Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Laboratorio de Redes Móviles, Inalámbricas y Distribuidas (ICARO)

Desarrollo de una Aplicación Móvil para Videollamadas Aplicada al Soporte de Consultas Médicas

Trabajo Especial de Grado
presentado ante la Ilustre
Universidad Central de Venezuela
por los Bachilleres:

Francisco Alonso
C.I.: 18.994.015
E-mail: francisco.a.alonso.s@gmail.com

Tony Briceño
C.I.: 18.841.614
E-mail: tonybp18@gmail.com

para optar al título de Licenciado en Computación

Tutora: Ana Morales
Cotutora: María Villapol

Caracas, Mayo de 2015

Universidad Central de Venezuela

Facultad de Ciencias
Escuela de Computación
Laboratorio de Redes Móviles, Inalámbricas y Distribuidas (ICARO)

ACTA DEL VEREDICTO

Quienes suscriben, Miembros del Jurado designado por el Consejo de la Escuela de Computación para examinar el Trabajo Especial de Grado, presentado por los Bachilleres Francisco Alonso C.I.: 18.994.015 y Tony Briceño C.I.: 18.841.614, con el título **"Desarrollo de una Aplicación Móvil para Videollamadas Aplicada al Soporte de Consultas Médicas"**, a los fines de cumplir con el requisito legal para optar al título de Licenciado en Computación, dejan constancia de lo siguiente:

Leído el trabajo por cada uno de los Miembros del Jurado, se fijó el día 14 de Mayo de 2015, a las 11:00 am, para que sus autores lo defendieran en forma pública, en Aula Internet II, Planta Alta, Galpón 10, lo cual estos realizaron mediante una exposición oral de su contenido, y luego respondieron satisfactoriamente a las preguntas que les fueron formuladas por el Jurado, todo ello conforme a lo dispuesto en la Ley de Universidades y demás normativas vigentes de la Universidad Central de Venezuela. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobarlo.

En fe de lo cual se levanta la presente acta, en Caracas el 14 de Mayo de 2015, dejándose también constancia de que actuó como Coordinador del Jurado el Profesor Tutor Ana Morales

Ana Morales
(Tutor)

Prof. Walter Hernández
(Jurado Principal)

Prof. Nancy Urbina
(Jurado Principal)

Agradecimientos

En esta página se realiza una mención en agradecimiento a las personas que colaboraron de alguna manera a la realización de este trabajo de investigación y desarrollo.

En este sentido se agradece a la Profa. María Elena Villapol por su labor de tutor y profesora durante la investigación, al Prof. David Pérez por su importante asesoría en cuanto al desarrollo de la videollamada y pruebas de rendimiento de la aplicación y a la Profa. Ana Morales por su labor de tutor en los últimos meses de trabajo, los cuales fueron muy importantes y decisivos para la culminación exitosa de este trabajo. Igualmente, se agradece al equipo de SOS Telemedicina por su colaboración y disposición durante todo el proceso.

También merecen una mención importante mis padres José Alonso y Carmen Sequera, y mi novia Andrea Torres, por su tiempo, apoyo y colaboración durante todo el tiempo y esfuerzo invertido en este trabajo.

-Francisco Alonso

Quiero agradecer a los profesores María Villapol, David Pérez y Ana Morales por el gran apoyo que nos brindaron en este trabajo de investigación y por otórganos el uso de los espacios del Laboratorio de Redes Móviles, Inalámbricas y Distribuidas (ICARO). De igual forma muestro mis agradecimientos a las personas que pertenecen al Centro de Informática Médica (CIM) de la Universidad Central de Venezuela, por su apoyo y colaboración. También quiero agradecer a mi compañero de trabajo Francisco Alonso por el gran esfuerzo y constancia a lo largo del desarrollo del proyecto estando en las buenas y malas.

El Programa Samuel Robinson (PSR) perteneciente a la secretaría de la Universidad Central de Venezuela me brindó la oportunidad de aprender y tener una mejor base en el estudio de la carrera, el cual fue de mucha ayuda. Por esto agradezco a todas esas personas que logran que el PSR siga en píe.

Por último pero no menos importante quiero dar un profundo agradecimiento a mis padres Antonio Briceño y Angelina Perea por su constante esfuerzo para que mi única preocupación fueran los estudios, y también agradezco a mi novia Ana Hernández por su apoyo incondicional durante toda la carrera.

-Tony Briceño

Resumen

Título:
Desarrollo de una Aplicación Móvil para Videollamadas Aplicada al Soporte de Consultas Médicas

Autor(es):
Francisco Alonso
Tony Briceño

Tutor:
Ana Morales

Cotutor*:*
María Villapol

El programa “SOS Telemedicina para Venezuela” del Centro de Análisis de Imágenes Biomédicas Computarizadas (CAIBCO) dependiente del Instituto de Medicina Tropical de la Universidad Central de Venezuela fue diseñado para dar soporte médico especializado a profesionales y estudiantes de la salud que prestan atención médica en localidades remotas. Este programa cuenta con soluciones propias de la telemedicina que permiten brindar este soporte de manera asíncrona, como el Sistema de Historia Médica Electrónica y el Sistema de Segunda Opinión Médica y Referencias Médicas, sin embargo no cuenta aún con una solución para prestar atención en tiempo real. El enfoque de este trabajo estuvo basado en la expansión del Sistema de Segunda Opinión Médica y Referencias Médicas con una aplicación móvil de videollamadas sobre Android, proporcionando así la capacidad de dar respuesta en tiempo real a médicos en localidades remotas. Esto se logró mediante la combinación de varias tecnologías, en el lado del servidor se instaló Asterisk y FreePBX, ésta es una solución que incluye una central telefónica IP-PBX, la cual se encarga del establecimiento de las videollamadas entre dos usuarios y la posibilidad de poder grabar el audio de las mismas, también se desarrolló un servicio web (API REST) para realizar operaciones a la base de datos del sistema; mientras que en el lado del cliente se hizo uso del *framework* Doubango para establecer la videollamada desde la aplicación desarrollada. Este proyecto se desarrolló siguiendo la metodología de desarrollo ágil SCRUM.

Palabras Claves: SOS telemedicina, aplicación Android, REST API, Asterisk, FreePBX.

Tabla de Contenido

Índice de Figuras

Índice de Tablas

1 Introducción

Llevar la atención médica a todos ha sido una tarea compleja que requiere de una gran cantidad de recurso humano, tecnológico y de infraestructura. Es por esto que el uso de las tecnologías de telecomunicaciones ha formado parte importante de todos estos esfuerzos en llevar atención médica a las personas que las necesiten. Especialmente en zonas más alejadas o de difícil acceso, donde no existe una infraestructura completa y no están presentes las ventajas en cuanto a medicina que ofrece una ciudad, es donde figura como una inmensa ventaja el uso de las telecomunicaciones.

Es por esta razón, que existen organizaciones cuyo fin es llevar a cabo las labores necesarias para proporcionar atención médica a zonas alejadas donde establecer una infraestructura compleja y mantener un gran personal médico conlleva muchos costos en recursos económicos y humanos. Estas organizaciones y programas se apoyan en el uso de las tecnologías de la telecomunicación para lograr su objetivo. SOS Telemedicina es uno de estos programas que funciona en Venezuela, elaborando proyectos cuyo objetivo es llevar atención médica especializada a zonas rurales y alejadas en varios estados del país.

Con la facilidad para la comunicación que proveen los teléfonos inteligentes y su gran popularidad en los grandes centros poblados más grandes, se hace fácil aprovechar este recurso para incrementar el alcance de los servicios médicos. El uso de estos teléfonos permite la comunicación en tiempo real, además de la transferencia y procesamiento de información desde prácticamente cualquier lugar donde exista acceso a Internet. Considerando estas facilidades el equipo de SOS Telemedicina propuso el diseño, desarrollo e implementación de un sistema que se integre con los proyectos ya existentes de referencias médicas y segunda opinión médica. Este nuevo proyecto tiene como base permitir una consulta médica como segunda opinión, a distancia y en tiempo real mediante una videollamada.

Este sistema consta de una aplicación en Android que funciona como cliente, desde esta se establecen las videollamadas y se accede a la base de datos de casos médicos y dos servidores, uno que permite establecer las videollamadas y otro que se encarga de manejar el acceso a la base de datos de casos de SOS Telemedicina. La implementación del sistema tiene como fin expandir el alcance del sistema actual de Referencias Médicas y Segunda Opinión Médica, proporcionando una herramienta que permita una respuesta en tiempo real para solventar distintas patologías que requieran consultas a especialistas, aprovechando las ventajas que proporciona el uso de un dispositivo móvil.

1.1 Definición del problema

El proyecto “SOS Telemedicina para Venezuela” del Centro de Análisis de Imágenes Biomédicas Computarizadas (CAIBCO) dependiente del Instituto de Medicina Tropical de la

Universidad Central de Venezuela está diseñado para proveer soporte a profesionales y estudiantes de la salud en instalaciones de atención primaria en zonas remotas y poco servidas donde la medicina móvil, la telemedicina y aplicaciones de software libre tienen particular aplicación.

La comunicación con otros profesionales en centros urbanos es básica para brindar el apoyo necesario. Los participantes del proyecto presentan necesidades relacionadas a la comunicación desde las áreas remotas donde se encuentran, especialmente en cuanto a la telemedicina ya que ésta demanda consideraciones propias que requieren de otros mecanismos de comunicación no tradicionales. Existen soluciones implementadas que proveen atención asíncrona a casos médicos mediante el **Sistema SOS Historia Médica Electrónica** que se implementó un sistema de referencias médicas electrónicas [1], el **Sistema de Segunda Opinión Médica SOS Telemedicina** consiste en una aplicación web desde la cual se gestiona la segunda opinión médica solicitada por médicos en zonas rurales o remotas a médicos especialistas [2]; existe también el **Sistema Móvil de Segunda Opinión Médica y de Referencias Médicas**, este permite el acceso móvil al Sistema de Referencias Médicas y al Sistema de Segunda Opinión Médica del Programa SOS Telemedicina [3][4]. Sin embargo, aún no hay una forma de proveer una segunda opinión médica de manera síncrona permitiendo a dos médicos mantener una consulta de manera síncrona, la implementación de un módulo que permita una conexión entre los médicos rurales y especialistas en tiempo real conlleva muchas ventajas sobre la comunicación y la transferencia de información. Por esta razón se plantea la siguiente pregunta ¿Puede una aplicación de video conferencia sobre dispositivos móviles proveer de una plataforma para realizar consultas médicas, transferencia de información y diagnósticos entre especialistas de la salud en localidades remotas?

1.2 Objetivo general

Desarrollar un sistema para realizar videollamadas a través de una aplicación para *smartphones* (teléfonos inteligentes) sobre el sistema operativo Android, que permita realizar consultas médicas y transferencia de información relacionada a dicha consulta.

1.3 Objetivos específicos

Los objetivos específicos del trabajo son:

- Capturar los requerimientos para la aplicación móvil de videollamada para segunda opinión médica.
- Diseñar en base a los requerimientos capturados los distintos diagramas que representen la aplicación, el flujo de comunicación y el flujo de interacción.
- Definir la arquitectura del sistema en base a los requerimientos.
- Implementar la aplicación de videollamada para Android.

- Instalar y configurar el ambiente de prueba para la aplicación desarrollada.

- Realizar pruebas a la aplicación en función a su usabilidad, funcionalidad y parámetros de rendimiento (por ejemplo, uso de CPU, consumo de RAM, retardo, *jitter*) en ambientes controlados y en ambientes reales.

1.4 Justificación

Las necesidades que presentan los participantes del proyecto SOS Telemedicina para Venezuela motivan al desarrollo de una aplicación sobre la plataforma que Android provee para cubrir las exigencias de un medio móvil de comunicación más completo en el ámbito de la telemedicina, especialmente una videollamada mediante la cual se pueda llevar a cabo una consulta médica y una transferencia de información relacionada, de esta manera se proporciona los médicos en localidades remotas la posibilidad de conectarse en tiempo real con un especialista. Adicionalmente es necesario que esta solución se integre al sistema existente y quede registrada toda la información relacionada a los casos y segundas opiniones para futuras consultas y casos de estudio.

En segundo lugar, las ventajas de un smartphone van desde el poder de cómputo el cual es superior al de un teléfono móvil convencional hasta las distintas formas de conectividad que dispone, esto lo convierte en una excelente plataforma cubriendo necesidades de recolección, almacenamiento, procesamiento y comunicación de datos. Según la investigación de *Strategy Analytics*, el número de smartphones en uso en todo el mundo superó la marca de mil millones de unidades en el tercer trimestre de 2012 [5]. Con esto se demuestra la gran aceptación que tienen estos dispositivos.

Por último, Android es un sistema operativo desarrollado para *smartphones*, es de software libre y posee una amplia documentación para el desarrollo de aplicaciones, Android posee la cuota de mercado del 46,9% [6], y está disponible para una gran variedad de dispositivos de distintas marcas y modelos. Esto provee una facilidad en la implementación y puesta en producción del sistema dado que los integrantes del proyecto tienen un mayor acceso a dispositivos Android.

1.5 Antecedentes

Este trabajo de investigación y desarrollo tiene como base y surgió a partir del proyecto **SOS Telemedicina para Venezuela**, este busca establecer una red de Telemedicina en Venezuela que contribuya de manera significativa a elevar la calidad de los servicios médicos asistenciales, y ofrecer alternativas para la formación permanente de personal de dichos servicios en zonas rurales, distantes o urbano-marginales, con el fin de mejorar las condiciones de salud de los habitantes y elevar los niveles de información y formación en salud del personal y la población asociada al servicio público de salud en Venezuela. Todo esto aplicando las tecnologías de la información y las telecomunicaciones [7].

A partir del proyecto SOS Telemedicina para Venezuela surge el proyecto **Sistema SOS Historia Médica Electrónica**. Este sistema consta de una serie de módulos que permiten la administración de la data de los médicos y referencias y realizar el triaje de los pacientes a los especialistas [1]. Esta data es almacenada en una base de datos que se convierte en el punto de integración de la Aplicación de Videollamada para Segunda Opinión Médica con el Sistema de historia Médica Electrónica.

El Sistema de Segunda Opinión Médica para SOS Telemedicina se encarga de permitir a médicos rurales obtener segundas opiniones médicas de especialistas a través de una aplicación web de manera asíncrona. Adicionalmente, ofrece acceso a la información de los casos, historias médicas e información de importancia sobre los pacientes, agilizando el proceso de atención de casos médicos [2].

Así mismo se desarrolló el **Sistema Móvil de Segunda Opinión Médica y de Referencias Médicas** para SOS Telemedicina, este proyecto tuvo como objetivo la elaboración e implementación de un sistema móvil integrado al Sistema de Referencias Médicas de SOS Telemedicina. Está basado en una aplicación web móvil y en una aplicación nativa para el sistema operativo Android y permite la gestión de Segunda Opinión Médica [3][4].

1.6 Distribución del documento

En los próximos capítulos se describen el marco teórico y todos los aspectos que cubre el proceso de desarrollo del sistema, estos están estructurados de la siguiente manera:

- **Introducción**: en este capítulo se presenta una descripción general del proyecto, se plantean la problemática que se intenta resolver y los objetivos generales y específicos que se definieron para lograrlo.

- **Aplicaciones de Telecomunicaciones en Salud**: en este capítulo se hace una reseña al uso de las tecnologías de telecomunicaciones en el área de la salud, se define el concepto de Telemedicina, su impacto, limitaciones y se describen algunos proyectos desarrollados en Venezuela.

- **Transferencia de Audio y Video**: en este capítulo se describe la arquitectura básica de una red y los componentes necesarios para la transferencia de audio y video (pila de protocolos SIP y RTP). Así como los estándares y técnicas de codificación de audio (AAC y MP3) y video (H.264).

- **Metodología y Herramientas de trabajo**: en este capítulo se describe la metodología de desarrollo y qué herramientas se utilizaron para llevar a cabo el proyecto.

- **Diseño del Sistema**: en este capítulo se explica el diseño de cada componente del sistema mediante los diagramas de casos de uso y diagrama de objetos.

- **Implementación del sistema**: en este capítulo se describen los detalles de la arquitectura del sistema, configuración de los servicios y desarrollo de la aplicación.

- **Pruebas y Resultados**: en este capítulo se presentan y describen con detalle las pruebas de rendimiento y funcionalidad realizadas sobre el sistema, además se describen y analizan los resultados que arrojaron dichas pruebas.

- **Conclusiones**: análisis de los resultados del presente trabajo y recomendaciones para trabajos futuros.

- **Instalación de Asterisk y FreePBX en Debian 6.0 (Squeeze)**: guía detallada del proceso de instalación de Asterisk y FreePBX sobre Debian 6.0 (Squeeze).

2 Aplicaciones de Telecomunicaciones en Salud

El acceso a los servicios de salud es un factor determinante del progreso y el bienestar de una nación, sin embargo, en muchos países en desarrollo y especialmente en las zonas rurales situadas a grandes distancias de los centros urbanos, la prestación de servicios de salud muestra serios inconvenientes debido a las dificultades de acceso, a la escasez del personal médico capacitado y de los recursos adecuados, entre otros factores; lo que hace que gran cantidad de personas en los lugares remotos carezcan de un adecuado acceso a los servicios de salud. Este problema aumenta por razones de poca demanda de dichos servicios que justifiquen la presencia permanente de tales especialistas, por carencia de equipos médicos con tecnología de punta, o por razones de orden público, económico y social.

En la actualidad los grandes avances en telecomunicación están contribuyendo a dar solución a estos inconvenientes, permitiendo que los médicos especialistas centralizados en las grandes ciudades, puedan proveer servicios asistenciales de salud a distancia, mediante la coordinación eficaz de los recursos disponibles. A esta integración de las ciencias médicas con el desarrollo de las telecomunicaciones y la informática para brindar atención o intercambio de información médica a distancia se le conoce como telemedicina.

2.1 Telemedicina

La Organización Mundial de la Salud (OMS), define la telemedicina como: “el suministro de servicios de atención sanitaria, en cuanto la distancia constituye un factor crítico, por profesionales que apelan a las tecnologías de la información y de la comunicación con objeto de intercambiar datos para hacer diagnósticos, preconizar tratamientos y prevenir enfermedades y heridas, así como para la formación permanente de los profesionales de atención de salud y en actividades de investigación y de evaluación, con el fin de mejorar la salud de las personas y de las comunidades en que viven” [8].

La telemedicina permite la disminución de los tiempos de atención, diagnósticos y tratamientos más oportunos, mejora la calidad del servicio, reduce los costos de transporte, hace posible la atención continuada, tratamientos más apropiados, disminución de riesgos profesionales, posibilita la interconsulta, amplía la cobertura de atención, campañas de prevención oportunas, entre otras muchas virtudes.

Dada la variedad de especialidades existentes en la medicina y las diversas maneras de adaptar o utilizar las aplicaciones de la tecnología para hacer telemedicina, se presentan distintas maneras de clasificarla: de acuerdo al tiempo, a las especialidades y al tipo de aplicación médica.

La clasificación en el tiempo hace referencia al momento en que se realiza la intervención médica a distancia y la comunicación entre el proveedor del servicio y quien lo solicita. En este sentido se tienen dos tipos de telemedicina [9]:

- La telemedicina asíncrona o de tiempo diferido; la cual se desarrolla cuando se envía o se solicita información clínica, y su asesoramiento ocurre tiempo después.

- La telemedicina sincrónica; que involucra la participación en tiempo real tanto de los pacientes como de los profesionales en salud en el envío de la información.

A continuación se muestra las clasificaciones por especialidad y por el tipo de aplicación médica [10]:

Tabla 2.1: Clasificación de la telemedicina.

Clasificación por especialidades médicas	Clasificación por tipo de servicio
• Telerradiología. • Telepatología. • Telecardiología. • Teleendoscopia. • Teledermatología. • Teleoftalmología. • Telecirugía. • Teleneurofisiología. • Telepediatría. • Teleobstetricia. • Teleoncología. • Telepsiquiatría. • Teleotorrinolaringología.	• Teleconsulta. • Telediagnóstico. • Telemetría. • Teleeducación. • Teleadministración. • Teleterapia. • Telefarmacia. • Telecuidado o Teleatención.

Podría decirse que los antecedentes de la telemedicina se remontan a la aparición del telégrafo y posteriormente comenzó a efectuarse por radio. En 1910, se utilizaron redes telefónicas análogas para la transmisión de electrocardiogramas (ECG) y electroencefalogramas. En los años 1920, varios países ofrecieron asesoramiento médico desde los hospitales a su flota de buques mercantes y servicio de consulta médica para los salvavidas, utilizando el código Morse. El hospital de la Universidad de Sahlgrens de Gotemburgo (Suecia) comenzó a prestar tales servicios en 1923 [9].

En los años 50 la telemedicina se difundió mediante circuitos cerrados de televisión en los congresos de medicina. En los 60 la Administración Nacional de Aeronáutica y del Espacio (*National Aeronautics and Space Administration*, NASA) desarrolló un sistema de asistencia médica que incluía el diagnóstico y el tratamiento de urgencias médicas durante las misiones espaciales. En1965 se realizó una demostración de operación de corazón abierto con la ayuda de un sistema de telemedicina entre el Methodist Hospital en Estados Unidos y el Hôpital Cantonal de Genève en Suiza. La transmisión se realizó por medio del primer satélite de interconexión continental creado por Comsat llamado ***EarlyBird*** [8].

Realmente casi ninguno de los programas de las décadas de los 60, 70 y 80 consiguió mantenerse por sí solos al terminar las subvenciones. No obstante la década de los 80 fue una década de gran actividad que dio lugar a muchos proyectos. En la década de los 90 se presentó un resurgimiento de la telemedicina que se ha denominado la “segunda era de la telemedicina”. Esta década supone la gran proliferación de experimentos de telemedicina a nivel mundial, muchos de ellos con un objetivo de continuidad y rentabilidad bien sustentado.

2.2 Experiencias en telemedicina en el mundo

En los países industrializados como Estados Unidos, Francia o Noruega han procedido a la masificación y a la integración de los servicios de telemedicina. Todas las regiones de Francia han desarrollado al menos una red de telemedicina y se trabaja por la integración de las historias clínicas con las imágenes para que estén disponibles en una red de alcance nacional.

En Estados Unidos la telemedicina comenzó a fines de 1950 con una serie de proyectos piloto en zonas rurales y urbanas que conectaban clínicas rurales, hogares de ancianos, prisiones y reservas indígenas con centros de atención sanitaria distantes. Más de 35 estados llevan a cabo actualmente proyectos de telemedicina y muchos de ellos desarrollan redes de telecomunicaciones estatales conectando los hospitales centrales con las zonas rurales. Las aplicaciones incluyen: atención sanitaria básica, medicina preventiva, salud pública, sistemas de información sanitaria, enseñanza médica permanente, servicios consultivos, entre otros [8].

En Noruega existen más de 300 aplicaciones de telemedicina en centros de salud basadas en videoconferencia con un ancho de banda de 384 Kbps, así mismo es ampliamente usada para teleeducación [8].

En España, el Ministerio de Sanidad y Consumo definió el Plan de telemedicina INSALUD, el cual marca las pautas para el desarrollo de la telemedicina. La mayoría de las experiencias giran en torno a la Telerradiología y se llevan a cabo importantes experiencias en Televigilancia y Teleatención [8].

En Chile las experiencias más importantes son las del Centro Diagnóstico de la Universidad Católica con el Hospital Soterró del Río, la de la Clínica Indisa con Isla de Pascua, la del Hospital Fuerza Aérea de Chile con la Base Aérea de la Antártida y las del sector público, coordinadas y patrocinadas por el Ministerio de Salud en distintos servicios asistenciales de regiones tanto del sur como del norte del país [8].

En Costa Rica el Ministerio de Salud desarrolló un proyecto con el objetivo de llevar consulta médica especializada a todas las regiones del país y extender de igual forma los servicios de educación continua al personal de salud. En mayo de 1997 se realizó la primera "Videoconferencia Internacional de Telemedicina". Con tal proyecto Costa Rica se colocó a la vanguardia en éste tipo de desarrollos, ya que viene a ser el primer país del mundo con un plan de telemedicina de cobertura nacional [11].

En Colombia se encuentran variadas experiencias, una de las más importantes ha sido la de Telerradiología entre el Seguro Social y la empresa VTG con más de 160.000 estudios anuales a nivel de la capital, además de experiencias en universidades y centros de investigación en desarrollo de software con manejo de historias clínicas para dar servicio a comunidades remotas en diversas especialidades médicas [8].

En Perú, el Proyecto EHAS – Alto Amazonas es el primer piloto cuyo objetivo principal es la provisión de servicios de acceso a información para el personal de salud en la provincia de Alto Amazonas, departamento de Loreto. El proyecto trabaja en zonas donde no ha llegado el servicio de telefonía básica, desarrollando redes de comunicación de bajo costo [8].

En Ecuador, La Universidad Técnica Particular de Loja promovió una campaña desde junio a diciembre de 2010, con el objetivo de ofrecer servicios de prevención en salud a la población del cantón Yacuambi con la finalidad de que las personas conozcan medidas básicas de prevención y que eso influya en el mejoramiento de su salud [12].

2.3 Proyectos de telemedicina en Venezuela

En Venezuela se encuentran varias iniciativas para el desarrollo de la telemedicina, específicamente en grupos de investigación de la Universidad de Carabobo (UC), de la Universidad de Los Andes (ULA) y de la Universidad Central de Venezuela (UCV). En tal sentido, el Grupo de Procesamiento de Imágenes (GPI) de la UC, trabajó en una propuesta para un proyecto piloto en telemedicina y actualmente el Grupo de Ingeniería Biomédica de la ULA desarrolla una propuesta para la implementación de sistemas de telemedicina en Mérida. La red de centros venezolanos de Bioingeniería y Telemedicina, formada por la Universidad

Simón Bolívar (USB), la Universidad de los Andes y la Universidad de Carabobo, participa en el Programa de Cooperación de Postgrado de Telemedicina entre Francia y Venezuela.

Por su parte en la Universidad Central de Venezuela el Centro de Análisis de Imágenes Biomédicas Computarizadas (CAIBCO) dependiente del Instituto de Medicina Tropical, desarrolla el programa "SOS Telemedicina para Venezuela", el cual contempla en su etapa inicial, el diseño, desarrollo e implementación de soluciones tecnológicas que acerquen oportunamente a las poblaciones de localidades apartadas de los estados Amazonas, Anzoátegui, Nueva Esparta, Miranda y Zulia a servicios especializados de salud, en el ámbito de la Teleconsulta y el Telediagnóstico, para realizar consultas a distancia a los especialistas médicos de la UCV y hospitales generales, con miras a obtener una segunda opinión y la orientación necesaria para el tratamiento adecuado de pacientes en zonas remotas de esos estados.

2.4 Ventajas de la telemedicina

La telemedicina es un servicio que permite ofrecer mejoras en los aspectos sociales, económicos y de desarrollo de los países. Posee el potencial de hacer la diferencia en la vida de cualquier persona. Unas de las tantas ventajas que ofrece este servicio se describen a continuación:

- Permite la comunicación a zonas rurales remotas, donde los pacientes y los profesionales en salud más cercanos pueden estar separados cientos de kilómetros, es decir, se ofrece el acceso a cuidados de salud a lugares donde no han sido disponibles antes, de modo que el aislamiento geográfico ya no sea un obstáculo para solucionar las necesidades básicas de tiempo y calidad del servicio médico.

- Ayuda a mantener a los profesionales de la salud en las áreas rurales otorgando entrenamiento y colaboración de otros profesionales de la salud, promoviendo además el acceso a la educación continuada y la actualización de éste personal.

- También permite un mejor aprovechamiento de la infraestructura disponible en el área de salud, tanto desde los puntos de vista de las prestaciones como de los costos, beneficiando a las zonas menos desarrolladas y modernizando los medios de comunicación en las instituciones.

Un buen ejemplo donde la telemedicina es de gran utilidad sería en aquellos casos de emergencias por desastres naturales o conmoción social, este servicio puede ser la diferencia entre la vida y la muerte de uno o más individuos. En particular, en estos casos donde el rápido tiempo de respuestas médicas y los cuidados especiales son necesarios, la disponibilidad de esta ayuda puede ser crítica.

Además permite optimizar la búsqueda, localización y recuperación de la información estadística incrementando el uso de servicios de información especializados para ponerlos a

disposición de profesionales, académicos, investigadores y de las instituciones encargadas del desarrollo de políticas de promoción de la salud.

2.5 Limitantes de la telemedicina

Así como los beneficios de la telemedicina son claros a través de experiencias y aplicaciones en diversos países del mundo, las limitantes e inconvenientes existen. La telemedicina requiere de una adecuada política de difusión de sus posibilidades y aplicaciones, para poder generalizarse de modo tal que sea accesible a la mayor cantidad posible de usuarios que realmente requieran de ella, ya que fallas en este proceso podrían generar serios inconvenientes.

Como muchas personas, algunos profesionales de la salud, pueden ser renuentes a utilizar una nueva tecnología si no la comprenden o la manejan fluidamente. La despersonalización producto de la disminución de la relación directa paciente-médico, causa alteración en la percepción y podría afectar la confianza entre el paciente y el profesional en salud en el desarrollo de Teleconsulta. Así mismo, ocasiona limitaciones ya que algunos procedimientos deben realizarse necesariamente en persona. Para obtener buenos resultados, los proveedores de servicios de telemedicina deben centrarse en las necesidades de los profesionales sanitarios y de los pacientes. En algunos sistemas y servicios de telemedicina, los usuarios deben disponer de un equipo compatible en ambos extremos del enlace de comunicaciones, lo cual reduce las posibilidades de funcionamiento y las ventajas del acceso a distintas fuentes de conocimientos que puede aportar la telemedicina.

El valor clínico de una imagen transmitida a través de una red de telecomunicaciones probablemente aumentará cuanto mejor sea su resolución. Si ésta fuese baja, la utilidad de la imagen es limitada y los médicos vacilarán en hacer un diagnóstico. Los datos médicos confidenciales sobre los pacientes deben estar protegidos contra el acceso no autorizado, de lo contrario ello podría generar conflictos de carácter médico legal. La financiación puede ser compleja y suele estar sujeta a limitantes de índole burocrática, dado que en las aplicaciones de telemedicina intervienen a menudo distintos participantes por ejemplo, operadores de telecomunicaciones, hospitales, instituciones gubernamentales, instituciones educativas o de investigación, entre otros.

2.6 Herramientas tecnológicas de uso en la telemedicina

Generalmente se piensa que las aplicaciones de telemedicina exigen una infraestructura de telecomunicaciones compleja y costosa, pero lo cierto es que algunas técnicas necesitan únicamente una infraestructura elemental para prestar el acceso a los servicios de salud en zonas remotas. Las aplicaciones de la telemedicina pueden clasificarse según el ancho de banda que se requiera para la transmisión. El suministro de los servicios de telemedicina siempre exige un análisis previo de las posibilidades, ventajas, herramientas disponibles, costos, resultados y sobre todo del nivel de desarrollo técnico en general.

3 Transferencia de Video y Audio

La transferencia de video y audio se denomina videoconferencia. *Video Development Initiative* (ViDe) [13] explica que una videoconferencia es en su forma más básica la transmisión de la imagen sincronizada (vídeo) y voz (audio) de ida y vuelta entre dos o más ubicaciones físicamente separadas, simulando un intercambio como si los participantes estuviesen en la conversación en físico. Para poder llevar a cabo este tipo de comunicación es necesario el uso de cámaras (para capturar y enviar video desde el punto final local), pantallas de vídeo (para visualizar el vídeo recibido desde terminales remotas), micrófonos (para capturar y enviar audio desde su punto final local), y los altavoces (para reproducir audio recibido desde terminales remotas).

Los usos más comunes de una videoconferencia van desde reuniones, aulas de clases y propósitos colaborativos. Sin embargo, estas aplicaciones no son las únicas que se le dan a una videoconferencia, también existen aplicaciones más específicas orientadas a ayudar y dar soporte a uno o más personas que se encuentren a grades distancias y requieran una comunicación visual y auditiva, entre estas aplicaciones específicas se tienen: la telemedicina, el teletrabajo, la teleeducación, aplicaciones judiciales, vigilancia y seguridad del campus, laboratorios remotos y respuesta a emergencias.

Las ventajas que trae la videoconferencia son múltiples, algunas de las más resaltantes son:

- El intercambio de ideas es más rápido y frecuente.
- Reunir personas situadas en diferentes lugares geográficos es mucho más sencillo.
- La toma de decisiones es más rápida.
- Aumenta la productividad y ventaja competitiva.
- Reducción de costos de tiempos y desplazamientos.
- Intercambio de imágenes y datos en tiempo real.

Varios sectores han observado estas ventajas y han optado por la videoconferencia como solución a sus problemáticas, entre ellos se encuentran: empresas, gobiernos, justicia, pequeñas oficinas y oficinas en casa, instituciones financieras, educación, sanidad, grandes recintos y auditorios, difusión televisiva, entre otros.

3.1 Arquitectura básica para una videoconferencia móvil

En la Figura 3.1 se muestra una arquitectura ejemplo de una red y los componentes necesarios en ésta para que una aplicación de videoconferencia pueda implementarse.

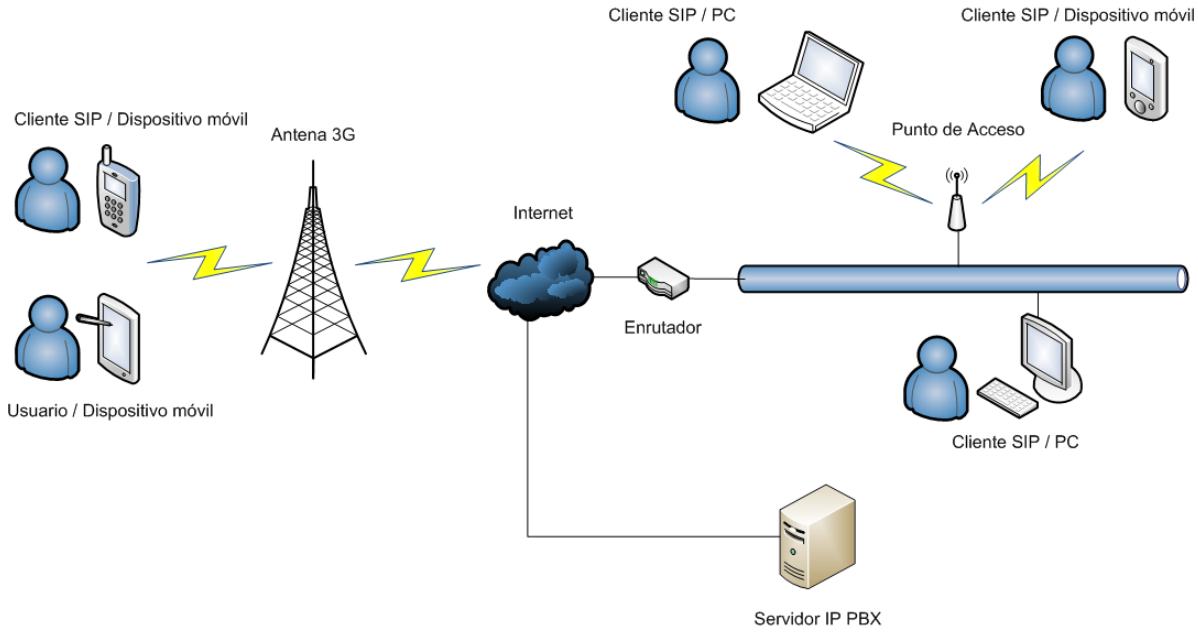


Figura 3.1: Arquitectura de red propuesta para una aplicación de videoconferencia.

Luego, en la Figura 3.2 se pueden observar los principales elementos de software que junto con los componentes de red permiten a la aplicación de videoconferencia establecer la comunicación entre los dispositivos finales, algunos de estos elementos de software como el Protocolo de Inicio de Sesiones (*Session Initiation Protocol*, SIP), el Protocolo de Transporte de Tiempo Real (*Real-time Transport Protocol*, RTP), Protocolo de Descripción de Sesiones (*Session Description Protocol*, SDP), Sistema de Nombres de Dominio (*Domain Name System*, DNS), Protocolo de Configuración Dinámica de Host (*Dynamic Host Configuration Protocol*, DHCP) y los codificadores de video y audio forman parte de la aplicación mientras que otros protocolos como el Protocolo de Internet (*Internet Protocol*, IP), el Protocolo de Datagrama de Usuario (*User Datagram Protocol*, UDP) y el Protocolo de Control de Transmisión (*Transmission Control Protocol*, TCP) además de la comunicación vía la red 3G son provistos por el sistema operativo. Los componentes relevantes se detallarán en las próximas secciones de este capítulo.

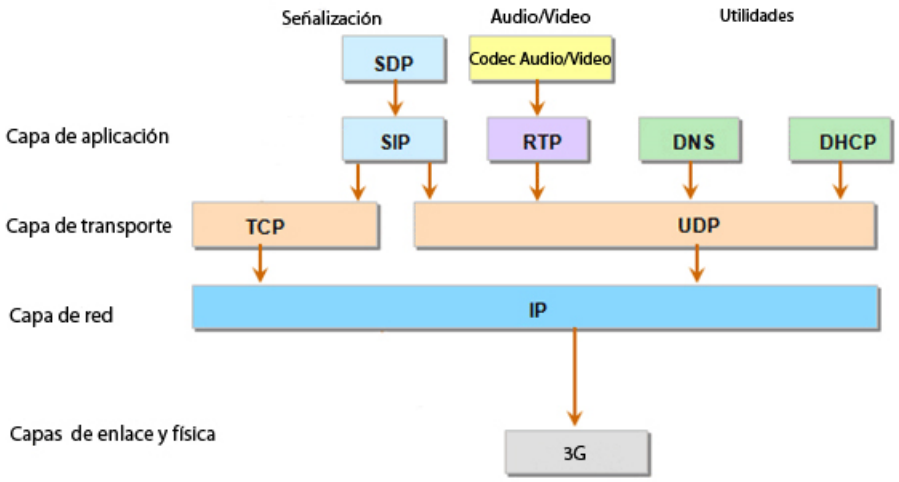


Figura 3.2: Elementos de software en la aplicación y el sistema operativo.

3.1.1 Redes 3G

3G es la tercera generación de tecnologías inalámbricas. Viene con mejoras con respecto a anteriores tecnologías inalámbricas, como transmisión de alta velocidad, acceso multimedia avanzada y roaming mundial. 3G se utiliza sobre todo con los teléfonos móviles como medio para conectarse a Internet u otras redes IP y de esta forma realizar llamadas de voz y video, descargar y cargar datos y para navegar por la red.

3G posee las siguientes mejoras sobre las redes 2.5G y anteriores en cuanto a mayor velocidad de datos, audio mejorado y *streaming* de vídeo, soporte de video-conferencia, navegación web y WAP a alta velocidad, soporte del Protocolo de Televisión en Internet (*Internet Protocol Television*, IPTV), entre otras.

A diferencia de Wi-Fi que se puede obtener de forma gratuita en algunos lugares públicos, es necesario estar suscrito a un proveedor de servicios para obtener conectividad a la red 3G. A menudo llaman a este tipo de servicio de un plan de datos o un plan de red. El dispositivo está conectado a la red 3G a través de su de su Módulo de Identificación de Suscriptor (*Subscriber Identity Module*, SIM) en el caso de un teléfono móvil o su tarjeta de datos 3G por ejemplo del tipo Bus Serial Universal (*Universal Serial Bus*, USB), el cual es proporcionado/vendido por el proveedor de servicios [14].

3.1.2 Servidores IP-PBX

Una IP-PBX o Ramal Privado de Conmutación IP (*IP Private Branch Exchange*, IP-PBX) es una pequeña es una pequeña central privada (sistema de conmutación telefónica dentro de una empresa) que conmuta las llamadas usando Voz sobre IP (*Voice over IP*, VoIP) entre los usuarios de las líneas locales mientras que permite a todos los usuarios compartir un número determinado de líneas telefónicas externas. Es capaz de ofrecer servicios de voz o de vídeo a

través de una red de datos y se integran con la Red Telefónica Pública Conmutada o (*Public Switched Telephone Network,* PSTN).

Una IP-PBX proporciona servicios similares a los de un Ramal Privado de Conmutación (*Private Branch Exchange*, PBX), puede conmutar llamadas entre un usuario de VoIP y un usuario de teléfono tradicional, o entre dos usuarios de telefonía tradicionales de la misma manera que un PBX convencional lo hace, pero a través de una Red de Área Local (*Local Area Network*, LAN) o una Red de Área Amplia (*Wide Area Network*, WAN) en lugar de una red de conmutación de circuitos.

Un PBX es un sistema cliente de telefonía local que administra los teléfonos de la empresa y actúa como la puerta de entrada a las redes de voz externo. Un conmutador/enrutador dirige los paquetes entrantes a la red de datos apropiada. Con un PBX convencional, redes separadas son necesarias para las comunicaciones de voz y datos. En lugar de dos redes separadas, cada una para voz y datos, sólo se necesita una red si se empaqueta la voz (VoIP) y se envía a través de la red de datos.

Un IP-PBX reemplaza una central tradicional. Se puede utilizar con un teléfono IP (con un chip incorporado que convierte la voz en paquetes IP y viceversa), un *softphone* (aplicación de software que también convierte a los paquetes de voz y viceversa) que se utiliza con un auricular y teléfonos estándar conectados a las computadoras.

Las ventajas de un IP-PBX en comparación con un PBX convencional son los siguientes:

- Maneja tanto voz como datos.
- Es más barato, ya que sólo requiere una red a instalar y mantener en lugar de dos.
- Reduce los gastos en equipos (ya que sólo se requieren productos basados en IP).
- Reduce los costos de larga distancia para las llamadas de oficina interprofesionales (mediante el uso de la red de datos).
- Fácil disposición (sólo tiene que conectarse desde cualquier lugar).
- Soporta servicios tales como mensajería unificada.
- Es más flexible.
- Es más escalable.
- Facilita la prestación de nuevos servicios, como los datos y la colaboración de video.
- Permite la configuración remota (a través de la red).

- Soporta actualizaciones de software modular y las nuevas tecnologías son fáciles de incorporar.

Algunos estándares utilizados en software IP-PBX son:

- **G.711**: un protocolo estándar internacional utilizado para codificar voz telefónica en un canal de 64 Kbps.

- **G.723.1**: un códec de audio para la voz que comprime el audio en tramas de 30 milisegundos. Existe dos tasas a la cual G.723.1 puede trabajar, en 5,3 Kbps y 6,3 Kbps.

- **G.726**: un protocolo estándar internacional de compresión de voz que cubre la transmisión de voz a tasas de 16, 24, 32, y 40 Kbps.

- **H.323**: recomendación para el servicio de señalización y comunicación audio-visual para la transmisión de paquetes IP que representen cualquier combinación de voz, vídeo y datos.

- **Respuesta de Voz Interactiva (*Interactive Voice Response*, IVR)**: un sistema telefónico de voz que interactúa con las personas que llaman con un menú de voz.

- **SIP**: un protocolo de señalización y comunicación similar a H.323 pero más simple.

3.1.3 Dispositivos finales

Los dispositivos finales o también conocidos como dispositivos terminales se le denomina a aquellos dispositivos con los cuales los usuarios interactúan para establecer y realizar la comunicación entre ambas partes.

Un dispositivo final puede ser un equipo como una computadora, una laptop, un *smartphone*, un equipo de video conferencia (video-teléfono VoIP), un *softphone* o un programa similar. Elementos como la cámara, una pantalla, micrófono y altavoces, son necesarios para establecer una videoconferencia, los *smartphones*, video-teléfonos VoIP, incluso laptops ya cuentan con estos requerimientos integrados, a diferencia de una computadora de escritorio por ejemplo.

3.2 Protocolo de Inicio de Sesiones (SIP)

SIP es un protocolo de control (señalización) a nivel de aplicación para conferencias en internet, telefonía, notificación de eventos y mensajería instantánea. SIP fue desarrollado por el grupo de trabajo IETF MMUSIC con la continuación de dicho trabajo por el grupo de trabajo IETF SIP desde septiembre de 1999.

SIP es un protocolo basado en texto, es muy parecido a HTTP, el protocolo usado para Web, o al Protocolo de Transferencia Simple de Correo Electrónico (*Simple Mail Transfer Protocol*, SMTP). Los mensajes consisten de encabezados y un cuerpo de mensaje. Los cuerpos de los mensajes SIP para las llamadas telefónicas son definidos por el protocolo SDP, éste es el encargado de establecer la sintaxis de dichos mensajes.

Las funciones básicas del protocolo incluyen:

- Determinar la ubicación de los usuarios, aportando movilidad.
- Establecer, modificar y terminar sesiones multipartitas entre usuarios.

3.2.1 Identificador Uniforme de Recurso SIP (SIP URI)

Las entidades SIP se identifican mediante el Identificador Uniforme de Recurso (*Uniform Resource Identifier,* URI) SIP. Un SIPURI tiene la forma de ***sip:usuario@dominio***, por ejemplo, sip:joe@company.com. Como se observa, un SIP URI consta de una parte con el nombre de usuario y otra con el nombre de dominio delimitado por el carácter arroba '@'. Las SIP URI's son fáciles de recordar ya que son similares a las direcciones de correo electrónico, con lo cual, por ejemplo, es posible utilizar el mismo URI para el correo electrónico y la comunicación SIP [15].

3.2.2 Arquitectura SIP

En la configuración más simple es posible usar sólo dos agentes de usuario que envían mensajes SIP directamente el uno al otro, sin embargo, en una red SIP típica contendrá más de un tipo de elementos SIP. Los elementos básicos SIP son: los Agentes de Usuario (*Agent User*, UA), *proxies*, los registrar y servidores de redirección. Brevemente estos se describen a continuación [15].

- **Agente de usuario SIP:** los dispositivos finales que usan SIP para encontrarse y negociar características de la sesión se denominan agentes de usuario. Los agentes de usuario por lo general, pero no necesariamente, residen en la computadora de un usuario en forma de una aplicación, este es actualmente el método más utilizado, pero los agentes de usuario pueden ser también teléfonos celulares, *gateways* PSTN, PDA's, sistemas automáticos de IVR, entre otros.
    - Los Agentes de Usuario a menudo se refieren como Agente de Usuario Servidor (*User Agent Server*, UAS) y el Agente de Usuario Cliente (*User Agent Client*, UAC). UAS y UAC son entidades lógicas únicas, cada agente de usuario contiene un UAC y UAS como se puede observar en la Figura 3.3. UAC es la parte del Agente de Usuario que envía solicitudes y recibe respuestas. UAS es la parte del Agente de Usuario que recibe peticiones y envía respuestas.

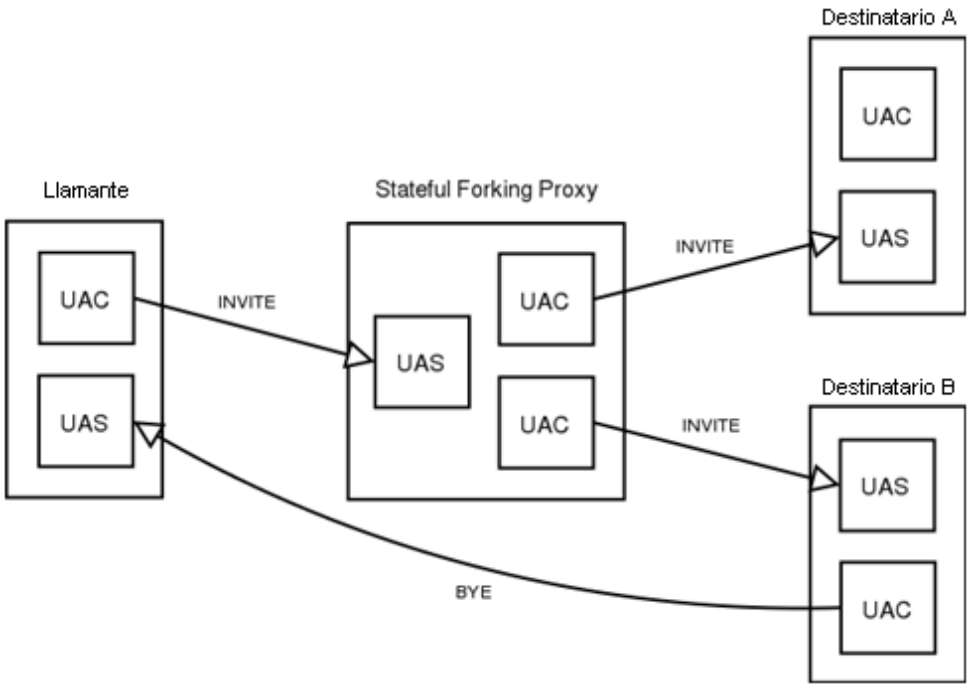


Figura 3.3: Ejemplos de UAC y UAS.

- La Figura 3.3 muestra tres Agentes de Usuario y un servidor proxy (*stateful forking proxy*). La parte del proxy que recibe el INVITE del llamante, de hecho, actúa como un UAS. Cuando se remite la solicitud el proxy crea dos UAC's, cada uno de ellos es responsable de una rama.

- En el ejemplo anterior el destinatario B atendió y después, cuando se desea colgar la llamada se envía un mensaje BYE. En este momento, el Agente de Usuario que fue previamente UAS se convierte en un UAC y viceversa.

- **Servidor proxy SIP:** los servidores proxy son entidades muy importantes en la infraestructura SIP. Realizan rutas de invitaciones de una sesión de acuerdo a la ubicación actual del invitado, la autenticación, su cuenta y muchas otras funciones importantes.

  - La tarea más importante de un servidor proxy es de enrutar las invitaciones de sesión "más cerca" al destinatario. La invitación de sesión usualmente atravesará un conjunto de proxies hasta que encuentra uno que conoce la ubicación real del destinatario. Dicho proxy enviará la invitación sesión directamente al destinatario y el destinatario entonces decide si aceptar o rechazar la invitación sesión.

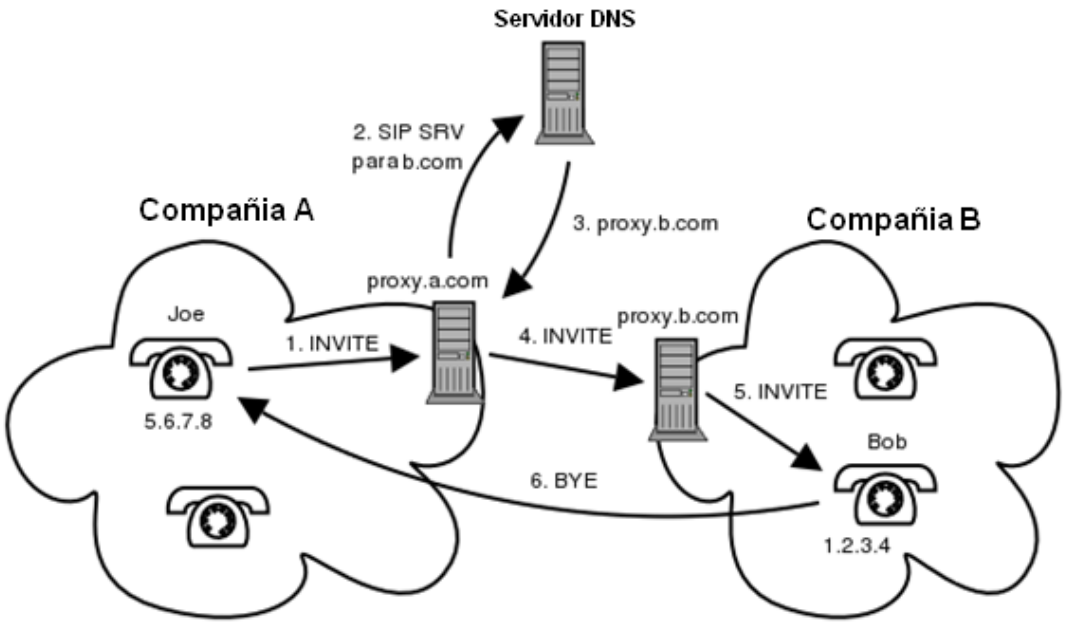


Figura 3.4: Invitación de una sesión SIP.

- En la Figura 3.4 se muestra cómo una invitación de sesión del empleado Joe que pertenece a la compañía A llegará a Bob que es un empleado en la compañía B.

- **Servidor de redirección SIP:** un servidor de redirección recibe peticiones y busca el destinatario de la solicitud en la base de datos de ubicación creada por un registrar. A continuación, crea una lista de ubicaciones actuales del usuario y la envía al emisor solicitud en una respuesta de clase 3xx.

    - El autor de la solicitud se utiliza para extraer la lista de destinos y envía otra solicitud directamente a ellos. La Figura 3.5 muestra una redirección típica.

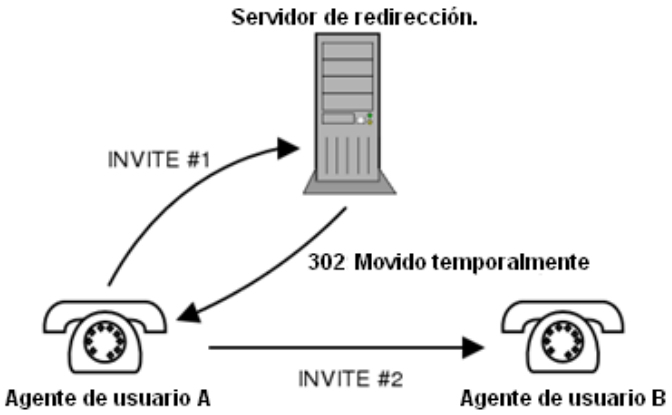


Figura 3.5: Redirección SIP.

- **Registrar SIP:** el registrador es una entidad SIP especial que recibe registros de usuarios, extrae información sobre su ubicación actual (dirección IP, puerto y nombre de usuario en este caso) y almacena dicha información en la base de datos de ubicación. El propósito de la base de datos de ubicación es mapear *sip:bob@b.com* a algo como *sip:bob@1.2.3.4:5060*. Cuando el proxy recibe una invitación para *sip:bob@b.com* va a buscar en la base de datos de ubicación. Encuentra *sip:bob@1.2.3.4:5060* y enviará la invitación allí. Un registrador es muy a menudo una entidad lógica única. Debido a su estrecha conexión con los registradores proxies, suelen ser coubicadas con servidores proxy.

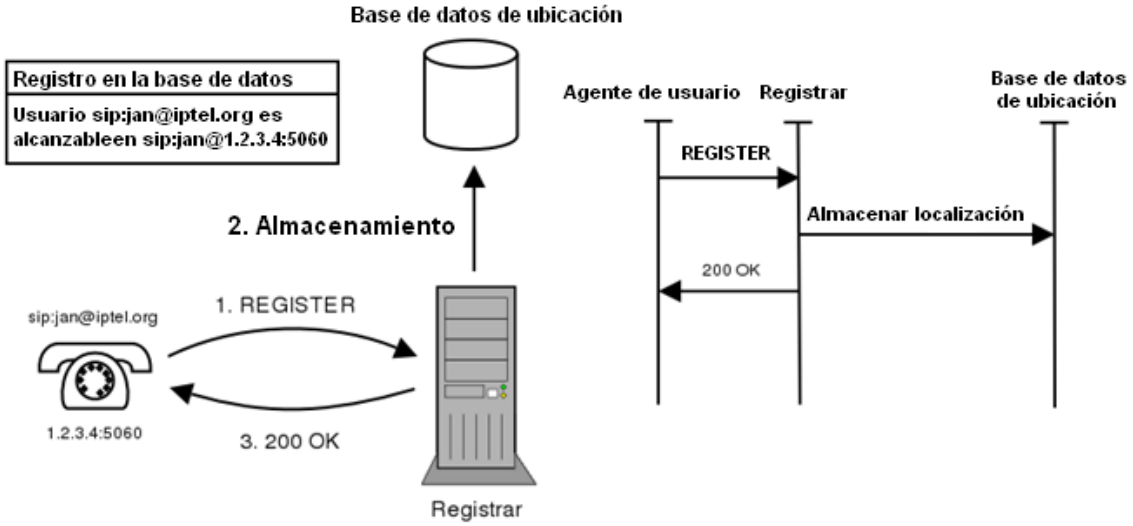


Figura 3.6: Descripción de un Registro SIP.

  - La Figura 3.6 muestra un típico registro SIP. Un mensaje REGISTER que contiene la Dirección de Registro *sip:jan@iptel.org* y la dirección de contacto *sip:jan@1.2.3.4:5060* donde 1.2.3.4 es la dirección IP del teléfono, se envía al Registrar. El registrar extrae esta información y la almacena en la base de datos de ubicación. Si todo ha ido bien, entonces el registrar envía una respuesta 200 OK en el teléfono y el proceso de registro ha finalizado.

3.2.3 Formato de los mensajes

SIP utiliza métodos para realizar solicitudes y sus respuestas correspondientes, pudiendo de esta forma establecer una sesión de llamada. La comunicación a través de SIP (a menudo llamado señalización) se compone de una serie de mensajes. Los mensajes pueden ser transportados de forma independiente por la red. Por lo general son transportados en un datagrama UDP separado cada uno. Cada mensaje consiste de la “primera línea”, encabezado del mensaje, y el cuerpo del mensaje. La primera línea identifica el tipo del mensaje. Hay dos tipos de mensajes, las solicitudes y las respuestas. Las solicitudes se suelen utilizar para iniciar una acción o informar a quien reciba la solicitud de algo. Las respuestas se utilizan para

confirmar que se recibió una solicitud y fue procesada, y contienen el estado del procesamiento.

Una típica petición SIP es como la que se muestra en la Figura 3.7, ésta se conforma de varios encabezados llamados: primera línea, cabecera, separador y mensaje, a continuación se describen cada uno de ellos.

El encabezado **Primera línea** de la Figura 3.7 indica que es un mensaje del tipo INVITE el cual se usa para establecer una sesión. El URI que se encuentra allí (*sip:7170@iptel.org*) se llama URI de solicitud y contiene el URI del siguiente salto del mensaje. En este caso, será el host iptel.org.

Una petición SIP puede contener uno o más campos *Via* en la **Cabecera** que se utilizan para registrar el camino de la solicitud. Luego, son usadas para enrutar las respuestas SIP exactamente por el mismo camino. En la Figura 3.7 el mensaje INVITE contiene sólo un campo *Via* en la cabecera ya que fue creado por el agente de usuario que envió la solicitud. Con el campo *Via* se puede decir que el agente de usuario se está ejecutando en el host 195.37.77.100 y por el puerto 5060.

Los campos *From* y *To* de la **Cabecera** identifican al iniciador (llamador) y al receptor (destinatario) de la llamada respectivamente. En el campo *From* del encabezado contiene un parámetro (*tag*) que sirve como un identificador de diálogo.

El campo *Call-ID* de la **Cabecera** es un identificador de diálogo y su propósito es identificar los mensajes que pertenecen a la misma llamada. Estos mensajes tienen el mismo identificador de llamada ID. *CSeq* se utiliza para mantener el orden de las solicitudes.

Dado que las solicitudes se pueden enviar sobre un transporte poco fiable donde los mensajes serán reordenados, un número de secuencia debe estar presente en los mensajes, de modo que el destinatario pueda identificar retransmisiones y solicitudes fuera de servicio.

El campo *contact* de la **Cabecera** contiene la dirección IP y el puerto en el que el remitente está en espera de nuevas solicitudes enviadas por el destinatario.

El **Mensaje** está delimitado de la **Cabecera** por una línea en blanco. Este encabezado es opcional y se utiliza entre otras cosas para transportar las descripciones de las sesiones que se quieren establecer, utilizando la sintaxis del protocolo SDP.

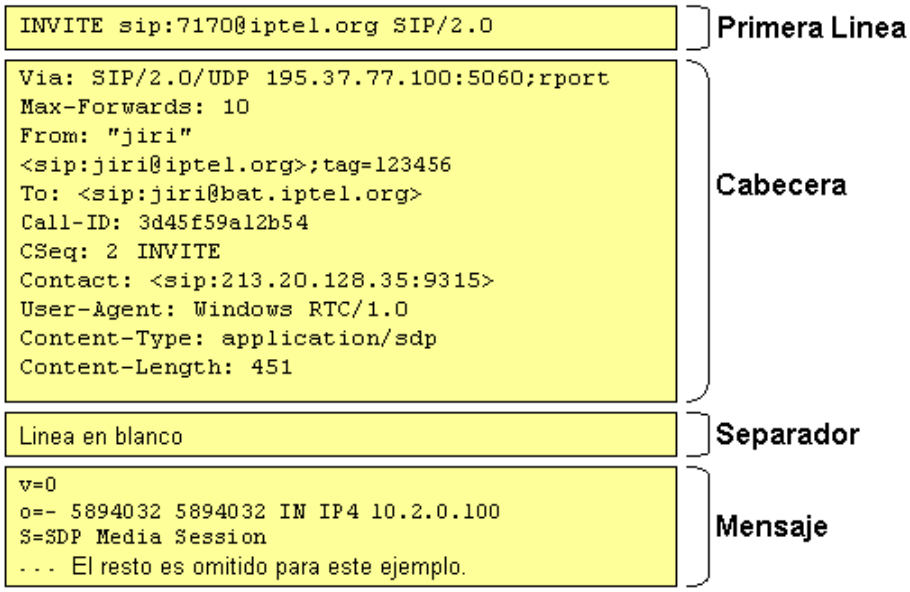


Figura 3.7: Ejemplo de un mensaje INVITE.

3.2.4 Mensajes de solicitud SIP

Existen seis tipos de mensajes de solicitud SIP básicos: **INVITE**, **ACK**, **BYE**, **CANCEL**, **REGISTER** y **OPTION**, los cuales son explicados brevemente a continuación.

- **Método INVITE:** indica un cliente que está siendo invitado a participar en una sesión de llamada.

- **Método de reconocimiento (*Acknowledgement*, ACK):** se utiliza para facilitar el intercambio fiable de los mensajes INVITE's, confirma una solicitud INVITE.

SIP implementa un protocolo de saludo de tres vías (*three-way handshake*) para establecer la comunicación:

- El emisor envía un mensaje INVITE.

- El destinatario de la llamada envía un 200 OK para aceptar la llamada.

- La persona que llama envía un mensaje ACK para indicar que el *three-way handshake* va a ser configurado y se hace la llamada.

Si el primer mensaje INVITE incluye una descripción de la llamada SDP, el mensaje 200OK incluye el SDP del destinatario.

- **Método BYE:** cuelga una sesión. El mensaje BYE se utiliza para culminar sesiones multimedia. La parte que desee culminar una sesión envía un pase directo a la otra parte.

- **Método CANCEL:** cancela una invitación. El mensaje CANCEL se utiliza para cancelar la sesión que aún no está completamente establecida. Se utiliza cuando el destinatario de la llamada no ha respondido con una respuesta definitiva todavía, pero la persona que llama desea abortar la llamada (por lo general cuando un destinatario de la llamada no responde por algún tiempo).

- **Método REGISTER:** registra una ubicación con un servidor de registro SIP. El objetivo de la solicitud REGISTER es dar a conocer al REGISTRAR de la ubicación actual del usuario. La información sobre la dirección IP actual y el puerto en el que se llegó a un usuario se realiza en los mensajes REGISTER. REGISTRAR extrae esta información y la pone en una base de datos de ubicación. La base de datos puede ser posteriormente utilizada por los servidores proxy SIP y así enrutar las llamadas para el usuario. Las inscripciones son limitadas en el tiempo y deben ser periódicamente renovadas.

- **Método OPTIONS:** permite que un agente de usuario consulte a otro agente de usuario o a un servidor proxy para conocer sus capacidades, es decir, esto le permite a un cliente obtener información acerca de los métodos admitidos, tipos de contenido, extensiones, códecs, entre otras cosas sin la necesidad de una llamada de la otra parte.

3.2.5 Código de los mensajes de respuesta

Los códigos de respuesta están relacionados con los códigos de respuesta de HTTP/1.1. No todos los códigos de respuesta HTTP/1.1 son apropiados, y sólo aquellos que sean apropiados se dan a continuación. Otros códigos de respuesta HTTP/1.1 no deben ser utilizados. Además, SIP define una nueva clase, 6xx [16].

- **1xx: provisional** - solicitud recibida, sin dejar de procesar la solicitud; las respuestas provisionales, también conocidas como respuestas informativas, indican que el servidor contactado está realizando una acción adicional y todavía no tengo una respuesta definitiva. Un servidor envía una respuesta 1xx si espera tener más de 200 ms para obtener una respuesta final. Hay que tener en cuenta que las respuestas 1xx no se transmiten de manera fiable. No causa que el cliente envía un ACK. Las respuestas provisionales (1xx) pueden contener cuerpos de mensajes, incluyendo descripciones de la sesión.

- **2xx: éxito** - la acción fue recibida con éxito, entendida y aceptada.

- **3xx: redirección** - nuevas medidas se deben tomar para completar la solicitud.

- **4xx: error del cliente** - la solicitud contiene una sintaxis incorrecta o no se puede cumplirse en este servidor.

- **5xx: error del servidor** - el servidor no pudo cumplir una petición aparentemente válida.

- **6xx: fallo global** - la solicitud no puede cumplirse en ningún servidor.

3.2.6 Flujo de establecimiento de una sesión SIP

Para el establecimiento y finalización de una sesión SIP se realizan una serie de intercambios de mensajes, en la Figura 3.8 se muestra a un Agente de Usuario 1 que intenta llamar al Agente de Usuario 2 y los mensajes involucrados.

Para iniciar la sesión el agente de usuario SIP UAC envía la petición con el método INVITE al servidor que tiene configurado (1). El agente de usuario destino si se encuentra desocupado comenzará a alertar al usuario destino y envía una respuesta hacia el usuario origen con un código de estado que indica esta situación (2). La respuesta sigue el camino inverso hacia el usuario origen. Cuando el usuario destino finalmente acepta la invitación, se genera una respuesta con un código de estado que indica que la petición fue aceptada (3). La recepción de la respuesta final es confirmada por el UAC origen mediante una petición con el método ACK (4), esta petición no genera respuestas y completa la transacción de establecimiento de la sesión. Las UA's establecen directamente canales RTP para el transporte de la voz o del video en forma de paquetes sin implicación del servidor proxy en este transporte.

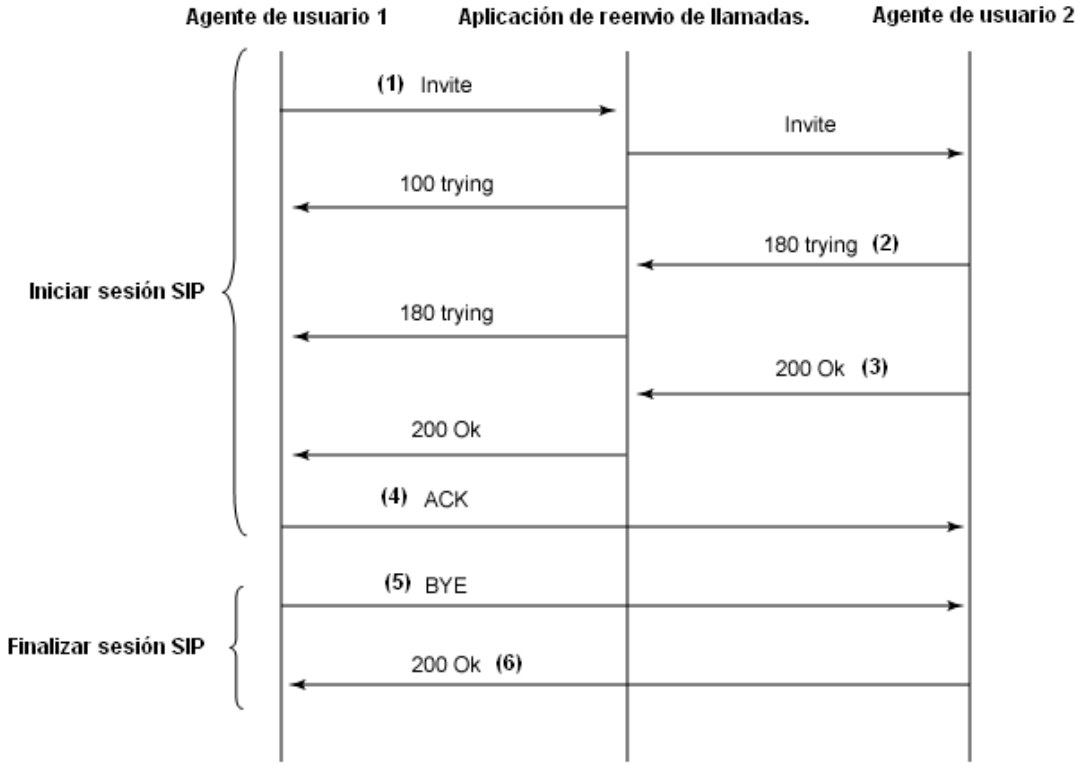


Figura 3.8: Flujo de establecimiento de una sesión SIP.

Al terminar la sesión, que lo puede hacer cualquiera de las partes, el agente de usuario de la parte que terminó la sesión, actuando como UAC, envía hacia la otra una petición con el método BYE, en este caso es el Agente de Usuario 1 (5). Cuando lo recibe el UAS genera la respuesta con el código de estado correspondiente (6).

3.3 Protocolo de Transporte de Tiempo Real (RTP)

El diseño de RTP se basó en dos esquemas que son el **entramado a nivel de aplicación** y el **principio *end-to-end*** con el fin de ofrecer un mecanismo robusto para la transmisión en tiempo real de data multimedia sobre un protocolo de transporte no confiable.

La base del **entramado a nivel de aplicación**, concepto introducido en 1990, es que sólo la aplicación tiene suficiente conocimiento de la data que maneja para tomar las decisiones acertadas en cuanto a cómo debe transportarse la data. Esto implica que el protocolo de transporte debe aceptar y entregar unidades de data significativas para la aplicación, conocidas como Unidades de Datos de la Aplicación (*Application Data Units*, ADU), además de exponer la información y detalles de su entrega de manera que la aplicación pueda responder adecuadamente frente a un error [17].

El objetivo del entramado a nivel de aplicación discrepa con el diseño de TCP que esconde las pérdidas producidas en la red IP, sin embargo tiene un buen funcionamiento con una capa de transporte UDP y aplicaciones multimedia de tiempo real. Una red basada en este principio no debe ser específica para un solo tipo de aplicación, debe exponer las limitaciones de las capas de transporte genéricas de modo que la aplicación obtenga flexibilidad para responder y alcanzar mejores resultados. El entramado a nivel de aplicación implica debilitar hasta cierto punto las capas definidas de manera estricta por el Modelo de Interconexión de Sistemas Abiertos (*Open Systems Interconnection*, OSI), esto en búsqueda de una solución pragmática que aparte de hacer el uso necesario de las capas también reconoce que deben exponerse detalles a las capas superiores.

Este esquema requiere que las aplicaciones sean inteligentes, conscientes de la red y puedan reaccionar ante los problemas.

El **principio *end-to-end*** se basa en dejar la responsabilidad de la data en manos de los entes finales, asegurando la confiabilidad de la comunicación incluso si los puntos intermedios no son confiables, RTP al igual que TCP siguen este principio. La consecuencia principal de la aplicación de este esquema es que la inteligencia tiende a agruparse en las capas superiores de la pila de protocolos por lo que el sistema que forma parte de la ruta de la red no tiene por qué ser robusto.

Estos dos esquemas mencionados son adecuados para la trasferencia de data en tiempo real sobre la Internet, en especial para aplicaciones multimedia que generalmente pueden soportar pérdidas pero están limitadas por lapsos de tiempo estrictos. Sin embargo, requieren un gran trabajo extra por parte del diseñador de la aplicación.

3.3.1 Elementos estándar de RTP

La mayoría de las implementaciones de RTP se presentan como aplicaciones o bibliotecas que funcionan sobre las interfaces UDP/IP que provee el sistema operativo. Como RTP no requiere UDP/IP este no es el único esquema posible, puede funcionar tanto sobre protocolos TCP/IP como en redes no-IP, por ejemplo, Modo de Transmisión Asíncrono (*Asynchronous Transfer Mode*, ATM). RTP provee un *framework* para el transporte de multimedia en tiempo real, este necesita ser complementado con perfiles y formatos asociados de carga útil para su correcto funcionamiento de acuerdo a diversos formatos de video y audio.

Existen dos componentes fundamentales en RTP que son el protocolo de transferencia de data y el protocolo de control asociado al primero. El protocolo de transferencia de data se encarga de definir un nivel adicional en la pila de protocolos empaquetando la carga útil multimedia incorporando varios elementos adicionales. Junto a éste funciona el Protocolo De Control de RTP (*Real Time Control Protocol*, RTCP) que provee un *feedback* de la calidad de la comunicación además de ser necesario para la sincronización e identificación de los participantes.

Se puede decir que RTP no fue completado explícitamente en dos aspectos, en primer lugar no especifica algoritmos para reproducción de multimedia, corrección de errores o control de congestión, dado que estos elementos pueden variar dependiendo de la aplicación no es conveniente limitar el protocolo a un solo comportamiento. En segundo lugar, algunos detalles de transporte de la data están abiertos a ser modificados mediante los perfiles y formatos de carga útil mencionados anteriormente, esto incluye características como la resolución de las marcas de tiempo, señalamiento de eventos importantes en la transferencia multimedia y uso del campo de tipo de carga útil, además el perfil puede definir las siguientes características:

- Mapeo entre el identificador de tipo de carga útil en el encabezado RTP y las especificaciones del formato de carga útil (éste describe cómo han de ser usados los distintos codificadores multimedia con RTP).

- El tamaño del campo del identificador del tipo de carga útil en el encabezado RTP y el número de bits usados para el señalamiento de eventos importantes en la transferencia multimedia.

- Adiciones al encabezado de transferencia de data RTP en casos donde este sea insuficiente para algún tipo de aplicación en particular.

- Intervalo de reporte del RTCP.

- Limitaciones en cuanto a que tipos de paquetes RTCP serán usados, en caso de que alguna información no sea útil para esa clase de aplicación.

- Mecanismos de seguridad adicionales como nuevos algoritmos de encriptación y autenticación.

- Mapeo de RTP y RTCP hacia capas de transporte inferiores.

En el *framework* RTP el **formato de carga útil** especifica cómo tipos particulares de data son transportados mediante RTP. Los formatos de carga útil son referenciados por los perfiles.

La relación entre un formato de carga útil y el perfil es principalmente un espacio de nombres, aunque el perfil también puede especificar algún comportamiento en el formato de carga útil. El espacio de nombres relaciona el identificador de tipo de carga útil con las especificaciones del formato de carga útil, de esta manera la aplicación puede relacionar la data a un códec específico. Un formato de carga útil puede especificar el uso de ciertos campos en el encabezado RTP, de igual manera puede definir un encabezado adicional en la carga útil. Luego la salida producida por un códec es traducida en una serie de paquetes de data RTP, algunas partes en el encabezado RTP, otras en el encabezado de la carga útil y el resto forma parte de la carga útil. La complejidad de este mapeo depende del códec y del grado de resistencia a errores que se quiera lograr.

Además dentro del protocolo RTP se definen **sesiones**, una sesión consiste en un grupo de participantes que se están comunicando haciendo uso de RTP. Una sesión sólo puede ser usada para transportar un sólo tipo de data multimedia, por lo que cada participante puede tener más de una sesión activa, por ejemplo una sesión para video y otra para audio. Una sesión está definida para cada participante con una dirección y un par de puertos por el que se enviará la data y otro par por el que se recibirá la data, los puertos para enviar y recibir pueden ser los mismos. Cada par de puertos consiste en dos puertos adyacentes, un puerto con número par para la trasferencia de los paquetes de data RTP y el siguiente puerto (con número impar) para la transferencia de los paquetes RTCP.

3.3.2 Protocolo de transferencia de data RTP

El protocolo RTP es el medio por el cual se transmite la data multimedia en tiempo real, se discutirán tanto el formato del paquete RTP como los campos que lo conforman, la extensión opcional del encabezado, el encabezado opcional de la carga útil y la carga útil como tal.

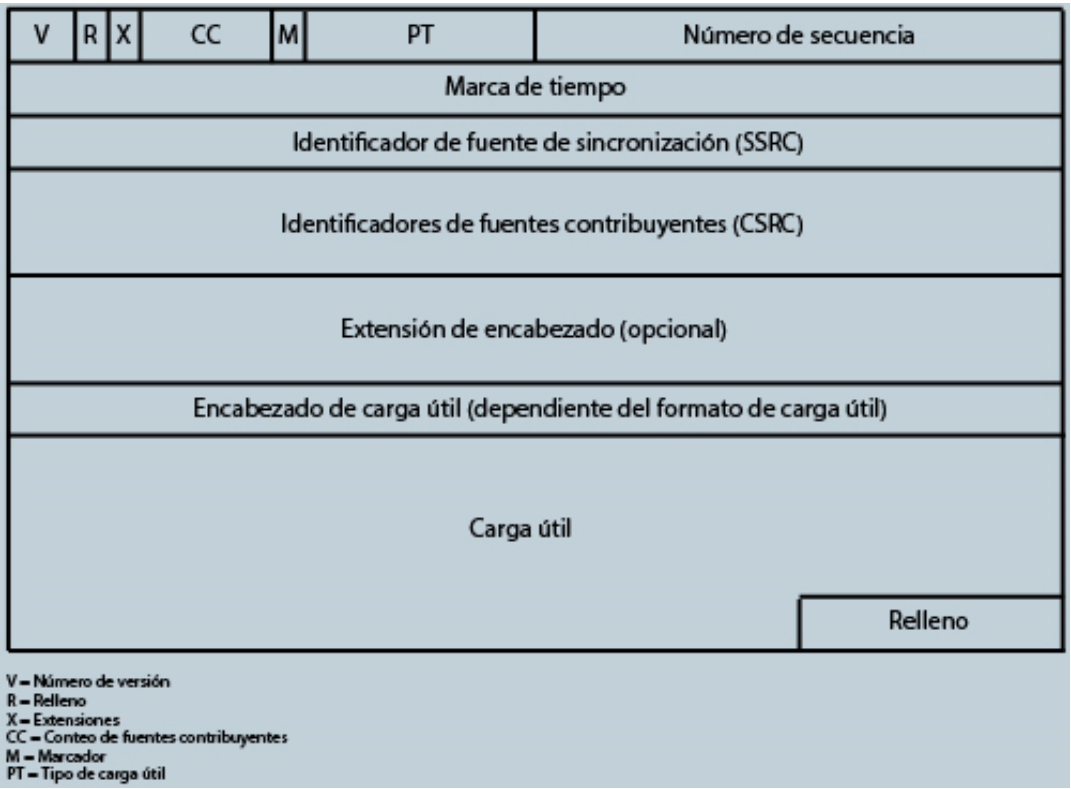


Figura 3.9: Formato de un paquete RTP.

El paquete RTP está contenido en la carga útil de una capa inferior, generalmente UDP/IP. El encabezado del paquete RTP tiene 12 octetos de longitud pero puede ser expandido con 4 a 60 octetos más. Los campos obligatorios en el encabezado son los siguientes:

- **Tipo de carga útil:** este campo identifica el tipo de media transportado por el paquete RTP. La aplicación examina el tipo de carga útil, para determinar cómo se debe tratar la data, la exacta interpretación se define por el perfil RTP que mapea los números del tipo de carga útil con las especificaciones de formato de carga útil.

- **Numero de secuencia:** este identificador en cada paquete permite a la aplicación determinar si se perdió algún paquete o fue entregado fuera de orden. Es un entero de16 bits de longitud, esto causa que se alcance el valor límite muy a menudo, por lo que se reinicia a cero y se tiene una cuenta del número de veces que se ha reiniciado el número de secuencia. A fines de disminuir la efectividad de un ataque el número de secuencia se inicia con un valor aleatorio.

- **Marca de tiempo:** en este campo se almacena el instante de tiempo del muestreo del primer octeto de la data en el paquete, es usado para realizar la reproducción de la data. La marca de tiempo es un entero sin signo de 32 bits que es reiniciado a cero una vez que se alcanza el valor límite. Con los codificadores comúnmente usados se utiliza una velocidad de reloj de 90KHz, lo que hace que se reinicie el campo cada 13 horas, esta velocidad de reloj es especificada por el perfil o el

formato de carga útil, debe ser suficiente para realizar la sincronización de labios y medir la variación en el tiempo de tránsito. Al igual que el número de secuencia se inicia con un valor aleatorio.

- **Fuente de sincronización (*Synchronization Source*, SSRC):** este entero de 32 bits identifica a cada participante en una sesión RTP, es generado localmente de manera aleatoria por lo que puede ocurrir una colisión, en este caso el participante que detecta la colisión al recibir un paquete con su mismo identificador debe enviar un paquete RTCP BYE y seleccionar otro identificador de fuente de sincronización.

- **Fuentes contribuyentes (*Contributing Sources*, CSRC):** cuando la data pasa a través de un mezclador o un traductor donde distintas fuentes contribuyen en un paquete RTP se usa una lista de fuentes contribuyentes, esta identifica a los participantes que contribuyeron al paquete RTP pero no son responsables de su tiempo ni sincronización. Cada identificador es un entero de 32 bits correspondiente al SSRC de cada contribuyente, la longitud de la lista se indica en el campo CC del encabezado RTP.

- **Marcador (M):** el bit marcador es usado para indicar eventos de interés dentro de una transmisión multimedia, su significado es dado por el perfil y el tipo de data usada. Por ejemplo para una aplicación con un perfil de audio el bit marcador se enciende para indicar el envío de un paquete luego de un período de silencio, mientras que en una aplicación con un perfil de video el bit marcador es usado para indicar el último paquete perteneciente a un *frame*. Esta utilidad sirve sólo como una pista, las aplicaciones deben ser diseñadas para funcionar incluso si la información del bit marcador se pierde.

- **Relleno o Padding (P):** el bit de relleno es usado para indicar que la carga útil ha sido rellenada más allá de su longitud original. Cuando se rellena la carga útil este bit se enciende y el último octeto de la carga útil contiene el número de octetos de relleno.

- **Encabezado de carga útil:** en varios casos el formato de carga útil necesita más información de la que puede proporcionar el encabezado RTP mandatorio, por eso se añade un encabezado adicional en la carga útil que es definido como parte del formato de carga útil. Este encabezado se agrega luego de la lista de fuentes contribuyentes. La razón principal por la que se incluye el encabezado de carga útil es para proveer resistencia a errores.

- **Carga útil:** uno o más *frames* de data que siguen luego de cualquier encabezado de carga útil, el tamaño de la carga útil depende del formato de carga útil y parámetros acordados durante el establecimiento de la sesión.

3.3.3 Protocolo de control RTP (RTCP)

El protocolo de control de RTP provee de reportes periódicos de calidad de recepción, identificación de participantes, descripción de fuentes, notificaciones de los cambios ocurridos en la sesión y la información requerida para sincronizar la transmisión multimedia. Está conformado básicamente por tres partes: **formatos de paquete**, **reglas de tiempo** y una **base de datos de los participantes**.

Existen cinco formatos de paquete RTCP definidos en la especificación RTP, estos siguen la estructura básica mostrada en la Figura 3.10.

Aunque la información específica al formato cambia de acuerdo al tipo de paquete. Los campos básicos en el paquete son el número de versión, relleno, conteo de objetos para los paquetes que transportan una lista de elementos, el tipo de paquete y el campo longitud que denota el tamaño del contenido del paquete luego del encabezado básico.

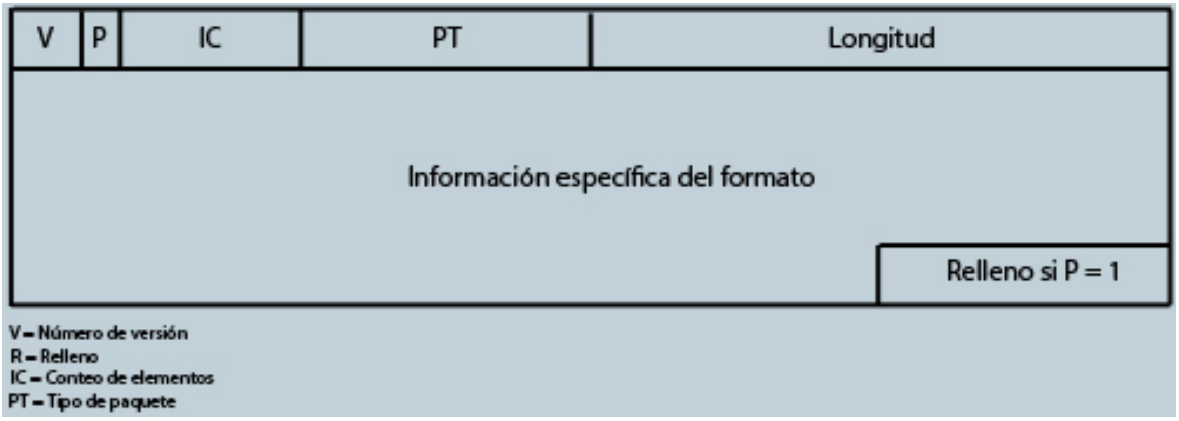


Figura 3.10: Formato de un paquete RTCP.

Los cinco tipos de paquetes se describen a continuación:

- **Paquete de Reporte de Receptor (*Receiver Reports*, RR):** estos son enviados por cualquier participante que reciba data. Es identificado con el tipo de paquete 201 y contiene el SSRC del participante que envió el reporte. Cada bloque de reporte describe la calidad de una fuente de sincronización de la cual el receptor ha recibido paquetes RTP durante el intervalo de reporte actual. Un total de 31 bloques de reporte puede ser transmitido en cada paquete RR RTCP.

- **Paquete de Reporte de Transmisor (*Sender Reports*, SR):** este paquete es identificado con el tipo 200 y contiene marcas de tiempo así como el conteo de paquetes RTP y octetos de carga útil dentro de esos paquetes, esto permite calcular la tasa de transferencia en espacios de tiempo donde no se recibió data. Además las marcas de tiempo son usadas para crear una correspondencia entre los relojes con una referencia externa y hacer posible la sincronización labial.

- **Paquete de Descripción de Fuente:** el identificador de este tipo de paquete es 202. Es usado para proveer información adicional sobre los participantes, datos como ubicación, dirección de correo electrónico y número telefónico. Contiene una entrada para cada SSRC/CSRC con una lista de los datos pertenecientes a dicha entrada. Los datos estándar definidos en la especificación RTP son CNAME, NAME, EMAIL, PHONE, LOC, TOOL, NOTE y PRIV.

- **Paquete de control de membrecía (BYE):** es usado para indicar que algún participante abandono la sesión o cambió de SSRC. Es identificado con el tipo de paquete 203. Los paquetes BYE pueden perderse por lo que un receptor debe estar preparado para revocar la sesión de un participante del que no se haya recibido data durante un período de tiempo.

- **Paquete RTCP definido por la aplicación (APP):** es usado para extensiones que no forman parte del estándar RTCP, así como fines experimentales. Se identifica con el tipo de paquete 204. Una gran variedad de aplicaciones usan este tipo de paquetes por lo que también deben ser diseñadas para ignorar paquetes no reconocidos definidos por otras aplicaciones.

Las **reglas de tiempo** definen la tasa de envío de los paquetes RTCP, ésta no es fija, varía en base al tamaño de la sesión y el formato multimedia utilizado, generalmente se busca restringir el tráfico RTCP a un 5% del ancho de banda de la sesión. Mientras más participantes existan en la sesión el intervalo de reporte se incrementa de modo que se disminuya la congestión de la red. Se toman en consideración para determinar el intervalo de reporte el ancho de banda asignado a RTCP, el tamaño promedio de los paquetes RTCP, el número total de participantes y la fracción de éstos que son remitentes.

Por último cada aplicación debería mantener una **base de datos de participantes** almacenando la información obtenida de los paquetes RTCP que recibe, ésta información es usada para llenar los paquetes de reporte de receptor que se envían de manera periódica, además de la sincronización labial entre los flujos de audio y video y mantenimiento de información y descripción de las fuentes [18][19].

3.4 Estándares para la codificación de audio y video

Para enviar la información de audio y video es necesario codificarla para llevarla de un formato analógico a uno digital y luego comprimir esta data de manera que se reduzca su tamaño para facilitar su transmisión en el medio, aunque los estándares para la codificación de audio y video deben manejar un equilibrio entre la compresión y la calidad del audio y la imagen que el receptor va a percibir según sus requerimientos y capacidades, los últimos estándares desarrollados permiten la transmisión de gran cantidad de data comprimida con muy poca pérdida de información lo cual proporciona la capacidad de compartir video de alta definición y audio de gran calidad. A continuación se revisarán el estándar H.264 para la

codificación video, sus características y su uso en la transmisión de video y audio en tiempo real a través de redes inalámbricas.

3.5 Estándar H.264 para codificación de video

El estándar H.264, especificado en la Recomendación H.264 de la Unión Internacional de Telecomunicaciones (*International Telecommunication Union*, ITU) y la Organización Internacional para Estandarización (*International Organization for Standardization*, ISO)/Comisión Electrotécnica Internacional (*International Electrotechnical Commission*, IEC) [20], describe la compresión de video, de manera tal que el resultado sea un formato en el que el video ocupe menos espacio al ser transmitido a la vez que provee una gran calidad de video. En el documento del estándar se define el formato o la sintaxis para la compresión del video y el método para decodificar ésta sintaxis y obtener un video reproducible. El estándar H.264/Códec de Video Avanzado (*Advanced Video Coding*, AVC) fue publicado por primera vez en el año 2003, realizado en base a estándares anteriores como MPEG-2 y MPEG-4 pero ofrece mayor flexibilidad y capacidad de compresión.

El funcionamiento de un codificador H.264 se resume en tres pasos los cuales son predicción, transformación y codificación, mientras que el decodificador realiza las tres operaciones complementarias que son decodificación, transformación inversa y reconstrucción como lo muestra la Figura 3.11.

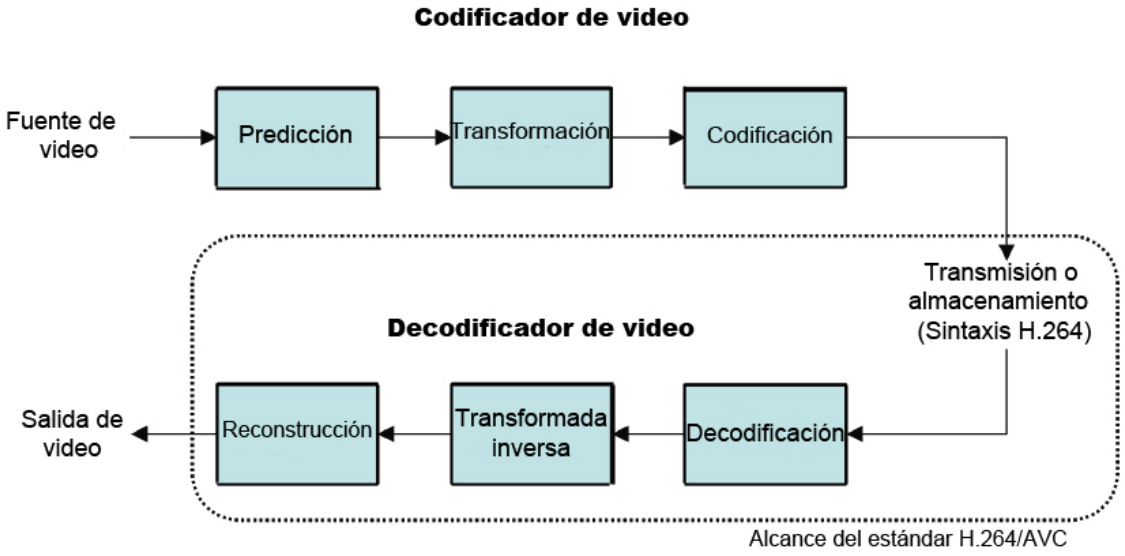


Figura 3.11: Codificador de vídeo.

3.5.1 Predicción

Dentro del codificador se divide cada trama en unidades de 16x16 pixeles llamadas **macrobloques**, luego para cada macrobloque se forma una predicción en base a información

codificada anteriormente, sea del *frame* actual (intra-predicción) o de *frames* anteriores, (inter-predicción) ésta predicción se resta al macrobloque actual creando un **residual.**

Para llevar a cabo la intra-predicción el macrobloque de 16x16 pixeles se divide en bloques de 4x4 y cada muestra dentro de este bloque es predicha usando las muestras que la rodean que ya han sido reconstruidas tanto en el codificador como en el decodificador. Existe nueve modos de realizar la predicción, uno corresponde a tomar la media de todas las muestras que rodean al bloque de 4x4 y las ocho restantes a las posibles direcciones en las que se pueden combinar las muestras como se muestra en la Figura 3.12.

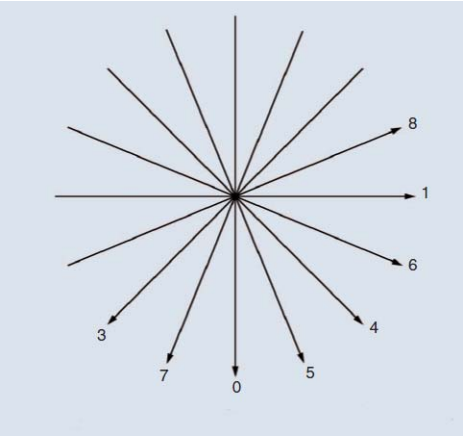


Figura 3.12: Nueve modos de la intra-predicción.

Una alternativa a los modos anteriores es predecir las componentes de luminosidad del macrobloque sin dividir este en bloques, para esto se tienen cuatro modos: predicción vertical, predicción horizontal, predicción DC y predicción plana, están ilustrados en la Figura 3.13. Esta metodología tiene mejor cabida en áreas de la imagen donde las muestras son homogéneas.

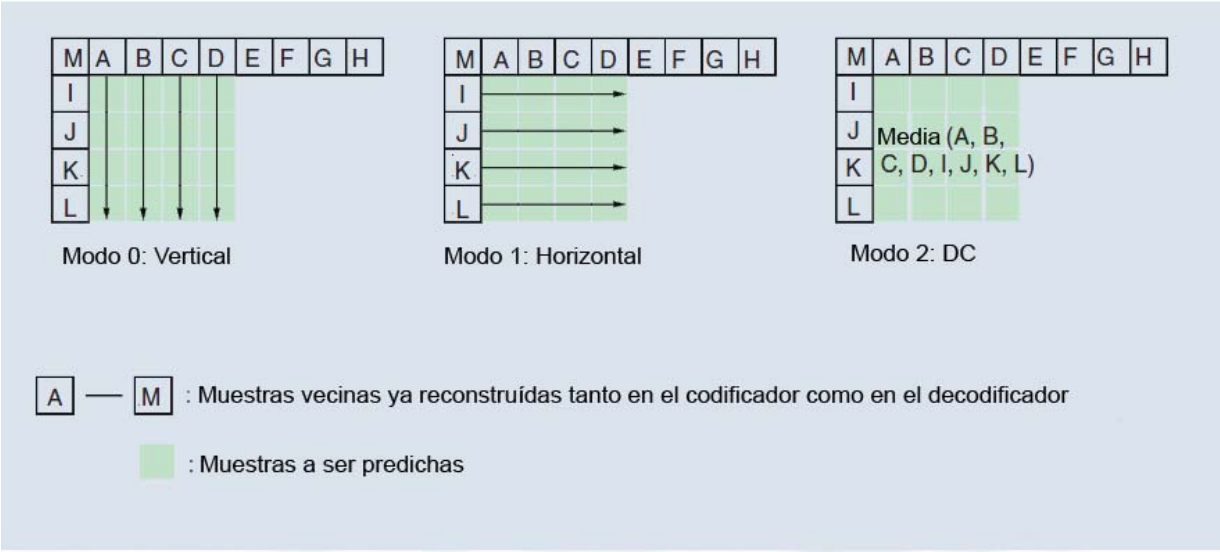


Figura 3.13: Otros modos de la intra-predicción.

En el caso de la inter-predicción, donde se usa información de otros *frames* previamente procesados, los macrobloques pueden ser divididos en particiones de tamaños 16x16, 16x8, 8x16 y 8x8, en éste último caso es posible subdividir la partición de 8x8 de manera similar. Es necesario crear un vector de desplazamiento que refiere a la posición de las muestras en la imagen de referencia ya transmitida, en estándares anteriores esta imagen es la imagen precedente más reciente, mientras que en H.264/ACV es posible hacer referencia a varias imágenes anteriores, para lo cual hay que codificar junto con el vector de desplazamiento un parámetro adicional que indica a cual imagen de referencia corresponde.

La precisión del vector de desplazamiento es de un cuarto de pixel, esta precisión fraccionaria puede hacer que un vector haga referencia a posiciones en la imagen de referencia que estén ubicadas entre dos muestras, para solucionar este inconveniente se realiza una interpolación usando las muestras adyacentes para generar las muestras necesarias para la predicción. Para esta interpolación se emplea un filtro de Respuesta Finita al Impulso (*Finite Impulse Response*, FIR) de sexto orden, lo que implica que cada muestra de medio pixel es la suma ponderada de seis muestras adyacentes, luego cada muestra de un cuarto de pixel es obtenida mediante la interpolación bilineal de las muestras adyacentes de medio o un pixel completo.

Para codificar los vectores de desplazamiento pueden ser necesarios un número significante de bits, en especial si se eligen tamaños pequeños de particiones, como los vectores de desplazamiento están altamente relacionados para las particiones vecinas se pueden predecir dichos vectores. El vector es predicho en base a la media de los vectores de desplazamiento arriba, diagonalmente arriba y a la derecha y a la izquierda de la partición o sub-particiones del macrobloque actual, si el macrobloque es omitido (no transmitido), se genera un vector de desplazamiento como si se hubiera codificado el macrobloque con una partición de 16x16.

3.5.2 Transformación y cuantización

Para la transmisión cada bloque residual es transformado usando una transformada entera de 8x8 o 4x4 que es una aproximación de la Transformada Discreta del Coseno (*Discrete Cosine Transform*, DCT) [20]. La transformada arroja como salida una serie de coeficientes, cada uno de los cuales es un valor ponderado para un patrón base estándar.

El bloque de coeficientes producto de la transformada, es luego cuantizado, es decir cada coeficiente es dividido por un valor entero y el resultado se redondea al valor entero más cercano, cuanto mayor sea el valor de las coordenadas del coeficiente mayor es el número o Parámetro de Cuantización (*Quantization Parameter*, QP) por el que se va a dividir lo que hace más probable que el resultado sea nulo, este proceso reduce la precisión de los coeficientes transformados de acuerdo al parámetro de cuantización, si este es menor se produce una menor compresión pero una imagen con menos perdidas, si este parámetro se incrementa la compresión es mayor pero produce más perdida en la calidad de la imagen decodificada.

Este método es utilizado en estándares anteriores con buenos resultados, en el caso de H.264 al usar una transformada entera exacta con matrices de 8x8 se logra mejorar la compresión en áreas altamente correlacionadas de la imagen y usando matrices de 4x4 se reduce el efecto de "*ringing*" (artefactos y errores pequeños en la imagen), estas transformadas son similares a la DCT pero simplificadas y modificadas para proveer una decodificación más precisa que en estándares anteriores.

3.5.3 Codificación

El proceso de codificación de video produce una serie de valores que deben ser codificados para formar un flujo de bits comprimido, entre estos valores están:

- Coeficientes transformados y cuantizados.
- Información que permita al decodificador recrear las predicciones.
- Información sobre la estructura de los datos comprimidos y las herramientas usadas para la codificación.
- Información sobre la secuencia de video completa.

Todos estos parámetros y valores son convertidos en códigos binarios, dentro del estándar H.264 se especifican dos tipos de codificación entrópica para lograr esto: una técnica de poca complejidad que usa un conjunto de códigos de longitud variable conmutados adaptables al contexto, también llamado Codificación de Longitud Variable Adaptable al Contexto (*Context Adaptative Variable Length Coding*, CAVLC) y otra técnica que demanda más poder de computo basada en codificación binaria aritmética adaptable al contexto, Codificación Aritmética Binaria Adaptable al Contexto (*Context Adaptive Binary Arithmetic Coding*, CABAC). Ambos métodos representan mejoras respecto a técnicas de codificación estática de estándares anteriores, donde se usaban para cada elemento o conjunto de elementos de sintaxis Codificación de Longitud Variable (*Variable Length Coding*, VLC) especialmente definido pero fijo, donde se asumía que las estadísticas subyacentes eran estacionarias, pero en la práctica este no es el caso. La data de los residuales de un codificador con predicción por compensación de movimiento muestra un comportamiento no estacionario que depende del contenido del video, las condiciones de codificación y de la precisión del modelo de predicción.

3.5.4 CAVLC (*Context Adaptative Variable Length Coding*)

CAVLC es el método básico para codificación de los bloques residuales ya transformados en el estándar H.264/AVC, mediante este procedimiento se aprovecha varias características de los bloques transformados y cuantizados:

- Luego de los procesos de predicción, transformación y cuantización, los bloques son dispersos, conteniendo muchos ceros en la mayoría de los casos por lo que CAVLC representa de manera compacta estas cadenas de ceros.

- Los coeficientes diferentes de cero con frecuencias más altas son generalmente secuencias de +/-1, llamados "*Trailing 1s*" o "T1", son también codificados de manera compacta por este método.

- El número de coeficientes diferentes de cero en bloques vecinos está correlacionado, este valor es codificado usando una tabla de búsqueda, dicha tabla se elige dependiendo del número de coeficientes en bloques vecinos.

- El nivel o magnitud de los coeficientes al principio del arreglo tiende a ser mayor cerca del coeficiente DC y se reduce hacia las frecuencias más altas. CAVLC aprovecha esta cualidad adaptando la elección de la tabla de búsqueda VLC para el parámetro del nivel dependiendo de los niveles de magnitud recientemente codificados.

Para codificar un bloque los coeficientes son escaneados haciendo un recorrido en zigzag formando un arreglo como se muestra en la Figura 3.14, luego son convertidos en una serie de códigos VLC, para lo cual se eligen una serie de tablas dependiendo de estadísticas locales al bloque actual y a bloques vecinos. La codificación un sub-bloque de 4x4 requiere codificar los siguientes parámetros:

- El número de coeficientes diferentes de cero (*Total-Coeffs*) y filas de unos (T1).

- El patrón de las filas de unos (T1).

- Los coeficientes diferentes de cero (Niveles).

- Número de ceros entre los coeficientes diferentes de cero (*Total_zeros*).

- La ubicación de dichos ceros entre los coeficientes diferentes de cero (*run_before*).

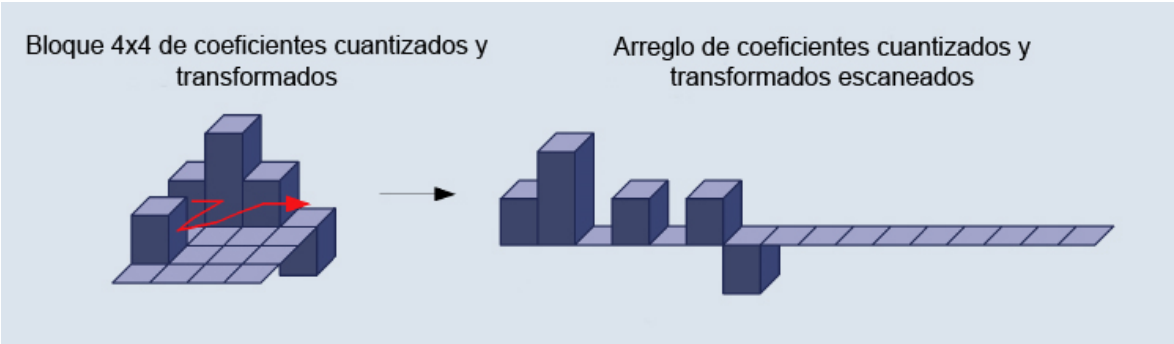


Figura 3.14: Codificación en zigzag.

Para realizar la codificación de cada bloque se procede de la siguiente manera:

Codificar el número de coeficientes y *Trailing* 1s.

El primer código de longitud variable, *coeff_token*, codifica los dos primeros parámetros mencionados anteriormente, *Total-Coeffs* y T1, el número de coeficientes diferentes de cero puede ser desde cero (lo que significa que no hay coeficientes en el sub-bloque de 4x4) hasta 16, el valor de T1 puede estar desde 0 hasta 3, sólo los 3 últimos +/-1 son tomados como caso especial, si existen más unos en fila son tomados como coeficientes normales.

Para la elección de la tabla VLC correspondiente al parámetro *coeff_token* se toma en consideración una variable *nC*, cuyo valor es determinado por el número de coeficientes en el sub-bloque a la izquierda (*nA*) y el sub-bloque superior (*nB*) codificados previamente al actual, *nC* es calculado según los dos siguientes casos:

- Si tanto *nA* como *nB* están disponibles entonces *nC* = (*nA* + *nB* + *1*) >> 1, donde el operador >> indica el desplazamiento binario hacia la derecha.

- Si alguno de los dos o ambos, *nA* o *nB*, no está disponible entonces su valor es cero y *nC* = (*nA* + *nB*).

Con este método se logra, en base al valor de *nC* (Tabla valores *nC*), que la elección de la tabla VLC se adapte al contexto, ya que depende del número de coeficientes en los sub-bloques anteriores, de modo que en la tabla VLC 1 dirigida a un número pequeño de coeficientes se asignan códigos muy cortos a los valores bajos de *Total-Coeffs* (0 y 1) mientras que a los valores altos le son asignados códigos largos, para la tabla VLC 2 que favorece a un número medio de coeficientes, se asignan a los valores medios de *Total-Coeffs* (2 a 4) códigos relativamente cortos, en el caso de la tabla VLC 3 para una cantidad mayor de coeficientes se asignan un Código de Longitud Fija (*Fixed Length Code*, FLC) de 6 bits a cada uno de los valores de *Total-Coeffs*.

Tabla 3.1: Elección de la tabla VLC correspondiente al parámetro coeff_token.

nC	**Tabla para *coeff_token***
0, 1	Tabla VLC 1
2, 3	Tabla VLC 2
4, 5, 6, 7	Tabla VLC 3
8 o más	FLC

Codificar los signos de los *Trailing* 1s

Para cada uno en la fila de unos se codifica su signo con un bit, donde 0 equivale a un valor positivo y 1 a un valor negativo (0=+, 1=-), codificados en orden inverso, es decir, desde los coeficientes de más alta frecuencia hasta los de más baja frecuencia. Se tendrá un máximo de 3 bits para los signos, correspondiente al máximo de unos en la fila de unos.

Codificar los niveles de los coeficientes diferentes de cero restantes

El nivel se refiere al signo y la magnitud de los coeficientes en el bloque distintos de cero y no están entre la fila de unos. La codificación se realiza en orden inverso y la elección del código VLC para cada nivel es adaptativa al contexto ya que depende de la magnitud del nivel anterior.

El código VLC se compone de los siguientes elementos:

- En primer lugar un prefijo llamado *level_prefix*.
- Una cantidad *b* de ceros terminando seguida de un 1.
- Por último un sufijo que es un código de longitud igual al valor *suffixlength*.

El valor de *suffixlength* es determinado en base a umbrales, cuando la magnitud del coeficiente actual supera el umbral se incrementa el valor de *suffixlength* según la Tabla 3.2, la longitud del código se adapta a que generalmente la magnitud de los coeficientes aumenta conforme estos tienen frecuencias bajas, códigos más cortos son asignados a los coeficientes con menor magnitud y códigos más largos son asignados a los coeficientes con mayor magnitud lo cual presenta un incremento de la eficiencia en la codificación de los valores. El incremento del sufijo se realiza de la siguiente manera:

- Se inicializa *suffixlength* en 0 a menos que haya más de 10 coeficientes distintos de cero y menos de 3 unos en la fila de unos (*Trailing 1s*) en cuyo caso se inicializa en 1.
- Se procede a codificar el primer coeficiente distinto de cero del bloque de 4x4.
- Si la magnitud de este coeficiente supera el umbral actual se incrementa *suffixlength*, hasta un máximo de 6.

Tabla 3.2: Umbrales para determinar cuándo incrementar el valor suffixlength.

***suffixlength* actual**	**Umbral para incrementar *suffixlength***
0	0
1	3
2	6
3	12
4	24
5	48
6	Máximo valor alcanzado

Codificación del número de ceros antes del último coeficiente

La codificación con un código VLC de la suma de los ceros que preceden al coeficiente diferente de cero más alto del arreglo reordenado (*Total_zeros*) se realiza en base a que muchos bloques tienen coeficientes iguales a cero al principio del arreglo, lo que significa que los ceros al principio del arreglo no necesitan ser codificados.

Codificación de los ceros que preceden a cada coeficiente distinto de cero (*run_before*)

Se codifica un valor *run_before* para cada coeficiente distinto de cero comenzando desde el de más alta frecuencia, se plantean dos excepciones:

- Si no hay más ceros para codificar (∑ [*run_before*] = *Total_zeros*) no es necesario codificar más valores *run_before*.

- No es necesario codificar un valor *run_before* para el coeficiente de menor frecuencia.

Para el código VLC de los valores *run_before* se toma el número de ceros que no han sido codificados aún y *run_before*, por ejemplo, si restan 2 ceros, *run_before* sólo puede tomar 3 valores (0, 1 ó 2) de modo que no se necesitan más de dos bits, si restaran 6 ceros los valores estarían entre 0 y 6 para lo cual la tabla VLC debe ser lo suficientemente grande [20].

Ejemplo:

Bloque 4 x 4:

0	3	-1	0
0	-1	1	0
1	0	0	0
0	0	0	0

Bloque reordenado:

0, 3, 0, 1, -1, -1, 0, 1, 0…

Total-Coeffs = 5, indexado desde la frecuencia más alta, 4, hasta la frecuencia más baja, 0.

Total_Zeros = 3

T1s = 3. Hay cuatro *Trailing 1s* pero sólo tres pueden ser codificados como un caso especial.

Codificación:

Elemento	Valor	Código
coeff_token	*Total-Coeffs = 5, T1s = 3 (usa Num_VLC0)*	0000100
T1 sign (4)	+	0
T1 sign (3)	-	1
T1 sign (2)	-	1
Level (1)	*+1 (*levelprefix = *1;* suffixlength = *0)*	1

Level (0)	*+3 (levelprefix = 001, suffixlength = 1)*	0010
Total_Zeros	*3*	111
run_before(4)	*ZerosLeft = 3; run before =1*	10
run_before(3)	*ZerosLeft = 2; run before =0*	1
run_before(2)	*ZerosLeft = 2; run before =0*	1
run_before(1)	*ZerosLeft = 2; run before =1*	01
run_before(0)	*ZerosLeft = 1; run before =1*	Código no necesario, último coeficiente.

El conjunto de bits transmitidos para éste bloque es 000010001110010111101101.

Decodificación:

El arreglo de salida es reconstruido a partir de los valores decodificados como se muestra a continuación:

Código	Elemento	Valor	Arreglo de salida
0000100	*coeff_token*	*Total-Coeffs = 5, T1s = 3*	*Empty*
0	*T1 sign*	*+*	1
1	*T1 sign*	*-*	-1, 1
1	*T1 sign*	*-*	-1, -1, 1
1	*Level*	*+1 (suffixlength = 0; incrementsuffixlength)*	1, -1, -1, 1
0010	*Level*	*+3 (suffixlength = 1)*	3, 1, −1, −1, 1
111	*Total_Zeros*	*3*	3, 1, −1, −1, 1
10	*run_before*	*1*	3, 1, −1, −1, 0, 1
1	*run_before*	*0*	3, 1, −1, −1, 0, 1
1	*run_before*	*0*	3, 1, −1, −1, 0, 1
01	*run_before*	*1*	3, 0, 1, −1, −1, 0, 1

El decodificador ya ha insertado dos ceros, *Total_Zeros* es igual a 3 por lo que un cero más es insertado antes del coeficiente más bajo, esto nos da el arreglo de salida final:

0, 3, 0, 1, −1, −1, 0, 1

3.5.5 CABAC (Context-Adaptive Binary Arithmetic Coding)

Este es un método de codificación opcional dentro del estándar H.264, generalmente usado para el perfil principal (*Main*) y alto (*High*).CABAC es el método a elegir para una mejora significativa en la eficiencia del proceso de codificación, da una reducción de la tasa de bits que va desde un 5% hasta un 15% con respecto a CAVLC. Se basa en la binarización, el modelado del contexto y la codificación aritmética binaria. La calidad de la compresión se logra mediante 3 procesos:

- Selección de modelos de probabilidad para cada elemento de sintaxis de acuerdo al contexto de dicho elemento.

- Adaptación de los estimados de probabilidad basándose en estadísticas locales.

- Uso de codificación aritmética en lugar de codificación mediante códigos de longitud variable (VLC).

El proceso de codificación se realiza a través de las siguientes fases [21]:

- Es necesaria una binarización de la data ya que CABAC maneja una codificación aritmética binaria, esto quiere decir que sólo se codifican decisiones binarias (1 ó 0), por ejemplo, un símbolo no binario es binarizado o convertido a un código binario antes de la codificación aritmético binaria, similar a convertir un símbolo de data en un código de longitud variable, sólo que el código binario VLC es codificado nuevamente por el codificador aritmético antes de ser transmitido.

- Un modelo de contexto es un modelo de probabilidad para uno o más *bins* del símbolo binarizado que contiene la probabilidad de que cada *bin* sea 1 ó 0. Este modelo es elegido de un grupo de modelos disponibles dependiendo de las estadísticas de símbolos recientemente codificados.

- Un codificador aritmético codifica cada uno de los *bins* de acuerdo al modelo de probabilidad seleccionado, cabe destacar que sólo habrá dos sub-rangos para cada *bin*, que corresponden a las probabilidades de los valores 0 y 1. En caso de que la probabilidad del *bin* de contener 0 ó 1 permanezca en 0,5 se codifica el *bin* mediante un codificador bypass simple.

- Luego se actualiza el modelo de probabilidad en base al *bin* codificado, de modo que si es 1 se incrementa la probabilidad del valor 1 [20].

3.5.6 Técnicas de resistencia a errores y adaptación a la red

En el estándar H.264, VLC especifica una representación de la señal de video codificada bastante eficiente, provee varias características para la adaptación a la red y la resistencia a errores que son esenciales para servicios de tiempo real como *streaming* y videoconferencia, a su vez la Capa de Abstracción de Red (*Network Adaptation Layer*, NAL) define una interfaz entre el codificador de video y los elementos con los que interactúa. Mediante unidades NAL se garantiza el soporte para redes de conmutación de paquetes. El Decodificador de Referencia Hipotética (*Hypothetical Reference Decoder*, HRD) aplica restricciones sobre las unidades NAL para habilitar la implementación de un decodificador que permita variar la relación entre costo y efectividad introduciendo un modelo *leaky-bucket* múltiple.

Las aplicaciones cuya finalidad es mantener una conversación (video-conferencia), necesitan condiciones bajo retraso en la comunicación y dado que las imperfecciones en las

demás capas y la congestión producen errores que no pueden ser evitados se añade una gran complejidad al problema por lo que este tipo de aplicaciones de video en tiempo real necesitan características que le proporcionen resistencia a errores. Por estas razones dentro del estándar H.264/AVC se toman en consideración varios esquemas, mencionados en las siguientes líneas, para tratar estos inconvenientes, que, a pesar de su utilidad deben ser implementados con cautela de manera que no afecten la eficiencia de la compresión y más aún, si es posible aplicar corrección de errores en capas inferiores. Generalmente se asume que con una mayor compresión se reduce la resistencia a errores, pero si se aplica correctamente una mayor compresión puede dar lugar a más data para la Corrección de Errores Hacia Adelante (*Forward Error Correction*, FEC).

La codificación estructurada por segmentos (*slice structured coding*) reduce la probabilidad de pérdida de paquetes y la degradación visual por la pérdida de paquetes. El Ordenamiento Flexible de Macrobloques es una técnica más avanzada que permite la transmisión de los macrobloques de una manera más flexible. En tercer lugar, ordenar arbitrariamente los segmentos (*slices*) permite decodificarlos sin seguir el orden que dicta la llegada de las unidades NAL, de manera que se puede reducir el retraso en la decodificación por la llegada de unidades NAL fuera del orden. H.264/AVC permite segmentar la data en 3 particiones, una para el encabezado y la información de desplazamiento y la segunda partición usada en estándares anteriores se separa ahora para contener la información de la intra-predicción y la inter-predicción, esto da la posibilidad de asignar mayor prioridad a información de intra-predicción más importante.

A pesar de estas técnicas sigue siendo inevitable la perdida de paquetes y la propagación de errores, la recuperación de estos errores sólo es posible usando regiones de imágenes codificadas con intra-predicción, para esto el estándar H.264/AVC permite el envío de simples macrobloques que no pueden ser predichos de manera correcta y eficiente, también se limita la propagación de errores enviando un número de macrobloques intra-codificados anticipándose a errores de transmisión [21].

3.6 Estándar AAC para la codificación de audio

El estándar de Codificación Avanzada de Audio (*Advanced Audio Coding*, AAC) forma parte de la última especificación del comité MPEG, es considerado el sucesor de MP3 y está respaldado por compañías importantes como Dolby, Sony y Nokia [22].

Todas las mejoras sucedidas desde la creación de MP3 llevaron a una especificación aparentemente compleja con muchos tipos de AAC disponibles. Usualmente AAC está envuelto dentro de un contenedor MP4 para proveer metadata como etiquetas y permitir búsquedas. Por esto AAC también puede ser referido como audio MP4. En términos de calidad el formato AAC está a la par con otros códecs actuales como Vorbis, LAME MP3 y WMA Pro, provee alta calidad a baja tasa de bits. Los desarrollos recientes llegan hasta AACplus, usado en radio vía Internet con muy buenos resultados a bajas tasas de bits (aproximadamente 40 Kbps).

Existen diversos codificadores de empresas como Apple, Real Networks y Nero AG, también codificadores de software libre como FAAC o LAME. También existe la posibilidad de crear una implementación propia ya que hay demos y especificaciones disponibles.

Éste estándar provee una alta flexibilidad soportando tasas de muestreo entre 8000 Hz y 96000 Hz, varias tasas de bits y hasta 48 canales. AAC Alcanza transparencia en la mayoría de las muestras alrededor de los 150 Kbps. Estas ventajas son muy llamativas pero incrementan la complejidad del códec.

Como parte del estándar MPEG-4, un archivo codificado mediante AAC incluye hasta 48 canales de audio con todo el ancho de banda (96KHz), 15 canales de baja frecuencia para mejora y realce y 15 flujos de data.

Los métodos de codificación en AAC están organizados en Perfiles o Tipos de Objetos. Estos tipos de objetos no necesariamente compatibles entre ellos, algunos se listan a continuación:

- MPEG-2 AAC LC / *Low Complexity*.
- MPEG-2 AAC *Main*.
- MPEG-2 AAC SSR / *Scalable Sampling Rate*.
- MPEG-4 AAC LC / *Low Complexity*.
- MPEG-4 AAC *Main*.
- MPEG-4 AAC SSR / *Scalable Sampling Rate*.
- MPEG-4 AAC LTP / *Long Term Prediction*.
- MPEG-4 AAC HE / *High Efficiency*.
- MPEG-4 AAC LD / *Low Delay*.

La diferencia entre estos radica en su complejidad, por lo que algunos tardan más en codificar y decodificar que otros. Generalmente lo beneficios obtenidos mediante métodos más complejos no valen el costo de procesamiento necesario para llevarlos a cabo. Esto da como resultado que el tipo de objeto MPEG-4 AAC Baja Complejidad (*Low Complexity*, LC) se ha convertido en el más popular.

Entre las tecnologías usadas para la compresión en AAC están:

- *Huffman coding*.
- *Quantization and scaling*.

- *M/S matrixing*.
- *Intensity stereo*.
- *Channel coupling*.
- *Backward adaptive prediction*.
- *Temporal Noise Shaping* (TNS).
- *Modified Discrete Cosine Transform* (IMDCT).
- *Gain control and hybrid filter bank* (*poly phase quadrature filter* (IPQF) + IMDCT).

3.7 Estándar MP3 para la codificación de audio

MPEG-1 Audio Layer 3, más conocido como MP3, es un formato popular de codificación de audio digital y compresión con pérdida, diseñado para reducir considerablemente la cantidad de datos necesarios para representar el audio, pero suena como una reproducción fiel del sonido original sin comprimir para la mayoría de los oyentes. Fue inventado por un equipo de ingenieros europeos que trabajaron en el marco del programa EUREKA 147 DAB de radio digital de investigación, y se convirtió en un estándar ISO/IEC en 1991 [23].

Para la codificación de audio MP3 el estándar MPEG-1 no incluye una especificación precisa de un codificador MP3. El algoritmo de decodificación y formato de archivo, al contrario si están bien definidos. Como resultado, hay diferentes codificadores MP3 disponibles, cada uno produciendo archivos de distinta calidad. Las comparaciones están ampliamente disponibles, por lo que es fácil para un usuario potencial de un codificador buscar la mejor opción.

El sistema de codificación que utiliza MP3 es un algoritmo de pérdida. Es decir, el sonido original y el que se obtiene posteriormente no son idénticos. Esto se debe a que MP3 aprovecha las deficiencias del oído humano y elimina toda aquella información que no somos capaces de percibir.

Para realizar ésta acción de “pérdida de información” se utiliza un sistema llamado Codificación de Sub Bandas, proceso por el cual la señal se descompone en sub bandas a través de un banco de filtros. Estas sub bandas se comparan a continuación con el original mediante un modelo psicoacústico que es el encargado de determinar que bandas se pueden eliminar y cuáles no. Dependiendo de la calidad que se desea obtener, se eliminarán más o menos bandas. Para finalizar el proceso, se cuantifican y codifican las sub bandas resultantes, y el resultado final se comprime mediante un algoritmo estándar, obteniendo así el archivo MP3 resultante.

Por otro lado la decodificación de audio MP3 está cuidadosamente definida en el estándar. La mayoría de los descodificadores son “flujo de bits compatible”, lo que significa que la salida descomprimida que producen a partir de un determinado archivo MP3 será el mismo (dentro de un grado de tolerancia especificado de redondeo) como el de salida especificado matemáticamente en el documento ISO/IEC. La información de cabecera y ayuda en el lado del decodificador para decodificar los datos codificados de *Huffman* asociados correctamente.

La estructura de un archivo MP3 se constituye de diferentes tramas MP3 que a su vez se componen de una cabecera MP3 y los datos MP3. Esta secuencia de datos es denominada “flujo elemental”. Cada uno de las tramas es independiente, es decir, se pueden cortar las tramas de un archivo MP3 y después reproducirlos. La cabecera consta de una palabra de sincronismo que es utilizada para indicar el principio de una trama válida. A continuación siguen una serie de bits que indican que el archivo analizado es un archivo estándar MPEG y si usa o no la capa 3. Después de todo esto, los valores difieren dependiendo del tipo de archivo MP3.

4 Metodología y herramientas de trabajo

Para el desarrollo de cualquier trabajo de investigación es necesario tener una metodología de trabajo definida con el objetivo de estructurar, planear y controlar todo el proceso. En este capítulo se describe la metodología de desarrollo de software que fue implementado en la elaboración de este trabajo así como el conjunto de herramientas que fueron utilizadas.

4.1 Metodología de desarrollo de software

Para el proceso de desarrollo de la aplicación se implementó la metodología de desarrollo ágil Scrum [24], sin embargo existen aspectos definidos en diversas fuentes sobre éste método de desarrollo de software que no se adaptaron fácilmente a las características de éste proyecto por lo que fueron modificados u omitidos. La metodología se implementó siguiendo las siguientes fases:

- Durante la **fase inicial de planificación** se realizó la captura de requerimientos del equipo de SOS Telemedicina y se definió una arquitectura para el proyecto. Ésta estuvo sujeta a cambios pero fue necesaria para dar comienzo a la siguiente etapa.

- Luego se procedió a la **fase de desarrollo**, en esta el objetivo fue completar el proyecto de manera incremental por medio de una serie de sub fases llamadas *sprints*, cada una de éstas tiene una corta duración, generalmente de una a dos semanas, en éste intervalo de tiempo se culminan las tareas asignadas para ese *sprint.* Las tareas consideradas más complejas y que conllevan más tiempo se subdividen en tareas más pequeñas y/o simples para poder cumplir los objetivos de cada *sprint*.

- Las tareas a cumplir son organizadas por prioridad en una lista llamada *backlog*, ésta lista es revisada y se actualizan las prioridades de cada tarea antes de cada *sprint*. Una parte resaltante de esta metodología es que plantea realizar reuniones del equipo muy seguidas, en donde se rinde cuenta de los avances realizados, que obstáculos se presentaron y que cosas se planean cumplir antes de la siguiente reunión [24].

- Luego de dar por culminada la fase de desarrollo se procedió a la **fase de pruebas**. Se realizaron pruebas para medir el rendimiento de la aplicación en cuanto a uso de CPU y RAM; el desempeño y comportamiento en la red se midió a través del jitter y el porcentaje de paquetes perdidos y por último se realizaron las pruebas de funcionalidad.

4.2 Herramientas

En el desarrollo del sistema se utilizaron varias herramientas en conjunto, entre ellas se encuentran lenguajes de programación, herramientas de soporte para la metodología Scrum, herramientas de desarrollo del sistema, y los *frameworks* usados tanto para el desarrollo del API como de la aplicación. A continuación se explica cada una de estas herramientas y sus características más resaltantes por la cual fueron seleccionadas.

4.2.1 Lenguajes de programación

Para el desarrollo de la aplicación se utilizó el lenguaje de programación Java. Éste es el lenguaje estándar para desarrollar sobre Android y provee los elementos necesarios para la aplicación. Se utilizó el *Software Development Kit* (SDK) que provee Android en conjunto con Eclipse.

Java es un lenguaje de programación de propósito general, concurrente y orientado a objetos desarrollado por Sun Microsystems y ahora a cargo de Oracle, uno de sus objetivos principales fue proporcionar portabilidad en el software desarrollado [26].

Aunque el lenguaje oficial para desarrollar para el sistema operativo Android es Java, estos dos mantienen importantes diferencias, el API de Java no es igual al API de Android, esta última no usa la máquina virtual Java sino una máquina virtual llamada Dalvik o ADR (*Android Runtime*).

El servicio web fue desarrollado bajo el lenguaje PHP, PHP es un lenguaje de programación de propósito general popular que es especialmente adecuado para el desarrollo web. PHP se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy, lo que ha atraído el interés de múltiples sitios con gran demanda de tráfico para optar por el mismo como tecnología de servidor [27].

Uno de los atractivos de usar PHP es su extensión PDO (*PHP Data Object*) que define una interfaz de acceso a una base de datos, su principal ventaja es que permite una abstracción para el acceso a datos. Es decir que las funciones para realizar consultas y para obtener datos son las mismas en cualquier servidor SQL utilizado (***MySQL***, ***PostgreSQL***...). Por ejemplo, si se cambia de ***MySQL*** a ***PostegreSQL***, no será necesario transformar todo el código como sería el caso de las funciones ***mysql_*** (***mysql_connect*** se reemplazaría por ***pgsql_connect***, ***mysql_query*** por ***pgsql_query***...). Pero como cada servidor SQL no ejecuta las consultas SQL de la misma manera, es posible que sea necesario reescribir algunas consultas.

SQL es un lenguaje de programación declarativo y orientado a sets que permite la manipulación de tablas de información, así como la obtención y actualización de data sobre las mismas [28]. Este lenguaje ha dado base a varias extensiones como MySQL, Oracle y Microsoft T-SQL.

4.2.2 Herramientas de soporte para la metodología Scrum

Durante todo el proceso de desarrollo se usaron un conjunto de herramientas que dieron soporte a la metodología ágil Scrum. Estas herramientas facilitan la organización de tareas en el proyecto y el mantenimiento del código de la aplicación.

Para mantener un orden, priorizar y establecer fechas límites se creó una cartelera de tareas en Trello. Trello es una página web que provee una interfaz para manejar proyectos mediante tarjetas que representan tareas [29]. Las tareas se crearon con estado “Por hacer”, se pasaban a estado “En progreso” y finalmente al culminarse se establecía el estado “Terminada”. Las tareas se corresponden con requerimientos, funcionalidades y errores a corregir dentro de la aplicación. También se manejó el progreso de la documentación haciendo uso de esta herramienta.

Para llevar el control de los cambios sobre el código de la aplicación se utilizó un repositorio de código Git usando Bitbucket para su administración. Bitbucket permite almacenar y manejar proyectos y repositorios Git en la nube, sin costo para cinco o menos usuarios [30]. Esta herramienta se utilizó en conjunto con el programa SourceTree para manejar copias locales del proyecto y desplazar los cambios realizados al repositorio Git en la nube.

4.2.3 Herramientas de desarrollo del sistema

Durante la etapa de desarrollo e implementación del sistema fueron primordiales las herramientas de virtualización. Para el desarrollo de la aplicación se utilizó VirtualBox, en esta herramienta de acceso gratuito de Oracle se instaló la central PBX, mientras que en la fase de implementación se realizó la instalación del sistema sobre una máquina virtual en un servidor de virtualización Xen. Esta diferencia de plataformas condujo a un cambio de usar Elastix como solución para la central PBX y su interfaz de administración, a usar Asterisk en conjunto con FreePBX para proporcionar las mismas funcionalidades.

Se utilizó Asterisk como una central IP PBX para soportar la comunicación en tiempo real, en este caso la videollamada. Asterisk es un *framework* para el desarrollo de aplicaciones y soluciones de comunicación en tiempo real multi-protocolo y representa la plataforma subyacente del sistema [31].

En la etapa de desarrollo de la aplicación se hizo uso de Elastix como solución integrada de IP PBX y una interfaz web para su configuración y administración, también por que proporciona una serie de herramientas que dan pie a posibles expansiones en cuanto a los servicios de comunicación que podría proveer el sistema. Elastix es una solución de software que integra un conjunto de medios y alternativas de comunicación como videollamadas, VoIP, servidor de correos, mensajería instantánea y fax.

Para la implementación del sistema se optó por una solución más simple que consiste en la instalación de Asterisk junto a FreePBX para proporcionar una interfaz gráfica web que facilitara la configuración y administración de Asterisk.

Se usó WampServer durante la etapa de desarrollo como ambiente para el desarrollo de los servicios web sobre Windows. Es una pila de soluciones para servicios web que comprende el servidor HTTP Apache, un sistema manejador de base datos MySQL y el lenguaje de programación PHP.

Las respuestas del servicio web se encuentran codificadas en formato JSON (*JavaScript Object Notation*) [33], este es un formato de intercambio de datos ligero. Este formato es fácil de leer para una persona y a su vez es sencillo generarlo y analizarlo dentro de un programa o aplicación.

JSON se basa en dos estructuras:

- Una colección de pares clave/valor.
- Una lista ordenada de valores.

La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX, La plataforma Android incluye las bibliotecas ***json.org*** que permiten trabajar de manera eficiente con archivos JSON. Pero también se puede integrar fácilmente otra solución para analizar archivos JSON.

Para el diseño del servicio web se implementó una arquitectura REST (*Representational State Transfer*) [34], se basa en un protocolo de comunicación que es, cliente-servidor, sin estado, y cacheables, haciendo uso del protocolo HTTP.

REST es un estilo de arquitectura para el diseño de aplicaciones en red. La idea es que, en lugar de usar mecanismos complejos, tales como CORBA, RPC o SOAP es utilizar HTTP para la comunicación entre las máquinas.

Generalmente los cuatro verbos HTTP principales son:

- **GET**: lee un recurso específico (por un identificador) o una colección de recursos.
- **PUT**: actualizar un recurso específico (por un identificador) o una colección de recursos. También se puede utilizar para crear un recurso específico si el identificador de recursos se conoce de antemano.
- **DELETE**: remueve/Elimina un recurso específico de un identificador.
- **POST**: crear un nuevo recurso. También funciona como un verbo por defecto para aquellas que no encajan en las otras categorías.

Las constantes pruebas de funcionalidad del servicio web fueron realizadas con una herramienta de Chrome llamada ***Advanced Rest Client***, este plugin facilita la visualización de

las solicitudes y respuestas HTTP realizadas así como los mensajes propios que retorna el servicio web. Este plugin se encuentra en el Chrome web store.

4.2.4 Frameworks

Para el desarrollo del sistema se hizo uso de una serie de *frameworks* que permitieron el desarrollo de la aplicación en Android y del servicio web. Estos fueron los *frameworks* utilizados:

- ***NGN Doubango Framework***: es una implementación para aplicaciones sobre Android de la pila NGN (*Next Generation Network*). Ésta implementación en software libre está basada en el *framework* Doubango desarrollado por la empresa Doubango Telcom. Éste conjunto de bibliotecas proveen las funcionalidades necesarias para establecer la comunicación en tiempo real de video y audio, además ofrece un conjunto de funcionalidades adicionales como mensajería instantánea y compartición de contenidos [35].

- ***Slim Framework***: es un micro *framework* en PHP que ayuda en el desarrollo de aplicaciones web y API's [36], sus principales características de este micro *framework* son:

    - Enrutamiento de alto nivel.
        - Métodos HTTP estándar y personalizados.
        - Enrutamiento con wildcards y condicionado.
        - Enrutamiento para redireccionar, detener y, de pase.
    - Plantilla para render con vistas personalizadas.
    - Cookies seguras con cifrado AES -256.
    - HTTP caché.
    - Manejo de errores y depuración.
    - Configuración sencilla.

5 Diseño del sistema

Luego de precisar los requerimientos del sistema se procedió al diseño de cada uno de los componentes que se desarrollaron y su integración con los ya existentes cómo la base de datos de SOS Telemedicina y el servidor de llamadas Asterisk.

5.1 Requerimientos del sistema

Para la aplicación se determinaron los siguientes requerimientos con los que debe cumplir el sistema que comprende la aplicación de Videollamada para Segunda Opinión Médica:

- **Autenticación del usuario**: el usuario debe ser autenticado mediante un nombre de usuario y una contraseña contra el servicio web y la central IP-PBX, así mismo cada operación al servicio web debe autenticarse con un hash asociado al usuario que se obtiene únicamente al iniciar sesión de forma exitosa.

- **Configurar parámetros de acceso a los servidores**: el usuario debe poder configurar las direcciones de acceso al servicio web y a la central IP-PBX, así como las credenciales necesarias para la autenticación.

- **Desconectarse**: el usuario podrá desconectarse y salir de la aplicación.

- **Integración con el sistema de Segunda Opinión Médica**: el sistema de Videollamada para Segunda Opinión Médica debe integrarse funcionar en conjunto con el sistema existente sin afectar su funcionamiento.

- **Crear caso**: el usuario podrá crear un caso médico, introduciendo los campos necesarios para ello, estos son: especialidad del caso médico, descripción y fecha de nacimiento del paciente.

- **Editar caso**: el usuario podrá editar un caso que haya creado, introduciendo los campos necesarios durante la creación del caso médico.

- **Eliminar caso**: el usuario podrá eliminar un caso que haya creado.

- **Consultar caso**: el usuario tendrá acceso al contenido de cada caso médico.

- **Consultar opinión**: el usuario tendrá acceso al contenido de las opiniones de cada caso médico.

- **Solicitar opinión médica**: el usuario podrá realizar una videollamada a un especialista disponible para agregar una segunda opinión a un caso médico.

- **Guardar opinión**: el usuario podrá guardar una segunda opinión médica perteneciente a un casó médico introduciendo los campos necesarios (nombre y cuerpo de la opinión).

- **Guardar y recuperar audio de segunda opinión**: el audio de cada videollamada será almacenado en la central IP-PBX. El audio de la videollamada podrá ser recuperado desde la aplicación por el usuario.

- **Editar segunda opinión**: el usuario podrá editar la opinión de un caso introduciendo los campos requeridos (nombre y cuerpo de la opinión).

- **Eliminar opinión**: el usuario podrá eliminar una opinión.

- **Instalación del sistema**: el sistema debe funcionar sobre los servidores que posee CAIBCO.

Adicionalmente se plantearon los siguientes requerimientos no funcionales:

- El sistema debe ser **usable**, debe tener una interfaz intuitiva y agradable al usuario, con colores y diseño acordes a la imagen del proyecto SOS Telemedicina.

- El sistema debe ser **escalable y mantenible**, la implementación e integración de nuevos módulos sobre el sistema de Videollamada para Segunda Opinión Médica debe ser fácil. Así mismo debe ser fácil el mantenimiento del sistema para futuras actualizaciones o correcciones.

- El sistema debe ser **robusto**, debe recuperarse errores sin producir pérdidas o inconsistencias de datos.

5.2 Diseño de la aplicación

El esquema de diseño que impone el desarrollo sobre Android basado en Activities nos llevó a diseñar la aplicación como una serie de pantallas, cada una con una o varias funciones específicas. Se optó por un modelo bastante simple, en donde luego de autenticarse el usuario, se presenta una pantalla principal desde la cual se pueden acceder a todas las demás usando el esquema básico de navegación de los dispositivos Android para realizar cualquier operación ofrecida. Cada una de estas pantallas corresponde a una clase. Además de estas clases se crearon otro conjunto de clases que representan elementos de data tanto de usuario como de casos médicos. En las siguientes secciones se encuentran los diagramas de casos de uso y de objetos correspondientes para la aplicación en donde se detallan las características de los elementos mencionados.

5.2.1 Casos de uso nivel 0

En este caso de uso se muestra las opciones básicas de la aplicación y el actor involucrado como se observa en la Figura 5.1. En la Tabla 5.1 se describe detalladamente el actor involucrado en la aplicación.

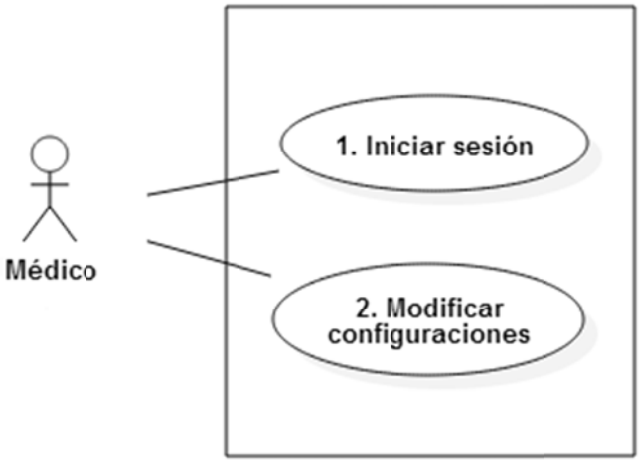


Figura 5.1: Caso de uso nivel 0 de la aplicación.

Tabla 5.1: Descripción del actor "Médico".

Actor	ACT.1 Médico
Casos de uso	1. Iniciar sesión, 2. Modificar configuraciones, 3. Conectarse, 4. Desconectarse, 5. Crear caso, 6. Listar todos los casos, 7. Modificar configuraciones, 8. Cerrar sesión, 9. Seleccionar caso, 10. Agregar opinión, 11. Llamar especialista, 12. Listar opiniones, 13. Seleccionar opinión, 14. Descargar audio.
Tipo	Primario
Descripción	Es el usuario final que hará uso de la aplicación, podrá crear casos nuevos en el sistema, agregar una opinión a un caso, visualizar las opiniones hechas a un caso, realizar una videollamada para hacer una consulta y obtener el audio de videollamadas realizadas referente a un caso.

5.2.2 Casos de uso nivel 1

Para este nivel de caso de uso se presentan las funcionalidades principales de la aplicación como se puede observar en la Figura 5.2, las descripciones de los casos de uso de

dicha figura se encuentran en las siguientes tablas: Tabla 5.2, Tabla 5.3, Tabla 5.4, Tabla 5.5, Tabla 5.6, Tabla 5.7, Tabla 5.8 y Tabla 5.9.

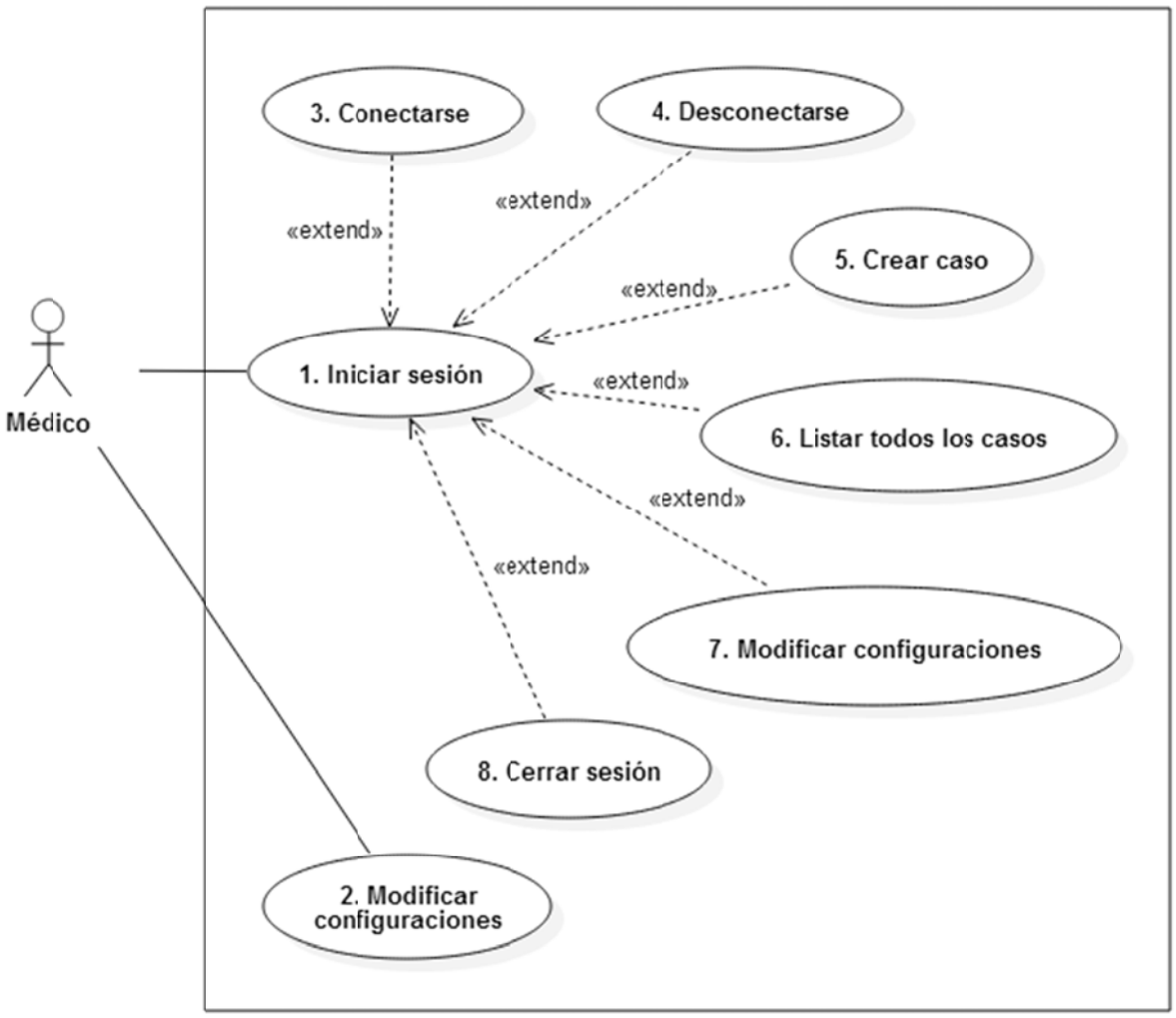


Figura 5.2: Caso de uso nivel 1 de la aplicación.

Tabla 5.2: Descripción de caso de uso "1. Iniciar Sesión".

Caso de uso	CU.1. Iniciar sesión
Actores	ACT.1 Médico
Descripción	Este caso de uso permite al actor ingresar sus credenciales para hacer uso de la aplicación.
Precondición	Ninguna.
Flujo básico	1. Se muestra una pantalla donde se solicitan las credenciales, usuario y contraseña. 2. El usuario ingresa el usuario y contraseña. 3. Las credenciales son correctas y se muestra la pantalla principal de la aplicación.
Flujos alternos	• La dirección del servidor web es incorrecta. Al momento de iniciar sesión no es posible comunicarse con el servicio web. Se mantiene presente la misma pantalla de solicitud de credenciales para el inicio de sesión. • Las credenciales son incorrectas. Se mantiene presente la misma pantalla de solicitud de credenciales para el inicio de sesión.
Postcondición	El actor tiene acceso a las opciones principales de la aplicación.

Tabla 5.3: Descripción de caso de uso "2. Modificar configuraciones".

Caso de uso	CU.2. Modificar configuraciones
Actores	ACT.1 Médico
Descripción	Este caso de uso permite al usuario configurar la dirección del servidor web donde se encuentra alojado el servicio web (*/sos_triaje*) para el inicio de sesión.
Precondición	Ninguna.
Flujo básico	1. El usuario ingresa la dirección del servidor web. 2. Guarda los cambios.
Flujos alternos	Ninguno.
Postcondición	La aplicación podrá comunicarse con el servidor web y verificar las credenciales del usuario para el inicio de sesión.

Tabla 5.4: Descripción de caso de uso "3. Conectarse".

Caso de uso	CU.3. Conectarse
Actores	ACT.1 Médico
Descripción	Este caso de uso permite al usuario iniciar sesión en el servidor de VoIP.
Precondición	Que el usuario haya iniciado sesión de manera exitosa.
Flujo básico	1. El usuario selecciona la opción de conectarse. 2. Se muestra un indicador (punto verde) para notificar al usuario que se realizó de forma exitosa la conexión al servidor de VoIP.
Flujos alternos	La configuración del servidor VoIP es incorrecta. No se conecta con el servidor de VoIP
Postcondición	El usuario podrá hacer uso del servicio de VoIP y se encontrará visible para que otros usuarios puedan contactarlo.

Tabla 5.5: Descripción de caso de uso "4. Desconectarse".

Caso de uso	CU.4. Desconectarse
Actores	ACT.1 Médico
Descripción	Este caso de uso permite al usuario cerrar sesión en el servidor de VoIP.
Precondición	• Que el usuario haya iniciado sesión de manera exitosa. • Que el usuario se encuentre conectado al servidor de VoIP.
Flujo básico	1. El usuario selecciona la opción de desconectarse. 2. Se muestra un indicador (punto rojo) para notificar al usuario que se realizó de forma exitosa la desconexión al servidor de VoIP.
Flujos alternos	Ninguno.
Postcondición	El usuario ya no estará conectado al servidor de VoIP y no será visible para los otros usuarios.

Tabla 5.6: Descripción de caso de uso "5. Crear caso".

Caso de uso	CU.5. Crear caso
Actores	ACT.1 Médico
Descripción	Este caso de uso permite al usuario crear un caso nuevo en el sistema.
Precondición	Que el usuario haya iniciado sesión de manera exitosa.
Flujo básico	1. El usuario selecciona la opción de crear un nuevo caso. 2. Se muestra una pantalla con un formulario para capturar los datos del caso. 3. El usuario completa los campos solicitados. 4. Se guarda el caso.
Flujos alternos	Falta algún campo obligatorio. Se muestra una notificación y se mantiene en la pantalla con el formulario.
Postcondición	El caso se encontrará almacenado en el sistema y podrá recibir opiniones de otros especialistas.

Tabla 5.7: Descripción de caso de uso "6. Listar todos los casos".

Caso de uso	CU. 6. Listar todos los casos
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario ver todos los casos que se encuentran creados en el sistema.
Precondición	Que el usuario haya iniciado sesión de manera exitosa.
Flujo básico	1. El usuario selecciona la opción de ver todos los casos. 2. Se muestra una lista con todos los casos del sistema.
Flujos alternos	Ninguno.
Postcondición	El usuario podrá seleccionar un caso para ver la información del caso y las opiniones que pueda tener.

Tabla 5.8: Descripción de caso de uso "7. Modificar configuraciones".

Caso de uso	CU. 7. Modificar configuraciones
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario configurar los parámetros de identidad y red para la conexión con el servidor de VoIP.
Precondición	Que el usuario haya iniciado sesión de manera exitosa.
Flujo básico	1. El usuario selecciona la opción de configuraciones. 2. Cambia los valores necesarios. 3. Se guardan automáticamente dichos cambios.
Flujos alternos	Ninguno.
Postcondición	Se guardan los cambios realizados y estos serán los nuevos parámetros para la conexión con el servidor de VoIP.

Tabla 5.9: Descripción de caso de uso "8. Cerrar sesión".

Caso de uso	CU. 8. Cerrar sesión
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario
Precondición	Que el usuario haya iniciado sesión de manera exitosa.
Flujo básico	1. El usuario selecciona la opción cerrar sesión. 2. Se cierra la sesión y se muestra la pantalla principal de la aplicación.
Flujos alternos	Ninguno.
Postcondición	Se muestra la pantalla principal de la aplicación donde se solicitan las credenciales, usuario y contraseña.

5.2.3 Casos de uso nivel 2

En este nivel de caso de uso se muestran las opciones que posee el usuario para realizar sobre un caso, estas opciones pueden observarse en el diagrama de caso de uso mostrado en la Figura 5.3, las descripciones de los casos de uso de dicha figura se encuentran en las siguientes tablas: Tabla 5.10, Tabla 5.11, Tabla 5.12, Tabla 5.13, Tabla 5.14 y Tabla 5.15.

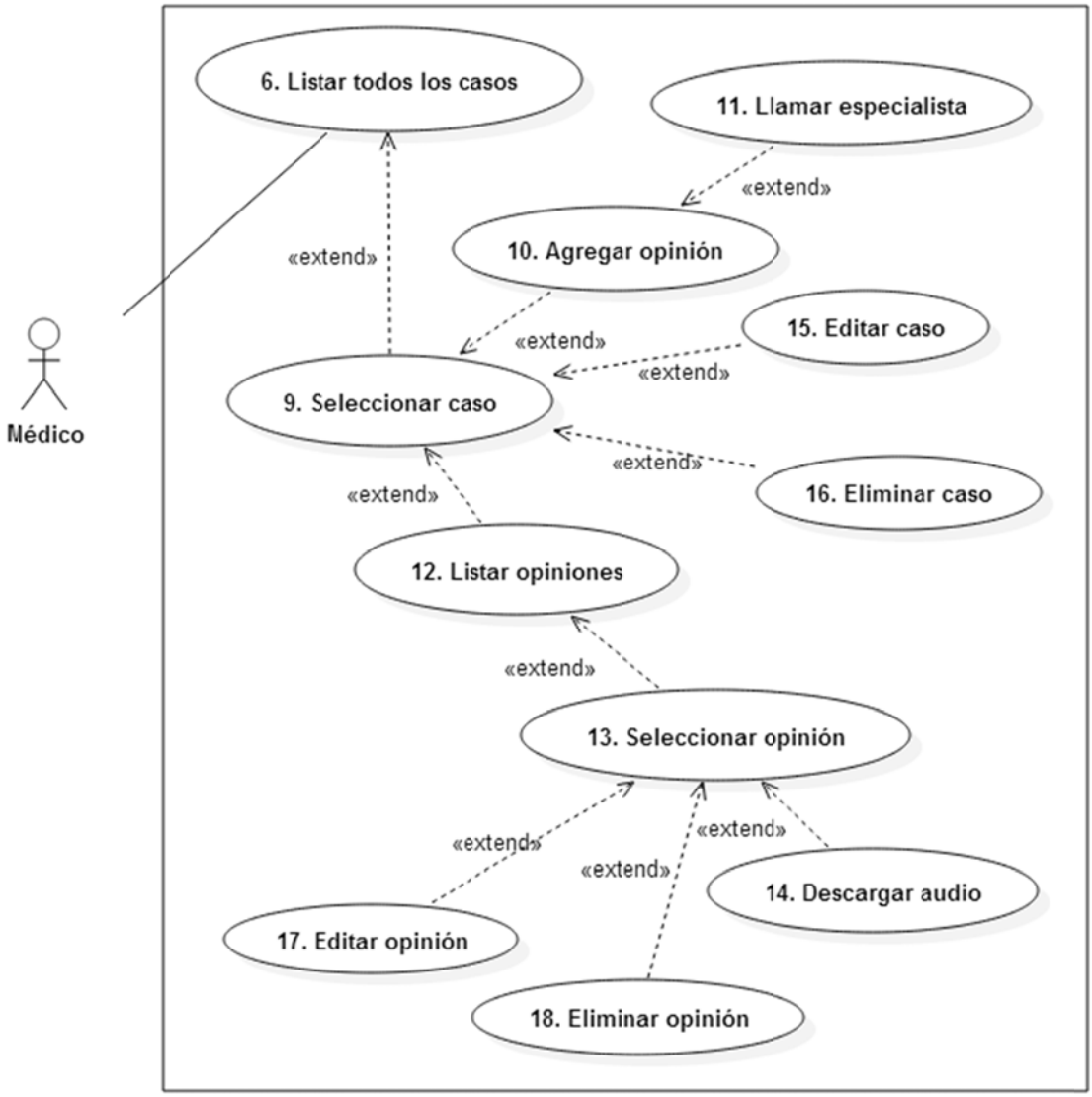


Figura 5.3: Caso de uso nivel 2 de la aplicación.

Tabla 5.10: Descripción de caso de uso "9. Seleccionar caso".

Caso de uso	CU. 9. Seleccionar caso
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario seleccionar y ver la información referente a un caso.
Precondición	Haber listado todos los casos.
Flujo básico	1. El usuario selecciona un caso de la lista. 2. Se muestra la información detallada del caso.
Flujos alternos	Ninguno.
Postcondición	El usuario verá la información referente a un caso y podrá agregar una nueva opinión o listar las opiniones que tiene el caso.

Tabla 5.11: Descripción de caso de uso "10. Agregar opinión".

Caso de uso	CU. 10. Agregar opinión
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario agregar una nueva opinión a un caso.
Precondición	Que el usuario haya seleccionado un caso.
Flujo básico	1. El usuario selecciona agregar opinión. 2. Se muestra una pantalla con un formulario. 3. Se completan los campos. 4. Se guarda la opinión el en caso.
Flujos alternos	Ninguno.
Postcondición	La opinión se encontrará almacenada en el sistema asociada al caso.

Tabla 5.12: Descripción de caso de uso "11. Llamar especialista".

Caso de uso	CU. 11. Llamar especialista
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario llamar a un especialista que se encuentre disponible para realizar una consulta de segunda opinión referente al caso.
Precondición	• Que el usuario haya seleccionado un caso. • Que el usuario se encuentre creando una opinión referente al caso seleccionado.
Flujo básico	1. El usuario selecciona la opción de “Llamar especialista” desde la pantalla de “crear opinión”. 2. Se listan las especialidades que existen en el sistema. 3. Se escoge una especialidad con relación al caso. 4. Se realiza la llamada a ese conjunto de usuarios que pertenezcan a ese grupo (Especialidad). 5. Se realiza la videollamada.
Flujos alternos	Ninguno.
Postcondición	Si la videollamada fue grabada se buscará en la base de datos de Asterisk el ID de la última llamada para almacenarla junto con la opinión y de esta forma poder obtener el audio en un futuro.

Tabla 5.13: Descripción de caso de uso "12. Listar opiniones".

Caso de uso	CU. 12. Listar opiniones
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario ver todas las opiniones que se encuentran creados en un caso.
Precondición	Que el usuario haya seleccionado un caso.
Flujo básico	1. El usuario selecciona la opción de ver todas las opiniones. 2. Se muestra una lista con todas las opiniones de ese caso.
Flujos alternos	Ninguno.
Postcondición	El usuario podrá seleccionar una opinión para ver información más detallada de la opinión.

Tabla 5.14: Descripción de caso de uso "13. Seleccionar opinión".

Caso de uso	CU. 13. Seleccionar opinión
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario seleccionar y ver la información de una opinión referente a un caso.
Precondición	• Que el usuario haya seleccionado un caso. • Haber listado todas las opiniones de ese caso.
Flujo básico	1. El usuario selecciona una opinión de la lista. 2. Se muestra la información detallada de la opinión.
Flujos alternos	Ninguno.
Postcondición	El usuario verá la información de una opinión referente a un caso y si existe una grabación asociada podrá descargarlo.

Tabla 5.15: Descripción de caso de uso "14. Descargar audio".

Caso de uso	CU. 14. Descargar audio
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario descargar el audio asociado a la opinión.
Precondición	Que el usuario haya seleccionado una opinión.
Flujo básico	1. El usuario selecciona una opinión. 2. Se muestra la información de la opinión. 3. El usuario selecciona la opción de descargar audio. 4. Se abre el enlace en un navegador para su descarga.
Flujos alternos	Ninguno.
Postcondición	Se obtiene el archivo de audio relacionado con la opinión.

Tabla 5.16: Descripción de caso de uso "15. Editar caso".

Caso de uso	CU. 15. Editar caso
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario editar la información de un caso seleccionado.
Precondición	Que el usuario haya seleccionado un caso.
Flujo básico	1. El usuario selecciona un caso. 2. El usuario presiona el botón editar. 3. El usuario edita el caso. 4. El usuario presiona el botón guardar.
Flujos alternos	Ninguno.
Postcondición	Se muestra la pantalla del caso editado.

Tabla 5.17: Descripción de caso de uso "16. Eliminar caso".

Caso de uso	CU. 16. Eliminar caso
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario eliminar un caso seleccionado.
Precondición	Que el usuario haya seleccionado un caso.
Flujo básico	1. El usuario selecciona un caso. 2. El usuario presiona el botón eliminar.
Flujos alternos	Ninguno.
Postcondición	Se muestra la pantalla de lista de casos.

Tabla 5.18: Descripción de caso de uso "17. Editar opinión".

Caso de uso	CU. 17. Editar Opinión
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario editar la información de una opinión seleccionada.
Precondición	Que el usuario haya seleccionado una opinión.
Flujo básico	1. El usuario selecciona una opinión. 2. El usuario presiona el botón editar. 3. El usuario edita la opinión. 4. El usuario presiona el botón guardar.
Flujos alternos	Ninguno.
Postcondición	Se muestra la pantalla de la opinión editada.

Tabla 5.19: Descripción de caso de uso "18. Eliminar opinión".

Caso de uso	CU. 18. Eliminar opinión
Actores	ACT. 1 Médico
Descripción	Este caso de uso permite al usuario eliminar una opinión seleccionada.
Precondición	Que el usuario haya seleccionado una opinión.
Flujo básico	1. El usuario selecciona una opinión. 2. El usuario presiona el botón eliminar.
Flujos alternos	Ninguno.
Postcondición	Se muestra la pantalla de lista de opiniones.

5.2.4 Diagrama de objetos

En la Figura 5.4 se presenta el diagrama de objetos de la aplicación, en este se muestran las extensiones entre los diferentes objetos y las relaciones principales entre ellos. El objeto ***GlobalBroadCastReceiver*** es el encargado de interceptar los intentos de tipo *broadcast* dentro de la aplicación principalmente para manejar las llamadas realizadas. El objeto ***NativeService*** presenta los servicios necesarios para la aplicación mientras que desde el objeto ***Engine*** se obtiene el acceso a estos.

El objeto ***BaseScreen*** representa la base para todas las pantallas e implementa la interfaz ***IBaseScreen***. Las pantallas son manejadas desde un ***ActityGroup Main***.

El objeto ***UserData*** representa toda la información que se necesita del usuario, una parte es almacenada localmente usando una base de datos SQLite manejada con el objeto ***DatabaseHandler*** y otro conjunto es obtenido del servicio web mediante los métodos que posee el objeto ***ServiceHTTP***. A su vez este último es el encargado de realizar las consultas al servidor para recuperar y almacenar la información de los casos médicos ***medicCase*** y opiniones ***Opinion*** desde y hacia la base de datos de SOS Telemedicina.

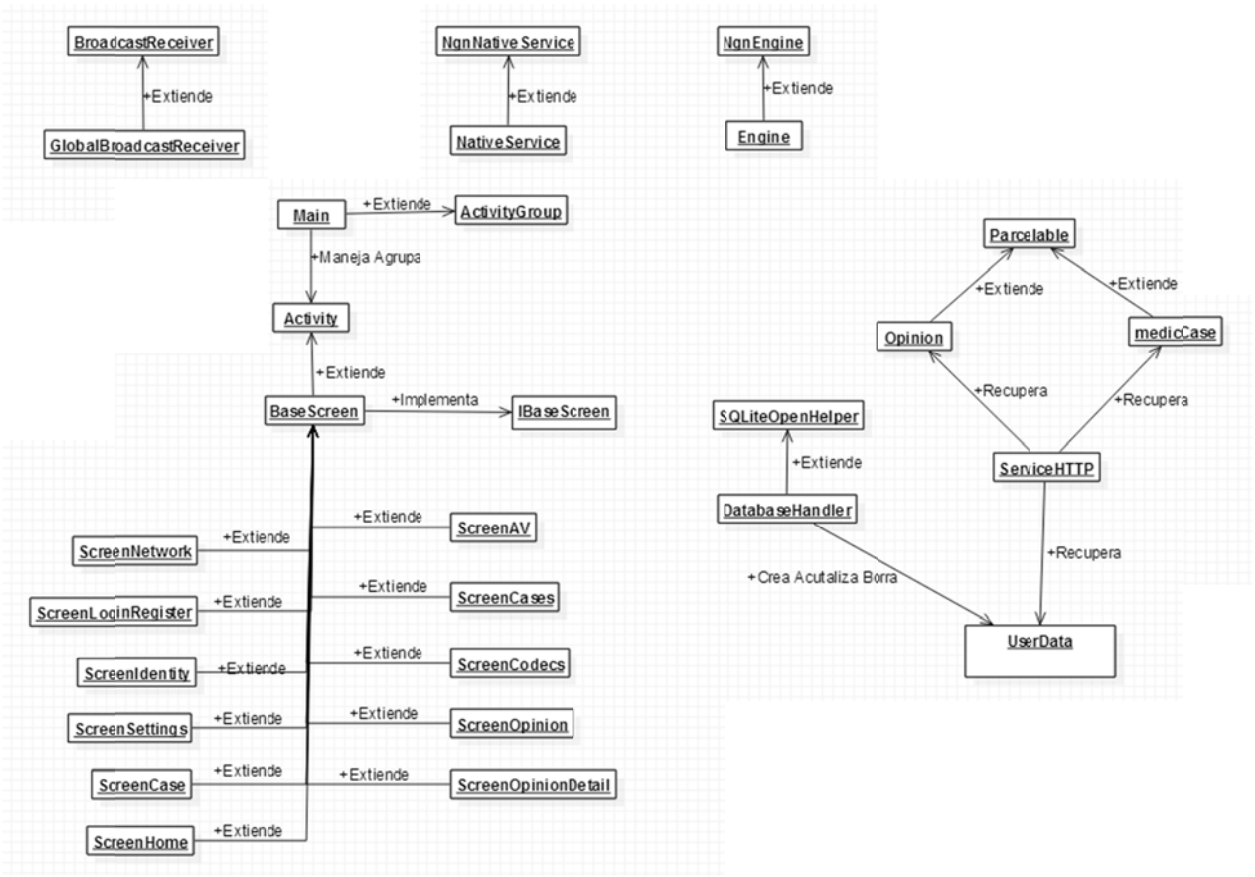


Figura 5.4: Diagrama de Objetos de la aplicación.

5.3 Modificaciones a la base de datos de SOS Telemedicina - sos_triaje

Dada la existencia de una base de datos para el sistema de segunda opinión médica, (Base de Datos SOS Telemedicina, *sos_triaje)*, se analizó la estructura de sus tablas y relaciones para comprender la organización de la data y como obtener la información de ella ya que en la documentación no se encuentra esta información técnica. En la Figura 5.5 se muestra el diagrama de Entidad-Relación de la base de datos de SOS Telemedicina.

Con el análisis realizado se crearon las consultas a la base de datos que ofrece el servicio web, a su vez se determinaron varias debilidades tanto en el diseño como en su estructura, realizar los ajustes necesarios para solventar dichas debilidades escapa de los objetivos de este trabajo, es por ello que las modificaciones realizadas en la base de datos fueron hechas de tal forma que no afecte el desarrollo del sistema original.

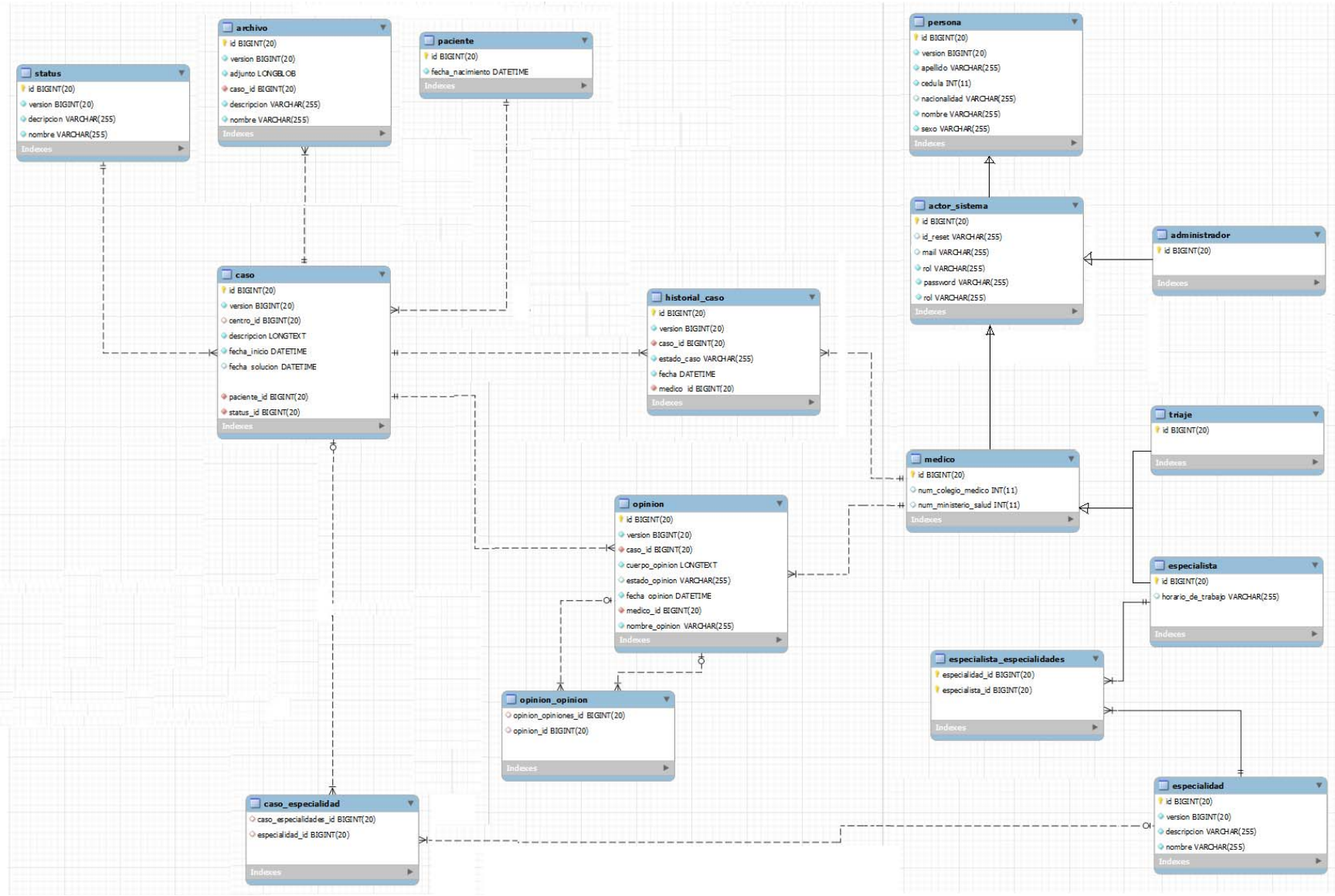


Figura 5.5: Diagrama de entidad-relación original del sistema SOS Telemedicina.

Las tablas de la base de datos de SOS Telemedicina que fueron modificadas son las siguientes:

- **actor_sistema**:
    - Se agrega la columna ***api_key***, este contendrá un hash MD5 para autenticar las peticiones del cliente con el servicio web, si se encuentra en NULL el servicio web creará un valor para ese usuario.
    - Se agrega la columna ***user_extension***, este campo almacenará el número de extensión que posee el usuario en Elastix.
- **archivo**: se agrega la columna ***mime_type***, esto es para identificar el tipo de archivo binario que se encuentra almacenado.
- **caso**: se agrega la columna ***FK_actor_sistema,*** esto es para identificar el usuario que creó el caso en el sistema, de esta forma ese usuario sólo podrá editar y eliminar casos creados por él.
- **especialidad**: se agrega la columna ***group_extension***, al igual que ***user_extension*** este campo almacenará el número de extensión que poseen en FreePBX para el grupo.
- **opinión**:
    - Se agrega la columna ***cdr_uniqueid***, en esta columna se almacenará el identificador univoco que generó la llamada.
    - Se agrega la columna ***calldate***, esta columna almacenará la fecha cuando fue realizada la grabación.
    - Se agrega la columna ***recordingfile***, esta columna almacenará el nombre del archivo de audio.
- **paciente**: se modifica la columna ***id*** para que tenga la propiedad de AUTO_INCREMENT esto con la finalidad de que la base de datos seleccione un ID automáticamente en lugar de ser asignado por la aplicación al momento de crear un registro en esta tabla. Es necesario haber eliminado todas las referencias (claves foráneas) en la base de datos a esta tabla para poder aplicar esta propiedad.

En el Apéndice A (Figura 9.1, Figura 9.2 y Figura 9.3) se encuentran los scripts SQL que realizan estos cambios para la base de datos de SOS Telemedicina.

5.4 Diseño del servicio web

El servicio web ***sos_triaje*** está diseñado bajo la arquitectura REST, es un API el cual permite realizar operaciones tanto a la base de datos de SOS Telemedicina como a las de

Elastix mediante solicitudes HTTP, estas solicitudes pueden ser de tipo ***POST***, ***GET***, ***PUT*** o ***DELETE*** y todas las respuestas poseen un formato genérico codificado en ***JSON***. Al iniciar sesión de forma exitosa se obtiene en la respuesta HTTP un ***API Key*** el cual es un *hash MD5* necesario para poder realizar operaciones al servicio web que requieran autenticación. Para aquellas solicitudes al API que requieran autenticación es necesario enviar en la cabecera de la solicitud HTTP el campo ***Authorization*** con el valor del API Key obtenido en el inicio de sesión, de lo contrario la solicitud será negada.

5.4.1 Estructura de directorios

Este servicio web se conforma de una serie de directorios donde cada uno agrupa elementos en común, dicha distribución puede observarse en la Figura 5.6, a continuación se describe el propósito de cada directorio.

Figura 5.6: Estructura de directorios del servicio web

- **/sos_triaje**: este es el directorio raíz el cual aloja el servicio web.
- **/sos_triaje/protected**: es un directorio protegido donde no se permite el acceso y es usado para almacenar los módulos del servicio web.
- **/sos_triaje/protected/API**: en este directorio se almacena código referente al API así como la definición de las rutas y las funciones que utiliza.
- **/sos_triaje/protected/clases**: este directorio contiene la definición de las clases utilizadas en el servicio web.
- **/sos_triaje/protected/consts**: este directorio contiene la definición de las constantes utilizadas en el servicio web.

- **/sos_triaje/protected/controllers**: este directorio contiene los controladores de recursos, estos se encargan de las operaciones que se puedan realizar (creación, lectura, actualización y eliminación).

- **/sos_triaje/protected/docs**: este directorio contiene archivos de documentación referente al servicio web.

- **/sos_triaje/protected/libs**: este directorio es usado para colocar códigos de terceros así como plugins y/o *frameworks* de ayuda para el servicio web.

- **/sos_triaje/protected/.htaccess**: este archivo es el que da la propiedad de protegido al directorio actual, negando el acceso a cualquiera (*deny from all*).

- **/sos_triaje/.htaccess**: este archivo contiene las configuraciones que requiera el servicio web.

- **/sos_triaje/index.php**: este archivo es la entrada principal y única del servicio web.

5.4.2 Formato de respuesta

Las respuestas obtenidas por el API se encuentran codificadas en formato JSON y poseen una estructura genérica que consta de dos bloques, ***metadata*** y ***data***. En el bloque de metadata se encontrará la información relacionada con la consulta y en el bloque data se encontrará la información solicitada. Dentro del bloque de metadata se encuentra:

- **errorCode**: un valor numérico que indica si la solicitud fue realizada con éxito o no. Cero (0) es el valor para indicar que se realizó con éxito la solicitud; Uno (1) para indicar que no pudo ser realizada la solicitud, cuando aparece este valor estará presente el campo ***msgDescription*** en el bloque de data ofreciendo una descripción detallada de la causa del error como se observa en la Figura 5.7.

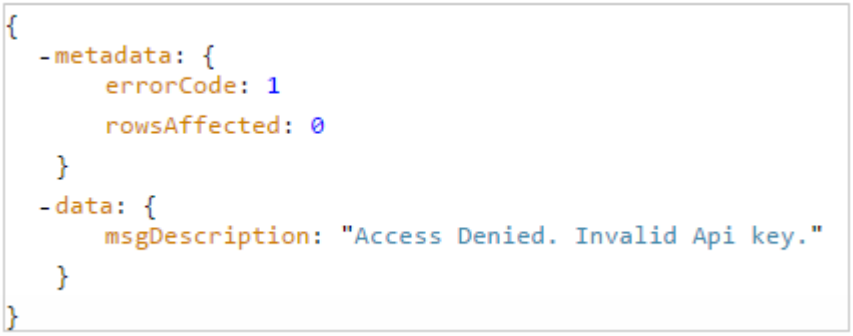


Figura 5.7: Respuesta del API de una solicitud no procesada.

- **rowsAffected**: indica la cantidad de filas afectadas por la solicitud realizada.

- **debug**: este bloque muestra información adicional acerca de la consulta realizada, para recibir este bloque en el API el valor de la variable global ***DEBUG_MODE*** debe estar en TRUE (*index.php*), la Figura 5.8 muestra una respuesta del API con el bloque debug presente, este bloque posee tres campos:
    - **errorMessage**: mensaje correspondiente al valor del campo ***errorCode***.
    - **requestMethod**: muestra el tipo de solicitud HTTP que fue realizado (*POST*, *GET*, *PUT* o *DELETE*).
    - **queryString**: el query que fue ejecutado para dicha solicitud.

Figura 5.8: Respuesta del API de una solicitud procesada con información adicional (Debug).

5.4.3 Documentación del API

En la Tabla 5.20, Tabla 5.21, Tabla 5.22, Tabla 5.23 y Tabla 5.24 se muestran la documentación del API ***sos_triaje***, estas se encuentran agrupadas por el tipo de solicitud HTTP (POST, GET, PUT y DELETE).

.

- Todos los métodos requieren autenticación exceptuando aquellos que indiquen lo contrario.
- Si el método requiere autenticación se debe enviar en la cabecera de la solicitud HTTP el ***API Key*** (Campo: *Authorization*).
- Los parámetros se envían en el cuerpo de la solicitud HTTP.
- Los filtros son parámetros (*query string*) que se agregan en el Localizador Uniforme de Recurso (*Uniform Resource Locator*, URL) de la solicitud. Todos los métodos de lectura (*/GET*) poseen dos filtros: ***limit*** y ***offset***, *limit* indica la máxima cantidad de registros a obtener en la consulta y *offset* es el desplazamiento para indicar a partir de donde se tomarán los registros, este último se usa en conjunto con *limit*.

Tabla 5.20: Documentación del API del servicio web sos_triaje.

URL	Parámetros	Filtros	Descripción
/POST (*Create*)			
/login *(No requiere autenticación)*	• **user**: correo o login del usuario en el sistema. • **password**: contraseña del usuario en el sistema.	-	Verifica si el usuario está registrado en el sistema y de ser así retorna datos del usuario incluyendo el ***API Key*** el cual es necesario para realizar operaciones que requieran autenticación.
/casos	• **versión** *(opcional, default = versión del API)* • **centro_id** *(opcional)* • **descripcion** • **fecha_solucion** *(opcional)*	-	Crea un caso.

Tabla 5.21: Continuación de la Tabla 5.20.

URL	Parámetros	Filtros	Descripción
/POST (*Create*)			
	• **id_casosos** *(opcional)* • **paciente_id** *(opcional)* • **status_id** *(opcional, default = 1, "En espera")* • **especialidad_id** *(opcional)* • **fecha_nacimiento**	-	
/casos/:casoId/opiniones	• **versión** *(opcional, default = versión del API)* • **cuerpo_opinion** • **estado_opinion** *(opcional, default = "Creado por videollamada")* • **nombre_opinion** • **user_extension**	-	Crea una opinión a un caso.
/pacientes	• **fecha_nacimiento**	-	Crea un paciente.
/GET (*Read*)			
/especialidades	-	-	Retorna el conjunto de especialidades que existen en el sistema.

Tabla 5.22: Continuación de la Tabla 5.21.

URL	Parámetros	Filtros	Descripción
/GET (*Read*)			
/centro_sos	-	-	Retorna el conjunto de centros SOS que existen en el sistema.
/centro_sos/:centro_id	-	-	Retorna los detalles de un centro SOS.
/casos	-	• **own**=<*true o false*> Filtra los casos que tenga un historial con la persona logueada. • **especialidad**=<*ID*> Filtra los casos que pertenezcan a una especialidad. • **status**=<*ID*> Filtra los casos por estatus. • **centro**=<*ID*> Filtra los casos que pertenezcan a un centro SOS en específico.	Retorna el conjunto de casos que existen en el sistema.
/casos/:casoId	-	-	Retorna los detalles de un caso.

Tabla 5.23: Continuación de la Tabla 5.22.

URL	Parámetros	Filtros	Descripción
/GET (*Read*)			
/casos/:casoId/historial	-	-	Retorna el historial de un caso.
/casos/:casoId/archivos	-	-	Retorna los archivos relacionados a un caso.
/casos/:casoId/archivos/:archivoId	-	-	Retorna un archivo en formato blob.
/casos/:casoId/opiniones	-	-	Retorna las opiniones de un caso.
/casos/:casoId/opiniones/:opinionId	-	-	Retorna los detalles de una opinión asociado al caso.
/pacientes	-	-	Retorna el conjunto de pacientes que existen en el sistema.
/pacientes/:paciente_id	-	-	Retorna información de un paciente.
/call_groups	-	-	Retorna una lista de grupos de llamada (*Ring Groups*).

Tabla 5.24: Continuación de la Tabla 5.23.

URL	Parámetros	Filtros	Descripción
/PUT (*Update*)			
/casos/:casoId	• **versión** *(opcional)* • **centro_id** *(opcional)* • **descripcion** *(opcional)* • **fecha_solucion** *(opcional)* • **id_casosos** *(opcional)* • **status_id** *(opcional)* ***Nota**: debe enviarse al menos un parámetro.*	-	Modifica un caso.
/casos/:casoId/opiniones/:opinionId	• **versión** *(opcional)* • **cuerpo_opinion** *(opcional)* • **estado_opinion** *(opcional)* • **nombre_opinion** *(opcional)* ***Nota**: debe enviarse al menos un parámetro.*	-	Modifica una opinión asociada a un caso.
/DELETE (*Delete*)			
/casos/:casoId	-	-	Elimina un caso.
/casos/:casoId/opiniones/:opinionId	-	-	Elimina una opinión asociada a un caso.

6 Implementación del sistema

Para el desarrollo y las pruebas se implementó una configuración compuesta por computadores personales de escritorio en donde se virtualizaron los servidores utilizando VirtualBox y para la red inalámbrica se utilizaron enrutadores simples de uso en oficinas o en el hogar.

Cuando el desarrollo culminó, se realizó la implementación sobre los servidores de SOS Telemedicina para lo cual se creó en estos una máquina virtual Xen con los requerimientos de hardware necesarios como se muestra en la Tabla 6.1. En un principio se intentó la instalación de Elastix sobre Debian como el sistema operativo por defecto en las máquinas virtuales Xen del servidor de SOS Telemedicina pero se encontraron problemas de incompatibilidad respecto al sistema operativo Debian y Elastix, esto, sumado a la dificultad de reproducir la instalación nuevamente sobre éste sistema operativo, hizo que se buscara otra solución la cuál fue realizar la instalación de la central PBX Asterisk y FreePBX como interfaz web para su administración.

Tabla 6.1: Requerimientos mínimos de hardware.

Sistema Operativo	Debian 6.0 (*Squeeze*)
Memoria RAM	1 GB
Espacio libre en disco duro	10 GB

6.1 Arquitectura del sistema

El sistema se desarrolló para funcionar en paralelo a la implementación existente del sistema de segunda opinión médica de SOS Telemedicina el cual comprende cuatro elementos principales que son: la aplicación de videollamada de Segunda Opinión Médica sobre Android, el servicio web que permite el acceso a la base de datos, el servidor de llamadas Asterisk y la base de datos del Sistema de Segunda Opinión Médica de SOS Telemedicina. En la Figura 6.1 se muestra la arquitectura del sistema final.

La aplicación de videollamada de Segunda Opinión Médica en Android es el punto con el que el usuario interactúa con el sistema. Ésta se conecta y autentica contra la central Asterisk IP-PBX y el servicio web, permite realizar todas las actividades necesarias para crear y consultar casos y opiniones médicas además de realizar llamadas a los especialistas.

El servicio web que se implementó para ésta solución se encarga de atender las consultas a la base de datos de SOS Telemedicina. La aplicación debe autenticarse contra este servidor y recibir una clave para poder realizar consultas adicionales luego de la autenticación.

El servidor de llamadas Asterisk IP-PBX mantiene las sesiones de cada usuario una vez que se autentican exitosamente. Su rol es establecer la comunicación entre dos usuarios mediante el protocolo SIP y luego mantener la comunicación y transmisión de audio y video a través de la red entre dos usuarios, adicionalmente permite almacenar el audio de cada llamada para futuras consultas. Para este sistema se instaló en conjunto con una interfaz web que facilita su configuración y administración.

Por último la base de datos de SOS Telemedicina contiene toda la data de usuarios, casos y opiniones necesarias. Ésta base de datos tuvo que ser modificada para lograr ciertas funcionalidades, estas modificaciones se hicieron de manera tal que no afectara el funcionamiento del sistema original.

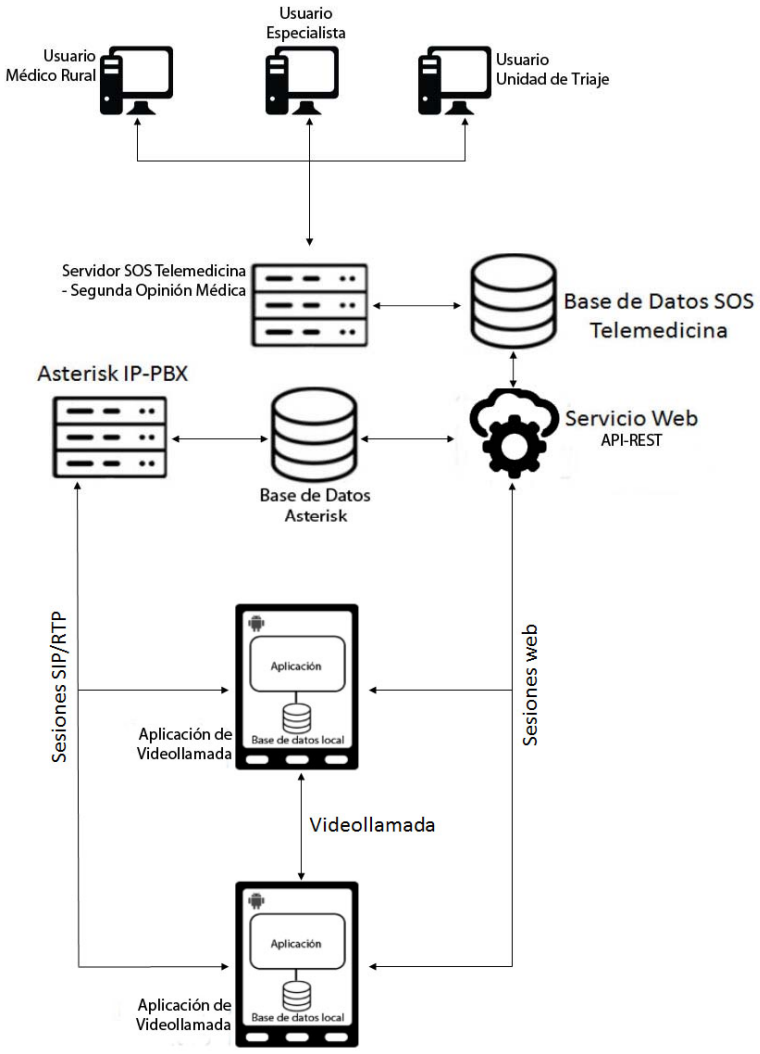


Figura 6.1: Arquitectura del sistema de videollamada.

6.2 Configuración de Asterisk + FreePBX

En esta sección se explicará cómo configurar Asterisk mediante la interfaz de FreePBX para tener un ambiente básico de videollamada, se explicará cómo permitir el acceso del servicio web, habilitar es soporte de video, crear extensiones, crear grupos de llamadas o *Ring Groups.* Para mayor información puede consultar la documentación oficial de FreePBX [37].

6.2.1 Permitir el acceso del servicio web

Es necesario realizar ciertas configuraciones para la conectividad del servicio web con el servidor Asterisk ya que el mismo se encuentra configurado para que el acceso sea únicamente local, es decir por medio de la interfaz web de FreePBX.

El servicio web realiza consultas (sólo lectura) a dos bases de datos de Elastix ***asteriskcdrdb*** y ***asterisk***. Realizando una consulta a la base de datos ***asteriskcdrdb*** se puede obtener el último registro de llamada realizada por el usuario, si esa llamada fue grabada el valor del ***uniqueid*** será asociado a la opinión en la base de datos de SOS Telemedicina sí y sólo sí el valor no se encuentra asociado ya a otra opinión. La consulta a la base de datos ***asterisk*** es para obtener el listado de los grupos de llamadas existentes en el sistema (*Ring Groups*), así la aplicación obtiene las extensiones de los grupos y de esta forma poder realizar las llamadas.

Para permitir la conexión de otro servidor al servidor de MySQL de FreePBX se debe editar el archivo que se encuentra en "***/etc/mysql/conf.d/mysqld_safe_syslog.cnf***" y agregar las líneas que se muestran en la siguiente figura.

```
# ...
[mysqld]
bind-address = 0.0.0.0
```

Figura 6.2: Modificación para permitir conexiones externas a MySQL.

Una vez hecho este cambio, se procede a ejecutar comandos para permitir que el usuario "***root***" con contraseña "***123456***" pueda tener acceso a la BD como se indica en la siguiente figura.

```
# mysql -u root -e "GRANT ALL PRIVILEGES ON *.* TO 'root'@'%'
IDENTIFIED BY '123456'; "
# mysql -u root -e "FLUSH PRIVILEGES;"
# service mysql restart
```

Figura 6.3: Comandos para dar acceso al usuario root a la BD de FreePBX.

6.2.2 Obtener las grabaciones vía servidor web

Para que la aplicación pueda recuperar las grabaciones de las opiniones vía el servidor web, es necesario crear un enlace simbólico en el directorio home del servidor web que apunte al directorio donde se encuentre estas grabaciones *(/var/spool/asterisk/monitor*), en la siguiente figura se muestran los comandos para realizar dicha operación.

```
# cd /var/www/html/
# ln -s /var/spool/asterisk/monitor monitor
```

Figura 6.4: Comandos para obtener las grabaciones via servidor web.

6.2.3 Habilitar el soporte de video

Asterisk por defecto no posee el soporte de video habilitado, es por esto que es necesario seguir los siguientes pasos para poder realizar videollamadas:

- Ingrese a la interfaz de administración de FreePBX.
- Seleccione "**Settings**" > "**Asterisk SIP Settings**" en el menú principal.
- En la sección de "**Video Codecs**" haga click en "**Enabled**" para la opción "**Video Support**".
- Seleccione los códecs que desea utilizar.
- Ir al final de la página y hacer click en "**Submit Changes**".
- Por último seleccione la opción "**Apply Config**" que se encuentra en el menú principal, esto reiniciará los servicios y una vez finalizado los cambios se encontrarán aplicados.

6.2.4 Crear una extensión

Una extensión es un usuario dentro del sistema de IP-PBX, para crear un usuario siga los siguientes pasos:

- Ingrese a la interfaz de administración de FreePBX.
- Seleccione "**Applications**" > "**Extensions**" en el menú principal.
- Una vez en la pantalla de "**Add an Extension**" seleccione la opción "**Generic SIP Device**" y haga click en "**Submit**".
- Llene los siguientes campos:
    - **User Extension**: numero para identificar a este usuario.

- **Display Name**: nombre a mostrar para este usuario.
- **Secret**: contraseña para este usuario.
- **Nat**: seleccione yes para habilitar el soporte de NAT a este usuario.
- **Recordings Options**: seleccionar todas en “Always”.
- **iSymphony Settings**: si se encuentra configurado el modulo, desactivar todas las casillas seleccionando “No”.

- Ir al final de la página y hacer click en “**Submit**”.
- Por último seleccione la opción “**Apply Config**” que se encuentra en el menú principal, esto reiniciará los servicios y una vez finalizado los cambios se encontrarán aplicados.

6.2.5 Crear una cuenta de usuario

En FreePBX es posible crear usuarios que puedan ingresar al módulo de administración pero con acceso limitado, en el siguiente ejemplo se creará un usuario que puede ingresar al módulo de administración y consultar las grabaciones del sistema.

- Ingrese a la interfaz de administración de FreePBX.
- Seleccione “**Admin**” > “**Administrators**” en el menú principal.
- Llene los siguientes campos:
  - **Username**: nombre de usuario.
  - **Password**: contraseña del usuario.
  - **Admin Access**: seleccione “**CDR Reports**”.
- Ir al final de la página y hacer click en “**Submit Changes**”.
- Por último seleccione la opción “**Apply Config**” que se encuentra en el menú principal, esto reiniciará los servicios y una vez finalizado los cambios se encontrarán aplicados.

6.2.6 Crear un grupo de llamada (ring groups)

En Asterisk es posible configurar un número de extensión en el cual un grupo de usuarios puedan pertenecer, a esto se le denomina *Ring Groups*, de esta forma cualquiera de los usuarios puede atender la llamada que se realice al grupo. Esta configuración es apropiada para la agrupación de médicos especialistas según sea su especialidad. Para crear un *Ring Group* en la interfaz de FreePBX debe seguir los siguientes pasos:

- Ingrese a la interfaz de administración de FreePBX.
- Seleccione “**Applications**” > “**Ring Groups**”.

- Llene los siguientes campos:
    - **Ring-Group Number**: el número de extensión asociado al *Ring Group*.
    - **Group Description**: descripción del *Ring Group*, es decir, especialidad.
    - **Ring Strategy**: seleccionar una política para la notificación a los usuarios del grupo.
    - **Extension List**: lista de usuarios que pertenecerán a este *Ring Group*.
    - **Skip Busy Agent**: marcar checkbox (true) para que no ocurra una autollamada al momento de ser el único usuario disponible.
    - **Record Calls**: marcar "**Always**" para grabar las conversaciones que ocurran con este *Ring Group*.
    - **Destination if no answer**: seleccionar "**Terminate Call**" y "**Hangup**".
- Ir al final de la página y hacer click en "**Submit Changes**".
- Por último seleccione la opción "**Apply Config**" que se encuentra en el menú principal, esto reiniciará los servicios y una vez finalizado los cambios se encontrarán aplicados.

6.3 Configuración del servicio web

Una vez se tenga el servicio web copiado al servidor donde estará alojado es necesario realizar ciertas configuraciones tanto en el servidor como al servicio web para su correcto funcionamiento. A continuación se lista una serie de pasos a seguir con su respectiva explicación.

1. Verificar si se encuentra habilitado el módulo de *rewrite* (***rewrite_module***) de apache, el servidor apache debe ser capaz de interpretar las URL con el formato RESTFULL (que no esté presente ***index.php*** dentro de la URL), por esto es necesario que se encuentre habilitado este módulo, en la siguiente figura se muestran los comandos para realizar esta operación.

```
a2enmod rewrite
service apache2 restart
```

Figura 6.5: Comandos para habilitar el módulo de rewrite de apache.

Verificar que se encuentre habilitado modificaciones por ***.htaccess***, para que el servicio web pueda ejecutar las reglas que se encuentra en el archivo .htaccess es necesario que el servidor apache lo permita, para esto se debe cambiar en el archivo ***/etc/apache2/sites-available/default*** el valor de ***AllowOverride*** de ***None*** a

All tal como se muestra en la siguiente figura y luego reinicia el servicio web con el comando "***service apache2 restart***".

```
<Directory /var/www/html>
     Options Indexes FollowSymLinks MultiViews
     AllowOverride All
     Order allow,deny
     allow from all
</Directory>
```

Figura 6.6: Habilitar modificaciones por .htaccess en el servidor web.

2. Haber ejecutado los script SQL suministrado en la Figura 9.1, Figura 9.2 y Figura 9.3, ya que el servicio web realiza consultas a tablas y campos creados con las modificaciones que se encuentran allí presentes.

3. Establecer los parámetros correspondientes en el archivo de configuración (**db**_***config.php***) para la conexión con la base de datos de segunda opinión médica (***sos_triaje***) y con las bases de datos de Asterisk (***asterisk*** y ***asteriskcdrdb***). En la siguiente figura se muestra un ejemplo de configuración donde la base de datos de SOS Telemedicina se encuentra en el mismo servidor donde se encuentra alojado el servicio web y el servidor Asterisk se encuentra en la dirección 192.168.2.44, ambos con las credenciales ***root@123456***.

```
1. define('SOS_DB_SERVER'     , 'localhost');
2. define('SOS_DB_NAME'       , 'sos_triaje');
3. define('SOS_DB_USER'       , 'root');
4. define('SOS_DB_PASSWORD'   , '123456');
5. define('ELASTIX_DB_SERVER' , '192.168.2.44');
6. define('ELASTIX_DB_NAME'   , 'asterisk');
7. define('ELASTIX_DB_USER'   , 'root');
8. define('ELASTIX_DB_PASSWORD', '123456');
```

Figura 6.7: Parámetros de configuración para conexiones a BD del servicio web.

A continuación se explica el uso de cada variable global del archivo de configuración:

- Línea 1: dirección del servidor de la BD (*SOS Telemedicina*).
- Línea 2: nombre de la BD a conectar.

- Línea 3: usuario de la BD de SOS Telemedicina.
- Línea 4: contraseña de la BD de SOS Telemedicina.
- Línea 5: dirección del servidor de la BD de asterisk.
- Línea 6: nombre de la BD por defecto a conectar (asterisk).
- Línea 7: usuario de la BD de asterisk.
- Línea 8: contraseña de la BD de asterisk.

6.4 Implementación de la aplicación

La aplicación sobre Android se desarrolló en su totalidad usando el lenguaje de programación Java. Las funcionalidades que corresponden al establecimiento de la comunicación con el servidor de llamadas fueron proporcionadas por la biblioteca NGN basada en el *framework* Doubango, siendo éstas el establecimiento de la sesión mediante el protocolo SIP y luego la transferencia y recepción de audio y video mediante el protocolo RTP. Siguiendo el paradigma de aplicación de Android se crearon las *Activities* necesarias y el flujo entre estas para cubrir cada funcionalidad, desde la consulta y la creación de casos y opiniones hasta realizar y recibir llamadas. Para cada *Activity* se programó la lógica necesaria que cubre su funcionalidad.

6.4.1 Uso del *framework* NGN Doubango

Este *framework* permite establecer sesiones y mantener comunicaciones en tiempo real de audio y video sobre Android. El *framework* provee tres niveles de abstracción, el más bajo permite una gran flexibilidad pero su uso es muy complejo por lo que para los objetivos de la aplicación se usó el nivel más alto que provee las clases necesarias para la comunicación. Su estructura consiste en una clase base llamada ***NgnEngine*** que provee todos los servicios necesarios para la aplicación. Todos los servicios deben ser obtenidos a través de esta clase.

Estos servicios heredan de la clase ***NgnBaseService*** y son iniciados automáticamente al crear la instancia de la clase ***NgnEngine***.

Para manejar los eventos relacionados al establecimiento y mantenimiento de la sesión SIP en el servidor de llamadas se usa el servicio ***NgnSipService***, que maneja dichos eventos usando la clase ***NgnSipStack***, siendo esta la que define la pila del protocolo SIP.

La sesión de audio y video utilizada para realizar y recibir las videollamadas se define mediante la clase ***NgnAVSession***, donde se declaran los consumidores y productores de audio y video y provee los métodos para realizar una llamada usando la data de la pila SIP y el URI del destinatario. Mientras que para recibir una llamada se utiliza la clase ***NgnNativeService***,

éste servicio captura los eventos identificados como una llamada entrante y muestra la pantalla de llamada entrante a la vez que reproduce el tono mediante el servicio de sonido ***NgnSoundService***.

6.4.2 Consultas al servicio web desde Android

Para acceder a la base de datos de SOS Telemedicina se implementó una clase llamada ***ServiceHTTP***. Ésta se encarga de construir las solicitudes GET y POST a partir de un URL especifico determinado por el API REST desarrollado para el sistema.

Los métodos de solicitud retornan una cadena de caracteres en formato JSON que contiene la respuesta a la solicitud realizada además de la metadata de dicha respuesta. Ésta cadena de caracteres es transformada en un objeto de tipo JSON del que se puede extraer toda la data para su procesamiento. Éste mecanismo es utilizado para todas las operaciones sobre la base de datos así como para la autenticación del usuario contra el servicio web.

6.4.3 Desarrollo en Java/Android

El código de la aplicación consiste en el conjunto de *activities* que representan cada pantalla junto a un archivo XML que define la disposición, forma y color de los elementos de interfaz de usuario como botones, campos de datos y títulos.

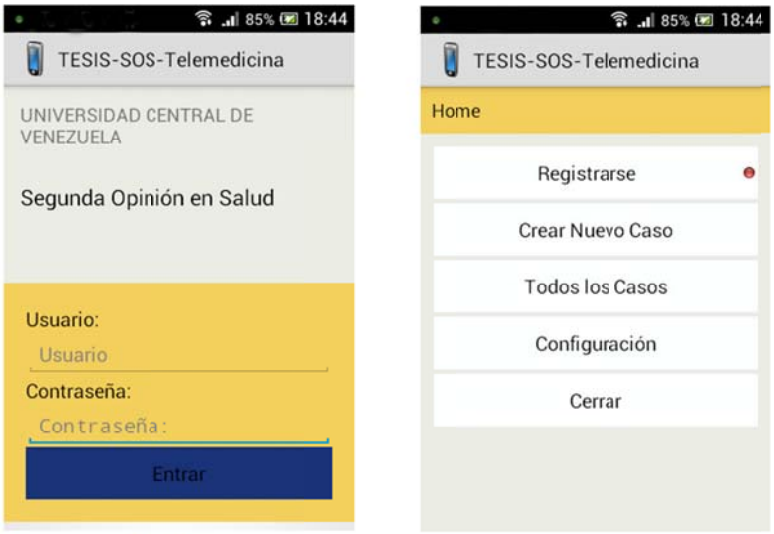


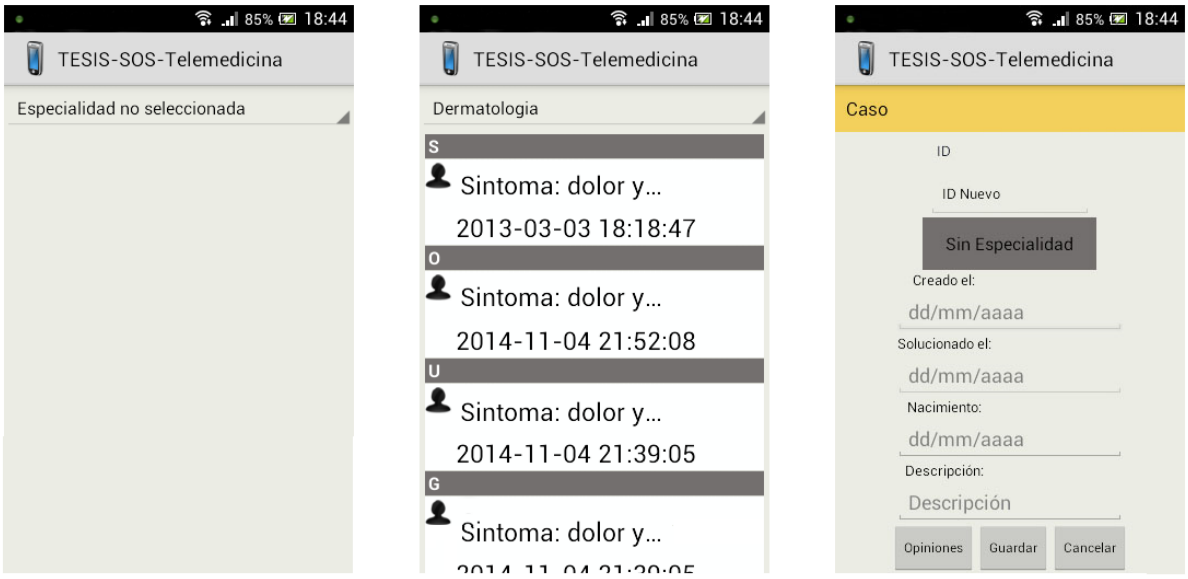
Figura 6.8: Captura de pantallas, (a) Pantalla de login y (b) Pantalla de home.

Después de acceder al sistema autenticándose con éxito en la primera pantalla que se muestra al abrir la aplicación, Figura 6.8 (a), se presenta al usuario el conjunto principal de operaciones que puede realizar, Figura 6.8 (b). En la pantalla de autenticación se permite al usuario establecer la dirección del servidor contra el cuál se realizará la autenticación.

Los datos de autenticación y configuración de los servidores pertenecientes al usuario son almacenados en una base de datos local ***SQLite*** que es manejada por la clase ***DataBaseHandler***, ésta hereda de la clase ***SQLiteOpenHelper*** y se encarga de crear y/o abrir la base de datos. Además en esta se implementan todos los métodos necesarios para crear la tabla perteneciente al usuario y realizar las consultas de creación, borrado, actualización y obtención de registros.

Al presionar el botón "**Registrarse**" se realiza el intento de inicio de sesión contra el servidor de llamadas Asterisk. Esta operación se ejecuta llamando al método ***NgnSipSession.Register(Context context)*** que usa las credenciales configuradas en la pantalla de Configuración para autenticarse y permitir realizar y recibir llamadas. Esta autenticación es independiente de la realizada contra el servicio web lo que permite consultar y crear casos sin haber iniciado sesión en el servidor de llamadas.

Para la consulta de los casos se implementó una pantalla, Figura 6.9 (a), que realiza una solicitud al servicio web requiriendo los casos pertenecientes a una especialidad indicada por el usuario, Figura 6.9 (b). Permite seleccionar un caso lo que lleva al usuario a una nueva pantalla en donde se muestra el caso en detalle, Figura 6.9 (c). Al final de la pantalla se encuentran los botones para agregar una opinión que implica realizar una llamada a un especialista o volver a la pantalla que lista los casos.

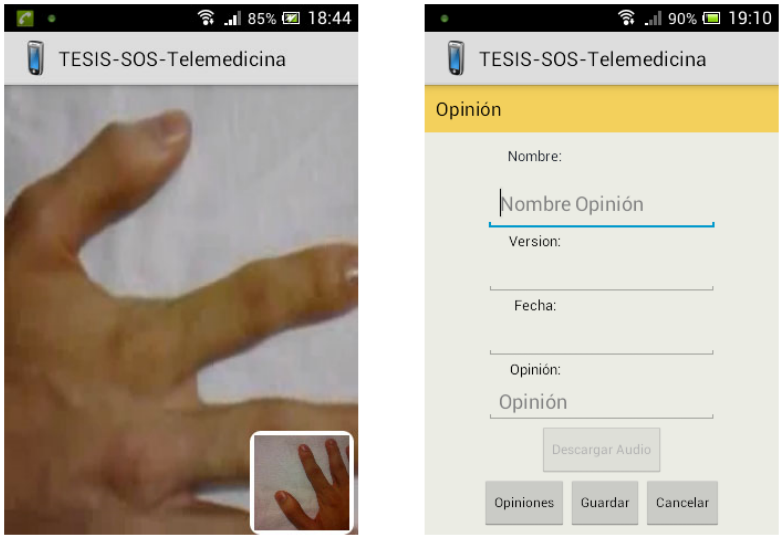


(a) **(b)** **(c)**

Figura 6.9: Captura de pantallas, (a) Pantalla para listar casos, (b) Lista de casos filtrada y (c) Detalles de un caso.

El botón "**Nueva Opinión**" realiza el intento de una videollamada que tiene como destino un grupo de llamada (*call group*) definido en Asterisk perteneciente a la especialidad del caso según el cual se redirige la llamada a cualquier especialista que esté disponible para atenderla,

Figura 6.10 (a). Luego de finalizar la llamada con éxito se muestra la pantalla de nueva opinión dónde se registran los detalles de la opinión creada, Figura 6.10 (b).



(a) (b)

Figura 6.10: Captura de pantallas, (a) Videollamada en curso y (b) Pantalla de opinión.

De regreso a la pantalla de menú principal, se puede crear un nuevo caso médico pulsando el botón "**Crear Nuevo Caso**", esta acción muestra la pantalla con el formulario que contiene los campos a llenar en donde "**Especialidad**" y "**Descripción**" son requeridos para poder guardar el caso en la base de datos; el campo fecha se ingresa automáticamente con la fecha actual.

En la pantalla "**Configuración**", Figura 6.11 (a), se presentan las opciones para establecer las credenciales de autenticación para registrarse contra el servidor de llamadas Asterisk en la opción "**Identidad**", Figura 6.11 (b), establecer la configuración de red en la opción "**Red**", Figura 6.11 (c), y el la opción "**Codecs**" se pueden activar y desactivar los códecs de audio y video a utilizar durante las videollamadas, Figura 6.11 (d).

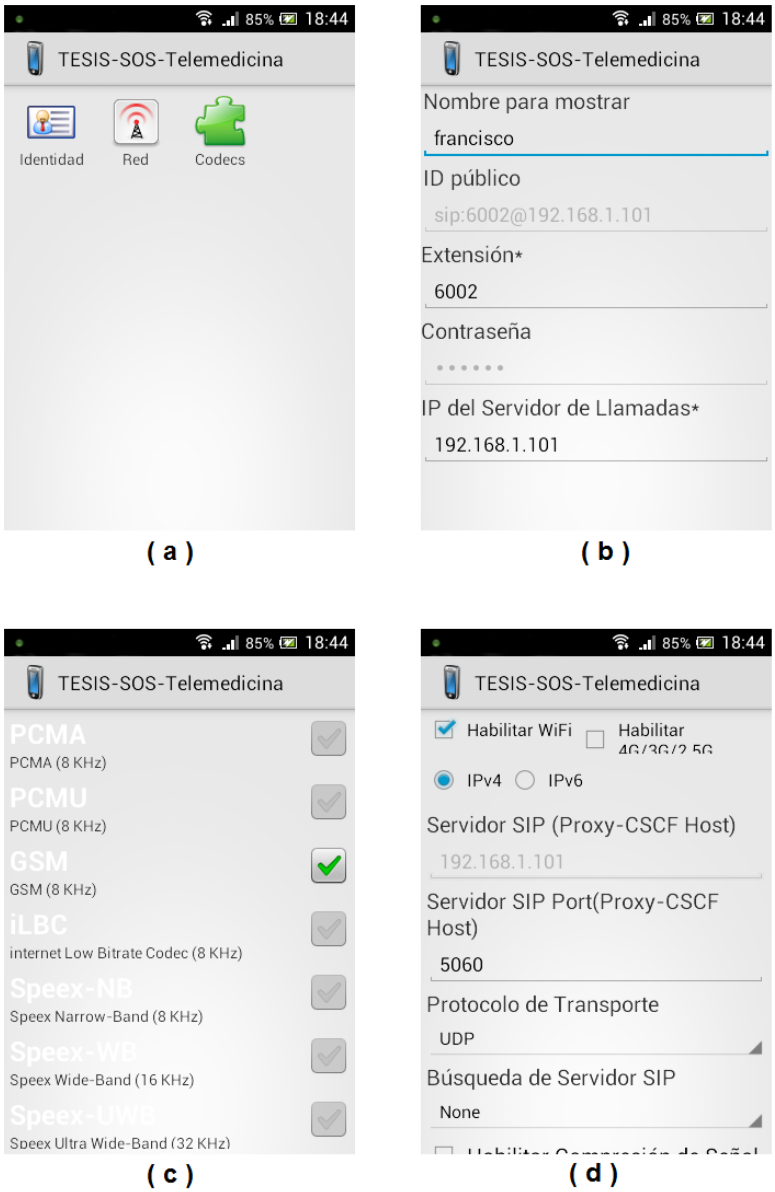


Figura 6.11: Captura de pantallas, (a) Menú de configuraciones, (b) Credenciales SIP (c) Codecs y (d) Red.

Por último, el botón "**Cerrar**" lanza una interfaz de dialogo confirmándole al usuario si desea salir de la aplicación y en caso afirmativo se cierra las sesiones en el servidor de base de datos y en el servidor de llamadas y se muestra la pantalla de autenticación inicial, Figura 6.12.

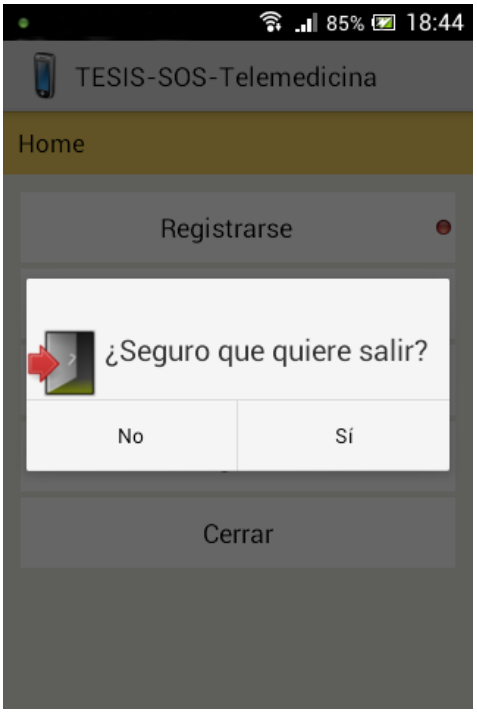


Figura 6.12: Captura de pantalla, salida de la aplicación.

7 Pruebas y resultados

Las pruebas que se realizaron tienen como objetivo evaluar el rendimiento y el comportamiento de la aplicación, para esto se tomaron en cuenta el porcentaje de uso del procesador y el espacio ocupado en la memoria principal (*Random Access Memory*, RAM) del dispositivo, el ancho de banda mínimo necesario, la variación del tiempo de retardo (jitter) y el porcentaje de paquetes perdidos. Éstas métricas son relevantes para determinar cuáles son los requerimientos mínimos de hardware del dispositivo y los requerimientos mínimos de red para que la aplicación funcione correctamente.

Para medir el rendimiento de la aplicación se realizaron las pruebas en dos partes, la primera comprende la evaluación del desempeño de la interfaz y las consultas al servidor de la base de datos y la segunda comprende la evaluación del desempeño solamente durante una videollamada. Las pruebas se realizaron sobre un dispositivo con sistema operativo Android, marca **Sony** modelo **XperiaGo ST27i** sus especificaciones se muestran en la siguiente tabla.

Tabla 7.1: Especificaciones del dispositivo de pruebas.

Marca	Sony
Modelo	XperiaGo ST27i
Sistema Operativo	Android v2.3 actualizable a v4.1.2
Procesador (CPU)	Dual-Core 1 GHz - Cortex-A9
Procesador Gráfico (GPU)	Mali-400
Cámara	5 MP - 2592 x 1944 pixeles – autofoco - flash LED
Video	720p a 30fps

También se realizaron pruebas de funcionalidad sobre la aplicación con el fin de determinar si la aplicación cumple con su propósito en manos del usuario además de arrojar recomendaciones para implementar correcciones y mejoras futuras.

7.1 Pruebas de rendimiento en interfaz y consultas las bases de datos

Éstas pruebas miden el rendimiento de la aplicación en cuanto a uso del procesador, espacio en RAM, consumo de ancho de banda y ancho de banda mínimo necesario durante la creación y consulta de casos y opiniones, la configuración de los diversos parámetros y la

navegación entre las *activities* que la componen, sin incluir el realizar videollamadas a otros usuarios.

7.1.1 Uso del procesador

El objetivo de ésta prueba fue evaluar el consumo de procesador de la aplicación fuera de una videollamada, de ésta forma se pudo saber qué operaciones utilizan más este recurso para plantear posibles mejoras y encontrar anormalidades que deban ser corregidas.

Para obtener los valores de porcentaje de uso del procesador se utilizó el comando "***top***", éste se ejecutó durante **100 segundos** arrojando una salida cada segundo, durante ese tiempo un usuario recorrió todos los flujos y operaciones dentro de la aplicación sin realizar una videollamada. Esta prueba se repitió un total de 10 veces.

En la Figura 7.1 se muestra a manera de ejemplo una de las ejecuciones de la prueba, el promedio del porcentaje de uso de procesador para esa ejecución fue de 6,8%, y el promedio de las 10 ejecuciones fue de 6.73%. El valor más alto registrado durante toda la prueba fue de un 25%.

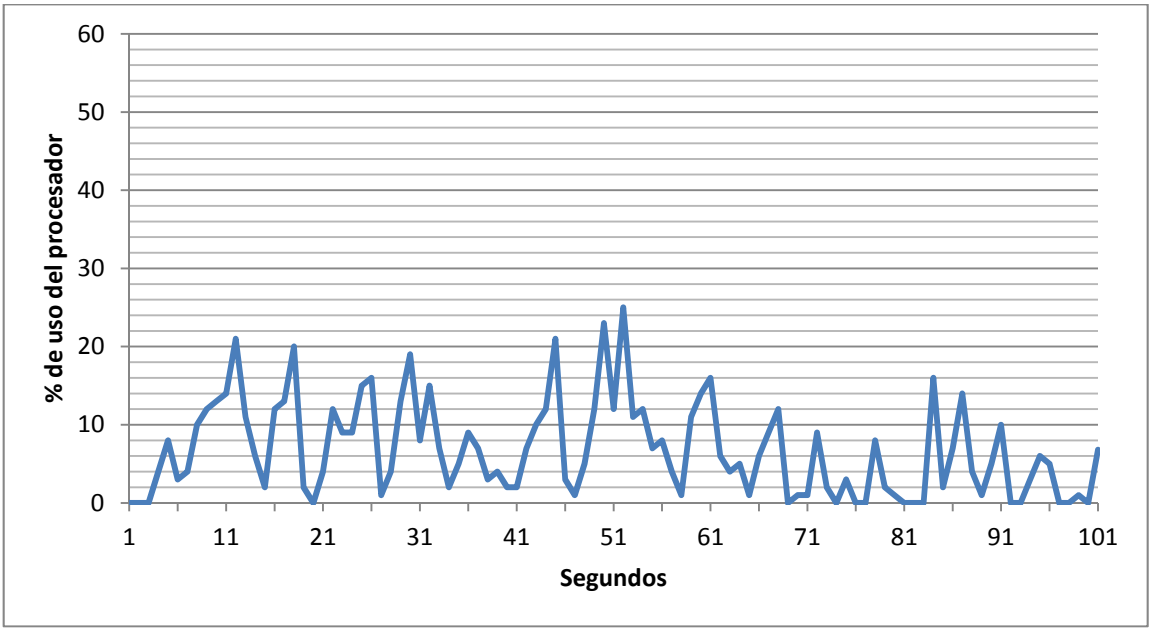


Figura 7.1: Gráfica de uso del procesador de la aplicación.

Los valores más altos en la gráfica (mayores a 15) corresponden a las peticiones que se realizan al servidor, mientras que los valores más bajos entre 0 y 2 sucedieron cuando la aplicación no estaba en uso. Esto indica que la interfaz y las peticiones al servidor de base de datos no conllevan una sobrecarga mayor sobre el procesador del dispositivo. También se observó que la duración de las peticiones en donde se produce un mayor consumo del procesador no fue mayor a 2 segundos y al ser una operación que no se realiza

frecuentemente permite, generalmente, un funcionamiento sin retardos en lo que respecta al uso de procesador.

7.1.2 Consumo de memoria

Haciendo uso nuevamente del comando "***top***" se obtuvo el espacio en RAM ocupado por la aplicación cada segundo durante **100 segundos**, la salida del comando arroja los valores para la **memoria virtual** y para la **memoria residente** del proceso en la RAM del dispositivo. Ésta prueba se ejecutó 10 veces obteniendo un promedio de 347.151 Kilobytes (339 Megabytes) para la memoria virtual ocupada por la aplicación y un promedio de 37.244 Kilobytes (37 Megabytes) para la memoria residente. En la Figura 7.2 se muestra una traza del espacio para la memoria virtual y en la Figura 7.3 se observan los valores correspondientes a la memoria RAM para la misma traza.

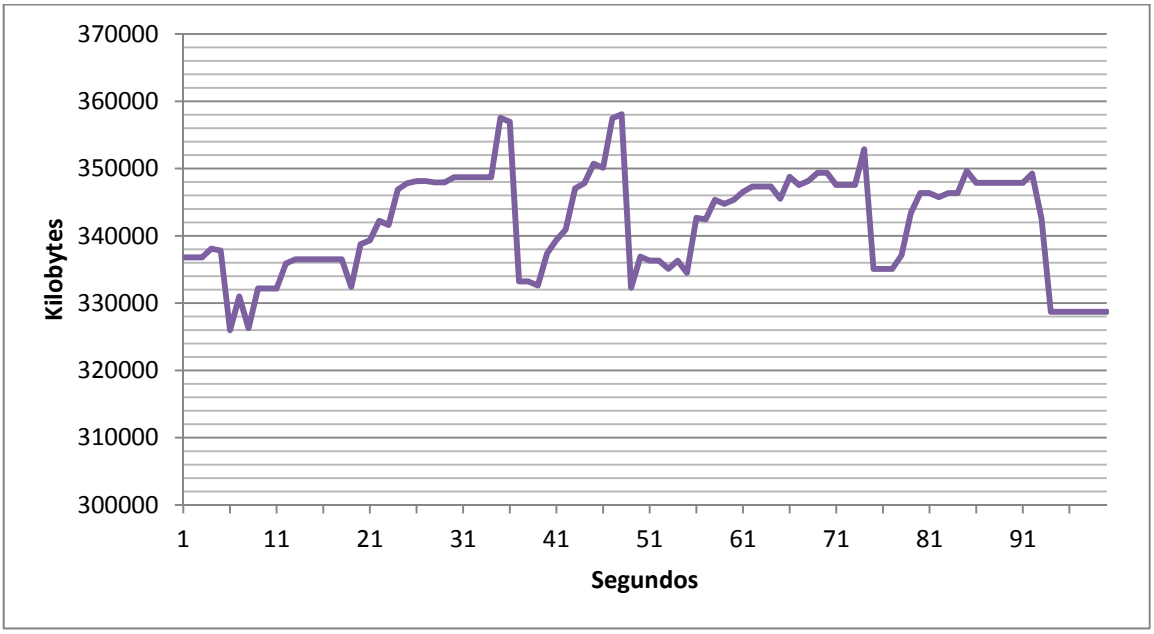


Figura 7.2: Gráfica de uso del memoria virtual de la aplicación.

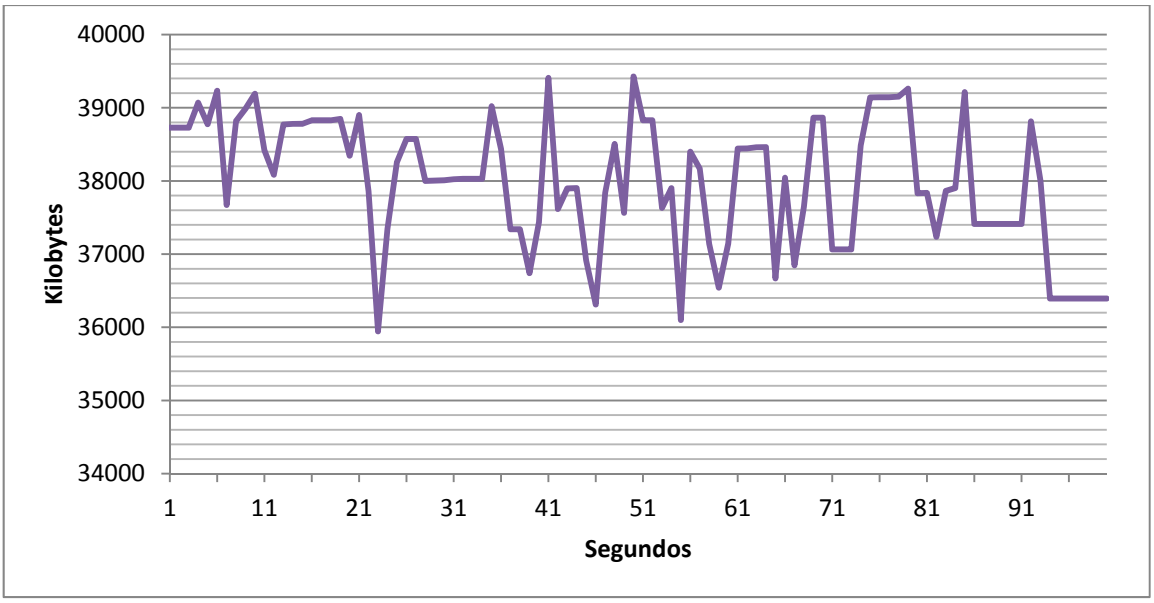


Figura 7.3: Gráfica de uso de la memoria residente de la aplicación.

7.2 Pruebas de rendimiento durante una videollamada

Este conjunto de pruebas es similar al realizado para la interfaz y las peticiones al servidor, busca determinar el rendimiento de la aplicación durante una videollamada en cuanto a uso del procesador, espacio en RAM, consumo de ancho de banda y ancho de banda mínimo necesario. También se compara el rendimiento usando los códecs de video H.263 y H.264 y los códecs de audio GSM y G729 por lo que se hicieron pruebas combinando cada uno de ellos.

7.2.1 Uso del procesador

Para evaluar el uso del procesador en el curso de una videollamada se usó también el comando "***top***" y se tomaron muestras cada segundo durante **60 segundos**, en éste intervalo se transmitió tanto audio como video. Se realizaron cuatro pruebas correspondientes a la combinación de cada códec usado. Para cada prueba se realizaron 5 ejecuciones cada una representa una videollamada.

- **H.263 + GSM**

  En esta prueba se evalúa el rendimiento usando el códec H.263 para video y GSM para audio. las cinco ejecuciones para estos códecs arrojaron un promedio de 34,51% de uso del procesador, siendo el valor más alto encontrado 42%, en la Figura 7.4 se muestran los valores que ejemplifican una llamada.

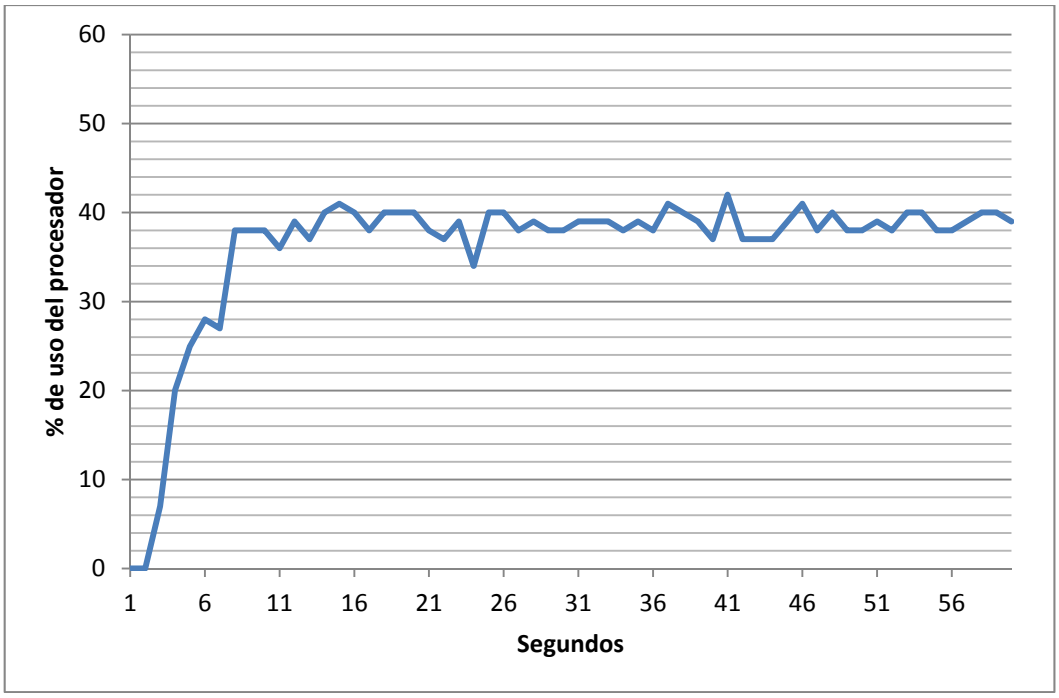


Figura 7.4: Gráfica de uso del procesador durante videollamada (H.263 + GSM).

- **H.263 + G729**

  Para estás cinco ejecuciones el promedio de uso del procesador fue de 33,96% y el valor más alto encontrado fue también de 42% como se muestra en la Figura 7.5.

Figura 7.5: Gráfica de uso del procesador durante videollamada (H.263 + G729).

- **H.264 + GSM**

  El promedio obtenido durante esta serie de ejecuciones fue de 29,24% y el mayor valor encontrado fue de 41% de consumo de procesador, ésta ejecución se muestra como ejemplo en la Figura 7.6.

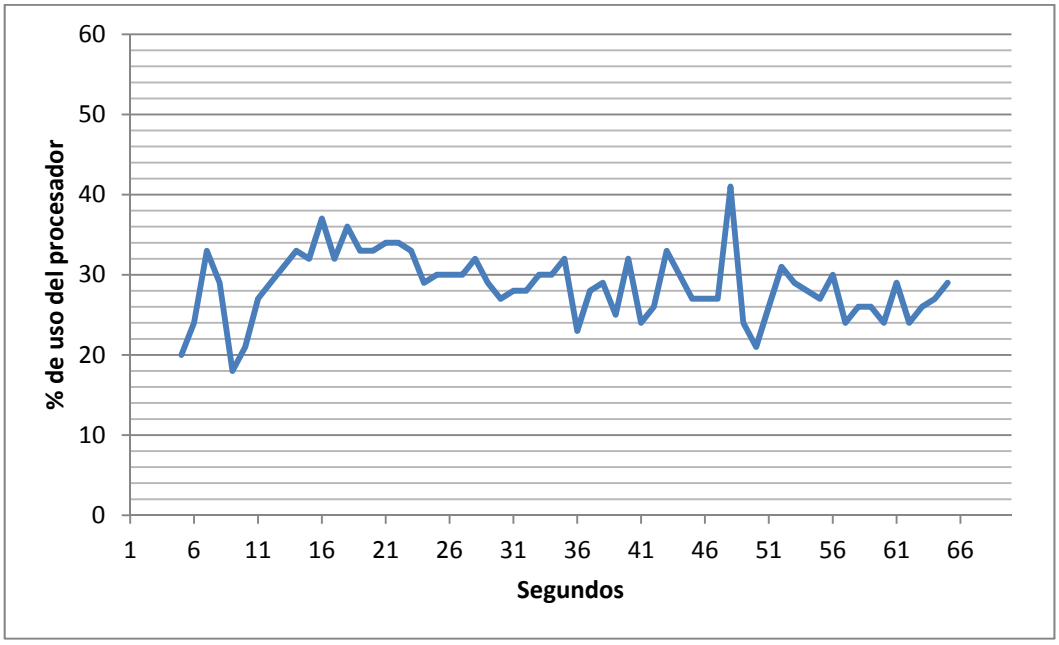


Figura 7.6: Gráfica de uso del procesador durante videollamada (H.264 + GSM).

- **H.264 + G729**

  En ésta última prueba el promedio de uso de procesador fue de 35,71% con el mayor valor registrado en 51%. Se muestra un ejemplo de traza para una llamada en la Figura 7.7.

Figura 7.7: Gráfica de uso del procesador durante videollamada (H.264 + G729).

Con respecto a los resultados obtenidos de las pruebas de rendimiento sobre el procesador se observan pocas diferencias en el consumo de este recurso exceptuando la combinación de los códecs H.264 y G729 que obtuvo unos valores un poco más elevados. Pero para todos estos casos es necesario un dispositivo con una capacidad de procesamiento similar a la que posee el dispositivo sobre el cual se llevaron a cabo las pruebas; un porcentaje de consumo de procesador entre 29% y 42% es alto y si se somete el dispositivo a carga similar por parte de otras aplicaciones simultáneamente se podrían presentar dificultades durante la video-llamada.

Entre las observaciones que se pudieron realizar durante las pruebas cabe mencionar que los valores más altos para el consumo de procesador se obtuvieron cuando la imagen era muy inestable y presentaba muchas diferencias de un frame a otro lo que impide al códec aprovechar frames anteriores para predecir los siguientes trayendo como consecuencia un aumento en el consumo de recursos.

7.2.2 Consumo de memoria

Para determinar el consumo de memoria de la aplicación durante una llamada se procedió de igual forma que para el consumo de procesador, obteniendo los valores usando el comando "***top***" y evaluando los datos para los códecs seleccionados. Para el consumo de memoria virtual se obtuvieron los valores promedio expuestos en la Figura 7.8 y para el consumo de memoria RAM los valores promedio mostrados en la Figura 7.9.

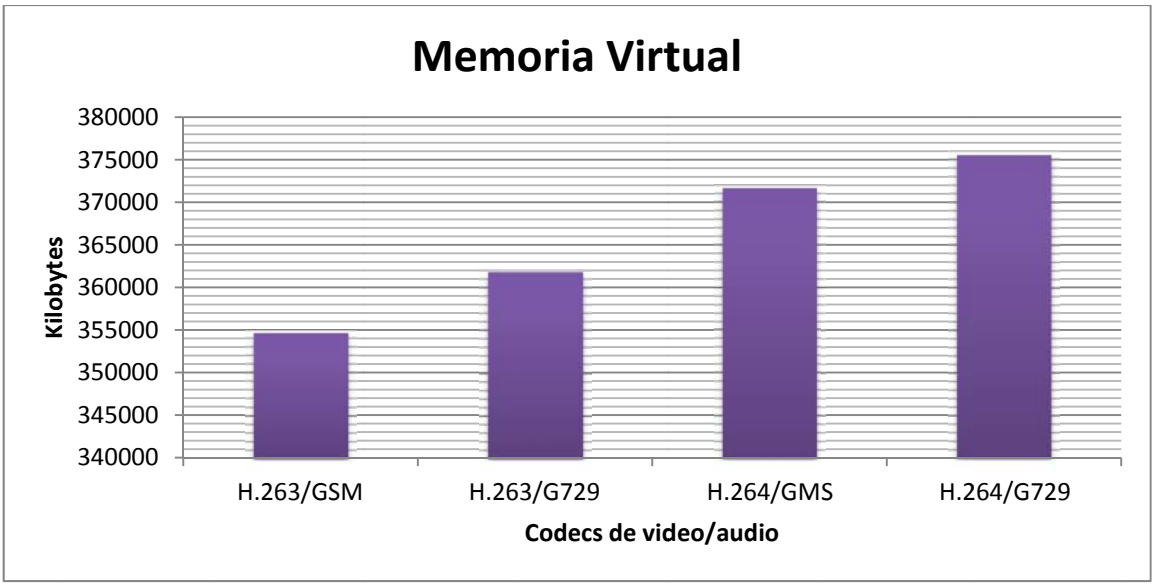


Figura 7.8: Gráfica de uso de memoria virtual durante una videollamada.

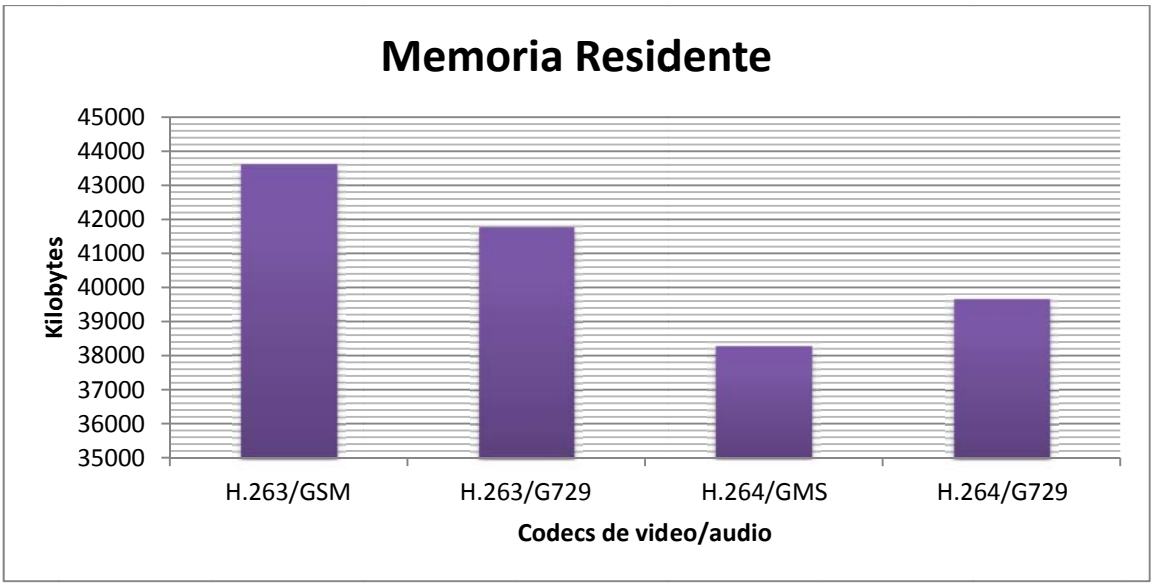


Figura 7.9: Gráfica de uso de memoria residente durante una videollamada.

Estos valores se pueden ver afectados tanto por los métodos de compresión y codificación de cada códec como por su implementación en el *framework*. Los valores de memoria virtual y memoria RAM parecen estar correlacionados, sin embargo, siendo la RAM como recurso del dispositivo más restrictiva que la memoria virtual, los valores obtenidos de su consumo se consideran con mayor peso para decidir que códecs utilizar.

En estos resultados se observa que la memoria RAM no representa una carga excesiva para el dispositivo, el máximo valor observado en las pruebas tanto de interfaz como durante una video-llamada es aproximadamente 37 Megabytes en memoria residente, esta cantidad de memoria no es limitante para el rendimiento apropiado de la aplicación comparándola con la

cantidad disponible de 512 Megabytes de los cuales dispone el dispositivo usado en las pruebas. La aplicación podría ejecutarse en dispositivos similares con menor cantidad de RAM teniendo en cuenta la carga que representan otras aplicaciones ejecutándose simultáneamente.

7.2.3 Pruebas de red

Éste conjunto de pruebas se realizó para obtener los valores necesarios con los que la aplicación puede funcionar correctamente en una red. Los valores medidos fueron la variación del tiempo de retraso entre cada paquete (jitter), el porcentaje de paquetes perdidos durante la llamada y el ancho de banda mínimo para reproducir el audio y el video. Estas métricas se obtuvieron usando el programa de captura de tráfico de red ***Wireshark***. Se configuró un ambiente en el cual dos dispositivos móviles establecieron un total de 24 videollamadas, tres para cada uno de ocho anchos de banda distintos.

Para la red en la que se encontraban los dispositivos móviles se usó un enrutador al cual se le estableció una velocidad de transmisión de 768 KBps para las primeras 15 llamadas y luego se redujo a 128 KBps para las 9 restantes. Para controlar la carga de la red dentro de cada una de estas velocidades establecidas en el router se utilizó el generador de tráfico de red D-ITG (*Distributed Internet Traffic Generator*). Las videollamadas tuvieron una duración de 60 segundos cada una, durante los cuales se capturaron los paquetes y se obtuvieron los valores mostrados en las siguientes figuras.

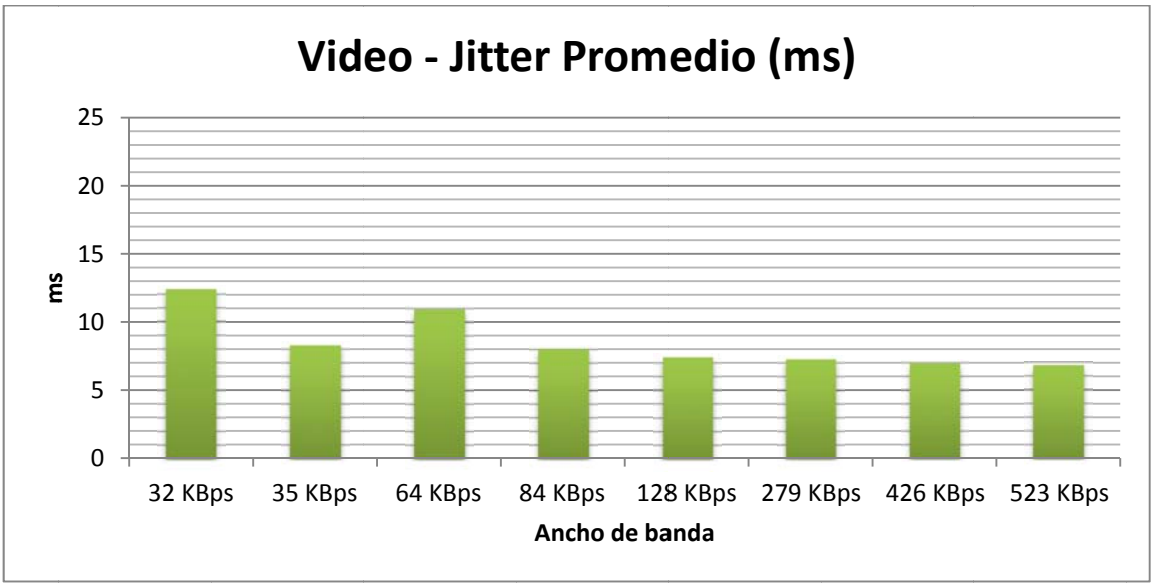


Figura 7.10: Gráfica de jitter promedio en video con diferentes ancho de banda.

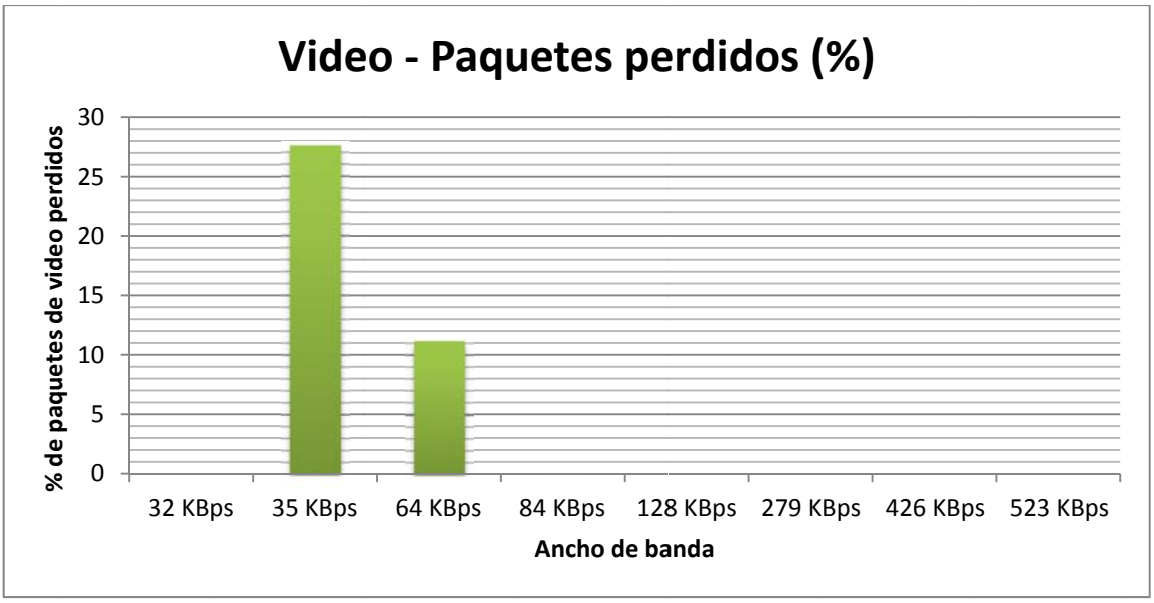


Figura 7.11: Gráfica de porcentaje de paquetes de video perdidos con diferentes ancho de banda.

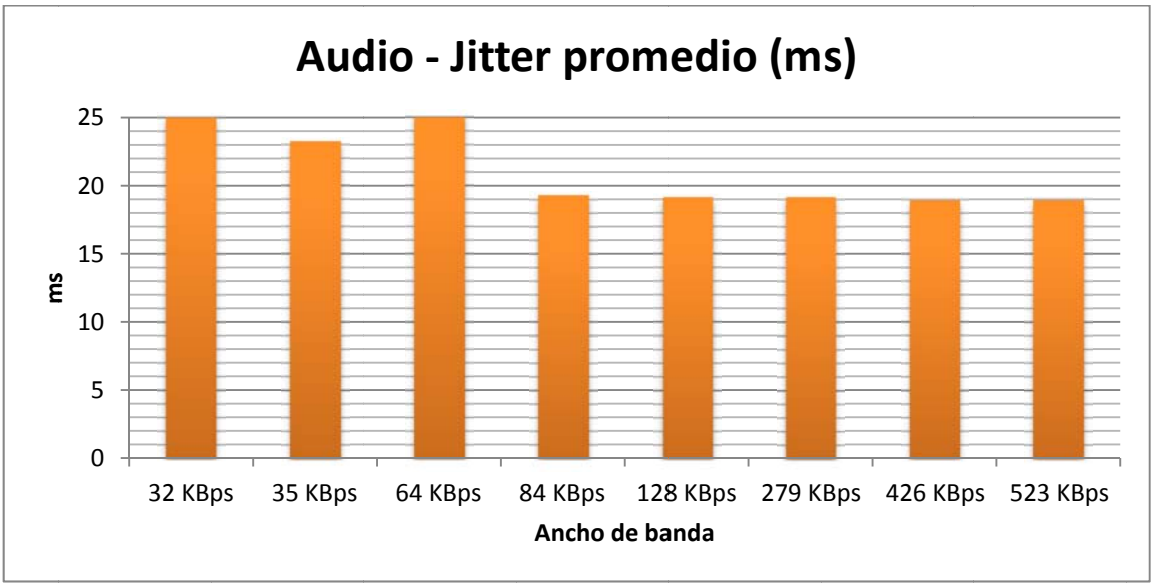


Figura 7.12: Gráfica de jitter promedio en audio con diferentes ancho de banda.

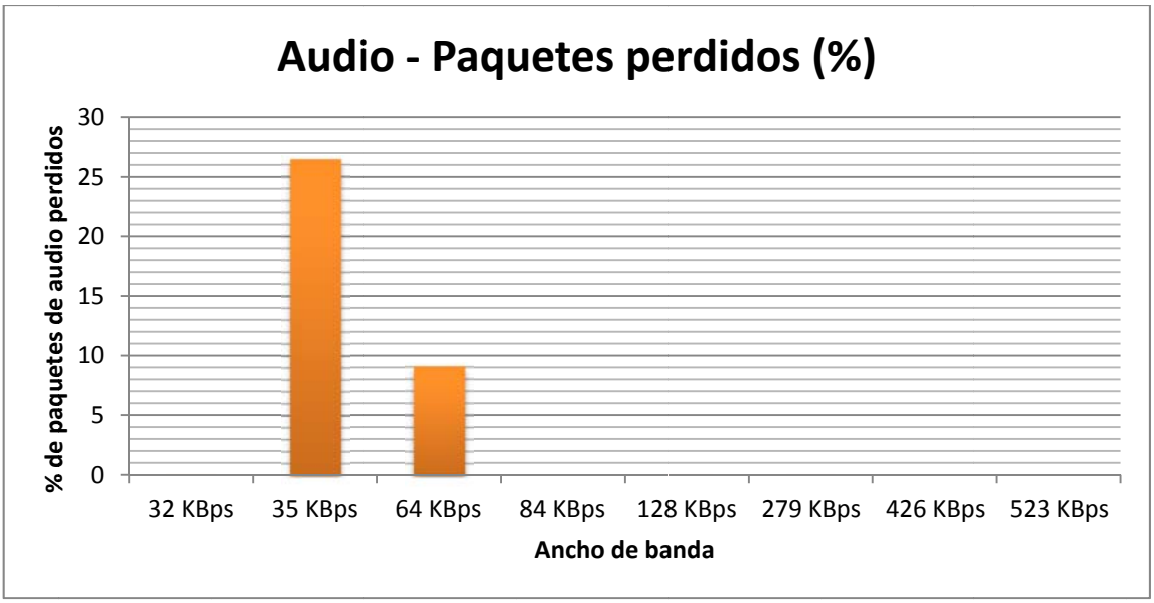


Figura 7.13: Gráfica de porcentaje de paquetes de audio perdidos con diferentes ancho de banda.

En los resultados de las pruebas de red se observó que el jitter fue la métrica más afectada por el ancho de banda a diferencia del porcentaje de paquetes perdidos que estuvo muy cercano a 0% luego de incrementar el ancho de banda a 84 KBps. Sin embargo el porcentaje de paquetes perdidos en este caso no es un indicativo definitivo para medir el rendimiento de red de la aplicación, ya que los paquetes al llegar podrían ser descartados por la aplicación a causa de un retardo muy alto y unos pocos paquetes no afectan el desempeño general de la videollamada. Por esto, considerando un jitter promedio muy alto (cercano a 20 ms para audio y cercano a 10 ms para video), se puede establecer un mínimo de ancho de banda de aproximadamente 128 KBps, se observó que este ancho de banda permite la comunicación sin artefactos aunque presenta un retardo de aproximadamente un segundo desde el punto de vista del usuario.

7.3 Pruebas de funcionalidad

Se realizaron las pruebas de funcionalidad sobre la aplicación desde el punto de vista del usuario, se probaron las funcionalidades y requerimientos de la aplicación una por una para determinar posibles errores y fallas luego del proceso de desarrollo. Las pruebas comprenden las operaciones que el usuario puede realizar y se evaluó el resultado esperado contra el resultado obtenido.

Se evaluaron las operaciones que involucran el acceso a la base de datos mediante el API REST, de esta manera se corroboró el correcto funcionamiento del servicio web, estás son:

- **Iniciar sesión / Cerrar sesión:** resultado satisfactorio, el usuario puede ingresar al sistema con las credenciales correctas; el usuario puede finalizar la sesión de forma correcta.

- **Obtener lista de casos:** resultado satisfactorio, se listan los casos según la especialidad seleccionada.

- **Obtener un caso en específico:** resultado satisfactorio, se muestra el caso seleccionado con sus respectivos detalles.

- **Obtener lista de opiniones de un caso:** resultado satisfactorio, se listan las opiniones pertenecientes a un caso.

- **Obtener opinión específica de un caso:** resultado satisfactorio, se muestra los detalles de la opinión seleccionada para dicho caso.

- **Guardar caso:** resultado satisfactorio, el caso nuevo es almacenado correctamente en la base de datos.

- **Guardar opinión:** resultado satisfactorio, la nueva opinión es almacenada correctamente en la base de datos.

Luego se procedió a probar las siguientes funcionalidades involucradas en la videollamada, estas pruebas buscaron conseguir posibles errores en la conexión de la aplicación móvil con la central IP-PBX Asterisk, así como en el establecimiento de las sesiones SIP/RTP, la transferencia de video y audio y por último almacenar el audio de la llamada en el servidor:

- **Llamar especialista:** resultado satisfactorio, se realizó la llamada sin ningún inconveniente.

- **Atender llamada:** resultado satisfactorio, el usuario que recibe la llamada atiende y se establece la transferencia de audio y video sin inconvenientes.

- **Detener/Reanudar video:** resultado satisfactorio, es posible pausar y reanudar la transferencia de video sin problemas.

- **Terminar llamada:** resultado satisfactorio, el usuario puede terminar la llamada sin problemas.

- **Almacenar audio de llamada:** resultado satisfactorio, el audio de cada llamada es almacenado en la base de datos de Asterisk.

Las pruebas de funcionalidad que se realizaron no arrojaron inconvenientes en las funcionalidades probadas, por lo que es posible para el usuario cumplir con todos los flujos requeridos para crear casos, opiniones y realizar videollamadas a especialistas.

8 Conclusiones

El producto final de este proceso de investigación y desarrollo fue el sistema que comprende la Aplicación de Videollamadas sobre Android aplicada al Sistema de Segunda Opinión Médica de SOS Telemedicina. La aplicación se integra con el Sistema de Segunda Opinión Médica proporcionando a los médicos participantes del proyecto la posibilidad de comunicación en tiempo real móvil para la atención de casos.

El sistema desarrollado comprende la aplicación de videollamada sobre Android, un servicio web basado en un API REST y una central IP-PBX Asterisk usando FreePBX como interfaz de administración y configuración. La integración se da a través del acceso a la base de datos del Sistema de Segunda Opinión Médica, desde esta se extrae y/o almacena toda la información pertinente a los casos y opiniones mediante el servicio web.

El desarrollo de la Aplicación de Videollamada se hizo en base al *framework* Doubango, haciendo uso de su versión libre para Android para el lado del cliente. Se logró realizar la correcta implementación de las funcionalidades relacionadas a la videollamada usando este *framework*.

Se implementó la metodología de desarrollo ágil Scrum para el manejo del proyecto, esta permitió organizar los requerimientos y dividirlos en tareas más sencillas de abordar. También proporcionó flexibilidad en las iteraciones y cambios que se presentaron.

Esta solución proporciona un avance más en el proyecto de SOS Telemedicina, con el fin de proporcionar a poblaciones rurales de servicios de telemedicina y salud especializados y de calidad. La Aplicación permite a los médicos rurales un acceso síncrono y en tiempo real a la información que pueden proporcionar otros especialistas de manera de solventar a través del Sistema de Segunda Opinión Médica patologías específicas.

8.1 Contribuciones

El sistema de la Aplicación de Videollamada de Segunda opinión Médica provee un conjunto importante de beneficios que está alineado con el objetivo principal del proyecto de SOS Telemedicina y del área de telemedicina en general, que es llevar la atención médica especializada y de calidad a zonas rurales o de difícil acceso, así como proveer al médico de movilidad y la capacidad de acceder a la información perteneciente a los casos desde casi cualquier lugar donde exista una conexión a Internet.

Además la videollamada permite una interacción y comunicación con mayor nivel, poder hablar directamente con el médico especialista e incluso ver una imagen enriquece la transferencia de información de manera muy significativa, sumado a la atención del caso en

tiempo real lo que agiliza enormemente el tiempo de respuesta de los médicos rurales frente a casos que ameriten una atención con un grado de urgencia mayor.

La integración de la aplicación con el sistema existente se da de una manera limpia y transparente por lo que lleva a una ampliación del sistema de Segunda Opinión Médica que no conlleva cambios que dificulten el flujo de las operaciones y funciones que se llevan a cabo antes de la implementación del sistema de la Aplicación de Videollamada.

El elemento primordial que permitió esta integración fue el API REST desarrollado para dar acceso tanto a la base de datos del Sistema de Segunda Opinión Médica como a la base de datos de la central IP-PBX Asterisk desde la Aplicación de Videollamada sobre Android. En el API se encuentran definidos un conjunto de URL para obtener la información requerida mediante solicitudes HTTP permite realizar cualquier operación necesaria a la base de datos.

El API REST, por sí solo, provee una herramienta importante en cuanto a la extensibilidad del proyecto que comprende el sistema de Segunda Opinión Médica, ya que permite a cualquier módulo un acceso de manera segura a la base de datos del sistema completo.

8.2 Limitaciones

La necesidad de implementar una nueva solución para la interfaz de administración de la central IP-PBX Asterisk fue un factor importante en el tiempo de culminación del proyecto. En un principio se seleccionó Elastix para cubrir estos requerimientos pero dado que no fue posible instalarlo en los servidores de SOS Telemedicina se optó por realizar una instalación y configuración completa de Asterisk junto a FreePBX, la cual funciona correctamente y cubre con los requerimientos propuestos.

Durante el proceso de desarrollo se presentó la dificultad importante de no poder probar sobre una cantidad considerable de dispositivos, esto no afecta el funcionamiento de la videollamada pero si puede repercutir negativamente en la estructura visual e interfaz de la aplicación móvil. Otra limitante importante durante el desarrollo sobre Android fue la falta de documentación sobre el *framework* Doubango, fueron necesarias muchas horas dedicadas a entender el funcionamiento del *framework*, pero que finalmente dieron los resultados esperados.

Un aspecto a considerar es la configuración de la aplicación móvil por parte del usuario, al no existir una entrada DNS para estos servicios que corren en los servidores de SOS Telemedicina puede ser engorroso para el usuario final realizar la configuración de la URL de cada servidor. Sin embargo, para mitigar esta limitante se provee una documentación que aclara estos procedimientos.

8.3 Trabajos futuros

El uso del *framework* Doubango provee otro conjunto de posibles extensiones al sistema, ya que además de las funcionalidades que permiten llevar a cabo una videollamada,

proporciona las herramientas para implementar un módulo de transferencia de archivos y un módulo de mensajería instantánea.

La instalación de la central IP-PBX también provee la capacidad de extender el uso de videollamadas a otras aplicaciones o soluciones con capacidad de establecer sesiones SIP/RTP en la central IP-PBX Asterisk. De esta manera se puede dar mayor alcance al proyecto cubriendo una mayor cantidad de plataformas.

Es posible implementar la aplicación sobre otras plataformas (Como iOS y Windows Phone) usando el *framework* Doubango. Esto daría más alcance y facilidades al proyecto al no estar limitado a un solo sistema operativo.

Para dar soporte a otras aplicaciones que puedan ser clientes del servicio web se puede implementar el envío de respuestas en formato XML.

Por último es recomendable actualizar el *framework* Doubango para tener acceso a mejoras en el rendimiento de la videollamada.

Apéndices

9 Apéndice A – Scripts de MySQL para la modificación de la base de datos SOS Telemedicina

En las siguientes figuras se muestran los comandos a ejecutar en la base de datos de SOS Telemedicina (*sos_triaje*) que realiza las modificaciones necesarias para el funcionamiento del proyecto.

```
ALTER TABLE archivo DROP FOREIGN KEY `FKD368E0CC575F15CE`;

ALTER TABLE caso DROP FOREIGN KEY `FK2E7B3AC4E46216`;

ALTER TABLE caso DROP FOREIGN KEY `FK2E7B3AD8BF583C`;

ALTER TABLE caso DROP FOREIGN KEY `FK2E7B3ADBDDF23E`;

ALTER TABLE caso_especialidad DROP FOREIGN KEY
`FK4E7036174A6E53D6`;

ALTER TABLE caso_especialidad DROP FOREIGN KEY
`FK4E703617C08DA023`;

ALTER TABLE especialista_especialidades DROP FOREIGN KEY
`FK21C94C5421186A7F`;

ALTER TABLE especialista_especialidades DROP FOREIGN KEY
`FK21C94C544A6E53D6`;

ALTER TABLE historial_caso DROP FOREIGN KEY
`FKD8E3C9CA575F15CE`;

ALTER TABLE historial_caso DROP FOREIGN KEY
`FKD8E3C9CADDADB17F`;

ALTER TABLE opinion DROP FOREIGN KEY `FKB4EDB382575F15CE`;

ALTER TABLE opinion DROP FOREIGN KEY `FKB4EDB382DDADB17F`;

ALTER TABLE opinion_opinion DROP FOREIGN KEY
`FKDDC86BC5387563EE`;

ALTER TABLE opinion_opinion DROP FOREIGN KEY
`FKDDC86BC5CA99929D`;

ALTER TABLE actor_sistema ADD api_key VARCHAR(255);

ALTER TABLE actor_sistema ADD user_extension VARCHAR(255);
```

Figura 9.1: Script SQL para modificaciones a la BD sos_triaje - Parte 1.

```
ALTER TABLE archivo ADD mime_type VARCHAR(255) default
'image/jpg';

ALTER TABLE caso ADD FK_actor_sistema bigint(20);

ALTER TABLE especialidad ADD group_extension VARCHAR(255);

ALTER TABLE opinion ADD cdr_uniqueid VARCHAR(255);

ALTER TABLE opinion ADD calldate datetime;

ALTER TABLE opinion ADD recordingfile VARCHAR(255);

ALTER TABLE paciente MODIFY id bigint(20) AUTO_INCREMENT;


ALTER TABLE `archivo`

ADD CONSTRAINT `FKD368E0CC575F15CE` FOREIGN KEY (`caso_id`)
REFERENCES `caso` (`id`) ON UPDATE CASCADE ON DELETE CASCADE;


ALTER TABLE `caso`

ADD CONSTRAINT `FK2E7B3AC4E46216` FOREIGN KEY (`status_id`)
REFERENCES `status` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `FK2E7B3AD8BF583C` FOREIGN KEY (`paciente_id`)
REFERENCES `paciente` (`id`) ON UPDATE CASCADE,

ADD CONSTRAINT `FK2E7B3ADBDDF23E` FOREIGN KEY (`centro_id`)
REFERENCES `centrosos` (`id`) ON UPDATE CASCADE,

ADD FOREIGN KEY (`FK_actor_sistema`) REFERENCES `actor_sistema`
(`id`) ON UPDATE CASCADE;


ALTER TABLE `caso_especialidad`

ADD CONSTRAINT `FK4E7036174A6E53D6` FOREIGN KEY
(`especialidad_id`) REFERENCES `especialidad` (`id`) ON UPDATE
CASCADE ON DELETE CASCADE,

ADD CONSTRAINT `FK4E703617C08DA023` FOREIGN KEY
(`caso_especialidades_id`) REFERENCES `caso` (`id`) ON UPDATE
CASCADE ON DELETE CASCADE;
```

Figura 9.2: Script SQL para modificaciones a la BD sos_triaje - Parte 2.

```
ALTER TABLE `especialista_especialidades`

ADD CONSTRAINT `FK21C94C5421186A7F` FOREIGN KEY
(`especialista_id`) REFERENCES `especialista` (`id`) ON UPDATE
CASCADE ON DELETE CASCADE,

ADD CONSTRAINT `FK21C94C544A6E53D6` FOREIGN KEY
(`especialidad_id`) REFERENCES `especialidad` (`id`) ON UPDATE
CASCADE ON DELETE CASCADE;


ALTER TABLE `historial_caso`

ADD CONSTRAINT `FKD8E3C9CA575F15CE` FOREIGN KEY (`caso_id`)
REFERENCES `caso` (`id`) ON UPDATE CASCADE ON DELETE CASCADE,

ADD CONSTRAINT `FKD8E3C9CADDADB17F` FOREIGN KEY (`medico_id`)
REFERENCES `medico` (`id`) ON UPDATE CASCADE;


ALTER TABLE `opinion`

ADD CONSTRAINT `FKB4EDB382575F15CE` FOREIGN KEY (`caso_id`)
REFERENCES `caso` (`id`) ON UPDATE CASCADE ON DELETE CASCADE,

ADD CONSTRAINT `FKB4EDB382DDADB17F` FOREIGN KEY (`medico_id`)
REFERENCES `medico` (`id`) ON UPDATE CASCADE;


ALTER TABLE `opinion_opinion`

ADD CONSTRAINT `FKDDC86BC5387563EE` FOREIGN KEY (`opinion_id`)
REFERENCES `opinion` (`id`) ON UPDATE CASCADE ON DELETE CASCADE,

ADD CONSTRAINT `FKDDC86BC5CA99929D` FOREIGN KEY
(`opinion_opiniones_id`) REFERENCES `opinion` (`id`) ON UPDATE
CASCADE ON DELETE CASCADE;
```

Figura 9.3: Script SQL para modificaciones a la BD sos_triaje - Parte 3.

10 Apéndice B – Instalación y configuración de Asterisk + FreePBX en Debian 6.0 (Squeeze)

Esta guía cubre la instalación y configuración de Asterisk y FreePBX en Debian 6.0 (*Squeeze*). Se asume que posee un servidor con una instalación de Debian básica sin el ambiente de escritorio, en pocas palabras, debe ejecutar el sistema en modo consola y no en modo grafico (*Graphic User Interface*, GUI) antes de comenzar con el proceso de instalación. Si ya tiene instalada una GUI es recomendado que sea desactivada en el arranque del sistema. También se asume que posee conexión a internet y permisos de raíz (root, #), es decir, que no sea un usuario con una cuenta limitada ($). El proceso de instalación fue probado con: Debian *Squeeze* 6.0, Asterisk 11 y FreePBX 2.11.

10.1 Actualización del sistema

Antes de comenzar con el proceso de instalación es necesario realizar la actualización del sistema, para ello se debe ejecutar el comando que muestra la Figura 10.1, si lo desea puede agregar o cambiar la lista de repositorios ubicado en ***/etc/apt/sources.list*** antes de ejecutar este comando. Para más información de cómo actualizar Debian puede consultar el siguiente enlace: https://www.debian.org/doc/manuals/debian-faq/ch-uptodate.en.html.

```
# apt-get update && apt-get upgrade –y
```

Figura 10.1: Comando para la actualización del sistema (Debian).

Al finalizar el proceso de actualización del sistema con éxito se procederá a instalar paquetes que son requeridos para Asterisk y FreePBX así como otros paquetes de utilidad y sus respectivas dependencias (Figura 10.2).

```
# apt-get install -y sudo build-essential openssh-server apache2
mysql-server mysql-client bison flex php5 php5-curl php5-cli
php5-mysql php-pear php-db php5-gd curl sox libncurses5-dev
libssl-dev libmysqlclient-dev mpg123 libxml2-dev libnewt-dev
sqlite3 libsqlite3-dev pkg-config automake libtool autoconf git
subversion uuid uuid-dev libiksemel-dev tftpd postfix mailutils
nano ntp chkconfig libspandsp-dev libcurl4-gnutls-dev unixodbc
unixodbc-dev libmyodbc xinetd e2fsprogs linux-headers*
```

Figura 10.2: Comando para instalar paquetes requeridos de Asterisk y FreePBX, otros paquetes de utilidad y sus dependencias.

Luego de ejecutar el comando de la Figura 10.2 se procederá a instalar cada uno de los paquetes, y a medida que esto ocurre se le solicitará asistencia para la configuración de postfix y para establecer la contraseña de root de MySQL. Por defecto el root de MySQL no posee contraseña y se puede continuar la instalación (no recomendado). Si establece una contraseña recuerde documentarla, más adelante le será solicitada.

Al culminar la instalación de todas las dependencias se debe instalar la versión de 1.7.14 de *Pear DB*, recibirá una advertencia de este tipo: ***WARNING: "pear/DB" is deprecated in favor of "pear/MDB2",*** puede ignorar esta advertencia sin problema.

```
# pear install db-1.7.14
```

Figura 10.3: Comando para la instalación de Pear 1.7.14.

Ahora proceda a **reiniciar** el servidor.

10.2 Instalación de Asterisk

Una vez ya con el sistema al día, daremos inicio al proceso de instalación y configuración de Asterisk y FreePBX, en la siguiente figura se muestran los comandos para la descarga de Asterisk y sus dependencias (Dahdi y Libpri) en el directorio ***/usr/src*** del servidor.

```
# cd /usr/src

# wget http://downloads.asterisk.org/pub/telephony/dahdi-linux-
complete/dahdi-linux-complete-current.tar.gz

# wget http://downloads.asterisk.org/pub/telephony/libpri/libpri-
1.4-current.tar.gz

# wget
http://downloads.asterisk.org/pub/telephony/asterisk/asterisk-11-
current.tar.gz
```

Figura 10.4: Comandos para descargar Asterisk y sus dependencias (Dahdi y Libpri).

Los comandos en la Figura 10.5, Figura 10.6 y Figura 10.7 se encargan de extraer, compilar e instalar Dahdi, Libpri y Asterisk respectivamente. Luego de ejecutar los comandos para instalar Asterisk (Figura 10.7) ***NO ejecute 'make samples'*** ya que puede dar conflicto con las configuraciones autogeneradas de FreePBX y causar problemas en el sistema.

```
# cd /usr/src
# tar zxvf dahdi-linux-complete*
# cd dahdi-linux-complete*
# make all
# make install
# make config
```

Figura 10.5: Comandos para extraer, compilar e instalar Dahdi.

```
# cd /usr/src
# tar zxvf libpri*
# cd libpri*
# make
# make install
```

Figura 10.6: Comandos para extraer, compilar e instalar Libpri.

```
# cd /usr/src
# tar zxvf asterisk-11*
# cd asterisk-11*
# ./configure CFLAGS=-mtune=native
# contrib/scripts/get_mp3_source.sh
# make menuselect
# make
# make install
# make config
```

Figura 10.7: Comandos para extraer, compilar e instalar Asterisk.

10.3 Instalación de FreePBX

Una vez instalado Asterisk es hora de instalar FreePBX el cual es una interfaz web para la administración de Asterisk. En la siguiente figura se muestran los comandos para la descarga de FreePBX desde un repositorio GIT en el directorio ***/usr/src*** del servidor y finalmente se posiciona el repositorio sobre la rama ***release/2.11***.

```
# export VER_FREEPBX=2.11
# cd /usr/src
# git clone http://git.freepbx.org/scm/freepbx/framework.git freepbx
# cd freepbx
# git checkout release/${VER_FREEPBX}
```

Figura 10.8: Comandos para descargar FreePBX desde un repositorio GIT.

Luego se procederá a la configuración del entorno, en la siguiente figura se crea el usuario "***asterisk***" y el directorio raíz donde se encontrará el acceso a la interfaz de FreePBX.

```
# adduser asterisk --disabled-password --no-create-home --home /var/lib/asterisk --shell /sbin/nologin --gecos "Asterisk User"
# mkdir /var/www/html
```

Figura 10.9: Comandos para la creación del usuario "asterisk" y del directorio raíz de FreePBX.

Se ajustan valores del servidor de apache para que se ejecute con el usuario "***asterisk***" recién creado y aumentar la capacidad máxima de carga de archivos a 20 MB.

```
# sed -i 's/\(APACHE_RUN_USER=\|APACHE_RUN_GROUP=\)\(.*\)/\1asterisk/g' /etc/apache2/envvars
# sed -i 's/\(^upload_max_filesize = \).*/\120M/' /etc/php5/apache2/php.ini
# service apache2 restart
```

Figura 10.10: Comandos para los ajustes del servidor apache.

Se configura la base de datos para Asterisk en MySQL, en la siguiente figura se utiliza la variable ***ASTERISK_DB_PW*** para establecer la contraseña de la BD de Asterisk, la cual por defecto es **"*amp109*"**. En estos comandos de ***mysqladmin*** y ***mysql*** se asume que root no posee contraseña, si se estableció una contraseña al ejecutar el comando de la Figura 10.2 debe incorporar el parámetro **"*-p*"** a los comandos relacionados para que le sea solicitado.

```
# cd /usr/src/freepbx/
# export ASTERISK_DB_PW=amp109
# mysqladmin -u root create asterisk
# mysqladmin -u root create asteriskcdrdb
# mysql -u root asterisk < SQL/newinstall.sql
# mysql -u root asteriskcdrdb < SQL/cdr_mysql_table.sql
# mysql -u root -e "GRANT ALL PRIVILEGES ON asterisk.* TO
asteriskuser@localhost IDENTIFIED BY '${ASTERISK_DB_PW}';"
# mysql -u root -e "GRANT ALL PRIVILEGES ON asteriskcdrdb.* TO
asteriskuser@localhost IDENTIFIED BY '${ASTERISK_DB_PW}';"
# mysql -u root -e "flush privileges;"
```

Figura 10.11: Comandos para configurar la base de datos de Asterisk en MYSQL.

Luego se procede a la instalación de FreePBX con los comandos que se muestran en la Figura 10.12.

```
# cd /usr/src/freepbx*
# ./start_asterisk start
# ./install_amp
# amportal a ma installall
# amportal a reload
# ln -s /var/lib/asterisk/moh /var/lib/asterisk/mohmp3
# amportal start
```

Figura 10.12: Comandos para instalar FreePBX.

Para configurar que FreePBX se ejecute al iniciar el servidor debe modificar el archivo ***/etc/rc.local*** y agregar **"*/usr/local/sbin/amportal start*"** antes del **"*exit 0*"** que se encuentra allí, como se muestra en la Figura 10.13.

```
# ...
/usr/local/sbin/amportal start
exit 0
```

Figura 10.13: Configuración para que FreePBX inicie con el sistema.

Ahora es necesario verificar que en el archivo de configuración ***/etc/apache2/sites-available/default*** la raíz del servidor web sea ***/var/www/html***, es decir, que se encuentre como se muestra en la Figura 10.14 tanto el ***DocumentRoot*** como la etiqueta ***Directory*** correspondiente.

```
# ...
DocumentRoot /var/www/html
     # ...
     <Directory /var/www/html/>
          # ...
     </Directory>
# ...
```

Figura 10.14: Modificación de la raíz del servidor web.

Si se realizaron cambios en el archivo antes mencionado debe reiniciar el servicio web con el comando **"*service apache2 restart*"** para que se apliquen los cambios.

Como todos los comandos fueron ejecutados bajo el usuario ***root*** es necesario asignar los directorios al usuario asterisk mediante el comando ***chown*** como se muestra en la siguiente figura.

```
# chown asterisk. /var/run/asterisk
# chown -R asterisk. /etc/asterisk
# chown -R asterisk. /var/{lib,log,spool}/asterisk
# chown -R asterisk. /usr/lib/asterisk
# chown asterisk. /var/lock/apache2
# chown -R asterisk. /var/www/
```

Figura 10.15: Comandos para ajustes de permisos al usuario asterisk.

Para acceder a la interfaz web de FreePBX debe ingresar la dirección IP del servidor en el navegador (**<http://DIRECCION_IP_SERVIDOR_FREEPBX>**), se le solicitará registrar unas credenciales para el administrador el sistema, recuerde documentar esta información, luego aparecerá tres opciones: "***FreePBX Administration***", "***User Control Panel***" y "***Get Support***", seleccionando la primera opción (*FreePBX Administration*) se le solicitará las credenciales de administrador y una vez autenticado ya podrá hacer uso de FreePBX.

Para el máximo provecho de FreePBX es necesario realizar una actualización del sistema, a continuación se describen los pasos a realizar:

- Seleccione **"Admin"** > "**Module Admin**" en el menú principal.
- Haga click en "**Check Online**".
- Haga click en "**Download All**" y en "**Upgrade All**".
- Haga click en "**Process**".
- Se mostrará un resumen de los módulos que se instalarán, haga click en "**Confirm**" si está de acuerdo.
- Aparecerá una ventana emergente mostrando el estatus de la instalación, cierre la ventana emergente.
- Por último seleccione la opción "**Apply Config**" que se encuentra en el menú principal, esto reiniciará los servicios y una vez finalizado los cambios se encontrarán aplicados.

Con esto se ha culminado el proceso de instalación de Asterisk 11 + FreePBX 2.11 sobre Debian 6.0.

11 Citas Bibliográficas

[1] A. Prieto y Ángel. Rodríguez, **Sistema de historia médica electrónica, programa SOS telemedicina para Venezuela**. Tesis de grado. Universidad Central de Venezuela. 2012.

[2] A. Gómez, **Apoyo tecnológico en el diseño, desarrollo, instalación y configuración de aplicaciones, programa sos telemedicina para Venezuela**. Tesis de grado. Universidad Católica Andrés Bello. Febrero, 2013.

[3] V. Rodríguez, A. Bay, **Sistema móvil de referencias médicas. Programa sos telemedicina para venezuela.** Tesis de grado. Universidad Central de Venezuela. Mayo, 2014.

[4] A. Morales, V. Rodríguez, A. Bay, **Aplicación Web Móvil Integrada al Sistema de Referencias Médicas del Programa SOS Telemedicina para Venezuela**. Segunda Conferencia Nacional de Computación, Informática y Sistemas (CoNCISa 2014). Sesión de Gestión de Tecnologías de Información. ISBN: 978-980-7683-00-5. Del 1 al 3 de octubre 2014. Universidad Católica Andrés Bello (UCAB), Caracas, Venezuela.

[5] businesswire.com, **Strategy Analytics: Worldwide Smartphone Population Tops 1 Billion in Q3 2012**, [En línea]. [Consulta: 03 de Diciembre de 2012]. Disponible en: <http://www.businesswire.com/news/home/20121017005479/en/Strategy-Analytics-Worldwide-Smartphone-Population-Tops-1>.

[6] A. Charlton, **Android Holds 46.9% Smartphone Market Share**, [En línea]. [Consulta: 03 de Diciembre de 2012]. Disponible en: <http://blog.net-essence.co.uk/2012/06/21/android-holds-46-9-smartphone-market-share>. Junio, 2012.

[7] El Centro de Análisis de Imágenes Biomédicas Computarizadas CAIBCO de la UCV, **Proyecto SOS Telemedicina**, [En línea]. [Consulta: 25 de Marzo de 2015]. Disponible en: http://caibco.ucv.ve/SOSTelemedicina.pdf

[8] A. Kopec, A. Salazar, **Aplicaciones de telecomunicaciones en salud en la subregión andina**: Telemedicina. Washington DC: Organización Panamericana de la Salud, OPS/OMS; 2002.

[9] C. Ruiz, A. Zuluaga, A. Trujillo, **Telemedicina: Introducción, aplicación y principios de desarrollo**. Rev CES Med. 2007.

[10] A. Poliszuk, A. Salazar, **Telemedicina**. Aplicaciones De Telecomunicaciones En Salud En La Subregión Andina. (Sin fecha).

[11] A. Díaz, **Telemedicina. Avances más recientes en el mundo**, [En línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <http://neutron.ing.ucv.ve/fernandezl/Multimedia/Tareas%202004-3/telemedicina-AlbertoDiaz.pdf>. (Sin fecha).

[12] R. Valero, J. Pérez, **Telemedicina y sus aplicaciones**, [En línea]. [Consulta: 03 de Diciembre de 2012]. Disponible en: <http://es.scribd.com/doc/53198255/Telemedicina-y-sus-aplicaciones>. (Sin fecha).

[13] Video Development Initiative (ViDe) et al., **Videoconferencing Cookbook - Version 4.1**, [En línea]. [Consulta: 03 de Diciembre de 2012]. Disponible en: <http://www.vide.net/cookbook/cookbook.en>. Abril, 2012.

[14] N. Unuth, **3G Technology – What Is 3G?** [En línea]. [Consulta: 03 de Diciembre de 2012]. Disponible en: <http://voip.about.com/od/mobilevoip/p/3G.htm>. (Sin fecha).

[15] J. Janak, **SIP Introduction**, [En línea]. [Consulta: 03 de Diciembre de 2012]. Disponible en:<http://ftp.iptel.org/pub/ser/0.8.14/doc/html/sip_introduction.html>. 2003 FhG FOKUS.

[16] J. Rosenberg, H. Schulzrinne, G. Camarillo, et al., **SIP: Session Initiation Protocol**. RFC 3261. June, 2002.

[17] L. Peterson, B. Davie, **Computer Networks – A Systems Approach**. San Francisco – United States. 2007.

[18] H. Schulzrinne, S. Casner, R. Frederick, et al., **RTP: A Transport Protocol for Real-Time Applications**. RFC 3550. Julio, 2003.

[19] C. Perkings, **RTP: Audio and Video for the Internet**. Addison-Wesley Professional. Junio, 2003.

[20] I. Richardson, **The H.264 Advanced Video Compression Standard – Second Edition**. A John Wiley and Sons, Ltd., Publication. Agosto, 2011.

[21] J. Ostermann, J. Bormans, P. List, et al., **Video coding with H.264/AVC: Tools, Performance, and Complexity**. IEEE Circuits and Systems Magazine. 2004.

[22] The Hydrogen audio Knowledgebase, **Advanced Audio Coding**, [En línea]. [Consulta: 03 de Diciembre de 2012]. Disponible en: <http://wiki.hydrogenaudio.org/index.php?title=AAC>. (Sin fecha).

[23] The Hydrogen audio Knowledgebase, **MP3**, [En línea]. [Consulta: 03 de Diciembre de 2012]. Disponible en: <http://wiki.hydrogenaudio.org/index.php?title=Mp3>. (Sin fecha).

[24] L. Rinsing, N. Janoff, **The Scrum Software Development Process for Small Teams**. Software, IEEE. Agosto, 2000.

[25] D. Ehringer, **The Dalvik Virtual Machine Architecture**; [En línea]. [Consulta en: 16 de Abril de 2015]. Disponible en: <http://www.davidehringer.com/software/android/The_Dalvik_Virtual_Machine.pdf>. Marzo 2010.

[26] J. Martínez, **Fundamentos de Programación en Java,** [En línea]. [Consulta: 25 de Marzo de 2015]. Disponible en: <http://pendientedemigracion.ucm.es/info/tecnomovil/documentos/fjava.pdf>. (Sin fecha).

[27] PHP.net, **PHP**, [En línea]. [Consulta: 25 de Marzo de 2015]. Disponible en: <http://php.net>. (Sin fecha).

[28] IBM Knowledge Center, **SQL**. [En Línea]. [Consulta: 25 de Marzo 2015]. Disponible en: <http://www-01.ibm.com/support/knowledgecenter/SSPK3V_6.3.0/com.ibm.swg.im.soliddb.sql.doc/doc/tables.rows.and.columns.html>. (Sin fecha).

[29] Trello.com, **Trello**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <https://trello.com/>. (Sin fecha).

[30] Bitbucket.org, **Atlassian Bitucket**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <https://bitbucket.org>. (Sin fecha).

[31] Asterisk.org, **Asterisk**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <http://www.asterisk.org>. (Sin fecha).

[32] Wampserver.com, **WampServer**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: < http://www.wampserver.com/en/>. (Sin fecha).

[33] json.org, **JSON**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <http://www.json.org>. (Sin fecha).

[34] M. Elkstein, **Learn REST: A Tutorial**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <http://rest.elkstein.org>. (Sin fecha).

[35] Doubango Telecom, **Doubango**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <http://www.doubango.org>. (Sin fecha).

[36] slimframework.com, **Slim**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <http://www.slimframework.com>. (Sin fecha).

[37] wiki.freepbx.org, **Wiki FreePBX**. [En Línea]. [Consulta en: 25 de Marzo 2015]. Disponible en: <http://wiki.freepbx.org>. (Sin fecha).