Gonçalo Nuno Afonso Azevedo

Licenciado em Ciências da Engenharia Electrotécnica e de Computadores

Wireless Triple Play System

Dissertação para obtenção do Grau de Mestre em Engenharia Electrotécnica e Computadores

Orientador :	Prof. Doutor Luís Bernardo, Professor Auxiliar com Agregação, FCT-UNL
Co-orientador :	Engenheiro Tiago Duarte, Partner, Softconcept

	Júri:
Presidente:	Prof. Doutor Pedro Amaral
Arguente:	Prof. Doutor José Manuel Fonseca

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Março, 2014

Wireless Triple Play System

To my mother

Acknowledgements

I would like to acknowledge several people that helped me through the development and writing of my thesis.

First I would like to show my gratitude towards both my supervisors Prof. Luís Bernardo and Eng. Tiago Duarte for their insightful guidance and support when it was most needed.

I would like to thank Nuno Malaguerra and Maurice Dini from Meru Networks for borrowing the testing equipments and for their endless availability during the development of my dissertation.

Bruno Lima from 2007com was a valuable help during the development of my thesis. I would like to thank him for his knowledge and method as also with the exemption of the Alcatel-Lucent testing equipments.

I would like to thank Tim June and Chunlei Yi from OpenVox for their value help during the development of my thesis. Their great knowledge on installation and configuration process of SIP interfaces were a great help for the developing of this work.

Eng. Hugo Azevedo for its constant guiding, patience and companionship throughout my dissertation. I would like to thank him for his thorough knowledge and method, otherwise I would not accomplish this lengthy milestone.

Also I would like to thank to all colleagues from FCT-UNL for their fellowship, great ambience and their advices to overcome all the difficulties during the thesis development: Vitor Astucia, Vanessa Chamorinha, Raquel Melo , Catarina Lucena, Eduardo Santos, João Chalaça, Marcio Mateus, Tiago Teixeira, Pedro Almeida, Carlos Simão, Bruno

Duarte, Marcelo Rodrigues, José Manuel, Fabio Oliveira, Bruno Caixinha, Pedro Antunes and to Filipa Pacheco for all of comprehension, support and encouragement that she always transmitted me.

Finally, I would like to show the most kind and special gratitude to my family: my Mother and Father. They have always supported me during the most difficult times of my life.

A special thanks to Softconcept and all of their collaborators by the technical, financial and logistic support through the development of this project.

To all of you, thank you!

Almada, February 2014

Gonçalo Azevedo

Abstract

Triple play is a service that combines three types of services: voice, data and multimedia over a single communication channel for a price that is less than the total price of the individual services. However there is no standard for provisioning the Triple play services, rather they are provisioned individually, since the requirements are quite different for each service. The digital revolution helped to create and deliver a high quality media solutions. One of the most demanding services is the Video on Demand (VoD). This implicates a dedicated streaming channel for each user in order to provide normal media player commands (as pause, fast forward).

Most of the multimedia companies that develops personalized products does not always fulfil the users needs and are far from being cheap solutions. The goal of the project was to create a reliable and scalable triple play solution that works via Wireless Local Area Network (WLAN), fully capable of dealing with the existing state of the art multimedia technologies only resorting to open-source tools.

This project was design to be a transparent web environment using only web technologies to maximize the potential of the services. HyperText Markup Language (HTML), Cascading Style Sheets (CSS) and JavaScript were the used technologies for the development of the applications. Both a administration and user interfaces were developed to fully manage all video contents and properly view it in a rich and appealing application, providing the proof of concept.

The developed prototype was tested in a WLAN with up to four clients and the Quality of Service (QoS) and Quality of Experience (QoE) was measured for several combinations of active services. In the end it is possible to acknowledge that the developed prototype was capable of dealing with all the problems of WLAN technologies and successfully delivery all the proposed services with high QoE.

Keywords: Wireless, IPTV, VoD, VoIP, Triple play, QoE, QoS

Resumo

Triple play é um serviço que combina diferentes tipos de serviços: voz, dados e multimédia num único canal de comunicação, por um valor monetário inferior ao total de cada um dos serviços. No entanto, não existe nenhum protocolo que defina na totalidade os serviços de Triple play, devido às diferenças de requisitos de cada um dos serviços. Cada um dos serviços tem que ser distribuído individualmente. A evolução digital ajudou na criação de várias soluções multimédia de alta qualidade. Uma das mais promissoras é o VoD, que implica um canal dedicado para cada utilizador, de forma a possibilitar comandos de utilização normais como, por exemplo, a pausa ou o avançar rápido.

As empresas oferecem produtos personalizados que nem sempre viabilizam todas as capacidades e necessidades que os clientes procuram. Estas ofertas estão longe de serem soluções viáveis ao nível monetário. O objectivo desta dissertação foi criar um sistema Triple play escalável e fidedigno, com capacidade de operar através de redes sem fios. Este sistema deve ser capaz de funcionar com o atual estado da arte das tecnologias multimédia e com recurso apenas a sistemas de código aberto.

Este projeto foi desenhado para ser um ambiente transparente e utilizar apenas tecnologias web de forma a maximizar o potencial dos serviços Triple play. Foram utilizadas tecnologias como o HTML, CSS e o JavaScript para o desenvolvimento das aplicações. Mostrou-se necessária a criação de uma interface de administração, bem como uma interface para o utilizador, de forma a possibilitar a manutenção dos conteúdos de multimédia e visualizá-los num ambiente apelativo para o utilizador.

O protótipo desenvolvido foi testado numa rede sem fios com até quatro clientes e a qualidade de serviço e a qualidade de experiência dos clientes foram medidas para várias combinações de serviços ativos. Em conclusão, foi possível verificar que o protótipo desenvolvido, ainda que longe de ser uma solução profissional, se apresentou capaz de lidar com todos os problemas das redes sem fios e fornecer todos os serviços propostos com elevada qualidade de experiência.

Palavras-chave: Redes sem fios, IPTV, VoD, VoIP, Triple play, QoE, QoS

Contents

List of Acronyms

3GPP 3rd Generation Partnership Project

ACL Access Control List

ADC Analogue to Digital Converter

AGC Automatic Gain Controller

AP Access Points

API Application Programming Interface

ARM Adaptive Radio Management

ATIS Alliance for Telecommunications Industry Solutions

AVC Advanced Video Coding

CA Conditional Access

CAT Conditional Access Table

COFDM Coded Orthogonal Frequency Division Multiplexing

CP Cyclic Prefix

CSCF Call Session Control Function

CSS Cascading Style Sheets

DAHDI Digium Asterisk Hardware Driver Interface

DB Data Base

DCCP Datagram Congestion Control Protocol

DCT Discrete Cosine Transform

DE-MUX Demultiplexer

DHCP Dynamic Host Configuration Protocol

DMO Dynamic Multicast Optimization

DNS Domain Name System

DRM Digital Rights Management

DVB Digital Video Broadcasting

DVB-T Digital Video Broadcasting - Terrestrial

EPG Electronic Program Guide

ES Elementary Streams

ETSI European Telecommunications Standards Institute

FEC Forward Error Correction

FFT Fast Fourier Transform

fps Frames Per Second

FR Full-Reference

FXO Foreign Exchange Office

FXS Foreign Exchange Station

GI Guard Interval

GNOME GNU Network Object Model Environment

GOP Group Of Pictures

GPL General Public License

GUI Graphical User Interface

HDTV High-Definition Television

HTB Hierarchical Token Bucket

HTML HyperText Markup Language

HTTP Hypertext Transfer Protocol

HVS Human Visual System

HSS Home Subscriber Server

IASF IPTV Application Service Functions

ICT Information and Communication Technologies

IETF Internet Engineering Task Force

IF Intermediate Frequency

IGMP Internet Group Management Protocol

IIF IPTV Interoperability Forum

IMS IP Multimedia Subsystem

IP Internet Protocol

IPTV Internet Protocol Television

IR Infrared

ITF IPTV Terminal Functions

ITU International Telecommunications Union

ITU-R International Telecommunications Union Radiocommunication Standardization Sector

ITU-T International Telecommunications Union Telecommunications Standardization Sector

IVR Iteractive Voice Response

LIRC Linux Infrared Remote Control

MC Multipoint Controller

MCE Media Center Edition

MCF Media Control Functions

MCU Multipoint Control Units

MDF Media Delivery Functions

MGCP Media Gateway Control Protocol

MOS Mean Option Score

MP Multipoint Processors

MPEG Motion Pictures Experts Group

MPEG-2 TS MPEG-2 Transport Stream

MPL Mozilla Public Licensing

MRO Multicast Rate Optimization

MSE Mean Square Error

MUX Multiplexer

NAFC Network Attachment Control Functions

NAL Network Abstraction Layer

NASS Network Attachment Subsystem

NGN Next Generation Network

NIT Network Information Table

NPAPI Netscape Plugin Application Programming Interface

NR No-Reference

OFDM Orthogonal Frequency Division Multiplexing

OIF Open IPTV Forum

OS Operation System

PAT Program Association Table

PBX Private Branch Exchange

PES Packetized Elementary Streams

PESQ Perceptual Evaluation of Speech Quality

PHP PHP Hypertext Preprocessor

PID Packet Identifier

PMT Program Map Table

POE Power over Ethernet

PSI Program Specific Information

PSNR Peak-to-peak Signal-to-Noise Ratio

PSTN Public Switched Telephone Networks

QoE Quality of Experience

QoS Quality of Service

RACF Resource and Admission Control Functions

RACS Resource and Admission Control Subsystem

RF Radio Frequency

RMS Root Mean Square

RR Reduced-Reference

RTP Real-Time Transport Protocol

RTCP Real-Time Control Protocol

RTMP Real-Time Messaging Protocol

RTSP Real-Time Streaming Protocol

SCF Service Control Functions

SCCP Skinny Call Control Protocol

SCTP Stream Control Transmission Protocol

SDF Service Discovery Function

SDK Software Development Kit

SDP Session Description Protocol

SE Standard Edition

SER SIP Express Router

SIP Session Initiation Protocol

SSF Service Selection Function

SSH Secure Shell

STB Set-Top Box

TCP Transmission Control Protocol

TIF Transport Information Filter

TISPAN Telecommunications and Internet converged Services and Protocols for Advanced Networking

TOS Type of Service

TS Transport Stream

TTY	Teletype write
TV	Television
UAC	User Agent Client
UAS	User Agent Server
UE	User Equipment
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
USB	Universal Serial Bus
VCL	Video Coding Layer
VLAN	Virtual LAN
VLM	VideoLAN Manager
VoD	Video on Demand
VoIP	Voice over Internet Protocol
VQM	Video Quality Metric
VQEG	Video Quality Experts Group
WLAN	Wireless Local Area Network
WMM	Wireless Multi-Media

List of Figures

List of Tables

1

Introduction

1.1 Thesis focus and scope

The main propose of this thesis is to study the feasibility of using Wireless Local Area Network (WLAN) technology to deliver an integrated web based triple-play solution (video, voice and data), which has high Quality of Service (QoS) and Quality of Experience (QoE) requirements.

The work comprises the study of video coding and transmission, QoS mechanisms for wireless networks and assessment of methodologies to be used for the implementation of the system model. It includes the implementation of a full architecture including the client and the servers. It also includes the assessment of the QoE and QoS achieved.

It was early decided that all the system would be implemented using adequate existing open-source resources, and the other necessary applications would be developed.

1.2 Problem statement and goals of the research

Networks have changed dramatically over the last few years. The availability of broadband access networks led to an enormous increase in the usage of Internet based multimedia applications such as video streaming, voice over ip and internet television. With this enormous increase triple-play services started to appear. Triple-play means to merge radio transmission, communications transmission and computer transmission in the information transmission. It does not mean the telecommunication network, computer network and cable television network will be converged in a physical way, but it refers to the integration of high-level business application. Triple-play service is a concept that means delivering video, voice and data through a single broadband connection, in this case WLAN. However, there is no standard for provisioning the triple-play services. Rather they are provisioned individually, since the requirements are quite different for each of the services. Such integration of network, communications and broadcasting technology gathers a variety of media onto a single platform and thus it will change the way the system actually works. Within the development of multimedia applications, the main problem faced by triple-play is the integration between the telecommunication network, which functions as the promotion of information transmission channel, and radio and television network, which possesses the advantages of disseminating information. Triple-play puts an end to the times when radio and television network monopolized information transmission while telecommunications monopolized the broadband operation [89].

Another challenge are the WLAN characteristics. WLAN are more sensitive to interference and therefore more lossy and unreliable. Indeed, it is a challenging task to provide a guaranteed level of QoS for real-time applications in WLANs. Future WLAN standards, like 802.11ad [18], promise to respond to this challenge, but this thesis considers the current 802.11n standard. The potential of using WLAN technology in this way is promising as it can significantly reduce infrastructure and maintenance costs of cabled architectures. The streaming of media (both video and audio) to one or many wireless displays is becoming more common. If bandwidth was unlimited, and the bit error rate negligible, then reliable streaming would be relatively easy to implement. However, as the bit rate increases with higher definition video formats such as High-Definition Television (HDTV), maintaining high QoS becomes more difficult [92].

With the huge growth of multimedia services consumption, users tend to be more sensitive to the experimented quality when playing an on-line game, doing an audio conference or simply watching a video. While the relation between subjective evaluation and objective quality measurements may not be known exactly, losses and delays in networks will most certainly impact with a service. Multimedia, and particularly video and audio traffic, generally have strict requirements on delay, jitter, and losses [88].

The goal of this thesis is to answer the aforementioned question by extending existing mechanisms and creating new ones to increase the performance of triple-play systems in today's WLAN. We also evaluate the usage of simple statistics from multimedia transmissions as a metric that approximates sufficiently the estimation of user satisfaction done by more complex means. The validity and applicability of these mechanisms are proven through assessment and prototype implementation, which in combination lead to the final results of this thesis. A prototype for the triple-play service client is implemented, which is the final link with the user. This thesis also incorporates the development of a streaming server that is responsible for delivering content to the customers.

1.3 Dissertation's Structure

This document is structured as follows. Chapter 2 provides a background study of the technologies used to provide Triple-play services, some background in video coding and transmissions architectures, QoS technologies and techniques to improve multimedia experiences through for WLAN. In chapter 3 is introduced and discussed the proposed architecture for the triple-play system, with the specification of both server architecture and client architecture. Chapter 4 deals with the development aspects and all the implementation decisions that have been made during the development process of the project. Tests with several scenarios to validate the proposed concepts are presented in Chapter 5. Finally, we conclude this thesis and discuss some ideas for future research in Chapter 6.

2

Background and Related work

In recent years there has been an explosive growth in the use of multimedia traffic. One consequence of this huge growth is the improvement on all aspects of video distribution services, and the growing importance of the user experience quality metrics. Nevertheless multimedia traffic evaluation considers both objective and subjective quality meters, which are very complex to estimate due to a multiplicity of factors that have to be taken into account.

This chapter presents an examination of the standards made by the international entities like, the International Telecommunications Union (ITU) [27] and the European Telecommunications Standards Institute (ETSI) [24]. It includes a description of the state-of art mechanisms used to prepare and transmit multimedia content, including key concepts, architectures, protocols and standards, to deliver high QoS and QoE requirements with resource to WLAN. Furthermore, a brief introduction to transport layer protocols, synchronization, encapsulation, packetization and coding techniques is presented, to give the reader a better understanding of how multimedia contents are transmitted. Alongside with this, a quick explanation about Triple-play services is done. The last section describes the techniques being used to evaluate multimedia services in terms of network and application performance. These techniques are used to determinate the QoS and QoE of multimedia services.

2.1 Network standards for multimedia

Figure 2.1 shows the main standards and protocols that can be used for multimedia transmission. The upper part of the figure shows commonly used coding standards, which are

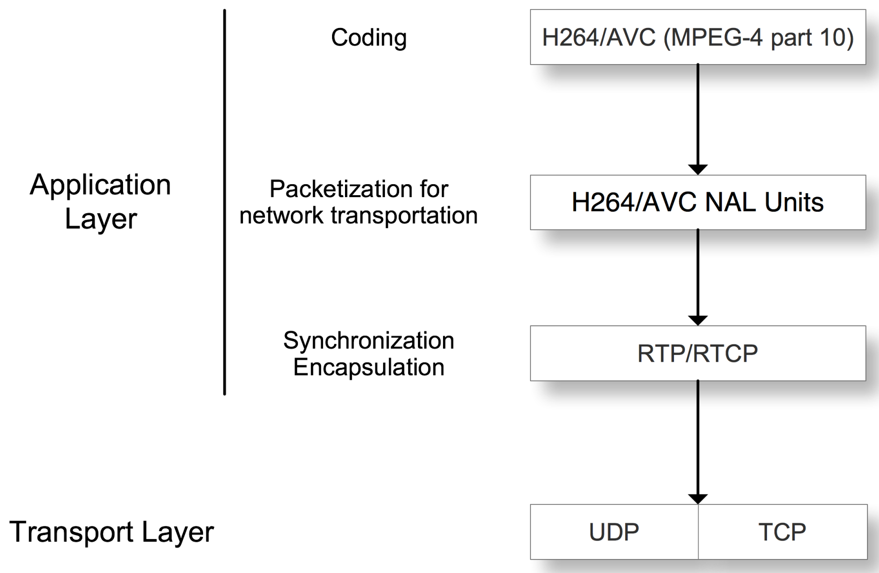

Application Layer
Coding
H264/AVC (MPEG-4 part 10)
Packetization for network transportation
H264/AVC NAL Units
Synchronization Encapsulation
RTP/RTCP
Transport Layer
UDP
TCP

Figure 2.1: Video transmission over an IP network (based on [88]).

part of the Motion Pictures Experts Group (MPEG) system. The MPEG codec (coder/decoder) is responsible for reorganizing multimedia information by reducing redundancies in spacial and temporal dimensions. Spatial reduction physically reduces the size of the video data by selectively discarding unneeded parts of the original data in a frame. On the other hand, temporal reduction, significantly reduces the amount of data pixels needed to store a video frame [35]. In order to be carried in packets and multiplexed for a successful delivery, audio and video outputs are encoded into audio and video Elementary Streams (ES) packetized as Packetized Elementary Streams (PES). The codec standards showed in figure 2.1 are the MPEG-4 [57] and the H.264/Advanced Video Coding (AVC) [84]. Next a separate packetization function is applied, which defines an MPEG-2 container format for the transmission MPEG-2 Transport Stream (MPEG-2 TS) [58]. For encapsulation and synchronization, Real-Time Transport Protocol (RTP) [45] and Real-Time Control Protocol (RTCP) [45] can be used, providing some quality feedback. Finally, the figure shows some of the transport protocols that can be used: they are the lightest User Datagram Protocol (UDP) [100] and the popular Transmission Control Protocol (TCP) [101]. The next subsections deepen the analysis of the most relevant options.

Transport layer protocols

The following subsection discusses the four main transmission protocols in the Internet Protocol (IP) stack used for multimedia data transportation that have been deployed by the Internet Engineering Task Force (IETF) [34]. The transport layer protocols are responsible for end-to-end transport of data. They have no knowledge of the network topology. They take an amount of data, which we call a packet if we use UDP or a segment if we use TCP, and transmit it. The most important of the transmission layer protocols are the UDP [100], TCP [101], Datagram Congestion Control Protocol (DCCP) [91] and Stream Control Transmission Protocol (SCTP) [109].

The UDP is a connectionless, message-oriented transport protocol [100]. It simply sends a certain amount of data from a source to a destination address. It neither gives any guarantees about the order in which packets arrive, nor about whether they arrive at all, but nevertheless, it does not worsen the levels of latency and jitter that the network provides. On the other hand, UDP involves very little overhead, with respect to both time and space. UDP is therefore the most useful for applications where delay is more critical than reliability. The TCP [101] is a transport protocol that provides mechanisms of flow and congesting control for supporting a reliable data transmission. This comes at the cost of more overhead and lower transmission speed than UDP. Arbitrarily long delays can be caused by the mechanisms of in-order delivery and congestion control of TCP [88]. The TCP specification also provides a number of congestion control algorithms. These algorithms adapt the send rate to the load of the network: if more bandwidth is available, the send rate is increased, if network congestion occurs, the send rate is decreased. This guarantees a near-optimal transmission speed of TCP packets except when transmission errors occur. Another important fact, is that TCP requires a connection set up between the sender and the receiver, before any data transmission can take place, which sometimes can cause a delay of twice the latency before the data can be sent. According to Wang, Kurose el al. [119] TCP may offer good performance for multimedia streams if its throughput is at least twice the media bit-rate and a few seconds of playback starting delay are allowed [88].

The DCCP [91] is a rarely implemented transport protocol. It is a message-oriented protocol that offers bidirectional unicast connections of congestion-control unreliable datagrams. DCCP is suitable for applications that may benefit from controlling the trade-off between delay and reliability in the transfer of large amounts of data. It is believed that compelling advantages will be given if DCCP is the chosen transport layer for the RTP streams [88].

The SCTP [109] is a connection-oriented protocol with congestion control that was initially meant to transport telephony signalling over IP networks. The SCTP places messages and control information into separate chunks (data chunks and control chunks), each identified by a chunk header. For performance and efficiency, SCTP transmits several independent streams of chunks in parallel, which some may be multimedia friendly and others may have error recovery features. Security features include a 4-way handshake to avoid SYN attacks and identity certifications using cookies. The SCTP packet is composed by a header and a payload that consists of control and data chunks. SCTP has been tested as an alternative to UDP for transporting multimedia streams [99], however it is not widely implemented yet [88].

Application layer protocols

The RTP [105] is an application layer protocol that is specially designed for multimedia streaming. In other words, RTP provides specifications for media transportation from the server to the client using data packets. Typically, RTP is carried over UDP, for reasons of performance, although it could also be built upon other transport layer protocols, like TCP. However this is not frequent because it lacks for efficient time delivery mechanisms, as its major concern is reliability. The RTP specification does not guarantee reliability, but it provides mechanisms to compensate jitter and losses of packets. In multimedia streaming, these two types of problems are the ones that have the most impact in the perceived quality of audio and video. Besides providing encapsulation, media synchronization and signalling for quality of service, RTP also has extended functionalities like error protection, codec control messages, and security. RTP is designed to work in

conjunction with the auxiliary control protocol RTCP [105] to get feedback on quality of data transmission and information about participants in the on-going session. RTCP also provides quality feedback reports with statistics on network status; these statistics may include performance data as packet loss, number of sent packets, jitter and delay.
Other entities generally present when RTP is used are the signalling protocols, which may include Session Description Protocol (SDP) [61] and Session Initiation Protocol (SIP) [103] or the Real-Time Streaming Protocol (RTSP) [106].
Three coding standards were focused in the scope of this thesis: MPEG-2, MPEG-4 and H.264. Information about these video coding standards can be seen in appendix A.1.

RTSP

The RTSP protocol is an application level protocol as RTP, which is used for controlling the delivery of data with real-time properties. It provides an extensible framework that enables controlled, on-demand delivery of real-time data, such as audio and video. This protocol is intended to control multiple data delivery sessions and provide means for choosing what is the best transport protocol stack to use. This protocol stack usually specifies two different protocols to use, the lower transport protocol TCP/UDP or the high level protocol RTP. RTSP consists of an exchange of text messages between the client and the server, similar to Hypertext Transfer Protocol (HTTP) [56]: the clients sends request messages, and the server answers to each request. The protocol defines several commands for the communications between the client and the server. The most important commands are the OPTIONS, DESCRIBE, SETUP, PLAY, PAUSE and TEARDOWN. Figure 2.2 illustrates a message flow example between the RTSP client and the RTSP server. Later on section 3.4 more RTSP flow messages examples will be discussed.

- OPTIONS This request method inquires the method supported by the server.
- DESCRIBE This method retrieves initialization information from the server about the media session that is going to be played. This initialization information contains the number and type of individual media streams composing the media session and also some information about the available transport methods supported on the server's side.

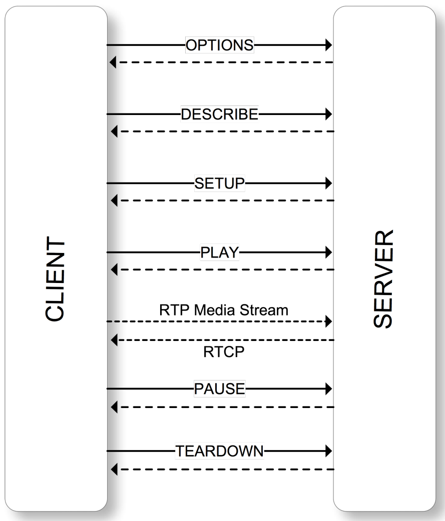


Figure 2.2: RTSP Message flow example.

- SETUP This method executes the request for individual media streams (audio and video media stream). The most important field in this method is the transport field, because it contains the information about the data transport possibilities on the client side. The answer from the server also contains a transport field that actually choose one of the transport methods proposed by the client. If everything is allright, the server generates a session identifier in response to SETUP requests.
- The PLAY command is sent to the server so that it can start sending the media stream data.
- The PAUSE request causes the stream delivery to be halted temporarily.
- A TEARDOWN request is used to stop all the media streams and freeing all session related data on the server's side.

SIP

SIP is an application layer protocol that uses the "request response" template, similar to HTTP, to manage interactive communication sessions between users, ensuring the translation of service name, location, negotiating attributes of the session management and participants. The SIP purpose is to control the communication between end multimedia devices, supported by the protocols RTP/RTCP and SDP. The RTP protocol is used for carrying multimedia data in real time, while the SDP protocol is used for negotiating the capabilities of participants. SIP was adopted by 3rd Generation Partnership Project (3GPP) as a signalling protocol in IP Multimedia Subsystem (IMS) [47] control architecture. The target applications of this protocol architecture are the signalling and call control for Voice over Internet Protocol (VoIP) [87], with the possibility of integrating other services, such as Internet Protocol Television (IPTV) [98]. As before, this protocol defines a set of specific methods: INVITE, BYE, OPTIONS, REGISTER, CANCEL, ACK. Figure 2.3 illustrates the use of these messages. More detailed information about SIP messages is given in section 3.4.

Figure 2.3 shows an example of SIP operation, where a client is inviting another client for a call. SIP client A sends an INVITE message for client B, which normally is sent via a proxy server. The server finds out that client A and client B are in the same network so it forwards this message directly to client B. Client B receives this message and returns an OK message to client A via the proxy server. The call between client A and B is established after client B receive the ACK message.

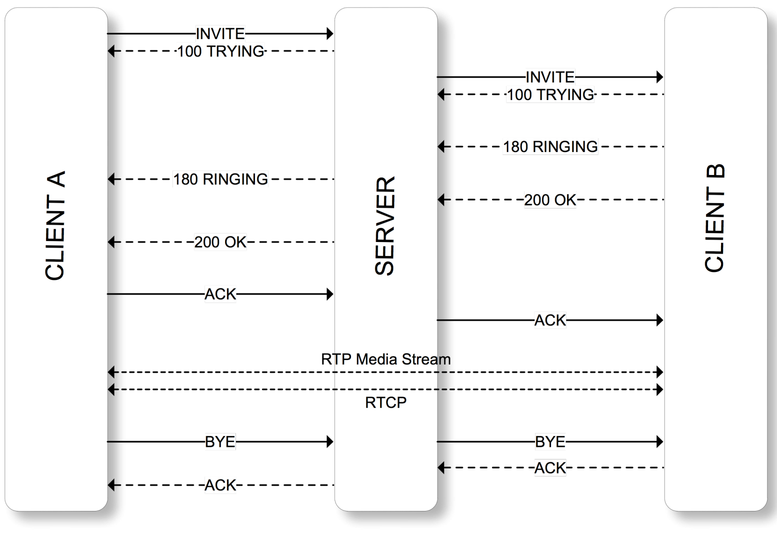


Figure 2.3: SIP Message flow example.

- REGISTER Gives information about the location of user to the SIP registration server.
- OPTIONS is a signalling message that gives information about the capabilities of a call.

- The INVITE request is used to invite the End User to join in a VoIP call. The header field usually consists of the addresses of the person calling, and the intended receiver, it gives information such as, the subject of the call, features to be used, preferences and routing information. The body field contains an object that gives information of the media content of the session. It gives information on what ports and what protocols are used to send media.

- The BYE message is used to disconnect a connection between two users in a VoIP call.

- ACK is used to maintain reliable message exchanges and it is used in response to invitations.

- CANCEL is used to terminate a particular request.

SDP

The SDP specifies a format for describing multimedia sessions, aimed at the announcement and the call session, the session description of attributes, among others. Its syntax is text-based. It is used together with other protocols as SIP and RTSP. The SDP includes information on streams (like audio or video), the formats used, the source and destination addresses, ports (to send and receive), the start and end times of the session, the originator, the level of quality, the number of video pictures per second, among others. In section 3.4 is showed an example of an SDP message.

2.2 Multimedia services

The ubiquity of the Internet led many multimedia content owners to search for a way to deliver multimedia content over an IP network. Video and audio are a more common way to communicate than the text and images used for the first decade of the Internet. A large number of multimedia services can be offered to end-users via IP networks. These services may include simple audio and video streaming, telephony or teleconferencing with or without video, and interactive games or other sessions. This section focuses on three multimedia services and their delivery over an IP network to end users: IPTV, Video on Demand (VoD) and VoIP.

Video

Internet Protocol Television (IPTV)

Internet Protocol Television (IPTV) has a lot of different definitions in literature and also in practice, resulting in ambiguous interpretations of the concept. In this thesis the reference from International Telecommunications Union Telecommunications Standardization Sector (ITU-T) is used as definition for IPTV [74].
"*IPTV is defined as multimedia services such as television/video/audio/text/graphics/data delivered over IP-based networks managed to support the required level of QoS/QoE, security, interactivity and reliability.*"
The main features of IPTV service are: the distribution of IPTV content over an private IP network; the delivery of standardized video formats; the delivery of multiple contents/channels; and the continuous streaming of content. To deliver these features continuously, the IPTV distribution network has to be carefully planned and designed. This is only possible using a private network managed by the provider and the use of multicast technology. Multicast reduces the use of network resources between the video server and the customer, by sending only a single copy of the media packets into the network. The network will replicate the stream to individual subscribers closer to the edge, saving this way bandwidth in the main core and also all traffic will be controlled by ensuring adequate levels of QoS [108].
Nowadays there are a lot of web-based video services which do not offer managed delivery of multimedia services and do not give any QoS guarantees. These services are often known as Internet TV services. For example, one of the most popular Internet TV service is Youtube [38], which works with user contributed videos. The delivery of these videos is not controlled by Youtube or any type of related service provider. The video delivery is not guaranteed by the service provider but only by the type of internet connection that the user has available, without any guarantees that the requirements of latency in the internet connection are satisfied. The most representative aspects of managed IPTV

services are:

- **IPTV services use an end-to-end or semi-closed network**

  IPTV services are provided by one service provider, who is also responsible for creating the means of making the IPTV available. In other words the service provider is responsible for creating/providing the network infrastructure, the access to the content and the decoder to access, receive and display the IPTV content.

- **IPTV services availability are geographically bound**

  The IPTV services are only deployed at locations where the service provider has full control of the infrastructure and also where the network offers enough bandwidth for the complete delivery of IPTV services.

- **IPTV services are service provider driven**

  Normally the IPTV subscriber uses services offered by the service provider, the user himself does not offer any kind of services.

- **IPTV services make use of access and admission control**

  Before a user can use IPTV service, the user has to be authenticated, in order to have full access rights over the content. Furthermore the service will only be available when the infrastructure has sufficient bandwidth to correctly deliver the service.

The main differences between IPTV and Internet TV are summarized in Table 2.1:

Table 2.1: A comparison of IPTV and Internet TV services.

	IPTV	**Internet TV**
Users	Requires IPTV infrastructure	Anyone with Internet access
QoS Guarantees	Yes	No
Security	Yes	No
Distribution Network	Closed	Open, Internet
Video Formats	MPEG-2/MPEG-4 H.264	Windows Media/Flash Video/H.264
Mobility	Yes	No
Content access	Subscription	Free

Typical IPTV services can be divided into three main groups [75]:

- Linear broadcast Television (TV): In Linear broadcast TV television, stations broadcast their channels via the air, satellite, or cable and the users can select a channel to view the program that the television station is currently broadcasting. IPTV broadcast television is similar to television broadcast, the difference lies in the distribution method: IPTV broadcast television uses IP multicast transport. The subscriber can select from numerous live television broadcasts, which are being transmitted using multicast delivery.

- Video on Demand: VoD services are described in the section bellow.
- Time-shifted TV: Time-shifted TV is a combination of linear broadcast TV and VoD. It provides a flexible viewing window for television broadcasts, allowing users to watch the beginning of a program, when the broadcast has actually already started.

Video on Demand (VoD)

VoD is a relatively well-known concept. The term of VoD is widely used for systems that allow watching video content via communications systems like Internet or cable TV. VoD allows video contents to be available on previous request from users, that is, the user can select the exact program that he wishes to see at any given time. Typically, the contents are available as pre-recorded files located in streaming servers [112]. This method of delivering video contents has several advantages such as [46]:

- The video contents are available in the transport network, only when the user asks to consume them, saving this way resources for another tasks.
- The security of this type of service - when the user asks for a service, the application system can easily identify if the client is valid, and when to allow the access to the content.
- It allows the use of video control commands, such as pause, volume up and down or fast-forward.

Since the data transmission is done for each individual user, it is typically performed at unicast, which facilitates the session customization. However, the cost of this individual transmission increases the bandwidth in use, due to the replication of open streams to the same content.

VoD services are segmented into several categories, classified by the allowing level of interactivity [94]:

- Live Broadcast (No-VoD):
  The user is a passive participant and has no control over the session. One example of this type of VoD is the traditional TV broadcast channels.
- Pay-per-view (PPV):
  Viewers sign up and pay for a specific program but still cannot make any influence on the way, when or how the content is shown.
- Quasi Video on Demand (Q-VoD):
  Viewers can be grouped, based on a threshold of interest. The users have rudimentary level of temporal control activities by the ability of changing to a different group. For example, viewers interested in sport events are joined in a group where soccer games are broadcast.
- Near Video on Demand (N-VoD):
  Services where functions like forward and reverse are simulated by transitions in discrete intervals of time (on the order of 5 minutes). This capability can be provided by multiple channels with the same programming skewed on time.
- True Video on Demand (T-VoD):
  Viewer has complete control over the session presentation. The user can jump to any position and perform operations like forward and reverse, play, freeze and random positioning. This service has one unique streaming for each client who is granted access to the media content. This means that several customers can start watching the media content whenever they want.

From now on, in this thesis VoD is synonymously referred to a T-VoD system.

IPTV Standards and Architectures

Nowadays, many areas of IPTV systems are covered by standards in an attempt to make IPTV system an interoperable system. The delivery of IPTV system over a Next Generation Network (NGN) has become the main objective of several standardization bodies such as ITU-T, ETSI Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN), Open IPTV Forum (OIF) and Alliance for Telecommunications Industry Solutions (ATIS). A brief overview of the IPTV standards is made in table 2.2 .

Table 2.2: IPTV Standardization overview [62].

Name	**Focus**
DVB Over IP Networks	IPTV and interactive television, primarily for broadcasters
ITU-T Focus Group IPTV	Definition of IPTV standard
ETSI TISPAN IPTV	IPTV based on IMS and referencing relevant standards for the transport layer
ATIS IPTV Interoperability Forum	IPTV for cable-TV providers
IETF	Defines protocols for IPTV (e.g. SIP, RTSP, RTP)
Open IPTV Forum	End-to-end IPTV service, including interaction and quality of service

- DVB Over IP Networks:

  Digital Video Broadcasting over Internet Protocol (DVB over IP), provides a set of technical specifications that cover the delivery of Digital Video Broadcasting (DVB) MPEG-2-Based services over bi-direction IP networks, including specifications of the transport encapsulation of MPEG-2 services over IP and the protocols to access such services [7].

- ATIS IPTV Interoperability Forum:

  The Alliance for Telecommunications Industry Solutions (ATIS) approved the formation of the subgroup called IPTV Interoperability Forum (IIF). This Forum is responsible for developing standards to enable the interoperability, interconnection, and implementation of IPTV systems and services, including VoD and interactive TV services. ATIS analyses several important aspects of IPTV and is also responsible for defining the IPTV logical domains as, IPTV reference architectures (IMS- and non-IMS-Based IPTV), content delivery concepts with QoE, Digital Rights Management (DRM) requirements and interoperability standards as well as testing requirements for the components [44].

- IETF:

  The Internet Engineering Task Force (IETF) is responsible for the definition of protocols that IPTV standards use such as SIP for session control, RTSP for media control of VoD services, Internet Group Management Protocol (IGMP) for IPv4 multicast-based services and RTP for media delivery.

- Open IPTV Forum:

  The Open IPTV Forum (OIF) is an initiative with the objective of working with other standardization bodies to define a common and open end-to-end solution for supplying a variety of IPTV services across a wide variety of different network architectures [62].

- ITU-T Focus Group IPTV:

  ITU-T Focus Group IPTV has been created with the mission of coordinating and developing the global IPTV standard architecture based on a client-server architecture with the addition of a service delivery platform while considering the flowing key areas: DRM, QoS/QoE, meta-data and interoperability [76].

- ETSI TISPAN IPTV:

  The ETSI/TISPAN emerged with the mission of developing specifications for the next generation wireless and fixed networking infrastructures, defining IPTV as a NGN service and using the IMS. This architecture can connect to legacy networks via gateways that make part of the IMS or any other SIP-based architecture [62].

Two main architectures have been approved for IPTV delivery: they are the ITU-T Focus Group IPTV Architecture and the ETSI TISPAN IPTV Architecture.

ITU-T Focus Group IPTV Architecture

The IPTV model lay over four main functional domains in the provision of an IPTV service. They are: the end user domain, responsible for presenting services to the end user; the network provider domain, allowing the connection between the consumer domain and the service domain; the service provider domain, which is responsible for providing consumers with services; and finally the content provider domain that owns the content or has a license to sell contents or content assets. Figure 2.4 shows the domains responsible for the provision of IPTV services over a NGN [107].

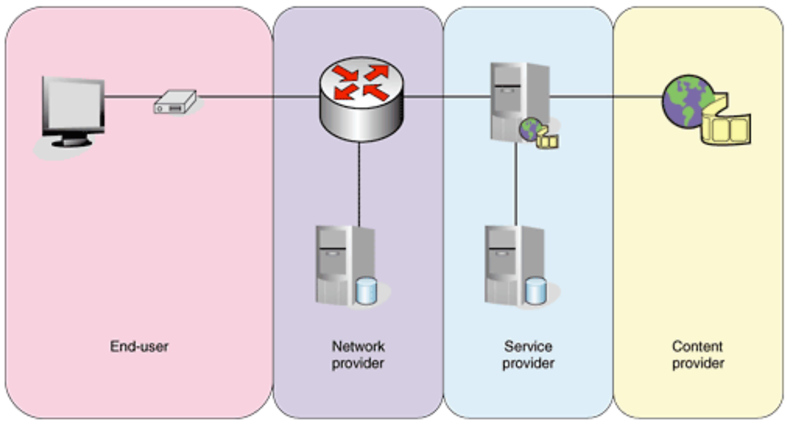


Figure 2.4: IPTV domain model [93].

The ITU-T identifies three IPTV architecture approaches that enable the service providers to deliver IPTV services, they are: the non-NGN IPTV architecture, the NGN-based non-IMS IPTV architecture and the NGN IMS-based IPTV architecture [73].

- Non-NGN IPTV Functional Architecture (Non-NGN IPTV): The Non-NGN IPTV architecture is based on the existing network components and protocols/interfaces. The use of this approach generally requires a separate service control and an application layer developed specially for the IPTV services.

- NGN-based non-IMS IPTV Functional Architecture (NGN-Non-IMS IPTV): The NGN-Non-IMS IPTV architecture uses components of the NGN framework reference architecture as defined in [ITU-T Y.2012 [81]] to support the provision of IPTV services, in conjunction with other NGN services. This approach uses a dedicated IPTV subsystem within the NGN to provide the necessary IPTV functionalities.

- NGN IMS-based IPTV Functional Architecture (NGN-IMS-IPTV): The NGN-IMS based IPTV architecture uses components of the NGN architecture including the IMS components to support the provision of IPTV services, in conjunction with other IMS services if required. The IMS functions and associated function such as

Service User Profile Function Block are defined in [ITU-T Y.2021 [72]] to provide service control functions.

The IMS-based and the non-IMS-based approaches only differ in terms of the inclusion of the core IMS *Service Control Functions (SCF)* in the IMS-based IPTV solution. The *IMS Core* provides a SIP-based session management for all types of application, and a common charging mechanism for all applications.

The IPTV Functional Architecture Framework is classified into seven functional groups, as showed in Figure 2.5. The *End-User Functions* perform mediation between the end-user and the IPTV infrastructure. Among the *IPTV Terminal Functions (ITF)*, the *Session Client Functional Block* provides the functions to handle service request, such as session initiation, modification and termination. The *Application Functions* enable the *End-User Functions* to select and purchase a content item. The *Service Control Functions* provide the functions to request and release network and service resources to support the IPTV services. The *Content Delivery Function*s initiate the content delivery, allocate temporary storages, and request the *Network Functions* to reserve network bandwidth for streaming contents. For the delivery of contents, the *IMS Core Functions* offer session control mechanisms in order to provide authentication and authorization functions based on the user profiles. By interacting with *Resource and Admission Control Functions (RACF)*, the relevant related resources are reserved. The *IMS Core Functions* interact with *ITF*, *IPTV Application Functions*, *Content Delivery Functions* and can be used for *Service Discovery Function* such as charging and roaming. The *Management Functions* perform overall system management, status monitoring and configuration. The *Content Provider Functions* are provided by the entity that owns or is licensed to sell content or content assents. Finally the *Network Functions* provide transport connectivity through *NGN Transport Functions*. Their network resources are shared among end-users and provide a certain QoS classes with the help of *Transport Control Functions*, *Network Attachment Control Functions (NAFC)* and *RACF* [73].

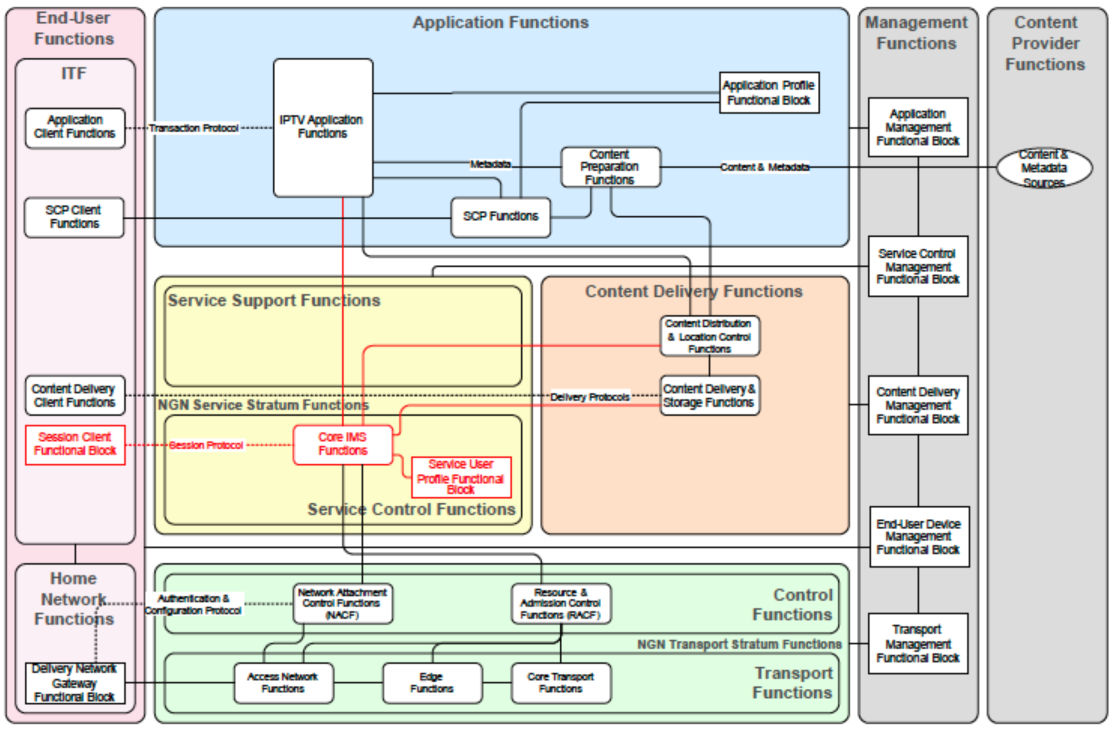

End-User Functions
ITF
Application Client Functions
SCP Client Functions
Content Delivery Client Functions
Session Client Functional Block
Home Network Functions
Delivery Network Gateway Functional Block
Application Functions
IPTV Application Functions
Application Profile Functional Block
Content Preparation Functions
SCP Functions
Transaction Protocol
Metadata
Content & Metadata
Service Support Functions
NGN Service Stratum Functions
Core IMS Functions
Service User Profile Functional Block
Session Protocol
Service Control Functions
Content Delivery Functions
Content Distribution & Location Control Functions
Content Delivery & Storage Functions
Delivery Protocols
Authentication & Configuration Protocol
Network Attachment Control Functions (NACF)
Resource & Admission Control Functions (RACF)
Control Functions
NGN Transport Stratum Functions
Access Network Functions
Edge Functions
Core Transport Functions
Transport Functions
Management Functions
Application Management Functional Block
Service Control Management Functional Block
Content Delivery Management Functional Block
End-User Device Management Functional Block
Transport Management Functional Block
Content Provider Functions
Content & Metadata Sources

Figure 2.5: ITU-T NGN-IMS-based IPTV Architecture [73].

ETSI TISPAN IPTV Architecture

The ETSI/TISPAN follows a parallel approach to ITU-T classifying IPTV architecture into the NGN-based non-IMS IPTV architecture and the NGN IMS-based IPTV architecture. In addiction, it considers two subsystems that are responsible for the *Transport Control Functions*: the *Resource and Admission Control Subsystem (RACS)* and the *Network Attachment Subsystem (NASS)*. The first one is responsible for the police management and implementation, ensuring that the user is allowed to access the services being requested. It also reserves and allocates the required and subscribed amount of bandwidth for the service. The second one (*NASS*) is responsible for provisioning the IP address and other configurations of the terminal, as well as authentication and authorization of the subscriber. Both NASS and RACS form an integral part of the NGN [16].

The overall functional architecture for IPTV service according to ETSI/TISPAN is shown in figure 2.6.

IMS-based functional architecture for IPTV services

The main functions and reference points defined in ETSI/TISPAN NGN IMS IPTV concept are, the *SCF*, the *Media Control Functions (MCF)* and the *Media Delivery Functions (MDF)* [16].

As showed in figure 2.6, the *User Equipment (UE)* can communicate with the IPTV application servers (including *SCF*) over various interfaces for different proposes, namely, over *Gm* interface via the *IMS Core* for session management propose, directly over a *Ut* interface for the service profile configuration purpose, or over the *Xa* interface to interact with service selection functionalities. Each *UE* has at least four interfaces for media control over *Xc* and media delivery over *Xd*, as well as the *Gm* interface to the *IMS Core* and the *Ut* interface to *IPTV Application Service Functions (IASF)*. *MCF* can control *MDF* over *Xp* interface, that also allows building a really scalable and distributed media delivery infrastructure.

IMS Core elements

The *IMS Core* specification comprises two main nodes: the *Call Session Control Function (CSCF)* and the *Home Subscriber Server (HSS)*. The *CSCF* is the heart of the IMS architecture and is used to process SIP signalling. The main function of the *CSCF* is to provide session control for terminals and applications using the IMS network. Three types of *CSCF* are defined:

- The *Proxy-CSCF* that is the first contact point for the IMS users. The main goals of the *P-CSCF* are to guarantee the signalling messages between the networks and subscribers and the resource allocation for media flows by interaction with the *RACS*.

- The *Serving-CSCF* is the main control entity within the IMS. It processes registrations from subscribers and stores their current location and also is responsible for subscriber authentication and session management. Subscriber policies stored in the *HSS* control the operations performed by the *S-CSCF* for a particular subscriber.

- The *Interrogating-CSCF* that is responsible for quering the *HSS* in order to discover the appropriate *S-CSCF* for the subscriber. The *HSS* is the database that contains user and subscriber information to support the network entities handling calls and sessions.

Service Discovery and Selection

The *Service Discovery Function (SDF)* and *Service Selection Function (SSF)* provide the necessary information that is required for an *UE* to select an IPTV service. The *SDF* is responsible for providing service attachment information about accessible IPTV services (personalized service discovery). The *SSF* is used to provide service selection information, as well as personalized user preference information.

IPTV Service Control Functions

IPTV *SCF* is responsible for handling the IPTV related requests and execute service and session control of all IPTV services. These functions are also responsible for interworking with the *IMS Core* on the service layer.

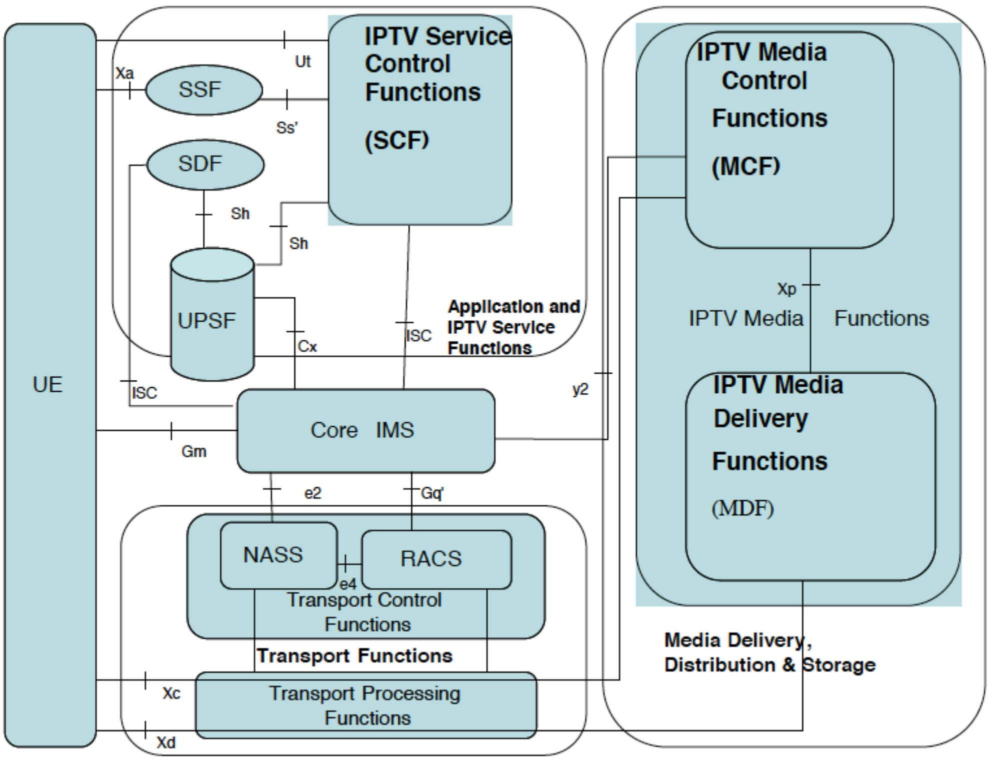

UE
Xa
SSF
Ut
Ss'
SDF
Sh
Sh
UPSF
Cx
ISC
ISC
IPTV Service Control Functions (SCF)
Application and IPTV Service Functions
Core IMS
Gm
e2
Gq'
y2
NASS
RACS
e4
Transport Control Functions
Transport Functions
Transport Processing Functions
Xc
Xd
IPTV Media Control Functions (MCF)
Xp
IPTV Media Functions
IPTV Media Delivery Functions (MDF)
Media Delivery, Distribution & Storage

Figure 2.6: TISPAN NGN-IMS-based IPTV Architecture [16].

Voice

Voice over Internet Protocol (VoIP)

Voice Over Internet Protocol (VoIP) may be defined as a set of technologies and components that together, offer the user the possibility of establishing voice calls through an IP network[52].

Traditional phone networks, known as Public Switched Telephone Networks (PSTN) use circuit-switching technology. In circuit-switch, resources are reserved along with the entire communication channel for the duration of the call. On the other hand, IP uses packet-switching, where the information is digitally transmitted using individual packets. Packets know their destination and may be able to arrive via different paths.

VoIP allows users to make voice calls to other VoIP users but also to traditional users (PSTN users). Through VoIP these users will obtain various additional services beyond voice calls, including voice mail, fax, Iteractive Voice Response (IVR), conferencing call and instant message[50].

VoIP has very strict network delay requirements, so for guaranteed QoS VoIP we should use private IP networks intended for voice traffic or use VoIP service providers (in this case an internet connection with traffic differentiation is preferable).

VoIP Architectures

Figure 2.7 presents a generalized model of a VoIP network. While the details of various protocols differ, all major VoIP architectures are sufficiently similar so that their major features can be described uniformly.

A VoIP system contains two types of components: end systems and signalling servers. The end systems can originate a call. When this call is placed, the call establishment request can proceed by a variety of routes through components of the network. At first, the originating end system must decide where to send its requests: either the originator is configured to send all its requests to a single local server, or it may resolve the destination address to locate the remote signalling server. When the request arrives at the signalling server, that server uses its user location database, its local policy, its Domain Name System (DNS) resolution, or any other methods to determinate the next signalling

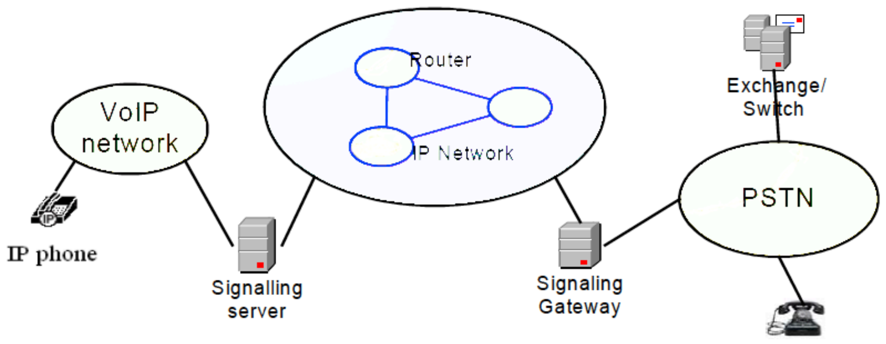


Figure 2.7: Generalized model of VoIP architecture [104]

.

server or end system to which the request should be sent.

The two VoIP architectures, SIP and H.323, are based on similar concepts. The tables 2.3 and 2.4 show the components and protocols of the two approaches. Both architectures comprise a lot of analogies with respect to function split and service location. In SIP as well as in H.323, basic call and feature control are performed in the terminals. For features requiring network support, servers are foreseen in the networks [85].

Table 2.3: SIP and H.323 components [85].

	Client	**Network Servers**		
SIP (IETF)	User Agent	Proxy Server	Registration Server	Redirect Server
H.323 (ITU-T)	Terminal	Gatekeeper	MCU	Gateway

Table 2.4: Protocols used by H.323 and SIP [85].

	Real-Time Data Transmission	Call Control
SIP (IETF)	RTP/RTCP	SIP and SDP
H.323 (ITU-T)	RTP/RTCP	H.225 and H.245

Information about H.323 can be found in appendix A.2.

SIP

The basic elements that compose a SIP architecture are: the User Agents and the Network Servers. There are tree types of servers within a network: A Proxy Server, a Registration Server and a Redirect Server [104].

- User Agent: User agent can be defined as IP terminals using SIP to find each other and negotiate the characteristics to establish a session. Each User Agent is composed of two parts: a client and a server. the client portion is called the User Agent Client (UAC) and it is the part that sends requests and receives responses. The server portion is called the User Agent Server (UAS) and it is the part that receives requests and sends responses. For example, when an entity is initiating a call, the User Agent behaves like UAC when sending the INVITE request and receive responses to the request. For anyone who receives a call, the User Agent behaves like UAS when receiving an INVITE request and sends responses.

- Proxy Server: The proxy servers act as an intermediary program, receiving requests, forwarding them to the next server. A proxy server also provides mechanisms for authentication and accounting.

- Registration Server: The Registration Server is a SIP entity that receives user records, extracts information about their current location and stores them into a location database. The propose of the database is to map sip:user@softconcept.pt to something like sip:user@192.168.1.1:5060 latter used by the Proxy Server.

  When the proxy receives an INVITE to sip:user@softconcept.pt the address translation will search in the location database by the real address (sip:user@192.168.1.1:5060), the INVITE will be then redirect to there.

- Redirect Server: The Redirect Server is a SIP entity that receives a request and sends back a response containing a list with the location/address of the recipient. His function is to provide name resolution and user location, using the location database created by the Registration Server. Thus, the customer may contact the address of a user directly. The Redirect Server can reduce the processing on the proxy server that is responsible for forwarding requests and make more robust routing signalling.

2.3 Quality measurement

User experience

The delivery of multimedia services requires different levels of quality like limited jitter or delay, or a significant amount of bandwidth in order to operate properly. Although, independently of the specific requirements that this services can have, QoS is just a component of the overall QoE; a significant part of it is dictated by psychological effects [88]. ETSI identifies the main components of the user experience as: the technical measures related to QoS and the subjective and objective psychological measures. Objective psychological measures can be, for example, measurements on user effectiveness and efficiency. On other hand, subjective psychological measurements are related to the subjective evaluation given by the user and are expressed in terms of degrees of satisfaction [12].
QoE has a lot of different definitions, however the standardization groups recognized the need for contextualizing the meaning of it:

- ETSI defines QoE as a measure of user performance based on both objective and subjective psychological measures when using service or product within the domain of Information and Communication Technologies (ICT). ETSI says that QoE is a measure of both the process and the outcomes of communications, and that technical parameters like QoS should be taken into account along with specific context variables [12].

- ITU-T defines QoE as being the overall acceptability of an application or service, as perceived subjectively by the user in an end-to-end context. Although ITU-T makes a remark that the user expectations and circumstances may influence the judgement [78].

To show the effect of different kinds of impairment, it is necessary to have a brief description of the major common impairments in multimedia services [88][66]:

- Block distortion (also known as blocking or blockiness): Distortion of an image characterized by the appearance of an underlying block encoding structure. This is often caused by an excessive data loss during the execution of the compression algorithm. Recent codec methods like H.264 have a deblocking filter in order to reduce the visibility of this kind of artefact.

- Blurring: Global distortion of the entire image, characterized by reduced sharpness of edges and spatial detail. The image appears unfocused. It is due to the suppression of higher frequency coefficients by the quantization process.
- Busyness: Spatial edge noise, is a variable distortion occurring on the image edges.
- Noise: There are two types of noise: Mosquito noise that is a form of edge busyness distortion usually associated with movement and characterized by moving artefacts and/or blotchy noise patterns superimposed over the objects, and Quantization noise that looks like snow effect, similar to a random noise process but not uniform over the image.
- Jerkiness: Motion that was originally smooth and continuous perceived as a series of distinct snapshots.
- Loss of information: In video the loss of reference will cause the appearance of black spots in the image, in audio it will result in an interruption of sound for a certain period of time.
- Echo: When part of the data sent in a bidirection channel return to the sender. This happens when a proper echo cancellation mechanism is not in use or if the delay is too big.
- Double Talking: When the delay its too big, data may be sent in two different directions at the same time.
- Time shifting: when a content is moved ahead in time.
- Packet discard at reception: When packets delay is too big, the packets are useless for a correctly stream decoding, so they are discarded.
- Slow channel navigation: Interactive commands coming from the user, take too long to produce effect, sometimes resulting in multiple interactions.

- Wrong synchronization between video and audio: This occurs because of different propagation delays for audio and video channels. Another reason is the error in the coding/decoding process.

Multimedia services

Quality of Service

QoS is a key requirement in IP-based multimedia applications. Its fundamental objective is to ensure predictable behaviour and high level of performance of the IP network. Different applications have different requirements regarding the handling of their traffic in the network. Applications generate traffic at various rates and this generally requires the network to be able to carry this traffic at the same rates that it is generated. In addition, applications are more or less tolerant to traffic delays in the network and to variation in traffic delay. Certain applications can tolerate some degree of traffic loss while others cannot. These requirements are expressed using the following QoS-related parameters: bandwidth, losses, jitter and delay [115].

The required bandwidth is related to the rate in which an application must be carried by the network. This calculation is complex because it is inherent to the content variation. However it is possible, if requested, for the video coder to generate different types of Bit Rates, in order to achieve a more adequate traffic variation. Requirements on losses and delays must be strictly obeyed because these regard to the period of time that an application can tolerate in delivering a packet of data. Small drops in packets can impair a significant number of video frames. On most cases, jitter is satisfactory when an adequate buffer size is established [88].

Multimedia services have different quality requirements when talking about audio or video services. In IPTV services the requirements are stricter, because the service is highly affected by jitter and losses and where high bandwidth is needed. VoD is less sensitive to delay and jitter. On the other hand audio services like VoIP calls are stricter in all QoS parameters (Delay, Jitter and Losses) because of the bidirectional data flow.

Quality of Experience

The QoE is defined as the overall acceptability of an application or service, perceived subjectively by users. QoE is a measure of end-to-end performance at the services level from the user perspective and an indication of how well the system meets the user's needs. The recommendation of the ITU-T defines QoE requirements from the user's perspective, in order to be agnostic to the transport network and the protocols used [78].

There is a nonlinear relation between the QoE subjective measurement as measured by the perceptual impact of forms of service degradation and the QoS objective parameters. Typically, when the QoS disturbance parameter increases, the QoE metric and user perception of quality decreases. 2.8.

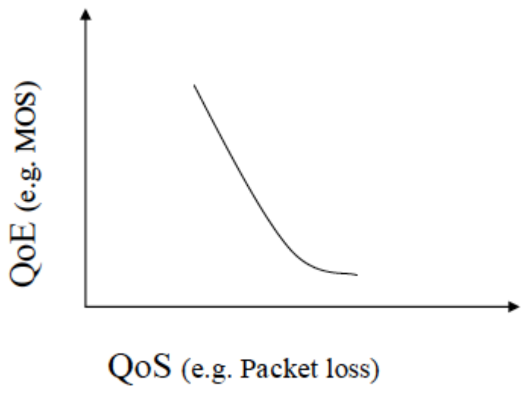


Figure 2.8: QoE relationship with QoS [1].

QoE have different disturbance in different kinds of multimedia services. Both audio and video services coding and the associated lossy compression can lead to effects like jerkiness and distortion of the original sound and/or image. Video services are more sensitive to impairments, but sometimes besides the jerkiness and the block distortion, also busyness can be associated with the coding process. Transcoding processes in audio services lead to noise and several losses of information. Besides these two impairments, video may also have some blurring or unfocused image. Delays can be responsible for impairing the synchronization between video and audio. Other losses and transmission errors can be translated as black spots on the screen. Table 2.5 resumes the impairment factors caused in multimedia services [88].

Table 2.5: Impairments to video and audio services (Based on [88]).

Service	Coding and Compression	Transcoding	Delay	Jitter	Losses and Transmission Errors
VOIP	Loss of information, Jerkiness, Block distortion	Loss of information, Noise	Echo, Double talking	Time shifting, Packet discard at reception	Jerkiness
IPTV	Loss of information, Jerkiness, Block distortion, Busyness	Loss of information, Noise, Blurring	Time shifting, Slow channel navigation, Impaired video/audio sync.	Time shifting, Packet discard at reception	Blank spots, Jerkiness
VOD	Loss of information, Jerkiness, Block distortion, Busyness	Loss of information, Noise, Blurring	Time shifting, Impaired video/audio sync.	Time shifting, Packet discard at reception	Blank spots, Jerkiness

Standards and models for video quality assessment

The QoE can be assessed in several ways, in the particular case of video, image quality can be assessed in three ways[1]:

- Subjective measurements: Using an experimental evaluation method where a number of participants assign a rating on a predefined scale of values.
- Objective measurements: Analysing parameters of the video signal (eg, Peak-to-peak Signal-to-Noise Ratio (PSNR), Mean Square Error (MSE)) using specialized measuring equipment. Compare the video signal that is transmitted over the network with the source video signal.
- Indirect measurements: Performing measurements of various network parameters (delay, jitter, packet loss) for the purpose of estimating the impact of these parameters on the visual quality using the relationship between QoS and QoE.

Although in the scope of this work just two of the main quality assessment methodologies are discussed, namely subjective and objective assessment methods, figure 2.9 gives an overview of the main quality assessment methodologies.

Subjective quality assessment involves human participation subjects and controlled viewing experiments, with standardized equipment in order to achieve results. Due to the high level of complexity of the subjective assessment process, an electronic approach is often used, where a software measures various aspects of video signal and predicts the user satisfaction using objective measurements [88].

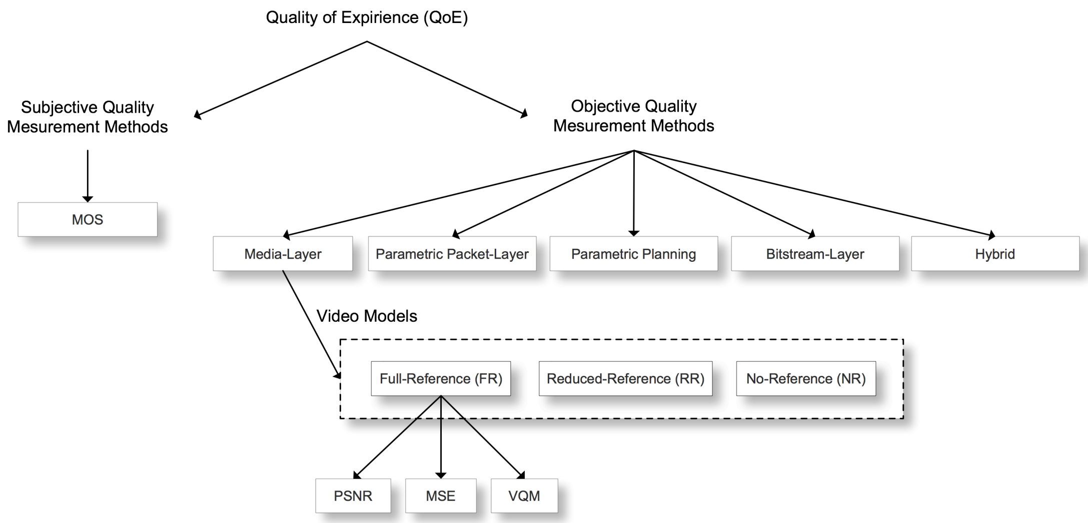


Figure 2.9: Classification of media-layer objective and subjective video quality models [49].

Subjective quality assessment

Subjective measurements are performed using human viewers to rate the quality in one of several possible testing methodologies specified by the ITU-T Recommendation P.910 (quality assessment methods for video applications) [68], Recommendation P.911 (quality assessment methods for multimedia applications) [67] and the International Telecommunications Union Radiocommunication Standardization Sector (ITU-R) Recommendation BT.500 [83]. Subjective measurements can provide an accurate assessment of the video quality, as well as reflect the video experienced by the end user. However, since each user has different interests and expectations, the subjectivity of the viewer evaluation cannot be completely eliminated. Besides the user interference in the subjective assessment, subjective experiments try to minimize factors through precise instructions and using training and controlled environments. Yet it is important to remember that a quality score is a noisy measurement that is defined by a statistical distribution rather than an exact number. The most used subjective assessment technique is the Mean Option Score (MOS).

MOS is a subjective measurement indication, which ranks the video quality based on the user feedback. In MOS users determinate the quality by rating the quality of the video displayed on a scale of 1 (very bad) to 5 (very good) as showed in Table 2.6. This scale is used when users are rating videos with single sequences without comparing them with a video reference.

On the other hand, table 2.7 shows a scale to establish a relative judgement between a pair of video sequences to evaluate degradation of the image[117].

Table 2.6: Quality and Impairment scale with MOS conversion [88].

MOS	Perceived Quality	Impairment
5	Excellent	Imperceptible
4	Good	Perceptible, but not annoying
3	Fair	Slightly annoying
2	Poor	Annoying
1	Bad	Very annoying

Table 2.7: MOS scale for the comparison between samples [88].

Score	Impairment
-3	Much worse
-2	Worse
-1	Slightly worse
0	The same
+1	Slightly better
+2	Better
+3	Much better

The main drawbacks of this approach are: it is expensive in cost, time consuming, cannot be used in real time and lacks repeatability. These limitations motivated the development of objective tools that predict subjective quality solely from physical characteristics.

Objective quality assessment

The objective assessment method is an instrumental measurement process that outcomes variables, based on physical features of the media signal, service and transmission in order to approximate subjective evaluation and to allow faster results and conclusions [12].

Video objective assessment methods can be mapped into five different models depending on the type of input information that is being used for quality assessment, as listed in Table 2.8 [111].

- **Media layer model:** This model uses video signals to approach human quality benchmarking, focusing on the emulation of Human Visual System (HVS). It does not require any information about the system under testing, such as codec type or packet loss rate, so it can be applied to the evaluation of unknown systems such as codec comparison and/or optimization. However if media signals are not available, this model cannot be used.
- **Parametric packet layer:** Predicts QoE only from the packet header information, and is not adequate to judge content-related features like distortion or to monitor individual user flows. It does a lightweight measurement solution for predicting QoE that does not include inspecting the media signals. This type of model is aimed for real-time non-intrusive monitoring at the network layer. Possible examples for the use of this model are the evaluation of services as VoD or IPTV.
- **Parametric planning model:** This model makes use of quality planning parameters for networks and terminals in order to predict the QoE. However this type of model requires a priori information about the system under testing. A good example for the use of this model is the ITU-T Recommendation G.107, the E-Model [79], which has been used as a network planning tool for the PSTN, but recently it was adopted to the planning of VoIP and IPTV networks [111].
- **Bitstream layer model:** This model is a recent concept; it occupies a position between the Media layer model and the Packet layer model. This model obtains the QoE by extracting and analysing content characteristics from the encoded bitstream and from the packet layer informations that is used in the packet layer model. This model is a useful non-intrusive monitoring with the advantage of allowing QoE monitoring for individual users.
- **Hybrid Layer:** It is a combination of the previously mentioned models. It is effective in terms of exploring as much information as possible to successfully predict

the QoE. This model is also aimed at non-intrusive monitoring, however accepts a combination of different types of inputs.

Table 2.8: Categories of objective quality assessment models (Based on [111]).

	MODELS				
	Media Layer	**Parametric Packet Layer**	**Parametric Planning**	**Bitstream Layer**	**Hybrid**
Input information	Media signal	Packet header information	Quality design parameters	Packet header and payload information	Combination of any
Application scenario	Quality benchmarking	In Service non-intrusive monitoring (e.g. Network probe)	Network planning terminal/ application designing	In Service non-intrusive monitoring	In Service non-intrusive monitoring

In the scope of this thesis, our attention was focused on the media layer model because the media layer quality model approximates the human subjective evaluation, which allows us to access the contribution of objective QoS in the overall perceived quality of video.

Media Layer Model

Media Layer Model estimates the audio and/or visual QoE by using media signals such as the speech waveform and video pixel data. Table 2.9 shows the ITU-T recommendations for media layer model[111].

Table 2.9: ITU-T Recommendations for objective assessment methods.

Service Type	**Media layer model**
Voice	ITU-T P.862
Audio (general)	ITU-R BS.1387
Video	ITU-T J.144 (SD) ITU-T J.341 (HD) ITU-T J.247 (PC)
Multimedia	ITU-T J.148

For voice, ITU-T developed the Recommendation P.862 [71] (Perceptual Evaluation of Speech Quality (PESQ)) which predicts the speech quality for narrow band transmissions. For audio (general), ITU-R standardized recommendation BS.1387-1 [69] that describes a method for predicting the perceptual audio quality objectively. However, BS.1387

cannot be used to evaluate the effects of packet loss, which is one of the most important quality factors in recent applications. For video ITU-T standardized recommendations J.144 [70], J.341 [82] and J.247 [77]. The first standard describes techniques for objective estimation of the perceptual video quality for digital cable television when Full-Reference (FR) is available. Recommendation J.247 does the same for general multimedia video and J.341 for HDTV. Last but not least, recommendation J.148 defines the requirements for multimedia perceptual model for audio-visual services [88].
Objective measurements compare the output video signal with the source video signal in order to find the differences. These differences are used to define how the decoded stream derives from the original. Greater differences between the original and the decoded stream are associated to lower quality of the received stream. The Media layer objective quality assessment methods can be further categorized to three different approaches: FR, No-Reference (NR) or Reduced-Reference (RR) depending on the quantity of information about the reference that is used to assess the quality [70].

- Reduced Reference (RR): RR method evaluates the video quality by comparing processed video, subjected to distortion by coding and transmission losses, with a small amount of information extracted from a source video (Figure 2.10). Since RR method uses features from the source video and degraded video signal, this evaluation is fairly accurate. RR model assumes that a side channel is available to transmit the reference signal parameter data.

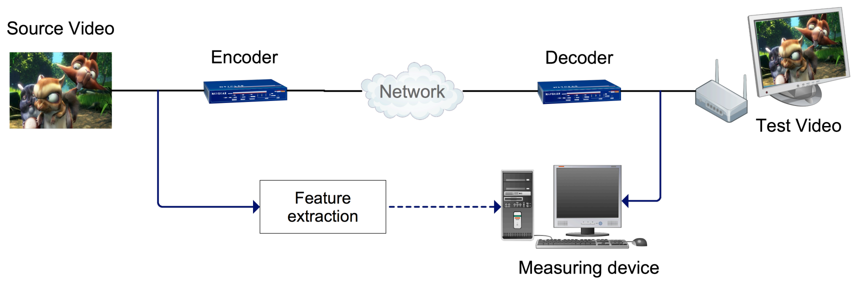


Figure 2.10: Reduced Reference method.

- No Reference (NR): The NR method operates solely on information extracted from the processed signal, that was subjected to distortion from coding and transmission losses (Figure 2.11). Since this method does not require any information from the source video, it can be used in environments where the source stream is not available, which typically is the case of IPTV or broadcast TV. This method is less accurate because it does not use information from the source.

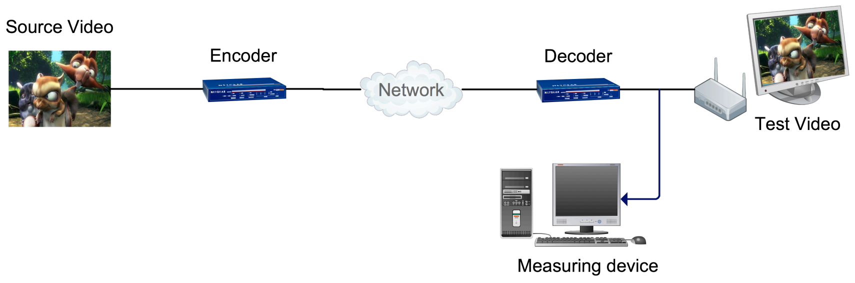


Figure 2.11: No Reference method.

- Full Reference (FR): The FR method compares the information of the source video (before any distortion has occurred), with the processed video after distortion caused by the coding and transmission losses, in order to objectively evaluate the degraded video (figure 2.12). Since the FR model compares source video with processed video, it provides a highly accurate objective assessment of the quality, reflecting its basic characteristics. The FR method is best suited for video quality metrics for VoD networks. However, it needs a large amount of resources in order to obtain data from the original video and to make a valid comparison.

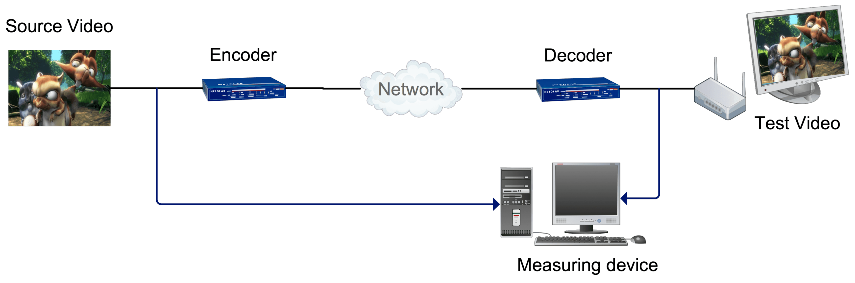


Figure 2.12: Full Reference method.

Some examples of metrics for FR assessment are showed below:

Mean Square Error (MSE) is computed by averaging the squared intensity of the deviation of the original image and the resultant image so it measures the difference between the original and the distorted image in a pixel basis as showed in equation 2.1.

$$MSE = \frac{1}{M.N}\sum_{i=1}^{M}\sum_{j=1}^{N}[f(i,j) - F(i,j)]^2 \tag{2.1}$$

Where $M.N$ is the number of pixels in the image or video signal, $f(i,j)$ is the original signal at pixel (i,j) and $F(i,j)$ its the reconstructed signal.

The Peak-to-peak Signal-to-Noise Ratio (PSNR) is a mathematical measure that is computed by taking the Root Mean Square (RMS) value from the differences (errors) of the original and the received video frames, often normalized to be expressed in dB. The value of PSNR can be calculated using equation 2.2,

$$PSNR = 10.\log_{10}\left(\frac{s^2}{MSE}\right)[dB], \tag{2.2}$$

where s is maximum possible pixel of the image, when the pixels are represented using 8 bits per sample, $s = 255$ ($2^8 - 1 = 255$).

Video Quality Metric (VQM) is an HVS metric and measures the perceptual effects of different kinds of video impairments such as blurring, jerky motion, block distortion, global noise and combines them into a single metric. VQM has a high correlation with subjective quality assessment.

PSNR and MSE are among the most important and widely used as reference methods for comparing the performance of video coding algorithms. There are a number of reasons for their popularity, partly due to their easy understanding and implementation and also the fast computation and broad acceptance by the scientific community. Another reason for their popularity is that both metrics are very well understood from a mathematical point of view. Despite the popularity of these metrics PSNR does not take into account basic concepts like spatial relationship between their pixels or concepts more complex like interpolation of images and image differences by HVS [120]. In 2007, PSNR was included by the Video Quality Experts Group (VQEG) as a reference to objective models in the test plans. These tests lead the ITU-T standards on FR assessment models [110]. Differently from the PSNR and MSE, VQM metric is based on simplified human spatial-temporal contrast sensitivity. Xiao, F. [55] proposed a modified Discrete Cosine Transform (DCT) based on VQM, which exploits the properties of visual perception, using the existing DCT coefficients, so it only incurs slightly more computation overhead. Starting in 2004, VQM method was adopted as ITU recommendations ITU-T J.144 and ITU-BT.1683. Based on the results obtained in [122], a mapping of VQM to a nominal 5-point MOS scale is showed in Table 2.10, which provides an approximation to subjective QoE.

Table 2.10: Mapping of MOS and VQM [88].

MOS	**VQM**
5 (Excellent)	< 0.2
4 (Good)	≥ 0.2 and < 0.4
3 (Fair)	≥ 0.4 and < 0.6
2 (Poor)	≥ 0.6 and < 0.8
1 (Bad)	> 0.8

3

Triple-play Model

3.1 Introduction

This chapter presents the architecture of the solution developed for the Triple-play system based on a web platform and targeted for Wireless Local Area Network (WLAN). The architecture shows both client and server topologies and includes both audio and video media. The chapter begins by presenting an overview of the Triple-play model and then describes all the functionalities and various modules that compose the full system.

3.2 System Model

This work includes the development of a customized Triple-play client and server. The server deploys the contents (Internet Protocol Television (IPTV), Video on Demand (VoD) and Voice over Internet Protocol (VoIP)) to the clients. The client was designed in parallel with the development of the server. The propose of this system is to distribute IPTV, VoD and VoIP content to customers in a wireless environment.

The client prototype (Set-Top Box (STB)) integrates the services into a web based architecture that can be connected over wireless networks. The STB implements a client application, that includes a streaming client, session control functions and video/voice services.

Initially, for video services the client obtains the addresses of IPTV streams and VoD streams that it can consume. Each stream can be associated with a broadcast channel or pre-recorded film. The client logs on to the server by choosing the IPTV or VoD stream desired. If the content is available, the server starts sending the stream to the client. On the other hand, for voice services each client must be authenticated with the VoIP server. After the authentication clients can initiate/receive calls from other clients that are authenticated. During the preview, clients can interact with the application. Functions such as Pause, Play, Stop, re-Play, Volume Up, Volume Down and changing the video can be used for VoD; changing Television (TV) channel, Volume Up and Volume Down can be used for IPTV and finally phone calls can be started for VoIP.

The server is responsible for providing the contents of IPTV, VoD and VoIP to the clients. These contents are available through three different modules that compose the server: IPTV, VoD and VoIP. Each of these modules use different types of protocols and different types of compression methods for the delivery of contents. The server may use a WLAN environment for the delivery of contents to the clients.

The Triple-play system includes an application module for monitoring Quality of Service (QoS) in the network. Statistics about packet loss and latency are periodically collected with this module. This data allows the application to monitor possible changes in network conditions and to react, with the objective of minimizing the negative impact on viewing experience, increasing the Quality of Experience (QoE).

3.3 Triple-play architecture

This section presents the general Triple-play system architecture of both server and client. The overall architecture is composed by three components: the server, the client and the transport network. The server is responsible for creating the connection between the server and the clients (STB). This includes everything used to capture the video streams as well as all connections and streaming functionalities. The client receives complete media samples, synchronizes them and plays them, finally the transport network receives media samples from the server, forms packets and transports the packets to the client. The client recombines them and makes them available for rendering.

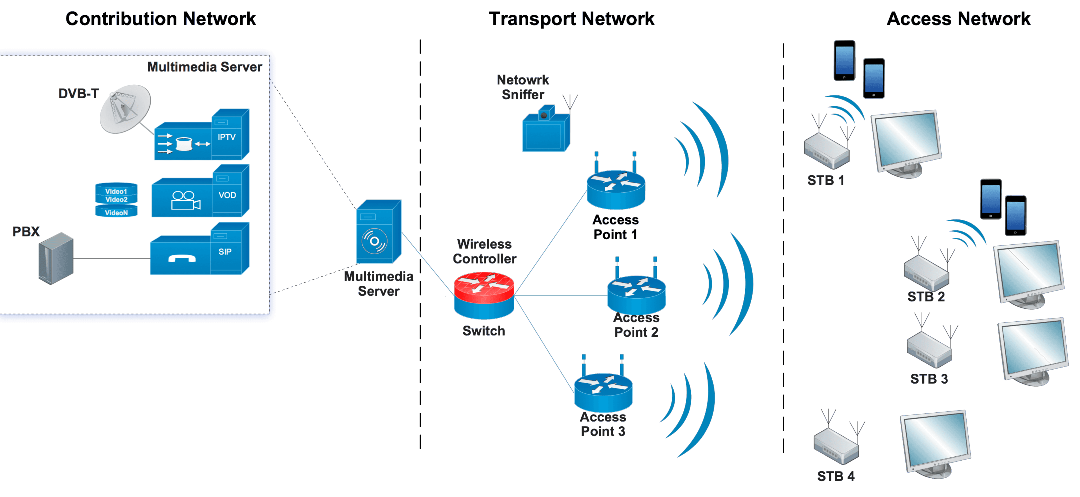


Figure 3.1: General Triple-play architecture.

The topology used is a basic network topology, that is composed by three main networks: the *Contribution network*, the *Transport network* and the *Access network*, as shown in figure 3.1. The contribution network contains the multimedia server that is divided into three modules: the IPTV module, the VoD module and the VoIP module. These three modules are responsible for delivering different types of traffic and media information. The first module, IPTV, is responsible for the delivery of television contents to the client. The transmission is done via Real-Time Transport Protocol (RTP) with multicast traffic between the server and the client. The VoD module delivers pre recorded videos and the transmission is done via Real-Time Streaming Protocol (RTSP) unicast, which allows the client to use Play, Pause and Stop functions. The third module (VoIP) delivers audio conversations using a Session Initiation Protocol (SIP) architecture. In this case, signalling is done using SIP. Later in this chapter each of these modules will be scrutinized.

The *Transport network* is responsible for the content transportation between the two peers (server and client). As shown in figure 3.1 this network is composed by three components: the wireless controller, the network sniffer and the Access Points (AP). The wireless controller is responsible for controlling all the connected AP. It is used as a Dynamic Host Configuration Protocol (DHCP) server and it also comprises some monitoring features. The network sniffer is used for traffic monitoring and is responsible for gathering information about the network traffic. This information will be later used to get some conclusions about jitter, packet loss and bandwidth network metrics. Finally, the *Access network* is basically composed by the STB, which is responsible for consuming the media contents provided by the server.

3.4 Architecture module specification

This section presents detailed information about all of the three networks (*Contribution*, *Transport* and *Access*), as well as the modules that compose the system.

IPTV

Module requirements

The system is composed by three main components: a multimedia server, a transport network and a client. Based on the scenario description the following IPTV module requirements are formulated:

- The server must support a scalable TV channel delivery.
- The client, the server and the network should be able to process RTP/Real-Time Control Protocol (RTCP) packets in order to achieve a successful streaming transmission.
- Network should support the transmission of multicast packets, adjusting its transmission in the WLAN according to the clients' interests.
- The IPTV client must support the reception of IPTV streams which are transmitted via multicast and must be able to decode and display the multicast stream.
- It must support human interaction in order to be able to change between TV Channels and do simple operations as volume up and volume down.
- Stream transactions or TV channel swaps have to be done as fast as possible (the prototype measured latency is below 5 seconds).
- The IPTV service must work over WLAN with a successful adaptation for multicast.
- The system aims to provide the user with acceptable QoE (Peak-to-peak Signal-to-Noise Ratio (PSNR) $\sim$ 40 dB).

Module description

The IPTV has a modular architecture. Each module is responsible for a specific function. The diagram in figure 3.2 represents the modules that compose both client and server structures, as also the transport network between them.

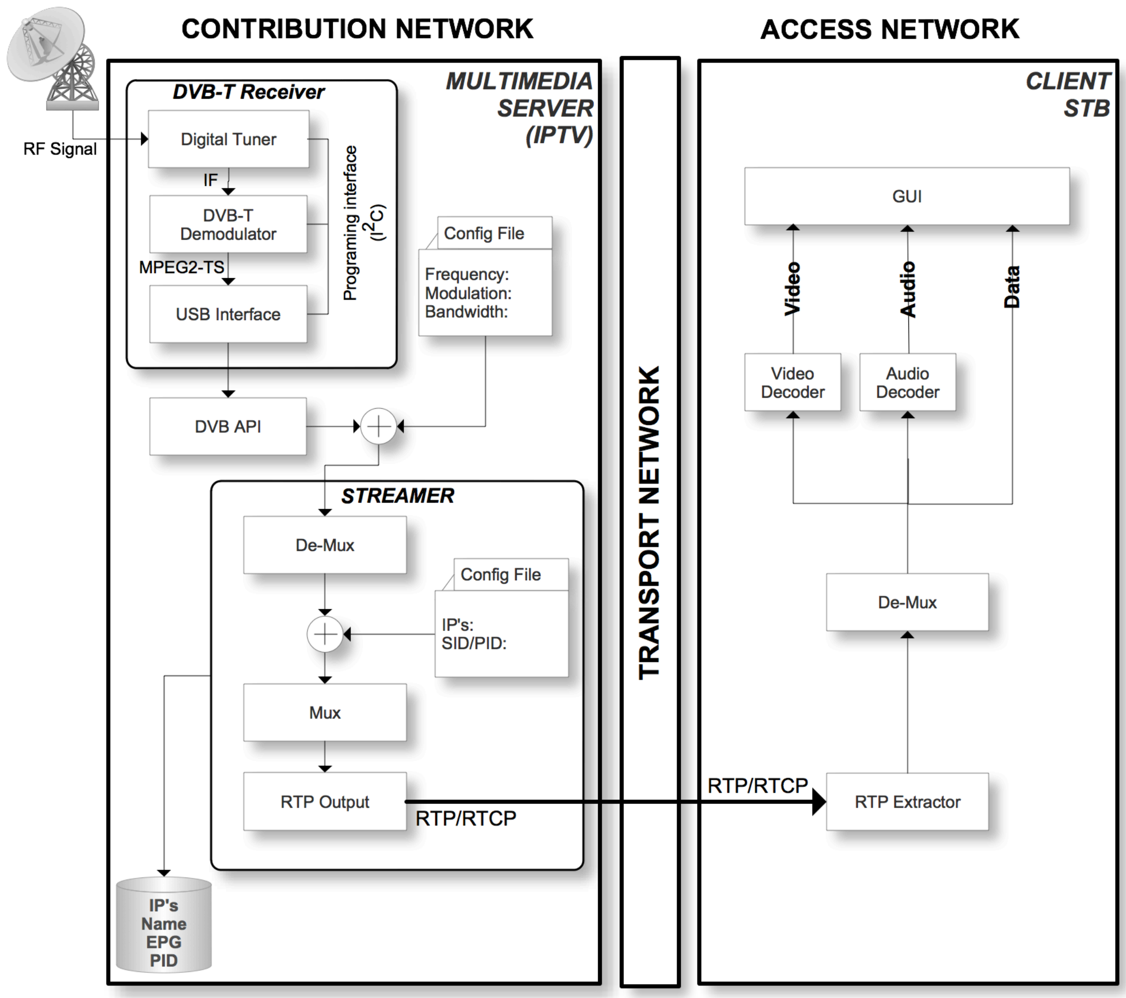


Figure 3.2: IPTV Module.

The process of IPTV streaming starts with the image receiver that captures video data. This image receiver is a Digital Video Broadcasting - Terrestrial (DVB-T) receiver that receives the Radio Frequency (RF) signal from the broadcaster using an antenna. The DVB-T receiver is composed by three main modules: the digital tuner, the DVB-T demodulator and the Universal Serial Bus (USB) interface.

The digital tuner receives the RF signal and is responsible for "tuning" the desired frequency and providing an analogue signal to the DVB-T demodulator (in digital transmissions the signal that comes out of the tuner is an analogue waveform commonly known as Intermediate Frequency (IF)). The demodulator is responsible for taking the analogue waveform and "demodulate" it to its digital form. The demodulator outputs MPEG-2 Transport Stream (MPEG-2 TS) packets and send them over a USB interface. The USB interface is another central part of the digital tuner as it makes the connection to the server and is responsible for all the communication between them (digital tuner, DVB-T and USB interface). Another important component of the DVB-T receiver is the programming interface (I2C) that is responsible for the initialization and configuration of the digital tuner. In other words, it is through this programming interface that the tuner knows which frequency has to tune. The frequency is specified in a configuration file and is forwarded to the digital tuner by the Digital Video Broadcasting (DVB) Application Programming Interface (API).

The DVB API module is responsible for making the communication between the DVB-T receiver and the configuration file. The configuration file is where the information about the signal frequency, bandwidth and codification is stored.

The next stage in the streaming process starts with the demultiplexing of the MPEG-2 TS signal of the DVB-T receiver. This part of the process is done by the streamer which is responsible for pre-processing and preparing the RTP packets. The streamer is composed by three modules: the Demultiplexer (DE-MUX), the Multiplexer (MUX) and the RTP output. The DE-MUX is responsible for the demultiplexing of the DVB-T signal. The MUX is responsible for the multiplexation of the audio, video and data, according with the specific Packet Identifier (PID). The RTP output module is responsible for the packetization into RTP of the multiplexed packets. These RTP packets are then transferred to ethernet using User Datagram Protocol (UDP). The streamer module is also responsible for storing the stream information (Internet Protocol (IP) address, name, PID).

The client communicates with the multimedia server using the channel's unique IP address that is stored into the data base. This process starts with the reception of RTP packets that are processed by the RTP extractor module. The DE-MUX extracts the video, audio and information data from the packets. The recovered data are presented through the Graphical User Interface (GUI).

This section brief explained how the packet is processed both in the server and in the client of the IPTV service. More specific information about the components of the DVB-T receiver and the streamer is discussed later in this section.

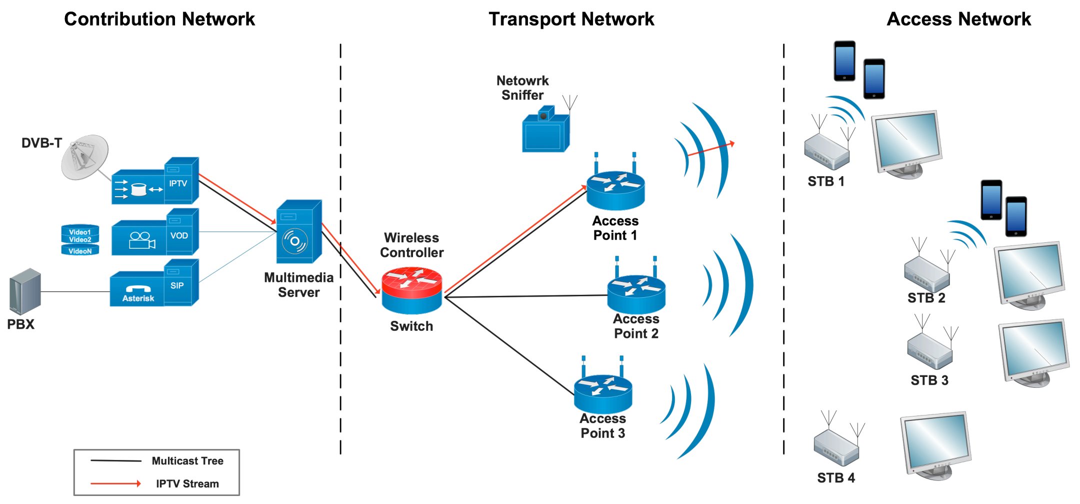


Figure 3.3: IPTV distribution network.

IPTV service offers real-time TV Channels to clients, which are being transmitted via multicast distribution, as seen in figure 3.3. All IPTV channel streams are constantly available in the transport network of the IPTV service and are only sent to the IPTV client when the user starts watching a certain TV channel. When a user selects a TV channel, the STB (client) joins the multicast group which is used for the transmission of the IPTV stream. The packets are sent to the multicast address via the access link and are forwarded to the client's STB. As the IPTV clients are connected via WLAN, the quality of the different access links from the client to the server over the transport network may differ and some access links may suffer some packet loss. This packet loss may influence the IPTV service, and may result on visual impairments or playback problems, leading to a unsatisfactory TV viewing experience. Section 5 will show the influence of these impairments in different scenarios of a WLAN system.

Server design specification

The purpose of the IPTV server is to provide IPTV channels using RTP over UDP. The server continually sends an IPTV packet stream to a multicast group address. The detailed process of streaming is discussed in the subsequent sections.

1. Capture and conversion: Information about capture and conversion methods can be found in appendix B.1.

2. Streaming:

   - Encapsulation

     Once the data are received by the DVB-T receiver the process of streaming begins. The streaming block operation starts with the demultiplexation of the MPEG-2 TS. The MPEG-2 TS (as seen in appendix A.1) is formed by multiplexed Packetized Elementary Streams (PES) packets and its header is composed by the PID, the unique address identifier. During the formation of the Transport Stream (TS), a set of database tables called Program Specific Information (PSI) tables are added, which describe the structure of the TS. The streamer block demultiplexes the MPEG-2 TS, filters the right packets based on the configuration file (where the information about the PID of the chosen programs is stored) and rebuilds the mandatory PSI tables with the new information.

     The RTP encapsulation starts after the multiplexing of the programs. In RTP encapsulation, data samples are picked from the multiplexing block and placed in the data field of the RTP packet. A RTP header is added in order to make the stream more "network friendly" and the synchronization of the streams is done by the RTCP.

   - Transportation

     As soon as a packet is formed, it is forwarded to the UDP socket associated to the physical port and is broadcasted. The packet is routed to the clients using the IP address.

Client design specification

The client is formed by an application that plays the received video data in real-time. The client program separates data from protocol headers and plays it. The client uses the same communication network deployed on the server (section 2 - Transportation). The RTP extractor extracts the RTP header and sends the data to demultiplexing process. The demultiplexer splits the stream into four outputs. The first output delivers the PSI packets to the Transport Information Filter (TIF); the second one delivers the video stream to the MPEG-2 decoder; the third one delivers the audio to the audio decoder; and the fourth one delivers the data. The basic function of the decoder is to take out the header data from the input stream, extract the picture, audio and user data information and decode compressed Elementary Streams (ES) that has been generated by the broadcaster. This information is finally presented to the user with resource to a GUI that reproduces all the video, audio and data.

VoD

Module requirements

Following a similar components structure as IPTV module, the requirements for the VoD module are:

- The client, the server and the network should be able to process RTSP packets in order to achieve a successful streaming transmission.
- The client and the server should be able to support human interaction, should be able to execute commands such as start, stop, pause and volume up/down in the VoD stream.
- The VoD service must work in a WLAN with an acceptable QoS.
- The VoD client must support the reception of VoD streams via unicast and should be able to decode and display the unicast stream.

Module description

The VoD module works similarly to the IPTV module. The specific VoD distribution architecture that was designed in the scope of this thesis is showed in figure 3.4. It is composed as before by the three main modules: the server, the client, and the transport network.

The server stores the original version of the video object. This server decides which content it offers and when this new content is offered. The suppliers of the streamer module are the video objects which are responsible for supplying the video content. Besides, there is one file that is responsible for the mapping between the videos stored in the content storage and the local addresses of the streams. When the streamer collects all these information and configuration parameters, it stores everything into a Data Base (DB). The streamer module also has the function of creating the RTSP packets and injecting them into the transport network to be available for the client.

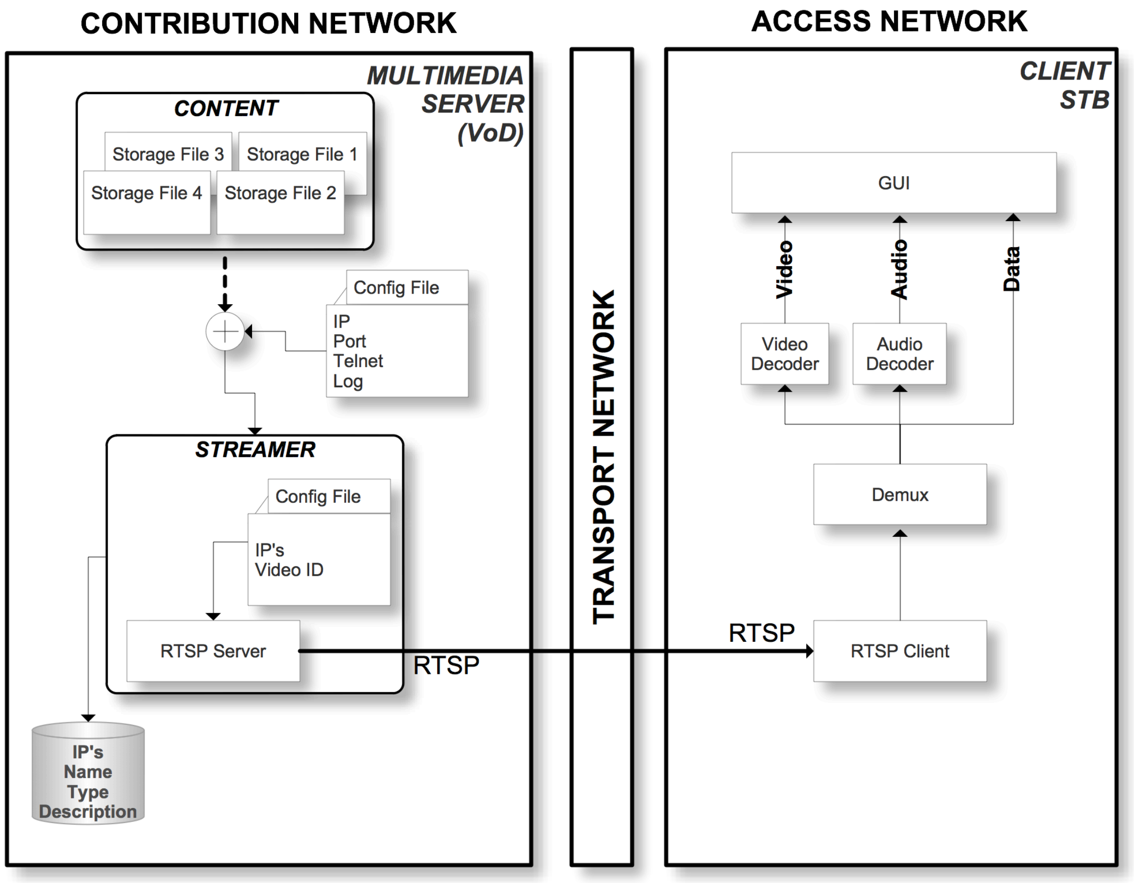


Figure 3.4: VOD Module.

The client is responsible for receiving the RTSP packets that are being sent from the server. The client uses a module called RTSP client that handles the RTSP session initiation and the receiving of the RTSP packets. After this, the RTSP packets are processed in order to have a full visual stream. The stream is separated into three types: audio, video and data. In the final stage, audio, video and data is decoded to be presented by the GUI.
From the Transport Network point of view, VoD service offers videos to its clients via an unicast distribution network. As the clients open the RTSP streams, new connections are created between the server and the client. When the client asks the server for a new RTSP stream, the server creates a new instance and a new connection to this client and the transmission is started. The packets are sent via unicast and are forwarded to the client, which will decodes these data and display the video. Figure 3.5 shows how this process is done.

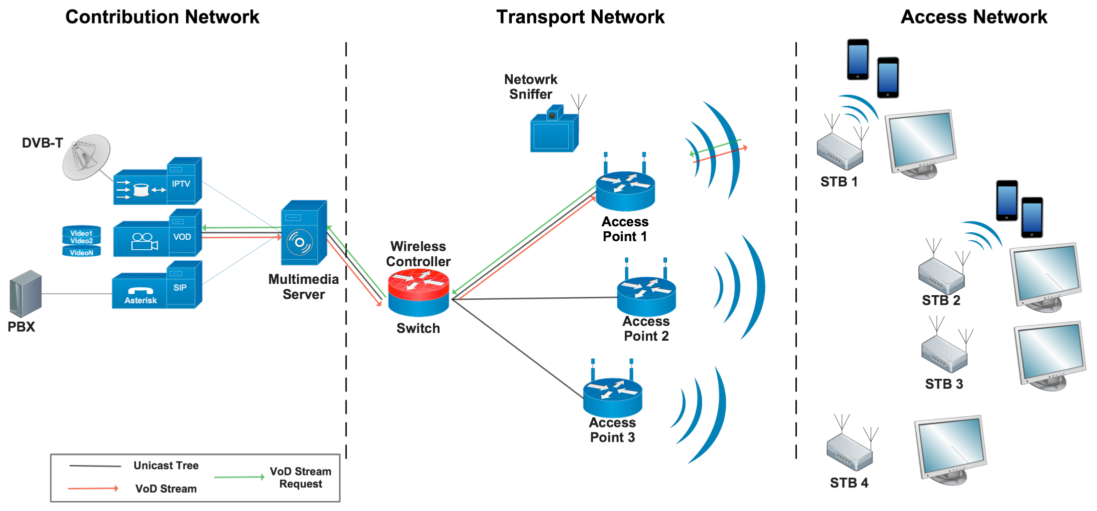
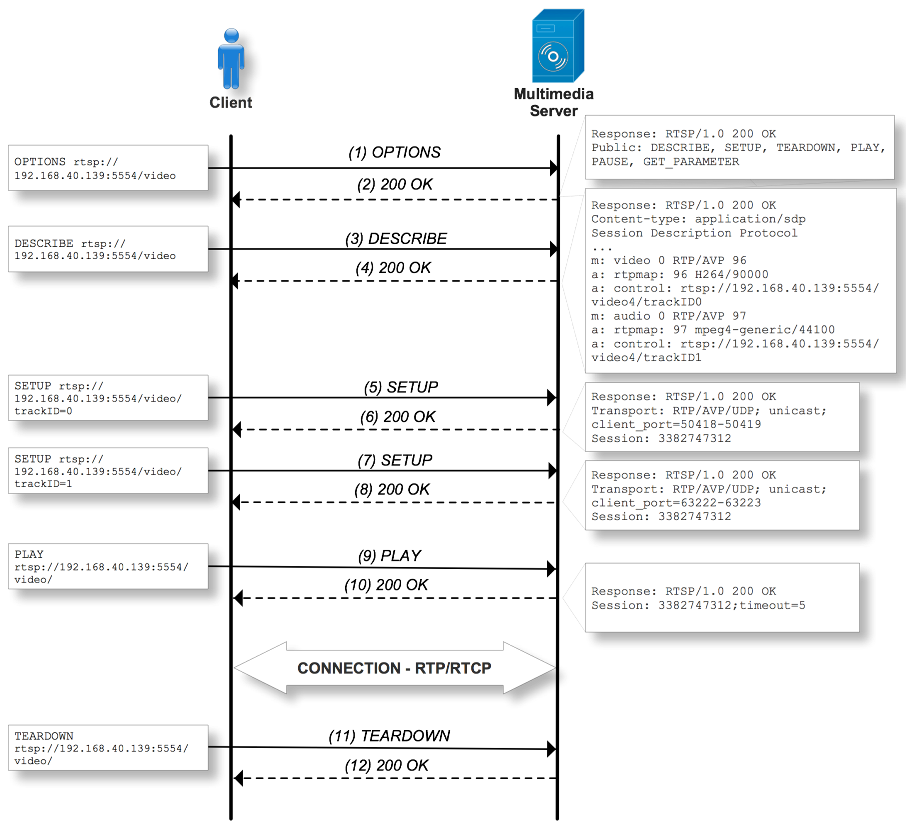


Figure 3.5: VOD distribution network.

Communication model

The client in this module has an important role, since it is the one that establishes the VoD session through the use of commands, as seen in section 2.1. A possible message diagram of a RTSP starting session is showed in figure 3.6. Several phases are identified:



```
OPTIONS rtsp://
192.168.40.139:5554/video
```

```
Response: RTSP/1.0 200 OK
Public: DESCRIBE, SETUP, TEARDOWN, PLAY,
PAUSE, GET_PARAMETER
```

```
DESCRIBE rtsp://
192.168.40.139:5554/video
```

```
Response: RTSP/1.0 200 OK
Content-type: application/sdp
Session Description Protocol
...
m: video 0 RTP/AVP 96
a: rtpmap: 96 H264/90000
a: control: rtsp://192.168.40.139:5554/
video4/trackID0
m: audio 0 RTP/AVP 97
a: rtpmap: 97 mpeg4-generic/44100
a: control: rtsp://192.168.40.139:5554/
video4/trackID1
```

```
SETUP rtsp://
192.168.40.139:5554/video/
trackID=0
```

```
Response: RTSP/1.0 200 OK
Transport: RTP/AVP/UDP; unicast;
client_port=50418-50419
Session: 3382747312
```

```
SETUP rtsp://
192.168.40.139:5554/video/
trackID=1
```

```
Response: RTSP/1.0 200 OK
Transport: RTP/AVP/UDP; unicast;
client_port=63222-63223
Session: 3382747312
```

```
PLAY
rtsp://192.168.40.139:5554/
video/
```

```
Response: RTSP/1.0 200 OK
Session: 3382747312;timeout=5
```

```
TEARDOWN
rtsp://192.168.40.139:5554/
video/
```

Figure 3.6: RTSP message flow client-server.

- Session establishment

  1. The VoD session establishment begins with the sending of an option message to the server, which contains the Uniform Resource Identifier (URI) of the video. This message is sent in order to know which methods the VoD server supports.
  2. The server answers with an RTSP 200 OK and with the compatible list of commands.
  3. With the available list of commands, the client sends a describe message to know which streams compose the video.
  4. The server answers with an OK and with the initialization data.
  5. The client requests for a specific stream, by sending a message of setup with the desired session parameters and transport mechanisms.
  9. To conclude the session establishment, the client sends a play message to the server with the desired URI of the stream. This message informs the server to start sending data via the transport mechanism specified in the setup message.

10. The server answers with an OK and the RTP packet flow starts. In case of unavailability of the stream requested, an error message is returned and the session is terminated.

- Control session

  While viewing the stream, the user can control the session, for example by pausing the stream. This command is done by sending a message of pause to the server. This message indicates to the server that it must stop sending the RTP stream. The server is responsible for storing the session state, namely the instance where the video should resume.

- Session closing

  The client or the server may send a teardown message (**11**) indicating the end of the session and the release of the process resources.

Open source software streaming solutions

This section presents the different open source multimedia server solutions available at the moment in the market. There are mainly three streaming media open source software: Red5, Darwin and VLC.

- Red5 open source media server

  Red5 [19] is an open source software streaming server that delivers Flash files. Red5 is based on Java and supports video/audio streaming. Red5 also supports real-time multiplayer gaming, video conferences, live streaming and client stream recording. As a summary Red5 is a clone of the Adobe Flash Media Player. Red5 supports file formats such as mp3, FLV and Real-Time Messaging Protocol (RTMP).

- Darwin streaming server (DSS)

  Darwin [17] is an open source streaming server based on the same code as Apple's QuickTime streaming server. Darwin has support for streaming MPEG-4 and Quicktime media on alternative platforms such as Mac OS , Windows server, Linux Fedora. Darwin allows streaming media to clients using standard protocols as RTP and RTSP.

- VideoLAN streaming solutions

  The VideoLAN [37] project offers several open source multimedia streaming solutions. The VideoLAN project hosts the VLC Media Player and the VLS, also known as VideoLAN Server. VLC was created to be a universal media client capable of playing streams from multimedia files, multimedia drives or network. VLC is fairly complete because it incorporates features of server and client in the same software. VLS is a multimedia streaming server that provides interfaces to a number of multimedia input devices such as digital satellite and terrestrial TV cards or encoding hardware. However VLS project is no longer available and VideoLAN has focused on VLC project. VLC is composed by several modules and the VideoLAN Manager (VLM) is one of them. The VLM is a media manager designed to control multiple streams with VLC. Thus VLC can act as a RTSP server with streaming functionalities. VLC supports a variety of audio and video formats including MPEG-2, MPEG-4, DIvX, mp3, H.264 and others [36]. VLC can be used as a server to stream multimedia data in unicast or multicast. VLC software is able to run on most operation systems such as: Linux systems, BSD systems, windows, Mac OS.

In the scope of this work the VLC streaming solution was chosen for the implementation of the server (both IPTV and VoD modules). VLC Plugin is the chosen application for the web integrated solution and is also part of the Videolan project. The choice of VLC for the streaming server is due to the large number of available features, in particular the diversity of formats of input and output encoders and decoders. Another important feature of VLC is that it is a free open-source cross-platform software with no hardware requirements. The fact that it has a large community working on it is another important feature.

VoIP

The VoIP module has a similar network structure and is composed by the same modules as the two previous ones, however this one has slightly different features and a distinct structural model. The VoIP server is mainly composed by a modular architecture that is implemented using the Asterisk Private Branch Exchange (PBX) module. This module is composed by smaller modules that can be seen in figure 3.7. There are five modules in Asterisk IP PBX: the PBX core, channel API, the codec translator API, the file format API and the application API [10]. The use of these APIs allows the complete abstraction between the protocols and the hardware.

- The PBX core is the heart of the asterisk module and it is the essential component that takes care of the bridging calls. The core is also responsible for reading the configuration files and the loading of the other modules.

- The channel API handles all the connections required by a call. In this model there are two types of channel modules: the Digium Asterisk Hardware Driver Interface (DAHDI) and the SIP. The DAHDI is responsible for the connection to the ordinary telephone line through a Foreign Exchange Office (FXO) interface. The other channel module SIP is a basic signalling protocol to perform call processing in Asterisk module. This module uses RTP/RTCP to transmit data.

- The codec translator API executes codec modules that support encoding and decoding of the various audio formats.

- The file format API enables the reading and writing of various types of files in order to allow the data storage in the file system.

- Finally the application API allows external modules with specific tasks to be executed. These applications can be voice mail, conference calling, data transmission, call waiting, among others.

The client in this case is also composed by one main block: the applet. The applet is an independent platform based on the open standard SIP that allows making VoIP calls. This block is responsible for making the RTP communication between the client and the server. Another important block in the client structure is the java script API that makes the customization and integration of the applet with a friendly user interface. This means that the applet can be controlled from outside by the java script API.

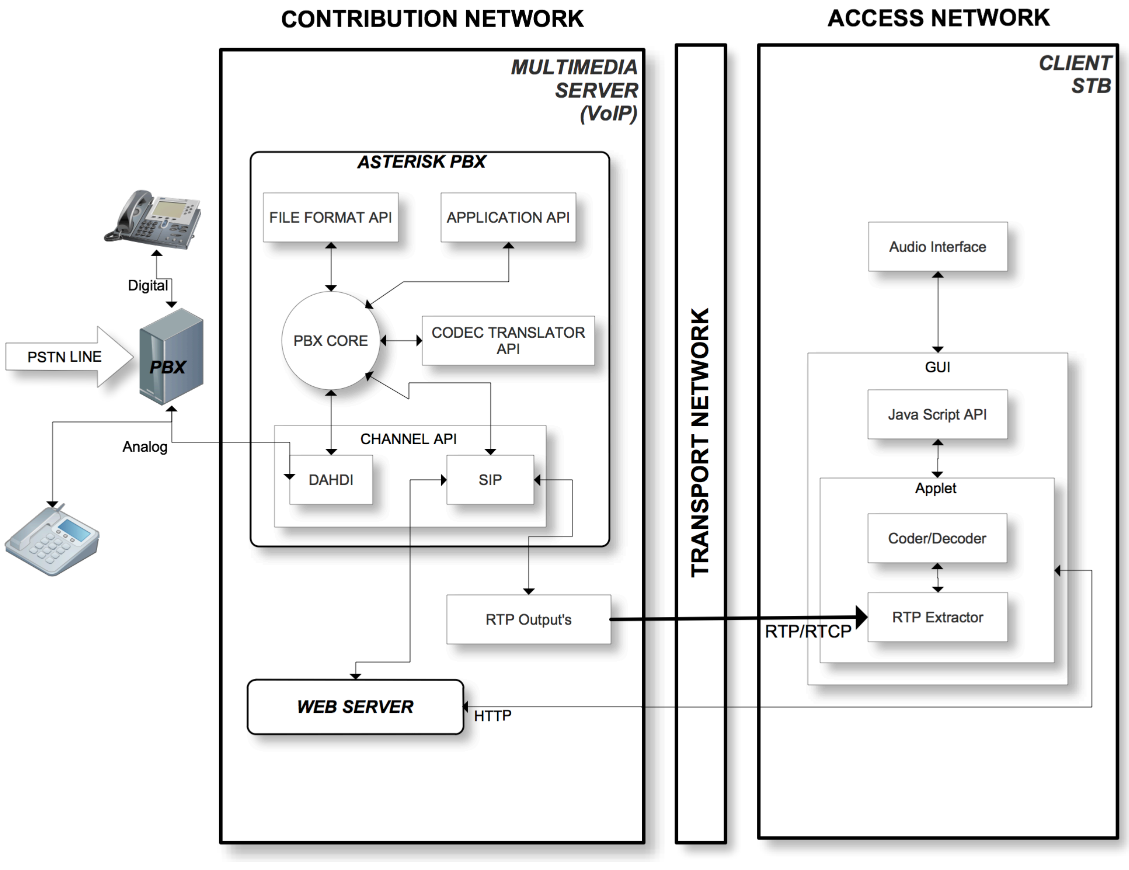


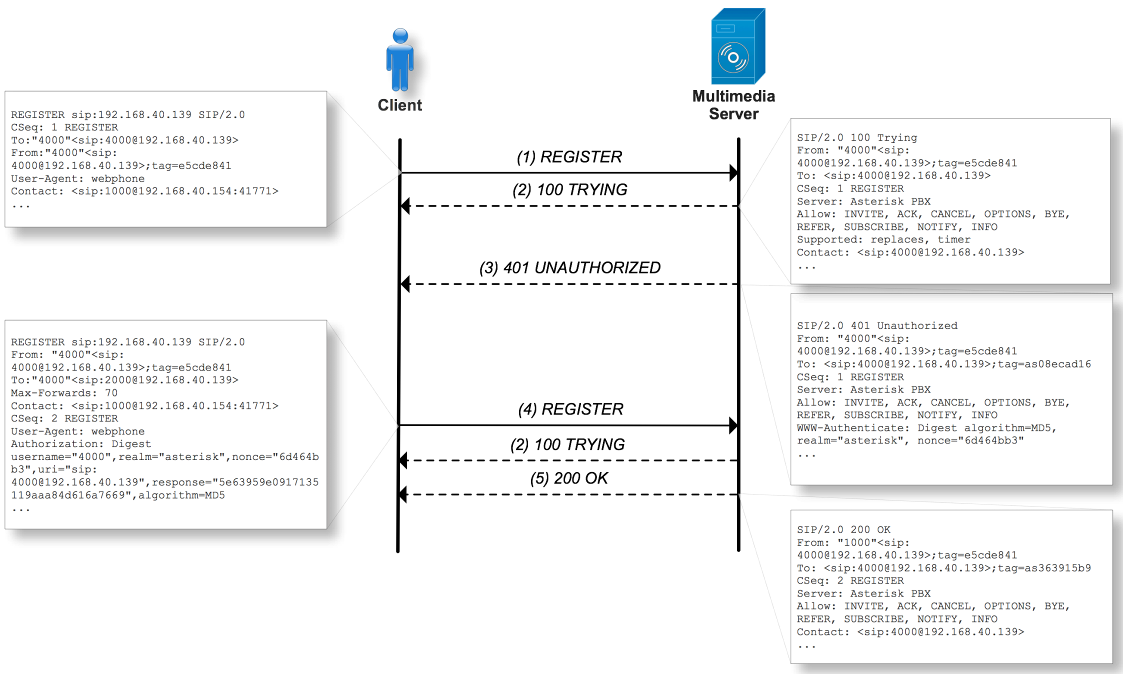
Figure 3.7: VOIP Module (based on [13]).

Communication model

As seen in section 2.1, the SIP exchange process consists of message flows between the client and the server. This message flow can be split into three main stages: the registration process, the session establishment and the session closing. The following subsections describe the message transactions between the server and the client in all the three stages.

- Registration process

  The registration process starts when the client sends a register request to inform the server about its current address. With this message, the client also informs that all the incoming requests should be redirected to its address. Instead of receiving a 200 OK message, the client receives an authentication message for security issues. The client answers with a new request containing the authentication credentials. These two requests are a security mechanism that the server uses to check the true identity of the caller. The process of registering in the server is showed in figure 3.8.



```
REGISTER sip:192.168.40.139 SIP/2.0
CSeq: 1 REGISTER
To:"4000"<sip:4000@192.168.40.139>
From:"4000"<sip:
4000@192.168.40.139>;tag=e5cde841
User-Agent: webphone
Contact: <sip:1000@192.168.40.154:41771>
...
```

```
SIP/2.0 100 Trying
From: "4000"<sip:
4000@192.168.40.139>;tag=e5cde841
To: <sip:4000@192.168.40.139>
CSeq: 1 REGISTER
Server: Asterisk PBX
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE,
REFER, SUBSCRIBE, NOTIFY, INFO
Supported: replaces, timer
Contact: <sip:4000@192.168.40.139>
...
```

```
SIP/2.0 401 Unauthorized
From: "4000"<sip:
4000@192.168.40.139>;tag=e5cde841
To: <sip:4000@192.168.40.139>;tag=as08ecad16
CSeq: 1 REGISTER
Server: Asterisk PBX
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE,
REFER, SUBSCRIBE, NOTIFY, INFO
WWW-Authenticate: Digest algorithm=MD5,
realm="asterisk", nonce="6d464bb3"
...
```

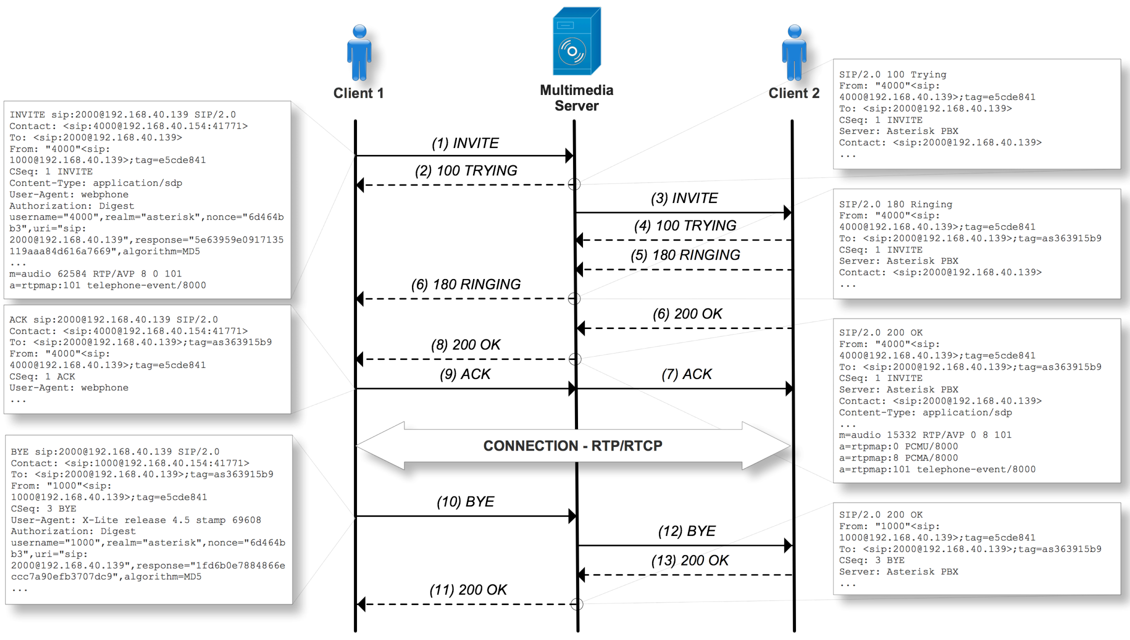
```
REGISTER sip:192.168.40.139 SIP/2.0
From: "4000"<sip:
4000@192.168.40.139>;tag=e5cde841
To:"4000"<sip:2000@192.168.40.139>
Max-Forwards: 70
Contact: <sip:1000@192.168.40.154:41771>
CSeq: 2 REGISTER
User-Agent: webphone
Authorization: Digest
username="4000",realm="asterisk",nonce="6d464b
b3",uri="sip:
4000@192.168.40.139",response="5e63959e0917135
119aaa84d616a7669",algorithm=MD5
...
```

```
SIP/2.0 200 OK
From: "1000"<sip:
4000@192.168.40.139>;tag=e5cde841
To: <sip:4000@192.168.40.139>;tag=as363915b9
CSeq: 2 REGISTER
Server: Asterisk PBX
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE,
REFER, SUBSCRIBE, NOTIFY, INFO
Contact: <sip:4000@192.168.40.139>
...
```

Figure 3.8: SIP register message flow client-server.

- Session establishment

  After finishing the registration process, the client is able to call another client (that is registered in the server), by sending an invite request through the server. The process of session establishment starts with the invite request (as can be seen in figure 3.9). The server forwards the invite message to the other client (called client). The called client receives the invite request and alerts its user to answer the call. The called client replies with a 180 ringing message. This message is routed back along the reverse path to the caller client. After the called user answers the call, it sends a 200 OK message to the caller client which contains the Session Description Protocol (SDP) message. After the caller client receives the 200 OK message, both clients have learned each other's endpoint IP address and media description. So, the caller client sends an ACK message to the called client to acknowledge the session is built. Therefore the media session can be started between both SIP clients.



```
INVITE sip:2000@192.168.40.139 SIP/2.0
Contact: <sip:4000@192.168.40.154:41771>
To: <sip:2000@192.168.40.139>
From: "4000"<sip:
1000@192.168.40.139>;tag=e5cde841
CSeq: 1 INVITE
Content-Type: application/sdp
User-Agent: webphone
Authorization: Digest
username="4000",realm="asterisk",nonce="6d464b
b3",uri="sip:
2000@192.168.40.139",response="5e63959e0917135
119aaa84d616a7669",algorithm=MD5
...
m=audio 62584 RTP/AVP 8 0 101
a=rtpmap:101 telephone-event/8000
```

```
ACK sip:2000@192.168.40.139 SIP/2.0
Contact: <sip:4000@192.168.40.154:41771>
To: <sip:2000@192.168.40.139>;tag=as363915b9
From: "4000"<sip:
4000@192.168.40.139>;tag=e5cde841
CSeq: 1 ACK
User-Agent: webphone
...
```

```
BYE sip:2000@192.168.40.139 SIP/2.0
Contact: <sip:1000@192.168.40.154:41771>
To: <sip:2000@192.168.40.139>;tag=as363915b9
From: "1000"<sip:
1000@192.168.40.139>;tag=e5cde841
CSeq: 3 BYE
User-Agent: X-Lite release 4.5 stamp 69608
Authorization: Digest
username="1000",realm="asterisk",nonce="6d464b
b3",uri="sip:
2000@192.168.40.139",response="1fd6b0e7884866e
ccc7a90efb3707dc9",algorithm=MD5
...
```

```
SIP/2.0 100 Trying
From: "4000"<sip:
4000@192.168.40.139>;tag=e5cde841
To: <sip:2000@192.168.40.139>
CSeq: 1 INVITE
Server: Asterisk PBX
Contact: <sip:2000@192.168.40.139>
...
```

```
SIP/2.0 180 Ringing
From: "4000"<sip:
4000@192.168.40.139>;tag=e5cde841
To: <sip:2000@192.168.40.139>;tag=as363915b9
CSeq: 1 INVITE
Server: Asterisk PBX
Contact: <sip:2000@192.168.40.139>
...
```

```
SIP/2.0 200 OK
From: "4000"<sip:
4000@192.168.40.139>;tag=e5cde841
To: <sip:2000@192.168.40.139>;tag=as363915b9
CSeq: 1 INVITE
Server: Asterisk PBX
Contact: <sip:2000@192.168.40.139>
Content-Type: application/sdp
...
m=audio 15332 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000
```

```
SIP/2.0 200 OK
From: "1000"<sip:
1000@192.168.40.139>;tag=e5cde841
To: <sip:2000@192.168.40.139>;tag=as363915b9
CSeq: 3 BYE
Server: Asterisk PBX
...
```

Figure 3.9: SIP session establishment message flow client-server.

- Session closing

  The process of finishing the call is showed in the figure 3.9. It is important to notice that the session is closed whenever one of the two intervenients leaves it. The session can be closed using the BYE message.

Module requirements

The VoIP module presented above introduces the following requirements:

- The clients should be able to start and receive calls for both SIP and Public Switched Telephone Networks (PSTN) users.
- The clients, the server and the network should be able to process both SIP and RTP/RTCP protocols.
- The client interface should be able to support human interaction (e.g. dial a phone number, end a call).
- The VoIP service must work in a WLAN with acceptable QoS.

Open source software for VoIP servers

- SIP Express Router (SER)

  The SIP Express Router (SER) [6] is a high performance free SIP server licensed under GNU General Public License (GPL) open source. It can act as a SIP server, register server, proxy server or redirect server. SER can be configured for specialized tasks such as load balancing or SIP front end for application servers. SER supports the main database management systems (MySQL and Oracle) and also offers complete support of Recommendation 3261 functionality (SIP) [103]. The SER works under numerous operating systems and their distributions: Debian, FreeBSD, Gentoo, NetBSD, OpenBSD, OpenSuse, Solaris.

- Kamailio

  Kamailio [30] is an open source SIP server released under GNU GPL. Kamailio can work as a register server, location server, proxy server, SIP application server or redirect server. It supports most of the transport layers like UDP, Transmission Control Protocol (TCP) and Stream Control Transmission Protocol (SCTP). Kamailio also supports the main database (MySQL, Oracle, PostgreSQL). It can be used to build large VoIP servicing platforms or to scale up SIP to PSTN gateways or PBX systems. Kamailio focus on performance, flexibility and security.

- OpenSIPS

  OpenSIPS [33] is one of the fastest open source SIP servers. The focus of OpenSIPS is to offer reliability and high-performance. OpenSIPS is a SIP proxy/router that includes an application level functionality. OpenSIPS can also be configured for specialized tasks such as load balancing.

- FreeSWITCH

  FreeSWITCH [26] is a free open source communication platform written in C for the creation of voice and messaging systems. FreeSWITCH is licensed under Mozilla Public Licensing (MPL). It includes many types of modules which provide applications by default, including conferencing, Iteractive Voice Response (IVR), PSTN interconnection, with the ability for supporting both analogue and digital circuits.

FreeSWITCH supports various types of VoIP protocols as SIP and H.323.

- Asterisk

  Asterisk [20] is an open source software implementation of a telephone PBX. Asterisk is an integrated telecommunication platform, distributed under GPL as free software. Asterisk powers IP PBX, VoIP gateways, conference servers and other custom solutions. One of the great advantages of Asterisk is allowing real time communications, applications and solutions. Asterisk offers both classical PBX functionality and advanced features, and interoperates with traditional standards-based telephony systems and VoIP systems. Asterisk supports almost all types of VoIP protocols such as SIP, H.323, Skinny Call Control Protocol (SCCP) and Media Gateway Control Protocol (MGCP). The impact of this software is the ability to turn a simple low cost computer running Linux into a unified communication server.

Asterisk was chosen for the developing of the VoIP module. Any other open source software mentioned before would be a possible solution. However for the intended purpose Asterisk was the most complete in terms of available functions. As SIP server OpenSIPS or FreeSWITCH are superior as far as the server is concerned, however in terms of available functions Asterisk is better. As mentioned before, Asterisk supports various protocols (H.323, MGCP, SCCP and SIP) and since it is a modular architecture, it also supports lots of additional PBX services. Later on, the Asterisk software will be discuss in a more detail.

4

Implementation

4.1 Introduction

This section describes the implementation of the client and server prototypes, taking into account the architecture presented in the previous chapter. It describes the methodology for software development, the libraries used, the programming language and the mapping of these tools into the client and server modules. The chapter ends with a brief description of the system working process and about some limitations of the implementation.

4.2 Development stages

The developing stages of the implementation process is represented in figure 4.1. It tarted with the planning stage, where the requirements and objectives of the implementation were defined. In the research stage, several streaming platforms and Internet Protocol (IP) Private Branch Exchange (PBX) servers were tested, in order to know which one would be the best solution that satisfy the requirements. Several approaches to client implementations, both in hardware and in software were studied, as also the encoding and decoding video tools. Based on this study it was possible to carry out an assessment of the capabilities and limitations related with the requirements. After an in-depth work in planning and researching the tools, we chose the platforms and libraries to use in the development solution. In the third stage the architecture for the solution was defined. The architecture is based on three main modules, Internet Protocol Television (IPTV), Video on Demand (VoD) and Voice over Internet Protocol (VoIP), as described in the previous chapter.

The development phase began with the creation of the streaming server modules, the IPTV and VoD server modules. Alongside with these modules, the IPTV and VoD clients were also developed. In a second phase of the development stage, the VoIP server and client were developed. All these modules were integrated into the architecture of the solution described before. The first full version of the prototypes server and client was completed when all the components were integrated with the hardware solution.

The final stage was the creation of the testing scenario, in order to achieve a complete evaluation of the final solution. This last stage will be fully detailed further in Chapter 5.

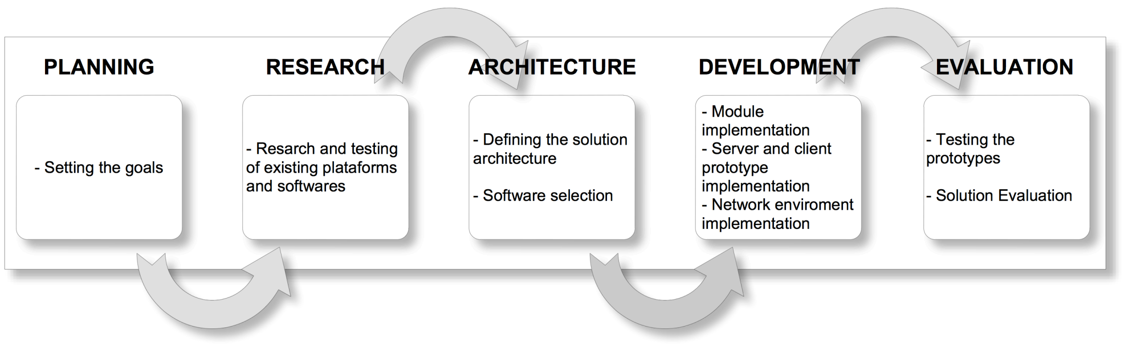

PLANNING
- Setting the goals
RESEARCH
- Resarch and testing of existing plataforms and softwares
ARCHITECTURE
- Defining the solution architecture
- Software selection
DEVELOPMENT
- Module implementation
- Server and client prototype implementation
- Network enviroment implementation
EVALUATION
- Testing the prototypes
- Solution Evaluation

Figure 4.1: Developing stages of the implementation process.

4.3 Solution requirements

The prototypes of the triple-play client and server developed in this work took into account the following requirements for the deployment of the system services:

- Open source software

  The server should deploy all the three main modules of the triple-play system (IPTV, VoD, VoIP) only using open source software.

- Wireless network environment

  The client should be able to access the modules from different network interfaces, however the main focus in this case is the Wireless Local Area Network (WLAN) access.

- Web Based technology

  The client should be able to run using web based technology. This means that the client has to run everything inside a browser with all the limitations and resources that this type of technology presents.

4.4 Client

Requirements

The requirements define the basic functions of the client (Set-Top Box (STB)). This section lists the requirements identified for the client.

- The client must support the reception of IPTV (multicast) and VoD (unicast) streams and must be able to decode and display the streams.
- To play the multimedia stream a Real-Time Transport Protocol (RTP)/Real-Time Streaming Protocol (RTSP) connection must be used in the user interface.
- The client must support start and receive calls for Session Initiation Protocol (SIP) and Public Switched Telephone Networks (PSTN) users.
- Protocols such as RTP/Real-Time Control Protocol (RTCP) and SIP has to be supported by the user interface to receive calls.
- The client can use different access technologies, therefore in this work we focus the WLAN connection.
- The client must share internet with other clients (phones, laptops) via WLAN.

- The client must support user interaction functions such as changing the Television (TV) channel, volume up, volume down, mute, pause, dial phone number and start/end call via the remote control.
- The STB Operation System (OS) has to be fully locked to avoid system security breaks.
- All video content available must be displayed in the user interface.
- The user interface shall use a web-based interface.

Set-top box prototype

The STB prototype was one of the most important stages in the development section. The STB prototype has been developed in order to allow users to see multimedia contents with a restricted user interface. Thus, a fully customized OS was necessary to protect privacy and maximize security, in order prevent users of changing any configuration settings. This system is based on a browser-only OS, specifically tuned to show multimedia in a full-screen mode with a restricted navigation. The STB OS is based on Ubuntu 12.04 LTS (Precise Pangolin). It has all unnecessary utilities of Ubuntu such as software and mouse/keyboard controls disabled and has new features such as remote control navigation. The Ubuntu 12.04 uses as default Graphical User Interface (GUI), the GNU Network Object Model Environment (GNOME). Thus, the first step of the customization was disabling the navigation features, and the definition of a specific permission to the user. A restricted user is a user with specific permissions that does not have master (sudo) access and can not change system settings. This is an important measure not to compromise the security of the STB. To achieve this level of customization it was necessary to create and edit some Ubuntu OS start-up scripts. The main start-up scripts created were:

- lockdown: This script is responsible for locking the navigation and interaction with the user. An important step of customization was the processing of disabling the following features: the gnome-panels, the seven Teletype write (TTY) consoles and the keyboard/mouse navigation and shortcuts.

- start-up: This script is used to specify what software and libraries should run in the initialization process of the OS. The main start-up software and libraries are the starting-up browser with the initialization of the specific add-ons, the starting-up of the wireless and ethernet configurations, the starting-up of the remote administration access and finally the starting-up of the remote control library. These topics will be discussed further in this chapter.

As mentioned before, since this is a browser-only OS, the chosen browser was Mozilla Firefox version 16.02. During the development of this project this was the only one which would support the project features. The browser was also customized with some external extensions to achieve the desired level of security and restriction. These extensions were used to disable all menus from the browser, to customize shortcuts, to maximize the browser window to full screen and to initialise the player to VoD and VoIP contents. This subject will be further discussed.
An important question after all the features and system configurations done was how to access the STB. To solve this, a Secure Shell (SSH) channel was initialized in the start-up scripts for remote administration purposes. This is the only method available to externally access the STB.

Physical components

To implement the STB prototype the platform chosen was a nanoPC computer from Foxconn (nT-535) [60]. This computer is composed by an Intel Atom D525 1.8 GHz processor, with 2 GB of memory and a SSD hard drive. The SSD was used to get a better OS performance and a faster start-up process. This STB also has a Realteck 8191SE wireless interface for WLAN communication purposes. The STB also contains two additional external hardware interfaces: the wireless adapter and the Infrared (IR) receiver/remote control. These external hardware interfaces have different functions in the overall project.
The wireless adapter is used to add another wireless interface to the STB (since the STB already has one internal wireless interface). This subject will be discuss further in this chapter.
The IR receiver and the remote control are used to give the end-user the possibility of a

friendly interaction with the STB. These types of hardware are a standard Media Center Edition (MCE) receiver and controller. The library that allows the STB OS to decode the IR signals coming from IR receiver is Linux Infrared Remote Control (LIRC) [14]. In order to use LIRC some configurations in the OS were needed. First it is necessary to create a configuration file to identify the hardware and the port connection where the IR receiver is connected to. Under the LIRC folder */etc/lirc* a file *hadrware.conf* is created, which has the information about the IR receiver and remote control. The following code presents the configuration used:

```
REMOTE="Windows Media Center Transceivers/Remotes (all)"
REMOTE_MODULES-"lirc_dev_mceusb"
REMOTE_DRIVER=""
REMOTE_DEVICE="/dev/lirc0"
REMOTE_LIRCD_CONF="mceusb/lircd.conf.mceusb"
REMOTE_LIRCD_ARGS=""

TRANSMITTER="None"
TRANSMITTER_MODULES=""
TRANSMITTER_DRIVER=""
TRANSMITTER_DEVICE=""
TRANSMITTER_LIRCD_CONF=""
TRANSMITTER_LIRCD_ARGS=""

START_LIRCD="true"
START_LIRCMD="false"
LOAD_MODULES="true"
LIRCMD_CONF=""
FORCE_NONINTERACTIVE_RECONFIGURATION="false"
```

After the identification of the IR receiver and remote control, it was necessary to configure the remote keys. It is one of the most important part of the process, since this enables the STB to identify the signals that are being send by the remote control (via the IR receiver).

The file responsible for this configuration is called *lircd.conf.mceusb* and is stored under *usr/share/lirc/remotes/mecusb*. Part of this file can be seen below:

```
begin remote
  name        mceusb
  bits                 16
  flags  RC6|CONST_LENGTH
  ...
  begin codes
        KEY_OK         0x00007bdd
        KEY_RIGHT      0x00007bde
        KEY_LEFT       0x00007bdf
        KEY_DOWN       0x00007be0
        KEY_UP         0x00007be1
        KEY_0          0x00007bff
        KEY_1          0x00007bfe
        KEY_2          0x00007bfd
        KEY_3          0x00007bfc
        KEY_4          0x00007bfb
   ...
  end codes
end remote
```

The code shows the decoded key codes that are known by LIRC. The more complete list of codes can be found in Appendix C. Setting up the remote control is only half of the equation. The other half is getting LIRC to communicate with the software that needs to be controlled. In this case, since the project is based on a browser and a web application for the client, LIRC has to communicate with Mozilla Firefox. So it was necessary to map the remote control keys to the actions to be performed in the application. The key mapping process is showed below and the full version is showed in appendix C.

```
begin
       prog = irxevent
       button = KEY_Down
       config =  Key Down CurrentWindow
end
begin
       prog = irxevent
       button = KEY_OK
       config =  Key Return CurrentWindow
end
begin
       prog = irxevent
       button = KEY_1
       config =  Key 1 CurrentWindow
end
```

Basically with this code, the buttons in the remote control are being mapped to specify keyboard commands into Mozilla Firefox. Another function implemented with LIRC was the execution of terminal commands, as showed next.

```
begin
    prog = irexec
    button = KEY_POWER
    config =  xset dpms force off; xset dpms force on
end
```

In this case the *xset dpms force off*, will force the screen to turn off and the *xset dpms force on* will force to turn on. This is a terminal execution, instead of an application execution, and that is why *irexec* is used instead of the *irxevent*.

User interface

The user interface is the software layer that provides the control of the STB. It is the final link between the hardware prototype and the user. Without this integrated user interface,

to show how a user application would work or how it should look like, everything done so far would be rather useless. Some advantages of using a web-based user interface are that it can be moved to another STB without any problems of configuration, and the fact that it can be updated on the web server, instead of updating all the users' application in each STB. The user interface uses HyperText Markup Language (HTML) to provide an interface that can be used in any browser. To start the development of the user interface there are several important components: The PHP Hypertext Preprocessor (PHP) framework, the HTML, the Cascading Style Sheets (CSS), the JavaScript, the JQuery, the VLC plugin, the Java applet and the IMDB scraper Application Programming Interface (API). An overview of the user interface can be seen in figure 4.2. The PHP framework was used to work with the Data Base (DB) where the list of channels and available multimedia contents are stored. The HTML, CSS, JavaScript and JQuery tools were used to provide an appealing web interface. The VLC plugin was used to receive and play the streams and the Java applet was used to act as a SIP client.
To retrieve all the movie's information another PHP framework was used to scrap IMDB information about the desired movie. The chosen tool was IMDB Scraper API since this library is always up to date with the IMDB official website [2].

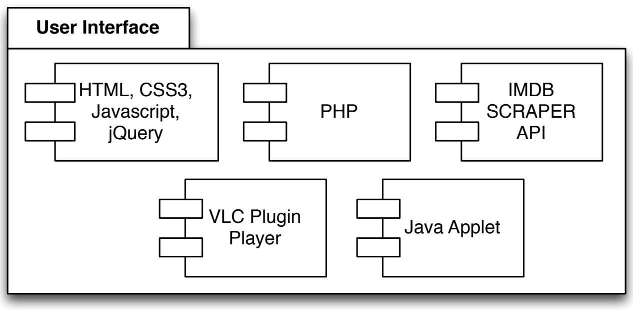


Figure 4.2: User interface components.

User interface design

Nowadays, powerful enterprise web pages are designed with the help of several HTML tools, for example, HTML 5, CSS 3, JavaScript, PHP, JQuery. All these tools provide different options for building web interfaces. To keep things simple, the tools used to design the user web interface were HTML, CSS, JavaScript and JQuery. PHP was used to process all the required connections with the DB, in order to retrieve the information about the channels and video movies.

The user interface is composed by three modules: The television contents (IPTV), the movie contents (VoD) and the telephone interface (VoIP). Appendix C.2 presents the full layout of the user interface, which is summarized in this section. All this modules are presented in one single screen, displaying each one according to the user's inputs. When the user selects IPTV, the television contents start playing. In this screen the user can change between channels, choose a specific channel by selecting the corresponding number and change the sound volume. When the user selects VoD the whole movie list is retrieved from the DB. This list is displayed in a new scene using their thumbnails. By choosing a thumbnail the movie picture will zoom in for a proper view of the movie content and information (such as rating and actors). A play option can be selected to start watching the movie. After the movie starts the user can interact with the STB using functions such as pause, play and change the sound volume. Finally, when the VoIP option is selected the telephone interface is showed. After this the user can star/receive calls using the remove control to start/stop the calls and dial the phone numbers.

IPTV and VOD

The main purpose of these modules is to carry IPTV and VoD to the final user by a web interface and in a transparent way. The IPTV and VoD modules work in a similar way as far as the video receiving process is concerned. As mentioned before, the VLC plugin version 2.0.3 was used to receive the video contents (IPTV and VoD) from the server inside the Mozilla Firefox browser.

Figure 4.3 shows an overall view of the user interface from the STB point of view. The Mozilla Firefox web browser receives the user interface (3) from the web server. The user interface uses the user-supplied Java Script (1) to identify the VLC plugin. The web browser includes the plugin interface that uses a generic browser plugin API (2) to interact with a browser specific plugin API (Netscape Plugin Application Programming Interface (NPAPI)), which will interact with the browser core, creating an interface between the generic browser plugin API and the browser.

One of the several advantages of using the VLC plugin is that it can easily be embedded

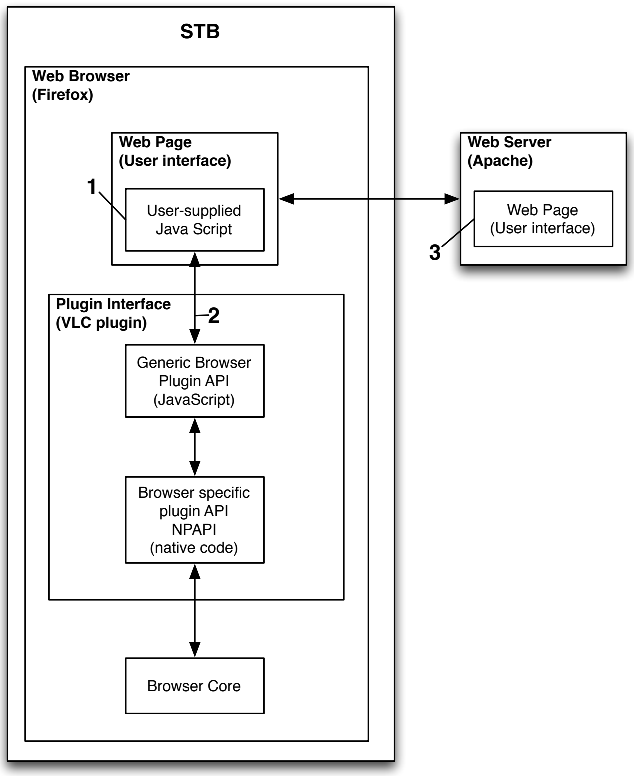

STB
Web Browser (Firefox)
Web Page (User interface)
1
User-supplied Java Script
Plugin Interface (VLC plugin)
2
Generic Browser Plugin API (JavaScript)
Browser specific plugin API NPAPI (native code)
Browser Core
Web Server (Apache)
Web Page (User interface)
3

Figure 4.3: IPTV and VoD user interface block architecture.

into a web page. More information about the process of embed VLC plugin into an HTML page can be seen in appendix C.3. Another important feature offered by the VLC plugin is the direct interaction with the VLC object embedded in the web page. The code below shows two different interactions with the VLC object: the first function toggles between mute and sound of the VLC object and the second one sends a play command to the VLC object.

```
function mute(){
        vlc.audio.toggleMute();
}
function play(){
        vlc.playlist.play();
}
```

Java Script is used to control the VLC object inside the web browser and implement all the necessary controls on the video window.

Finally another Java Script was implemented to remote control the VLC object. This Java Script is called *navigation.js* and is responsible for the capture of keys pressed by the remote control and process the corresponding functions inside the VLC object (part of this code can be seen in appendix C.4).

VoIP

The VoIP client (introduced in 3.4) works using a Java applet, which is platform independent and runs on any Java enabled web browser. The VoIP client is customizable via Java Script and uses the SIP protocol stack for communication purposes. The applet is responsible for all the SIP communication process and the Java Script implements both the functions to control the applet, and the user interface. This interface can be seen in appendix C.2, figure C.6.

The figure 4.4 shows the overall process of the Java applet in the STB. The Mozilla Firefox web browser receives the user interface from the web server (2). The user interface uses the user-supplied Java Script (1) to identify the Java applet (4). In order to run this applet, the web browser uses a Java plugin that allows the applet file (3) to be downloaded from the web server.

The user supplied Java Script acts as a controller block since it orders the applet to run the specific SIP commands.

The following code shows how the applet is embedded into the user interface (1). The archive parameter identifies the name of the applet file stored in the web server. The Java Script was used to create an user friendly phone application and the applet was minimized to one pixel. The *serveraddress* variable is used to identify the VoIP server address.

```
<applet
      archive = "webphone.jar"
      codebase = "."
      code = "webphone.webphone.class"
      name = "webphone"
      width = "1"
```

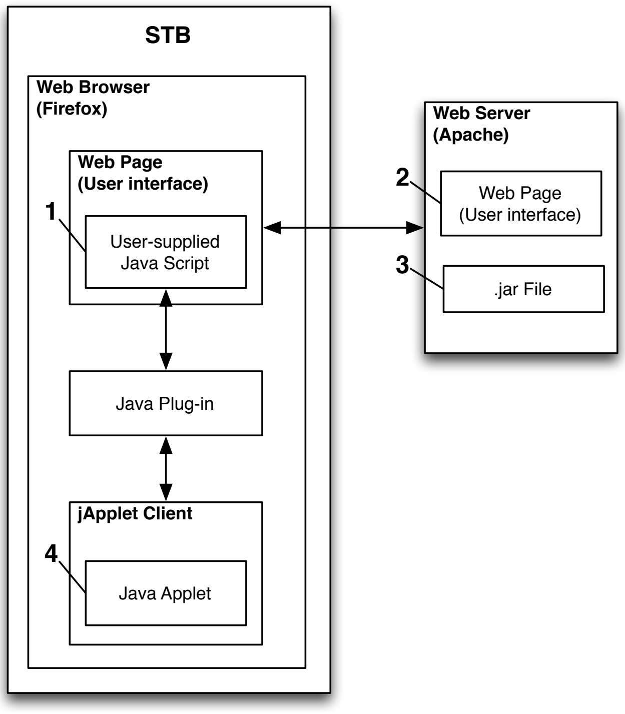


Figure 4.4: VoIP user interface block architecture.

```
        height = "1"
        hspace = "0"
        vspace = "0"
        align = "middle"
    >
    <param name = "serveraddress" value = "192.168.40.139:5060">
    </applet>
```

An example of functions used in Java Script to connect with the applet is *API_Register*, showed below.

```
boolean API_Register(String server, String username,
     String password, String authname, String displayname)
```

Function *API_Register* is used to register the SIP client into the VoIP server. This function only needs to be called once, since the subsequent re-registrations are done automatically. When called subsequently, the old registration is deleted and a new one will be created with a new call-id. For the registration process, the VoIP server address and the username/password are necessary. Another example is the *API_Call* function, that is used to initiate a call to a number or SIP username and is showed below.

```
boolean API_Call(int line,String peer)
```

Other services

As mentioned before, the client (STB) is composed by two wireless interfaces: one internal and one external. The external interface is a wireless adapter connected to the STB, that is used for communication with the multimedia server. The internal interface is used to share internet with other clients such as laptops and mobile phones. Figure 3.1 from the previous chapter can help understanding this concept of internet sharing. To accomplish this, the internal interface needs to be configured to work as a hotspot, while the external interface is used for communication with the multimedia server. This process was implemented, however due to limited development time, this feature was not properly tested.

4.5 Multimedia Server

The tools chosen for the multimedia server, and some of the implementations that were needed in order to achieve the solution requirements are presented in this section.

Sections 4.4 and 4.4 showed the specific components that make up the client's user interface and physical hardware components. However the client is only one part of the project. The other part is the multimedia server, responsible for delivering all the contents to the clients. This section is split into three main subsections, each one explains how IPTV, VoD and VoIP contents are received, processed and delivered to the client.

An overview of the multimedia server components can be seen in figure 4.5. The multimedia server is running under Linux Ubuntu 10.04 LTS (Lucid Lynx) and its main components are the IPTV, VoD and VoIP modules, the Apache server and the Java Standard Edition (SE) API. Each one of this components is responsible for one task in the whole system. The IPTV and VoD modules are responsible for delivering the streaming contents. The selected tools for these modules were DVBlast tool (project developed by VideoLAN community) [23] for the IPTV and VLC together with VideoLAN Manager (VLM) for content management for the VoD. The VoIP module is responsible for the calling process between two users and the server. In this case the chosen tool was Asterisk PBX. The Java SE API makes the execution of the server application interface possible to both IPTV and VoD modules. This interface is used to start/stop and manipulate the streams of these modules. Finally the Apache server is a web server application, used to supply the user interface to the clients STB.

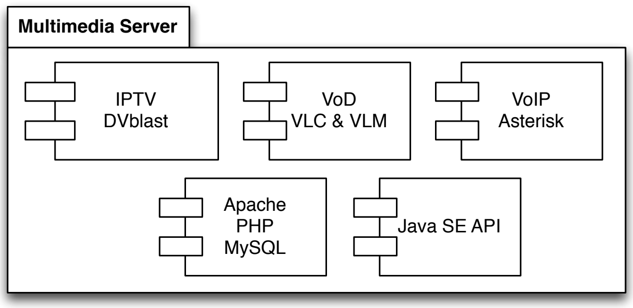


Figure 4.5: Multimedia server components.

IPTV

One of the implementation goals in this module is the ability to receive and stream Digital Video Broadcasting - Terrestrial (DVB-T). To allow the system to receive DVB-T signals a Terratec Cinergy T Stick Dual RC Universal Serial Bus (USB) adapter [9] was used. The DVB-T USB adapter was shipped with a small antenna, however reception is poor with this kind of small antennas. An external antenna was used instead to solve this issue in the implementation process .

The adapter's main components are the ITE Tech IT9137 and ITE Tech IT9133 [28]. However in this work only one chip was used (ITE Tech IT9137, the main chip in the DVB-T receiver) because only one receiver was needed. This chip is three-in-one and it contains: a Radio Frequency (RF) tuner, a Coded Orthogonal Frequency Division Multiplexing (COFDM) demodulator and a USB interface. To use a USB DVB-T adapter in Linux, the Digital Video Broadcasting (DVB)-stack must be included in the Linux kernel. For

this, reconfiguration of the kernel is needed as also the compilation of the Linux kernel drivers for the specific DVB-T USB adapter.
When the DVB-stack is enabled in the kernel, a software is needed to interface with the Linux kernel DVB API. The DVB API is used for operations such as tuning and scanning channels. The DVB API and the DVB-stack are developed, maintained and documented by the LinuxTV project [31]. The DVB-stack name is different for each of the Linux distributions but in this case, since the Linux distribution that is being used is Ubuntu, the name is dvb-utils [22].
The IPTV is splitted into three main procedures to accomplish the delivery of the contents. They are the scanning procedure, the processing procedure and the streaming procedure.

Scanning

The first stage in the distribution of IPTV service is the scanning of the frequencies for available channels. To perform the scanning, a utility needs to be executed, which returns a file with the list of channels that are available in the specified region. The following code shows how this command should be executed and how the returned file with the list of channels looks like.

```
w_scan -c pt >> channelsPT.conf
```

The parameter *-c pt* specifies the country frequencies. This specification is necessary because the utility *w_scan* is not a frequency detector, since its function is limited to look for channels inside the region frequencies that are specified for the region. The list of scanned channels is stored in the file *channelsPT.conf*. An example of the file containing the list of channels is represented below:

```
RTP 1:754000000:INVERSION_AUTO:BANDWIDTH_8_MHZ:FEC_2_3:FEC_1_2:QAM_64:
TRANSMISSION_MODE_8K:GUARD_INVERVAL_1_4:HIERARCHY:256:257:1101

RTP 2:754000000:INVERSION_AUTO:BANDWIDTH_8_MHZ:FEC_2_3:FEC_1_2:QAM_64:
TRANSMISSION_MODE_8K:GUARD_INVERVAL_1_4:HIERARCHY:256:257:1102
```

```
SIC:754000000:INVERSION_AUTO:BANDWIDTH_8_MHZ:FEC_2_3:FEC_1_2:QAM_64:
TRANSMISSION_MODE_8K:GUARD_INVERVAL_1_4:HIERARCHY:256:257:1103

TVI:754000000:INVERSION_AUTO:BANDWIDTH_8_MHZ:FEC_2_3:FEC_1_2:QAM_64:
TRANSMISSION_MODE_8K:GUARD_INVERVAL_1_4:HIERARCHY:256:257:1104

HD:754000000:INVERSION_AUTO:BANDWIDTH_8_MHZ:FEC_2_3:FEC_1_2:QAM_64:
TRANSMISSION_MODE_8K:GUARD_INVERVAL_1_4:HIERARCHY:256:257:1111
```

The fields have the follow specification:
Channel name : Frequency (in Hz) : Inversion mode : Bandwidth (in MHz) : Forward Error Correction (FEC) scheme high priority stream : FEC scheme low priority stream : Modulation scheme : Transmission mode : Guard interval : Hierarchy information : Video Packet Identifier (PID) : Audio PID : PID

Processing

Once the data from the TV channels that are being broadcast are collected, the configuration files for the streaming software (DVBlast) has to be created. The channel configuration file is *channel.cfg* and can be stored anywhere in the server since this file will be later pointed to DVBlast. The following code exemplifies this configuration file.

```
;RTP 1
239.255.1.10:5004 1 1101
;RTP 2
239.255.1.20:5004 1 1102
;SIC
239.255.1.30:5004 1 1103
;TVI
239.255.1.40:5004 1 1104
```

The lines starting with a semicolon are comments, and are used only for internal identification. After the comment, the line presents three columns: the multicast IP address (and port) to stream the channel, the 'always-on' flag and the PID to identify the channel.

Streaming

The streaming process is the last stage of the delivery of the contents to the clients.

- DVBlast

  In the previous two sections (scanning and processing) the goals were to create/-configure all the configuration needs in order to successfully stream the TV channels. In the streaming section the information collected in the previous sections is used and the streaming process is started. As mentioned before the application used for streaming is DVBlast. The DVBlast is a lightweight application designed to run under extreme memory and CPU conditions. As seen in section 3.4, the streaming module is responsible for editing the Program Specific Information (PSI) tables in order to rearrange the MPEG-2 Transport Stream (MPEG-2 TS) packets coming from the DVB-T USB adapter. To accomplish this process, DVBlast uses biTStream library [21], which is a set of C headers that allow a simpler access to the binary structures. In order to start DVBlast, a terminal application is needed (this is just an example since the process of lauching DVBast with resource to the Java SE API application will be later explained). A set of parameters has to be specified to start DVBlast. The following code shows how to start an individual instance of DVBlast.

  ```
  dvblast -a 0 -c /home/admserver/IPTV/channels.cfg -f 754000000
  -m qam_64 -b 8 -e
  ```

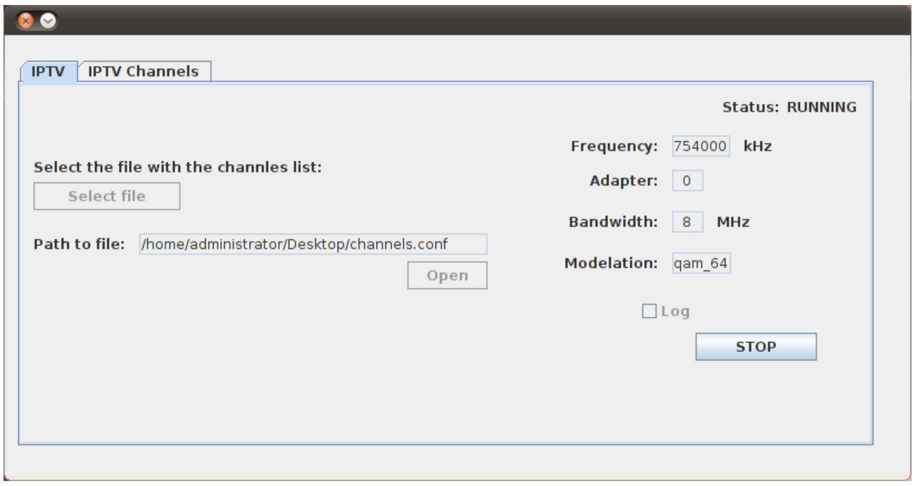
  Parameters:

  - ***-a n*** tells DVBlast to use tuner n. The number of the adapters starts in 0 according with dvb-utils.
  - ***-c nameoffile.cfg*** tells DVBlast to use the specific configuration file.
  - ***-f xxxxxxxxx*** specifies the frequency.

  - *-m qam_x* specifies the modulation type.
  - *-b n* specifies the bandwidth.
  - *-e* tells DVBlast to also stream Electronic Program Guide (EPG) data. In the development of this work the EPG is being transported, however it is not being showed to the user interface since this feature was not developed.

- Java SE API

  The Java SE API was used to build a Java application to replace the poor interface made available by DVBlast. This application is responsible for grabbing the configuration files and starting DVBlast. A preview of the application can be seen in figure 4.6. More examples of the application functionalities and layout can be seen in appendix C.5.



Status: RUNNING

Select the file with the channles list:

Path to file: /home/administrator/Desktop/channels.conf

Frequency: 754000 kHz

Adapter: 0

Bandwidth: 8 MHz

Modelation: qam_64

☐ Log

Figure 4.6: IPTV Java application.

In order to activate the IPTV server, the administrator starts selecting the configuration file with the channel list and inserts the country parameters like frequency bandwidth and modulation. The information collected by the application about the channels being streamed is then stored into a DB. The DB structure can be seen in figure 4.7.

Figure 4.8 shows the class diagram with the main classes and functions developed for the Java IPTV module application.

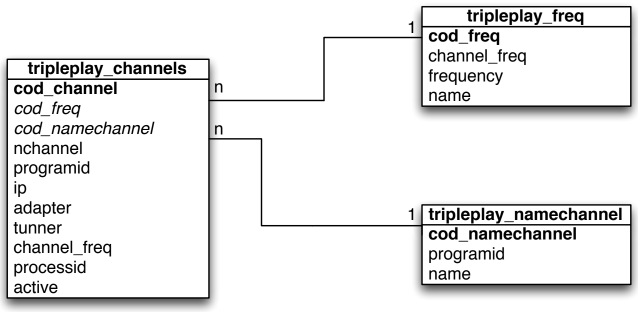


Figure 4.7: Design of the DB where the channels info are stored.

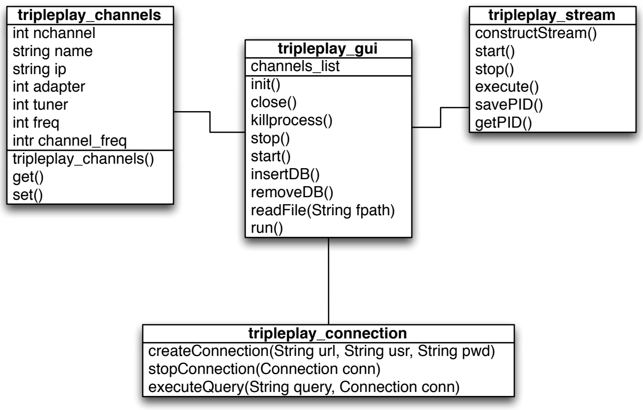


Figure 4.8: Class diagram of Java application.

Operation overview

To explain the "working mode" it is important to keep in mind that the main purpose of this module is to deliver TV channels to an end-user using a web browser.

Figure 4.9 shows the process of requesting a TV channel. The portal application is the customized Mozilla Firefox browser. It loads and displays the HTML page (user interface) from the Hypertext Transfer Protocol (HTTP) web server. This customized browser uses Java Script for the remote control interaction with the user. As mentioned before the stream transmission is done via multicast. The reason why video multicast is used is because each TV channel is a different multicast stream on a different IP address. The DVBlast is used for multicast streaming of the TV channels (the signal with the TV channels is received from the DVB-T adapter). The multicast streams address has the following syntax *rtp://@239.255.1.X:5004* and its IPs are stored in the DB by the Java application that interacts directly with the DVBlast.

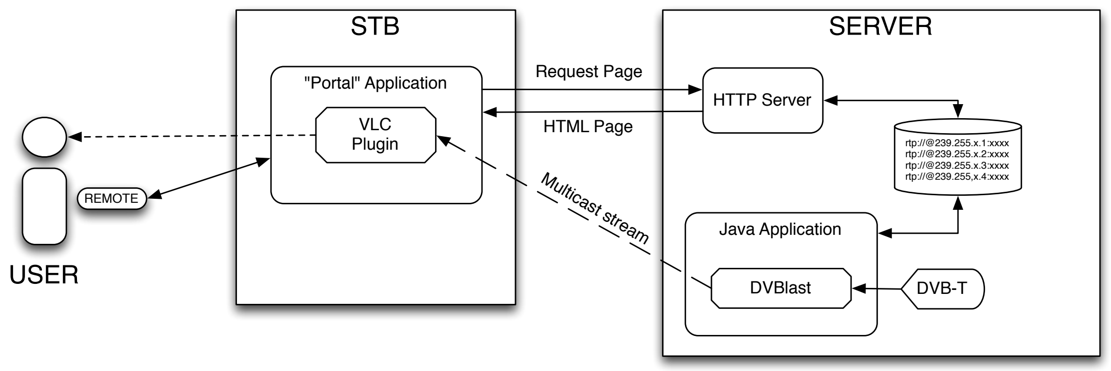


Figure 4.9: Conceptual diagram of the IPTV working process.

VoD

The objective of this module is to offer video contents to its clients. These video contents are generally stored in the hard disk, however they can be stored in any removable device, with the obvious bitrate limitation introduced by the interface which connects the removable device to the server. The VoD applications are extensive, since VoD can be used to share videos, TV shows or even commercial advertising.

VLC and VLM

The VoD module is responsible for delivering video contents to the clients, as described on sections 3.4 and 3.4. It includes the libVLC version 1.1.13, which uses the LiveMedia library from LIVE555 [32] to support RTSP, RTP and Session Description Protocol (SDP). VLC uses VLM as a small media manager to control multiple streams by an administration interface. VLM allows the creation of a single file with all the configuration parameters to initiate the stream, similarly to what happened in DVBlast.
An example of the configuration parameters for the VoD contents can be seen below:

```
new video1 vod enabled
setup video1 input "file:///home/administrator/Videos/ssredp.mkv"
new video2 vod enabled
setup video2 input "file:///home/administrator/Videos/arrow.mkv"
new video3 vod enabled
```

```
setup video3 input "file:///home/administrator/Videos/wdead.mkv"
new video4 vod enabled
setup video4 input "file:///home/administrator/Videos/borgia.mp4"
```

There are two types of administration interfaces to manage the contents: the HTTP interface and the Telnet interface. The major difference between then is that the HTTP interface is more user friendly than the Telnet, therefore HTTP was used in this work. The VLM interfaces (HTTP and Telnet) can be initiated using any of the command lines described below. Appendix C.6 presents a preview of both interfaces.

- **HTTP interface**

  ```
  vlc --vlm-conf /home/admserver/VOD/vod_config.vlm --ttl 12
  -vvv --color -I http --rtsp-host 0.0.0.0:5554
  ```

- **Telnet interface**

  ```
  vlc --vlm-conf /home/admserver/VOD/vod_config.vlm --ttl 12
  -vvv --color -I telnet --telnet-password videolan --rtsp-host
  0.0.0.0:5554
  ```

The parameter *–rtsp-host* defines the IP address of the server that the clients should use to receive the stream. In this case *0.0.0.0* is the address of the local host. The path to *vod_config.vlm* is passed to the command line, VLC parses it and executes the specified instructions. After parsing the file, VLC does not need to access it during the media session. VLC performs an initial tuning of all videos defined on the *vod_config.vlm* file to check if there is any anomaly and if everything is correct, then HTTP or the Telnet interface is started. Control commands such as play, pause and stop can be executed from any of these interfaces.
A Java application was developed to start the VLM and HTTP interface. The Java application has to grab the configuration file with the video content objects and start the VLM with the administration interface (HTTP interface). A preview of the application can be seen in appendix C.7. To use this application, the administrator needs to select the configuration file with the video content list, then the information about the video contents is collected and stored into the DB (figure 4.10).

tripleplay_video
cod_video
nvideo
idvideo
ip
processid
active

Figure 4.10: Design of the DB table where the video information is stored.

Operation overview

Figure 4.11 illustrates the process of requesting a video content from the web interface. The process starts with the user requesting a video content through the portal application. The process of streaming is similar to the previous module, however in this case the streams are sent via unicast. Each user requests a specific video content that is available in the list and then the server starts sending the content. The administration user can modify the video contents directly through the HTTP interface. This functionality is a great advantage since the server does not need to stop, while the video contents are being edited. The unicast streams have the following syntax *rtsp://192.168.X.X:5554* and its IPs are stored in the DB by the Java application that interacts directly with VLC.

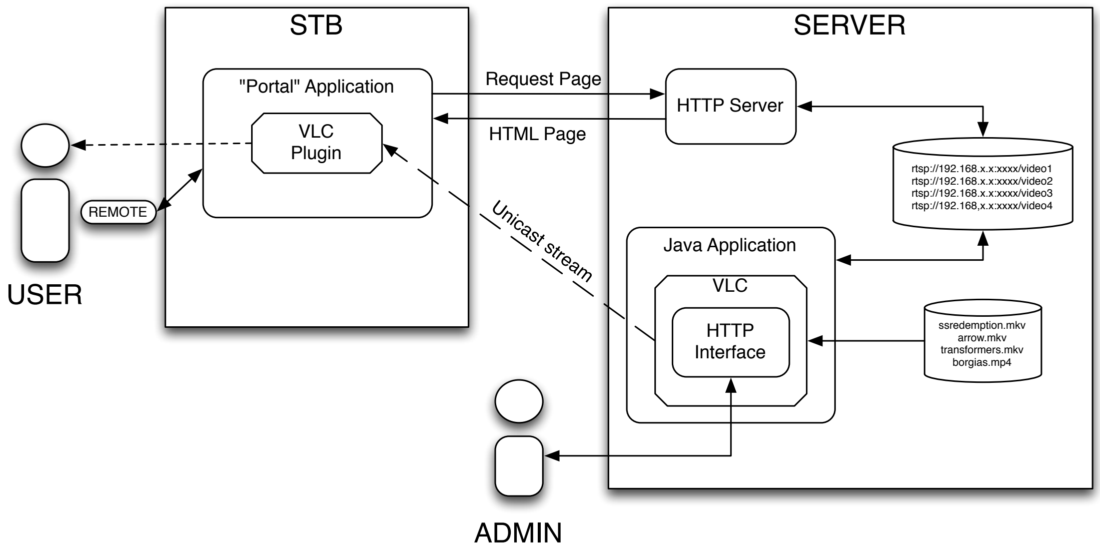


Figure 4.11: Conceptual diagram of the VOD working process.

VoIP

In the VoIP module the main objective is to start and receive calls between different users. Asterisk IP PBX was used to accomplish this objective, however extra hardware was needed in order to connect to a common telephone line. As seen in chapter 3.4, to use the ordinary analogue telephone line, a Foreign Exchange Office (FXO) module was needed. The chosen interface was the OpenVox A400P [8] with a FXO module, that is a modular analogue telephone interface. This interface contains four module banks. Each bank can be filled either with FXO or Foreign Exchange Station (FXS) modules. One of the key benefits of this interface is that it can be easily scaled to more users, adding additional of modules to the system. In order to make full use of A400P, a Digium Asterisk Hardware Driver Interface (DAHDI) software driver has to be compiled, installed and configured on Linux and Asterisk. This process was done according to the user manual from OpenVox [15]. One of the biggest obstacles in the development of the VoIP module was the integration of all the software and hardware under the Linux Ubuntu, because Asterisk is more commonly used under other Linux distributions (CentOS for example) and documentation for Asterisk in Linux Ubuntu is rare.

Asterisk and FreePBX

The Asterisk server version 1.8 was installed from the Ubuntu packages, using a customized configuration. The configuration process was done via the Asterisk command line interface. However, a friendly user interface, FreePBX [25], was also added to provide a more attractive GUI to Asterisk complex file system. FreePBX is one among many GUI that works together with Asterisk. In this work FreePBX was the chosen GUI since it is a very stable platform and is continuously being updated. Version 2.8 of FreePBX was installed to facilitate the administration of the full system. An example of the FreePBX interface is shown in appendix C.9.

- **Compiling, installing and loading**

  The process of installation both Asterisk and FreePBX is fully described in appendix C.8.

- **Configuration**

  The main files of the configuration process on Asterisk are explained in the next sections:

  Adding clients

  The SIP clients configuration in Asterisk system is done by the *sip.conf* file. This configuration file creates a SIP register for each user. The SIP file starts with a general section (denoted by [*general*]), which contains the general parameters for all users. The second section defines the specific SIP account parameters for each user.

  More information and an example of *sip.conf* can be found in appendix C.10.

  Telephone numbers allocation

  To establish a communication, there are some needed requirements such as, having registered users and defined rules to start or receive a call. These rules are defined on *extensions.conf* file and generally called dial plan. The configuration file *extensions.conf* contains the dial plans of Asterisk. This plan controls and executes all operations, such as incoming and outgoing calls. The content of *extensions.conf* is organized in sections which are referred as contexts. There are two setting sections denoted by [*general*] and [*globals*], which define the general settings for the *extensions.conf*. After these two sections, new contexts can be created. The names of these new contexts are entirely defined by the system administrator. Each context is composed by one or more extensions. These extensions are simply a named set of actions. As summary, the dial plan is a collection of contexts and each context is a collection of extensions.

  More information and an example of *extensions.conf* can be found in appendix C.10. More information about dial plan and extensions can be found in the Asterisk documentation website [11].

4.6 Solution limitations

This section describes some of the main limitations of the prototype implemented as well as some of the difficulties experienced during the development phase. It presents some solutions for minimizing these limitations and the choices made regarding features not considered in the implementation.

- **Stream transactions**:
  LibVLC has limitations in stream transactions. The process associated with the exchange stream on the server, begins by stopping the current media content, opening the desired content and advancing it to the right position. For the client this process has some consequences such as possibility of the viewing window to close and re-open, or the contents being affected by presence of visual artefacts for a few seconds.

- **Traffic control**:
  The development plan also includes the implementation of an Hierarchical Token Bucket (HTB) filter for queuing and prioritizing the traffic of the internal wireless interface, in order not to compromise the receiving of multimedia contents. However due to limited development time this was not developed.

- **QoS monitoring**:
  VLC (libVLC) does not allow management at levels as the video frame, or the control of the RTP traffic received. Therefore, it was not possible to measure the RTP packet loss using this library. Losses are monitored by embedded applications - wrappers - which can sometimes lead to wrong conclusions. This limitation could be overcome with the implementation of an extra module on the client, that listens to the network and analyses the structure and content of received RTP packets, such as sequence numbers, type of transported video frames, number of lost video frames, etc. This solution was not implemented, due to the additional complexity and constrained development time.

- **Security**:
  The client (STB) prototype and the multimedia server does not include security features such as access control, users registration, server authentication or content encryption. This choice is justified by the fact that the security aspects are not a priority of this work. However, the modular architecture allows these features to

be added in a future development.

- **Stream failure recovery**:
  Another limitation of the client (STB) is its dependency on server availability, in other words, the program works correctly while the server is available. In case of failure during the session, the client detects the lost connection with the server or with the stream, but does not perform automatic switching to a secondary server or stream. In this case, the user is responsible for starting a new stream.

- **Web-browsers, plugins**:
  During the development stage, it was possible to verify that the VLC plugin for browsers is not compatible with most of them. In fact it only works at full potential with Mozilla Firefox. This issue limits the range of possibilities on the client although Firefox is available for almost all platforms. The VLC community will develop in a near future a better compatibility with the browsers available on the market.

5

Experimental results and discussion

5.1 Introduction

This chapter provides some of the experimental results and conclusions achieved with the developed modules. This chapter starts by introducing the testing environment, where the used equipments, the testing utilities and the configuration process are described. Afterwords, some experiments were conducted with the complete solution, in order to know if it is a good system for future expansion and to assess the quality experience being delivered.

5.2 Environment and testing methodology

The modules implemented for Internet Protocol Television (IPTV), Video on Demand (VoD) and Voice over Internet Protocol (VoIP) were tested independently during the development phase, before any integration in the final solution. Tests were performed after the development of new features for each module. Although no formal testing methodology was followed, these tests were crucial in the developing and testing phases since it is quicker and easier to identify errors in small modules than in a large solution. These tests had three main objectives: the first was to assess the video quality, both Quality of Service (QoS) and Quality of Experience (QoE) of video contents; the second was to test the involved technologies and devices to detect their limitations and to consider a possible commercial application; and the third one was to study the feasibility of using Wireless Local Area Network (WLAN) to deliver multimedia contents to the clients with a satisfactory QoE.

The simplified test environment is shown in figure 5.1. All the VoD tests were done with resource to 720p videos with a 23.98 frame rate. To access the video quality some type of transcoding process were needed. This subject is further discussed in the following sections.

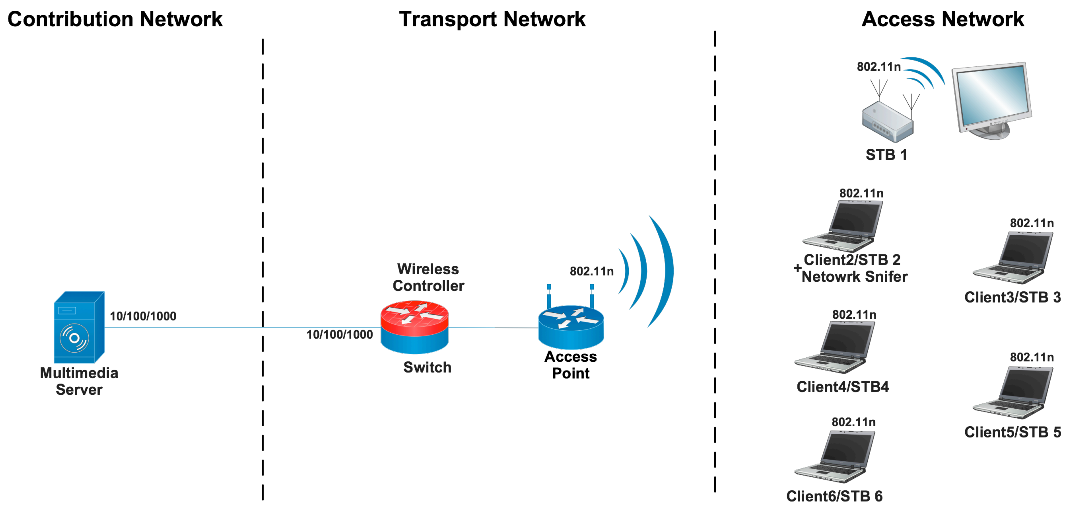


Figure 5.1: Testing scenario used.

The multimedia server is connected via a gigabit ethernet to the wireless controller. The wireless controller is then connected to all the Access Points (AP)s. These APs are responsible for the creation of the connection between the clients and the server. The APs can be power supplied by the controller via Power over Ethernet (POE) standard.

The following subsections describe the equipments used in the testing environment, the tools used for QoE and QoS assessment and the configuration used in the testing stages.

Equipments

The following equipments were used for testing and developing the entire solution:

Wireless Controller:

The wireless controller used for the tests was the Alcatel-Lucent OmniAccess 4306 [40] [42] with an IAP105 [39] AP.

Multimedia Server:

The server used was a Pentium D 2.8 GHz with 2 GB of memory running Linux Ubuntu 10.04 LTS with kernel 2.6.32.36. The server has a Broadcom NetExtreame BCM5755 gigabit ethernet and two hardware interfaces, the OpenVox A400P with one Foreign Exchange Office (FXO) module for Asterisk and the Terratec Cinergy T Stick Dual Universal Serial Bus (USB) adapter for IPTV module.

Clients:

- Set-top box:

The Set-Top Box (STB) is the client build for the triple-play solution. The STB is an Intel Atom D525 1.8 GHz with 2 GB of memory, with a SSD hard drive and a Realteck RTL 8191SE wireless interface (supporting 802.11n) [60]. The STB also has a standard Infrared (IR) Media Center Edition (MCE) receiver, a standard MCE remote control and an Asus USB-N53 wireless adapter (supporting 802.11n) [43]. The remaining clients were standard laptops with standard specifications. The following laptops were used:

- Dell Studio XPS 1340:
- Toshiba Satellite A200-286:
- LG F1 Pro-Express Dual:
- Fujitsu Siemens Esprimo v5515:
- Macbook Pro 8.1:

Most of the laptops used had an Intel core 2 Duo 2 GHz processor or above with 2 Gb of memory and with Windows 7 professional and ultimate editions. The external wireless adapters used were respectively: TP-Link TL-WDN3200 [114], D-Link DNA-127 [53], TP-Link TL-WN821N [113] and Edimax EW-7733Und [54].

All of these wireless adapters support 802.11n standard, however four of them also support dual-band (2.4 GHz and 5 GHz), respectively the TP-Link TL-WDN3200, the Edimax EW-7733Und, the Asus USB-N53 and the internal wireless adapter from Macbook Pro. Both Fujitsu and Toshiba Laptops were used to generate traffic into the network. This subject is discussed in the following sections.

Used tools

To test the video QoE and the network capabilities while using the solution developed, several software and utilities were used. The most important software used for the mentioned tests were MSU Video Quality Measurement Tool (VQMT) [118], Wireshark [121] and the Alcatel Web User Interface (WebUI) [41]. The Wireshark and the Alcatel controller web UI were used for network analysis and the VQMT was used for pos-processing and evaluation of the QoE.

MSU VQMT

The MSU VQMT is a software used to perform deep comparative objective analysis of video quality. The main functionality of this software is to calculate objective quality metrics for video content using Full-Reference (FR) or No-Reference (NR) types of analysis. The main application areas of this tool are the video/image codec quality analysis, the video/image processing algorithm comparison and the different algorithms for image and video analysis. MSU VQMT is able to use several objective quality metrics. It has special Software Development Kit (SDK) for objective metrics development and implementation, which handles different video formats and color spaces. It also has options to visualize objective metrics values and other useful features for a complete, fast and accurate objective quality assessment for multimedia content.

Wireshark

Wireshark is an open source tool for profiling network traffic and analysing packets. This tool is often referred to as a network analyser, network protocol analyser or sniffer. It can be used to examine the details of traffic at a variety of levels ranging from connection-level information to the bits that make up a single packet. Packet capture can provide a network administrator with information about individual packets such as transmission time, source, destination, protocol type and header data. This information can be useful for evaluating security events, troubleshooting network device issues and network analysis.

Alcatel WebUI

The Alcatel WebUI is one of the available interfaces that can be used for remote administration of the wireless controller. This WebUI also includes network monitoring features that provide enhanced visibility into the wireless network performance and usage. These allow to easily locate and diagnose any WLAN issues in the controller. This interface also allows monitoring and troubleshooting Radio Frequency (RF) issues in the WLAN.

Configuration

Distributing video over WLAN has long been the promise, but the results have been less than satisfactory due to the limitations of slow 802.11 clients, inadequate multicast capabilities, and the lack of control over client behaviour. To optimize video content from end-to-end over WLAN there are considerations that need to be made in the environment. The wireless controller is the key equipment in the transport network, since all the traffic goes through it. Several configuration adjustments were made in the wireless controller and in the AP to achieve a successful video deployment in final testing environment [41]:

- Configure Dynamic Host Configuration Protocol (DHCP) server and Virtual LAN (VLAN).
- Enable Internet Group Management Protocol (IGMP): Enabling IGMP facilitates the video delivery only to those APs with subscribers.
- Set traffic tagging and prioritization:
  - Enable Wireless Multi-Media (WMM) shaping policy: The WMM traffic shaping provides bandwidth allocation among specific categories; in this case the priority was set to video and voice traffic.
  - Create video Access Control List (ACL) rules: The ACL rule will tag all traffic destined to a multicast address with a Type of Service (TOS) value and direct it towards the high priority queue in order to ensure end-to-end QoS.

- Enable Adaptive Radio Management (ARM): The ARM feature is a distributed approach to enable self configuring and self healing wireless networks. In order to fully utilize the available spectrum, increase the system capacity, and the number of supported users, ARM dynamically learns about the RF medium and adapts accordingly. This is accomplished by the AP that periodically scan available spectrum channels, analyse the interference level seen on other channels and reports it back to the controller. The controller then computes the most optimized setting and instructs the AP to work accordingly. There is a small amount of time when the AP leaves its channel to conduct scanning operations that are essential to ARM. When there are real time applications such as video/voice running on the network, this periodic scan might result in latency, jitter and eventually degrade the quality of the signal.

- Enable the APs to allow both dual-band (2.4 GHz and 5 GHz) and single-band (only 2.4 GHz).

- Enable band steering : This feature enhances performance and user experience by automatically moving clients to the 5 GHz band when capable and reserving the 2.4 GHz band for single-band clients.

- Enable Dynamic Multicast Optimization (DMO): WLAN transmissions can benefit from unicast transmissions depending on the number of clients in use. If only a small number of clients are subscribed to a multicast group, it can be more efficient to convert over-the-wire multicast to over-the-air unicast due to the faster data rates and prioritization capabilities of unicast connections.

- Enable multicast rate optimization: In cases where DMO determines that it is more efficient to send traffic over-the-air as multicast, Multicast Rate Optimization (MRO) supports higher data rate multicast frame transmissions increasing multicast traffic capacity in a given channel. The AP transmits multicast traffic at the lowest common rate sustainable for all associated subscribers instead of using the lowest common supported rate for all clients. This allows conservation of wireless bandwidth and higher video density.

Before defining the actual testing environment, several other testing environments were tried. The entire system was tested using different wireless interfaces both, 802.11n single-band (only 2.4 GHz) adapters and 802.11g. The system was also tested with other types of wireless controller, although all these results were not as good as expected: the video had jerkiness and had a very high level of block distortion. The other wireless controllers tested were Meru MC1500 [97] with an 320i AP [96] and a Cisco ESW 520 switch [51]. Besides these results, adjustments and configurations were done in both server and client at the cache level. Two types of cache were added: a network cache and a Real-Time Streaming Protocol (RTSP) cache, both directly supported by VLC. The threads responsible for the decoding process in VLC were increased in order to try to solve the jerkiness and block distortion, however none of these adjustments were fully successful.

Testing scenarios

Three main scenarios were defined for the testing process, to check the system behaviour both in terms of network metrics and in terms of video quality metrics, for the mix of different types of load on the network, and considering the topology represented in 5.1.

The scenarios are briefly described below:

1. Scenario 1: In scenario 1 it is analysed the behaviour of the system for a mix of DATA and VoD traffic. The measurements include the QoE metrics Mean Square Error (MSE), Peak-to-peak Signal-to-Noise Ratio (PSNR) and Video Quality Metric (VQM) for video quality, and jitter delay and bandwidth for QoS network metrics.

2. Scenario 2: This scenario analyses the influence of adding Session Initiation Protocol (SIP) traffic to scenario 1. It tests the same metrics for video, but it also verifies if VoIP audio is correctly received in the uplink.

3. Scenario 3: It considers the same conditions described in scenario 1 with additional multicast traffic (IPTV). In this case RTSP, Real-Time Transport Protocol (RTP) and DATA are tested simultaneously.

It was necessary to parametrize how many clients can consume media over the

WLAN. This number of clients is defined to the maximum value such that the video quality metrics are not affected at any of the clients connected over WLAN to the multimedia server. In this test a growing number clients (STB) is added in order to measure how many clients the system supports. Number N is defined as the number of clients who can simultaneously view the video sample without any noticeable loss of quality. The number N measured in the initial set of tests was four and was used as a reference to all the remaining testing scenarios.

The DATA traffic used in the testing process were User Datagram Protocol (UDP) traffic. To generate this traffic the used tool was Iperf [64]. Iperf is a client server traffic generator with text-based interface. the command *iperf -s -u* was executed in the server to start the software. The parameter *-s* says that Iperf should act as a server, and the *-u* indicates that the protocol to be used is UDP. On the client side, was used the command: *iperf -u -c -b <bandwidth> -t 180* to start Iperf as a client. The parameter *-c* defines the Iperf to act as a client, *-u* indicates that the protocol used is UDP, *-b <bandwidth>* denotes the bandwidth to be used in transmission packages, finally *-t <180>* indicates the duration of the costumer transmission, defined in seconds.

Tables 5.1 show an overview of the 3 scenarios described before.

Table 5.1: Testing scenarios parameters.

Type	Data (in Kbps)	N°. Clients (N=4)	Testing parameters	
VoD Data	0 Kbps	1,2, ... N	Network Measurements	Jitter, Bandwidth, Packet Loss, Delay
			Quality of Experience	MSE, PSNR, VQM
VoD Data SIP	100 Kbps	1,2, ... N	Network Measurements	Jitter, Bandwidth, Packet Loss, Delay
			Quality of Experience	MSE, PSNR, VQM
VoD IPTV Data	200 Kbps	1,2, ... N	Network Measurements	Jitter, Bandwidth, Packet Loss, Delay
			Quality of Experience	MSE, PSNR, VQM

5.3 Testing results

Some adjustments were done in the video samples to accomplish the testing scenarios defined before and due to limitations of the VQMT tool. Figures 5.2 and 5.3 describe the adjustments done. Figure 5.2 shows that the original video (the video sample that is stored in the multimedia server) and the video sample that is streamed and received in the client (STB) are converted to adapt them to the VQMT tool.

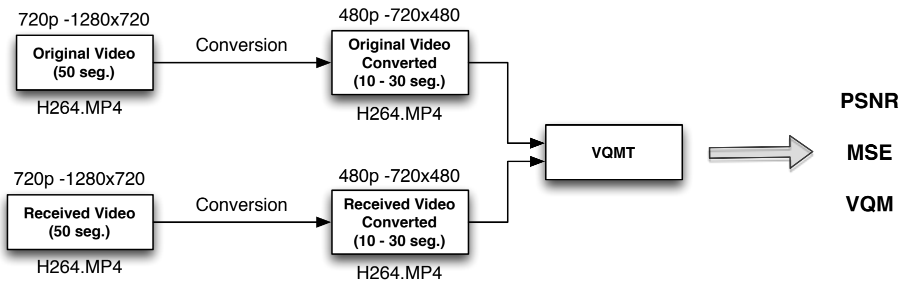
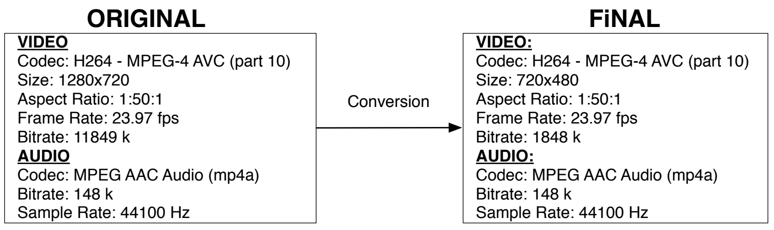


Figure 5.2: Video sample conversion diagram.

The VQMT software version used only supports standard quality video samples as input for video quality assessment. As mentioned before the video samples used were 720p quality. Therefore a conversion to 480p quality had to be done in the video samples (the maximum supported quality video by VQMT). Another important conversion was the duration of the video sample. The initial video samples had 50 seconds, however to avoid jerkiness and block distortion caused by not using buffers, the video sample used in the video quality assessment was only 20 seconds (from 10$^{\text{th}}$ second to 30$^{\text{th}}$ second, excluding the first 9 seconds because of the initial jerkiness and block distortion).


ORIGINAL
VIDEO
Codec: H264 - MPEG-4 AVC (part 10)
Size: 1280x720
Aspect Ratio: 1:50:1
Frame Rate: 23.97 fps
Bitrate: 11849 k
AUDIO
Codec: MPEG AAC Audio (mp4a)
Bitrate: 148 k
Sample Rate: 44100 Hz

Conversion

FiNAL
VIDEO:
Codec: H264 - MPEG-4 AVC (part 10)
Size: 720x480
Aspect Ratio: 1:50:1
Frame Rate: 23.97 fps
Bitrate: 1848 k
AUDIO:
Codec: MPEG AAC Audio (mp4a)
Bitrate: 148 k
Sample Rate: 44100 Hz

Figure 5.3: Video sample conversion codec data.

Figure 5.3 shows detailed information about the codec and video/audio properties of both original video sample and converted video sample.

In resume, 480p video samples were used for quality assessment purposes, however the project is working with full support for 720p video samples.

Because many statistical information was retrieved from this experiments, only the ones considered more relevant are shown in this section. Yet, more information can be seen in appendix D, regarding all the experiments presented in this chapter.

Network Metrics

Table 5.2 shows the streaming usage both in the channel AP and in the client (STB), considering an incremental number of clients for the scenario 1 with 100 Kbps of traffic. In this case two clients were used for traffic generation (100 Kbps) and four clients for streaming opening. For each client, one portal application interface was open, representing one unicast connection to the multimedia server. After all the clients terminate, a single video file (representing each used computer/STB) is saved in the system, converted for the right proportions and compared with the original file (this comparison will be discussed later in this chapter). As can be seen in the table, there were no significant losses (<2%) in the content received for most of the cases. In a viewing situation this means that the viewer would experience some jitter and/or block distortion during a certain period of time. It can also be seen that each client/STB uses approximately 13 Mbps of bandwidth, that represents 7% of the capacity of the AP channel. When four clients are consuming media the total usage increases to approximately 33% of channel capacity. A very low percentage of the packets were retransmitted or dropped.

Figure 5.4 shows how the multimedia server bandwidth varies during the video file consumption in the same testing scenario as before. These values show that with the increase of the number of clients from one to four, the bandwidth varies from approximately [1.5,2] MiB/s to [6,6.5] MiB/s (that corresponds to approximately [12.6,16.7] Mbps to [50.3,54.5] Mbps).

In scenario 2 it was expected to have slightly more traffic since VoIP traffic has been added to the connection. However, as can be seen in table 5.3, the traffic from each client is approximately the same as the previous scenario, 13 Mbps. From table 5.3 we can also conclude that adding VoIP traffic (i.e. SIP signalling plus the audio streams traffic) does not denote major changes in VoD QoS. However, it can be seen an increase of loss rate

Table 5.2: Network metrics for 100Kbps of data traffic in scenario 1. The channel capacity measures the AP's estimation for the total datarate available, the utilization measures the channel utilization.

data = 100 Kbps					
AP Channel statistics					
N. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interference [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
1	-90	7 %/2 %	<1 %	214.1	13.7
2	-90	14 %/3 %	1 %	209.4	26.3
3	-90	24 %/5 %	2 %	176.3	38.6
4	-90	33 %/5 %	2 %	226.8	53.2

Clients (STB) statistics				
N. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retried [%]**	**Frame Dropped [%]**
1	216.8	13.7	<1 %	0 %
2	201.2	12.7	<1 %	0 %
	194.3	13.6	<1	<1
3	181.6	13.1	<1 %	<1 %
	152.1	12.2	<1 %	<1 %
	200.7	13.3	2 %	<1 %
4	196.2	13.7	<1 %	0 %
	232.1	12.8	1 %	0 %
	229.9	13.8	<1 %	0 %
	250.3	12.8	<1 %	0 %

since VoIP traffic, contrarily to VoD and IPTV, adds traffic in the uplink channel. This traffic leads to less network capacity, despite the less bandwidth that VoIP connections use. In the table, it can also be seen higher values of frames retried (3 % - 4%), contrarily from scenario 1. These values are most likely because of an external interference in the WLAN environment. It was also verified that all the VoIP connections were in perfect quality conditions. The only less positive results obtained were the delay of one or two seconds between the caller and the called user and the noise due to the low quality microphones used in the tests.

Figure 5.5 shows the multimedia server bandwidth used in scenario 2. It can be seen that, with one open VoD stream and three VoIP connections the bandwidth used is approximately 3 MiB/s (25 Mbps). With two VoD streams and two VoIP connections the bandwidth is approximately 4.5 MiB/s (37.75 Mbps) and finally with three VoD streams the bandwidth is approximately 6 MiB/s (50.33 Mbps). The presented values were inside

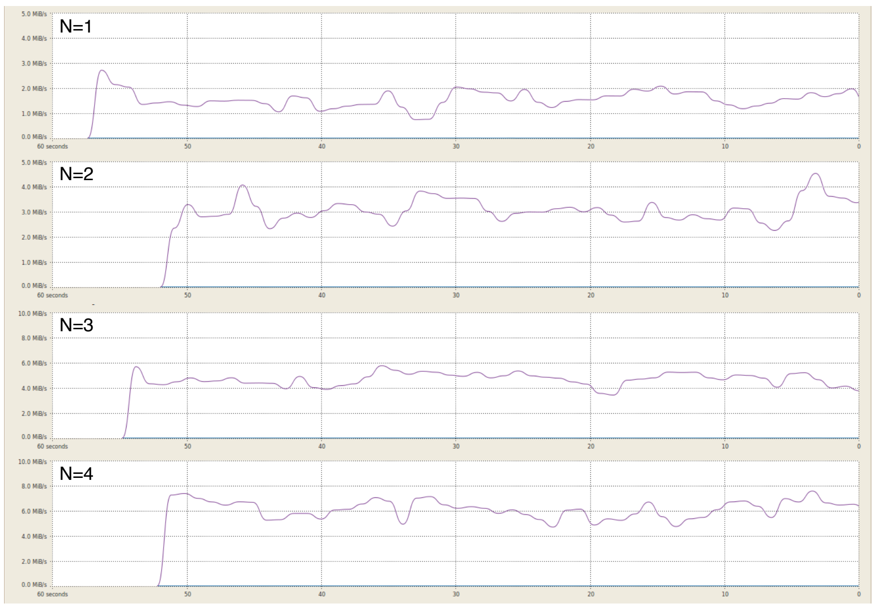


Figure 5.4: Network usage in scenario 1 from the multimedia server.

of what is expected from the multimedia bandwidth usage in the multimedia server for scenario 2, given that each VoIP audio connection uses only about approximately 300 Kbps.

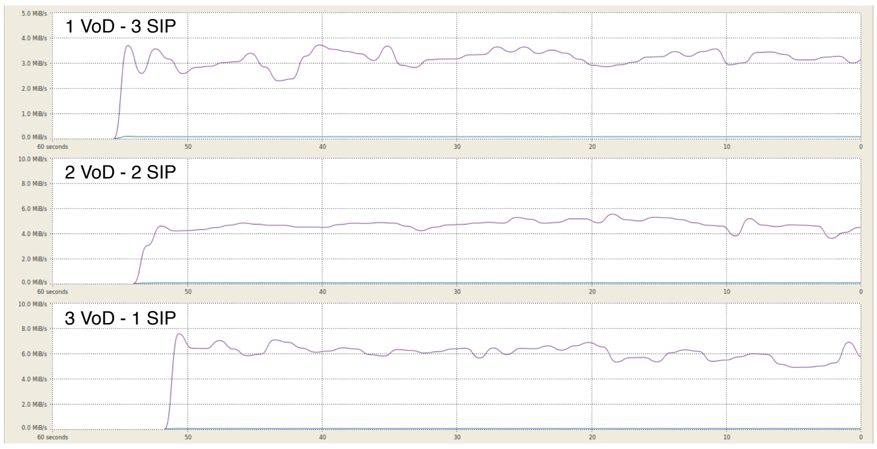


Figure 5.5: Network usage in scenario 2 from the multimedia server.

Table 5.4 shows the results for the experiments conducted in scenario 3, with multiple types of traffic (VoD, IPTV and 100 Kbps of data traffic) and an incremental number of clients. It can be seen that the increase in the number of opened VoD streams leads to an increase of the channel usage from 10% to 39%, the highest load tested. The channel

Table 5.3: Network metrics for 100Kbps of data traffic in scenario 2.

data = 100 Kbps					
AP Channel statistics					
N°. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interference [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
3 VoD 1 SIP	-91	26 %/5 %	2 %	201.4	39.7
2 VoD 2 SIP	-91	17 %/4 %	1 %	183.9	26.7
1 VoD 3 SIP	-90	8 %/4 %	1 %	197.3	14

Clients (STB) statistics				
N°. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retried [%]**	**Frame Dropped [%]**
3 VoD 1 SIP	172	12.9	<1 %	0 %
	226	11.7	3 %	0 %
	202	12.9	4 %	0 %
2 VoD 2 SIP	205.3	13.5	<1 %	<1 %
	193.1	13	4 %	0 %
1 VoD 3 SIP	235.8	13.5	<1 %	<1 %

usage variation is mainly due to the increment of clients, and in the scenario 3, to the increment of the IPTV traffic that, in this case, is a constant value (near 6% of channel capacity).

Figure 5.6 shows a normal usage of bandwidth, suggesting that the multimedia server could support even more users. The traffic monitor is turned at the time 50 seconds, having no significance the values before this moment. In this case, the bandwidth is slightly bigger since there is additional IPTV traffic.

The network metrics show that this quantity of data traffic is not a key factor for the system disturbance. The main variation of bandwidth usage both in the multimedia server and in the channel AP came from the increase in the number of open VoD streams. Besides the VoD streams, SIP and IPTV can also influence the system. However, their traffic was not expressive in the system bandwidth occupation, in the tests conducted. An important factor is the interference from external sources. As is known, WLAN is susceptible to RF interference coming from other wireless, equipments present in the

Table 5.4: Network metrics for 100Kbps of data traffic in scenario 3.

data = 100 Kbps					
AP Channel statistics					
N°. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interference [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
1	-90	10 %/3 %	1 %	202.3	16.3
2	-90	16 %/3 %	1 %	206.1	29.9
3	-90	25 %/4 %	2 %	193.5	40.2
4	-91	39 %/5 %	2 %	184.9	54.3

Clients (STB) statistics				
N°. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retried [%]**	**Frame Dropped [%]**
1	210.4	16.3	<1 %	<1 %
2	135.7	13.2	<1 %	<1 %
	223.1	13.3	<1 %	<1 %
3	177.8	13.7	<1 %	0 %
	186.7	13.3	<1 %	0 %
	220.5	13.2	1 %	0 %
4	208.6	13.3	<1 %	<1 %
	154.6	12	3 %	0 %
	157.4	12.2	<1 %	<1 %
	246.9	12.2	<1 %	<1 %

testing environment. These external factors can be seen as the main problem in the contamination of the experimental measurements. Although the good results of noise values showed in tables 5.2, 5.3 and 5.4, the results shown in the following sections suggest that the experiments were contaminated by external factors, which resulted in video quality metrics slightly different from what could be expected from the noise values measured at the network level.

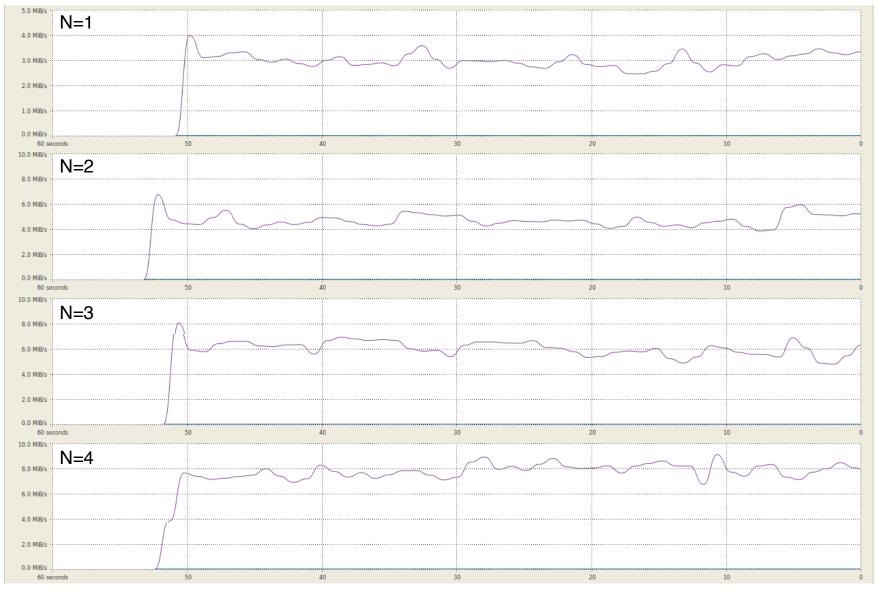


Figure 5.6: Network usage in scenario 3 from the multimedia server.

PSNR Metric

Figures 5.7, 5.10 and 5.11 show the PSNR metric for the three testing scenarios analysed in the section above. Figure 5.7 strangely shows that with more clients (VoD open streams) the PSNR metric gets better results than with fewer clients. After evaluating the tested videos, it was verified that these two bad results (with one and two clients) came from a failure in time synchronization of the two videos analysed. Since all the tests were done with a FR metric and since the videos were not synchronized, these two results do not express the correct result for the PSNR metric. For the remaining tests, all clients show good results (the slightly variation of results came from external interferences in the WLAN environment). Figures 5.8 and 5.9 depict screen shots of "before" and "after" transmission video frames. In the first case these frame pictures clearly show the lack of synchronization. The second case shows two synchronized frames. This second result leads to much better results when measuring video quality metrics for the obvious reasons.

Figure 5.10 also shows some lack of synchronization for the test with two open VoD streams and two open SIP calls. This synchronization issue was very small (mostly notorious in the sound) and lead to a smaller difference than in the figure 5.7. The results

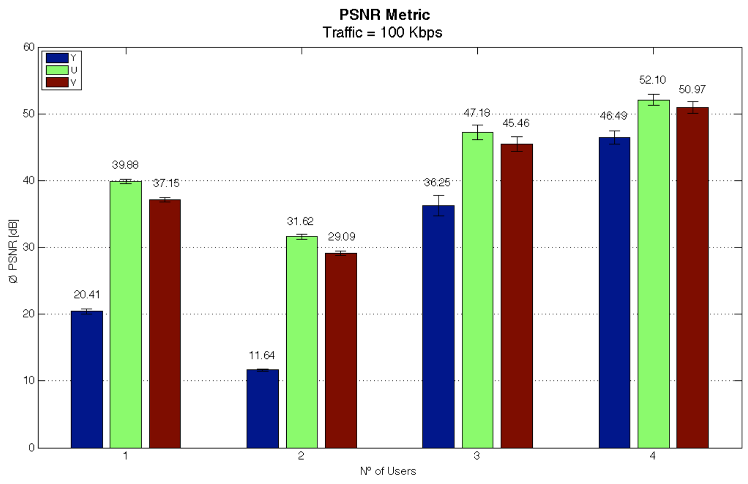


Figure 5.7: PSNR Metric for 100Kbps of DATA traffic in scenario 1.

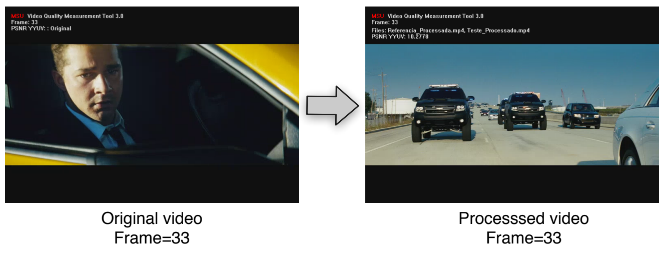

MSU Video Quality Measurement Tool 3.0
Frame: 33
PSNR YYUV: : Original
MSU Video Quality Measurement Tool 3.0
Frame: 33
Files: Referencia_Processada.mp4, Teste_Processado.mp4
PSNR YYUV: 10.2778
Original video
Frame=33
Processsed video
Frame=33

Figure 5.8: Frame comparison with N=2 clients in PSNR Metric for 100Kbps of DATA traffic in scenario 1.

from one VoD stream and three VoD streams are good and shows a good performance of the overall system.

Figure 5.11 shows two good results for one and two clients in scenario 3. With three clients a small synchronization issue is visible, and with four clients the frame desynchronization is visible in various points of the video.

Another interesting result that can be seen during the manual analysis of the video samples is that some good PSNR results also show some block distortion and jerkiness. For example, for the scenario 1 with N=3, the slightly different PSNR value for N=4 case is result not only of external WLAN interferences but also of some block distortion and jerkiness that the video sample showed. This block distortion and jerkiness could also be seen in scenario 2 (2VOD-2SIP) and in scenario 3 (N=4).

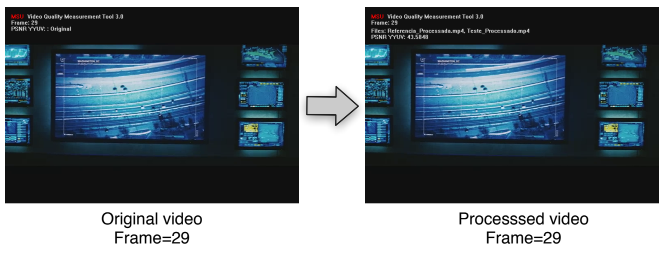

MSU Video Quality Measurement Tool 3.0
Frame: 29
PSNR YYUV: : Original
MSU Video Quality Measurement Tool 3.0
Frame: 29
Files: Referencia_Processada.mp4, Teste_Processado.mp4
PSNR YYUV: 43.5848
Original video
Frame=29
Processsed video
Frame=29

Figure 5.9: Frame comparison with N=4 clients in PSNR Metric for 1000Kbps of DATA traffic in scenario 1.

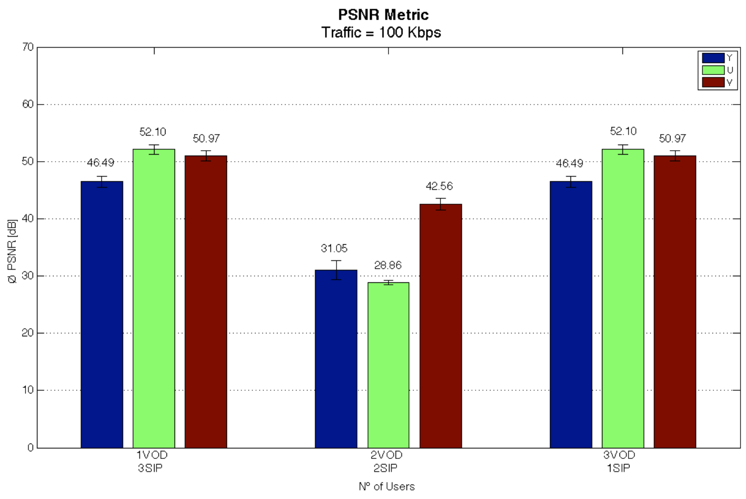


Figure 5.10: PSNR Metric for 100Kbps of DATA traffic in scenario 2.

MSE Metric

The MSE metric has a lower error tolerance threshold than the PSNR metric, leading to a sharp increase of the MSE values (meaning lower quality). For this reason the results lead to what looks like very bad video quality results. However, after some depth analysis of the video samples, it was concluded that just a small synchronization issue or some block distortion/jerkiness cause a huge increase of the MSE metric. Figures 5.12, 5.13 and 5.14 show the MSE metric analysis for scenario 1, 2 and 3 respectively. They confirm the conclusions taken before using the PSNR metric. Scenario 1 for N=2 has a synchronization problem that leads to the high value of the MSE metric shown in figure 5.12. For the rest of the cases, it can be seen a small increase due to the external WLAN interface. It should be noticed the result for N=4 with a low MSE value, demonstrating that for this number of clients and this quantity of traffic the system is fully stable.

Following the same reasoning, figure 5.13 also shows a high MSE value for two

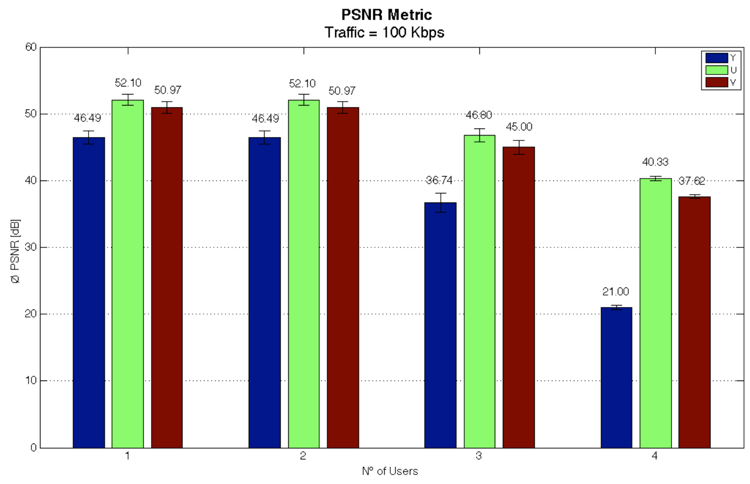


Figure 5.11: PSNR Metric for 100Kbps of DATA traffic in scenario 3.

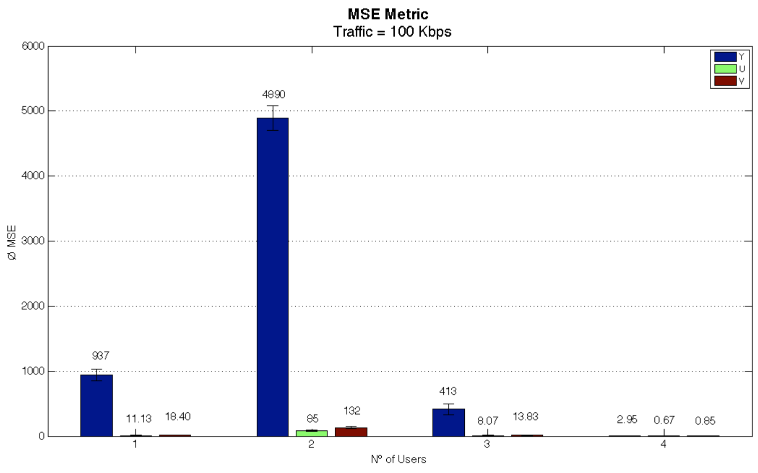


Figure 5.12: MSE Metric for 100Kbps of DATA traffic in scenario 1.

VoD clients and two VoIP connections in scenario 2, like in figure 5.10. However, since PSNR metric does not grow so fast as MSE metric, figure 5.10 highlights more the synchronization problem. Similar conclusions apply for figure 5.14.

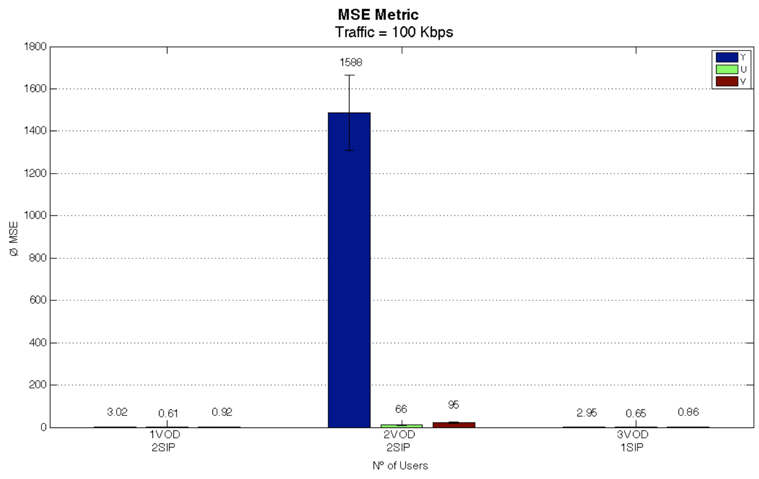


Figure 5.13: MSE Metric for 100Kbps of DATA traffic in scenario 2.

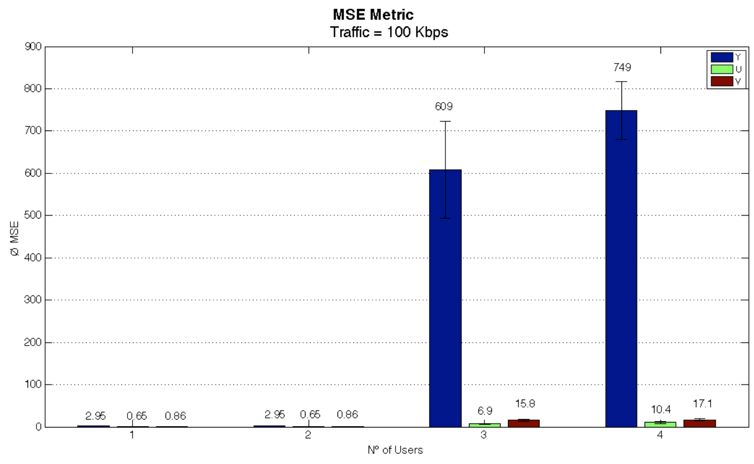


Figure 5.14: MSE Metric for 100Kbps of DATA traffic in scenario 3.

VQM Metric

The VQM is very sensitive to errors and its variation is similar to the MSE metric showed above. The VQM metric signals better quality as more close to zero the value is. Figure 5.15 shows one good result for N=4, meaning that the other three (N=1,2,3) have either interference or block distortion/jerkiness. Figure 5.16 shows good results for both one VoD stream and three VoD streams, and sync problems for the last case (as in the rest of the metrics). Finally, figure 5.17 shows good results for N=1 and N=2, a slightly block distortion and jerkiness for the N=3 and a sync problem for N=4.

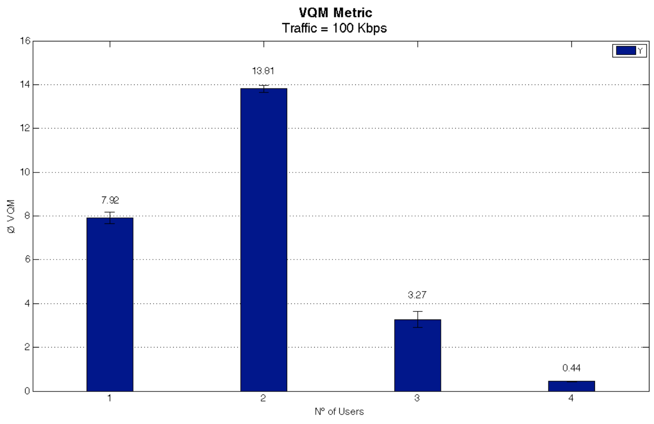


Figure 5.15: VQM Metric for 100Kbps of DATA traffic in scenario 1.

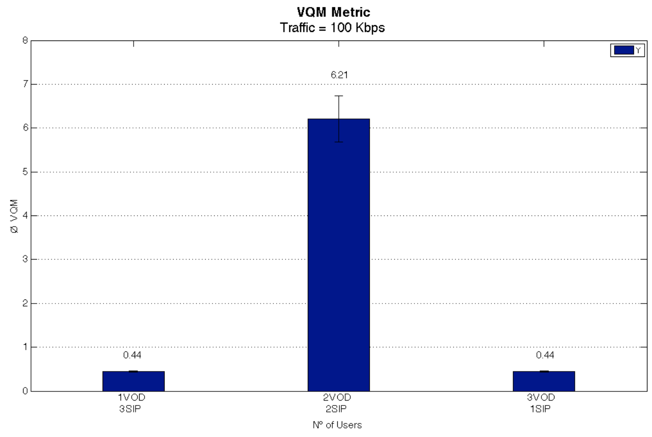


Figure 5.16: VQM Metric for 100Kbps of DATA traffic in scenario 2.

The results were conclusive. The system deployment proved to be capable of dealing with the service delivery circumstances. Besides some inferior results from some testing metrics, the overall objective of the testing phase was achieved since the system was

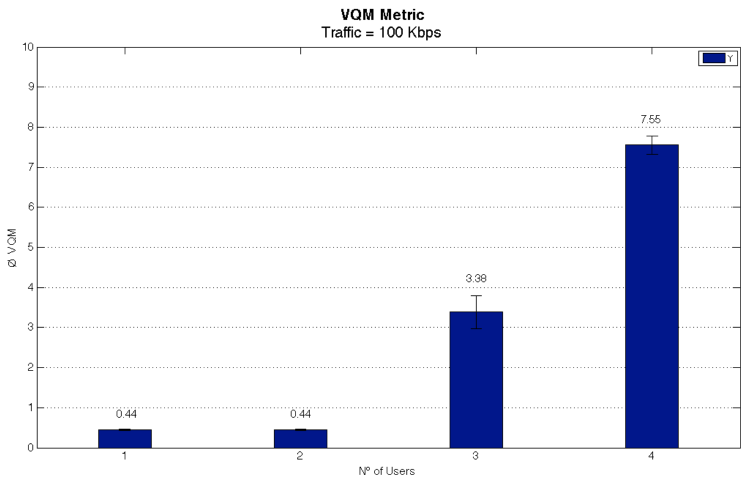


Figure 5.17: VQM Metric for 100Kbps of DATA traffic in scenario 3.

able to deliver full service via WLAN to the clients. Although the experiments were carried out with a small number of test sequences and a relatively small number of clients, some useful conclusions could be taken. To achieve more conclusive results, more tests with more clients, and in different types of scenarios should be preformed.

From the results and experiments done, it can also be concluded that any of the three metrics, respectively PSNR MSE and VQM, could be used to measure the video quality with similar results. However, the effect of the desynchronization in time affected the results, thus the effective value of QoE is higher than the ones actually showed in the results of the tests, as the quality perceived by the viewer is not so sensitive to frame desynchronization.

6

Conclusions and Future Work

6.1 Conclusions

The purpose of this thesis was to create a feasible structure for a triple-play system capable of providing IPTV, VoD and VoIP services in a reliable manner. To achieve this propose, it was a system capable of streaming both video and audio content to a client, and to run through a web-browser interface with a WLAN infrastructure. After an extensive research in the available technologies, it was possible to have a better perception of all the pieces that were needed to build this system.

Perhaps the biggest challenge was to properly deliver the video and audio contents (IPTV, VoD and VoIP) inside the web browser, using the common available tools. The client application gave a good perception of what can be done with web based tools and what is the potential of this type of technologies. During the development, only JApplet was able to deliver VoIP functionalities, however webRTC and HTML5 is starting to appear with new developments in this subject.

On the other hand, the VLC choice was very helpful and complete enough to deal with the needed streaming services. Also Asterisk and DVBlast were essential, since all these tools were able to correctly deliver their services.

The experimental tests allowed to conclude that it is possible to provide triple play services through WLAN, however this transmission is highly affected by external elements, as seen in test results for video quality metrics. Despite the test results, the visual video does not suffer significant visible loss of quality. As expected, with the increase of the number of clients it was visible an increase of bandwidth used for both WLAN and multimedia server. In the studied cases, the number of clients did not affect significantly the used bandwidth since the number of clients was small.

Triple-play solutions do not come cheap. They have high network costs, besides hardware and software. The use of open-source tools joined into one single application, proved to be possible the creation of a free triple-play structure that could easily level with any available product.

6.2 Future Work

The system created was not fully completed and can be extended by adding new features that at this point were not considered essential. In this line of thought, this section proposes some suggestions for future work that is compatible with the implemented architecture.

In terms of administration, a centralized administration portal can to be created unifying all the administration platforms (IPTV, VoD and VoIP). This process will help the service administrate to easily detect problems and act with more accuracy in the solution of the problems.

One of the main improvements to be done is the QoS monitoring. The implementation of a extra module on the client that listens and analyse the RTP packets received, including missing sequence numbers, type of video frames transported, video frames lost or corrupted is suggested. These analysis will allow the monitor module to act with more accuracy in the dynamic process of choosing the most appropriate level of quality to the state of the network, increasing the this way the QoE.

Aspects related to security are another point to develop. In this area, an AAA server (Authentication, Authorization and Accounting), which would be responsible for storing and managing all profiles and authentication credentials of customers could be implemented. The authentication server and the content is also important for the customer, to ensure that it receives the desired contents. For this, it can implement a system-encryption of content through the exchange of keys and digital certificates. This mechanism would also be important to ensure the intellectual property of content on the network.

More robust and sophisticated solutions could be added to the current implementation. The Data Base (DB) could provide some sort of special accounts where the users would have access only to certain provide specific video contents. Scaling those accounts through different privileges would also be an interesting option with several available packages. A billing solution could provide those choices to the user. Here a billing solution to the calls would also be another interesting implementation, since an agreement can be done with the phone contractors and profit could be taken from the same calls.

Bibliography

[1] Triple-play services quality of experience QoE requirements. Technical Report TR-126, DLS Forum.

[2] IMDB. http://www.imdb.com, 1990-2013. Accessed: 2013-10-14.

[3] DVEO - Pro Boradcast Division . http://www.dveo.com/broadcast-systems/transport-stream.shtml/, 2003. Accessed: 2013-08-15.

[4] Introduction to H.264. Technical report, ATI Technologies, 2005.

[5] Understanding MPEG-4: Technologies, Advantages and Markets. Technical Report MP-IN-40182, The MPEG Industry Forum, 2005.

[6] About IPTEL. http://www.iptel.org/about, 2007. Accessed: 2013-09-5.

[7] Digital Video Broadcasting (DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks. Technical Report TR-102034 V1.4.1, European Telecommunications Standards Institute, August 2009.

[8] OpenVox - Asterisk Cards. http://www.openvox.cn/web/en/products/telephony-cards/fxofxs-cards/a400p.html?page=shop.product_details&flypage=flypage.tpl&product_id=19&category_id=1, 2009. Accessed: 2013-11-25.

[9] Terratec - Cinergy T Stick Dual RC. http://www.terratec.net/en/products/Cinergy_T_Stick_Dual_RC_102261.html, 2009. Accessed: 2013-11-11.

[10] Asterisk - Asterisk Architecture, Types of Asterisk Modules. https://wiki.asterisk.org/wiki/display/AST/Types+of+Asterisk+Modules, September 2010. Accessed: 2013-08-30.

[11] Asterisk documentation. https://wiki.asterisk.org/wiki/display/AST/Dialplan+Fundamentals, 2010. Accessed: 2013-12-7.

[12] Human factores (HF) QoE (quality of experience) requirements for real-time communication services. Technical Report TR-102643, European Telecommunications Standards Institute, January 2010.

[13] Asterisk - Asterisk Architecture, The Big Picture. https://wiki.asterisk.org/wiki/display/AST/Asterisk+Architecture, May 2011. Accessed: 2013-08-30.

[14] Lirc - Linux Infrared Remote Control. http://www.lirc.org, 2011. Accessed: 2013-10-14.

[15] OpenVox - A400P User Manual. http://downloads.openvox.cn/pub/manuals/Release/English/A400P_on_DAHDI_User_Manual.pdf, 2011. Accessed: 2013-12-5.

[16] Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); IPTV Architecutre; IPTV functions supported by the IMS subsystem. Technical Report TS-182027, European Telecommunications Standards Institute, March 2011.

[17] Darwin Streaming Server. http://dss.macosforge.org/, 2012. Accessed: 2013-08-15.

[18] Ieee standard for information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific

requirements-part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications amendment 3: Enhancements for very high throughput in the 60 ghz band. *IEEE Std 802.11ad-2012 (Amendment to IEEE Std 802.11-2012, as amended by IEEE Std 802.11ae-2012 and IEEE Std 802.11aa-2012)*, Dec 2012.

[19] Red5 - The Open Source Media Server. `http://www.red5.org/downloads/docs/red5-reference-1.0.pdf`, 2012. Accessed: 2013-08-15.

[20] Asterisk. `http://www.asterisk.org/`, 2013. Accessed: 2013-09-5.

[21] biTStream. `http://www.videolan.org/developers/bitstream.html`, 2013. Accessed: 2013-11-15.

[22] dvb-apps. `http://www.linuxtv.org/wiki/index.php/LinuxTV_dvb-apps`, 2013. Accessed: 2013-11-15.

[23] DVBlast. `http://www.videolan.org/projects/dvblast.html`, 2013. Accessed: 2013-11-4.

[24] European Telecommunications Standards Institute. `http://www.etsi.org/`, 2013.

[25] FreePBX documentation. `http://www.freepbx.org/support/documentation`, 2013. Accessed: 2013-12-7.

[26] FreeSWITCH. `http://www.freeswitch.org`, September 2013. Accessed: 2013-09-5.

[27] Internation Telecommunication Union. `http://www.itu.int`, 2013.

[28] IT9130 DVB-T. `http://www.ite.com.tw/EN/products_more.aspx?CategoryID=6&ID=21,120`, 2013. Accessed: 2013-11-10.

[29] ITU Video Compression - H.263, H.264, MPEG-4 Information site. `http://www.h2631.com`, March 2013.

[30] Kamailio Features. `http://www.kamailio.org/w/features/`, 2013. Accessed: 2013-09-5.

[31] Linux TV. http://www.linuxtv.org/wiki/index.php/Main_Page, 2013. Accessed: 2013-11-10.

[32] LIVE555. http://www.live555.com/liveMedia/, 2013. Accessed: 2013-11-17.

[33] OpenSIPS About. http://www.opensips.org/About/About, June 2013. Accessed: 2013-09-5.

[34] The IETF home page. http://www.ietf.org, 2013.

[35] The MPEG home page. http://mpeg.chiariglione.org/, 2013.

[36] Video LAN Organization - VLC Playback features. http://www.videolan.org/vlc/features.html, 2013. Accessed: 2013-11-20.

[37] VideoLAN Organization. http://www.videolan.org/vlc/, 2013. Accessed: 2013-08-15.

[38] Youtube. http://www.youtube.com, March 2013.

[39] IAP-105 Wireless Access Point. http://support.arubanetworks.com/DOCUMENTATION/tabid/77/DMXModule/512/Command/Core_Download/Default.aspx?EntryId=5929/, 2013. Accessed: 2013-12-15.

[40] Aruba 600 Series Mobility Controllers. http://www.arubanetworks.com/?product=aruba-600-controller-series, 2013. Accessed: 2013-12-15.

[41] AOS-W 6.1 - Alcatel Lucent - User Guide. https://service.esd.alcatel-lucent.com/pr/retrieveFile?filename=bt_qQk/BqdeRouHil/wpWSJnRcS9lJql&filepath=QuPbIUiWe26ZFOXl0N1YcUOWAWHOFYmUToycm4i0KMTifGigjLecN9AxcaQVy88l, 2013. Accessed: 2013-12-15.

[42] Aruba 620 Branch Office Controller documentation. http://www.arubanetworks.com/pdf/products/DS_A620.pdf, 2013. Accessed: 2013-12-15.

[43] Asus USB-N53. `http://www.asus.com/Networking/USBN53/#overview`, 2013. Accessed: 2013-12-15.

[44] ATIS Interoperability Forum (IIF). *ATIS IPTV High Level Architeutre Standard (ATIS 080007)*, 2007.

[45] Audio-Video Transport Working Group, H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RFC 1889: RTP: A transport protocol for real-time applications. RFC 1889 (Proposed Standard), January 1996. Obsoleted by RFC 3550.

[46] David Austerberry. *The Technology Of Video And Audio Streaming*. Focal Press, Burlington, USA, 2nd edition, 2004.

[47] Gonzalo Camarillo and Miguel-Angel Garcia-Martin. *The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the Cellular Worlds, Third Edition*. John Wiley & Sons, 2008.

[48] Tien-Hsin Chang. Demodulator Device and Demodulation Method for reducing PCR Jitter, January 2012.

[49] Shyamprasad Chikkerur, Vijay Sundaram, Martin Reisslein, and Lina J. Karam. Objective Video Quality Assessment Methods: A Classification, Review, and Performance Comparison. *IEE Transactions on Broadcasting*, 57(2):165–182, June 2011.

[50] Lili Chu. Implementation and application of voip network. In *Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), 2011 2nd International Conference on*, pages 2139–2141, 2011.

[51] Cisco ESW500 Series Switches. `http://www.cisco.com/en/US/prod/collateral/switches/ps5718/ps10143/data_sheet_c78-521740.html`, 2013. Accessed: 2013-12-15.

[52] Luís António Pereira de Sousa. Avaliação de desempenho do PBX Asterisk. Master's thesis, UNIVERSIDADE TÉCNICA DE LISBOA, Instituto Superior Técnico, Departamento de Engenharia de Redes e de Computadores, September 2008.

[53] D-Link Wireless N150 High-Gain USB Adapter. http://www.tp-link.com/lk/products/details/?model=TL-WDN3200, 2013. Accessed: 2013-12-15.

[54] Edimax - EW7733Und. http://www.edimax.com/au/produce_detail.php?pd_id=399&pl1_id=1&pl2_id=44, 2013. Accessed: 2013-12-15.

[55] Xiao Feng. DCT-based Video Quality Evaluation. *Final Project for EE392J on*, 2000.

[56] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. RFC 2616: HTTP/1.1 Hypertext Transfer Protocol. RFC 2616 (Proposed Standard), June 1999. Obsoleted by RFC 2068.

[57] International Organization for Strandardization. Iso/iec jtc1/sc29/wg11/n4668 - overview of the MPEG-4 standard, March 2002.

[58] International Organization for Strandardization. Iso/iec jtc1/sc29/wg11/nmpeg00 - Overview of the MPEG-2 standard, March 2002.

[59] International Organization for Strandardization. Iso/iec 14496.2:2004 - mpeg-4 part 2: Coding of audio-visual objects - part 2: Visual, February 2004.

[60] Foxconn nT-535. http://www.foxconnchannel.com/ProductDetail.aspx?T=NanoPC&U=en-us0000006/, 2013. Accessed: 2013-12-15.

[61] M. Handley, V. Jacobson, and C. Perkins. RFC 4566: SDP: Session Description Protocol. RFC 4566 (Proposed Standard), July 2006.

[62] Johan Hjelm. *Why IPTV? : Interactivity, Technologies and Services*, chapter IPTV Standards and Solution. Telecoms Explained - Unravel Emerging Technologies. Chichester, U.K. Wiley, 2008.

[63] D. Hoffman, G. Fernando, V. Goyal, and M. Civanlar. RTP Payload Format for MPEG1/MPEG2 Video. RFC 2250 (Proposed Standard), January 1998.

[64] IPERF. http://code.google.com/p/iperf/, 2013. Accessed: 2014-1-12.

[65] ISO/IEC. Iso/iec 13818-1:2000(E) - Information technology - Generic coding of moving pictures and associated audio information: Systems, December 2000.

[66] ITU-T. Recommendation P.930 - Principles of a reference impairment system for video, August 1996.

[67] ITU-T. Recommendation P.911 - Subjective audiovisual quality assessment methods for multimedia applications, December 1998.

[68] ITU-T. Recommendation P.910 - Subjective video quality assessment methods for multimed applications, September 1999.

[69] ITU-T. Recommendation BS.1387-1 - Method for objective measurements of perceived audio quality, November 2001.

[70] ITU-T. Recommendation J.144 - Objective perceptual video quality measuremnet techniques for digital cable television in the presence of full reference, March 2001.

[71] ITU-T. Recommendation P.862 - Perceptual evaluation of speech quality (PESQ): an objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs, February 2001.

[72] ITU-T. Recommendation Y.2021 - IMS for Next Generation Networks, September 2006.

[73] ITU-T. IPTV Focus Group Proccedings. In *IPTV Architecture*, pages 89–166, 2008.

[74] ITU-T. IPTV Focus Group Proccedings. In *IPTV Services Requirements*, pages 29–88, 2008.

[75] ITU-T. IPTV Focus Group Proccedings. In *IPTV Services Scnearios*, pages 167–200, 2008.

[76] ITU-T. IPTV Focus Group Proccedings. In *Part II - Texts of the FG IPTV Deliverables*, pages 1–19, 2008.

[77] ITU-T. Recommendation J.247 - Objective perceptual multimedia video quality measurement in the presence of a full reference, August 2008.

[78] ITU-T. Recommendation P.10/G.100 Amendment 2 - Vocabulary for performance and quality of service, July 2008.

[79] ITU-T. Recommendation G.107 - The E-Model, a computation model for use in transmission planning, April 2009.

[80] ITU-T. Recommendation H.323 - Packet-based multimedia communication systems, December 2009.

[81] ITU-T. Recommendation Y.2012 - Functional Requirements and Architecture of Next Generation Networks, April 2010.

[82] ITU-T. Recommendation J.341 - Objective perceptual multimedia video quality measurement of hdtv for digital cable television in the presence of a full reference, January 2011.

[83] ITU-T. Recommendation BT.500-13 - Methodology for the subjective assessment of the quality of televison pictures, January 2012.

[84] ITU-T. Recommendation H.264 - Advanced video coding for generic audiovisual services, January 2012.

[85] J. Glasmann and W. Kellerer and Harald Muller. Service Architectures in H.323 and SIP: A Comparison, 2003.

[86] Young Jiang, Wen Xu, and Grassmann Cyprian. Implementing a DVB-T/H Receiver on a Software-Defined Radio Platform. *International Journal of Digital Multimedia Broadcasting*, 2009:7, February 2009.

[87] Alan B Johnston. *SIP: understanding the Session Initiation Protocol, Third Edition*. Artech House, 2004.

[88] Laercio Crivnel Júnior. *Dynamic Support to Quality of Service in DiffServ-based IP Networks*. PhD thesis, UNIVERSIDADE TÉCNICA DE LISBOA, Instituto Superior Técnico, Departamento de Engenharia Inofrmática e de Computadores, Fevereiro 2011.

[89] A. Kern, I. Moldovan, and T. Cinkler. On the optimal configuration of metro ethernet for triple play. In *Next Generation Internet Design and Engineering, 2006. NGI '06. 2006 2nd Conference on*, pages 334–341, 2006.

[90] Y. Kikuchi, T. Nomura, S. Fukunaga, Y. Matsui, and H. Kimata. RFC 3016: RTP Payload Format for MPEG-4 Audio/Visual Streams. RFC 3016 (Proposed Standard), , Internet Engineering Task Force (IETF), November 2000.

[91] E. Kohler, M. Handly, and S. Floyd. RFC 4340: Datagram Cogestion Control Protocol (DCCP). RFC 4340 (Proposed Standard), March 2006.

[92] Jay Kraut. An examination of wireless media streaming in a an examination of wireless media streaming in a home network environment. Master's thesis, UNIVERSITY OF MANITOBA, Faculty of Graduate Strudies, Department of Eletrical and Computer Engineering, May 2005.

[93] Jun kyun Choi, Gyu Myoung Lee, Hyo-Jin Park, and Il Young Chong. Open iptv services over ngn. In *Optical Internet, 2008. COIN 2008. 7th International Conference on*, pages 1–6, 2008.

[94] T. D C Little and D. Venkatesh. Prospects for interactive video-on-demand. *Multi-Media, IEEE*, 1(3):14–25, 1994.

[95] Arlindo Maia da Silva. Prespectivas de Evolucao de VoIP na Internet. Master's thesis, Instituto Superior de Engenaria do Porto, Faculdade de Engenharia do Porto, Departamento de Engenharia Electrotecnica e de Computadores, September 2003.

[96] Meru 320i. `http://www.merunetworks.pl/zalaczniki/DS_AP320i_0410_v3.pdf`, 2013. Accessed: 2013-12-15.

[97] Meru MC1500. `http://www.merunetworks.com/collateral/data-sheets/2012-ds-mc1500-wireless-lan-controller.pdf.pdf`, 2013. Accessed: 2013-12-15.

[98] E. Mikoczy, D. Sivchenko, Bangnan Xu, and J.I. Moreno. IPTV Systems, Standards and Architectures: Part II - IPTV Services over IMS: Architecture and Standardization. *Communications Magazine, IEEE*, 46(5):128–135, 2008.

[99] S. Nosheen, S.A. Malik, Y. Bin Zikria, and M.K. Afzal. Performance Evaluation of DCCP and SCTP for MPEG4 Video over Wireless Networks. In *Multitopic Conference, 2007. INMIC 2007. IEEE International*, pages 1–6, 2007.

[100] J. Postel. RFC 768: User Datagram Protocol. RFC 768 (Standard), August 1980.

[101] J. Postel. RFC 793: Transmission Control Protocol. RFC 793 (Standard), September 1981. Updated by RFC 3168.

[102] Ulrich H. Reimers. DVB- The Family of International Standards for Digital Video Broadcasting. In *Proceedings of the IEEE*, volume 94, pages 173–182, January 2006.

[103] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. RFC 3261: SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), June 2002. Updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621,5626, 5630, 5922, 5954, 6026, 6141.

[104] Stephan. Rupp and Hoang Anh Tong. SIP-Based VoIP Service - Architecture and Comparison. *INFOTECH Seminal Advanced Comunication Services (ACS)*, pages 1–10, 2005.

[105] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RFC 3550: RTP: A transport protocol for real-time applications. RFC 3550, Internet Engineering Task Force (IETF), July 2003.

[106] H. Schulzrinne, A. Rao, and R. Lanphier. RFC 2326: RTSP: Real Time Streaming Protocol. RFC 2326 (Proposed Standard), April 1998.

[107] I. Shinji, D. Mikami, K. Masahito, L Katsuhiko, I. Hideo, N. Isami, and G. Yoshinori. Report on ITU-T FG IPTV Internation Standardization Activities. Technical Report Vol.6 No.8, NTT Technical Review, August 2008.

[108] W. Simpson and H. Greenfield. *IPTV and Internet Video: new markets in television broadcasting.* Focal Press : National Association of Broadcasters, Amsterdam ; Boston ; Washington, D.C., 2007.

[109] R. Stewart. RFC 4960: Stream Control Transmission Protocol. RFC 4960 (Proposed Standard), September 2007.

[110] A. Takahashi, D. Hands, and V. Barriac. Standardization activities in the ITU for a QoE assessment of IPTV. *IEEE Communications Magazine on*, 46(2):78–84, 2008.

[111] Akira Takahashi. Framework and Standardization of Quality of Experience (QoE) Design and Management for Audio Visual Communication Services. Technical Report Vol.7 No.4, NTT Technical Review, 2009.

[112] Michael Topic. *Streaming Media Demystified*. McGraw Hill, New York ; Chicago ; San Francisco ; Lisbon, 2002.

[113] TP-Link 300Mbps Wireless N USB Adapter. `http://www.tp-link.com/en/products/details/?model=TL-WN821N`, 2013. Accessed: 2013-12-15.

[114] TP-Link N600 Wireless Dual Band USB Adapter. `http://www.tp-link.com/lk/products/details/?model=TL-WDN3200`, 2013. Accessed: 2013-12-15.

[115] T. Uhl. Qos measurement aspects for triple play services. In *Ultra Modern Telecommunications Workshops, 2009. ICUMT '09. International Conference on*, pages 1–5, 2009.

[116] J. van der Meer, D. Mackie, V. Swaminathan, D. Singer, and P. Gentric. RFC 3640: RTP Payload Format for Transport of MPEG-4 Elementary Streams. RFC 3640 (Proposed Standard), Internet Engineering Task Force (IETF), November 2003.

[117] Juan Velasco. Video Quality Assessment. In Amal Punchihewa, editor, *Video Compression*, pages 129–154. in Tech, Spain, March 2012.

[118] MSU Video Quality Measurement Tool. `http://compression.ru/video/quality_measure/video_measurement_tool_en.html`, 2005-2013. Accessed: 2013-12-15.

[119] Wang, Bing and Kurose, Jim and Shenoy, Prashant and Towsley, Don. Multimedia streaming via TCP: An analytic performance study. *ACM Trans. Multimedia Comput. Commun. Appl.*, 4(2):16:1–16:22, may 2008.

[120] S. Winkler and P. Mohandas. The evolution of video quality measurement: From psnr to hybrid metrics. *Broadcasting, IEEE Transactions on*, 54(3):660–668, 2008.

[121] Wireshark documentation. `http://www.wireshark.org/docs/`, 2013. Accessed: 2013-12-15.

[122] Thomas Zinner, Oliver Hohlfeld, Osama Abboud, and Tobias Hossfeld. Impact of Frame Rate and Resolution on Objective QoE Metrics. In *Quality of Multimedia Experience (QoMEx), 2010 Second International Workshop on*, pages 29–34, 2010.

A

Appendix

A.1 Video coding standards

Video consists of a stream of images (or frames) which are displayed in order and captured at regular time intervals (Frames Per Second (fps)). However, this raw format wastes too much bandwidth when transported over a shared network. The most successful way for solving this problem is achieved through the use of coding mechanisms that, besides compressing spatially the contents of each frame also obtain significant reductions of transmission data using temporal redundancies among different frames of the sequence. Spacial reduction reduces the size of the video data by selectively discarding unneeded parts of the original data in a frame. Temporal reduction, on the other hand, applies delta compression or motion compression in order to reduce the amount of data pixels needed to store a video frame by encoding only the pixels that change between consecutive frames in a sequence [88].

The implementation of a coding standard is a coder/decoder often addressed as *codec*, which is an acronym for compression/decompression. Another mechanism used by coders/decoders to reduce the bandwidth of transmitted video contents is the adaptive quantization of the signal. This process consists on the discretization of the analogue signal at the input of the encoder. The continuous signal is splitted in a number of discrete values according to a quantization scale. This process takes advantage of the fact that human vision has a different sensitivity at high and low spacial frequencies. Several important standards like the Motion Pictures Experts Group (MPEG) standard and H.264 are the most commonly used techniques for video compression.

Video encoders use different types of compression for different types of frames. There are three types of picture frames:

- I-Frames, or Intra pictures are coded as a stand-alone picture frame, and represent the original image compressed, exploiting the spatial redundancy.
- P-Frames, or Predicted pictures use the same techniques of I-frames but explore temporal redundancy by motion compensation. They use the previous I or P-frames as reference. They have a higher compression ratio than the I-frames.
- B-Frames, or Bidirectional pictures have the highest compression ratio, because they may use as references the last and the next I or P-frames.

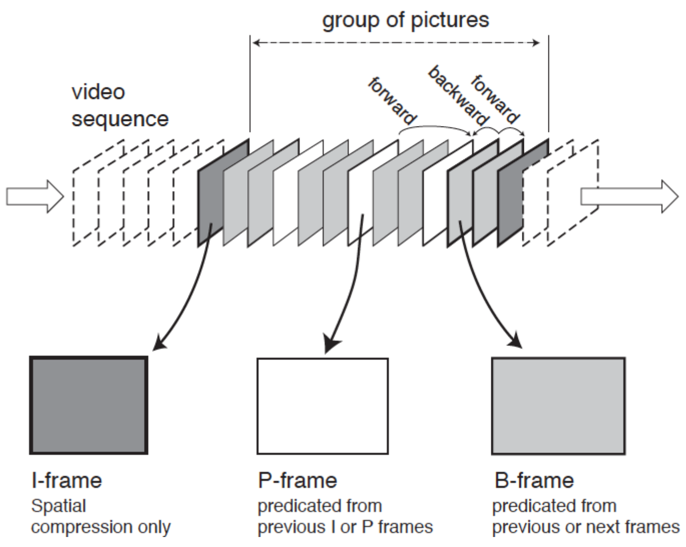


Figure A.1: GOP structure and dependencies between I,P and B frames [46].

The Group Of Pictures (GOP) is a group possibly composed by I,P and B-frames, shown in Figure A.1, that is self contained, i.e. it does not depend of the other external frames. It is easy to realize the importance of each type of video frame: failures to deliver I-Frames are the most severe because all the information of the GOP is lost, while a lost

P-Frame will compromise only all subsequent frames of the GOP. Losses of B-Frames have the least impact on the final result because, no other frames uses a B-Frame as a reference (excepts for MPEG4, where two groups of B-Frames can be defined). Since video playback must respect a specific frame rate, the receiver's side must provide some error concealment mechanism, to avoid having gaps in the frame sequence and excessive different regions in each frame.

Figure A.2 illustrates the evolution of MPEG and International Telecommunications Union (ITU) video coding standards. It shows that the two major standards were a combined effort of both groups. Three key formats for video compression will be considered in the scope of this thesis: MPEG-2, MPEG-4 and H.264.

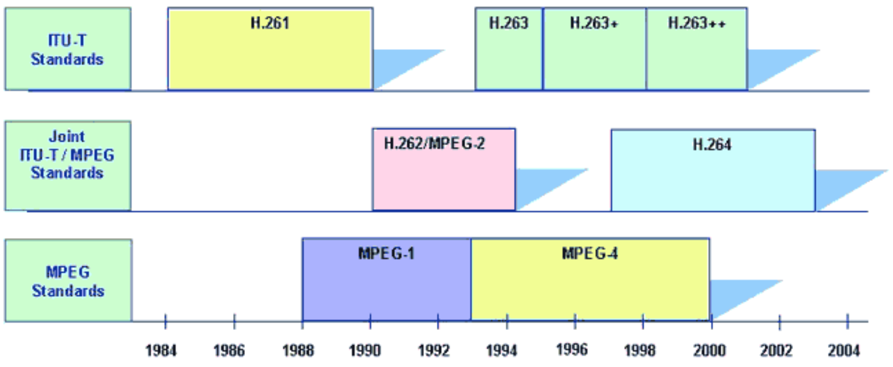


Figure A.2: Evolution of video coding standards [29].

MPEG-2

The MPEG-2 video standard is one of the most widely used standard for both, digital cable TV and satellite TV. The Elementary Streams (ES) is usually defined as the output of an audio or/video encoder. The MPEG-2 uses the specific model's ES, which consists on video audio and data components encoded. Also other information such as time stamps and network parameters can be added to ES. To send a ES through a possible lossy network, MPEG-2 uses a Transport Stream (TS). TS is formed by the process of multiplexing Packetized Elementary Streams (PES), which consists on the packetization of ES by spliting the data into chunks and adding a packet header [58]. One of the key fields of the header is the Packet Identifier (PID). The PID is the unique address identifier that is provisioned by the MPEG multiplexing equipment and is the field that helps the receiver to identify the specific packets. One of the major advantages of a PES over an ES is that it enables both random access and error protection. During the formation of the TS, a set of database tables describing the structure of the TS are added. This tables are called Program Specific Information (PSI) tables and there are various types of PSI tables: Program Association Table (PAT), Program Map Table (PMT), Conditional Access Table (CAT) and Network Information Table (NIT) each one with a specific job. The PAT is responsible for describing all the PMT's in the TS; The PMT provides the information on each program present in the TS, including the program number and a list of ES that composes the program; The CAT describes the Conditional Access (CA) systems that are in use in the transport stream, and provides information about how to decode them; Finally the NIT describes how transport streams are organized on the current network, and also describes some of the physical properties of the network [65] [3]. PSI tables consists of a normative data which are required for demultiplexing of the TS and the successful regeneration of the programs [65]. Figure A.3 indicates the sequence of PSI table decoding required to assemble and decompress the contents of a program.

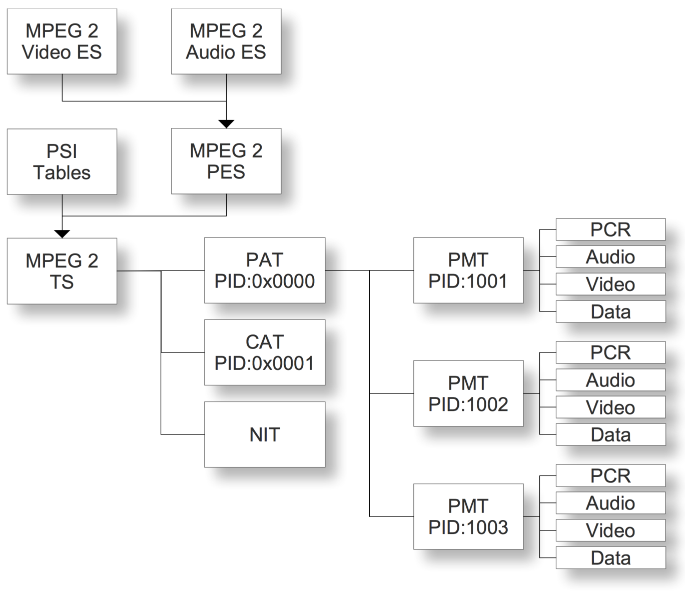


Figure A.3: MPEG-2 transport stream structure with PSI table sequence (Based on [3]).

Diagram A.4 illustrates a simplified block diagram of how a RTP stream is constructed using a TS[63].

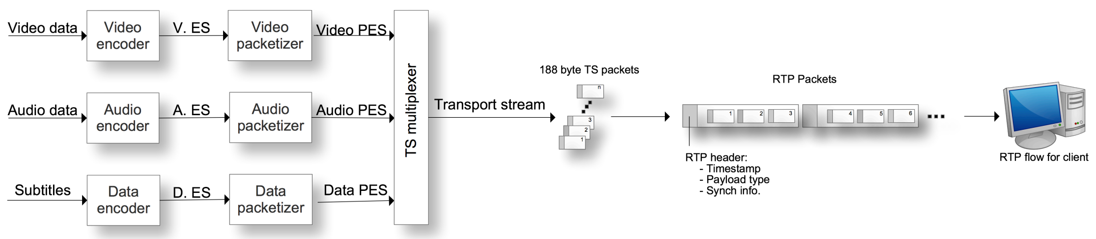


Figure A.4: Construction and encapsulation of MPEG-2 TS by RTP (Based on [88]).

MPEG-4

MPEG-4 is a group of audio and video coding standards for storage and delivery of digital multimedia content [5]. The initial goal of MPEG-4 was to create a standard that supports lower bit rates, although this was changed into a standard covering high compression ratios for both low and high bitrates. The MPEG-4 standard is the successor to the MPEG-2, extending its application to distribution over Internet Protocol (IP) networks and providing features for interaction. MPEG-4 specifies two different video encoding standards: MPEG-4 Part 2 and MPEG-4 Part 10 (known as H.264/AVC). From now on MPEG-4 format will be referred to as MPEG-4 part 2.

The MPEG-4 is a high performance video codec with scalability, error resilience and efficiency features [57], that uses an object-video coding, in opposition to the frame-based approach from the previous standard. This way, MPEG-4 provides higher compression ratios than the MPEG-2 for the same resulting video quality [59]. MPEG-4 media objects can be represented individually, allowing the composing of scenes with multiple objects; For example, foreground and background objects can be coded differently to each other. Text, graphics, and synthetic contents are other standardised media objects that are coded as descriptive elements and can be handled in an audiovisual scheme [57].

The transportation of media objects is done by ES that are generated by an encoder, which adds to each stream descriptors for synchronization and optionally, hints to the QoS requirements. Finally these streams are multiplexed and sent for transmission, for instance to a MPEG-4 TS, to RTP or directly to UDP [116] [90] (figure A.5 shows the encapsulation process).

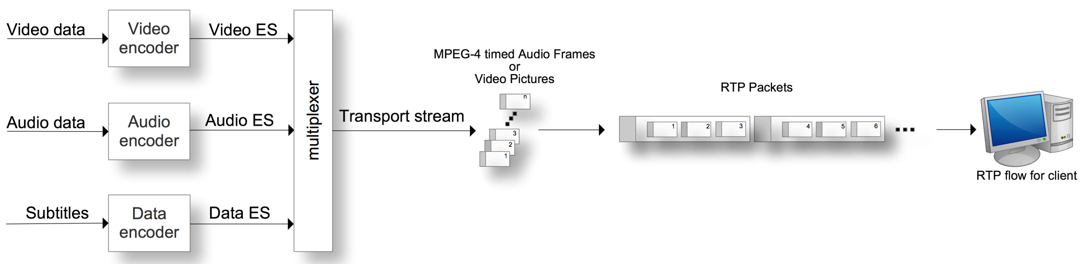


Figure A.5: Construction and encapsulation of MPEG-4 by RTP (Based on [88]).

H.264/AVC

The H.264/AVC video coding standard was jointly developed by the International Telecommunications Union Telecommunications Standardization Sector (ITU-T) and the MPEG group, approved as recommendation H.264/MPEG-4 AVC (or MPEG4 part 10) and provides significantly better compression efficiency compared to the previous standards[84] (figure A.6 ilustrates a more detailed comparison between MPEG-2 and H.264).

The H.264/AVC standard was designed for high compression efficiency, error resilience and flexibility, in order to support a wide variety of applications and different transport environments (such as wired and wireless networks). The H.264/AVC provides higher compression ratios at lower storage coasts, which leads to a reduction of the network bandwidth requirements. This however comes at a price: the processing power requirements for decoding H.264/AVC content are much higher than for MPEG-2. To achieve the flexibility mentioned before, the standard was designed to contain two coding layers [84]:

- Video Coding Layer (VCL) - Represents the core video encoding process (which carries the actual video compression) and the VCL data that consists of coded bits.
- Network Abstraction Layer (NAL) - handles the transportation of VCL data and other header information by encapsulating them in NAL units (packets), allowing efficient customization for a large number of systems.

The separation of video coding into two layers ensures that the VCL provides an efficient representation of video content, while the NAL transports the coded data and

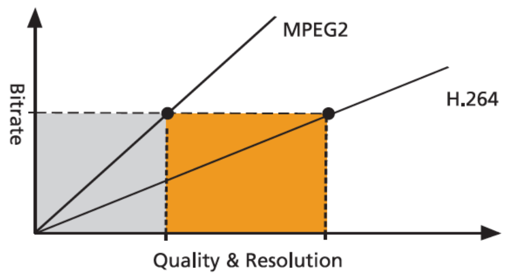


Figure A.6: Difference in compression rates between MPEG2 and H264 [4].

other header information in a flexible manner by adapting it to a variety of delivery frameworks.

The NAL layer is responsible for formatting the data and providing a heading information, to adequate the encoding for transmission. The contents of the layer VCL are mapped into NAL units and then carried to the transport layer. These units can be transmitted over a packet network, for example by RTP [105].

A.2 H.323

H.323 protocol specifies the components and protocols that provide multimedia communication services, real-time audio, video, and data communications over packet-based networks. The H.323 is part of a family of ITU-T recommendations called H.32x that is responsible for providing multimedia communication services over a variety of networks [80].

The H.323 standard specifies four types of components that together provide multimedia connection services as illustrated in figure A.7. They are: the Terminals, the Gateways, the Gatekeepers and the Multipoint Control Units (MCU) [80].

- Terminals: An H-323 Terminal can be a personal computer or a standalone device, running an H.323 stack and multimedia communications applications. Terminal support audio communications and can optionally support video or data communications. Terminals must support two types of protocols: H.245 for exchanging terminal capabilities and creation of media channels and H.225 for call signalling and call setup.

- Gatekeepers: The Gatekeeper is one of the most important components on H.323 system. It is the focal point for all calls within the H.323 network. It is responsible for providing several functions such as: address translation, admission control, bandwidth control and zone management.
- Gateways: An H-323 gateway provides connectivity between an H.323 network and a non-H.323 network, e.g. a gateway can connect and provide communication between an H.323 terminal and the Public Switched Telephone Networks (PSTN). This connectivity of dissimilar networks is achieved by translating protocols for call setup and release, converting media formats between different networks, and transferring information between networks connected by the gateway.
- Multipoint Control Units (MCU): An end point on the network that provides the capability for three or more terminals and gateways to participate in a multipoint conference. The MCU consists of a mandatory Multipoint Controller (MC) and optional Multipoint Processors (MP).

In resume, the terminals are devices used by end users. The gateways provide access to other networks (PSTN or other VoIP networks). The gatekeepers act as regulators of who can and can not talk on the network. A terminal requests permission to the gatekeeper before starting to communicate with another terminal. The MCU are used to allow several participants in a conference. The complexity of H.323 results in a large number of messages needed to be sent to establish a basic call. Networks can be segmented and the different segments be managed by different Gatekeepers. When multiple gatekeepers are involved in establishing calls, the messages can become very complex.

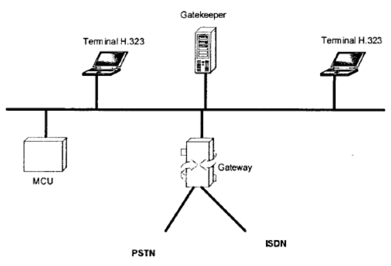


Figure A.7: H.323 network architecture [95].

B

Appendix

B.1 Capture and conversion

Figure B.1 shows a block diagram of a Digital Video Broadcasting - Terrestrial (DVB-T) receiver. As seen before the DVB-T receiver is where all the process starts and is composed by three main blocks: The digital tuner, the DVB-T demodulator and the USB controller. Each one of these blocks is then framed into simpler blocks.

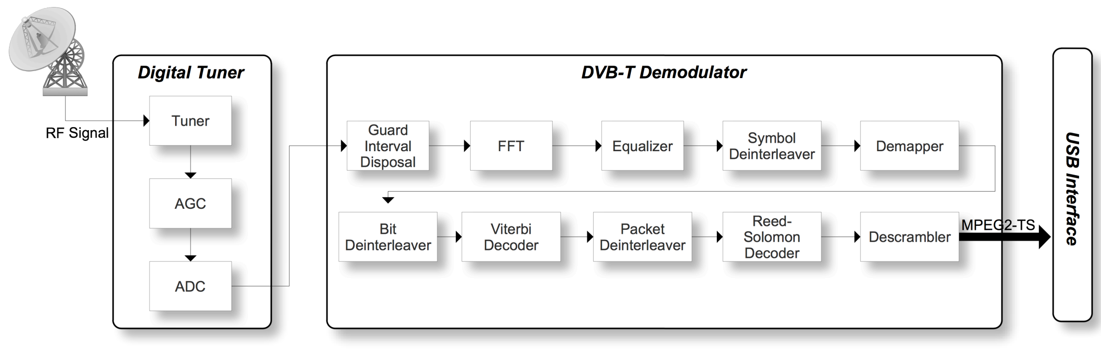


Figure B.1: Functional DVB-T receiver block diagram

The digital tuner includes a tuner, an Analogue to Digital Converter (ADC) and an Automatic Gain Controller (AGC). The tuner is responsible for amplifying the analogue

RF signal received through the antenna, converting and filtering it into a base-band signal. Afterwards, it is scaled by the AGC and then the base-band signal is ready for the digital conversion. The ADC converts the based-band signal into a digital signal in order to be sent to the DVB-T demodulator.

The DVB-T demodulator is an Orthogonal Frequency Division Multiplexing (OFDM) demodulator and is composed by: a guard interval disposal, a Fast Fourier Transform (FFT) unit, an equalizer, a symbol deinterleaver, a demapper, a bit deinterleaver, a Viterbi decoder or inner Forward Error Correction (FEC) decoder, a packet deinterleaver, a Reed-Solomon decoder or external FEC decoder and finally a descrambler [86] [48] [102]. Each of the units of the DVB-T demodulator has the following functions:

- The guard interval disposal is responsible for eliminating the Guard Interval (GI) and the Cyclic Prefix (CP).
- The FFT unit is responsible for the transformation of the time domain signal into a frequency domain signal.
- The equalizer can compensate the distortion caused by the amplification and transmission.
- The symbol deinterleaver can provide symbol-based deinterleaving on a block basis.
- The demapper can convert the symbol-deinterleaver data constituted by a complex number of vector's (QPSK, 16 QAM or 64 OAM) to a simple bit stream.
- The bit deinterleaver provides the bit-based deinterleaveing that is a bit-wise deinterleaving.
- The Viterbi decoder performs the Viterbi decoding to reverse the coding process that has been performed by the internal coder at the transmitter.
- The packet deinterleaver applies the packet-based deiterleaver that is a byte-wise deiterleaving within each packet.
- The Reed-Solomon decoder reverses the coding process that has been performed by the Reed-Solomon coder at the transmitter to correct the transmission error.

- the descrambler, descrambles the decoded data for removing the dispersal energy that has been added in the transmission and recovers the original serial bit stream as the final output of the DVB-T demodulator.

C

Appendix

C.1 LIRC

File: *usr/share/lirc/remotes/mecusb/lircd.conf.mceusb*.

```
begin remote
  name        mceusb
  bits                  16
  flags  RC6|CONST_LENGTH
  eps                   30
  aeps                 100
  header       2667   889
  one           444   444
  zero          444   444
  pre_data_bits         21
  pre_data         0x37FF0
  gap               105000
  toggle_bit            22
  rc6_mask     0x100000000
```

```
begin codes
  KEY_BLUE       0x00007ba1
  KEY_YELLOW       0x00007ba2
  KEY_GREEN       0x00007ba3
  KEY_RED           0x00007ba4
  KEY_BACK          0x00007bdc
  KEY_OK            0x00007bdd
  KEY_RIGHT         0x00007bde
  KEY_LEFT          0x00007bdf
  KEY_DOWN          0x00007be0
  KEY_UP            0x00007be1
  KEY_AGAIN         0x00007be4
  KEY_NEXT          0x00007be5
  KEY_STOP          0x00007be6
  KEY_PAUSE         0x00007be7
  KEY_RECORD        0x00007be8
  KEY_PLAY          0x00007be9
  KEY_REWIND        0x00007bea
  KEY_FORWARD       0x00007beb
  KEY_CHANNELDOWN   0x00007bec
  KEY_CHANNELUP     0x00007bed
  KEY_VOLUMEDOWN    0x00007bee
  KEY_VOLUMEUP      0x00007bef
  KEY_MUTE          0x00007bf1
  KEY_HOME          0x00007bf2
  KEY_POWER         0x00007bf3
  KEY_ENTER         0x00007bf4
  KEY_CLEAR         0x00007bf5
  KEY_9             0x00007bf6
  KEY_8             0x00007bf7
  KEY_7             0x00007bf8
```

```
        KEY_6                0x00007bf9
        KEY_5                0x00007bfa
        KEY_4                0x00007bfb
        KEY_3                0x00007bfc
        KEY_2                0x00007bfd
        KEY_1                0x00007bfe
        KEY_0                0x00007bff
      end codes
end remote
```

File: *lircrc*.

```
begin
prog = irexec
button = KEY_POWER
config = xset dpms force off ; xset dpms force on
end
begin
prog = irxevent
button = More
config = Key F6 CurrentWindow
end
begin
prog = irxevent
button = KEY_DOWN
config = Key Down CurrentWindow
end
begin
prog = irxevent
button = KEY_UP
config = Key Up CurrentWindow
end
```

```
begin
prog = irxevent
button = KEY_LEFT
config = Key Left CurrentWindow
end
begin
prog = irxevent
button = KEY_RIGHT
config = Key Right CurrentWindow
end
begin
prog = irxevent
button = KEY_OK
config = Key Return CurrentWindow
end
begin
prog = irxevent
button = KEY_BACK
config = Key Cancel CurrentWindow
end
begin
prog = irxevent
button = KEY_1
config = Key 1 CurrentWindow
end
begin
prog = irxevent
button = KEY_2
config = Key 2 CurrentWindow
end
begin
```

```
prog = irxevent

button = KEY_3

config = Key 3 CurrentWindow

end

begin

prog = irxevent

button = KEY_4

config = Key 4 CurrentWindow

end

begin

prog = irxevent

button = KEY_5

config = Key 5 CurrentWindow

end

begin

prog = irxevent

button = KEY_6

config = Key 6 CurrentWindow

end

begin

prog = irxevent

button = KEY_6

config = Key 6 CurrentWindow

end

begin

prog = irxevent

button = KEY_7

config = Key 7 CurrentWindow

end

begin

prog = irxevent
```

```
button = KEY_8

config = Key 8 CurrentWindow

end

begin

prog = irxevent

button = KEY_9

config = Key 9 CurrentWindow

end

begin

prog = irxevent

button = KEY_0

config = Key 0 CurrentWindow

end

begin

prog = irxevent

button = KEY_CLEAR

config = Key BackSpace CurrentWindow

end

begin

prog = irxevent

button = KEY_ENTER

config = Key Select CurrentWindow

end

begin

prog = irxevent

button = KEY_RED

config = Key F9 CurrentWindow

end

begin

prog = irxevent

button = KEY_GREEN
```

```
config = Key F10 CurrentWindow
end
begin
prog = irxevent
button = KEY_BLUE
config = Key F12 CurrentWindow
end
begin
prog = irxevent
button = KEY_VOLUMEUP
config = Key plus CurrentWindow
end
begin
prog = irxevent
button = KEY_VOLUMEDOWN
config = Key minus CurrentWindow
end
begin
prog = irxevent
button = KEY_PAUSE
config = Key Pause CurrentWindow
end
begin
prog = irxevent
button = KEY_PLAY
config = Key P CurrentWindow
end
begin
prog = irxevent
button = KEY_MUTE
config = Key Mute CurrentWindow
```

```
end

begin

prog = irxevent

button = KEY_STOP

config = Key s CurrentWindow

end

begin

prog = irxevent

button = KEY_REWIND

config = Key B CurrentWindow

end

begin

prog = irxevent

button = KEY_FORWARD

config = Key F CurrentWindow

end
```

C.2 User interface

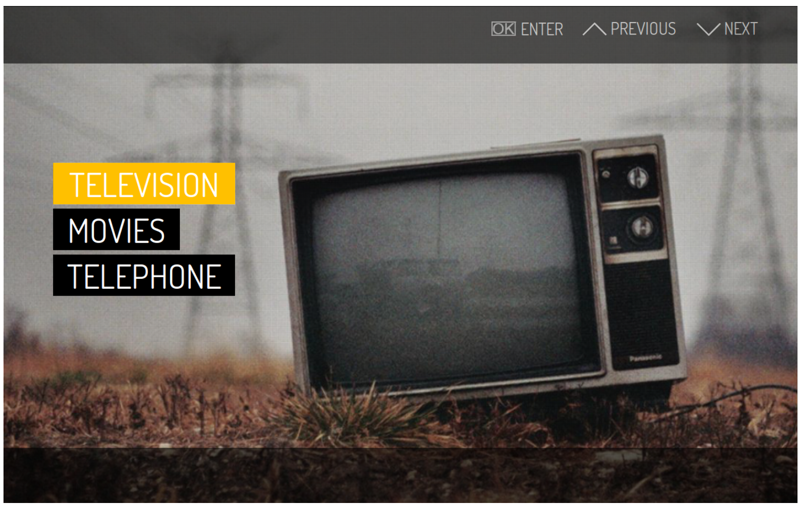


Figure C.1: User interface - IPTV.

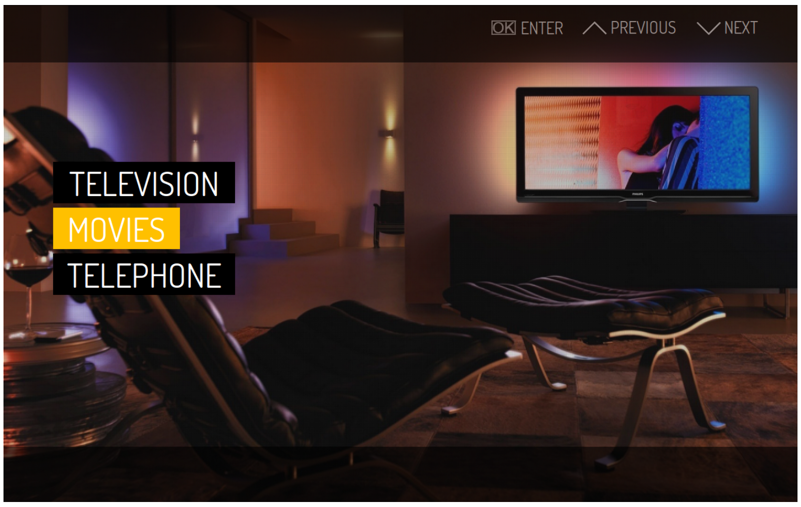


Figure C.2: User interface - VoD.

C.3 VLC plugin

The code below shows how to embed a VLC object into a HyperText Markup Language (HTML) page.

```
function WriteVLC(vlc){
var object = document.createElement('object');
object.version = 'VideoLAN.VLCPlugin.2';
```

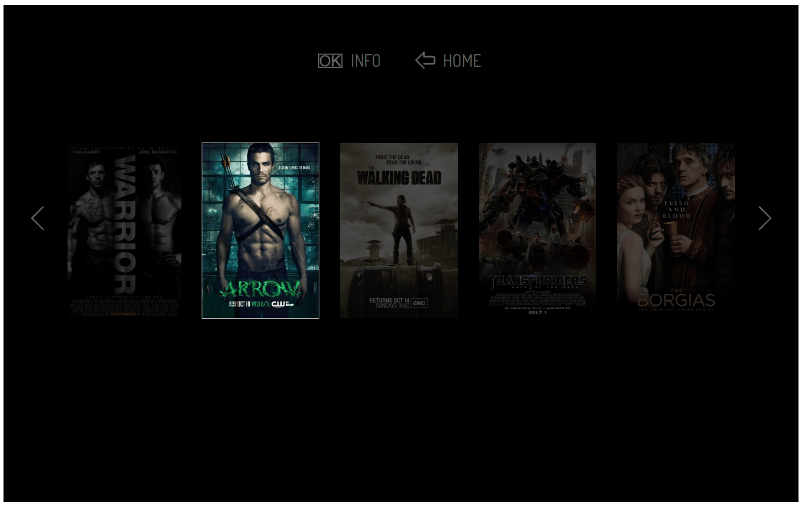
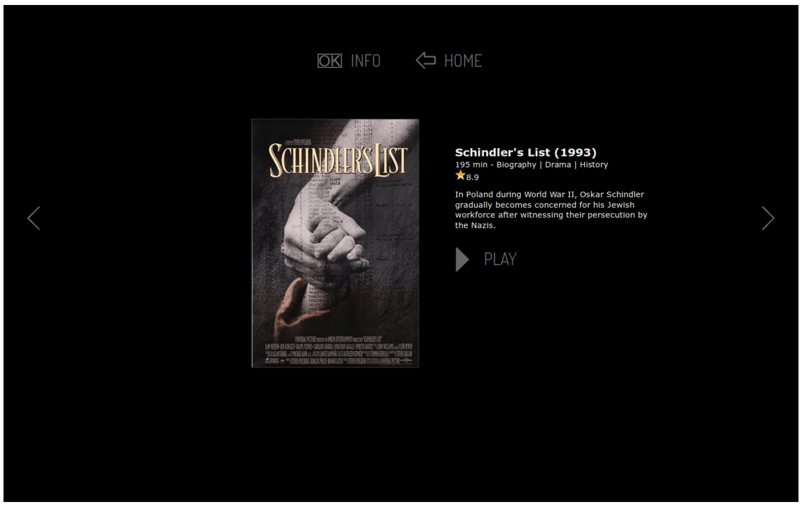


Figure C.3: User interface - thumbnail selection.



Schindler's List (1993)
195 min - Biography | Drama | History
★8.9

In Poland during World War II, Oskar Schindler gradually becomes concerned for his Jewish workforce after witnessing their persecution by the Nazis.

Figure C.4: User interface - movie description.

```
object.pluginspage = 'http://www.videolan.org';
object.type = 'application/x-vlc-plugin';
   object.id = vlc;
    object.name = vlc;
    object.width = screen.width;
    object.height = screen.height;
object.events = true;
object.tabindex = '2';
var param1 = document.createElement('param');
param1.name = 'toolbar';
```

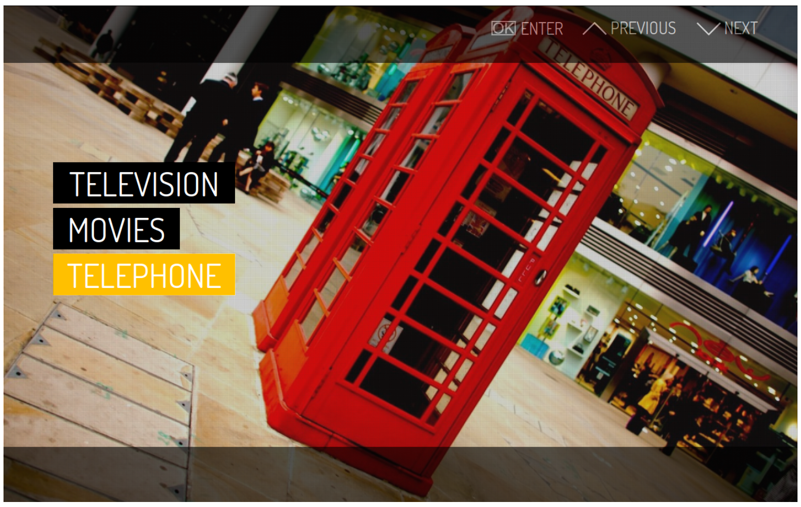


Figure C.5: User interface - VoIP.

```
param1.value = 'none';
object.appendChild(param1);
document.getElementById('screen').appendChild(object);
}
```

C.4 Remote Navigation

File: *navigation.js*.

```
document.onkeydown = function(e) {
    var x=0;
    if(window.event) {
        keynum = e.keyCode;
    } else if(e.which) {
        keynum = e.which;
    }else {
        keynum = e.charCode;
    }
    switch(keynum) {
        ...
        case 171://volume up
```

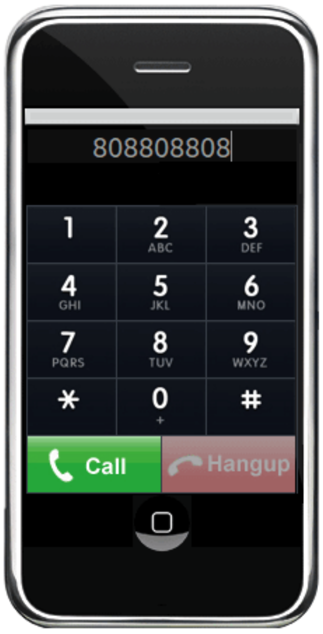
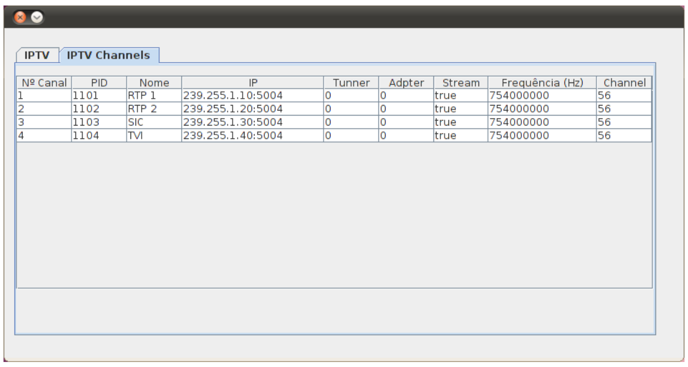


Figure C.6: User interface - VoIP client.

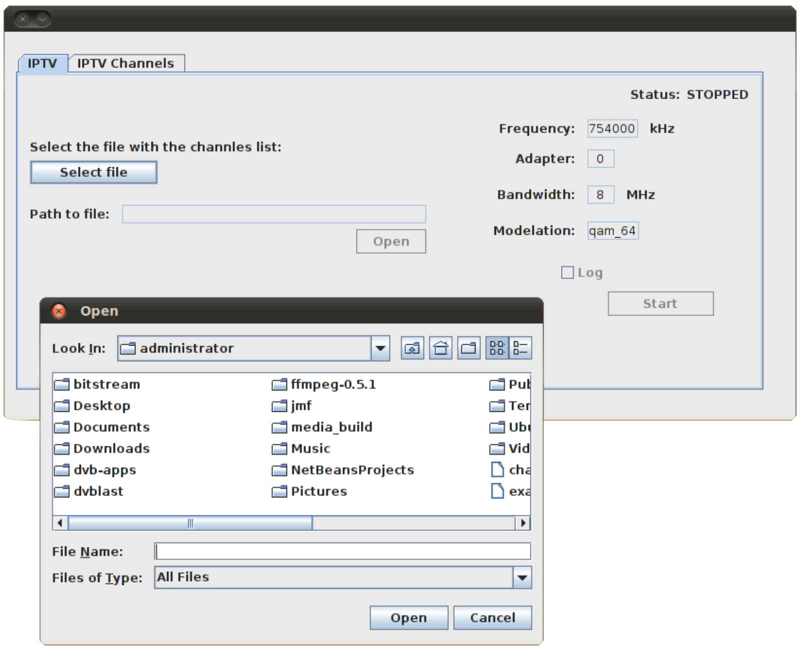
```
                vlc.audio.volume=vlc.audio.volume+10;
            break;
            case 173://volume down
                vlc.audio.volume=vlc.audio.volume-10;
            break;
            default:
            break;
        }
    };
```

C.5 IPTV module application



Nº Canal	PID	Nome	IP	Tunner	Adpter	Stream	Frequência (Hz)	Channel
1	1101	RTP 1	239.255.1.10:5004	0	0	true	754000000	56
2	1102	RTP 2	239.255.1.20:5004	0	0	true	754000000	56
3	1103	SIC	239.255.1.30:5004	0	0	true	754000000	56
4	1104	TVI	239.255.1.40:5004	0	0	true	754000000	56

Figure C.7: IPTV java application - list of channels tab



Status: STOPPED

Select the file with the channles list:

Path to file:

Frequency: 754000 kHz

Adapter: 0

Bandwidth: 8 MHz

Modelation: qam_64

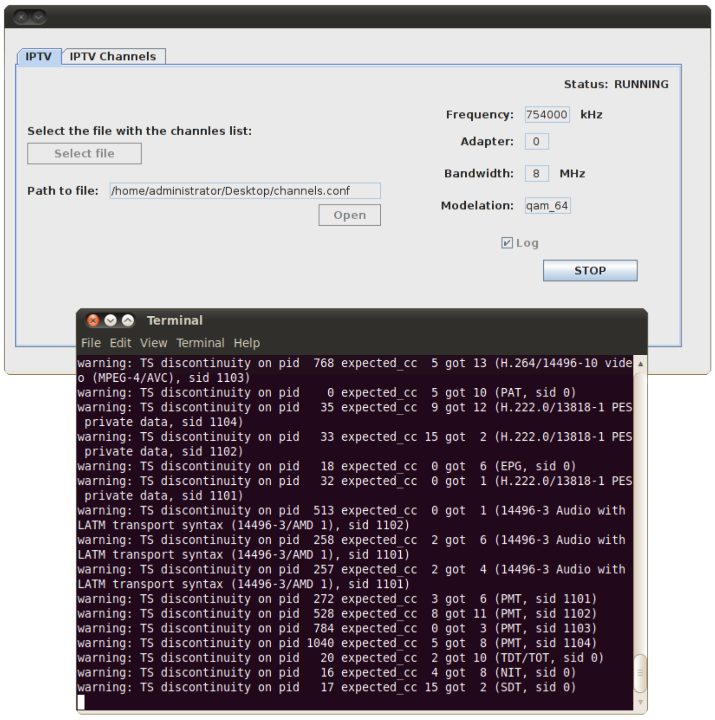
Look In: administrator

bitstream
Desktop
Documents
Downloads
dvb-apps
dvblast
ffmpeg-0.5.1
jmf
media_build
Music
NetBeansProjects
Pictures
Pul
Ter
Ubu
Vid
cha
exa

File Name:

Files of Type: All Files

Figure C.8: IPTV java application - channels.cfg selection



Status: RUNNING

Select the file with the channles list:

Frequency: 754000 kHz

Adapter: 0

Bandwidth: 8 MHz

Path to file: /home/administrator/Desktop/channels.conf

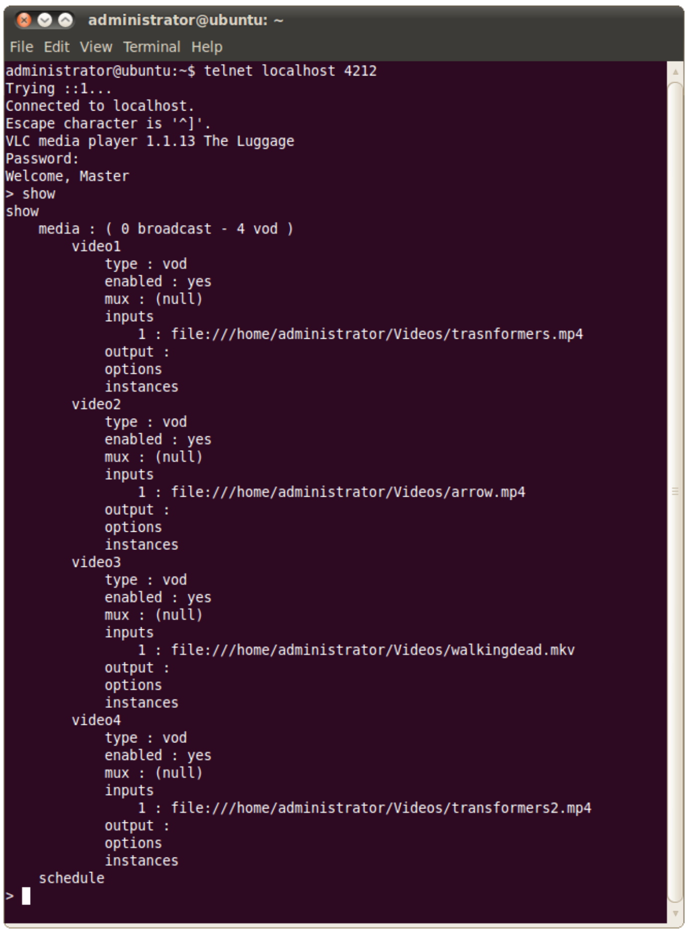
Modelation: qam_64

[✓] Log

```
warning: TS discontinuity on pid  768 expected_cc  5 got 13 (H.264/14496-10 vide
o (MPEG-4/AVC), sid 1103)
warning: TS discontinuity on pid    0 expected_cc  5 got 10 (PAT, sid 0)
warning: TS discontinuity on pid   35 expected_cc  9 got 12 (H.222.0/13818-1 PES
 private data, sid 1104)
warning: TS discontinuity on pid   33 expected_cc 15 got  2 (H.222.0/13818-1 PES
 private data, sid 1102)
warning: TS discontinuity on pid   18 expected_cc  0 got  6 (EPG, sid 0)
warning: TS discontinuity on pid   32 expected_cc  0 got  1 (H.222.0/13818-1 PES
 private data, sid 1101)
warning: TS discontinuity on pid  513 expected_cc  0 got  1 (14496-3 Audio with
LATM transport syntax (14496-3/AMD 1), sid 1102)
warning: TS discontinuity on pid  258 expected_cc  2 got  6 (14496-3 Audio with
LATM transport syntax (14496-3/AMD 1), sid 1101)
warning: TS discontinuity on pid  257 expected_cc  2 got  4 (14496-3 Audio with
LATM transport syntax (14496-3/AMD 1), sid 1101)
warning: TS discontinuity on pid  272 expected_cc  3 got  6 (PMT, sid 1101)
warning: TS discontinuity on pid  528 expected_cc  8 got 11 (PMT, sid 1102)
warning: TS discontinuity on pid  784 expected_cc  0 got  3 (PMT, sid 1103)
warning: TS discontinuity on pid 1040 expected_cc  5 got  8 (PMT, sid 1104)
warning: TS discontinuity on pid   20 expected_cc  2 got 10 (TDT/TOT, sid 0)
warning: TS discontinuity on pid   16 expected_cc  4 got  8 (NIT, sid 0)
warning: TS discontinuity on pid   17 expected_cc 15 got  2 (SDT, sid 0)
```

Figure C.9: IPTV java application - Log console selected

C.6 VoD module interfaces



```
administrator@ubuntu:~$ telnet localhost 4212
Trying ::1...
Connected to localhost.
Escape character is '^]'.
VLC media player 1.1.13 The Luggage
Password:
Welcome, Master
> show
show
    media : ( 0 broadcast - 4 vod )
        video1
            type : vod
            enabled : yes
            mux : (null)
            inputs
                1 : file:///home/administrator/Videos/trasnformers.mp4
            output :
            options
            instances
        video2
            type : vod
            enabled : yes
            mux : (null)
            inputs
                1 : file:///home/administrator/Videos/arrow.mp4
            output :
            options
            instances
        video3
            type : vod
            enabled : yes
            mux : (null)
            inputs
                1 : file:///home/administrator/Videos/walkingdead.mkv
            output :
            options
            instances
        video4
            type : vod
            enabled : yes
            mux : (null)
            inputs
                1 : file:///home/administrator/Videos/transformers2.mp4
            output :
            options
            instances
    schedule
> 
```

Figure C.10: VLM - Telnet interface

C.7 VoD module application

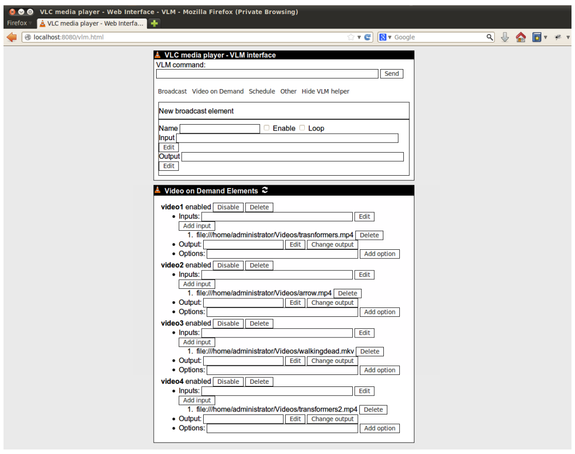
C.8 Asterisk installation and configuration

Asterisk

This is just a small gathering of commands and best practices for installing Asterisk and FreePBX on Ubuntu.

The following packages will be installed:

Asterisk 1.8



localhost:8080/vlm.html

VLC media player - VLM interface

VLM command:

New broadcast element

Name ☐ Enable ☐ Loop
Input
Output

Video on Demand Elements

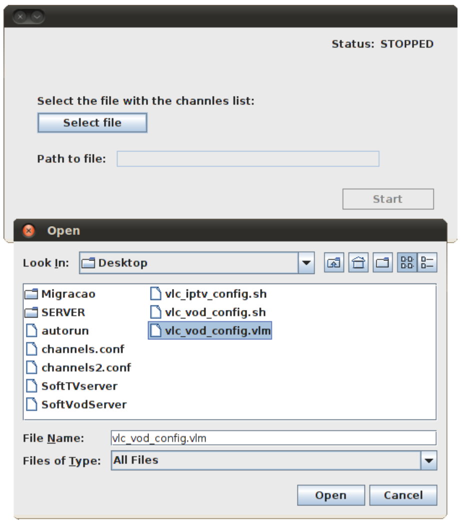
video1 enabled
- Inputs:
  1. file:///home/administrator/Videos/trasnformers.mp4
- Output:
- Options:

video2 enabled
- Inputs:
  1. file:///home/administrator/Videos/arrow.mp4
- Output:
- Options:

video3 enabled
- Inputs:
  1. file:///home/administrator/Videos/walkingdead.mkv
- Output:
- Options:

video4 enabled
- Inputs:
  1. file:///home/administrator/Videos/transformers2.mp4
- Output:
- Options:

Figure C.11: VLM - HTTP interface



Status: STOPPED

Select the file with the channles list:

Path to file:

Look In: Desktop

Migracao
SERVER
autorun
channels.conf
channels2.conf
SoftTVserver
SoftVodServer
vlc_iptv_config.sh
vlc_vod_config.sh
vlc_vod_config.vlm

File Name: vlc_vod_config.vlm
Files of Type: All Files

Figure C.12: VOD java application - vod_config.vlm selection

FreePBX 2.8

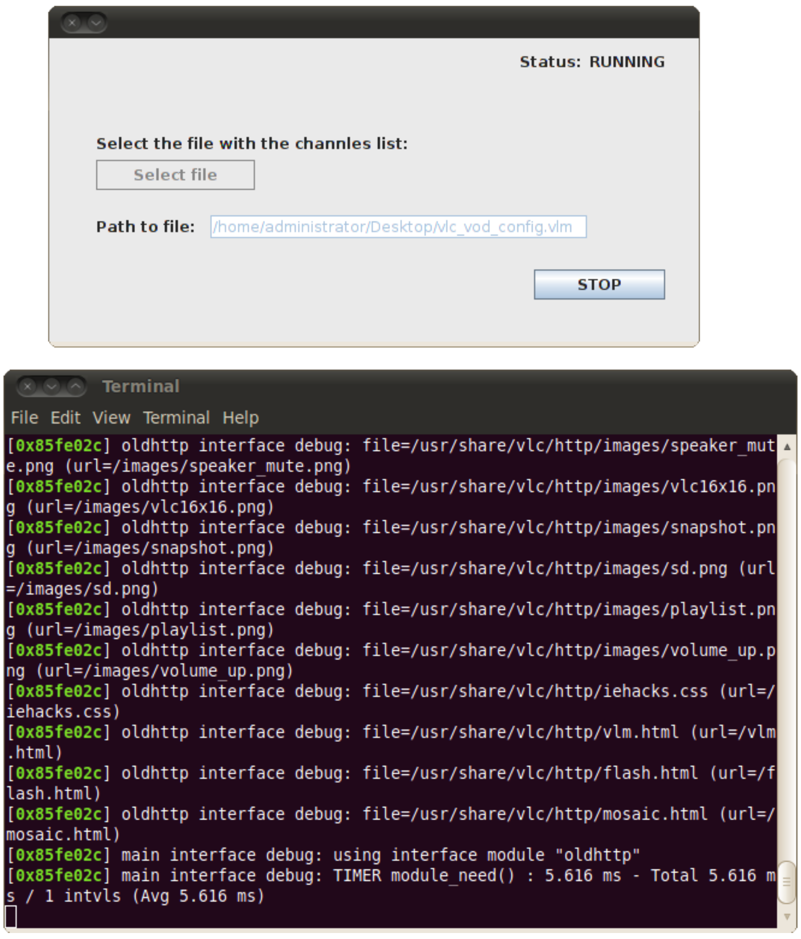
Update the system:

```
# aptitude update
```

Install necessary dependencies:

```
# aptitude install -y build-essential linux-headers-'uname -r' openssh-
server bison curl sox libncurses5-dev libssl-dev libmysqlclient15-dev
mpg123 libxml2-dev subversion
```

To install FreePBX need to install this packages also:



Status: RUNNING

Select the file with the channles list:

Path to file: /home/administrator/Desktop/vlc_vod_config.vlm

```
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/images/speaker_mut
e.png (url=/images/speaker_mute.png)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/images/vlc16x16.pn
g (url=/images/vlc16x16.png)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/images/snapshot.pn
g (url=/images/snapshot.png)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/images/sd.png (url
=/images/sd.png)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/images/playlist.pn
g (url=/images/playlist.png)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/images/volume_up.p
ng (url=/images/volume_up.png)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/iehacks.css (url=/
iehacks.css)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/vlm.html (url=/vlm
.html)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/flash.html (url=/f
lash.html)
[0x85fe02c] oldhttp interface debug: file=/usr/share/vlc/http/mosaic.html (url=/
mosaic.html)
[0x85fe02c] main interface debug: using interface module "oldhttp"
[0x85fe02c] main interface debug: TIMER module_need() : 5.616 ms - Total 5.616 m
s / 1 intvls (Avg 5.616 ms)
```

Figure C.13: IPTV java application - Log console

```
# aptitude install -y flex apache2 php5 php5-curl php5-cli php5-mysql
php-pear php-db php5-gd libapache2-mod-php5  mysql-server
```

Go to the directory */usr/src* and download source code packages:

```
# wget http://downloads.asterisk.org/pub/telephony/
dahdi-linux-complete/dahdi-linux-complete-2.6.0+2.6.0.tar.gz
# wget  http://downloads.asterisk.org/pub/telephony/asterisk/releases
/asterisk-1.8.5.0.tar.gz
```

Compile and install DAHDI and Asterisk:

```
# tar -zxvf dahdi-linux-complete/dahdi-linux-complete-2.6.0+2.6.0.tar.gz
# cd dahdi-linux-complete/dahdi-linux-complete-2.6.0+2.6.0
# make;make install;make config
# tar -zxvf asterisk-1.8.5.0.tar.gz
# cd asterisk-1.8.5.0
# contrib/scripts/get_mp3_source.sh
# ./configure
# make menuselect
```

Go to Add-ons:

```
select app_mysql, app_saycountpl, cdr_mysql, format_mp3, res_config_mysql
```

Go to Extras Sound Packages:

Select *EXTRAS-SOUNDS-EN-GSM*

Press *Esc* and *S* to save and exit.

```
# make;make install;make samples
```

PHP

PHP5 is installed just need to install some additional packages.

Edit the file */etc/php5/apache2/php.ini* and make the next changes in the settings:

```
max_execution_time = 30
memory_limit = 100M
error_reporting = E_COMPILE_ERROR|E_RECOVERABLE_ERROR|E_ERROR|E_CORE_ERROR
display_errors = Off
log_errors = On
error_log = /var/log/php.log
register_globals = Off
;PHP's built-in default is text/html
default_mimetype = "text/html"
;default_charset = "ISO-8859-1"
default_charset = "utf8";
magic_quotes_gpc = Off
# a2enmod php5
```

Apache

Modify the */etc/apache2/sites-available/default* like this:

```
<VirtualHost *:80>
    DocumentRoot /var/www/html
    <Directory />
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Order allow,deny
        allow from all
    </Directory>
    <Directory /var/www/html/admin>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Order allow,deny
        allow from all
    </Directory>
</VirtualHost>
```

Create an htpasswd file to protect the access to FREEPBX.

```
sudo htpasswd -c /etc/apache2/freepbx-passwd admin
```

Restart Apache.

```
sudo /etc/init.d/apache2 restart
```

FreePBX

Download the source code of FreePBX.

```
# cd /usr/src
# wget http://mirror.freepbx.org/freepbx-2.8.1.tar.gz
# tar -xvzf freepbx-2.8.1.tar.gz
# cd freepbx-2.8.1/
```

Create the database and the user used to access it, and populate the basic tables. This will create and import the basic tables to Asterisk and Asterisk CDR database, run this from the recently extracted directory.

```
mysqladmin create asterisk -u root -p
mysqladmin create asteriskcdrdb -u root -p
mysql -u root -p asterisk < SQL/newinstall.sql
mysql -u root -p asteriskcdrdb < SQL/cdr_mysql_table.sql
```

Create the user with privileges to access and edit the tables.

Open a mysql prompt with:

```
mysql -u root -p
```

On the prompt run the following queries:

```
GRANT ALL PRIVILEGES ON asterisk.* TO
asterisk@localhost IDENTIFIED BY 'amp109';
GRANT ALL PRIVILEGES ON asterisk.* TO
asterisk@127.0.0.1 IDENTIFIED BY 'amp109';
GRANT ALL PRIVILEGES ON asteriskcdrdb.* TO
asterisk@localhost IDENTIFIED BY 'amp109';
GRANT ALL PRIVILEGES ON asteriskcdrdb.* TO
asterisk@127.0.0.1 IDENTIFIED BY 'amp109';


flush privileges;
quit;
```

Before running the install command, make a backup of /etc/asterisk/modules.conf.

```
sudo cp /etc/asterisk/modules.conf ~/asterisk-modules.conf
```

Install FreePBX to */var/www/...*:

```
# ./safe_asterisk start
# ./install_amp
```

The install script will ask for some configuration information, such as were to install FreePBX (*/var/www/html*, sql password, asterisk password, etc.

The output from the install script is somewhat like this (with default settings):

```
...Enter your USERNAME to connect to the 'asterisk' database:
 [asteriskuser] asterisk
Enter your PASSWORD to connect to the 'asterisk' database:
 [amp109]
Enter the hostname of the 'asterisk' database:
 [localhost]
Enter a USERNAME to connect to the Asterisk Manager interface:
 [admin]
Enter a PASSWORD to connect to the Asterisk Manager interface:
 [amp111]
Enter the path to use for your AMP web root:
[/var/www/html]
Enter the IP ADDRESS or hostname used to access the AMP web-admin:
[xx.xx.xx.xx] your IP address
Enter a PASSWORD to perform call transfers with the
Flash Operator Panel:
 [passw0rd]
Use simple Extensions [extensions] admin or separate
Devices and Users [deviceanduser]?
 [extensions]
Enter directory in which to store AMP executable scripts:
 [/var/lib/asterisk/bin]...
```

Restore *asterisk-modules.conf* file, which have been backed up before installing FreePBX:

```
sudo cp ~/asterisk-modules.conf /etc/asterisk/modules.conf
```

Change permissions.

Add asterisk to asterisk group:

```
# adduser asterisk --disabled-password --no-create-home --
gecos "asterisk PBX user"


# adduser www-data asterisk
```

Change the permissions of a series of directories:

```
# chown -R asterisk:asterisk /var/run/asterisk
# chown -R asterisk:asterisk /etc/asterisk
# chown -R asterisk:asterisk /var/{lib,log,spool}/asterisk
# chown -R asterisk:asterisk /var/www/
```

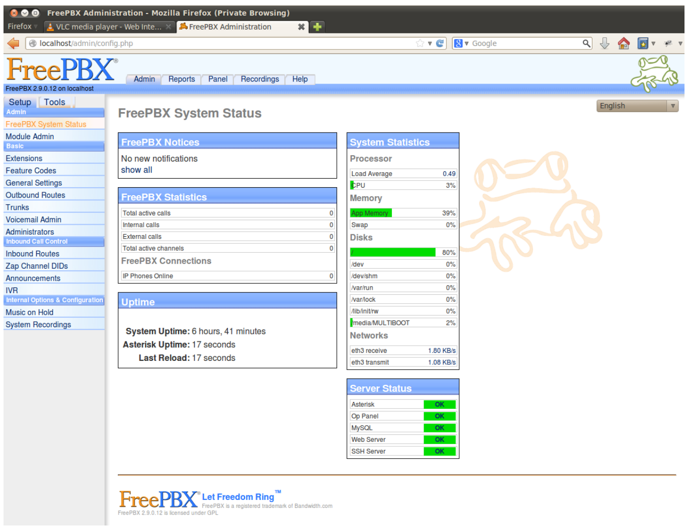
Fix permissions from amportal, add these lines to the end of file */etc/amportal.conf*:

```
AMPASTERISKUSER=asterisk
AMPASTERISKGROUP=asterisk
AMPASTERISKWEBUSER=asterisk
AMPASTERISKWEBGROUP=asterisk
```

Start amportal:

```
# amportal start
```

C.9 FreePBX interface



FreePBX System Status

FreePBX Notices

No new notifications
show all

FreePBX Statistics

Total active calls	0
Internal calls	0
External calls	0
Total active channels	0

FreePBX Connections

IP Phones Online	0

Uptime

System Uptime: 6 hours, 41 minutes
Asterisk Uptime: 17 seconds
Last Reload: 17 seconds

System Statistics

Processor

Load Average	0.49
CPU	3%

Memory

App Memory	39%
Swap	0%

Disks

/	80%
/dev	0%
/dev/shm	0%
/var/run	0%
/var/lock	0%
/lib/init/rw	0%
/media/MULTIBOOT	2%

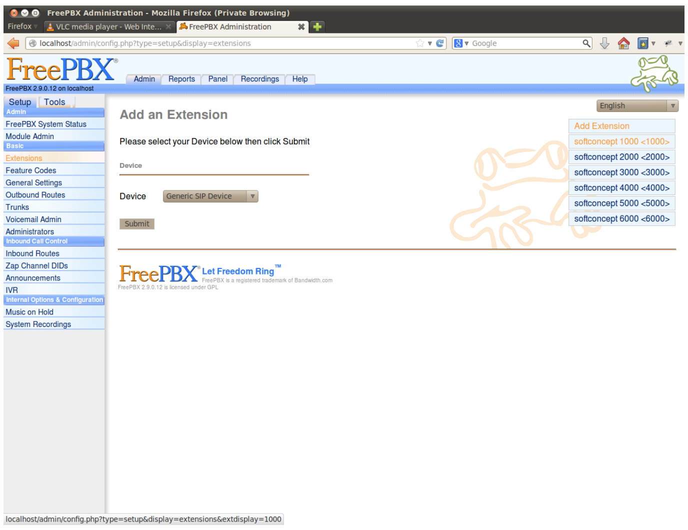
Networks

eth3 receive	1.80 KB/s
eth3 transmit	1.08 KB/s

Server Status

Asterisk	OK
Op Panel	OK
MySQL	OK
Web Server	OK
SSH Server	OK

Figure C.14: FreePBX interface - home page



Add an Extension

Please select your Device below then click Submit

Device

Device: Generic SIP Device

Figure C.15: FreePBX interface - extensions

C.10 Asterisk files

The first part of the *sip.conf* informs the system on the general settings to all users.

- Port: defines the port used by the SIP protocol
- Disallow: defines the codecs that will not be used
- Allow: defines the codecs that will be used
- CallerID: defines the default SIP register identity
- Context: defines the default content
- Bindadr: difines the IP address on which Asterisk is running

The second part defines the specific attributes for each of the SIP accounts.

- Type: defines the permissions to the user
- Username: defines the user name associated
- Secret: defines the password
- Host: client finder method
- Callerid: User identification which has to be attached to do a call

File: *usr/asterisk/sip.conf*.

```
[general]
port=5060
context=incoming
bindaddr=192.168.40.139
disallow=all
allow=all
callerid=Unknown
language=en
```

```
[1000]
deny=0.0.0.0/0.0.0.0
secret=123456
dtmfmode=rfc2833
canreinvite=no
host=dynamic
trustrpid=yes
sendrpid=no
type=friend
nat=no
context=internal
qualify=yes
qualifyfreq=60
callgroup=
pickupgroup=
dial=SIP/1000
mailbox=1000
permit=0.0.0.0/0.0.0.0
callerid=device <1000>
callcounter=yes
faxdetect=no

[2000]
deny=0.0.0.0/0.0.0.0
secret=111qqq
dtmfmode=rfc2833
canreinvite=no
host=dynamic
trustrpid=yes
sendrpid=no
type=friend
```

```
nat=no
context=internal
qualify=yes
qualifyfreq=60
callgroup=
pickupgroup=
dial=SIP/2000
mailbox=2000@device
permit=0.0.0.0/0.0.0.0
callerid=device <2000>
callcounter=yes
faxdetect=no
```

The syntax used to define an extension on *extensions.conf* is the following:

```
exten => <extension>,<priority>,<action>
```

- <extension>: Is the label of the extension.
- <priority>: Is a sequence number for each action of an extension. The first executable command of an extensions has the priority "1", so when Asterisk transfers a call to an extension, it will look for a command with priority "1". After executing the priority "1" command, Asterisk will change the priority to "2" and so on.
- <action>: Is the name of the command (action) to execute.

Some of the most common *action* commands that can be used are:

- Answer: This action accepts the call.
- Goto(*context*,*extension*,*priority*): This action sends the call to the specified *context*, *extension*, and *priority*.
- Dial(type/identifier,option,timeout): This action tells the Asterisk to ring the phone. When the call is answered, Asterisk bridges the call. *Type* specifies the channel type

and *identifier* specifies the "phone number" to dial on that channel. The *timeout* specifies a maximum time, in seconds, that the Dial command is to wait for a channel to answer. The *options* parameter, which is optional, is a string containing zero or more of the following flags and parameters:

  - t: Allows the called user to transfer the call by pressing a specific key.
  - T: Allows the calling user to transfer the call by pressing a specific key.

- Hang up(): Unconditionally hangs up the current call.

File: *usr/asterisk/extensions.conf*.

```
[incoming]

; Ring on extension 1000, 2000
exten => s,1,Answer()
exten => s,n,Dial(SIP/1000,150,r,t,)

; not answered? Pass unanswered calls to voicemail
exten => s,n,Voicemail(100,u)
exten => s,n,Hangup

[outgoing]

; Outbound calls can be routed based on the number of digits dialled
exten => _XXXXXXXXXXXXXXX,1,Dial(DAHDI/g0/${EXTEN:3})
exten => _XXXXXXXXX,1,Dial(DAHDI/g0/${EXTEN:3})

[internal]

; Calls between employees (between extensions)
exten => _XXX,1,Dial(SIP/${EXTEN})
```

```
; Calls to ext 1000

exten => 1000,1,Dial(SIP/1000,20)

exten => 1000,n,VoiceMail(1000,u)

exten => 1000,n,Hangup


; Calls to ext 2000

exten => 2000,1,Dial(SIP/1000,20)

exten => 2000,n,Hangup
```

D

Appendix

D.1 Testing results

Scenario 1

Network Metrics

Table D.1: Nerwork metrics without DATA traffic in scenario 1.

data = 0 Kbps					
AP Channel statistics					
N°. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interferance [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
1	-90	7 %/1 %	<1 %	218.6	13.9
2	-90	16 %/3 %	2 %	171.2	26.1
3	-90	26 %/4 %	2 %	187.9	38.6
4	-90	38 %/2 %	2 %	164	51.5

Clients (STB) statistics				
N°. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retrived [%]**	**Frame Droped [%]**
1	230.6	13.9	1 %	0 %
2	198.6	12.6	1 %	0 %
	148.5	13.5	<1	<1
3	219	12.4	1 %	1 %
	165.3	12.9	1 %	0 %
	222.4	13.3	2 %	0 %
4	174.5	13.1	<1 %	0 %
	224.4	12	2 %	1 %
	158.2	13	<1 %	0 %
	169.7	13.4	<1 %	0 %

Table D.2: Nerwork metrics for 200Kbps of DATA traffic in scenario 1.

data = 200 Kbps					
AP Channel statistics					
N°. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interferance [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
1	-90	7 %/2 %	1 %	222.5	13.7
2	-90	16 %/3 %	<1 %	175.6	26.3
3	-90	24 %/5 %	2 %	200.4	39.4
4	-90	38 %/6 %	2 %	176.6	54.6

Clients (STB) statistics				
N°. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retrived [%]**	**Frame Droped [%]**
1	225	13.7	<1 %	<1 %
2	209.1	12.6	<1 %	<1 %
	143.2	13.6	<1 %	<1 %
3	191.5	13.3	<1 %	<1 %
	145.3	12.7	1 %	<1 %
	200.4	13.4	<1 %	0 %
4	270.5	13.5	<1 %	<1 %
	178.3	13.8	<1 %	0 %
	223.2	13.6	<1 %	0 %
	162.7	13.7	<1 %	<1 %

PSNR Metric

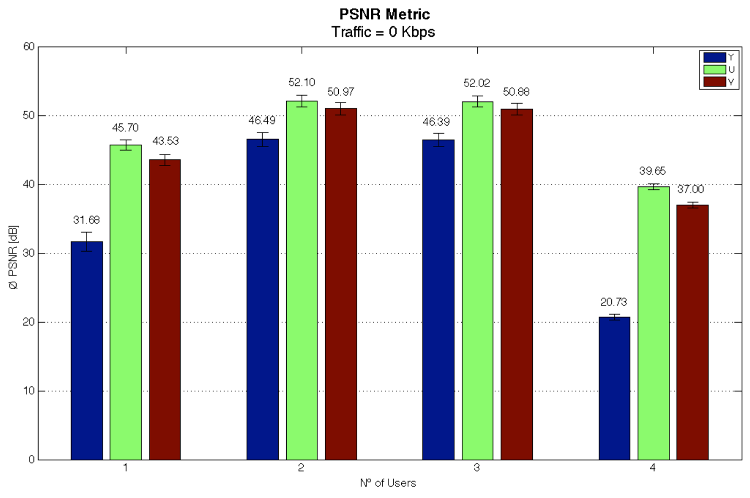


Figure D.1: PSNR Metric without DATA traffic in scenario 1.

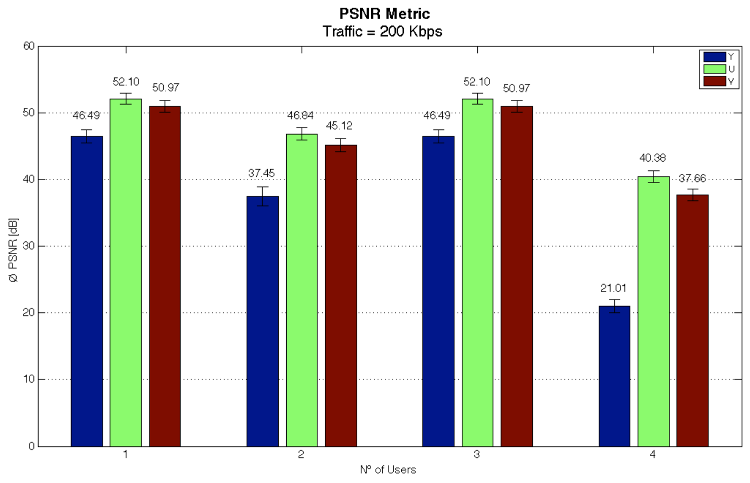


Figure D.2: PSNR Metric for 200Kbps of DATA traffic in scenario 1.

MSE Metric

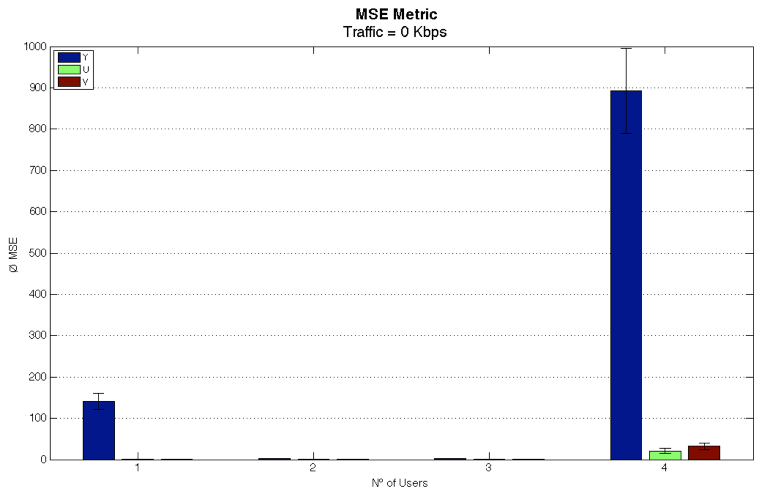


Figure D.3: MSE Metric without DATA traffic in scenario 1.

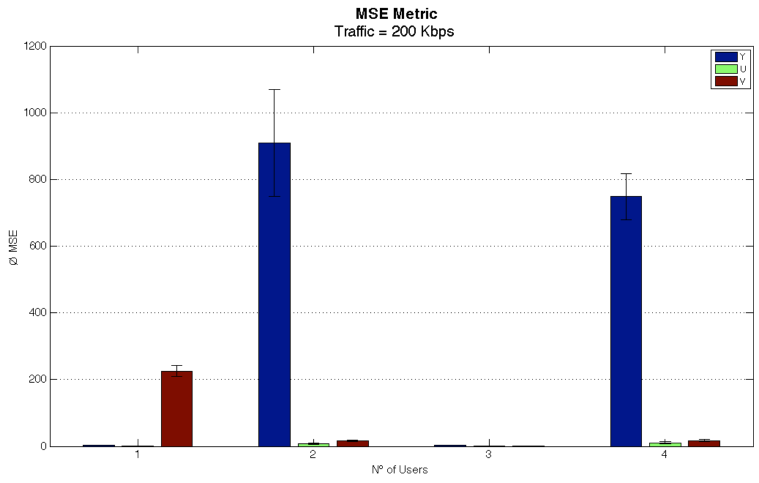


Figure D.4: MSE Metric for 200Kbps of DATA traffic in scenario 1.

VQM Metric

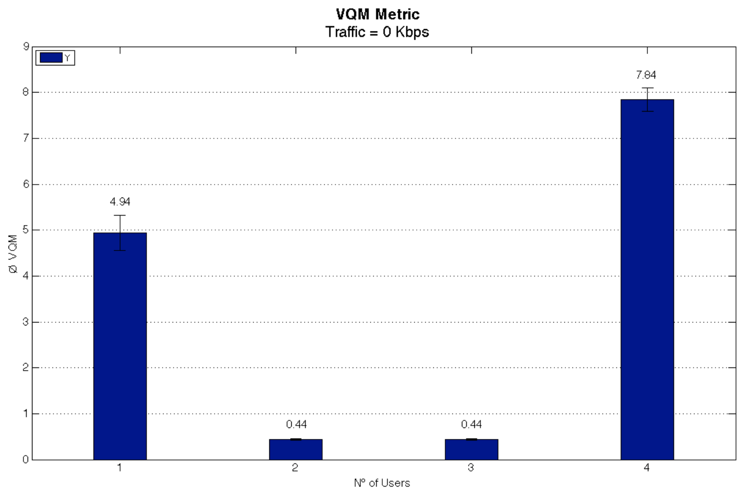


Figure D.5: VQM Metric without DATA traffic in scenario 1.

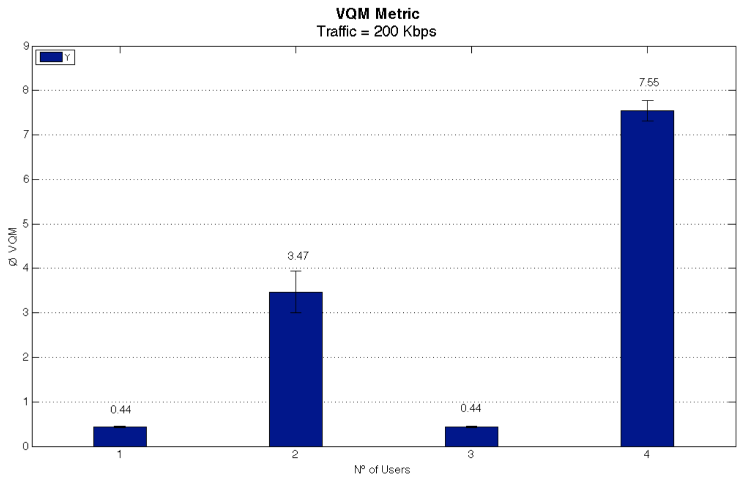


Figure D.6: VQM Metric for 200Kbps of DATA traffic in scenario 1.

Scenario 2

Network Metrics

Table D.3: Nerwork metrics without DATA traffic in scenario 2.

data = 0 Kbps					
AP Channel statistics					
N°. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interferance [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
3 VoD 1 SIP	-91	27 %/5 %	2 %	195.4	39.5
2 VoD 2 SIP	-90	19 %/3 %	1 %	200.5	26
1 VoD 3 SIP	-90	8 %/4 %	1 %	211.3	13.6

Clients (STB) statistics				
N°. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retrived [%]**	**Frame Droped [%]**
3 VoD 1 SIP	179.5	13.4	<1 %	<1 %
	118.1	12.9	3 %	<1 %
	166.5	13.1	6 %	<1 %
2 VoD 2 SIP	175.5	12.8	3 %	<1 %
	208.7	13.1	<1 %	<1 %
1 VoD 3 SIP	232.7	13.6	<1 %	<1 %

Table D.4: Nerwork metrics for 200Kbps of DATA traffic in scenario 2.

data = 200 Kbps					
AP Channel statistics					
N°. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interferance [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
3 VoD 1 SIP	-91	27 %/5 %	2 %	186.6	38.6
2 VoD 2 SIP	-90	19 %/5 %	1 %	189.4	27.1
1 VoD 3 SIP	-90	8 %/4 %	1 %	220.8	13.9

Clients (STB) statistics				
N°. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retrived [%]**	**Frame Droped [%]**
3 VoD 1 SIP	185.4	13.1	<1 %	<1 %
	211.6	12.2	3 %	<1 %
	189.3	13	4%	<1 %
2 VoD 2 SIP	191.2	13.4	<1 %	<1 %
	207.5	13.3	3 %	0 %
1 VoD 3 SIP	239.4	13.8	<1 %	<1 %

PSNR Metric

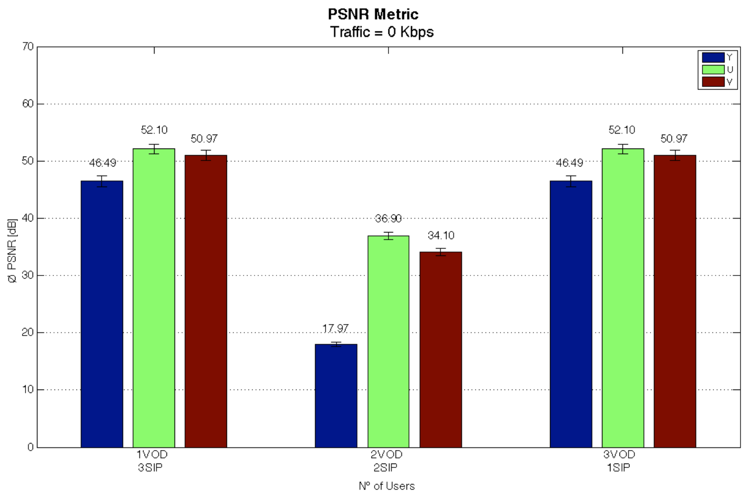


Figure D.7: PSNR Metric without DATA traffic in scenario 2.

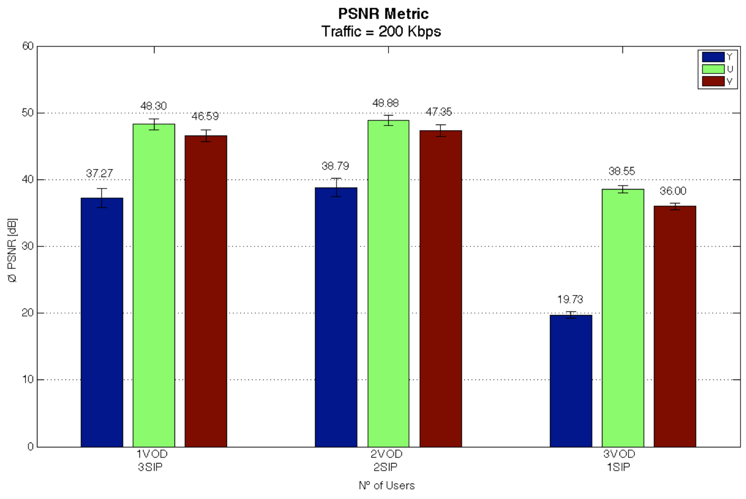


Figure D.8: PSNR Metric for 200Kbps of DATA traffic in scenario 2.

MSE Metric

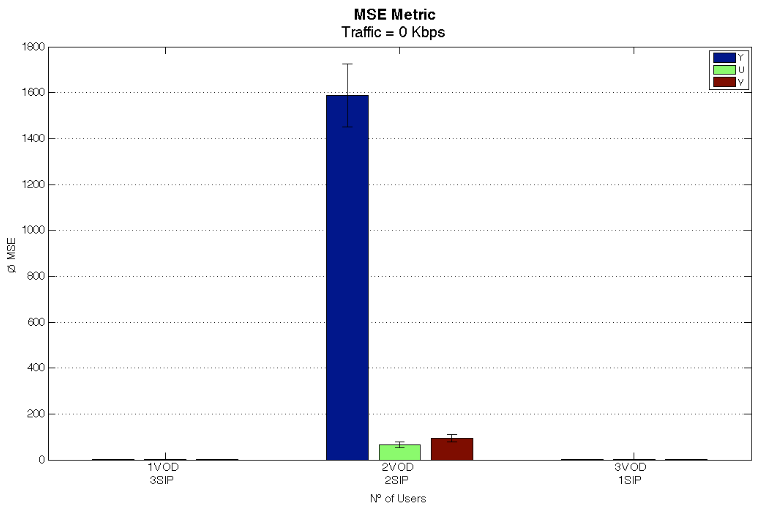


Figure D.9: MSE Metric without DATA traffic in scenario 2.

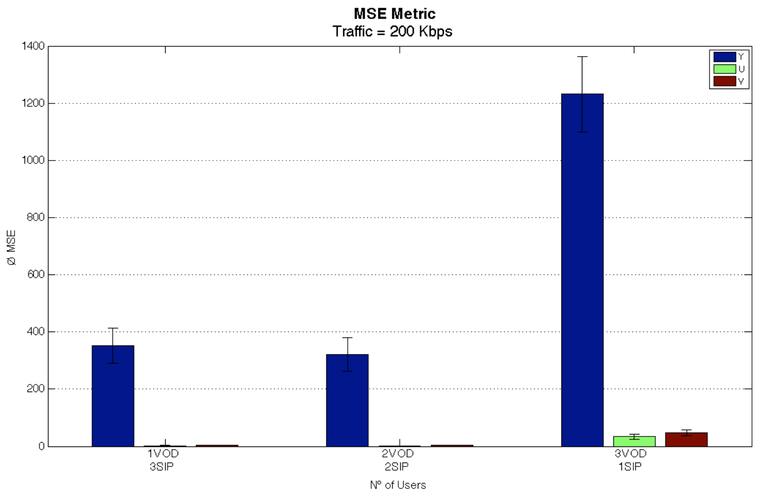


Figure D.10: MSE Metric for 200Kbps of DATA traffic in scenario 2.

VQM Metric

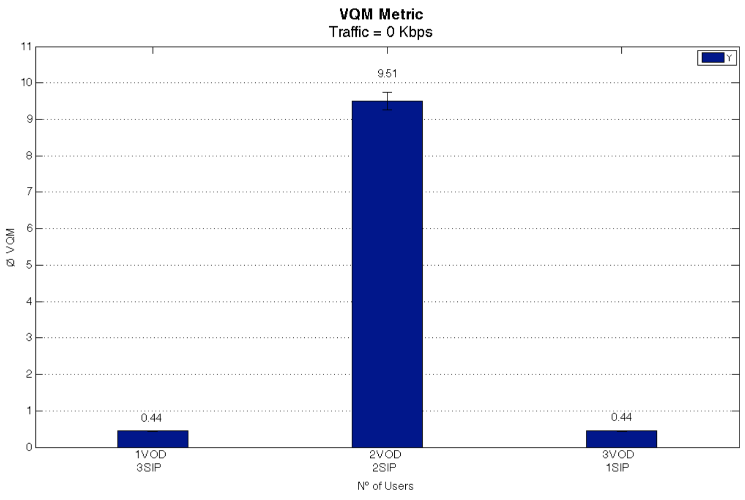


Figure D.11: VQM Metric without DATA traffic in scenario 2.

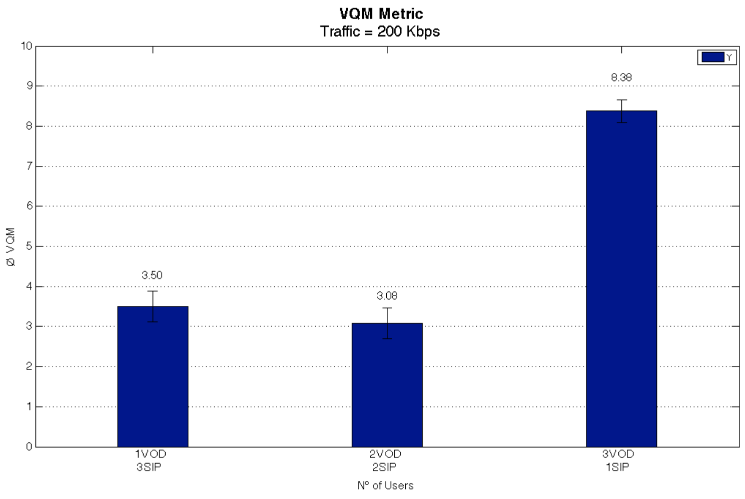


Figure D.12: VQM Metric for 200Kbps of DATA traffic in scenario 2.

Scenario 3

Network Metrics

Table D.5: Nerwork metrics without DATA traffic in scenario 3.

data = 0 Kbps					
AP Channel statistics					
N°. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interferance [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
1	-90	7 %/1 %	<1 %	214	13.8
2	-90	16 %/3 %	1 %	159.4	24.7
3	-90	25 %/4 %	1 %	193.5	39
4	-90	38 %/6 %	2 %	170.3	52.1

Clients (STB) statistics				
N°. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retrived [%]**	**Frame Droped [%]**
1	223.7	13.8	<1 %	0 %
2	213.2	11.8	<1 %	<1 %
	215.3	12.9	<1 %	<1 %
3	195.8	12.8	<1 %	<1 %
	206.8	12.7	1 %	<1 %
	132.6	13.5	<1 %	<1 %
4	150.1	13.2	<1 %	<1 %
	181.8	12.8	3 %	<1 %
	187	13.2	<1 %	0 %
	174.5	12.9	<1 %	<1 %

Table D.6: Nerwork metrics for 200Kbps of DATA traffic in scenario 3.

data = 200 Kbps					
AP Channel statistics					
N°. Clients	**Noise [dBm]**	**Utilization [TX/RX]**	**Interferance [%]**	**Capacity [Mbps]**	**Usage [Mbps]**
1	-90	7 %/1 %	1 %	216.6	13.6
2	-90	16 %/3 %	1 %	211.9	26.3
3	-91	26 %/5 %	2 %	173.2	40.7
4	-91	39 %/5 %	2 %	165.6	54.5

Clients (STB) statistics				
N°. Clients	**Capacity [Mbps]**	**Usage [Mbps]**	**Frame Retrived [%]**	**Frame Droped [%]**
1	217.4	13.6	<1 %	<1 %
2	215.2	12.6	<1 %	<1 %
	218.3	13.8	<1 %	0 %
3	181.8	13.8	<1 %	<1 %
	141.7	13.5	<1 %	2 %
	209.9	13.5	2 %	2 %
4	156.6	13.4	<1 %	<1 %
	164.4	13.3	3 %	<1 %
	263.4	14	<1 %	<1 %
	183.4	13.3	<1 %	<1 %

PSNR Metric

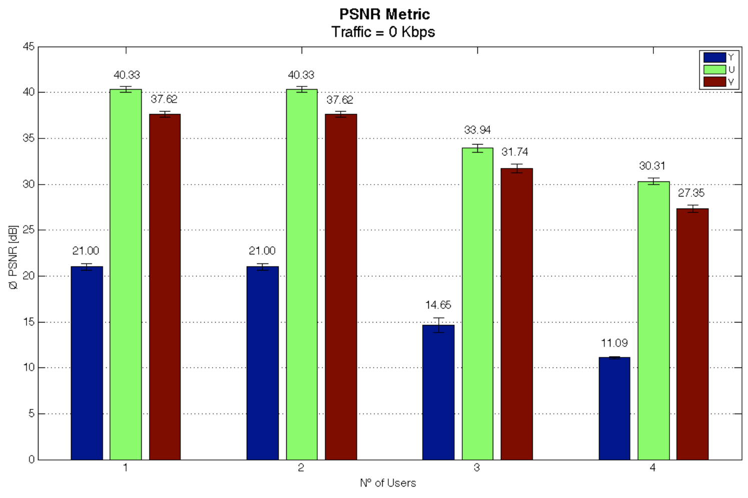


Figure D.13: PSNR Metric without DATA traffic in scenario 3.

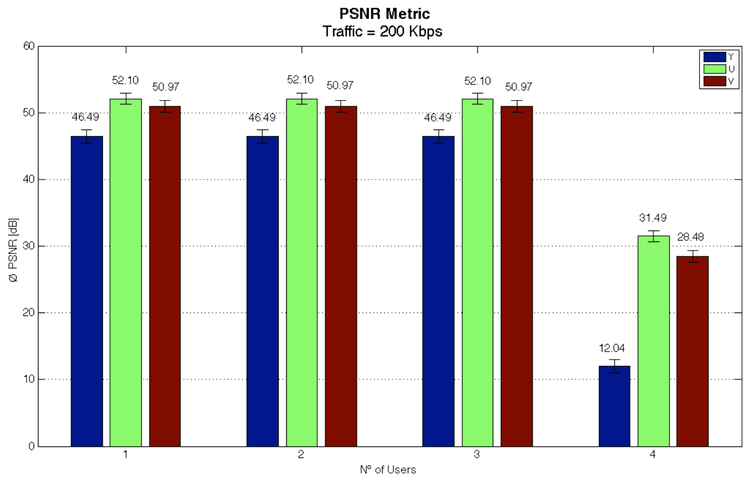


Figure D.14: PSNR Metric for 200Kbps ofDATA traffic in scenario 3.

MSE Metric

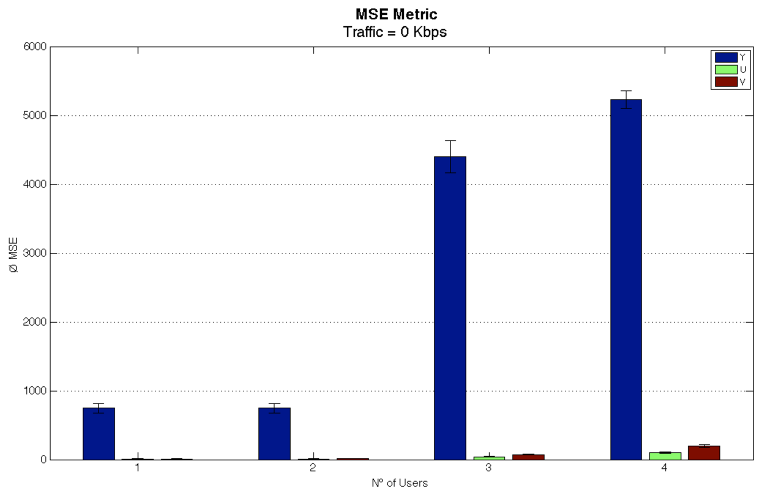


Figure D.15: MSE Metric without DATA traffic in scenario 3.

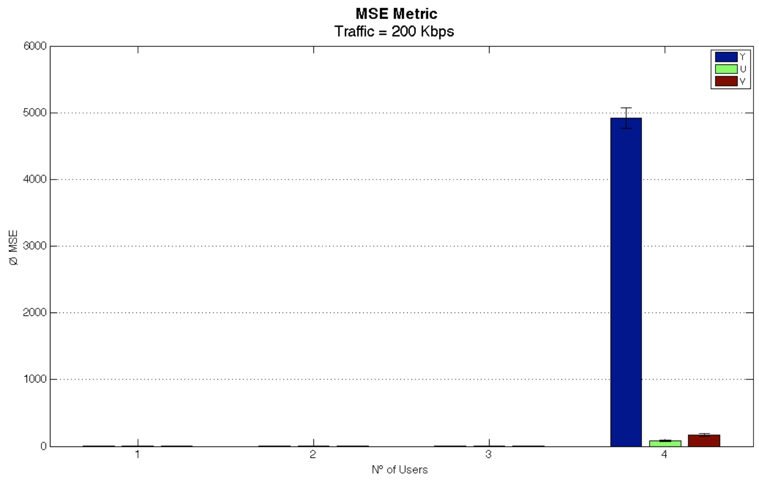


Figure D.16: MSE Metric for 200Kbps DATA traffic in scenario 3.

VQM Metric

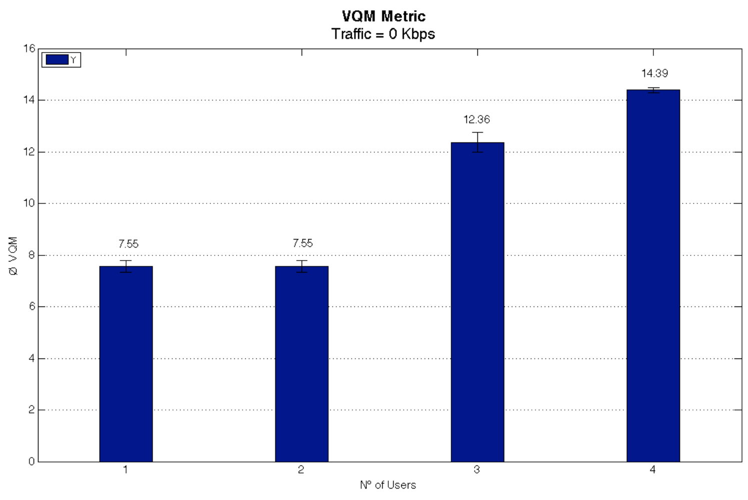


Figure D.17: VQM Metric without DATA traffic in scenario 3.

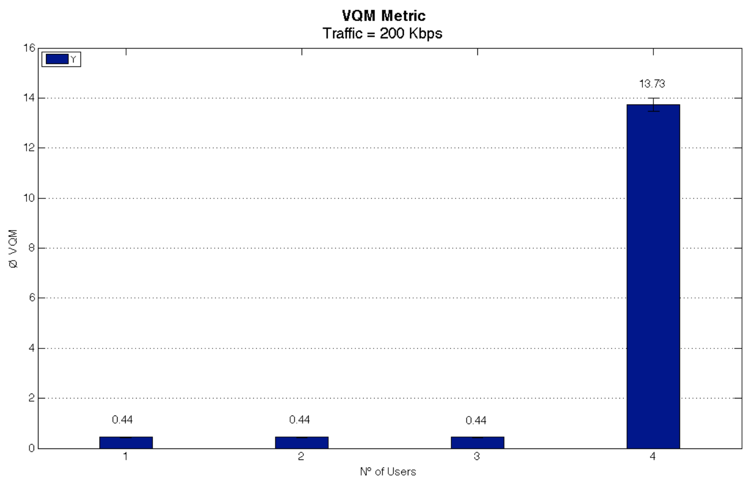


Figure D.18: VQM Metric for 200Kbps of DATA traffic in scenario 3.