UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA DE TELECOMUNICACIÓN
ESPECIALIDAD EN SISTEMAS Y REDES DE TELECOMUNICACIONES

PROYECTO FINAL DE CARRERA

Design and implementation of an on-line demonstrator for a video telephony system over heterogeneous networks

Gerardo Pinar Loriente

TUTOR: Dr.-Ing. José Ignacio Moreno Novella
DIRECTOR: Dr.-Ing. Błażej Lewcio
DIRECTOR: Dr.-Ing. Nico Bayer

Octubre de 2013

TÍTULO: *Design and implementation of an on-line demonstrator for a video telephony system over heterogeneous networks.*

AUTOR: *GERARDO PINAR LORIENTE*

TUTOR: *JOSÉ IGNACIO MORENO NOVELLA*

La defensa del presente Proyecto Fin de Carrera se realizó el día 25 de Octubre de 2013; siendo calificada por el siguiente tribunal:

PRESIDENTE: *DANIEL DÍAZ SÁNCHEZ*

SECRETARIO: *JULIO VILLENA ROMÁN*

VOCAL: *JUAN JOSÉ GARCÍA FERNÁNDEZ*

Habiendo obtenido la siguiente calificación:

CALIFICACIÓN:

Presidente **Secretario** **Vocal**

To the memory of my father.

To the memory of my grandmother, for whom I am an engineer since long ago.

To María, because I dedicate everything to her.

To my uncle Ángel, the most eager on me to finish.

To my family and friends.

To my mother.

* * *

A mi padre.

A mi abuela, que me dio el título de ingeniero hace ya mucho tiempo.

A María, porque todo se lo dedico a ella.

A mi tío Ángel, quien más ganas tenía de que terminara.

A mi familia y amigos.

A mi madre.

Acknowledgements

Thanks to José Ignacio, for the enormous support during the stay in Berlin as when preparing the document, always available and attentive. Thanks for your time and advices.

Thanks to Błażej, for your outstanding guide during my internship at T-Labs, for your help when approaching the thesis, for giving me the fantastic opportunity of working with you, for showing me how to do magic with a computer.

Thanks to Nico, for your help, your time and supervision.

Thanks to my father, for instilling in me the love for learning and showing me the taste of the joy which personal growth gives to us. For showing me how to see a problem from multiple diverse perspectives. For trusting me. Thanks to my mother, for everything: for your unconditional support, for being an example of struggle, of sacrifice, always keeping your sense of humor. Thanks for the army of saints conspiring for me to do well at every exam, thanks for making possible that I enjoyed so many experiences that made me grow. Thanks to my grandmother, because I can only remember you with a smile; I'm sorry for not having reached the end of this way one year before, although for you I had done it long time ago. Thanks to Rossana and Mari Tere, for appearing again in my life. Thanks to my family, for all the encouragement and support.

Thanks to Miguel Pagán, for showing me the way of self-improvement, sacrifice; the power of mind and spirit. Thanks for demonstrating me that there is always one more step to walk before falling exhausted. Thanks for educating me in a winning mentality. Thanks to Javi and my karate mates, with you I keep living this, with you I feel a privileged person.

Thanks to Julio, for showing me a way of life and an example of humility and mastery of life. Thanks to my aikido mates. With you a new and magical universe arised, a universe still expanding... just as Hara.

Thanks to Guillermo Ballenato, for the magnificent experience of attending your personal development courses, for your guide and example, for showing me that being happy is an imperative.

Thanks to José Manuel Ruiz. Thanks for showing me how to "pass the inspection test" over the problems, for teaching me to think as an engineer.

Thanks to my teachers David and Mari Paz. The knowledge you gave me have becomed blurred with time, but your example and wisdom are still as vivid in my mind as when I was seventeen.

Thanks to Jesusito, my brother-in-arms at ČVUT. I will never forget those anecdotes of university, which we so much like remembering in front of a beer.

Thanks to Víctor, Santi and Gabi, for that evening of itinerancy and eigenvalues.

Thanks to Álvaro and Carlos, for that effervescent February in the Southern Area.

Thanks to María, my guide, my motor, my partner.

Thanks to my classmates, my fellow sufferers. Thanks for all that laughing, support, energy, coffees... which made less hard laboratory and study hours. Thanks for teaching me so much.

Thanks to my friends from university, you have made of this a golden time.

Thanks to my friends in Erasmus, my Praguer brothers. Together, we fulfilled one of the best experiences that one can ever have.

Thanks to my mates at T-Labs, to my friends in Berlin. Thanks for those six magical months.

And thanks to my old friends, you, who are always there.

Thanks to all of you who have appeared in my way. Thanks to all of you who have walked with me along this journey, giving to me the best of you, and still do. I don't add more names, you know who you are.

THANKS, to you all.

Alcorcón, October 2013 Gerardo Pinar Loriente

Agradecimientos

Gracias a José Ignacio, por el enorme apoyo tanto durante la estancia en Berlín como de cara a la elaboración del documento, siempre disponible y atento. Gracias por tu tiempo y tus consejos.

Gracias a Błażej, por tu inmejorable guía durante mis prácticas en T-Labs, por tu ayuda al enfocar la tesis, por brindarme la fantástica oportunidad de trabajar contigo, por enseñarme cómo hacer magia con un ordenador.

Gracias a Nico, por tu ayuda, tu tiempo y supervisión.

Gracias a mi padre, por inculcarme el amor por el aprendizaje y mostrarme el sabor de la felicidad que regala el crecimiento personal. Por enseñarme a contemplar un problema desde múltiples perspectivas diferentes. Por confiar en mí. Gracias a mi madre, por todo: por tu apoyo incondicional, por ser un ejemplo de lucha, de sacrificio, siempre con sentido del humor. Gracias por tu ejército de santos conspirando para que me saliera bien cada examen, gracias por hacer posible que pudiera disfrutar de tantas experiencias que me han permitido crecer. Gracias a mi abuela, porque sólo puedo recordarte sonriendo; siento no haber llegado a la meta un año antes, aunque para ti ya había llegado hace mucho tiempo. Gracias a Rossana y a Mari Tere, por haber reaparecido en mi vida. Gracias a mi familia, por todo el ánimo y apoyo.

Gracias a Miguel Pagán, por mostrarme el camino de la autosuperación, del sacrificio; el poder de la mente y del espíritu. Gracias por demostrarme que no hay límites, que siempre se puede dar un paso más antes de caer exhausto. Gracias por educarme en una mentalidad ganadora. Gracias a Javi y a mis compañeros de karate, con vosotros sigo viviendo todo esto, con vosotros me siento un privilegiado.

Gracias a Julio, por enseñarme una forma de vida y un ejemplo de humildad y maestría de la vida. Gracias a mis compañeros de aikido. Con vosotros se abrió un universo nuevo y mágico, un universo que sigue expandiéndose... como Hara.

Gracias a Guillermo Ballenato, por la magnífica experiencia que fue asistir a tus cursos de desarrollo personal, por tu guía y ejemplo, por mostrarme que ser feliz es un imperativo.

Gracias a José Manuel Ruiz. Gracias por enseñarme a "pasar la itv" a los problemas, por enseñarme a pensar como un ingeniero.

Gracias a mis profesores David y Mari Paz. Los conocimientos que me transmitisteis se han difuminado con el tiempo, pero vuestro ejemplo y sabiduría siguen tan vívidos en mi mente como cuando tenía diecisiete años.

Gracias a Jesusito, mi compañero de armas en la ČVUT. Nunca olvidaré aquellas anécdotas de la universidad, que tanto nos gusta recordar ante una cerveza.

Gracias a Víctor, Santi y Gabi, por aquella tarde de itinerancia y autovalores.

Gracias a Álvaro y a Carlos, por aquel febrero efervescente en la Zona Sur.

Gracias a María, mi guía, mi motor, mi compañera.

Gracias a mis compañeros de clase, mis compañeros de fatigas. Gracias por todas esas risas, apoyo, energía, cafés... que hacían menos duras las prácticas y las horas de estudio. Gracias por enseñarme tanto.

Gracias a mis amigos de la universidad, que habéis hecho de esta una etapa dorada.

Gracias a mis amigos del Erasmus, mis hermanos pragueños. Juntos consumamos una de las mejores experiencias que se pueden vivir.

Gracias a mis compañeros en T-Labs, a mis amigos de Berlín. Gracias por aquellos seis meses mágicos.

Y gracias a mis amigos de siempre, vosotros, los que siempre estáis ahí.

Gracias a todos los que os habéis cruzado en mi camino. Gracias a todos los que me habéis acompañado en este viaje, regalándome lo mejor de vosotros, y aún lo hacéis. No añado más nombres, vosotros sabéis quiénes sois.

A todos, GRACIAS.

Alcorcón, Octubre de 2013 Gerardo Pinar Loriente

Abstract

In recent times, *Next Generation Mobile Networks* (NGMN) enable user's mobility, not needing to be in a fixed place anymore. For this service to be successful, seamless transitions between the different technologies become essential, in order to make possible the "always best connected" goal.

This brings an additional problematic, which is the impairments resulting from a handover between two networks. In order to succesfully plan and continue the development of "always on" services and mobility management, the approach must be based on user's perception of phenomena such us packet loss, the so-called *Quality of Experience* (QoE). This is the context in which Mobisense was born, intending a better understanding of NGMN transmission phenomena and resulting quality.

Hence, Mobisense project is focused on the evaluation of the quality of service from user's point of view and on the seamless switch provision between video codecs when connections are transferred between two networks. On that purpose, a NGMN test environment was developed for real-time multimedia services. In this environment, specific network conditions can be associated with user's assessments within a realistic model.

Mobisense project creates the foundations for the employment of advance prediction methods for real-time mobility management, to make decisions depending on the characteristics measured in the network and the predictions of quality resulting from them.

The present degree final project gathers the work carried out to develop an extension for Mobisense testbed, in order to deploy it in a real environment of network technologies, as well as the integration with *Quality of Service* (QoS) algorithms. Therefore, the aim of this project consists on the development of the software required for the creation of a video thelephony system over NGMN, taking Mobisense and MultiRAT testbeds as starting point. Mobisense brings adaptation in the application layer and user's perception, and MultiRAT provides QoS adaptation and new wireless technologies. Both testbeds combined to explore more QoE aspects in wireless networks of tomorrow.

Key words: NGMN, mobility, handover, QoE, user's perception, QoS, prediction, testbed, adaptation.

Resumen

En los últimos tiempos, las NGMN posibilitan la movilidad del usuario, sin ser ya necesaria su permanencia en un lugar fijo. Para el éxito de este servicio se hacen indispensables transiciones continuas entre las diferentes tecnologías, de manera que sea posible el objetivo de "siempre la mejor conexión".

Esto lleva consigo una problemática adicional, que son las deficiencias resultantes del *handover* entre dos redes. Para planificar y continuar satisfactoriamente el desarrollo de servicios *always on* y la gestión de la movilidad, el enfoque debe ser en base a la percepción del usuario de fenómenos tales como la pérdida de paquetes, la llamada QoE. En este contexto nació el proyecto Mobisense, buscando una mejor comprensión del fenómeno de transmisión NGMN y la calidad resultante.

El proyecto Mobisense se centra, por tanto, en la evaluación de la calidad de servicio desde el punto de vista del usuario y en la provisión de cambios continuos entre codecs de vídeo al transmitirse conexiones entre dos redes. Para tal propósito, un entorno de pruebas NGMN fue desarrollado para servicios multimedia en tiempo real. En este entorno, pueden asociarse determinadas condiciones en la red con valoraciones de calidad por parte del usuario en un modelo realista.

El proyecto Mobisense sienta las bases para el empleo de métodos avanzados de predicción para la gestión de movilidad en tiempo real, para tomar decisiones dependientes de las características de la red medidas y de las predicciones de calidad derivadas a partir de éstas.

El presente proyecto de fin de carrera recoge el trabajo realizado para desarrollar una extensión del *testbed* Mobisense, de cara a desplegarlo en un entorno real de tecnologías de red, así como la integración de algoritmos de QoS. Por tanto, el objeto de este proyecto consiste en el desarrollo software requerido para la creación de un sistema de videotelefonía sobre NGMN, tomando los *testbeds* Mobisense y MultiRAT como punto de partida. Mobisense aporta adaptación en la capa de aplicación y la percepción de usuario, y MultiRAT proporciona adaptación QoS y nuevas tecnologías de red. Ambos testbeds se combinan para la exploración más amplia de los aspectos de QoE en las redes inalámbricas del mañana.

Palabras clave: NGMN, movilidad, *handover*, QoE, percepción del usuario, QoS, predicción, *testbed*, adaptación.

Contents

List of Figures

List of Tables

List of Acronyms

ACELP *Algebraic Code-Excited Linear Prediction*

ADPCM *Adaptative Differential Pulse Code Modulation*

AP *Access Point*

BCCH *Broadcast Control Channel*

BS *Base Station*

BSS *Basic Service Set*

BOWL *Berlin Open Wireless Lab*

CELP *Code-Excited Linear Prediction*

CPU *Central Processing Unit*

CS *Coding of Speech*

DL-MAP *Downlink Map*

DOM *Document Object Model*

DS *Distributed System*

EDGE *Enhanced Data Rates for GSM Evolution*

ERP *Erns-Reuter-Platz*

ESS *Extended Service Set*

ETSI *European Telecommunications Standard Institute*

GUI *Graphical User Interface*

GPRS *General Packet Radio Service*

HSDPA *High-Speed Downlink Packet Access*

HTTP *Hyper Text Transfer Protocol*

IEEE *Institute of Electrical and Electronics Engineers*

IETF *Internet Engineering Task Force*

IP *Internet Protocol*

ITU *International Telecommunication Union*

ITU-T *ITU Telecommunication Standardization Sector*

LAN *Local Area Network*

LD-CELP *Low Delay CELP*

MANET *Mobile Ad hoc Networks*

MEGACO *Media Gateway Control Protocol*

MGCP *Media Gateway Control Protocol*

MOS *Mean Opinion Score*

NGMN *Next Generation Mobile Networks*

NGN *Next Generation Network*

NPCP *Network Parameters Control Protocol*

PCM *Pulse Code Modulation*

PLC *Packet Loss Concealment*

PSTN *Public Switched Telephone Network*

QoE *Quality of Experience*

QoS *Quality of Service*

QU *Quality and Usability*

RAT *Radio Access Technologies*

RTP *Real-time Transport Protocol*

RTCP *RTP Control Protocol*

SIP *Session Initiation Protocol*

SNR *Signal to Noise Ratio*

SS *Subscriber Station*

SS7 *Signalling System No. 7*

SSID *Service Set Identifier*

SVG *Scalable Vector Graphics*

TC *Traffic Controller*

TCP *Transmission Control Protocol*

UDP *User Datagram Protocol*

UMTS *Universal Mobile Telecommunications System*

VLAN *Virtual Local Area Network*

VNC *Virtual Network Computing*

VoIP *Voice over IP*

VPN *Virtual Private Network*

WFD *WinterfeldStraße*

WLAN *Wireless LAN*

XML *eXtensible Markup Language*

Chapter 1

Introduction and objectives

If one does not know to which port one is sailing, no wind is favorable.
(Lucius Annaeus Seneca)

Think, believe, dream, and dare.
(Walt Disney)

This document describes the final project prepared as a result of the work developed during an internship, under Erasmus Placement program, at Deutsche Telekom Innovation Laboratories (T-Labs) in Berlin, under the supervision of Błażej Lewcio and Nico Bayer.

The work was carried out in the period from October 2011 to March 2012, within *Quality and Usability* (QU) department, dedicated to develop methods to measure the quality and the usability ease of information and communication systems.

The first steps were to become familiar with Mobisense. *Mobisense goes video* is a research project at the QU department of T-Labs. Born in the context of *Next Generation Network* (NGN) and seamless mobility, Mobisense studies video and audio quality in mobile heterogeneous networks, contributing to achieve network performance metrics closer to the user's real perception of audio-visual services. With this approach, Mobisense gives -by focusing on the correlation of user's perception and network performance- a hint about how future audio-visual services will be experienced in this scope.

The idea of the new contribution was an extension for the testbed, intending a real scenario of heterogeneous wireless access technologies as well as integration of network-based QoS optimization algorithms. Additionally, a prototype of a QoE-aware mobility management was intended to be implemented, pursuing further development of QoE-aware multimedia streaming applications. This becomes a highly relevant question in future communication systems.

At the end of the internship, in March of 2012, the project was satisfactorily completed with the achievement of main goals of software production, with the exception of the final deployment completion and minor details. The demonstrator stayed ready (when totally deployed) for further quality of experience exploration.

1.1. Motivation and background

Most of telecommunication services which are demanded nowadays concern to mobile networks. However, users expect not only service connectivity, but also service quality. Therefore, to manage users' expectations and multimedia streaming optimization, it will be needed to make a trade-off between service quality and the resources allocated for the service offering. In this context, it is interesting to equip multimedia applications with QoE awareness: monitoring tools that are able to control streaming parameters across transmission layers. Another important role for the service operators consists on techniques for active service adaptation, because of the resulting changes of link quality during ongoing transmission.

The real user's perception must be known in order to choose the most suitable streaming parameters. Parameters such us transmission rate, audio and video codec or packet size, can be adjusted according to the network circumstances and the intended service quality. Because of these subjective aspects of quality perception, multimedia streaming configuration must be validated in experimental environments prior to offering them to the users. On this purpose, the experimental testbed Mobisense is developed as a playground for testing of new networking and streaming techniques.

The validation can be done in the form of system logs and an analysis can be performed with the collected output signal. In this way, it is possible to track bottlenecks and propose solutions to improve in terms of quality. With respect to the perceived quality, the impact of multimedia streaming configuration can be visualized using an integrated demonstration interface. Summing up, the contribution of the testbed is relevant in the field of designing user perception aware multimedia applications. This relevance comes because Mobisense is giving an answer to the question of how future audio-visual services will be experienced in the context of NGN, mobile technology and seamless mobility, and the answer is the correlation of user's perception and network performance, tightly coupling performance metrics and real user perception.

Mobisense goes video line of action is focused on merging user's experience with cross-layer parameters of multimedia streaming in wireless networks. A thorough analysis of user's perception is necessary in order to perform a proper quality evaluation, the parameters to study are: packetization, buffering, link, quality adaptation during active session...

One of the requirements for the project is the visualization of the effect caused by the audiovisual streaming configuration on perceived quality. To accomplish this specification a demonstration *Graphical User Interface* (GUI) is developed; in which received signal could be displayed, together with application control and monitoring widgets. Information about the streaming quality, correctness of the decoding process was provided. Moreover, information about the currently used bit rates and size of the jitter buffers could be found. As well as the names of currently used codecs and the name of the network which the client is currently connected to. Among the control features which are provided, there exists a network switch interface, and the

option to artificially impair the transmission by packet loss, delay or delay variation. These control widgets are available to be used at any moment during the demonstration, consequently, the resulting streaming quality can be observed in real time. Results have been generated in real time for several configurations. Tests can be automatically processed and data can be collected, the event-based testing concept is used to flexibly define test conditions.

In sum, the research line of *Mobisense goes video* testbed is oriented to identify the most effective strategies for quality management, from a perceptual point of view; as well as assurance of valid instrumental methods for quality prediction in heterogeneous networks. All in order to maximize the QoE in modern telephony systems [1].

Further extensions are intended, such as more wireless technologies integration, as WiMAX, LTE and Femtocells, as well as the integration of network based QoS optimization algorithms. Aiming to achieve a prototype of QoE-aware mobility management.

1.2. Objectives and contribution

The objective of this degree final project consists on providing software support for the Mobisense testbed extension. This extension entails a new scenario where Mobisense purposes could be implemented over real wireless network technologies: user's perception of transmission changes, like roaming between wireless networks; audio/video codecs switching or video bit rate changing during active calls. Hence, this new environment consisted of two testbeds: *Mobisense goes video*, which brings adaptation in the application layer and user perception; and *MultiRAT*, which provides QoS adaptation and new wireless technologies. Together, both testbeds pursue the exploration of more QoE aspects in future wireless networks.

This new approach demands several steps for its implementation, among all of them, the main tasks which I accomplished are described below:

- A traffic controller unit should be created, which purpose of enabling an expert user to send instructions to change parameters or artificially alter the conditions of the network. This functionality is already implemented in *Mobisense goes video*, but in the new testbed should be separated from the client.
- Since the control function is intended to be separated from the client, an appropriate protocol for control operations signaling is needed. This protocol should be designed and implemented to support communication between the traffic controller and the clients.
- Two GUI should be designed and implemented, for clients and traffic controller. The purpose of this is to enable visualization of the effect caused by audiovisual streaming configuration on perceived quality; as well as the presence of control widgets to act over the network parameters.

1.3. Project phases

The objective is clear: *Mobisense* testbed should be extended and integrated with *MultiRAT* testbed.

Two phases were established at the begining for this project:

- Phase 1: Extension of the existing demonstrator, intended to last 1 month.
- Phase 2: Design and deployment of an online demonstrator for video chat between ERP and WFD, it would take 3 months.

The steps defined for the phase 1 are:

1. Setup of the development environment.

1. Interconnection of the testbeds[1].

2. Integration of a network control widget with autodetection of active interfaces.

3. Integration of a network bandwidth control gauge (based on *netem* or similar).

4. Integration of QoS control functions from *MultiRAT*.

The steps for the phase 2:

1. Deployment of a second Video Node in the WFD.

2. Integration of a network socket in the video client for remote control.

   a) Support of video/audio codec and bitrate control, and status information retrieval, initially.

   b) Implementation based on XML markup-language for the signaling message syntax.

3. Implementation of a stand-alone testbed controller for online demonstrator/testbed.

   a) Task based on the extraction of the existing test controller code from the video client and creating a separate entity for control purposes only.

   b) This step will allow more modular experiments/demonstrations in the future.

4. Development of a simple but catchy GUI for video telephony.

[1] In collaboration with Theresa Enghardt.

1.4. Document structure

Chapter 2 summarizes the relevant technologies in the scope of this book. VoIP characteristics are briefly covered, such as protocols, codecs or quality measurement. The importance of cross-layer adaptation is introduced and a quick overview about wireless technologies is referred. The relation concludes with a short exposition of the state of the research at the beginning of this project, having a glimpse along Mobisense and MultiRAT testbeds.

Chapter 3 tackles the new testbed design, beginning with new requirements and the reason for both testbeds to be combined. The chapter continues along architecture and the protocol issues, such as types of message or message format. Thereafter, graphical user interfaces are presented. Finally, structure and elements of client and traffic controller applications are detailed in the software design section, including figures and diagrams for further understanding of the programs design and behaviour.

In Chapter 4, the diverse tasks performed in this project are evaluated, with focus on the possible improvements for the designed protocol. It is important to note the constraint of not relying on measures or test results, since no videocall could be performed within the time of the internship; this is why only a qualitative assessment is provided.

Chapter 5 summarizes the achievements and presents some possible lines to follow in future research.

Chapter 2

State of the art

Absorb what is useful, discard what is not.
(Bruce Lee)

In this chapter, a panoramic view of the current state of technologies involved in this project is presented. First technology addressed is VoIP, with a brief introduction of main issues related, such as: characteristics, protocols, etc., until the description leads into one of the main topics of the background research, which is quality and cross-layer adaptation. Thereafter, the exposition continues with a short review of the main network technologies covered throughout the project. Finally, the existing system in which the present project is integrated is introduced, passing over the testbeds Mobisense and MultiRAT.

2.1. VoIP overview

In the '90s, new topics started having relevance in the field of communication, such as voice and data convergence. The design of networks capable to transport different nature information was desired [2].

The classic paradigm was based on circuits switching, denominated *Public Switched Telephone Network* (PSTN), where a path was implemented between source and destination with a guarateed badnwidth, and resources exclusively dedicated to that communication [2].

The other approach is packets switching, where source sends information in form of data packets which are independently routed to the destination. This brings a more efficient -and cheaper- use of resources, since they are not exclusively dedicated to one call; but entails additional complications, such as packets arriving out of order, variable delays, or packets loss [2].

This conjugation of the -formerly dimensions- voice and data transmission, constitutes the foundation for VoIP. VoIP technology enables to encapsulate the voice in packets in order to be transported over data networks, without the need for PSTN circuits [2].

2.1.1. Characteristics

Below, the main characteristics of VoIP are presented:

1. Regarding the structure:
   - A link to traditional telephony network is provided [3].
   - Control of network traffic, reducing the probability of having important failures. [3].
2. Considering that VoIP is a technology supported by *Internet Protocol* (IP), some additional advantages are derived:
   - Independence from the employed hardware [3].
   - Independence from the supporting physical layer [3].
   - Both software and hardware implementation are feasible [3].

2.1.2. VoIP-PSTN comparative

Table 2.1.2 shows the existing differences between the two paradigms: switched interconnections vs. IP interconnection.

Switched (Telephony) Networks	Next Generation (IP) Networks
Circuits dimensioned for voice	Traffic types vary (different QoS needed)
Interconnection fee based on time	Packets have no time or distance dimensions
Fixed - Mobile interconnection asymmetric	Packets exchanged uniformly across platforms
Small but constant information delivery rate	Typocally "bursty" traffic patterns
Little tolerance for delays and sound distortions	Handle time sensitive and delay tolerant traffic
Regulated interconnection at agreed POIs	Unregulated peering and transit
Traffic routed over a circuit to a dialed number	Connectionless, 'best efforts' routed on IP headers

Table 2.1: Comparison of Legacy (PSTN) and IP NGN [4]

As networks are becoming digital, switched networks are giving way to NGN. In Figure 2.1 can be observed the growth of IP telephony, which means VoIP is gaining ground to PSTN.

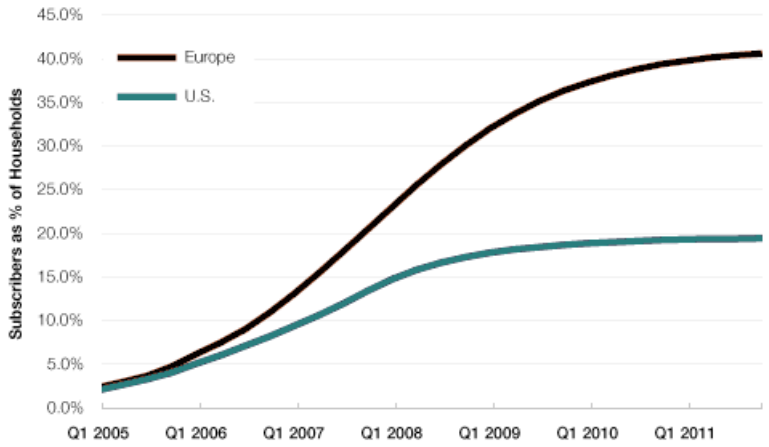


Figure 2.1: VoIP evolution [5]

2.1.3. Components

VoIP model consists mainly of three elements [5]:

1. *Client*: In charge of voice calls enstablishment and ending. The information coming from user's microphone output is coded, packetized and transmitted. Likewise, incoming information is received, decoded and played through speakers or any output device.

2. *Server*: Users operations are conducted, such as validation, accounting, routing, service control, users and services registration, etc.

3. *Gateway*: Interfaces with traditional telephony are provided. working as a platform for virtual clients. Moreover, this equipment also plays an important role about access security, accounting, QoS and service enhancement.

2.1.4. Codecs

Different algorithms have arisen to takle the transformation process of analog speech into discret binary samples and transport through a packet network, which is a complex procedure, with several degrees of freedom involved. The steps wich all these algorithms must follow are [6]:

1. Continuous analog signal transformation into a train of discret samples.
2. Quantization process: selection of a set of limited and standardized values, in order to enable conversion to encoded binary words. Non-uniform quantization is used to achieve smaller error, since distance between two standard values becomes smaller for smaller values of amplitude.
3. Transmission across the network.

Thanks to "intelligent" coding methods, the reduction of quality resulting from filtering and quantization process can be mitigated by reducing bandwidth consumption. These codecs can be grouped in three families: waveform, source and hybrid codecs; with diverse approaches. These families will not be described in this document, but some of the most relevant codecs are presented in table 2.1.4. The most widely used codecs are G.711 (optimal audio quality and moderated *Central Processing Unit* (CPU) consumption), G.729 (more opitimized bandwidth, but with higher CPU consumption) [7].

Standard codecs (ITU)	Description	bit rate (Kbps)
G.711	PCM	64
G.721	ADPCM	32, 16, 24, 40
G.722	PCM-ADPCM	48, 56, 64
G. 723.1	CELP	5.3,6.3
G.728	LD-CELP	16
G.729	CS-ACELP	8

Table 2.2: Standard codecs (ITU) [8]

2.1.5. Protocols

2.1.5.1. Media transport and quality feeback protocols

Real-time Transport Protocol (RTP) is nowadays the choice for most multimedia applications on the Internet. Originally specified in RFC 1889 and updated in RFC 3550 [9], it was design together with its companion protocol: *RTP Control Protocol* (RTCP), born with purposes of feedback from users, about communication quality. The aim of these protocols is merely to collect and transport extensive QoS information. It is the controlling application's task to use, or not, that information.

Multimedia Streaming using IP entails some important restrictions which can be satisfied neither by *Transmission Control Protocol* (TCP) nor *User Datagram Protocol* (UDP). These restrictions come from the need of sampling the signal at very short intervals, and its reproduction at the exact same rate. Therefore, VoIP traffic should have end-to-end delay no longer than 150 ms for good voice quality, as well as a strictly bounded jitter[1] [6].

2.1.5.2. Signaling protocols

Signaling performs an important role in VoIP, which is communication management between two points, including establishment, maintenance, administration and ending. Moreover, QoS is provided at every transmission channel . Below some of most important protocols in VoIP are described [8]:

1. *H.323*: Standards family developed by *International Telecommunication Union* (ITU) in 1996 aiming to offer a transport mechanism for multimedia services over QoS-non-guarateeing networks; it is widely used on IP networks. Despite H.323 is, technically, a strong protocol, the users' interest has decreased because of some complexity and inefficiency issues.

2. *SIP*: Protocol designed by *Internet Engineering Task Force* (IETF) in 1999 for multimedia sessions control and implementation of advance telephony services. *Session Initiation Protocol* (SIP) is based on *Hyper Text Transfer Protocol* (HTTP), in aspects such as syntax simplicity and a client-server structure conceived under request-reply model. The great virtue of SIP lies on its flexibility, since new services not defined by the recommendation can be programmed. The purpose of SIP is merely to initiate or terminate voice/video calls. All voice/video communications are supported by RTP.

3. MGCP-MEGACO: Standard for VoIP also developed by IETF, based on a master-slave model, in which transmission signaling is separated from information, simplifying integration with *Signalling System No. 7* (SS7) protocol. This advantage resulted in joint collaboration between IETF and ITU for a new spec based on *Media Gateway Control Protocol* (MGCP): *Media Gateway Control Protocol* (MEGACO), also referred as H.248 by ITU. In overall, SIP and H.323 are used for end-to-end signaling, whereas MEGACO is optimal for large telephony operators.

2.1.6. Architecture

A benefit of VoIP technology is the versatility of networks to have centralized or distributed architecture. This flexibility enables companies to consider two kind of network characterizations: simplified administration and terminal innovation,

[1] Factors affecting quality are described in section 2.1.7 on page 12.

depending on the used protocol [5]. Centralized approach is criticized for not being flexible enough to adopt future technological innovations. On the other hand, distributed architecture introduces higher complexity [10]. Therefore, flexibility becomes the best strength in the architectural dimension of VoIP.

1. *Centralized architecture*: It is usually associated with MGCP and MEGACO protocols. Network intelligence is centralized, and terminals are practically intelligenceless. The virtue of this model is a simple calls flow, repeating voice characteristics [5].

2. *Distributed architecture*: In this case, association is with protocols H.323 and SIP, which allow intelligence to be distributed among terminals and calls control devices. Intelligence here means calls establishment, characteristics, routing, provisioning or invoicing among others. Terminal is any device capable to initiate or terminate a VoIP call. Calls control devices are the so-called gatekeepers in a H.323 network, and proxy servers in a SIP networks [5].

2.1.7. Quality issues

Voice quality is affected mainly by the following factors [5]:

1. *Codecs*: see table 2.1.4.

2. *Frame loss*: As a result of network congestion or data corruption.
   Since frame relaying is not practical in real time, terminals are in the need of dealing with frame erasure. The effect of this losses in voice quality depends on how it is managed by the terminals:

   - Jerkiness: the simplest case, the terminal introduces a silent interval in the voice stream.
   - *Packet Loss Concealment* (PLC): lost frames are compensated in base of prior samples. PLC is included in codecs such as: PLC+G.711, PLC+CELP, G.723.1, G.728 and G.729.

3. *Delay*: two sources of delay can distinguished:

   - Algorithmic delay: introduced by the codec, inherent to the coding algorhithm.
   - Packetization delay: time required to complete an information packet, once the coding and compression processes are concluded. This time depends on the size of the required block for the voice coder and on the number of blocks per frame.
   - Serialization delay: time required to transmit an IP packet. Appears whenever a packet goes through a router or switch.

- Propagation delay: time that the optical or electrical signal takes to travel along the transmission media. Depends on geographical distance.
- Component delay: due to the different components operations within the transmission system.

Some impairments caused by delay are echo and speakers overlap.

4. *Delay variance (jitter)*: destination terminal doesn't receive frames in regular intervals. Storing frames in a buffer is the general strategy to circumvent jitter. The larger the buffer is the longer the margin will be for frames to arrive. However, an additional delay is introduced in this operation, a good approach for optimal decision is to use an adaptive memory.

5. *Global delay*: the sum of all the delays in the system.

2.1.7.1. Quality measurement

The two main tools to measure quality are subjective testing and instrumental monitoring. Tests are time consuming, since many users have to be asked to evaluate QoS according to a standard process and give a score[2], that is why there is a big investment in the development of instrumental measurement tools, as the *ITU Telecommunication Standardization Sector* (ITU-T) E-Model [6].

The main challenge for the successful deployment of VoIP services is the underlying technology, which inherently lacks QoS guarantees for the sensitive voice traffic.

2.2. Cross-layer adaptation

QoS can be decreased by the constraints experienced by multimedia data transmission rate. These constraints arise due to the nature of multimedia applications, which are characterized by three main properties: demand for high data transmission, sensitiveness to packet delays and tolerance to packet losses. Users demand ubiquitous connectivity, which motivates evolution from wired to wireless technologies. But it entails a change of paradigm, multimedia data transmission over wireless networks inherits all the characteristics and constraints related to free space propagation. Hence, in wireless networks packet loss is due not only to congestion in the path, as occured in wired networks, but also to low *Signal to Noise Ratio* (SNR), causing packets corruption; multi-path signal fading or interference from neighboring transmissions [11, 12].

Therefore, if packet loss occurs due to packet corruption, a congestion avoidance mechanism may be activated not being aware of the true nature of the loss. This mechanism at the transport layer would reduce the transmission rate, aiming to avoid congestion, and thus degrading the performance of the application, even when

[2]Typically from 1 to 5 (*Mean Opinion Score* (MOS) score)

congestion is not the cause of the packet loss. A better solution would be to share information about the specific application characteristics and network conditions among all layers, enabling the use of more efficient mechanisms against packet loss. [11, 12].

All the above factors set the challenge of developing new techniques to mitigate delay and packet loss ratio during multimedia transmissions. The classic approach, based on the layered model, does not seem to be valid anymore, since it cosiders communication only between adjacent layers without heeding specific characteristics of multimedia applications; and this is not suitable for wireless networks. On the other hand, a new paradigm opts for a high level of cooperation among layers in order to achieve a more effective performance. Here arises the concept of cross-layer adaptation, under the challenge of developing mechanisms to increase the level of cooperation and communication between layers without altering the layered model [11].

2.3. Network technologies overview

This section presents a review of main network technologies covered throughout this document.

2.3.1. WiFi - IEEE 802.11

Also known as *Wireless LAN* (WLAN), this networks are based on an architecture where the system is divided into cells or *Basic Service Set* (BSS), which define areas where mobile users are capable to achieve communication through an *Access Point* (AP) which controls the BSS. Association and authentication are indispensable phases for communication with AP at WLAN. In another type of networks, no AP exists and communication takes place directly between users, these networks are denominated *Mobile Ad hoc Networks* (MANET) [13].

The most frequent situation is that in which several AP exist and are interconnected through a common backbone, known as *Distributed System* (DS), which can be wired or wireless as well. DS becomes essential, since enables multiple BSS working integrated.

2.3.1.1. Network discovery

Network discovery is required before a user can communicate with an AP, and it can be performed both in an active or a passive manner. In passive exploration, the node listens to the messages sent by the AP. These messages (beacons) are sent periodically (usually 10 times per second) and contain the AP configuration. A client can have one or several network names (*Service Set Identifier* (SSID)) predetermined; when the client receives a beacon frame from a configured SSID, automatically connects to that AP, and if receives two frames with the same SSID, then the client will connect to the AP with stronger signal. On the other hand, active

exploration is performed by sending messages (*Probe Request*) to all the reachable AP. These requests contain the SSID that the client node is looking for; in case the AP receives the request, a *Probe Response* will be sent back; if no network is suitable for the client, then an empty SSID will be sent. In *ad hoc* networks, and only in this case, where no AP is involved, all the nodes send beacon frames. Continuous exploration allows the client to keep a list of best neighboring APs [13].

2.3.1.2. Features [13]

1. *Time stamp*: used for synchronization of the user with the AP.

2. *Channel information*: the *European Telecommunications Standard Institute* (ETSI) describes 13 channels for the use of WLAN.

3. *Transmission speed*: transmission speeds for IEEE 802.11b/g are: 1, 2, 5.5, 22 and 54 Mbps. IEEE 802.11n allow speeds up to 300 Mbps.

4. *Supported services*: the types of netwok are BSS, *Extended Service Set* (ESS) and *ad hoc*.

2.3.2. WiMax - IEEE 802.16

The objective of WiMax networks is to provide telecommunication services to all those clients in areas not reachable by wired and/or WiFi networks, or it entails a high cost. According to the standard, this technology enables transmission between points separated up to 50 Km with direct line of sight and 15 Km otherwise.

The operating mode is similar to WiFi networks. WiMax architecture contains two main components: *Base Station* (BS) and receiving clients. BS offers connectivity to all the *Subscriber Station* (SS). Communication occurs between BS and SS nodes. Additionally to point-to-multipoint mode, WiMax supports mesh network connection, with all the subscriber nodes communicating with each other. This configuration is similar to WiFi *ad hoc* (IEEE 802.11) networks, where no central point exists, and thereby, management and coordination tasks are carried out by the network nodes themselves [13].

2.3.2.1. Network discovery

Once again, this function is also similar to the one employed in WiFi networks. A client station can detect access points by listening for *Downlink Map* (DL-MAP) messages, which contains information as *Base Station ID* and *Operator ID*. In case of different options, the client will select the best suiting network [13].

2.3.3. Telephony networks

3G[3] systems can transmit both voice and data, just as *General Packet Radio Service* (GPRS). Nevertheless, 3G can reach higher speed (up to 7.2 Mbps or more) in data transmission, enabling video sending and reception, as well as high speed Internet access or video calls [13].

The services provided by 3G technology comprise: Internet access, wide band services, international roaming and interoperability. These systems enable the development of multimedia systems for real-time video transmission [13].

2.3.3.1. Main features

- High speed in data transmission: up to 144 Kbps mobile data (vehicular), 384 Kbps in portable data (pedestrian) and 7.2 Mbps in static data.
- Better capacity and spectrum efficiency compared to wired systems.
- Voice quality comparable to the one offered by wired systems.
- Symmetrical or asymmetrical data.
- Packets and circuits switching services.
- Incorporation of second generation systems and possibility of coexistence and interconnection with mobile services via satellite.
- Capability of simultaneous services provision to terminal users.

2.3.3.2. Network discovery

Similar as in WiFi and WiMax, base station sends a broadcast message through a special point-to-multipoint channel (*Broadcast Control Channel* (BCCH)) with general information for all the mobile terminals in the network. Once the information is received, the mobile user synchronizes with the base station, and will be on-line after registration and authentication phases [13].

2.4. Existing system

2.4.1. Mobisense testbed [14]

2.4.1.1. Introduction

As mentioned in Chapter 1, *Mobisense goes video* is born within the background of QoS evaluation in the context of the services which are nowadays offered to clients in mobile networks. Mobisense is an experimental testbed which makes possible to

[3]third technological generation of mobile telephony

merge user experience with cross-layer parameters of multimedia streaming in wireless networks. Clients do not expect only service connection anymore, but for multimedia services, like video telephony or mobile TV, they also expect service quality. Therefore, knowing the clients' needs and expectations is of vital importance for the service operators, as well as avoiding resources over provisioning in the networks. This is why QoE aware applications are attracting an increasing attention: by QoE monitoring, circumstances and state of the network is observed and some stream parameters across the layers can be controlled. These stream parameters can be audio and video codec, transmission rate or packet size; and it is important to remark that -in order to choose the more suitable ones- the knowledge of real user perception is required. Neglecting QoE can result in disastrous effects for the service. Just to give some examples: a delay when switching from narrow to wide band speech quality during an ongoing transmission may degrade user's perception; on the other hand, an increase of video bitrate beyond a certain threshold does not significantly improve user's experience. And it is because of these subjective aspects with quality perception that multimedia streaming configurations -prior to be offered to the users- must be validated in experimental environments. In this way, experimental testbeds are developed and used as playground for testing networking and streaming techniques: system bottlenecks can be tracked and improvements can be proposed ensuring the success in terms of service quality. The experimental testbed described in this section enables the merge of user experience with cross-layer streaming parameters during wireless transmission. The main components are:

(i) Audiovisual client: encodes, transmits and decodes video signals in real time. Diversity of video input and output possibilities, as well as transmission parameters: speech, audio and video codecs, transmission rate, size of transmitted packets, size of receiving jitter buffer. Also additional jitter buffer monitoring allows to compare network and application statistics related to data loss.

(ii) Network infrastructure: enables transmission in heterogeneous wireless networks with network handover support. As an additional feature, the link quality can be artificially degraded and background traffic generated.

Derived from this complexity, an additional testbed control unit is integrated in the client to make automatic test processing possible. The test conditions can be flexibly defined using XML-Markup language.

2.4.1.2. Test client

The software for the client is based on PJPROJECT 0.8.3 framework that inherently supports Voice over IP communication. The media stack has been extended towards video functionality and is based on SIP for multimedia sessions control and RTP for real-time multimedia data.

In Figure 2.2, a general view of the client software is presented. The picture is divided in two media flows of similar structure: audio and video processing. Within

each one there is -as central element- a bridge unit (Conference or video Bridge) which interconnects sources and sinks of uncompressed audio or video signal that can be decoded in Media Streams or acquired from peripheral devices like camera or sound card. There is additional support for files with pre-recorded uncompressed signals as input.

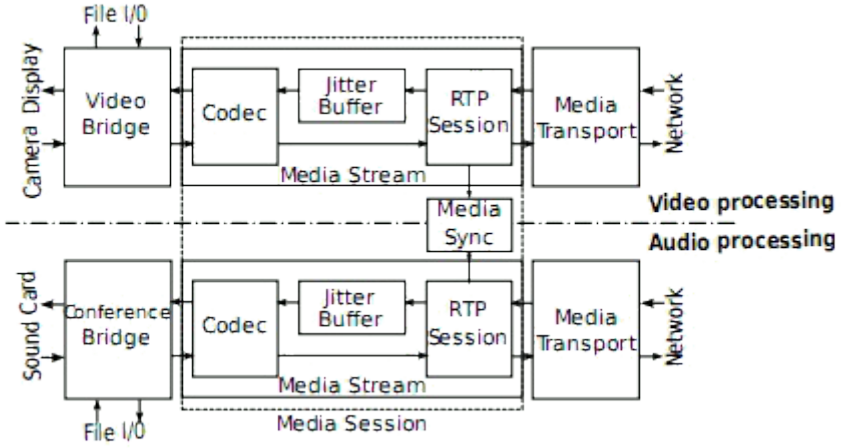


Figure 2.2: Application building blocks and media flow in the client [14]

Not to extend too much, just mention that the video processing functionality is implemented using *libavformat* (core functionality for uncompressed video processing) and *libavcodec* (implementation of several video codecs) libraries which belong to ffmpeg framework. The audio processing part is mainly implemented by the original developers of PJPROJECT, though additional audio and speech codecs have been integrated within the scope of this project.

2.4.1.3. Networking setup

The test environment where the video client has been deployed consists of two interconnected testbeds: the Mobisense (provides access to heterogeneous technologies: WiFi, *Universal Mobile Telecommunications System* (UMTS)/*High-Speed Downlink Packet Access* (HSDPA), ...) and Router Lab testbed (enables emulation of diverse backbone link characteristics: artificial transmission degradation and emulation of asymmetric ADSL links using Dummynet, and generation of background traffic). All the networks are IP-based and it is used Mobile IP Protocol for network handover support.

The components of the testbed architecture are:

- A Mobile Node (MN): connected via WiFi and LAN to the Dummynet computer in the Router Lab, and via HSDPA to the Internet.

- A Correspondent Node (CN): connected via a Gigabit Ethernet link to the home network.

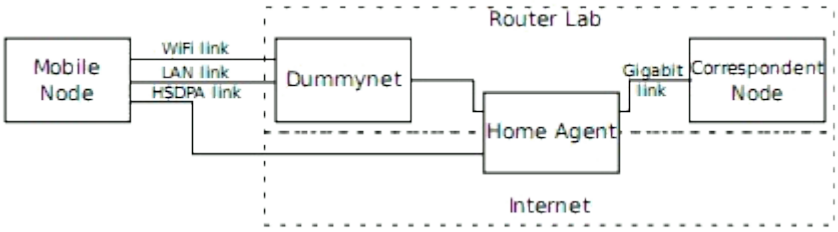


Figure 2.3: Networking setup [14]

- A Home Agent (HA): dual-homed. Communication between CM and MN occurs via HA, and selectively via the Internet or the Router Lab.

The networking infrastructure enables to investigate the multimedia streaming quality in heterogeneous networks, and -particularly- allows to address the streaming quality during network handovers.

2.4.1.4. Testbed control unit

As mentioned before, due to the high complexity of the system -coming from the high number of components and functions integrated in the testbed- an additional control unit has been developed and integrated into the client, in order to enable automatic test processing and to provide an interface for the demonstrator.

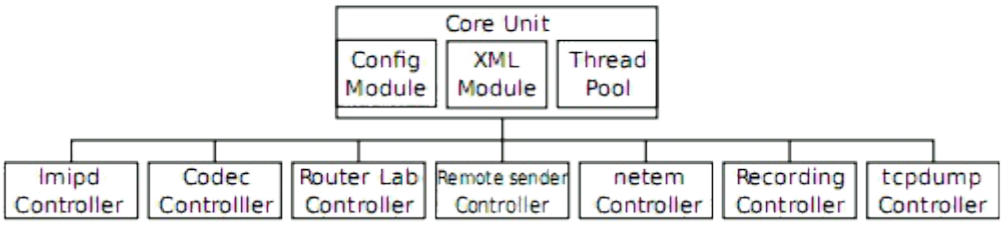


Figure 2.4: Building blocks of the Test Controller [14]

In Figure 2.4 an overview of the Test Controller is depicted. There are core components with some modules attached for the control of certain components and functionalities:

Core Unit

- *Configuration Module*: for testbed configuration. It is also used to provide user-friendly names for different parts of the system, e.g. *WiFi* instead of *wlan0*.

- *XML Module*: for test description files.

- *Thread Pool*: to execute related processes or functions that run simultaneously to the audiovisual transmission.

Control modules

- *lmip Controller*: for network handovers.
- *netem Controller*: for artificial degradation of the link quality.
- *Remote Sender Controller*: to start the video on Correspondent Node (CN).
- *Router Lab Controller*: for Dummynet PC control.
- *Codec Controller*: to change the audio and video codecs.
- *tcpdump Controller*: to trace the network traffic.
- *Recording Controller*: to record the received audio and video signal.

The automatic test processing has an event-based design: when any streaming parameter is changed it is considered as a single event, and it has timing information associated that describes when the parameter should be changed. Below is the list of currently -it can be extended- supported events:

- Network change.
- Packet loss change.
- Audio codec change.
- Video codec change.

Due to this modularity, it is possible to have multiple configuration events scheduled at different times during the processing for each test scenario.

2.4.1.5. Exemplary results

The described testbed was used to study the video call quality perception in mobile networks. In this section some of the results are described. First, the impact of the trade-off between audio ad video quality on the overall transmission quality is addressed. Second, strategies of streaming adaptation when packet loss occurs are evaluated.

Test sequences for a subjective test were generated with the testbed. Audiovisual clips of approx. 9 s were presented to the test participants, and rated by them according to audiovisual, video and audio quality. These rates were then processed. The users were non-expert as they were not concerned with multimedia quality as part of their work.

Some conclusions can be extracted from these tests:

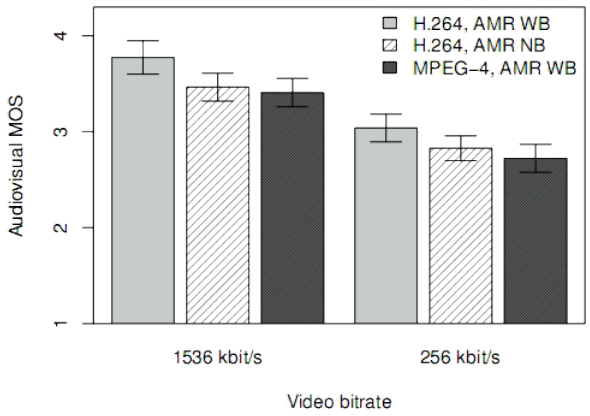


Figure 2.5: Comparison of the quality ratings collected for diverse audio and video streaming configuration in WiFi [14].

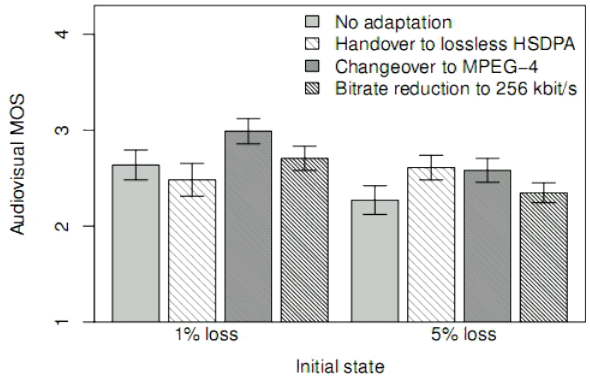


Figure 2.6: Comparison of the quality ratings collected when transmission affected by packet loss is adapted. Wideband speech encoding [14].

- Use of wideband codecs instead of narrowband speech codecs improves the QoE and the quality gain is slightly significant if the video quality is high.

- If lower video bit rate is used, the overall quality perception is degraded (as expected).

- If the video bit rate is reduced to counter the effect of packet loss, the quality is considerably improved. The network handover considerably degrades the user perception if the packet loss is low.

- If packet loss occurs, video codec changeover is the adaptation technique that always improves the user perception. However, if the packet loss is high, network handover is the most appropriate adaptation method.

2.4.1.6. Demonstrator

The way to visualize how the perceived quality is influenced by the configuration of audiovisual streaming was one of the requirements of the project; and, for this purpose, a demonstration GUI has been developed. The implementation of this GUI is in C++/Qt, based on Qt widgets available from Embedded Widgets Demo.

Figure 2.7: Screenshot of the demonstration interface [14].

The received video signal is displayed in the middle of the screen, surrounded by application control and monitoring widgets:

- Information about the streaming quality (on top of the screen). Visual indicators were used to inform about the correctness of the decoding process and the use of Packet Loss Concealment (PLC), flashing red in case of error.
- Information about the currently used bit rates and size of the jitter buffers.
- Audio and video codec switching.
- PLC or Voice Activity Detection (VAD) switch.
- Network information and switch.
- Audiovisual synchronization information.

These control widgets can be used anytime during the demonstration, and the resulting streaming quality can be observed in real time.

2.4.2. MultiRAT testbed

2.4.2.1. Introduction

Traditional research about mobile wireless networks is focused on ubiquitous access and large capacity. Nevertheless, energy save and environment protection demand attention, and current research efforts must be oriented towards an energy efficiency design [15]. Currently, a significant quantity of energy is wasted during periods of time when the demand of capacity is low. This excess could be avoided by implementing mechanisms for network reconfiguration based on load adaptation[16].

MultiRAT is a professional testbed suit including a number of nodes deploying different *Radio Access Technologies* (RAT). It is situated within the research frame of enabling optimal trade-off between provided QoE and energy consumption. MultiRAT is integrated in the ComGreen project, with emphasis on showing the potential energy save that could be attainable for heterogeneous networks. The opportunities for reduction of network energy consumption are related to energy aware network elements and systems, architectures, planning and operation. Energy consumption is proportional to offered resources in the network, therefore, one strategy is reduce energy consumption in networks with the same user QoE. MultiRAT testbed is focused on the investigation of heterogeneous access networks, relying on a mixture of different access technologies deployed indoors and outdoors [17].

The core of the testbed is the power management component, composed of: context manager, energy controller, control point and context collection agent [17].

2.4.2.2. Location

MultiRAT testbed is located in Berlin-Schöneberg, Germany, and nodes are distributed throughout the the building as well as outside of it, rather than in a laboratory setup.

The area covered by the MultiRAT network includes two trapezoidal and two rectangular courtyards, each with 400 - 900 square meters (figure 2.8. The whole area covered by the building is 9600 square meters, being the largest telecommunication building in Europe at the time of its construction in 1923-1929 [18].

As shown in Figure 2.9, MultiRAT testbed consists of nine nodes denoted as CAP or CMP. CAP nodes provide the clients with access to the network, CMP nodes are used for IP packets forwarding between specific CAP nodes. All nodes are connected to a mesh network by WLAN interfaces. In addition, a wired connection exists between each node and a common backhaul Ethernet network that enables any node to be used as a gateway (CGW1/CGW2). Beside the WLAN radio equipment, also two WiMAX capable nodes are deployed in the testbed. The arrows depict the main lobe direction of the antennas. Data transmission depending on the node configuration can take place through paths with up to six wireless hops between a node and a gateway. The WiMAX link at CAP9 can also be used for connection between the nodes and the backbone via a possible gateway situated with CAP9 [18].

Figure 2.8: Extract of Berlin city map showing the location environment in Berlin Shöneberg [18].

Various servers are available for different purposes. They provide an interface for virtualization and are installed in the backbone Ethernet network. A central management server enables the management of all the nodes in the MultiRAT testbed. Moreover, a *Virtual Private Network* (VPN) gateway, installed at the Ethernet network provides access to the testbed for remote users.

2.4.2.3. Planned extensions

1. *Measurements of energy consumption:*

   The capability to measure energy consumption in the testbed has a clear importance, with the purpose of implementing such functionality, an additional IP based switch with an integrated electrical power measurement module is intended. Every electrical connection of nodes would be equipped with this IP power switch. These measurements would be remotely controlled; additionally, it would be also remotely possible to switch on/off access nodes. According to these characteristics, a central monitor of energy consumption by all access nodes in the testbed would be implemented [18].

2. *Heterogeneous access*

   The future plan was to rely on three different access technologies, 2G, 3G and WLAN, available for users. Being 2G a basic coverage technology while 3G and WLAN would be used for broadband connection of mobile terminals.

3. *Planned demo scenarios: load-adaptive dynamic reconfiguration of access nodes*

   Access nodes of different access technologies could be switched on/off depending on actual traffic demand in the network. There would exist two function-

alities at every node: enabling context information gathering from the node and controlling state of the node. The final goal would be to achieve a network able to keep active only those resources that are needed, reducing the power consumption by switching off unused hardware. Likewise, no users in the network become off-line, since 2G access node would be guaranteed to be used by any user at any time.

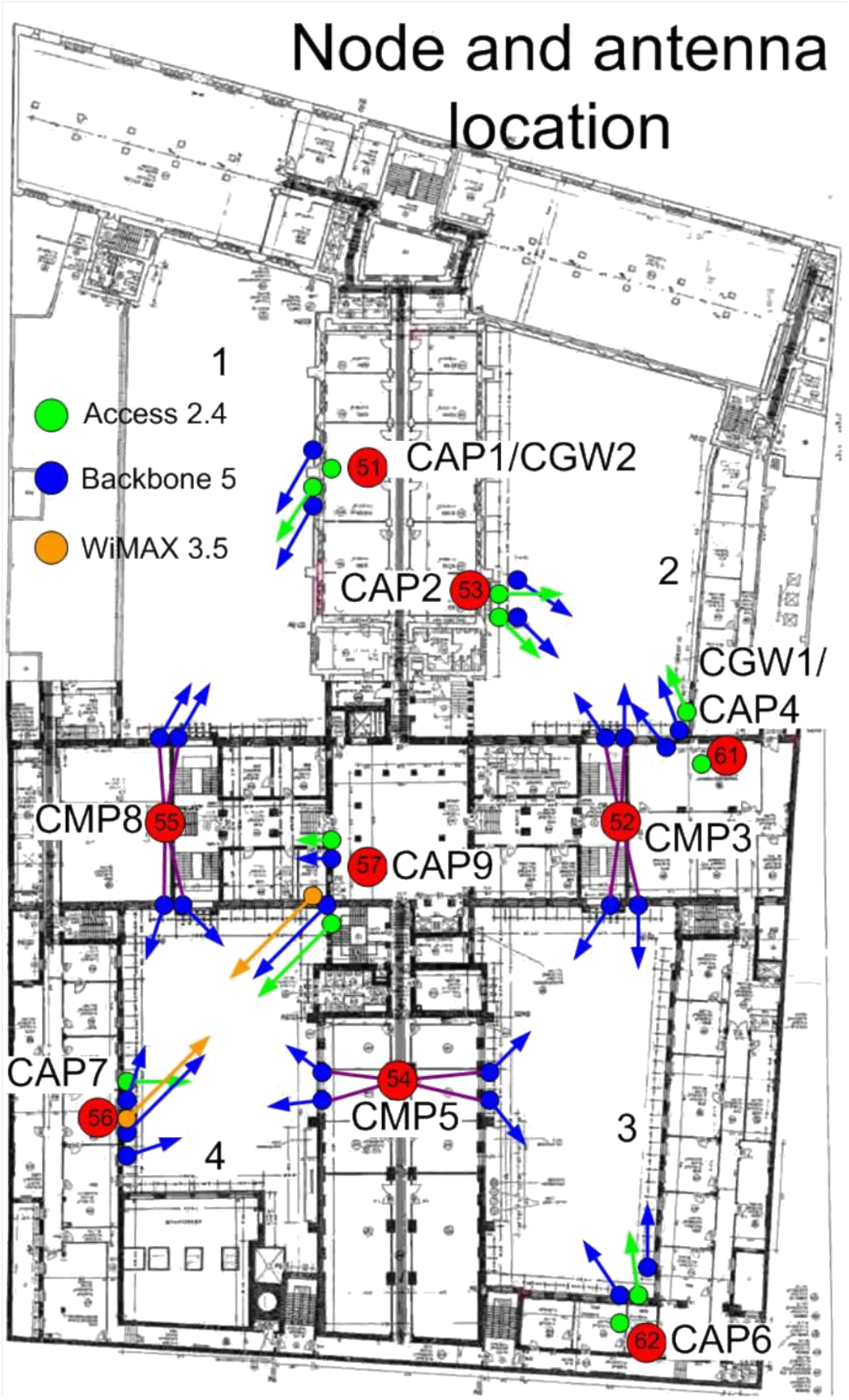


Figure 2.9: Sketch of deployed MultiRAT testbed architecture with the location of installed nodes and antenna directions [18].

Chapter 3

New testbed design

To search for the old is to understand the new.
The old, the new,
this is a matter of time.
In all things man must have a clear mind.
The Way:
who will pass it on straight and well?

(Gichin Funakoshi)

As already explained, this contribution was born in the research direction of QoE enhancement in nomadic users, together with optimization of used resources in the intelligent networks of future. The way to work on that purpose was by creating a prototype of a mobility management system, with focus on: QoE analysis, user's perception of quality adaptation process, quantification of perceptual effects and quality prediction; and adding the value of real and varied network scenario and QoS adaptation.

The new scenario of the project consisted on a video-chat communication between two clients (one in the building at Erns-Reuter-Platz, one in the building at WinterfeldStraße) with focus on the quality (QoS and QoE). Moreover, a new entity appeared: a traffic controller (mobile) unit, conceived to allow an expert user to influence some parameters in the network. The aim was to control different network impairments and enable to activate both QoE and QoS adaptation to test and demon-

strate their importance.

This is probably the most representative chapter, since it contains the main contribution presented in this document, which is the design and implementation (C/C++ and Qt) of a protocol to control the network parameters (*Network Parameters Control Protocol* (NPCP)) -this is, the language spoken between the clients with the traffic controller- as well as a suitable GUI for the demonstrator.

Below, a summary of the new requirements for the extension of the testbed can be found, continuing with a description of the interfaces and software design and programming.

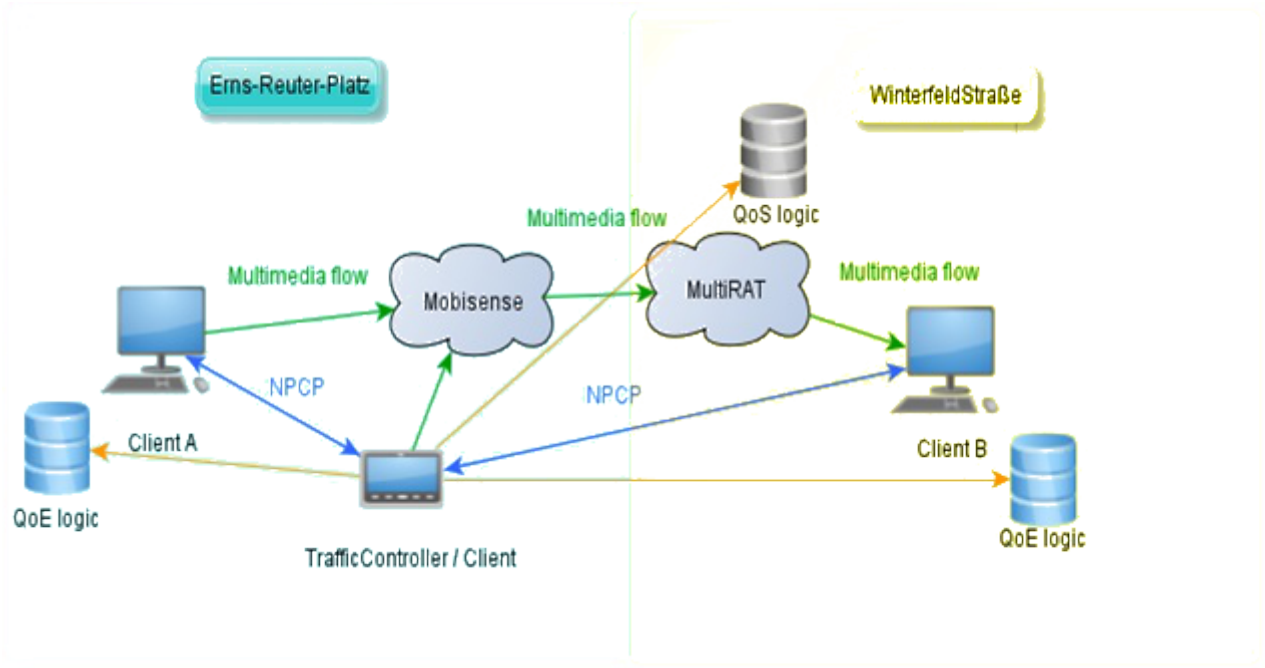


Figure 3.1: New scenario for the testbed integration

3.1. New requeriments

3.1.1. Mobisense-MultiRAT testbeds interaction

The aim of the Mobisense/Multi-RAT integration was to accomplish the Mobisense traffic being sent over a real network. The MultiRAT testbed offered the possibility to do this over various paths with different technologies. Among them, there were implemented and available: WLAN 802.11a/b/g, WiMAX, Ethernet, SoftToken. Being planned in future 2G, 3G and 4G technologies. In any case, in its status at that time, the possibility of routes through the Multi-RAT network were clearly relevant for Mobisense, in order to act in a more real scenario.

The technologies operating for Mobisense through the Multi-RAT network are which follow:

1. *Ethernet*
   Ethernet topology can be considered as a benchmark which does not use any wireless technology to route to the traffic through the Multi-RAT testbed but represents a pure wired connection.

2. *WLAN* and *WiMAX*
   Both topologies use a single WiFi/WiMAX link to route the traffic through the Multi-RAT testbed.

3. Mesh
   Mesh topology uses a multi-hop path, consisting on heterogeneous technologies to route the traffic through the Multi-RAT testbed

3.2. Network architecture

The setup consisted of two subnets: local Mobisense network (192.168.15.0/24), at *Erns-Reuter-Platz* (ERP) and the remote MultiRAT network (192.168.7.0/24) at *WinterfeldStraße* (WFD). Mobisense had a demonstration set up in a room for such purposes, it was made up of two desktop PCs and potentially some mobile clients such as tablet PCs/laptops. These devices were connected to the MultiRAT testbed in WFD using VPN and caracal as a gateway. The desktop PC directly connected to the plug acted as a bridge for the other desktop PC and for mobile clients by acting as an AP named *mobisense-ap*. Since the routes connecting these networks included video calls purpose, bitrates up to 2MB/s in both directions were expected, for instance, during short periods of demonstration.

The traffic to WFD was routed via the router/gateway caracal, which was being operated by the *Berlin Open Wireless Lab* (BOWL) team. Every device in the Mobisense network must have a route to the MultiRAT network, which needed to be set up via the gateway 192.168.15.1.

Client B at WFD was running a *Virtual Network Computing* (VNC) server which enabled everything on its screen to be displayed on any other PC. Therefore, VNC client on the Corresponding Node may be started to display the screen of B, while it was still located and running in WFD. The switching of the physical network (WiFi, WiMAX, Ethernet ...) worked through the remote Router Controller equipment (192.168.7.105), at WFD, which controlled the path of the packet stream within the MultiRAT testbed. Scripts on Router Controller were invoked by the testbed controller, which set routes on the nodes in the MultiRAT testbed.

During a demonstration, the routes through the networks could be selected by some buttons on the Traffic Controller's GUI. The election would be done among the following options:

- *Ethernet*: direct link to the client without going over a wireless network. A sort of benchmark.
- *WiMAX*: one-hop WiMax link.
- *WLAN*: one-hop WLAN link.
- *Mesh*: multi-hop WLAN links.
- *2G*: one-hop *Enhanced Data Rates for GSM Evolution* (EDGE) link [1].

[1] EDGE still under installation process at that time.

Hence, a different button for each route[2] should exist in the Traffic Controller's GUI. A configuration script was implemented in the remote network (MultiRAT), located at the Routing Controller unit, which could be executed from the Mobisense network in order to set the different routes. The usage was very simple, required the topology parameter.

Regarding to accessing the VNC, there was two ways to connect to the network: on one hand, by employing the available link in ERP to WFD, which was used to connect the two clients. On the other hand, connecting via VPN.

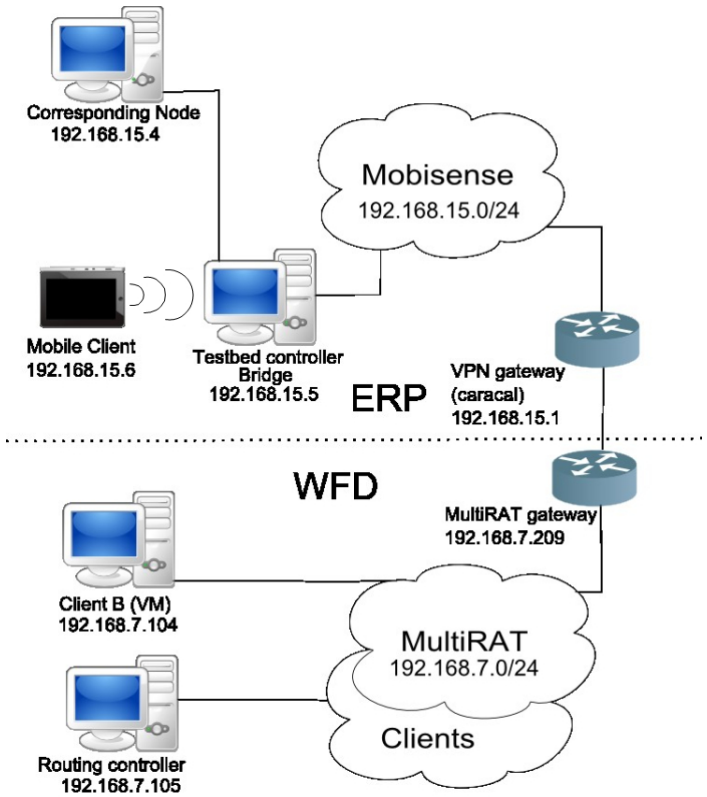


Figure 3.2: Mobisense-MultiRAT integration: network topology

3.3. Network Parameters Control Protocol (NPCP)

Regard this new scenario, both testbeds would act in order to maximize the quality when packet loss occurs. The two clients could be having a multimedia session and, in parallel, another communication between the clients taking place with the TC. NPCP is the protocol designed to support the communication between the client nodes and the TC node.

[2]These route buttons did not become operative at the end of the internship. Nevertheless, the integration of this functionality in the GUI was prepared, and no much effort should be needed for such task.

The clients receive, from the traffic controller unit, NPCP-messages with orders to change parameters or artificially alter the conditions of the network: packet losses, audio/video codecs, transmission rate, packetization, etc. Likewise, the traffic controller obtains messages about whether the ordered operation was successful or not. These changes on the network parameters make possible to test some algorithms meant to act in order to maximize the quality once these changes have occurred.

3.3.1. NPCP messages and example of session

3.3.2. Types of messages and example of NPCP session

Below is a general scheme of the possible messages there can be in NPCP:

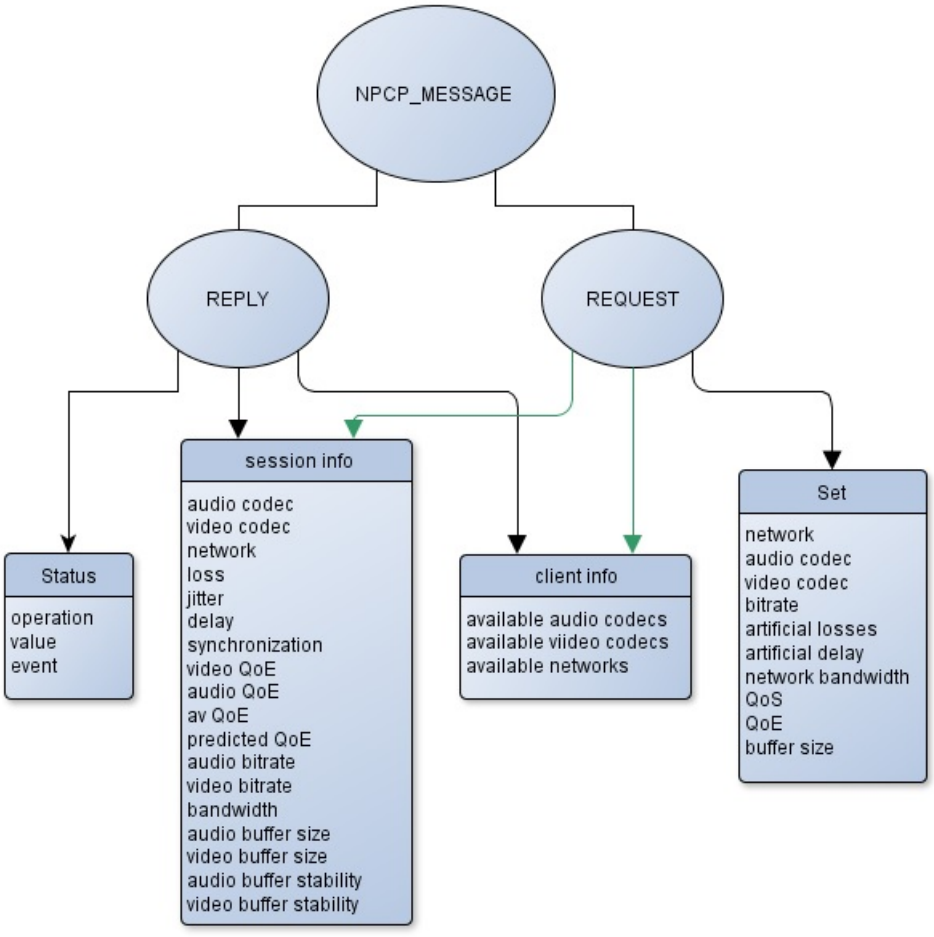


Figure 3.3: Types of NPCP messages

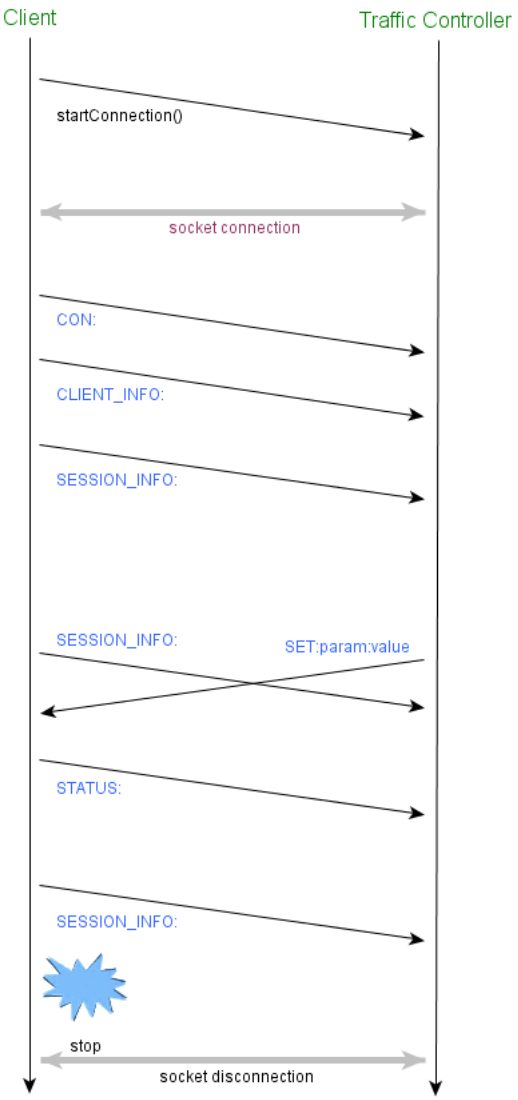
Actually, 'session info' and 'client info' requests were not implemented, but the client sent the information itself -directly- to the TC without any previous request. Therefore, the communication between the TC with the clients, according to this scheme, works as follows:

Client ⇒ Traffic Controller:

- Session information message: sent every few seconds (8 seconds, for consistency with PJVIDEO). The purpose is to enable TC to track the networks parameters evolution, displaying it on the statistics widget for observation by an expert user.
- Client information: sent at the beginning of the session. The information about available networks must be refreshed also periodically, given its relevance, since the availability of routes may change.
- Status message: sent to the TC after receiving a 'set instruction', informing about the success or not with carrying out the operation.

Traffic Controller ⇒ Client:

- Set instructions: sent when the expert user decides to change any characteristic in the network: additional losses, delay; codec change; network change; bandwidth; jitter buffers size; etc.



3.3.3. NPCP syntax

As already seen above, there exist four types of NPCP messages: *status*, *session information*, *client information* (replays) and *set* (request). These messages contain an identifier and a set of parameters. The identifier differentiates the type of message: "SESSION_INFO", "CLIENT_INFO", "CON", "MSGFROM", "SET", "STATUS", etc. "CON" and "MSGFROM" identifiers correspond to a client's acknowledgement and a user's chat message at a client respectively.

The format of *session information* messages is as shown below: identifier ("SESSION_INFO"), followed by the list of parameters depicted on Figure 3.3

<id>:$param_1$#$param_2$#...#$param_k$#\n\r

In the case of *client information* messages, the identifier is followed by the lists of available audio and video codec; the respective indices related to the current codec in each case; the list of available networks as well as the index of the current network in use. Token '#' is used for delimitation between lists and token '*' to mark out the different elements within a list.

<id>:audio_codec$_1$*...*audio_codec$_m$*#audio_codec_index*#video_codec$_1$*...*video_codec$_k$*#video_codec_index#network$_1$*...*netowrk$_p$*#netowrk_index#\n\r

Finally, the *set* messages consist on the identifier (using ":" to select a subtype) followed by the value which is intended to be set and the end message delimiter: "\n \r"

<id>:value\n\r

As can be appreciated, this is a very basic signaling message syntax. In further steps, a XML markup language-based implementation was intended, in order to attain a more robust and flexible mechanism, without affecting simplicity.

3.4. Interface design

3.4.1. Client Interface

Continuing with the requirement of visualization in the initial *Mobisense goes video* testbed, a demonstrator GUI was developed in order to appreciate how the perceived quality is influenced by the configuration of audiovisual streaming. In Figure 3.4 a screen shot of the interface is presented. Appearance is very simple, with the received video signal shown in the middle of the screen, together with a couple of buttons to start/stop the session. Additionally, for an expert user, the view of statistics of relevant parameters information can be enabled. These characteristics are the jitter buffer size, codec and bit rate for audio and video, as well as the network which is operating, losses and delay of the transmission and synchronization. They all are displayed on the left side. Moreover, the demonstrator interface is fitted out with a widget for written chat, which also resulted useful in terms of testing and debugging.

3.4.2. Traffic Controller Interface

The TC interface is where all the support for parameters setting and monitoring is given. On the central band, there are two graphs, each one for one of the two nodes: ERP and WFD, where the statistics are plotted. The selection of the

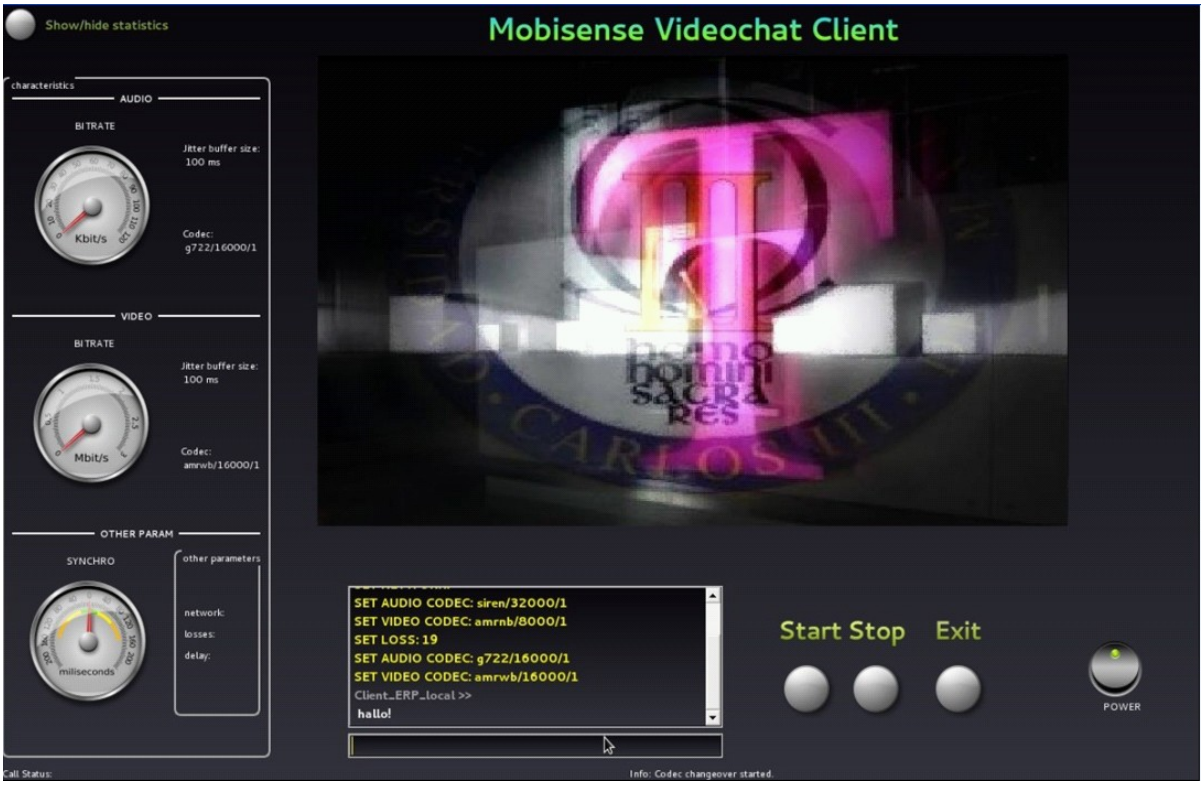


Figure 3.4: Client GUI

desired parameter to be represented is made by pressing the corresponding button in the middle of the screen: video/audio QoE, jitter, losses, delay, audio buffer size/stability, etc.

On the upper band, two control blocks can be found. In the first one, in the left side, the names of the currently in use codecs are displayed, as well as a set of buttons to change over. In the second block, in the right side, the name of the network to which the client is currently connected is displayed. Moreover, additional widgets for network interface switching and for artificial impairment by packet loss, delay, bit rate or delay deviation are available. The central box in this upper band was no longer needed, but meant to be replaced by a QoE gauge, which would be based on the results of the conducted tests.

At the bottom of the screen, information about the currently used jitter buffer sizes and bit rates are located. The first box, in the left, corresponds to the node at ERP and the second one, in the right side, to the node at WFD. In the middle, the events are being shown as they are registered. In the last box, two buttons for (de)activation of QoS/QoE logics can be found. Finally, one more button start-s/stops the NPCP session.

All these control widgets can be used at any time during the demonstration, being possible to observe the resulting effects in real time, with especial interest on the streaming quality.

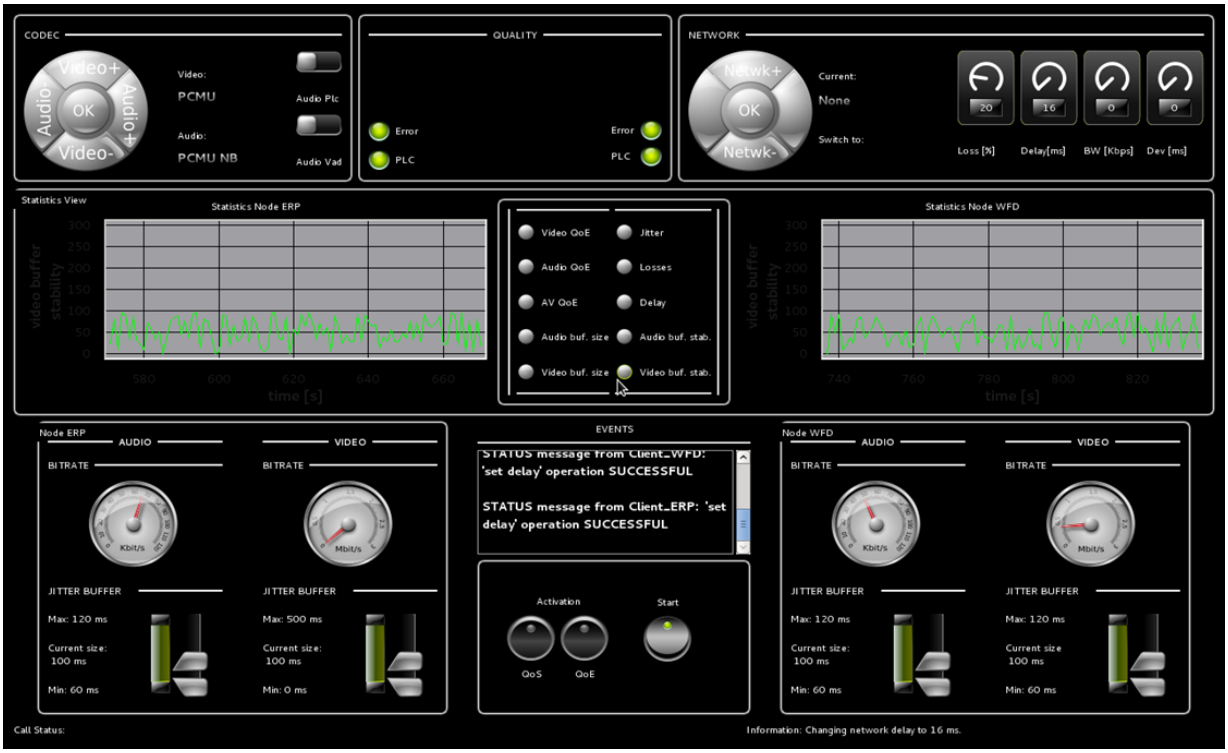


Figure 3.5: Traffic controller GUI

3.5. Software design

In this section, the software design of this extension for the testbed is presented. The starting point will be the new entity: TC, which is the main character in the NPCP scenario, continuing with the client. Only the aspects involved in NPCP and communication with TC will be commented, since the multimedia part belongs to *Mobisense goes video* and was already implemented, falling its intricacy out of the scope of this section, as it was reviewed in 2.4.1.

3.5.1. Traffic Controller

A detailed description of the TC module lies below, beginning with a glance of the blocks scheme and files and directories which form the structure of TC application. Secondly, a diagram of the classes is shown, emphasizing on the Qt signal-slot relation between the objects. Finally, different flow charts will help clarify the program functioning.

3.5.1.1. Blocks diagram

Figure 3.6 shows an overview of the TC, where a modular scheme is followed. The core is composed of NPCP module, which implements the functionality of the protocol; the Socket module, where the connection is managed; and Graphics mod-

ule, which supports statistics plotting. The rest of blocks are in charge of traffic and GUI control.

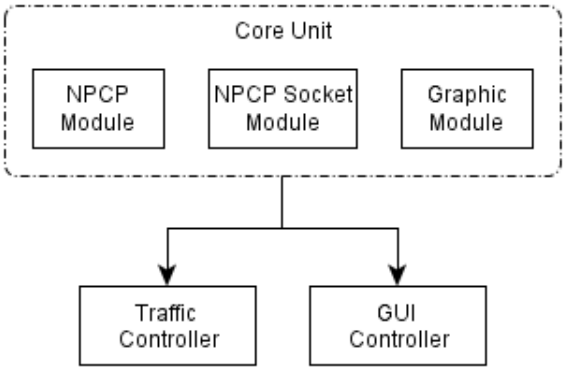


Figure 3.6: Building Blocks of the Traffic Controller

3.5.1.2. Directory tree and files structure

Below, the different directories are described, followed by figure 3.7, where the different files contents and purposes are explained. The three packages which make up this application are:

1. ./socket

   Contains all the socket functionality for Network Parameters Control Protocol (NPCP) which supports the connection with the PJVIDEO client.

2. ./graphics

   Contains the functionality for the statistics plot, which shows relevant information related to parameters' history.

3. ./tcg

   Contains the GUI implementation

```
./
├── socket
│   ├── npcp.cpp
│   │   Contains the implementation of the structures
│   │   which are used to carry the data in NPCP
│   │   sessions.
│   │
│   ├── npcp.h
│   ├── settings.h
│   │   Contains constants definitions.
│   │
│   ├── tcpconnection.cpp
│   │   Contains the implementation for all the
│   │   functionality related to decode and interpret
│   │   the NPCP messages coming from a specific
│   │   client.
│   │
│   ├── tcpconnection.h
│   │
│   ├── trafficcontroller.cpp
│   │   From this file the connections are managed
│   │   as well as their interaction with the GUI and
│   │   graphs.
│   │
│   └── trafficcontroller.h
│
├── graphics
│   ├── plots.c
│   ├── plots.h
│   │   Contains the constants for plots functionality
│   │   and the definition of the structs used to
│   │   handle the different types of plot, as well as
│   │   the necessary functions for their use.
│   │
│   ├── statisticsplot.cpp
│   │
│   └── statisticsplot.h
│       Contains the functions to -using
│       textitplots.h/plots.c support- properly
│       plot in a graph the parameters from
│       the corresponding client (mapped by a
│       TCConnection).
│
└── tcg
    ├── tcgcontroller.cpp
    │
    └── tcgcontroller.h
```

Figure 3.7: Directory tree of the TC application

3.5.1.3. Classes diagram

1. ***GUI-NPCP-Socket interaction***

   Figure 3.8 shows a big picture representation of the classes which take part in this system and the relations between them. TC GUI Controller *(TCGController)* class represents the highest abstraction level, there exists one instance of this class and deals directly with the GUI. Continuing in a descending level of abstraction, TrafficController class is next. Also from this class, only one instance is created, being in charge of the NPCP mechanisms management, the role of this object is also to be an intermediary for socket and graphics with the GUI. Finally, first level of abstraction comes with the class TCConnection, with as many instances as clients are connected. Therefore, one instance for the client at ERP and one more for the other at WFD; their mission is to deal directly with the socket (QTcpSocket).

   A couple of flows have been highlighted in the picture. On one side, in green color, a descending flow goes from high abstraction to lower. It shows how the orders from the expert user -through the control widgets- are being translated into basic orders: first, a NPCP message is built and thereafter sent through a socket. On the other side, in blue color, another flow goes the other way round, showing how basic data from the client comes through the socket resulting in a NPCP message which is decoded and properly processed.

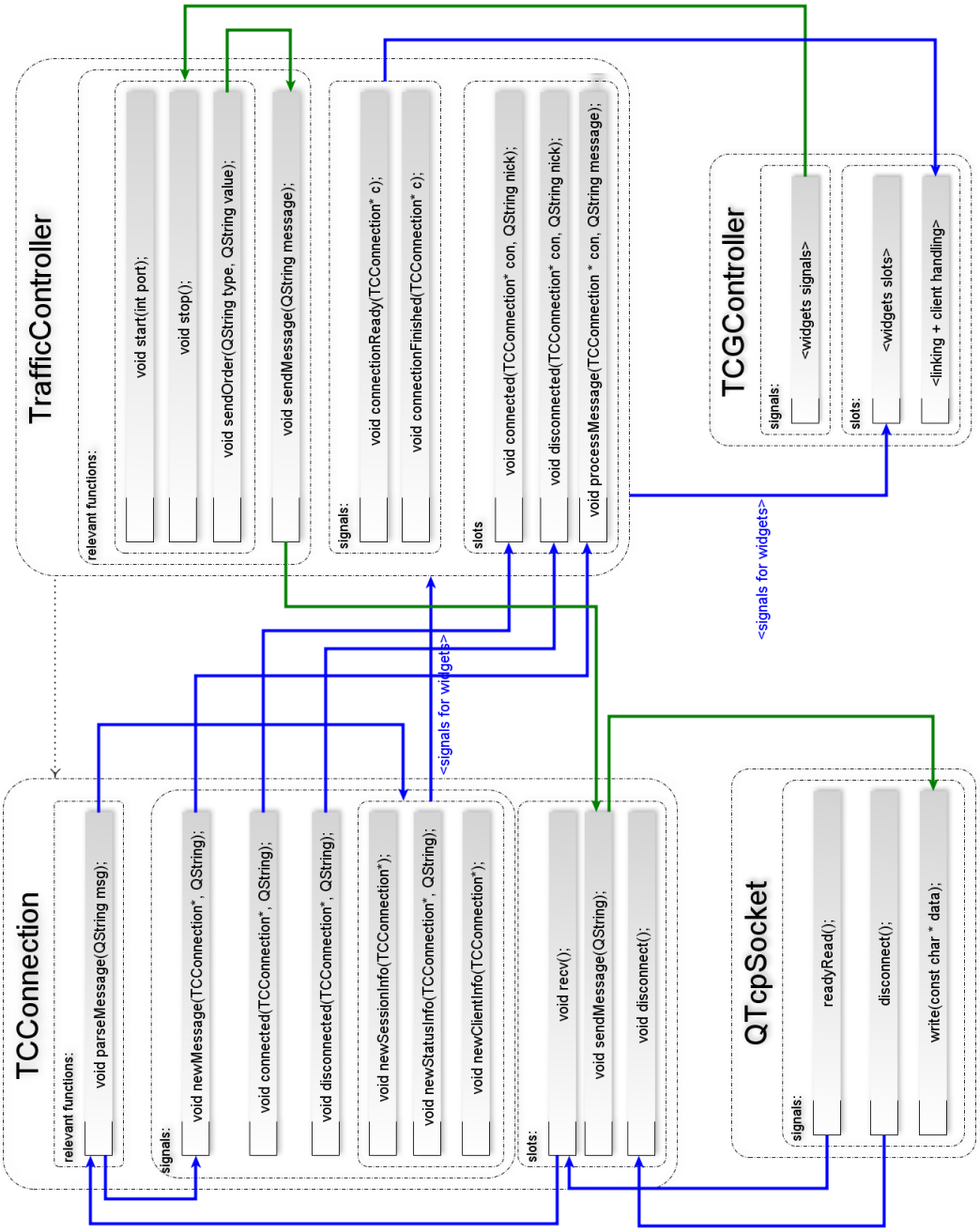


Figure 3.8: Classes diagram of the TC application

Below, sending and data reception are depicted in figures 3.9 and 3.10 respectively.

Orders are sent whenever a *set* operation is requested by the user by pushing the corresponding control widgets. These operations are managed by TrafficController class, where the appropriate NPCP messages are built and delegated on TCConnection to be sent.

On the other hand, when a message comes from a client, the corresponding TCConnection is notified by the socket. The message is parsed and passed to TrafficController to be processed. If, when being parsed, the message is identified as session, client or status information then it is managed directly by widgets slots at TCGController. This link is performed in the end of connection process (see figure 3.12) within the function for widgets linking with TCConnection, this is why in figure 3.10, the arrow crosses TrafficController, since the referencing is done through this class.

Connection and disconnection operations are explained in 3.5.1.4, where a flow chart is depicted.

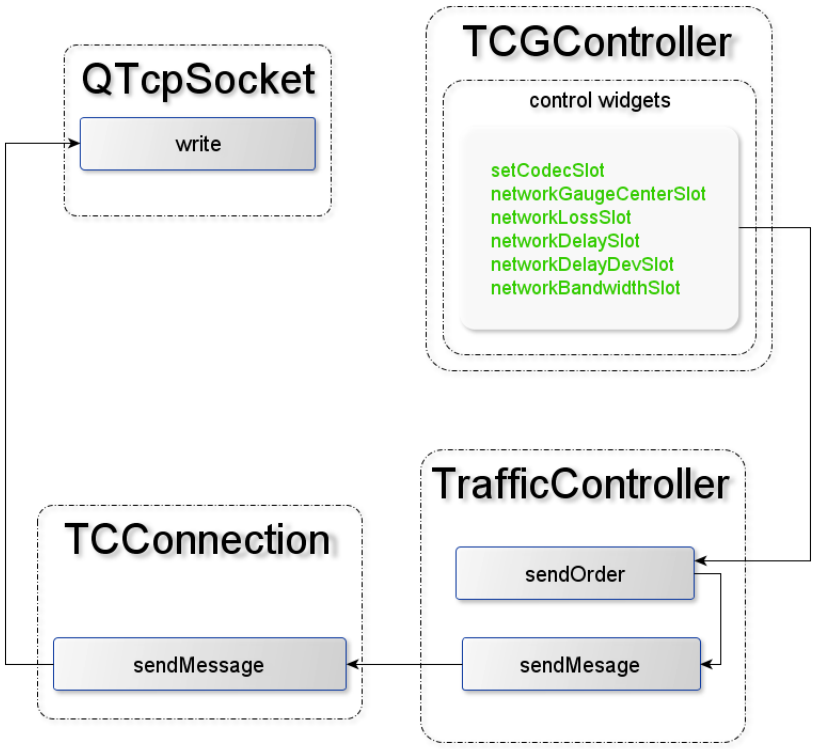


Figure 3.9: Data sending at TC side

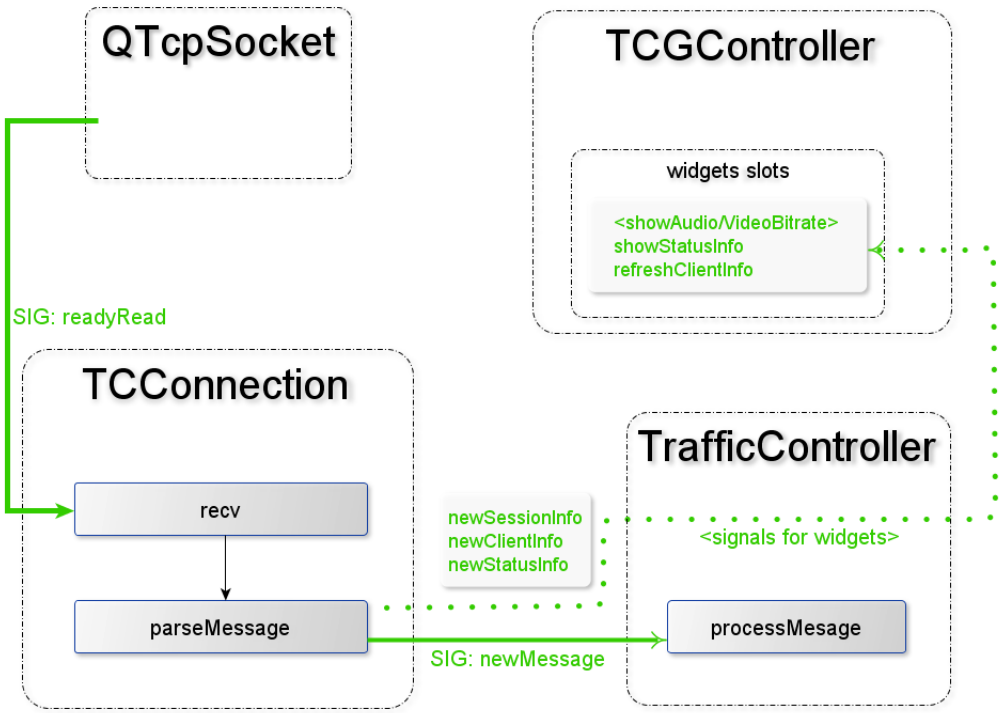


Figure 3.10: Data reception at TC side

2. ***Graphics functionality***

As seen in figure 3.5, TC GUI relies on two frames for statistics view. QwtPlot objects, from Qwt library, are used to depict the samples collected from the clients. As the samples are arriving, their value is plotted with every *newSessionInfo()* event notified from *TCConnection*; once the total width of the plot has been filled, the horizontal axis starts shifting with every data update. The desrired parameter to be depicted is selected with the buttons lying between the plots.

The main element for the statistics implementation is the class *Statisticsplot*, two instances of this class are responsible of parameters plot selection and data update. Plot selection entails activation of axes corresponding to desired parameter and deactivation of the others, this occurs after every parameter selection on the buttons. Data update takes place, as mentioned, after every *newSessionInfo()* event notified from *TCConnection*. Specific tasks as parameters mapping, color customization, and data structures management are stated at textitplots.h and textitplots.cpp files. Figure 3.11 shows graphically the above explained.

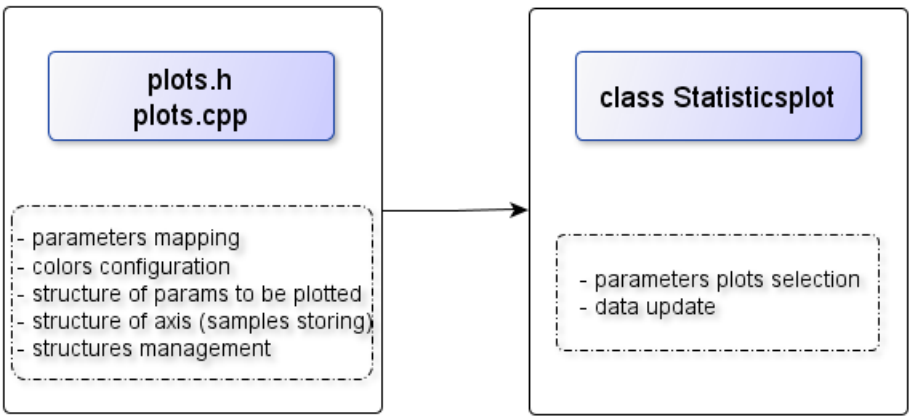


Figure 3.11: Graphics scheme

3.5.1.4. Flow charts

In Figure 3.12 it is shown a general scheme of the TC initialization. When the start/stop toggle button is switched, the *TC GUI Controller (TCGController)* decides by checking the boolean variable of the button which is the instruction to carry out. When starting, TC is booted by initiating the socket, if no error occurs, the socket keeps listening for clients wanting to connect.

When an event of new connection takes place, this connection is stored into a list by the TC. At this point, through this connection (TCConnection object), three types of events can arrive: a disconnection; a message, which would be decoded and processed; and the acknowledgement from the client informing that everything is fine by its side so the NPCP session can begin. As soon as this acknowledgement occurs, the TC emits a signal for the GUI controller and the last initialization operations are carried out, which are: linking of events, widgets and plot on one hand, and new client handling on the other hand. The reason of this linking is to establish a correspondence between widgets and graphs with the client, depending on whether it is the one in ERP or the one in WFD.

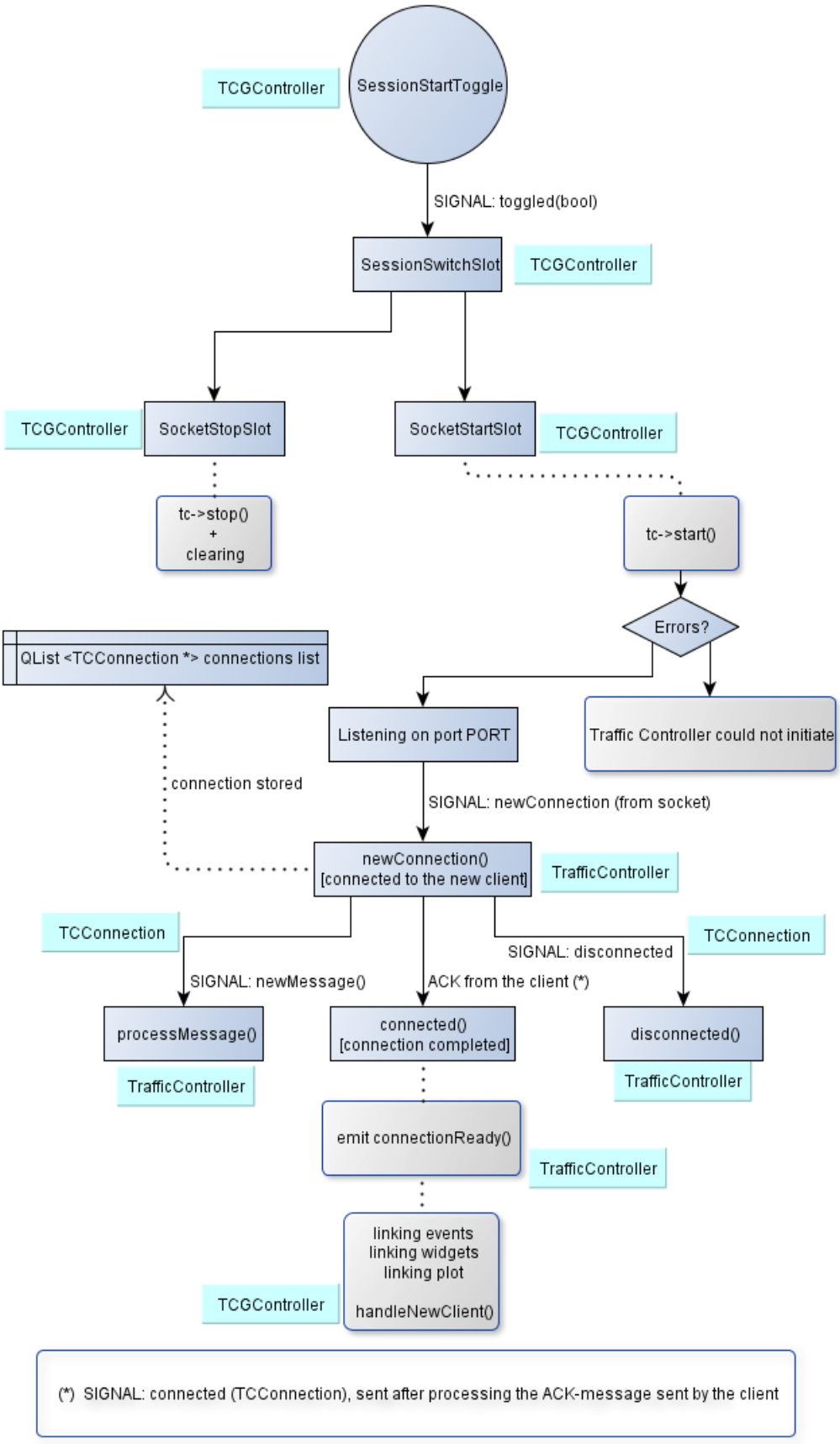


Figure 3.12: Traffic Controller initialization scheme

3.5.2. Client

3.5.2.1. Blocks diagram

The new components added to the client application are depicted on figure 3.13. As in the TC, a module is in charge of NPCP implementation, supported by two additional modules: one for socket connection management and one for timer operations, required for periodical sending of information. Two more blocks complete the scheme: NPCP controller, conducting the protocol operations and GUI Controller, to act on the widgets.

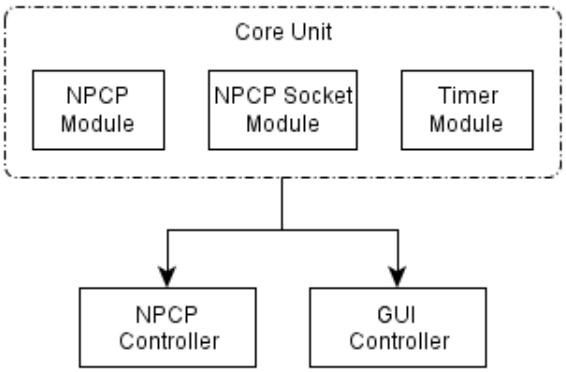


Figure 3.13: Building Blocks of the Client

3.5.2.2. Directory tree and files structure

Below, the different directories are described, followed by figure 3.7, where the relevant files contents and purposes are explained. The folders interesting to be examined for the present case are:

1. pjvideo/third_party/socket

   Contains all the socket functionality for Network Parameters Control Protocol (NPCP) which supports the connection with the TC.

2. pjvideo/third_party/gui/

   Contains GUI and GUI controller implementation.

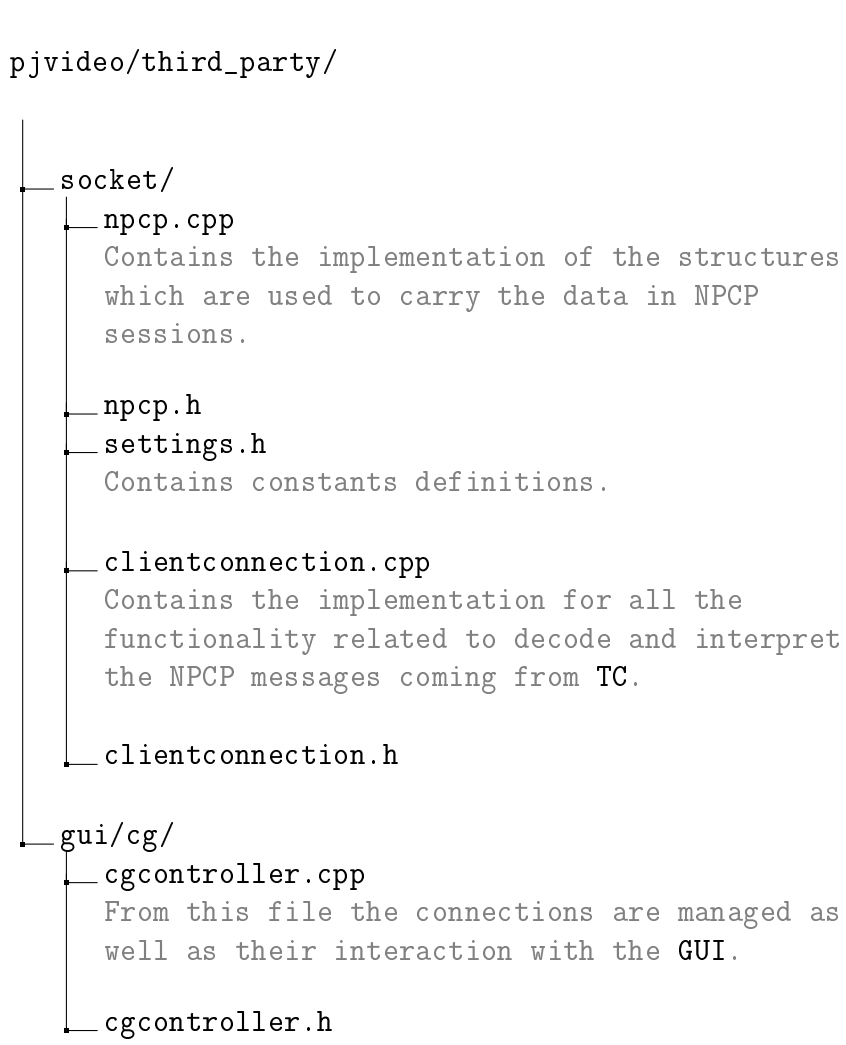


Figure 3.14: Directory tree of the Client application

3.5.2.3. Classes diagram

Within client application, five types of purposes can be distinguished: connection establishment and ending, session and client information sending, data reception and socket error at client side.

Figure 3.15 depicts the connection establishment. After the start button is pressed by the user, the resulting signal is directed in CGController class to the start slot, where the request is passed to a lower level, represented by ClientConnection class. This class attempts the connection to TC by invoking connectToHost() function at socket, when "connected" signal is received at ClientConnection it is notified to CGController, where the connection is considered correctly established. Moreover, the timer is started for periodic information sending control.

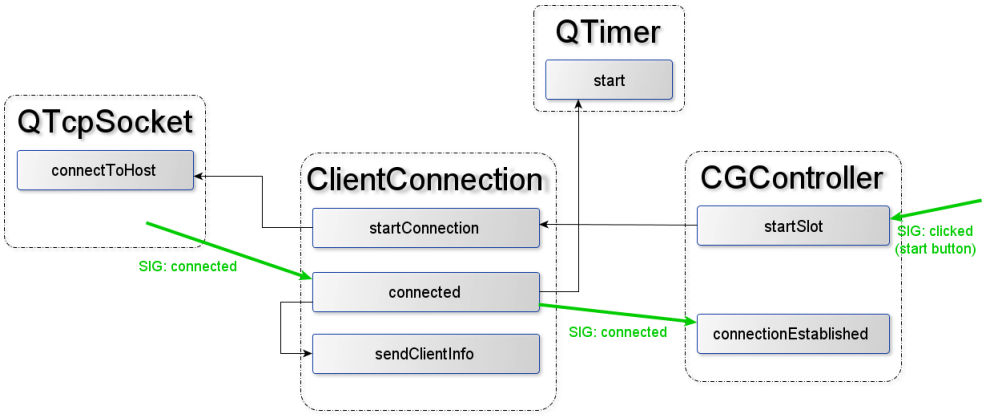


Figure 3.15: NPCP connection establishment: client side

On the other hand, connection ending is carried out in a similar process. When stop button is pressed by user, a descending sequence in level of abstraction, transfers the signal till the socket, which is closed, and the timer, which is stopped. Finally, the GUI receives back the notification of the disconnection completed. The scheme can be observed on figure 3.16.

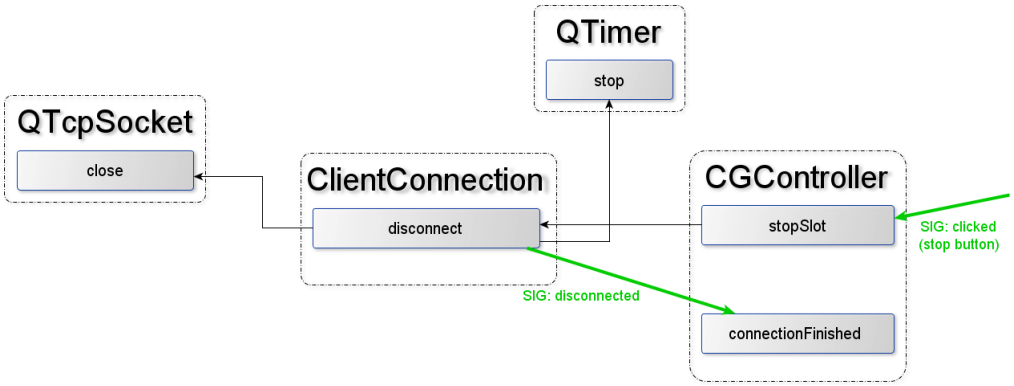


Figure 3.16: NPCP connection ending: client side

The data sending takes place periodically, as mention before, to keep TC informed of the network statistics as well as the availability of network interfaces. The timer regulates this sending, when timeout signal is caught at ClientConnection two signals are sent to the GUI controller to request session and client information to be updated. When the operation ends, two signals are sent back to ClientConnection, from where the information is formatted in an NPCP message and passed to the socket to be send for the TC. The process is described on figure 3.17.

Data reception, depicted on figure 3.18 takes place the other way round. A signal is sent from the socket to ClientConnection announcing data ready to be read. At ClientConnection the message is parsed and interpreted, resulting in the appropriate signal for the GUI controller, depending on the message sent by the TC, whether

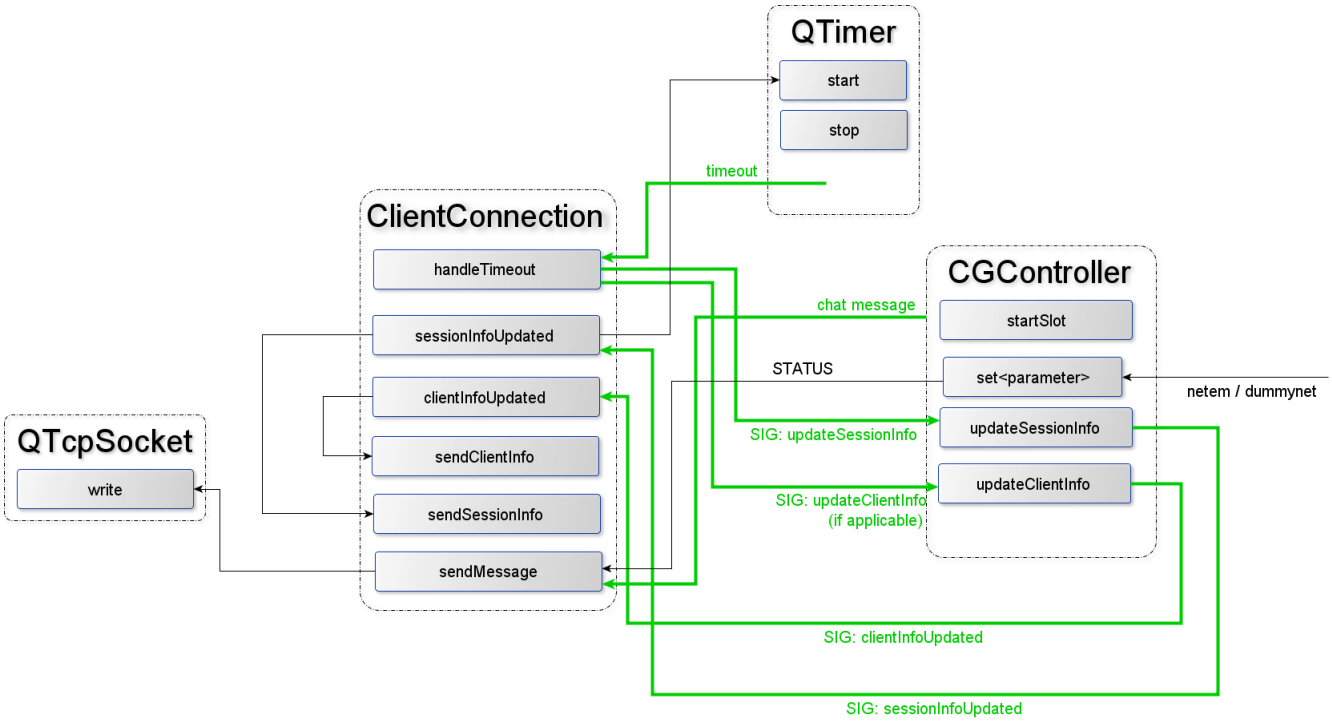


Figure 3.17: Data sending at client side

set instruction or text message.

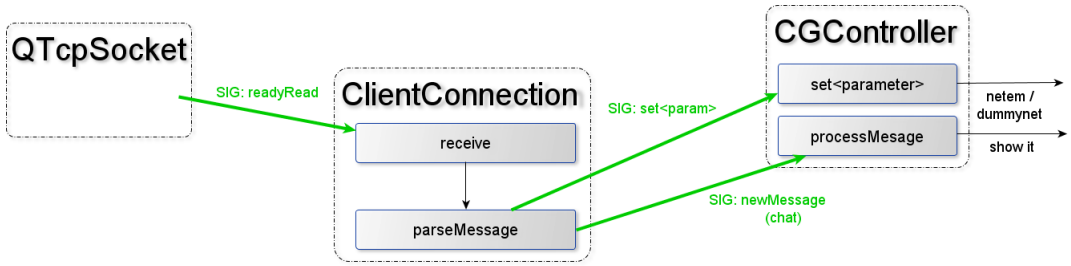


Figure 3.18: Data reception at client side

Finally, figure 3.19 schematizes the socket errors reporting mechanism. Two signals can be thrown by the socket: error or disconnection. In both cases, ClientConnection directs towards the GUI controller, which becomes in erroneous connection or connection finished state.

For a global and detailed glance of the functionality, figure 3.20 gathers up all the signal/slot mechanisms related to NPCP communication.

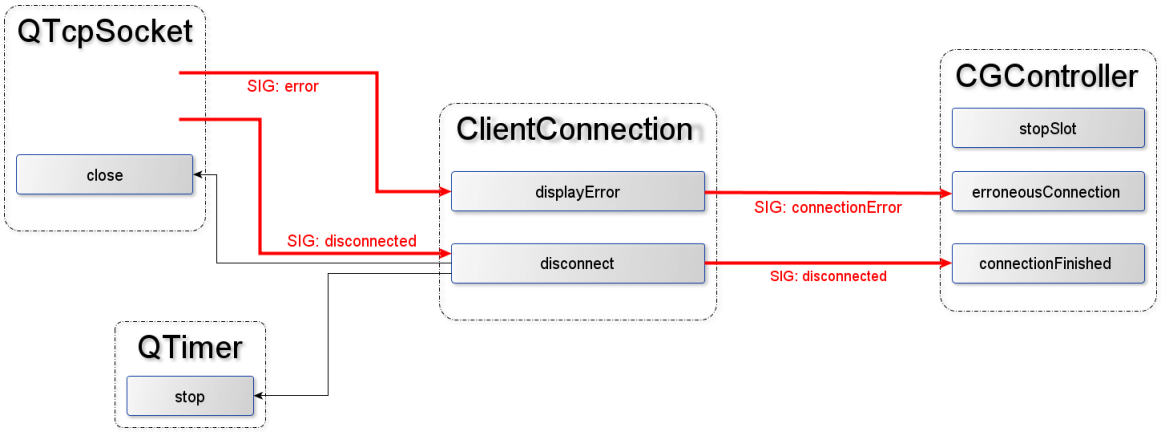


Figure 3.19: Socket error at client side

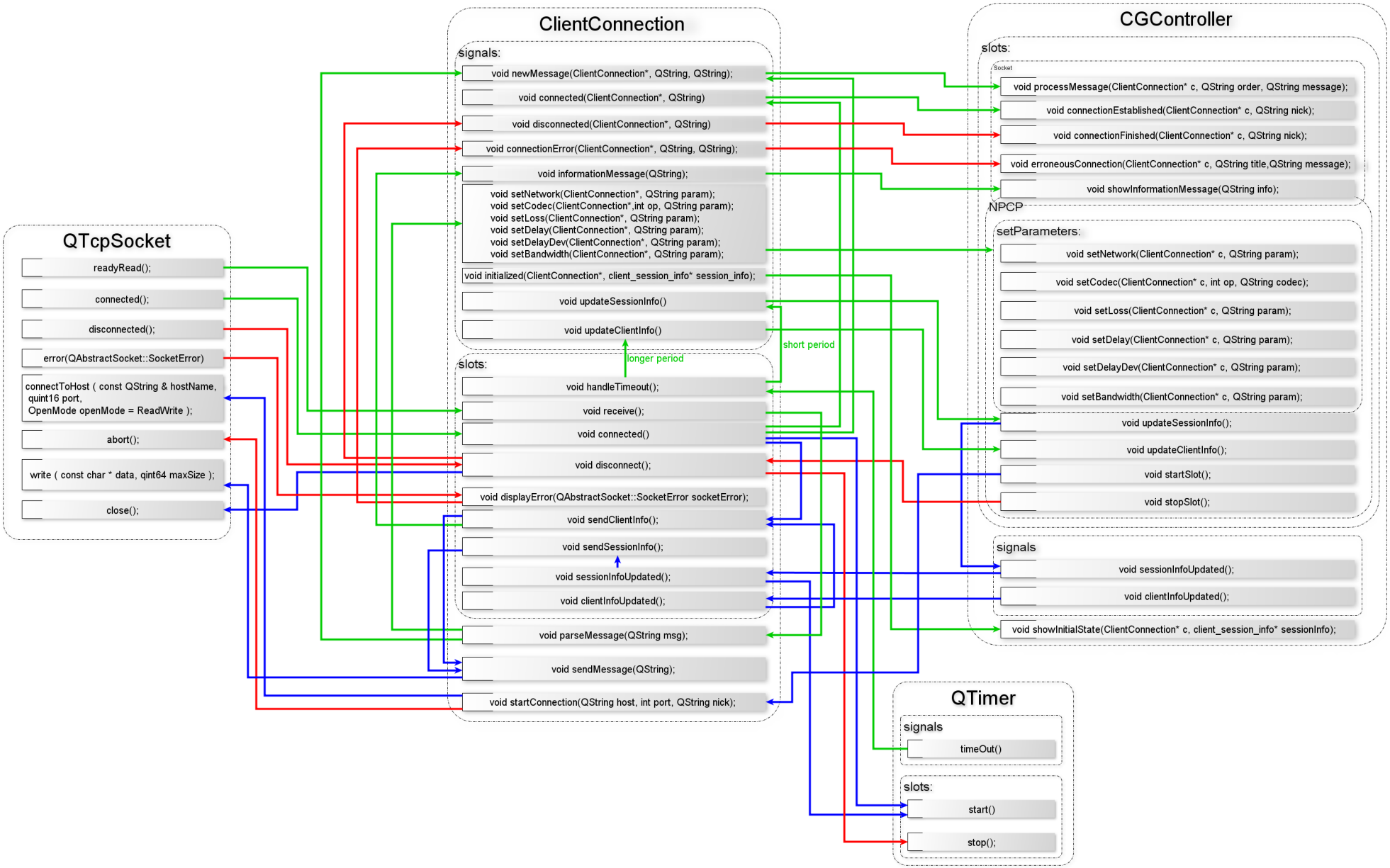


Figure 3.20: General classes diagram of client application

3.5.2.4. Flow charts

In the case of the client, the initialization mechanism presents two dimensions: the multimedia session on one side, and the NPCP session on the other. Stop and start routines are depicted on Figure 3.21. When pressing 'start' button, the *Client GUI Controller (CGController)* initiates the multimedia session and the socket is connected to the TC on the appropriate port. The socket management is done by using a ClientConnection object. As soon as the sockets notifies the successfully establishment of the connection, an acknowledgement is sent to the TC and a signa notifies the main window (GUI controller) to prepare the widgets and show a message informing about the connection successfully completed. On the other hand, when 'stop' button is pushed, the socket is disconnected -again through the ClientConnection object- and a signal is sent to the GUI Controller to display a message.

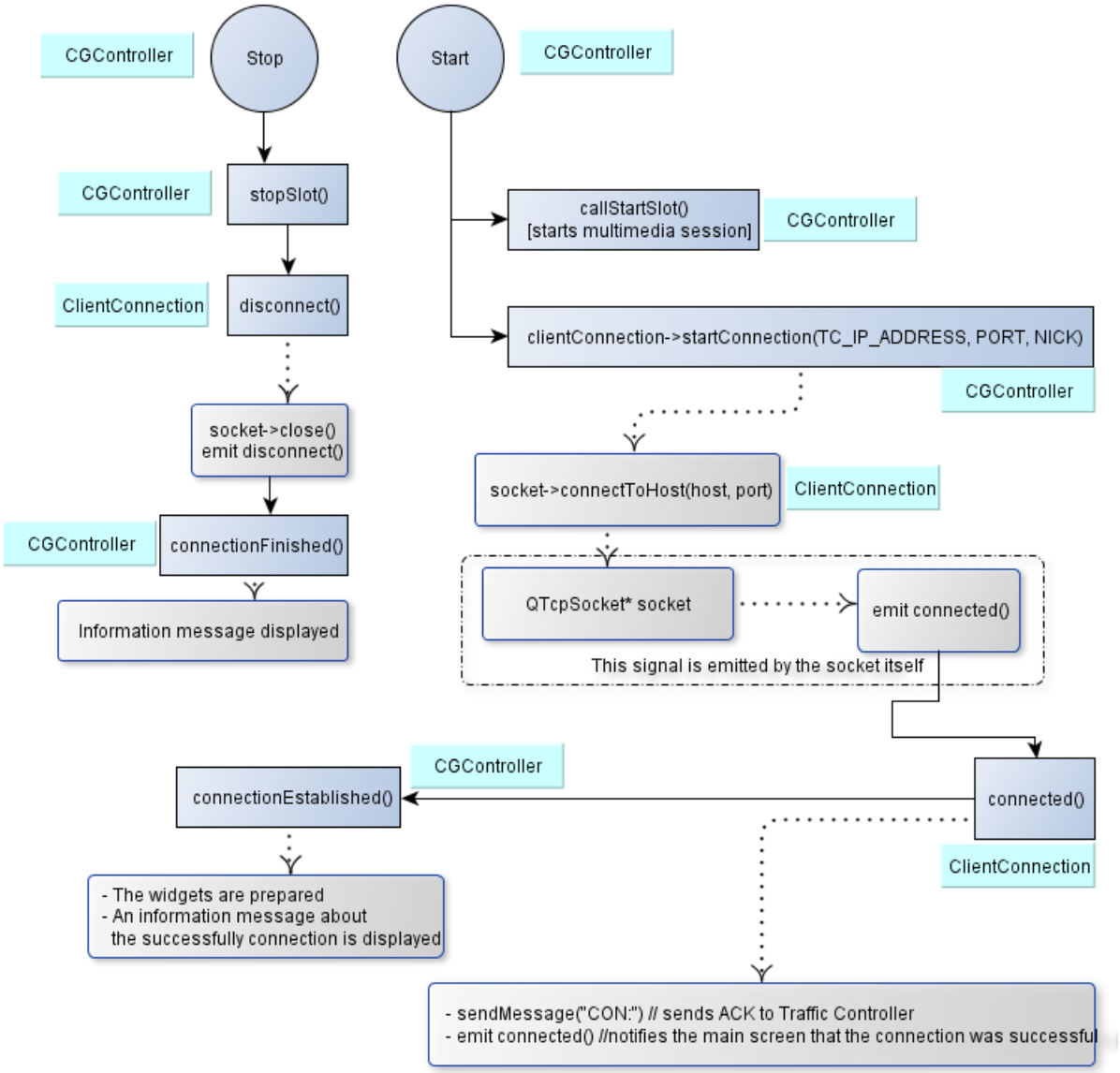


Figure 3.21: Client initialization scheme

Chapter 4

Evaluation

He who conquers others is strong.
He who conquers himself is mighty.
(Lao Tse)

Who looks outside, dreams. Who looks inside, awakes.
(Carl Jung)

Several issues are involved in the implementation of this prototype, among which, I am considering for evaluation mainly the ones related to software development, since they are the core of my work. I present only a qualitative analysis, since no experiments were performed. The analysis is divided in the diverse tasks that have been performed:

1. Reuse of the existing heterogeneous testbed, with separation between control and multimedia functionalities in two stand-alone units, and implementation of the traffic controller unit.
2. Architectural design and deployment.
3. Signaling protocol design and implementation for communication between multimedia clients and control unit.
4. Implementation of Graphical User Interfaces for the clients and the traffic controller.
5. QoS and QoE activation mechanisms.
6. Final deployment.

4.1. Reuse of the existing heterogeneous testbed and implementation of the traffic controller unit

In the client application, the new design has been built relocating the usable modules and leaving out the old ones. For the *Traffic Controller*, a simple design has been implemented to carry out NPCP actions as well as GUI control. As already mentioned, only a qualitative analysis is presented here:

1. Modularity

   Continuing with the scheme of Mobisense testbed, the line of modularity has been kept, achieving benefits as reusability and easy modifiability as well as complexity decrease. For the new testbed design, a relocation has been performed, reusing the useful modules and leaving the rest aside.

2. Reusability

   As mentioned above, the modularity scheme has been kept. Some new modules have been created which could be employed for future software; as an example: graphics module at TC implements the mechanisms to show statistics that become updated periodically with sample shifting once the buffer is full. If any other parameters were desired to be plotted, the same module would provide that functionality.

3. Modifiability

   Due to the modularity existing in the software architecture, future modifications could be performed only by operating on the involved module. An example of future modification likely to be implemented is the support for NPCP, replacing the current message format by an XML model, gaining with this change high robustness and flexibility while keeping it simple. Hence, to do this migration, only NPCP module would be affected.

4. Scalability

   Further features could be easily integrated in the system, such as new network interfaces or codecs.

5. Configurability

   Configuration options are set by modifying the appropriate *settings.h* file, that exist in both TC and client applications. The customization options include font color for text browser, orders mapping, wait times for periodical sending, IP addresses, ports, etc. Another configuration feature is that, in the client, statistics information is provided for an expert user, who can configurate its visibility from the GUI.

4.2. Architectural design and deployment

The design consists of an interconnection between the hosts at ERP and the MultiRAT testbed at WFD. They all are using local, private networks, therefore no gateways to the public Internet were involved. Traffic to WFD was routed via the VPN router/gateway, operated by the BOWL team over Gigabit Ethernet. It is worth noting that, while the design is a preliminary one still with some problems to be solved, the setup was running according to the requirements. The overall status of the *Virtual Local Area Network* (VLAN) is detailed below:

1. Interconnection was active and working. In case of any change in the VLAN, it would be transparent, not being required any additional configuration.

2. IP addresses and routes for all client devices at the demonstration room were configured manually, hence DHCP protocol should not be run at ERP side. Otherwise the port would be disabled and, anyway, the obtained IP address would not be valid.

4.3. Signaling protocol design and implementation

The protocol designed for the communication of the TC with the clients meets all the requirements described in section 3.3.2. The objective was satisfied, and consisted simply in providing a mechanism by which TC is able to send orders to the clients and to retrieve some information from them: current session parameters and availability of resources (network technologies), as well as the result of executing the orders. Nevertheless, several improvements could be performed in future:

1. *Efficiency*: periodical sending of "client info" (number of available network interfaces) can be problematic; because, in the rare case that the availability of a network interface changes, a low period results in inefficiency; since unuseful information traffic is traveling through the network. On the other hand, if the period is long, TC could send instructions to switch interface and -obviously- that would not be performed, receiving a "status" message notifying the failure. Therefore, an optimal solution would be to send this information only when needed, this is, when some value changes. A way to accomplish this would be with the client checking the interfaces locally and, when detecting any change, a notification would be sent to the TC. This strategy could be extended to "session info" messages: a value is sent only if changed.

2. *Robustness, adaptability, flexibility*: as already mentioned, the currently basic syntax of the messages could be changed to XML. This way, NPCP could be accessed and edited by using tools based on the *Document Object Model* (DOM). Thus, the mere strings processing would be abandoned in favor of a modular strategy.

Below, some examples are given of how this new format for NPCP messages would look like:

```
<!-- reply message for session information -->
<NPCP_MESSAGE type="reply" subtype="session_info">
        <SESSION_INFO qoe="0.74" video_qoe="0.74"
                audio_qoe="0.80" jitter_buffer_size="200ms
                   " bitrate="512Kbps"
                audio_codec="G.719" video_codec="H.264"
                   network="hsdpa"
                delay="15ms" packet_loss="5%"
                   synchronization="2ms" jitter="2.5ms" />
</NPCP_MESSAGE>
```

Listing 4.1: Example of "session info" message

Finally, "session info" messages (Listing 4.1) contain the current values of the parameters in the ongoing session, whereas "client info" messages (Listing 4.2) gather the available codecs and network interfaces.

```
<!-- reply message for client information -->
<NPCP_MESSAGE type="reply" subtype="client_info">
        <CLIENT_INFO>
                <!-- list of available audio codecs -->
                <AUDIO_CODEC>g722</AUDIO_CODEC>
                <AUDIO_CODEC>g7231</AUDIO_CODEC>
                <AUDIO_CODEC>g7291</AUDIO_CODEC>

                <!-- list of available video codecs -->
                <VIDEO_CODEC>h263</VIDEO_CODEC>
                <VIDEO_CODEC>h264</VIDEO_CODEC>
                <VIDEO_CODEC>mpeg2</VIDEO_CODEC>
                <VIDEO_CODEC>mpeg4</VIDEO_CODEC>

                <!-- list of available networks -->
                <NETWORK>wifi</NETWORK>
                <NETWORK>hsdpa</NETWORK>
                <NETWORK>2g</NETWORK>
                <NETWORK>mesh</NETWORK>
                <NETWORK>wimax</NETWORK>
        </CLIENT_INFO>
</NPCP_MESSAGE>
```

Listing 4.2: Example of a "client info" message

"Set" messages are sent everytime a parameter is changed at the TC (Listing 4.3). Nevertheless, if, for instance, there were a "send order" button, several

"set" orders could be grouped in one and sent all together when pushing the button, in the same NPCP message. This way, "set" messages (Listing 4.4) would collect the parameters to be changed and the values to set instead of the current ones: network interface, audio/video codec, jitter buffer, packet loss, etc.

```
sendMessage("REQUEST:SET:audiocodec:"+value + "\n\r");
```

Listing 4.3: Current instruction for "set" message: one message per parameter

```
<NPCP_MESSAGE type="request" subtype="set">
          <SET network="wifi" audio_codec="G.719"
             video_codec="MPEG4" bitrate="512Kbps"
             packet_loss="3.5%" delay="50ms" bandwidth="1
             Mbps" qos="0.74" qoe="0.80" buffer_size="20ms"
             />
</NPCP_MESSAGE>
```

Listing 4.4: Example of a "set" message

A similar strategy can be followed with "status" message (Listing 4.5): if several "set" operations have taken place, a single NPCP message could notify the result of all of them. In case of any failure when setting a parameter, this is notified to the TC

```
<NPCP_MESSAGE type="status">
         <STATUS operation="SET-network" value="failed"
            event="Unable to change network to hsdpa" />
         <STATUS operation="SET-audio_codec" value="
            successful"/>
         <STATUS operation="SET-video_codec" value="
            successful"/>
         <STATUS operation="SET-delay" value="successful"/>
         <STATUS operation="SET-buffer_size" value="
            successful"/>
</NPCP_MESSAGE>
```

Listing 4.5: Example of a "status" message

4.4. Implementation of Graphical User Interfaces

Graphical User Interfaces (Figures 3.4 and 3.5) played an essential role in thiel system. In the clients, the GUI enabled visualization of the configuration effect on perceived quality. The purpose was to present a simple but catchy design, emphasizing some aspects:

1. *Clear*: the GUI was intended to make it easy to figure out how the application works.
   The case of the client is simple: the video screen, buttons for starting/ending the session, a frame where chat is shown and where new text can be input. An additional button is in charge of enabling/disabling statistics and gauges view, addressed to expert users.

   The TC presents a much more complex GUI, but still well organized: control widgets for both sides in the top; two separated sides are set in the bottom, each side relies on widgets and gauges related to the corresponding client; a statistics field occupying the central band of the screen, once again distinguishing between the two clients. Start/stop and QoS/QoE activation widgets complete the list of elements.

2. *Intuitive*: a first-time user would find no problem to fathom the function of the elements in both GUIs. Even the TC GUI results quite familiar, despite the high number of features, and there should not be any problem figuring out the purpose of every widget.

3. *Responsive*: in general for any application, the user should not be subject to tedious waits. This is unlikely for this case, since no big data loads are entailed. Even though, Qt platform prevents such situations to occur, because of the excellent signal/slot threading system for events management.

4. *Attractive*: once again, Qt makes possible the use of fancy widgets making the GUI pleasant to use. Qt provides widgets and mechanisms to display *Scalable Vector Graphics* (SVG), which is an XML-based language for describing two-dimensional vector graphics [19] and makes possible complex images elaboration [20].

4.5. QoS and QoE activation mechanisms

This activation actions are started from the TC, where there exist a couple of buttons for such purpose. However, their functionality was not implemented at the end of the internship.

4.6. Final deployment

A further whole evaluation is not possible as, at the time of the internship end, any video call between ERP and WFD could not be performed yet; since not all the hardware resources were ready (e.g. web camera) and the remote installation, via VNC, of the client at WFD was not completed.

Chapter 5

Outlook and conclusions

In three words I can sum up everything I have learned about life: it goes on.
(R. Lee Frost)

Vitality shows in not only the ability to persist but the ability to start over.
(Francis Scott Fitzgerald)

This chapter concludes this document with a summary of the project subject and the final statement of the objectives fulfillment. Secondly, an overview of possible lines to continue this research is presented. Finally, a brief personal reflection about the experience of developing this project closes the chapter.

5.1. Summary

The background of this project was the ongoing research about QoE enhancement in modern networks, by the *Quality and Usability* department at Deustche Telekom Laboratories. The line of action of this research is focused on the evaluation of user experience in wireless networks, for a better understanding of user perception of changing quality. For such purpose Mobisense testbed was implemented. This prior system consisted of a network infrastructure, with an audio/video telephony client and a control module. Controlled laboratory conditions were created in passive and interactive situations in order to emulate the transmission phenomena of interest. All the effects perceived by users could be captured, being this the cornerstone of the study. My contribution to this research is the extension of this testbed, with the objective of implementing a QoE prototype in a real scenario with heterogeneous network technologies and integration of network-based QoS optimization algorithms; aiming not only to improve QoE of nomadic users, but also use of resources optimization in networks of the future.

The final statement of the project objectives fulfillment at the time of the end of my internship was:

- A new node deployed in WFD[1].
- A network socket was integrated in the video client for remote control.
  - Support for video/audio codecs, bit rate control and status information.
  - Design of a syntax for signaling messages.
- A stand-alone testbed controller for on-line demonstrator/testbed was implemented.
  - Based on the separation of existing test controller code from the video client, and creation of a separate entity only for control purposes.
  - This will allow more modular experiments/demonstrations in the future.
- GUIs for videotelephony and traffic control were developed.

As explained in section 4.6, a video call evaluation was not possible. Nevertheless, a dummy demonstration took place in order to show NPCP functionality. A tablet PC was running both client and TC applications and NPCP traffic could be visualized in the GUIs. As no real ongoing video call existed, simulated values were generated in the client side and shown in the statistics graphs and gauges. The tablet PC was chosen for traffic control and even for clients because of its attractiveness and mobility.

[1]Except for the installation of the client application and the fully hardware equipment.

5.2. Future work

The protoype implemented for this project will provide a more realistic platform for testing, enabling corroboration and even further exploration of conclusions reached in previous research supported by Mobisense testbed. This research approaches the elucidation of the trade-offs that quality provision must consider, such as: network coverage versus throughput and reliability of connection; efficiency versus robustness of signal compression. The main challange in tests development is the high number of transmission parameters and the design of switching conditions; not to mention the debugging and implementation efforts. Therefore, future researchs should state quality profiles with a limited number of changes, in order to safeguard data realibility.

Moreover, future studies could include additional parameters exploration, such as video resolution or frame rate.

5.3. Conclusions

I have great satisfaction for the opportunity that I had, as a student, to enjoy an internship in a great place as Deutsche Telekom Laboratories, thanks to Erasmus Placement program.

For me, it was a first approach to professional engineering. At the beginning, I had to deal with a steep learning curve and with a project much huger than what I was used to: many directories, many source files, and many technologies and concepts to keep in mind; as well as different team players to coordinate with. I faced challenging tasks in which I was expected to deliver results. I had to develop disciplined manners of working to move forward. Moreover, prototyping, design, programming and testing skills needed to be polished on me, so I focused my efforts on looking for the most efficient ways to be learnt and applied since the begining: daily planning, big picture thinking, effective designing before implementation, source control (Git), efficient coding and debugging (GDB), fast code edition (Vim),...

I also found an excellent guide and a fantastic environment plenty of opportunities to improve my skills and grow as an engineer and as a person. The working methodology consisted in periodical meetings, where the status and future steps were discussed, where I had the chance to listen and learn, as well as to present my own points of view. All these conditions resulted in a high improvement of my skills and, above all, of my passion and excitement for engineering and human relationships.

Budget

The tasks performed in this project could be grouped in seven global activities, which derive in other secondary tasks:

1. Setup of development environment and study of technologies.
2. Testbeds interconnection.
3. Traffic controller unit design and implementation.
   a) Existing features selection and reuse.
   b) Integration of new control features: network control widget, network bandwidth control gauge.
   c) QoS and QoE activation widgets.
   d) software architecture design according to NPCP.
   e) Integration of a network socket for NPCP communication with clients.
   f) GUI design and implementation.
4. Control operations signaling protocol design and implementation (NPCP).
   a) Protocol design according to requirements: types of messages, format, session, etc.
   b) Protocol implementation in client and traffic controller applications.
5. Client application update.
   a) Redesign.
   b) Integration of a network socket in the video client for remote control (via NPCP).
   c) GUI design and implementation.
6. Deployment of a second video client in WFD.
   a) Client application installation via VNC.

b) Integration of QoS control functions from MultiRAT.

c) Scripts generation to be remotely invoked for QoS functions management.

d) Web camera equipment.

7. Documentation.

First step was setup of development environment: QtCreator, QtDesigner and vim running over Linux Debian OS and installation of libraries needed for Mobisense. Additionaly Git platform was used as a source versions manager.

Once the equipment was ready to develop design phase started: how the GUIs should look like, how NPCP communication should be, software architecture, etc. In the middle of December, implementation phase began, lasting till March.

In a parallel to this activities testbeds interconnection was carried out, becoming ready at the end of January.

In March, the deployment of the video client in WFD was the focus of the job, together with documentation.

At the end of the internship, all the tasks had been successfully completed, with the exception of QoS/QoE widgets activation implementation and last steps for deployment of the client in WFD.

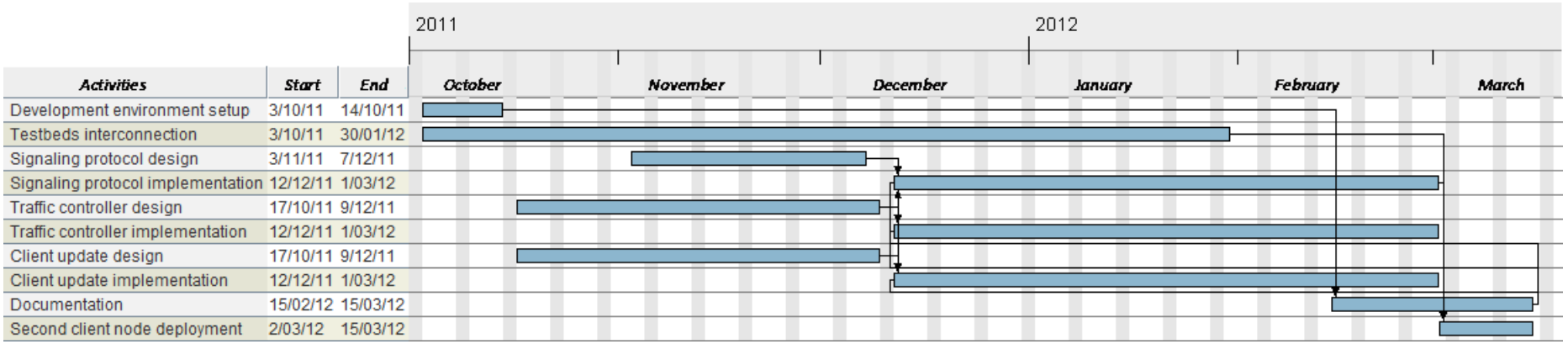


Figure 5.1: Gannt Diagram

1. Author:
   Gerardo Pinar Loriente

2. Department:
   Telematics Engineering

3. Project description:

Title:	Design and implementation of an online demonstrator for a video telephony system over heterogeneous networks.
Duration (months):	6
Indirect costs ratio:	20%

4. Total project budget (Euros):
   24,475.93 €

5. Project breakdown (direct costs):

HUMAN RESOURCES

Surname, Name	ID number	Category	Dedication (people per month)	people per month cost	Cost (Euros)	Signature
Pinar Loriente, Gerardo	-	Ingeniero	6	2,694.39	16,166.34	
	-	Ingeniero Senior	0.5	4,289.54	2,144.77	
	-	Ingeniero	2	2,694.39	5,388.78	
	-	Service technician	0.2	2,694.39	538.88	
	people per month		8.7	**Total**	24,238.77 €	

EQUIPMENT

Description	Cost (Euros)	% use in the project	Dedication (months)	Depreciation period	Chargeable cost
Tablet Asus EP 121	883.00	100	6	60	88.30
PC Optiplex 780	699.00	100	6	60	69.90
PC Optiplex 780	699.00	100	6	60	69.90
PC Optiplex 780	699.00	33	2	60	7.69
Web Camera Creative Live	125.00	2	3	60	1.38
Web Camera Creative Live	125.00	0	3	60	0.00
				Total	237.16 €

TASKS OUTSOURCING		
Description	Company	Chargeable cost
	Total	0.00

OTHER DIRECT COSTS OF THE PROJECT		
Description	Company	Chargeable cost
		0.00
	Total	0.00

6. Costs summary:

Direct costs	
Human resources	24.239
Amortization	237
Tasks outsourcing	0
Operation costs	0
Indirect costs	4,895
Total	29,371

The total budget of this project amounts to 29,371 €.

Leganés a 25 de October de 2013

The project engineer

Gerardo Pinar Loriente

Bibliography

[1] B. Lewcio, *Management of Speech and Video Telephony Quality in Heterogeneous Wireless Networks*. PhD thesis, 2013.

[2] D. R. de Transporte Multiservicio, "Entendiendo la tecnología voip," 2002.

[3] T. Morales Garnica and A. Lugo García, "Implementación de un servidor de comunicaciones basado en sip," 2007.

[4] "Ict regultation toolkit." `http://www.ictregulationtoolkit.org`. Accessed: 2013-09-28.

[5] J. C. Corrales Muñoz and Á. Redón Gallón, "Telefonía ip: conceptos y arquitectura," 2011.

[6] A. Sfairopoulou *et al.*, "A cross-layer mechanism for qos improvements in voip over multi-rate wlan networks," 2008.

[7] J. B. Plaza, *Implantación de un sistema VoIP basado en Asterisk*. PhD thesis, 2009.

[8] S. Moreno Urrea, "Estudio experimental de calidad de servicio de voz sobre ip: comparativa subjetiva versus objetiva," 2012.

[9] V. Jacobson, R. Frederick, S. Casner, and H. Schulzrinne, "Rtp: A transport protocol for real-time applications," 2003.

[10] F. J. M. Cirera, J. G. López, and F. G. Montoya, "Diseño e implementación de un sistema voip de alta disponibilidad y alto rendimiento," 2009.

[11] C. Bouras, A. Gkamas, and G. Kioumourtzis, "Challenges in cross layer design for multimedia transmission over wireless networks," in *Proceedings of WWRF-21st Meeting WG3–Future Architecture, Stockholm, Sweden, October*, 2008.

[12] S. Rao and K. Shama, "Cross layer protocols for multimedia transmission in wireless networks," *International Journal of Computer Science & Engineering Survey*, vol. 3, no. 3, pp. 15–28, 2012.

[13] F. Mesquita Buiati, "Diseño e implementación de un sistema de información de movilidad para redes heterogéneas," 2013.

[14] B. Lewcio and M. S., "A testbed for qoe-based multimedia streaming optimization in heterogeneous wireless networks," in *Signal Processing and Communication Systems (ICSPCS), 5th International Conference on*, pp. 1–9, Deutsche Telekom Labs., Tech. Univ. Berlin, 2011.

[15] Y. Chen, S. Zhang, S. Xu, and G. Y. Li, "Fundamental trade-offs on green wireless networks," *Communications Magazine, IEEE*, vol. 49, no. 6, pp. 30–37, 2011.

[16] N. Bayer, D. Sivchenko, H. J. Einsiedler, A. Roos, A. Uzun, S. Gondor, and A. Kupper, "Energy optimisation in heterogeneous multi-rat networks," in *Intelligence in Next Generation Networks (ICIN), 2011 15th International Conference on*, pp. 139–144, IEEE, 2011.

[17] "Institut für kommunikationsnetze und rechnersysteme." `http://www.ikr.uni-stuttgart.de/Content/itg/fg524/Meetings/2012-11-29-Berlin/04_ITG524_Berlin_Bayer.pdf`. Accessed: 2013-10-6.

[18] D. Sivchenko, B. Schubert, S. Bretzke, and K. Jonas, "Selection of the implementation platform - description of elements and functions used in the testbed," 2011.

[19] "Qt-project." `http://qt-project.org/doc/qt-4.8/painting-svgviewer.html`. Accessed: 2013-10-2.

[20] "wikipedia." `http://es.wikipedia.org/wiki/Scalable_Vector_Graphics`. Accessed: 2013-10-2.