UNIVERSIDAD DE GRANADA

TRABAJO FIN DE GRADO

INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Análisis de tráfico y QoE para Skype

Autor
Julio Elvira del Castillo

Director
Jorge Navarro Ortiz

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, junio de 2019

Análisis de tráfico y QoE para Skype

Autor
Julio Elvira del Castillo

Director
Jorge Navarro Ortiz

Análisis de tráfico y QoE para Skype

Julio Elvira del Castillo

Palabras clave: *Skype*, tráfico, *throughput*, *delay*, *jitter*, tasa de pérdidas, calidad de experiencia, *MOS*, *VoIP*

Resumen

El propósito de este Trabajo de Fin de Grado consiste en el desarrollo de modelos cuantitativos de la generación de tráfico y de la calidad experimentada por el usuario para el servicio de Skype. Estos modelos darán lugar a diferentes matrices, que recogen los valores medios de tráfico generado y de calidad de experiencia (*QoE*), expresada a través del *MOS* (*Mean Opinion Score*). Se ha elegido este servicio porque Skype es una de las herramientas de telefonía IP (*VoIP*) más utilizada a nivel mundial.

Para conseguir este objetivo, el presente proyecto se ha desarrollado en una serie de fases.

- En primer lugar se ha procedido al estudio de diferente tipo de documentación y bibliografía relacionada con el tema que abarca este proyecto.
- En segundo lugar se ha desarrollado una metodología de experimentos y pruebas.
- Mas tarde se ha procedido a llevar a cabo la serie de experimentos y pruebas, con el objetivo de obtener los valores de tráfico generado por *Skype* y la calidad de experiencia (*QoE*).
- Por último, se han procesado estos valores con Excel, obteniendo así los modelos propuestos.

Para cada experimento se obtienen múltiples indicadores de rendimiento, los cuales reflejan de forma gráfica y cuantitativa métricas objetivas como, por ejemplo, el tráfico generado por la aplicación, y métricas subjetivas como, por ejemplo, la calidad experimentada por el usuario final.

Las métricas objetivas, recogidas en la totalidad de los experimentos realizados, son las siguientes:

- Se han guardados los valores instantáneos de throughput instantáneo generados por la aplicación Skype durante los experimentos correspondientes al *throughput* generado por la herramienta *Skype*.

- Se almacenan los valores instantáneos correspondientes al retardo o *delay*, recogidos en intervalos de 10 segundos.
- Se guardan los valores de *jitter* o variación de retardo, recogidos en intervalos de 10 segundos.
- Valores instantáneos de la tasa de paquetes perdidos, expresada en porcentaje del total, recogida en valores de 10 segundos.

A su vez, para cada simulación se obtienen y almacenan las métricas instantáneas correspondientes a la calidad de experiencia (*QoE*, *Quality of Experience*). Se han obtenido y almacenado métricas de calidad de experiencia. tales como el eModel.

De esta forma, se obtienen las distintas métricas, tanto objetivas como subjetivas, las cuales se almacenenan y se estudian, tal como se especificará en el correspondiente apartado. Gracias al posterior procesado y estudio de los distintos datos obtenidos para las diferentes métricas de red se obtienen finalmente los modelos deseados, tanto de generación de tráfico como de calidad de experiencia *QoE*

A su vez se recogen distintas gráficas que reflejan los datos anteriormente, para completar el estudio del tráfico y calidad de experiencia *QoE* de *Skype*.

Analysis of traffic and QoE for Skype

Julio Elvira del Castillo

Keywords: *VoIP*, *Quality of Experience*, traffic, model, *Skype*, matrix, simulation

Abstract

The main purpose of this Bachelor thesis is the building of a Quality of Experience (*QoE*) model for the worldwide known Voice over IP *Skype*, as well as to build a model for the *Skype* generated traffic under diverse network conditions.

Both the Quality of Experience model and the traffic model consist of several matrixes, which express, in a quantitative manner, the average data.

This is a way to obtain a study about the expected Quality of Experience, as well as the expected generated traffic for each network condition in the well known VoIP application.

This project has been built in several phases, all of which are oriented to finally obtain the previously stated models.

There was an initial phase which concerns the gathering of the fundamental information, needed to design the structured model, a model which was in the successive phases implemented.

It is in this first phase when an exhaustive study of many different studies relating the modeling of *VoIP* traffic is conducted. It is through these studies, that a design of experiments is built. Once that all the appropriate information was gathered and exhaustively studied, a detailed design for the experiments is implemented, specifying in an accurate way the steps to be made in order build the desired Quality of Experience and the *Skype* traffic model.

Afterwards it is a must to carry out the design previously mentioned. As a previous step to the experiments, it is required to implement several changes in the tools needed in the project, as well as to implement a new Java method. The method is essential to build the matrixes models of this project, as it enables the main tool to gather in a few text files the different instantaneous values of the main network metrics.

It is also required to implement the different *XML* files which define the different network conditions under which the experiments are made. For each and every different network condition, a *XML* file is needed, as they characterize the network metrics of the experiment.

After all the required implementations, in the main phase of this project the experiment design is carried out, which leads to the building of both the Quality of Experience model and the *Skype* generated traffic model.

The project consists of many experiments. For each experiment several following files are obtained, through the Java method *crearFichero()*, which are meant to express in a studied, quantitative way, the generated traffic and the Quality of Experience.

Below are described each and every of the obtained network metrics:

- The instantaneous values of the generated *Skype* throughput gathered every 10.0 seconds.
- The instantaneous values of the experimented delay during the experiments, gathered every 10.0 seconds.
- The instantaneous values of the observed jitter, gathered every 10.0 seconds
- A file containing the values of the packet loss rate, also gathered in 10.0 seconds intervals, for every experiment.

These 4 metrics have been obtained for each and every simulation. Afterwards, the average value of those instantaneous values gathered in each file is calculated, and then put in its matrix, building this way the matrixes for both the Quality of Experience and the generated *Skype* traffic.

In addition to the quantitative models, and with the purpose of a better understanding of both the Quality of Experience and the generated traffic, several graphics are obtained, which are meant to display, in a graphic, intuitive way the Quality of Experience obtained in the simulations, as well as the generated traffic.

Yo, **Julio Elvira del Castillo**, alumno de la titulación Grado en Ingeniería de Tecnologías de Telecomunicación de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI XXX, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Julio Elvira del Castillo

Granada, junio de 2019.

D. **Jorge Navarro Ortiz**, Profesor del Área de Ingeniería Telemática del Departamento Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Análisis de tráfico y QoE para Skype***, ha sido realizado bajo su supervisión por **Jorge Navarro Ortiz**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada, junio de 2019.

El director:

Jorge Navarro Ortiz

Agradecimientos

A mi tutor, Jorge Navarro, por resolver de la mejor manera todas las dudas que me fueron surgiendo a lo largo de este proyecto y ayudarme a que resultase lo mejor posible.

A mis padres y mi hermana, que tanto han cuidado de mí, por su apoyo emocional a lo largo de este camino.

A mis amigos, por estar siempre y suponerme un pilar fundamental en la vida.

A todos, gracias.

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Contexto

En la actualidad las redes multimedia están cobrando cada vez una importancia mayor en el nicho de las telecomunicaciones. En concreto, una de las aplicaciones que mayor progresión ha experimentado en los últimos tiempos es la de la telefonía sobre IP (*VoIP*, *Voice over IP*), metodología que se encarga de la realización de llamadas, tanto de voz como videoconferencias, aprovechando la red IP ya existente.

La tecnología *VoIP* se basa en unos principios similares a la telefonía tradicional, con la excepción de que la información es transmitida por una red de conmutación de paquetes, en contraposición a la red de conmutación de circuitos que se utiliza en la telefonía convencional (RTC, Red Telefónica Conmutada).

El auge en la tecnología *VoIP* ha provocado que numerosas herramientas aprovechen esta funcionalidad para realizar videoconferencias a un coste más barato, lanzando al mercado atractivas propuestas para los clientes. En concreto, una de ellas, *Skype*, fue una de las aplicaciones la que sentó las bases de la telefonía IP, y la que más tarde se convertiría en una de las aplicaciones de *VoIP* mas utilizada alrededor del mundo sin lugar a dudas [1].

Skype es una herramienta multiplataforma que cuenta versiones tanto de escritorio (*Windows*, *Linux*, *macOS*) como para móviles (*Android*, *iOS*, *Windows Phone*) que utiliza una arquitectura *Peer 2 Peer*, en lugar del usual cliente- servidor. *Skype* utiliza un códec propio denominado *SILK* [2], este códec ha demostrado un rendimiento superior al de otros anteriormente utilizados tanto por *Skype* como por otras herramientas de *VoIP*.

Este codec es capaz de ajustar el tráfico a las condiciones de red, variando tanto la tasa de bits (bitrate) como la frecuencia de muestreo utilizada según como de buenas sean las condiciones de red. Esta es una característica muy interesante, y que será estudiada exhaustivamente en el posterior apartado de Pruebas.

En el presente Trabajo de Fin de Grado se estudiará y analizará el tráfico generado por la herramienta *Skype* y la calidad de experiencia (*QoE*, *Quality of Experience*) percibida por el usuario final bajo variadas condiciones de red, viendo cómo estas afectan al comportamiento de la herramienta.

Para conseguir este objetivo, se realizarán experimentos variando las condiciones de red mediante la modificación de tres parámetros principales: la capacidad del canal o *throughput*, el retardo o *delay*, y la tasa de pérdida de paquetes.

1.2. Motivación

Los servicios y redes multimedia son servicios que requieren de gran interactividad y de un tiempo de respuesta extremadamente bajo. Por este motivo, son servicios muy sensibles a posibles retardos o variaciones las condiciones en la red.

Debido a que requieren gran interactividad, lo más habitual es que utilicen mecanismos de tipo *best-effort*, muy útiles para reducir el retardo pero que, sin embargo, carecen de mecanismos para asegurar la correcta recepción de paquetes o para garantizar una *QoS* (*Quality of Service*, Calidad de Servicio por sus siglas en inglés).

Desde el primer teléfono comercial de voz sobre IP (*VoIP*), en unas dos décadas *VoIP* ha revolucionado la comunicación alrededor del mundo. *VoIP* funciona segmentando la señal de voz analógica inicial en tramas de voz. Estas tramas son posteriormente comprimidas y transmitidas entre equipos mediante sus direcciones IP. Al recibir estas tramas, se descomprimidos y recomponen, recreando la señal analógica inicial .

Debido al exponencial crecimiento de las redes IP, la *VoIP* ha experimentado un crecimiento similar. En 1995 se lanzó al mercado el primer teléfono comercial *VoIP* y, a finales del siglo XX, la telefonía a través de IP tan solo representaba el 1 % del total de la telefonía (datos globales). No obstante, tan solo unos años después, en 2003, *VoIP* representaba ya aproximadamente el 25 % del total de la telefonía, año en el que salió al mercado *Skype*.

Este auge que experimentó la Voz sobre IP (*VoIP*) se debió en gran parte a su gran aceptación en el mundo corporativo, debido al abaratamiento en costes que suponía a las empresas con respecto a la telefonía tradicional. Por ello, muchas redes corporativas fueron progresivamente sustituyendo su red telefónica tradicional por tecnologías de Voz sobre IP (*VoIP*) [3].

La tecnología de Voz sobre IP (*VoIP*) conllevaba un abaratamiento en costes, permitiendo tener muchas lineas de telefono en una única red corporativa [4], dependiendo del ancho de banda (*Bandwith*) disponible en la misma. La Voz sobre IP permite además la utilización de diferentes tecnologías para la realización de las llamadas [5]. En la Figura 1.1 se puede observar un ejemplo de lo expuesto, en el que una red corporativa hace uso de la Voz sobre IP (*VoIP*) para conectar las llamdas, a través de Internet, entre dos de sus campus.

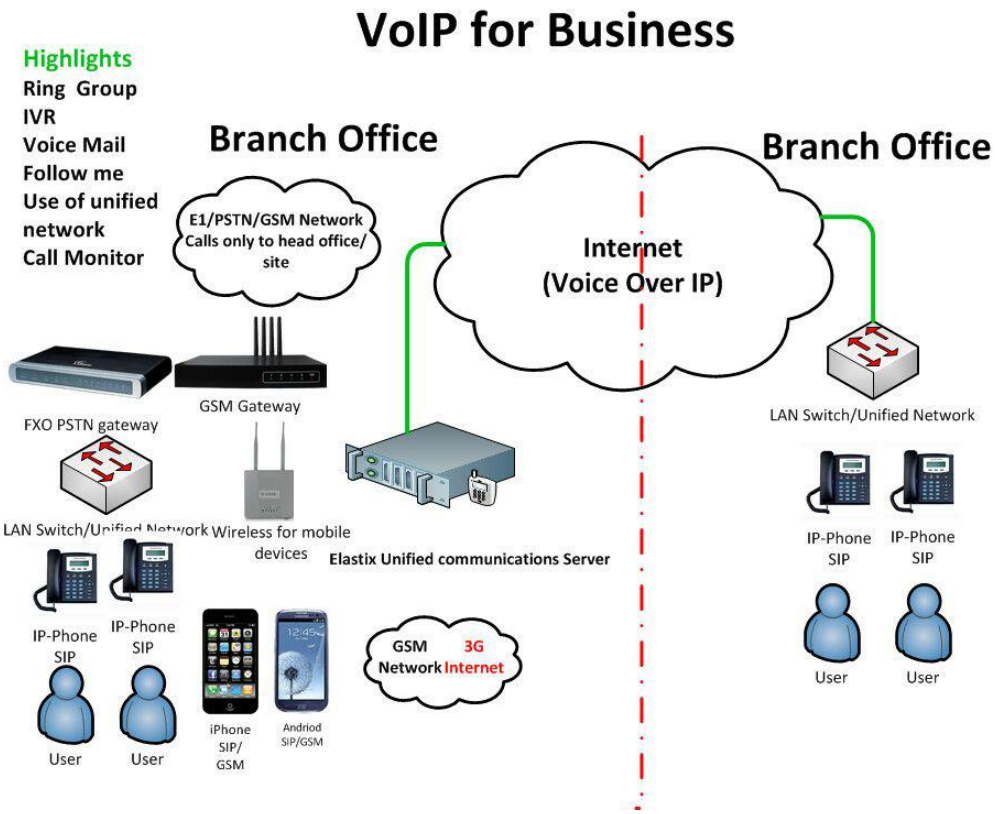


Figura 1.1: Conexión entre dos campus de una Red Corporativa

La cada vez mayor disponibilidad de acceso a Internet de banda ancha,

con el significado que esto tiene sobre la calidad de la llamada sobre IP, ha provocado que la telefonía IP haya continuado experimentado un progresivo aumento en el mercado global de la telefonía.

El abaratamiento que supone el uso de tecnología *VoIP* sobre la telefonía convencional ha sido un factor clave en el auge de la telefonía sobre IP, además las diversas opciones que ha ido progresivamente ofreciendo en el campo de la comunicación multimedia, integrando servicios de videollamada, videoconferencia, mensajes de texto o chat etc.

Es por todo esto que en su momento se eligió este Trabajo de Fin de Grado, fruto de una curiosidad por el estudio de este campo en auge de las Telecomunicaciones, y la posibilidad de poder poner en práctica diversos conocimientos adquiridos en el campo de las Telecomunicaciones y la Telemática. Concretamente, se centrará en la evaluación del funcionamiento de la herramienta *Skype*, desde los puntos de vista de generación de tráfico y de la calidad de experiencia percibida por el usuario.

1.3. Objetivos

El principal objetivo de este Trabajo de Fin de Grado es el análisis del funcionamiento de una aplicación de *VoIP*, en un entorno controlado, y bajo una gran variedad de condiciones de red. Se pretende mediante este estudio realizar un modelo detallado que refleje el comportamiento de *Skype*, su calidad de servicio (*QoS*) y su calidad de experiencia (*QoE*) bajo cualquiera de estas condiciones. En este sentido, se ha utilizado una red real con condiciones óptimas para que las degradaciones se debieran exclusivamente a las introducidas expresamente para la realización de los experimentos.

Para la realización de la totalidad de los experimentos, se han empleado dos máquinas virtuales [6] con Windows 7, que han hecho las veces de Cliente y Servidor de la llamada. Esto se ha hecho utilizando el software de virtualización VirtualBox [7].

Para la consecución de este objetivo primario se han llevado a cabo diferentes objetivos secundarios:

- Se establecerán los límites de los parámetros de red (*throughput*, retardo, pérdidas) para la realización de experimentos. Para ello, se estudiará qué valores hacen que la calidad de servicio (*QoS*) se haya degradado tanto que no la comunicación no sea posible. Este primer paso era necesario para trabajar en los siguientes objetivos.
- Se realizarán experimentos con diferentes señales de audio en formato

.waw, con la intención de obtener los valores de tráfico generado por *Skype*, así como de calidad de experiencia (*QoE*)

- Se han realizado diferentes simulaciones con diferentes tipos de audio, con el objetivo de ilustrar el comportamiento del entorno, así como de comprobar que todos los recursos utilizados funcionasen de forma óptima.

- Se realizarán experimentos con multitud de condiciones de red, con el propósito de ver cómo funciona *Skype* en diferentes situaciones.

- Se realizarán los experimentos necesarios para averiguar cómo *Skype* genera su tráfico y se adapta a condiciones cambiantes, y cómo esto afecta a la calidad experimentada por el usuario. En este sentido, se recogerá información de métricas objetivas (*throughput*, retardo, *jitter* y pérdidas) y de métricas subjetivas (valores de *MOS* (*Mean Opinion Score* [8]) utilizando para ello el modelo E (*eModel*) [9].

De esta forma, y mediante la consecución de los diferentes objetivos secundarios, se logrará evaluar la calidad de servicio (*QoS*) y la calidad de experiencia (*QoE*) en *Skype*, y esto permitirá crear un modelo tanto de generación de tráfico como de calidad experimentada.

1.4. Estructura de la memoria

En esta sección se procede a hacer una descripción resumida del contenido de los diferentes apartados que conformaran el Trabajo de Fin de Grado. Conseguimos así exponer una idea global de cómo se estructura el mismo. Los diferentes apartados que conforman este Trabajo de Fin de Grado son los siguientes:

- Introducción: En este capítulo se exponen los motivos, de diversa índole, por los cuales se eligió hacer este trabajo en concreto. Asimismo se realiza una exposición detallada de la cuestión que se pretende resolver, así como los objetivos propuestos.

- Estado del arte: Se describen programas o aplicaciones ya existentes que han ayudado a resolver bien problemas de naturaleza similar al planteado en este trabajo.

- Planificación y costes: Descripción exhaustiva de los materiales y recursos utilizados para llevar a cabo el Trabajo de Fin de Grado, así como de los costes detallados de los mismos y su planificación temporal.

- Especificación de requisitos: Este capítulo permite explicar los distintos requerimientos necesarios para la consecución de los diferentes objetivos, así como de la resolución de los problemas eventuales que han podido surgir a lo largo de todo el Trabajo de Fin de Grado. Como punto final se realiza una descripción exhaustiva de los requisitos funcionales y de los requisitos no funcionales necesarios para un correcto diseño de acuerdo a los objetivos inicialmente propuestos.

- Descripción de herramientas utilizadas: En este capítulo de describen todas las herramientas, hardware y software utilizadas a lo largo de la realización del proyecto.

- Implementación: Se detallan de forma detallada todos los procedimientos utilizados para llevar a cabo con éxito la consecución de los requerimientos de la aplicación.

- Realización de simulaciones y resultados obtenidos: A lo largo de todo este Trabajo de Fin de Grado se han llevado a cabo numerosos experimentos para lograr los objetivos propuestos, y en este capítulo se analizan los resultados obtenidos.

- Conclusiones y líneas futuras. Capítulo final del trabajo, en el que se recogen las valoraciones del autor y se describen varias líneas futuras que podrían servir para mejorar el trabajo o extenderlo.

Capítulo 2

Estado del arte

En este capítulo se procederá a realizar una exposición de las mejores herramientas de monitorización de la calidad de experiencia (*QoE*) y/o calidad de servicio (*QoS*) en el ámbito de la telefonía sobre IP (*VoIP*).

2.1. PRTG Network Monitor

PRTG Network Monitor [10] es una aplicación de monitorización que ofrece un entorno de monitorización de telefonía sobre IP (*VoIP*), con la posibilidad de hacer monitorización de la calidad de servicio (*QoS*), así como de monitorización por la tecnología de *IP SLA* (*Service Level Agreement*) de *Cisco*, la cual nos permite la simulación de datos *VoIP* para realizar pruebas en la red, siempre que los nodos utilizados sean los propios de *Cisco*.

Para la monitorización de la calidad de servicio (*QoS*) que el usuario experimentaría, esta herramienta se encarga de hacer una monitorización de los siguientes parámetros:

- Tasa de paquetes perdidos
- Tasa de paquetes duplicados
- Tasa de paquetes desordenados
- Retardos o *delay*
- Variación de retardo o *Jitter*

Esta herramienta funciona de una manera similar a la utilizada en este proyecto, siendo capaz de monitorizar parámetros de red tales como la *latency*, el *jitter*, o la perdida de paquetes, así como el *MOS*.

PRTG Network Monitor también ofrece la posibilidad de realizar un filtrado por protocolo utilizado, así como la recogida de información de monitorización según el protocolo, tal y como puede verse en la siguiente imagen. Además esta herramienta permite el envío de notificaciones cuando se traspasa cierto umbral preestablecido.

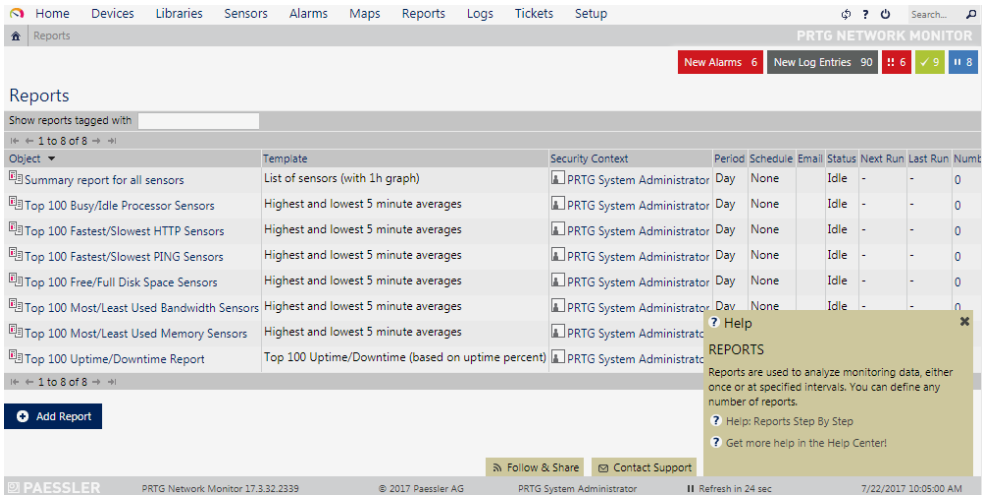


Figura 2.1: Log con información monitorizada mediante PRTG

Lo que hace particularmente interesante a esta herramienta es que incluye el sensor de *CISCO SLA*, el cual proporciona a la herramienta la capacidad de particularizar la monitorización de la calidad de servicio (*QoS*) a dispositivos *Cisco*.

2.2. VoIP Tester

Esta herramienta [11] fué desarrollada en el ámbito de un Proyecto Final de Carrera en la Universidad de Granada.

VoIP Tester es una herramienta desarrollada en Java, la cual permite al usuario realizar una configuración detallada de los parámetros de red mas usuales.

Cabe destacar que esta aplicación funciona sobre flujos *VoIP* reales, incluyendo todos los codecs que soporte la librería GStreamer [12], tales como Speex, ADPCM MS, Siren 7, CELT, PCM Ley A, PCM Ley Mu, AAC, WMA v1 etc

Una desventaja de *VoIP Tester* es que no es capaz de funcionar con códecs de bitrate variable (*VBR*), únicamente con códecs de bitrate constante (*CBR*), por esta razón esta herramienta no sirve para trabajar con aplicaciones de *VoIP* que usen códecs de bitrate variable, tales como *Skype* o *Discord*.

2.3. ThousandEyes

ThousandEyes [13] es una de las mejores herramientas que existen en el mercado actual en cuanto a monitorización de Voz sobre IP. Esta aplicación establece una correlación entre los parámetros básicos de la red utilizada, retardo (*delay*), tasa de paquetes perdidos y variaciones en el retardo (*jitter*) y la calidad de experiencia que el usuario tendría utilizando ese servicio *VoIP*.

La funcionalidad que hace especial a esta herramienta es que ofrece la posibilidad de simular llamadas de Voz sobre IP entre dos nodos para hacer pruebas sobre la red existente. Como extra, añade la posibilidad de hacer *troubleshooting* para detectar fallos en el enlace.

2.4. Solarwinds VoIP Network Quality Manager

Esta aplicación [14], desarrollada por *Solarwinds*, es una de las más utilizadas en cuanto a monitorización del rendimiento de la red a nivel mundial Permite al usuario monitorizar numerosas métricas de red, tales como:

- Retardo o *delay*
- Variación del *delay* o *Jitter*
- Tasa de paquetes perdidos

También permite al usuario realizar *troubleshooting* de la red, mediante el establecimiento de una correlación entre el rendimiento de la red *WAN* y posibles problemas en las llamadas a realizar. Todas las métricas de red (*Delay*, *Jitter*, *MOS*, Tasa de paquetes perdidos...) involucradas en la calidad de experiencia (*QoE*) que el usuario de la red experimentaría se pueden recolectar y almacenar para su posterior estudio. En la siguiente imagen se puede ver un ejemplo de su historial de notificaciones.

Solarwinds permite generar trafico sintético de Voz sobre IP para el estudio del rendimiento de la red. Además permite el uso de notificaciones cuando sucede un evento de red, o se traspasa cierto umbral (por ejemplo si

el *delay* en la red sube por encima de cierto umbral establecido).

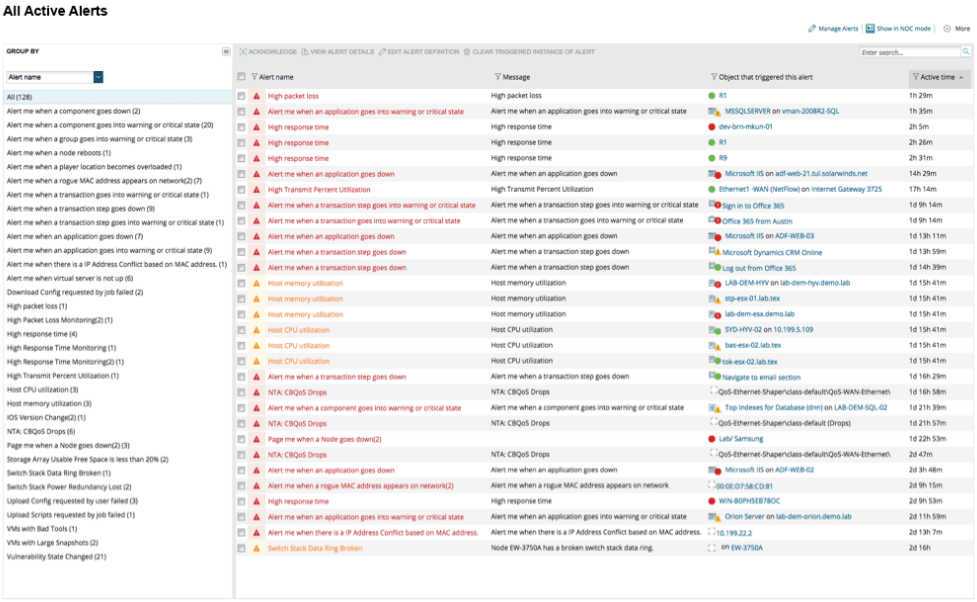


Figura 2.2: Historial de notificaciones SolarwindsVoIP

Como funcionalidad añadida, *VNQM* proporciona capacidad de monitorización para dispositivos Cisco, basada en el uso de *MIBs* (*Management Information Base*) propias que recolectan información relacionada con los dispositivos Cisco de la red mediante el protocolo de gestión de red *SNMP* (*Simple Network Management Protocol*).

2.5. WhatsUp Gold VoIP Monitor

Esta herramienta [15], desarrollada por *Ipswitch*, nos permite la monitorización de las métricas básicas de red, tales como el *delay*, *jitter* o tasa de perdida de paquetes y establecer una calidad de experiencia (*QoE*) en función a estos, Además incluye el envío automático de notificaciones mediante el protocolo *SNMP* configurable cuando alguno de estos parámetros supera cierto umbral preestablecido, proporcionándonos la seguridad de que la red monitorizada se encuentra siempre dentro de esos umbrales de aceptación.

Esta herramienta cuenta con varios monitores de rendimiento que nos dan una medida de la estabilidad, perdida de paquetes y retardo, tanto en el emisor como en el receptor. Proporciona además una puntuación de la

calidad de experiencia basada en *MOS*.

2.6. VoIPmonitor

VoIPmonitor [16] es una software de libre distribución capaz de medir la calidad de servicio (*QoS*) en Voz sobre IP. Analiza la calidad de la llamada *VoIP* en función de los parámetros de red de *Jitter* y tasa de perdida de paquetes de acuerdo con el E-model de la ITU-T G.107 [9].

Toda la estadística recogida en el transcurso de la llamada *VoIP* se almacena en una base de datos *MySQL*. Opcionalmente, existe la posibilidad de almacenar la llamada en un archivo *.pcap* (utilizado por analizadores de paquetes como *Wireshark*), los cuales también se pueden convertir a fichero de audio en formato *WAW*, formato el cual ha sido utilizado en la realización de este proyecto.

Dos de las ventajas de esta herramienta sobre otras es la cantidad de códecs que soporta, entre los cuales se encuentra *Silk*, el códec utilizado por *Skype*, y que es posible su uso en el mundo corporativo para propósitos de tarificación.

2.7. Conclusiones

En este capítulo del proyecto se han expuesto ver diversas herramientas de monitorización de Voz sobre IP, tanto comerciales como de libre distribución, que se basan en diferentes métricas o tecnologías (véase Cisco SLA) [17].

Para todas ellas se han podido ver tanto sus posibles desventajas como sus ventajas en cuanto a las posibilidades que ofrecen, e incluso como algunas de ellas no son adecuadas en algunos entornos de red. Con esto se ha pretendido dar una perspectiva de las posibilidades existentes en cuanto a la monitorización de *VoIP* en el mercado actual.

Capítulo 3

Herramientas

En esta sección se procede a realizar una descripción exhaustiva de todas las herramientas utilizadas a lo largo de este Trabajo de Fin de Grado.

3.1. VirtualBox

VirtualBox [7] es una herramienta de virtualización diseñado por Oracle para arquitecturas x86. Esta aplicación permite instalar sistemas operativos *invitados*, dentro de nuestro *host-anfitrión*.

En el presente proyecto, VirtualBox ha sido una herramienta fundamental, la cual nos ha permitido tener dos máquinas virtuales, que han hecho las veces de Cliente y Servidor, en los numerosos experimentos realizados.

3.2. VLC

VLC Media Player es un reproductor multimedia de libre distribución y gratuito. Desarrollado por *VideoLAN*, está disponible para todos los sistemas operativos más populares, tanto de sobremesa (*Windows*, *Linux*, *macOS*) como móviles (*Android*, *iOS*, *Windows Phone*). Lo interesante de este reproductor es que se puede configurar para que tome como salida de audio un dispositivo concreto, *Virtual Audio Cable* para el caso que nos ocupa.

VLC incluye la posibilidad de ser manejado mediante línea de comandos, lo cual ha sido de utilidad en el proyecto que nos ocupa, para poder manejar el programa de forma automatizada *VLC* mediante la herramienta diseñada en el Trabajo Fin de Grado [18].

Todas las utilidades en cuanto a línea de comandos se pueden consultar en la wiki oficial de *VideoLAN* [19].

3.3. Wireshark

Wireshark es un analizador de paquetes de libre distribución ampliamente utilizado para realizar análisis de red y *troubleshooting*.

Esta herramienta multiplataforma nos permite realizar la captura de paquetes de red, lo cual ha resultado de gran utilidad para capturar cierto tipo de paquetes generados durante las diferentes llamadas de *Skype*.

Existe una versión llamada *T-Shark*, optimizada para su utilización sin interfaz gráfica, desde una terminal.

Wireshark ha sido de gran utilidad en este proyecto para la captura de ciertos paquetes de red, que nos ha permitido realizar algunas comprobaciones previas a la etapa de pruebas, como conocer si la señalización de *Skype* es *out-of-band* (envío de las tramas de control en una banda distinta a las tramas de audio en este caso).

A continuación, se expone la Figura 3.1, en la que se observa un ejemplo de una captura que se realizó con *Wireshark* en la máquina virtual del servidor, en la etapa de Análisis y Diseño.

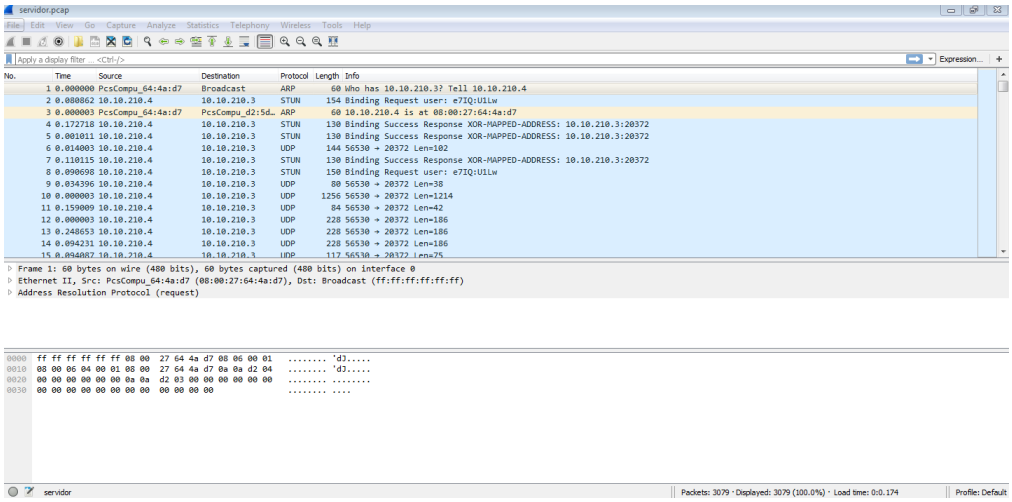


Figura 3.1: Captura de *Wireshark* realizada en la máquina virtual del servidor

3.4. Network Emulator for Windows Toolkit

Network Emulator for Windows Toolkit es un emulador que permite al usuario realizar la emulación de redes de comunicaciones para comprobar el rendimiento de aplicaciones reales sobre un entorno emulado, *Skype* en el caso que nos ocupa. Este emulador permite hacer comprobaciones sobre el rendimiento, predecir el impacto que puede llegar a tener sobre la red un cambio y en general optimizar la toma de decisiones sobre la red que se emula.

Network Emulator for Windows Toolkit permite modificar una gran cantidad de parámetros de red, destacando el ancho de banda disponible, el retardo y la pérdida de paquetes.

La característica clave de este emulador para el presente trabajo es que permite su ejecución por lineas de comandos gracias al uso de archivos *XML*. Esto ha permitido una versatilidad total a la hora de realizar todos los experimentos, siendo posible modificar cada parámetro con la mayor precisión simplemente definiendo un nuevo archivo *XML*.

3.5. SOX SourceForge

SOX es un software multiplataforma capaz de convertir varios formatos de audio a otros formatos. La herramienta posee la capacidad de grabar audio en la mayoría de plataformas. Adicionalmente *SOX* permite la edición del audio grabado mediante concatenación, repetición, cortado y pegado de audio así como normalización de audio. El software incluye también la funcionalidad de síntesis de ficheros simples de audio y de eliminación de ruido.

La herramienta funciona con diferentes formatos de archivo, tanto *self-describing*, archivos que contienen una cabecera la cual describe el tipo de audio y codificación empleada, como archivos *raw*, aquellos que no disponen de tal cabecera, y en los que para describir las características de los mismos se deberá emplear a través de la linea de comandos de *SOX*.

Además *SOX Sourceforge* permite el uso de diferentes características de audio. Para los experimentos se ha utilizado audio de un solo canal *mono* y con un tamaño de muestra de 16 bits.

Cabe destacar también que todos los audios utilizados en este proyecto están en formato *WAW*, es decir, que pertenecen al tipo *self-describing*, en cuya cabecera se encuentran las características del mismo.

La herramienta *SOX* permite combinar varios audios, lo que ha resultado de utilidad para generar audios de mayor duración. De esta forma, *SOX* permite la combinación de audios mediante concatenación, mezcla de los audios, fusión o *merge* de varios audios o mediante la utilización de una secuencia de varios audios de naturaleza diferente o *streams*.

A continuación, se incluyen los comandos que es posible utilizar en el software, obtenidos de la pagina web oficial del desarrollador de la herramienta [20]:

```
sox [global-options] [format-options] infile1
    [[format-options] infile2] ... [formatoptions] outfile
    [effect [effect-options]] ...
play [global-options] [format-options] infile1
    [[format-options] infile2] ... [formatoptions]
    [effect [effect-options]] ...
rec [global-options] [format-options] outfile
    [effect [effect-options]] ...
```

3.6. Stereo Mix Plus

Se trata de una herramienta software capaz de crear una tarjeta de audio virtual en un entorno *Windows* (*Windows 7 / 8 / Vista*) que utiliza su propio controlador de audio, sustituyendo el controlador de audio original del sistema. Cabe destacar que este software funciona para todas las tarjetas de audio.

Esta herramienta ha permitido utilizar como dispositivo de grabación la salida a los altavoces, algo que no era posible con el controlador de audio de las máquinas virtuales. En el caso de este proyecto, la herramienta ha sido de utilidad para hacer el envío de trafico de audio a la aplicación de *Skype*, gracias al audio enviado a los altavoces por parte de la herramienta de reproducción de sonido SOX.

Una vez descargada la herramienta, para habilitarla como la tarjeta de audio virtual en nuestro sistema *Windows* se han seguido los siguientes pasos:

- En el área de notificaciones de *Windows*, se hace click derecho y se va a la pestaña de *Dispositivos de grabación*.
- Se abre el panel de Dispositivos de grabación. Haciendo click derecho en este panel, las opciones *Ver dispositivos desactivados* y *Ver dispositivos desconectados* están por defecto desactivadas. Activándolas aparece la opción de *Stereo Mix*.

- Hacemos click derecho sobre la opción *Stereo Mix* y activamos la opción *Activar*.

Capítulo 4

Planificación y Costes

4.1. Descripción de las fases de desarrollo

Este Trabajo de Fin de Grado se ha realizado siguiendo las siguientes fases de desarrollo:

4.1.1. Fase 1 - Investigación y análisis

En esta fase se ha seguido un riguroso proceso de estudio de diferentes tipos de documentación, bibliografía y artículos, con el objetivo de estudiar la metodología a seguir en el desarrollo de los experimentos. Además, se han llevado a cabo pruebas para determinar de forma experimental el correcto funcionamiento de todas las herramientas utilizadas a lo largo de este proyecto.

Las principales fuentes utilizadas a lo largo de esta fase pueden consultarse en la bibliografía de este proyecto.

4.1.2. Fase 2 - Estudio de las herramientas de desarrollo

En esta fase se realizó un exhaustivo estudio de las diferentes herramientas empleadas a lo largo del desarrollo del proyecto, las cuales quedan debidamente reflejadas en el capítulo de Herramientas. Fue necesario aprender cómo funcionaban cada una de ellas por separado, y en conjunto, para comprender cómo se podían llevar a cabo los experimentos correspondientes a la fase de pruebas de este proyecto.

4.1.3. Fase 3 - Resolución de problemas y búsqueda de herramientas alternativas

Esta fase ha servido para corregir todos los errores que han surgido en la herramienta principal, así como en otras herramientas secundarias.

También se han determinado nuevas herramientas sustitutas a las herramientas secundarias en las que habían surgido errores, y se han realizado los testeos iniciales de dichas herramientas para comprobar qué funcionasen correctamente.

Las nuevas herramientas utilizadas son *SOX* [20] en sustitución a la aplicación *Pamela* [21], como herramienta de grabación de audio, y *Stereo Mix Plus* [22], una tarjeta virtual de audio que permitió utilizar como dispositivo de grabación la salida de los altavoces.

4.1.4. Implementación

Tercera fase de desarrollo que ha servido para implementar nuevas funcionalidades al proyecto que sirven para hacer correctamente los experimentos conducentes al estudio de la calidad de experiencia (*QoE*) y trafico de *Skype*.

Toda el desarrollo de esta fase queda expuesta en el capítulo de Pruebas de este Trabajo de Fin de Grado.

4.1.5. Pruebas

Fase clave del proyecto, en la que se han llevado a cabo las pruebas siguiendo la metodología que se determinó en la primera fase del trabajo.

4.1.6. Documentación

Última fase del Trabajo de Fin de Grado, la cual ha servido para escribir el presente documento del proyecto.

A continuación, se procede a realizar una descripción del tiempo planificado inicialmente previsto para el desarrollo de cada una de las fases de este proyecto.

En el siguiente Cuadro 4.1 se procede a realizar una exposicion acerca de los tiempos previstos inicialmente para la realización del presente proyecto.

Fase o subfase	Nombre	Fecha aproximada de comienzo	Fecha aproximada de finalización	Resultado final
1	Investigación y análisis	09/2018	10/2018	1 mes
2	Estudio de las herramientas de desarrollo	15/10/2018	11/2018	Dos semanas
3	Resolución de problemas y búsqueda de herramientas alternativas	11/2018	15/01/2019	2 meses y 2 semanas
4	Implementación	15/01/2019	15/04/2019	3 meses
5	Pruebas y simulaciones	15/04/2019	15/05/2019	1 mes
6	Documentación	15/05/2019	15/06/2019	1 mes

Cuadro 4.1: Planificación inicial de las fases del proyecto

El tiempo de desarrollo dedicado a cada fase del proyecto queda también ilustrado mediante el siguiente diagrama de Gantt:

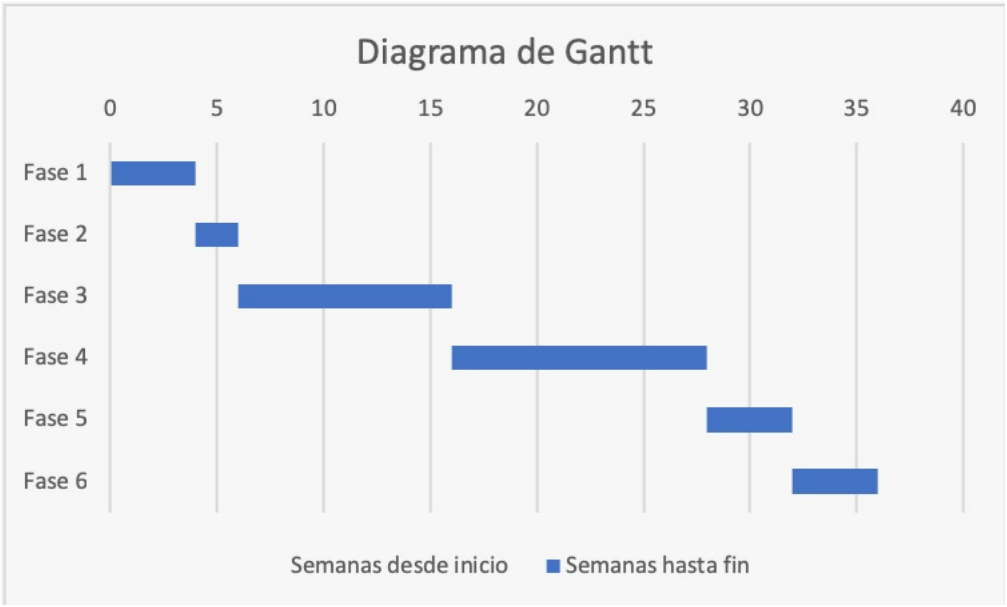


Figura 4.1: Diagrama de Gantt del proyecto

4.2. Recursos

4.2.1. Humanos

- D. Jorge Navarro Ortiz, Profesor Titular de Universidad del área de Ingeniería Telemática del Departamento de Teoría de la Señal, Telemática y Comunicaciones. Tutor del proyecto.

- Julio Elvira del Castillo, alumno del Grado en Ingeniería de Tecnologías de Telecomunicación en la Universidad de Granada. Autor del proyecto.

4.2.2. Hardware

MacBook Pro 2017.

4.2.3. Software

- Software de virtualización *VirtualBox 6.0.4*. Utilizado para implementar ambas máquinas virtuales que han funcionado como servidor y cliente.

- *Windows 7*, Sistema Operativo utilizado en ambas máquinas virtuales, tanto la del cliente como la del servidor. Este Sistema Operativo ha sido el único utilizado para el desarrollo de todo el Trabajo de Fin de Grado.

- *Skype*, aplicación telemática especializada en ofrecer videoconferencia y llamadas sobre *IP* (*VoIP*) utilizada durante todo el Trabajo para la realización de las simulaciones. No obstante, cualquier otra herramienta de llamadas VoIP se podría incluir de forma sencilla para poder analizar su comportamiento.

- *Windows Network Emulator Toolkit*, herramienta de simulación de redes utilizada para simular las llamadas con cualquier condición requerida. Las especificaciones de los parámetros se realizan mediante la creación de archivos *XML*.

- Reproductor multimedia de libre distribución *VLC*, el cual sirve para la reproducción del audio elegido en la máquina cliente.

- *IDE Neatbeans 8.1.* Entorno de desarrollo utilizado para la programación del cliente y del servidor en Java.

- *Java jfreechart 1.0.19*, que realiza la representación gráfica de los resultados obtenidos, generando gráficas de los valores almacenados de cada parámetro.

- *SOX*, herramienta de grabación de audio utilizable por línea de comandos.

- *Stereo Mix Plus*, *software* que nos permite crear una tarjeta de audio virtual en nuestras máquinas virtuales Windows 7.

4.3. Fases de desarrollo

4.3.1. Revisión del estado del arte

Primer apartado en el que se realiza una revisión de las diferentes aplicaciones en el mercado.

4.3.2. Especificación de requisitos

Se realiza un estudio exhaustivo de todos los requisitos, tanto funcionales como no funcionales, que deberán ser tenidos en cuenta e implementados durante todo el desarrollo de este Trabajo de Fin de Grado. Adicionalmente se especifica el objetivo u objetivos principales que se tuvo a la hora de implementar la solución.

4.3.3. Investigación e implementación

Desarrollo del esquema de estudio de la calidad de experiencia, para lo cual se ha llevado a cabo un proceso de estudio de todo tipo de bibliografía relacionada con este campo, con el objetivo de definir los umbrales de actuación para todos los parámetros.

4.3.4. Evaluación y pruebas

Uno de los apartados clave de este Trabajo Fin de Grado, en el que se han llevado a cabo numerosos experimentos y simulaciones, con el objetivo de llevar a cabo el estudio detallado de la calidad de experiencia (QoE) de *Skype* en todas las condiciones de red.

4.3.5. Documentación

Se lleva a cabo la documentación exhaustiva en el correspondiente informe de todo el proceso llevado a cabo para el desarrollo de este Trabajo Fin de Grado.

4.4. Estimación de costes

4.4.1. Herramientas

En este apartado se lleva a cabo el estudio de costes para la totalidad de las herramientas de hardware y software utilizadas para el desarrollo de este trabajo.

Herramientas hardware		
	Unidades	Precio
MacBook Pro	1	1505.59 euros * 0.1875
	Precio total hardware	282.2 euros

Cuadro 4.2: Coste del *hardware*

Herramientas software		
	Unidades	Precio
Licencia de macOS Mojave	1	0 euros (incluida en hardware)
Licencia de Windows 7	2	270 euros
Netbeans 8.1	2	0 euros
Skype 8.44	2	0 euros
Stereo Mix Plus	2	0 euros
SOX	2	0 euros
	Precio total software	270 euros

Cuadro 4.3: Coste del *software*

Cabe destacar que, en las herramientas *hardware* se ha incluido tan sólo la parte proporcional. Es decir, suponiendo que un ordenador tiene una vida útil de 4 años, y este se ha usado durante 9 meses, la parte imputada al proyecto es la proporcional, y no el total del coste del PC.

Pese a que las licencias utilizadas de Windows 7 tienen un coste de mercado de 270 euros, el Departamento TSTC tenía licencias de volumen, por lo que no ha sido necesario pagarlas, siendo por lo tanto el coste de las mismas de 0 euros. Se ha tratado de reducir los costes de software eligiendo, en la medida de lo posible, aquel software que cumpliera el cometido y fuera de libre distribución. En su defecto se han utilizado versiones con menores funcionalidades pero de licencia gratuita, como en el caso de la herramienta *SOX*.

Por lo tanto, esto nos da un coste total de 1505.59 euros, en los cuales quedan incluidos el ordenador de trabajo. Sin embargo, observando el desglose en hardware y software del presupuesto, se ve con claridad que el total del presupuesto va hacia la herramienta hardware utilizada para el desarrollo de este proyecto, esto se debe a tres razones.

- La primera razón es que la licencia de *Windows 7* se obtuvo de forma gratuita, gracias a la Universidad de Granada, cabe destacar sin embargo que, quizás, alguien que no tuviera esta ventaja, debería pagar dos licencias de *Windows 7*, las cuales tienen un precio unitario de 135 euros, luego las dos licencias habrían tenido un precio total de 270 euros.

- La segunda razón es que todas las herramientas utilizadas en los entornos simulados, han sido de libre distribución (véase *Netbeans 8.1* y *Stereo Mix Plus*), o se ha utilizado la versión gratuita de test con el objetivo de abaratar costes (véase *SOX Sourceforge*).

- La tercera razón es que el Sistema Operativo del ordenador utilizado para el desarrollo del proyecto (*macOS Mojave*) viene incluido en el precio del hardware.

4.4.2. Recursos Humanos

En la presente sección se procede al cómputo del coste de los Recursos Humanos. Se describen en el Cuadro 4.4.

Para el coste consideramos una jornada de 5 días laborables semanales, por una jornada laboral diaria de 8 horas para el autor del proyecto, más 30 horas en total para el tutor. Se supone además un coste de 25 euros la hora para el autor del proyecto, así como un coste de 50 euros la hora para el tutor.

Recursos Humanos			
Fase	Duración	Días Laborables	Coste autor
Investigación y análisis	1 mes	20 días	4000 euros
Estudio de las herramientas de desarrollo	2 semanas	10 días	2000 euros
Investigación y análisis	1 mes	20 días	4000 euros
Resolución de problemas y búsqueda de alternativas	2 meses y 2 semanas	50 días	10000 euros
Implementación	3 meses	60 días	12000 euros
Pruebas y simulaciones	1 mes	20 días	4000 euros
Documentación	1 mes	20 días	4000 euros
		Total	40000 euros

Cuadro 4.4: Coste de los Recursos Humanos

Por lo tanto, teniendo un coste total de 1500 euros para el tutor, así como un coste total de 40000 euros para el autor, el coste de los Recursos Humanos del presente proyecto sería de 41500 euros.

Capítulo 5

Análisis y diseño

5.1. Entorno de experimentación

En el presente proyecto se han realizado las diferentes pruebas, relatadas en el capítulo de Pruebas, en un entorno compuesto por dos máquinas virtuales [6] Windows 7, sobre un host-anfitrión MacOS X Mojave.

Ambas máquinas virtuales han hecho las veces de Cliente y Servidor de la llamada, y constaban tanto de *Skype* en su versión 8.46 como de las herramientas necesarias para llevar a cabo la totalidad de los experimentos, las cuales se describen en el capítulo de Herramientas.

Cabe destacar que, al principio del presente proyecto, la realización de los experimentos se basaban en unas herramientas, algunas de las cuales hubo que cambiar para poder realizar los experimentos del presente proyecto correctamente. A continuación se describen las herramientas antiguas que fue necesario cambiar, y se indica por cuales de las nuevas se cambiaron para construir el entorno de experimentación.

- Pamela [21], era la herramienta utilizada para la grabación de la conversación de *Skype*. Al dejar de funcionar la API de *Skype*, Pamela dejó de funcionar, al no detectar las conversaciones REF. Fué por esto que Pamela se sustituyó por SOX [20], herramienta que nos permite grabar el audio de la llamada en nuestro entorno de simulación. SOX se explica en profundidad en el capítulo de Herramientas del presente prpyecto. Cabe señalar que para poder utilizar la salida a los altavoces como dispositivo de grabación fue necesario introducir un programa que crea una tarjeta de audio virtual que permite hacerlo, ya que el driver de audio de nuestras maquinas virtuales no permitía hacer esto.

- Clisk [23], *Command-Line Interface for Skype* es una herramienta, la cual se basa en línea de comandos, que permite el responder a las llamadas de *Skype* de forma automática. Sin embargo, en una actualización de *Skype*, esta herramienta dejo de servirnos, debido a que ya no funcionaba ninguna API conocida de Skype para contestar las llamadas o para colgar.. Es por esto que se sustituyo por unos scripts basados en powershell de Windows que simulaban los *shortcuts* de responder la llamada y de colgar la misma.

De esta forma, además de la sustituación de las ya mencionadas herramientas cabe señalar que en el código de la herramienta en Java fue necesario introducir pausas, para dejar tiempo suficiente ente los eventos de llamar, descolgar y comenzar a capturar el audio.

5.2. Diseño de la metodología de la experimentación

En este apartado se procede a explicar el desarrollo de una metodología para la realización de experimentos. Dicha metodología tenía dos objetivos finales:

- Facilitar el estudio del autor acerca de la calidad de experiencia (QoE).
- Desarrollar un modelo que reflejase de manera intuitiva el tráfico y la calidad de experiencia de la aplicación.

La idea es generar dos matrices multidimensionales para el caso de Throughput VS Retardo y otras dos de manera análoga para el caso de Throughput VS Tasa de pérdida de paquetes. De esta forma, en una de las dos matrices se recogen los diferentes valores del tráfico generado por *Skype*, mientras que en la otra se recoge la calidad de experiencia (QoE) experimentada. Es así como para cada celda de estas matrices, se tiene el valor promedio,bien del tráfico generado por la aplicación, bien de la calidad de experiencia (QoE) experimentada, para el experimento realizado en las métricas de red dadas.

Inicialmente las matrices tendrían como dimensiones, el *throughput* de red, el retardo o *delay* y las perdidas, sin embargo, dado que el espacio a barrer sería demasiado grande, se ha optado por generar las matrices correspondientes a los casos de Throughput VS Retardo y al de Throughput VS Tasa de pérdida de paquetes.

Se acabó llegando a la conclusión de realizar un modelo de tráfico y otro de calidad de experiencia, ambos expresados en forma de matrices, donde las columnas de dichas matrices fueran el *throughput* de la llamada (expresado en kbps) y las filas de dichas matrices fueran bien la tasa de perdidas de paquetes (expresada en tanto por ciento) o el retardo o *delay* (expresado en milisegundos).

De esta forma, se busca el barrido de un espacio tridemensional (*throughput* de red, retardo y tasa de pérdida de paquetes) de una forma mas sencilla, barriendo dos espacios bidimensionales, el espacio de*throughput* de red y retardo y el de *throughput* de red y tasa de paquetes perdidos.

Es mediante estos barridos, que podemos conocer el MOS (expresado a traves del *eModel*) y el *throughput* generado por *Skype* en las condiciones de red mas típicas, y expresarlas en forma intuitiva a través de las ya mencionadas matrices.

Con esta metodología ya creada, se procedió a la realización de los ficheros de configuración para los experimentos realizados, conducente a realización de las pruebas. Dicha implementación se detalla exhaustivamente en el siguiente capítulo del proyecto Implementación.

Capítulo 6

Implementación

En el presente capítulo se procede a realizar una descripción detallada de la implementación llevada a cabo para para desarrollar las pruebas que estudien la calidad de experiencia (QoE) y generación de tráfico de la herramienta *Skype*, descritas en el capítulo posterior.

Cabe destacar que, como paso previo a la realización de la implementación, se ha llevado a cabo un profundo estudio de diferentes metodologías utilizadas en otros casos para el estudio del rendimiento en redes multimedia, y mas concretamente en aplicaciones de videollamada o videoconferencia. Para ello se han utilizado numerosos estudios, recogidos en publicaciones que han sido debidamente referenciadas en el capítulo de Análisis y Diseño.

Para el estudio del rendimiento de *Skype* se ha utilizado una herramienta implementada en Java. A continuación se procede a realizar una exposición acerca del funcionamiento de dicha herramienta.

- Arquitectura: La arquitectura de la herramienta consta de dos maquinas virtuales, una de las cuales actúa como cliente y la otra como servidor.

  Es la máquina virtual que actúa como cliente la que se, mediante una interfaz gráfica, permite al usuario de la herramienta fijar las métricas de red. Así como del posterior almacenamiento de los datos recogidos durante la llamada.

  A su vez, la máquina servidor se encarga del establecimiento y grabación de la llamada, y de indicar al emulador de red utilizado de que manera se modifican los parámetros de red.

- Realización de las llamadas: Para realizar las llamadas correctamente

se debe proceder de la siguiente manera. Primero se debe poner en marcha el servidor, tras lo cual este quedará en estado de RECIBIENDO. Una vez hecho esto se procede a encender el cliente, al cual se le especificarán las métricas de red adecuadas para cada experimento. Una vez la llamada finaliza, los datos recogidos, así como las gráficas, objetivas y subjetivas, quedan recogidas en la máquina virtual del cliente.

Sobre esta herramienta se han realizado algunas modificaciones, que incluyen la obtención de ficheros con los valores de las métricas objetivas y subjetivas obtenidas por la herramienta. Para generar estos ficheros, se ha implementado un nuevo método *crearFichero()*.

A continuación se expone el método *crearFichero()*:

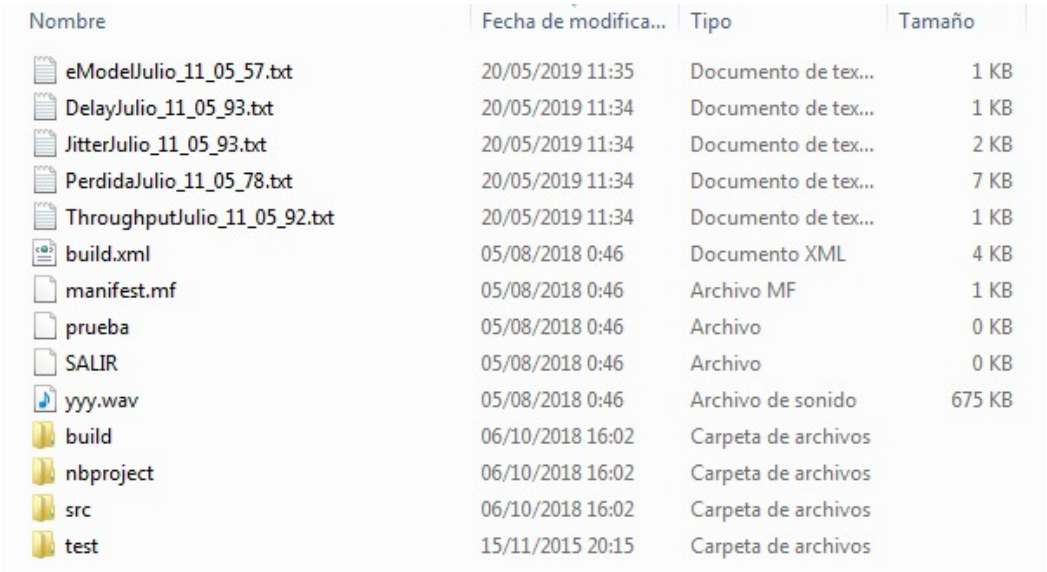
```
1  public void crearFichero(String nombreFichero, ArrayList ejeX,
       ArrayList ejeY){
2      DateTimeFormatter format = DateTimeFormatter.ofPattern('HH_MM_SS')
           ;
3      try (FileWriter fileWriter = new FileWriter(nombreFichero + '_' +
4          LocalDateTime.now().format(format) + '.txt')){
5              BufferedWriter bufferedWriter = new BufferedWriter(
                   fileWriter);
6              int limit = ejeX.size() < ejeY.size() ? ejeX.size() : ejeY
                   .size();
7              for (int i = 0 ; i < limit ; i++){
8                  bufferedWriter.write(ejeX.get(i).toString() + ' ' +
                       ejeY.get(i).toString());
9                  bufferedWriter.newLine();
10             }
11             bufferedWriter.close();
12     } catch (IOException e){
13         new RuntimeException('Error message', e).printStackTrace();
14     }
15 }
```

Así, este método *crearFichero()* nos permite obtener, al fin de cada experimento, un almacenamiento en fichero de texto (*.txt*) de los valores instantáneos, captados en intervalos de 10 segundos para las métricas representadas, es decir, *throughput* generado por *Skype*, tasa de pérdidas, retardo, *jitter* y *MOS*. El almacenamiento en ficheros de texto de todos estos valores era de vital importancia, ya que permite el posterior procesado de la información obtenida en cada experimento. Todo el proceso de procesado de la información de cada experimento, así como el diseño detallado del plan de experimentación llevado a cabo para el estudio, quedan debidamente detallados en el posterior capítulo de Pruebas.

Así pues, se ha desarrollado el método *crearFichero()*, dicho método crea los ficheros de texto en los cuales son almacenados los valores instantáneos de los parámetros en la ubicación de la máquina virtual de *Windows* (C:

TestSkype
Cliente).

Los ficheros generados nos permiten un posterior procesado automático mediante Excel o MatLab.



Nombre	Fecha de modifica...	Tipo	Tamaño
eModelJulio_11_05_57.txt	20/05/2019 11:35	Documento de tex...	1 KB
DelayJulio_11_05_93.txt	20/05/2019 11:34	Documento de tex...	1 KB
JitterJulio_11_05_93.txt	20/05/2019 11:34	Documento de tex...	2 KB
PerdidaJulio_11_05_78.txt	20/05/2019 11:34	Documento de tex...	7 KB
ThroughputJulio_11_05_92.txt	20/05/2019 11:34	Documento de tex...	1 KB
build.xml	05/08/2018 0:46	Documento XML	4 KB
manifest.mf	05/08/2018 0:46	Archivo MF	1 KB
prueba	05/08/2018 0:46	Archivo	0 KB
SALIR	05/08/2018 0:46	Archivo	0 KB
yyy.wav	05/08/2018 0:46	Archivo de sonido	675 KB
build	06/10/2018 16:02	Carpeta de archivos	
nbproject	06/10/2018 16:02	Carpeta de archivos	
src	06/10/2018 16:02	Carpeta de archivos	
test	15/11/2015 20:15	Carpeta de archivos	

Figura 6.1: Ubicación de los ficheros generados por el método *crearFichero()*

En este capítulo de implementación se detalla también el proceso seguido para darle a los diferentes ficheros *XML* las diferentes métricas de red que se pasan a la herramienta Network Emulator Toolkit en cada experimento.

XML (*Extensible Markup Language*) nos define un conjunto de reglas para codificar documentos en un formato que es legible tanto por un humano como por una máquina, en este caso la herramienta *SOX Sourceforge*.

Cada fichero *XML* consta de los siguientes parámetros:

- Declaración *XML*: Al comienzo de cada archivo *XML* se utiliza una declaración que describe el propio fichero *XML*, en nuestro caso da la información de que se trata de un fichero con especificación de *XML 1.0* (la mas común actualmente) y que utiliza una codificación *UTF-8*.
- Elemento *XML*: Un elemento es un componente lógico que empieza con una etiqueta de entrada (*start-tag*) y finaliza con otra etiqueta (*end-tag*).

- Atributo *XML*: Construcción formada por el par nombre-valor que se encuentra dentro de una etiqueta de entrada (*start-tag*) *XML* y que permite da valor a diversos parámetros de nuestro fichero. En este caso por ejemplo se utiliza para especificar el valor del *throughput* utilizado, o del retardo introducido.

Para ilustrar todo lo anteriormente expuesto sobre archivos *XML* se expone a continuación un ejemplo de fichero *XML* que sirve como ejemplo para ilustrar lo anteriormente expuesto.

```
<?xml version="1.0" encoding="UTF-8"?>
<Emulation xmlns="http://research.microsoft.com/asia">
    <VirtualChannel DispatchType="packet" name="VirtualChannel 1">
        <FilterList>
            <Filter name="FILTER_4" not="0"></Filter>
        </FilterList>
        <VirtualLink instances="1" name="LINK_1">
            <LinkRule dir="upstream">
                <Bandwidth>
                    <Speed unit="kbps">25</Speed>
                    <QueueManagement>
                        <NormalQueue>
                            <Size>100</Size>
                            <QueueMode>packet</QueueMode>
                            <DropType>DropTail</DropType>
                        </NormalQueue>
                    </QueueManagement>
                </Bandwidth>
                <Loss>
                    <Periodic>
                        <PerPackets>20</PerPackets>
                    </Periodic>
                </Loss>
            </LinkRule>
            <LinkRule dir="downstream">
                <Bandwidth>
                    <Speed unit="kbps">25</Speed>
                    <QueueManagement>
                        <NormalQueue>
                            <Size>100</Size>
                            <QueueMode>packet</QueueMode>
                            <DropType>DropTail</DropType>
                        </NormalQueue>
                    </QueueManagement>
                </Bandwidth>
                <Loss>
                    <Periodic>
                        <PerPackets>20</PerPackets>
                    </Periodic>
                </Loss>
            </LinkRule>
        </VirtualLink>
    </VirtualChannel>
</Emulation>
```

El fichero expuesto es uno utilizado para realizar una simulación con un

throughput de 25 kbps (especificado en el atributo *Speed unit*) y con una pérdida cada 20 paquetes (especificado en el atributo *PerPackets*, lo cual daría unas tasa de perdidas del 5 % para este caso.

Se han modificado tantos ficheros *XML* como variaciones de los parámetros de *throughput* de red, retardo y tasa de pérdidas, por lo que al inicio de cada simulación simplemente se ha hecho una llamada al correspondiente fichero *XML*.

Por último, para una correcta implementación ha sido necesario llevar a cabo una serie de modificaciones, las cuales se detallan a continuación.

Debido a una actualización de *Skype*, dejo de funcionar una de sus *APIs* utilizada por la herramienta *Pamela*. Esta herramienta se utilizaba para grabar el audio de las conversaciones.

Esto se solucionó modificando la herramienta que se utilizaba para la grabación de audio. Así, se pasó a utilizar la herramienta *SOX* descrita en el capítulo de Herramientas.

Esta modificación produjo a su vez un nuevo problema, ya que el controlador de audio que viene por defecto en las máquinas virtuales no permite usar como dispositivo de grabación la salida a los altavoces. Esto se solucionó introduciendo una nueva tarjeta virtual de audio, *Stereo Mix Plus*, descrita detalladamente en el Capítulo de Herramientas.

Una vez hechas estas modificaciones, las cuales solucionaban el problema anterior, fue necesario a su también cambiar cómo la herramienta llamaba a ciertas funcionalidades de *Skype*, como la realización de una llamada, contestar a la misma y colgar al finalizar la conversación. Esto se debe a que las APIs de *Skype* conocidas dejaron de funcionar tras la actualización.

Para poner solución a esto, aprovechamos que *Skype* tiene definidos atajos de teclado (*shortcuts*) para las principales funcionalidades, como la realización de llamadas (CTRL + MAY + P), contestar (CTRL + MAY + P) y colgar (CTRL + E). Aprovechando dichos *shortcuts*, se han realizados unos *scripts* basados en la herramienta *Powershell* de *Windows* que automatizan la pulsación de dichas teclas, simulando los comandos ya mencionados.

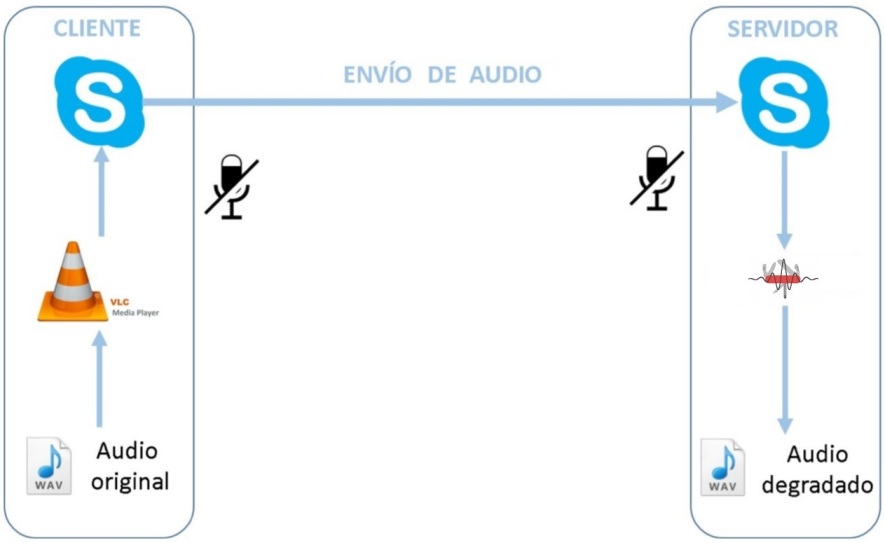


Figura 6.2: Diagrama del envío del audio del cliente al servidor

Capítulo 7

Pruebas

En este capítulo quedan documentadas todas las pruebas que se han ido desarrollando a lo largo de este Trabajo de Fin de Grado. Todas las pruebas han sido orientadas, tal y como queda explicado en capítulos anteriores, al estudio del tráfico y de la calidad de experiencia (*QoE*) de la herramienta *Skype*.

Como paso previo a la realización de este capítulo, cabe destacar la importancia de tener en consideración los factores que podrían afectar a la realización de cualquier experimentación de *VoIP* en un entorno emulado, se trate del utilizado en este proyecto o en cualquier otro. Dichos factores quedan expuestos en el capítulo de Análisis y Diseño.

Teniendo en cuenta los factores ya mencionados, se informa de que todos los datos recogidos, y que sirven como base de estudio en el presente capítulo, han sido meticulosamente contrastados, a lo largo de numerosos experimentos en diferentes condiciones de red.

Para el estudio del tráfico y la calidad de experiencia de *Skype*, se han tenido en cuenta cuatro métricas objetivas de la red: *throughput*, retardo, variaciones en el retardo (*jitter*) y tasa de paquetes perdidos.

Se ha tomado el *throughput* generado por *Skype* como parámetro fundamental para la caracterización del tráfico. Teniendo esto en cuenta, se han desarrollado las pruebas orientadas a caracterizar el tráfico creando dos matrices.

Se han calculado varias matrices, de forma que expresen de forma intuitiva el tráfico de la herramienta *Skype*, así como la calidad de experiencia (*QoE*). De esta forma se obtienen dos matrices para el barrido de Throughput VS Retardo (una que expresa el tráfico generado por *Skype* para este

caso y otra que expresa la calidad de experiencia (QoE)) y dos matrices para el caso de Throughput VS Tasa de paquetes perdidos (las cuales expresan de forma análoga al caso anterior el tráfico generado por la aplicación y la calidad de experiencia (QoE).

Para la creación de estas dos matrices, los experimentos utilizan un fichero de audio en formato *WAW* de 480 segundos de duración, el cual se procesa y se utiliza a modo de audio de la llamada. Cabe recordar que el entorno de las pruebas utiliza un emulador de red. Para evitar degradaciones por culpa de la red real a través de la cual se conectan ambos extremos de la comunicación, los equipos se han puesto muy próximos al *router WiFi* y se han evitado otros tipos de tráfico durante los experimentos. Así, se ha intentado siempre realizar las simulaciones en una red real lo mas óptima posible, para obtener así unos resultados lo mas fidedignos posibles y que reflejen solo las propias variaciones introducidas a través de la herramienta, y no variaciones en parámetros de la red real.

Como paso previo a la realización de los experimentos, se determinaron los rangos de las métricas de red en los cuales se desarrollarían los mismos. Para ello se hicieron una serie de experimentos previos, con el objetivo de determinar unos umbrales dentro de los cuales fuera interesante la realización de las pruebas.

Se llegó a la conclusión de que las métricas de red estarían en los siguientes rangos. De 10 kbps a 200 kbps para el caso del *throughput*, de 0 % a 33 % de tasa de pérdida de paquetes y de 0 milisegundos a 200 milisegundos de retardo.

Una vez determinados los rangos, se procedió a disponer las métricas de red en las matrices, de la forma en que se explica a continuación.

Las dos matrices recogen el *throughput* generado por *Skype* de la siguiente manera:

- La primera matriz recoge el *throughput* generado por *Skype* para cada caso variando dos parámetros, el *throughput* de red y el retardo introducido en la llamada (en milisegundos).
- Se varía el *throughput* de la llamada en sentido descendente de la siguiente forma: 200 kbps - 100 kbps - 50 kbps - 25 kbps - 10 kbps. Este rango de throughput se ha obtenido mediante la realización de experimentos previos para ver el rango de valores.
- Se varía el retado en milisegundos en sentido ascendente de la siguiente forma: 0 ms - 10 ms - 50 ms - 100 ms - 200 ms. Se han realizado

numerosos experimentos previos con el objetivo de obtener este rango para la variación del retardo o *delay*.

- Para todos los valores cruzados se realiza una simulación de 480 segundos de duración y con el audio anteriormente descrito.

- Tras el experimento, se extraen en ficheros de texto el valor instantáneo captado para las cuatro métricas objetivas y el *MOS* cada 10 segundos.

- Estos ficheros de texto se exportan a la herramienta *Excel* para realizar el cálculo del promedio y de la desviación típica del *throughput* generado por *Skype*, así como de la calidad de experiencia (QoE). La obtención de una baja desviación típica permite asegurar que el promedio obtenido para cada experimento es representativo de todos los valores instantáneos obtenidos.

- El valor promedio para cada par de valores cruzados se recoge en una tabla, creando así la primera matriz, que incluye el tráfico generado por *Skype* para los diferentes valores cruzados de *throughput* y retardo en la red.

- Se obtienen también a la vez los valores instantáneos que nos permitirán, a través de su procesado en *Excel*, obtener la matriz correspondiente a la calidad de experiencia (QoE) del primer caso, reflejando el *MOS* para los diferentes valores de *throughput* y retardo en la red.

- Para la creación de la segunda matriz se realiza un proceso análogo, pero esta vez se varían el *throughput* de red y la tasa de paquetes perdidos.

- De manera análoga al caso de la matriz se varia el *throughput* de la llamada en sentido ascendente de la siguiente manera: 200 kbps - 100 kbps - 50 kbps - 25 kbps - 10 kbps.

- La tasa de paquetes perdidos se varia en sentido ascendente de la siguiente manera: 0 % - 5 % - 16 % - 25 % - 33 %.

- Para cada valor cruzado se realiza una simulación, utilizando el audio *test2.waw*, de 480 segundos de duración.

- Se extraen todos los valores de los parámetros captados cada diez segundos de la misma forma que en el caso de la primera matriz, y se calcula el promedio, así como la desviación típica para cada caso.

- Finalmente se recoge en una segunda tabla cada valor cruzado, habiendo creado así la segunda matriz.

- Cabe destacar que de igual forma al caso de *throughput* de la llamada y retardo (*delay*), se obtiene también, después de realizar el procesado por *Excel*, la segunda matriz de calidad de experiencia (*QoE*) que depende del *throughput* de red y la tasa de paquetes perdidos.

Matriz Throughput VS Retardo

A continuación se muestran ambas matrices, así como las gráficas creadas por la implementación de Java que se han considerado mas significativas:

Throughput Retardo	200 kbps	100 kbps	50 kbps	25 kbps	10 kbps
0 ms	50.1	50.2	25.1	19.1	11.8
10 ms	50.8	48.1	21.6	15.2	7.4
50 ms	50.8	43.2	21.9	14.9	6.7
100 ms	50.4	37.7	21.5	15.0	7.5
200 ms	50.8	40.8	21.4	14.7	6.2

Cuadro 7.1: Throughput VS Retardo. *Throughput* generado por *Skype*

Cabe destacar que en el Cuadro 7.1 (Matriz *Throughput* VS Retardo) se exponen los valores promedio de los valores instantáneos captados durante cada simulación cada 10.0 segundos, variando ambas métricas de red.

En dicho cuadro vienen expuestos el *throughput* de red expresado en kilobit por segundo (kbps) y el retardo (ms), así como el *throughput* generado por *Skype* expresado en kilobit por segundo (kbps).

Para comprobar la bondad de los datos anteriormente expuestos, se ha construido una tabla en la que se expone la desviación típica del promedio de cada simulación. A continuación se expone dicha tabla:

Throughput Retardo	200 kbps	100 kbps	50 kbps	25 kbps	10 kbps
0 ms	1.4	1.5	0.9	0.1	0.1
10 ms	0.1	2.3	0.6	0.1	0.2
50 ms	0.2	2.03	2.1	1.96	2.6
100 ms	0.5	3.2	1.4	0.78	2.6
200 ms	0.3	0.2	1.9	2.4	2.3

Cuadro 7.2: Throughput VS Retardo. *D*esviación típica del throughput generado por *Skype*

Observando la tabla de desviación típica, se puede establecer que, al ser la desviación típica considerablemente menor que el valor promedio, es posible establecer que, para cada simulación realizada, el promedio de los datos recogidos nos da mucha información acerca de los mismos.

A continuación se exponen tres gráficas en la que se observan las métricas de red objetivas de la aplicación para unos ejemplos concretos, y se explica lo que sucede en el experimento que refleja la gráfica:

- En la Figura 7.1 se observan las gráficas que representan la evolución de las métricas de red objetivas para el caso de un experimento en el que se tiene una tasa de 200 kbps y un retardo que va en aumento, comenzando en 0 ms, hasta en finalizar en 200 ms. Pese a que el retardo aumenta considerablemente, el throughput generado por *Skype* desciende muy poco, debido a que la tasa de 200 kbps es bastante alta. Por esto, se observa también cómo el *jitter* se mantiene constante.

- En esta Figura 7.2 se reflejan las gráficas representativas de las métricas de red objetivas para el caso de un experimento para el que se tiene una tasa de 100 kbps, y la tasa de pérdida de paquetes se varía de forma análoga al caso de la Figura 7.1, comenzando en 0 ms hasta llegar a 200 ms. Para valores de retardo iguales al caso de la Figura 7.1 (200 kbps), se tiene un *throughput* generado por la aplicación que desciende considerablemente más. Así mismo, se observa cómo el *jitter* varía bastante más que en el caso anterior.

- Esta Figura 7.3 nos sirve para reflejar el caso en el que se tiene una tasa mucho mas baja, de 50 kbps. Para los mismos valores de retardo, se tiene un *throughput* generado por *Skype* que desciende drásticamente. Se observa que las métricas de red son mucho peores, para los mismos valores de retardo o *delay*, debido a que se tiene una tasa notablemente menor.

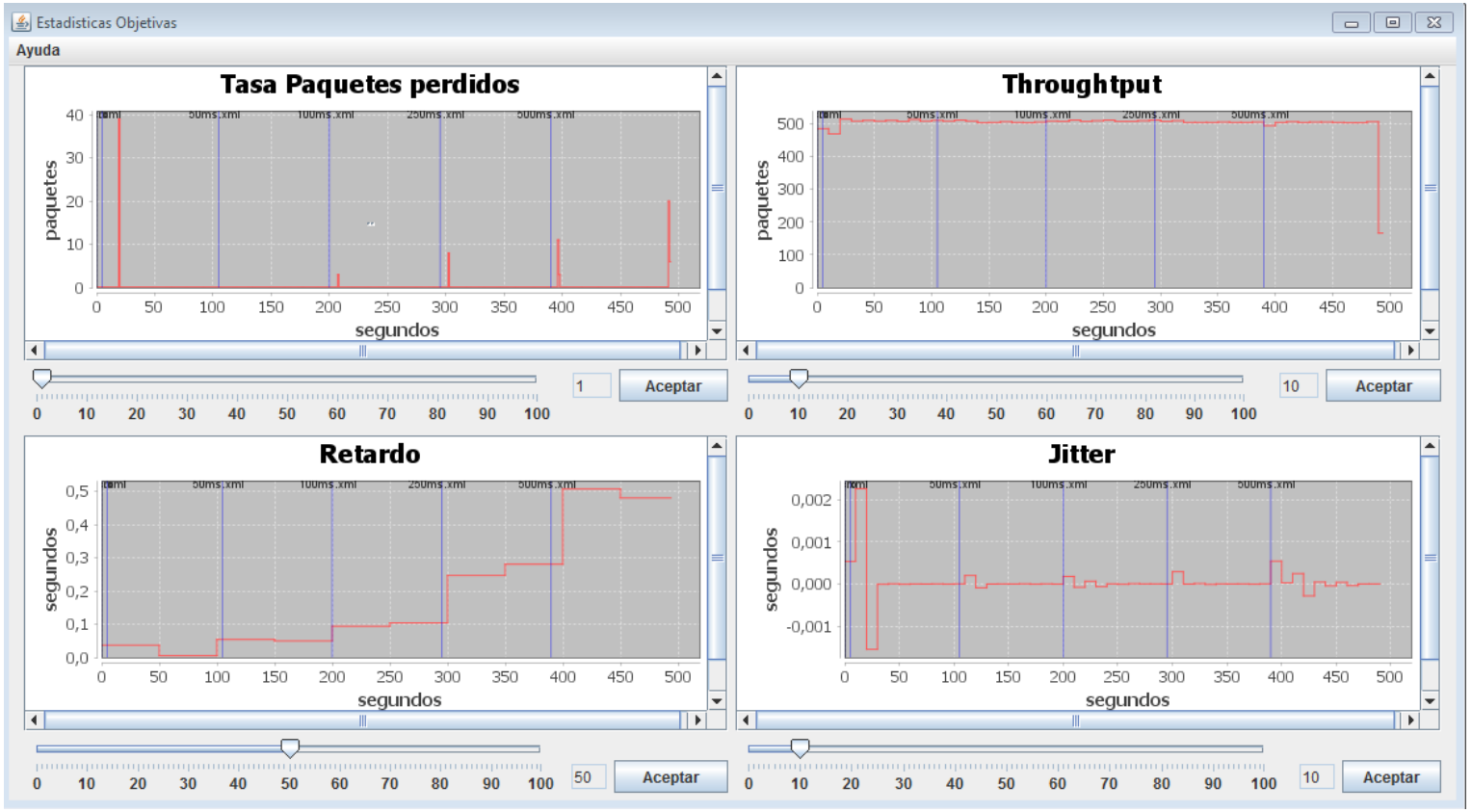


Figura 7.1: Tasa de transferencia de la aplicación para 200 kbps

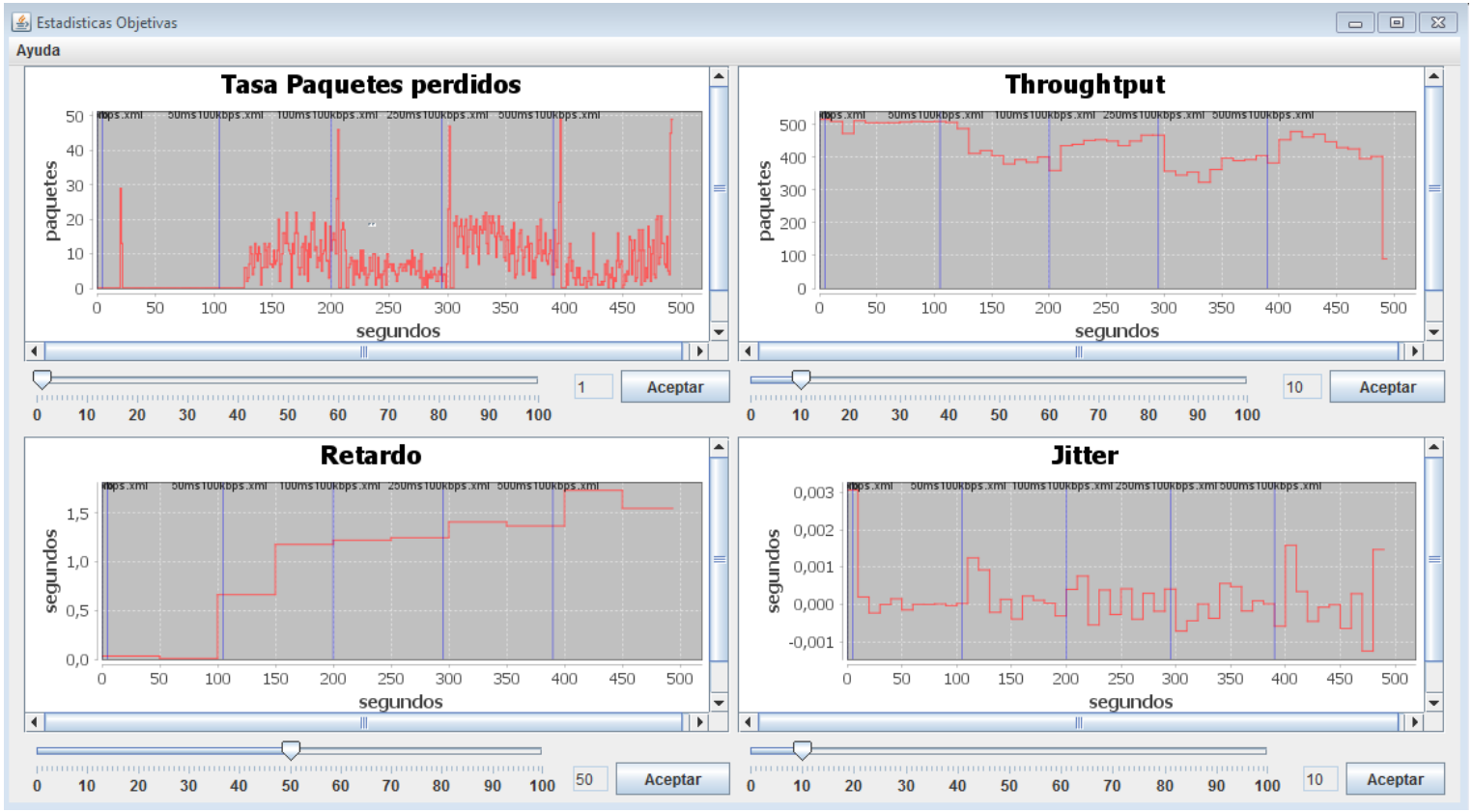


Figura 7.2: Tasa de transferencia de la aplicación para 100 kbps

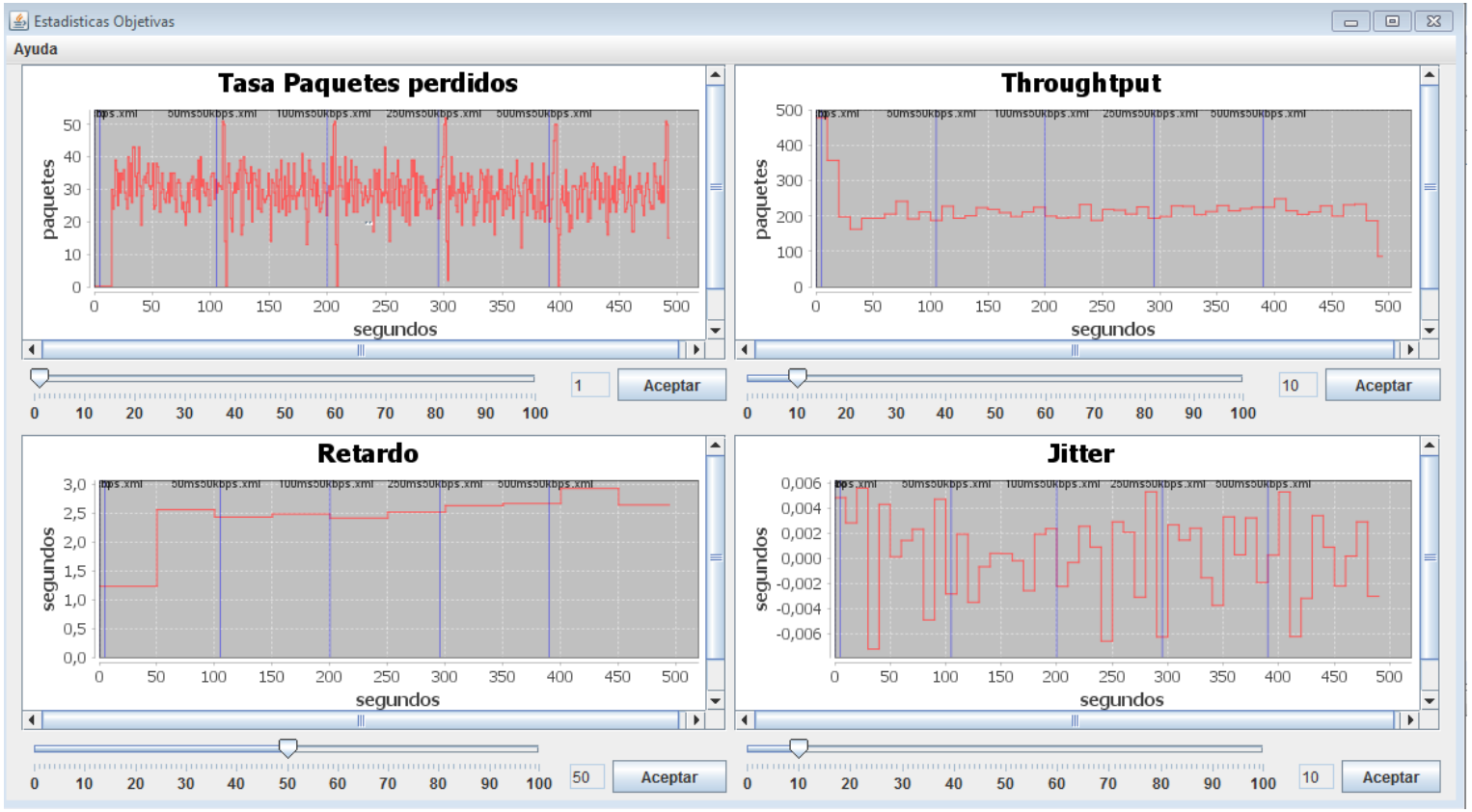


Figura 7.3: Tasa de transferencia de la aplicación para 50 kbps

Es posible concluir como, a medida que las condiciones de red empeoran, bien por un *throughput* de llamada menor, bien por un retardo mayor, *Skype* reacciona bajando el *throughput* generado, de forma que se trata de adaptar el tráfico generado a las condiciones de red. La aplicación *Skype* tiene un *bitrate* variable, que ajusta a las condiciones de la red, como se observa claramente en este apartado.

Matriz Throughput VS Pérdidas

Throughput Pérdidas	200 kbps	100 kbps	50 kbps	25 kbps	10 kbps
0 %	50.1	44.41	22.2	12.7	9.2
5 %	48.06	41.2	20.6	6.4	5.8
16 %	42.4	37.07	20.02	5.7	5.2
25 %	38.9	32.6	18.9	5.3	5.1
33 %	33.2	27.9	18.1	4.6	4.7

Cuadro 7.3: Throughput VS Pérdidas. *Throughput* generado por *Skype*

De igual forma que para el caso de la matriz anterior (*Throughput* vs Retardo), en esta matriz quedan reflejados los datos del tráfico generado por *Skype*, adaptándose a las variaciones de ambas métricas de red.

De forma análoga, para comprobar la fiabilidad de los datos expuestos, se visualiza la tabla de la desviación típica:

Throughput Pérdidas	200 kbps	100 kbps	50 kbps	25 kbps	10 kbps
0 %	1.3	2.3	9.6	9.5	5.05
5 %	0.7	1.8	2.3	1.6	2.7
16 %	0.3	1.6	2.4	0.8	0.4
25 %	0.6	2.2	1.0	0.7	1.47
33 %	0.4	1.4	2.0	0.6	0.2

Cuadro 7.4: Throughput VS Pérdidas. *D*esviación típica del throughput generado por *Skype*

Es posible comprobar, como, de la misma forma que sucedía en el caso de *Throughput* VS Retardo, la desviación típica es de varios ordenes menor

a los valores promedio del trafico generado, luego se puede establecer que los promedios expuestos en la matriz son muy representativos del total de los datos captados durante los experimentos.

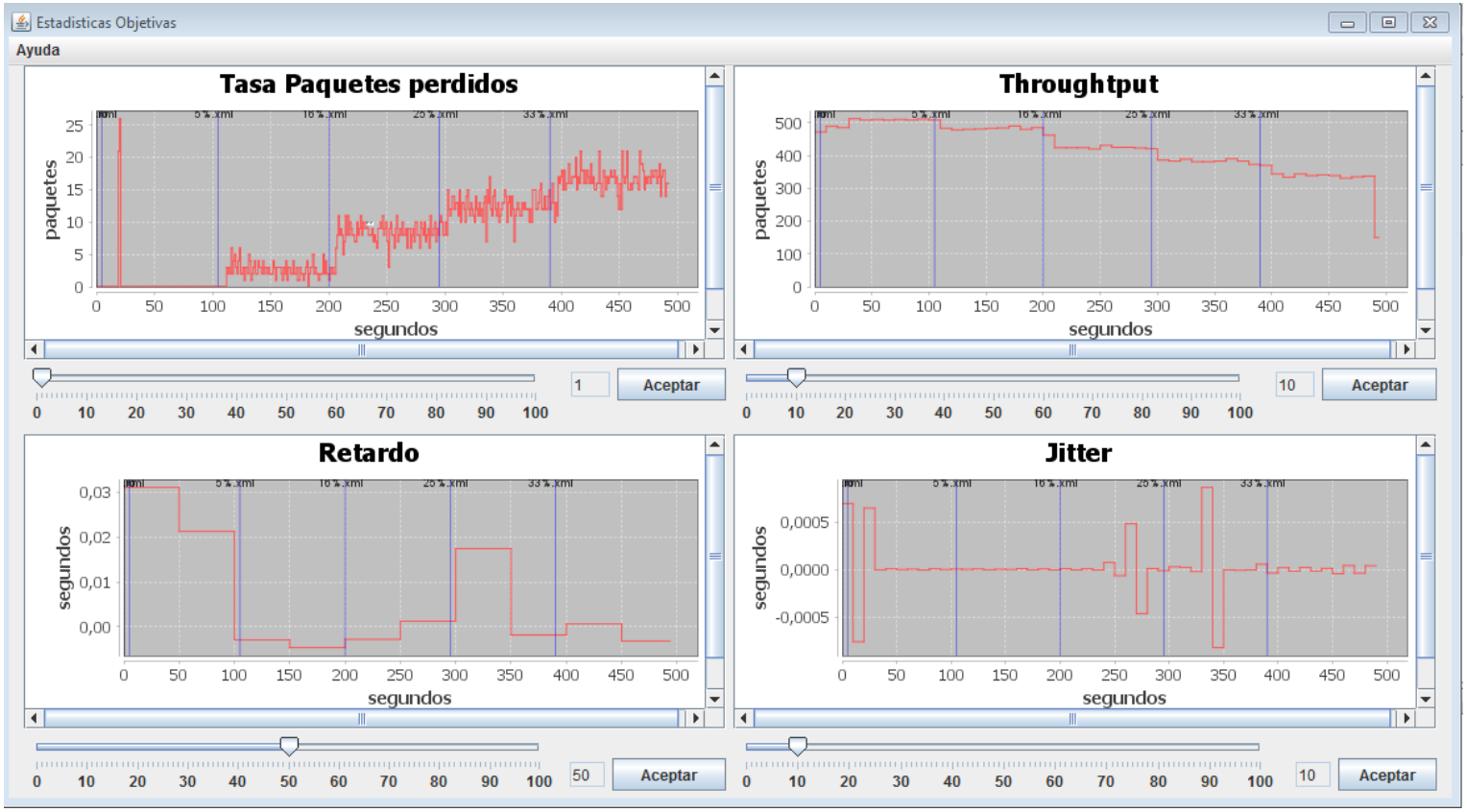


Figura 7.4: Pérdidas de la aplicación para 200 kbps

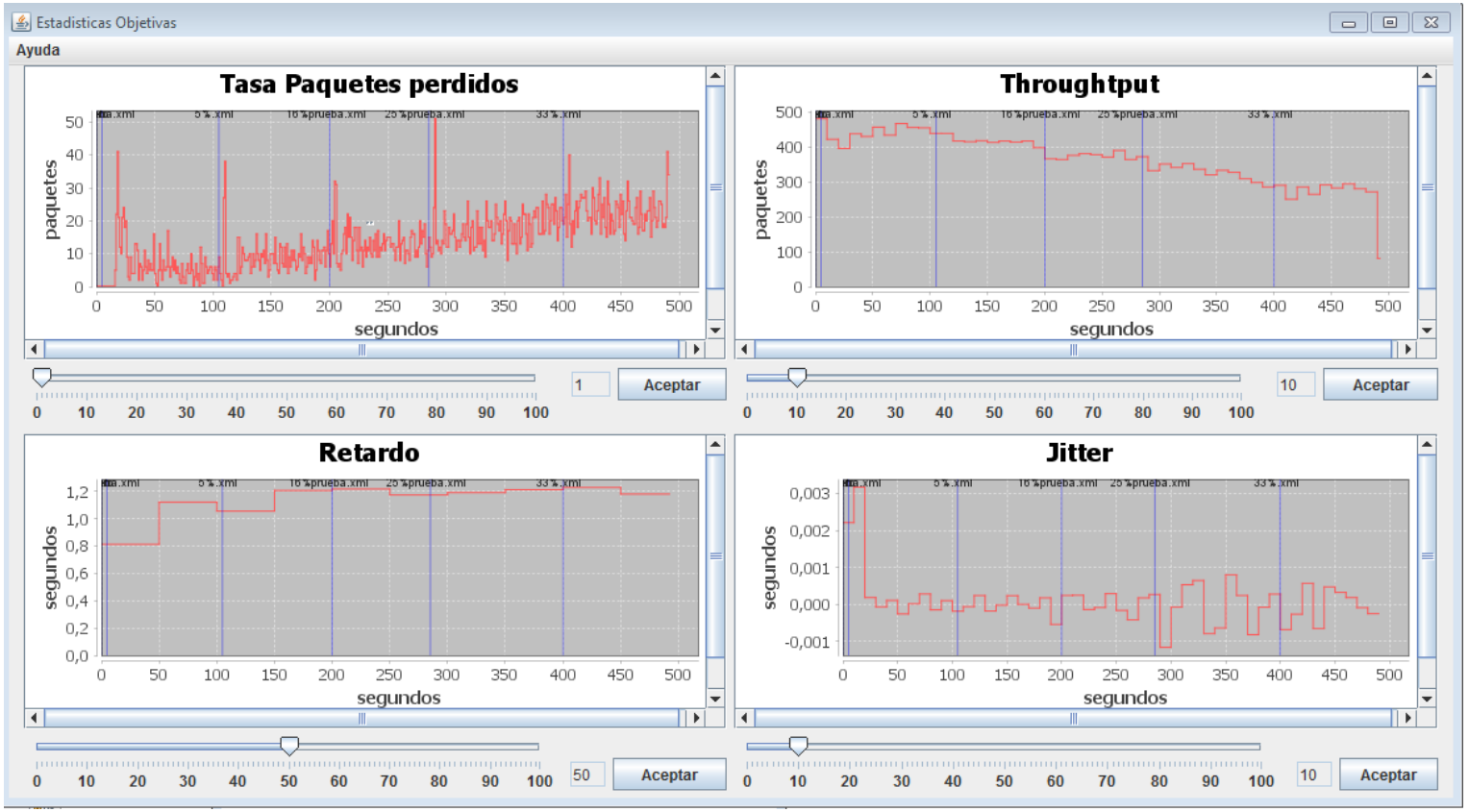


Figura 7.5: Pérdidas de la aplicación para 100 kbps

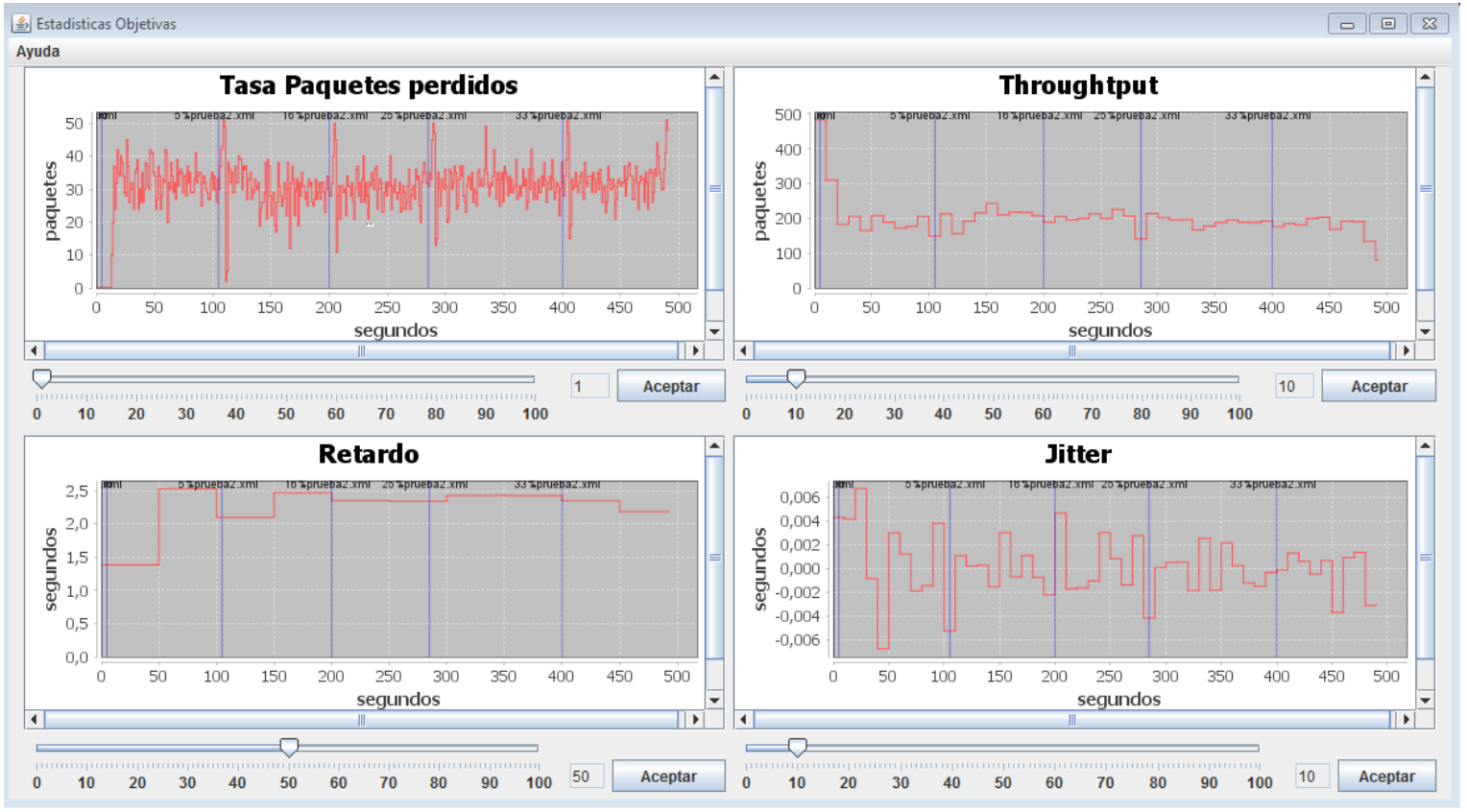


Figura 7.6: Pérdidas de la aplicación para 50 kbps

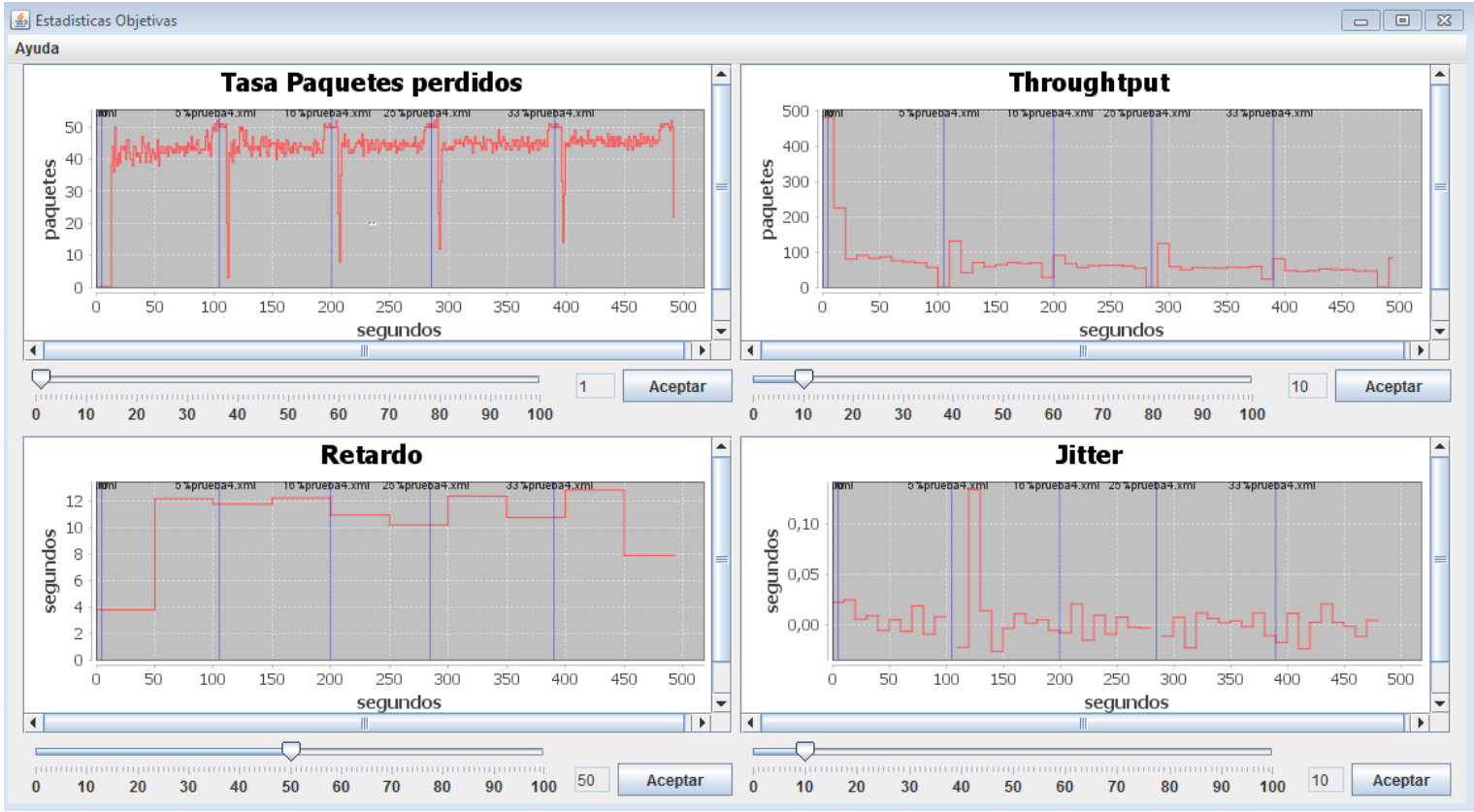


Figura 7.7: Pérdidas de la aplicación para 10 kbps

Como se observa en la Figura 7.5 , a medida que la tasa de paquetes perdidos aumenta del 0 %, hasta el 33 %, el tráfico generado (expresado en intervalos por segundo) va disminuyendo considerablemente. De esta forma se observa de forma gráfica el como un aumento en la tasa de paquetes perdidos afecta negativamente al tráfico generado por la herramienta *Skype*.

En la Figura 7.6 se refleja un experimento en el que la tasa de perdidas de paquetes se va aumentando, para un *throughput* de 50 kbps. Se puede observar en este caso un comportamiento parecido al de la Figura 7.5, a medida que aumenta la tasa de paquetes perdidos, el tráfico generado por *Skype* se va reduciendo. Sin embargo, en esta Figura 7.6, se ve como el *throughput* generado por *Skype* es bastante mas reducido en comparación al caso de 200 kbps.

En la Figura 7.7 se tiene una gráfica en la cual se exponen los valores de las métricas objetivas para el caso de tener una tasa de tan sólo 10 kbps. Es posible observar como, para los mismos valores de tasa de pérdida de paquetes, de 0 % a 33 %, el *throughput* generado por *Skype* desciende rápidamente, así como se tiene un *jitter* mucho mas variante. Esto es debido a tener una tasa mucho menor a los casos anteriores.

De esta forma, se puede establecer que, cuando se tiene un *throughput* de llamada menor, se puede tener un *throughput* generado por la aplicación mayor si la tasa de pérdidas es menor, hasta llegar al punto en el que la tasa de pérdidas aumenta lo suficiente, caso en el que el *throughput* generado por la aplicación disminuye considerablemente.

7.0.1. MOS

Para el estudio de la calidad de experiencia en *Skype* se ha realizado la misma aproximación. Se han captado los valores del *eModel* cada diez segundos para todas las simulaciones descritas anteriormente. Posteriormente se han procesado esos valores mediante *Excel*, calculando el valor promedio y la desviación típica, en un proceso análogo al realizado para el estudio de los parámetros objetivos.

Finalmente se crean las dos matrices para el estudio de la calidad de experiencia en *Skype*. Una primera matriz recoge la calidad de experiencia (QoE) variando el *throughput* y el retardo, y una segunda matriz que recoge la calidad de experiencia (QoE) variando de la misma manera el *throughput* y la tasa de paquetes perdidos.

Para todos los casos se expresa la calidad de experiencia de *Skype* mediante el *MOS*, una medida que nos da la representación la calidad general

de nuestro sistema.

Matrix QoE Throughput VS Retardo

Throughput Retardo	200 kbps	100 kbps	50 kbps	25 kbps	10 kbps
0 ms	4.2	4.2	1.3	1.3	1.3
10 ms	4.3	1.2	1	1	1
50 ms	4.3	1	1	1	1
100 ms	3.7	1	1	1	1
200 ms	2.2	1	1	1	1

Cuadro 7.5: Throughput VS Retardo. MOS (*Mean Opinion Score*)

Es posible observar como a medida que las condiciones de red empeoran, bien en términos de *throughput* de la llamada, bien en términos de retardo en la conexión, el *MOS* va disminuyendo progresivamente, hasta llegar a el umbral de ruptura, punto a partir del cual las condiciones de *throughput* y retardo son tan malas que la calidad de experiencia (*QoE*) de *Skype* disminuye drásticamente.

Exponemos ahora la tabla que refleja los datos de la desviación típica para cada simulación:

Throughput Retardo	200 kbps	100 kbps	50 kbps	25 kbps	10 kbps
0 ms	0.2	0.2	1.01	1.02	1.01
10 ms	0.04	0.8	0	0	0
50 ms	0.01	0	0	0	0
100 ms	0.3	0	0	0	0
200 ms	0.0	0	0	0	0

Cuadro 7.6: Throughput VS Retardo. Desviación típica del MOS

Donde es posible establecer, en base a los datos expuestos, que los valores promedio recogidos en la matriz de la calidad de experiencia de *Throughput* VS Retardo, nos da mucha información acerca de los datos recogidos, esto es así ya que la desviación típica es varios ordenes menor al promedio de los datos recogidos.

A continuación, en la Figura 7.8 se expone la calidad de experiencia (*QoE*) expresada a partir del MOS (*Mean Opinion Score*) de un experimento para el que se tiene una tasa de 200 kbps y una tasa de pérdidas que va del 0 % al 33 % de paquetes perdidos. Se puede observar como se mantiene razonablemente bueno conforme la tasa de pérdida de paquetes va aumentando, debido a tener una tasa de 200 kbps.

A medida en que la tasa de paquetes perdidos va aumentando la calidad de experiencia disminuye para la misma tasa de 200 kbps, pasando de tener una calidad de experiencia (*QoE* excelente, a una regular.

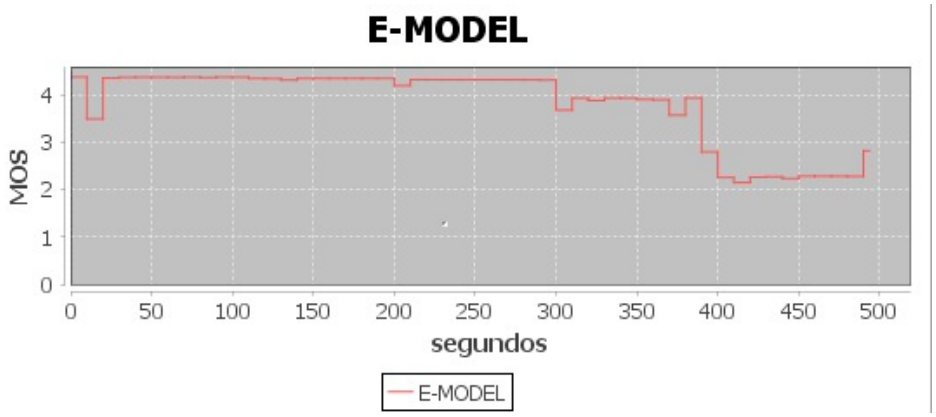


Figura 7.8: Throughput VS Retardo. MOS de la aplicación para 200 kbps

A continuación, en la Figura 7.9, se puede observar claramente como el *MOS* se va degradando a medida que aumenta el retardo, o disminuye el *throughput* de la llamada, manteniéndose en todo momento excelente bueno o aceptable para el caso de 200 *kbps*, y excelente para el caso de un *throughput* de 100 *kbps* y 0 ms de retardo, sin embargo a partir de ese punto es cuando la calidad de experiencia (*QoE*) de *Skype* experimenta una degradación muy significativa, pasando de un *MOS* de 4.25 (Muy Bueno) a un *MOS* de 1.25 (Pobre).

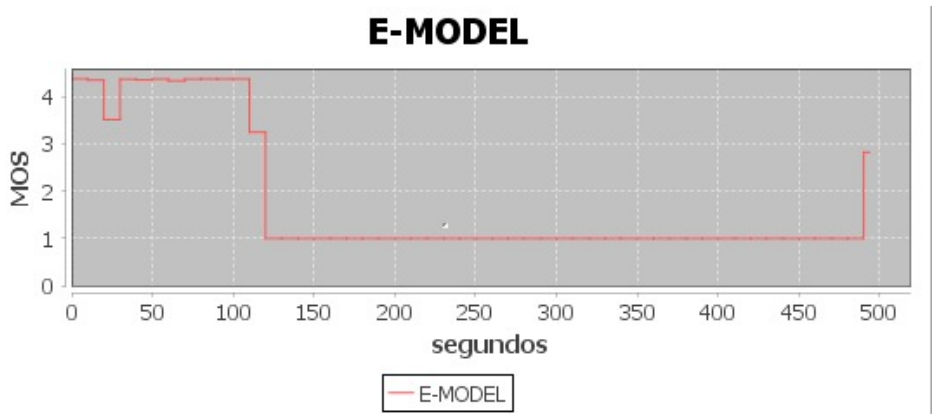


Figura 7.9: Throughput VS Retardo. MOS de la aplicación para 100 kbps

Matriz QoE Throughput VS Pérdida de paquetes

Throughput Pérdida de paquetes	200 kbps	100 kbps	50 kbps	25 kbps	10 kbps
0 %	4.2	4.3	1.3	1	1
5 %	3.7	3.2	1	1	1
16 %	3.3	2.1	1	1	1
25 %	3.2	1.1	1	1	1
33 %	3.1	1	1	1	1

Cuadro 7.7: Throughput VS Pérdidas. MOS (*Mean Opinion Score*)

De manera análoga al caso de *throughput* y retardo, se tiene para el caso de la tasa de pérdidas que la calidad de experiencia (*QoE*) de *Skype* disminuye progresivamente, hasta llegar al umbral de ruptura, punto a partir del cual baja drásticamente.

De igual forma al caso del estudio de la calidad de experiencia para *Throughput* VS Retardo, se hace necesario exponer la matriz de desviaciones típicas, con el objetivo de afirmar que los datos recogidos en la tabla anterior sean fiables. A continuación se expone dicha tabla:

Throughput Pérdidas	200 kbps	100 kbps	50 kbps	25 kbps	10 kbps
0 %	0.2	0.007	1.01	1.01	1.02
5 %	0.07	0.001	0	0	0
16 %	0.01	0	0	0	0
25 %	0.03	0	0	0	0
33 %	0.007	0	0	0	0

Cuadro 7.8: Throughput VS Pérdidas. Desviación típica del MOS

Observando la tabla de desviaciones típicas, es posible afirmar que, para cada simulación, se cumple que la desviación típica es mucho menor que el valor promedio, luego es posible establecer que los datos promedio recogidos en la matriz de calidad de experiencia son representativos del total de los datos captados en ese experimento, en relación a la calidad de experiencia (QoE).

A continuación se expone la Figura 7.10, el la cual se observa claramente una gráfica que plasma un experimento en el que se refleja la evolución de la calidad de experiencia (QoE) para un*throughput* de 200 kbps, a medida que se aumenta la tasa de pérdida de paquetes, desde un 0 % de pérdidas a un 33 %.

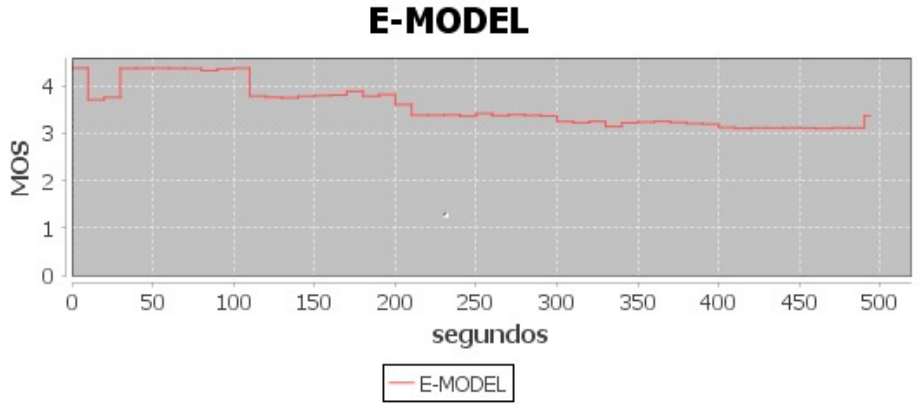


Figura 7.10: Throughput VS Pérdidas. MOS de la aplicación para 200 kbps

Se puede observar en esta gráfica como para el caso de un *throughput* de la llamada de 200 kbps el *MOS* se mantiene en todo momento excelente o bueno, para cualquier caso de pérdidas, pese a que evidentemente se degrada

conforme aumenta la tasa de paquetes perdidos, hasta llegar al máximo de degradación con un 33 % de paquetes perdidos, caso para el cual se tiene un *MOS* de 3.12, lo cual se sigue considerando buena calidad de experiencia (*QoE*).

7.0.2. Conclusiones de las pruebas

En este capítulo de Pruebas se ha procedido a exponer de una forma exhaustiva el tráfico generado por *Skype*, así como la calidad de experiencia (*QoE*). Esto se ha hecho mediante el modelo cuantitativo. Para dicho modelo cuantitativo se han obtenido las matrices que representan el tráfico generado por*Skype*, así como la calidad de experiencia (*QoE*) en *Skype*..

También se han detectado los umbrales de las métricas de red a partir de los cuales no sería viable la comunicación. Dichos umbrales dependen del *throughput* de red que se tenga (en kbps), del retardo o *delay* y de la tasa de perdida de paquetes.

Es así que, para el caso del Throughput VS Pérdidas, utilizando una tasa de 200 kbps la calidad de experiencia (*QoE*) se mantiene excelente o buena incluso hasta para una tasa de pérdidas del 33 %. Reduciendo la tasa a 100 kbps, el umbral para el cual la comunicación se degradaría demasiado, sería para una tasa de pérdidas del 25 %, tasa de pérdida de paquetes para la cual la comunicación sería inviable.

Para el caso de Throughput VS Retardo se tiene que, para una tasa o *throughput* de red de 200 kbps, la comunicación se mantiene en todo momento entre excelente o buena. Es reduciendo la tasa a 100 kbps, cuando se tendría un umbral, a partir del cual la calidad de experiencia (*QoE*) se reduce drásticamente, a partir de los 50 ms de retardo.

Capítulo 8

Conclusiones y vías futuras

8.1. Vista retrospectiva

Desde el primer momento en el que se decidió cual sería el Trabajo de Fin de Grado, y su propósito, hubo muchas ideas a modo de ampliación, algunas de las cuales se consiguieron implantar a tiempo, otras sin embargo han quedado para posibles proyectos futuros, debido en parte a la complejidad del propio proyecto que nos ocupa, en parte a una serie de errores en las herramientas que se han utilizado a lo largo del desarrollo del proyecto que fue necesario subsanar.

Sin embargo, el propósito principal de este proyecto, que era el análisis del tráfico generado y el estudio de la calidad de experiencia en la herramienta *Skype*, se ha cumplido satisfactoriamente.

Se han dado además una serie de pautas, un diseño a seguir en posibles vías futuras en cuanto al estudio de otras herramientas de *VoIP*, tales como *Discord* [24] o *TeamSpeak* [25]. La metodología seguida en la fase de pruebas del presente proyecto ha sido diseñada después de un exhaustivo proceso, que se extendió durante varios meses, de lectura de referencias y bibliografía, sobre diferentes metodologías utilizadas por diferentes autores sobre el análisis de la calidad de experiencia en Voz sobre IP.

Además, los errores surgidos durante el desarrollo del proyecto, han servido para depurar la herramienta de análisis del tráfico de *Skype*, siendo esta aplicable no solo a *Skype*, sino a cualquier otra herramienta de *VoIP* actual (véase el siguiente apartado de líneas futuras). Se ha implantado también un método a la herramienta que consigue obtener los valores instantáneos de trafico y calidad de experiencia para un experimento concreto, siendo necesario para el desarrollo del capítulo de Pruebas.

Otros cambios en herramientas utilizadas para el desarrollo de este proyecto han servido para mejorarlo, así como para hacerlo extensible con una mayor facilidad a otras aplicaciones *VoIP*. Todo esto queda debidamente detallado en el capítulo de Implementación.

Así pues, con este proyecto se ha conseguido no solo cumplir el propósito principal, realizar un estudio detallado de la calidad de experiencia (*QoE*) y trafico de la herramienta *Skype*, sino que gracias en parte a los errores surgidos por el camino, se han conseguido abrir o facilitar una serie de vías futuras, las cuales se procede a relatar a continuación.

8.2. Vías futuras

Se incluyen en este apartado posibles vías futuras o ampliaciones del presente proyecto:

- Mejora de la herramienta utilizada, de forma que se pueda utilizar como audio de la llamada, voz generada en ese preciso instante. En este proyecto se han utilizado diversos ficheros *WAW* de audio para la realización de las experimentos, todos ellos previamente almacenados. Sería interesante incluir la opción de generar audio llamarte con voz real, como vía futura.

- Análisis del tráfico de otras populares herramientas de Voz sobre IP. El auge de las tecnologías *VoIP* alrededor del mundo ha propiciado que existan numerosas herramientas que permiten a dos o mas usuarios realizar llamadas o videoconferencias *VoIP*, algunas de ellas incluso especializadas en determinados segmentos (videojuegos online, aplicaciones *VoIP* exclusivamente para smartphones etc). Se propone como vías futuras investigar el trafico *VoIP* en aplicaciones como *Discord* (para videojuegos online) o *Slack* (más orientada al mundo laboral, con una interfaz de usuario similar a la de *Discord*).

- Análisis del tráfico en otro tipo de redes, las posibilidades que los entornos emulados nos ofrecen son enormes. Esto se puede aprovechar para realizar todo tipo de estudios sobre tráfico o experiencia en otro tipo de redes o servicios. Podrían ser muy interesantes estudios de este tipo en plataformas de juego online por streaming (*PlayStation Now*) o sobre servicios de streaming (*HBO* o *Netflix*).

8.3. Conclusiones finales

Llegados aquí, cabe destacar que, pese a que el presente Trabajo de Fin de Grado ha supuesto un gran reto, con numerosos errores que han surgido

por el camino, y que han supuesto alguna que otra frustración, me siento muy satisfecho de todo el trabajo invertido a lo largo de estos meses en este proyecto. He podido aplicar lo aprendido a lo largo de la carrera, especialmente en la especialidad de Telemática, aplicando conceptos de redes, de multimedia o de programación Java. He seguido una metodología ágil de desarrollo y he intentado, finalmente, hacer una memoria lo más clara, amena y fácil de leer, pero que a la vez que documentase de la mejor manera posible el trabajo realizado en estos meses.

Cabe destacar también la gran cantidad de conocimientos que se han adquirido, no solo en cuanto a Voz sobre IP, sino en cuanto a redes multimedia, en las fases de investigación y análisis y en la de implementación.

Para resumir, este ha sido un proyecto con el que, pese a las eventuales dificultades que han ido surgiendo, me he sentido muy satisfecho al finalizarlo, así como de los conocimientos cimentados durante el mismo.

Bibliografía

[1] Aplicaciones VoIP más usadas
https://voip.review/2018/01/28/top-ten-voip-apps/

[2] Página oficial del códec SILK
https://web.archive.org/web/20130724103746/http://dev.skype.com/silk

[3] Historia de la Voz sobre IP
https://www.cosmocom.gr/wp-content/uploads/2013/05/Hallock_J_VoIP_Past.pdf

[4] Beneficios de la Voz sobre IP
https://www.centurylink.com/asset/business/enterprise/white-paper/business-benefits-voip-whitepaper-WP160048.pdf

[5] Diferentes tipos de tecnología VoIP
http://www.netlab.tkk.fi/~tsmura/publications/Smura_CICT04.pdf

[6] Conceptos básicos de las máquinas virtuales
https://www.csd.uoc.gr/~hy428/reading/smith_nair_2005.pdf

[7] Página oficial de Oracle VM VirtualBox
https://www.virtualbox.org

[8] International Telecommunications Union: P.800.1
https://www.itu.int/rec/T-REC-P.800.1

[9] ITU-T Recommendation G.107: The E-model, a computational model for use in transmission planning, 2008
https://www.itu.int/rec/T-REC-G.107

[10] Página oficial de PRTG Network Monitor
https://www.es.paessler.com/prtg

[11] Antonio Sánchez Navarro, Proyecto Fin de Grado: *VoIP Tester*
http://dtstc.ugr.es/it/pfc/proyectos_realizados/downloads/Memoria2010_AntonioSanchez.pdf

[12] Página oficial del framework GStreamer
https://gstreamer.freedesktop.org

[13] PDF explicativo de la herramienta de monitorización ThousandEyes
https://marketo-web.thousandeyes.com/rs/thousandeyes/images/ThousandEyes_Data_Sheet_VoIP_Monitoring.pdf

[14] Página oficial de Solarwinds
https://www.solarwinds.com/topics/voip-monitor

[15] Resumen y acceso a la prueba gratuita de WhatsUp Gold VoIP Monitor
https://www.ipswitch.com/recursos/hojas-de-datos/voip-monitor

[16] Página oficial de VoIPMonitor
https://www.voipmonitor.org/

[17] Guía de configuración de la tecnología Cisco IP SLA
https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipsla/configuration/xe-16/sla-xe-16-book/sla-overview.html

[18] César Senés, Proyecto Final de Grado: *Análisis del tráfico Skype* https://wpd.ugr.es/~jorgenavarro/thesis/2017_TFG_CesarSenesRomo.pdf

[19] Wiki oficial de VideoLAN
https://wiki.videolan.org/VLC_command-line_help

[20] Página oficial de SoX (Sound eXchange)
http://sox.sourceforge.net/

[21] Página oficial de Pamela, para la grabación de conversaciones de Skype
http://www.pamela.biz/en/

[22] Página oficial de la tarjeta virtual de audio Stereo Mix Plus
http://stereomixplus.com/

[23] Página oficial de Clisk, interfaz de línea de comandos para Skype
http://www.dlee.org/skype/clisk/

[24] Discord, VoIP para videojuegos online
https://discordapp.com

[25] Página oficial del software de comunicación TeamSpeak https://www.teamspeak.com/en/