UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS E INGENIERÍA

DESARROLLO DE UNA APLICACIÓN PARA LA GENERACIÓN Y ANÁLISIS DE TRÁFICO DE VOZ SOBRE IP MÓVIL (MVOIP).

TESIS
PARA OBTENER EL GRADO DE

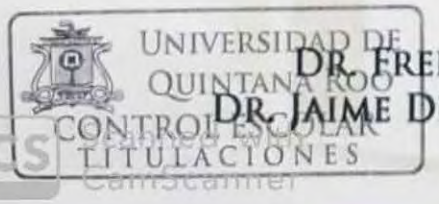
INGENIERÍA EN REDES

PRESENTA
LANDY ISABEL VARGAS COOT

DIRECTOR DE TESIS
DR. HOMERO TORAL CRUZ

ASESORES
M.T. MARTÍN ANTONIO SANTOS ROMERO
LIC. JORGE RICARDO GÓMEZ MONTALVO

SUPLENTES
DR. FREDDY IGNACIO CHAN PUC
DR. JAIME DIONISIO CUEVAS DOMÍNGUEZ



CHETUMAL QUINTANA ROO, MÉXICO, NOVIEMBRE DE 2019

UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS E INGENIERÍA

TRABAJO DE TESIS TITULADO
"Desarrollo de una Aplicación para la Generación y Análisis de Tráfico de Voz sobre IP Móvil (mVoIP)."

ELABORADO POR
Landy Isabel Vargas Coot

BAJO SUPERVISIÓN DEL COMITÉ DE ASESORÍA Y APROBADO COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE

INGENIERÍA EN REDES

COMITÉ DE TESIS

DIRECTOR: _____
Dr. Homero Toral Cruz

ASESOR: _____
M.T. Martín Antonio Santos Romero

ASESOR: _____
Lic. Jorge Ricardo Gómez Montalvo

SUPLENTE: _____
Dr. Freddy Ignacio Chan Puc

SUPLENTE: _____
Dr. Jaime Dionisio Cuevas Domínguez

Universidad de Quintana Roo
DCI División de Ciencias e Ingeniería

UNIVERSIDAD DE QUINTANA ROO
CONTROL ESCOLAR
TITULACIONES

CHETUMAL QUINTANA ROO, MÉXICO, NOVIEMBRE DE 2019

Agradecimientos

A mi familia, a mi hermana Karla por haberme apoyado en todo momento y haber soportado mi mal humor en muchas ocasiones, a mis padres por darme consuelo en cada momento, a mi tía que siempre creyó en mí y haberme animado a seguir.

A mis maestros por haberme enseñado a amar esta carrera, en especial al Dr. Homero por haberme apoyado y tenido paciencia a lo largo de esta tesis.

Dedicatoria: A mi familia.

Resumen

En este trabajo se presenta el desarrollo de una aplicación móvil bajo el sistema aperativo Android y el lenguaje de programación orientado a objetos Java. La aplicación desarrollada permite realizar llamadas reales de voz sobre el protocolo de Internet (VoIP) en una red inalambrica.

La aplicación propuesta, se implementó bajo el protocolo de inicio de sesión (SIP) y permite establecer comunicación y centralizar las llamadas mediante un servidor comercial Elastix. Se implementaron los codificadores de voz para telefonía G.711 u-law y G.711 a-law; además, la aplicación tiene la capacidad de evaluar en tiempo real los principales parámetros que determinan la calidad de servicio (QoS) o métricas de desempeño; tales como: jitter, retardo en un sentido (OWD), tasa de pérdida de paquetes (PLR), MOS (Mean Opinion Score) y factor R del Modelo E. También, permite guardar los parámetros de QoS en un base de datos local y exportarlos en formato .CSV a la memoria externa del teléfono.

Para el desarrollo de la aplicación se realizó el diseño de una interfaz de usuario dinámica mediante el uso de “Fragments”, funcional para “Smartphones” y “Tablets”.

Contenido

Índice de Ilustraciones

CAPÍTULO 1

Capítulo 1 INTRODUCCIÓN

Como resultado de la rápida evolución de las tecnologías inalámbricas y dispositivos finales (smartphones, tablets PC, PDAs), surge una nueva e interesante variante dentro de la tecnología VoIP, la cual consiste en la transmisión de llamadas telefónicas a través de dispositivos móviles sobre una red inalámbrica IP. Donde la tecnología de voz sobre IP móvil, resulta considerablemente menos costosa respecto a la tecnología celular. (A. Sfairopoulou, 2011)

Cuando el tráfico de voz es transportado sobre redes IP, la transmisión basada en paquetes puede introducir deterioros y tener influencia en la calidad de servicio percibida por el usuario final. La calidad de la voz en la tecnología voz sobre IP y voz sobre IP móvil (mVoIP) depende de diversos parámetros de QoS. Particularmente, el jitter, one way delay (OWD) y tasa de pérdida de paquetes (PLR) tienen un importante impacto en la calidad de la voz. Estos parámetros están estrechamente relacionados unos con otros y pueden ser usados para configurar otros parámetros (longitud de los datos de voz, tipo de codec, tamaño de redundancia FEC, y tamaño del de-jitter buffer) a valores óptimos para de calidad de voz.

Por otro lado, la alta demanda por parte de los usuarios de servicios multimedia a grandes velocidades con movilidad, y que, además; éstos sean ofrecidos con aceptable nivel de calidad de servicio, hace que hoy en día se tengan que desarrollar esquemas de transmisión de datos capaces de cumplir con las demandas de los usuarios. Para cubrir la primera demanda, referente a altas velocidades con movilidad, es necesario realizar la migración de las aplicaciones de video, voz y datos hacia las redes inalámbricas de próxima generación (4G). Por otro lado, para cubrir la segunda demanda, relacionada a QoS, es necesario realizar un estudio de los principales parámetros de desempeño de red que determinan la calidad de servicio en aplicaciones VoIP, donde, los principales objetivos de dicho estudios son: (1) realizar análisis y evaluación de desempeño mediante mediciones de red; (2) caracterizar el comportamiento del tráfico de red; (3) implementar modelos representativos del tráfico; e (4) implementar mecanismos de QoS.

Motivados por los puntos mencionados anteriormente, en este proyecto se desarrolló una aplicación para la generación de tráfico real de voz sobre IP móvil, la cual permite evaluar los principales parámetros que determinan la QoS (jitter, OWD y PLR) y el desempeño de una red mVoIP, mediante el mean opinion score (MOS) (ITU-T P. 800, 1996) y el factor R propuesto en el modelo E (ITU-T G.107, 2009). La aplicación propuesta se desarrollò bajo el sistema operativo Android, el protocolo SIP y permite realizar llamadas reales de voz mediante los esquemas de codificación G.711 u-law y G.711 a-law (UIT-T G.711, 1993).

1.1 Definición del problema:

Dentro de la evolución de los sistemas de comunicaciones de voz, surge una nueva e interesante variante dentro de la tecnología VoIP, la cual consiste en la transmisión de llamadas telefónicas a través de dispositivos móviles sobre las redes inalámbricas IP (mVoIP). Donde la tecnología de voz sobre IP móvil, resulta muy atractiva por presentar un menor costo respecto a la telefonía celular. Sin embargo, en la transmisión de tráfico de voz sobre las redes IP, la calidad de servicio no esta garantizada, debido a que Internet proporciona un servicio de mejor esfuerzo (best effort) y la presencia de altos umbrales de retardos, jitter y pédida de paquetes es altamente probable cuando la red sufre congestiones por la alta demanda de recursos de los diversos usuarios de Internet.

Una alternativa para realizar el monitoreo de las métricas de desempeño en aplicaciones mVoIP y tener un mejor control de la QoS de las llamadas telefónicas, es mediante una aplicación que permita evaluar los principales parámetros que determinan la QoS y el desempeño mediante el MOS y el factor R del modelo E.

1.2 Justificación:

Son muchas las ventajas que se tiene al momento de utilizar la tecnología mVoIP, entre las más importantes (Toral-Cruz, Pathan, & Ramírez Pacheco, 2019), podemos mencionar: integración de servicios de voz y datos sobre una misma infraestructura, alta movilidad de los usuarios, escalabilidad flexible al incrementar el ancho de banda, reducción de costos, uso óptimo del ancho de banda, etc. Por otro lado, disponer de una aplicación que permita realizar

llamadas telefónicas reales sobre una red inalámbrica IP, monitorear las métricas de desepeño y evaluar la calidad de servicio en tiempo real, es de alta utilidad para poder realizar análisis y caracterización de tráfico en redes mVoIP.

Con base en lo mencinado anteriormente, surge el interés de desarrollar la aplicación propuesta en el presente proyecto de tesis.

1.3 Objetivo General:

Desarrollar una aplicación para la generación y análisis de tráfico de la tecnología de Voz sobre IP Móvil (mVoIP).

1.4 Objetivos Particulares:

- Diseñar una interfaz de usuario dinámica mediante el uso de “Fragments”, funcional para “Smartphones” y “Tablets”.
- Implementar una aplicación Android para generar tráfico mVoIP bajo el CODEC más popular en telefonía (G.711 u-law y G.711 a-law).
- Implementar el sistema de señalización SIP estipulado en el RFC 3261 de la organización internacional abierta de normalización “Internet Engineering Task Force (IETF)”.

CAPÍTULO 2

Capítulo 2 CONCEPTOS BÁSICOS DE LA TECNOLOGÍA VoIP

2.1 Voz sobre el Protocolo de Internet

VoIP es la tecnología que nos permite transmitir voz a través de redes de datos bajo la pila de protocolos TCP/IP. Un sistema VoIP convierte las señales de voz analógicas en paquetes, los cuales son transportados a través de redes de datos o redes IP que no garantizan la calidad de servicio.

2.2 Protocolos de señalización

Para el establecimiento de llamadas telefónicas a través de redes de datos, los protocolo de señalización y control de llamada son de suma importancia, porque permite a los diversos componentes de la red, comunicarse entre sí, para crear, modificar y terminar llamadas. Dos importantes protocolos de señalización son H.323 y SIP. (Toral-Cruz, Ramirez-Pacheco, Velarde-Alvarado, & Pathan, 2013)

En el caso de la telefonía IP, se puede definir una llamada como la sesión multimedia entre dos o más participantes, mientras que la señalización asociada con una llamada se denomina conexión (Joskowicz, 2013). El rol de un protocolo de señalización puede desglosarse principalmnte en cuatro funciones:

- Ubicación del usuario: La persona que llama debe encontrar primero la ubicación de los llamados.
- Establecimiento de la sesión: Se decide sobre aceptar, rechazar o redirigir la llamada.
- Negociación de la sesión: Los puntos finales deben acordar un conjunto de propiedades para la sesión.
- Gestión de los participantes: Permite a los puntos finales unirse o abandonar una sesión existente.

2.2.1 H.323

La primera versión del protocolo de control/señalización denominado H.323, fue ratificada en 1996, varias enmiendas han sido propuestas y adoptadas, de hecho, H.323 no es un protocolo único. Es más bien un conjunto de protocolos diferentes que se utilizan para audio, video y aplicaciones de datos.

H.323 consta de tres tipos de interacciones. La primera se refiere a las operaciones administrativas de un usuario y se lleva a cabo utilizando la señalización de registro, admisión y estado (RAS). El segundo es responsable de la señalización sobre la configuración de la llamada. Para ello se emplea el protocolo H.225. El tercer tipo de interacción se dirige a la negociación de capacidades del terminal y al control de llamadas y está cubierta en H.245.

Las características de H.323 se pueden categorizar en tres clases:

- Características locales: Incluyen historial de llamadas, libreta de direcciones.
- Funciones basadas en la red: Incluyen autorización, admisión, las características que se requieren controlar.
- Servicios: Consiste en un conjunto suplementario de servicios que requieren una señalización especial. Los servicios suplementarios son características percibidas por el usuario. Como reenvío de llamadas, transferencia de llamadas y terminación de llamadas.

2.2.2 SIP

El protocolo de inicio de sesión (SIP) es un protocolo de la capa de aplicación que fue inicialmente desarrollado por el Grupo de Trabajo de Control de Sesión Multimedia Multipartita de la IETF (MMUSIC WG) en 1999 y actualizado por el SIP WG en 2002. SIP, que está delineado en RFC 3261, se utiliza para crear, modificar y terminar sesiones con uno o más participantes, y fue diseñado para ser independiente del protocolo de transporte subyacente. Como en H.323, las características de SIP también se clasifican en tres categorías similares, a saber, (1) características locales, (2) características basadas en la red, como autorización y (3) servicios suplementarios.

Las principales funciones de este protocolo de señalización son: (i) localización de recursos / partes; (ii) sesiones de invitación a servicio; y (iii) negociación de parámetros de servicio. Para el transporte de información sobre el contenido multimedia de la sesión SIP se basa en el protocolo de descripción de sesión (SDP).

SIP es similar a HTTP (HyperText Transfer Protocol) y comparte algunos de sus principios de diseño. En particular, adopta una arquitectura cliente/servidor (petición/respuesta) en la que las peticiones son generadas por el cliente y enviadas al servidor. El servidor procesa las solicitudes y envía una respuesta al cliente. Al igual que HTTP, SIP se basa en mensajes que son peticiones o respuestas intercambiadas entre los clientes y los servidores.

Los tipos más importantes de solicitudes son la solicitud INVITE que se utiliza para invitar a un usuario a una llamada, el ACK que envía al llamante al receptor para simplemente acusar recibo de la respuesta del último y la petición BYE que se utiliza para terminar el conexión entre dos usuarios en una sesión (ver Ilustración 1). Además de estos tipos, se pueden identificar otros tres tipos de solicitudes: CANCEL, OPTIONS y REGISTER. La solicitud CANCEL se utiliza para anular cualquier búsqueda pendiente de un usuario; sin embargo, no destruye una llamada en curso. La solicitud OPTIONS solo consulta las capacidades de los servidores. Finalmente, la solicitud REGISTER se emplea para registrar un usuario con un servidor SIP.

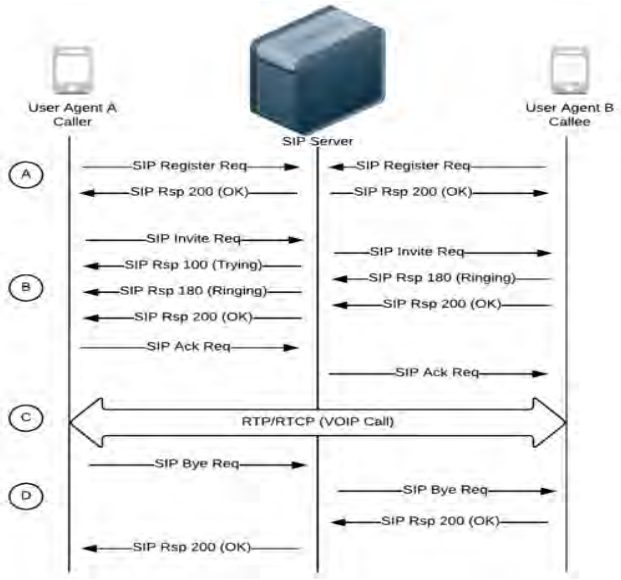


Ilustración 1. Diagrama de solicitudes

2.2.2.1 Arquitectura SIP

Los principales componentes de SIP son los agentes de usuario y los servidores de redes (ver Ilustración 2).

- Los agentes de usuario: son los puntos finales SIP que realizan y reciben llamadas. Un agente de usuario puede funcionar en dos modos:
  - Cliente de Agente de usuario (UAC) : Inicia peticiones SIP.
  - Servidor de Agente de Usuario (UAS): Que recibe solicitudes y responde en nombre del usuario.

  Un punto final SIP (por ejemplo, un teléfono IP) puede actuar tanto como un UAC o como un UAS. Sin embargo, sólo funciona como uno u otro por transacción, dependiendo de si inició o no la solicitud.

En lo que respecta a los servidores de red, existen cuatro tipos diferentes de ellos en una red SIP: servidores proxy, servidores de redireccionamiento, servidores de ubicación y servidores de registro.

- Servidor Proxy: Recibe las solicitudes generadas por los agentes de usuario y decide a qué servidor se debe reenviar una solicitud. Normalmente, una solicitud recorrerá muchos servidores antes de llegar a su destino.
- Servidor de Redireccionamiento: Es diferente de un servidor proxy. Un servidor de redireccionamiento no envía solicitudes. Más bien, notifica a la parte que llama de la ubicación del receptor. Para ello, contacta con un servidor de ubicación que guarda información sobre la posible ubicación de la parte llamada.
- Servidores de Registro: Aceptan solicitudes de REGISTRO de los agentes de usuario y normalmente se ubican conjuntamente con servidores proxy o servidores de redireccionamiento. Finalmente como en el caso de la arquitectura H.323, también se puede emplear un gateway para conectar puntos finales SIP con otros tipos de terminales.

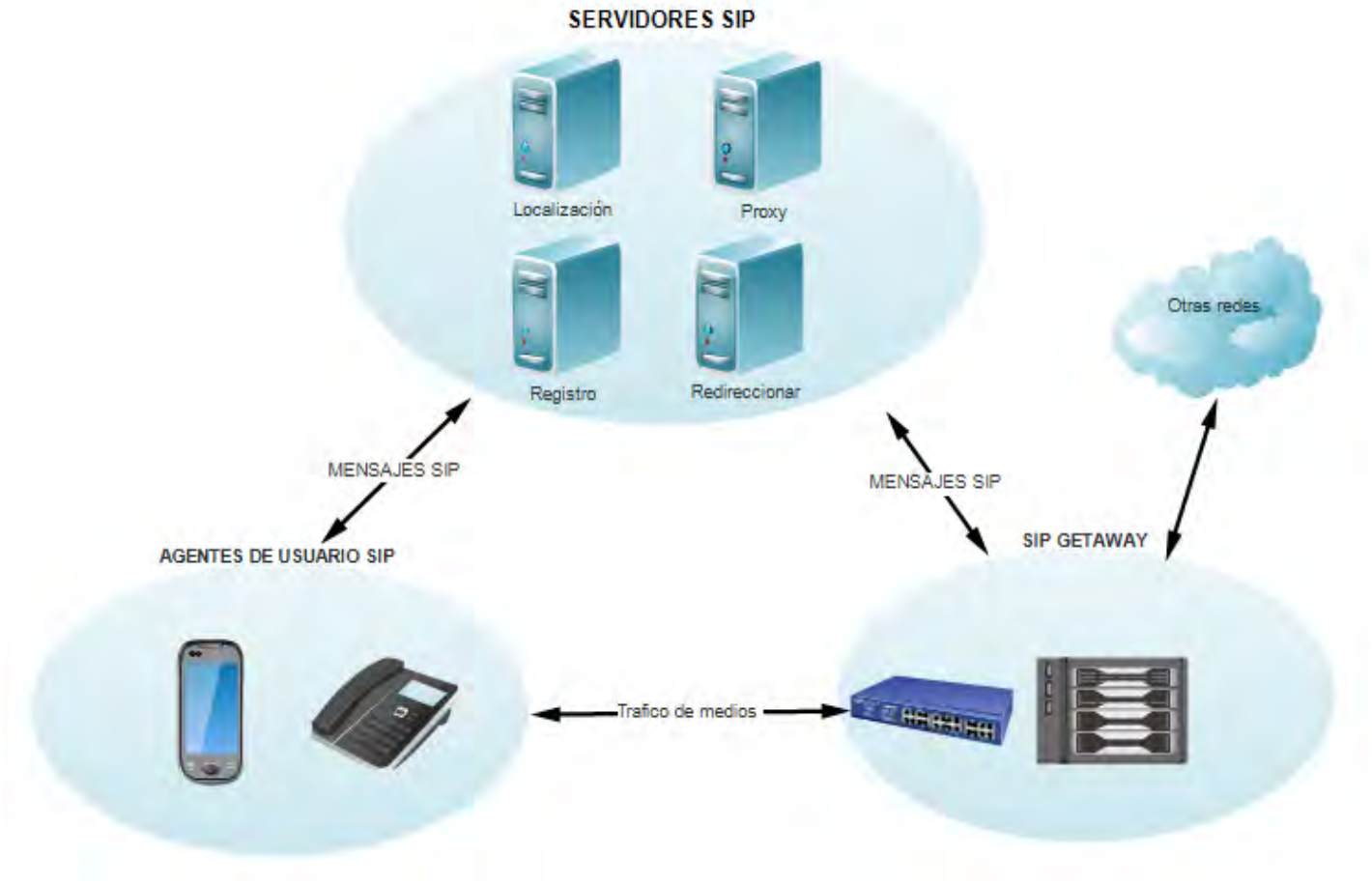


Ilustración 2. Arquitectura SIP

2.2.2.2 Etapas de Llamadas

En la Ilustración 3, se ejemplifica cómo se establece y finaliza una llamada en una arquitectura SIP. La parte llamante primero envía una solicitud INVITE a la parte llamada. Esta solicitud atraviesa algunos servidores proxy antes de llegar a su destino.

Cada servidor proxy que recibe esta solicitud devuelve un mensaje de respuesta de 180 TRYING, indicando que la solicitud está siendo procesada. Cuando la UAS de la parte llamada recibe la solicitud, comienza a sonar y luego envía una respuesta de 180 RINGING a la UAC de la parte que llama.

Los servidores proxy que reciben la respuesta 180 RINGING lo remiten a la parte llamante. Cuando la llamada es aceptada por el destinatario, se envía una respuesta de 200 OK a la parte

que llamante. El agente de usuario llamante responde a ese mensaje con un ACK que se envía directamente al receptor. Entonces las dos partes pueden comenzar la comunicación. Si el destinatario hubiera decidido rechazar la llamada, se habría enviado un mensaje 603 DECLINE a la persona llamante (cuando está ocupado, se envía una respuesta 600 BUSY). Si el ususario llamante desea colgar, se envía una solicitud BYE a través de los servidores intermedios proxy al receptor, y este último responde con una respuesta de 200 OK para indicar que la solicitud cumplió su propósito.

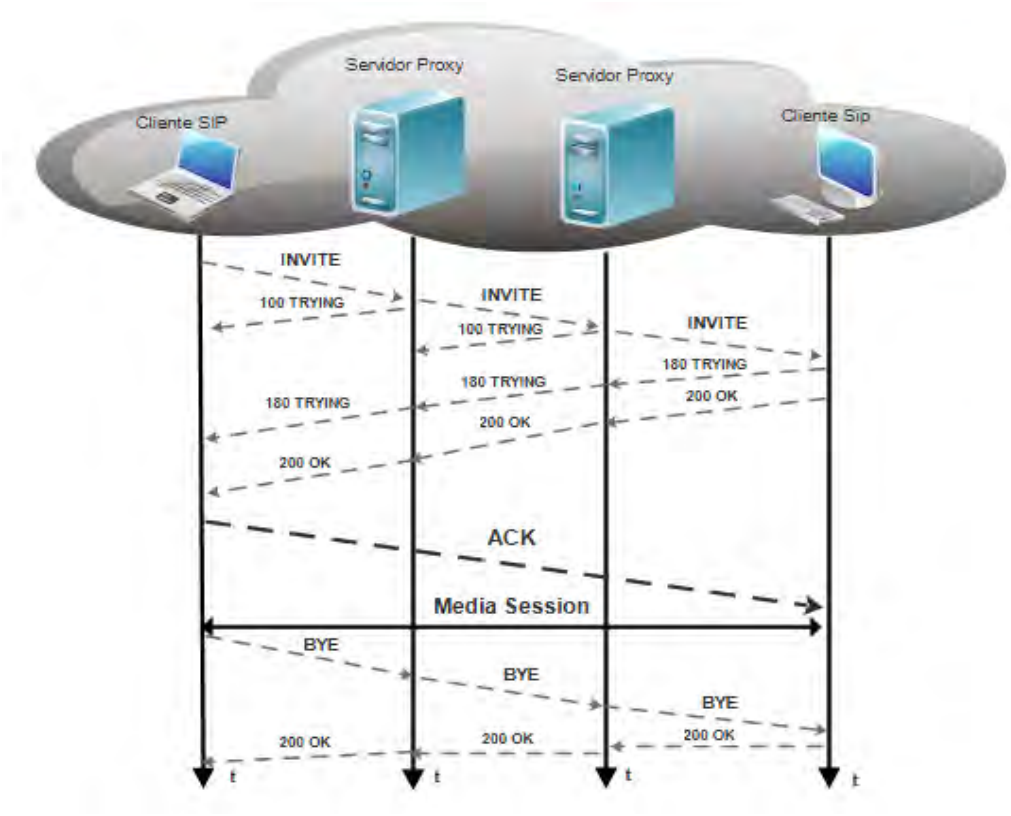


Ilustración 3. Establecimiento y finalización de llamada en una arquitectura SIP

2.3 Codificadores de Voz

Los codificadores de voz son los algoritmos que permiten al sistema convertir la voz analógica a digital. Existe una gran variedad de codificadores de voz, los cuales varían de acuerdo a su complejidad algorítmica, tasa de transmisión y calidad de voz. Mientras mayor sea la tasa de transmisión, mayor será la calidad de la voz.

2.3.1 Tipos de Codificadores

2.3.1.1 G.711

G.711 es un esquema de codificación, mejor conocido como modulación por impulsos codificados (PCM) que produce un valor de 8 bits cada 125 µs, resultando en un flujo de bits de 64 kb/s (UIT-T G.711, 1993). Cada dato de audio se codifica como ocho bits después del escalamiento logarítmico. Este estándar tiene dos formas, µ-Law (usada en Norteamérica y Japón) y A-Law (uso en Europa y el resto del mundo).

Un codificador PCM A-Law G.711 convierte muestras PCM lineales largas de 13 bits en muestras PCM comprimidas de 8 bits (forma logarítmica), y el decodificador realiza la conversión viceversa, mientras que un codificador PCM de µ-Law G.711 convierte 14 bits lineales PCM en muestras PCM comprimidas de 8 bits. G.711 es el codificador estándar utilizado en desde los orígenes de la red telefónica publica conmutada (PSTN)

2.3.1.2 G.723.1

G.723.1 es un estándar de códec de voz de doble velocidad del ITU-T, desarrollado originalmente para teléfonos de vídeo que ofrecen video y voz sobre líneas telefónicas regulares (PSTN) (ITU-T G.723.1, 2006). Se normalizó en 1996 como parte de la familia H.324 de estándares y puede operar a dos velocidades binarias:

- 6.3 kb/s (utilizando bloques de 24 bytes) utilizando un algoritmo de cuantificación de máxima verosimilitud de pulso (MPC-MLQ).
- 5.3 kb/s (utilizando trozos de 20 bytes) utilizando un algoritmo de predicción lineal estimulada por el código algébraico (ACELP).

2.3.1.3 G.726

El ITU-T G.726 reemplazó al UIT-T G.723. Funciona a cuatro tasas de bits, es decir, 16, 24, 32 y 40 kb/s. Específicamente, se recomienda este códec para la conversión de un solo canal PCM de µ-law o A-law de 64 kb/s codificado a 8 kHz a un canal de 16, 24, 32 ó 40 kb/s (ITU-T G.726,

1990). Funciona con base en el principio de ADPCM. No obstante, las velocidades de codificación de 16 y 24 kb/s no proporcionan una buena calidad.

Por lo tanto, el ITU-T G.726 recomienda que las velocidades de 16 y 24 kb/s se alternen con una codificación de velocidad de datos más alta para proporcionar un tamaño de muestra promedio de entre 3.5 y 3.7 bits por muestra.

2.3.1.4 G.728

ITU-T G.728 describe un códec de voz de retardo bajo para la codificación de señales de voz a 16 kb/s utilizando predicción lineal excitada de código de bajo retardo (LD-CELP) (ITU-T G.728, 2012).

G.728 El anexo G (G.728 G) es una especificación de punto fijo del códec que trabaja a una velocidad binaria de 16 kb/s. El Anexo I de G.728, es la técnica de ocultación de pérdida de paquetes (PLC) utilizada junto con el códec G.728. Se trata de un codec de voz muy robusto, con muy buena calidad de voz, comparable a 32 kb/s ADPCM.

2.4 Protocolos de Internet

La pila de protocolos empleadas en VoIP, consiste en técnicas/protocolos de la capa de aplicación (RTP y RTCP), la capa de transporte (TCP o UDP), la capa de red (IP) y la capa física/enlace.

TCP y UDP son los protocolos de la capa de transporte más utilizados. TCP es un protocolo de transporte orientado a la conexión y confiable. Sus características tales como retransmisión, control de flujo y control de congestión no son adecuadas para aplicaciones multimedia en tiempo real como VoIP, debido a que puede introducir retardos no deseados. UDP es un protocolo de transporte orientado a la no conexión y no fiable. Su simple encabezado, y la no retransmisión, lo hacen adecuado para aplicaciones en tiempo real.

El protocolo de transporte en tiempo real (RTP) se desarrolló para ayudar a la transferencia de flujos de medios en tiempo real sobre el protocolo UDP no fiable. RTP cuenta con doce campos, los cuales se describen en el RFC 3550 (RFC 3550, 2003). El protocolo de control RTP (RTCP)

asociado, también se desarrolló para ayudar al control de medios y la gestión de QoS / QoE para aplicaciones VoIP. En este capítulo se presentan los conceptos claves de RTP y RTCP, junto con el análisis detallado de sus encabezados. También se discuten los problemas de eficiencia de ancho de banda y RTP comprimidos (cRTP)

2.4.1 IP

El protocolo de capa de red, Protocolo de Internet (IP), es responsable de la transmisión de paquetes IP desde el remitente hasta el receptor a través de Internet y se refiere principalmente a dónde enviar un paquete y a cómo encaminar paquetes a través de la mejor ruta a través de Internet (con respecto a los protocolos de enrutamiento). La Ilustración 4 muestra los encabezados de la versión IPv4.

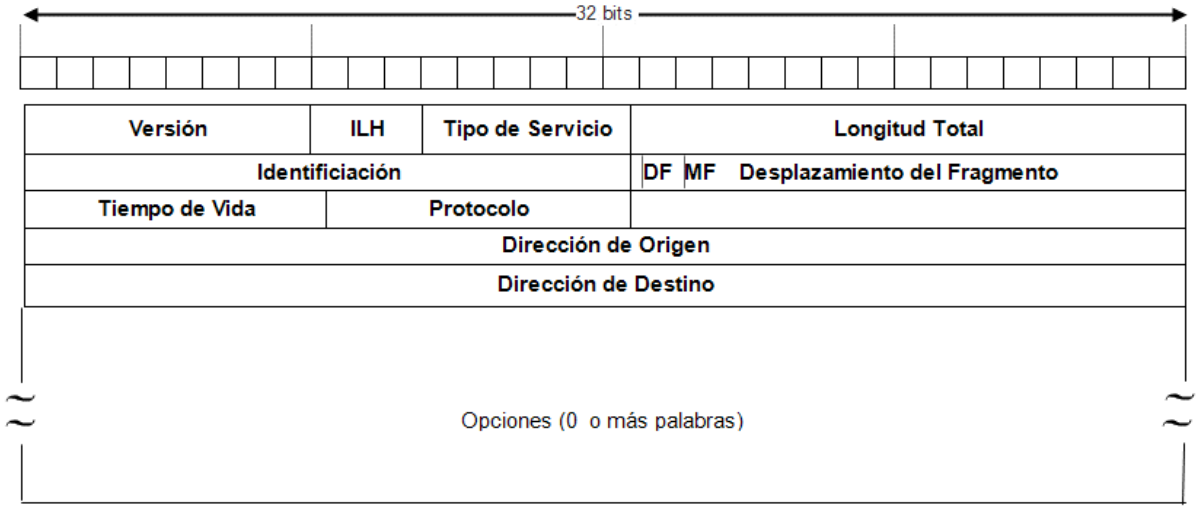


Ilustración 4 Cabecera IPv4

2.4.2 TCP

En la pila de protocolos TCP/IP, hay dos protocolos de la capa de transporte, el protocolo de control de transmisión o protocolo de control de transporte (TCP) y el protocolo de datagramas de usuario (UDP). Alrededor del 80% del tráfico actual de Internet proviene de aplicaciones basadas en TCP, como HTTP/Web, correo electrónico, transferencia de archivos, mensajería instantánea, juegos en línea y algunas aplicaciones de streaming de vídeo (YouTube). El 20% restante del tráfico de Internet pertenece a aplicaciones basadas en UDP, como el sistema de nombres de dominio (DNS) y aplicaciones de VoIP en tiempo real (Sawashima, Hori, & Sunahara, 1997).

El protocolo TCP, originalmente definido en RFC 793 en 1981, proporciona un servicio orientado a la conexión, punto a punto y es fiable. El encabezado TCP contiene un número de puerto de origen de 16 bits, un número de puerto de destino de 16 bits, un número de secuencia de 32 bits, un número de acuse de 32 bits, una longitud de encabezado de TCP de 4 bits, RST (Restablecer), PSU (Push 'data'), ACK (Acuse de recibo), URG (Urgent bit), suma de verificación de 16 bits, puntero de 16 bits urgente y campos de opciones. El encabezado TCP mínimo es 20 bytes (cuando no hay opciones).

El número de secuencia y el número de acuse se utilizan para indicar la ubicación del paquete de envío dentro del flujo de envío y para confirmar la recepción de paquetes relevantes (junto con el bit de indicador de ACK). Este mecanismo de reconocimiento junto con la retransmisión de paquetes perdidos es la clave para la transmisión fiable de paquetes TCP.

Otras características como el control de flujo (mediante el uso de un tamaño de ventana de 16 bits) garantizan los procesos de envío y recepción a una velocidad de coincidencia (no enviar demasiado rápido o demasiado lento). El mecanismo de control de la congestión se utiliza para ajustar la velocidad de envío de bits en respuesta a la congestión de la red. Debido a las características anteriores, TCP se utiliza principalmente para la transmisión de aplicaciones de datos de alta fiabilidad, insensibles al retardo (como correo electrónico, transferencia de datos ftp y aplicaciones web http). Las características de reconocimiento, retransmisión, control de congestión no son adecuadas para aplicaciones VoIP en tiempo real. El control punto a punto y de flujo tampoco es adecuado para aplicaciones de videoconferencia en las que se necesita enviar un flujo a varios clientes. Esto ha hecho de UDP una única opción para la transmisión de paquetes VoIP.

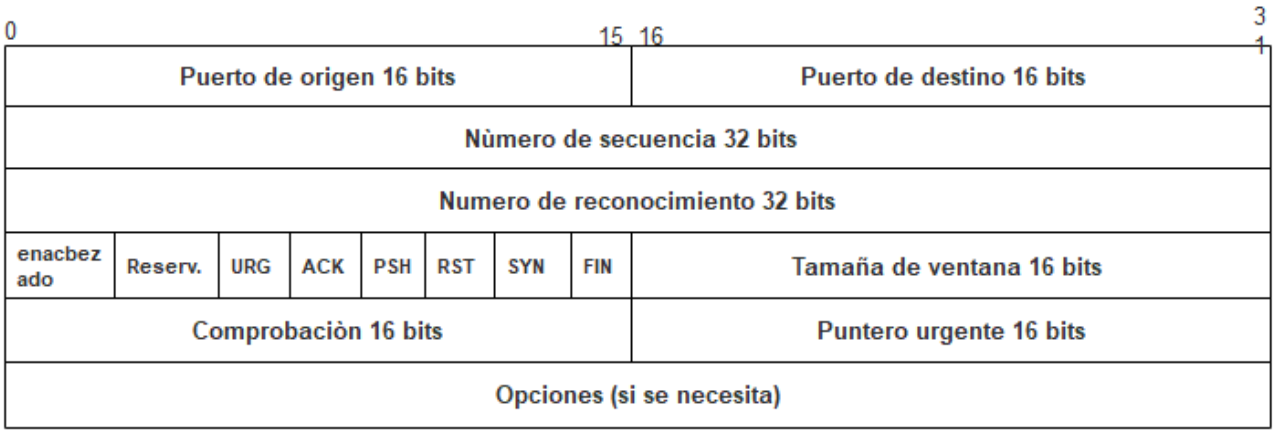


Ilustración 2. Cabecera TCP

2.4.3 UDP

En comparación con TCP, UDP (User Datagram Protocol), originalmente definido en RFC 768 en 1980, es muy simple en su estructura y funciones. Como se muestra en la Ilustración 6, el encabezado UDP que sólo contiene 8 bytes, con 16 bits para el número de puerto de origen, 16 bits para el número de puerto de destino, 16 bits para la longitud del paquete UDP y 16 bits restantes para el checksum UDP. No hay etapa de establecimiento de conexión, no hay control de flujo, control de congestión, ni mecanismos de retransmisión como se proporciona en TCP. Ninguna etapa de conexión (sin conexión) y ningún mecanismo de retransmisión significan que la transferencia UDP es más rápida que la transferencia TCP. Ningún número de secuencia y mecanismo de reconocimiento, significa que la transferencia de paquetes UDP no conocerá el orden de sus paquetes y no sabrá si se recibe o no un paquete. Esto hace que la transferencia UDP sea rápida, pero poco fiable. La naturaleza de transferencia rápida de UDP lo hace adecuado para aplicaciones multimedia en tiempo real, como VoIP, que también puede tolerar cierto grado de pérdida de paquetes.

Desde el punto de vista de implementación de socket, un paquete UDP se envía al lado de destino (a través de su socket de destino). Todo dependerá de la red si va a llegar al destino o no. Debido a la naturaleza de la red IP, algunos paquetes pueden ser duplicados, algunos paquetes pueden llegar fuera de la orden. Es evidente que UDP no puede resolver el problema en relación a poner los paquetes de voz o video en orden correcto para ser reproducidos correctamente en el lado del receptor para aplicaciones VoIP.

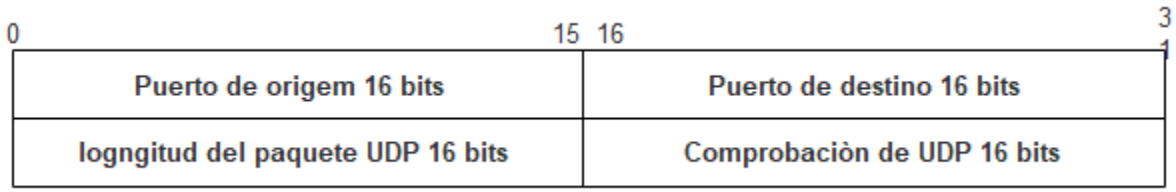


Ilustración 3 Cabecera UTP

2.4.4 RTP

Protocolo de transporte en tiempo real, RTP fue propuesto originalmente en RFC 1889 en 1996 (ahora obsoleto) y refinado en RFC 3550 en 2003. Su objetivo es apoyar la transferencia de datos multimedia en tiempo real a través del protocolo de transporte UDP. RTP agregó el número de secuencia con el fin de identificar la pérdida de los paquetes RTP. Junto con el campo timestamp, permite al receptor reproducir los paquetes de voz / video recibidos en el orden correcto y en la posición correcta. RTCP (RTP Control Protocol), en asociación con RTP, se utiliza para supervisar la calidad del servicio de una sesión VoIP y para transmitir información sobre los participantes en una sesión en curso. Los paquetes RTCP se envían periódicamente y contienen informes de emisor y/o receptor (por ejemplo, para la tasa de pérdida de paquetes y el valor de fluctuación).

El encabezado RTP incluye principalmente el tipo de carga útil (para codificadores de audio / vídeo), el número de secuencia y la estampa de tiempo.

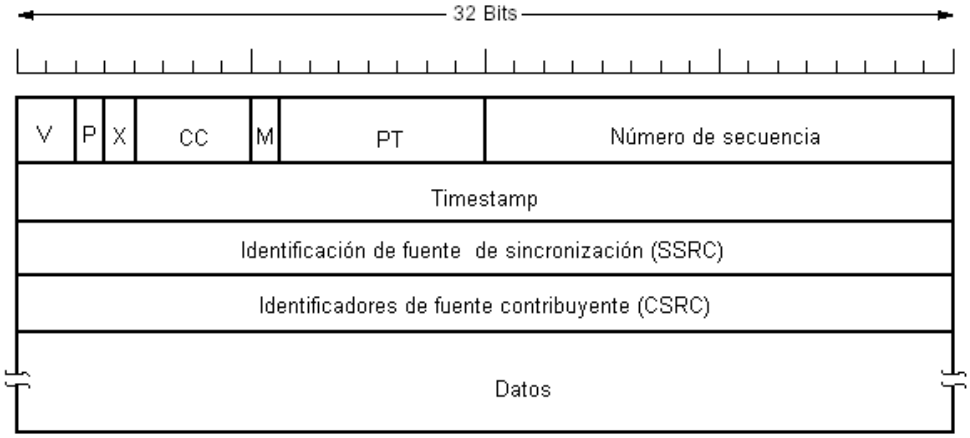


Ilustración 4 Cabecera RTP

En la Ilustración 7 se muestra el encabezado RTP que incluye los siguientes campos de encabezado:

- V: Este campo (2 bits) contiene la versión del protocolo RTP. La versión definida por RFC 1889 es dos.
- P: Este es el bit de relleno que indica si hay campos de relleno en el paquete RTP.
- X: Este es el bit extensión que indica si el encabezado de extensión está presente.
- CC: Este campo (4 bits) contiene el recuento CSRC, el número de Identificadores de origen.
- M: Este es el bit de marcador. Para voz, esto marca el inicio de una conversación de voz, si la supresión de silencio está habilitada en el codificador. Por ejemplo, M se establece en 1 para el primer paquete después de un período de silencio y es cero en caso contrario. Para el vídeo, el bit de marcador se establece en uno (true) para el último paquete de una trama de vídeo y cero en caso contrario. Por ejemplo, si una trama I se divide en 8 paquetes para transmitir a través del canal/red, los primeros siete paquetes tendrán el bit de marcador puesto a cero (falso) y el 8º paquete (último paquete para la trama I) tendrá el bit de marcador establecido en uno (verdadero). Si un P-frame se pone en dos paquetes. El primer paquete tendrá el bit marcador puesto a cero y el bit M del segundo paquete establecido como uno. Si sólo hay un paquete para una trama P o B, el bit de marcador siempre será uno.
- PT: Este campo (7 bits) contiene el tipo de carga útil para voz o vídeo.
- Número de secuencia: Este campo (16 bits) contiene el número de secuencia que se incrementará en uno por cada paquete RTP enviado para detectar la pérdida de paquetes.
- Timestamp: Este campo (32 bits) indica el instante de muestreo cuando se generó el primer octeto de los datos RTP. Se mide de acuerdo con la velocidad del reloj de medios. Para voz, la frecuencia de reloj de fecha y hora es de 8 kHz para la mayoría de codificadores y de 16 kHz para otros. Por ejemplo, para el códec G.723.1 con tamaño de fotograma de 30 ms (que contiene 240 muestras de voz a una frecuencia de muestreo de 8 kHz) y una trama de voz por paquete, la diferencia de marca de tiempo entre dos paquetes consecutivos será 240. En el caso del habla usando el

codificador G.723.1, la velocidad de reloj es la misma con la velocidad de muestreo y la diferencia de marca de tiempo basada en la velocidad de reloj de medios para dos paquetes consecutivos puede decidirse por el número de muestras de voz que un paquete contiene.

- Identificador SSRC: SSRC es para la fuente de sincronización. Este campo (32 bits) contiene el identificador de una fuente de voz o video. Los paquetes originados en la misma fuente tendrán el mismo número de SSRC.
- Identificador CSRC: CSRC es para Contribution Source. Sólo estará disponible cuando el valor del campo CC no sea cero, lo que significa que se han mezclado más de una fuente para producir el contenido de este paquete. Este campo (32 bits) contiene una entrada para el identificador de una fuente contribuyente. Se admiten hasta 16 entradas.

2.5 RTCP

El protocolo de control RTP (RTCP), también definido en RFC 1889 (ahora obsoleto) y RFC 3550, es un protocolo de control de transporte asociado con RTP. Puede proporcionar información de retroalimentación relacionada con la calidad para una sesión RTP en curso, junto con información de identificación para los participantes de una sesión RTP.

Puede utilizarse para el control y la gestión de la calidad de VoIP (por ejemplo, un remitente puede ajustar su velocidad de envío de bits de acuerdo con la información de calidad de VoIP y de red recibida) y también puede ser utilizado por las herramientas de supervisión de terceros. Los paquetes RTCP son enviados periódicamente por cada miembro participante a otros miembros de la sesión y su ancho de banda no debe ser más del 5% del ancho de banda de la sesión RTP.

Existen cinco tipos diferentes de paquetes RTCP, que son SR (informe del remitente), RR (informe del receptor), paquete SDES (descripción de la fuente), paquete BYE (adiós) y paquete APP (definido por la aplicación).

- SR: Informe del remitente: proporciona información de feed-forward sobre los datos enviados y la información de retroalimentación de las estadísticas de recepción de todas las fuentes desde las que el remitente recibe datos RTP.
- RR: Informe del receptor: proporciona información de retroalimentación de las estadísticas de recepción para todos los participantes durante el período de presentación de informes.
- SDES: Fuente descripción: proporciona información de identificador de fuente, como el nombre canónico (CNAME) de la fuente participante, por ejemplo, nombre de usuario y dirección de correo electrónico.
- BYE: Paquete de despedida: indica el final de un participante (por ejemplo, un participante cuelga una llamada VoIP).
- APP: Paquete definido por la aplicación: proporciona funciones específicas de la aplicación.

CAPÍTULO 3

Capítulo 3 CALIDAD DE SERVICIO

La calidad de servicio se define como "el efecto colectivo del rendimiento del servicio, que determina el grado de satisfacción de un usuario del servicio" en la Rec. E.800 (versión de 1988). En la red o en un campo más específico de voz y vídeo sobre IP, QoS se denomina en general calidad de servicio de red (NQoS) con un enfoque en la calidad del rendimiento de la red IP en contraste con QoS de extremo a extremo que también incluye la calidad de la terminal/dispositivo final o segmentos/dispositivos relacionados con redes de comunicaciones conmutadas (SCN) tales como PSTN, ISDN y GSM como se muestra en la Ilustración 8.

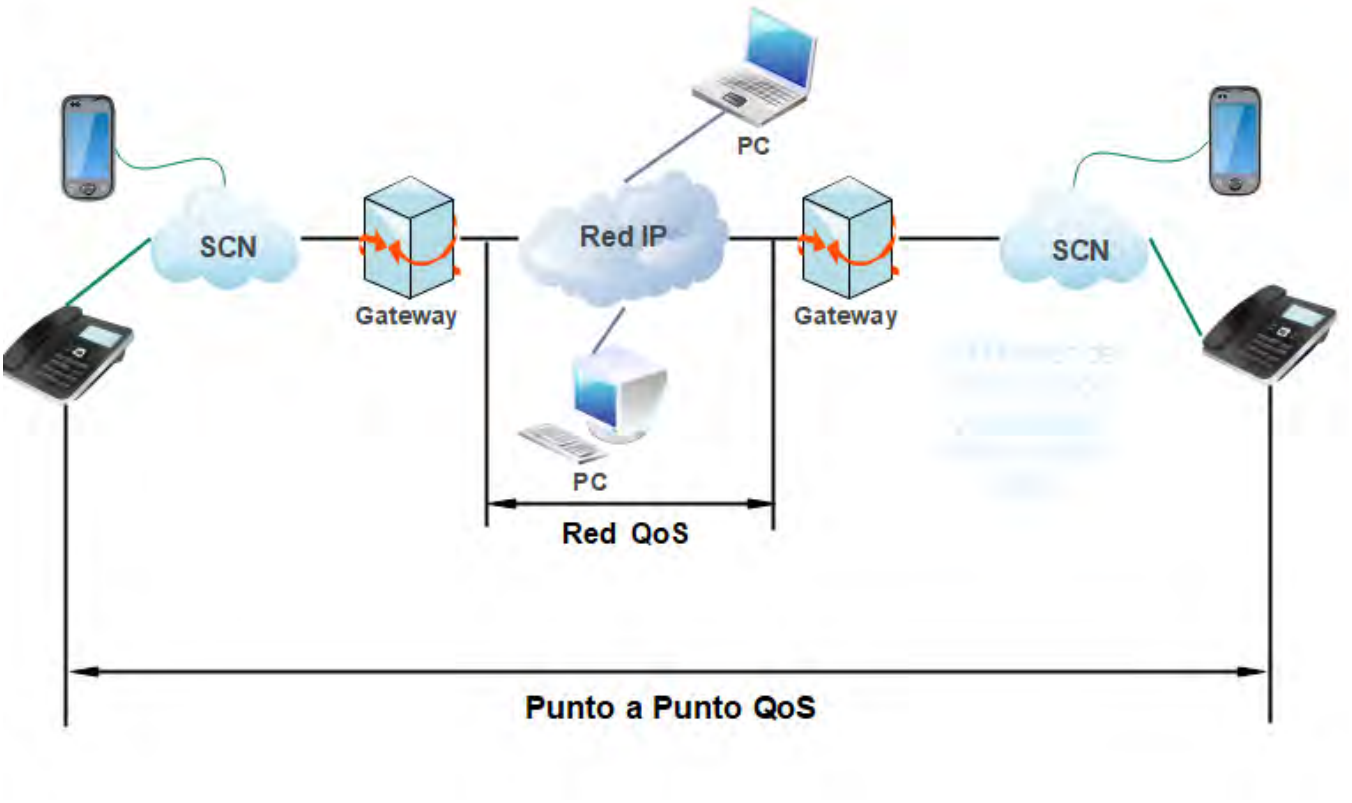


Ilustración 5. Red QoS Extremo a Extremo

En las aplicaciones de VoIP, la QoS de red cubre la calidad del segmento de transmisión IP, que comienza cuando los paquetes IP dejan el terminal o el dispositivo final (por ejemplo, una PC o una computadora portátil que ejecuta un softphone) a otro terminal VoIP a través de la red IP. Un escenario de llamadas de PC a PC o desde cuando los paquetes IP dejan un gateway de medios en una red combinada PSTN / IP a otro gateway de medios en un escenario de

llamada de teléfono a teléfono. En las aplicaciones de VoIP, la QoS de extremo a extremo también se considera como una calidad de boca a boca al reflejar la calidad de una llamada VoIP de un usuario que habla al micrófono de un teléfono en un extremo a otro que escucha en el teléfono en el otro extremo. Esto es principalmente para la calidad del habla de escucha de una vía sin tener en cuenta la interactividad (para la calidad de la conversación).

Recientemente, el término de calidad de servicio extremo a extremo, ha sido gradualmente reemplazado por la calidad de servicio percibida (PQoS) para reflejar la naturaleza de cómo el usuario final percibe la calidad proporcionada y la calidad de experiencia (QoE), con un enfoque en la experiencia del usuario en la calidad del servicio prestado.

3.1 Métricas de desempeño

La calidad de servicio de la red o el rendimiento de la red suele estar representado por métricas de desempeño, tales como pérdida de paquetes, retardo y variación de retardo (jitter). Estas métricas pueden medirse o monitorearse mediante medición activa (intrusiva) o pasiva (no intrusiva) (Mohan, Reddy, & Kalpana , 2011). En la medición se envían a la red paquetes de prueba (por ejemplo, paquetes ping de protocolo de mensajes de control de Internet (ICMP) como se usan en la herramienta común de "ping") y se comparan con el paquete de eco para obtener las métricas de rendimiento de red.

3.1.1. Pérdida de paquetes

Entre los diferentes elementos de calidad, la pérdida de paquetes es la degradación, lo que hace a VoIP perceptualmente más diferente de las redes de telefonía fija. La pérdida de paquetes puede ocurrir en la red o en el emplazamiento del receptor, por ejemplo, debido al excesivo retardo de la red en caso de congestiones. En las redes de paquetes inalámbricas, la pérdida de paquetes a menudo resulta de errores de bit introducidos en el enlace de radio. En este caso, no se pierden paquetes completos, sino sólo partes de ellos. Dependiendo de la importancia de la información perdida, las partes recibidas de los paquetes afectados pueden todavía ser usadas durante la decodificación. Los principales elementos que influyen en el deterioro de la pérdida de paquetes son la distribución de pérdida de paquetes, el tamaño de

paquete, la recuperación de pérdida de paquetes a nivel de paquete (Estrada, Torres, & Toral-Cruz, 2010), utilizando mecanismos como FEC, la decodificación de voz y medidas respectivas para PLC aplicado en el decodificador.

La pérdida de paquetes de red es un impedimento que afectará la calidad de voz y video en aplicaciones VoIP. Hay principalmente dos tipos de pérdida de paquetes.

- Uno es causado por congestiones de la red en acoplamientos del cuello de botella a lo largo de la trayectoria debido al desbordamiento de la cola intermedia del ranurador. Este tipo de pérdida se llama pérdida congestiva (congestión en la red).
- Pérdida no congestiva se debe principalmente a enlaces con pérdidas como redes móviles/inalámbricas y redes de acceso ADSL y es de naturaleza aleatoria. La pérdida de paquetes bursty tiene un efecto más adverso en la calidad de voz/video cuando se compara con la pérdida aleatoria de paquetes. Esto se debe a que el codificador incorpora un mecanismo de ocultamiento de pérdida de paquetes incorporado en el lado del decodificador que es capaz de ocultar los paquetes perdidos basándose en información de paquetes recibidos previamente.

Se han investigado las características de pérdida de paquetes basadas en la recopilación de datos de rastreo de Internet real a través de medición de QoS activa o pasiva. Se han desarrollado diferentes modelos de pérdida de paquetes para caracterizar las pérdidas de paquetes a través de Internet.

3.1.2 Retardo

El retardo en una red IP, es el tiempo invertido por un paquete que se transporta desde el remitente (punto A) hasta el receptor (punto B).

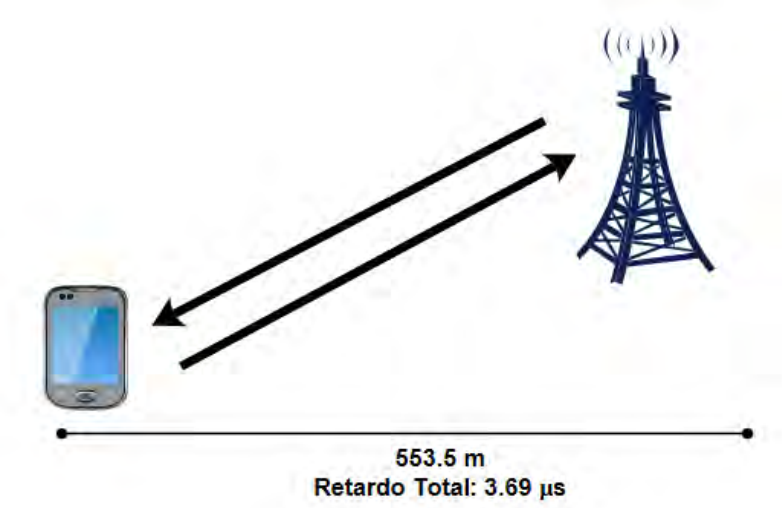


Ilustración 6. Calculo del Retardo

El retardo de la red IP consiste principalmente en los siguientes componentes (Toral-Cruz, Homero ; Torres, Deni; Estrada, Leopoldo, 2012):

- Retardo de propagación: depende únicamente de la distancia física de la vía de comunicación y del medio de comunicación.
- Retardo de transmisión: la suma del tiempo que tarda las interfaces de red en los routers para enviar el paquete a lo largo de la ruta.
- Retardo de procesamiento nodal: la suma del tiempo que tarda en los routers para decidir dónde (qué interfaz) enviar el paquete basado en el análisis de encabezado de paquetes y la tabla de enrutamiento.
- Retardo de cola: el tiempo que un paquete tiene que pasar en las colas de los routers a lo largo de la ruta. Se debe principalmente a la congestión de la red.

3.1.3 Jitter

La variación del retardo se denomina jitter, estas variaciones con valores superiores a 50ms afectan la calidad de servicio de la voz (Calyam, Sridharan, Mandrawa, & Schopis, 2004). Un buffer de reproducción en el lado del receptor se utiliza para atenuar el impacto del jitter y para garantizar una reproducción adecuada de audio o vídeo.

Debido al enrutamiento de los paquetes sobre diferentes rutas de red y las características asíncronas de la red, los paquetes dentro de un pico de conversación pueden llegar a su destino

con un retardo variable. La inestabilidad degrada significativamente la calidad del habla y tiene que ser compensada. Esto se hace generalmente en la red en el lado del receptor aplicando buffers de jitter que almacenan paquetes para una cantidad de tiempo estática o gestionada dinámicamente antes de la reproducción. Esta medida introduce un retardo adicional y potencialmente pérdida de paquetes, por ejemplo, si los paquetes llegan demasiado tarde para su reproducción. Sin embargo, estos inconvenientes son más aceptables que la degradación debida al jitter.

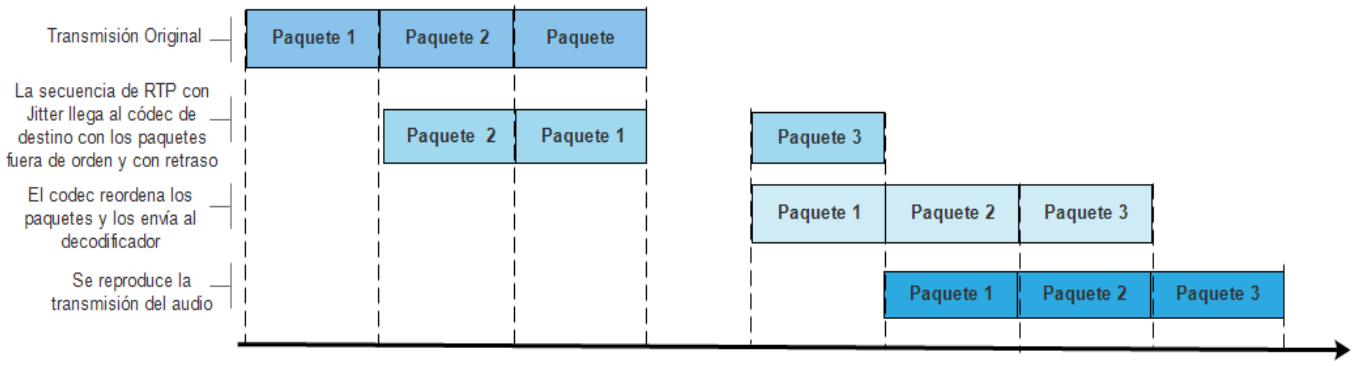


Ilustración 7. Jitter.

Por ejemplo, en una conversación telefónica, los usuarios tienen una determinada ventana de tiempo durante la cual esperan una reacción de su interlocutor. Según Kitawaki e Itoh (1991), el retardo se detecta cuando la reacción real excede esta ventana de tiempo. El tamaño de esta ventana de tiempo depende tanto de la velocidad de emisión como de la duración de la propagación precedente. La ventana se ensancha tanto con el aumento de la duración de la conversación como con la disminución de la velocidad de emisión. Dado que una interactividad más alta implica períodos de duración de la conversación más cortos y una velocidad de emisión más rápida, la detectabilidad del retardo aumenta con la interactividad. Curiosamente, en Krauss y Bricker (1966) se encontró que el creciente retardo en las conversaciones telefónicas con un tema de conversación libre conduce a un aumento de las duraciones de los enunciados debido a la adaptación por parte de los interlocutores.

Esta adaptación puede hacer que el retardo sea más tolerable debido al ensanchamiento asociado de la ventana de tiempo durante la cual los usuarios esperan la reacción de su pareja. A su vez, en el caso de conversaciones más estructuradas como en el intercambio interactivo

de números aleatorios, tal adaptación no se puede realizar, lo que explica por qué las tareas de conversación más interactivas son más sensibles al retardo.

3.2 Técnicas de Evaluación de Desempeño de la Voz

La calidad de la experiencia (QoE) se define como "la aceptabilidad general de una aplicación o servicio, tal como es percibido subjetivamente por el usuario final", de acuerdo con la recomendación P.10/G.100. QoE normalmente se considera como calidad percibida de servicio (PQoS) a diferenciación con la calidad de servicio (QoS) que generalmente se considera como calidad de servicio de red (NQoS) o métricas relacionadas para la calidad de red como pérdida de paquetes de red, retardo y jitter.

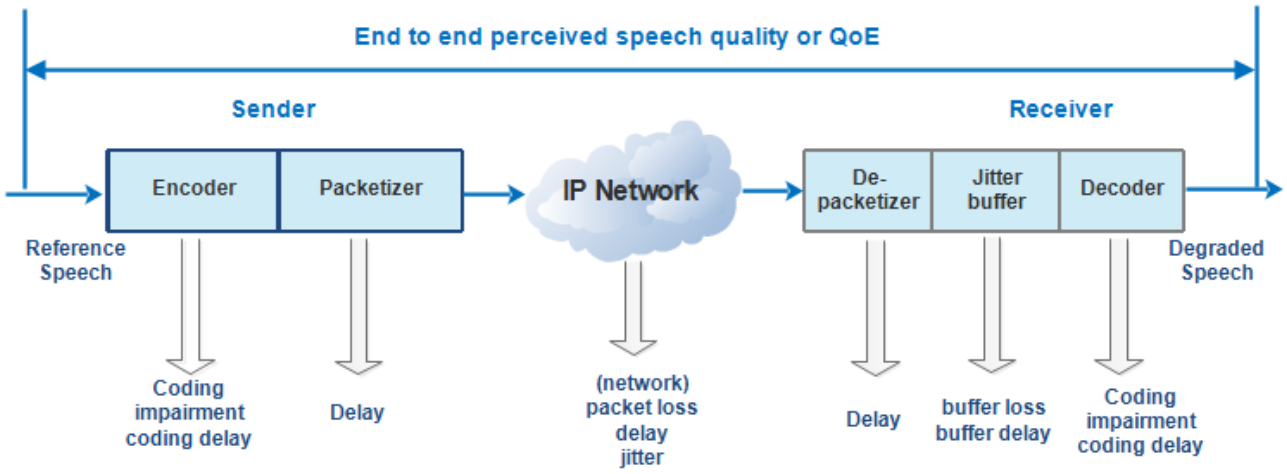


Ilustración 8 Factores que afectan la calidad del servicio de punto a punto

3.2.1 MOS

La métrica QoE clave para aplicaciones VoIP es el Mean Opinion Socore (MOS); la cual, es una métrica utilizada para representar una calidad general de voz proporcionada durante una llamada VoIP. Esta métrica se obtiene generalmente promediando las calificaciones globales de calidad proporcionadas por un grupo de usuarios (para el término de opinión media). MOS también se utiliza para representar la calidad de video percibida para videoconferencia o

aplicaciones de transmisión de video o calidad audiovisual percibida cuando tanto la voz como el video se tienen en cuenta en escenarios de videoconferencia.

La escala del MOS se especifica en la Tabla 1.

MOS	CALIDAD	DETERIORO
5	Excelente	Imperceptible
4	Bueno	Perceptible pero no molesto
3	Aceptable	Ligeramente molesto
2	Pobre	Molesto
1	Malo	Muy molesto

Tabla 1. Escala del MOS

3.2.2 Modelo E

Una herramienta útil para evaluar el impacto relativo de las decisiones de planificación de la transmisión en el rendimiento del habla es el E-Model. El Modelo E ha sido incluido en varias Recomendaciones de la ITU-T (International Telecommunication Union - Telecommunication, Standardization Sector) sobre la planificación de la transmisión. En particular, el Modelo E es el tema de la Recomendación G.107. No obstante, también ha sido adoptada por ETSI y TIA (Telecommunications and Industry Association) y se ha convertido en la herramienta más utilizada para la evaluación objetiva de la calidad del habla. Este modelo se basa en el supuesto de que los deterioros causados por los parámetros de transmisión tienen un efecto que degradan la calidad de la voz. De acuerdo con este modelo, la calidad del habla se determina mediante la siguiente ecuación:

$$\boldsymbol{R = R_0 - I_s - I_d - I_e + A}$$

Dónde:

- R_0 : Representa los efectos de ruido.

- I_s : Representa impedimentos tales como un nivel de voz demasiado alto, una audición no óptima y un ruido de cuantificación.
- I_d : Es la suma de los impedimentos debidos a efectos de retardo y eco.
- I_e : Representa deficiencias debidas a codificadores de voz de baja velocidad de bits.
- A representa una "ventaja de acceso" que algunos sistemas tienen en comparación con PSTN (por ejemplo, A para sistemas móviles es 10).

Cabe señalar que los dos primeros términos de la ecuación son intrínsecas a la propia señal de voz y no dependen de la transmisión de voz a través de Internet. El valor de esta función tiene un rango nominal de 0 para terrible hasta 100 para la voz perfecta. Sin embargo, existe una relación directa entre R y la puntuación MOS.

CAPÍTULO 4

Capítulo 4 SISTEMA OPERATIVO Y HERRAMIENTAS DE DESARROLLO

En el presente capitulo se describe el sistema operativo bajo el cual se desarrolló la aplicación y las principales herramientas utilizadas en la implementación de la misma.

4.1 Sistema Operativo Android

Android es un sistema operativo, inicialmente diseñado para teléfonos móviles como los sistemas operativos iOS (Apple), Symbian (Nokia) y Blackberry OS.
En la actualidad, este sistema operativo se instala no sólo en móviles, sino también en múltiples dispositivos, como tabletas, GPS, televisores, discos duros multimedia, mini ordenadores, microondas, lavadoras etc.
Está basado en Linux, el cual es un núcleo de sistema operativo libre, gratuito y multiplataforma. Este sistema operativo permite programar aplicaciones empleando una variación de Java llamada Dalvik, y proporciona todas las interfaces necesarias para desarrollar fácilmente aplicaciones que acceden a las funciones del teléfono (como el GPS, las llamadas, la agenda, etc.) utilizando el lenguaje de programación Java.
Su sencillez principalmente, junto a la existencia de herramientas de programación gratuitas, es la causa de que existan cientos de miles de aplicaciones disponibles, que extienden la funcionalidad de los dispositivos y mejoran la experiencia del usuario.

4.1.1 Versiones de Android

En la Tabla 2 e Ilustración 12 se presentan las versiones del sistema operativo Android hasta el 2018, desde la versión 1.0 Angel cake que fue lanzada en el año 2008 hasta la última actualización con la versión 9.0 Pie que fue lanzado en el 2018, desde la primera versión hasta la última han sido 16 versiones de Android, sin contar las API que se han ido actualizando en cada versión.

VERSIÓN	CODENAME	API
1.0	Angel Cake	1
1.1	Battenberg	2
15	Cupcake	3
1.6	Donut	4
2.0-2.1	Eclair	5, 6, 7
2.2	Froyo	8
2.3.3 – 2.3.7	Gingerbread	9, 10
3.0	Honeycomb	11,12,13,
4.0.3 – 4.0.4	Ice Cream Sandwich	14, 15
4.1.x	Jelly Bean	16
4.2.x		17
4.3		18
4.4	KitKat	19
5.0	Lollipop	21
5.1		22
6.0	Marshmallow	23
7.0	Nougat	24
7.1		25
8.0	Oreo	26
9.0	Pie	28

Tabla 2. Versiones de Android

Ilustración 9 Versiones de Android

4.2 Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan la productividad durante la compilación de apps para Android, como las que se mencionan a continuación:

- Un sistema de compilación basado en Gradle flexible
- Un emulador rápido con varias funciones

- Un entorno unificado en el que se puede realizar desarrollos para todos los dispositivos Android
- Instant Run para aplicar cambios mientras la app se ejecuta sin la necesidad de compilar un nuevo APK
- Integración de plantillas de código y GitHub para ayudar a compilar funciones comunes de las apps e importar ejemplos de código
- Gran cantidad de herramientas y frameworks de prueba
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK

4.3 Arquitectura

En las siguientes líneas se dará una visión global por capas de la arquitectura empleada en Android. Cada una de estas capas utiliza servicios ofrecidos por las anteriores, y ofrece a su vez los suyos propios a las capas de niveles superiores, tal como muestra en la Ilustración 13((c) Google).

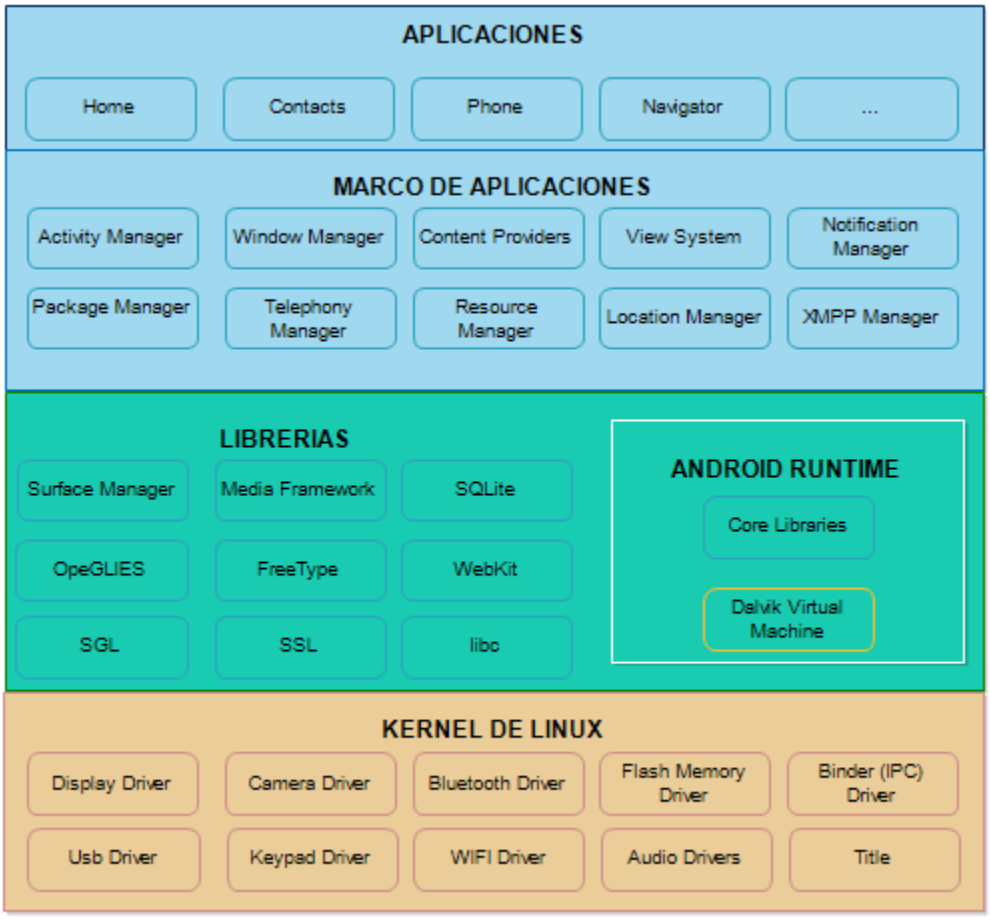


Ilustración 10.Arquitectura del Sistema Android

4.3.1 Aplicaciones

Este nivel contiene, tanto las incluidas por defecto de Android como aquellas que el usuario vaya añadiendo posteriormente, ya sean de terceras empresas o de su propio desarrollo. Todas estas aplicaciones utilizan los servicios, las API y librerías de los niveles anteriores.

4.3.2 Anatomía/Armazón de una Aplicación

Representa fundamentalmente el conjunto de herramientas de desarrollo de cualquier aplicación. Toda aplicación que se desarrolle para Android, ya sean las propias del dispositivo, las desarrolladas por Google o terceras compañías, o incluso las que el propio usuario cree, utilizan el mismo conjunto de API y el mismo "framework", representado por este nivel.

Entre las API más importantes ubicadas aquí, se pueden encontrar las siguientes:

- *Activity Manager:* Conjunto de API que gestiona el ciclo de vida de las aplicaciones en Android.
- *Window Manager:* Gestiona las ventanas de las aplicaciones y utiliza la librería Surface Manager.
- *Telephone Manager:* Incluye todas las API vinculadas a las funcionalidades propias del teléfono (llamadas, mensajes, etc.).
- *Content Provider:* Permite a cualquier aplicación compartir sus datos con las demás aplicaciones de Android. Por ejemplo, gracias a esta API la información de contactos, agenda, mensajes, etc. será accesible para otras aplicaciones.
- *View System:* Proporciona un gran número de elementos para poder construir interfaces de usuario (GUI), como listas, mosaicos, botones, "check-boxes", tamaño de ventanas, control de las interfaces mediante teclado, etc. Incluye también algunas vistas estándar para las funcionalidades más frecuentes.
- *Location Manager:* Posibilita a las aplicaciones la obtención de información de localización y posicionamiento.
- *Notification Manager:* Mediante el cual las aplicaciones, usando un mismo formato, comunican al usuario eventos que ocurran durante su ejecución: una llamada entrante, un mensaje recibido, conexión Wi-Fi disponible, ubicación en un punto determinado, etc. Si llevan asociada alguna acción, en Android denominada **Intent**, (por ejemplo, atender una llamada recibida) ésta se activa mediante un simple clic.
- *XMPP Service:* Colección de API para utilizar este protocolo de intercambio de mensajes basado en XML.

4.3.3 Librerías

La siguiente capa corresponde con las librerías utilizadas por Android. Éstas han sido escritas utilizando C/C++ y proporcionan a Android la mayor parte de sus capacidades más características. Junto al núcleo basado en Linux, estas librerías constituyen el corazón de Android.

Entre las librerías más importantes ubicadas aquí, se pueden encontrar las siguientes:

- *Librería libc:* Incluye todas las cabeceras y funciones según el estándar del lenguaje C. Todas las demás librerías se definen en este lenguaje.
- *Librería Surface Manager:* Es la encargada de componer los diferentes elementos de navegación de pantalla. Gestiona también las ventanas pertenecientes a las distintas aplicaciones activas en cada momento.
- *OpenGL/SL y SGL:* Representan las librerías gráficas y, por tanto, sustentan la capacidad gráfica de Android. OpenGL/SL maneja gráficos en 3D y permite utilizar, en caso de que esté disponible en el propio dispositivo móvil, el hardware encargado de proporcionar gráficos 3D. Por otro lado, SGL proporciona gráficos en 2D, por lo que será la librería más habitualmente utilizada por la mayoría de las aplicaciones. Una característica importante de la capacidad gráfica de Android es que es posible desarrollar aplicaciones que combinen gráficos en 3D y 2D.
- *Librería Media Libraries:* Proporciona todos los códecs necesarios para el contenido multimedia soportado en Android (vídeo, audio, imágenes estáticas y animadas, etc.)
- *FreeType:* Permite trabajar de forma rápida y sencilla con distintos tipos de fuentes.
- *Librería SSL:* Posibilita la utilización de dicho protocolo para establecer comunicaciones seguras.
- *Librería SQLite:* Creación y gestión de bases de datos relacionales.
- *Librería WebKit:* Proporciona un motor para las aplicaciones de tipo navegador y forma el núcleo del actual navegador incluido por defecto en la plataforma Android.

4.3.4 Android Runtime/Tiempo de ejecución de Android

Al mismo nivel que las librerías de Android se sitúa el entorno de ejecución. Éste lo constituyen las Core Libraries, que son librerías con multitud de clases Java y la máquina virtual Dalvik.

La máquina virtual de Android Runtime (ART) y Dalvik utilizan la paginación y la asignación de memoria (mmapping) para administrar la memoria. Esto significa que cualquier memoria que una aplicación modifique, ya sea asignando nuevos objetos o tocando páginas mapeadas,

permanece residente en la RAM y no se puede localizar. La única manera de liberar memoria de una aplicación es liberar referencias de objetos que la aplicación mantiene, haciendo que la memoria esté disponible para el recolector de elementos no utilizados. Eso es con una excepción: cualquier archivo mmapped sin modificación, como código, puede ser paginado fuera de RAM si el sistema quiere usar esa memoria en otra parte.

4.3.5 Kernel de Linux

Android utiliza el **núcleo de Linux 2.6** como una capa de abstracción para el hardware disponible en los dispositivos móviles. Esta capa contiene los drivers necesarios para que cualquier componente hardware pueda ser utilizado mediante las llamadas correspondientes. Siempre que un fabricante incluye un nuevo elemento de hardware, lo primero que se debe realizar para que pueda ser utilizado desde Android es crear las librerías de control o drivers necesarios dentro de este kernel de Linux embebido en el propio Android.

4.3.6 Android Manifest

Éste archivo es uno de los más importantes de cualquier aplicación Android. Se genera automáticamente al crear el proyecto, y en él se encuentra definida la configuración del proyecto en XML (actividades, intents, los permisos de la aplicación, bibliotecas, etc.).

4.4 Actividades

La actividad es el componente fundamental de la mayoría de aplicaciones Android. El concepto de actividad va ligado a la idea de interfaz de usuario (IU). Casi todas las actividades interactúan con el usuario, por lo que la clase de actividad se encarga de crear una ventana para colocar la interfaz de usuario setContentView(View). Mientras que las actividades se presentan a menudo al usuario como ventanas de pantalla completa, también se pueden utilizar de otras maneras: como ventanas flotantes (a través de un tema con windowIsFloatingconjunto) o incrustado dentro de otra actividad (utilizando ActivityGroup). Hay dos métodos que casi todas las subclases de Actividad implementan:

- onCreate(Bundle): Es donde el usuario inicializa su actividad. Lo más importante, aquí suele llamarse setContentView(int) con un recurso de diseño que define su interfaz de usuario y utiliza findViewById(int) para recuperar los widgets en esa interfaz de usuario que necesita interactuar mediante programación.
- onPause(): Es donde el usuario abandone su actividad. Lo que es más importante, cualquier cambio realizado por el usuario debe ser comprometido en este momento (por lo general a la ContentProvidercelebración de los datos).

Para ser de uso con Context.startActivity(), todas las clases de actividad deben tener una <activity> declaración correspondiente en su paquete AndroidManifest.xml.

4.4.1 Ciclo de vida de una actividad

Las actividades del sistema se gestionan como una *pila de actividades*. Cuando se inicia una nueva actividad, se coloca en la parte superior de la pila y se convierte en la actividad en ejecución: la actividad anterior siempre permanece por debajo de ella en la pila y no volverá al primer plano hasta que se cierre la nueva actividad.

Una actividad tiene esencialmente cuatro estados:

- Si una actividad está en primer plano de la pantalla (en la parte superior de la pila), está *activa* o en *ejecución*.
- Si una actividad ha perdido el foco pero sigue siendo visible (es decir, una nueva actividad no de tamaño completo o transparente se ha centrado en la parte superior de su actividad), se *detiene*. Una actividad en pausa está completamente viva (mantiene toda la información de estado y miembro y permanece conectada al administrador de ventanas), pero puede ser eliminada por el sistema en situaciones de memoria de muy baja intensidad.
- Si una actividad es completamente oscurecida por otra actividad, se *detiene*. Todavía conserva toda la información de estado y de miembro, sin embargo, ya no es visible para el usuario por lo que su ventana está oculta y que a menudo será asesinado por el sistema cuando la memoria se necesita en otro lugar.

- Si una actividad se detiene, el sistema puede dejar caer la actividad de la memoria ya sea pidiéndole que finalice o simplemente mate su proceso. Cuando se vuelve a mostrar al usuario, debe reiniciarse completamente y volver a su estado anterior.

El diagrama de la Ilustración 14 muestra las rutas de estado importantes de una actividad. Los rectángulos representan métodos de devolución de llamada que puede implementar para realizar operaciones cuando la actividad se mueve entre estados. Los óvalos coloreados son estados importantes en los que puede estar la actividad.

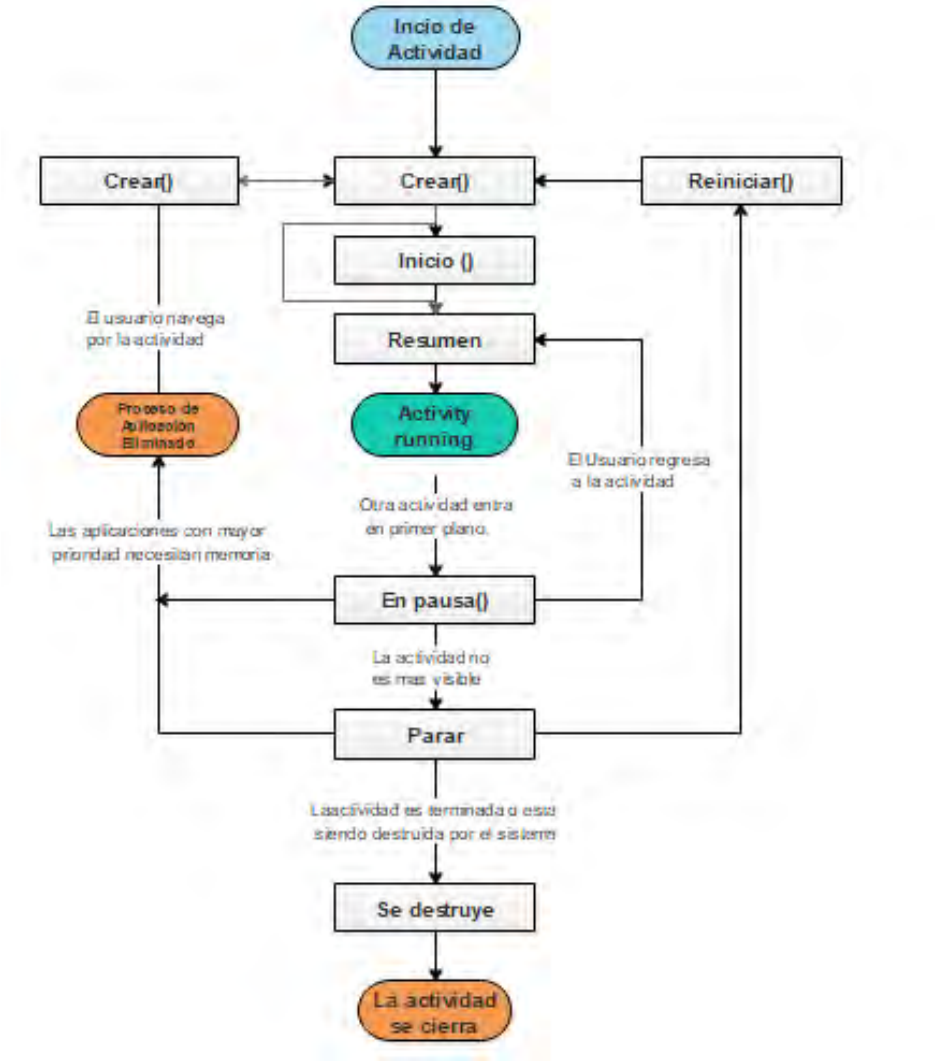


Ilustración 11. Ciclo de vida de una actividad

4.5 Java

Java es un lenguaje y una plataforma originada por Sun Microsystems. En esta sección, se describe brevemente este lenguaje. Para satisfacer diversas necesidades, Sun organizó Java en tres ediciones principales: Java SE, Java EE y Java ME.

4.5.1 El Lenguaje Java

Java es un lenguaje en el que los desarrolladores expresan el código fuente (texto del programa). La sintaxis de Java (reglas para combinar símbolos en características de lenguaje) está en parte modelada después de los lenguajes C y C ++ para acortar la curva de aprendizaje de los desarrolladores de C / C ++.

La siguiente lista identifica algunas similitudes entre Java y C / C ++:

- Java y C / C ++ comparten los mismos estilos de comentarios de línea única y multilínea. Los comentarios permiten documentar el código fuente.
- Muchas de las palabras reservadas de Java son idénticas a sus contrapartes de C / C ++ (para, si, cambiar y mientras) y las contrapartes de C ++ (catch, class, public y try).
- Java soporta caracteres primitivos de doble precisión, de coma flotante, de entero, de entero largo y de entero corto, y a través de las mismas palabras reservadas char, double, float, int, long y short.
- Java soporta muchos de los mismos operadores, incluyendo operadores aritméticos (+, -, *, / y%) y condicionales (? :).

4.5.2 Java como Plataforma

Java es una plataforma que consiste en una máquina virtual y un entorno de ejecución. La máquina virtual es un procesador basado en software que presenta un conjunto de instrucciones. El entorno de ejecución consiste en bibliotecas para ejecutar programas e interactuar con el sistema operativo subyacente.

El entorno de ejecución incluye una enorme biblioteca de archivos de clases preconfigurados que realizan tareas comunes, como operaciones matemáticas (por ejemplo, trigonometría) y comunicaciones de red. Esta biblioteca se conoce comúnmente como la biblioteca de clases estándar.

Un programa Java especial conocido como el compilador de Java traduce el código fuente en instrucciones (y datos asociados) que son ejecutadas por la máquina virtual.

Estas instrucciones se conocen como bytecode. El compilador almacena el bytecode de un programa y los datos en archivos que tienen la extensión *.class*. Estos archivos se conocen como archivos de clase porque normalmente almacenan el equivalente compilado de clases, una característica de lenguaje.

Un programa Java se ejecuta a través de una herramienta que carga e inicia la máquina virtual y pasa el archivo de clase principal del programa a la máquina. La máquina virtual utiliza un cargador de clases (una máquina virtual o un componente de entorno de ejecución) para cargar el archivo de clase.

4.5.3 JDK

Android Studio proporciona asistentes y plantillas que verifican los requisitos de tu sistema, como el Java Development Kit (JDK) y la memoria RAM disponible, y configuran los ajustes predeterminados, como una emulación optimizada predeterminada del Android Virtual Device (AVD) e imágenes del sistema actualizadas.

El kit de desarrollo de Java (JDK) contiene las herramientas y librerías necesarias para crear y ejecutar applets y aplicaciones en Java. A continuación se listan algunas de las utilidades que se pueden encontrar en el JDK:

- javac. Es el compilador de Java. Se encarga de convertir el código fuente escrito en Java a bytecode.
- java. Es el intérprete de Java. Ejecuta el bytecode a partir de los archivos class.
- appletviewer. Es un visor de applets. En la mayoría de las ocasiones puede utilizarse en lugar de un Navegador Web.
- javadoc. Se utiliza para crear documentación en formato HTML a partir del código fuente Java y los comentarios que contiene.
- javap. Es un desensamblador de Java.
- jar. Es una herramienta utilizada para trabajar con los archivos JAR.

CAPÍTULO 5

Capítulo 5 REQUERIMIENTOS

En este capítulo hablaremos de los requerimientos funcionales y no funcionales del sistema, los primeros son servicios que proveerá el sistema y la manera en que este reaccionara a cada una de las entradas particulares y los segundos son aquellos requerimientos que no se refieren directamente a las funciones específicas que entrega el sistema, sino a las propiedades emergentes de éste como la fiabilidad, la respuesta en el tiempo y la capacidad de almacenamiento

5.1 Requerimientos Funcionales

Los requerimientos funcionales son declaraciones de los servicios que proveerá el sistema, de la manera en que éste reaccionará a entradas particulares. En algunos casos, los requerimientos funcionales de los sistemas también declaran explícitamente lo que el sistema no debe hacer.

Para el desarrollo de este sistema en particular se presentan los siguientes requerimientos:

Registros y configuraciones.

En estos requerimientos se realizará el enlace al servidor y se harán las configuraciones necesarias para realizar las llamadas.

ID	NOMBRE	PRIORIDAD
01	Registro	Alta
DESCRIPCIÓN		
El cliente se registra con la cuenta SIP proporcionada por el servidor para obtener permiso para realizar operaciones. El servidor inicia la actividad del cliente, dando acceso a llamadas de voz.		
CRITERIOS DE ÉXITO		
Se establecerá comunicación entre el servidor y el cliente.		

Tabla 3. Requerimiento 01

ID	NOMBRE	PRIORIDAD
02	Establecimiento de comunicación	Alta
DESCRIPCIÓN		
El sistema permitirá a dos usuarios establecer una comunicación de VoIP.		
CRITERIOS DE ÉXITO		
Se establecerá la comunicación con éxito entre los clientes al realizar llamadas de voz. Los usuarios establecerán y realizarán una comunicación sin ningún problema.		

Tabla 4. Requerimiento 02

ID	NOMBRE	PRIORIDAD
03	Códec's	Alta
DESCRIPCIÓN		
El sistema permitirá al usuario elegir el códec con el cual desea establecer la comunicación. • G711 µlaw • G711 Alaw		
CRITERIOS DE ÉXITO		
Los usuarios establecerán una comunicación con el códec seleccionado problema.		

Tabla 5. Requerimiento 03

ID	NOMBRE	PRIORIDAD
04	Puerto	Alta
DESCRIPCIÓN		
El usuario elegirá el puerto que desee utilizar.		
CRITERIOS DE ÉXITO		
Los usuarios establecerán una comunicación a través de este puerto.		

Tabla 6. Requerimiento 04

ID	NOMBRE	PRIORIDAD
05	Protocolo	Alta
DESCRIPCIÓN		
El usuario elegirá el protocolo a utilizar		
CRITERIOS DE ÉXITO		
Los usuarios establecerán comunicación a través de este protocolo.		

Tabla 7. Requerimiento 05

ID	NOMBRE	PRIORIDAD
05	Parámetros de entrada	Alta
DESCRIPCIÓN		
El sistema permitirá introducir los siguientes parámetros, con los cuales se podrá establecer la comunicación. • Servidor • Usuario • Puerto • Protocolo • Tipo de red a usar		
CRITERIOS DE ÉXITO		
Se podrá establecer una comunicación con los parámetros introducidos al sistema.		

Tabla 8. Requerimiento 05

ID	NOMBRE	PRIORIDAD
06	Parámetros QoS Medidos	Alta

DESCRIPCIÓN
Durante las llamadas VoIP, el sistema realizará el cálculo y visualización en tiempo real de los siguientes parámetros: 1. Paquetes Transmitidos 2. Paquetes Recibidos 3. Paquetes Perdidos 4. Número de Secuencia 5. OWD 6. Jitter OWD 7. RTT 8. Jitter RTT 9. PLR 10. Factor R 11. MOS
CRITERIOS DE ÉXITO
El sistema realizara correctamente el cálculo y la visualización en tiempo real de estos parámetros.

Tabla 9. Requerimiento 06

ID	NOMBRE	PRIORIDAD
07	Resultados de las estimaciones de los parámetros QoS medidos	Alta
DESCRIPCIÓN		
Durante la llamadas de VoIP, el sistema permitirá guardar los siguientes parámetros: 1. Número de Secuencia 2. OWD 3. Jitter OWD 4. RTT 5. Jitter RTT 6. PLR 7. Factor R 8. MOS		
CRITERIOS DE ÉXITO		
El sistema permitirá guardar correctamente los valores de estos parámetros calculados durante la comunicación.		

Tabla 10. Requerimiento 07

5.2 Requerimientos No Funcionales

Son aquellos requerimientos que no se refieren directamente a las funciones específicas que entrega el sistema, sino a las propiedades emergentes de éste como la fiabilidad, la respuesta en el tiempo y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y la representación de datos que se utiliza en la interface del sistema.

Los requerimientos no funcionales surgen de la necesidad del usuario, debido a las restricciones en el presupuesto, a las políticas de la organización, a la necesidad de interoperabilidad con otros sistemas de software o hardware o a factores externos como los reglamentos de seguridad, las políticas de privacidad, entre otros. En la Tabla 11 se describen los requerimientos no funcionales.

ID	DESCRIPCIÓN
01	El sistema deberá funcionar en la plataforma Android.
02	El programa se desarrollará en la plataforma Android Studio (Java).
03	Se utilizará el emulador SDK para la creación de la aplicación.
04	La aplicación está preparada para funcionar en equipos de propósito general y acceder a las interfaces de red que dispongan.
05	La aplicación será mostrada al usuario mediante una interfaz gráfica de usuario de ventanas, con controles de fácil manipulación y que aparecerán en distintos paneles según la función que realicen.

Tabla 11 Requerimientos no funcionales

CAPÍTULO 6

Capítulo 6 DISEÑO DEL SISTEMA

En este capítulo se realiza el diseño del sistema, para resolver el problema planteado se aborda el diseño de la solución propuesta. El sistema queda comprendido por varios módulos en los cuales se realizaran las diferentes acciones del sistema.

Para la realización de la aplicación propuesta, se utilizaron diversas funciones contenidas en el softphone Sipdroid como base de nuestra aplicación y a partir de estas se fueron realizando las modificaciones necesarias para implementar las funciones de evaluación y mediciones de los parámetros QoS y almacenamiento en la base de datos.

Sipdroid es un software libre bajo la licencia GNU GPL Sipdroid. El proyecto, que ha comenzado con HSC en 2008 y luego llevado por ip-tel GmbH en 2009, sigue siendo la referencia y el pionero de los proyectos SIP en Android, ya que existe incluso antes de la llegada de las API de SIP (Android API 9). Lo más destacado de este cliente SIP es su compatibilidad con las versiones de Android desde 1.6 (Donut). Sipdroid MjSip se basa en: una tercera pila SIP escrito en Java en 2005 por Luca Veltri. Licencia pública general Hughes Systique Corporation (Estados Unidos) Universidad de Parma (Italia) Informe PCD 7 2012 – 2013.

En el presente proyecto de tesis, se propone el desarrollo de la aplicación titulada **TestingVoip**, que tiene como objetivo principal, realizar llamadas de voz sobre redes IP móviles y realizar las estadísticas de las métricas de calidad de servicio y evaluar los principales parámetros que determinan la QoS.

6.1 Casos de Uso

Los casos de uso son una técnica para especificar el comportamiento de un sistema. Son una secuencia de interacciones entre un sistema y alguien o algo que usa alguno de sus servicios. Cada caso de uso proporciona uno o más escenarios que indican cómo debería actuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico. (Torres, 2013)

Actor	Descripción	Acciones
Usuario	Es la persona que contesta/ realiza la llamada, guarda los datos y los visualiza	* Llamar * Responder * Terminar llamada * Guardar datos * Visualizar datos

Tabla 12. Casos de uso del usuario al sistema

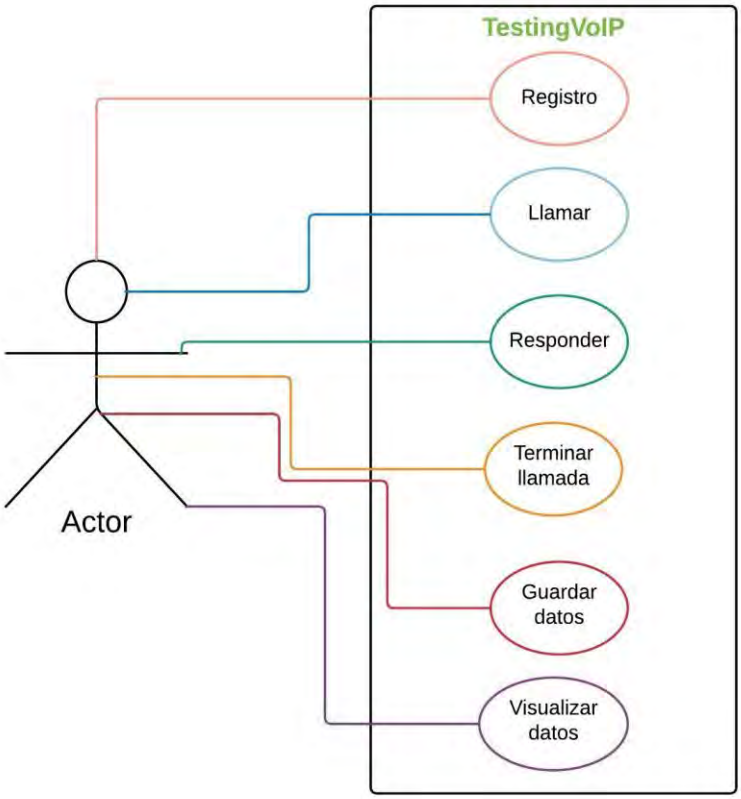


Ilustración 12. Caso de uso del usuario del sistema

A continuación se describen los casos de uso para los módulos que se desarrollan dentro de la aplicación TestingVoIP a partir de la Ilustración 14:

ID	**TestingVoIP1**
Nombre:	Registro
Descripción:	1. Se permite el registro del usuario, contraseña y servidor SIP. 2. Se concede permisos para realizar llamadas 3. Se realiza la selección de códec, puerto, protocolo y demás configuraciones básicas.
Actores:	Usuario
Precondiciones:	1. El usuario debió pedir su cuenta SIP con anterioridad.

Flujo:	2. El usuario debe ingresar usuario, contraseña y dominio, así como realizar la selección de las configuraciones de códec, puerto y protocolo.

Tabla 13. Caso de uso Registro

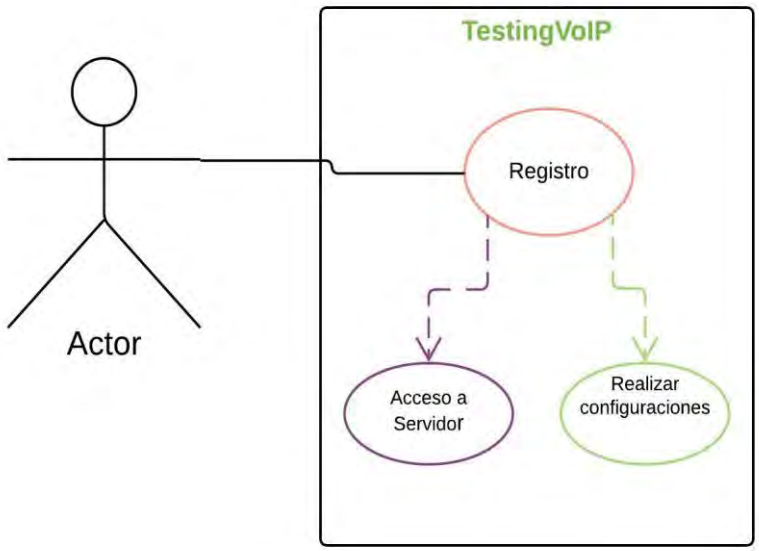


Ilustración 13. Caso de uso Registro

ID	**TestingVoIP2**
Nombre:	Llamar
Descripción:	1. Permite él opción de comunicación con un servidor SIP. 2. Permite la comunicación entre dos usuarios
Actores:	Usuario
Precondiciones:	1. La aplicación debe tener comunicación con el Servidor. 2. El usuario debe estar autenticado con el usuario y contraseña que le provee el servidor.
Flujo:	1. El usuario debe marcar el número de teléfono o extensión.

Tabla 14. Caso de uso Llamar

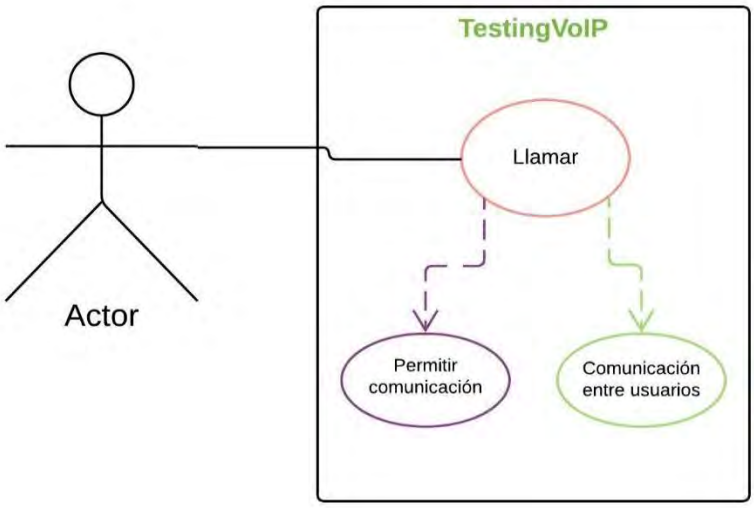


Ilustración 14. Caso de uso Llamar

ID:	TestingVoIP3
Nombre:	Responder
Descripción:	1. Permite la comunicación entre cliente/servidor. 2. Permite la comunicación entre dos usuarios.
Actores:	Usuario
Precondiciones:	1. La aplicación debe tener comunicación con el Servidor. 2. El usuario debe estar autenticado con el usuario y contraseña que le provee el servidor.
Flujo:	1. El usuario debe marcar debe aceptar la llamada.

Tabla 15 Caso de uso Responder

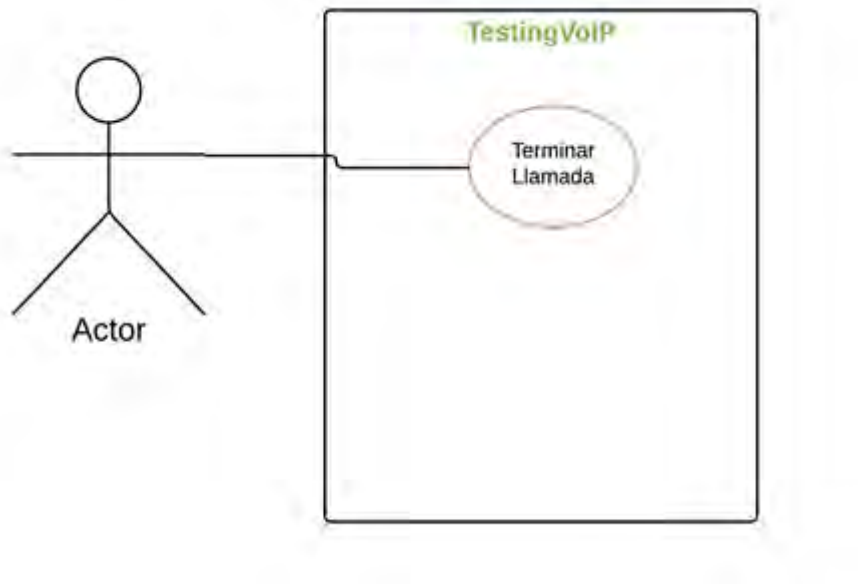


Ilustración 15. Caso de uso Terminar llamada

ID	**TestingVoIP4**
Nombre:	Visualizar Datos
Descripción:	1. Permite la visualización de los parámetros de QoS.
Actores:	Usuario
Precondiciones:	1. La aplicación debe haber terminado la llamada. 2. El usuario debe dirigirse a la sección para visualizar los resultados obtenidos.
Flujo:	

Tabla 16. Caso de uso Visualizar datos

ID	TestingVoIP5
Nombre:	Guardar Datos
Descripción:	1. Permite al usuario guardar los parámetros de QoS. 2. Permite guardar dichos parámetros en la base de datos local. 3. Permite guardar en la memoria externa del dispositivo móvil. 4. Permite visualizar la base de datos local.
Actores:	Usuario
Precondiciones:	1. La aplicación debe haber finalizado la llamada. 2. El usuario debe dirigirse a la sección para visualizar los resultados obtenidos.
Flujo:	1. El usuario debe terminar la llamada. 2. Dirigirse a la visualización de las métricas. 3. El usuario debe usar los botones para guardar. 4. Después podrá visualizar la base de datos en la aplicación y obtener un archivo .CSV que se guardara en la memoria del dispositivo

Tabla 17. Caso de uso Guardar datos

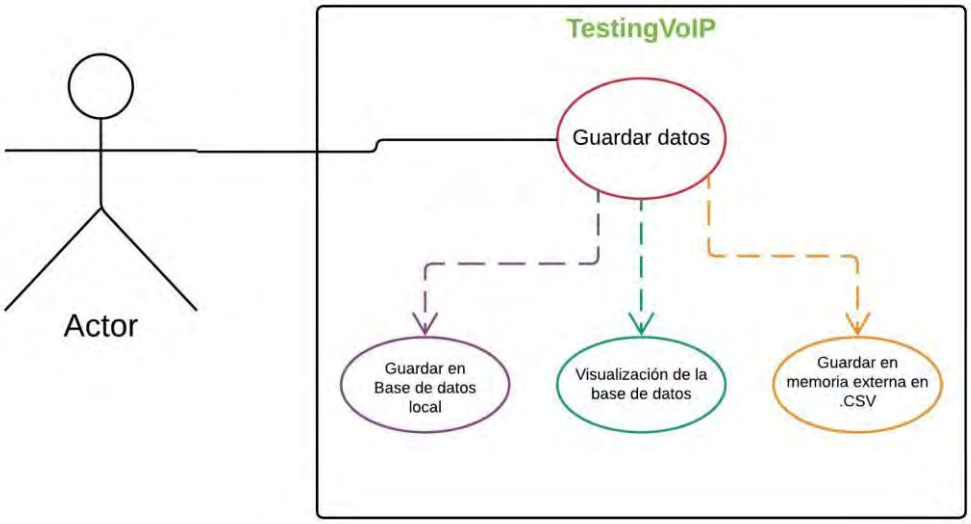


Ilustración 16. Caso de uso Guardar Datos

6.2 Funcionamiento del sistema

En este apartado se mostraràn las partes que conforman la aplicación y el funcionamiento de la misma. Primero se mostrara cómo funciona el sistema en conjunto.

Para tal fin, se realizó un diagrama de flujo que representa el funcionamiento de la aplicación, como se puede observar la pantalla principal le da acceso a las demás funciones de la aplicación, desde ella se realizan las llamadas, y desde allí se da acceso a la conexión de la cuenta SIP y los parámetros de QoS.

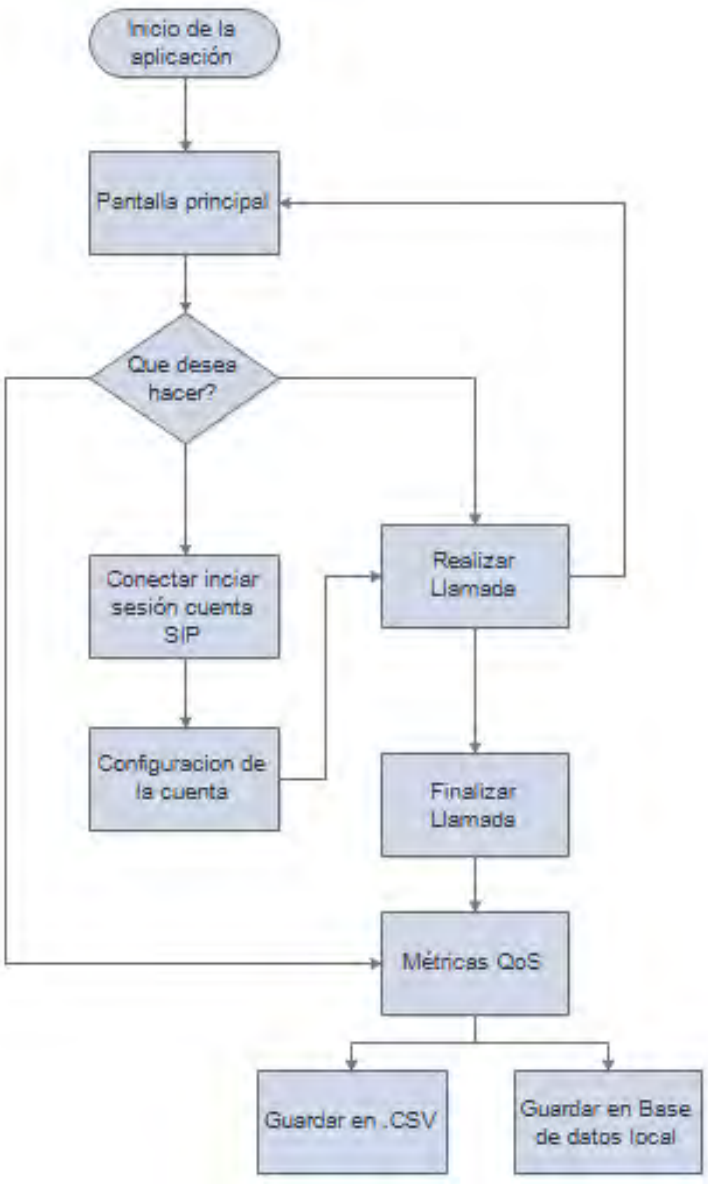


Ilustración 17. Diagrama de flujo del Funcionamiento del sistema

Este mismo funcionamiento puede ser explicado por medio de bloques como se muestra en la Ilustración 22, en donde se muestran tres módulos principales el transmisor, el receptor y las estadísticas.

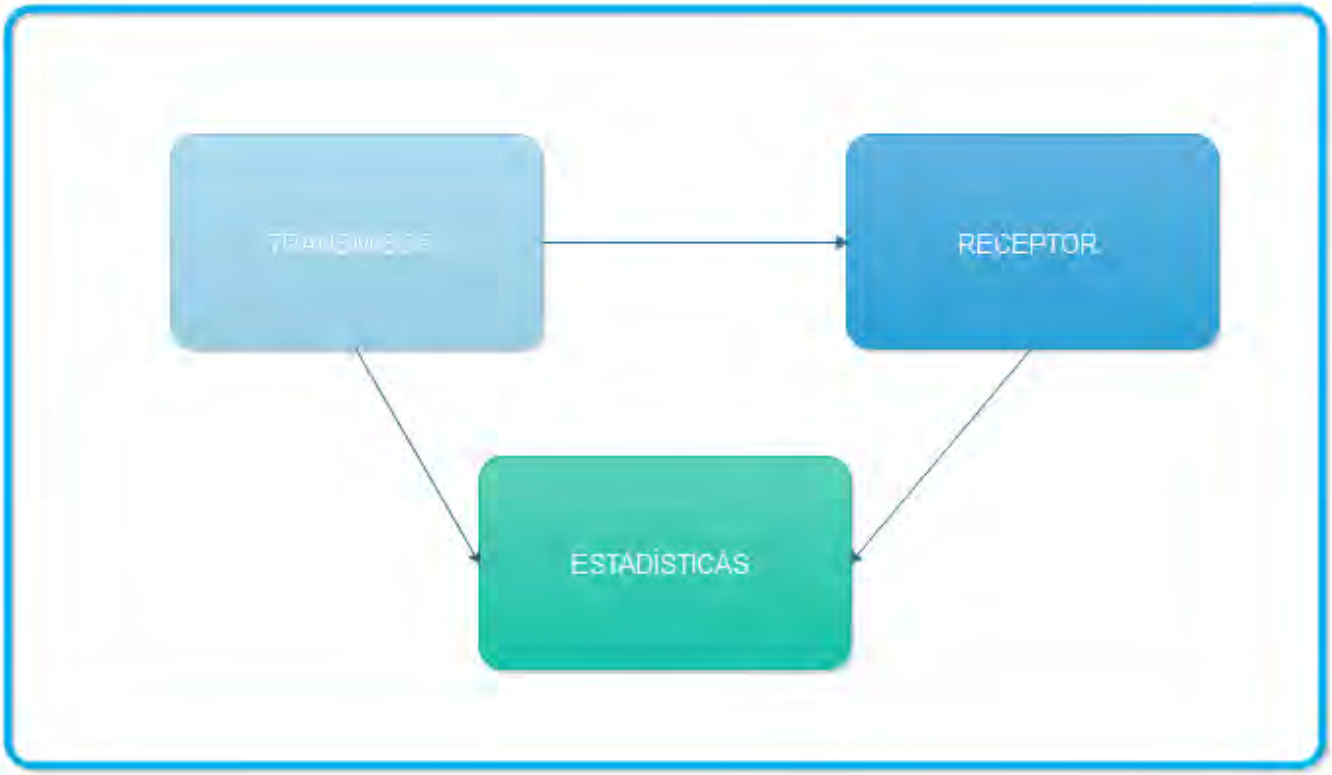
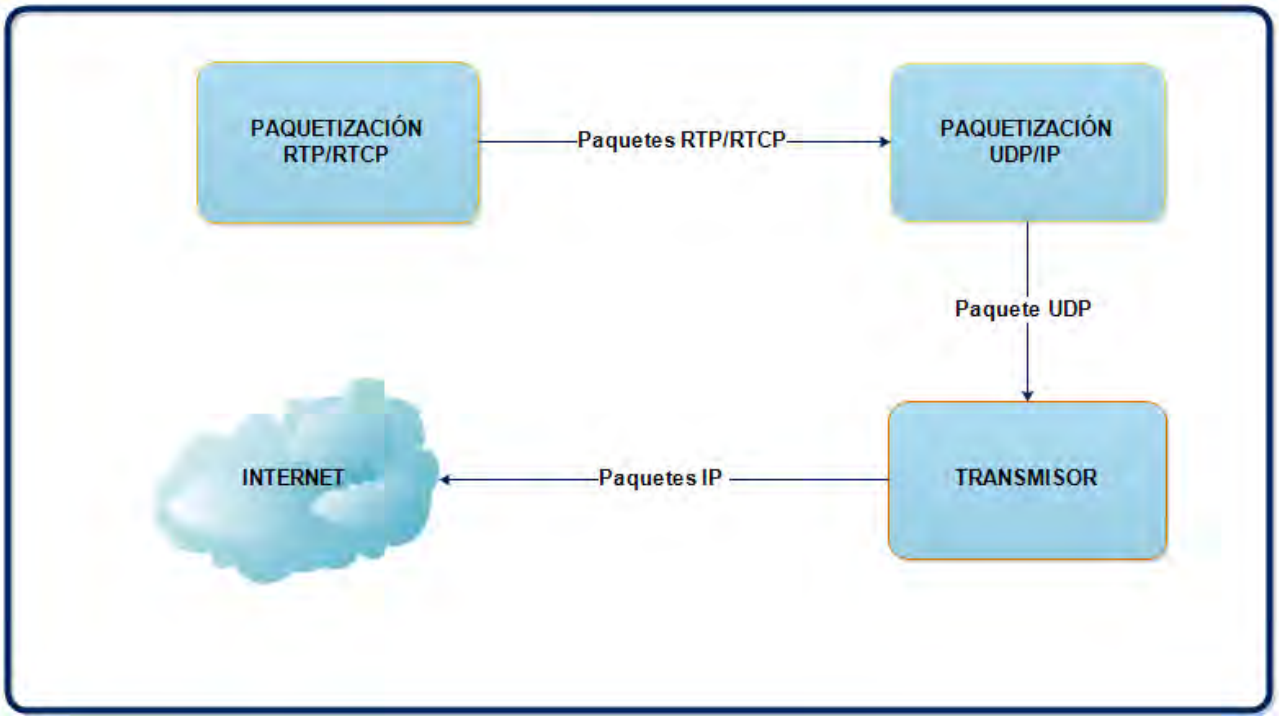


Ilustración 18. Diagrama de bloques

Módulo trasmisor: Es el encargado de realizar la comunicación y crear los paquetes, el transmisor realiza la paquetización RTP, en función del códec que el usuario defina (en las pruebas se usó el Códec G711) y este a su vez se encapsula en un paquete UDP para ser transmitido a través de un socket a la red IP.



Módulo receptor: Se encarga de recibir los paquetes por medio de un buffer, suprimir los

Ilustración 19. Diagrama del módulo transmisor

encabezados, extraer las muestras de voz y enviar el encabezado RTP al bloque de estadísticas, este módulo realiza la función inversa del módulo transmisor.

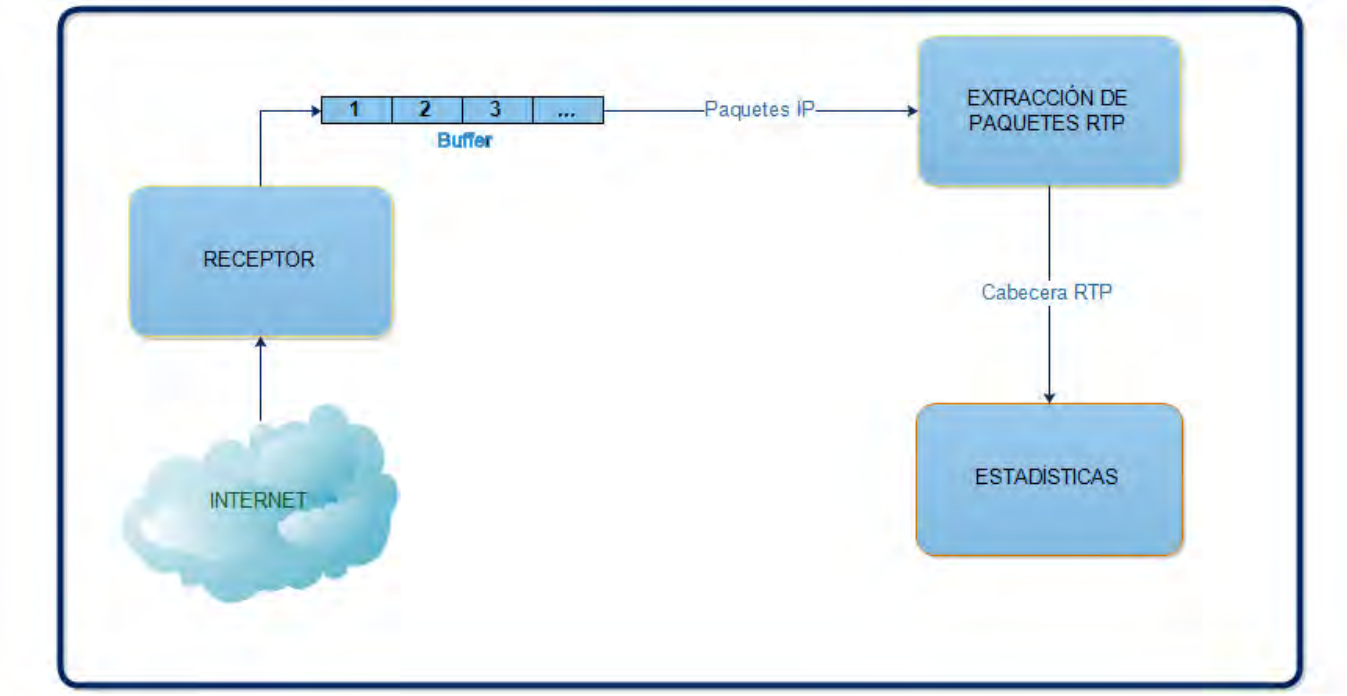


Ilustración 20. Diagrama del módulo receptor

Estadísticas QoS: es este módulo se recibe el encabezado RTP, del cual se toman los datos necesarios para obtener los parámetros de QoS.

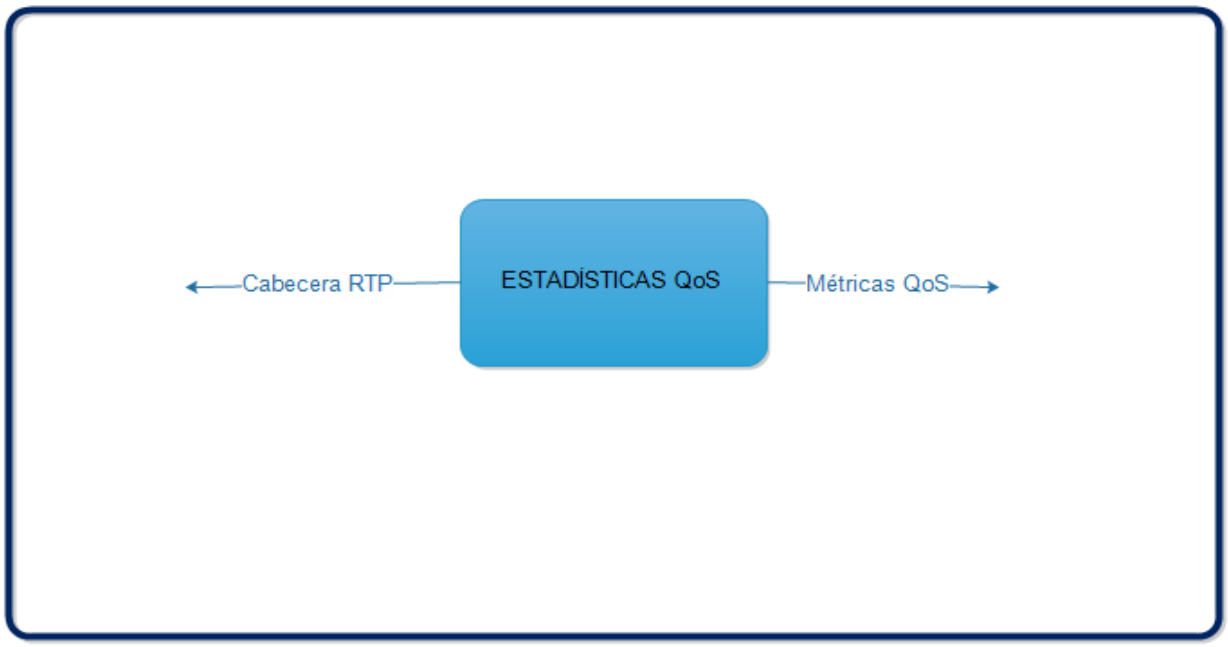


Ilustración 21. Diagrama del módulo de estadísticas

6.3 Arquitectura del Sistema

En el presente subcapítulo, centraremos nuestra atención en el diagrama de clases y paquetes del sistema implementado, el sistema se desarrolló en lenguaje Java en Android Studio, por lo cual se utilizó un prototipo orientado a objetos.

Para empezar se muestra un diagrama de paquetes del programa y cómo el sistema está dividido en agrupaciones lógicas y las dependencias entre esas agrupaciones.

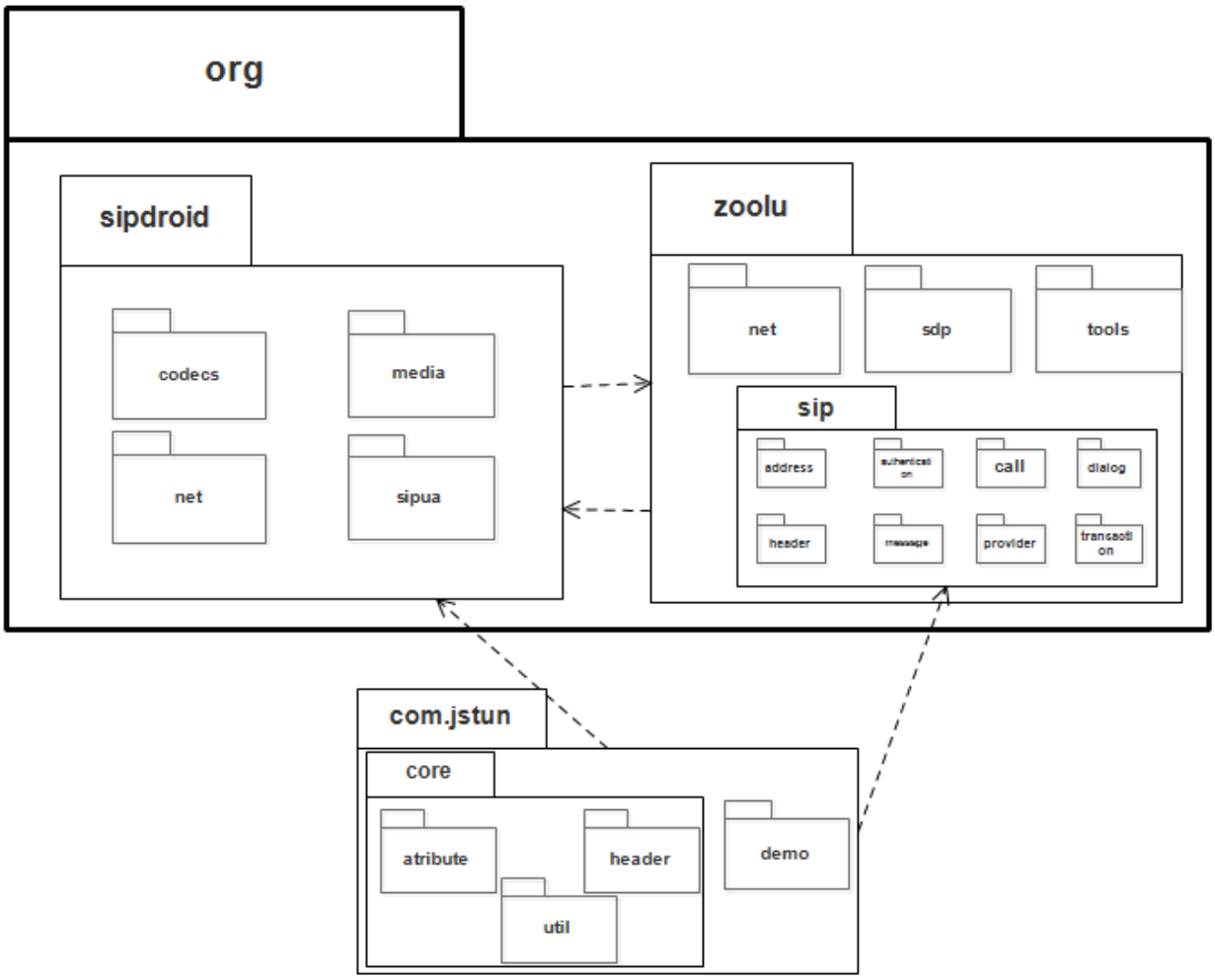


Ilustración 22. Diagrama UML de paquetes

En la Tabla 19, se muestra el nombre y la descripción de cada uno de los paquetes y su función que realiza en el sistema de la aplicación.

Nombre del Paquete	Descripción
Sipdroid	Este módulo contiene los paquetes de codecs, media, net y sipua.
Codecs (Sipdroid)	Este paquete contiene la interfaz básica para las clases de codecs. Los codecs implementan capacidades de codificación y decodificación básicas.

Media(Sipdroid)	Este módulo representa la Interfaz para las clases que inician aplicaciones de los medios de audio o video.
net(Sipdroid)	Este módulo contiene el socket de flujo de datos, la estimación de parámetros, la base de datos SQlite y el guardado externo de los datos.
Sipua(Sipdroid)	Este módulo contiene los paquetes para los registros de números telefónicos, usuarios y llamadas.
Nombre del Paquete	**Descripción**
Zoolu	Este módulo contiene los paquetes para los registros la comunicación de las llamadas, servidores, autentificación y demás herramientas necesarias en la conexión.
net (Zoolu)	Este módulo contiene los paquetes de los sockets, los servidores TCP, UDP y sus conexiones.
Sdp(Zoolu)	Este paquete maneja las clases de los cuerpos de mensaje SIP formateados de acuerdo con el protocolo de descripción de sesión (SDP).
Sip(Zoolu)	Este paquete contiene las clases para el manejo del protocolo SIP, el direccionamiento, autenticación, llamada, diálogos, cabeceras, entre otros.
Tools(Zoolu)	Este paquete contiene una colección de métodos estáticos para manejar archivos y archivos jar.
Nombre del Paquete	**Descripción**
com.jstun	Este paquete contiene las clases para el envío de mensajes.
Nombre del Paquete	**Descripción**
Res	Este paquete contiene todas las clases de recursos necesarios para el proyecto: imágenes, *layouts*, cadenas de texto, etc

Tabla 18. Descripción de paquetes de la aplicación

En este apartado se muestran los paquetes con sus clases un poco más extendidas y explicadas y también las dependencias entre cada paquete y clase.

6.3.1 Paquete Sipdroid

Este módulo contiene los paquetes de codecs, media, net y sipua, cada uno de ellos contiene paquetes con diferentes funciones que se irán explicando más adelante.

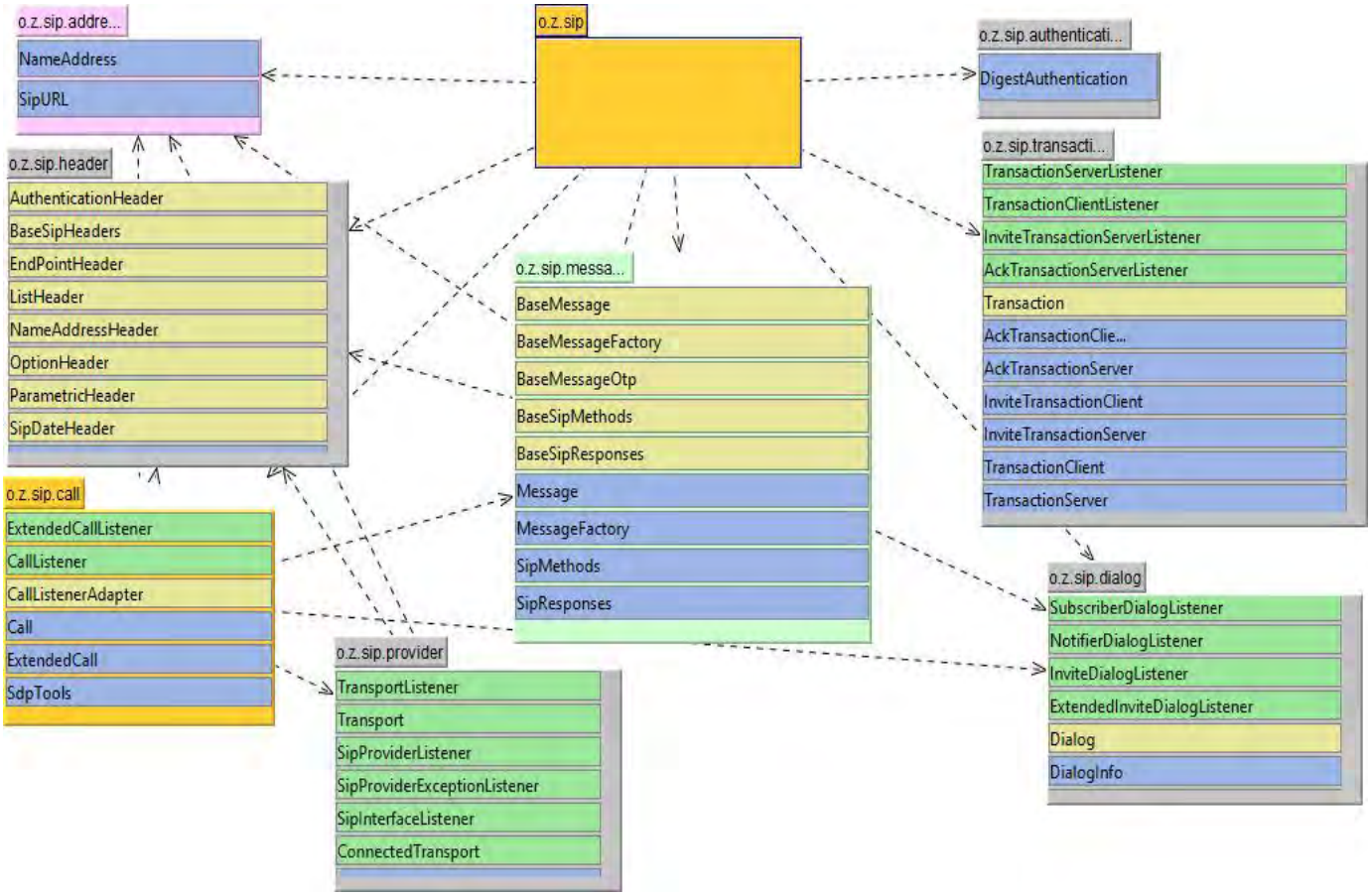


Ilustración 23. Diagrama UML de paquetes de Sipdroid

6.3.1.1 Paquete Codecs

El paquete codecs contiene la interfaz básica para las clases de Códec. Los codecs implementan capacidades de codificación y decodificación básicas. Los codecs que se encuentran activos son del códec g711 alaw y ulaw, los codecs GSM y g722 se encuentran desactivados.

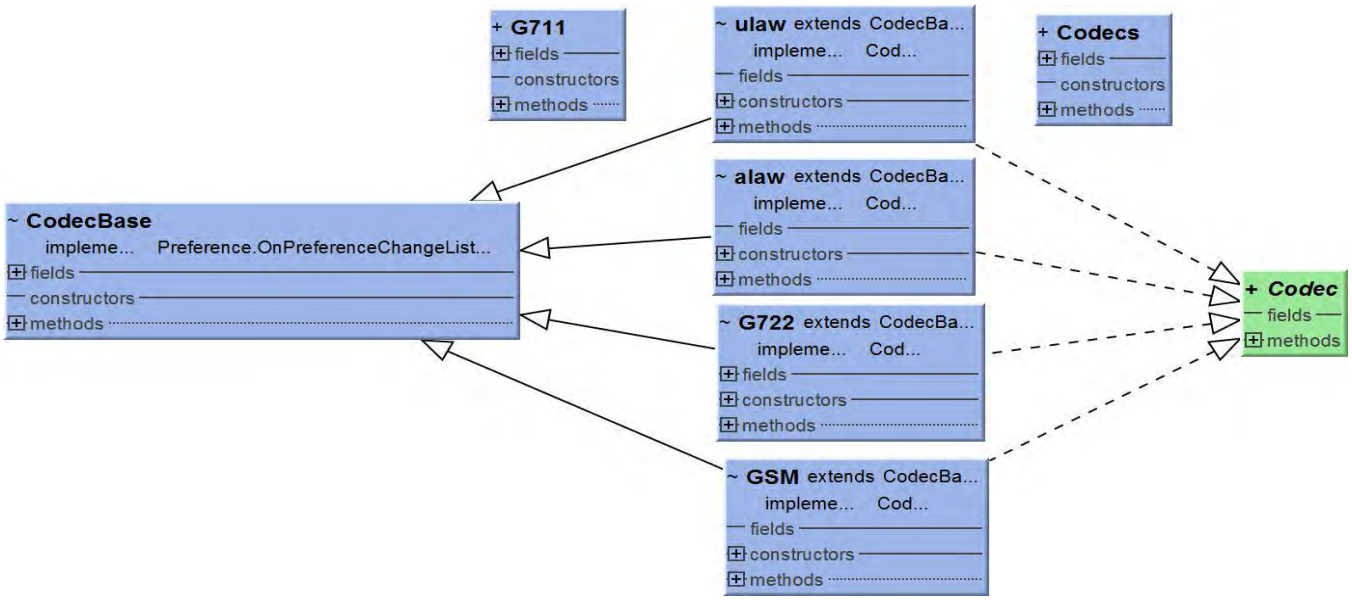


Ilustración 24. Diagrama UML de clases del paquete codecs

6.3.1.2 Paquete media

Este módulo representa la interfaz para las clases que inician aplicaciones de los medios de audio o video. En este se encuentran los paquetes para la recepción y envió de la transmisión de secuencias protocolo RTP.

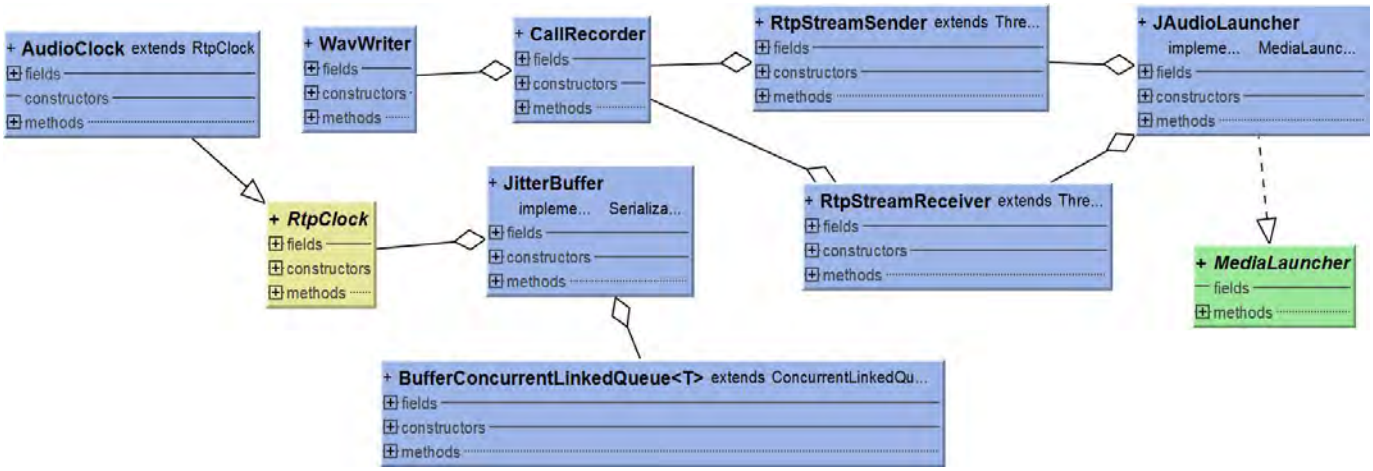


Ilustración 25. Diagrama UML de clases del paquete media

6.3.1.3 Paquete net

Este paquete contiene los socket del flujo de datos, la estimación de parámetros, la base de datos SQlite y el guardado externo de los datos.

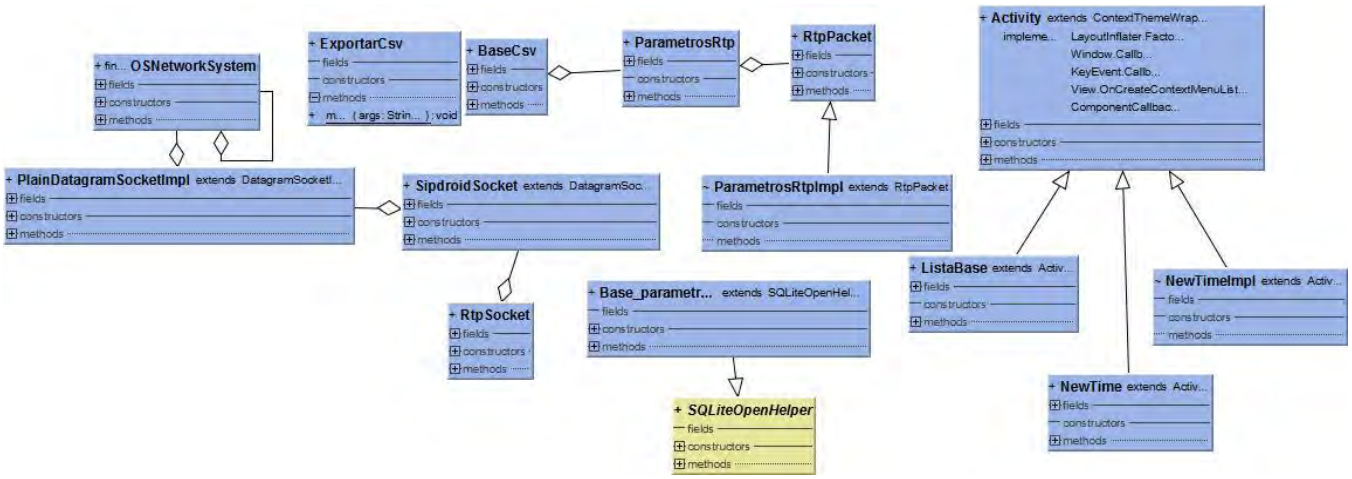
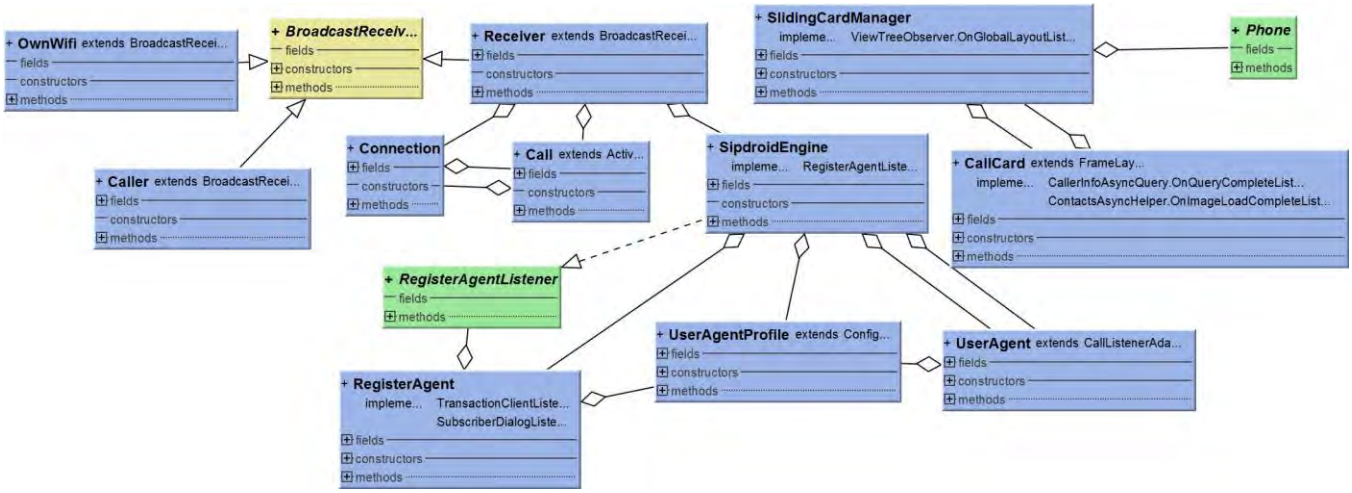


Ilustración 26. Diagrama UML de clases del paquete net

6.3.1.4 Paquete sipua

Este módulo contiene los paquetes para los registros de números telefónicos, usuarios y llamadas.



6.3.2 Zoolu

Esta carpeta contiene los paquetes net, sip, sdp y tools en ellos se contienen los registros de números telefónicos, usuarios y llamadas.

6.3.2.1 Net(Zoolu)

Este módulo contiene los paquetes de los sockets, los servidores TCP, UDP y sus conexiones.

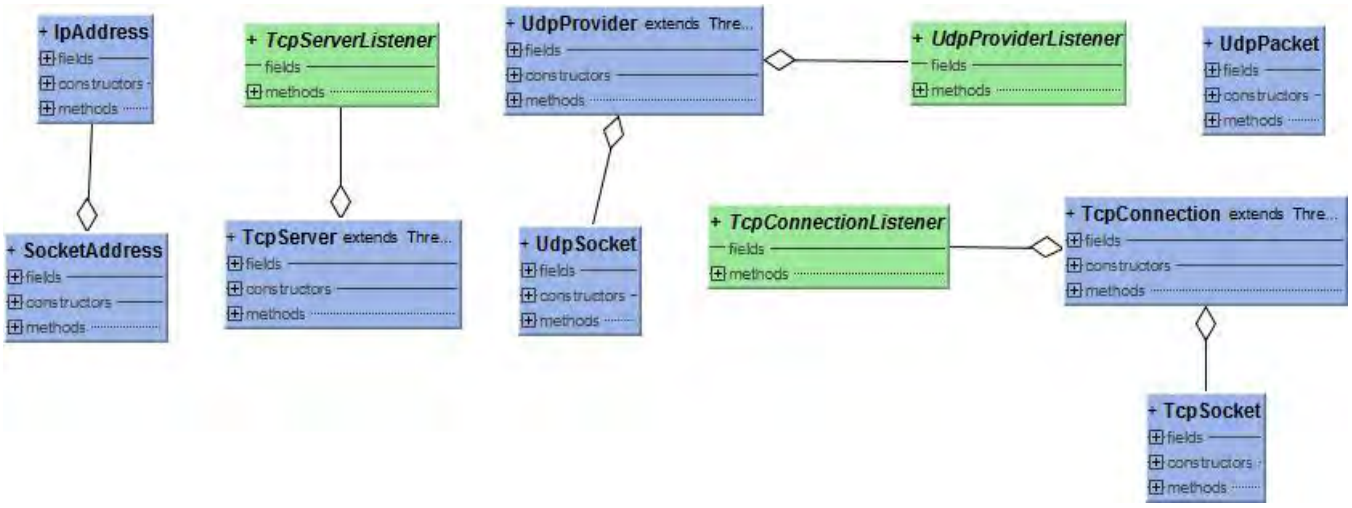


Ilustración 27. Diagrama UML de clases del paquete net (zoolu)

6.3.2.2 Paquete sdp (Zoolu)

Este paquete maneja las clases de los cuerpos de mensaje SIP formateados de acuerdo con el protocolo de descripción de sesión (SDP).

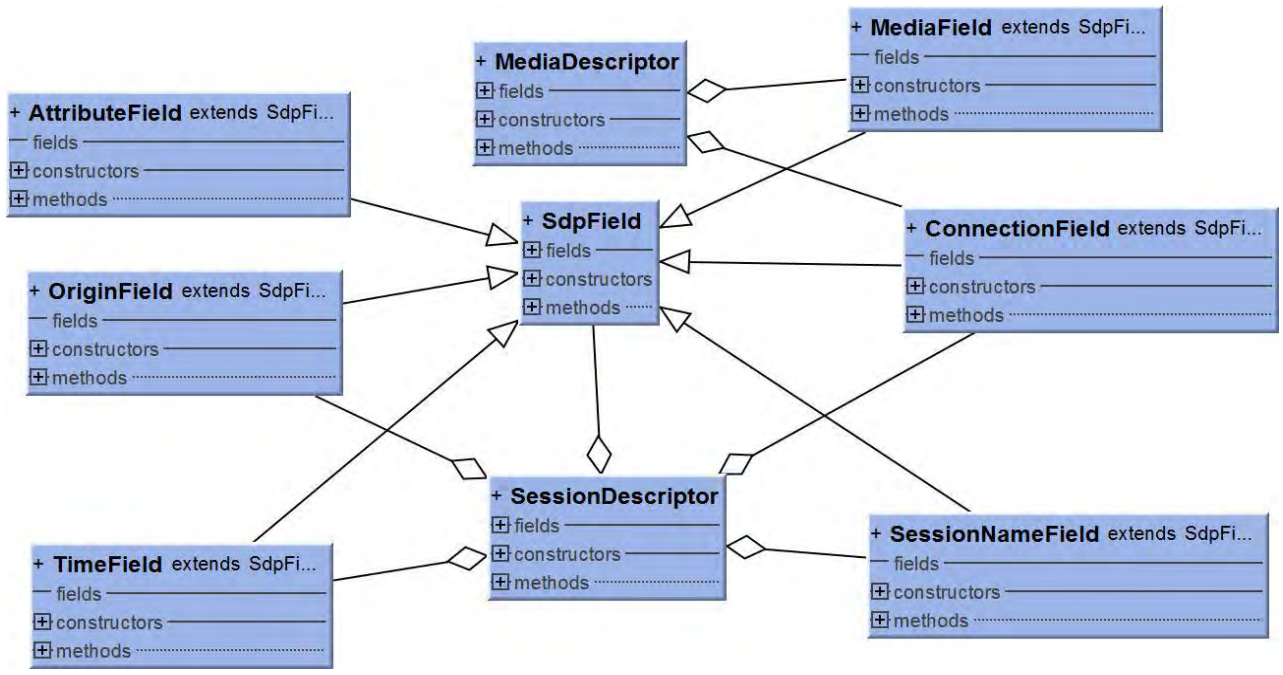


Ilustración 28. Diagrama UML de clase del paquete sdp

6.3.2.3 Paquete sip(Zoolu)

Este paquete contiene las clases para el manejo del protocolo SIP, el direccionamiento, autenticación, llamada, diálogos, cabeceras, entre otros.

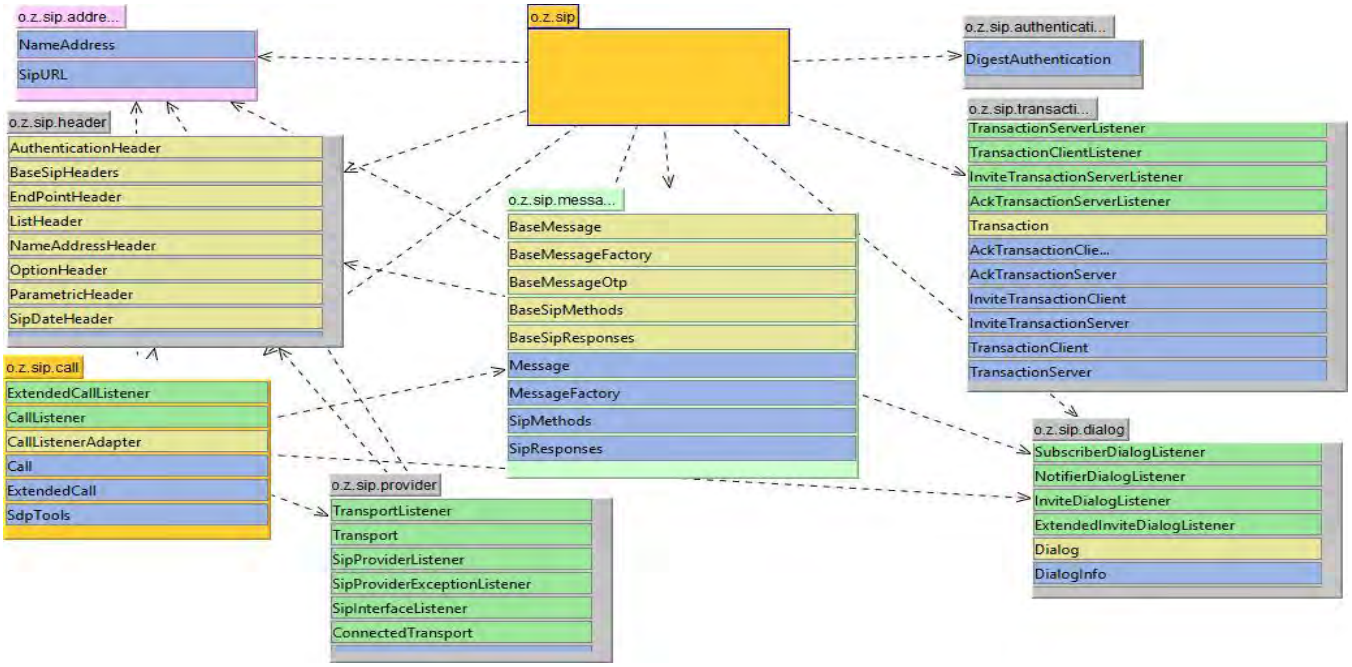


Ilustración 29. Diagrama UML de paquetes de sip

6.3.2.3 Paquete tolos(Zoolu)

Este paquete contiene una colección de métodos estáticos para manejar archivos y archivos jar.

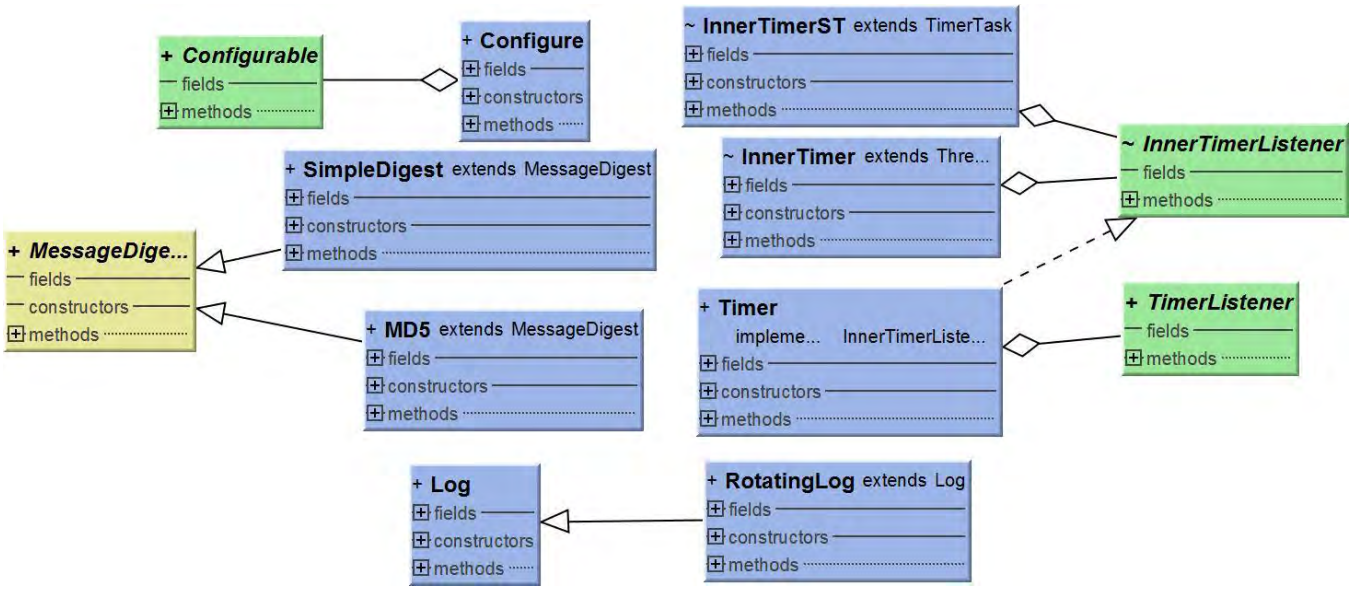


Ilustración 30. Diagrama UML de clases del paquete tools

6.3.3 Paquete Com.jstun

Este paquete contiene las clases para el envío de mensajes.

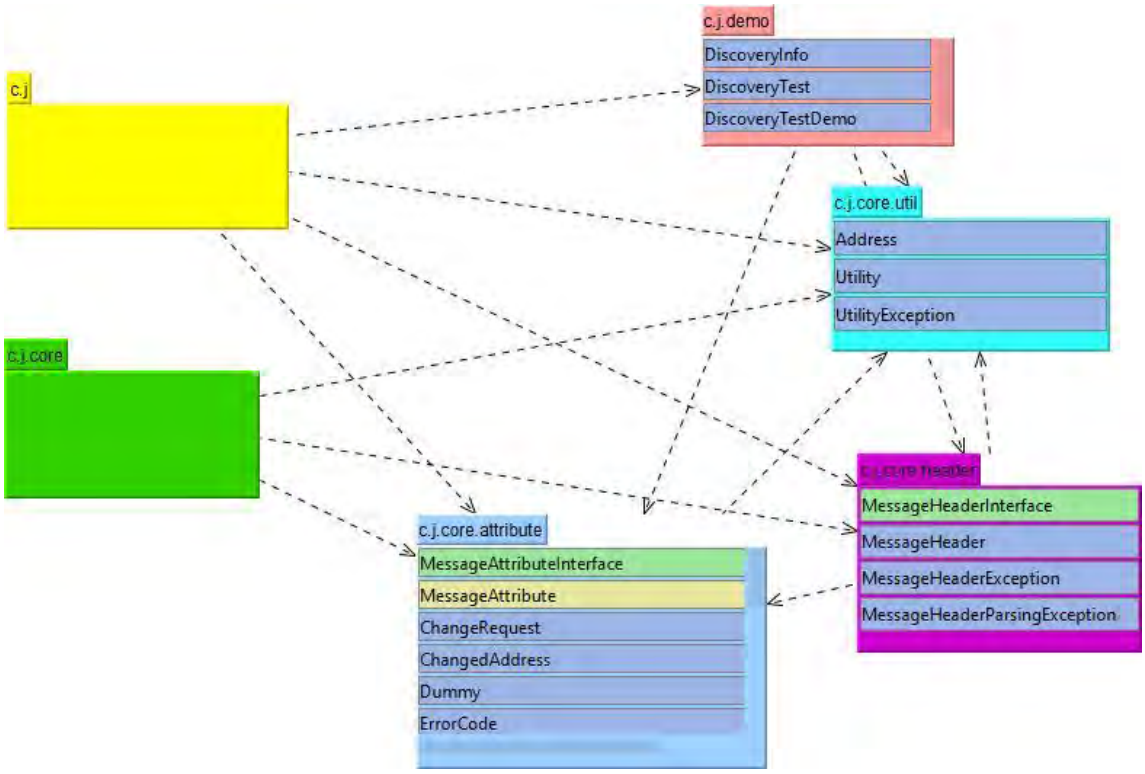


Ilustración 31. Diagrama UML de paquetes de com.jstun

6.4 Interfaz del sistema

La interfaz de usuario de una aplicación es todo aquello que el usuario puede ver y todo aquello con lo que este puede interactuar. Android ofrece una variedad de componentes de IU previamente compilados, como objetos de diseño estructurados y controles de IU que permiten compilar la interfaz gráfica de usuario para una aplicación. Android también ofrece otros módulos de IU para interfaces especiales, como diálogos, notificaciones y menús. La interfaz del sistema consta de 5 pantallas principales:

6.4.1 Pantalla de Inicio

En la pantalla de inicio se encuentra la modulo principal para llamadas y los botones que dan acceso al registro y configuraciones, agenda, salida y métricas QoS.

Ilustración 32. Pantalla de inicio de la aplicación

Botón de configuraciones: el botón color gris en forma de engrane lleva a las configuraciones del programa para realizar la llamada. En este se configuran el usuario y servidor SIP, además de la elección del tipo de códec, notificaciones, audio y video.

Botón de contactos: el botón azul en forma de teléfono envía al directorio de contactos del teléfono.

Panel para realizar las llamadas: en esta parte se puede teclear directamente el número de teléfono o extensión para realizar las llamadas.

Botón de parámetros de calidad de servicio: este botón permite el acceso a las estadísticas de calidad de servicio para visualizarlas y poder realizar el guardado y visualización de la base de datos local y el guardado en la memoria externa del dispositivo.

Botón para salir: este botón permite salir de la aplicación y hace que finalice el servicio de la misma.

6.4.2 Pantalla de Parámetros de Calidad de Servicio

En esta pantalla se accesa a un panel para visualizar los parámetros de QoS, los botones para guardar en la base de datos local, visualizar y guardar en .csv y para salir.

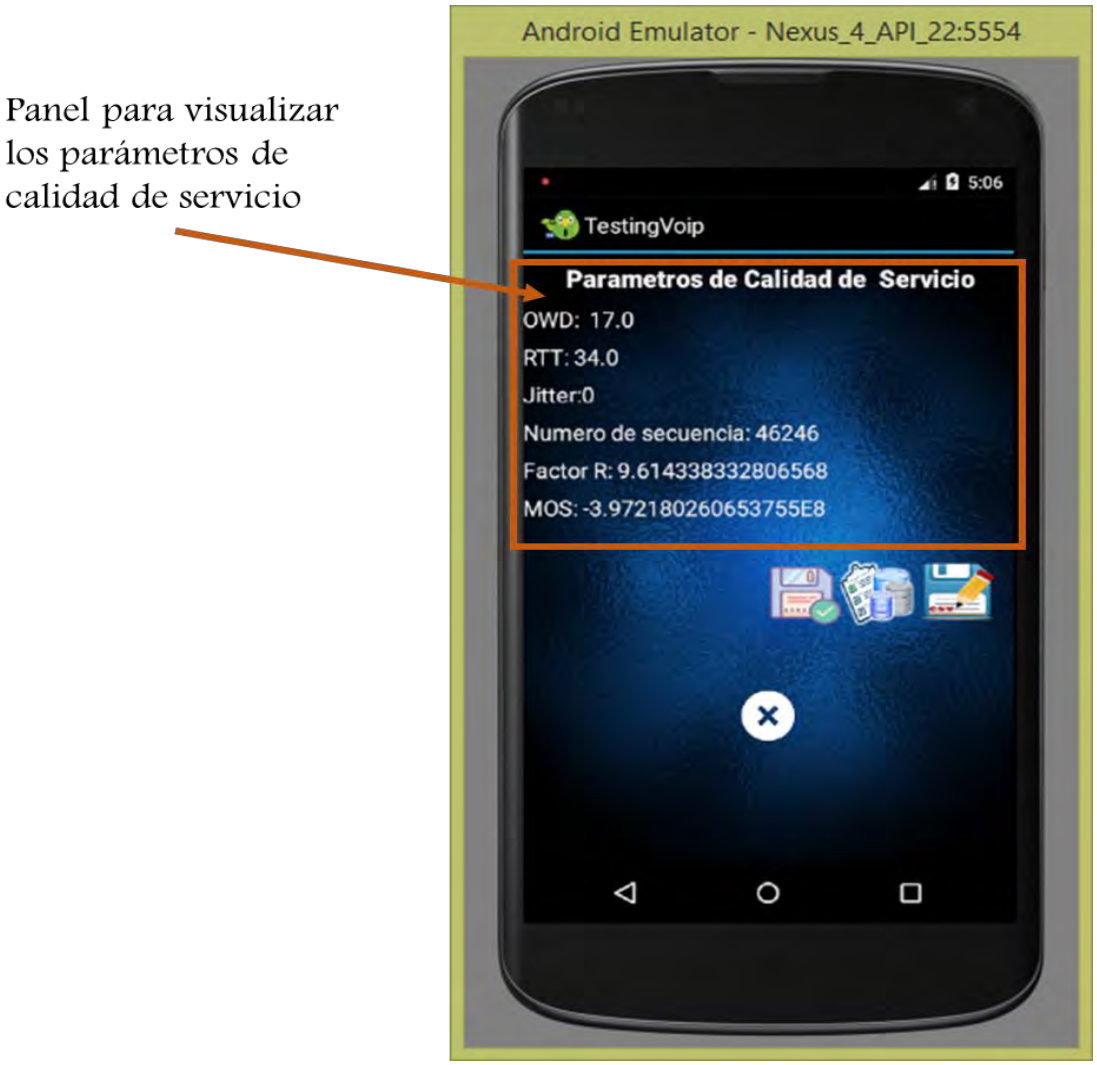


Ilustración 33. Pantalla de parámetros

Ilustración 34. Botones para guardar datos

Panel de visualización: en este panel se visualizan los parámetros de calidad de servicio (QoS).

Botón base de datos: este botón sirve para guardar los parámetros que se encuentran visualizados en la base de datos local.

Botón visualizar: este botón sirve para visualizar la base de datos local.

Botón guardar .csv: este botón sirve para guardar los parámetros que se visualizan en un archivo .csv en la memoria externa del teléfono.

Botón salir: este botón sirve para salir de la pantalla actual y regresar a la pantalla principal.

6.3.2.1 Base de datos local

En esta pantalla se visualizan los parámetros de calidad de servicio que se han guardado en la base de datos local.

Para la realización de la base de datos local se utilizó una base de datos en SQLite, esta se creó pensando en que se pueda enlazar a una la base de datos remota en SQL y exportar la información de forma permanente.

Para esto se realizó la base de datos Base_parametros para registrar las mediciones realizadas en tiempo real.

Base de datos local

Ilustración 35. Pantalla de la base de datos

6.4.3 Pantalla para configuración de servicios

En esta pantalla se realiza la configuración de servicios, como las cuentas SIP, las opciones de las llamadas, notificaciones, las opciones de audio y video, los tipos de codecs de audio y conexiones inalámbricas.

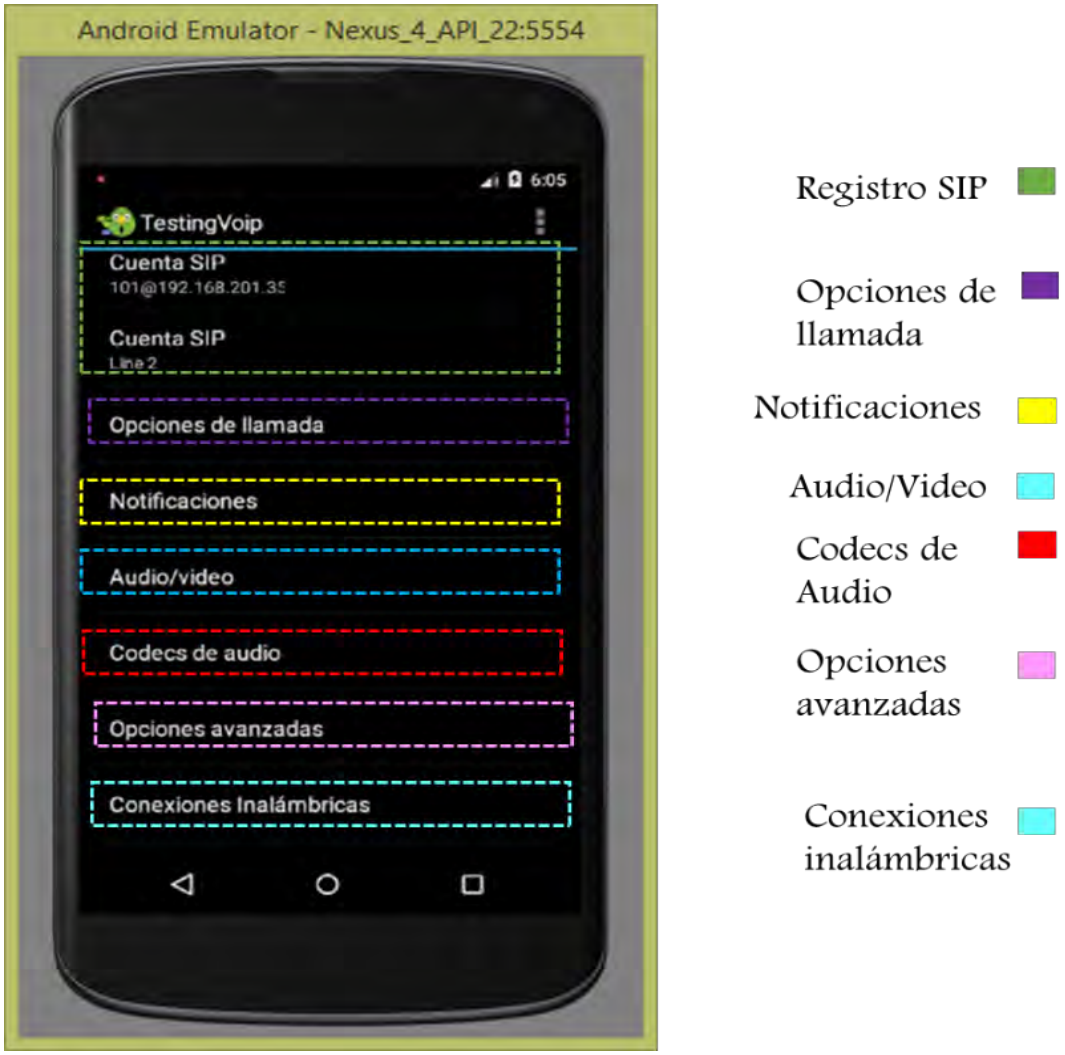


Ilustración 36. Configuración de servicios

En esta pantalla se configura la cuenta SIP, se debe introducir el usuario, la contraseña, el servidor y dominio, el puerto, el protocolo y seleccionar si se desea usar WLAN u otra.

Pantalla para configuración de la cuenta SIP

Ilustración 37. Cuenta SIP

En estas pantallas se muestran las configuraciones que pueden realizarse a las llamadas y las notificaciones.

Pantalla para las opciones de llamada

Pantalla para configuración de las notificaciones

Ilustración 38. Configuración de llamada y notificaciones

En estas pantallas se muestran las configuraciones que pueden realizarse a las opciones de audio y video y la pantalla para elegir el tipo de codec a utilizar.

Pantalla para las opciones de audio y video

Pantalla para las opciones códecs

Ilustración 39. Configuración de codecs y audio y video

En estas pantallas se muestran las configuraciones que pueden realizarse a las opciones de conexiones inalámbricas y la pantalla para elegir opciones avanzadas.

Pantalla para las opciones de conexiones inalámbricas

Pantalla para las opciones avanzadas

Ilustración 40. Configuración de las conexiones inalámbricas y opciones avanzadas

CAPÍTULO 7

Capítulo 7 PRUEBAS Y EVALUACIÓN DEL SISTEMA

Con la finalidad de probar y evaluar la aplicación desarrollada “TestingVoIP” se realizaron pruebas de conectividad y funcionamiento del sistema. Para la evaluación del sistema, probar la conectividad y evaluar las métricas QoS se realizó el siguiente escenario y se utilizó la red wlcampus del Universidad de Quintana Roo, con el objetivo de evaluar la calidad de servicio de la red en presencia de tráfico de voz sobre IP móvil (mVoIP).

Para lograr lo antes mencionado, se utilizó el analizador de tráfico Wireshark y se implementó un servidor Elastix. Wireshark permitirá comprobar la correcta implementación del protocolo SIP y de los cofificadores de voz, menconados anteriormete.

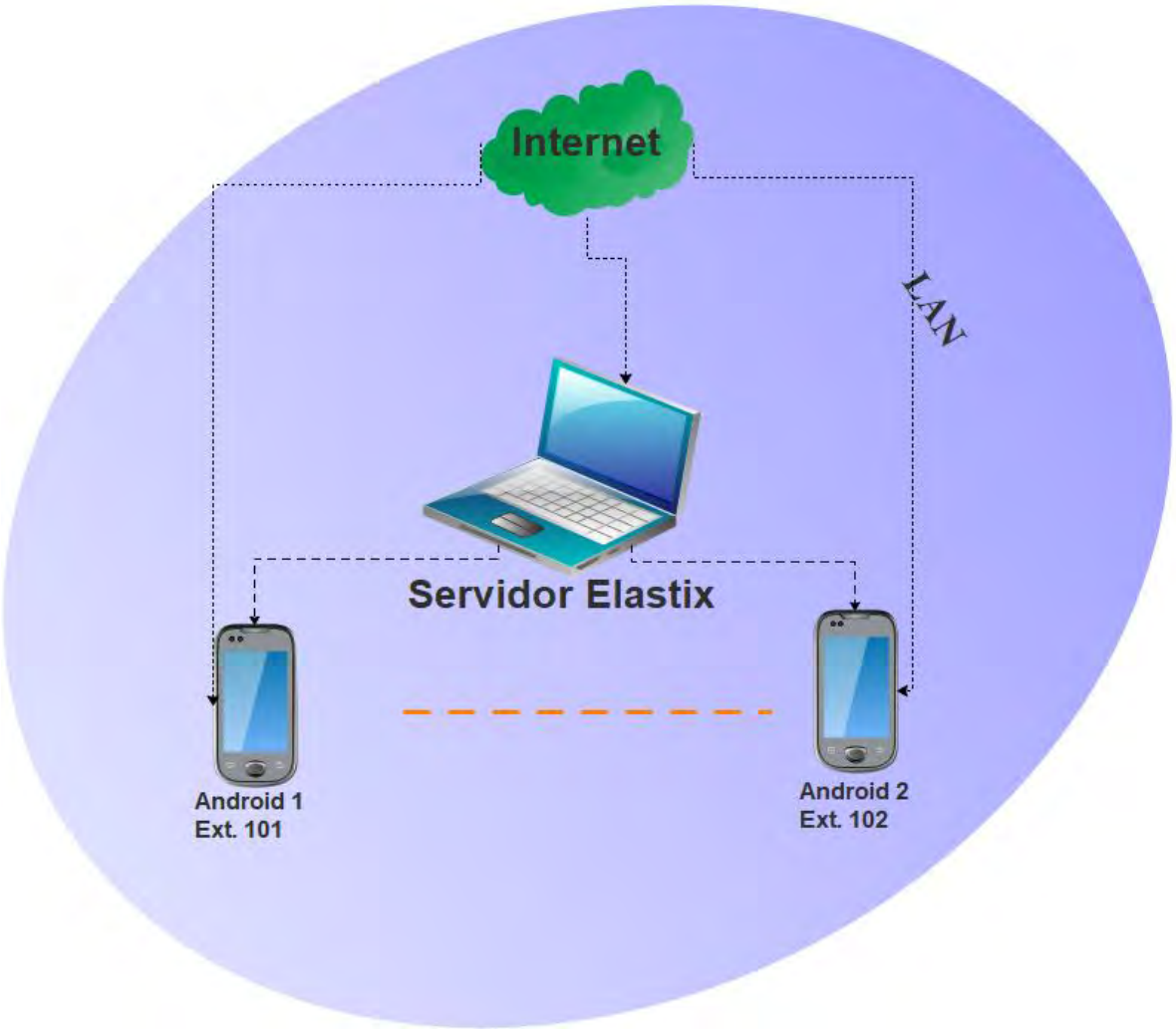


Ilustración 41. Escenario de prueba

Para la implementación del servidor se utilizó una computadora portatil en las cual se levantó un Servidor Elastix con una arquitectura SIP en Centos 5.10.

Ilustración 42. Pantalla de inicio Elastix

El servidor con la dirección 192.168.1.67 se instaló en una máquina virtual en VirtualBox.

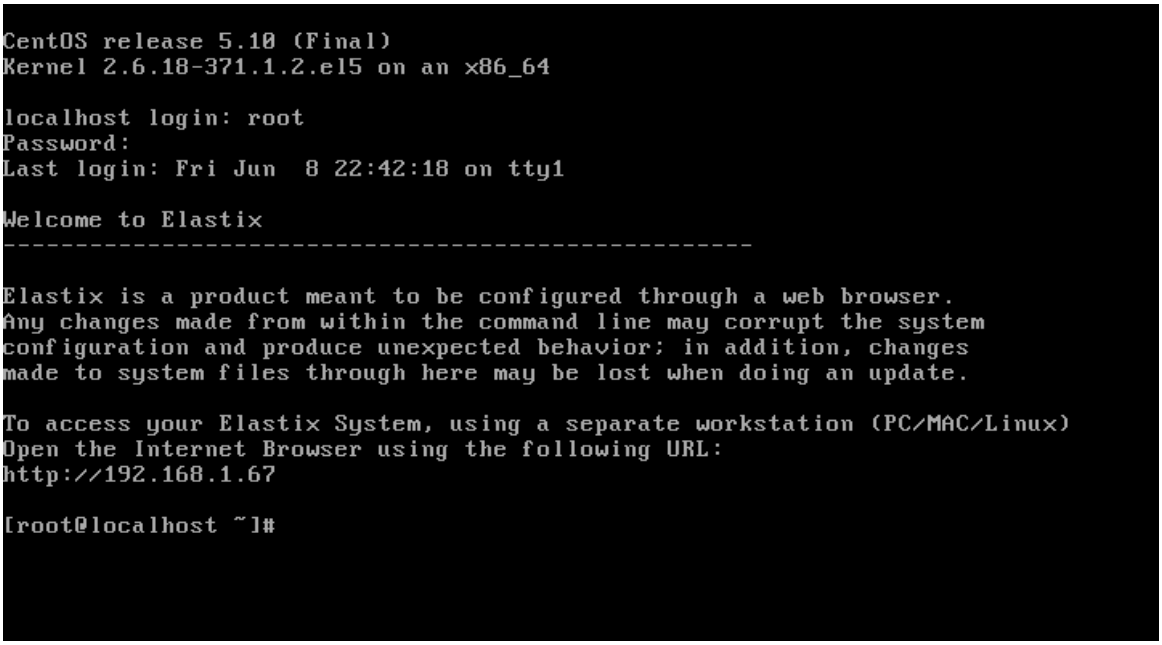


Ilustración 43. Dirección del Servidor

Para realizar las pruebas se crearon los usuarios Android 1 y Android 2 con las extensiones 101 y102 respectivamente como se muestra en la captura.

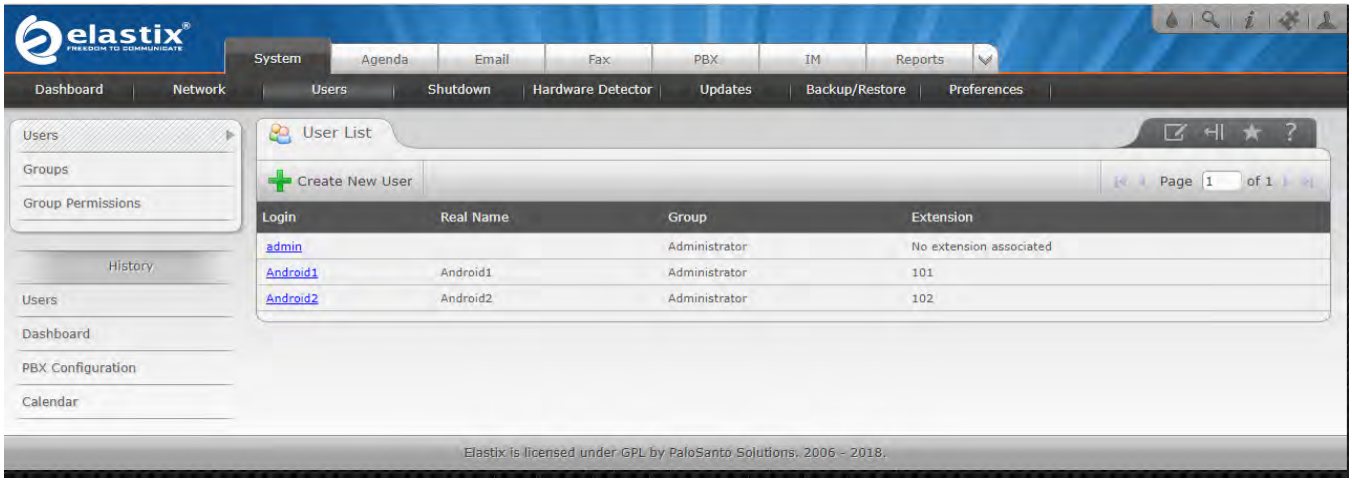


Ilustración 44. Creación de usuarios y extensiones de Elastix

También se utilizaron dos dispositivos móviles donde se instaló la aplicación TestingVoIP y se configuraron las cuentas de los usuarios Android 1 y Android 2.

Ilustración 45. Cuenta Android 1

Ilustración 46. Cuenta Android 2

7.1 Mediciones y Resultados

En el proceso de las pruebas, se realizaron un conjunto de llamadas y se utilizó el analizador de protocolos Wireshark para comprobar los codecs, los protocolos y las direcciones de origen y destino. La Ilustración 51, muestra la pantalla principal de wireshark

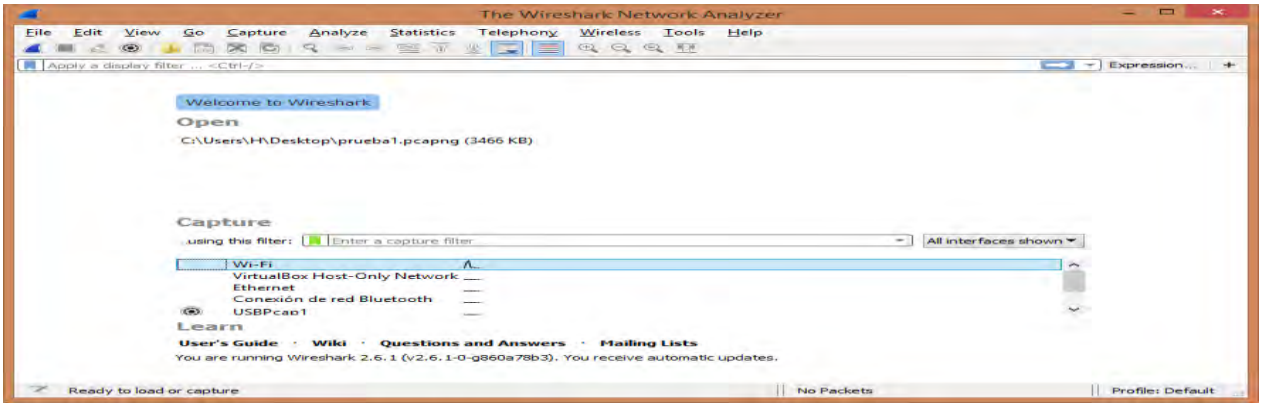


Ilustración 47. Pantalla principal de Wireshark

Después se aplicó un filtro para visualizar exclusivamente las llamadas de mVoIP.

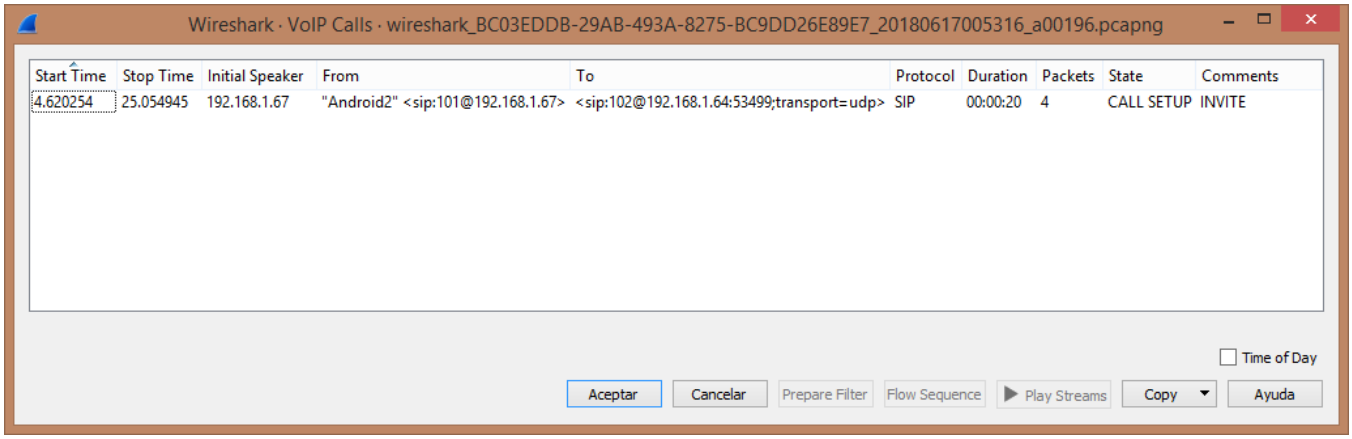


Ilustración 48. Filtro de llamadas de voz

Como se puede observar en la Ilustración 53, el protocolo de transporte usado es UDP.

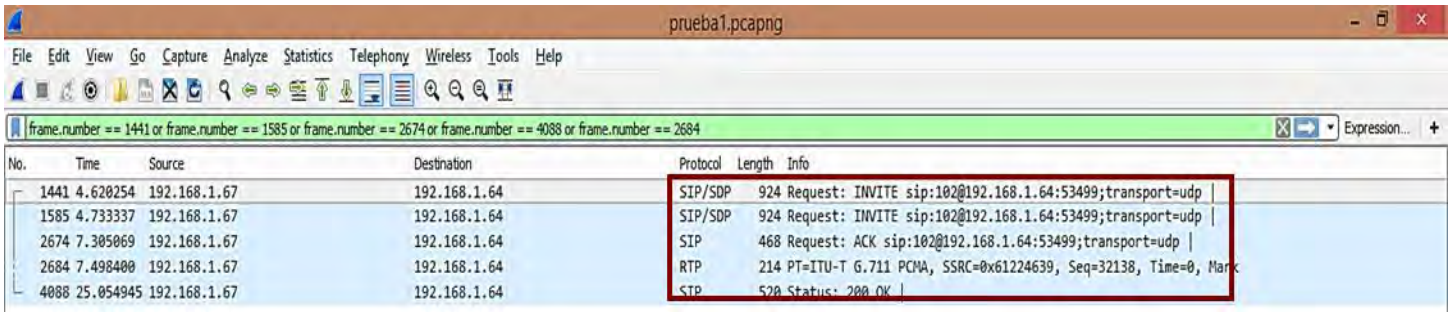


Ilustración 49. Filtro de UDP

En la Ilustración 54 se puede observar el tipo de codec configurado en la aplicación (G711 PCMA) y de igual forma en la Ilustración 53 y 55 se puede validar en la captura de Wireshark la utilización del códec G711 PCMA en la llamada de prueba.

Ilustración 50. Uso de G711 PCMA como codec en la aplicación

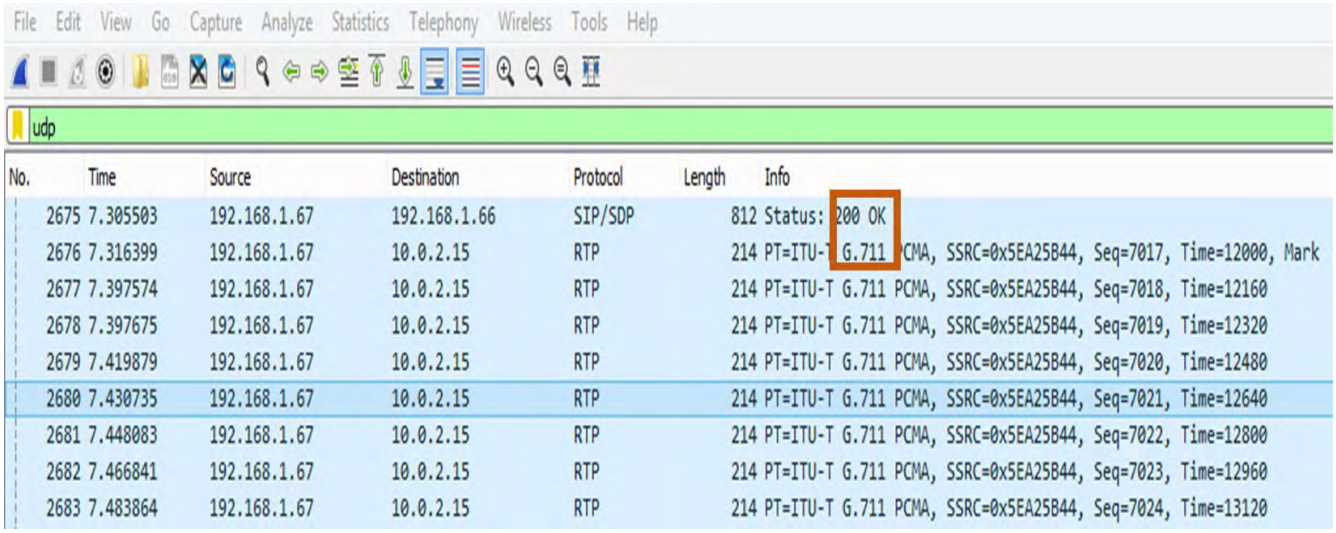


Ilustración 51. Uso de códec G711 PCMA en la llamada de prueba

Como se puede ver la aplicación desarrollada cumple con los parámetros que se están configurando y que se aplican al momento de realizarse las llamadas.

Como último punto, se evaluaron las principales métricas de calidad de servicio, para tener valores de referencia, se realizó de manera paralela mediciones mediante el analizador de protocolos comercial llamado CommView y se compararon los resultados obtenidos, como se muestran en las Tablas 20-25.

MOS

Android 1	CommView	Android 2	Commview
4.0616	3.965	4.1568	3.8521

Tabla 19. Evaliación del MOS

FACTOR R

Android 1	CommView	Android 2	Commview
75.1	70.1	73.8	69.5

Tabla 21. Evaliación del Factor R

OWD

Android 1	CommView	Android 2	CommView
185 ms	183 ms	199 ms	195 ms

Tabla 22. Evaliación del OWD

RTT

Android 1	CommView	Android 2	CommView
370 ms	367 ms	398 ms	396 ms

Tabla 20. Evaliación del RTT

PLR

Android 1	CommView	Android 2	CommView
0.09%	0.08%	0.08%	0.08%

Tabla 21. Evaliación del PLR

JITTER

Android 1	CommView	Android 2	CommView
20.504ms	20.452ms	21.604ms	21.520ms

Tabla 22. Evaliación del Jitter

Al analizar los resultados obtenidos se observa una pequeña diferencia entre los valores obtenidos mediante el analizador de protocolos CommView y la aplicación propuesta TestingVoip, estas diferencias son mínimas por lo cual se puede concluir que la aplicación TestingVoip es lo suficientemente fiable.

CAPÍTULO 8

Capítulo 8 CONCLUSIONES

En los últimos años, la denominada revolución móvil ha propiciado un despegue sin precedentes en el ámbito de las aplicaciones para terminales celulares. Estamos presenciando una etapa de popularización de la conectividad en redes inalámbricas. Por otro lado, los dispositivos móviles nos permiten comunicarnos casi desde cualquier lugar, y el rápido desarrollo de estas tecnologías ha desembocado en la incorporación de nuevas funcionalidades en los terminales, tales como: GPS, cámara fotográfica, acelerómetro, etc.

Esto aunado a la gran cantidad de aplicaciones en tiempo real, tales como VoIP, que se encuentran disponibles en las redes inalámbricas IP, con un costo de comunicación menor comparado con la PSTN. Sin embargo las redes IP de hoy en día solo ofrecen un servicio de "mejor esfuerzo", y no garantizan la calidad de servicio en este tipo de aplicaciones.

Una técnica para poder hacer frente a estos desperfectos es mediante el análisis y caracterización de los principales parámetros de QoS, mediante mediciones de red. La aplicación que se desrrollo en esta tesis tiene la finalidad de realizar llamadas de voz reales y evaluar los principales parámetros que determinan la calidad de servicio.

La aplicación propuesta se desarrollo bajo el sistema operativo Android, el protocolo SIP y permite realizar llamadas reales de voz mediante los esquemas de codificación G.711 u-law y G.711 a-law.

REFERENCIAS

Referencias

A. Sfairopoulou, B. B. (17 de Febrero de 2011). A Comparative Survey of Adaptive Codec Solutions for VoIP overMultirateWLANs: A Capacity versus Quality Performance Trade-Off. *EURASIP Journal on Wireless Communications and Networking*, págs. 5-11.

AndroidYA. (14 de 11 de 2018). *AndroidYA*. Obtenido de http://www.tutorialesprogramacionya.com/javaya/androidya/androidstudioya/index.php?inicio=0

Developers. (10 de Noviembre de 2017). *Developers*. Obtenido de https://developer.android.com/

Fundamentos de telefonia. (s.f.). Obtenido de http://elastixtech.com/fundamentos-de-telefonia/voip-telefonia-ip/

Herramientas Web para la enseñanza de protocolos de comunicación . (07 de Julio de 2017). Obtenido de http://neo.lcc.uma.es/evirtual/cdd/tutorial/Indice.html

Homero Toral-Cruz, A. D.-P. (2017). *"Advances and Challenges in Convergent Communication Networks", .* Wireless Personal Communications, Springer.

http://circe.univ-fcomte.fr/Docs/Java/JMF-Guide/RTPSending.html. (25 de 01 de 2019). Obtenido de http://circe.univ-fcomte.fr/Docs/Java/JMF-Guide/RTPSending.html

http://sipdroid.com/. (26 de Enero de 2017). *Sipdroid*. Obtenido de http://sipdroid.com/

International Telecommunications. (23 de Octubre de 2017). Obtenido de https://www.itu.int/ITU-T/studygroups/com12/emodelv1/tut.htm

ITU-T G.107. (2009). *The E-Model, A Computational Model for Use in Transmission Planning.* Geneva, Switzerland: Telecommunication Standardization Sector.

ITU-T P. 800. (1996). *Methods for Subjective Determination of Transmission Quality.* Geneva, Switzerland: Telecommunication Standardization Sector.

Joskowicz, J. (2013). *VOZ, VIDEO Y TELEFONIA SOBRE IP.* Montevideo, Uruguay: Instituto de Ingeniería Eléctrica, Facultad de Ingeniería.

Lukas Orcik, M. V.-C.-W. (2016). "Prediction of Speech Quality Based on Resilient Backpropagation Articial Neural Network". Wireless Personal Communications, Springer.

Mi granito de Java. (12 de 05 de 2018). Obtenido de http://migranitodejava.blogspot.com/2011/06/programacion-orientada-objetos-con-java.html

SGOLIVER.NET. (16 de 03 de 2018). Obtenido de http://www.sgoliver.net/blog/desarrollando-una-aplicacion-android-sencilla-android-studio/

STACKOVERFLOW. (15 de 12 de 2018). Obtenido de https://stackoverflow.com/questions/30948346/voip-rtp-streaming-from-to-server-in-java-to-from-android

Stylianos Karapantazis, F.-N. P. (29 de Marzo de 2009). VoIP: A comprehensive survey on a promising technology. *Computer Networks*.

Toral-Cruz, H., Pathan, A.-S., & Ramírez Pacheco, J. (2019). Accurate Modeling of VoIP Traffic in Modern Communication Networks. En M. Niazi, *Modeling and Simulation of Complex Networks* (págs. 175-208). United Kingdom: Institution of Engineering & Technology (IET).

Toral-Cruz, H., Ramirez-Pacheco, J., Velarde-Alvarado, P., & Pathan, A.-S. (2013). VoIP in Next Generation Converged Networks. En A.-S. Pathan, M. Monowar, & Z. Fadlullah, *Building Next-Generation Converged Networks: Theory and Practice* (págs. 337-360). USA: CRC Press, Taylor & Francis Group.

Torres, G. d. (2013). *DESARROLLO DE UN GENERADOR DE TRÁFICO.* Junio.

UIT-T G.711. (1993). *MODULACIÓN POR IMPULSOS CODIFICADOS (MIC) DE FRECUENCIAS VOCALES.* Geneva, Switzerland: Telecommunication Standardization Sector.

Xiuzhong Chen, C. W. (2003). Survey on QoS Management of VoIP. *Institute of Computing Technology, Chinese Academy of Sciences*, págs. 1-9.