TELÉFONO SOFTWARE IP BASADO EN SIP E IMPLEMENTACIÓN DE PESQ

MARCO AGUILAR JUNCA

PAOLA ANDREA RIAÑO CASTELLANOS

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA ELECTRÓNICA

BOGOTÁ D.C.

MAYO DE 2005

TELÉFONO SOFTWARE IP BASADO EN SIP E IMPLEMENTACIÓN DE PESQ

MARCO AGUILAR JUNCA

PAOLA ANDREA RIAÑO CASTELLANOS

DIRECTOR

Ing. HENRY FERNÁNDEZ

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA ELECTRÓNICA

BOGOTÁ D.C.

MAYO DE 2005

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA ELECTRÓNICA

RECTOR MAGNIFICO: R.P. GERARDO REMOLINA S.J.

DECANO ACADÉMICO: Ing. FRANCISCO JAVIER REBOLLEDO MUÑÓZ

DECANO DEL MEDIO UNIVERSITARIO: R.P. ANTONIO JOSÉ SARMIENTO NOVA S.J.

DIRECTOR DE CARRERA: Ing. JUAN CARLOS GIRALDO CARVAJAL

DIRECTOR DEL PROYECTO: Ing. HENRY FERNÁNDEZ

ARTICULO 23 DE LA RESOLUCIÓN No. 13 DE JUNIO DE 1946

"La universidad no se hace responsable de los conceptos emitidos por sus alumnos en sus proyectos de grado.

Sólo velará porque no se publique nada contrario al dogma y la moral católica y porque los trabajos no contengan ataques o polémicas puramente personales. Antes bien, que se vea en ellos el anhelo de buscar la verdad y la justicia".

AGRADECIMIENTOS

En primer lugar debemos dar las gracias especialmente a nuestros directores de trabajo de grado, Nelson Martínez y Henry Fernández. Su trabajo, apoyo y valiosa orientación han hecho posible la finalización de este trabajo.

También quisiéramos expresar nuestra sincera gratitud a Javier Villegas y a Edson Bárcenas por la confianza y el apoyo demostrados, así como al resto de miembros del Departamento de Electrónica.

Por último, queremos dar las gracias a nuestras respectivas familias, por su apoyo, comprensión y cariño, que nos han servido para no rendirnos en el tiempo que hemos estados dedicados a la realización de este Trabajo de Grado.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

GLOSARIO

- ACR Evaluación por categorías absolutas (absolute category rating)
- IRS Sistema intermedio de referencia (intermediate reference system)
- MOS Nota media de opinión (mean opinion score)
- PESQ Evaluación de la calidad vocal por percepción (perceptual evaluation of speech quality)
- PSQM Medida de la calidad vocal por percepción (perceptual speech quality measure)
- Loudness: Sonoridad, es un término subjetivo que describe la fuerza de la percepción de un sonido por el oído.
- Pitch: Es un atributo perceptivo del sonido que puede ser descrito como una sensación de la altura relativa de un sonido, su correlator físico es la frecuencia fundamental.
- Psicoacústica: Es el estudio de la percepción humana subjetiva de los sonidos. Estudio de la psicología de la percepción acústica.
- Escala Bark: Es una escala psico-acústica, varía de 1 a 24 y corresponde a las primeras 24 bandas críticas de escucha.
- Umbral absoluto de audición: Es la intensidad mínima de un tono puro que el oído puede oír en un ambiente sin ruido
- Cliente: Un cliente es cualquier elemento de una red, que envía peticiones SIP y recibe respuestas. Un cliente puede ser o no interactuar directamente con un usuario humano. Los UAC y proxies son clientes.
- Diálogo: Un diálogo es una relación SIP punto a punto, entre dos UAS que persiste por un tiempo. Un diálogo se establece por mensajes SIP, como

respuestas 2xx a una petición INVITACIÓN. Un diálogo es identificado por un identificador de llamada, una etiqueta local y una etiqueta remota. Un diálogo es formalmente conocido como una etapa de la llamada en el RFC2543.

- Encabezado: Un encabezado es un componente del mensaje SIP que transporta información del mensaje. Éste está estructurado como una secuencia de campos de encabezado.

- Campo de encabezado: Un campo de encabezado, es un componente del encabezado del mensaje SIP. Un campo de encabezado puede aparecer como uno o valores de campo de encabezado. Muchos valores de campo de encabezado dados en un campo de encabezado son separados por comas. Algunos campos de encabezado sólo pueden tener un valor, como resultado, siempre aparece como una fila de encabezado sencilla.

- Anillo o Loop: Es una petición que llega a un Proxy, devuelta, y más tarde regresa a mismo Proxy. Cuando llega por segunda vez, la petición URI es idéntica a la primera y los otros campos de encabezados que afectan la operación Proxy no son cambiados, por eso el Proxy debe generar la misma respuesta que dio para la primera petición. Peticiones en anillo son errores, y los procedimientos para detectarlos y manejarlo son descritos por el protocolo

- Mensaje: Información enviada entre los elementos SIP como parte del protocolo. Los mensajes SIP son tanto las peticiones como sus respuestas.

- Proxy fuera del límite: Es un Proxy que recibe peticiones de un cliente, aun cuando el servidor no puede resolver la comunicación con una petición URI. Típicamente un UA es configurado de forma manual con un Proxy fuera de límite, o puede aprender acerca de protocolos de auto-configuración.

INTRODUCCIÓN

El mundo de hoy en día, no sería lo mismo sin los sistemas de comunicación; tanto los sistemas de datos como los de voz, son una necesidad del mundo actual y por eso estos sistemas tienden a converger y de allí surge VoIP. VoIP es una categoría de "hardware y software" que permite actualmente, usar Internet como el medio de transmisión para llamadas telefónicas. La comunicación se logra enviando voz en paquetes de datos, usando el protocolo de Internet (IP), en lugar de la red conmutada de circuitos de voz PSTN (Public Switched Telephone Network). [11] La telefonía IP, provee llamadas muy económicas o gratis, en algunos casos a todo el mundo si se tiene un canal dedicado o pagando el costo de una llamada local. En las grandes empresas, se destinan unos recursos para pagar canales dedicados para el transporte de información a través de Internet, entonces ¿por qué no aprovechar estos mismos canales para enviar la voz y así ahorrarse el costo extra de la telefonía convencional a los diferentes operadores?

El protocolo de voz sobre IP (VoIP)[1], por sus siglas en inglés, hace referencia a la convergencia de la red de conmutada de circuitos de voz en la red de datos, para reducir costos y para proveer aplicaciones de telefonía a través de la interconexión de sistemas abiertos OSI[2]. [1][2] La telefonía sobre IP, es una de las tecnologías más importantes y de mayor crecimiento que provee una alternativa viable técnica y económica para las redes de comunicación actuales.

Este trabajo está dividido en dos secciones, la primera es un soft-phone[3] IP prototipo, que permite comunicación de voz entre dos usuarios de una misma red local a través de sus computadores personales. Aunque estas aplicaciones existen actualmente en el mercado, éste tiene cualidades como poder utilizarse tan sólo

[1] VoIP: Voice Over Internet Protocol.
[2] OSI: Open System Interconnection.
[3] Soft-phone: Teléfono realizado en software.

dentro de una red local, sin necesidad de tener conexión a Internet y la señalización será SIP[4]. Estas cualidades permiten ventajas como aprovechar la infraestructura de una red local de datos ya implementada, eliminando la necesidad de tener plantas telefónicas con extensiones, cableado, teléfonos y en general la administración y el mantenimiento de una red dedicada para voz, separada de la de datos.

Los siguientes son ejemplos de servicios provistos por una red de VoIP de acuerdo a los requerimientos del mercado: Teléfono a teléfono, PC a teléfono, teléfono a PC, PC a PC, voz a e-mail, número gratis (1-800), aplicaciones de centros de llamadas, red privada virtual (VPN), mensajes unificados, conectividad inalámbrica, y aplicaciones de red inteligente (IN) usando SS7.

La segunda parte de este trabajo realiza una evaluación perceptiva de la calidad del habla, sobre el canal de voz, implementado previamente. La evaluación perceptiva de la calidad del habla surge de la necesidad de saber en qué grado nosotros, los seres humanos, entendemos lo que escuchamos a través de un canal de comunicación de audio y así evaluar el comportamiento de la red de extremo a extremo. Para lograr esto, se han usado métodos subjetivos y objetivos.

Los métodos subjetivos permiten determinar empíricamente la calidad de archivos de audio, con la medida del codec o a través del uso de pruebas auditivas con personas; es decir, un número grande de personas que actúan como sujetos experimentales escuchan muestras de audio y califican según la escala MOS [5]de uno a cinco (malo a excelente). Las desventajas de este sistema son los elevados costos y la cantidad de tiempo que se requiere para obtener los datos, por lo tanto no puede ser aplicado dentro de un parámetro o ambiente práctico de la vida diaria.

[4] SIP: Session Initiation Protocol.
[5] MOS: Mean Opinion Score.

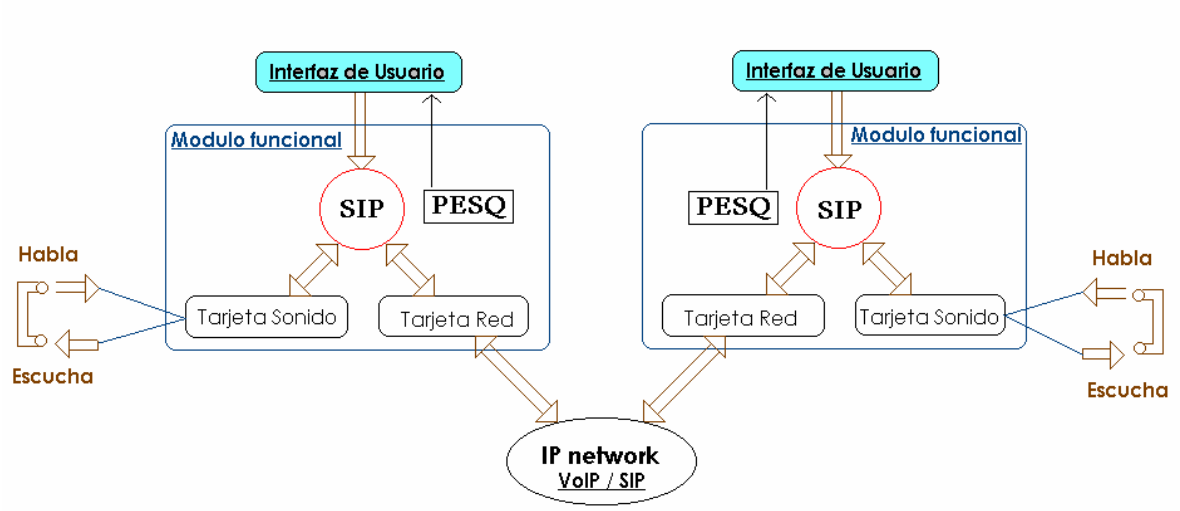

FIGURA 1 Diagrama en bloques de Sistema Software de Telefonía IP basado en SIP.

Los métodos objetivos tratan de recrear las pruebas subjetivas en un computador y reflejarlas con precisión a la escala MOS, sin el costo de hacerlas de forma subjetiva. Éste método está basado en la percepción, simulando un oído humano, a través de la psico-acústica[6], método que estudia la relación entre las propiedades físicas del sonido y la interpretación que hace de ellas el cerebro.

[6] Psicoacústica: Rama de la psicofísica que estudia la relación existente entre el estímulo de carácter físico y la respuesta de carácter psicológico que el mismo provoca.

1 MARCO TEÓRICO

1.1 VOZ SOBRE IP (VoIP)

Voz sobre IP, se define como el transporte de señales de voz de un transmisor a un receptor a través de una red que utiliza el protocolo de Internet para transmitir información [24]. Usando los protocolos de VoIP, las comunicaciones de voz se pueden implementar en cualquier red IP ya sea Internet, Intranets o redes de área local (LAN). En una red habilitada con VoIP, se digitaliza la señal de voz, se comprime, se convierte en paquetes IP y después se transmiten sobre la red. Los protocolos de señalización de VoIP se utilizan para establecer, terminar llamadas y llevar información requerida para localizar a usuarios y para negociar capacidades.

1.1.1 COMPONENTES DE UN SISTEMA DE VOIP. [24]

Una llamada de VoIP tiene un procedimiento básico cuando se realiza. Cada uno de los pasos de este procedimiento es ejecutado por uno de los componentes del sistema, en un sentido, se ejecutan unas funciones de transmisión y en el otro unas de recepción. A continuación se describen estos componentes y sus funciones.

- *CAPTURA Y REGENERACIÓN [24]*

Para enviar la información del habla a través de una red de datos, las señales deben ser convertidas previamente a un formato digital. La señal es captada por un micrófono y digitalizada por medio de la tarjeta de sonido del computador, este proceso de digitalización de la señal comprende el muestreo (en tiempo) y la cuantización (en amplitud). Esta señal se envía, y al otro lado se hace el proceso inverso pasando la señal digital a un formato análogo de audio, luego se

reproduce a través de los parlantes o audífonos. Los procesos mencionados anteriormente se conocen como captura y regeneración respectivamente y son ejecutados por conversores A/D y D/A, definiendo estos estados:

- Muestreo y Cuantización (A/D)
- Reconstrucción (D/A)
- Mezcla de señales de audio.

El proceso de muestreo se define en el dominio del tiempo como el proceso de convertir una señal análoga, en una secuencia correspondiente de muestras, que se espacian uniformemente en el tiempo. Sin duda, para que un procedimiento de este tipo tenga utilidad práctica, resulta necesario que elijamos adecuadamente la frecuencia de muestreo, de manera que la secuencia de muestras defina en forma única la señal análoga original. Ésta es la esencia del teorema de muestreo, el cuál dice lo siguiente:

- Una señal limitada en banda de energía finita, que no tiene componentes de frecuencia mayores a W hertz, se describe por completo especificando los valores de la señal en instantes separados por 1/(2W) segundos.
- Una señal limitada en banda de energía finita, que no tiene componentes de frecuencias mayores que W Hertz, puede recuperarse completamente a partir del conocimiento de sus muestras tomadas a una velocidad de 2W.

La frecuencia de muestreo de 2W muestras por segundo, para un ancho de banda de señal W Hertz, se denomina la frecuencia de Nyquist; su recíproco 1/(2W) (medido en segundos) se denomina el Intervalo de Nyquist. [25]. Las señales humanas de voz, pueden contener frecuencias hasta de 12KHz, sin embargo, en el sistema telefónico, sólo se transmite por debajo de 4000Hz obteniendo una comunicación de calidad. Usando esta información junto con el teorema de Nyquist, se concluye que la tasa de muestreo adecuada para digitalizar señales de voz es de 8000Hz.

Una señal continua, tal como la voz, tiene un intervalo continuo de amplitudes y, por tanto, sus muestras tienen un intervalo continuo de amplitud. En otras palabras, dentro del intervalo de amplitud finito de la señal, encontramos un número infinito de niveles de amplitud. De hecho, no es necesario transmitir las amplitudes exactas de las muestras. Cualquier sentido humano (el oído o el ojo), como un receptor final, puede detectar únicamente diferencias de intensidad finitas, lo cual quiere decir que la señal continua original puede aproximarse mediante una señal construida a partir de amplitudes discretas elegidas de un conjunto disponible sobre una base de error mínimo. [25]

Es conveniente saber que una tasa de muestreo grande y una pequeña cuantización, conllevan a una mejor representación de la señal original. Sin embargo, esto significa que entre más información digital tenga, más debe ser transmitido, por tanto se debe determinar que tanta información es necesaria para mantener una conversación telefónica con calidad aceptable.

Para cubrir el rango de amplitudes que las señales de voz pueden producir, son necesarios por lo menos doce bits cuando es usado un esquema uniforme de cuantización, sin embargo un esquema uniforme que se reduce a ocho bits es usualmente suficiente para tener conversaciones telefónicas de calidad. [24].

En cuanto a la cuantización logarítmica, existen dos tipos que vale la pena nombrar. En Europa es usada la ley A y en estados Unidos y Japón, la ley μ es el estándar de transmisión sobre redes, éste método reduce de trece bits uniformes a ocho bits en valores logarítmicos.

Finalmente podemos calcular que el ancho de banda necesario para una conversación telefónica, es de por lo menos 64Kbps, ya que la tasa de muestreo debe ser de 8000Hz y la cuantización de 8 bits. [24]

- *REQUISITOS DE COMUNICACIÓN [24]*

En una comunicación de VoIP, resultan algunos inconvenientes en la transmisión de los paquetes de voz debido a diferencias en los retardos de llegada al receptor, esto se conoce como Jitter. La calidad de la voz en este tipo de conversaciones no se puede garantizar ya que es muy complicado evitar que en el envío de estos paquetes, algunos lleguen con errores e inclusive que todos lleguen. De acuerdo a esto se han establecido ciertos requisitos en parámetros para asegurar de cierta forma la calidad de VoIP.

- Error de tolerancia: En contraste a una comunicación de datos, en donde el error más pequeño puede causar resultados devastadores, la comunicación de voz es más tolerante a la presencia de errores. Un error ocasional no degradará seriamente la conversación siempre y cuando el error no afecte una porción relativamente larga de la señal.

- Requisitos de retardo: cuando se envía información de datos no es muy relevante el tiempo que tarda un paquete en llegar a su destino, mientras que en una comunicación de voz estos retardos sí son muy importantes. Por esto se ha determinado que los retardos deben ser menores a 200ms para tener una conversación clara.

- Tolerancia de Jitter: Si cada paquete contiene una parte de voz, ésta debe ser reproducida una vez llegue al receptor. A veces se presenta que unos paquetes llegan de forma continua pero otras veces a hay un retardo adicional. Si la voz se reprodujera así, sería muy molesto para los participantes de la conversación, por eso este Jitter debe tratar de eliminarse.

1.1.2 PARÁMETROS DE RED CON VOIP

- *LONGITUD DE PAQUETES*

En una transmisión de voz sobre IP se pueden dañar o perder paquetes; para reducir en cierta forma la pérdida de información, los paquetes deben tener una

porción muy pequeña de la señal de voz. De esta forma, si un paquete se pierde, tan solo una pequeña porción es perdida y es muy improbable que distorsione la conversación. [24].

- *BUFFER*

Anteriormente, se mencionaron los efectos negativos del Jitter; una técnica para reducirlo durante la transmisión de la voz, es introducir una cantidad de retardo o buffer. En lugar de ir reproduciendo la señal de voz de cada paquete que va llegando, se introduce un retardo a cada paquete obteniendo así una alta probabilidad de que cuando termine de reproducirse un paquete el siguiente ya esté disponible y así hacer que la señal sea continua. [24].

- *RETARDO*

Un retardo muy largo es desastroso para una conversación. El retardo total puede ser clasificado en dos tipos, el retardo fijo que es el retardo total debido al buffer, la capacidad del enlace etc. y el retardo variable que es causado por filas de paquetes en los routers, congestiones en la red etc. [24].

- *SUPRESIÓN DE SILENCIO*

Cuando se está efectuando una conversación, es usual que tan sólo una persona esté hablando a la vez, cuando estamos usando un sistema de VoIP, tenemos oportunidad de “ahorrar” ancho de banda porque los paquetes que contienen silencio no es necesario enviarlos.

Antes de desechar paquetes, debemos determinar si contienen silencio o no, una forma de hacerlo es calcular la energía de la señal de voz en un paquete y los paquetes que no contengan la suficiente cantidad de energía pueden suponerse contenedores de silencio y así pueden ser desechados. [24].

La supresión de silencio no tiene mayores efectos en la conversación debido a que los paquetes de “silencio”, efectivamente no tienen sonido de la conversación

sino ruido ambiente; sin embargo para que la conversación no parezca que haya terminado, es introducido un sonido ambiente artificial en el lado del receptor.

- *JITTER*

El Jitter es una medida de la variación del tiempo que toman las comunicaciones para atravesar desde el remitente (aplicación) hasta el receptor. Se puede tender a pensar en el Jitter como la variación media estadística del tiempo de llegada entre paquetes o data gramas.

El Jitter puede crear problemas audibles de la calidad de la voz, si la variación es mayor a 20ms. Los síntomas de excesivo Jitter son muy similares a los síntomas de gran retardo, porque en ambos casos se desechan los paquetes si el retardo excede la mitad del tamaño del Jitter Buffer.

La topología de la red también puede afectar el Jitter porque por ejemplo, hay menos colisiones en una red jerárquica conmutada de datos que en una red plana.

- *PÉRDIDA DE PAQUETES*

La pérdida de paquetes se da cuando se envían paquetes por una red, pero no se reciben en el destino final debido a algún problema en la red. Para asegurar buena calidad de voz en una red de VoIP, la pérdida del paquete debe ser menor al 0.2% entre los puntos finales. Hay varios factores que hacen que exista pérdida de paquetes:

Los requisitos de la pérdida de paquetes son más exigentes para los tonos, (con excepción de DTMF) que para la voz, debido a que el oído es menos capaz de detectar la pérdida de paquetes durante discurso (pitch variable), que durante un tono (constante).

Los requisitos de la pérdida de paquetes son más exigentes para la pérdida continua de paquetes pequeños que para la pérdida de paquetes al azar en cierto

plazo, es decir, perder diez paquetes contiguos es peor que perder diez paquetes espaciados uniformemente en una hora.

La pérdida de paquetes puede ser más sensible con la pérdida de grandes cargas útiles de la voz que para las más pequeñas, porque se pierde más voz en una carga útil más grande.

- *DESORDEN DE LLEGADA*

El desorden de llegada de paquetes excesivo, es muy parecido a la pérdida de paquetes para la voz sobre IP. Si un paquete llega fuera en desorden, generalmente se desecha, pues no tiene ningún sentido reproducirlo en desorden. Específicamente, los paquetes se desechan

cuando llegan más tarde de lo que el Jitter Buffer puede mantenerlos. El desorden de llegada puede ocurrir cuando las redes envían paquetes individuales por diversas rutas. Acontecimientos previstos como carga balanceada, acontecimientos imprevistos como re-enrutar debido a congestiones u otras dificultades transitorias pueden causar desorden en la llegada de los paquetes. Los paquetes que atraviesan la red por diferentes rutas pueden llegar su destino en desorden.

- *TRANSCODIFICACIÓN*

La transcodificación es la conversión de una señal de voz análoga a digital o de digital a análoga (posiblemente con o sin compresión y descompresión). Si las llamadas son enrutadas usando múltiples codificadores de voz, las llamadas pueden experimentar múltiples transcodificaciones, lo que da lugar a una cierta degradación de la calidad de la voz.

- *ECO*

En una transmisión de VoIP existen muchas fuentes de eco por ejemplo se da eco por acústica, eco por impedancia etc. Existen varias causas por la que se

presenta el eco, por ejemplo cuando una llamada de VoIP sale de la LAN a través de una troncal análoga a la PSTN, o cuando hay desacople de impedancias entre sistemas de cuatro y dos cables o cuando se hace la conversión entre bus TDM (multiplexación por división de tiempo) y LAN, o por el desacople entre el receptor y su adaptador.

Los desacoples de impedancia causan una transferencia de energía ineficaz y el desequilibrio de la energía debe ir a alguna parte lo que se ve reflejado en forma de eco. El locutor oye generalmente eco pero el receptor no. Los supresores de eco, tienen gran cantidad de memoria, para comparar la voz recibida con el patrón de voz y si los patrones concuerdan, el supresor cancela el eco. Sin embargo los supresores de eco no son perfectos, bajo algunas circunstancias y el eco persiste a pesar del supresor.

- *DUPLEX*

La red ideal para el transporte del tráfico de VoIP es aquella que es completamente LAN conmutada de extremo a extremo porque reduce o elimina perceptiblemente colisiones. Una red que comparte segmentos (basada en Hubs) puede dar lugar a una calidad de voz más baja, debido a excesivas colisiones.

- *SELECCIÓN DE CODEC*

Dependiendo de la disponibilidad del ancho de banda y calidad de voz requerida, puede ser un gran logro seleccionar un codec que produzca audio comprimido.

El codec G.711 produce audio sin comprimir a 64 kbps

El codec G.729 produce audio comprimido a 8 kbps

El codec G.723 produce audio comprimido aproximadamente a 6 kbps

La siguiente tabla proporciona la comparación de varias consideraciones de la calidad de la voz se asociadas a algunos codecs:

Estándar	Tipo de codificación	Tasa de Bits (Kbps)
G. 711	PCM	64
G. 729	CS-ACELP	8
G. 723.1	ACELP MP-MQL	6.3 5.3

Tabla 1 Comparación de los estándares de codificación del habla.

1.2 ARQUITECTURA MULTIMEDIA IETF

Dado que SIP se basa en la arquitectura multimedia del IETF se hace necesario mencionar su pila de protocolos y la ubicación de SIP en ésta. La arquitectura está basada en un conjunto de protocolos independientes e intercambiables y compatibles con sistemas basados en H.323.

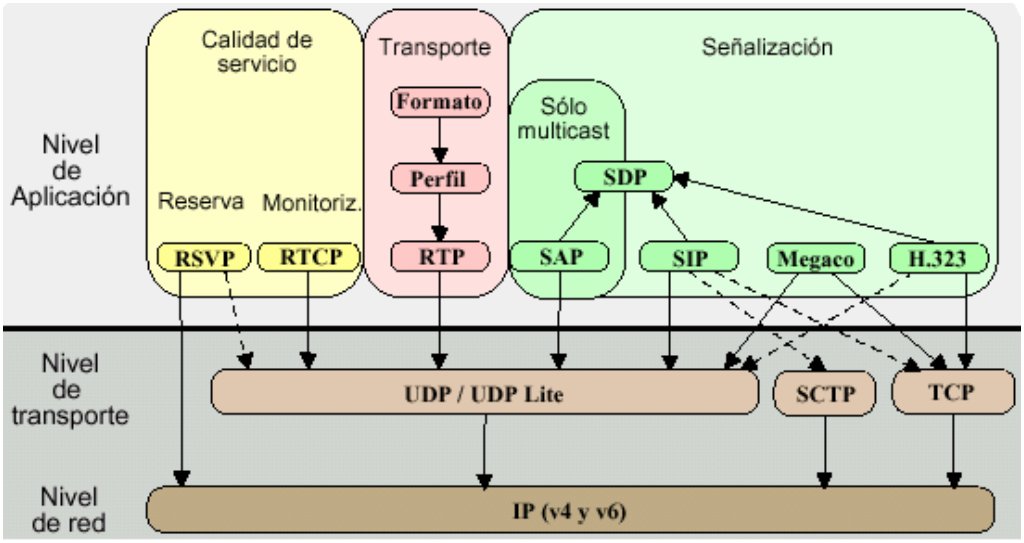

FIGURA 2 Arquitectura multimedia IETF.

1.2.1 NIVEL DE APLICACIÓN

- *CALIDAD DE SERVICIO*

Para que una solución de VoIP funcione bien, la red debe poder dar mayor prioridad a los paquetes de voz que a los paquetes ordinarios de datos. Existen

estrategias para dar prioridad a estos paquetes como clase de servicio (CoS), prioridad de puertos, servicios de prioritización y el uso de 802.1p/Q IEEE para fijar los bits de prioridad.

Clase de Servicio Vs. Calidad de Servicio. La clase del servicio (CoS) es un método solamente de clasificación, CoS no asegura calidad del servicio (QoS), pero es el método usado para mejorarla. La mayoría de las estrategias de CoS asignan un nivel de prioridad, generalmente 0-7 o 0-63, a un marco o a un paquete respectivamente. Los modelos comunes de CoS incluyen Tipo de Servicio (TOS) IP, DSCP (Differentiated Service o Differentiated Service Code Point, definido en RFC 2474 y otros) y el estándar 802.1p/Q de la IEEE. La calidad del servicio (QoS) implica dar tratamiento preferencial, reserva de la ancho de banda u otros métodos basados en cualidades del paquete, como la prioridad CoS y después se negocia la calidad de servicio, ejemplos de QoS son ATM (Asynchronous Transfer Mode), RSVP (Reservation Protocol (RFC 2205)) y MPLS (Multi-Protocol Label Switching (RFC 1117 y otros)). [5]

Differentiated Service. (DiffServ). Representa una solución para fijar la prioridad de diferentes tipos de tráfico. Este hace uso de los campos del encabezado IP versión 4 conocidos con el nombre de campos TOS (Types of Service), o los campos del encabezado IP versión 6 conocidos como TC (Traffic Class). Los campos Type of Service/Traffic Class los usa DiffServ al redefinirlos para marcar diferentes tipos de tráfico, los bits de estos campos se combinan para crear un numero determinado de clases de servicio las cuales representan varias categorías de aplicación, los cuales se quiere que se encuentren estandarizados entre todos los ISP y routers dentro de la red, por lo tanto todos los routers deben entender las categorías DiffServ QoS. DiffServ simplemente asegura que un tipo especifico de trafico, como por ejemplo trafico de voz será asignado de primero en la cola en la salida de un puerto de un router. En DiffServ se pueden usar los bits del campo TOS / TC para identificar el tipo de tratamiento que se le da a un trafico determinado, esto se conoce como PHB (Per Hop Behavior), estos se clasifican

según el DSCP, para crear clases de trafico diferentes. Un tipo de PHB se conoce como EF (Expedite Forwarding), usado generalmente para soportar servicios de VoIP, en el cual se garantiza baja perdida de paquetes, bajo retardo y baja variación del retardo, asegurando un ancho de banda dedicado entre los puntos finales. [17]

Uso de DSCP (TOS). El esquema de prioritización DSCP redefine el byte de tipo de servicio (TOS) en el encabezado IP combinando los primeros seis bits en 64 combinaciones posibles. El método original de TOS utiliza el byte TOS para asignar una clase de servicio según lo definido en RFC 795. Este byte IP se puede utilizar en la manera tradicional fijando los tres bits precedentes dando así ocho clases de servicio y los otros cuatro bits definen el retardo (normal o bajo), el rendimiento de procesamiento (alto o normal), la confiabilidad (alto o normal) y el costo (normal o bajo). [5]

RSVP (Resource Reservation Protocol). Habilita diferentes técnicas de reserva de recursos para una sesión o sesiones dadas antes de que esta(s) sea(n) establecida(s) y se intente intercambiar flujo de medios entre los participantes de dicha sesión. De las soluciones disponibles en el mercado RSVP es la solución más compleja pero es la que más se acerca a la emulación de circuitos dentro de una red IP. Consiste en señalización IP con el propósito de establecer un camino entre dos puntos finales, con el objetivo de brindarle QoS al flujo de datos, RSVP como tal no da QoS pero trata a los paquetes de una cierta clasificación de acuerdo a las reglas de dicha clasificación, de tal forma que el envió de punto a punto es un proceso predecible, RSVP fija un camino, el cual va a ser usado por los paquetes que fluyen entre los puntos finales. Las características principales de RSVP son:

- Es orientado a receptor, esto quiere decir que la reserva del camino propuesta es hecha por el receptor y no por el emisor, la razón principal para tener esta característica es el soporte de aplicaciones multicast.

- Soporta reservas heterogéneas, por cada sesión multicast, esto permite que los receptores en una sesión multicast realicen peticiones diferentes de QoS.
- Los receptores tienen la capacidad de modificar una QoS existente en el camino entre dos puntos finales.
- Soporta múltiples estilos de reserva, para diferentes necesidades de aplicaciones diferentes, como por ejemplo audio o video.

En RSVP la calidad de servicio se implementa para un flujo de datos en particular gracias a un mecanismo que en su conjunto se denomina control de tráfico. En este se incluye, el clasificador de paquetes, el control de admisión y políticas y el organizador de paquetes, los cuales son usados en conjunto para determinar cuando se debe reenviar un paquete. El clasificador de paquetes determina el tipo de QoS y la ruta para cada paquete, para cada interfaz de salida el organizador de paquetes brinda la QoS prometida. A través de estas funciones, el control de tráfico, puede implementar la QoS deseada. Durante el establecimiento de la reserva, se realiza una petición de QoS RSVP, la cual es transmitida a una entidad de decisión local, denominada control de admisión y políticas. El control de admisión determina si el nodo tiene suficientes recursos disponibles para suministrarle a la petición QoS, y el control de políticas determina si el usuario tiene permisos de tipo administrativo para hacer la reserva requerida. Si ambos controles son superados, los parámetros son establecidos en el clasificador de paquetes y en la interfaz de la capa de enlace para obtener la QoS deseada. Si alguno de los controles falla, el programa RSVP devuelve un error a la aplicación que hizo la petición. [17]

Uso de Puertos. Con el uso de puertos, se puede implementar un esquema del prioritización que asigna la prioridad de acuerdo a los números de los puertos UDP (User Datagram Protocol) que usan los paquetes de voz. Este esquema permite utilizar el equipo necesario para que la red pueda dar prioridad a todos paquetes de una gama portuaria. El UDP se utiliza para transportar voz por LAN porque,

contrario al TCP, es un protocolo no orientado a conexión. Debido a que la sensibilidad del oído humano para el retardo, es mejor que la p los paquetes más bien que retransmita la voz en un ambiente en tiempo real. Así pues, un protocolo sin conexión es preferible a un protocolo orientado a conexión. [5]

Uso de VLANs. Las redes locales virtuales (VLANs) proporcionan seguridad y crean pequeña difusión de dominios, a través de software creando subredes separadas virtualmente. La difusión es natural en los protocolos usados en la mayoría de las redes de datos por PC, servidores, switches y routers. Crear una VLAN separada para voz, reduce la cantidad de tráfico de difusión que el teléfono recibirá. La separación de VLANs da como resultado una eficaz utilización del ancho de banda y reduce la carga del procesador en los puntos finales, liberándolos de tener que analizar paquetes irrelevantes. Las VLANs son características de la capa 2 y se definen en los switches de datos, o pueden estar especificadas en una lista en los puertos del switch. La voz y los datos separados en VLANs es una opción que tiene sentido para la mayoría.

Estándar VLAN 802.1 Q. El estándar del instituto de ingenieros eléctricos y electrónicos (IEEE, Institute of Electric and Electronic Engineers) 802.1 Q asocia dispositivos físicos y puertos a cada VLAN, esto lo hace al añadir una etiqueta VLAN de dos bytes, a cada trama Ethernet, esta etiqueta VLAN identifica el grupo de trabajo virtual al que pertenece la trama y además incluye información de prioridad y QoS. Este identificador VLAN es un desarrollo del identificador propuesto por el protocolo 802.10 de IEEE ratificado en 1992, el cual define un encabezado de 4 bytes, su evolución conllevo a la creación de una etiqueta VLAN de 12 bits, que contiene información de membresía a una VLAN, prioridad, y QoS. El diagrama de una etiqueta VLAN 802.1Q dentro de una trama Ethernet se muestra en la Figura 3.

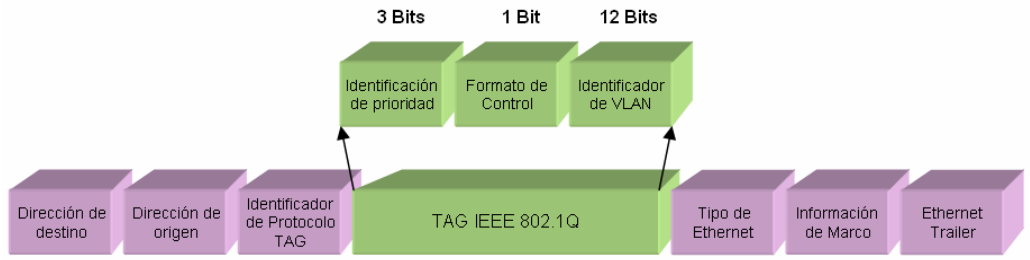

FIGURA 3 Trama Ethernet modificada por el estándar 802.1Q.

El estándar 802.1 Q además brinda la posibilidad de propagar la información de la configuración de VLAN a través de los switches en la red, haciendo uso del protocolo de registro de atributos genéricos (GARP, generic attribute registration protocol) y el protocolo de registro de VLAN GARP (GVRP, GARP VLAN Registration Protocol). El mismo estándar define el protocolo de registro de multicast GARP (GMRP, GARP multicast registration protocol) que brinda información del tráfico multicast y mejora la eficiencia y la limitación de este tipo de trafico en la red.

Estándar de prioritización de tráfico 802.1p. El estándar de IEEE 802.1p es un método de fácil implementación y efectivo para asignar prioridades de trafico en una red de área local, el estándar 802.1p recomienda el uso de las etiquetas VLAN de las tramas Ethernet definidas por el protocolo 802.1Q, como se pudo ver en la figura (VLAN TAG), existen 3 bits de la etiqueta que identifica el nivel de prioridad de cada trama, debido a que estos tres bits pueden definir 8 valores, 802.1p define 8 clases de trafico, que se ilustran en la tabla.

Clase de tráfico	Valor de la etiqueta (binario)	Tipo de tráfico
7	111	Control
6	110	Voz
5	101	Multimedia
4	100	Aplicaciones con carga controlada
3	011	Esfuerzo excelente
2	110	Repuestos
1	001	Fondo
0	000	Por defecto

Tabla 2 Clases de Tráfico 802.1 p

Una vez asignada la clase de tráfico a cada trama, las políticas de prioridad son configuradas y administradas de forma centralizada a través de software de administración. Después de que la trama ha sido etiquetada se transmite a la red, los switches compatibles con el protocolo 802.1 p reconocerán el valor de la clase de trafico y reenviaran la trama de acuerdo a su prioridad, para hacer esto los switches deben tener múltiples colas, de tal forma que el trafico con mas alta prioridad asignada será enviado a la cola de mas alta prioridad, y el trafico con la mas baja prioridad asignada será enviado a la cola de mas baja prioridad. [17]

- *SEÑALIZACIÓN*

Dentro de las funciones de señalización encontramos el establecimiento, modificación, finalización de llamadas, registro y localización de participantes, gestión del conjunto de participantes y de los componentes del sistema.

El Protocolo de inicio de sesión SIP, por sus siglas en inglés, es un protocolo de señalización de control simple de la capa de aplicación basado en texto creado para controlar sesiones multimedia entre dos o más participantes [14]. Dentro de sus funciones principales de SIP, se diferencian dos tipos la señalización que se encarga del establecimiento, modificación y finalización de sesiones de voz, y el intercambio de tráfico de voz mediante protocolos adicionales como RTP y RTCP. SIP está definido en el RFC 3261 de la IETF (Internet Engineering Task Force) para establecer y administrar cualquier tipo de comunicación sobre Internet de persona a persona en tiempo real, incluyendo llamadas telefónicas por Internet, distribución multimedia y conferencias multimedia.

SIP está definido como un protocolo cliente servidor, en donde el cliente puede ser un programa de aplicación que envía peticiones SIP y el servidor es una entidad que responde a esas peticiones.

Los servidores SIP se diferencian en dos clases:

- Servidores Proxy, los cuales ejecutan la señalización de llamada entre los clientes que están unidos a él.
- Servidores de re-direccionamiento, que son aquellos que determinan la dirección actual del cliente que llama iniciar señalización con el cliente llamado.

SIP soporta cinco facetas para establecer y terminar comunicaciones multimedia:

- Localización del usuario: Determina el punto final con el que se efectuará la comunicación.
- Disponibilidad del usuario: Determina la buena disposición de la llamada para establecer las comunicaciones.
- Capacidades del usuario: Determina el medio y los parámetros del medio a ser usado.
- Establecimiento de sesión: Timbrazo, establece los parámetros de la sesión de la llamada.
- Gestión de Sesión: Incluye la transferencia y finalización de la sesión al igual que la modificación de los parámetros de ésta.

SIP no es un sistema de comunicación verticalmente integrado pero puede ser usado con otros protocolos para construir una arquitectura de multimedia completa. Típicamente este tipo de arquitecturas incluyen protocolos como RTP (definido en el RFC 3550) para transporte de información en tiempo real que provee realimentación para la calidad de servicio, el RTSP (definido en el RFC 2326) para el control sobre la entrega de datos con propiedades de tiempo real como audio ó video y el protocolo de descripción de sesión SDP (definido en RFC 2237) usado por SIP como cuerpo de los mensajes. Describe sesiones multimedia con el propósito de anuncio e invitación de sesión y otras formas multimedia de iniciación de sesión. Sin embargo el funcionamiento y operación básica de SIP no depende de ninguno de estos protocolos [14].

SIP es un protocolo estructurado por capas, lo que significa que su comportamiento es descrito en términos de un grupo de estados de procesamiento independientes, con sólo una pequeña interacción entre cada capa. El comportamiento del protocolo es descrito en capas con el propósito de presentación, permitiendo la descripción de funciones comunes a través de elementos de una sección sencilla.

No todos los elementos especificados por el protocolo contienen cada capa, además los elementos especificados por SIP son lógicos y no físicos.

La capa más baja de SIP es la sintaxis y la codificación que específicamente es una gramática Backus-Naur Form BNF, por sus siglas en inglés (definido en el RFC 2234).

La segunda capa es la de transporte, que define cómo un cliente envía peticiones y recibe respuestas y cómo el servidor envía peticiones y recibe respuestas a través de la red. Todos los elementos de SIP contienen la capa de transporte.

La tercera capa es la de transacción, que son los componentes fundamentales de SIP. Una transacción es una petición enviada por una transacción de cliente una transacción de servidor, junto con todas las respuestas enviadas a esa petición de la transacción de servidor de vuelta al cliente.

La siguiente capa es llamada usuario de transacción (TU). Cada uno de los componentes de SIP, excepto el Proxy sin estados, son usuarios de transacción. Cuando un TU quiere enviar una petición, éste crea una transacción de cliente y la pasa junto con la dirección IP de destino, el puerto y lo que quiere transportar.

La arquitectura del modelo SIP, comprende características como su integración a la infraestructura WEB, su estructura cliente servidor, los mensajes de petición, respuesta y la reutilización de conceptos de otros servicios como WEB, correo y DNS.

Los Mensajes SIP. SIP es un protocolo basado en texto y utiliza un grupo de caracteres UTF-8. Un mensaje SIP es una petición desde un cliente a un servidor o una respuesta desde el servidor a un cliente.

Los dos mensajes, tanto la petición como la respuesta, usan el formato básico del RFC2882. Sin embargo la sintaxis difiere en el conjunto de caracteres y en la sintaxis específicamente ya que SIP permite encabezados que no permite el RFC2882.

Ambos paquetes de mensajes consisten en una línea de comienzo, uno o más encabezados, líneas vacías, que indican el final de los encabezados y un mensaje de cuerpo que es opcional.

La línea del comienzo en cada encabezado del mensaje y la línea vacía al final, aunque el cuerpo del mensaje no esté, deben terminar en una secuencia CRLF (carriage-return line-feed sequence).

Excepto por las diferencias en el conjunto de caracteres, muchos de los mensajes de SIP y la sintaxis de los encabezados son idénticos a HTTP/1.1. (RFC2616), sin embargo SIP no es una extensión de HTTP.

a) Peticiones

En SIP, las peticiones se realizan mediante un mensaje que se compone de tres elementos, la dirección de la entidad a la que es enviado, la versión de SIP y un método. SIP usa seis métodos principales que se muestran en la tabla [3], para conformar 6 tipos de requerimientos.

Estos parámetros van separados por un espacio sencillo (SP) y la línea de petición finaliza con una CRLF, como se ilustra a continuación:

Línea de Petición = Método SP Petición-URI SP Versión-SIP CRLF.

Métodos SIP	Descripción
INVITE	Primer mensaje enviado por el cliente que hace la llamada. Contiene información en el encabezado SIP, el cual identifica el cliente que llama, ID de llamada, cliente llamado, numero de secuencia, entre otros. También puede enviarse durante una llamada para modificar el estado de operación de la llamada (Ej. poner cliente en llamada en espera. Mensaje contiene una descripción SDP del cliente llamado, con parámetros tales como tipo de media, y direcciones de transporte.
ACK	El cliente llamado responde son un ACK solo a métodos INVITE que han sido aceptados satisfactoriamente con el código 200. El cuerpo del mensaje ACK contiene una descripción SDP de las capacidades de tipo de media del cliente llamado.
OPTIONS	Este mensaje pregunta las capacidades de un cliente. Determina que tipo de media soporta un usuario remoto antes de establecer una llamada.
BYE	BYE El cliente envía este mensaje al servidor para finalizar la llamada. El cliente que envía este mensaje termina el flujo de media y considera que la llamada ha sido terminada independientemente de la respuesta del cliente remoto.
CANCEL	CANCEL Método que cancela un requerimiento que este en proceso, pero no tiene efectos en la llamada establecida si no se están procesando requerimientos. Este método debe especificar explícitamente la llamada por el ID de llamada, la secuencia de llamada, y los campos TO y FROM del encabezado SIP.
REGISTER	REGISTER El cliente usa el método REGISTER para registrarse en la lista de direcciones del campo To en el encabezado con un servidor SIP.

Tabla 3 Peticiones SIP. [17].

Métodos adicionales pueden estar documentados en estándares extensiones de SIP.

Petición - URI: es una URI (Uniform Resource Indicator) que indica el usuario o el servicio para el que la petición se ha direccionado. Éste no debe contener espacios o caracteres de control, ni debe estar entre “<>“.

Los elementos de SIP deben soportar peticiones URI con esquemas diferentes a “SIP”, por ejemplo “TEL”. Los elementos de SIP deben poder traducir URIs que no sean SIP usando cualquier mecanismo del que dispongan, resultando en un URI SIP o cualquier otro esquema.

Versión - SIP: Los dos mensajes, petición y respuesta, incluyen la versión de SIP en uso.

b) Respuestas

Las respuestas SIP se diferencian de las peticiones, en que las primeras tienen una línea de estado en su línea de comienzo. Una línea de estado consiste en la versión del protocolo mostrado por un código numérico y es asociado textualmente con cada elemento separado por un espacio sencillo.

El código de estado es un número entero de tres dígitos resultante del código que indica el resultado de una prueba para entender y satisfacer una petición. El código de estado se hizo para la máquina por tanto se creó “La frase razón” que pretende dar una descripción corta textual del código de estado para el ser humano, aunque no es necesario que el cliente examine esta “Frase de razón”.

El primer dígito del código de estado define la clase de respuesta. Los siguientes dos dígitos no tienen ningún papel de categorización, por esta razón cualquier respuesta del código de estado entre 100 y 199 se refiere a una “repuesta 1xx”, entre 200 y 299 “respuesta 2xx” y así sucesivamente. SIP 2.0 permite seis valores para el primer dígito.

1xx: Provisional -- Petición recibida, continúa el proceso de la petición.

2xx: Éxito -- La acción fue exitosamente recibida, entendida y aceptada.

3xx: Redirección -- Esta acción necesita ser ejecutada para completar la petición.

4xx: Error de Cliente -- La petición contiene mal la sintaxis o no puede ser cumplida por este servidor.

5xx: Error de Servidor -- El servidor falla en la ejecución de una petición aparentemente válida.

6xx: Falla Global -- La petición no puede ser ejecutada en ningún servidor.

Categoría	Código de Estado	Frase Razón	Explicación
Informational Procediendo con la ejecución del requerimiento.	100	Trying	Procediendo con la llamada.
	180	Ringing	Alertando sobre la llamada
	181	Call is being forwarded	Enviado por un servidor Proxy, agregando la ubicación del destino de envió de llamada
	182	Queued for service	Se utiliza en el caso de que se quiera poner la llamada en una cola de llamadas
Success Requerimiento entendido, y acción desarrollada	200	OK	Petición ejecutada satisfactoriamente.
Redirection. La llamada necesita más procesamiento antes de que se pueda determinar si puede ser completada.	300	Multiple Choices	La dirección en la petición al ser resuelta da como resultado más de una opción, los cuales son enviados de vuelta al que llama para que escoja una opción y re-direccione la llamada.
	301	Moved Permanently	El usuario llamado ha cambiado de dirección, y el usuario que llama debe intentar llamarlo a una nueva dirección que le es devuelta en el encabezado de la respuesta
	302	Moved Temporarily	El usuario llamado ha cambiado de dirección temporalmente, y puede ser encontrado en la dirección devuelta.
	305	Use Proxy	El usuario no puede ser llamado directamente, pero debe ser llamado a través de un servidor Proxy. Solo los servidores pueden enviar esta respuesta
	380	Alternative Service	Los servicios requeridos no están permitidos pero son posibles servicios alternativos
Request Failure. Indica que el cliente no esta autorizado a hacer el requerimiento	400	Bad Request	El mensaje de texto no puedo ser entendido por el usuario remoto.
	401	Unauthorized	El usuario requiere autenticación antes de hacer la petición.
	402	Payment Required	El usuario debe dinero
	403	Forbidden	El mensaje fue entendido pero el requerimiento no puede ser atendido.
	404	Not Found	Mensaje enviado cuando el usuario nunca ha existido, o cuando los registros de el han sido borrados del servidor.
	405	Method not allowed	El método enviado no es permitido por el usuario remoto.

	406	Not acceptable	El cliente remoto generara respuesta que no van a ser entendidas por el cliente que llama.
	407	Proxy authentication required	Autenticación requerida ante el servidor Proxy, antes de seguir con la llamada.
	408	Request timeout	El servidor no puede producir una respuesta dentro del tiempo requerido por el que llama en el encabezado de la petición. Generalmente enviada por servidores de red que estén ocupados.
	409	Conflict	Hay un conflicto entre la petición actual y otras condiciones del servidor
	410	Gone	El usuario o servicio requerido no esta en el servidor y no hay dirección de su ubicación.
	411	Length required	El servidor requiere que el cliente ubique la longitud del cuerpo del mensaje en el encabezado.
	413	Request Entity too large	El tamaño del mensaje es muy grande y no puede ser
	414	Request-URL too long	El servidor tiene dificultad en interpretar (Request URI) la dirección de la entidad a la que se ele envía la petición, debido a su tamaño.
	415	Unsupported media type	El servidor no puede aceptar el requerimiento debido a su codificación. El servidor puede indicar el método apropiado para codificar la petición.
	420	Bad extension	El servidor no entiende la extensión del protocolo SIP que intenta usar el que llama.
	480	Temporarily not available	El cliente llamado no esta disponible temporalmente
	481	Call leg/transaction does not exist	El servidor recibe un método CANCEL por una petición que no existe, o un BYE por una llamada que no existe.
	482	Loop detected	Loop en la ruta del mensaje detectado.
	483	Too many hops	El número de hops requerido para llegar al cliente llamado excede el máximo permitido. Protección contra rutas inestables o largas, y routers mal configurados.
	484	Address incomplete	Dirección de destino incompleta.
	485	Ambiguous	Dirección de destino ambigua.

	486	Busy here	El cliente llamado esta ocupado o no quiere contestar la llamada.
Server Failure. El requerimiento puede ser valido pero el servidor no lo puede ejecutar.	500	Server Internal Error.	Error a nivel de hardware, software o cualquier error interno.
	501	Not implemented	El servidor no puede atender la petición porque el servicio no esta implementado.
	502	Bad Gateway	Respuesta errónea recibida por el servidor desde un gateway u otro servidor presente en el camino de la llamada.
	503	Service Unavailable	Servicio temporalmente no disponible.
	504	Gateway Timeout	El servidor finalizo tiempo de espera mientras solicitaba acceso a un gateway en el camino de la llamada al endpoint remoto
	505	SIP versión not supported	Versión de SIP no soportada.
Global Failure. El requerimiento del cliente no puede ser atendido por ningún servidor.	600	Busy everywhere	El cliente llamado esta ocupado.
	603	Decline	El cliente llamado rechaza la llamada
	604	Does not exist anywhere	El usuario llamado no existe en ningún lugar.
	606	Not aceptable	El usuario acepta la llamada pero hay incompatibilidades en la media requerida, y entonces no se puede aceptar la llamada. (Ej. Requerimiento para soporte de codificación g.728, pero solo se acepta codificación g.711)

Tabla 4 Respuestas SIP. [17].

c) Parámetros de encabezado

Los mensajes de encabezado son un tipo de información incluida en las peticiones o respuestas con el fin de dar información adicional del mensaje, o facilitar el manejo del mensaje enviado por parte del servidor o el cliente.

Los parámetros de encabezado de SIP son similares a los de HTTP tanto en sintaxis como en semántica. En particular, los encabezados de SIP muestran las definiciones de la sintaxis, para el encabezado del mensaje y las reglas para encabezados extendidos sobre líneas múltiples.

Los parámetros de encabezado se dividen en 4 categorías como se muestra en la tabla 5.

Encabezados Generales Pueden ser usados tanto en peticiones como en respuestas.	Encabezados de Requerimiento Aplican solo a los requerimientos SIP y son usados para dar información adicional al servidor concerniente a los requerimientos y concernientes al cliente.	Encabezados de Respuesta Aplican solo a las respuestas SIP. Usados para dar información adicional acerca de la respuesta-aquella que no se puede incluir en la línea de estado.	Encabezados de Entidad Indica el tipo y el formato de la información incluido en el cuerpo de un mensaje SIP.
Accept	Hide	Proxy-Authenticate	Allow
Accept-Encoding	In-Reply-To	Retry-After	Content-Disposition
Accept-Language	Max-Forwards	Server	Content-Encoding
Authorization	Priority	Unsupported	Content-Language
Call-ID	Proxy-Authorization	Warning	Content-Length
Contact	Proxy-Require	WWW-Authenticate	Content-Type
Cseq	Route		Expires
Date	Response-Key		
Encryption	Subject		
From			
Organization			
Record-Route			
Require			
Supported			
Timestamp			
To			
User-Agent			
Via			
MIME-Version			

Tabla 5 Encabezados SIP. [17].

d) Cuerpo

Una petición incluye nuevas peticiones definidas en extensiones de la especificación RFC3261 y pueden contener cuerpos de mensaje a menos que se especifique lo contrario. La interpretación del cuerpo del mensaje depende del método de la petición. Para mensajes de respuesta el método de petición y el código de estado determinan el tipo de interpretación de cualquier cuerpo de mensaje, todas las respuestas pueden incluir un cuerpo de mensaje.

e) Formato de los mensajes SIP

Diferente a HTTP, las implementaciones SIP pueden usar UDP o protocolos de data gramas no confiables. Cada uno de estos data gramas lleva una petición o una respuesta. La longitud del valor del encabezado es usado para localizar el final de cada mensaje SIP en una cadena. Esto siempre se presenta cuando los mensajes SIP son enviados sobre una cadena de transporte orientada.

A continuación se describen los comportamientos específicos que deben asumir los participantes dentro de una conversación de VoIP con señalización SIP.

- *COMPORTAMIENTO GENERAL DEL AGENTE USUARIO*

Un Agente Usuario representa un sistema final; éste contiene un Agente Usuario Cliente (UAC – User Agent Client), que genera peticiones y un Agente Usuario servidor (UAS – User Agent Server) que responde estas peticiones. Un UAC es capaz de generar peticiones basadas en estímulos externos, por ejemplo el usuario oprime un botón o una señal de la red pública conmutada, y se procesa una respuesta. Un UAS es capaz de recibir peticiones y generar respuestas basadas en la entrada del usuario, es decir estímulos externos, por ejemplo el resultado de la ejecución de un programa o cualquier otro mecanismo.

Cuando un UAC envía una petición, ésta pasa a través de un número de servidores Proxy (servidores delegados), que reenvían esta petición hacia el UAS. Cuando el UAS genera una respuesta, la respuesta es reenviada de nuevo al UAC.

Los procedimientos de UAC y UAS dependen básicamente de dos factores, el primero basado en si la petición o la respuesta está dentro o fuera de un diálogo, y el segundo basado en el método de la petición. Los diálogos representan la relación que hay punto a punto entre el agente usuario y el establecimiento de los métodos SIP específicos, como la INVITACIÓN.

Un conjunto limitado de características privadas son soportadas por cuerpos encriptados usando S/MIME.

- *COMPORTAMIENTO DEL UAC*

Esta sección cubre el comportamiento de un UAC por fuera de un diálogo.

a) Generación de una petición

Una petición SIP válida formulada por un UAC, mínimo debe contener los siguientes parámetros en el encabezado: Para (To), De (From), Número de secuencia (Cseq), Identificación de llamada (Call-ID), Max-Forwards y Via. Todos estos encabezados son obligatorios en todas las peticiones SIP. Estos seis parámetros del encabezado son los bloques fundamentales de construcción de un mensaje SIP. La unión de todos estos proporciona todos los servicios críticos de enrutamiento de mensajes incluyendo su direccionamiento, enrutamiento de respuestas, propagación limitada de mensajes, ordenamiento y una única identificación de transacciones. Estos encabezados se agregan a la línea obligatoria de petición que contiene el método, petición URI y la versión de SIP.

Los ejemplos de peticiones enviadas fuera del diálogo incluyen una INVITACIÓN para establecer una sesión y las OPCIONES para preguntar por las capacidades.

b) Petición – URI

La petición inicial – URI del mensaje, se debe poner el valor de URI en el parámetro de encabezado “Para” (To). Una excepción notable es el método REGISTRO. Esto puede ser inconveniente por razones de privacidad, o conveniente por poner estos parámetros con el mismo valor.

En circunstancias especiales, la presencia de una ruta pre-existente puede afectar la petición-URI del mensaje. Una ruta pre-existente fija es un grupo de URIs que identifican una cadena de servidores, a los cuales un UAC enviará peticiones salientes, que están fuera de un diálogo. Comúnmente están

configurados en el agente usuario por una persona, por un proveedor de servicio manual o a través de otro mecanismo que no sea SIP.

Call ID: Sirve para comparar peticiones con sus correspondientes respuestas, detecta duplicados, y diferentes Call ID sirven para cambiar parámetros en una conferencia. Ejemplo: paolari@192.190.132.20, donde 'paolari' es único para cada host y la dirección IP lo hace globalmente único. Se genera un nuevo Call ID para cada llamada.

Cseq: Compuesto por un número de secuencia y el nombre del método (ejemplo: 1 INVITE), este número se incrementa con cada petición (excepto por las peticiones ACK y Cancel), el servidor copia este número para las correspondientes respuestas.

From: Debe estar en todas las peticiones y respuestas, contiene un nombre opcional y la dirección del originador del mensaje de petición.

To: Debe estar en todas las peticiones y respuestas e indica el destino de la petición, se copia para las repuestas.

Via: Se usa para grabar la ruta de una petición con el fin de permitir que servidores intermediarios de SIP reenvíen las respuestas por el mismo camino.

Encryption: Especifica que el cuerpo del mensaje y posiblemente algunos encabezados de mensajes han sido encriptados

Content –Type: Describe el tipo de datos del contenido del cuerpo del mensaje, ejemplo: "application/sdp", el mensaje contiene una descripción de sesión usando SDP.

Content –length: El número de octetos del cuerpo del mensaje

Versión: Los Mensajes de petición y respuesta incluyen la versión de SIP en uso, y siguen H3.1, (como HTTP es reemplazado por SIP, HTTP/1.1 es reemplazado por

SIP/2.0), en referencia a la petición de la versión, requerimientos de cumplimiento y actualización de números de versión. Para cumplir con esta especificación, las aplicaciones que envíen mensajes SIP deben incluir la versión de SIP como “SIP/2.0”. [14]

- *COMPORTAMIENTO DEL UAS*

Cuando es procesada una petición por el UAS por fuera de un diálogo, hay un conjunto de procedimientos que son seguidos de forma independiente del método. La sección 12 del RFC3261 da una guía de cómo un UAS puede determinar cuando una petición está fuera o dentro de un diálogo.

Si una petición es aceptada, todos los cambios de estado asociados deben ser ejecutados. Si es rechazada, todos los cambios de estado no deben ser ejecutados.

Los UAS, deben procesar peticiones, en el orden de pasos que muestra ésta sección (que comienza con autenticación, luego método de inspección, los parámetros de encabezados y todo lo que concierne a ésta sección).

a) *Método de inspección*

Una vez ha sido autenticada la petición (o la autenticación ha sido omitida), el UAS debe inspeccionar el método de la petición. Si el UAS reconoce pero no soporta el método de la petición, esto debe generar una repuesta 405 (método no permitido). Los procedimientos para generar respuestas son descritos en la sección 8.2.6. del RFC3261. El UAS debe igualmente agregar un parámetro de encabezado de Permitir, a la respuesta 405. El parámetro de respuesta Permitir, debe listar el conjunto de métodos soportados por el UAS generando un mensaje. Si el método es soportado por el servidor el proceso continúa.

b) Inspección de encabezado

Si un UAS no comprende un parámetro de encabezado en una petición (que es el parámetro de encabezado que no está definido en esta especificación o en ninguna extensión soportada), el servidor debe ignorar el parámetro de encabezado y continuar el procesamiento del mensaje. Un UAS debe ignorar cualquier parámetro de encabezado malformado que no es requerido para en procesamiento de la petición.

c) Parámetro Para y la petición – URI

El parámetro de encabezado Para, identifica el receptor original de la petición designada por el usuario identificado en el parámetro De. El receptor original puede o no ser un procesador de peticiones, debido a la llamada reenviada o a las operaciones Proxy. Un UAS puede aplicar cualquier política que quiera, para determinar que acepte las peticiones cuando el parámetro de encabezado Para no es la identidad del UAS; sin embargo, es recomendado que el UAS acepte las peticiones uniformemente si no reconoce el esquema URI en el parámetro de encabezado Para, o si el parámetro de encabezado Para, no direcciona un usuario conocido o actual por éste UAS. Si de otra forma, el UAS decide rechazar la petición, esto puede generar una respuesta con un código de estado 403 (Prohibido) y pasa al servidor de transacción para la transmisión.

Sin embargo la petición URI, identifica el UAS que procesa la petición. Si la petición URI usa un esquema que no es soportado por el UAS, éste debe rechazar la petición con una respuesta 416 (Esquema URI no soportado). Si la petición URI no identifica una dirección tal que el UAS esté dispuesto a aceptar la petición, debe rechazar la petición con una respuesta 404 (No encontrado). Típicamente, un UA que usa el método de REGISTRO para unir su dirección de registro a una dirección de contacto específica, verá peticiones URI iguales que las direcciones de contacto. Otras fuentes potenciales de Peticiones URI recibidas

incluyen el parámetro de encabezado Contacto de las peticiones y respuestas enviados por el UA que establece o refresca los diálogos.

d) Peticiones combinadas

Si la petición no tiene una etiqueta en el parámetro de encabezado Para, el núcleo del UAS debe comprobar la petición frente a las transacciones en ejecución. Si las etiquetas De (From), Identificación de llamada (Call ID) y Cseq coinciden exactamente con aquellas asociadas con las transacciones en ejecución, pero la petición no coincide con esa transacción (basado en la reglas de coincidencia en la sección 17.2.3, el núcleo del UAS debe generar una respuesta 482 (Anillo detectado) y pasarla al servidor de transacción.

La misma petición ha llegado al UAS más de una vez, ejecutando diferentes trayectorias, muchas debidas a bifurcaciones. El UAS procesa la primera petición recibida y responde con un 482 (Anillo detectado) al resto de ellas.

e) Requisito

Asumiendo que el UAS decide que es el elemento apropiado para procesar la petición, éste examina el parámetro de encabezado Requisito, si se presenta.

El parámetro de encabezado Requisito es usado por el UAC para decirle al UAS acerca de las extensiones SIP que el UAC espera que el UAS soporte en orden de procesar su petición apropiadamente. Su formato es descrito en la sección 20.3.2 del RFC3261. Si un UAS no entiende la etiqueta Opción, listada en el parámetro de encabezado Requisito, éste debe responder generando un código de estado 420 (Extensión Mala). El UAS, debe agregar un parámetro de encabezado No soportado y listarlo en las opciones que no son entendidas en el parámetro de encabezado Requisito.

Hay que tener en cuenta que Requisito y Requisito Proxy no deben ser usados en una petición Cancelar de SIP o en una petición Reconocimiento enviada por una

respuesta que no sea 2xx. Éste parámetro de encabezado debe ser ignorado si esto está presente en estas peticiones.

Una petición de Reconocimiento de respuesta 2xx debe contener sólo valores de Requisito y Requisito Proxy que están presentes en la petición inicial.

Ejemplo:

UAC->UAS: INVITE sip:watson@bell-telephone.com SIP/2.0

Require: 100rel

UAS->UAC: SIP/2.0 420 Bad Extension

Unsupported: 100rel

Éste comportamiento asegura que la interacción cliente-servidor procederá sin retardo cuando todas las opciones son entendidas por ambos lados, y sólo se introduce retardo si hay algunas que no son entendidas. Para una buena relación cliente-servidor, la interacción procede rápidamente, ahorrando un viaje redondo siempre requerido por la negociación de los mecanismos. Adicionalmente, éste también remueve las ambigüedades cuando el cliente requiere características que el servidor no entiende. Algunas características, como el parámetro de manejo de llamadas, son de interés sólo para los sistemas finales.

f) Contenido de procesamiento

Asumiendo que el UAS entiende todas las extensiones requeridas por el cliente, el UAS examina el cuerpo del mensaje y los parámetros de encabezado que éste describe. Si éste es un cuerpo cuyo tipo (indicado por el Tipo de contenido), el lenguaje (Indicado por el Contenido de Lenguaje) o la codificación (Indicada por el Contenido de codificación) no son entendidos, y esa parte del cuerpo no es opcional, (Como indica el parámetro de encabezado de Contenido de disposición), el UAS debe rechazar la petición con una respuesta 415 (Tipo de

medio no soportado). La respuesta debe contener un parámetro de encabezado de Aceptación listando los tipos de todos los cuerpos que son entendidos, en el evento que el tipo de cuerpo que contiene la petición no sea soportado por el UAS. Si la petición tiene contenido codificado que no es entendido por el UAS, la respuesta debe contener un parámetro de encabezado de Aceptación codificada listando los códigos entendidos por el UAS. Si la petición tiene lenguajes que no son entendidos por el UAS, la respuesta debe contener un parámetro de encabezado de Aceptación de lenguaje indicando los lenguajes que son entendidos por el UAS. Más allá de estas comprobaciones, el manejo del cuerpo depende del método y del tipo. Para información futura acerca del procesamiento del contenido específico de los parámetros de encabezado, vea la sección 7.4 al igual que la sección 20.11 y 20.15 del RFC3261.

g) Aplicando extensiones

Un UAS que quiere aplicar alguna extensión cuando está generando la respuesta, no debe hacerlo sin soporte para esta extensión, que es indicada en el parámetro de encabezado de la petición. Si la extensión deseada no es soportada, el servidor debe confiarse sólo de la línea base SIP y de ninguna otra extensión soportada por el cliente. En circunstancias extrañas, cuando el servidor no puede procesar una petición sin la extensión, el servidor puede enviar una respuesta 421 (Extensión requerida). Ésta respuesta indica que no se puede generar la respuesta apropiada sin soporte de la extensión específica. La extensión necesitada se debe incluir en el parámetro de encabezado de Requisito. Éste comportamiento no es recomendado si éste generalmente va a romper la interoperabilidad.

Cualquier extensión a la que no se aplica una respuesta 421 debe ser listada en el parámetro de encabezado de Requisito incluido en la respuesta. Por supuesto, que el servidor no debe aplicar extensiones que no estén listadas en el parámetro de encabezado Requisito; como resultado de esto el parámetro de encabezado

Requisito en una respuesta, sólo contendrá etiquetas opcionales definidas en el estándar del RFC.

h) Procesamiento de una petición

Asumiendo que todas las comprobaciones de las secciones previas han pasado, el procesamiento del UAS comienza un método específico. La sección 10 cubre el REGISTRO de la petición, la sección 11 cubre las OPCIONES de la petición, la sección 13 cubre la INVITACIÓN de la petición y la sección 15 cubre el BYE de la petición.

i) Generando la respuesta

Cuando un UAS quiere construir una respuesta a una petición, éste ejecuta los procedimientos generales detallados en las siguientes subsecciones. Los comportamientos adicionales específicos al código de respuesta en cuestión, que no son detallados en esta sección, de todas formas son requeridos.

Una vez todos los procedimientos asociados con la creación de una respuesta son completados, el UAS retorna la respuesta al servidor de transacción, desde donde recibió la petición.

j) Enviando una respuesta provisional

Una guía específica para la generación de respuestas, es que el UAS no debe considerar una respuesta provisional para una petición de no invitación. Mejor el UAS debe generar un respuesta final a una petición de no invitación tan pronto como sea posible.

Cuando una respuesta 100 (intento) es generada, cualquier parámetro de encabezado Timestamp presente en la petición, debe ser copiado en la respuesta 100. Si hay un retraso generando la respuesta, el UAS debe agregar un valor de retardo en el valor del tiempo estampado en la respuesta. Éste valor debe

contener la diferencia entre el tiempo de envío de la respuesta y el recibo de la petición, medida en segundos.

k) Encabezados y etiquetas

El parámetro de encabezado De, de la respuesta debe ser igual al parámetro de encabezado De, de la petición. El parámetro de encabezado Call ID de la respuesta, debe ser igual al parámetro de encabezado de Call ID de la petición. El parámetro de encabezado CSeq de la respuesta debe ser igual al parámetro de encabezado Cseq de la petición. Los valores del parámetro de encabezado Vía de la respuesta deben ser igual a los valores del parámetro de encabezado Vía de la petición y deben mantener el mismo orden.

Si una respuesta contiene una etiqueta Para en la petición, el parámetro de encabezado Para en la respuesta, debe ser igual en la respuesta. Sin embargo si el parámetro de encabezado Para en la petición, no contiene una etiqueta, el URI en el parámetro de encabezado Para en la respuesta, debe ser igual al URI en el parámetro de encabezado Para; adicionalmente, el UAS debe agregar una etiqueta al parámetro de encabezado Para en la respuesta (a excepción de la respuesta 100, en la cual una etiqueta puede estar presente). Esto sirve para identificar el UAS que está respondiendo, que posiblemente resulte en un componente de identificación del diálogo. La misma etiqueta debe ser usada para todas las respuestas Para de la petición, ambas, final y provisional (de nuevo exceptuando la respuesta 100). Los procedimientos para la generación de etiquetas están definidos en la sección 19.3 del RFC3261.

l) Comportamiento sin estado del UAS

Un UAS sin estado, es un UAS que no mantiene el estado de la transacción. Éste responde normalmente a las peticiones, pero descarta cualquier estado que ordinariamente es retenido por el UAS, después de haber enviado una respuesta. Si un UAS sin estado, recibe una retransmisión de una petición, éste genera la respuesta y vuelve a reenviarla, así haya respondido a la primera instancia de la

petición. Un UAS no puede ser sin estado, a menos que la petición procesada para este método, siempre resulte en la misma respuesta si las peticiones son idénticas. Estas reglas, sacan los registros sin estado. Un UAS sin estado no usa la capa de transacción, ellos reciben las peticiones directamente de la capa de transporte, y envían las respuestas directamente a la capa de transporte.

El papel de los UAS sin estado, es necesario principalmente para manejar peticiones no autenticadas, para la cuales se requieren respuesta. Si se manejan peticiones con todos los estados, entonces inundaciones malignas de peticiones no autenticadas, pueden crear una cantidad de transacciones de estados, que pueden demorar o detener por completo el proceso de llamada en el UAS, creando efectivamente una negación de la condición del servicio.

Los comportamientos más importantes de un UAS sin estado se nombran a continuación:

- Un UAS sin estado no debe enviar respuestas provisionales 1xx.
- Un UAS sin estado no debe retransmitir respuestas.
- Un UAS sin estado debe ignorar las peticiones de reconocimiento.
- Un UAS sin estado debe ignorar las peticiones de cancelación.
- Las etiquetas de encabezado Para, deben ser generadas por respuestas de manera sin estado, de una manera que genere le misma etiqueta para la misma petición de forma consistente.

En los demás aspectos, el UAS sin estados se comporta de la misma manera, como un UAS con todos los estados. Un UAS puede operar modo todos los estados o en modo sin estados, para cada respuesta nueva.

- *COMPRESIÓN DE VOZ*

Los codificadores de voz (COder/DECoder) también llamados Voice CODEC’s, de aquí en adelante codecs son algoritmos de codificación que convierten la señal de

voz analógica en un flujo digital de datos mediante muestreo, que es la discretización de la señal en el tiempo y cuantificación, es decir la discretización de la señal en amplitud.

Los algoritmos agregan facilidades como detección de inactividad, cancelación de eco y reducción de ruido. Las características de calidad y retardo varían según cada implementación y no hay una clase predominante.

Las siguientes recomendaciones de la ITU-T establecen las características de los codec's cuyos requerimientos de ancho de banda de la voz comprimida se indican en la tabla 6.

CODEC	Método de compresión	BIT Rate (Kbps)
G.711	PCM Pulse Code Modulation	64
G.723.1	CELP Code Excited Linear prediction	5.3
G.723.1	MP-MPQL Low Bit Rate Vocoder for Multimedia	6.4
G.726	ADPCM Adaptive Diferencial PCM	32
G.728	LD CELP Low Delay CELP	16
G.729	CS ACELP Conjugate Structure Algebraic CELP	8

Tabla 6 Codificadores de Voz y su requerimiento de ancho de banda.

G.711 es el estándar internacional para la codificación de audio en un canal de 64 kbps. Su esquema de modulación es por codificación de pulso (PCM) la cual funciona a una tasa de muestreo de 8 kilociclos por segundo, con 8 bits por muestra. G.711 puede codificar frecuencias entre 0 y 4 kilociclos y puede seleccionar entre dos tipos de cuantizaciones logarítmicas, ley A y ley μ donde la ley A es el estándar para los circuitos internacionales.

Estos esquemas cuantización se diseñan con funciones logarítmicas, es decir, se asignan más bits a los valores más bajos de la señal y pocos bits a los valores más altos de la señal. Esto se asegura de que las señales de amplitud baja sean representadas adecuadamente, mientras que se mantienen suficientes gamas para codificar amplitudes grandes.

La codificación real no utiliza funciones logarítmicas, sin embargo, el intervalo de entrada está dividido en segmentos, en donde cada segmento usa un intervalo diferente para valores de decisión. La mayoría de los segmentos contienen 16 intervalos, y el tamaño del intervalo se duplica segmento a segmento. La ilustración demuestra tres segmentos con cuatro intervalos en cada uno.

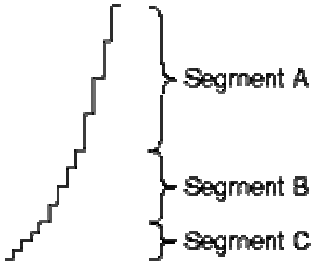

FIGURA 4 Cuantización logarítmica.

Ambas la ley A y la ley μ simétricas alrededor de cero. La ley μ, utiliza 8 segmentos de 16 intervalos cada uno, en cada una de las direcciones positivas y negativas, comenzando con un tamaño de intervalo de 2 en el segmento 1, y va aumentando el tamaño del intervalo a 256 en el segmento 8. La ley A, utiliza 7 segmentos, el segmento más pequeño, usa un intervalo de 2, es dos veces el tamaño de los otros (32 intervalos). Los seis segmentos restantes son "normales", con 16 intervalos cada uno, y aumenta el tamaño del intervalo a 128, en el segmento 7, es por eso, que la ley A puede hacer representación de señales más pequeñas con mayor fidelidad.

El estándar G.711 tiene dos apéndices, el primero (A high quality low-complexity algorithm for packet loss concealment with G.711) proporciona un método PLC para la recomendación G.711 y el segundo (A comfort noise payload definition for ITU-T G.711 use in packet-based multimedia communication systems) define un formato de carga útil de ruido no perturbador (bit-stream) para el uso de ITU-T G.711 en sistemas de comunicación basados en paquetes multimedia. El uso del formato de la carga útil, se piensa para los sistemas basados en paquetes con

encabezados largos donde la transmisión de paquetes desempeña un papel significativo en el bit-rate del sistema. En este caso, el uso de VAD/DTX/CNG puede reducir apreciablemente el bit-rate y por lo tanto mejorar la eficiencia del ancho de banda. [16][26]

A continuación se enumeran aplicaciones más comunes de G.711 [16]

- WIFI phones VoWLAN
- Wireless GPRS EDGE systems.
- Personal Communications
- Wideband IP telephony
- Audio and Video Conferencing
- Wideband IP telephony

1.2.2 NIVEL DE TRANSPORTE

- *PROTOCOLO UDP*

El protocolo UDP, User Datagram Protocol por sus siglas en inglés, a diferencia del TCP, no es un protocolo seguro, no es orientado a conexión, no es fiable, no envía acuses de recibo, no ordena los mensajes, no controla la velocidad a la que fluye la información, sus mensajes pueden llegar fuera de secuencia o perderse y reduce la información suplementaria a enviar.

Puerto UDP de origen	Puerto UDP de destino
Longitud de mensaje UDP	Suma de verificación UDP
Datos	
.....	

FIGURA 5 Formato de la Trama UDP.

Puerto de origen: es opcional y se refiere al puerto de origen del data grama. Puede tener valor 0.

Puerto de destino: es el puerto al que va dirigido el data grama.

Longitud: número de bytes en el data grama incluyendo los del encabezado.

Suma de verificación: es opcional. Si su valor es cero significa que no se computó

El protocolo UDP tiene las siguientes funciones como incorporar el direccionamiento a un puerto como el puerto de origen y el puerto de destino e incorporar un campo de longitud de datos y la suma de comprobación (si se detecta un error se descarta el data grama).

- *PROTOCOLO TCP*

Es un protocolo seguro, orientado a conexión y fiable debido a que controla la llegada de la tramas (ACK), solicita reenvíos, utiliza el sistema de ventana deslizante y utiliza buffers para la transferencia haciéndola más eficiente. Por lo tanto acumula datos hasta que tiene suficientes para llenar un data grama, o se puede forzar el envío además la conexión es full-duplex lo que permite la transferencia en ambas direcciones.

La fiabilidad de TCP se ve reflejada cuando hay pérdida de tramas y responde a la llegada de estas mediante asentimientos (ACK).

Además utiliza temporizadores para controlar la pérdida de tramas, cada pérdida falsa será detectada y se enviará un duplicado, el receptor detectará el doble envío gracias al número de secuencia y descartará la trama.

En cuanto a la eficiencia y control de flujo, se utiliza un sistema de ventana deslizante para gestionar el flujo, también se utiliza el sistema de superposición para el ahorro de ancho de banda consumido por los ACKs y se aprovecha la utilización de buffers para reducir el número de data gramas.

La ventana es de tamaño variable y está controlada por el receptor, no controla el número de tramas recibidas, si no el número de bytes.

Puerto de origen y destino: contiene el número del puerto origen y el destino.

Número de secuencia: identifica el número de secuencia de la trama, TCP no numera los paquetes si no los bytes que transmite.

Número de ACK: identifica el número de byte que se espera recibir como siguiente.

Longitud de cabecera: nos indica la longitud de la cabecera medida en palabras de 32 bits

<table>
<tr><td colspan="2">PUERTO DE ORIGEN</td><td colspan="2">PUERTO DE DESTINO</td></tr>
<tr><td colspan="4">NÚMERO DE SECUENCIA</td></tr>
<tr><td colspan="4">NUMERO DE ACK</td></tr>
<tr><td>LONGITUD</td><td>RESERVA</td><td>CONTROL</td><td>VENTANA</td></tr>
<tr><td colspan="2">SUMA DE COMPROBACIÓN</td><td colspan="2">PUNTERO URGENTE</td></tr>
<tr><td colspan="4">OPCIONES + RELLENO</td></tr>
</table>

FIGURA 6 Formato de la trama TCP. [28].

A continuación se describen cada uno de los bits de control de la trama TCP.

URG: indica que el campo puntero de datos urgentes se encuentra activo.

ACK: cuando está a 1 indica que sirve como asentimiento de un data grama.

PSH: indica la entrega inmediata de los datos al nivel superior, no van a esperar que se llene el buffer.

RST: si se encuentra a 1 indica el reseteo de la conexión.

SYN: Es la bandera de establecimiento de la conexión. Se utiliza para establecer la conexión.

Ventana: indica el tamaño de la ventana. Con un valor 0 indica que el receptor necesita un descanso.

Suma de comprobación: asegura la integridad del data grama.

Puntero urgente: indica un desplazamiento a partir del cual aparecen datos urgentes, sustituye al concepto de interrupción.

Opciones: se creó para añadir características extras no cubiertas por la cabecera normal.

Los puerto de comunicación. Puede pensarse en ellos como una cola donde el protocolo sitúa los data gramas TCP, los utiliza para averiguar el destino último dentro de la máquina que recibe la trama permite que varios programas dentro de una misma máquina se comuniquen concurrentemente TCP se encargará de multiplexar los datos entrantes de diferentes programas.

DECIMAL	PALABRA CLAVE	DESCRIPCIÓN
80	http	Acceso a la WEB
79	finger	Finger
25	SMTP	Simple Mail Transfer Protocol
23	telnet	Conexión con Terminal
21	ftp	Transferencia de archivos

Tabla 7 Ejemplos de puertos bien conocidos

- *RTP/RTCP*

Real-time Transport Protocol o Protocolo de transporte en tiempo real, es definido en RFC 3550 por la EITF y adoptado por la ITU-T. El protocolo de transporte de tiempo real (RTP), proporciona servicios de entrega de datos con características de tiempo real punto a punto, como por ejemplo audio y video interactivos. Esos servicios incluyen identificación de la carga útil, enumeración de secuencia, etiquetado de tiempo y supervisión de entrega. Las aplicaciones RTP típicamente se implementan encima de UDP para hacer uso de su multiplexación y sus servicios de verificación de suma; ambos protocolos contribuyen con partes de la funcionalidad del protocolo de transporte. Sin embargo, RTP se puede utilizar con otros protocolos de red o de transporte subyacentes convenientes.

Debe observarse que RTP por sí mismo no proporciona ningún mecanismo para asegurar entrega oportuna o para proporcionar otras garantías del calidad de servicio, pero confía en servicios de la capa inferior para hacer eso. Los números de serie incluidos en RTP permiten que el receptor reconstruya la secuencia del paquete del remitente, pero los números de serie también se pueden utilizar para determinar la localización apropiada de un paquete, por ejemplo en descifrar video, sin que los paquetes estén necesariamente en secuencia. Mientras que RTP se diseña sobre todo para satisfacer las necesidades de las conferencias multimedia multi-participante, no se limita a ese uso en particular. El almacenaje de datos continuos, simulación distribuida interactiva, divisa activa y usos de control y de medida pueden también aplicarse a RTP

El protocolo de transporte en tiempo real (RTP), se utiliza entonces para llevar los datos que tienen características de tiempo real, mientras que el protocolo del control de RTP (RTCP), se usa para supervisar la calidad del servicio y transportar la información sobre los participantes en una sesión en curso. El segundo aspecto de RTCP puede ser suficiente para sesiones controladas libremente, por ejemplo donde no hay control y disposición explícitos de la calidad de miembro, pero no se piensa necesariamente para apoyar todos los requisitos de control de la comunicación en uso. Esta funcionalidad se puede incluir completamente o parcialmente por un protocolo separado del control de sesión. [11] [12]

RTCP implica la transmisión periódica de paquetes de control a todos los participantes en una sesión, su función principal es proporcionar mecanismos de realimentación para informar sobre la calidad en la distribución de los datos. Esta información se puede utilizar para diagnosticar fallos en la distribución y construir codificadores adaptables (Sure-Stream de Real-Networks).

El RTCP aporta un identificador para la capa de transporte denominado identificador canónico CNAME (Canonical Name), este se utiliza para identificar a cada participante. [28]

Formato del paquete RTP

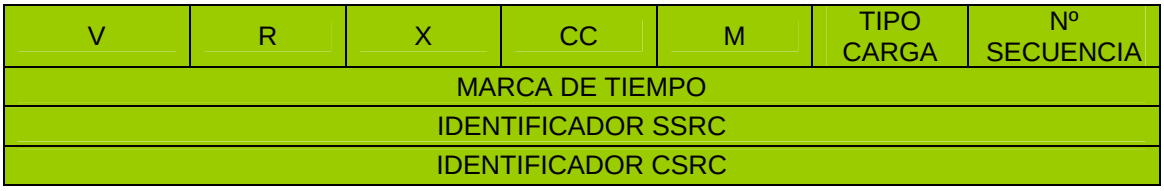

FIGURA 7 Formato de paquete RPT

Versión (V): identifica la versión del RTP. La versión actual es la 2 (2 bits).

Relleno (R): este campo indica que el paquete contiene uno o más bytes de relleno que no son parte de la carga útil. El último byte de la carga útil indica cuantos bytes tienen que ser ignorados (1 BIT).

Extensión (X): cuando este BIT está a 1 se indica que la cabecera está seguida de una extensión de cabecera.

Cuenta CSRC (CC): especifica el número de CSRC que siguen a la cabecera (4 bits).

Marcador (M): este campo se define en el perfil. Se utiliza para marcar eventos significativos (1 BIT).

Tipo de carga útil (TC): identifica el formato de la carga útil y determina como las

aplicaciones tienen que interpretarla (7 bits).

Número de Secuencia: se incrementa cada vez que se envía un nuevo paquete. Este número se utiliza en el destino para detectar pérdidas y ordenar los paquetes en la secuencia original. El valor inicial se elige de forma aleatoria

SSRC: es un número aleatorio. Se utiliza para identificar a la fuentes de sincronización dentro de la misma sesión RTP. Indica donde se combinaron los datos o la fuente de los mismos (si solo hay una fuente).

Lista CSRC: contiene de 0 a 15 ítems de 32 bits cada uno de los cuales especifica la fuente que a contribuido a la carga útil del paquete. El número de identificadores se especifica en CC.

Los paquetes RTCP tienen las siguientes características:

Informe del emisor (IE): distribuye las estadísticas de emisión y de recepción de los emisores activos (los emisores también pueden ser receptores)

Información del receptor (IR): distribuye las estadísticas de recepción de los participantes que no sean emisores activos.

Descripción de la fuente (DESF): proporciona información para describir la fuente. Por ejemplo mediante: el CNAME, el nombre, el número de teléfono, la localización y la versión de las herramientas de aplicación.

BYE: indica el final de la participación de un emisor

APP: esta paquete se utiliza para transportar información específica de las diferentes aplicaciones que pueden utilizar el protocolos RTP.

Los paquetes IE proporcionan un informe del emisor y varios del receptor:

Informe del Emisor: Referencia de reloj dada la marca de tiempo dada por el protocolo de temporización de red, NTP (Network Time Protocol), número de segundos desde las 0 horas del 1 de enero de 1990, el mismo instante temporal que la marca NTP pero utilizando el reloj utilizado para generar las marcas de tiempo RTP (esta correspondencia se puede utilizar para realizar la correspondencia intra-media e inter-media), número de paquetes transmitidos, total de bytes de carga útil transmitidos desde que comenzó a transmitir

1.2.3 NIVEL DE RED (IP V.4)

El Protocolo Internet definido en el RFC791, está diseñado para uso en sistemas interconectados de redes de comunicación de ordenadores por intercambio de paquetes. A un sistema de este tipo se le conoce como "catenet". El protocolo Internet proporciona los medios necesarios para la transmisión de bloques de datos llamados data gramas desde el origen al destino, donde origen y destino son hosts identificados por direcciones de longitud fija. El protocolo Internet también se encarga, si es necesario, de la fragmentación y reensamblado de grandes data gramas para su transmisión a través de redes de trama pequeña. [28]

- *INTERFACES*

Este protocolo es utilizado por protocolos host-a-host en un entorno Internet. Este protocolo utiliza a su vez protocolos de red locales para llevar el data grama Internet al próximo gateway o host destino.

Por ejemplo, un módulo TCP llamaría al módulo Internet para tomar un segmento TCP (incluyendo la cabecera TCP y los datos de usuario) como la parte de datos de un data grama Internet. El módulo TCP suministraría las direcciones y otros parámetros de la cabecera Internet al módulo Internet como argumentos de la llamada. El módulo Internet crearía entonces un data grama Internet y utilizaría la interfaz de la red local para transmitir el data grama Internet.

- *OPERACIÓN*

El protocolo Internet implementa dos funciones básicas: direccionamiento y fragmentación. Los módulos Internet usan las direcciones que se encuentran en la cabecera Internet para transmitir los data gramas Internet hacia sus destinos. La selección de un camino para la transmisión se llama rutado.

Los módulos Internet usan campos en la cabecera Internet para fragmentar y reensamblar los data gramas Internet cuando sea necesario para su transmisión a través de redes de "trama pequeña".

El modelo de operación es que un módulo Internet reside en cada host involucrado en la comunicación Internet y en cada gateway que interconecta redes. Estos módulos comparten reglas comunes para interpretar los campos de dirección y para fragmentar y ensamblar data gramas Internet. Además, estos módulos (especialmente en los gateways) tienen procedimientos para tomar decisiones de rutado y otras funciones.

El protocolo Internet trata cada data grama Internet como una entidad independiente no relacionada con ningún otro data grama Internet. No existen conexiones o circuitos lógicos (virtuales o de cualquier otro tipo).

El protocolo Internet utiliza cuatro mecanismos clave para prestar su servicio: Tipo de Servicio, Tiempo de Vida, Opciones, y Suma de Control de Cabecera.

El Tipo de Servicio se utiliza para indicar la calidad del servicio deseado. El tipo de servicio es un conjunto abstracto o generalizado de parámetros que caracterizan las elecciones de servicio presentes en las redes que forman Internet. Esta indicación de tipo de servicio será usada por los gateways para seleccionar los parámetros de transmisión efectivos para una red en particular, la red que se utilizará para el siguiente salto, o el siguiente gateway al enrutar un data grama Internet.

El Tiempo de Vida es una indicación de un límite superior en el periodo de vida de un data grama Internet. Es fijado por el remitente del data grama y reducido en los puntos a lo largo de la ruta donde es procesado. Si el tiempo de vida se reduce a cero antes de que el data grama llegue a su destino, el data grama Internet es destruido. Puede pensarse en el tiempo de vida como en un plazo de autodestrucción.

Las opciones proporcionan funciones de control necesarias o útiles en algunas situaciones pero innecesarias para las comunicaciones más comunes. Las opciones incluyen recursos para marcas de tiempo, seguridad y rutado especial.

La Suma de Control de Cabecera proporciona una verificación de que la información utilizada al procesar el data grama Internet ha sido transmitida correctamente. Los datos pueden contener errores. Si la suma de control de cabecera falla, el data grama Internet es descartado inmediatamente por la entidad que detecta el error.

El protocolo Internet no proporciona ningún mecanismo de comunicación fiable. No existen acuses de recibo ni entre extremos ni entre saltos. No hay control de errores para los datos, sólo una suma de control de cabecera. No hay retransmisiones. No existe control de flujo.

Los errores detectados pueden ser notificados por medio del Internet Control Message Protocol (ICMP) (Protocolo de Mensajes de Control de Internet) [3] el cual está implementado en el módulo del protocolo Internet.

- *RELACIÓN CON OTROS PROTOCOLOS*

El siguiente diagrama ilustra el lugar del protocolo Internet en la jerarquía de protocolos:

TELNET	FTP	TFTP	....
TCP	UDP	....	
PROTOCOLO DE INTERNET & ICMP			
PROTOCOLO DE LA RED LOCAL			

FIGURA 8 Relación entre Protocolos

El protocolo Internet interactúa por un lado con los protocolos host- a-host de alto nivel y por otro con el protocolo de la red local. En este contexto una "red local" puede ser una pequeña red en un edificio o una gran red como ARPANET.

- *MODELO DE OPERACIÓN*

El modelo de operación para transmitir un data grama de una aplicación a otra se ilustra en el siguiente escenario:

Suponemos que esta transmisión involucra a una gateway intermedia.

La aplicación remitente prepara sus datos y llama a su módulo Internet local para enviar esos datos como un data grama y pasa la dirección de destino y otros parámetros como argumentos de la llamada.

El módulo Internet prepara una cabecera de data grama y adjunta los datos a él. El módulo Internet determina una dirección de la red de área local para esta dirección Internet, que en este caso es la dirección de una gateway.

Envía este data grama y la dirección de red local al interfaz de red local.

El interfaz de red local crea una cabecera de red local, le adjunta el data grama y entonces envía el resultado a través de la red local.

El data grama llega a un host gateway encapsulado en la cabecera de red local, el interfaz de red local desprende esta cabecera y dirige el data grama hacia el módulo Internet. El módulo Internet determina a partir de la dirección Internet que el data grama debe ser reenviado a otro host en una segunda red. El módulo Internet determina una dirección de red local para el host destino. Llama al interfaz de red local de esa red para enviar el data grama.

Este interfaz de red local crea una cabecera de red local y le adjunta el data grama enviando el resultado al host destino.

En este host destino el interfaz de red local le quita al data grama la cabecera de red local y se lo pasa al módulo Internet.

El módulo Internet determina que el data grama va dirigido a una aplicación en este host. Pasa los datos a la aplicación en respuesta a una llamada al sistema, pasando la dirección de origen y otros parámetros como resultado de la llamada.

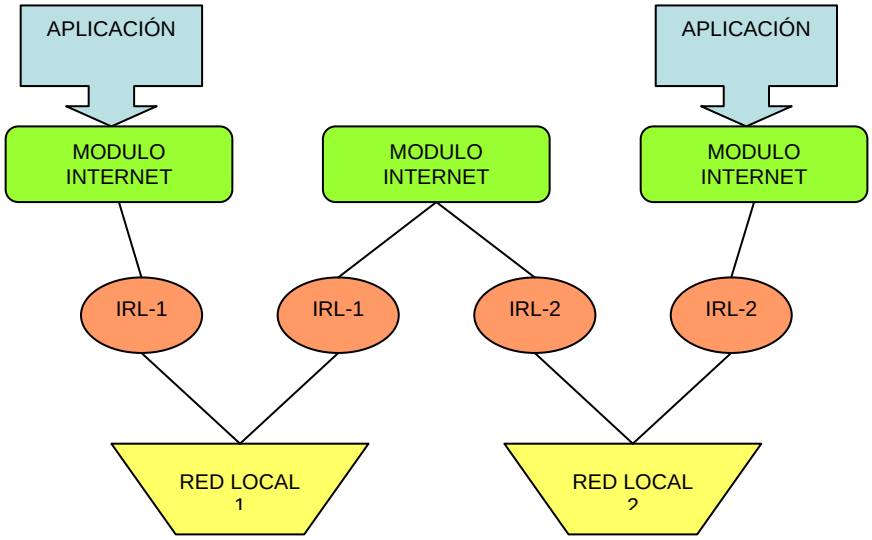

FIGURA 9 Trayectoria de la transmisión

- *DESCRIPCIÓN DE FUNCIONES*

La función o propósito del Protocolo Internet es mover data gramas a través de un conjunto de redes interconectadas. Esto se consigue pasando los data gramas desde un módulo Internet a otro hasta que se alcanza el destino. Los módulos Internet residen en hosts y gateways en el sistema Internet. Los data gramas son enrutados desde un módulo Internet a otro a través de redes individuales basándose en la interpretación de una dirección Internet. Por eso, un mecanismo importante del protocolo Internet, es la dirección Internet.

En el enrutamiento de mensajes desde un módulo Internet a otro, los data gramas pueden necesitar atravesar una red cuyo tamaño máximo de paquete es menor que el tamaño del data grama. Para salvar esta dificultad se proporciona un mecanismo de fragmentación en el protocolo Internet.

- *DIRECCIONAMIENTO*

Se establece una distinción entre nombres, direcciones y rutas, un nombre indica qué buscamos, una dirección indica dónde está y una ruta indica cómo llegar allí. El protocolo Internet maneja principalmente direcciones. Es tarea de los protocolos de mayor nivel (es decir, protocolos host-a-host o entre aplicaciones) hacer corresponder nombres y direcciones. Y es tarea de los procedimientos de menor nivel (es decir, redes locales o gateways) realizar la correspondencia entre direcciones de red local y rutas.

Las direcciones son de una longitud fija de 4 octetos (32 bits). Una dirección comienza por un número de red, seguido de la dirección local (llamada el campo "resto"). Hay 3 formatos o clases de direcciones Internet:

Clase A. Su BIT más significativo es el 0, los 7 bits siguientes son la red, y los 24 bits restantes son la dirección local.

Clase B. Sus dos bits más significativos son uno-cero ("10"), los 14 bits siguientes son la red y los últimos 16 bits son la dirección local.

Clase C. Sus tres bits más significativos son uno-uno-cero ("110"), los 21 bits siguientes son la red y los 8 restantes son la dirección local.

Se debe tener cuidado al relacionar direcciones Internet con direcciones de red local; un host individual físicamente hablando debe ser capaz de actuar como si fuera varios hosts distintos, hasta el punto de usar varias direcciones Internet distintas. Algunos hosts tendrán también varios interfaces físicos (multi-homing).

Esto quiere decir que se debe establecer algún mecanismo que permita a un host tener varios interfaces físicos de red, cada uno de ellos con varias direcciones lógicas Internet.

- *FRAGMENTACIÓN*

La fragmentación de un data grama Internet es necesaria cuando éste se origina en una red local que permite un tamaño de paquete grande y debe atravesar una red local que limita los paquetes a un tamaño inferior para llegar a su destino.

Un data grama Internet puede ser marcado como "no fragmentar" y todo data grama Internet marcado así no será fragmentado entre distintas redes bajo ninguna circunstancia. Si un data grama Internet marcado como "no fragmentar" no puede ser entregado en su destino sin fragmentarlo, entonces debe ser descartado.

La fragmentación, transmisión y reensamblado a través de una red local invisible para el módulo del protocolo Internet se llama fragmentación intranet y puede ser utilizada.

El procedimiento de fragmentación y reensamblado en Internet tiene que ser capaz de dividir un data grama en un número casi arbitrario de piezas que puedan ser luego reensambladas. El receptor de los fragmentos utiliza el campo de identificación para asegurarse de que no se mezclan fragmentos de distintos data gramas. El campo offset le indica al receptor la posición de un fragmento en el data grama original. El offset y longitud del fragmento determinan la porción de data grama original comprendida en este fragmento. El indicador "más-fragmentos" indica (puesto a cero) el último fragmento. Estos campos proporcionan información suficiente para reensamblar data gramas.

El campo identificador se usa para distinguir los fragmentos de un data grama de los de otro. El módulo de protocolo origen de un data grama Internet establece el campo identificador a un valor que debe ser único para ese protocolo y par origen-destino durante el tiempo que el data grama estará activo en el sistema Internet. El módulo de protocolo de origen de un data grama completo pone el indicador "más-fragmentos" a cero y el offset del fragmento a cero.

Para ensamblar los fragmentos de un data grama Internet, un módulo de protocolo Internet (por ejemplo en un host destino) combina todos los data gramas Internet que tengan el mismo valor en los cuatro campos: identificación, origen, destino y protocolo. La combinación se realiza colocando el trozo de datos de cada fragmento en su posición relativa indicada por el offset del fragmento en la cabecera Internet de ese fragmento. El primer fragmento tendrá offset cero, y el último fragmento tendrá el indicador "más fragmentos" puesto a cero.

- *FORMATO DE LA CABECERA INTERNET*

A continuación vemos un resumen del contenido de la cabecera Internet:

Versión	IHL	Tipo de Servicio	Longitud total
Identificación	Flags	Posición	
Tiempo de Vida	Protocolo	Suma de control de cabecera	
Dirección Origen			
Dirección Destino			
Opciones		Relleno	

FIGURA 10 Ejemplo de Cabecera de un data grama Internet

Versión: 4 bits. El campo Versión describe el formato de la cabecera Internet. Este documento describe la versión 4.

IHL: 4 bits. Longitud de la cabecera Internet (Internet Header Length), es la longitud de la cabecera en palabras de 32 bits y por tanto apunta al comienzo de los datos. Nótese que el valor mínimo para una cabecera correcta es 5.

Tipo Servicio: 8 bits. El Tipo de Servicio proporciona una indicación de los parámetros abstractos de la calidad de servicio deseada. Estos parámetros se usarán para guiar la selección de los parámetros de servicio reales al transmitir un data grama a través de una red en particular.

Longitud Total: 16 bits. La Longitud Total es la longitud del data grama, medida en octetos, incluyendo la cabecera y los datos.

Identificación: 16 bits. Es un valor de identificación asignado por el remitente como ayuda en el ensamblado de fragmentos de un data grama.

Flags (indicadores): 3 bits. Son diversos indicadores de control.

Posición del Fragmento: 13 bits. Este campo indica a que parte del data grama pertenece este fragmento. La posición del fragmento se mide en unidades de 8 octetos (64 bits). El primer fragmento tiene posición 0.

Tiempo de Vida: 8 bits. Este campo indica el tiempo máximo que el data grama tiene permitido permanecer en el sistema Internet.

Protocolo: 8 bits. Este campo indica el protocolo del siguiente nivel usado en la parte de datos del data grama Internet.

Suma de Control de Cabecera: 16 bits. Suma de Control de la cabecera solamente. El campo suma de control es el complemento a uno de 16 bits de la suma de los complementos a uno de todas las palabras de 16 bits de la cabecera. A la hora de calcular la suma de control, el valor inicial de este campo es cero.

Dirección Origen: 32 bits. La dirección de origen.

Dirección Destino: 32 bits. La dirección de destino.

Opciones: variable. Las opciones pueden o no aparecer en los data gramas. Deben ser implementadas por todos los módulos IP (host y gateways). Lo que es opcional es su transmisión en cualquier data grama en particular, no su implementación.

Compartimentos (Campo C): 16 bits. Cuando la información no está compartimentada se usa un valor todo ceros. Otros valores para el campo Compartimentos pueden obtenerse de la Agencia de Inteligencia de Defensa.

Manejo de Restricciones (campo H): 16 bits. Los valores para los marcadores de control y liberación.

Código de Control de Transmisión (Campo TCC): 24 bits. Proporciona un mecanismo para segregar el tráfico y definir comunidades de interés controladas entre diversos suscriptores.

Valor de Relleno: variable. El Valor de Relleno se usa para asegurar que la cabecera Internet ocupa un múltiplo de 32 bits. El valor de relleno es cero. [28]

1.3 CALIDAD DEL HABLA

1.3.1 QUÉ ES CALIDAD DEL HABLA (VOICE SPEECH QUALITY - VSQ)

Existen muchos factores que influencian VSQ y muchos de estos factores pueden ser medidos, sin embargo, la calidad del habla tiene el mayor significado desde la perspectiva de los usuarios finales.

Después de todo, los usuarios finales son quienes deciden por cuales servicios de comunicación van a pagar y a quien se los van a comprar, es por esta perspectiva del cliente final que la calidad del habla es necesaria.

VSQ esta compuesta de un número de características incluyendo la “claridad” de la voz, efectos como el eco y caídas, tiempos de retardo entre las frases habladas etc. de todos estos parámetros, claridad y retardo de tiempo se consideran los más importantes. Otros parámetros como recorte de tiempo y sonoridad, son factores actuales que influencian la “claridad” del habla y son comúnmente reflejados en una clase de medida o evolución de claridad.

Otro parámetro clave es el eco, que afecta la calidad de percepción de un emisor. El eco es una función de retardos de la red y otras causas, en muchos casos sin

embargo, el oyente percibe toda la calidad del habla en términos de la claridad y retardo de tiempo.

Mientras el retardo de tiempo es un parámetro muy objetivo y puede ser medido efectivamente y expresarse en milisegundos, la claridad es un parámetro subjetivo, que se puede definir como la fidelidad de la voz como es reproducida por una red y percibida por el oyente. En resumen claridad es:

- Dependiente de diferentes tipos de distorsiones introducidas por varios elementos de una red.
- Independiente del retardo, es decir, la claridad del habla puede conservarse y ser transportada con cero, poco o mucho retardo a través de la red. Variaciones en el retardo (conocido como Jitter), puede sin embargo influenciar la claridad.
- Independiente del eco, desde que el eco es percibido por el emisor y la claridad percibida por el oyente.

Muchos tipos de distorsión en una red pueden alterar la claridad, esto incluye

- Codificación y decodificación de voz. Esto incluye la codificación de una onda con modulación por codificación de pulso (PCM) uniforme y no uniforme, modulación por codificación de pulso diferencial adaptativo (ADPCM) y codecs para compresión de voz de baja tasa de bits (BIT rate). Se introduce distorsión lineal y no lineal.
- Recorte de tiempo (conocido como FEC) introducido por los detectores de actividad de voz.
- Pérdida temporal de señal y caídas introducidas por pérdida de paquetes.
- Variación de retardo (Jitter).
- Ruido ambiental, incluyendo ruido de fondo.
- Varianza en la atenuación y/o ganancia de la señal.

- Nivel de recorte.
- Errores de transmisión en el canal.

Mientras que se ha definido VSQ como una colección de parámetros, es la claridad la que más se usa considerando la calidad de la voz.

- *CLARIDAD DEL HABLA EN REDES CONVENCIONALES*

Los estándares aceptables para la claridad del habla en una llamada telefónica han sido definidos por un siglo por la red pública telefónica conmutada (PSTN). Los diseñadores originales de la PSTN recorrieron grandes distancias, para encontrar el balance óptimo entre un nivel aceptable de calidad y costo. A mayor calidad del habla mayor costo. Muchos de los componentes y características de la PSTN que fueron intencionalmente diseñados para asegurar ese nivel de calidad incluyen:

- Filtrado del auricular del teléfono.
- Muestreo digital y la sobre modulación a la tasa Nyquist para la voz. (8kHz).
- Codificación PCM no uniforme con Ley μ y ley A.
- Garantía de un ancho de banda de 56/64 Kbps.
- Planes de pérdida de retorno de eco (para minimizar o eliminar el eco).

Como resultado de estas decisiones de diseño, los niveles de calidad de la PSTN (particularmente la claridad) son bastante predecibles y confiables. La claridad del habla en la PSTN es aceptable aunque lejos de la perfección. Existen muchos deterioros de la claridad en la PSTN, esto incluye:

- Filtrado análogo y atenuación en el auricular de un teléfono.
- Filtrado, atenuación y posible interferencia electromagnética (EMI) en las líneas de transmisión análogas.

- Codificación PCM, codificadores de onda en particular PCM 64kbps (G. 711), introduce distorsión de cuantización que tiene un impacto mínimo en la claridad del habla. Este impacto es absorbido por estándares de claridad esperados en la PSTN ampliamente aceptados. En realidad, la codificación G.711 sobre una red grande mejora la claridad eliminando deterioros introducidos por transmisiones largas hechas sobre líneas análogas.

- Errores de bit debido a ruido en el canal.

- Eco debido a múltiples empalmes de cables híbridos en el camino de una llamada.

En general, estos deterioros son identificados fácilmente o sus efectos son aceptados por los estándares de claridad de la PSTN.

- *CLARIDAD DEL HABLA EN LAS REDES DE NUEVA GENERACIÓN*

El despliegue de las redes convergentes hoy en día y en los próximos años, introduce nuevos retos de mantener estándares aceptables de la claridad del habla, porque las nuevas tecnologías se están usando para liberar servicios de voz, ahora están presentes nuevos deterioros en las redes de voz. Muchos de estos deterioros son comunes en VoIP, ATM, Frame relay y redes inalámbricas.

Muchas llamadas atraviesan redes híbridas incluyendo PSTN-VoIP-PSTN o inalámbrica PSTN. Desde ahora los deterioros tradicionales introducidos por la PSTN se aumentan y a veces se destacan por los deterioros introducidos por las redes de nueva generación. Hasta ahora, la claridad del habla tenía menos posibilidades de ser degradada, pero con las redes de nueva generación de varias maneras introducen deterioros a la claridad del habla. Algunos ejemplos claves son:

- Baja tasa de codificación por bits de la voz. Los codecs introducen una distorsión y filtrado no lineal, retardo y baja alguna condiciones puede introducir pérdidas severas y fragmentación de audio.

- Supresión de silencio. El uso de detectores de actividad de voz introduce recorte front-end.
- Perdida de paquetes que introduce caídas y recortes de tiempo.
- Jitter que puede dar como resultado “deformaciones de audio” en donde ese habla no esta libre en un flujo constante. Aunque algunos dispositivos de VoIP tienen Jitter buffers para cancelar el Jitter, la reproducción a la tasa de audio no siempre es perfecta.
- Retardo de paquetes. Mientras que el retardo de paquetes no afecta directamente a la claridad, un incremento del retardo de paquetes puede aumentar la pérdida y el Jitter. Aunque el eco no afecta la claridad del emisor percibida por un oyente, si tiene impacto sobre la calidad del habla, un parámetro clave de la evaluación subjetiva. Percibir eco se hace peor por el retardo excesivo introducido por el proceso y transporte de VoIP.

- *LAS MÉTRICAS TRADICIONALES NO SON DEL TODO ADECUADAS*

Las métricas tradicionales usadas para evaluar la calidad de redes conmutadas convencionales son más precisas cuando se aplican a sistemas lineales e invariables con el tiempo (LTI). Algunas métricas pueden ser adaptadas a sistemas que no sean LTI, estimando esos sistemas como LTI en segmentos muy pequeños; sin embargo, estas métricas están limitadas para detectar deterioros debidos a la transmisión análoga y a la codificación de la onda. Antes de describir porque las métricas tradicionales son menos útiles en redes convergentes no LTI, es útil una explicación de linealidad y variación en el tiempo.

Linealidad y variación en el tiempo. LTI es una característica reconocida a menudo en circuitos de audio y canales que describen, de manera general, como los circuitos de audio o los canales actúan cuando procesan una señal de entrada. Un sistema o red el lineal si cumple las propiedades de aditividad y homogenidad.

Propiedad Aditiva. La respuesta resulta de la entrada x (t)+ (yt) es igual a la respuesta de x (t) más la respuesta de y (t). Es decir, la función de salida [F(x+y)] (t) = Fx (t) + Fy (t), la señal de salida de una entrada combinada de la señal x y de la señal y es igual que la suma de las señales de salida entradas individuales sin combinar.

Propiedad Homogénea. La respuesta resulta de la entrada a [x (t)] y es igual a la respuesta de x (t) como entrada, multiplicada por a que es un valor escalar. Es decir, la función de salida [F (ax)] (t) = a (Fx) (t), una señal multiplicada por una constante a es la entrada de un sistema, produciría la misma salida, si la señal fuera la entrada (sin multiplicar por a) y después multiplicar la salida por a.

Las propiedades de linealidad se pueden resumir así:

$$\left[F(ax+bx)\right](t) = a(Fx)(t) + b(Fy)(t)$$ **Ecuación 1**

Un sistema o una red es invariante en el tiempo si para cualquier x (t -t0), la respuesta es y (t - t0). Es decir, la forma de onda de la respuesta es independiente del retardo. Una variación en el retardo de una señal de puede causar que una señal invariante se convierta en variante en el tiempo. Así, incluso al usar técnicas de codificación lineales, el retardo de Jitter introducida por una red no determinística, (tal como las redes de paquetes) puede producir un sistema variante en el tiempo.

- *BREVE DESCRIPCIÓN DE LAS MÉTRICAS TRADICIONALES DE CLARIDAD*

Las métricas usadas tradicionalmente para evaluar sistemas de LTI se describen a continuación.

Relación Señal a Ruido. La relación señal a ruido (SNR) se utiliza para medir niveles de ruidos relativos en señales análogas y distorsión de cuantización introducida por los codificadores PCM. SNR puede también proporcionar una

medida exacta de los efectos que los errores de bit tienen en una señal reproducida.

Para los sistemas con canales con ruido aditivo, la señal recibida es la suma de la señal transmitida más ruido, o

$$r(t) = s(t) + n(t)$$ **Ecuación 2**

para los sistemas con filtración pasa banda, éstas son funciones complejas que representan las envolventes complejas de las señales respectivas.

Para los sistemas de codificación lineal digital (por ejemplo PCM uniforme) con posibles errores de bit, la salida SNR del promedio de la señal respecto al promedio del ruido se expresa como:

$$\left(\frac{S}{N}\right)_{OUT} = \frac{M^2}{1+4(M^2-1)P_e}$$ **Ecuación 3**

Donde: hay M pasos de cuantización en el cuantizador uniforme. Pe es la probabilidad de error de BIT y la relación del promedio de la energía de la señal debido a la distorsión del cuantizador lineal es:

$$\left(\frac{S}{N}\right)_{OUT} = M^2$$ **Ecuación 4**

Para los cuantizadores no uniformes (ley μ y ley A), se utiliza una expresión más complicada logarítmica, la salida SNR es relativamente constante y deteriora rápidamente la señal de entrada que disminuyendo los niveles en sistemas PCM uniformes. Las pausas en el habla pueden dar lugar a SNR falso; por lo tanto, se utiliza un SNR segmentario en el cual SNR se obtiene solamente para los intervalos del habla y no para los períodos silenciosos entre las elocuciones. SNR es útil solamente cuando el proceso de codificación mantiene generalmente formas de onda de la entrada en la salida. En los casos donde se utilizan los codecs bajos y la compresión de índice binario, sin embargo, se ha encontrado que SNR y los resultados segmentarios de la medida de SNR demuestran poca

correlación a la claridad percibida del habla. Ésta es una de las razones por la que son necesarios los nuevos algoritmos perceptivos de claridad.

- *DISTORSIÓN TOTAL ARMÓNICA Y DISTORSIÓN POR INTERMODULACIÓN*

Las medidas de la Distorsión Total Armónica (THD) y de la Distorsión por intermodulación (IMD), son técnicas usadas para evaluar la distorsión no lineal introducida por los procesadores de la señal tales como amplificadores. THD se determina introduciendo un solo tono sumando las relaciones de la energía para las salidas de mayor orden. IMD se determina de una entrada de tono dual.

Mientras que estas métricas son útiles para medir la distorsión no lineal para entradas de tono, no reflejan adecuadamente la calidad de una señal que ha sido procesada por codecs sin forma de onda y redes de paquetes. Una vez más otra razón por la que la nueva métrica de claridad se debe emplear en redes convergentes que emergen.

- *ERROR DE TASA DE BIT*

El error de tasa de BIT BER, es una medida muy eficaz de calidad en redes de transmisión físicas; sin embargo, son obviamente insuficientes para los sistemas no LTI. Por ejemplo, la configuración de bits puede ser muy errónea por un codec de baja velocidad de transmisión de bites con un impacto insignificante en la calidad subjetiva del habla. Esto es porque los codecs de bajo índice binario se diseñan para codificar y para descifrar las características audio que son importantes para la opinión humana y no necesariamente las configuraciones exactas de bits. Así, la métrica de claridad que hace su trabajo basado en la opinión humana debe ser utilizada cuando los sistemas no LTI son parte de la trayectoria de la telefonía.

- *LAS REDES MODERNAS REQUIEREN NUEVAS MÉTRICAS*

Las redes de la siguiente generación utilizan muchas tecnologías como codecs de baja tasa de bits y transmisión de paquetes que no son LTI. Como se ha descrito previamente, para estas tecnologías muchas métricas tradicionales de calidad son insuficientes. Cuando la trayectoria de transmisión no es LTI las medidas objetivas simples, por ejemplo las especificadas en la recomendación G.712 para las características de funcionamiento de los sistemas PCM, no son adecuadas. Además, incluso esas métricas tradicionales que se pueden aplicar a los sistemas no LTI no predicen adecuadamente la opinión de claridad de una persona, por ejemplo, SNR y THD segmentarios no explicarán la capacidad de una persona de adaptarse a los componentes que falta de la energía de tiempo-frecuencia como resultado de la codificación de voz. Los resultados de métricas tradicionales que pudieron indicar, características pobres de claridad del habla en la señal de salida, a veces no son realmente perceptibles por el oyente humano, por ejemplo, variaciones lentas o locales en aumento o la atenuación, las variaciones lentas de retardo, truncamiento corto del tiempo, una cierta filtración y especialmente codificación del sin forma de onda que puede deformar la forma de onda (dando por resultado SNR o BER muy pobres, por ejemplo), son todas las características de salida que probablemente no causarían al oyente ningún malestar. Estas características de salida, en muchos casos, son detectadas por métricas tradicionales. Por otra parte, algunos problemas perceptibles de claridad pueden no detectarse realmente con métricas tradicionales.

Comenzando en los últimos años 80 y a través de los 90 se fueron desplegando nuevas tecnologías, la industria reconoció la necesidad de nuevas técnicas de medición que representen exactamente claridad de la misma manera que los seres humanos la perciben. [23]

1.3.2 PUNTAJE DE OPINIÓN (MEAN OPINION SCORES MOS)

La primera técnica significativa usada para medir claridad del habla estaba basada en encuestas de una gran cantidad de oyentes humanos para producir puntajes de claridad estadísticos subjetivos. Esta técnica se conoce como puntaje opinión (MOS) donde se calcula el valor medio de muchos puntajes de opinión humana. Las técnicas para probar redes con MOS se describen en la Recomendación P.800 de la ITU. La recomendación P.830 proporciona métodos más específicos para la prueba subjetiva de codecs del habla. Estas dos recomendaciones de la ITU, describen métodos para probar, métodos para obtener puntajes subjetivos, valores de los puntajes, características de las muestras de habla que se utilizarán y otras condiciones bajo las cuales la prueba debe ser realizada.

La prueba MOS se puede basar en pruebas conversacionales de dos vías o en pruebas de una sola vía de escucha, en donde se usan muestras de habla estandardizadas. Los oyentes oyen las muestras transmitidas sobre un sistema o una red, y clasifican la calidad total de la muestra, de acuerdo con las escalas de opinión.

P.800 especifica varios tipos de prueba subjetiva:

- Prueba de opinión de conversación
- Prueba de clasificación absoluta de categoría (Absolute Category Rating - ACR)
- Prueba Quantal-Response Detectability
- Método de de clasificación de Degradación de Categoría (DCR)
- Método de de clasificación de comparación de Categoría (CCR)

Cada uno de estos métodos define sus escalas de opinión correspondientes. Por ejemplo las pruebas de Opinión de conversación y ACR tienen escalas similares, denominadas Puntaje de Opinión de Conversación y Puntaje de Calidad de Escucha respectivamente. La prueba de Opinión de Conversación pregunta a los

oyentes su opinión acerca de la conexión que acaban de utilizar. Las pruebas del ACR piden que los oyentes clasifiquen la calidad del habla. Los puntajes para estas dos escalas son:

Puntaje	Calidad del Habla
5	Excelente
4	Bueno
3	Aceptable
2	Pobre
1	Malo

Tabla 8 Escala MOS.

Esta escala 1-5 es la que se utiliza más a menudo para la prueba MOS, otro ejemplo común es la escala ACR en la que se pide a los oyentes clasificar el esfuerzo requerido para entender el significado de oraciones, como se muestra a continuación:

Puntaje	Esfuerzo requerido para entender el significado de oraciones
5	Completa relajación posible. No requiere esfuerzo.
4	Se necesita atención. No se requiere un esfuerzo apreciable.
3	Se requiere un esfuerzo moderado
2	Se requiere un esfuerzo considerable
1	No es posible entender el significado con cualquier esfuerzo posible.

Tabla 9 Escala ACR.

Obviamente, la prueba MOS tiene varias desventajas:

- Es subjetiva porque los resultados dependen de muchas cualidades incontrolables de los oyentes que hacen la prueba incluyendo humor, actitud y cultura. Prácticamente hablando, la prueba MOS no es un método repetible o constante para probar.

- Es costosa porque requiere a una gran cantidad de gente y elabora las disposiciones de prueba.
- Es ineficaz e impráctico realizar la prueba tan frecuente como se necesita hacer cambios de diseño y configuración en una red y para la supervisión rutinaria de la misma.

Las desventajas de prueba del MOS sugieren que los métodos de prueba objetivos, automatizados, y repetibles son necesarios para medir la claridad subjetiva del habla.

1.3.3 ANTECEDENTES

- *MEDIDA PERCEPTIVA DE LA CALIDAD DEL HABLA - PSQM*

Para justificar a la necesidad de una prueban con un método objetivo, automatizado y repetible de la claridad del habla que considera la naturaleza subjetiva y la opinión humana de la claridad, se desarrolló una técnica por Juan G. Beerends y J. A. Stemerdink de la investigación de KPN en los Países Bajos. Fue llamada la medida perceptiva de calidad del habla PSQM por sus siglas en inglés.

A partir de 1993-1996, se investigaron varios métodos para la medida objetiva de la claridad del habla, incluyendo PSQM, por parte de NTT y la ITU. Estas organizaciones compararon los resultados de estos métodos para determinar cuáles dieron las estimaciones más exactas de la claridad subjetiva. La ITU concluyó que PSQM correlacionó lo mejor posible los resultados de la prueba subjetiva (tales como los producidos por MOS).

PSQM fue aprobado por el grupo de estudio de ITU-T 12 y publicado posteriormente por ITU como recomendación P.861 en 1996, desde donde tiene gran aceptación como medida constante y exacta de la claridad del habla basada en factores de opinión humana.

Visualización del proceso de PSQM. PSQM es un proceso matemático que proporciona una medida subjetiva de la calidad del habla. El objetivo de PSQM es producir puntajes que predicen confiablemente los resultados de pruebas subjetivas, particularmente esos métodos que describe P.830 (MOS). Sin embargo, los puntajes de PSQM, están en una escala diferente y reflejan una diferencia de la medida perceptiva, es decir, los puntajes de PSQM reflejan la cantidad de divergencia de la señal original respecto a otra distorsionada después de haber sido procesada por un sistema telefónico.

PSQM esta diseñado para ser aplicado a señales de banda telefónica (300-3400 Hz) procesadas por codecs y medir la distorsión introducida por los codecs según factores de opinión humana. Es particularmente útil para medir claridad del habla cuando se utilizan codecs o vocoders de baja velocidad de transmisión bits en la compresión de la voz.

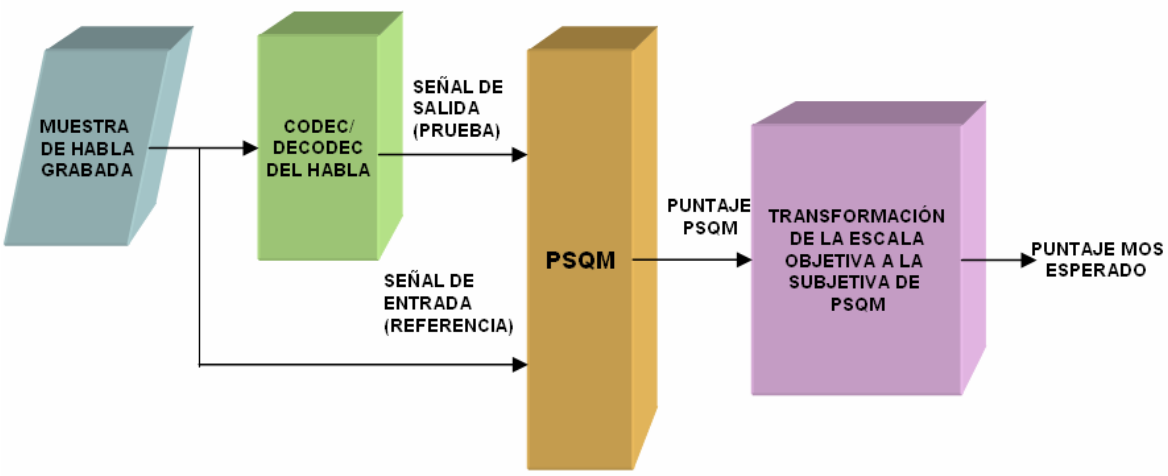

FIGURA 11 Proceso de prueba de PSQM.

Para realizar una medida PSQM, se introduce y procesa una muestra del habla humana grabada, en un sistema por cualquier codec; las características de la señal de entrada son las mismas utilizadas para la prueba MOS y están especificadas en ITU P.830.

Las señales de entrada pueden ser habla humano verdadero o habla artificial de acuerdo a la recomendación P.50 de la ITU.

La ITU-T recomienda que las señales de entrada estén filtradas según el IRS modificado (sistema intermedio de referencia especificado en ITU P.48) que recibe las características definidas el anexo D/P.830 ya que esto simula las características de la frecuencia de recepción del auricular del teléfono.

La señal de salida es grabada mientras que se recibe, entonces se sincroniza en tiempo con la señal de entrada y son comparadas por el algoritmo PSQM. Esta comparación se realiza en segmentos individuales del tiempo (marcos) en el dominio de la frecuencia (conocido como componentes tiempo-frecuencia), actuando sobre las componentes de los parámetros tiempo-frecuencia derivados de las densidades espectrales de energía de la entrada y de la salida. La comparación se basa en factores de opinión del ser humano, por ejemplo sensibilidad de la frecuencia y de la intensidad.

Los puntajes que resultan con PSQM en el rango de cero a infinito, representan la distancia perceptiva entre las señales de entrada y salida. Por ejemplo, un puntaje resultante de 0 indica que no hay diferencia entre la entrada y la salida, interpretado a menudo como claridad perfecta. Puntajes más altos de PSQM, indican niveles de aumento en la distorsión, a menudo interpretado como baja claridad. En la práctica, los límites superiores de PSQM están alrededor de15-20.

- *MEDIDA DE BLOQUES NORMALIZADOS (MNB)*

En 1997, basado en un informe de Stephen D. Voran del instituto para Ciencias De las Telecomunicaciones, se publicó como anexo propuesto P.861 (PSQM), este anexo fue aceptado en 1998 como apéndice II de P.861, describe una técnica alternativa a PSQM para medir la distancia perceptiva entre las representaciones internas de las señales de entrada y salida, esta técnica se conoce como medida de bloques Normalizados (MNB).

Visualización del proceso MNB. Se observó que los "oyentes se adaptan y reaccionan de formas distintas a desviaciones espectrales que atraviesan diversas escalas de tiempo y frecuencia", MNBs aprovecha esto analizando la distancia perceptiva a través escalas más pequeñas de tiempo y frecuencia. La técnica de MNB está recomendada para medir el impacto de la claridad del habla con las siguientes características:

- Errores de transmisión del canal
- CELP y codecs híbridos con tasas de bit menores a < 4 KBs
- Vocoders

Hay dos tipos de MNB: Medida de Bloques Normalizados en tiempo y Medida de Bloques Normalizados en Frecuencia. El algoritmo crea uno solo, el valor no negativo llamado Distancia Auditiva (AD), el cuál es una medida de distancia perceptiva entre las señales de entrada (referencia) y la de salida (prueba) en pro de predecir la calidad subjetiva.

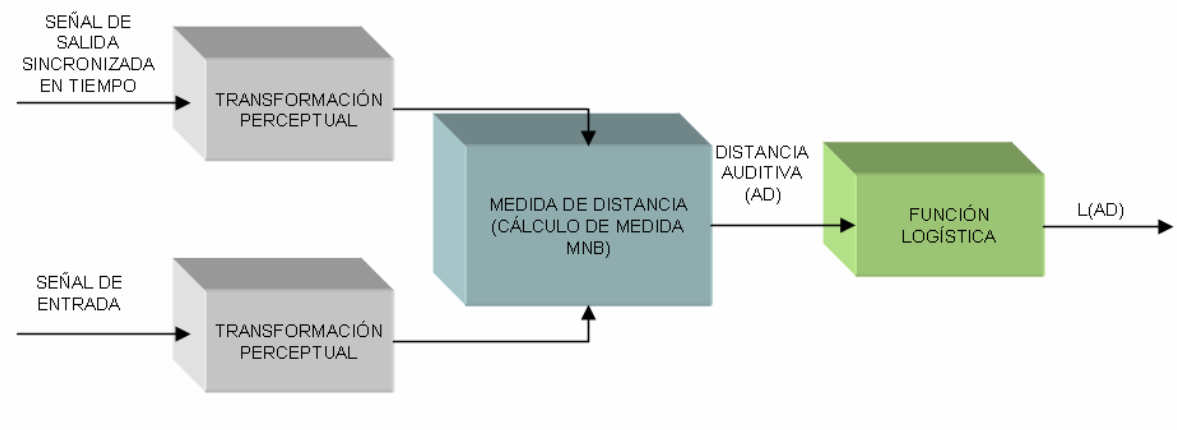

FIGURA 12 Diagrama de Bloques del Proceso de MNB.

- *PSQM+*

Debido al rápido despliegue de las redes de nueva generación se dio una fuerte necesidad para probar la calidad del habla (particularmente claridad del habla),

con lo que PSQM ganó gran aceptación. PSQM llegó a ser popular para medir la claridad de la voz no solamente a través de vocoders sino a través de redes enteras, incluyendo VoIP. Una desventaja de PSQM, es eso que no muestra exactamente el impacto de la distorsión cuando es causada por pérdida de paquetes u otros tipos truncamiento de tiempo. Es decir PSQM mostraría una mejor claridad bajo de éstas condiciones que un oyente humano.

Se reconoció la necesidad tener en cuenta estas degradaciones en la transmisión de la red, por lo que J. G. Beerends, E. J. Meijer, y A. P. Hekstra, desarrollaron una mejora al modelo de PSQM este nuevo modelo fue repasado por el grupo de estudio ITU 12 y publicado en 1997 como PSQM+ y se convirtió rápidamente en el método preferido para medir la calidad del habla capaz de medir en ambientes de red. PSQM+ (se basa directamente en el modelo de ITU P.861 PSQM y representa solamente una leve desviación) mejora la técnica de PSQM que es aplicada a un sistema con distorsiones severas debido al truncamiento en tiempo y a la pérdida de paquetes. Para sistemas que codifican el habla solamente, PSQM y PSQM+ dan el mismo puntaje.

Descripción de la mejora de PSQM+. PSQM trata la distorsión por ganancia de señal, representada dentro de una celda tiempo-frecuencia, diferente a la distorsión por pérdida de señal, porque la distorsión aditiva de la señal (energía agregada) tiene un impacto más grande en claridad percibida, que la distorsión substractiva (falta energía). PSQM escala encima del disturbio aditivo para dar lugar a puntajes más altos y reducen el disturbio substractivo para dar lugar a puntajes más bajas de PSQM. Para distorsiones pequeñas, que probablemente son debidas a un codec, PSQM proporciona una excelente correlación con los resultados de las pruebas subjetivas. Para distorsiones grandes, debido al truncamiento de tiempo y pérdida de paquetes, en la cual todas las células dentro de un marco experimentan pérdidas severas de energía de señal, PSQM produce puntajes bajos, muy lejanos, en comparación con los resultados de las pruebas subjetivas.

Para explicar este problema en el proceso asimétrico de PSQM y para hacer que PSQM+ correlacionara de forma más precisa con los resultados de la prueba subjetiva que PSQM, PSQM+ agrega un segundo factor de posicionamiento, que contradice el factor de posicionamiento de PSQM, bajo condiciones de distorsiones severas representadas por grandes pérdidas de energía en una célula de tiempo-frecuencia (este nuevo factor se aplica a cada marco). Cuando la señal de energía de entrada y salida son casi iguales en un marco, el nuevo factor está alrededor 1.0. por lo tanto PSQM+ produce casi el mismo puntaje que PSQM.

Cuando una gran distorsión como la ocasionada por truncamiento de tiempo o pérdida de paquetes se introduce (haciendo el algoritmo original de PSQM reducir el disturbio del ruido), el algoritmo de PSQM+ aplica otro factor de posicionamiento que tiene un efecto opuesto y escala encima del disturbio del ruido, esto da lugar a puntajes más altos de PSQM+, los cuales se correlacionan mejor con los resultados de las pruebas subjetivas.

En PSQM+, el segundo factor de posicionamiento se aplica siempre, pero cuando las distorsiones son pequeñas, se iguala a 1, teniendo así poco o ningún impacto. A medida que las distorsiones aumentan, se contradice el primer factor de posicionamiento y para distorsiones ruidosas (energía agregada), el segundo factor de posicionamiento obliga a bajar el puntaje y para la pérdida severa (energía que falta), obliga a puntajes más altos. Para distorsiones pequeñas debido a los codecs, PSQM y PSQM+ producen puntajes casi iguales, los cuáles correlacionan bien con las pruebas subjetivas. Para distorsiones por pérdida severa y truncamiento de tiempo, PSQM+ producirá puntajes más altos que correlacionan mejor que PSQM y para distorsiones ruidosas, PSQM+ producirá puntajes más bajos que correlacionan mejor que PSQM.

- *PAMS (PERCEPTUAL ANALYSIS MEASUREMENT SYSTEM)*

Para tratar el problema de medir objetivamente la claridad subjetiva del habla, se emprendió un esfuerzo independiente por parte de la KPN (donde fue desarrollado

PSQM), el resultado fue una técnica llamada Sistema de Medida de Análisis perceptivo o PAMS. PAMS fue publicado en agosto de 1998, y lanzado con la versión 3.0 en febrero del 2000. PAMS ofrece un modelo diferente a PSQM+ pero con la misma meta: predecir de forma objetiva los resultados de las pruebas subjetivas de la calidad del habla para los sistemas con distorsiones en la codificación debidas al truncamiento de tiempo y pérdida de paquetes. PAMS primero ganó gran aceptación en Europa, y a medida que se fue experimentando aumentó su uso en Norteamérica.

Descripción de PAMS. PAMS utiliza un modelo basado en factores de opinión humana para medir la claridad del habla en una señal de salida con respecto a la señal de entrada. Aunque es similar a PSQM en muchos aspectos, PAMS utiliza diferentes técnicas de procesamiento de señales y un modelo perceptivo distinto.

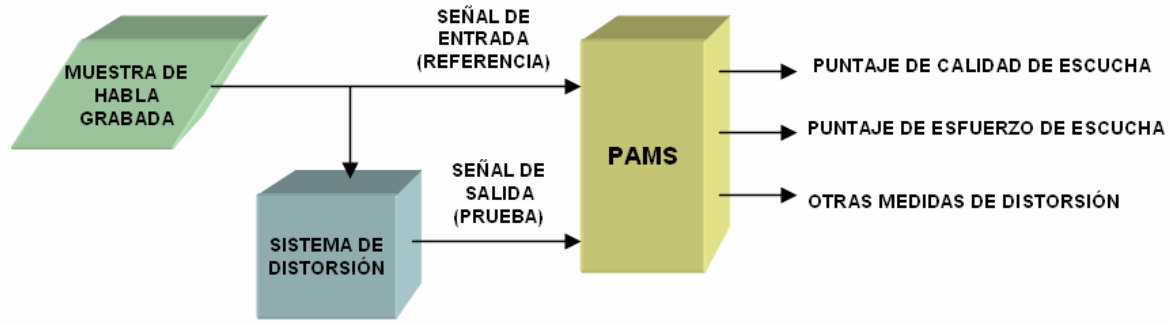

FIGURA 13 Proceso de Prueba PAMS.

Para realizar una medida con PAMS, se introduce una muestra del habla humana grabada, en un sistema o una red. Las características de la señal de entrada son las mismas que se utilizan para MOS especificados en P.830.

La señal de salida se graba mientras es recibida, luego las señales de entrada y salida se introducen al modelo de PAMS el cual realiza la alineación de tiempo, nivel alineación, igualación para quitar los efectos de retardo, ganancia total del sistema y filtración análoga en el teléfono. La alineación de tiempo se realiza en segmentos de tiempo, de modo que los efectos negativos de grandes variaciones

de retardo sean eliminados. Sin embargo, los efectos perceptibles de variación de retardo se preservan y reflejan en los puntajes de PAMS.

PAMS entonces compara la señal de entrada y salida en el dominio tiempo-frecuencia, comparando las células tiempo-frecuencia dentro de marcos de tiempo, esta comparación se basa en factores de opinión humana. Observe que la definición de la célula de tiempo-frecuencia y el marco de tiempo es similar a los usados en PSQM.

Los resultados de las comparaciones de PAMS son puntajes en un rango entre 0 – 5 y están correlacionados con la misma escala MOS. En particular, PAMS produce un puntaje de calidad de escucha y un puntaje de esfuerzo de entendimiento que corresponde a las dos escalas ACR en P. 800 y la escala de opinión en P. 830.

Muchos otros parámetros de distorsión son medidos por PAMS, incluyendo el cálculo de error de superficie.

1.3.4 EVALUACIÓN PERCEPTIVA DE LA CALIDAD DEL HABLA - PESQ

Con el fin de correlacionar mejor las pruebas subjetivas, PSQM fue mejorado de nuevo mediante la investigación de KPN en 1999, esta mejora se conoce como PSQM99. En el período de estudio a partir de 1998-2000, la ITU publicó cinco correcciones sobre los borradores para las nuevas medidas perceptivas de la calidad del habla. Se incluyeron en esta revisión PSQM99 y PAMS, que demostraron ser pruebas bien correlacionadas con las subjetivas. Se determinó que cada uno tenía características significativas y que sería beneficioso para la industria, combinar las características de cada una en una nueva técnica de medición. Un borrador de colaboración, fue sometido a la ITU en mayo de 2000 por Juan G. Beerends y a Andries P. Hekstra de la investigación de KPN, y por Anthony W. Rix y Mike Hollier de las telecomunicaciones británicas. El nuevo algoritmo perceptivo de la claridad del habla se llamaría evaluación perceptiva de

la calidad del habla (PESQ), y está actualmente en la recomendación P.862 del 2001.

PESQ es un método de evaluación objetiva de la calidad del habla, que se realiza a través del computador, simulando un oído humano, no solo aplicable a codecs de voz sino a medidas de extremo a extremo.

Se debe aclarar que PESQ, al ser un modelo de evaluación objetivo, presenta ciertas condiciones en las que se debe usar para obtener predicciones acertadas, por ejemplo, se recomienda su uso para la evaluación de la calidad del habla de banda angosta 3.1kHz y también de codecs de habla de banda angosta. PESQ no presenta una evaluación detallada de la calidad de transmisión, sino que simplemente mide los efectos de la distorsión del habla, y el ruido sobre la señal del canal, en un sentido. Otros efectos como eco, retardo etc., no se ven reflejados en los puntajes obtenidos por PESQ, por eso es posible obtener altos puntajes con PESQ y tener una baja calidad de conexión. [22]

El objetivo de PESQ es comparar la señal original X(t) con ésta misma degradada por el canal Y(t), para obtener la predicción objetiva de la calidad del habla percibida por una persona, como si fuera una prueba subjetiva de escucha.[22]

PESQ está estandarizado por la Recomendación ITU-T P.862, en la cual describe el procedimiento para procesar las señales, con el fin de obtener un resultado de la escala MOS. Éste procedimiento se divide en tres módulos a saber: pre-procesamiento de señales, modelamiento perceptivo, modelamiento cognitivo. [22] [23]

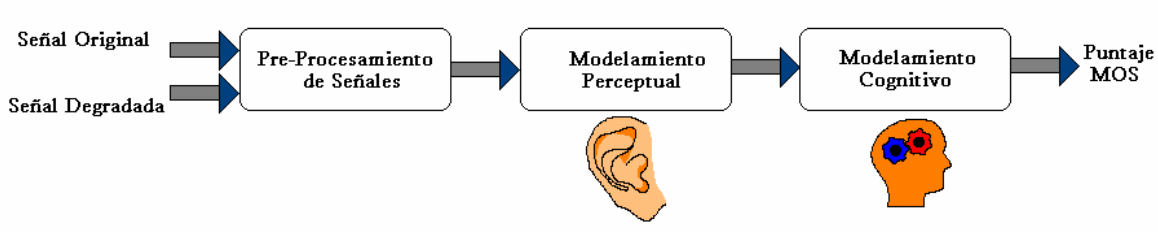

FIGURA 14 Diagrama en bloques de PESQ.

- *DESCRIPCIÓN DE PESQ*

Como PSQM y PAMS, PESQ también fue diseñado para señales telefónicas de banda angosta. Esto es aplicable a sistemas con codificación de habla (incluyendo vocoders de baja tasa de bits), retardo variable, filtrado, pérdida de paquetes o de celdas, truncamiento de tiempo y errores de canal. Los puntajes PESQ predicen los puntajes de las pruebas ACR de calidad de escucha.

PESQ reúne las mejores características de PAMS y PSQM99, combinando las robustas técnicas de alineamiento de tiempo de PAMS con el preciso modelo perceptivo de PSQM99, y adiciona nuevos métodos incluyendo la función de transferencia de ecualización y el nuevo método de promedio de distorsión en el tiempo.

La metodología de alineamiento de tiempo de PAMS suprime los efectos de retardo y varianzas lentas de retardo mientras que preserva los efectos de deterioros perceptibles debidos a variaciones rápidas de retardo. El modelo perceptivo de PSQM99 difiere del de PSQM y del de PSQM+ en el proceso de asimetría y escalamiento. PSQM99 presenta una correlación más precisa con las pruebas subjetivas que PSQM y PSQM+.

- *SUPOSICIONES Y FACTORES DE PESQ*

PESQ es efectivo para medir hasta el mínimo impacto de las siguientes condiciones o procesos en la claridad subjetiva del habla:

- Codecs de forma de onda.

- Codecs sin forma de onda (4kb/s), incluyendo aquellos con tasa de bits múltiples
- Transcodificaciones (conversión de un formato digital a otro)
- Niveles de entrada del habla a un codec
- Errores de transmisión en el canal
- Ruido adicionado por el sistema (que no está presente en la señal de entrada)
- Deformaciones largas y cortas de tiempo
- Pérdida de paquetes y truncamiento de tiempo (PESQ es más sensible al truncamiento punto a punto y menos sensitivo al truncamiento de tiempo que las pruebas subjetivas, la correlación varía)

La precisión de PESQ no es conocida o PESQ no pretende medir los impactos de los siguientes parámetros:

- Retardo (es cancelado por la alineación en tiempo)
- Niveles de escucha y ganancia o atenuación sobre todo el sistema (cancelado por alineamiento de niveles)
- Dependencia de la persona que habla, locutor (características vocales como el intervalo del tono, el idioma, el género, la edad etc.)
- Locutores múltiples y simultáneos.
- Ruido de fondo presente en la señal de entrada
- Señales de habla artificiales como señales de entrada
- Música como señal de entrada
- Codecs <4kbps
- Truncamiento de nivel
- Eco en el oyente

- *PROCESO DETALLADO DE PESQ*

La siguiente sección describe el pre-proceso y análisis realizado por el algoritmo de PESQ.

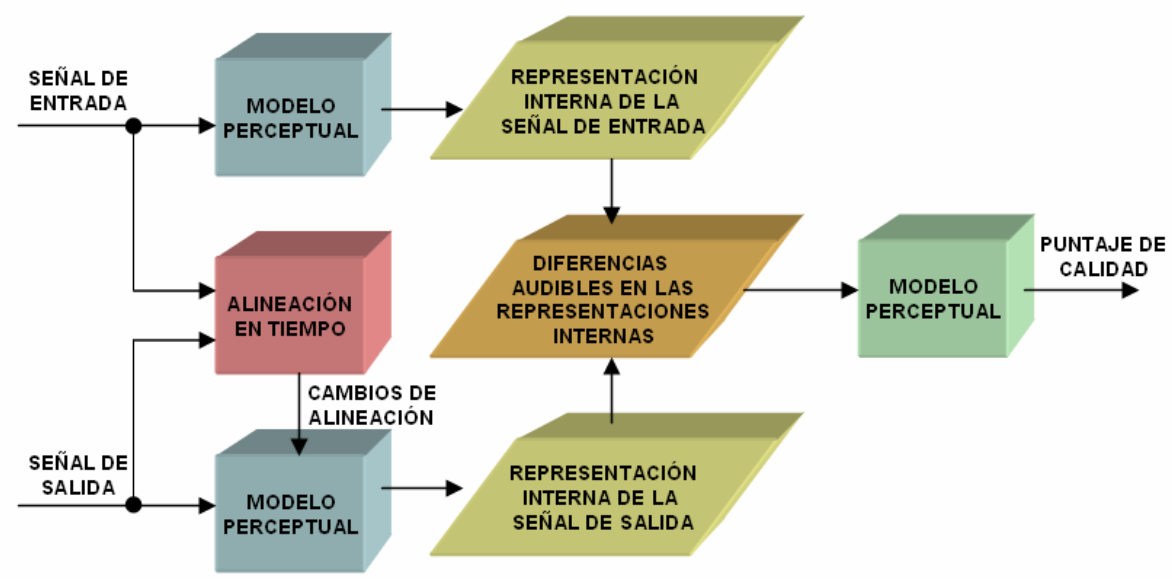

FIGURA 15 Diagrama de Bloques del Proceso de PESQ.

Las características de la señal de entrada para PESQ son consistentes con las de PSQM y PAMS, se puede usar muestreo de habla natural y se pueden seguir los lineamientos de P.830, incluyendo las especificaciones de niveles de escucha y filtrado. Se puede usar habla artificial pero representando estructuras fonéticas de habla natural. Las señales deben ser almacenadas con muestreos de 8kHz o 16kHz, con codificación lineal PCM de 16 bits. Describiendo la figura 13, los pasos del proceso de PESQ son:

Paso 1. Pre-procesamiento de señal:

Antes de analizar las señales con el modelo perceptivo se incluyen las siguientes operaciones:

- Alineación de nivel. Las señales de entrada y salida son alineadas en nivel al mismo valor de energía constante para tener en cuenta la ganancia o atenuación total del sistema y los cambios lentos en ganancia o atenuación. Primero se filtran las señales, luego son computados los valores promedio de energía y finalmente se aplica una ganancia para alinear las dos señales.

- Alineación de Tiempo. A la señal de salida se le aplica un corrimiento en tiempo para alinearla con la señal de entrada. Esto es ejecutado en segmentos individuales de tiempo, o articulaciones de habla de longitud variable. De hecho, muchos cálculos de retardo son ejecutados dentro de una articulación y una articulación puede determinar si hay grandes variaciones en los cálculos. Los retardos durante el habla y el silencio son contados para ejecutar el alineamiento de tiempo.

Paso 2. Modelo Perceptivo

El modelo perceptivo transforma las señales de entrada en representaciones internas (percepción humana). Este modelo incluye el mapeo tiempo-frecuencia (similar a PSQM), deformaciones en frecuencia (usando una escala Bark modificada) y un escalamiento comprimido de sonoridad en donde el orden de la escala de dBm es deformada en una función no lineal para reflejar la sensitividad humana.

- Filtrado. El filtrado es aplicado para asignar características de la banda telefónica a las señales. Hasta aquí las características de filtrado de un auricular de teléfono, no afectan la medida PESQ.

- Mapeo Tiempo-Frecuencia. Como en PSQM, se usa una FFT con una ventana de 32ms, o 356 muestras con una frecuencia de muestreo de 8kHz, para segmentar las señales de entrada y salida en celdas de tiempo frecuencia individuales.

- Deformación en Frecuencia. Se ejecuta una deformación en frecuencia a una escala Bark modificada para reflejar la sensitividad humana en frecuencia.
- Deformación de la intensidad. Se realiza una deformación de la intensidad de la energía espectral a una escala Sone de sonoridad.

El proceso del modelo perceptivo produce representaciones internas de las señales de entrada y salida, es decir, la salida de este bloque, es una representación tiempo-frecuencia de las señales de entrada y salida que consideran características de percepción del ser humano.

Paso 3. Modelo Cognitivo

Su objetivo es calcular dos valores promedio de ruido, que al final son combinados para producir un puntaje MOS.

- Diferencia entrada salida en la celda Tiempo-Frecuencia. Para cada celda tiempo-frecuencia se calcula una diferencia entre la señal de entrada y salida. Una diferencia positiva indica que se han adicionado componentes como ruido y una diferencia negativa indica que se han omitido componentes debido a distorsión de codificación o a la pérdida de señal.
- Enmascaramiento de pequeña distorsión. Se aplica un umbral de clases a los niveles de distorsión dentro de cada celda y se realiza el escalamiento a cero, para enmascarar el impacto de distorsiones pequeñas que no son perceptibles en señales ruidosas.
- Proceso de asimetría. El proceso de asimetría es similar al realizado por PSQM, se calcula el disturbio asimétrico usando un factor de posicionamiento para aplicar diversos pesos a los disturbios positivos y negativos. El disturbio asimétrico es aquel con el que las celdas quedan con disturbios positivos solamente. El contador de escala que es parte de PSQM+ no se realiza en PESQ. Los resultados del proceso de asimetría de PESQ son valores de disturbios asimétricos únicamente positivos y normales (estos incluyen disturbios positivos y negativos).

- Disturbios de Marco (Frame). Los disturbios normales (no asimétricos) y los asimétricos son calculados y agregados a través de bandas de frecuencia, dando por resultado disturbios en el Marco.

- Detección de la variación del retardo. Desde el proceso de alineamiento en tiempo, PESQ puede detectar cuando varía el retardo e identifica los marcos involucrados. Los disturbios pequeños en un marco, son eliminados durante las variaciones de retardo para prevenir falsos puntajes pobres.

- Alineamiento de tiempo reasignado. Se asigna otro alineamiento de tiempo para marcos consecutivos hasta un umbral. Si se determina que el alineamiento de tiempo es un resultado impreciso en un marco con muchos disturbios, se repite el alineamiento en tiempo y se recalculan los disturbios del marco.

- Adición de valores de disturbios y Predicción MOS. Se adicionan los valores de disturbio y disturbio asimétrico de marco, sobre el tiempo en niveles progresivos. Se calcula un puntaje MOS como una combinación lineal del promedio del valor de disturbio y el promedio del valor de disturbio asimétrico. Como resultado de éste cálculo el intervalo de los puntajes MOS resultantes del análisis PESQ actualmente es 0.5 a 4.5, pero hay que notar que en muchos casos los puntajes MOS resultantes de PESQ, estarán entre 1.0 y 4.5. [23]

2 ESPECIFICACIONES

Este proyecto como un sistema práctico de comunicación de voz por medio de computadores personales, comprende dos partes esenciales: la primera permite efectuar llamadas dentro de una misma red local con direcciones IP estáticas y la otra efectúa un análisis perceptivo de la calidad del habla sobre todo el sistema (red local y teléfono VoIP).

Se requiere que el software sea instalado en cada uno de los equipos que pertenezcan a la misma red local intercomunicados por tarjetas de red y un conmutador de datos, bajo un sistema operativo Windows®.

Las direcciones IP asignadas de forma estática que se van usando, van quedando guardas en el cuadro de marcación, para poder llamar a los demás usuarios de forma ágil.

Todas estas operaciones se efectúan en una misma ventana de diálogo con un ambiente visual fácil de manejar para el usuario.

Respecto al análisis perceptivo de la calidad del habla, se escoge el archivo que se quiera reproducir dentro del mismo software, para que sea enviado una vez se establezca una llamada, en el otro Terminal se pone a grabar este archivo y luego se comparan el original que está en el Terminal que grabó con el archivo guardado. De ésta forma se puede calificar la calidad perceptiva del habla en dicho canal.

El programa se desarrolló en el lenguaje de programación Visual C++, utilizando librerías ya existentes de SIPFOUNDRY y el código de la sección de implementación de referencia del estándar P.862 para la parte de PESQ.

Sipfoundry es una comunidad internacional de código abierto dedicada a mejorar y expandir aplicaciones SIP y su tecnología base. Se Fundó en febrero de 2004. Las

comunidades de usuarios y desarrolladores de SIPfoundry se han expandido rápidamente incluyendo participantes de 63 países. Sus patrocinadores son PINGTEL y TELTEL

Los proyectos sipX fueron dados a SIPfoundry por Pingtel Corp. Ahora son proyectos de código abierto disponibles para quien quiera usarlos bajo licencia LGPL (Lesser General Public License).

Como se mencionó previamente, el software tiene dos ventanas principales de operación, una permite hacer llamadas de VoIP con señalización SIP, y la otra está diseñada para hacer un análisis perceptivo de la calidad del habla PESQ, dando un puntaje en la escala MOS, por eso se hace una descripción más detallada a continuación.

2.1 TELÉFONO SOFTWARE IP CON SEÑALIZACIÓN SIP

El teléfono IP con señalización SIP, se basa en el RFC 3261 y realiza cada una de las etapas de petición, establecimiento y finalización de llamadas de VoIP.

En una llamada SIP, el extremo que llama se denomina Cliente y el que es llamado, Servidor. El primer paso es abrir una conexión de señalización entre los dos puntos, los cuales pueden usar TCP o UDP y la sintaxis del mensaje es independiente del protocolo de transporte. En nuestro caso usaremos UDP (dado que no requerimos confiabilidad), entonces la dirección y puerto que usan las respuestas a las peticiones SIP, van contenidas en el parámetro Vía del encabezado de la petición SIP. [2]

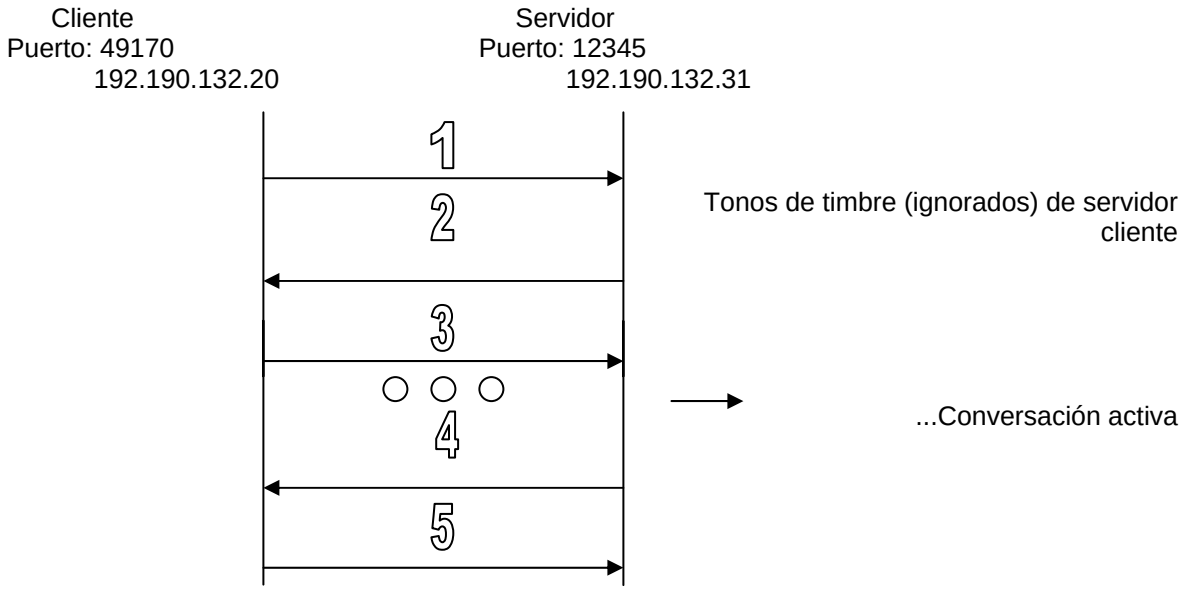

FIGURA 16 Proceso de una llamada con señalización SIP

Paso 1: El cliente llama al servidor al enviar un mensaje de invitación INVITE, cuyas características las describen parámetros de SDP; la codificación escogida por el cliente va como parte del encabezado RTP.

Paso 2: Aquí se hace la negociación de codec. Cuando el servidor acepta la invitación, envía un mensaje de respuesta OK. Si el codec no es compatible responde con un no aceptable, mostrando sus codecs para que el cliente reinicie el proceso con un INVITE mas apropiado. Si el servidor no puede o no desea aceptar, manda un REPLY especificando la razón.

Paso 3: El cliente necesita reconocer que ha recibido apropiadamente el mensaje de OK o REPLY, para esto se envía un mensaje de reconocimiento ACK. Si nunca recibió el paso 2, vuelve a intentar el paso 1.

Paso 4: Luego de establecida y cursada la llamada, se termina por cliente o servidor, en este caso es el servidor quien manda una petición de adiós BYE.

Paso 5: El cliente necesita reconocer que ha recibido apropiadamente el mensaje de Bye, para esto se envía un mensaje de reconocimiento BYE.

Existen diversas librerías que ofrecen quienes han implementado SIP en diferentes proyectos y la escogencia de alguna de ellas, se basó en factores como soporte, escalabilidad, modularidad de la arquitectura y lenguaje de desarrollo entre otros.

Con la utilización de alguna de estas librerías como herramienta de ayuda, se redujo el tiempo de implementación de la señalización considerablemente, haciendo uso del principio de reutilización de software.

2.2 EVALUACIÓN PERCEPTIVA DE LA CALIDAD DEL HABLA – PESQ

La evaluación perceptiva de la calidad del habla o PESQ, se basa en el estándar ITU-T P.862, y realiza cada uno de los pasos del procedimiento, pre-procesamiento de la señal, modelamiento perceptivo y cognitivo, para llegar a un resultado numérico MOS de la calidad perceptiva del habla.

Se ha decidido la utilización del código que ofrece el estándar para uso de evaluación del mismo, debido a que sigue por completo los lineamientos y se encuentra implementado en C, esto facilitó la integración de las aplicaciones para hacer de forma uniforme la programación del software. Por eso no fue necesario buscar otras alternativas para esta parte del desarrollo del programa.

El código del estándar está subdividido en siete archivos los cuales están enumerados a continuación:

- dsp.c Rutinas básicas DSP
- dsp.h Archivo de encabezamiento para el dsp
- pesq.h Archivo de encabezamiento general

- pesqdsp.c Rutinas DSP PESQ
- pesqio.c Archivo entrada – salida
- pesqmain.c Programa principal
- pesqmod.c Modelo PESQ de alto nivel
- pesqpar.h Definiciones del modelo perceptivo PESQ

De las cuales se extrajeron las más importantes para explicar el proceso que lleva a cabo el programa durante el análisis.

A continuación se describen las etapas específicas de cada uno de los módulos del algoritmo de PESQ que sugiere la recomendación P.862 de la ITU-T.

2.2.1 PRE-PROCESAMIENTO DE SEÑALES

El Pre-Procesamiento es necesario, para tener en cuenta el filtrado en el camino de envío del auricular y para asegurar que los niveles de potencia sean fijados en un rango apropiado. La clave de este proceso está en la transformación de las señales original y degradada, a una representación que es análoga a la psicofísica de las señales de audio, en el sistema auditivo humano, tomando en cuenta la frecuencia perceptiva (Bark) y "loudness[7]". Esto se logra en varias etapas: alineación de tiempo, alineación de nivel a un nivel calibrado, mapeo de tiempo a frecuencia entre otras. [23]

- *CÓMPUTO DE LA GANANCIA DE TODO EL SISTEMA (ALINEACIÓN DE NIVEL)*

La ganancia del sistema bajo prueba, no se conoce con anterioridad y puede variar considerablemente; además, no existe un nivel calibrado al cual esté grabada la señal, por esto es necesario alinear en nivel las dos señales, original X(t) y degradada Y(t) al mismo nivel constante de potencia. PESQ asume que el

[7] Loudness: Sonoridad, es un término subjetivo que describe la fuerza de la percepción de un sonido por el oído.

nivel de escucha subjetivo, es una constante de 79dB SPL[8] en el punto de referencia del oído. El algoritmo de alineación de nivel procede así:

- Se calculan versiones filtradas de la señal inicial y de la señal degradada. El filtro suprime todos los componentes con frecuencias inferiores a 250 Hz y deja pasar las señales de hasta 2000 Hz con una respuesta uniforme; a partir de esta frecuencia presenta una caída gradual, pasando por los siguientes puntos: {2000 Hz, 0 dB}, {2500 Hz, –5 dB}, {3000 Hz, -10 dB}, {3150 Hz, –20 dB}, {3500 Hz, –50 dB}, {4000 Hz y frecuencias superiores, –500 dB}. Estas versiones filtradas de las señales sólo se utilizan para el cálculo de la ganancia del sistema global.

- Se computan las muestras de habla de las señales original y degradada, elevándolas al cuadrado y sacándoles raíz cuadrada, de ésta forma se llega a un valor promedio y se calculan las muestras de las señales vocales degradadas filtradas.

- Se calculan y aplican diferentes ganancias para alinear ambas señales X(t) y Y(t) a un nivel constante, éste es el objetivo, con lo que se obtienen las versiones sometidas a escala XS(t) e YS(t) de estas señales.

  - *FILTRADO IRS*

Se asume que las pruebas de escucha se llevaron a cabo con un auricular de características de recepción IRS. Un modelo perceptivo de la evaluación humana de calidad de habla, debe tener en cuenta esto, para modelar las señales que los sujetos realmente escucharon; por lo tanto, se computan las versiones con características de recepción IRS filtradas de las señales original y degradada.

Esto es implementado en PESQ por una FFT a lo largo del archivo, filtrando con una respuesta lineal por partes similar a la característica de recepción IRS no modificada (P.830); seguido a esto se hace una FFT inversa a lo largo del archivo,

[8] SPL: sound pressure level. Ver P.830, sección 8.1.2

así resultan las versiones filtradas XIRSS(t) y YIRSS(t) de las señales sometidas a escala XS(t) y YS(t).

Estas señales filtradas IRS son usadas en el procedimiento de alineación de tiempo y en el modelo perceptivo.

- *ALINEACIÓN DE TIEMPO*

La rutina de alineación de tiempo, proporciona al modelo de percepción, valores de retardo, para permitir la comparación de las correspondientes partes de las señales original y degradada. Éste proceso de alineación conlleva una serie de etapas:

- Estimación de retardo basado en la envolvente, usando las señales enteras original y degradada.
- División de la señal original en un número de sub-secciones llamadas articulaciones.
- Estimación de retardo basado en la envolvente sobre las articulaciones.
- Identificación de retardo basado en una fina correlación o histograma, a la muestra más cercana en las articulaciones.
- Separación de articulaciones y re-alineación de los intervalos de tiempo para buscar cambios de retardo (jitter) durante el habla.
- Después del modelo perceptivo, identificar y realinear largas secciones de errores grandes, para buscar errores de alineación.

a) Alineación basada en la envolvente

De las señales sometidas a escala XS(t) y YS(t), se calculan las envolventes XES(t)K y YES(t)K. La envolvente es definida como:

$$LOG\left(MAX\left(\frac{E(k)}{Ethresh}, 1 \right) \right)$$ **Ecuación 5**

Donde E(k) es la energía en una trama k de 4ms, y Ethresh es el umbral de habla determinado por un detector de actividad de voz.

Se usa la cross-correlación de las envolventes de la señal inicial y de la señal degradada, para estimar el retardo bruto entre ellas, con una resolución aproximada de 4ms.

b) Alineación fina de tiempo

Debido a que los modelos preceptivos son sensitivos a desplazamientos de tiempo, se hace necesario calcular un valor de retardo exacto para la muestra. Esto es computado así:

- Se aplica una ventana Hanning a las tramas de 64ms (con una superposición de 75%), las cuales, tras una alineación basada en la envolvente, son correlacionadas entre la señal inicial y la degradada.
- El máximo de la correlación, elevado a la potencia 0.125, es usado como una medida de la confianza de la alineación en cada trama. El índice del máximo da la estimación del retardo para cada trama.
- Se calcula un histograma de estas estimaciones de retardo, ponderado por la medida de confianza. Este histograma es alisado por convolución con un kernel triangular simétrico de 1ms de ancho.
- El índice del máximo en el histograma, dividido entre la suma del histograma antes de la convolución con el kernel, da una medida de confianza entre 0 (no hay confianza) y 1 (confianza plena).

Los resultados después de ejecutado la alineación fina de tiempo, son un valor de retardo y una confianza de retardo para cada articulación, tomando en cuenta cambios de retardo en periodos de silencio. El conocimiento de los puntos de comienzo y final de cada articulación, permite identificar el retardo de cada trama en el modelo perceptivo.

c) Separación de articulaciones

Los cambios del retardo durante el habla se prueban dividiendo los intervalos de tiempo en cada articulación y alineándolos de nuevo. El proceso de separación de articulaciones se repite en muchos puntos diferentes dentro de cada articulación, y la división que produce la mayor confianza es identificada. Si la confianza así obtenida es mayor que la obtenida sin división, y si cada una de las dos partes tiene un retardo significativamente diferente, entonces la articulación se divide. Esta prueba es aplicada de forma recursiva a cada parte al hacer cada separación, para la prueba de cambios de retardo posteriores.

De esta forma se tienen en cuenta los cambios en el retardo durante habla y silencio, y se calculan el retardo por intervalo de tiempo (di), junto con las muestras de comienzo y final. El número de intervalos de tiempo se determina por el número de cambios de retardo.

d) Re-alineación perceptiva

Luego de ser aplicado el modelamiento perceptivo, se identifican y realinean por cross-correlación las secciones que presentan una perturbación muy grande (que rebasa cierto umbral). Éste paso mejora la exactitud del modelo, aportando un pequeño número de archivos difíciles de alinear, donde no se identificaron correctamente los cambios en el retardo por el previo proceso de alineación. Esto se explicará mejor en la sección Re-alineación de intervalos malos.

2.2.2 MODELAMIENTO PERCEPTIVO

Éste proceso transforma las señales de entrada y salida, en señales de representaciones de la percepción humana; es decir, la salida es una representación en tiempo – frecuencia, que tiene en cuenta características de percepción. Éste modelo se usa para calcular una distancia entre la señal de habla original y degradada, puntaje PESQ; éste puntaje es mapeado a una escala MOS, es decir un número entre -0.5 y 4.5.

- *CÁLCULO PREVIO DE LOS VALORES ATRIBUIDOS A LAS CONSTANTES*

Se ejecuta un pre-cómputo a ciertos valores de constantes y funciones. Para aquellos que dependen de la frecuencia de muestreo, se guardan versiones de muestreo de 8kHz y 16kHz en el programa.

a) El tamaño de ventana de la FFT según la frecuencia de muestreo (8 o 16kHz)

En PESQ, las señales en el dominio del tiempo, se hacen corresponder a señales en el dominio de la frecuencia, usando una transformada rápida de Fourier FFT a corto plazo, con una ventana Hanning de 32ms; En el caso de la velocidad de muestreo de 8kHz, esto representa 256 muestras por ventana y en el caso de 16kHz, la ventana cuenta 512 muestras a la vez que tramas adyacentes se superponen en un 50%.

b) Umbral absoluto de audición

El umbral absoluto de audición P0(f), se interpola para obtener los valores en el centro de las bandas Bark que se utilizan; estos valores son guardados en un arreglo que se usa en la fórmula de sonoridad de Zwicker[9].

c) Factor de escala de potencia

En el análisis de tiempo – frecuencia después de la FFT, se utiliza una constante de ganancia arbitraria; ésta constante se calcula partiendo de una onda sinusoidal con una frecuencia 1kHz y amplitud 29.54 (40dB SPL), y es transformada al dominio de la frecuencia usando la FFT con ventana en 32ms. El eje de frecuencia (discreto), es luego convertido en una escala Bark modificada, para lo cual se confinan bandas FFT. La amplitud pico del espectro confinado a la escala de frecuencias Bark (llamada la "densidad de potencia del pitch[10]"), debe luego ser

[9] Zwicker: Consideró que una ley de potencia entre la suma de excitación evocada por un sonido y el ruido interno y la suma de sonoridad específica de ambos y halló la fórmula de sonoridad.
[10] Pitch: Altura del sonido.

10.000 (40 dBL). Este último valor es hecho cumplir por una multiplicación con una constante, el factor de escala de potencia Sp.

d) El factor de escala de sonoridad

El mismo tono de referencia de 40dB SPL se usa para calibrar la escala de sonoridad psicoacústica (Sone). Después del confinamiento a la escala bark modificada, el eje de intensidad es deformado para formar una escala de sonoridad utilizando la ley de Zwicker, basada en el umbral absoluto de audición. La integral de la densidad de sonoridad, a lo largo de la escala de frecuencias Bark usando un tono de calibración de 1kHz y el tono de referencia de 40dB SPL, debe entonces producir el valor de 1 Sone. Este último valor es hecho cumplir por una multiplicación con una constante, el factor de escala de sonoridad Si.

- *FILTRADO IRS EN RECEPCIÓN*

Como se dijo anteriormente, se asume que las pruebas de escucha fueron llevadas a cabo usando una característica de recepción IRS en el auricular. El filtrado necesario a las señales de habla ya ha sido aplicado en el pre-procesamiento.

- *CÓMPUTO DEL INTERVALO DE TIEMPO DE HABLA ACTIVO*

Si el archivo de habla original y degradado, comienza o termina con grandes intervalos de silencio, esto puede influenciar el cálculo de ciertos valores de distorsión promedio en los archivos; por lo tanto, se hace un estimado de los periodos en silencio, en el comienzo y final de éstos archivos. La suma de los valores absolutos de cinco muestras sucesivas debe ser superior a 500, desde el comienzo y final del archivo de habla original, con el fin de considerar esa posición como comienzo o final del intervalo activo. El intervalo entre éste comienzo y final es definido como el intervalo de tiempo de habla activo. Con el fin de ahorrar ciclos de cálculo y/o espacio de almacenamiento, algunos cálculos pueden limitarse al intervalo activo.

- *TRANSFORMADA RÁPIDA DE FOURIER A CORTO PLAZO*

El oído humano efectúa una transformación de tiempo – frecuencia, en PESQ esto es implementado por una FFT a corto plazo con tamaño de ventana de 32ms. La superposición entre ventanas de tiempo (tramas) sucesivas es del 50%. Los espectros de potencia, es decir, la suma de los valores cuadráticos de la parte real y de la parte imaginaria de los componentes FFT complejos, son guardados en arreglos distintos, uno que contiene los valores reales de la señal inicial y el otro los de la señal degradada. En PESQ, la información de fase dentro de una ventana Hanning individual, se descarta y todos los cálculos son basados sólo en las representaciones de potencia PXWIRSS(f)n y PYWIRSS(f)n.

Los puntos de comienzo de las ventanas en la señal degradada, se desplazan por el retardo. El eje de tiempo de la señal de habla original, se deja sin modificar. Si el retardo aumenta, se excluyen partes de la señal degradada del procesamiento, mientras que si el retardo disminuye, algunas partes son repetidas.

- *CÁLCULO DE LAS DENSIDADES DE POTENCIA DE LA ALTURA DEL SONIDO (PITCH)*

La escala Bark, refleja el hecho que, a frecuencias bajas, la resolución de frecuencias del sistema auditivo humano es más fina que a frecuencias altas; esto es implementado al confinar las bandas de la FFT y sumar las correspondientes potencias de las bandas FFT, con una normalización de las partes sumadas. La función de deformación que hace corresponder la escala de frecuencia en Hertz a la escala de pitch en Bark, no sigue exactamente los valores dados en la literatura. Las señales resultantes se conocen como las densidades de potencia del pitch PPXWIRSS(f)n y PPYWIRSS(f)n.

- *COMPENSACIÓN PARCIAL DE LA DENSIDAD DE POTENCIA ORIGINAL DEL PITCH PARA LA ECUALIZACIÓN DE LA FUNCIÓN DE TRANSFERENCIA*

Para manejar el filtrado en el sistema bajo prueba, los espectros de potencia de las densidades de potencia del pitch, de las señales original y degradada, son promediadas en el tiempo; éste promedio es calculado para las tramas activas de habla, usando sólo aquellas celdas de tiempo – frecuencia, cuyas potencias excedan por un factor de más de 1000 el umbral absoluto de audición. Por cada Bark confinado modificado, se calcula un factor de compensación parcial a partir de la razón del espectro degradado al espectro inicial; la máxima compensación nunca es mayor a 20dB. La densidad de potencia del pitch original PPXWIRSS(f)n de cada trama n, es luego multiplicada por su factor de compensación parcial, para ecualizar la señal original a la degradada; de esto resulta una densidad de potencia de la altura del sonido filtrada inversamente PPX'WIRSS(f)n.

Ésta compensación parcial, es usada porque un filtrado severo puede ser molesto para el oyente. La compensación se lleva a cabo en la señal original porque la señal degradada es la que es juzgada por los sujetos en un experimento ACR.

- *COMPENSACIÓN PARCIAL DE LA DENSIDAD DE POTENCIA DEL PITCH DISTORSIONADA PARA VARIACIONES DE LA GANANCIA EN FUNCIÓN DEL TIEMPO ENTRE LA SEÑAL DISTORSIONADA Y LA ORIGINAL*

Las variaciones de ganancia a corto plazo (corta duración), son compensadas parcialmente al procesar las densidades de potencia del pitch trama por trama. Para las densidades de potencia del pitch de las señales original y degradada, se calcula la suma en cada trama n, de todos los valores que rebasan el umbral absoluto de audición. La razón de la potencia en el archivo inicial a la potencia en el archivo degradado, se calcula y se limita a al rango [3*10-4,5]; a ésta razón se aplica un filtro pasabajos de primer orden (a lo lago del eje de tiempo). La densidad de potencia del pitch distorsionada en cada trama n, es luego

multiplicada por ésta razón, con lo que se obtiene la densidad de potencia del pitch distorsionada con compensación parcial de la ganancia PPY'WIRSS(f)n.

- *CÁLCULO DE LAS DENSIDADES DE SONORIDAD*

Después de la compensación parcial para el filtrado y las variaciones de ganancia de corta duración, las densidades de potencia del pitch original y degradada, son transformadas a una escala de sonoridad Sone utilizando la ley de Zwicker.

$$LX(f)_n = S_l \cdot \left(\frac{P_0(f)}{0.5}\right)^{\gamma} \cdot \left[\left(0.5 + 0.5 \cdot \frac{PPX'_{WIRSS}(f)_n}{P_0(f)}\right)^{\gamma} - 1\right]$$ **Ecuación 6**

Donde P0(f) es el umbral absoluto y Sl, el factor de escala de sonoridad del numeral 6.2.2.1.

Por encima de 4 Bark, la potencia Zwicker, , es 0.23, que es el valor dado en la literatura. Por debajo de 4 Bark, la potencia Zwicker es incrementada levemente para tener en cuenta el llamado efecto de reclutamiento. Los arreglos bidimensionales resultantes LX(f)n y LY(f)n son llamados densidades de sonoridad.

2.2.3 MODELAMIENTO COGNITIVO

Este proceso se ejecuta para calcular dos tipos de valores de molestia o ruido promedio, que al final son combinados, para producir un puntaje predictivo MOS. Éste proceso incluye: [22][23]

- *CÁLCULO DE LA DENSIDAD DE PERTURBACIÓN*

Se calcula la diferencia con signo entre la densidad de sonoridad distorsionada y la original; cuando ésta diferencia es positiva, quiere decir que se han sumado componentes como ruido, cuando es negativa, se han omitido componentes de la señal original. Éste arreglo de diferencias es llamado densidad de perturbación bruta.

Se calcula el mínimo de densidad de sonoridad original y degradada para cada celda de tiempo – frecuencia. Éstos mínimos se multiplican por 0.25 y el correspondiente arreglo bidimensional es llamado arreglo de máscara. Seguido a esto, en cada celda de tiempo – frecuencia se aplican las siguientes reglas:

- Si la densidad de perturbación bruta es positiva y mayor que el valor de máscara, el valor de máscara es restado de la perturbación bruta.
- Si la densidad de perturbación bruta está comprendida entre más y menos la magnitud del valor de máscara, la densidad de perturbación se fija a cero.
- Si la densidad de perturbación bruta es más negativa que menos el valor de máscara, el valor de máscara es sumado a la densidad de perturbación bruta.

El efecto neto hace que las densidades de perturbación brutas tiendan a desplazarse a cero. Esto representa una zona muerta antes que una celda de tiempo – frecuencia real sea percibida como distorsionada. Lo anterior modela el tratamiento de pequeñas diferencias inaudibles en presencia de señales de gran sonoridad (señales enmascaradoras), en cada celda de tiempo – frecuencia. El resultado es una densidad de perturbación en función del tiempo (número de ventana n) y de frecuencia D(f)n.

- *MULTIPLICACIÓN CELDA A CELDA POR UN FACTOR DE ASIMETRÍA*

El efecto de asimetría es causado por el hecho que cuando un codec distorsiona la señal de entrada, será en general muy difícil introducir un nuevo componente de tiempo – frecuencia, que se integre con la señal de entrada, por lo que la señal de salida resultante se descompondrá en dos partes diferentes, la señal de entrada y la distorsión; esto conduce a una distorsión claramente audible. Cuando el codec deja fuera un componente de tiempo – frecuencia, la señal de salida resultante no puede descomponerse de la misma manera y la distorsión se nota menos. Éste efecto es modelado al calcular una densidad de perturbación asimétrica DA(f)n por trama mediante la multiplicación de la densidad de perturbación D(f)n por un factor

de asimetría; éste factor de asimetría es igual a la razón de las densidades de potencia de la altura del sonido distorsionada y original, elevada a la potencia 1.2. Si el factor de asimetría es inferior a 3, se fija a cero; si excede 12, se pone en ése valor. De esta forma, sólo quedan con valores diferentes de cero, las celdas de tiempo-frecuencia para las cuales la densidad de potencia de la altura del sonido degradada, es mayor que la densidad de potencia de la altura del sonido original.

- *SUMA DE LAS DENSIDADES DE PERTURBACIÓN EN FUNCIÓN DE LA FRECUENCIA Y ACENTUACIÓN SOBRE LAS PARTES DÉBILES DE LA SEÑAL ORIGINAL*

La densidad de perturbación D(f)n y la densidad de perturbación asimétrica DA(f)n son sumadas, a lo largo del eje de la frecuencia utilizando dos normas Lp diferentes y una ponderación sobre las tramas débiles (tramas de baja sonoridad):

$$D_n = M_n \sqrt[3]{\sum_{f=1,\ldots \text{Número de bandas Bark}} \left(\left| D(f)_n \right| W_f \right)^3}$$ **Ecuación 7**

$$DA_n = M_n \sum_{f=1,\ldots \text{Número de bandas Bark}} \left(\left| DA(f)_n \right| W_f \right)$$ **Ecuación 8**

Donde Mn es un factor multiplicador, 1/(potencia de la trama original más una constante)0.04, de lo que resulta una acentuación de las perturbaciones que se producen durante silencios en el fragmento de habla original, y Wf, es una serie de constantes proporcionales a la anchura de los confines Bark modificados. Después de ésta multiplicación, los valores de perturbaciones de trama son limitados a un máximo de 45. Éstos valores agregados, Dn y DAn, son llamados perturbaciones de trama.

- *PUESTA EN CERO DE LAS PERTURBACIONES DE AQUELLAS TRAMAS DURANTE LAS CUALES EL RETARDO DISMINUYÓ APRECIABLEMENTE*

Si el retardo de la señal deformada es de más de 16 ms (la mitad de la ventana), se modifica la estrategia de repetición mencionada en el numeral 2.4. Se ha

observado que a los efectos de determinar la calidad del habla por métodos objetivos, es mejor no tener en cuenta las perturbaciones durante tales eventos; como consecuencia de esto, las perturbaciones de trama se fijan a cero en tales casos, entonces las perturbaciones de trama se designan por D'n y DA'n.

- *REALINEACIÓN DE INTERVALOS MALOS*

Las tramas consecutivas con una perturbación de trama superior a un determinado umbral, se denominan intervalos malos. En pocos casos, la medida objetiva predice grandes distorsiones en un número mínimo de tramas malas, debido a retardos de tiempo incorrectos observados por el pre-procesamiento. Se estima un valor de retardo nuevo para ésos llamados malos intervalos, maximizando la correlación cruzada entre la señal original absoluta y la señal degradada absoluta, ajustada de acuerdo a los retardos observados por el pre-procesamiento. Cuando la correlación cruzada máxima esta por debajo de cierto umbral, se concluye que el intervalo está haciendo concordar ruido con ruido, y el intervalo ya no es llamado malo, por tanto el procesamiento para ese intervalo se detiene; en caso contrario, se vuelve a calcular la perturbación de trama para las tramas durante los intervalos malos, y el valor hallado, si es menor, sustituye a la perturbación de trama original. Los resultados son las perturbaciones de trama finales D''n y DA''n que se utilizan para calcular la calidad percibida.

- *SUMA DE LA PERTURBACIÓN DENTRO DE LOS INTERVALOS DE FRACCIÓN DE SEGUNDO*

Seguido a esto, los valores de perturbación de trama y los valores de perturbación de trama asimétrica se suman en intervalos de fracción de segundo de 20 tramas (teniendo en cuenta la superposición de tramas: aproximadamente 320ms), utilizando normas L6, un valor de p mayor que el empleado en la suma a lo largo del archivo de habla. Estos intervalos también presentan una superposición en 50% y no se necesita una función de ventana.

- *SUMA DE LA PERTURBACIÓN POR LA DURACIÓN DE LA SEÑAL DE HABLA (ALREDEDOR DE 10S), INCLUYENDO UN FACTOR DE HECHO RECIENTE*

Los valores de perturbación en la fracción de segundo y los valores de perturbación asimétrica en la fracción de segundo, se suman en el intervalo activo (las tramas correspondientes) de los archivos de habla, ahora utilizando normas L2. El valor más alto de p para la suma dentro de los intervalos de fracción de segundo, en comparación con el valor más bajo de p para la suma en el archivo de habla, es debido al hecho que cuando partes de esa fracción de segundo están distorsionadas, esa fracción de segundo pierde significado, por ejemplo si la primera frase de un archivo de habla esta distorsionada, la calidad de otras frases puede permanecer intacta.

- *CÓMPUTO DEL PUNTAJE PESQ*

El puntaje final de PESQ, es una combinación lineal del valor de perturbación promedio y del valor de perturbación asimétrica promedio. El rango del puntaje PESQ es de -0.5 a 4.5, sin embargo para la mayoría de los casos, el rango de salida será un puntaje de calidad de escucha tipo MOS entre 1.0 y 4.5, que es el rango normal de valores de MOS que se dan en un experimento sobre la calidad de escucha ACR.

- *DIAGRAMAS DE BLOQUES DE PESQ. [22]*

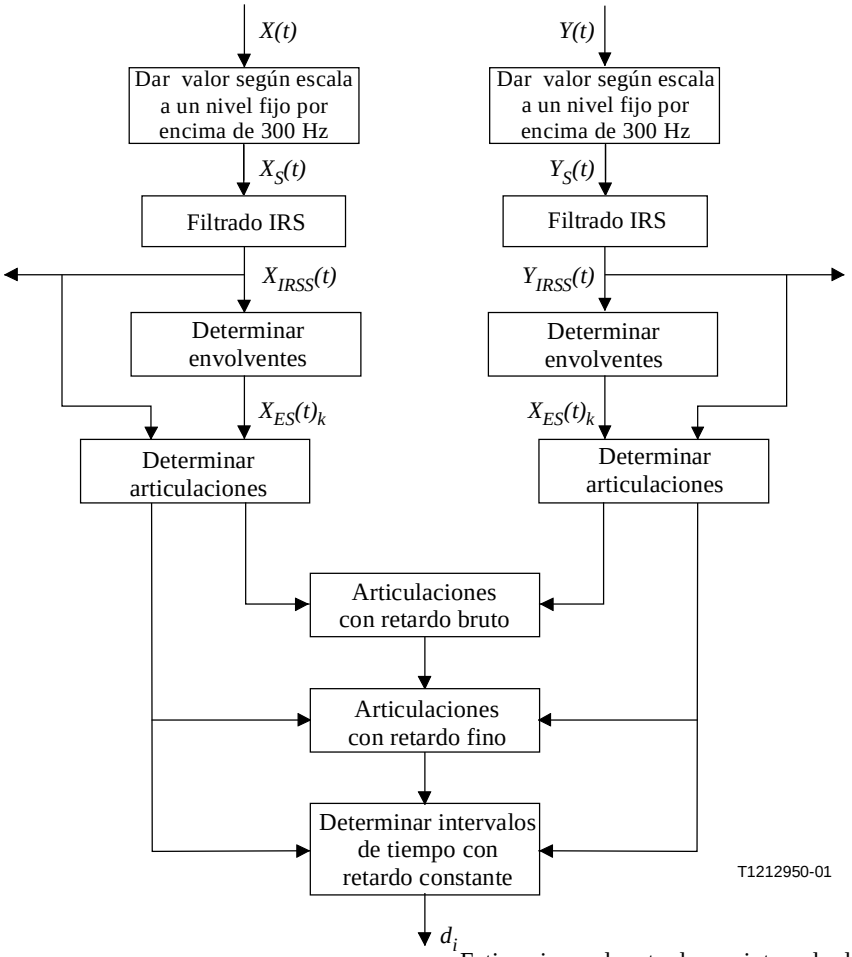

FIGURA 17 Pre-Procesamiento de señales[11]

[11] Descripción de la rutina de alineación utilizada en PESQ para determinar el retardo por intervalo de tiempo di.

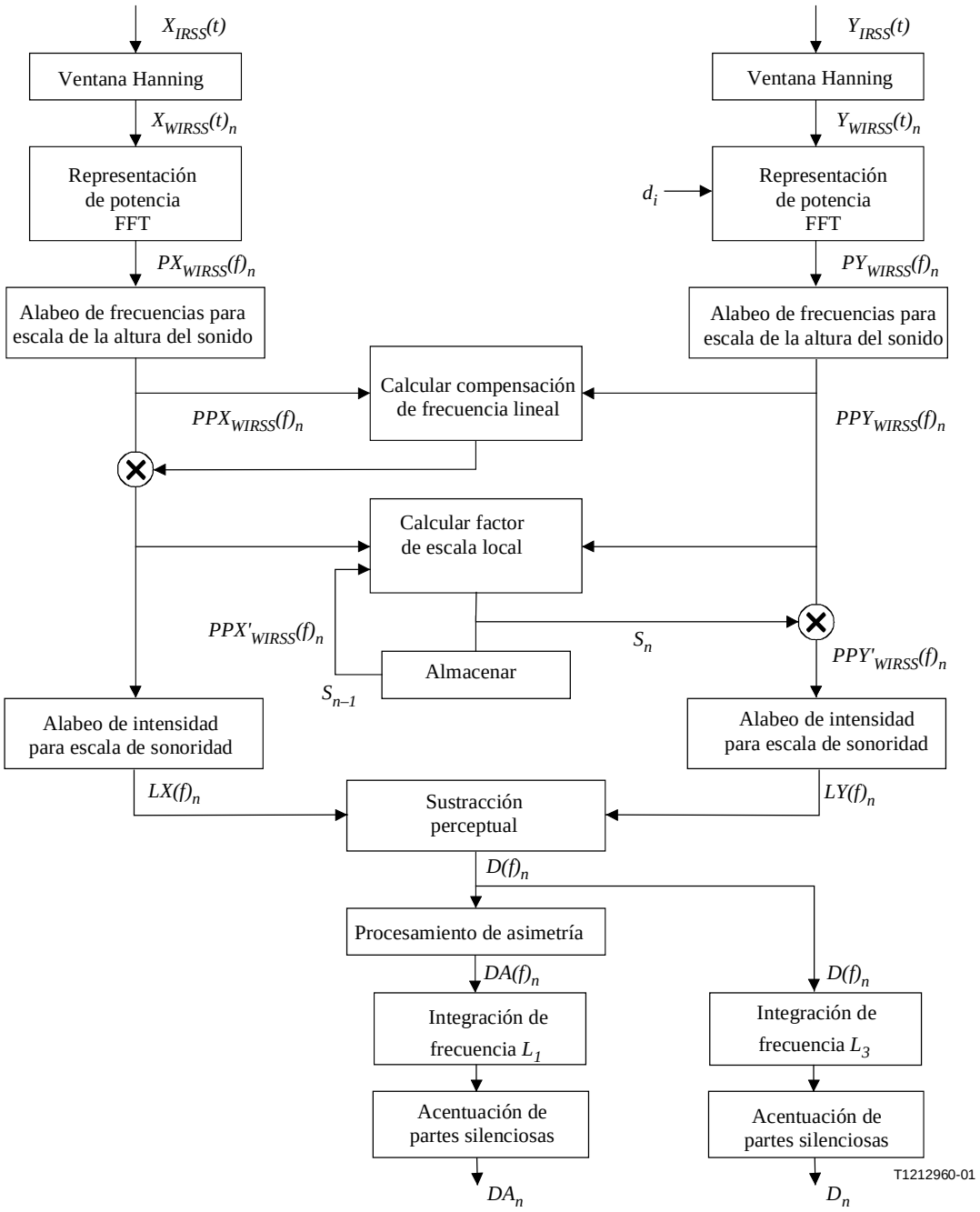

FIGURA 18 Modelamiento Perceptivo [12]

[12] Las distorsiones por trama Dn y DAn deben ser agregados en el tiempo para obtener las molestias finales.

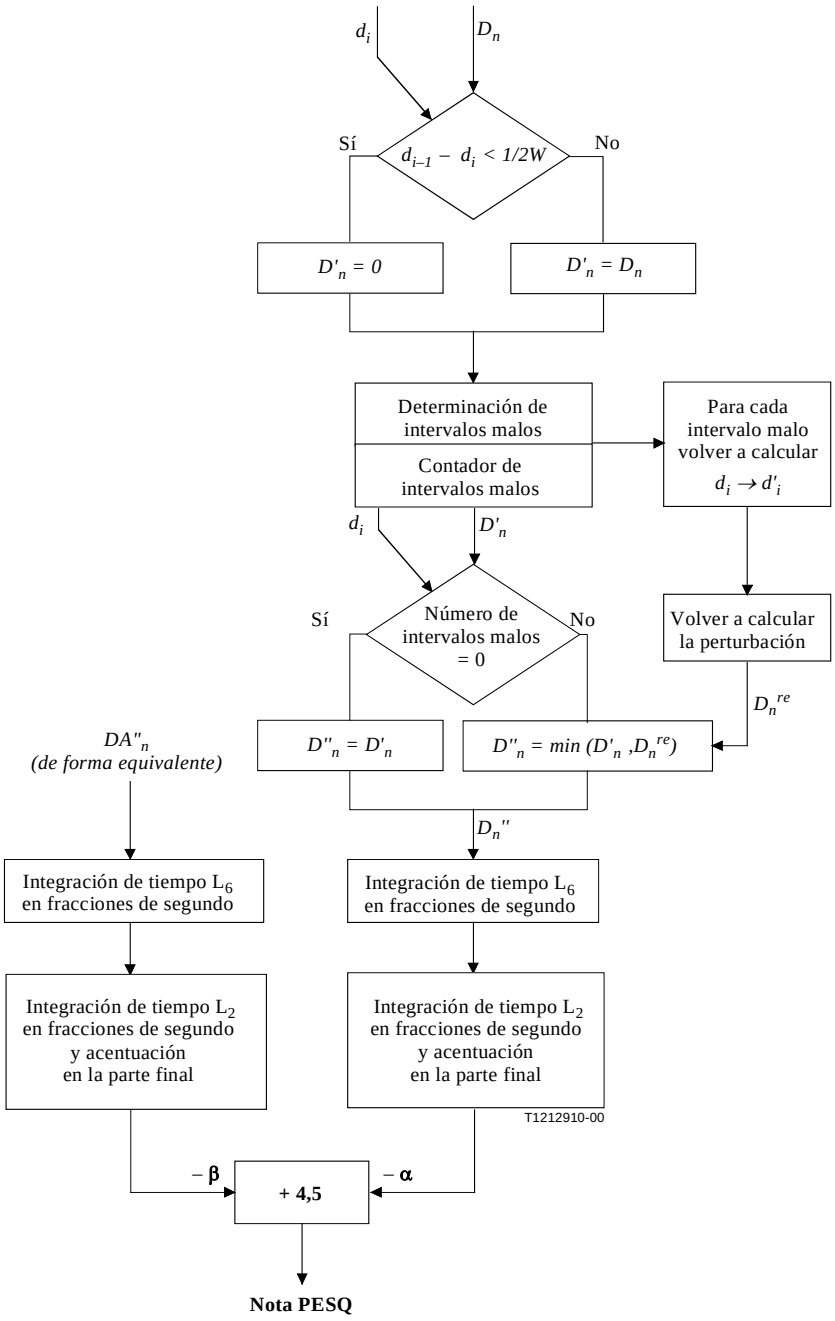

FIGURA 19 Modelamiento Cognitivo [13]

[13] Después del re-alineamiento de los intervalos malos, las distorsiones por tramas d''n y da''n son integradas en el tiempo y mapeadas al puntaje PESQ. W es la longitud de la ventana de la FFT en muestras.

2.3 DIAGRAMA EN BLOQUES

A continuación se presenta el diagrama en bloques general con las dos partes principales del programa, llamadas de VoIP con señalización SIP y Análisis PESQ.

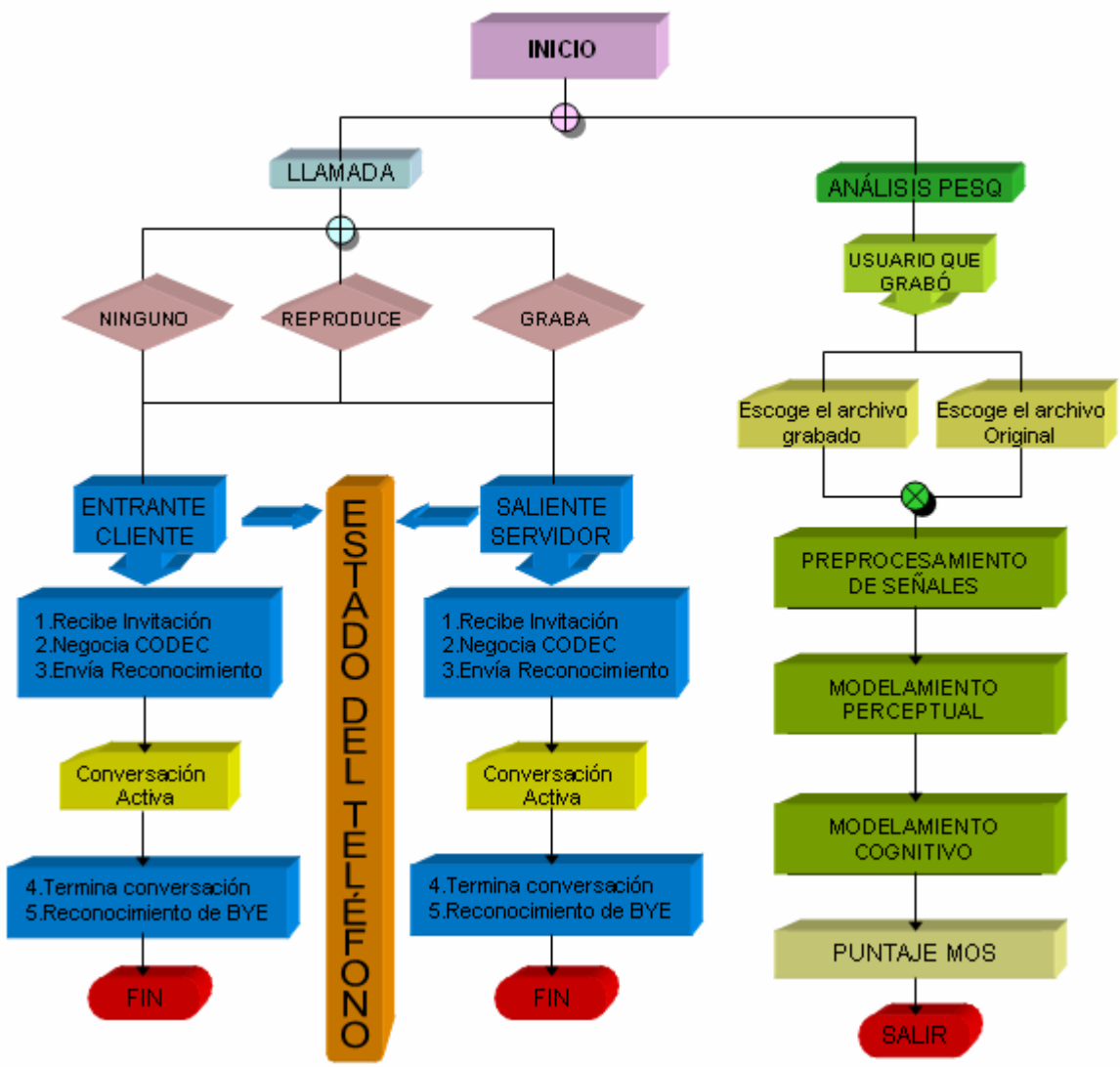

FIGURA 20 Diagrama de bloques del funcionamiento del sistema .

3 DESARROLLOS

3.1 TELÉFONO SOFTWARE IP CON SEÑALIZACIÓN SIP

La primera tarea ejecutada para el desarrollo de éste trabajo fue la investigación de librerías de código libre del stack SIP basadas en RFC3261 y en C++, además de otras que simplificaran el manejo de RTP y la portabilidad al sistema operativo.

Sólo tres librerías cumplían total o parcialmente con éstos requisitos en ésta investigación;

LibOsip2: Unido con eXosip como API, con licencia GPL carece de un manejo de RTP y de compilación en Windows.

ReSIProcate: Proyecto de SIPFOUNDRY, provee de una capa DUM como API fue escrito en .NET, carece de manejo de RTP y su licencia VOCAL/BSD, escasa documentación.

Familia SIPX: Proyecto de SIPFOUNDRY, provee Licencia LGPL, manejo de RTP y portabilidad y SipXtapi SDK para desarrollo de teléfonos IP.

Se sumarían otras características necesarias como una buena documentación de las librerías y una implementación ejemplo de una llamada activa así como una lista activa de correos de usuarios y desarrolladores, entonces se decidió estudiar a fondo SIPX.

En principio se partió de los ejemplos recibir y hacer llamada de la librería sipXcallLib que hace uso específicamente de SipXtapi SDK, para hacer pruebas y comprender a fondo SipXtapi SDK, entonces se desarrolló una interfaz gráfica en MFC; debido a la falta de clases que manejaran los estados del teléfono y a la necesidad de sumar características de control activo de volumen y ganancia del micrófono se estudió la posibilidad de trabajar en base a sipXezPhone basado a

su vez en SipXtapi SDK, y por ende en las 4 librerías y destinado específicamente para servir como implementación de ejemplo para desarrolladores que desearan aprender SipXtapi SDK. Debido a una infraestructura de clases de estado existentes en el proyecto, se hizo uso del principio de reutilización de código para así desechar componentes que no se necesitaban (por ejemplo llamada en espera y transferencia de llamada, entre otros) y adherir nuevas características (por ejemplo reproducción y grabación de archivos .wav) por medio de métodos específicos de SipXtapi SDK y clases del Dr. Fred DePiero para lectura y escritura de archivos .wav para así finalmente desarrollar en conjunto con el componente de PESQ la interfaz de usuario programada en wxWidgets.

3.1.1 ARQUITECTURA DEL TELÉFONO

El teléfono SipPesq, se basa en la arquitectura de las librerías de Sipfoundry como ya se había mencionado, a continuación se presenta un esquema de la jerarquía de éstas.

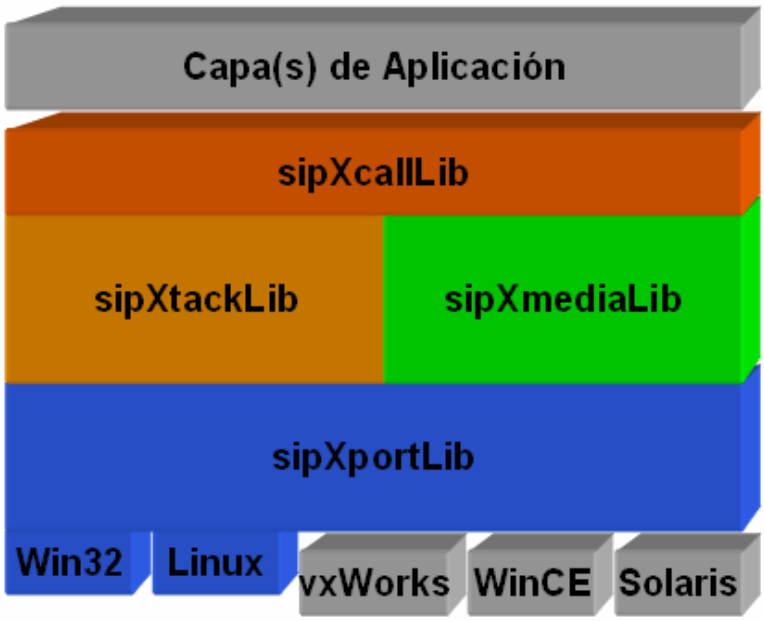

FIGURA 21 Arquitectura de las librerías

A partir de la arquitectura general de estas librerías, se obtuvo un esquema más detallado del Teléfono SipPesq el cuál se muestra a continuación.

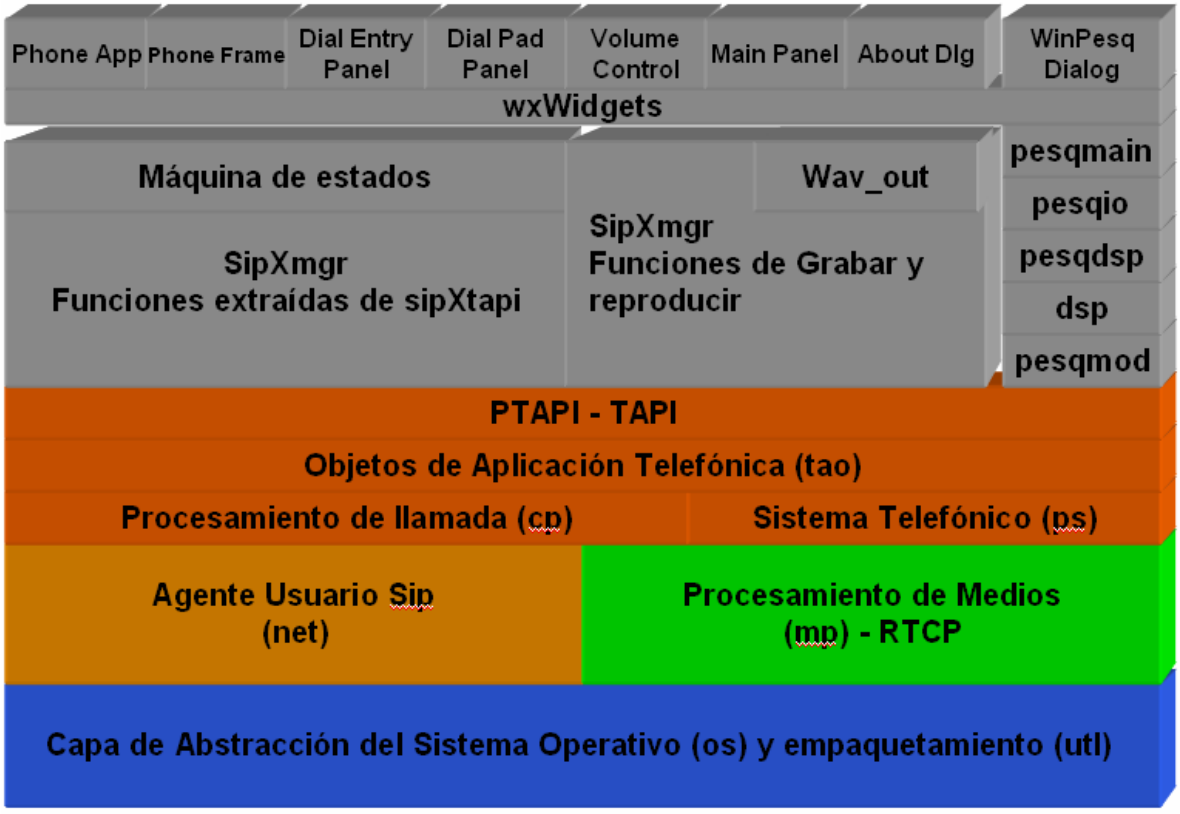

FIGURA 22 Arquitectura del teléfono SipPesq

3.1.2 CAPA DE APLICACIÓN

La capa superior o capa de aplicación está dividida en cuatro secciones básicas que a su vez están subdivididas de acuerdo al siguiente esquema.

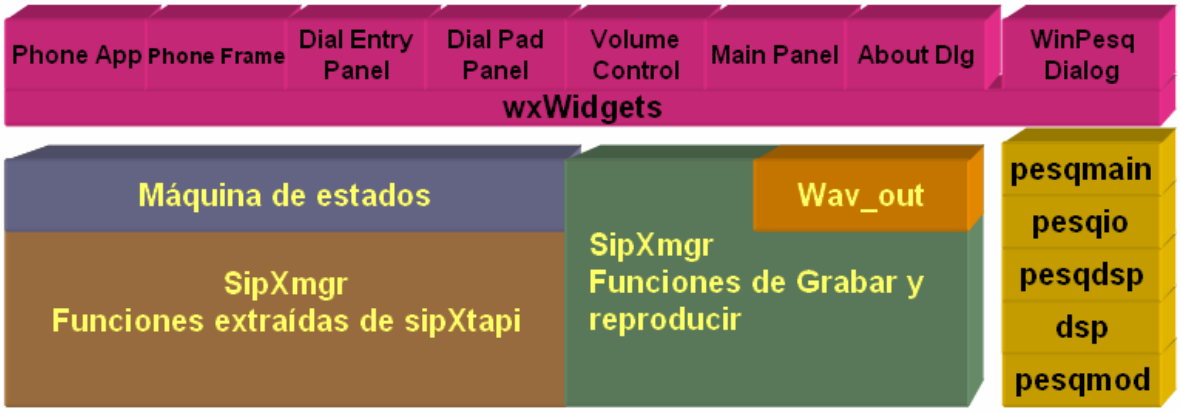

FIGURA 23 Esquema detallado de la Capa de Aplicación

La parte superior, representa toda la interfaz gráfica, es decir todos los elementos que componen las ventanas del programa.

- Phone Application. Crea la ventana principal del programa especificando su ubicación su tamaño y su icono entre otros.
- Phone Frame. Crea los enlaces para llamar a las demás ventanas que integran el programa.
- Dial Entry Panel. Crea algunos de los elementos que componen al Main Panel.
- Dial Pad Panel. Crea el teclado alfanumérico del teléfono.
- Volume Control. Crea los “potenciómetros” de volumen para el micrófono y los audífonos.
- Main Panel. Crea los demás elementos que lo conforman y llama a los que han sido creados por otros archivos.
- About Dialog. Crea la ventana “Acerca de” del programa.
- WinPesqDlg. Crea la ventana de WinPesq con todos los elementos que la conforman.

Toda la interfaz gráfica se basa en la librería wxWidgets debido a que el software abierto pretende ser independiente del sistema operativo en el que se instale.

- SipXmgr. La librería SipXmgr es una extracción de las funciones que se usan en SipPesq, de la librería SipXtapi, esto se hace con el fin de que la aplicación sea más liviana y no contenga funciones que no utiliza. La máquina de estados pretende monitorear el estado en el que se encuentra el teléfono (desocupado, marcando, conectado etc.) y está dentro de SipXmgr. Wav_Out es una función que se implementó para poder generar archivos .wav al igual que las funciones de grabar y reproducir y también se encuentran incluidas en la librería SipXmgr.
- Pesqmain – pesqio – pesqdsp – dsp – pesqmod. Son los archivos que contienen el código de PESQ que suministra el estándar ITU-T P.862.

Esta fue la capa en la que se trabajó para llegar al resultado de SipPesq y en la siguiente figura se muestran los bloques que fueron creados o modificados en este trabajo de grado.

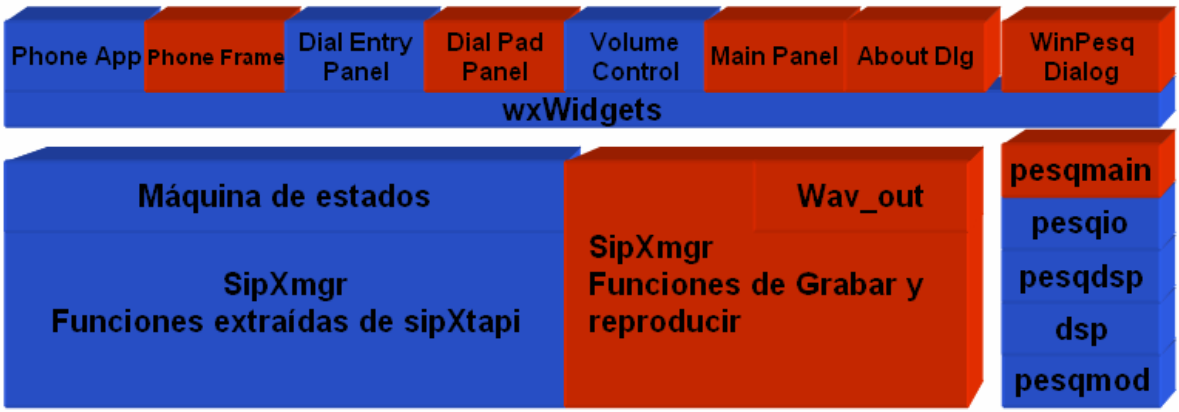

FIGURA 24 Esquema detallado de los bloques modificados o creados de la Capa de Aplicación

3.1.3 SIPXCALLLIB (PROCESAMIENTO DE LLAMADAS)

Provee el núcleo de todos los puntos finales (end points) de sipX. Esta capa está construida sobre de SIPSTACK (Net Layer) y la capa de procesamiento de medios (mp). La capa de procesamiento de llamadas es dividida en: PTAPI, Telephony Application Object (tao), Call processing (cp) y Phoneset(ps) .

ptapi – Es una versión en C++ Telephony Application Interface.

tao – Capa de aplicación del procesamiento de la llamada, usa colas de mensajes para transferir información entre las capas de aplicación y el procesamiento de llamadas.

cp – Modelo abstracto de procesamiento de llamadas. El modelo básico incluye abstracciones para llamadas y conexiones. Esta capa une el protocolo SIP de establecimiento de llamada y al subsistema multimedia mp. Adicionalmente, ésta capa provee las capacidades para preguntar o escuchar cambios de estado a través de un camino de observación.

ps – una abstracción de los objetos del teléfono virtual o físico. Phoneset contiene objetos que representan los componentes físicos de un teléfono o softphone. Por ejemplo, los objetos modelan el "hookswitch", el botón de habla "speaker", los botones de marcado, control de volumen, lámparas, micrófono, etc.

3.1.4 SIPXTACKLIB (PRUEBA RFC 3261)

Provee una librería de acuerdo al RFC 3261 y 3263, está diseñado para ser usado en sistemas embebidos. La primera interfaz es Sip User Agent. Este proyecto depende de sipXportlib y tiene herramientas para enviar y recibir transacciones SIP como Siptest.

El SipUserAgent es la interface principal de mensajes SIP entrantes y salientes. Este maneja la confiabilidad, el reenvío, cancelación y los detalles de la capa del protocolo IP. Los desarrolladores que quieren enviar y recibir mensajes SIP pueden utilizar SipUserAgent sin tener que preocuparse de detalles de la transacción y transporte.

3.1.5 SIPXMEDIALIB

Marco de procesamiento de medios G711, RTP, RTCP, streaming, recorders, mezcladores. Incluye todo el procesamiento de audio que utiliza el teléfono. Por ejemplo la librería contiene puentes de audio, divisores de audio, supresores de eco, generadores de tono, soporte de flujo, RTCP, codecs G711, etc.

El subsistema de procesamiento de medios provee un marco dentro del cual todos los procesos “Trayectos Rápidos” (real time Audio) toman un lugar. El subsistema de medios permite que sea controlado por el gestor Terminal de la llamada.

El subsistema de procesamiento de medios es implementado como una colección de clases en C++. Las clases más importantes en el subsistema son aquellos que implementan el diagrama de flujo y los recursos de procesamiento de medios.

Como se muestra siguiente figura, el procesamiento de medios para una llamada es organizada como una gráfica dirigida, de procesamiento de recursos, los recursos de procesamiento de medios corresponden a operaciones de procesamiento de bajo nivel y estos recursos pueden ser adicionados dinámicamente o eliminados del diagrama de flujo.

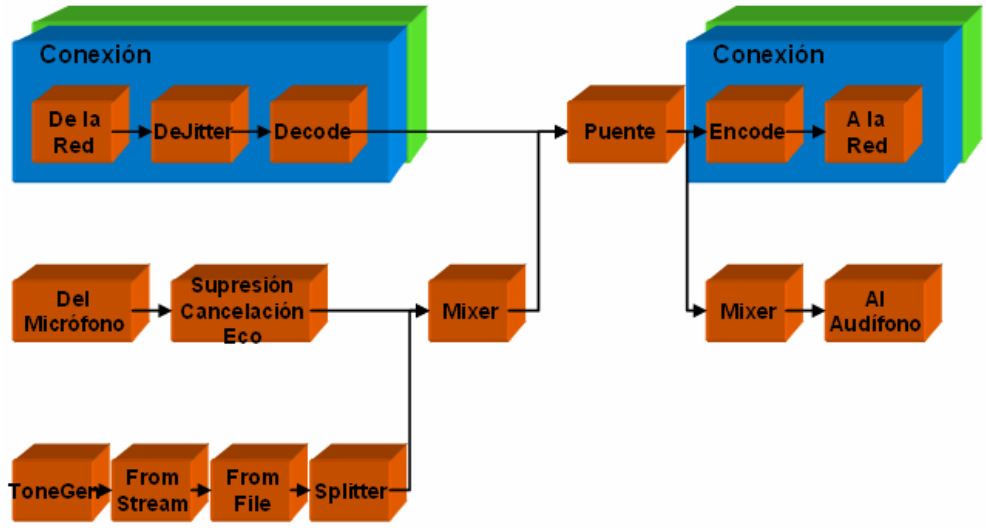

FIGURA 25 Diagrama de flujo de procesamiento de Medios

Cada llamada tiene su propio diagrama de flujo que es ejecutado cada 10ms para procesar las muestras de audio como ellas se encadenan al sistema de procesamiento de medios.

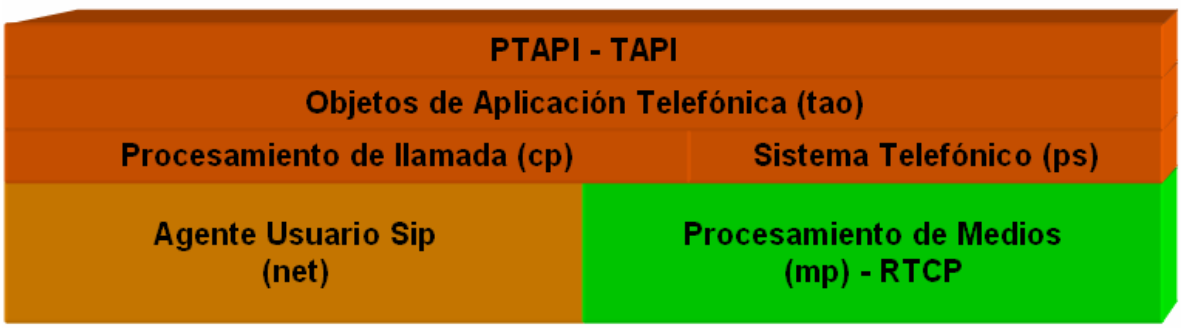

FIGURA 26 Esquema de jerarquía de capas SipXcalllib, SipXtacklib y SipXmedialib

3.1.6 SIPXPORTLIB (CAPA PORTABLE)

Provee un set de clases que dan al sistema operativo una abstracción de las demás funciones de otros Sistemas Operativos. Todos los proyectos de sipX usan esta librería para asegurar una fácil "portabilidad" en cualquier sistema operativo.

Como se muestra en la figura la OSAL está entre el sistema operativo nativo y el resto de las capas superiores del software.

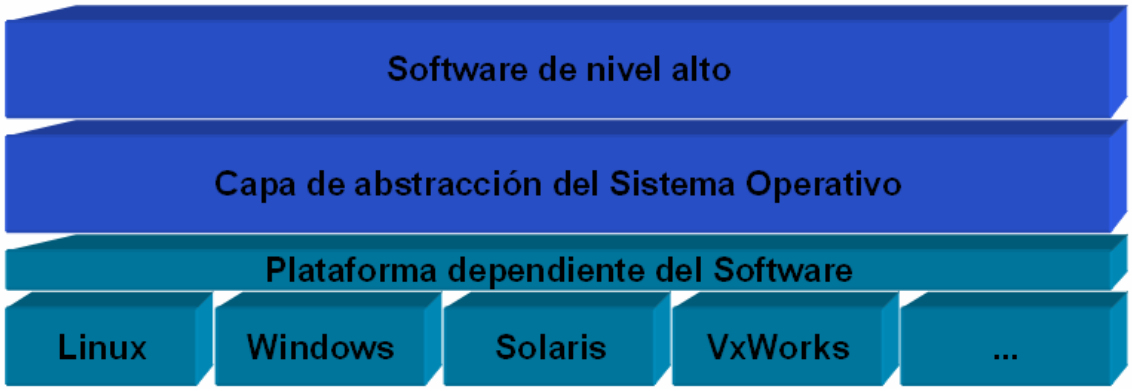

FIGURA 27 Esquema detallado de SipXportlib

La abstracción del sistema operativo aísla las capas superiores del software de las demás capas del sistema operativo nativo proveyendo una interfaz abstracta de un gran set de funciones del sistema operativo. Este nivel de abstracción lo hace muy apto para que el software funcione en diferentes plataformas.

3.2 EVALUACIÓN PERCEPTIVA DE LA CALIDAD DEL HABLA – PESQ

Los desarrollos en cuanto a la Evaluación perceptiva de la calidad del habla – PESQ, se realizaron básicamente dos pasos, la integración de éste código al teléfono de voz sobre IP y la reproducción y grabación de los archivos .wav con la ventaja de poder escoger el tiempo del proceso de grabación como se ilustrará a continuación.

3.2.1 INTEGRACIÓN DEL PESQ AL TELÉFONO DE VoIP

Básicamente la integración de PESQ al teléfono consistió en la reutilización del código brindado por el estándar P.862 con el trabajo de “traducirlo” de C a Visual C++ para poder mostrarlo en la misma interfaz gráfica del teléfono de VoIP.

3.2.2 REPRODUCCIÓN Y GRABACIÓN DE ARCHIVOS .WAV

Para poder utilizar PESQ con el teléfono de VoIP, fue necesario implementarle al mismo teléfono, una sección de grabación y reproducción de archivos .wav que posteriormente son analizados por el código PESQ.

El funcionamiento básico de esta sección es que uno de los terminales a comunicarse por el teléfono escoge entre las opciones de “grabar” o “reproducir”, mientras que el otro Terminal escoge la opción contraria, esto se hace antes de realizar la llamada, y se escogen los nombres de los archivos a grabar y reproducir respectivamente. Luego cualquiera de los terminales marca y cuando se logre establecer la llamada cada uno de los terminales ejecuta la tarea preestablecida

(grabar o reproducir) generando un archivo .wav en el Terminal que había escogido grabar.

Se recomienda que estos archivos sean los que ofrece el estándar o las Harvard Sentences ya que estos archivos fueron especialmente generados para este tipo de pruebas.

3.3 PRUEBAS REALIZADAS CON EL SOFTWARE

El estándar aconseja una serie de pruebas de las cuales se mencionarán las que realizamos.

3.3.1 COMPARACIÓN CON ARCHIVOS DE RETARDO DE VARIABLE

Las pruebas del software para evaluar PESQ, se tomaron del estándar que ofrece una base de datos, compuesta a partir de 40 condiciones (pares de archivos) que provienen de dos pruebas subjetivas, que abarcan conexiones VoIP real y simulado y que presentan un retardo variable en función del tiempo. Los nombres de archivo inicial y degradado para cada par de archivos, y la nota PESQ dada por la implementación de referencia, se proporcionan en el archivo voipref.txt, que está incluida en el código ANSI-C.

Una implementación pasa esta prueba cuando la diferencia absoluta en la nota PESQ comparada con la implementación de referencia es menor que 0,05 en 39 de los 40 pares de archivos. Se permite que un solo par de archivos tenga una diferencia absoluta en la nota PESQ de menos de 0,5. Éste puede ser cualquiera de los 40 pares de archivos.

Esta prueba de conformidad es obligatoria para todas las implementaciones.

El cuadro que se muestra a continuación contiene los resultados de esta prueba que concuerdan exactamente con los suministrados por el estándar.

ARCHIVO ORIGINAL	ARCHIVO DEGRADADO	PUNTAJE PESQ
Conform\or105.wav	Conform\dg105.wav	2.237
Conform\or109.wav	Conform\dg109.wav	3.180
Conform\or114.wav	Conform\dg114.wav	2.147
Conform\or129.wav	Conform\dg129.wav	2.680
Conform\or134.wav	Conform\dg134.wav	2.365
Conform\or137.wav	Conform\dg137.wav	3.670
Conform\or145.wav	Conform\dg145.wav	3.016
Conform\or149.wav	Conform\dg149.wav	2.558
Conform\or152.wav	Conform\dg152.wav	2.768
Conform\or154.wav	Conform\dg154.wav	2.694
Conform\or155.wav	Conform\dg155.wav	2.606
Conform\or161.wav	Conform\dg161.wav	2.608
Conform\or164.wav	Conform\dg164.wav	2.850
Conform\or166.wav	Conform\dg166.wav	2.527
Conform\or170.wav	Conform\dg170.wav	2.452
Conform\or179.wav	Conform\dg179.wav	1.828
Conform\or221.wav	Conform\dg221.wav	2.774
Conform\or229.wav	Conform\dg229.wav	2.940
Conform\or246.wav	Conform\dg246.wav	2.205
Conform\or272.wav	Conform\dg272.wav	3.288

Tabla 10 Puntajes PESQ con archivos de voipref del estándar

3.3.2 COMPARACIONES ADICIONALES

Para evitar que los implementadores acomoden específicamente un algoritmo para que sea conforme con los requisitos de los archivos descritos anteriormente, se proporciona una prueba adicional. Una implementación de PESQ que sea conforme a esta Recomendación debe, en al menos el 95% de los casos, producir una nota de salida que esté dentro de un margen de 0,05 de la nota PESQ dada por la implementación de referencia en ANSI-C.

3.3.3 PRUEBAS PROPIAS

El desempeño de la implementación del algoritmo PESQ, y las tareas de reproducción y grabación del teléfono se evaluaron con las siguientes pruebas:

- *DESCRIPCIÓN*

La prueba consistió en enviar 11 veces el mismo archivo de forma consecutiva de un Terminal a otro en once llamadas activas, con la configuración que se muestra más adelante. De estos 11 envíos se hizo el análisis PESQ y se sacó una media.

- *TIPOS DE ARCHIVOS UTILIZADOS*

Los archivos utilizados fueron algunos de los que brinda el estándar para hacer las pruebas de voipref. A continuación se enumeran los que se utilizaron:

- Or105.wav
- Or109.wav
- Or114.wav
- Or134.wav
- Or149.wav
- Or170.wav
- Or179.wav
- Or221.wav

- *TOPOLOGÍA*

FIGURA 28 Conexión directa de dos PCs.

- *RESULTADOS*

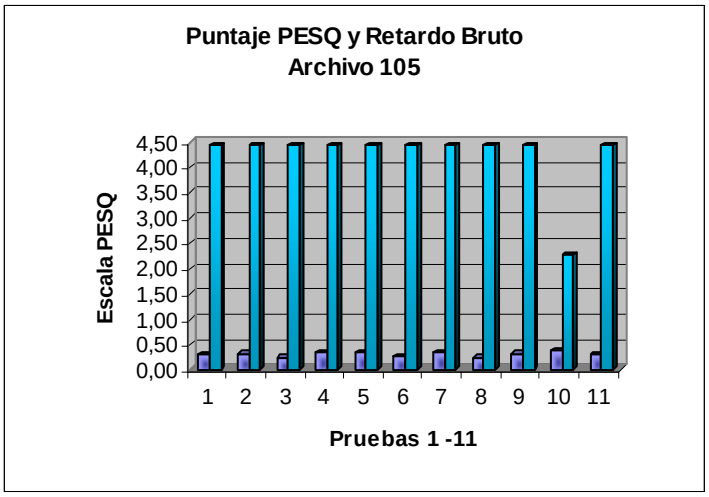

FIGURA 29 Diagrama de barras para el archivo or105.wav

Media R Bruto	Puntaje PESQ
0,308	4,236

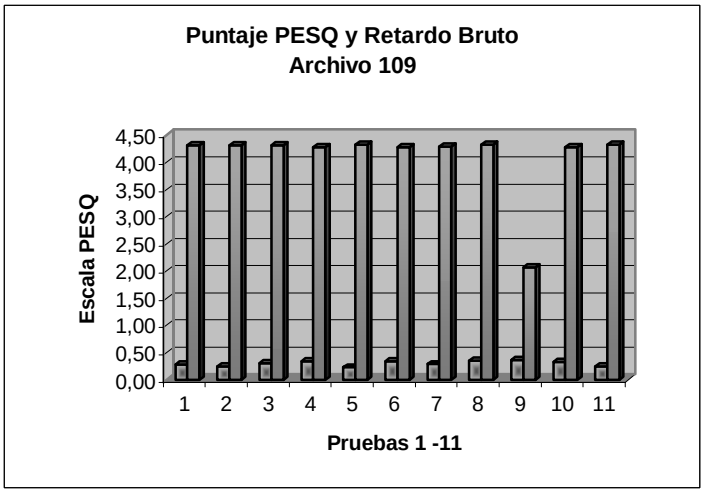

FIGURA 30 Diagrama de barras para el archivo or109.wav

Media R Bruto	Puntaje PESQ
0,301	4,100

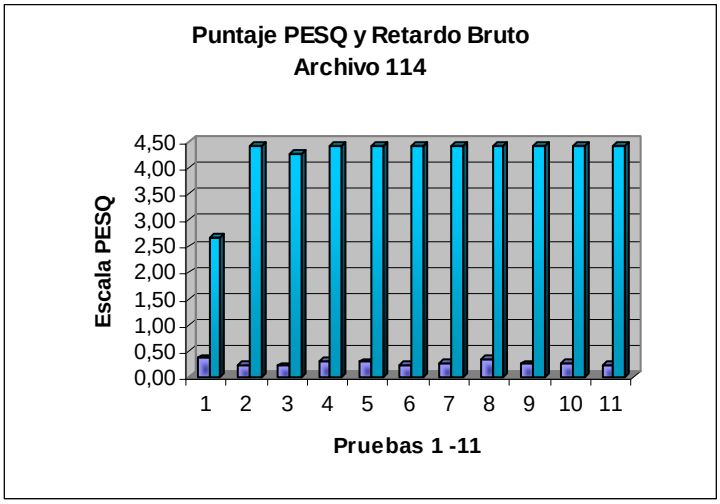

FIGURA 31 Diagrama de barras para el archivo or114.wav

Media

R Bruto	Puntaje PESQ
0,277	4,240

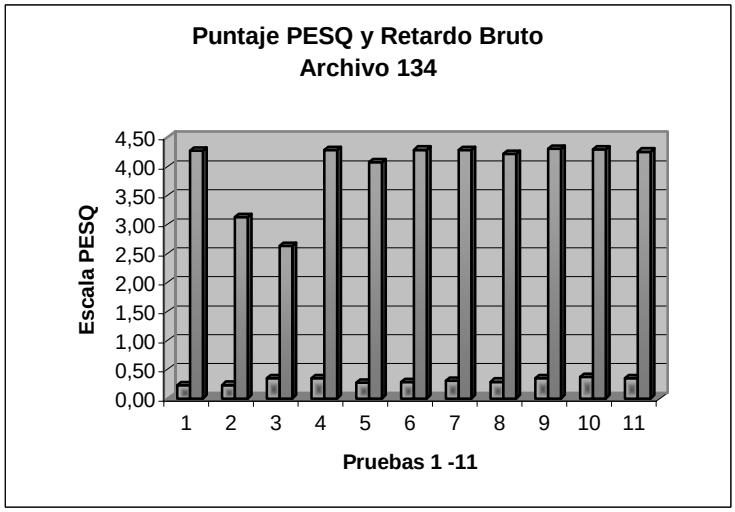

FIGURA 32 Diagrama de barras para el archivo or134.wav

Media

R Bruto	Puntaje PESQ
0,306	4,004

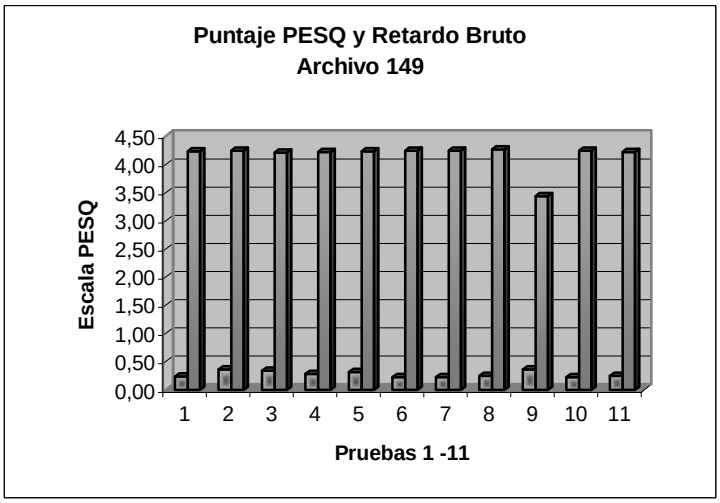

FIGURA 33 Diagrama de barras para el archivo or149.wav

Media R Bruto	Puntaje PESQ
0,276	4,163

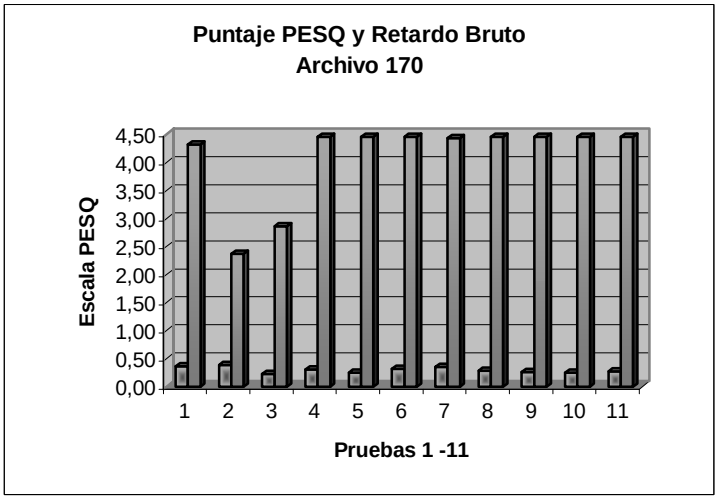

FIGURA 34 Diagrama de barras para el archivo or170.wav

Media R Bruto	Puntaje PESQ
0,295	4,110

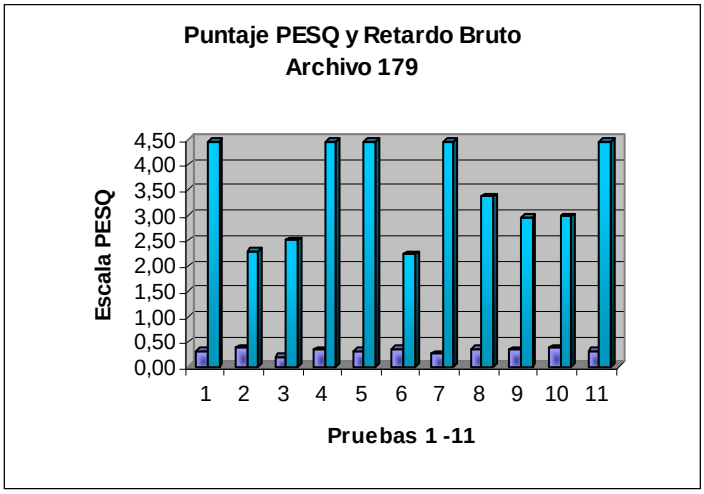

FIGURA 35 Diagrama de barras para el archivo or179.wav

Media

R Bruto	Puntaje PESQ
0,328	3,509

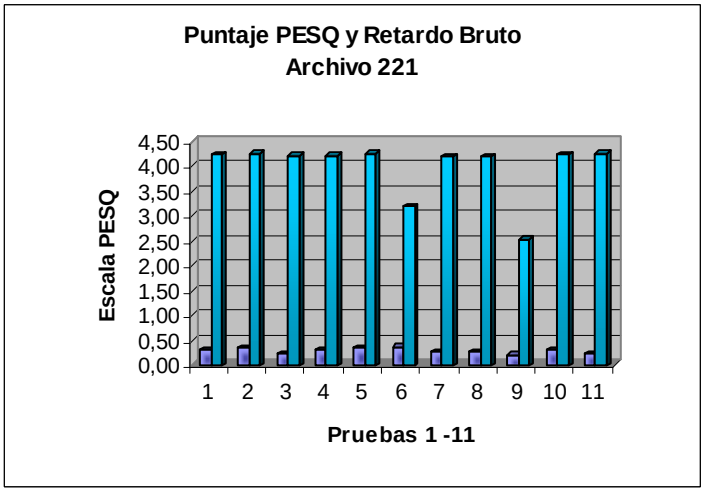

FIGURA 36 Diagrama de barras para el archivo or221.wav

Media

R Bruto	Puntaje PESQ
0,291	3,977

3.4 MANUAL DEL PROGRAMADOR

3.4.1 SIPXTAPI SDK

El Kit de Desarrollo de Software sipXtapi es una interfaz de programación de uso de C para comunicación de VoIP. Específicamente, SipXtapi provee una interfaz generalizada de telefonía encima del protocolo SIP (RFC3261) y del protocolo de tiempo real RTP (RFC1889). Mientras que los protocolos SIP y RTP proveen señalización e infraestructura de transporte de tiempo real, sipXtapi también incluye otras implementaciones de protocolo y estándares necesarias para comunicaciones de voz.

sipXtapi esta desarrollado bajo código abierto y hace parte de la línea de proyectos sipX disponibles en SIPfoundry. Su licencia es bajo LGPL, la cual permite uso comercial de la librería sin la "infección viral" asociada a GPL. Esencialmente se puede usar el código como está y construir sobre éste sin excepción alguna.

La tecnología usada bajo sipXtapi fue donada por Pingtel Corp en Marzo de 2004. La tecnología base fue y aún es usada por sus servidores Proxy, teléfonos software y equipos. La tecnología se considera bien probada y muy interoperable con otros dispositivos SIP.

a) Objetivos

El principal objetivo de sipXtapi es proveer una interfaz de programación simple para desarrolladores de aplicaciones y proveer una solución simple que abstraiga muchos de los detalles intrincados de SIP. Con sipXtapi no se necesita entender la sintaxis y semántica de los protocolos subyacentes y se puede enfocar en un modelo de llamada más familiar.

Otro objetivo fue construir una API que fuese familiar para los desarrolladores de aplicaciones en telefonía. Un número de modelos de llamada existen hoy en día,

sin embargo, Microsoft TAPI 2.2 fue escogido como base conceptual debido a su popularidad y buena separación de las características de end-point o del canal y las de un centro de llamadas.

SipXtapi SDK se construye y corre bajo Windows y Linux, las tecnologías subyacentes también han sido ejecutadas en Solaris, Linux Embebido y vxWorks.

b) Características

-API tipo TAPI

-Llamadas múltiples simultáneas

-Retener/silenciar llamada

-Control de volumen y ganancia

-Dispositivos seleccionables: entrada audio, parlante, ringer

-Separación lógica entre ringer y altavoz

-Redireccionamiento y rechazo de llamada silenciosa

-Generación de Tonos DTMF

-Reproducción de audio desde un archivo

-Enlace a interfaces específicos de la red

-Stack probado SIP RFC3261

-Tonos DTMF fuera de banda RFC2833

-Supresión de RTP cuando llamada silenciada

-Puertos configurables (UDP, TCP y RTP)

-Identidades SIP múltiples

-Autenticación de línea

-Control de receso DNS SRV

-Control de Proxy SIP

-Rport

-Codec G.711

-Jitter Buffer ajustado no adaptativo

c) Fundamentos de SipXtapi

SIPX_handles. Casi todos los métodos de sipXtapi API requieren uno más handles como argumentos de función. Los SIPX_handles representan todos los datos asociados a una llamada lógica, conferencia, identidad de línea, o instancia de user-agent. A continuación se provee una breve descripción de cada tipo de hadle.

SIPX_INST. Este handle representa una instancia de un user-agent. Un user-agent incluye un stack SIP y un marco de trabajo de procesamiento de medios. sipXtapi no soporta instancias múltiples de user-agents en el mismo espacio de proceso, sin embargo, ciertas características de procesamiento de medios se vuelven limitadas o ambiguas; por ejemplo, sólo un user-agent debería controlar los dispositivos de entrada y salida del sistema local.

SIPX_LINE. Este handle representa una identidad de entrada o de salida. Cuando se hacen llamadas de salida, el programador de la aplicación debe definir la línea de salida. Cuando se reciben llamas entrantes, la aplicación puede preguntar la línea.

SIPX_CALL. Este handle representa una llamada o conexión entre el user-agent y otro. Todas las operaciones de llamada requieren el handle de llamada como parámetro.

SIPX_CONF. Este handle representa una colección de SIPS_CALLs que tienen audio puenteado (mezclado). Los desarrolladores de aplicaciones pueden manipular cada parte de la conferencia a través de varias funciones de conferencia.

Los ciclos de vida de los handles son manejados por el marco de trabajo y la aplicación del desarrollador. Los handles SIPX_CONF y SIPX_CALL son creados explícitamente por el desarrollador de la aplicación, sin embargo, son destruidos automáticamente al fin de la sesión. Cuando se recibe una llamada entrante, un handle SIPX_CALL es creado implícitamente por el marco de trabajo.

d)Grupos de métodos funcionales

Todas las funciones del API en sipXtapi pueden ser clasificadas en grupos funcionales; ésta agrupación se deriva de los nombres de los métodos, por ejemplo, “sipXcallAccept(…)”, “sipXcallReject(…)” y “sipXcallRedirect(…)” son funciones relacionadas con la llamada, mientras que “sipxConferenceGetCalls(…)” es una función de conferencia. A continuación se provee un breve resumen de cada área funcional.

Config. SiXtapi incluye un número de ajustes de configuración que permiten a los desarrolladores de aplicaciones fijar (ajustar) el servidor Proxy SIP, ajustes de tiempo agotado, habilitar/deshabilitar SIP específico tal como señalización simétrica.

Los ajustes pueden ser cambiados en cualquier punto; sin embargo, todos los ajustes pueden no afectar las llamadas en progreso.

Call. Las características de llamada incluyen, aceptar, rechazar, y redireccionar nuevas llamadas entrantes; responder, poner en espera, silenciar, reproducir tonos, reproducir archivos de audio, y transferir llamadas activas, y acceder al ID del que llama y al llamado.

Line. SipXtapi provee métodos par definir las líneas (identidades SIP). Generalmente las líneas representan las líneas externas PSTN, y colas internas como la cola de venta y soporte. En sipXtapi, las líneas se definen en términos de identidades SIP; cada identidad se configura opcionalmente para registrarse con un registrar SIP. Las Credenciales de autenticación se configuran línea a línea.

Éste mecanismo permite al ambiente peer-to-peer donde los usuarios establecen llamadas usando direcciones IP o nombres host y a ambientes centrales orientados a directorios con clientes autenticados registrados con un bien conocido registrar.

Audio. SipXtapi provee métodos para enumerar dispositivos de audio, seleccionar el dispositivo de altavoz de llamada entrante, seleccionar el dispositivo de timbre, y fijar el dispositivo de entrada; adicionalmente, las APIs son capaces de fijar el volumen del altavoz y los niveles de ganancia del micrófono. Para los servidores, los desarrolladores de aplicaciones pueden deshabilitar los micrófonos y altavoces.

Conference. Las conferencias de cliente-mixto adhoc se establecen y manipulan a través de una serie de APIs en conferencia. Los desarrolladores de aplicaciones pueden adherir y remover participantes de una conferencia y poner a los participantes individuales en espera.

Events. Un mecanismo callback (patrón observador) se usa para comunicar transiciones de estados de llamada a la capa de aplicación. Un pequeño número de eventos mayores permiten las máquinas de estado de aplicaciones simples y el

procesamiento aerodinámico. Los eventos menores proveen información adicional y causas de transiciones de eventos mayores.

Hooks. Los desarrolladores de aplicaciones pueden "enganchar" (hook) fuentes de audio y destinos para consumir o manipular audio; éste mecanismo permite registrar, inyectar y capturar audio.

Adicionalmente, éste mecanismo ha sido usado para tender un puente sobre clientes de voz no SIP a clientes de voz SIP usando un acercamiento de user-agent back to back (B2BUA).

La API sipXtapi usa eventos para comunicar transiciones de estados a la capa de aplicación; ya que muchas de las llamadas API son asíncronas, las notificaciones de eventos deben ser revisadas por ambas operaciones como establecer una llamada y eventos generados externamente como el usuario remoto desconectándose. A continuación se listan los eventos mayores.

NEWCALL. Este evento indica que una nueva llamada ha sido creada automáticamente por sipXtapi; éste evento es generado frecuentemente como respuesta a una petición de llamada entrante.

DIALTONE. Este evento indica que una nueva llamada ha sido creada con el propósito de establecer una llamada saliente; La capa de aplicación debería determinar si necesita simular tonos de marcado para en usuario final (remoto).

REMOTE_OFFERING. Este evento indica que una invitación a establecer una llamada ha sido enviada al usuario remoto; la invitación puede o no recibir una respuesta. Si una respuesta no es recibida en un tiempo determinado, sipXtapi moverá la llamada a un estado de desconectado; si se llama a otro user-agent de tipo sipXtapi, el estado recíproco es OFFER.

REMOTE_ALERTING. Este estado indica que una invitación a establecer una llamada ha sido aceptada y el usuario remoto está en el estado de alerta (ringing);

dependiendo de la configuración SIP, end points, y servidores Proxy involucrados, éste evento debería durar sólo hasta 3 minutos, después el estado se moverá automáticamente a DISCONNECTED. Si se está llamando a otro user-agent de tipo sipXtapi, el estado recíproco es ALERTING.

CONNECTED. Este estado indica que se ha establecido una llamada entre el usuario local y remoto; El audio debería estar fluyendo y el micrófono y los altavoces deberían engancharse.

DISCONNECTED. Este estado indica que una llamada fue desconectada o falló en conectar; Una llamada puede moverse al estado DISCONNECTED desde casi cualquier otro estado. Para más información repasar los códigos DISCONNECTED de estados menores.

OFFERING. Este estado indica que una nueva invitación de llamada ha sido extendida a éste user-agent; Los desarrolladores de aplicaciones deberían invocar en respuesta sipxCallAccept(), sipxCallReject() o sipxCallRedirect(), el no responder resultará en una llamada implícita sipxCallReject().

ALERTING. Este estado indica que una llamada entrante ha sido aceptada y la capa de aplicación debería alertar al usuario final; éste estado es limitado a 3 minutos en la mayoría de las configuraciones; después la llamada será cancelada. Las aplicaciones generalmente reproducirán algún tipo de tono de timbrado en respuesta a éste evento.

DESTROYED. Este evento indica que los recursos subyacentes han sido removidos para una llamada; éste es el último evento que la aplicación recibirá para cualquier llamada; el handle llamada es inválido después que éste evento es recibido.

UNKNOWN. Este evento es generado cuando el estado para una llamada ya no se conoce; esto es generalmente una condición de error, ver el evento menor para causas específicas.

En ésta figura, el diagrama de estados representa el ciclo vital típico para una llamada saliente. Un evento es enviado a los desarrolladores de la aplicación en transiciones de estados.

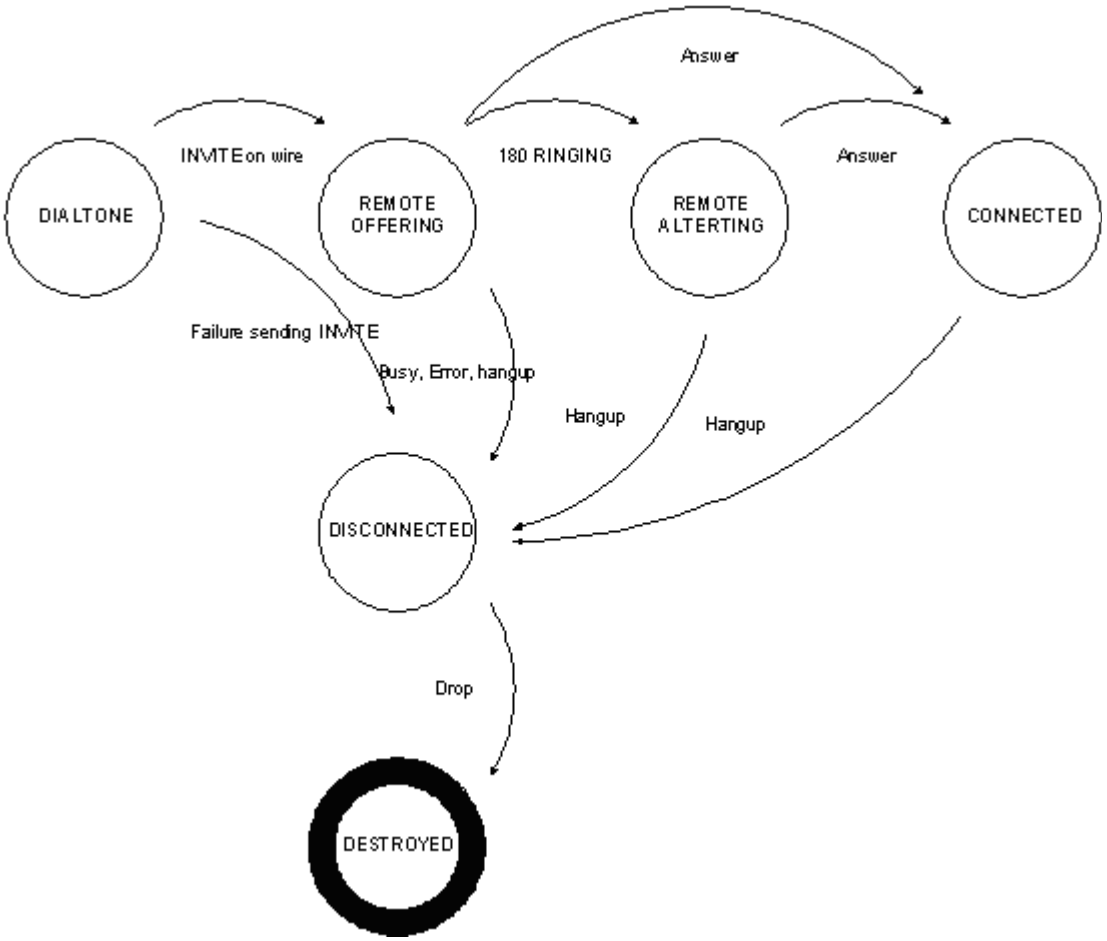

FIGURA 37 Eventos para una llamada saliente

En ésta figura, el diagrama de estados describe el ciclo vital típico para una llamada entrante. El evento OFFERING señala una petición para una conexión y

el desarrollador de la aplicación puede escoger aceptar, rechazar o redireccionar la llamada. Nota: Aceptar la llamada es a priori de notificar (o timbrar) al usuario.

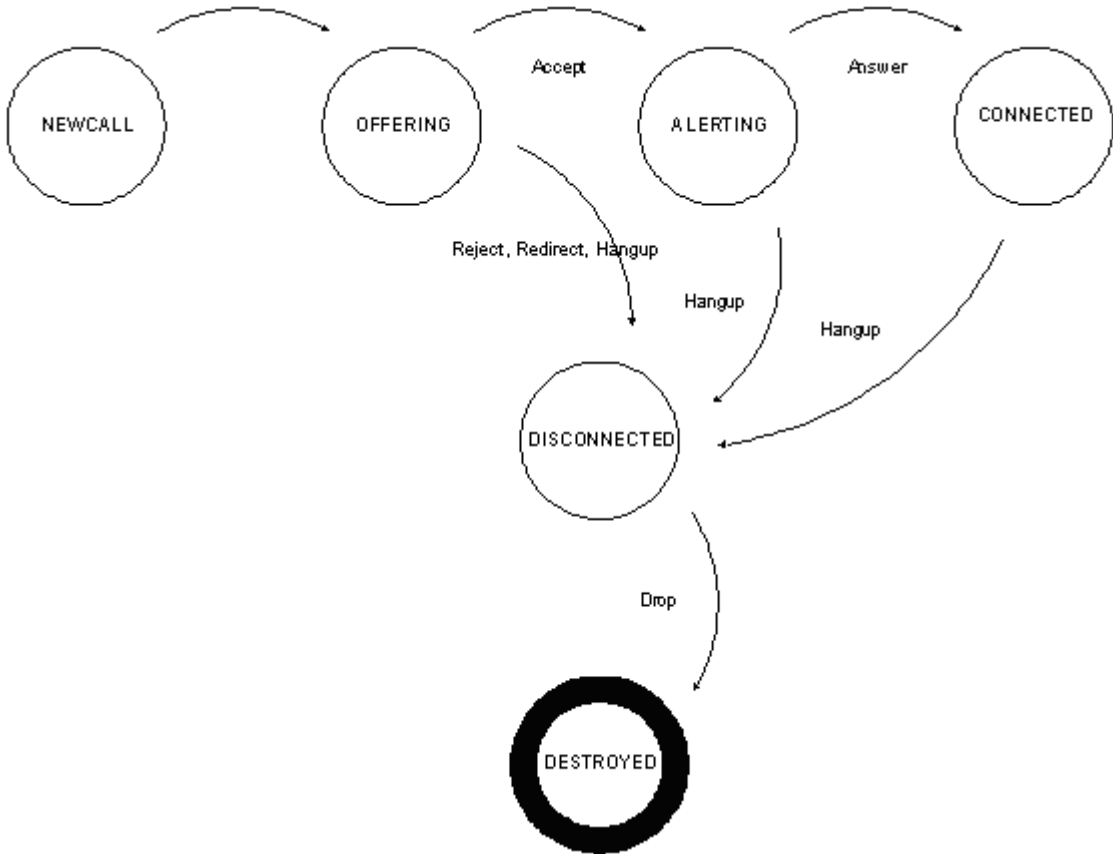

FIGURA 38 Eventos para una llamada entrante

e) Estructuras de Datos

SIPX_CONTACT_ADDRESS : Ésta estructura incluye información de contacto como (IP y puerto), tipo de dirección fuente e interface.

SIPX_CONTACT_TYPE	eContactType *Address type/source.*
char *	cInterface [32] *Source interface.*
char	CIpAddress [32] *IP Address.*
int	Sport *Port.*

Tabla 11 Esrtuctura de datos de SIPX_CONTACT_ADDRESS

b) Lista de archivos

SipXtapi.h Declaraciones principales del API

SipXtapiEvents.h Declaraciones de eventos

ARCHIVO DE REFERENCIA SIPXTAPI.H :

#include <stddef.h>

Iniciación

SIPXTAPI_API SIPX_RESULT sipxInitialize()

Inicia la capa sipX tapi-like API.

Éste método inicia el stack SIP básico y los recursos de procesamiento de medios, y debe ser llamado antes que cualquier otro método de tipo sipxXXX; además. Éste método llena un parámetro SIPX_INST el cual debe ser pasado a un número de métodos sipX.

Parámetros:

- phInst: Un apuntador a hInst que debe ser otras varias rutinas sipX.
- udpPort: El puerto UDP por defecto para el stack del protocolo SIP; el puerto no puede ser cambiado después de la iniciación. Se pasa un valor de 0 (cero) para seleccionar automáticamente un puerto abierto o –1 para deshabilitar UDP.
- tcpPort: El puerto TCP por defecto para el stack del protocolo SIP; el puerto no puede ser cambiado después de la iniciación. Se pasa un valor de 0 (cero) para seleccionar automáticamente un puerto abierto o –1 para deshabilitar TCP.
- tlsPort: **aún no soportado**

- rtpPortStart: El puerto donde se comienza a recibir tráfico RTP. La capa sipX usará puertos comenzando en rtpPortStart y terminando en (rtpPortStart + 2 * maxConnections) - 1. Se pasa un valor de 0 (cero) para seleccionar automáticamente un puerto abierto.
- maxConnections: El número máximo de conexiones simultáneas que la capa sipX soportará.
- szIdentity: La identidad de salida por defecto usada por el satck SIP si no se definen líneas.
- szBindToAddr: Define cual dirección IP escuchará el user-agent sobre el stack RTP. La dirección por defecto "0.0.0.0" escucha todas las interfaces. La dirección debe estar en forma decimal y punteada como se muestra; los nombres de host no funcionarán.

SIPXTAPI_API SIPX_RESULT sipxUnInitialize()

Des-inicia la capa sipX tapi-like API .

Éste método finaliza el stack SIP básico y los recursos de procesamiento de medios y debería llamarse antes de salir del proceso.

Parámetros:

-hInst: Un handle de instancia obtenido de sipxInitialize.

c) Métodos de Llamada

SIPXTAPI_API SIPX_RESULT sipxCallAccept ()

Acepta una llamada entrante y procede inmediatamente a alertar.

Éste método es invocado en respuesta a un evento NEWCALL; cuando se recibe una nueva llamada, el desarrollador de la aplicación debería usar ACCEPT

(proceder a timbrar), REJECT (enviar mensaje de ocupado), o REDIRECT para redireccionar la llamada.

Parámetros:

- hCall: Handle a una llamada. Los handles de una llamada se obtienen al invocar sipxCallCreate o son pasados a la aplicación a través de una interface de escucha.
- contactType

SIPXTAPI_API SIPX_RESULT sipxCallReject()

Rechaza una llamada entrante (antes de alertar al usuario).

Éste método debe ser invocado antes que el usuario final es alertado (antes de sipXcallAccept). Cuando una nueva llamada es recibida, el desarrollador de la aplicación debería usar ACCEPT (proceder a timbrar), REJECT (enviar mensaje de ocupado), o REDIRECT para redireccionar la llamada.

Parámetros:

- hCall

SIPXTAPI_API SIPX_RESULT sipxCallRedirect()

Redirecciona una llamada entrante (antes de alertar al usuario).

Parámetros:

- hCall
- szForwardURL: URL de tipo SIP a donde se reenvía o redirecciona la llamada.

SIPXTAPI_API SIPX_RESULT sipxCallAnswer( )

Responde una llamada en alerta.

Parámetros:

- hCall

SIPXTAPI_API SIPX_RESULT sipxCallCreate()

Crea una nueva llamada con el propósito de crear una conexión o llamada saliente.

Como efecto secundario, un evento DIALTONE se dispara para simular el mundo PSTN. Generalmente una aplicación simularía tonos de marcado en reacción a ése evento.

Parámetros:

- hInst
- hLine: Identidad de línea para la llamada saliente.
- phCall: Apuntador a un handle de llamada. Al haber éxito, éste valor es reemplazado con un handle de llamada válido. El éxito se determina por el código de resultado SIPX_RESULT.

SIPXTAPI_API SIPX_RESULT sipxCallGetLocalContacts()

Consigue la dirección de contacto local disponible para señalización entrante/saliente y audio.

Las direcciones de contacto locales siempre incluirán las direcciones IP locales y también pueden incluir direcciones derivadas externas NAT (tal como STUN).

Parámetros:

- hCall

- addresses: Una lista pre-asignada de estructuras SIPX_CONTACT_ADDRESS. Estos datos se llenarán por la llamada de la API.
- nMaxAddresses: Número máximo de direcciones provistas por el parámetro addresses.
- nActualAddresses

SIPXTAPI_API SIPX_RESULT sipxCallConnect()

Conecta una llamada a una dirección designada como destino.

Parámetros:

- hCall
- szAddress
- contactType

SIPXTAPI_API SIPX_RESULT sipxCallHold ( )

Pone la llamada especificada en espera.

Parámetros:

- hCall

SIPXTAPI_API SIPX_RESULT sipxCallUnhold()

Quita la llamada especificada de espera.

Parámetros:

- hCall

SIPXTAPI_API SIPX_RESULT sipxCallDestroy()

Destruye la llamada especificada.

Parámetros:

- hCall

SIPXTAPI_API SIPX_RESULT sipxCallGetID()

Consigue la identidad (ID) SIP de la llamada representada por el handle llamada especificado.

Parámetros:

- hCall
- szId: Buffer para guardar la ID.
- iMaxLength: Longitud máxima del Buffer ID.

SIPXTAPI_API SIPX_RESULT sipxCallGetLocalID()

Consigue la ID de la conexión local

La ID representa 1) quien fue llamado en caso de una llamada entrante, o 2) la línea ID usada en una llamada saliente

Parámetros:

Los mismos de SIPXTAPI_API SIPX_RESULT sipxCallGetID()

SIPXTAPI_API SIPX_RESULT sipxCallGetRemoteID()

Consigue la identidad SIP de la conexión remota.

Parámetros:

Los mismos de SIPXTAPI_API SIPX_RESULT sipxCallGetID()

SIPXTAPI_API SIPX_RESULT sipxCallStartTone()

Reproduce un tono (DTMF, tono de marcado, timbrado de retorno, etc) en el sistema local o remoto.

Parámetros:

- hCall
- toneId: ID del tono a reproducir.
- bLocal
- bRemote

SIPXTAPI_API SIPX_RESULT sipxCallStopTone()

Para de reproducir un tono en el sistema local o remoto.

Parámetros:

- hCall

SIPXTAPI_API SIPX_RESULT sipxCallPlayFile()

Reproducir el archivo designado.

El archivo puede ser 16 BIT signed PCM a 8000 muestras/segundo, mono, little endian.

SIPXTAPI_API SIPX_RESULT sipxCallBlindTransfer()

Transferencia invisible de la llamada especificada a otro sistema. Advertencia: Aún en desarrollo.

Parámetros:

- hCall
- szAddress

d) Métodos de Conferencia

SIPXTAPI_API SIPX_RESULT sipxConferenceCreate()

Crea un handle de conferencia.

SIPXTAPI_API SIPX_RESULT sipxConferenceJoin()

Une o adhiere una llamada existente a una conferencia (aún no implementado).

SIPXTAPI_API SIPX_RESULT sipxConferenceSplit()

Quita una llamada existente de una conferencia (aún no implementado).

SIPXTAPI_API SIPX_RESULT sipxConferenceAdd()

Adherir un nuevo participante a una conferencia existente.

SIPXTAPI_API SIPX_RESULT sipxConferenceRemove()

Quita un participante de la conferencia al colgarles la llamada.

SIPXTAPI_API SIPX_RESULT sipxConferenceGetCalls()

Pone a todas las llamadas participantes en una conferencia.

SIPXTAPI_API SIPX_RESULT sipxConferenceDestroy()

Destruye una conferencia.

e) Métodos de Audio

SIPXTAPI_API SIPX_RESULT sipxAudioGetInputDevice()

Consigue el nombre o identificador del dispositivo de entrada en la posición index.

SIPXTAPI_API SIPX_RESULT sipxAudioGetNumInputDevices()

Consigue el número de dispositivos de entrada disponibles en el sistema.

SIPXTAPI_API SIPX_RESULT sipxAudioGetNumOutputDevices()

Consigue el número de dispositivos de salida disponibles en el sistema.

SIPXTAPI_API SIPX_RESULT sipxAudioGetOutputDevice()

Consigue el nombre o identificador del dispositivo de salida en la posición index.

SIPXTAPI_API SIPX_RESULT sipxAudioSetCallInputDevice()

Fija el dispositivo de entrada de la llamada (micrófono).

SIPXTAPI_API SIPX_RESULT sipxAudioSetCallOutputDevice()

Fija el dispositivo de salida de la llamada (parlante/altavoz).

SIPXTAPI_API SIPX_RESULT sipxAudioSetRingerOutputDevice()

Fija el dispositivo de timbrado/alertado de llamada.

Métodos De Línea / Identidad

SIPXTAPI_API SIPX_RESULT sipxLineAdd()

Adhiere una línea aspecto (appearance).

Una línea aspecto define la dirección como se guarda y es usada como el identificador caller-id “From” y como la identidad pública en la cual se reciben llamadas.

Parámetros:

- hInst

- szLineURL: Dirección para la identidad línea.
- bRegister: El user-agent debería registrar automáticamente esta línea aspecto? A menos que el user-agent sea designado con una dirección IP estática o nombre DNS y ésta información de enrutado sea provista en un servidor SIP, se debe registrar este aspecto.
- phLine: Apuntador a un handle línea. Si hay éxito, un handle a la nueva línea adherida es devuelto.
- contactType

SIPXTAPI_API SIPX_RESULT sipxLineAddCredential()

Adhiere credenciales de autenticación a la línea aspecto designada.

Las credenciales son requeridas frecuentemente por servicios de registro para verificar que la línea esta siendo usada por la línea aspecto/dirección de aquel que posee el registro.

Parámetros:

- hLine: Handle una línea aspecto.
- hLine: Apuntador de instancia obtenido por sipxLineAdd.
- szUserID: ID de usuario para la línea aspecto.
- szPasswd: Contraseña usada por la línea aspecto.
- szRealm: Realm para el cual el usuario y contraseña son válidos.

SIPXTAPI_API SIPX_RESULT sipxLineGet()

Parámetros:

- hInst
- lines: Arregle pre-asignado de handles de línea.

- max: Número máximo de líneas a retornar o devolver.
- actual: Número de líneas válidas devuelto.

SIPXTAPI_API SIPX_RESULT sipxLineGetURI()

Consigue la URI de la línea para el handle de línea designado.

Parámetros:

- hLine
- szBuffer
- nBuffer
- nActual

SIPXTAPI_API SIPX_RESULT sipxLineRemove()

Quita la línea aspecto designada.

Parámetros:

- hLine

f) Métodos de configuración

SIPXTAPI_API SIPX_RESULT sipxConfigDisableStun()

Deshabilitar el uso de STUN. Ver sipxConfigEnableStun para detalles sobre STUN.

Parámetros:

- hInst

SIPXTAPI_API SIPX_RESULT sipxConfigEnableLog()

Este método permite registrarse (logging) en sipXtapi API, procesamiento de medios y de llamada, stack SIP, y capa de abstracción OS. Registrarse esta deshabilitado por defecto. El marco de trabajo subyacente no hace intentos para limitar el archivo de registro a un tamaño fijo.

Parámetros:

- szFileName
- logLevel

SIPXTAPI_API SIPX_RESULT sipxConfigEnableRport()

Habilita o no el uso de “rport”.

Parámetros:

- hInst
- bEnable

SIPXTAPI_API SIPX_RESULT sipxConfigEnableStun()

Permite soporte STUN (Simple Traversal of UDP through NAT) para señalización UDP SIP y UDP audio/video (RTP). STUN ayuda a los user-agent a determinar su dirección IP externa desde dentro de un NAT/Firewall. Éste método debería ser invocado inmediatamente después de llamar sipxInitialize y antes de crear cualquier línea o llamada.

SIPXTAPI_API SIPX_RESULT sipxConfigSetDnsSrvFailoverTimeout()

Especifica el tiempo a esperar antes de intentar el siguiente registro DNS SRV

SIPXTAPI_API SIPX_RESULT sipxConfigSetDnsSrvTimeouts()

Modifica los valores de tiempo (time out) usados por las búsquedas DNS SRV

SIPXTAPI_API SIPX_RESULT sipxConfigSetMicAudioHook()

Designa una rutina callback como reemplazo o suplemento del micrófono. La rutina es invocada con la información (datos) desde el micrófono y ésta información puede ser revisada, modificada, reemplazada o descartada.

Esta rutina callbackproc no debe bloquearse y debe retornar la información rápidamente. Además, éste método no debe llamar ninguna función de bloqueo (como operaciones IO, malloc, new, etc). Los datos deber estar formados como mono 16-BIT signed PCM, little endian, 8000 muestras por segundo. A la rutina callback se le dan 80 muestras (10ms) de datos cada vez.

SIPXTAPI_API SIPX_RESULT sipxConfigSetOutboundProxy()

Define el Proxy SIP usado para peticiones salientes

SIPXTAPI_API SIPX_RESULT sipxConfigSetRegisterExpiration()

Especifica la expiración del periodo de registro. Después de fijar ésta configuración, los siguientes mensajes REGISTER serán enviados con el nuevo periodo de registro especificando el periodo de expiración (en segundos) para el registro.

SIPXTAPI_API SIPX_RESULT sipxConfigSetSpkrAudioHook()

Designa una rutina callback para post-mezclar datos de audio (por ejemplo: datos dirigidos hacia el parlante). Éste gancho (hook) puede revisar, modificar, reemplazar, o descartar datos. Esta rutina callbackproc no debe bloquearse y debe retornar la información rápidamente. Además, éste método no debe llamar ninguna función de bloqueo (como operaciones IO, malloc, new, etc). Los datos deber estar formados como mono 16-BIT signed PCM, little endian, 8000 muestras por segundo. A la rutina callback se le dan 80 muestras (10ms) de datos cada vez.

SIPXTAPI_API SIPX_RESULT sipxConfigSetSubscribeExpiration()

Especifica el periodo (en segundos) para la suscripción. Después de fijar ésta configuración, todos los siguientes mensajes SUBSCRIBE serán enviados con el nuevo periodo de suscripción.

g) Documentación de Typedef

typedef void(* fnMicAudioHook)(const int nSamples, short *pSamples)

Typedef para el procedimiento de gancho de la fuente de audio (micrófono). Éste typedef junto con el API sipxConfigSetMicAudioHook, permite a los desarrolladores ver, modificar o sustituir datos de micrófono.

Parámetros:

- nSamples: Número de muestras tipo 16 BIT unsigned PCM.
- pSamples: Apuntador al arreglo de muestras.

typedef void(* fnSpkrAudioHook)(const int nSamples, short *pSamples)

Typedef para el procedimiento de gancho de audio destino(parlante). Éste typedef junto con el API sipxConfigSetSpkrAudioHook, permite a los desarrolladores interceptar y modificar audio destinado para el parlante.

Parámetros:

Los mismos que typedef void(* fnMicAudioHook)(const int nSamples, short *pSamples).

typedef unsigned int SIPX_CALL

El handle SIPX_CALL representa una llamada o conexión entre el user-agent y otro sistema. Todas las operaciones de llamada requieren al handle llamada como parámetro.

typedef unsigned int SIPX_CONF

El handle SIPX_CONF representa una colección de CALLs que tienen audio puenteado (mezclado). Los desarrolladores de la aplicación pueden manipular capa parte de la conferencia a través de varias funciones de conferencia.

typedef void* SIPX_INST

El handle SIPX_INST representa una instancia de un user-agent. Un user-agent incluye el stack SIP y un marco de trabajo (infraestructura) de procesamiento de medios. SipXtapi soporta múltiples instancias de user-agents en el mismo espacio de proceso, sin embargo, ciertas características de procesamiento de medios se vuelven limitadas o ambiguas; por ejemplo, sólo un user-agent debe controlar los dispositivos de entrada y salida del sistema local.

typedef unsigned int SIPX_LINE

El handle SIPX_LINE representa una identidad entrante o saliente. Cuando se hacen llamadas salientes, el programador de la aplicación debe definir la línea saliente. Cuando se reciben llamadas entrantes, la aplicación puede preguntar por la línea.

typedef enum SIPX_LOG_LEVEL SIPX_LOG_LEVEL

Existen varios niveles de registro (log) disponibles para el método sipxConfigEnableLog. Los desarrolladores pueden escoger la cantidad de detalles disponibles en el registro; cada nivel incluye mensajes generados en bajos niveles; por ejemplo, LOG_LEVEL_EMERG limitará el registro a mensajes de emergencia, mientras que LOG_LEVEL_ERR incluye mensajes de emergencia, alerta, críticos y errores. LOG_LEVEL_ERR es probablemente mejor para situaciones generales de ejecución. LOG_LEVEL_INFO o LOG_LEVEL_DEBUG es mejor para diagnosticar problemas.

typedef enum SIPX_RESULT SIPX_RESULT

Códigos de resultado y retorno (devolución) de sipX api.

typedef enum SPEAKER_TYPE SPEAKER_TYPE

Los tipos de salida del altavoz se usan para diferenciar entre el timbrado lógico (usado para alertar al usuario de una llamada entrante) del altavoz (dispositivo de audio de entrada de llamada) .

typedef enum TONE_ID TONE_ID

Identificaciones para tonos DTMF y otros usadas con sipxCallStartTone / sipxCallStopTone.

h) Documentación del tipo de Enumeración

enum SIPX_CONTACT_TYPE

SIPX_CONTACT_TYPE es una enumeración de posibles tipos de direcciones para usar con contactos SIP e información de conexión SDP.

enum SIPX_LOG_LEVEL

Existen varios niveles de registros disponibles para el método sipxConfigEnableLog.

enum SIPX_RESULT

Códigos de resultado y retorno (devolución) de sipX api.

enum SPEAKER_TYPE

Los tipos de salida del altavoz se usan para diferenciar entre el timbrado lógico (usado para alertar al usuario de una llamada entrante) del altavoz (dispositivo de audio de entrada de llamada) .

enum TONE_ID

Identificaciones para tonos DTMF y otros usadas con sipxCallStartTone / sipxCallStopTone.

i) Archivo de Referencia SIPXTAPIEVENTS.H :

#include "sipXtapi.h"

Documentación de funciones

SIPXTAPI_API char* sipxEventToString()

Crea una versión imprimible tipo string de las identificaciones de eventos de estado de llamada. Esto generalmente se usa para hacer debug.

Parámetros:

- eMajor: Código de un evento mayor de SipXtapi.
- eMinor: Código de un evento menor de SipXtapi.
- szBuffer: Buffer para guardar el string del evento.
- nLength: Longitud del buffer del string szBuffer.

SIPXTAPI_API char* sipxLineEventToString()

Crea una versión imprimible tipo string de identificaciones designadas de eventos de línea. Esto generalmente se usa para hacer debug.

Parámetros:

- lineTypeMajor: Tipo de ID de un evento mayor.
- szBuffer
- nLength

SIPXTAPI_API SIPX_RESULT sipxLineListenerAdd ()

Adhiere un callback u observador con el propósito de recibir eventos de Línea.

Parámetros:

- hInst
- pCallbackProc: Función para recibir eventos sipx.
- pUserData: Datos del usuario pasados junto con datos del evento.

SIPXTAPI_API SIPX_RESULT sipxLineListenerRemove ()

Quita un callback/observador de eventos de Línea. Provee los mismos valores pCallbackProc y pUserData como sipxLineListenerAdd.

Parámetros:

- hInst
- pCallbackProc:
- pUserData: Datos del usuario especificados como parte de sipxListenerAdd.

SIPXTAPI_API SIPX_RESULT sipxListenerAdd ()

Adhiere un callback/observador con el propósito de recibir eventos de llamada

Parámetros:

Los mismos que SIPXTAPI_API SIPX_RESULT sipxLineListenerAdd ()

SIPXTAPI_API SIPX_RESULT sipxListenerRemove ()

Quita un callback/observador de eventos de llamada. Provee los mismos valores pCallbackProc y pUserData como sipxLineListenerAdd.

Parámetros:

Los mismos que SIPXTAPI_API SIPX_RESULT sipxLineListenerRemove ()

j) Documentación de Typedef

Typedef void (* CALLBACKPROC) ()

Firma para un callback/observador de eventos. Los desarrolladores de la aplicación no debe bloquear el hilo de llamada.

Parámetros:

- hCall: Fuente del evento.
- hLine: Una llamada con Línea fue hecha.
- eMajor: ID de un evento mayor.
- eMinor: ID de un evento menor.
- pUserData: Datos del usuario provistos cuando se adhirió un listener.

Typedef void (* LINECALLBACKPROC) ()

Firma para un callback/observador de eventos de Línea. Los desarrolladores de la aplicación no debe bloquear el hilo de llamada.

Parámetros:

Los mismos que typedef void (* CALLBACKPROC) ()

typedef enum SIPX_CALLSTATE_MAJOR SIPX_CALLSTATE_MAJOR

Los eventos mayores de estados de llamada identifican cambios significativos en el estado de la llamada.

typedef enum SIPX_CALLSTATE_MINOR SIPX_CALLSTATE_MINOR

Los eventos menores de estados de llamada identifican la razón de ser de un evento SIPX_CALLSTATE_MAJOR o proveen más detalle.

typedef enum SIPX_LINE_EVENT_TYPE_MAJOR SIPX_LINE_EVENT_TYPE_MAJOR

k) Enumeración de posibles eventos de Línea.

Documentación de tipo de Enumeración.

enum SIPX_CALLSTATE_MAJOR

Los eventos mayores de estados de llamada identifican cambios significativos en el estado de la llamada.

enum SIPX_CALLSTATE_MINOR

Los eventos menores de estados de llamada identifican la razón de ser de un evento SIPX_CALLSTATE_MAJOR o proveen más detalle.

enum SIPX_LINE_EVENT_TYPE_MAJOR

Enumeración de posibles eventos de Línea.

3.4.2 CÓDIGO DE REFERENCIA DEL ESTÁNDAR P.862 – PESQ

El código de referencia del estándar P.862, tiene restringido su uso para ciertas aplicaciones, los usos permitidos, se encuentran incluidos en el software, en donde cada usuario puede observar el origen de ese código.

A continuación se enumeran algunas de las funciones con sus respectivos parámetros de entrada y salida.

Primero se definen dos estructuras, una recibe los datos de cada archivo .wav, y la otra va guardando los estados de error a través de los cálculos:

```
typedef struct {

  char  path_name[512];
```

```
  char  file_name [128];
  long  Nsamples;
  long  apply_swap;
  float * data;
  float * VAD;
  float * logVAD;
} SIGNAL_INFO;
typedef struct {
  long Nutterances;
  long Largest_uttsize;
  long Nsurf_samples;
  long  Crude_DelayEst;
  float Crude_DelayConf;
  long  UttSearch_Start[MAXNUTTERANCES];
  long  UttSearch_End[MAXNUTTERANCES];
  long  Utt_DelayEst[MAXNUTTERANCES];
  long  Utt_Delay[MAXNUTTERANCES];
  float Utt_DelayConf[MAXNUTTERANCES];
  long  Utt_Start[MAXNUTTERANCES];
```

```
  long  Utt_End[MAXNUTTERANCES];

  float pesq_mos;

  float subj_mos;

  int   cond_nr;

} ERROR_INFO;
```

- *ALINEAMIENTO DE TIEMPO*

```
double align_filter_dB [26] [2] = {{0.,-500}, {50., -500}, {100., -500}, {125., -500}, {160., -500}, {200., -500}, {250., -500}, {300., -500}, {350.,  0}, {400.,  0}, {500.,  0}, {600.,  0},{630.,  0}, {800.,  0}, {1000., 0}, {1250., 0}, {1600., 0}, {2000., 0}, {2500., 0}, {3000., 0}, {3250., 0}, {3500., -500}, {4000., -500}, {5000., -500}, {6300., -500}, {8000., -500}};

double standard_IRS_filter_dB [26] [2] = {{  0., -200}, { 50., -40},{100., -20},{125., -12},{160., -6},{200.,  0},{250.,  4},{300.,  6},{350.,  8},{400., 10},{500., 11}, {600., 12},{700., 12},{800., 12},{1000., 12},{1300., 12},{1600., 12},{2000., 12},{2500., 12},{3000., 12},{3250., 12},{3500., 4},{4000., -200},{5000., -200},{6300., -200},{8000., -200}};

void fix_power_level (SIGNAL_INFO *info, char *name, long maxNsamples)

void calc_VAD( SIGNAL_INFO * sinfo )

void utterance_locate( SIGNAL_INFO * ref_info, SIGNAL_INFO * deg_info, ERROR_INFO * err_info, float * ftmp )

void id_utterances (SIGNAL_INFO * ref_info, SIGNAL_INFO * deg_info, ERROR_INFO * err_info)
```

void crude_align(SIGNAL_INFO * ref_info, SIGNAL_INFO * deg_info, ERROR_INFO * err_info, long Utt_id, float * ftmp)

void time_align(SIGNAL_INFO * ref_info, SIGNAL_INFO * deg_info, ERROR_INFO * err_info, long Utt_id, float * ftmp)

- *MODELO PERCEPTIVO Y COGNITIVO*

float Whanning [Nfmax];

double pow_of (const float * const x, long start_sample, long stop_sample, long divisor)

power_ref = (float) pow_of (ref_info-> data, SEARCHBUFFER * Downsample, maxNsamples - SEARCHBUFFER * Downsample + DATAPADDING_MSECS * (Fs / 1000), maxNsamples - 2 * SEARCHBUFFER * Downsample + DATAPADDING_MSECS * (Fs / 1000));

power_deg = (float) pow_of (deg_info-> data, SEARCHBUFFER * Downsample, maxNsamples - SEARCHBUFFER * Downsample + DATAPADDING_MSECS * (Fs / 1000), maxNsamples - 2 * SEARCHBUFFER * Downsample + DATAPADDING_MSECS * (Fs / 1000));

void RealFFT(float *x, unsigned long N)

void RealIFFT(float *x, unsigned long N)

void freq_warping (int number_of_hz_bands, float *hz_spectrum, int Nb, float *pitch_pow_dens, long frame)

float pseudo_Lp (int n, float *x, float p)

#define MAX_SCALE 5.0

#define MIN_SCALE 3E-4

void intensity_warping_of (float *loudness_dens, int frame, float *pitch_pow_dens)

void multiply_with_asymmetry_factor (float *disturbance_dens, int frame, const float * const pitch_pow_dens_ref, const float * const pitch_pow_dens_deg)

disturbance_dens_asym_add = (float *) safe_malloc (Nb * sizeof (float));

deadzone = (float *) safe_malloc (Nb * sizeof (float));;

#define THRESHOLD_BAD_FRAMES 30

#define MINIMUM_NUMBER_OF_BAD_FRAMES_IN_BAD_INTERVAL 5

err_info-> pesq_mos = (float) (4.5 - D_WEIGHT * d_indicator - A_WEIGHT * a_indicator);

3.5 MANUAL DEL USUARIO

El programa comprende dos ventanas básicas de operación, la primera es el teléfono de VoIP en donde encontramos los siguientes elementos:

- (1) Un menú de WinPesq en el frame del programa.
- (2) Un menú de Ayuda en el frame del programa.
- (3) Un cuadro que nos indica el estado en el que se encuentra el teléfono.
- (4) Un cuadro de edición en donde se pone la dirección IP del Terminal remoto al que se va a conectar denominado “Marcar”.
- (5) Un botón contiguo al cuadro de direcciones IP para marcar.
- (6) Dos barras para controlar el volumen del micrófono y de los parlantes.
- (7) Un botón para colgar o descolgar para terminar o para admitir una llamada entrante respectivamente.
- (8) Un teclado alfanumérico.

- (9) Un selector de función a realizar cuando se establezca una llamada. (Ninguno, reproducir o grabar).
- (10) Un cuadro de edición en donde se debe poner el nombre del archivo en donde grabar o reproducir respectivamente.
- (11) Un botón contiguo que abre el menú de abrir archivos .WAV.
- (12) Un cuadro de edición en donde se deben poner el tiempo de grabación, en segundos, que se requiere.
- (13) Dos botones, uno para grabar y otro para reproducir en cualquier momento durante una llamada ya establecida.

Todos estos elementos y su configuración se muestran en la siguiente figura.

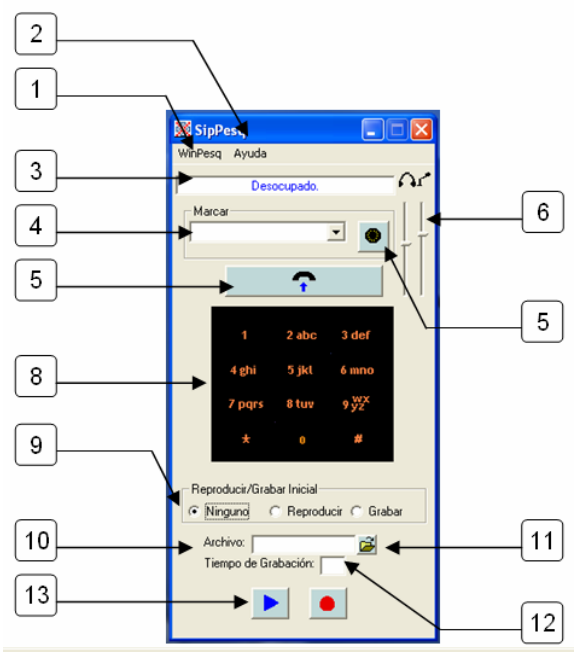

FIGURA 39 Interfaz gráfica del Teléfono de VoIP

Para efectuar una llamada se debe poner la dirección IP del Terminal con el que se quiere establecer comunicación en el cuadro “Marcar” y luego presionar el botón contiguo. En ese momento el cuadro superior indica que el estado en el que

se encuentra es “Marcando”. El selector de función por defecto está en “Ninguno” para indicar que se realizará una llamada normal.

FIGURA 40 Teléfono VoIP marcando a la dirección IP 10.5.7.50

Cuando se quiere realizar un análisis PESQ, se deben seguir los pasos que se enumeran a continuación.

- Primero se debe poner el selector de función en “Reproducir” en uno de los terminales. Esto se ilustra a continuación.

FIGURA 41 Selector de función en Reproducir en uno de los terminales

- Luego se debe abrir el archivo que se quiere reproducir, esto se hace presionando el botón “abrir” el cual despliega un menú en donde se pueden buscar el archivo correspondiente. El archivo debe tener extensión .wav para poderlo reproducir.

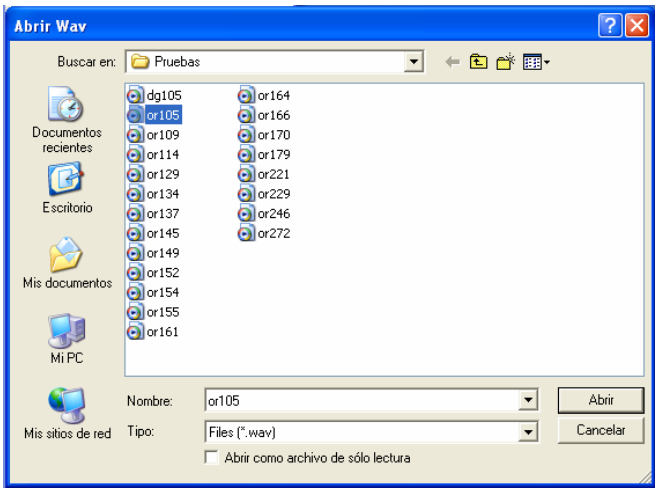

FIGURA 42 Menú para abrir el archivo que se quiere reproducir

A continuación se muestra el teléfono después de efectuar esta operación, debe notarse que ya aparece el nombre del archivo escogido en el cuadro “Archivo”.

FIGURA 43 Teléfono IP con el nombre del archivo que se va a reproducir

- Ahora en el otro Terminal, se debe poner la función contraria “Grabar” en el selector de función.

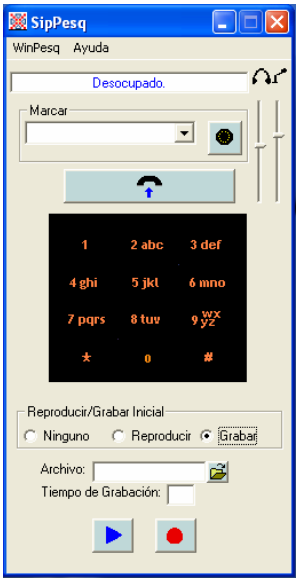

FIGURA 44 Selector de función en Grabar en el otro Terminal

A continuación se debe escoger el archivo donde se quiere grabar, esto se hace desplegando el menú de “Abrir archivo”. Si no se quiere sobre escribir un archivo existente, simplemente se pone un nombre en el cuadro “Nombre” y se presiona el botón “Abrir”. El programa creará un archivo nuevo con el nombre especificado.

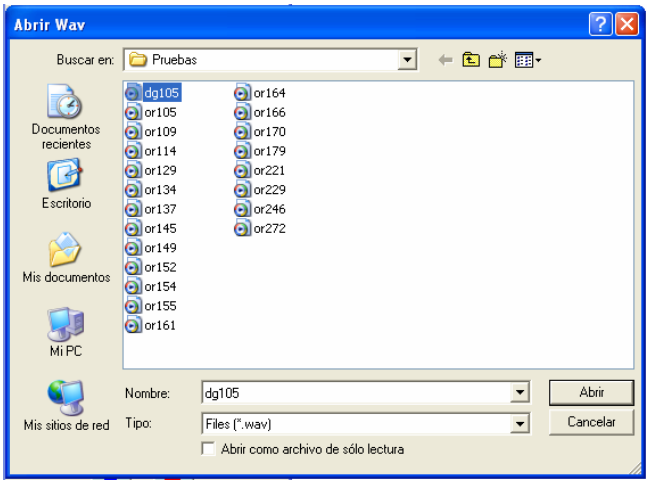

FIGURA 45 Menú para abrir el archivo en el que se va a grabar

A continuación se muestra el teléfono después de efectuar esta operación, debe notarse que ya aparece el nombre del archivo escogido en el cuadro “Archivo”. En el Terminal que va a ejecutar la grabación se debe hacer un paso adicional el cual es poner el tiempo, en segundos, durante el que se quiere grabar. Este valor deber ser un número entero no mayor a 50.

FIGURA 46 Teléfono IP con el nombre del archivo en el que se va a grabar

- Luego se presiona el botón de “Marcar” en cualquiera de los dos terminales y se espera a que se establezca la llamada. Una vez estén conectados, cada uno ejecutará la tarea previamente indicada.

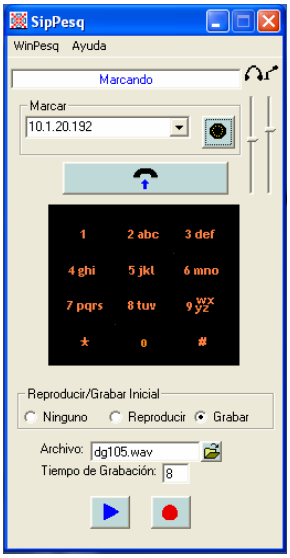

FIGURA 47 Teléfono IP configurado para grabar cuando se establezca la llamada.

Finalmente para completar el análisis PESQ, se despliega el menú de WinPesq con el título que aparece en el frame del programa, a continuación se muestra la ventana y cada uno de sus componentes.

- (1) Cuadro de texto que arroja el resultado.
- (2) Campo para indicar el archivo original para hacer el análisis.
- (3) Campo para indicar el archivo degradado para hacer el análisis.
- (4) Botón de salir para cerrar la ventana de WinPesq
- (5) Botón para realizar el análisis una vez indicados los archivos original y degradado.
- (6) Cuadro de texto que indica el uso permitido del código de PESQ del estándar ITU-T P.862.

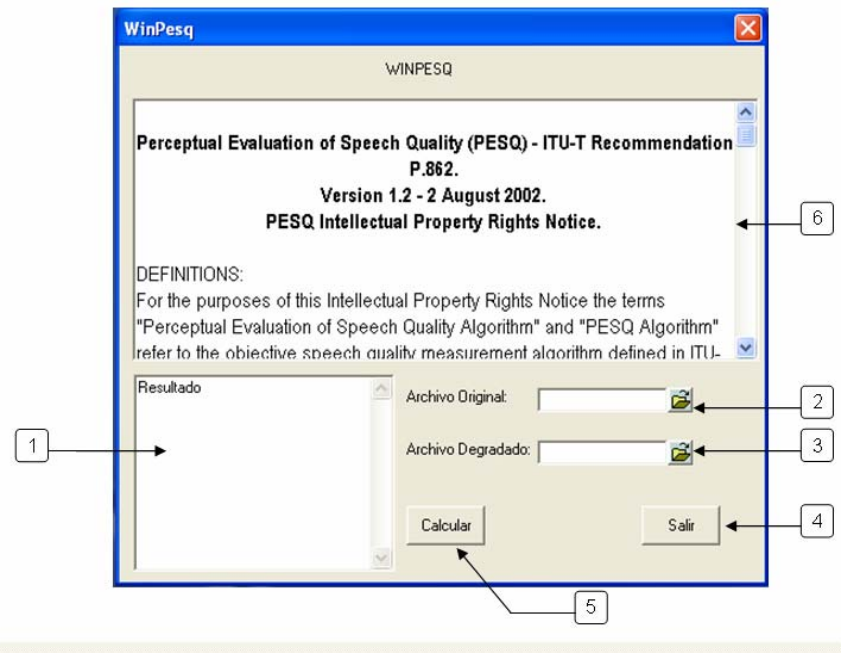

FIGURA 48 Ventana para hacer análisis PESQ.

Se deben indicar los archivos originar y degradado en los campos respectivos.

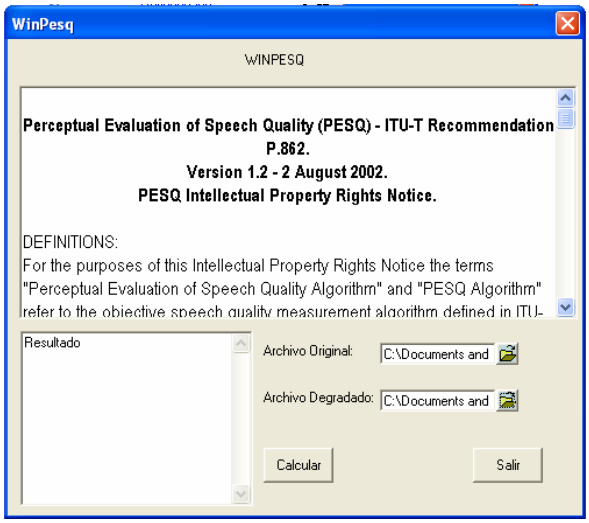

FIGURA 49 Ventana para hacer análisis PESQ con los archivos

Por último se presiona el botón de “Calcular” para que el software calcule el puntaje MOS entre los dos archivos.

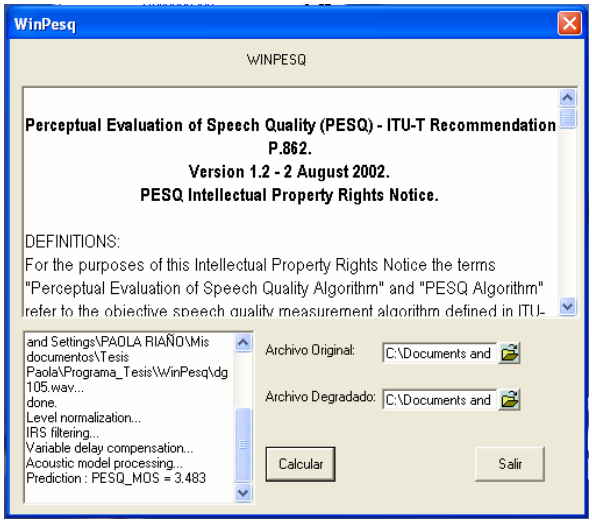

FIGURA 50 Ventana para hacer análisis PESQ con el resultado de PESQ.

4 ANÁLISIS DE RESULTADOS

4.1 RESPECTO A LOS OBJETIVOS

A continuación se enumeran cada uno de los objetivos propuestos en este trabajo de grado con su respectiva sustentación de por que y cómo se alcanzaron.

El objetivo principal fue diseñar e implementar un sistema prototipo de telefonía IP en software, capaz de establecer llamadas punto a punto en una LAN con señalización SIP y predecir la calidad de habla basado en la recomendación ITU-T P.862 (PESQ), el cual fue alcanzado con todo éxito apoyado en los objetivos específico que se nombran más adelante. Como resultado, se llegó a la aplicación denominada SipPesq la cual tiene dos ventanas de operación básicas, la primera es un teléfono IP con señalización SIP que logra establecer llamadas entre dos terminales dentro de una red local y la segunda permite hacer un análisis perceptivo de la calidad del habla comparando dos archivos .wav realizados para este propósito.

El primer objetivo específico fue elaborar una aplicación para la emulación de un teléfono IP, el cual mediante la interfaz multimedia de dos terminales PC, establezca una comunicación de voz a través de una red LAN Ethernet con direcciones de red estáticas, el cual se alcanzó de forma satisfactoria al desarrollar como producto final la aplicación sipPesq con todos los requisitos básicos de un teléfono software, basado en la interfaz de programación sipXtapi SDK de Sipfoundry.

El segundo objetivo específico fue comprobar que la aplicación generara la señalización SIP con un método paso a paso gráfic,o durante una llamada. Éste objetivo se reemplazó por la ventana de estados del teléfono (ocupado, marcando,

desconectado etc), puesto que la aplicación fue desarrollada en particular sobre un stack SIP comprobado, perdiendo sentido la implementación del método gráfico paso a paso.

El tercer objetivo específico fue implementar el algoritmo PESQ siguiendo la recomendación P.862 de la ITU-T utilizando los módulos descritos en la descripción general y los diagramas de bloques, el cual fue alcanzado con éxito debido a que se pudo integrar todo el código que brinda el estándar en la aplicación de teléfono software y evaluar su funcionalidad de forma adecuada bajo la plataforma de programación Visual C++.

El cuarto objetivo específico fue evaluar el desempeño de la implementación del algoritmo PESQ, comparando los resultados con la sección de implementación de referencia y pruebas de validación de la recomendación, las cuales fueron realizadas y se obtuvieron los resultados esperados. Este objetivo fue complementado con el diseño pruebas adicionales que están documentadas previamente en este trabajo.

El quinto objetivo específico se basó en el análisis de forma cualitativa la calidad perceptiva del habla del canal de comunicación a través del algoritmo PESQ, en una escala MOS, lo cual se realizó a través de las pruebas adicionales diseñadas.

El sexto y último objetivo específico fue elaborar los manuales correspondientes de la aplicación desarrollada, para el usuario y el programador los cuales se encuentran incluidos en este trabajo.

4.2 RESPECTO A LAS LIBRERÍAS ESCOGIDAS

Gran parte del trabajo, consistió en la investigación de los desarrollos que existen actualmente en código y librerías para teléfonos de VoIP. Aunque el hecho de

necesitar que la señalización fuera SIP, limitaba un poco el tema, existen varias posibilidades para alcanzar el objetivo propuesto.

Después de observar muchas posibilidades, se llegó a un grupo de tres librerías básicas para el desarrollo del teléfono de VoIP con señalización SIP, LibOSip2, ReSIProcate y SipX, todas con licencia LPGL, de las cuales se escogió trabajar con SipX por tener la mayoría de requisitos necesarios como se mencionó en el capítulo de desarrollos.

En cuanto a las librerías para la implementación de PESQ, fueron escogidas las que brinda el estándar, debido a que el estándar es relativamente nuevo y no hay muchas fuentes de código de donde escoger que cumplan con todas las expectativas esperadas. Además es permitido su uso para fines académicos y evaluativos lo cual fue conveniente para este trabajo específicamente.

4.3 RESPECTO A LAS PRUEBAS

Al comparar las pruebas de los archivos en las primeras gráficas en la sección de resultados, se pudieron apreciar dos factores importantes, el retardo y las diferencias en las formas de onda.

Al escuchar una conversación activa se apreciaba que la media del retardo bruto se mantenía bastante alta (casi 300ms), se ve que éste resultado no concuerda con la topología de la red, es decir, en una red de una única conexión PC a PC la comunicación debería ser inmediata y la pérdida de paquetes teórica es cero; mientras que la nota PESQ se mantiene alta alrededor de 4.2 el cual es un resultado lógico como se explica a continuación.

Al revisar los usos para los cuales se ha demostrado que PESQ tiene una exactitud aceptable como lo son los errores en el canal de transmisión,

transcodificadores, el efecto de la variación del retardo y deformación de la señal de audio en función del tiempo, se comprueba que éste retardo no influye en la nota PESQ por lo que es lógico tener puntajes altos.

El retardo en la conexión que ha sido impuesto, no por la red sino por el teléfono, lo que significa que el tamaño del jitter buffer es demasiado grande, debido a varios posibles factores como la fijación dinámica del tamaño del buffer durante silencios, lo que conlleva a variaciones del retardo en silencios; con lo que se concluye que para evitar el retardo, el jitter buffer debería ser lo más pequeño posible mientras cumpla su función.

Al observar casos de puntajes significativamente bajos se entendió que había ocurrido otro tipo de degradación en éstos archivos y que no dependen de la topología o trafico de la red, por lo que se pasa a analizar cada muestra de audio generalizando estos resultados para una LAN.

A continuación se muestran las gráficas de los archivos que obtuvieron bajos resultados en el puntaje PESQ. En la parte superior se encuentra la gráfica de amplitud versus el tiempo de archivo original e inmediatamente inferior la gráfica de amplitud versus tiempo del mismo archivo grabado por el teléfono en una de las dos terminales cuando hay una llamada activa, es decir el archivo degradado.

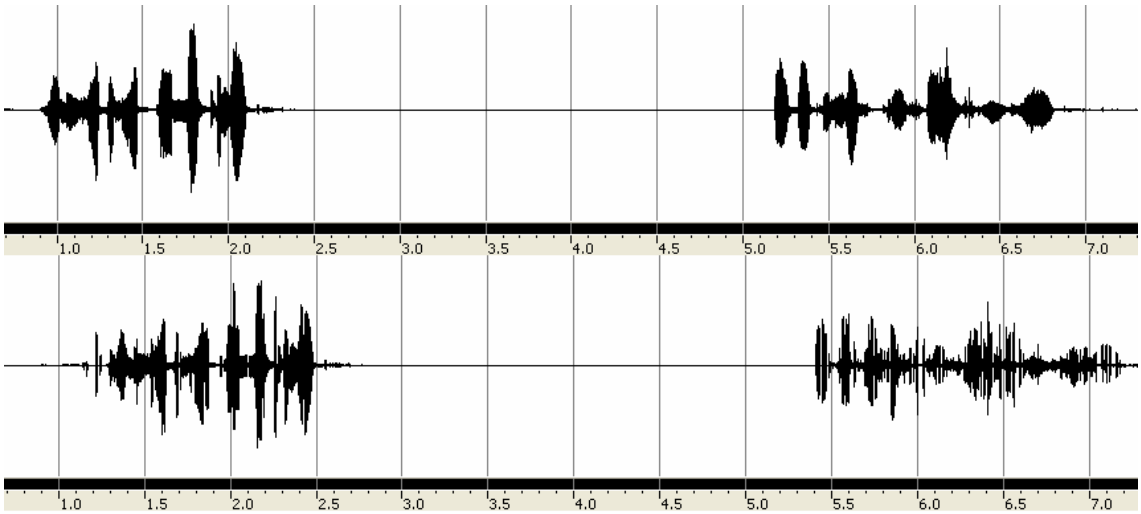

FIGURA 51 Archivo 105 Prueba 10 PESQ = 2.278, Retardo Bruto =0.3800

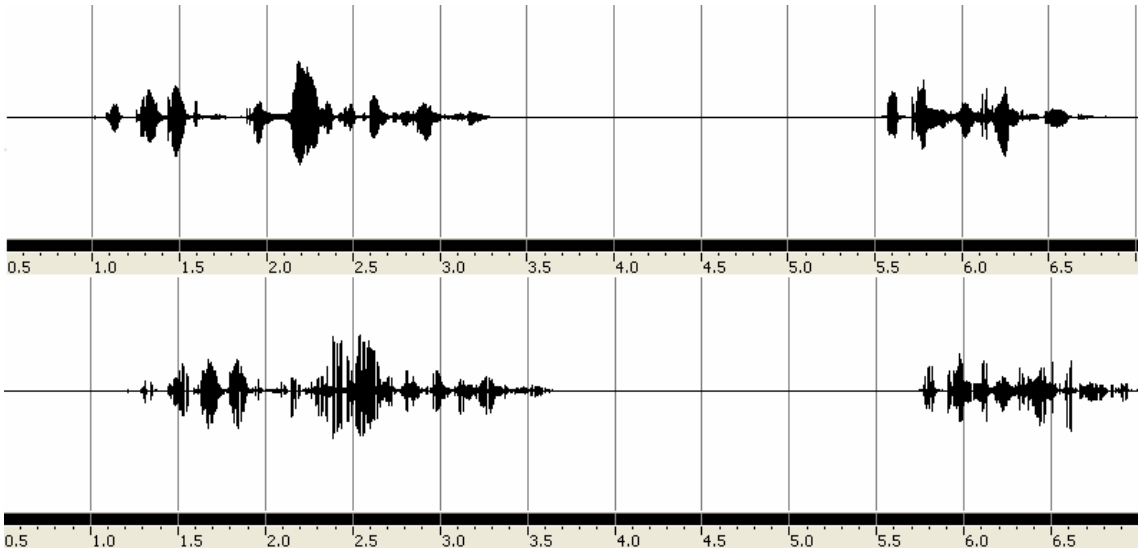

FIGURA 52 Archivo 109 Prueba 9 PESQ = 2.070, Retardo Bruto =0.3600

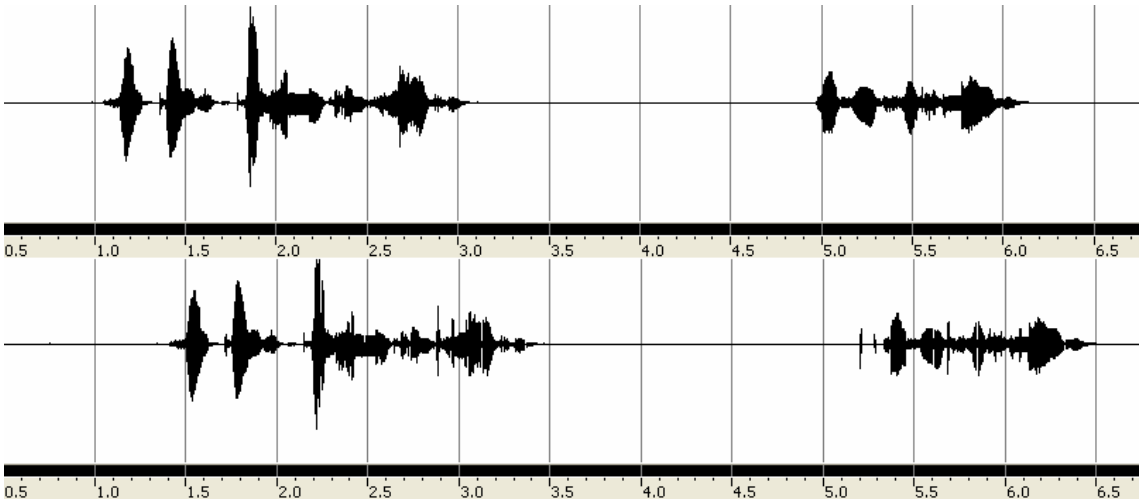

FIGURA 53 Archivo 114 Prueba 1 PESQ =2.673, Retardo Bruto =0.3600

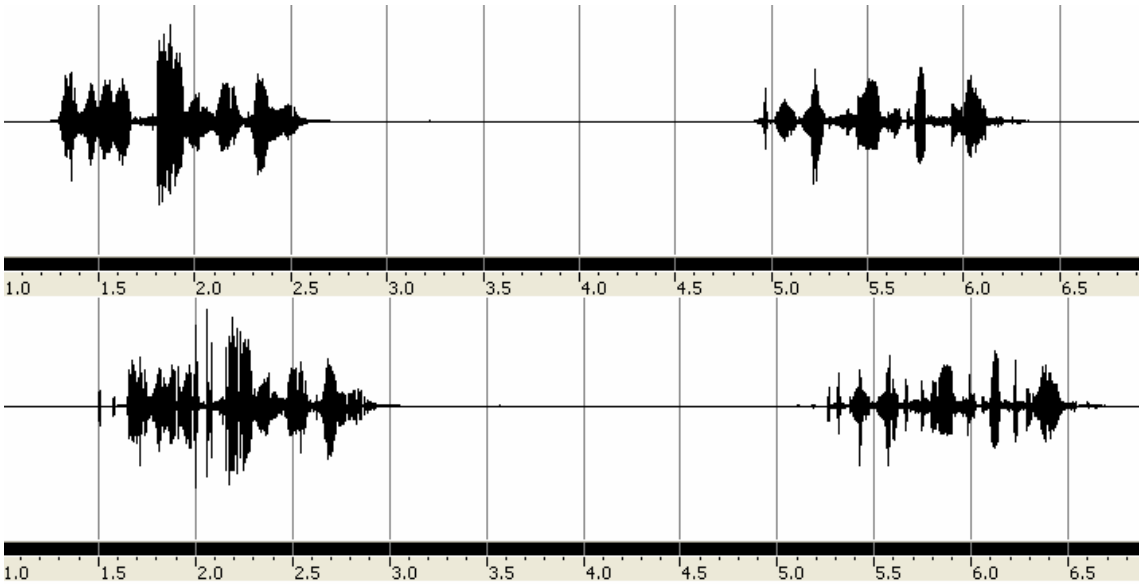

FIGURA 54 Archivo 134 Prueba 3 PESQ = 2.634, Retardo Bruto =0.3480

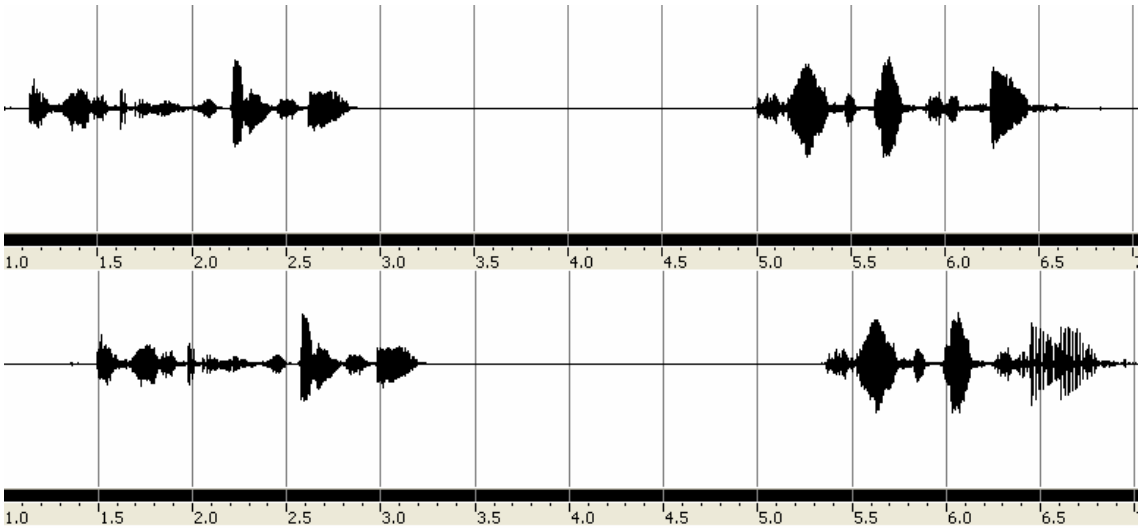

FIGURA 55 Archivo 149 Prueba 9 PESQ = 3.435, Retardo Bruto =0.3600

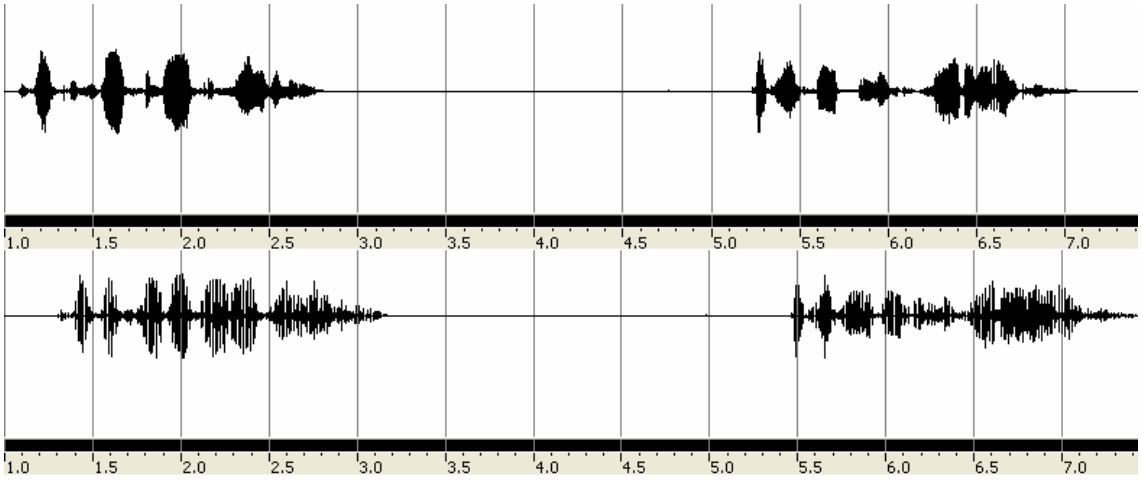

FIGURA 56 Archivo 170 Prueba 2 PESQ = 2.368, Retardo Bruto =0.3800

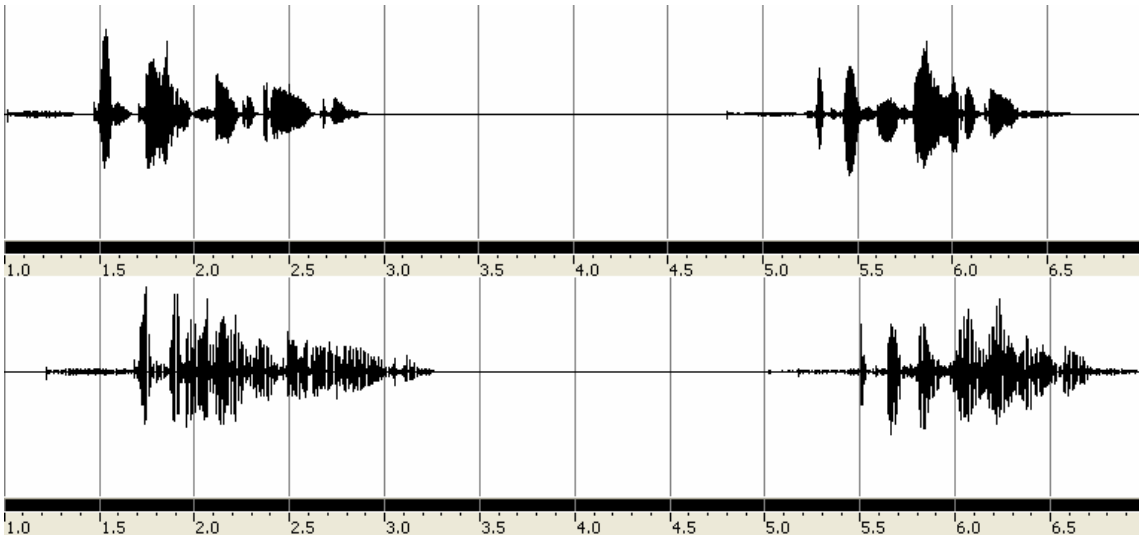

FIGURA 57 Archivo 179 Prueba 2 PESQ = 2.292, Retardo Bruto =0.3720

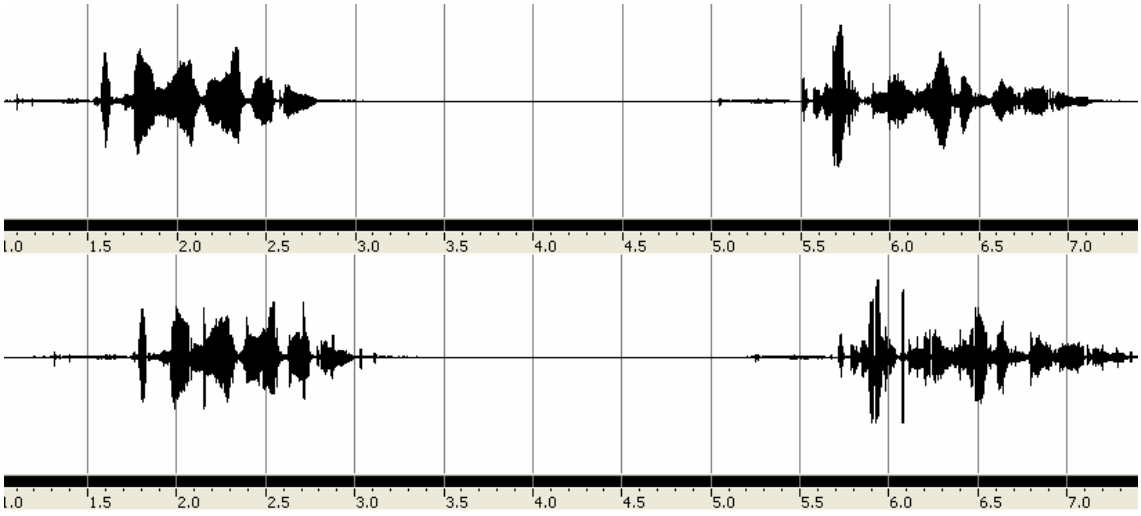

FIGURA 58 Archivo 221 Prueba 9 PESQ = 2.537, Retardo Bruto =0.2080

El segundo punto importante se da al comparar las diferencias en las formas de onda, donde se observa un común denominador en todas las gráficas que concuerda con el resultado del retardo bruto y es éste mismo, el que indica cuales son las características de relevancia, como supresión de silencios y secciones de las articulaciones repetidas, es decir eco. Estos son factores para los cuales

PESQ no ha sido validado; luego el eco se fue computado por el algoritmo como retardo variable y errores en el canal de transmisión y ha dado a concluir que algoritmo PESQ se ha desempeñado bien en su campo y la poca pero dramáticamente baja calidad ha sido causada significativamente en el teléfono por un nivel de supresión de silencios muy alto y el cancelador de eco, definidos junto con el jitter buffer en la librería de procesamiento de medios sipXmediaLib.

4.4 CUADRO DE COSTOS Y COMPARACIÓN

El valor total en costo directo del producto aumentó respeto al que se había planteado en el anteproyecto debido a que el tiempo de desarrollo aumentó en 1 mes y faltó tener en cuenta algunos insumos que fueron necesarios a lo largo del proceso.

No se puede estimar un precio para la venta para el producto porque el software contiene código cuyo uso es restringido para evaluación y fines académicos. Es totalmente prohibido hacer un producto para uso comercial con ese código fuente a menos de que sea solicitada la licencia respectiva.

CUADRO DE COSTOS Y FUENTES DE FINANCIACIÓN PARA EL PROYECTO

Costos Directos					
A.- Potencial Humano	***Unidad***	***Cantidad***	***Valor Unitario***	***Valor Total***	***Fuente***
Director de Tesis	hr	150	$ 60.000	$ 9.000.000	Universidad Javeriana
Asesor de Tesis	hr	120	$ 30.000	$ 3.600.000	Universidad Javeriana
Desarrollador 1	hr	1500	$ 20.000	$ 30.000.000	Recursos propios
Desarrollador 2	hr	1500	$ 20.000	$ 30.000.000	Recursos propios
Valor total potencial Humano				**$ 72.600.000**	
B.- Materiales	***Unidad***	***Cantidad***	***Valor Unitario***	***Valor Total***	***Fuente***

Computador personal con software preinstalado y accesorios.	un	2	$ 1.500.000	$ 3.000.000	Universidad Javeriana
Tarjeta de red para computador personal.	un	2	$ 30.000	$ 60.000	Universidad Javeriana
Micrófono para computador personal.	un	2	$ 10.000	$ 20.000	Universidad Javeriana
Parlantes para computador personal.	jg	2	$ 30.000	$ 60.000	Universidad Javeriana
Impresora para computador personal	un	1	$ 300.000	$ 300.000	Universidad Javeriana
Papel bond tamaño carta.	resma	4	$ 13.000	$ 52.000	Recursos propios
Fotocopias	un	500	$ 50	$ 25.000	Recursos propios
Internet	hr	600	$ 1.500	$ 900.000	Universidad Javeriana
Libros de consulta	un	5	$ 200.000	$ 1.000.000	Biblioteca Central de la Universidad Javeriana

Valor total Materiales	**$ 5.417.000**

VALOR TOTAL COSTO DIRECTO	**$ 78.017.000**

Tabla 12 Cuadro de costos planteado en el Anteproyecto

CUADRO DE COSTOS Y FUENTES DE FINANCIACIÓN PARA EL PROYECTO

Costos Directos

A.- Potencial Humano	***Unidad***	***Cantidad***	***Valor Unitario***	***Valor Total***	***Fuente***
Director de Tesis	hr	100	$ 60.000	$ 6.000.000	Universidad Javeriana
Asesor de Tesis	hr	150	$ 30.000	$ 4.500.000	Universidad Javeriana
Asesor de Tesis	hr	150	$ 30.000	$ 4.500.000	Universidad Javeriana
Desarrollador 1	hr	1800	$ 20.000	$ 36.000.000	Recursos propios
Desarrollador 2	hr	1800	$ 20.000	$ 36.000.000	Recursos propios

Valor total potencial Humano	**$ 87.000.000**

B.- Materiales	***Unidad***	***Cantidad***	***Valor Unitario***	***Valor Total***	***Fuente***

Computador personal con software preinstalado y accesorios.	un	2	$ 2.000.000	$ 4.000.000	Universidad Javeriana
Tarjeta de red para computador personal.	un	2	$ 30.000	$ 60.000	Universidad Javeriana
Diademas con micrófono y audífono	un	2	$ 25.000	$ 50.000	Recursos propios
Impresora para computador personal	un	1	$ 400.000	$ 400.000	Recursos propios
Consumibles Impresora	un	1	$ 200.000	$ 200.000	Recursos propios
Papel bond tamaño carta.	resma	5	$ 15.000	$ 75.000	Recursos propios
Fotocopias	un	500	$ 50	$ 25.000	Recursos propios
Empastes	un	9	$ 6.000	$ 54.000	Recursos propios
Internet	hr	600	$ 1.500	$ 900.000	Universidad Javeriana
Libros de consulta	un	15	$ 200.000	$ 3.000.000	Biblioteca Central de la Universidad Javeriana

Valor total Materiales	**$ 8.764.000**

VALOR TOTAL COSTO DIRECTO	**$ 95.764.000**

Tabla 13 Cuadro de costos final

5 CONCLUSIONES

Los resultados obtenidos en la etapa de pruebas acerca del funcionamiento de la aplicación, permite saber de antemano que la solución propuesta aporta experiencia en el tema de análisis perceptivos de la calidad del habla y la telefonía IP.

Aunque no se implementó toda la capacidad de las librerías de SipX, es importante anotar que a partir de la utilización de más métodos, se puede avanzar más en el desarrollo de otros teléfonos IP y aplicaciones afines.

La implementación de funciones de reproducción y grabación de archivos de voz en tiempo real dentro del teléfono, es una herramienta muy útil que permitió la integración de PESQ y su análisis con archivos enviados a través del mismo teléfono. Eso constituye una ayuda adicional en la que se podría implementar nuevas características como por ejemplo correo de voz.

De acuerdo a las observaciones preliminares realizadas y la evolución de los algoritmos, se puede afirmar que la aplicación cumple con el objetivo general propuesto satisfaciendo las expectativas.

Se pudo comprobar que la respuesta del sistema frente al retardo no es lo suficientemente acertado como debe ser un sistema de comunicación de este tipo. Esta observación indica que la modificación de ciertos métodos de la librería de procesamiento de medios, puede llegar a ser útil para el mejoramiento del comportamiento.

SipPesq como resultado final, puede llegar a utilizarse como una herramienta de análisis de redes y teléfonos IP en cuanto a calidad perceptiva del habla, además

de brindar una aplicación fácil de utilizar para comunicación IP dentro de una red LAN.

Es interesante trabajar con la telefonía IP debido a que se estudian todas las capas que intervienen en un sistema de comunicación de este tipo.

El software SipPesq es un trabajo basado en la investigación, recopilación, selección y desarrollo de una serie de aportes de conocimiento e información relacionados con la telefonía IP y su calidad. Esto no sería posible sin el manejo de los estándares ya que es una gran herramienta que todo ingeniero, independientemente de su afinidad, debe contemplar en su aprendizaje.

Es importante resaltar que el sistema presenta un comportamiento eficiente en la comunicación sobre redes IP y los análisis de las mismas.

6 BIBLIOGRAFÍA Y FUENTES DE INFORMACIÓN

[1] HERSENT, Olivier ; GURLE, David y PETIT, Jean Pierre. IP telephony : packet - based multimedia communications systems. Harlow : Addison Wesley, 2000. 451p. ISBN 0-201-61910-5

[2] DAVIDSON, Jonathan y PETERS, James. Voice over IP fundamentals : a systematic approach to understanding the basics of voice over IP. Indianapolis : Cisco Press, 2000. 349p. ISBN 1-57870-168-6.

[3] KEAGY, Scott. Integrating voice and data networks : practical solutions for the new world of packetized voice over data networks. Indianapolis : Cisco Press, 2000. 752 p.

[4] TAYLOR, Steven. VoIP Special Report VoIP Implementation: Who ’s Doing It, and Why [online]. AVAYA Communication. White Paper. 2002. < http://www.spenser.biz/VoIPImplementationsandWhy.pdf >

[5] AVAYA INC. Avaya IP Voice Quality : Network Requirements. AVAYA Communication. White Paper. Julio 2001.

[6] MORRISSEY, Peter. It's time to take a look at SIP [online]. ProQuest document, Network Computing. Manhasset : Apr 17, 2003. Vol. 14, Iss. 7; pg. 76, 3 p. < http://www.nwc.com/showArticle.jhtml?articleID=15000545>

[7] BORTHICK, Sandra. SIP for the enterprise : Work in progress [online]. ProQuest document, Business Communications Review. Hinsdale : Feb 2003. Vol. 33, Iss. 2; p. 7-20. < http://www.bcr.com/bcrmag/2003/02/p20.php>

[8] WALDRON, Gerard J. y WELCH, Rachel. Voice-over-IP: The Future of Communications[online]. Global Internet Policy Initiative. Washington D.C. : Abril 29 2002. 11p. <www.Internetpolicy.net/practices/voip.pdf >

[9] POUZOLS, Federico Montesino. SIP : Session Initiation Protocol [online]. España : Mayo 2003. 30p. <www.rediris.es/mmedia/gt/gt2003_1/sip-gt2003.pdf>

[10] SISALEM, Dorgham y KUTHAN, Jiri. Understanding SIP [online]. Mobile Integrated Services - GMD Fokus. 187p. <iptel.org/sip/siptutorial.pdf>

[11] RAD COM. Voice over IP [online]. Rad Com : 2004. <www.protocols.com/pbook/VoIPFamily.htm>

[12] SCHULZRINNE, et al. RTP: A Transport Protocol for Real-Time Applications [online]. IETF. Estados Unidos : Julio 2003. 104 p. < www.ietf.org/rfc/rfc3550.txt>

[13] SCHULZRINNE, et al. Real Time Streaming Protocol (RTSP)[online]. IETF. Estados Unidos : Abril 1998. 92 p. <www.ietf.org/rfc/rfc2326.txt>

[14] ROSENBERG, J. SIP: Session Initiation Protocol[online]. IETF. Estados Unidos : Junio 2002. 269 p. <www.ietf.org/rfc/rfc3261.txt>

[15] HANDLEY, M. y JACOBSON, V. SDP: Session Description Protocol [online]. IETF. Estados Unidos : Abril 1998. 42 p. <www.ietf.org/rfc/rfc2327.txt>

[16] VOCAL TECHNOLOGIES LTD. G.711 : Pulse Code Modulation (PCM) of Voice Frequencies [online]. Vocal Technologies ltd. New York : 2004. 1p. <http://www.vocal.com/data_sheets/g711.html>

[17] MONTOYA, Jaime Andrés. Sistema de Diagnóstico de Redes Corporativas para implementación de Telefonía IP. Bogotá : Julio 2003, 237 p. Trabajo de Grado (Ingeniero Electrónico). Pontificia Universidad Javeriana. Facultad de Ingeniería. Carrera de Electrónica.

[18] GONZÁLEZ, Felipe y VÁSQUEZ, Carlos Andrés. Guía de Diseño de Redes de Voz por Paquetes en un Entorno Corporativo. Bogotá : 2001. Trabajo de Grado (Ingeniero Electrónico). Pontificia Universidad Javeriana. Facultad de Ingeniería. Carrera de Electrónica.

[19] CORREDOR, Fernando y ROLDAN, Miguel Ángel. Prácticas de Laboratorio En X.25, Frame Relay E IP. Bogotá : 2001. Trabajo de Grado (Ingeniero Electrónico). Pontificia Universidad Javeriana. Facultad de Ingeniería. Carrera de Electrónica.

[20] MATIZ, Mauricio. Estudio para la implementación del servicio de videoconferencia en la Pontificia Universidad Javeriana. Bogotá : 2003. Trabajo de Grado (Ingeniero Electrónico). Pontificia Universidad Javeriana. Facultad de Ingeniería. Carrera de Electrónica.

[21] VALENZUELA, Germán. Implementación de los servicios de voz y datos vía Frame Relay en una empresa de Colombia. Bogotá : 1999. Trabajo de Grado (Ingeniero Electrónico). Pontificia Universidad Javeriana. Facultad de Ingeniería. Carrera de Electrónica.

[22] INTERNATIONAL TELECOMMUNICATIONS UNION. Recommendation P.862 : Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrow_band TELEPHONE networks and speech codecs. Geneva. 2001.

[23] ANDERSON, John. Methods for Measuring Perceptual Speech Quality [online]. Agilent Technologies. White Paper. Estados Unidos : Octubre 2001. 36 p. < http://itpapers.zdnet.com/whitepaper.aspx?scid=118&docid=13095>

[24] LIESENBORGS, Jori. Voice over IP in networked virtual environments. Alemania : 24 Mayo 2000. Tesis (Knowledge Technology, option Computer Science-Multimedia). School for Knowledge Technology.

[25] HAYKIN Simon. Sistemas de Comunicación. Primera Edición. México: Editorial Limusa Wiley, 2002. 817 p. ISBN 9681863070.

[26] FREESOFT. G.711 Protocol Overview[online].1p. Disponible en: <http://www.freesoft.org/CIE/Topics/127.htm>

[27] COHEN, Mike et. al. Sipfoundry Libraries documentation [online]. <http://www.sipfoundry.org>

[28] AREA DE INGENIERÍA TELEMÁTICA. Tema 1 : El nivel de transporte [online]. España : 2004. Universidad de Oviedo. 17 p. <http://ferre.edv.uniovi.es/material/cursos/InternetNG_EU_072004>

7 ANEXOS

ANEXO A. Diagramas UML de las librerías de SipFoundry

Nota: Refiérase a la carpeta UML anexa en la documentación. Esa carpeta contiene:

Primera parte del diagrama UML de SipXportlib

Segunda parte del diagrama UML de SipXportlib

Diagrama UML de SipXtacklib

Diagrama UML de SipXmedialib

Diagrama UML de SipXcalllib (cp)

Diagrama UML de SipXcalllib (ps)

Diagrama UML de SipXcalllib PTAPI

Diagrama UML de SipXcalllib (tao)

Diagrama UML de SipXcalllib TAPI

Donde Las estructuras Unnamed, tienen los siguientes nombres:

Unnamed49 : AEC_SETTING

Unnamed103 : LINE_LISTENER_DATA

Unnamed73 : SIPX_CALL_DATA

Unnamed36 : MIC_SETTING

Unnamed94 : SIPX_LINE_DATA

Unnamed110 : LISTENER_DATA

Unnamed84 : SIPX_CONF_DATA

Unnamed55 : SIPX_INSTANCE_DATA

Unnamed192 : SIPX_CONTACT_ADDRESS