ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"IMPLEMENTACIÓN DE TRANSACCIONES DE ENVÍO Y RETIRO DE DINERO UTILIZANDO IVR (INTERACTIVE VOICE RESPONSE)"

PROYECTO TÓPICO DE GRADUACIÓN

Previo la obtención del Título de:

INGENIERO EN COMPUTACIÓN
ESPECIALIDAD: SISTEMAS TECNOLÓGICOS

Presentado por:

JAVIER EDUARDO LÓPEZ CASSÁN

GUAYAQUIL – ECUADOR
2007

AGRADECIMIENTO

Deseo exteriorizar mi sincero agradecimiento y reconocimiento al personal docente de la Escuela Superior Politécnica del Litoral, quienes impartieron sabios conocimientos en el transcurso de mi carrera universitaria.

Quiero dejar constancia de mi profundo agradecimiento al Ing. José Escalante pues sin su ayuda y colaboración, todo este trabajo habría sido imposible realizar.

Sin ánimo de olvidar a nadie y en particular a todas aquellas personas que de una u otra manera han compartido mi vida durante el transcurso de estos últimos años mi más sincero agradecimiento, a su comprensión, estimulo y ayuda, por que todos son parte de mi vida.

DEDICATORIA

A Dios todopoderoso y eterno;

A mi hijo Jeremy López;

A Jorge López y Gladys Cassán, mis Padres;

A mis hermanos y amigos.

Las formas y herramientas de trabajo han evolucionado a un ritmo acelerado, sobre todo en los países del primer mundo, vemos en estas últimas décadas el desarrollo vertiginoso de la tecnología. Ésta se encuentra cada vez más palpable en la vida cotidiana de los seres humanos y los beneficios son muchos y muy importantes, ¿Quién puede menospreciar lo útil que resulta el computador, y las posibilidades que ofrece el Internet?.

Al encontrarme viviendo en la era cibernética, es mi deber y responsabilidad brindar mi aporte al desarrollo tecnológico-intelectual de mi país. Nuevas expectativas, ambiciones y retos permitirán desenvolverme en el mundo competitivo cada vez más vulnerable al impacto de la globalización mundial.

Con humildad y sencillez enfrento este nuevo reto profesional y personal en bienestar de la sociedad ecuatoriana.

TRIBUNAL

_____
Ing. Holger Cevallos
SUB-DECANO

_____
Ing. José Escalante
DIRECTOR DE TOPICO

_____
Ing. Edgar Leyton.
VOCAL PRINCIPAL

_____
Ing. Juan Carlos Aviles.
VOCAL PRINCIPAL

DECLARACIÓN EXPRESA

"La responsabilidad por los hechos, ideas y doctrinas expuestas en este trabajo, corresponden exclusivamente al autor; y el patrimonio intelectual de la misma, a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL".

(Reglamento de Exámenes y Títulos Profesionales de la ESPOL).

_____

Javier Eduardo López Cassán

RESUMEN

El presente trabajo trata de brindar soluciones a los siguientes puntos:

- Automatizar el envío de dinero desde usuarios de USA a través de una llamada telefónica la cual interactuará con un servidor IVR en GYE.
- Automatizar el retiro de dinero en Ecuador a través de una llamada telefónica la cual interactuará con un servidor IVR en GYE.
- Optimizar el uso de canales de datos para la comunicación de voz entre sucursales dentro y fuera del País.
- Economizar en llamadas nacionales e internacionales usando infraestructura de enlaces para transmisión de datos instaladas.

CAPÍTULO 1:

En este capítulo se describen los conceptos básicos de las tecnologías a utilizar para llevar a cabo la implementación del presente proyecto; para lo cual introducimos a conceptos voz sobre IP y una de sus aplicaciones como lo es IVR (INTERACTIVE VOICE RESPONSE).

CAPÍTULO 2:

Aquí se detalla todo el proceso actual para llevar a cabo el proceso de envió y retiro de dinero, también se hace notar la necesidad del cliente de poder realizar transacciones de una forma más ágil.

CAPÍTULO 3:

En este capítulo se analizan, desarrollan e implementan el sistema de envió y retiro de dinero.

CAPÍTULO 4:

Se analizan las factibilidades técnicas y comerciales del proyecto para poderlo ofertar.

ÍNDICE GENERAL

APÉNDICES:

ÍNDICE DE TABLAS

ÍNDICE DE GRÁFICOS

INTRODUCCIÓN

El crecimiento y fuerte implantación de las redes IP, tanto en redes locales como en redes remotas, el desarrollo de técnicas avanzadas de digitalización de voz, mecanismos de control y priorización de tráfico, protocolos de transmisión en tiempo real, así como el estudio de nuevos estándares que permitan la calidad de servicio en redes IP, han creado un entorno donde es posible transmitir telefonía sobre IP. Si a todo lo anterior, se le suma el fenómeno Internet, junto con el potencial ahorro económico que éste tipo de tecnologías puede acarrear, la conclusión es clara: El VoIP (Protocolo de voz sobre internet - Voice over internet protocol) es un tema estratégico para las empresas.

La telefonía sobre IP abre un espacio muy importante dentro del universo que es Internet. Es la posibilidad de estar comunicados a costos más bajos dentro de las empresas y fuera de ellas, es la puerta de entrada de nuevos servicios apenas imaginados y es la forma de combinar una página de presentación de Web con la atención en vivo y en directo desde un call center, entre muchas otras prestaciones. Lentamente, la telefonía sobre IP está ganando terreno... y todos quieren tenerla.

Hubo un tiempo en que la voz sobre Internet era un "chiché" más de los tantos que permitía la web. Los estándares eran dudosos y la performance del sistema dejaba mucho que desear. Aun así, muchos carriers en los Estados Unidos vieron amenazado su negocio y trataron de frenar por vías legales el avance de lo que, meses después, se planteaba como "Telefonía sobre Internet".

En 1996 las siglas ACTA y VON (la agrupación de carriers y un organismo llamado Voice On the Net, respectivamente) resumían las posturas en pugna. Sin embargo, los grandes de la telefonía (AT&T y MCI) se mostraban un poco ambiguos a la hora de alinearse con sus colegas: ellos sabían que no había vuelta atrás.

El concepto original es relativamente simple, se trata de transformar la voz en "paquetes de información" manejables por una red IP (con protocolo Internet, que también incluye a las intranets y extranets). Gracias a otros protocolos de comunicación, como el RSVP, es posible reservar cierto ancho de banda dentro de la red que garantice la calidad de la comunicación.

La voz que se obtiene desde un micrófono conectado a la tarjeta de sonido de la PC, o desde un teléfono común: existen gateways (dispositivos de interconexión) que permiten intercomunicar las redes de telefonía tradicional con las redes de datos. El sistema telefónico podría desviar sus llamadas a Internet para que, una vez alcanzado el servidor más próximo al destino, esa llamada vuelva a ser traducida como información analógica y sea transmitida hacia un teléfono común por la red telefónica tradicional. Se pueden mantener conversaciones teléfono a teléfono.

Existen objeciones de importancia, que tienen que ver con la calidad del sistema y con el uptime (tiempo entre fallas) de las redes de datos en comparación con las de telefonía. Sin embargo, la versatilidad y los costos del nuevo sistema hacen que las Telcos estén comenzando a considerar la posibilidad de dar servicios sobre IP y, de hecho aunque todavía el marco regulatorio no lo permite en forma masiva, y a pesar de que difícilmente lo admitan, algunas están realizando pruebas.

CAPÍTULO 1

1. TECNOLOGÍAS

VoIP viene de Voice Over Internet Protocol, intenta permitir que la voz viaje en paquetes IP y obviamente a través de internet o una red de datos.

La telefonía IP conjuga dos mundos históricamente separados: la transmisión de voz y de datos. Se trata de transportar la voz, previamente convertida a datos utilizando las redes de datos para efectuar las llamadas telefónicas, y desarrollar una única red convergente que se encargue de cursar todo tipo de comunicación, sea voz, datos, video o cualquier tipo de información. La voz IP, por lo tanto, no es un servicio, sino una tecnología que permite encapsular la voz en paquetes para poder ser transportados sobre redes de datos sin necesidad de disponer de los circuitos conmutados convencionales PSTN, las redes desarrolladas a lo largo de los años para transmitir las conversaciones vocales, se basaban en el concepto de conmutación de circuitos, o sea, la realización de una comunicación que requiere el establecimiento de un circuito físico durante el tiempo que dura ésta, lo que significa que los recursos que intervienen en la realización de una llamada no pueden ser utilizados en otra hasta que la primera no finalice, incluso durante los silencios que se suceden dentro de una conversación típica. En cambio, la telefonía IP no utiliza circuitos para la conversación, sino que envía múltiples de ellas (conversaciones) a través del mismo canal codificadas en paquetes y flujos independientes. Cuando se produce un

silencio en una conversación, los paquetes de datos de otras conversaciones pueden ser transmitidos por la red, lo que implica un uso más eficiente.

Son evidentes las ventajas que proporciona el segundo tipo de red, que con la misma infraestructura podrían prestar más servicios y además la calidad de servicio y la velocidad serian mayores; pero por otro lado también existe la gran desventaja de la seguridad, que no es posible determinar la duración del paquete dentro de la red hasta que éste llegue a su destino y además existe la posibilidad de perdida de paquetes, por que el protocolo IP no cuenta con esta herramienta.

1.1. Llamadas telefónicas VoIP.

El constante cambio de las telecomunicaciones, la telefonía sobre IP es prometedora ante un mercado global cada vez más competitivo, las compañías telefónicas existentes, los proveedores de servicios de Internet (ISPs), las operadoras locales competitivas emergentes (CLECs) y las PTTs (autoridades de correo, teléfonos y telégrafos) buscan, en forma constante, formas de aumentar sus ofertas de servicios. La telefonía sobre IP ha captado la atención de dichos proveedores de servicios en todo el mundo, ofreciendo una amplia gama de servicios nuevos y reduciendo al mismo tiempo sus costos de infraestructura. La voz sobre IP está cambiando el

paradigma de acceso a la información, fusionando voz, datos, facsímile y funciones multimedia en una sola infraestructura de acceso convergente.

Mediante la telefonía sobre IP, los proveedores de servicios pueden ofrecer servicios de voz básicos y ampliados a través de Internet, incluyendo la llamada en espera en Internet, el comercio en la web por telefonía ampliada y comunicaciones interactivas de multimedia. Estos servicios se integrarán de forma ininterrumpida a las redes conmutadas existentes (PSTN) a fin de permitir que se originen o terminen llamadas en teléfonos tradicionales según sea necesario. Dado que IP es una norma abierta, VoIP le brinda a los proveedores flexibilidad para personalizar sus servicios existentes e implementar nuevos servicios con mayor rapidez y eficiencia en función de los costos.

1.1.1. Como funciona la voz sobre IP.

Años atrás se descubrió que mandar una señal a un destino remoto también podía hacerse de forma digital: antes de enviar la señal se debía digitalizar con un ADC (analog to digital converter), transmitirla y en el extremo de destino transformarla de nuevo a formato análogo con un DAC (digital to analog converter). VoIP funciona de esa forma, digitalizando la voz en paquetes de datos, enviándola a través de la red y reconvirtiéndola a voz

en el destino. Básicamente el proceso comienza con la señal análoga del teléfono que es digitalizada en señales PCM (pulse code modulación) por medio del codificador/decodificador de voz (codec). Las muestras PCM son pasadas al algoritmo de compresión que comprime la voz y la fracciona en paquetes que pueden ser transmitidos para éste caso a través de una red privada WAN. En el otro extremo de la nube se realizan exactamente las mismas funciones en un orden inverso. El flujo de un circuito de voz comprimido es el mostrado en la figura 1.

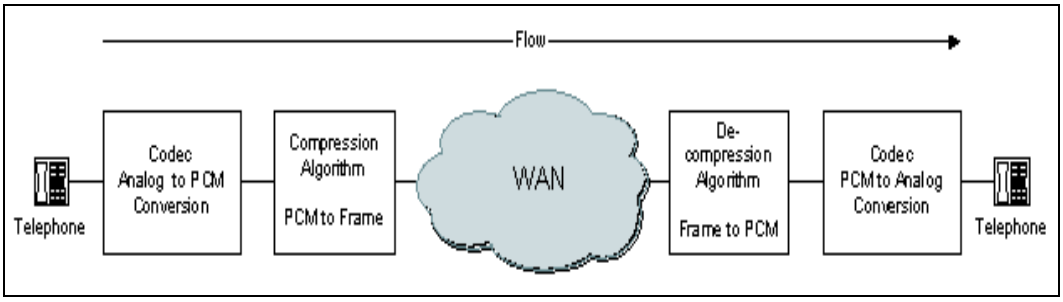


Figura 1. *Circuito de voz comprimido.*

Dependiendo de la forma en la que la red esté configurada, el enrutador o el gateway puede realizar la labor de codificación, decodificación y/o compresión. Por ejemplo, si el sistema usado es un sistema análogo de voz, entonces el enrutador o el gateway realizan todas las funciones mencionadas anteriormente como lo muestra la figura 2:

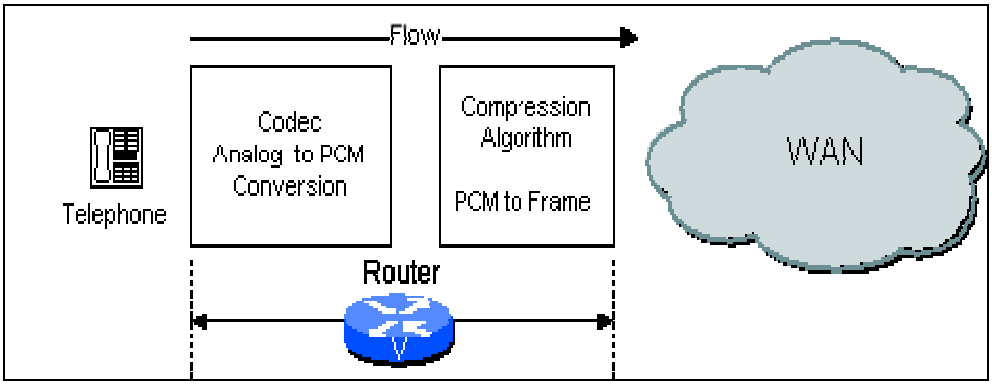


Figura 2. *Gateway de Voz completo.*

Si, por otro lado, el dispositivo utilizado es un PBX digital, es entonces éste el que realiza la función de codificación y decodificación, y el enrutador solo se dedica a procesar las muestras PCM que le ha enviado el PBX (ver figura 3).

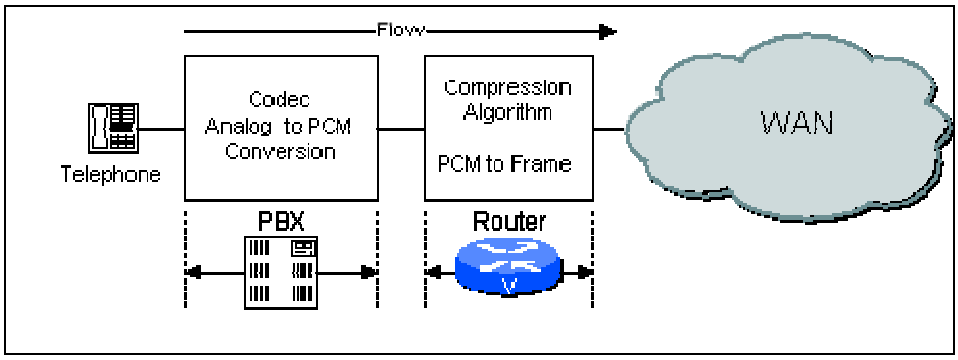


Figura 3. *Gateway procesador de muestras PCM.*

Para el caso en el que el transporte de voz se realiza sobre la red pública Internet, se necesita una interfaz entre la red telefónica y la red IP,

que se denomina gateway y es el encargado en el lado del emisor de convertir la señal analógica de voz en paquetes comprimidos IP para ser transportados a través de la red, del lado del receptor su labor es inversa, dado que descomprime los paquetes IP que recibe de la red de datos, y recompone el mensaje a su forma análoga original conduciéndolo de nuevo a la red telefónica convencional en el sector de la última milla para ser transportado al destinatario final y ser reproducido por el parlante del receptor.

Todas las redes deben tener de alguna forma las características de direccionamiento, enrutamiento y señalización. El direccionamiento es requerido para identificar el origen y destino de las llamadas, también es usado para asociar clases de servicio a cada una de las llamadas dependiendo de la prioridad. El enrutamiento por su parte encuentra el mejor camino a seguir por el paquete desde la fuente hasta el destino y transporta la información a través de la red de la forma más eficiente, que ha sido determinada por el diseñador. La señalización alerta las estaciones terminales y a los elementos de la red su estado y la responsabilidad inmediata que tienen al establecer una conexión.

1.1.2. Escenario de la voz IP en servicios de telefonía.

Deben distinguirse dos escenarios de aplicación de la voz IP en servicios de telefonía. El primero es cuando la voz IP es transportada a través de redes privadas empresariales y el segundo, cuando la red de transporte usada entre los dos extremos de la conversación es Internet. En el primer el caso, ésta se considera voz viajando sobre el protocolo IP, mas no en Internet éste último caso es cuando se habla de telefonía por Internet. La diferencia entre los dos escenarios no son únicamente el medio de transporte sino también las posibilidades de establecer mecanismos de control de calidad que garanticen la misma calidad en todo momento. Los mecanismos y las técnicas aplicadas en ambos casos son diferentes pero los niveles de calidad que se consiguen son muy similares y, en algunos casos, superiores a la telefonía convencional. Por ejemplo, los proveedores de servicio que utilizan Internet como una red de transporte para voz IP usan una técnica a partir de software que evita que los puntos de congestión causen pérdidas de calidad. Cuando se trata de transportar la voz IP a través de redes privadas hay medios más simples y efectivos que aseguran que los paquetes de voz se dirigen a cada uno de los dispositivos de la red antes que los datos y así, se evitan potenciales atrasos en caso de saturación de la red.

Llamadas teléfono a teléfono

En éste caso tanto el origen como el destino necesitan ponerse en contacto con un gateway. Supongamos que el teléfono **A** descuelga y solicita efectuar una llamada a **B**. el gateway de **A** solicita información al gatekeeper sobre como alcanzar a **B**, y éste le responde con la dirección IP del gateway que da servicio a **B**; entonces el gateway de **A** convierte la señal analógica del teléfono **A** en paquetes IP y lo envía hacia el gateway de **B**, el que va regenerando la señal analógica a partir de los paquetes IP que recibe con destino al teléfono **B**. El gateway de **B** se encarga de enviar la señal analógica al teléfono **B**. Por tanto tenemos una comunicación telefónica convencional entre el teléfono **A** y el gateway que le da servicio (gateway **A**), una comunicación de datos a través de una red IP, entre el gateway **A** y el **B**, y una comunicación telefónica convencional entre el gateway que da servicio al teléfono **B** (gateway **B**), y éste. Es decir, dos llamadas telefónicas convencionales, y una comunicación IP. Si las dos primeras son metropolitanas, que es lo normal, el margen con respecto a una llamada telefónica convencional de larga distancia o internacional, es muy grande.

Llamadas PC a teléfono o viceversa

En éste caso sólo un extremo necesita ponerse en contacto con un gateway. El PC debe contar con una aplicación que sea capaz de establecer y mantener una llamada telefónica. Supongamos que un ordenador **A** trata de llamar a un teléfono **B**, en primer lugar la aplicación telefónica de **A** ha de solicitar información al gatekeeper, que le proporcionará la dirección IP del gateway que da servicio a **B**. entonces la aplicación telefónica de **A** establece una conexión de datos, a través de la red IP, con el gateway de **B**, el cuál va regenerando la señal analógica a partir de los paquetes IP que recibe con destino al teléfono **B**. Como el gateway de **B** se encarga de enviar la señal analógica al teléfono **B**. Por tanto tenemos una comunicación de datos a través de una red IP, entre el ordenador **B** y el gateway de **B**, y una comunicación telefónica convencional entre el gateway que da servicio al teléfono **B** (gateway **B**), y éste. Es decir, una llamada telefónica convencional, y una comunicación IP. Si la primera es metropolitana, que es lo normal, el margen con respecto a una llamada telefónica convencional de larga distancia o internacional, es muy grande.

Llamadas PC a PC

Ambos ordenadores sólo necesitan tener instalada la misma aplicación encargada de gestionar la llamada telefónica, y estar conectados a la red IP, internet generalmente, para poder efectuar una llamada IP.

1.1.3. Convergencia hacia VoIP.

Todo el mundo conoce las ventajas que brinda la voz sobre IP; pero cómo adoptar y desplegar esta nueva alternativa sigue siendo una incógnita para muchos usuarios, a continuación se mencionarán algunas cuestiones a tener en cuenta para adentrarse en el mundo de las redes convergentes.

El argumento inicial en favor de éste nuevo modelo de redes se basa en la gran presencia actual de las infraestructuras IP en los entornos corporativos de datos, así como en la suposición de que parte de la capacidad de estas redes está siendo desaprovechada, hay que emplear el ancho de banda inutilizado para soportar el tráfico de voz y fax. De esta forma no sólo aumentaría la eficiencia global de la red, sino también las sinergias entre su diseño, despliegue y gestión.

1. La convergencia plantea un serio reto: las redes de voz y datos son esencialmente diferentes. Las redes de voz y fax, que emplean conmutación de circuitos, se caracterizan por:

- Para iniciar la conexión es preciso realizar el establecimiento de llamada.
- Se reservan recursos de la red durante todo el tiempo que dura la conexión.
- Se utiliza un ancho de banda fijo (típicamente 64 Kbps por canal de voz) que puede ser consumido o no en función del tráfico.
- Los precios generalmente se basan en el tiempo de uso.
- Los proveedores están sujetos a las normas del sector, regulados y controlados por las autoridades pertinentes.
- El servicio debe ser universal para todo el ámbito estatal.

Por el contrario, las redes de datos, basadas en la conmutación de paquetes, se identifican por las siguientes características:

- Para asegurar la entrega de los datos se requiere el direccionamiento por paquetes, sin que sea necesario el establecimiento de llamada.

- El consumo de los recursos de red se realiza en función de las necesidades, sin que, por lo general, sean reservados siguiendo un criterio de extremo a extremo.
- Los precios se manejan exclusivamente en función competitiva de la oferta y la demanda.
- Los servicios se prestan de acuerdo a los criterios impuestos por la demanda, variando ampliamente en cuanto a cobertura geográfica, velocidad de la tecnología aplicada y condiciones de prestación.
- Implementar una red convergente supone estudiar las diferencias existentes entre las características de las redes de voz y de datos, comprendiendo los problemas técnicos que implican dichas diferencias sin perder de vista en ningún momento la perspectiva del usuario final.

2. Las diferencias entre la operación de las redes de voz y datos requieren distintos enfoques de gestión. Tradicionalmente, la industria de la telefonía trabaja con unas altas exigencias de fiabilidad, conocidas como los "cinco nueves”: 99,999 por ciento. Esto se traduce en objetivos de diseño de centrales públicas de conmutación que garantizan niveles de caída del servicio de sólo dos horas cada cuarenta años de operación. Cuarenta años son aproximadamente 350.400 horas; y dos horas sin servicio representaría

sólo un 0,0000057 de todo ese tiempo. O lo que es lo mismo, una disponibilidad del 99,9994 por ciento.

3. Factores de Calidad de Servicio (QoS). La entrega de señales de voz, vídeo y fax desde un punto a otro no se puede considerar realizada con un éxito total a menos que la calidad de las señales transmitidas satisfaga al receptor. Entre los factores que afectan a la calidad se encuentran los siguientes:

- Requerimientos de ancho de banda: La velocidad de transmisión de la infraestructura de red y su topología física.
- Funciones de control: Incluye la reserva de recursos, provisión y monitorización requeridos para establecer y mantener la conexión multimedia.
- Latencia o retardo: De la fuente al destino de la señal a través de la red.
- Jitter: Variación en los tiempos de llegada entre los paquetes. Para minimizar éste factor los paquetes entrantes deben ser introducidos en un buffer y, desde allí, enviados a intervalos estándar.
- Pérdida de paquetes: Cuando un paquete de vídeo o de voz se pierde en la red es preciso disponer de algún tipo de compensación de la señal en el extremo receptor.

4. Implementación de nuevos estándares. Los estándares vienen a ser el anteproyecto necesario para diseñar, implementar y gestionar las comunicaciones de voz y datos. En su desarrollo trabajan diferentes entidades reconocidas como organizaciones de estándares internacionales, entre los que se encuentran ANSI (American National Standards Institute), IEEE (Institute of Electrical and Electronics Engineers), ISO (International Organization for Standardization), UIT (Unión Internacional de Telecomunicaciones) e IETF (Internet Engineering Task Force). Gracias a un estricto cumplimiento de los estándares internacionales (ITU H.323, H.245, H.225) el Gateway VoIP puede integrarse fácilmente en redes en las que existan Gateways H.323 de otros fabricantes de forma que se puedan intercambiar llamadas entre ellos. De igual forma el Gateway VoIP podrá integrarse en una red gestionada por un Gatekeeper H.323.

5. Interoperatividad multifabricante. Pueden operar y comunicarse equipos de diferentes marcas.

6. Otros factores significativos. Además de las cuestiones de gestión y diseño, existen otros factores, algunos fuera del control de los usuarios, que afectarán la migración a las redes convergentes. Por ejemplo, la Comisión Europea ha determinado que, de momento, dadas las características y el estado de desarrollo de VoIP, hay que considerarlo como un servicio

desregulado y no sometido a limitaciones normativas. No obstante, la Comisión se ha encargado de aclarar que seguirá de cerca los pasos de la telefonía IP por si su posterior evolución exigiera introducir cambios en su regulación.

En muy poco tiempo, el interés por la voz sobre IP irá más allá de las simples llamadas gratuitas de voz y fax por Internet para extender su influencia a cómo las comunicaciones de empresa darán servicio a los usuarios finales en el próximo milenio, y a las potenciales economías de escala que promete.

1.1.4. Ventajas y desventajas de la telefonía IP.

Ventajas

- Es evidente que el hecho de tener una red en vez de dos, es beneficioso para cualquier operador que ofrezca ambos servicios.
- Realmente se trata de una solución verdaderamente fantástica, facturas de teléfono muy bajas, oficinas virtuales, dirección centralizada y un rápido despliegue, son sólo algunos de los beneficios, el éxito de algunas grandes compañías combinado con el

crecimiento de las redes wireless, puede mover esta tecnología desde las empresas a los pequeños negocios y a todo el mercado en general.

- Como si el ahorro de ancho de banda no fuera suficiente, el despliegue de la voz sobre IP reduce el costo y mejora la escalabilidad empleando componentes de redes de datos estándares (routers, switches...), en vez de los caros o complicados switches para teléfonos, ahora el mismo equipo que dirige las redes de datos puede manejar una red de voz.

- VoIP posibilita desarrollar una única red convergente que se encargue de cursar todo tipo de comunicación, sea voz, datos, video o cualquier tipo de información.

- La telefonía IP no requiere el establecimiento de un circuito físico durante el tiempo que toma la conversación, por lo tanto, los recursos que intervienen en la realización de una llamada pueden ser utilizados en otra cuando se produce un silencio, lo que implica un uso más eficiente de los mismos.

- Las redes de conmutación por paquetes proveen alta calidad telefónica utilizando un ancho de banda menor que el de la telefonía clásica, debido a los algoritmos de compresión pueden reducir hasta 8kbps el ancho de banda para digitalización de la voz, produciendo un desmejoramiento en la calidad de la misma apenas perceptible.

Desventajas

- Transportan la información dividida en paquetes, por lo que una conexión suele consistir en la transmisión de más de un paquete, estos paquetes pueden perderse, y además no hay una garantía sobre el tiempo que tardarán en llegar de un extremo al otro de la comunicación.

- El aspecto de seguridad es muy relevante como se explicó anteriormente.

- Se cambia confiabilidad por velocidad.

- Finalmente, tenemos que resaltar que así como PSTN, VoIP no puede prestar servicio a todos sus clientes (por ejemplo, una llamada GSM

no puede manejar más de algunos cientos o un par de miles de clientes).

- Por ahora, el servicio está restringido a redes privadas (y en consecuencia a pocos usuarios), por que en un ambiente como una red pública Internet, los niveles de calidad telefónica son bajos pues tal red no puede proveer anchos de banda reservados ni controlar la fluctuación de carga que se presenta.

- El control de congestión de TCP hace reducir la ventana de transmisión cuando detecta pérdida de paquetes, y el audio y el video son aplicaciones cuya velocidad de transferencia no permite disminuciones de éste tipo en la ventana de transmisión.

1.2. La Arquitectura H.323

1.2.1. Estándar y protocolos VoIP/H.323

Direccionamiento:

Tomando de nuevo el ejemplo de una intranet con direccionamiento IP, podríamos ver que las interfaces de voz aparecerían como anfitriones IP adicionales, como extensiones del esquema de numeración existente o como nuevas direcciones IP.

La traducción de los dígitos marcados del PBX al host IP se realizan por medio del plan de numeración. El número de teléfono de destino o alguna parte de éste será vinculado a la dirección IP de destino. Cuando el número es recibido del PBX, el enrutador lo compara con los que se han vinculado con alguna dirección IP y están relacionados en la tabla de ruteo, si hay alguna coincidencia la llamada será enrutada al host IP al cual este relacionada, después de que la conexión es establecida, el enlace de la intranet es transparente hacia el suscriptor.

1. RAS (Registration, Admision and Status). Protocolo de comunicaciones que permite a una estación H.323 localizar otra estación H.323 a través del Gatekeeper.
2. DNS (Domain Name Service). Servicio de resolución de nombres en direcciones IP con el mismo fin que el protocolo RAS; pero a través de un servidor DNS.

Señalización:

La señalización VoIP tiene 3 áreas distintas: Señalización del PBX al ruteador, señalización entre ruteador y señalización del ruteador al PBX. Por ejemplo para el caso de una intranet corporativa, esta aparece como la troncal al PBX, quien dará la señalización a los usuarios de la intranet, por lo cual el PBX reenvía los números digitados al ruteador de la misma forma en la que los dígitos hubiesen sido reenviados al switch de una central telefónica. Cuando el enrutador remoto recibe la llamada solicitante, éste envía una señalización al PBX. Luego que el PBX envía un acuse de recibo, el ruteador envía los dígitos marcados al PBX, y tramita un acuse de recibo de llamada al ruteador de origen.

En una arquitectura de red no orientada a la conexión (como IP), la responsabilidad del establecimiento de la comunicación y de la señalización

es de las estaciones finales (end stations). Para prestar exitosamente servicios de voz a través de una re IP, es necesario realizar mejoras en la señalización. Por ejemplo, un agente de h.323 es adicionado al enrutador para facilitar soporte para el transporte de cadenas de audio y señalización. El protocolo q.931 es usado para el establecimiento y desconexión de la llamada entre agentes h.323 o estaciones terminales. RTCP (real time control protocol) es usado para establecer canales de audio. Un protocolo confiable orientado a la conexión, TCP, es utilizado entre estaciones terminales para transportar los canales de señalización.

RTP, protocolo de transporte en tiempo real, el cual está soportado en UDP, es usada para el trasporte del caudal de audio en tiempo real. RTP usa UDP como mecanismo de transporte porque posee un menor retardo que TCP, y además porque el trafico de voz en la actualidad, sin importar que sean datos o señalización, toleran menos niveles de perdida y no tienen la facilidad de retransmisión.

CAPA SEGÚN OSI	ITU H.323 ESTÁNDAR
Presentación	g.711,g.729, g.729a, etc.
Sesión	h.323, h.245, h.225, RTCP
Transporte	RTP, UDP
Red	IP, RSVP, WFQ
Enlace	rfc1717(PPP/ML), Frame, ATM, etc.

Tabla 1. *Modelo de referencia OSI y estándar H.323*

1. Q.931 Señalización inicial de llamada.
2. H.225 Control de llamada: señalización, registro y admisión, y paquetización / sincronización del stream (flujo) de voz.
3. H.245 Protocolo de control para especificar mensajes de apertura y cierre de canales para streams de voz**.**

Compresión de voz:

Los algoritmos de compresión usados en los ruteadores y en los gateways analizan un bloque de muestras PCM entregadas por el codificador de voz (voice codec), estos bloques tienen una longitud variable que depende del codificador, por ejemplo el tamaño básico de un bloque del

algoritmo g.729 es 10 ms, mientras que el tamaño básico de un bloque del algoritmo g.723.1 es 30ms. un ejemplo de cómo funciona el sistema de compresión g.729 es mostrado en la siguiente figura:

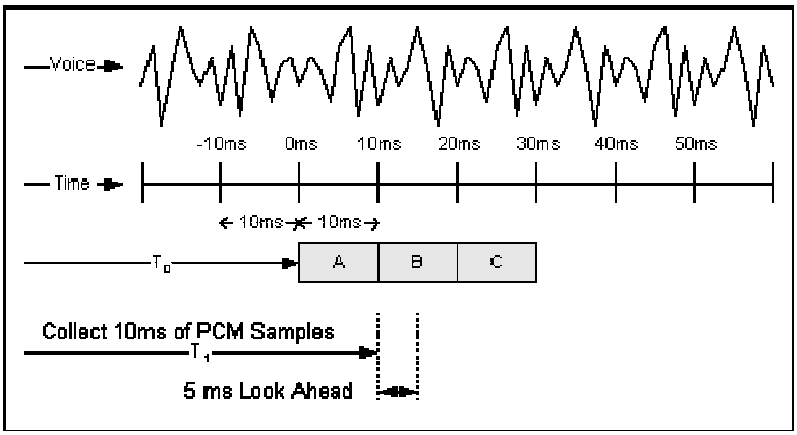


Figura 4. *Compresión de voz*

La cadena de voz análoga es digitalizada en tramas PCM, y así mismo entregadas al algoritmo de compresión en intervalos de 10 ms.

1. Requeridos: G.711 y G.723
2. Opcionales: G.728, G.729 y G.722

Transmisión de voz:

1. UDP. La transmisión se realiza sobre paquetes UDP, pues aunque UDP no ofrece integridad en los datos, el aprovechamiento del ancho de banda es mayor que con TCP.
2. RTP (Real Time Protocol). Maneja los aspectos relativos a la temporización, marcando los paquetes UDP con la información necesaria para la correcta entrega de los mismos en recepción.

RTP (Real Time Transport Protocol) o protocolo de transporte en tiempo real, es un protocolo que como su nombre lo indica, está orientado a la transmisión de información en tiempo real, como la voz o el video. Este es un protocolo de las capas superiores de usuario que funciona sobre UDP (user datagram protocol) haciendo uso de los servicios de checksum y multiplexión, para proporcionarle a los programas que generan éste tipo de datos, una manejo de transmisiones en tiempo real a través de difusiones unicast o multicast, en el UDP se cambia confiabilidad por velocidad, lo cual es básico para manejo de transmisiones en tiempo real como la VoIP.

Aunque RTP no es lo suficientemente confiable por si solo, éste proporciona “ganchos” con protocolos y aplicaciones de capas inferiores y recursos proporcionados por los switches y ruteador para garantizar

confiabilidad. Los paquetes RTP no contienen campo de longitud, por que al funcionar sobre UDP, éste protocolo es quien encapsula la voz comprimida en datagramas.

Las herramientas de las que se vale RTP para lograr transmisiones en tiempo real son el RTCP (RTP control protocol) que proporciona un feedback a cerca de la calidad de distribución y la congestión, con esto, la empresa que ofrece el servicio puede monitorear la calidad y puede diagnosticar los problemas que pueda presentar la red, además de esto, RTCP sincroniza el audio y el video, conoce el número de usuarios presentes en una conferencia y con esto calcula la velocidad a la cual deben ser enviados los paquetes, todas estas opciones son obligatorias cuando RTP se usa en entornos multicast IP; pero existe otra aplicación opcional y es una administración de sesiones con bajo manejo de información de control para aquellas aplicaciones donde hay uso masivo de usuarios entrando y saliendo constantemente.

Para la compresión RTP usa una aplicación llamada “vocoder” pudiendo reducir de 64 kbps hasta a 8 kbps el ancho de banda para digitalización y compresión de voz produciendo un desmejoramiento en la calidad de la voz poco perceptible, además de esto usa h.323 g.729 y otros protocolos más para transmisiones en tiempo real.

RTP es capaz de correr sobre protocolos WAN de alta velocidad como ATM sin ningún problema, también en redes asimétricas como ADSL, cable-modem o por enlace satelital pero cumpliendo con ciertas características de ancho de banda para ambas direcciones y uso exclusivo para la aplicación RTP.

A pesar de que TCP es un protocolo de transporte de información "pesada", y eventualmente podría llegar a transportar video y voz, éste y otros protocolos como XTP son inapropiados por tres razones básicas:

- El hecho de que ante la pérdida de paquetes éste tipo de protocolos emplean retransmisión de paquetes. TCP no soporta multicast.

- El control de congestión de TCP hace reducir la ventana de transmisión cuando detecta pérdida de paquetes, y el audio y el video son aplicaciones cuya velodicad de transferencia no permite disminuciones de éste tipo en la ventana de transmisión.

- Adicionalmente otra desventaja es que los encabezados de estos protocolos son más largos que los de RTP.

Control de la transmisión*:*

Figura 5. *Protocolos RTP y RTCP*

RTCP (Real Time **Control** Protocol). Se utiliza principalmente para detectar situaciones de congestión de la red y tomar, en su caso, **acciones** correctoras.

Este protocolo permite completar a RTP facilitando la comunicación entre extremos para intercambiar datos y monitorear de esta forma la calidad de servicio y obtener información acerca de los participantes en la sesión. RTCP se fundamenta en la transmisión periódica de paquetes de control a todos los participantes en la sesión usando el mismo mecanismo de RTP de distribución de paquetes de datos. El protocolo UDP dispone de distintas puertas (*UDP Port*) como mecanismo de identificación de protocolos.

La función primordial de RTCP es la de proveer una realimentación de la calidad de servicio. Se relaciona con el control de congestión y flujo de datos. El RTCP involucra varios tipos de mensajes, por ejemplo:

- **Send report** para emisión y recepción de estadísticas (en tiempo aleatorio) desde emisores activos.
- **Receiver Report** para recepción estadística desde emisores no activos.
- **Source Description** para un identificador de nivel de transporte denominado CNAME (*Canonical Name*).
- **Bye** para indicar el final de la participación en la conexión.
- **Application** para aplicaciones específicas.

El mensaje Send Report, uno de los más interesantes, disponen de 3 secciones bien diferenciadas:

- Los primeros 8 Bytes se refieren a un encabezado común.
- La segunda parte de 20 Bytes permite la evaluación de diferentes parámetros (retardo, jitter, eficiencia de datos, etc).
- La tercera parte de 24 Bytes lleva reportes que se han obtenido desde el último reporte informado. Incluye los siguientes reportes: cantidad total de paquetes RTP perdidos y a la proporción de los mismos; la

cantidad de paquetes recibidos y el jitter entre paquetes; el horario del último paquete recibido y el retardo de transmisión del mismo.

1.2.2. Elementos de una red VoIP

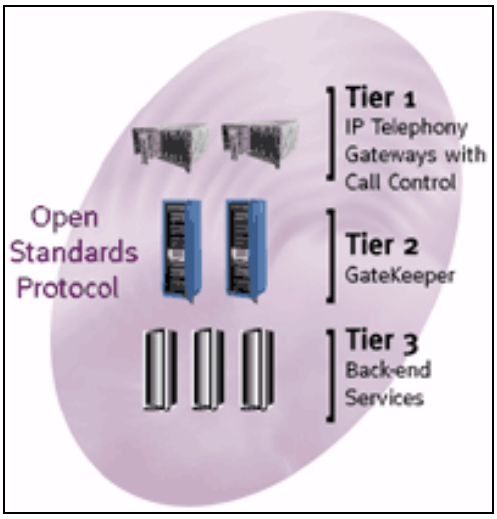


Figura 6. *División por capas VoIP*

El sistema de telefonía sobre IP de clase carrier de 3Com se basa en una arquitectura abierta de tres niveles de gateways, gatekeepers y servidores de backend interconectados mediante protocolos abiertos basados en normas. La arquitectura modular de 3Com presenta APIs estándar en cada nivel a fin de brindarle a los carriers flexibilidad para personalizar el sistema, facilitando la diferenciación de servicios y la integración de las "mejores" aplicaciones de oficina back-to-back "de su clase". Este sistema modular llave en mano basado en normas soporta la

telefonía sobre IP de teléfono a teléfono y de PC a teléfono en redes conmutadas por paquetes.

1.2.2.1. VoIP Gateway.

Los gateways de VoIP proveen un acceso ininterrumpido a la red IP. Las llamadas de voz se digitalizan, codifican, comprimen y paquetizan en un gateway de origen y luego, se descomprimen, decodifican y rearman en el gateway de destino.

El procesamiento que realiza el gateway de la cadena de audio que atraviesa una red IP es transparente para los usuarios. Desde el punto de vista de la persona que llama, la experiencia es muy parecida a utilizar una tarjeta de llamada telefónica. La persona que realiza la llamada ingresa a un gateway por medio de un teléfono convencional digitando un número de acceso. Una vez que fue autenticada, la persona digita el número deseado y oye los tonos de llamada habituales hasta que alguien responde del otro lado. Tanto quien llama como quien responde se sienten como en una llamada telefónica "típica".

Podemos considerar al Gateway como una caja que por un lado tiene un interface LAN y por el otro dispone de uno o varios de los siguientes interfaces:

- FXO. Para conexión a extensiones de centrales ó a la red telefónica básica.
- FXS. Para conexión a enlaces de centrales o a teléfonos analógicos.
- E&M. Para conexión específica a centrales.
- BRI. Acceso básico RDSI (2B+D)
- PRI. Acceso primario RDSI (30B+D)
- G703/G.704. (E&M digital) Conexión especifica a centrales a 2 Mbps.

1.2.2.2. VoIP Gatekeeper.

Los gateways se conectan con los gatekeepers de VoIP mediante enlaces estándar H.323v2, utilizando el protocolo RAS H.225. Los gatekeepers actúan como controladores del sistema y cumplen con el segundo nivel de funciones esenciales en el sistema de VoIP de clase carrier, es decir, autenticación, enrutamiento del servidor de directorios, contabilidad de llamadas y determinación de tarifas. Los gatekeepers utilizan la interfaz estándar de la industria ODBC-32 (Open Data Base Connectivity – Conectividad abierta de bases de datos) para acceder a los servidores de

backend en el centro de cómputos del carrier y así autenticar a las personas que llaman como abonados válidos al servicio, optimizar la selección del gateway de destino y sus alternativas, hacer un seguimiento y una actualización de los registros de llamadas y la información de facturación, y guardar detalles del plan de facturación de la persona que efectúa la llamada.

Las funciones del GK son:

- Traslación de direcciones desde una dirección “alias” del terminal hacia dirección de capa 3/4 (socket).
- Control de admisión para autorizar el acceso a la red mediante mensajes ARQ/ACF/ARJ (protocolo RAS).
- Control de ancho de banda mediante mensajes BRQ/BRJ/BCF (protocolo RAS).
- Señalización de control de llamada para autorización o rechazo de llamadas.
- Servicios de directorio.
- Servicio de reservación de ancho de banda, etc.

1.2.2.3. Servidores backend.

El tercer nivel de la arquitectura de VoIP de clase carrier de 3Com corresponde a la serie de aplicaciones de backoffice que constituyen el corazón del sistema operativo de un proveedor de servicios. Las bases de datos inteligentes y redundantes almacenan información crítica que intercambian con los gatekeepers durante las fases de inicio y terminación de las llamadas. En el entorno de una oficina central, resulta vital preservar la integridad de los datos de las bases de datos de backend. La solución de 3Com ofrece un enfoque único que garantiza la resistencia de los servidores de backend y la seguridad de sus bases de datos. Los servidores SQL de Microsoft están integrados dentro de la arquitectura del sistema de Backend y administran las bases de datos SQL para las funciones de autenticación, mapeo de directorios, contabilidad y determinación de tarifas. Este nivel de la arquitectura fue optimizado a fin de responder a las necesidades exclusivas de seguridad y disponibilidad de los proveedores de servicios. Para implementaciones a menor escala, el sistema ofrece flexibilidad para consolidar las bases de datos en un solo servidor robusto o en la plataforma de un gatekeeper.

1.2.2.4. MGC (*Media Gateway Controller*) o Softswitch.

Es el control de procesamiento con la red pública PSTN. El MGC es un software que contiene en su interior al GK. Realiza las siguientes funciones:

- Control de llamada (asimilable al punto de conmutación en las PBX).
- Identificación del tráfico H.323 y aplicación de las políticas apropiadas.
- Limitación del tráfico H.323 sobre la LAN y WAN.
- Entrega archivos CDR (*Call Detail Records*) para la facturación (*Billing*).
- Realiza la interfaz con las redes inteligentes.
- Inserta calidad de servicio e implementa políticas de seguridad.

Los MGC pueden colocarse en configuración Failover para protección ante fallas. Los GW son controlados por el MGC mediante el protocolo MGCP (Media Gateway Control Protocol). Como protocolo de señalización hacia la PSTN se utilizan ISUP/TCAP de la serie SS7 o el MFC-R2 para centrales sin facilidad SS7. En las redes de Telefonía-IP públicas, el GK se encuentra integrado al MGC. También se dispone de servidores para RADIUS (para gestión de seguridad), para LDAP (servicio de directorio y memoria) y para AAA (funciones de autentificación y cobro).

Las funciones del MGC pueden ser realizadas mediante dos técnicas distintas. La primera toma del mundo de la telefonía pública convencional las partes que pueden ser utilizadas (procesador central, memoria, cómputo de tráfico, etc.) y eliminan aquellas que no corresponden (red de conmutación de circuitos). En la segunda, se trata de un software absolutamente nuevo (conocido como Softswitch) que corre sobre una plataforma genérica (por ejemplo, Linux). De acuerdo con la nomenclatura de la norma H.323 el controlador de llamada es el Gatekeeper GK; sin embargo, se ha popularizado también la denominación MGC para una mayor extensión de funciones.

1.2.2.5. Procedimiento de comunicación H.323.

El procedimiento de funcionamiento de los protocolos H.323 se describe con detalle a continuación. En H.323 se encuentran 3 tipos de mensajes de señalización diferentes:

H.245: Se describen estos mensajes en forma de texto concatenado en letras tipo bold (por ejemplo se menciona el mensaje: **maximumDelayJitter**).

RAS: Se representa mediante 3 letras (por ejemplo ARQ).

H.225/Q.931: Representado en una o dos palabras con la primera letra en mayúsculas (ejemplo: Call Proceeding). Es usado para encapsular los mensajes H.245 de señalización entre terminales y originalmente fue diseñado como protocolo DSS1 en capa 3/7 para los accesos ISDN.

Fase de Mantenimiento de la registración. Contiene un intercambio de mensajes para mantener activa la conexión entre los Gateways GW y el Gatekeeper GK. Ver la Figura 3 para el intercambio de mensajes de RAS.

1- Discovery. Este primer paso es el proceso por el cual el GW determina cual es el GK que atiende a la red en ese momento. El mensaje desde el GW es del tipo multicast y se denomina **GRQ** (*Gatekeeper Request*). El GK responde con la aceptación **GCF** (*GK Confirmation*) o rechazo **GRJ** (*GK Reject*). El GK puede indicar un GK alternativo mediante mensajes **alternateGatekeeper**. Si no se está en condiciones de procesar el request, se puede enviar un mensaje **RIP** (*Requst in Progress*) para indicar que se está procesando el request; esto resetea el timeout de la conexión.

2- Registration. El GW informa de sus direcciones de transporte y alias mediante **RRQ** (*Registration Request*) y el GK responde con **RCF** (*Registration Confirmation*) o **RRJ** (*Registration Reject*). El RRQ se emite en forma periódica. La registración tiene un tiempo de duración (expresado en

segundos) para lo cual se utiliza el mensaje **timeToLive**. El terminal o el GK pueden cancelar la registración mediante el mensaje **URQ** (*Unregister Request*) al cual le corresponde la confirmación **URF** (*Unregister Confirmation*).

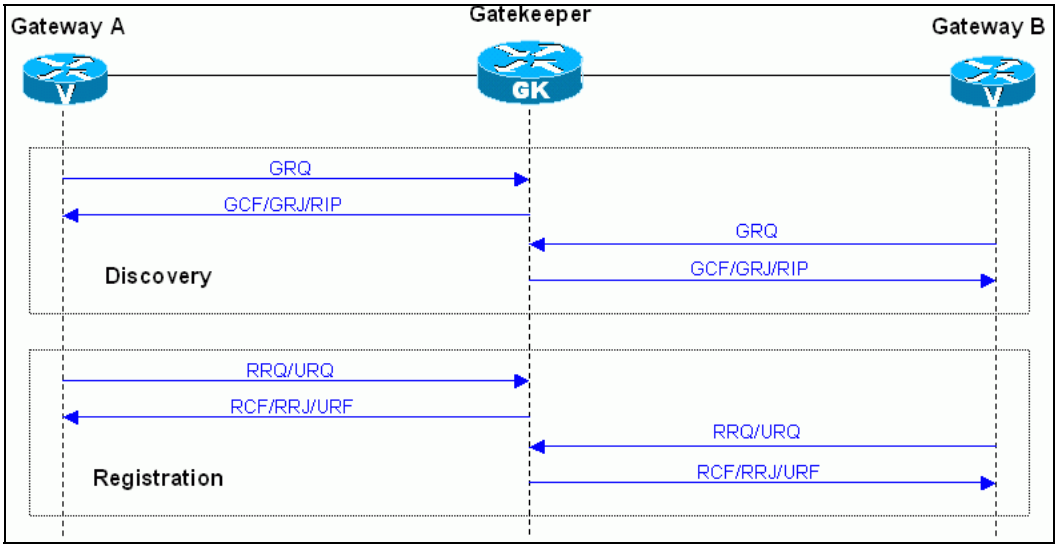


Figura 7. Fase de mantenimiento de la registración entre GW y GK.

3- Location. Un GW o GK que tiene un alias para un GW y quiere determinar su información de contacto, puede emitir el mensaje de requerimiento de localización **LRQ** (*Location Request*). Al cual le corresponde la confirmación **LCF** (*Location Confirmation*) con la información requerida. La dirección puede ser del tipo E.164 si se trata de un GK fuera de la red.

De existir varios GK se disponen de mensajes para intercomunicación, por ejemplo, **LRQ** para *Locate Request* y **LCF** para *Locate Confirm*.

4- Status. Se trata de un mensaje periódico (mayor a 10 segundos) que emite el GK al terminal para determinar el estado y requerir un diagnóstico. Se trata de los mensajes **IRQ** (*Information Request*) y **IRR** (*Information Response*). La habilitación se realiza mediante **willRespondToIRR** enviado en el mensaje RCF o ACF.

Fase de conexión de la llamada. Representa las distintas etapas para establecer una llamada.

5- Admission. En la Figura 8, el proceso se inicia cuando desde la PSTN se recibe un mensaje de Setup para inicio de una llamada entrante en protocolo ISUP (de la suite de protocolos de señalización telefónica SS7). El GW responde a la PSTN mediante el mensaje Call Proceesing, para mantener la conexión en espera.

El GW requiere iniciar una llamada mediante el pedido de admisión desde GW al GK. Este mensaje es **ARQ** (*Admissions Request*) y contiene un requerimiento Call Bandwidth (en formato Q.931). El GK puede reducir las características de la solicitud en el mensaje de confirmación **ACF**

(*Admissions Confirm*). En el mismo mensaje ARQ se dispone de la funcionalidad **TransportQOS** para habilitar la funcionalidad de reservación de ancho de banda RSVP, para servicios unidireccionales (orientado-al-receptor).

6a- Setup modo no-ruteado. Una vez admitido el GW-B por el GK el procedimiento se bifurca en el modo ruteado y no-ruteado. En el modo de operación no-ruteado, el GK informa al GW-B cual es la dirección IP del GW-A al cual va dirigida la llamada, de acuerdo con la dirección E.164 recibida en el mensaje ARQ. Ahora, el GW-B se comunica con el GW-A que fue indicado por el GK y le envía el mensaje Setup. Este mensaje (en protocolo Q.931) es respondido mediante el mensaje Call Proceesing. El GW-A se ocupa de registrarse mediante ARQ y recibe desde el GK el mensaje ACF. Con estas acciones cumplidas, el GW-A se ocupa de informar al usuario de la llamada entrante (corriente de llamada al teléfono) y hacia el GW-B le envía el mensaje de Alerting en Q.931 para indicar el estado de llamada. El GW-B envía el mensaje de Alerting a la PSTN, ahora en formato de protocolo ISUP.

6b- Setup modo ruteado. Para el caso de trabajar con Modo Ruteado, el mensaje de Setup entre GW pasa por el GK. En el caso No-Ruteado anterior, el GK se desentiende de la conexión y solo se ocupa de la traslación entre direcciones E.164 y IP. En el modo ruteado el GK seguirá toda la conexión,

de forma que haciendo uso de las funcionalidades de Softswitch se podrán ofrecer servicios de valor agregado.

7- Conect. Cuando el usuario en el GW-A responde se genera el mensaje Q.931 de Connect. Este mensaje se emite hacia el GK (Modo Ruteado), quien hace lo mismo hacia el GW-B y éste lo imita hacia la PSTN pero en protocolo ISUP.

El paso siguiente es establecer las capacidades de los terminales utilizando el protocolo H.245 entre GWs. Se trata del mensaje **TerminalCapabilitySet** de solicitud y el **TerminalCapabilityAck** de respuesta, que permite determinar capacidad del terminal, tipo de codificador, canal lógico, etc. Finalmente, se envía el mensaje **OpenLogical Channel** para abrir un canal lógico.

8- Canal Vocal. El canal vocal se transporta sobre los protocolos RTP de la suite IP. Para más detalles se puede consultar el siguiente ítem.

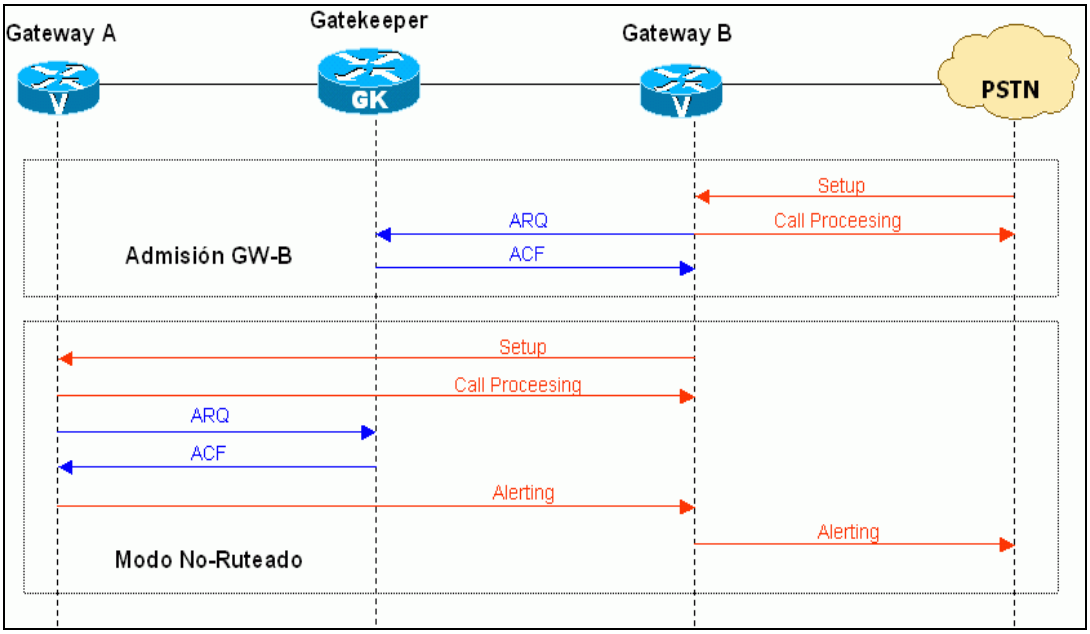


Figura 8. *Operación de Conexión mediante el Modo No-Ruteado.*

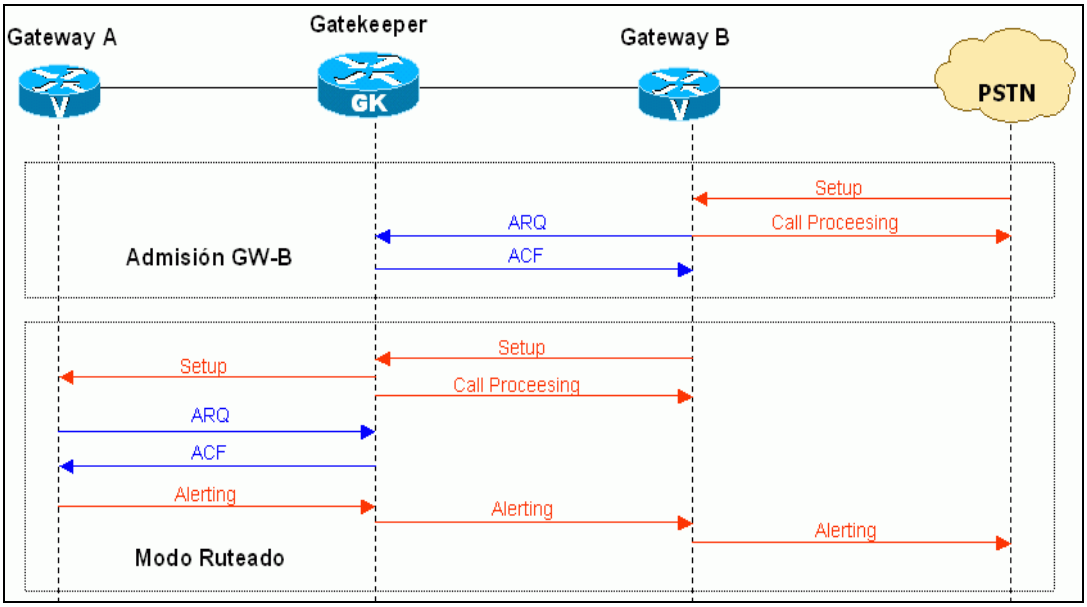


Figura 9. *Operación de Conexión mediante el Modo Ruteado.*

9- Bandwidth. Durante una conexión el terminal o el GK pueden requerir el cambio de ancho de banda del canal mediante el mensaje **BCR** (*Bandwidth Change Request*).

Fase de desconexión de la llamada. En la Figura 10 se indica la fase de desconexión de la llamada. La misma se realiza con mensajes Release Complete de Q.931 y **DRQ** (*Delete Request*) y **DCF** (*Delete Confirm*) de RAS.

Sobre el paquete Q.931 (H.225) se disponen de distintos tipos de mensajes:

- Mensajes para establecimiento de llamada: Alerting, Call Proceeding, Connect, Setup, Progress, etc.
- Mensajes para la fase de información de llamada: Resume, Suspend, User Information, etc.
- Mensajes para el cierre de la llamada: Disconnect, Release, Restart, etc.
- Mensajes misceláneos: Segment, Congestion Control, Information, Notify, Status, Status Enquiry, etc.

Los mensajes manejados en el ámbito de H.245 (durante la fase de comunicación telefónica) son:

- **multimediaSystemControl** para efectuar el control del sistema; las variantes del mensaje son request, response, command and indication.
- otros mensajes de interés son: **masterSlaveDetermination, terminalCapability, MaintenanceLoop, communication Mode, communicationMode, conferenceRequest and Response, terminal-ID.**

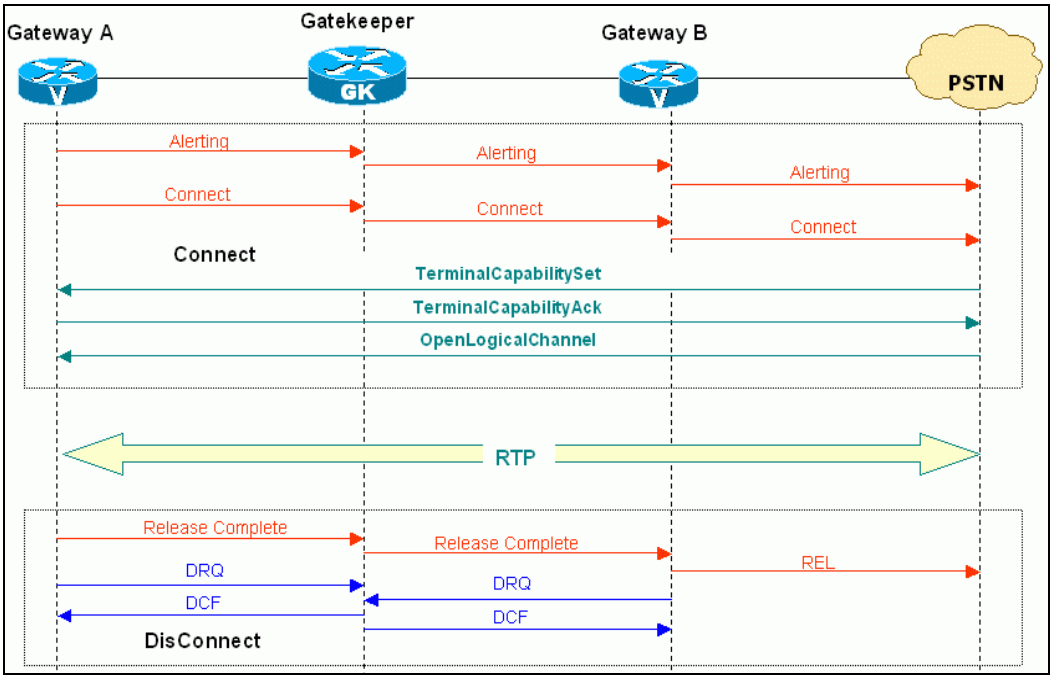


Figura 10. *Conexión final de la llamada y desconexión de la misma en Modo Ruteado.*

1.3. La Arquitectura SIP

Las arquitecturas definidas en torno a los protocolos **H.323** y **SIP** son sustancialmente distintas; sin embargo, al comparar la evolución de ambos estándares durante los últimos años, se extrae la conclusión de que, a medida que se definen nuevas ampliaciones a **SIP** y aparecen nuevas versiones de H.323, cada vez se diferencian menos en cuanto a las funciones y posibilidades que ofrecen.

Tanto la recomendación H.323 como el **protocolo SIP** definen mecanismos de señalización para establecer y terminar llamadas, así como otras funciones de control de conferencia, negociación de capacidades y servicios adicionales sobre redes de conmutación de paquetes. **SIP** se ha diseñado con posterioridad con la pretensión de que, desde la perspectiva de los estándares y prácticas habituales en Internet, presente las siguientes ventajas frente a H.323:

• Implementación más fácil de realizar y depurar

• Mayor flexibilidad para incorporar nuevas funciones

• Mayor integración con otras aplicaciones y servicios Internet

Aunque **SIP** se creó como un **protocolo** de inicio de sesiones, ha evolucionado, mediante la definición de nuevas funciones y servicios en forma de módulos complementarios basados en un núcleo de funciones básicas flexibles y ampliables, hasta constituirse en el **protocolo** de señalización y control propuesto por el IETF como base para los servicios de telefonía y comunicación multimedia en general en Internet, así como el **protocolo** de señalización de la red telefónica de tercera generación y la base de algunos de los sistemas de mensajería instantánea más extendidos.

1.3.3. Que es SIP.

SIP es un **protocolo** de señalización cliente-servidor de nivel de aplicación válido para redes unicast y multicast. Generalmente, los mensajes **SIP** constan de un conjunto de cabeceras y un cuerpo que contiene descripciones de sesiones multimedia, siendo SDP el formato utilizado en la actualidad.

Puesto que el formato de los mensajes **SIP** es textual, basado en HTTP y SMTP, y sigue principios similares a los de HTTP, es posible desarrollar servicios **SIP** mediante las procedimientos extendidos en la Web.

SIP cumple las siguientes funciones: establecimiento, modificación y finalización de sesiones multimedia, registro y localización de participantes, gestión del conjunto de participantes y de los componentes del sistema, así como descripción de las sesiones y negociación de capacidades.

1.3.2. Integración con protocolos IETF.

SIP no es un protocolo integrado verticalmente. SIP puede utilizar otros protocolos estándares para construir las sesiones de una aplicación basada en SIP. Por ejemplo:

TCP/UDP: Para transportar la información de señalización.

TLS: Para establecer sesiones seguras.

DNS: Para resolver nombres de servidores de acuerdo a la dirección de destino.

RSVP, DiffServ: Para asegurar la calidad de servicio de la sesión.

RTP (Real Time Protocol): Para transportar las comunicaciones interactivas de voz, datos y video.

RTSP (Real Time Streaming Protocol): Para controlar el envío de streaming media.

SAP (Session Advertisement Protocol): Para publicar sesiones multimedia vía multicast.

SDP (Session Description Protocol): Para describir sesiones multimedia.

MIME (Multipurpose Internet Mail Extension): Estándar para describir contenido en Internet.

http (Hypertext Transfer Protocol): Toma parte de la sintaxis y semántica, los mecanismos de autenticación, etc.

SMTP (Simple Mail Transport Protocol): Reutiliza headers, mecanismos de enrutamiento, modo de direccionamiento, etc.

COPS (Common Open Policy Service): Para establecer políticas de calidad y seguridad.

OSP (Open Settlement Protocol): Para automatizar el provissioning de los usuarios.

XML (eXtensible Markup Language): Para crear servicios y transmitir información de eventos.

1.3.3. Escalabilidad, simplicidad y movilidad del protocolo SIP.

Escalabilidad

La arquitectura SIP es escalable, flexible y distribuída. Las funcionalidades tales como proxy, redirección, locación y registro pueden residir en un único servidor o en varios servidores distribuidos. La funcionalidad distribuída permite incorporar nuevas funciones o procesos sin afectar los demás componentes.

El protocolo conserva información de estado en los extremos, permitiendo recuperarse de fallas de alguno de los componentes. La escalabilidad y redundancia se logra bajo el paradigma de N+1. No es necesario un control centralizado.

Simplicidad

Rápido, simple en el centro, inteligente y con menor volumen en el borde. Basado en texto para una implementación y depuración simple. Este protocolo utiliza de “primitivas” (métodos y respuestas) para el establecimiento de sesiones.

Movilidad

SIP permite implementar dos tipos de movilidad diferentes:

1. La movilidad personal, que permite que el usuario pueda ser alcanzado en un dispositivo cualquiera, mediante los servicios de proxy y redirección.

2. La movilidad intrínseca provista por la ubicuidad del protocolo IP.

El registro permite mantener las direcciones actuales del usuario de forma dinámica Basado en la locación actual el proxy server enrutará las llamadas a la locación actual del usuario. Ejemplos de aplicaciones de movilidad incluyen presencia y forking de llamadas.

5555@pbxgtwy.acme.com

5555555@cellcarrier.com

beth@bethpda.acme.com

beth@bethpc.acme.com

beth@bethipphone.acme.com

Llamar a:

beth@acme.com

1.3.4. Elementos de una red SIP.

Los dos tipos de elementos presentes en la arquitectura **SIP** son los *agentes de usuario (UA)* y los *servidore*s.

Los UA constan a su vez de dos componentes: Los agentes de usuario clientes (UAC) y los agentes de usuario servidores (UAS). Ambos se encuentran en todos los agentes de usuario, permitiendo la comunicación entre diferentes agentes de usuario mediante peticiones y respuestas de tipo cliente-servidor. Los UAC tienen como misión el envío de peticiones **SIP**, mientras que los UAS están encargados de atender tales peticiones y remitir las correspondientes respuestas.

Los servidores **SIP** pueden ser de tres tipos no excluyentes: *servidores de redirección, de registro* y *proxy*s. Aunque una llamada básica se puede realizar con **SIP** sin que intervengan servidores, las funciones avanzadas del **protocolo** no se pueden llevar a cabo sin su participación. La interacción entre UAC, UAS y servidores se realiza mediante el intercambio de mensajes **SIP** cuyo cuerpo contiene generalmente descripciones de sesiones multimedia. **SIP** define seis métodos (concepto tomado de HTTP) básicos: INVITE, ACK, BYE, CANCEL, OPTIONS y REGISTER. De ellos, los cuatro primeros intervienen en el establecimiento de llamadas. El método OPTIONS

permite intercambiar información sobre las capacidades de los componentes de un sistema **SIP** y el método REGISTER es la base para los servicios de localización.

Uno de los mecanismos de ampliación de **SIP** es la definición de nuevos métodos, como puede ser el método INFO, válido para el intercambio de información de control durante sesiones establecidas mediante **SIP**. Asimismo, las peticiones y respuestas **SIP** pueden contener cabeceras que permiten que elementos de una red **SIP** con diferentes niveles de implementación y actualización del estándar interoperen mediante un conjunto mínimo de métodos y reglas de procesamiento de mensajes.

Los nombres de los usuarios de los sistemas **SIP** tienen forma de dirección de correo electrónico, lo que permite el uso de URL **SIP** (del tipo **sip**:usuario@ejemplo.rediris.es) dentro de la infraestructura Web de forma similar a los URL *mailt*o. Del mismo modo, la localización de servidores **SIP** se basa en el DNS, y existen ampliaciones DHCP para la configuración dinámica de servidores proxy **SIP**.

Por otra parte, el esquema utilizado por **SIP** para la negociación de capacidades en el inicio y modificación de sesiones se basa en el **modelo** «**Offer/Answer**», **modelo** genérico y simple para el intercambio de

descripciones de sesiones, según el cual, el mensaje de oferta enviado por un UAC especifica las características deseadas de la sesión, mientras que el UAS receptor del mensaje genera una respuesta indicando las características admitidas.

Las funciones básicas de los servidores **SIP** son la localización de usuarios y la resolución de nombres. Puesto que generalmente los agentes de usuario clientes no conocen la dirección IP del destinatario de una llamada, sino su nombre de usuario o un número de teléfono al que habrá que acceder a través de un gateway, necesitan enviar en primer lugar un mensaje de invitación al servidor correspondiente al nombre o número para que localice al destinatario. El servidor puede conocer la dirección del destinatario o recurrir a otros servidores para continuar la búsqueda. Cuando las llamadas se redirigen, la ruta seguida se registra en los mensajes **SIP**, de modo que a la hora de generar respuestas se pueda conocer el camino de retorno hasta el origen del mensaje inicial.

Los servidores **SIP** actúan generalmente como varios tipos de servidores de forma simultánea. Gracias a una infraestructura de servidores **SIP**, es posible gestionar las llamadas de forma distribuída entre equipos personales, equipos de proveedores de servicios y pasarelas corporativas, con la consiguiente flexibilidad y control por parte del usuario, que puede

mantener la privacidad de sus datos personales en todo momento. Asimismo, un mensaje **SIP** puede pasar por un número indeterminado de servidores desde que un agente de usuario cliente lo envía hasta que llega al agente de usuario servidor destinatario. Los tipos de servidores **SIP** definidos hasta el momento son los siguientes:

• **Servidores de registro.-** El uso más común de estos servidores es registrar un dispositivo después de su arranque, de modo que cuando lleguen invitaciones destinadas a él, los servidores **SIP** puedan proporcionar su dirección. Se contempla la existencia de un tiempo máximo de validez de cada registro, definible por el servidor, tras el cual se debe renovar el registro.

Asimismo, existen mecanismos para cancelar todos los registros contenidos en un servidor.

• **Proxys.-** Su función es similar a la de los proxys HTTP: recibir solicitudes y decidir a qué otro servidor se deben remitir, alterando además algunos campos de la solicitud. Por tanto, actúan como intermediarios en las transacciones que procesan. Normalmente, los proxys actúan como servidores de registro para todos los dispositivos representados por ellos. Para ello, aceptan y procesan peticiones de registro (método REGISTER).

Téngase en cuenta que el elemento al que el proxy reenvía la petición sólo es el agente de usuario servidor destinatario en el caso más simple, pudiendo ser tanto otro servidor proxy como un servidor de redirección. Asimismo, el proxy puede recurrir a un servidor de localización para determinar la dirección en la que actualmente está disponible el destinatario.

Los proxys son, al igual que los proxys HTTP, particularmente útiles como representantes de salida/entrada de y a redes corporativas, proporcionando servicios de búsqueda de direcciones, control de cortafuegos y gestión de normas de administración corporativas. Asimismo, pueden cumplir funciones de control de salida a pasarelas para redes telefónicas tradicionales.

• **Servidores de redirección.-** A diferencia de los proxys, no inician transacciones, sino que, cuando reciben solicitudes desde un agente de usuario cliente, remiten al mismo agente un mensaje indicando el o los servidores con los que debe ponerse en contacto, en un procedimiento similar al de búsqueda iterativa del sistema DNS. Asimismo, a diferencia de los agentes de usuario servidores, no aceptan llamadas.

Normalmente, los servidores de redirección gestionan mayor número de mensajes que los proxys, pero con menores necesidades de

procesamiento. Nótese que, puesto que en sesiones controladas por **SIP** la redirección se realiza mediante mensajes **SIP**, las respuestas se pueden generar con flexibilidad y adecuación a servicios de conferencia multimedia, modificándose en función de parámetros tales como la hora del día, el origen o urgencia de la llamada, o cualquier otro criterio específico aplicado por el servidor **SIP**.

Los servidores pueden mantener el *estado de las llamadas* a dos niveles de detalle o no mantener estado alguno. Al igual que en la red telefónica convencional, pueden mantener el estado completo de cada llamada; sin embargo, este comportamiento es opcional, puesto que limitaría la escalabilidad de los sistemas **SIP**. El modo de funcionamiento recomendado y más frecuente es que los servidores mantengan únicamente el estado de cada transacción por separado. Las transacciones **SIP** se pueden definir como *el conjunto de peticiones y respuestas intercambiadas desde que un agente de usuario cliente envía una petición hasta que recibe una respuesta definitiva originada en el agente de usuario servidor que recibió la petición inicia*l. Por éste motivo, si los servidores sólo mantienen el estado de cada transacción por separado, no tienen conocimiento de las llamadas existentes en un cierto instante. Por tanto, los servidores no tienen que mantener una máquina de estados para cada llamada, constituyendo así un sistema altamente escalable. Asimismo, el comportamiento de los

servidores **SIP** en cuanto al mantenimiento de estados se puede modificar de forma dinámica en función de las circunstancias.

1.3.5. Mensajes SIP – métodos y respuestas

Métodos SIP:

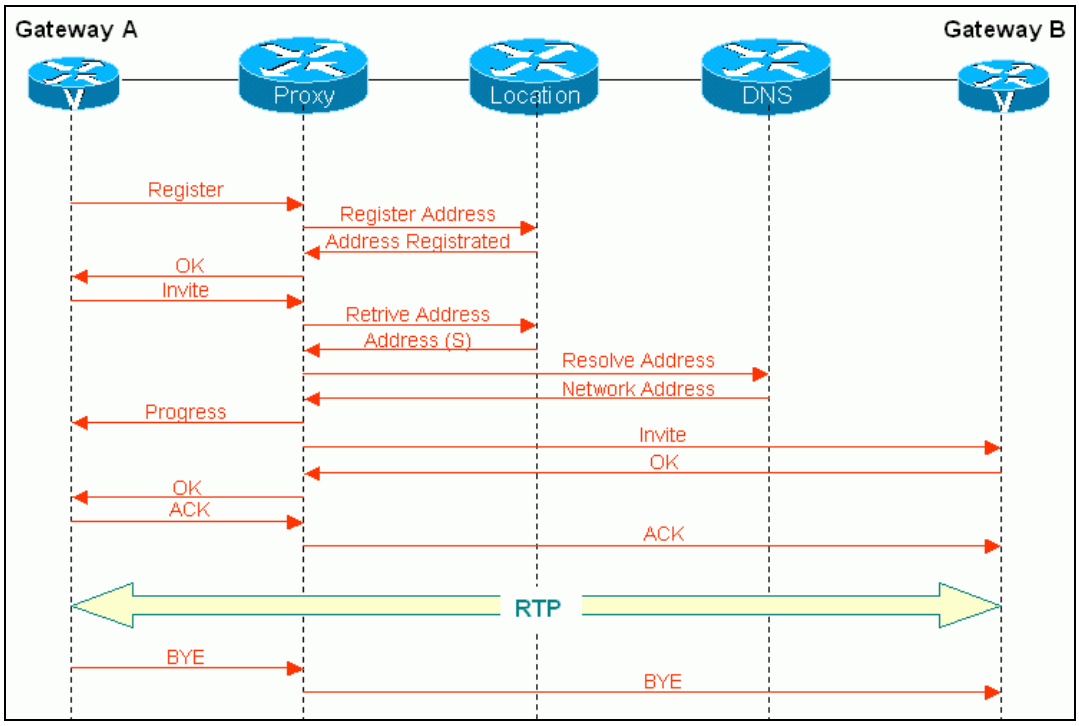


Figura 11. *Intercambio de mensajes para establecer una comunicación con protocolo SIP.*

INVITE: Inicia una llamada invitando a un usuario a participar en una sesión.

ACK: Confirma que el cliente ha recibido una respuesta final a un método INVITE.

BYE: Indica la terminación de la llamada.

CANCEL: Cancela un requerimiento pendiente.

REGISTER: Registra al user agent.

OPTIONS: Usado para consultar las capacidades de un servidor.

INFO: Usado para transportar información fuera de banda, como dígitos DTMF.

MESSAGE: Transporta mensajes de texto entre user agents.

REFER: Solicita generar una sesión desde una tercera parte.

SUSCRIBE: Suscribe al user agent a ser notificado sobre eventos que ocurran en otro user agent.

NOTIFY: Notifica los eventos suscriptos.

UPDATE: Modifica elementos del diálogo activo.

PRACK: Confirmación provisoria.

PUBLISH: Publica la notificación de eventos.

Respuestas SIP:

1xx – Mensajes provisionales

100 Trying

180 Ringing

183 Session Progress

2xx – Respuestas de éxito

200 OK

202 Accepted

3xx – Respuestas de redirección

300 Multiples Choices

301 Moved Permanently

302 Moved Temporarily.

4xx – Respuestas de falla de método

400 Bad request

401 Unauthorized

404 Not found

407 Proxy authentications required

486 Busy here

487 Request terminated

5xx – Respuestas de fallas de servidor

500 Sever internal error

502 Bad gateway

6xx – Respuestas de fallas global

600 Busy everywhere

603 Decline

Los componentes SIP se comunican intercambiando mensajes SIP:

Encabezado SIP

SIP toma prestado mucha de la sintaxis y semántica de HTTP. Un mensaje SIP se ve como un mensaje HTTP – Formateo de mensaje, encabezado y soporte MIME.

Un ejemplo de encabezado SIP:

Método:

INVITE sip:bob@biloxi.com SIP/2.0

Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds

Max-Forwards: 70

To: Bob <sip:bob@biloxi.com>

From: Alice <sip:alice@atlanta.com>;tag=1928301774

Call-ID: a84b4c76e66710@pc33.atlanta.com

CSeq: 314159 INVITE

Contact: <sip:alice@pc33.atlanta.com>

Content-Type: application/sdp

Content-Length: 142

Respuesta:

SIP/2.0 200 OK

Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds

To: Bob <sip:bob@biloxi.com>;tag=a6c85cf

From: Alice <sip:alice@atlanta.com>;tag=1928301774

Call-ID: a84b4c76e66710@pc33.atlanta.com

CSeq: 314159 INVITE

Contact: <sip:bob@192.0.2.4>

Content-Type: application/sdp

Content-Length: 131

Direcciones SIP

Las direcciones SIP están identificadas por una URI (Uniform Resource Identifier) con la forma: user@host.

Ejemplos de URIs SIP:

sip:mstokle@nortelnetworks.com

sip:bob@192.168.10.1

sip:14083831088@gateway.nortel.com

sips:mstokle@nortelnetworks.com

Los proxy server pueden resolver y transformar URIs del tipo tel, que contienen direcciones E.164

tel:+541148277237

Dependiendo del tipo de user agent, estos también pueden utilizar otros tipos de URIs como http: o mailto:

http://www.nortelnetworks.com

mailto:mstokle@nortelnetworks.com

Las URIs se diferencian de las URLs en que estas últimas apuntan a una ubicación física específica (ejemplo: un archivo)

1.3.6. Proceso para establecer una comunicación

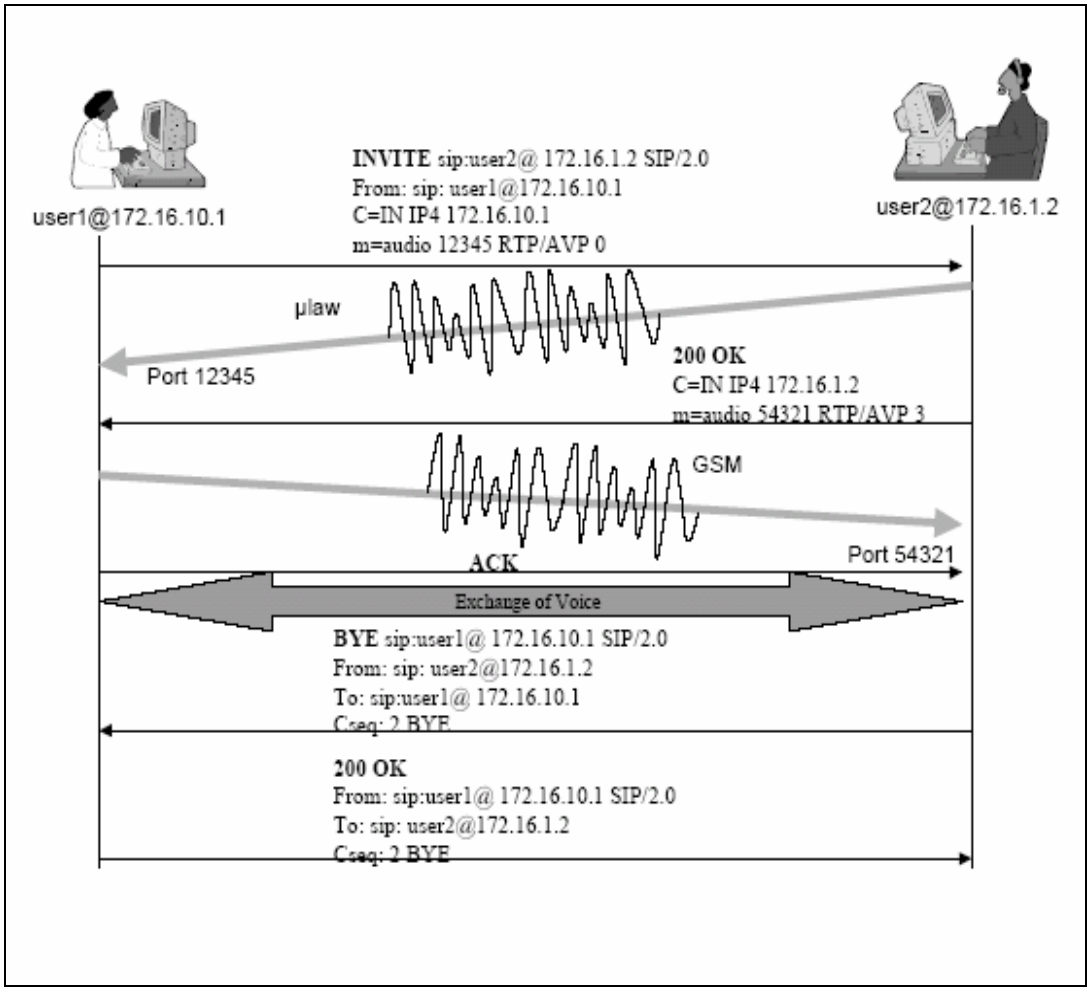


Figura 12. *Establecimiento de comunicación usando SIP.*

Establecer una comunicación usando SIP ocurre usualmente en 6 pasos:

1. Localización del usuario.

2. Determinación del medio a utilizar. Se efectúa por medio de un modelo de oferta/respuesta por intermedio de SDP (Session Description Protocol)

3. Determinación de la parte llamada de aceptar la llamada. Aceptar o rechazar.

4. Establecimiento del medio.

5. Modificación de la llamada o manejo de la misma. Ejemplo, transferencia.

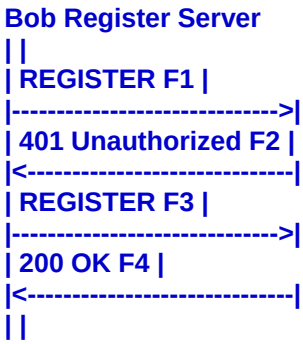
6. Terminación de la llamada.

Flujos de ejemplo: Registro

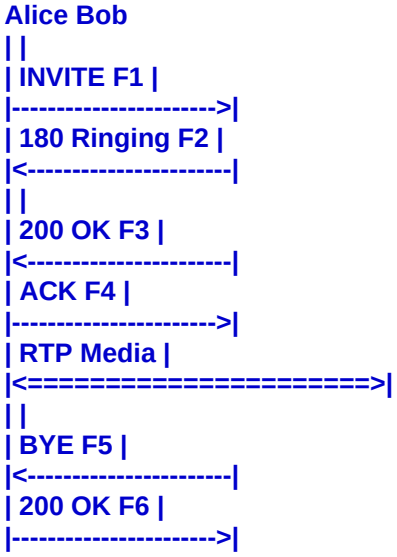
Cada vez que el usuario enciende su dispositivo (Teléfono SIP, PC, u otro dispositivo SIP), el cliente se registra con el registrar server. La registración también ocurre cuando el usuario modifica su locación física o enciende un nuevo dispositivo.

La información de registro se refresca periódicamente y cada usuario debe de registrarse con el registrar server. El registrar server actualiza la base de datos del location server. La registración puede hacerse por otros

medios (páginas web, scripts, admin, etc.). La direccion del server enlaza la dirección de registro del tipo sip:mstokle@nortelnetworks.com con las direcciones físicas de los dispositivos, del tipo sip:mstokle@47.46.208.11 o sip:mstokle@pda.mstokle.cala.nortel.com

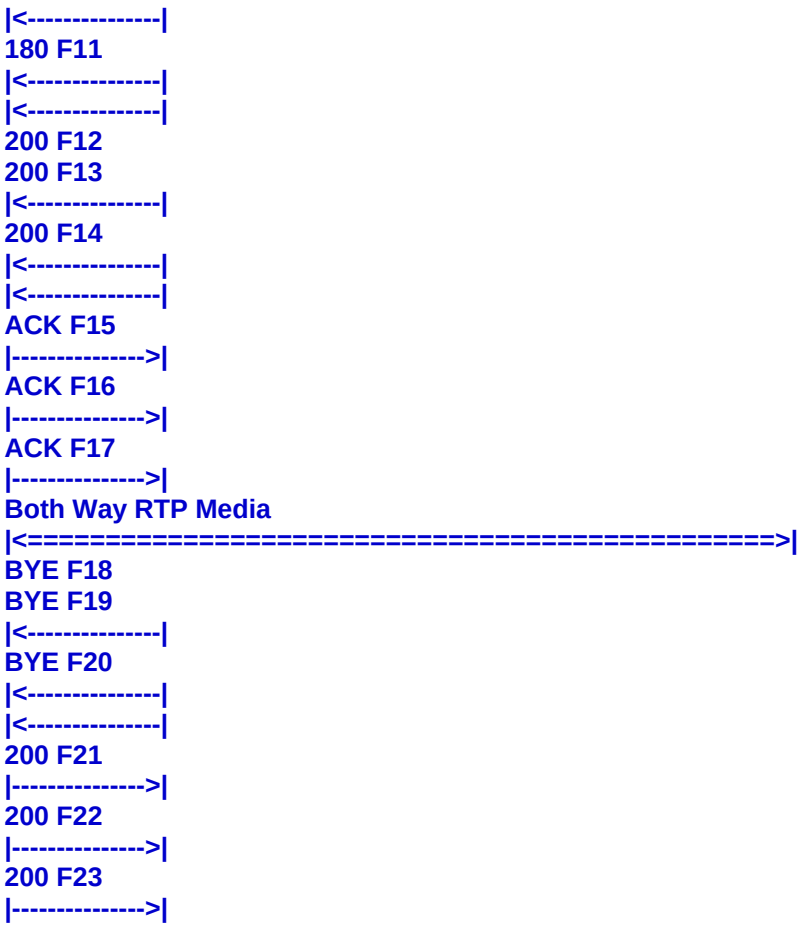


Flujos de ejemplo: Llamada básica

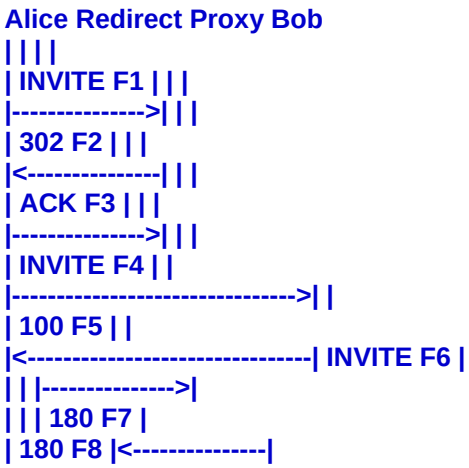


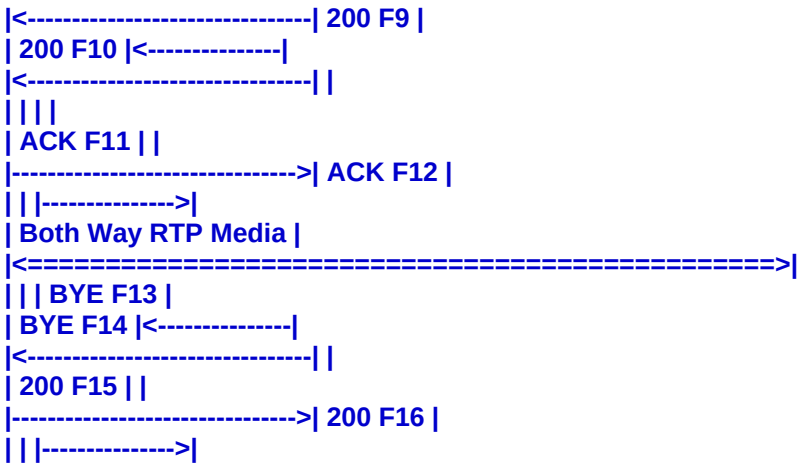
En éste caso, Alice y Bob conocen su locación actual y pueden contactarse directamente, Alice envía un INVITE a Bob. En el cuerpo del INVITE ofrece sus capacidades de media usando SDP (audio, video, juegos, etc). El user agent de Bob envía una respuesta provisional (180 Ringing), una vez que el usuario contesta la llamada, el user agent envía la respuesta definitiva (200 OK) y en el cuerpo de esa respuesta se envía la respuesta sobre la media utilizar usando SDP. El user agent de Alice envía el ACK y se establece el path de media utilizando los **protocolos** apropiados (RTP en éste caso). Bob corta la llamada, su user agent envía un BYE. El user agent de Alice envía su respuesta exitosa (200 OK).

Flujos de ejemplo: Llamadas con proxy

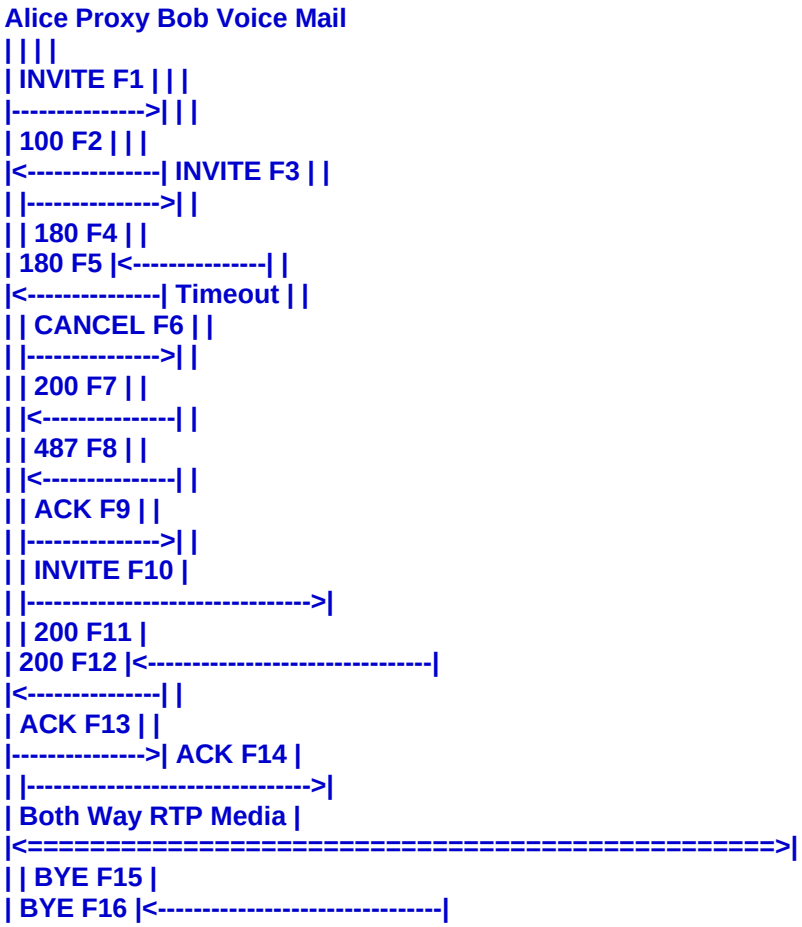


Flujos de ejemplo: Llamadas con redirect





Flujos de ejemplo: CFNA al Voice Mail



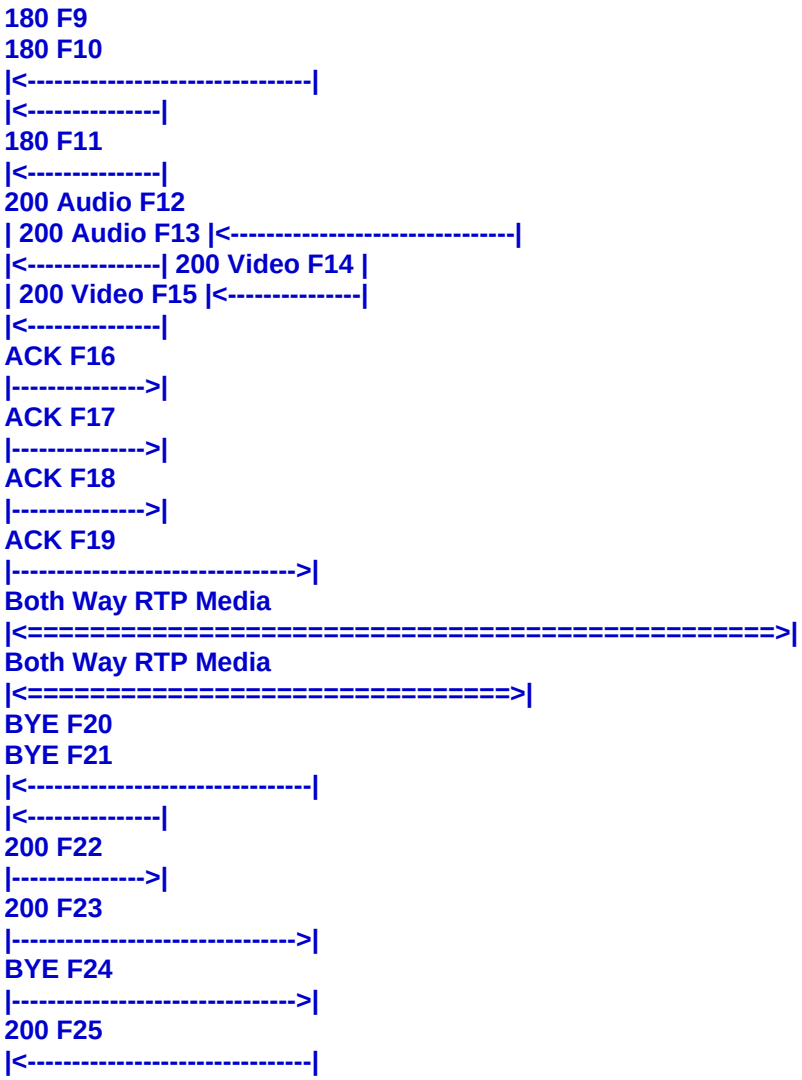
```
|<---------------| | |
| 200 F17 | | |
|--------------->| 200 F18 |
| |-------------------------------->|
```

Flujos de Ejemplo: MWI usando Suscribe

```
Alice Voice Mail
| SUBSCRIBE (new) F1 |
|---------------------->|
| 200 OK F2 |
|<----------------------|
| NOTIFY (sync) F3 |
|<----------------------|
| 200 OK F4 |
|---------------------->|
| NOTIFY (change) F5 |
|<----------------------|
| 200 OK F6 |
|---------------------->|
| (re)SUBSCRIBE F7 |
|---------------------->|
| 200 OK F8 |
|<----------------------|
| NOTIFY (sync) F9 |
|<----------------------|
| 200 OK F10 |
|---------------------->|
```

Flujos de ejemplo: Forking a converged desktop

```
Alice Proxy 1 Bob PC Client Bob Teléfono
|
INVITE F1
|--------------->|
407 F2
|<---------------|
ACK F3
|--------------->|
| INVITE (A/V) F4 |
|--------------->|
100 F6
INVITE (A/V) F7
|<---------------|-------------------------------->|
| INVITE (A/V) F8 |
|--------------->|
```



1.4. Optimización de canales de datos para telefonía.

1.4.1. Problemas en una red IP.

Dos son los mitos que involucran a la telefonía IP. Uno se refiere a la baja calidad de Internet. Se confunden las prestaciones de los accesos dial-up con el uso de canales de transporte punto-a-punto con calidad contratada.

Otro se refiere al medio de transportar a los paquetes IP. Aquí se menciona que solo ATM está en condiciones de garantizar la calidad de servicio. Nuevamente se ignora la serie de herramientas que posee una red IP y Gigabit-Ethernet para garantizar una calidad de servicio.

Los problemas que son evidentes en una red de VoIP, son la latencia, el jitter y el eco. En telefonía-IP estos problemas son resueltos mediante diversas técnicas.

Latencia.

Cuando diseñamos redes que transportan voz en paquetes, marcos, o infraestructura de célula, es importante entender todos los posibles causales de retardos teniendo en cuenta cada uno de los factores, es posible mantener la red en un estado aceptable. La calidad de la voz es función de muchos factores, como lo son, los algoritmos de compresión, los errores y las perdidas de tramas, la cancelación del eco y los retardos. A continuación se esbozan los posibles retardos para VoIP y algunos apartes de la recomendación G.114 de la UIT.

Rango(ms)	Descripción
0-150	Aceptable para las aplicaciones más comunes.
150-400	Aceptable, teniendo en cuenta que un administrador de red conozca las necesidades del usuario.
Sobre 400	Inaceptable para la mayoría de planeaciones de red, sin embargo, este límite puede ser excedido en algunos casos aislados.

Tabla 2. *Limite de los retardos según UIT G.114.*

Estas recomendaciones se estipulan para conexiones con control de eco adecuado, eso implica el uso de equipos canceladores de eco. Estos equipos son requeridos cuando el retardo de una vía excede los 25 ms. (UIT G.131)

Fuentes del retardo.

Se clasifican en dos tipos:

- Retardo fijo, se adiciona directamente al total del retardo de la conexión.
- Retardo variable, se adiciona por demoras en las colas de los buffer, se nota como (Δn).

A continuación se identifican todos los posibles retardos, fijos o variables, en una red.

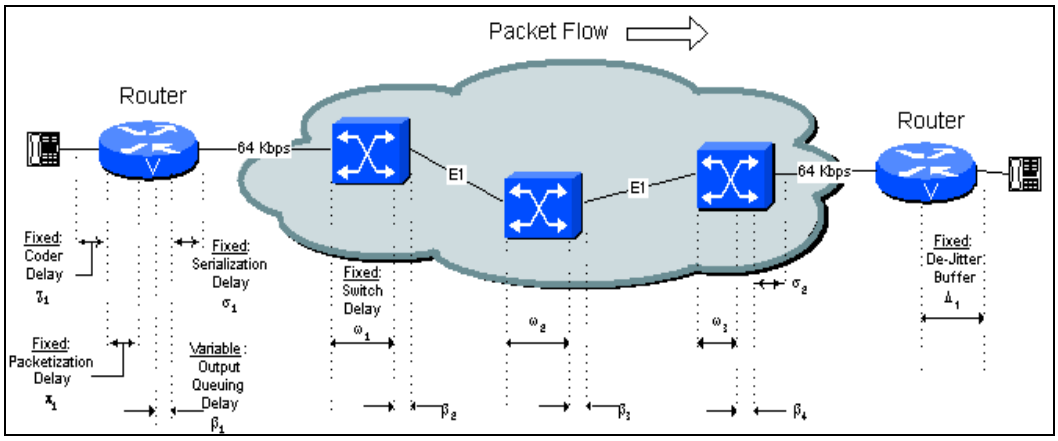


Figura 13. *Retardos en una comunicación VoIP.*

Retardo por codificación.- También llamado retardo de proceso (χn), es el tiempo que tarda el DSP en comprimir un bloque de muestras PCM, como los codificadores trabajan en diferentes formas, este retardo varia dependiendo del codificador de voz y de la velocidad y carga del procesador.

Codificador	Rata (Kbps)	Tamaño de muestra (ms)	Mejor opción (ms)	Peor opción (ms)
ADPCM, G.726	32	10	2.5	10
CS-ACELP, G.729A	8.0	10	2.5	10
MP-MLQ, G.723.1	6.3	30	5	20
MP-ACELP, G.723.1	5.3	30	5	20

Tabla 3. *Mejor y peor alternativa de retardo por codificación.*

Retardo algorítmico.- El algoritmo de la compresión, que depende de características conocidas de voz para procesar correctamente el bloque N de la muestra, debe tener algún conocimiento de lo que está en el bloque N + 1 en reproducir exactamente el bloque de la muestra N. Esta mirada adelante, que es realmente una demora adicional, se llama la demora algorítmica y aumenta efectivamente la longitud del bloque de la compresión.

El retardo acumulado del codificador se rige por la siguiente ecuación.

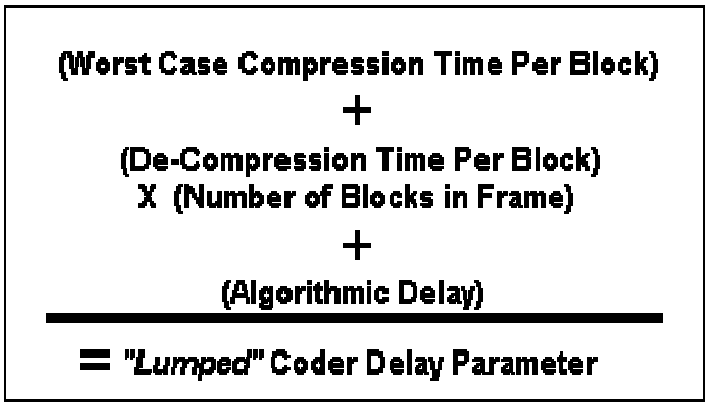


Figura 14. *Fórmula para cálculo del retardo.*

***Retardo por paquetización*.-** Es la demora para llenar un paquete de información, carga útil, de la conversación codificada y comprimida. Este retardo es función del tamaño de bloque requerido por el codificador de voz y el número de bloque de una sola trama.

Codificador	Rata Kbps	Carga útil (Bytes)	Retardo de paquetización (ms)	Carga útil (Bytes)	Retardo de paquetizacion (ms)
PCM, G.711	64	160	20	240	30
ADPCM, G.726	32	80	20	120	30
CS-ACELP, G.729	8.0	20	20	30	30
MP-MLQ, G.723.1	6.3	24	24	60	48
MP-ACELP, G.723.1	5.3	20	30	60	60

Tabla 4. *Retardos de paquetización más comunes.*

Cuando cada muestra de voz experimenta, ambos retardos, retardo algorítmico y retardo por paquetización, en realidad, los efectos se superponen.

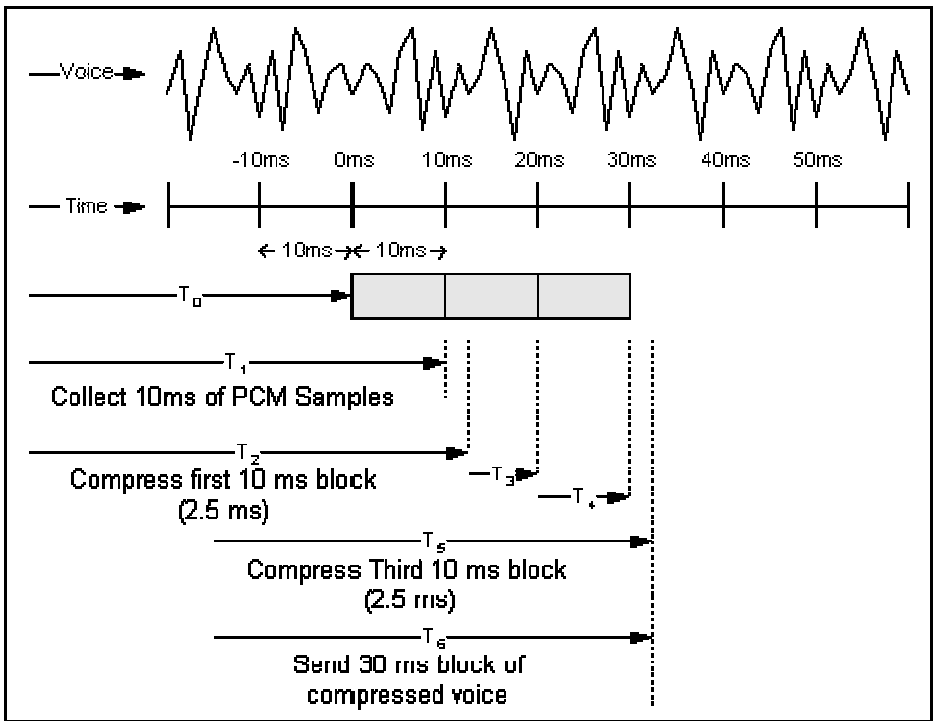


Figura 15. *Retardo por paquetización.*

Retardo de serialización.- Es un retardo fijo dependiente de los relojes del muestreo de la voz, o de las tramas de red, está relacionado directamente a la tasa del reloj de la transmisión. Recuerde que con reloj bajo y tramas pequeñas, se debe adicionar banderas extras para separar tramas significativas.

Tamaño de trama (bytes)	Velocidad de línea (Kbps)										
	19.2	56	64	128	256	384	512	768	1024	1544	2048
38	15.83	5.43	4.75	2.38	1.19	0.79	0.59	0.40	0.30	0.20	0.15
48	20.00	6.86	6.00	3.00	1.50	1.00	0.75	0.50	0.38	0.25	0.19
64	26.67	9.14	8.00	4.00	2.00	1.33	1.00	0.67	0.50	0.33	0.25
128	53.33	18.29	16.00	8.00	4.00	2.67	2.00	1.33	1.00	0.66	0.50
256	106.67	36.57	32.00	16.00	8.00	5.33	4.00	2.67	2.00	1.33	1.00
512	213.33	73.14	64.00	32.00	16.00	10.67	8.00	5.33	4.00	2.65	2.00
1024	426.67	149.29	128.00	64.00	32.00	21.33	16.00	10.67	8.00	5.31	4.00
1500	625.00	214.29	187.50	93.75	46.88	31.25	23.44	15.63	11.72	7.77	5.86
2048	853.33	292.57	256.00	128.00	64.00	42.67	32.00	21.33	16.00	10.61	8.00

Tabla 5. *Demora de serialización para diferentes tamaños de tramas.*

Retardo por Cola/Buffering.- Posteriormente a la compresión de la información, se adiciona un encabezado, y se apila para trasmitirse a la red, como los paquetes de voz tienen prioridad para el enrutador, una trama de voz solo debe esperar cuan do otra trama de voz este siendo atendida. Por tanto éste retardo solo depende del estado de la cola y la velocidad del enlace.

Retardo por conmutador de red.- Las redes publicas de Frame Relay o ATM conectan nodos finales y son las causantes de los grandes retardos de las conexiones de voz, a su vez son los más complejos de cuantificar.

Retardo en el buffer estabilizador.- Como la conversación es un servicio de rata constante de transmisión, las inestabilidades de todos los posibles retardos deben ser descartadas cuando la señal abandone la red, éste buffer especial de los ruteadores de CISCO, permite transformar un retardo variable en uno fijo, con el fin de excluir variables inestables de retardo.

Jitter.- Es el efecto por el cual el retardo entre paquetes no es constante. Se trata de una latencia variable producida por la congestión de tráfico en el backbone de red, por distinto tiempo de tránsito de paquetes debido al *connectionless*, etc. Se puede utilizar un buffer para distribuír los paquetes y reducir el jitter, pero introduce un retardo adicional. Lo correcto es incrementar el ancho de banda del enlace; solución posible en un backbone pero de menor posibilidad en los enlaces WAN. Otra posibilidad es la formación de colas para prioridad de tráfico de telefonía sobre los de datos.

Eco.- Las características anteriores (latencia y jitter) pueden producir eco sobre la señal telefónica, lo cual hace necesario el uso de canceladores de eco (ITU G.168). Se tienen 2 tipos de eco. Uno tiene alto nivel y poco retardo

y se produce en el circuito híbrido de 2 a 4 hilos local; mientras que otro es de bajo nivel y gran retardo y se produce en el circuito separador híbrido remoto. El cancelador de eco se construye mediante la técnica de ecualización transversal autoadaptativa. Consiste en usar una parte de la señal de transmisión para cancelar el eco producido por la desadaptación de impedancias en el circuito híbrido que convierte de 4 a 2 hilos.

Throughput.- Es la capacidad de un enlace de transportar información útil. Representa a la cantidad de información útil que puede transmitirse por unidad de tiempo. No tiene relación directa con el delay. (Por ejemplo, se puede tener un enlace de alto throughput y alto delay o viceversa, como sería por ejemplo un enlace satelital de 2Mbps y 500 mseg de delay).

Packet Loss.- Es la tasa de perdida de paquetes. Representa el porcentaje de paquetes transmitidos que se descartan en la red. Estos descartes pueden ser producto de alta tasa de error en alguno de los medios de enlace o por sobrepasarse la capacidad de un buffer de una interfaz en momentos de congestión. Los paquetes perdidos son retransmitidos en aplicaciones que no son de Tiempo Real; en cambio para telefonía, no pueden ser recuperados y se produce una distorsión vocal.

El delay afecta a la performance de aplicaciones interactivas (por ejemplo, Telnet). El throughput afecta a la performance de aplicaciones que mueven grandes volúmenes de información (por ejemplo, Mail y FTP). El packet loss afecta a ambos tipos de aplicaciones. El jitter afecta a aplicaciones de tiempo real como la voz y el video por IP.

1.4.2. QoS (Calidad de servicio)

Esta función tiene primordial importancia en relación con la QoS experimentada por el usuario final. En esto influyen dos factores fundamentales:

- La calidad de la voz extremo a extremo, determinada por los sucesivos procesos de codificación – decodificación, y las pérdidas de paquetes en la red.
- La demora extremo a extremo, debido a las sucesivos procesos de codificación – decodificación, paquetización y "encolados". Afecta la interactividad en la conversación, y por tanto a la QoS.

Las redes IP son redes del tipo best-effort y por tanto no ofrecen garantía de QoS, pero las aplicaciones de telefonía IP si necesitan algún tipo de garantía de QoS en términos de demora, jitter y pérdida de paquetes. En

tal sentido existen dos mecanismos se señalización para QoS, esto es, IntServ y DiffServ. Ambos son "mecanismos" de cara a la red.

Por tanto, es necesario buscar QoS no solo en la red, sino también en los terminales, y en los procesos que en los mismos se desarrollan, de ahí que sea necesario también decir que la sensibilidad a la pérdida de paquetes, a las demoras y sus fluctuaciones, que experimentan los servicios de voz sobre IP, dependen en buena medida de los mecanismos implementados en los terminales.

La preparación de los medios en los terminales para ser enviados y transferidos por la red IP involucra varios procesos: digitalización, compresión y empaquetado en el extremo emisor, y los procesos inversos en el extremo receptor. Todo esto se lleva a cabo mediante un complejo procesamiento que sigue determinado algoritmo, lo cual a su vez se desarrolla en cierto intervalo de tiempo, esto es, implica demora de procesamiento y demora de empaquetado:

- Demora de procesamiento: demora producida por la ejecución del algoritmo de codificación, que entrega un stream de bytes listos para ser empaquetados;

- Demora de paquetización: es el tiempo que se requiere para formar un paquete de voz a partir de los bytes codificados.

Debe señalarse que el resultado de esta codificación – paquetización incide directamente en la QoS, y también la forma en que se lleve a cabo. Así, cuando se reduce la velocidad de codificación los requerimientos de ancho de banda también se reducen, lo que posibilita de cara a la red poder manejar más conexiones simultáneas, pero se incrementa la demora y la distorsión de la señales de voz. Lo contrario ocurre al aumentar la velocidad de codificación.

Otro aspecto a tener en cuenta es el compromiso entre la demora de paquetización y la utilización del canal (relación entre bytes de información y bytes de cabecera en cada paquete de voz), es decir, la búsqueda de mayor utilización del canal conduce a mayor demora de paquetización para cierto estándar de codificación. Claro está, según el estándar de codificación que se utilice será la demora resultante en relación con la utilización del canal, diferencias que se acentúan cuando la utilización del canal está por encima del 50%, con un crecimiento de la demora en forma exponencial en el caso de los codecs de baja velocidad como el G.723.1. La demora de paquetización también puede ser reducida mediante multiplexación de varias conexiones de voz en el mismo paquete IP.

A las demoras de procesamiento y empaquetado se suma también la demora que introduce el proceso de buffering en los terminales, y la demora de "encolado" en la red. Todo esto da una demora extremo a extremo que percibe el usuario final en mayor o menor medida. Demoras extremo a extremo por debajo de 400 milisegundos no comprometen la interactividad en la conversación, pero por encima de 150 milisegundos se requiere control del eco. Las demoras antes comentadas son resultado lógico de las características y modo de operación de las redes IP, así como también de la naturaleza de las señales de voz.

La calidad de servicio (QoS) es el rendimiento de extremo a extremo de los servicios electrónicos tal como lo percibe el usuario final. Los parámetros de QoS son: el retardo, la variación del retardo y la pérdida de paquetes. Una red debe garantizar que puede ofrecer un cierto nivel de calidad de servicio para un nivel de tráfico que sigue un conjunto especificado de parámetros.

La implementación de políticas de calidad de servicio se puede enfocar en varios puntos según los requerimientos de la red, los principales son:

- Asignar ancho de banda en forma diferenciada.

- Evitar y/o administrar la congestión en la red.

- Manejar prioridades de acuerdo al tipo de tráfico.

- Modelar el tráfico de la red.

Como se ha dicho, la comunicación sobre IP (al igual que la telefonía convencional) debe tener características de tiempo real, desafortunadamente TCP/IP no puede garantizar éste tipo de particularidad siempre, de modo que se deben introducir algunas políticas que puedan manejar el flujo de paquetes en todos los enrutadores que deban intercambiar paquetes. Estas son:

Campo tos en el protocolo IP para describir el tipo de servicio: los altos valores indican poca urgencia, mientras que los más bajos indicarán urgencia, es decir que se solicita respuesta en tiempo real.

Métodos de solución para paquetes en cola:

FIFO (first in first out), es el método más común, donde sale primero el paquete que llegó en primer lugar.

WFQ (weighted fair queuing), consiste en un paso justo de paquetes en consideración con el ancho de banda disponible (por ejemplo, FTP no puede consumir todo el ancho de banda disponible del enlace en cuestión), dependiendo del tipo de flujo de datos que se esté dando, por ejemplo en un ambiente justo, porcada paquete UDP habrá uno TCP.

CQ (custom queuing), donde los usuarios deciden la prioridad del paquete.

PQ (priority queuing), se establece un número de colas (típicamente 4), cada una con una nivel de prioridad diferente: se comienza enviando los paquetes de la primera cola y luego (cuando la primera cola está vacía) se envían los paquetes de la segunda cola y así sucesivamente.

CB-WFQ (class based weighted fair queuing), es muy similar a WFQ pero se adiciona el concepto de clases (hasta 64) y además un valor de ancho de banda es asociado.

Capacidad de limitación, la cual permite restringir a la fuente llegar a un ancho de banda determinado para:

- Descarga (download).
- Carga (upload).

- Prevención de congestión.

1.5. Seguridades VoIP.

Desafortunadamente, las nuevas tecnologías traen también consigo detalles a tener en cuenta respecto a la seguridad. De pronto, se presenta la necesidad de proteger dos infraestructuras diferentes: voz y datos.

Los dispositivos de redes, los servidores y sus sistemas operativos, los protocolos, los teléfonos y su software, todos son vulnerables.

La información sobre una llamada es tan valiosa como el contenido de la voz. Por ejemplo, una señal comprometida en un servidor puede ser usada para configurar y dirigir llamadas, del siguiente modo: una lista de entradas y salidas de llamadas, su duración y sus parámetros. Usando esta información, un atacante puede obtener un mapa detallado de todas las llamadas realizadas en una determinada red, creando grabaciones completas de conversaciones y datos de usuario y poder retransmitir todas las conversaciones sucedidas en la red. La conversación es en sí misma un riesgo y el objetivo más obvio de una red VoIP. Consiguiendo una entrada en

una parte clave de la infraestructura, como una puerta de enlace de VoIP, se pueden capturar y volver a montar paquetes con el objetivo de escuchar una conversación.

Las llamadas son también vulnerables al "secuestro". En éste escenario, un atacante puede interceptar una conexión y modificar los parámetros de la llamada. Se trata de un ataque que puede causar bastante pavor, por que las víctimas no notan ningún tipo de cambio. Las posibilidades incluyen diversas técnicas como robo de identidad, y redireccionamiento de llamada, haciendo que la integridad de los datos estén bajo un gran riesgo.

La enorme disponibilidad de las redes VoIP es otro punto sensible. En PSTN, la disponibilidad era raramente un problema. Una pérdida de potencia puede provocar que la red se caiga por lo que es mucho más sencillo hackear una red VoIP. Los efectos demoledores de los ataques traen como consecuencia la denegación de servicio. Si se dirigen a puntos clave de la red, podrían incluso destruir la posibilidad de comunicación vía voz o datos.

Los teléfonos y servidores son blancos por sí mismos. Aunque sean de menor tamaño o parezcan elementos simples, son en base, ordenadores con software. Obviamente, éste software es vulnerable con los mismos tipos de falencias de seguridad que pueden hacer que un sistema operativo pueda

estar a plena disposición del intruso. El código puede ser insertado para configurar cualquier tipo de acción maliciosa.

En resumidas cuentas, los riesgos que comporta usar el protocolo VoIP no son muy diferentes de los que nos podemos encontrar en las redes habituales de IP. Desafortunadamente, en los esquemas iniciales y en diseños de hardware para voz, software y protocolos, la seguridad no es su punto fuerte.

Internet, generalmente es poco confiable para transportar voz de alta calidad telefónica, porque los actuales protocolos TCP/IP no proveen reservas de ancho de banda ni garantizan la calidad del servicio. Por consiguiente, la calidad de las llamadas sobre IP serán adversamente afectadas por la congestión de la red que origina que los paquetes se tarden o se pierdan. Un ambiente como una red pública Internet, está marcada por una incontrolable y dramática fluctuación de carga, razón por la cual no puede garantizar una conexión de voz aceptable.

La encriptación es la única forma de prevenirse de un ataque, desafortunadamente se consume ancho de banda. Existen múltiples métodos de encriptación: VPN (virtual personal network), SRTP (secure RTP). La clave, de cualquier forma, es elegir un algoritmo de encriptación rápido,

eficiente, y emplear un procesador dedicado de encriptación. Otra opción podría ser QoS (quality of service); los requerimientos para QoS asegurarán que la voz se maneja siempre de forma oportuna, reduciendo la pérdida de calidad.

Estas limitaciones de los servicios de voz basados en IP, están siendo solucionadas por nuevos protocolos que proveen diferentes clases de servicios o prioridades de paquetes y la habilidad de reservar ancho de banda a través de la red para la duración de una llamada telefónica. Nuevos protocolos para tráfico, otorgan la habilidad, no sólo de destinar ancho de banda por prioridad de paquetes, sino que también dan preferencia al procesamiento de los mismos dentro de los límites del enrutador (enrutador), de forma que los paquetes de alta prioridad son procesados primero. Estas mejoras a los algoritmos y protocolos en los enrutadores y conmutadores están reduciendo la tenencia y la pérdida de paquetes para lograr una mejor calidad de servicio, y estos avances han comenzado a permitir a los proveedores de servicios de VoIP encontrar los estándares necesarios para servicios de voz.

Es preciso tener en cuenta la certeza de todos los elementos que componen la red VoIP: servidores de llamadas, enrutador, switches, centros de trabajo y teléfonos. Se necesita configurar cada uno de esos dispositivos

para asegurarse de que están en línea con las demandas en términos de seguridad. Los servidores pueden tener pequeñas funciones trabajando y sólo abiertos los puertos que sean realmente necesarios. Los enrutador y switches deben estar configurados adecuadamente, con acceso a las listas de control y a los filtros. Todos los dispositivos deben estar actualizados. Se trata del mismo tipo de precauciones que es necesario tomar cuando se añaden nuevos elementos a la red de datos; únicamente habrá que extender éste proceso a la porción que le compete a la red VoIP.

Es posible emplear un firewall y un IDS (intrusion detection system) para ayudar a proteger la red de voz. Los firewalls de VoIP son complicados de manejar y tienen múltiples requerimientos. Los servidores de llamada están constantemente abriendo y cerrando puertos para las nuevas conexiones. Este elemento dinámico hace que su manejo sea más complicado. No obstante, el costo es equiparable la cantidad de beneficios. Se debe prestar especial atención al perfeccionamiento los controles de acceso. Un IDS puede monitorizar la red para detectar cualquier anomalía en el servicio o un abuso potencial. Las advertencias son una clave para prevenir los ataques posteriores.

Sin embargo, las redes privadas basadas en IP pueden proporcionar alta calidad de servicios de voz. Adicionalmente al uso de las capacidades de

la red, planeando y activando el manejo de las cargas para evitar la congestión, estas redes pueden aprovechar las ventajas de las mejoras realizadas a los protocolos TCP/IP, que permiten asignar altas prioridades para tráfico en tiempo real (como la voz) a diferencia de la rata tradicional.

Las redes de conmutación por paquetes pueden transportar llamadas de voz eficientemente, utilizando un ancho de banda de 8 kbps que provee de alta calidad telefónica, comparadas a las redes de conmutación de circuitos (tradicionales) que hacen uso de un ancho de banda de 64 kbps. Además, los costos de infraestructura, asociados a la implementación de redes de conmutación por paquetes, son mucho más bajos que las alternativas tradicionales. Como resultado, nuevos proveedores de servicios telefónicos, están utilizando cada vez más éste tipo de arquitecturas.

1.6. Aplicaciones VoIP

El IP-PBX es una implementación por software de una central telefónica denominada PBX por sus siglas en ingles (private branch exchange. Como cualquier otra central telefónica (PBX), esta permite que un conjunto de extensiones (teléfonos) conectados a ella pueda realizar llamada entre ella y así mismo poder conectarse a otos servicios telefónicos incluyendo las líneas

de servicio publico de telefonía (PSTN. El software del IP-PBX ha sido diseñado para trabajar bajo el sistema operativo (SO) Linux CentOS 3.5.

El software del IP-PBX incluye varias características que antes solo estaban disponibles en los costosos sistemas propietarios de PBX — correo de voz, llamadas de conferencias, respuesta interactiva de voz o menú de voz (IVR), y distribución automática de llamadas. Los usuarios pueden agregar nuevas funcionalidades al escribir planes de marcados dirigidos a solucionar y priorizar enrutamientos óptimos de llamadas.

Para poder agregar teléfonos ordinarios a un servidor de Linux que tenga el software del IP-PBX, o conectar líneas tróncales de telefonía tradicional (PSTN), dicho servidor debe estar integrado de un hardware especial. (Un módem ordinario no seria útil.) Digium y un determinado número de firmas se encarga de diseñar y comercializar tarjetas PCI para poder agregar teléfonos, líneas telefónicas, líneas T1 y E1, y otros servicios análogos y digitales para el servidor.

Hoy en día el IP-PBX soporta una amplia gama de protocolos de voz sobre IP (VoIP), incluyendo dentro de ellos el SIP, H.323, IAX, etc. El IP-PBX puede ínter operar con la mayoría de teléfonos SIP, actuando tanto como un registrar así como también como una puerta de enlace (gateway) entre los

teléfonos IP y el PSTN. Los desarrolladores del IP-PBX han diseñado un protocolo moderno denominado IAX, para la troncalización eficiente de llamadas ente los IP-PBXs.

Al soportar una mezcla del servicio de telefonía tanto tradicional como de VoIP, el IP-PBX permite que las personas que lo utilizan puedan utilizar los nuevos sistemas de telefonía de forma más eficiente, o gradualmente estos puedan emigrar a los nuevos sistemas de tecnología que existen hoy en día. Algunos lugares están utilizando los servidores IP-PBX para reemplazar sus antiguas unidades de PBX; otos las utilizan para proveer funciones adicionales (tales como el correo de voz o menú de teléfono) o para bajar costos de llamadas de larga distancia sobre el Internet.

Características:

Las soluciones de telefonía basadas en software ofrecen un amplio y flexible conjunto de características. El IP-PBX ofrece tanto la clásica funcionalidad de un PBX así como también características avanzadas, permitiendo operar con los sistemas telefónicos tradicionales y los sistemas sobre IP de VoIP.

Servicios de telefonía:

- Sistema de correo de voz (Voicemail)
    - Protegido por clave.
    - Mensajes por separado tanto de no disponibilidad como de alejado.
    - Mensajes predefinidos o personalizados
    - Múltiple carpetas de correo
    - Interfase vía web para revisar su correo de voz
    - Notificación por e-mail de correo de voz
    - Reenvío de correo de voz
    - Indicador visual de mensaje en espera
- Asistente automático
- Respuesta interactiva por voz (IVR)
- Overhead Paging
- Lógica flexible de extensiones
    - Extensiones de líneas múltiples
    - Control de acceso de multi-capas
    - Sistema de acceso directo Interno
- Listado de directorio
- Puenteo de conferencias
    - Cuartos de conferencias ilimitados

    - Control de acceso
- Cola de llamadas
- Sistema de menú ADSI
    - Soporte para características avanzadas de telefonía
    - Sistemas de menú visuales conducidos por el PBX
    - Notificación visual de correo de voz
- Registro de los detalles de llamadas (CDR)
- Agentes de llamadas locales
- Agentes de llamadas remotas
- Puenteo de protocolo
    - Permite la integración de tecnologías
    - Ofrece un conjunto unificado de servicios a los usuarios independientemente del tipo de conexión
    - Permite la interoperabilidad de los sistemas de VoIP

Características de las llamadas

- Música en espera
- Música en transferencia
    - Sistema flexible basado en mp3
    - Control de volumen
    - Toque aleatorio

    - Toque lineal
- Llamada en espera
- Identificador de llamadas
- Bloqueo de identificación de llamadas
- Identificador de llamadas en llamadas en espera
- Reenvío de llamadas cuando está ocupado
- Reenvío de llamadas al no contestar
- Reenvío de llamadas variable
- Transferencia de llamadas
- Paqueo de llamadas
- Recuperación de llamadas
- Contestación remota de llamadas
- No molestar

Escalabilidad:

- TDMoE
    - Permite la conexión directa del IP-PBX
    - Ofrece latencia cero
    - Utiliza hardware de commodity ethernet

Voz sobre IP

- Permite la integración de instalaciones físicamente separadas
- Utiliza conexiones de datos comúnmente desplegados
- Permite un plan de marcado unificado a través de múltiples oficinas

Interoperabilidad de voz sobre IP:

EL IP-PBX provee un puenteo transparente entre los protocolos de voz sobre IP y los equipos de telefonía tradicional. Además se pueden transferir llamadas de un sistema a otro, vía el protocolo de Inter Asterisk Exchange.

- Inter-Asterisk Exchange (IAX)
- H.323
- Session Initiation Protocol (SIP)
- Media Gateway Control Protocol (MGCP)

Interoperabilidad con telefonía tradicional

- Tipos de señalización Robbed Bit
    - FXS y FXO

    - Loop start
    - Ground start
    - Kew start
    - E&M
    - E&M Wink
    - Característica Grupo D

- Protocolos PRI
    - 4ESS
    - Lucent 5E
    - DMS100
    - National ISDN2
    - EuroISDN
    - BRI (ISDN4Linux)

Soporte de codec

- GSM
- G.729 (disponible a través de la compra de licencias comerciales)
- G.723.1 (de paso)
- Lineal

- Mu-Law
- A-Law
- ADPCM
- G.726
- ILBC
- LPC-10
- MP3 (solamente decodificación)

CAPÍTULO 2.

2. SITUACIÓN ACTUAL

2.1. Proceso actual

Para el envío y retiro de dinero en muchas de las casas de cambio se usa el siguiente proceso:

En el caso de envió de dinero:

1) Cliente quien envía dinero debe acercarse a la agencia más cercana de las empresas que realizan estas transferencias.
2) Cliente debe presentarse con sus nombres y apellidos, el monto a enviar, nombre completo de la persona a quien envía, dirección de la persona a quien envía.
3) La empresa que realiza la transferencia cobra una comisión por el envió del dinero y se le asigna un código de giro, para que a través de éste código pueda realizar el retiro del dinero la otra persona.
4) Todo éste proceso es ingresado manualmente en un sistema que se comunica con una base de datos central que servirá para poder consultar el destinatario del giro.

En el caso de retiro de dinero:

1) Cliente recibe confirmación por teléfono de la empresa que realiza el giro o de la persona que envía dinero.
2) Cliente debe acercarse a la sucursal más cercana para solicitar el retiro del dinero, dando sus datos y el código de transacción.
3) La agencia local dispone de dinero en efectivo como caja para poder realizar la entrega del dinero.
4) Una vez que son verificados los datos la agencia procede a la entrega del dinero.

2.2. Diseño de la red actual.

Actualmente estos sistemas se manejan con una red LAN, la cual cuenta con una conexión a Internet, y a través de esta usando la tecnología de Túneles IP se conectan a la base de datos central usando una aplicación web y protocolos TCP/IP que sirve para ingresar los envíos y retiros de dinero a la base de datos central.

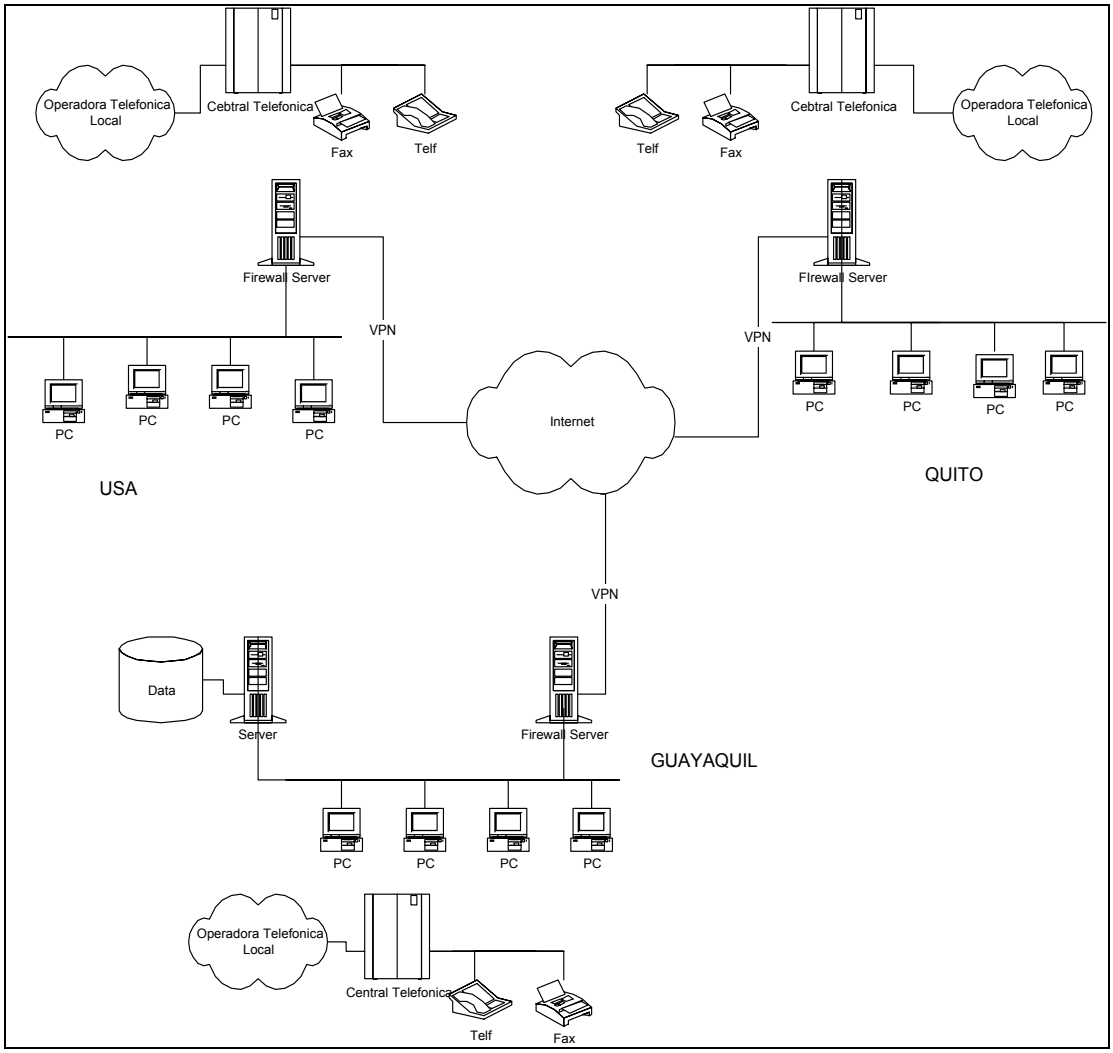


Figura 16. *Diseño de red actual.*

Como notamos en la grafica la red de telefonía solo sirve de paso entre el operador del sistema y la base de datos. No existe conexión alguna entre la red de voz y la red de datos.

Se dispone de una aplicación cliente servidor vía web mediante la cual se tienen las siguientes opciones para la administración del sistema:

- Registro de Usuario.
- Administración de usuario.
- Administración de Servicios y Tarifas.
- Registro de Envío y Retiro de Dinero.

Se dispone de central telefónica digital y líneas telefónicas locales, las cuales básicamente sirven para comunicación entre sucursales, o también para recibir llamadas de clientes y verificar a través del código de giro si ha recibido dinero.

Todas las sucursales se comunican entre si a través de enlaces privados locales o a través de VPN usando el Internet.

La red de datos de cada sucursal se encuentra protegida con un sistema Firewall que impiden el acceso de cualquier intruso.

En estos sistemas se tiene una base de datos central en la cual se registran todos los giros realizados.

No disponen de configuración en su red de datos la optimización del servicio de voz.

2.3. Tecnología actual.

Las redes LAN usan topología Ethernet 10/100 base T. La conexión a Internet que disponen es a través de enlaces wireless, conexiones ADSL, conexiones DialUp, con las cuales se configuran las VPNs para la conexión a la base de datos. Localmente disponen de una aplicación cliente/servidor con conexiones a la base de datos central. Como nos damos cuenta en el diseño de la red actual vemos un completo divorcio entre la red telefónica y la red de datos.

2.4. Necesidades de los clientes.

Los clientes necesitan disponer de mayor agilidad y seguridad en el envío y retiro de dinero.

También requieren de un servicio personalizado, en el cual puedan llevar un registro de todas sus transacciones realizadas.

CAPÍTULO 3.

3. PROYECTO

3.1. Descripción de proyecto.

El proyecto se basa en la implementación de transacciones de envío y retiro de dinero usando un sistema IVR, para lo cual el sistema demostrativo debe cumplir los siguientes requisitos:

Para el proyecto se ha considerado que se lo implementa en una institución ficticia llamada ABC que tiene una matriz en Guayaquil y 2 sucursales tanto en Quito como en Miami. Además la institución ABC posee dominio propio y tiene publicado un sitio web en el internet.

Cualquier usuario antes de poder realizar una transacción debe primeramente registrarse en el sistema, acercándose a la institución ABC más cercana y realizar el registro, en el cual le darán una clave única para poder realizar cualquier tipo de transacción, o podrá realizar el registro en el internet utilizando un browser e ingresando la dirección web de la institución ABC y escoger la opción registrarse.

Cuando el usuario se registró en el sistema, éste automáticamente simula una conexión con bases de datos externas de las instituciones financieras locales registradas en el sistema, para así poder buscar las

cuentas o tarjetas de crédito del usuario y registrarlas permanentemente en el sistema para poder realizar transacciones sobre estas.

El usuario podrá llamar al servicio telefónico de transacciones de la institución ABC y con la clave de acceso poder realizar cualquier transacción de envío o retiro de dinero utilizando el sistema IVR.

Cabe recalcar que también la institución ABC en el sitio web permitirá la realización de transacciones de envío y retiro de dinero.

En la parte administrativa del sistema vía web deberá realizar lo siguiente:

- Conexiones con nuevos bancos que deseen registrase en el sistema, esto se lo realizará vía web.
- Visualizar y buscar los usuarios registrados en el sistema.
- Realizar el registro de nuevos usuarios.
- Reporte de transacciones por rangos de fecha.

Como parte del proyecto también se implementaran centrales telefónicas IPs con funciones básicas de centrales telefónicas modernas en cada una de las sucursales de la institución ABC.

Se deberá realizar la integración de la central telefónica convencional con la nueva central telefónica IP.

3.2 Esquema de la red a implementar.

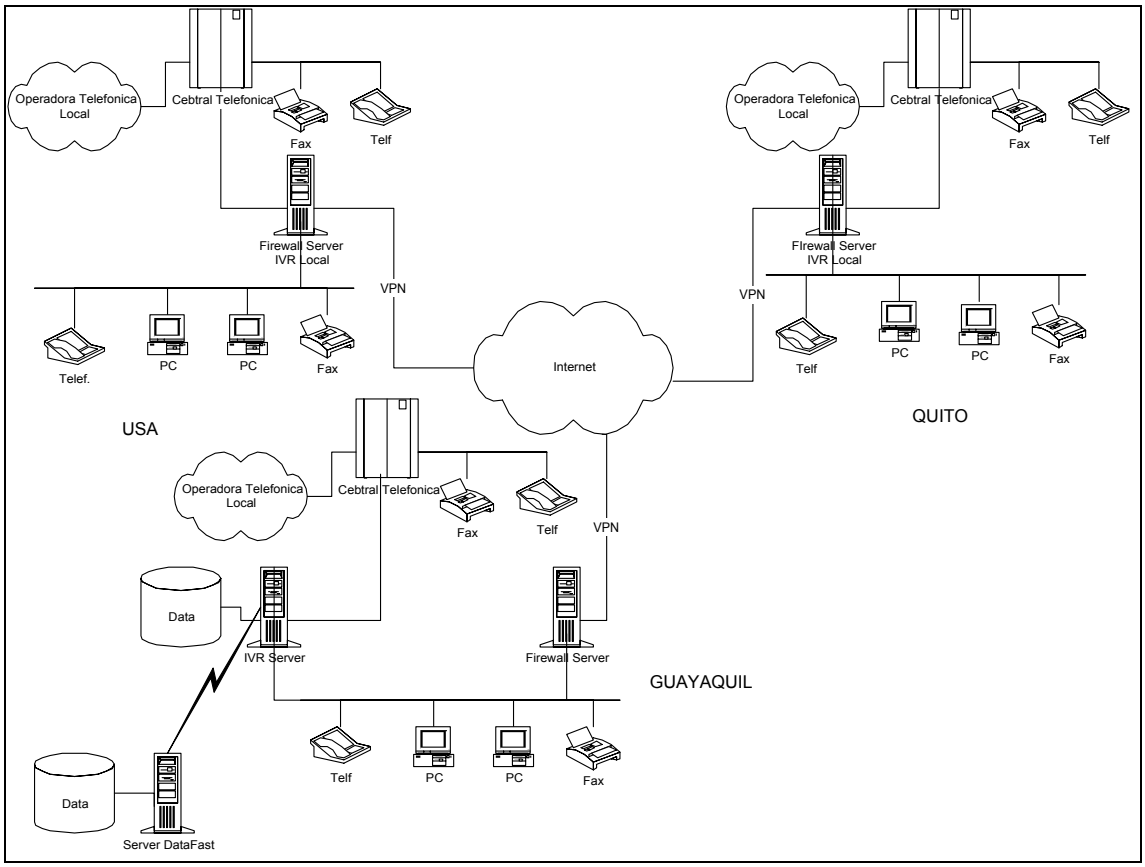


Figura 17. *Diagrama de red a implementar para la institución ABC*

El objetivo del proyecto es integrar la voz y datos en una sola red y también poder interactuar entre paquetes voz con datos.

Se plantea el hecho de poder enviar dinero desde USA a personas tanto de Guayaquil como de Quito y viceversa, aunque en el caso contrario las transacciones serán de muchísima menor escala, dado que ingresa más dinero del exterior a nuestro País, respecto a lo que se envía.

Tanto en Guayaquil, Quito y USA existirán Servidores IVR con líneas telefónicas locales que atenderán los requerimientos de transacciones de los usuarios.

Estos servidores IVR también dispondrán de características de centrales telefónicas con funciones de transfer, followme, multiring, voice mail, conference room que servirá para la comunicación interna entre empleados de las sucursales.

Se dispondrá de un Servidor Web el cual contendrá una aplicación que permita al usuario realizar las siguientes opciones:

- Registrarse en el sistema.
- Visualizar vía web el detalle de las transacciones realizadas.
- Realizar Transacciones vía Web.

Este Servidor Web también contendrá una aplicación para la administración del sistema, el cual contendrá las siguientes opciones:

- Administración de Usuarios.
- Administración de Servicios y Tarifas.
- Administración de Bancos.

Todas estas opciones interactúan con una base de datos MySQL, en la cual se queda registrado todo movimiento.

Para la comunicación entre sucursales se utilizará la misma infraestructura levantada entre ellas, lo único que se realizarán en estos enlaces es verificar su óptimo funcionamiento y también la configuración de estos enlaces para que funcione la telefonía perfectamente.

Existirá una simulación de conexión a las bases de datos de las instituciones financieras registradas en el sistema.

Las centrales telefónicas digitales se las integrará con el IPBX, permitiendo así una comunicación entre usuarios de la central telefónica convencional y usuarios de la central telefónica IP y viceversa.

3.3. Hardware, software y tecnologías a utilizar.

La implementación del proyecto consiste en seis partes, en cada una de ellas se detalla el equipamiento y tecnologías a utilizar:

1) Optimización de red privada de datos del cliente.
2) Implementación del servicio de VoIP en la red del cliente.
3) Implementación del IPBX.
4) Diseño e implementación de la Base de Datos.
5) Implementación del registro, transacciones del usuario y administración del sistema.
6) Implementación de transacciones usando IVR.

1) Optimización de la red privada de datos del cliente.

Para nuestro caso de estudio se utiliza servidores Linux en los cuales estarán levantadas las VPNs y se implementará la solución de optimización de la red privada de datos usando el servicio IPTABLE propio del sistema operativo Linux. Tanto la red de voz como de datos de cada sucursal estará detrás de este servidor el cual a parte de la función de optimizar el canal para la voz, tambien servirá como seguridad para la red privada voz-datos de cada sucursal.

Si el cliente dispone de un enlace frame relay o compartido, se le sugerirá cambiar éste enlace por un canal puro o clear channel, y la capacidad de éste, dependerá del número de canales de voz simultáneas que requiera el cliente. Se considerará 16 Kbps por cada canal de voz.

2) Implementación del servicio de VoIP en la red del cliente.

Para la implementación del servicio de VoIP en la red del cliente se utilizaran los siguientes equipos:

- Equipos Terminales Sipura modelo SPA841.
- Tarjetas FXO.
- Licencias para soportar protocolo G.729 bajo Linux.
- Gramstream modelo GXP-2000 como teléfono de central.

3) Implementación del IPBX.

Para la implementación del IPBX se requerirá.

1 PC con las siguientes características mínimas:

Procesador Pentium 3, 1Ghz.

Disco Duro 20Gbyte.

Memoria RAM 256Mbyte.

2 Tarjetas de Red 10/100 BASE T.

Sistema Operativo: Linux Red Hat Enterprise 4.0.

Asterisk para la implementación de la central telefónica.

Los sistemas IPBX para la institución ABC tendrán las siguientes características en cada una de las sucursales:

Localidad	**Líneas externas**	**Número extensiones**
Guayaquil	8	24
Quito	8	24
Miami	8	24

Tabla 6. *Distribución de líneas externas y extensiones en sucursales.*

4) Diseño e implementación de Base de Datos.

Se utilizará MySQL 5.0 para Linux centOS 4.2, para la base de datos del sistema.

5) Implementación del registro, transacciones del usuario y administración del sistema.

En esta parte se desarrolla la parte del sistema que se interactúa vía Web. Esta parte se la programa usando las siguientes herramientas de desarrollo:

Servidor Web Tomcat.

Java JDK 1.5.

Microsoft Front Page.

6) Implementación de transacciones usando IVR.

Para el desarrollo de esta programación se utiliza Asterisk add-on que es la libreria que permite realizar conexión y operaciones a la base de datos MySQL usando IVR.

3.4 Optimización de RED de datos para el servicio de VoIP.

Uno de los pasos primordiales en la implementación del servicio de VoIP en la red será la priorización o segmentación de paquetes de voz en la

red, para lo cual se utiliza comandos IPTABLE bajo el sistema operativo Linux.

Se está partiendo del hecho que el cliente posee un canal privado clear channel y libre de errores de comunicación entre las sucursales.

Consideremos dos puntos como se muestra en Figura 18 para poder implementar segmentación de ancho de banda.

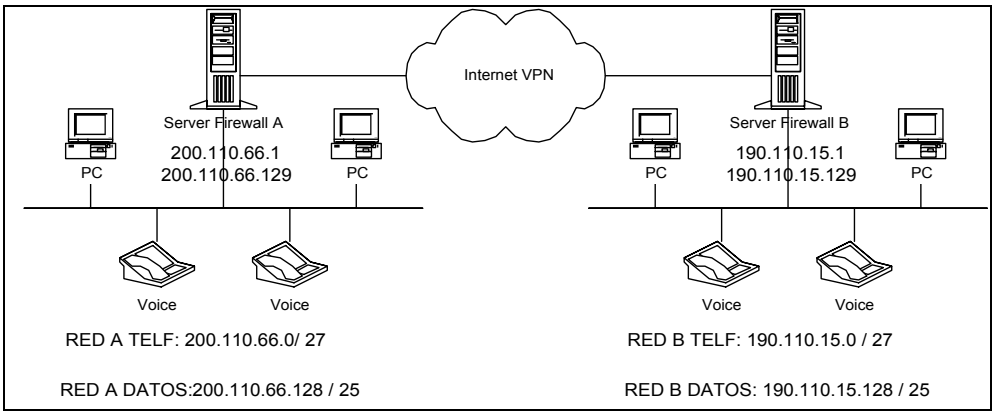


Figura 18. *Red de ejemplo para segmentario de ancho de banda*

La segmentación se la realiza en cada uno de los servidores Linux, en el cual se configura cuanto ancho de banda se asignará a la red de datos y cuanto ancho de banda a la red de telefonía. En el servidor Linux como requisito debe estar levantado el servicio IPTABLE para poder implementar la segmentación de ancho de banda.

Para el caso de la Institución ABC tendrá la siguiente distribución de ancho de banda simétricos (igual ancho de banda tanto de subida como de bajada):

Enlaces	**Ancho de banda Datos (Kbps)**	**Ancho de banda Voz (Kbps)**	**Ancho de banda total (Kbps)**
Guayaquil – Miami	128	256	384
Guayaquil – Quito	128	256	384
Quito – Miami	128	256	384

Tabla 7: *Distribución de ancho de banda entre sucursales.*

A continuación se detalla el código a ingresar en cada uno de los servidores para realizar la segmentación de ancho de banda:

SERVIDOR FIREWALL A:

```
#!/bin/sh
IPTABLES=/sbin/iptables
TC=/sbin/tc
CBQ=/sbin/cbq.init
OK="\033[1;0m [  \033[00;32mOK  \033[1;0m]\033[0m"
echo -e "CBQ: Cargando Reglas de Control de Ancho de Banda:        $OK"
$CBQ stop
# ===>> Control Ingreso de Paquetes de la Red <<=== #
#=== Marcamos los paquetes de entrada de telefonía con el número 1 ===#
```

```
$IPTABLES -t mangle -A PREROUTING -j MARK -p tcp -s 200.110.66.1 --set-mark 1
$IPTABLES -t mangle -A INPUT      -j MARK -p tcp -s 200.110.66.1 --set-mark 1
$IPTABLES -t mangle -A OUTPUT     -j MARK -p tcp -s 200.110.66.1 --set-mark 1
$IPTABLES -t mangle -A PREROUTING -j MARK -p udp -s 200.110.66.1 --set-mark 1
$IPTABLES -t mangle -A INPUT      -j MARK -p udp -s 200.110.66.1 --set-mark 1
$IPTABLES -t mangle -A OUTPUT     -j MARK -p udp -s 200.110.66.1 --set-mark 1
#=== Marcamos los paquetes de entrada de Datos con el número 3 ===#
$IPTABLES -t mangle -A PREROUTING -j MARK -p tcp -s 200.110.66.129 --set-mark 3
$IPTABLES -t mangle -A INPUT      -j MARK -p tcp -s 200.110.66.129 --set-mark 3
$IPTABLES -t mangle -A OUTPUT     -j MARK -p tcp -s 200.110.66.129 --set-mark 3
$IPTABLES -t mangle -A PREROUTING -j MARK -p udp -s 200.110.66.129 --set-mark 3
$IPTABLES -t mangle -A INPUT      -j MARK -p udp -s 200.110.66.129 --set-mark 3
$IPTABLES -t mangle -A OUTPUT     -j MARK -p udp -s 200.110.66.129 --set-mark 3
# ===>> Control Egreso de Paquetes de la Red <<=== #
#=== Marcamos los paquetes de entrada de telefonía con el número 2 ===#
$IPTABLES -t mangle -A PREROUTING -j MARK -p tcp -s 200.110.66.4 --set-mark 2
$IPTABLES -t mangle -A INPUT      -j MARK -p tcp -s 200.110.66.4 --set-mark 2
$IPTABLES -t mangle -A OUTPUT     -j MARK -p tcp -s 200.110.66.4 --set-mark 2
$IPTABLES -t mangle -A PREROUTING -j MARK -p udp -s 200.110.66.4 --set-mark 2
$IPTABLES -t mangle -A INPUT      -j MARK -p udp -s 200.110.66.4 --set-mark 2
$IPTABLES -t mangle -A OUTPUT     -j MARK -p udp -s 200.110.66.4 --set-mark 2
#=== Marcamos los paquetes de entrada de telefonía con el número 4 ===#

$IPTABLES -t mangle -A PREROUTING -j MARK -p tcp -s 200.110.66.129 --set-mark 4
$IPTABLES -t mangle -A INPUT      -j MARK -p tcp -s 200.110.66.1 --set-mark 4
$IPTABLES -t mangle -A OUTPUT     -j MARK -p tcp -s 200.110.66.1 --set-mark 4
$IPTABLES -t mangle -A PREROUTING -j MARK -p udp -s 200.110.66.1 --set-mark 4
```

```
$IPTABLES -t mangle -A INPUT      -j MARK -p udp -s 200.110.66.1 --set-mark 4
$IPTABLES -t mangle -A OUTPUT     -j MARK -p udp -s 200.110.66.1 --set-mark 4

# ===>> Segmentación de Ancho de Banda a 256Kbit voz y 128Kbit datos <<=== #

$TC qdisc add dev eth0 root handle 1: cbq bandwidth 10Mbit avpkt 1000 cell 8
$TC qdisc add dev eth1 root handle 1: cbq bandwidth 10Mbit avpkt 1000 cell 8

$TC class add dev eth0 parent 1:0 classid 1:1 est 1sec 2sec cbq bandwidth 10Mbit rate
96Kbit allot 1514 cell 8 weight 1 prio 1 maxburst 20 avpkt 1000 bounded
$TC class add dev eth1 parent 1:0 classid 1:2 est 1sec 2sec cbq bandwidth 10Mbit rate
96Kbit allot 1514 cell 8 weight 1 prio 1 maxburst 20 avpkt 1000 bounded
$TC class add dev eth0 parent 1:0 classid 1:3 est 1sec 2sec cbq bandwidth 10Mbit rate
64Kbit allot 1514 cell 8 weight 1 prio 1 maxburst 20 avpkt 1000 bounded
$TC class add dev eth1 parent 1:0 classid 1:4 est 1sec 2sec cbq bandwidth 10Mbit rate
64Kbit allot 1514 cell 8 weight 1 prio 1 maxburst 20 avpkt 1000 bounded

$TC filter add dev eth0 protocol ip handle 1 fw classid 1:1
$TC filter add dev eth1 protocol ip handle 2 fw classid 1:2
$TC filter add dev eth0 protocol ip handle 3 fw classid 1:3
$TC filter add dev eth1 protocol ip handle 4 fw classid 1:4
```

SERVIDOR FIREWALL B:

```
#!/bin/sh
```

```
IPTABLES=/sbin/iptables
TC=/sbin/tc
CBQ=/sbin/cbq.init
OK="\033[1;0m [  \033[00;32mOK  \033[1;0m]\033[0m"
echo -e "CBQ: Cargando Reglas de Control de Ancho de Banda:        $OK"
$CBQ stop

# ===>> Control Ingreso de Paquetes de la Red <<=== #

$IPTABLES -t mangle -A PREROUTING -j MARK -p tcp -s 190.110.15.1 --set-mark 1
$IPTABLES -t mangle -A INPUT      -j MARK -p tcp -s 190.110.15.1  --set-mark 1
$IPTABLES -t mangle -A OUTPUT     -j MARK -p tcp -s 190.110.15.1  --set-mark 1
$IPTABLES -t mangle -A PREROUTING -j MARK -p udp -s 190.110.15.1 --set-mark 1
$IPTABLES -t mangle -A INPUT      -j MARK -p udp -s 190.110.15.1  --set-mark 1
$IPTABLES -t mangle -A OUTPUT     -j MARK -p udp -s 190.110.15.1 --set-mark 1

$IPTABLES -t mangle -A PREROUTING -j MARK -p tcp -s 190.110.15.129 --set-mark 3
$IPTABLES -t mangle -A INPUT      -j MARK -p tcp -s 190.110.15.129 --set-mark 3
$IPTABLES -t mangle -A OUTPUT     -j MARK -p tcp -s 190.110.15.129 --set-mark 3
$IPTABLES -t mangle -A PREROUTING -j MARK -p udp -s 190.110.15.129 --set-mark 3
$IPTABLES -t mangle -A INPUT      -j MARK -p udp -s 190.110.15.129 --set-mark 3
$IPTABLES -t mangle -A OUTPUT     -j MARK -p udp -s 190.110.15.129 --set-mark 3

# ===>> Control Egreso de Paquetes de la Red <<=== #

$IPTABLES -t mangle -A PREROUTING -j MARK -p tcp -s 190.110.15.1 --set-mark 2
$IPTABLES -t mangle -A INPUT      -j MARK -p tcp -s 190.110.15.1  --set-mark 2
```

```
$IPTABLES -t mangle -A OUTPUT     -j MARK -p tcp -s 190.110.15.1  --set-mark 2
$IPTABLES -t mangle -A PREROUTING -j MARK -p udp -s 190.110.15.1 --set-mark 2
$IPTABLES -t mangle -A INPUT      -j MARK -p udp -s 190.110.15.1  --set-mark 2
$IPTABLES -t mangle -A OUTPUT     -j MARK -p udp -s 190.110.15.1 --set-mark 2

$IPTABLES -t mangle -A PREROUTING -j MARK -p tcp -s 190.110.15.129 --set-mark 4
$IPTABLES -t mangle -A INPUT      -j MARK -p tcp -s 190.110.15.129 --set-mark 4
$IPTABLES -t mangle -A OUTPUT     -j MARK -p tcp -s 190.110.15.129 --set-mark 4
$IPTABLES -t mangle -A PREROUTING -j MARK -p udp -s 190.110.15.129 --set-mark 4
$IPTABLES -t mangle -A INPUT      -j MARK -p udp -s 190.110.15.129 --set-mark 4
$IPTABLES -t mangle -A OUTPUT     -j MARK -p udp -s 190.110.15.129 --set-mark 4

# ===>> Segmentación de Ancho de Banda a 96Kbit voz y 64Kbit datos <<=== #

$TC qdisc add dev eth0 root handle 1: cbq bandwidth 10Mbit avpkt 1000 cell 8
$TC qdisc add dev eth1 root handle 1: cbq bandwidth 10Mbit avpkt 1000 cell 8

$TC class add dev eth0 parent 1:0 classid 1:1 est 1sec 2sec cbq bandwidth 10Mbit rate
96Kbit allot 1514 cell 8 weight 1 prio 1 maxburst 20 avpkt 1000 bounded
$TC class add dev eth1 parent 1:0 classid 1:2 est 1sec 2sec cbq bandwidth 10Mbit rate
96Kbit allot 1514 cell 8 weight 1 prio 1 maxburst 20 avpkt 1000 bounded
$TC class add dev eth0 parent 1:0 classid 1:3 est 1sec 2sec cbq bandwidth 10Mbit rate
64Kbit allot 1514 cell 8 weight 1 prio 1 maxburst 20 avpkt 1000 bounded
$TC class add dev eth1 parent 1:0 classid 1:4 est 1sec 2sec cbq bandwidth 10Mbit rate
64Kbit allot 1514 cell 8 weight 1 prio 1 maxburst 20 avpkt 1000 bounded

$TC filter add dev eth0 protocol ip handle 1 fw classid 1:1
```

```
$TC filter add dev eth1 protocol ip handle 2 fw classid 1:2
$TC filter add dev eth0 protocol ip handle 3 fw classid 1:3
$TC filter add dev eth1 protocol ip handle 4 fw classid 1:4
```

3.5. Implementación de IP-PBX.

3.5.1. Instalación.

Requerimientos de hardware.

Entre más rápido sea el sistema que se utiliza para correr el software del IP-PBX permitirá manejar mayor número de llamadas simultáneas, por ejemplo un Pentium III de 500MHz con 128 Megas de RAM puede fácilmente cumplir las necesidades de una mediana empresa.

Instalando Asterisk@Home.

Para la implementación del IPBX, se lo hará utilizando una aplicación para programación de centrales telefónicas IPs llamado Asterisk@Home disponible en el internet.

Bajar Asterisk@Home desde http://asteriskathome.sourceforge.net.

Una vez bajado en el disco se ejecutan las siguientes líneas:

```
mkdir /var/aah_load
cp asteriskathome-2.1.tar.gz /var/aah_load
cd /var/aah_load
tar xvfz asteriskathome-2.1.tar.gz
./install.sh
```

Cambiando de contraseñas al servidor IP-PBX.

Mientras la conexión de red está desconectada o por lo menos conectada a un hub o switch sin nada más conectado a éste, podemos cambiar las contraseñas sin el riesgo de intentos de hackeados. Es muy fácil ingresar y controlar cualquier tipo de servidor que no se han cambiado sus entradas por defecto ni sus claves. El servidor IP-PBX no es diferente de esto. Es recomendable que se escriban estas claves y se las guarde en un lugar muy seguro.

Cambiando la contraseña por defecto del AMP (Asterisk Management Portal).

Asterisk consta con una aplicación de administración llamado AMP (Asterisk Management Portal), el cual nos permite facilitar la configuración y administración de la central telefonica. Para accesar al AMP se ingresa la siguiente linea desde un navegador de web:

HTTP://IP-PBXipaddress

El ingreso y la contraseña por defecto para un AMP recientemente instalado es:

Username: maint

Password: password

Para cambiar la contraseña por defecto, en la línea de comando del CentOS se digita el siguiente comando:

Passwd maint

Deberá aparecer lo siguiente:

Set password for AMP web GUI and maint GUI

User: maint

New password:

Re-type new password:

Updating password for user maint

Pedirá la nueva contraseña. Luego pedirá que confirme la contraseña. También se puede cambiar la contraseña de wwwadmin utilizando.

```
passwd-amp
```

Deberá aparecer lo siguiente:

```
--------------------------------------------
Set password for AMP web GUI and maint GUI
User: wwwadmin
--------------------------------------------
New password:
Re-type new password:
Updating password for user wwwadmin
```

Cambiando la contraseña FOP por defecto.

Asterisk consta de una aplicación que permite la visualización de todas las llamadas que transitan en la central telefónica, y es muy útil para la recepcionista. Esta aplicación es le FOP (Flash Operador Panel) el cual es como un teléfono de central, manipulable vía web.

La contraseña por defecto para un Flash Operador Panel recientemente instalado:

Password: password

Para cambiar la contraseña por defecto se debe ingresar al sistema operativo CentOS utilizando el ingreso root login y la contraseña e ingresar al directorio de FOP se digita lo siguiente:

cd /var/www/html/panel

Usando un editor de texto, se abre el archivo de configuración op_server.cfg y se busca la línea que dice security code=password. Reemplaza “password” con la contraseña requerida.

security_code=lacontraseñaquedesee

Luego se presiona CTRL-X para salir y luego "Y" para guardar los cambios. Ahora se reinicia el servidor.

Cambiando la contraseña por defecto del MeetMe.

Para cambiar la contraseña por defecto del MeetMe se digita lo siguiente en la línea de comando del sistema operativo CentOS.

passwd-meetme

Pedirá ingresar la contraseña nueva dos veces.

Asegurando la tecla ALT-F9 en la consola de monitoreo CLI #9 característica / seguridad para riesgo.

El IP-PBX tiene una característica llamada “riesgo de seguridad escondida”. Se puede presionar “Alt” y “F9” simultáneamente, para acceder a la consola de corrida del IP-PBX sin ingresar al sistema y sin ninguna restricción. Esta pequeña característica se puede considerar un riesgo de la seguridad sino puede garantizar la seguridad física del servidor IP-PBX. En su consola CentOS se ingresa al archivo "safeasterisk" y se escribe lo siguiente:

nano /usr/sbin/safe_asterisk

se cambia

CONSOLE=yes

a

CONSOLE=no

3.5.2. Configuración y administración del IPBX.

ASTERIK posee una interfaz gráfica llamada AMP(Asterisk Management Portal) con la cual se puede fácilmente configurar el IP-PBX. AMP provee un método de interfase gráfica (a través de un navegador de Web) para configurar la configuración textual de los archivos que el IP-PBX necesita para su funcionamiento.

Las siguientes características es lo que AMP puede configurar en el IP-PBX:

- Llamadas entrantes: especifica a donde enviar las llamadas desde el exterior.

- Extensiones: Agrega extensiones y configura propiedades del correo de voz.

- Grupos de timbrado: Grupo de extensiones que deben timbrar simultáneamente.

- Recepcionista digital: Crea menú de voz para dar una bienvenida a las personas que llaman.

- Troncales: Configurar las troncales para que se conecten al mundo exterior.

  Enrutamiento de Salida: Maneja que troncal debe enviar la llamada.

- Música en espera: Hace una subida (Upload) de archivos MP3 para que sea reproducido mientras los usuarios están en espera.

- Grabaciones (del sistema): Graba o sube (upload) mensajes para extensiones especificas.

- Respaldo y restauración: Crea, respalda, y restaura archivos en su sistema.

- Configuraciones generales: Configura la marcación básica, el directorio de la empresa y la configuración del fax.

Para ingresar al AMP y realizar cambios, se ingresa por medio del navegador hacia la dirección IP del servidor IP-PBX.

HTTP://IP-PBXhomeipaddress

El ingreso y contraseña por defecto del AMP es:

Login: maint

Password: password

Configurando una extensión.

1) Se utiliza una PC de la red que tenga un navegador de Web y luego a través de éste se conecta al servidor IP-PBX utilizando HTTP://IP-PBXipaddress.

2) Se hace click en el AMP y luego en configuración.

3) Se realiza un click en extensiones y luego se agregan las extensiones.

4) Luego se utiliza la extensión por defecto 200 y se escribe una contraseña para registración por ejemplo "abc123". A continuación se digita el nombre de la persona utilizando esa extensión.

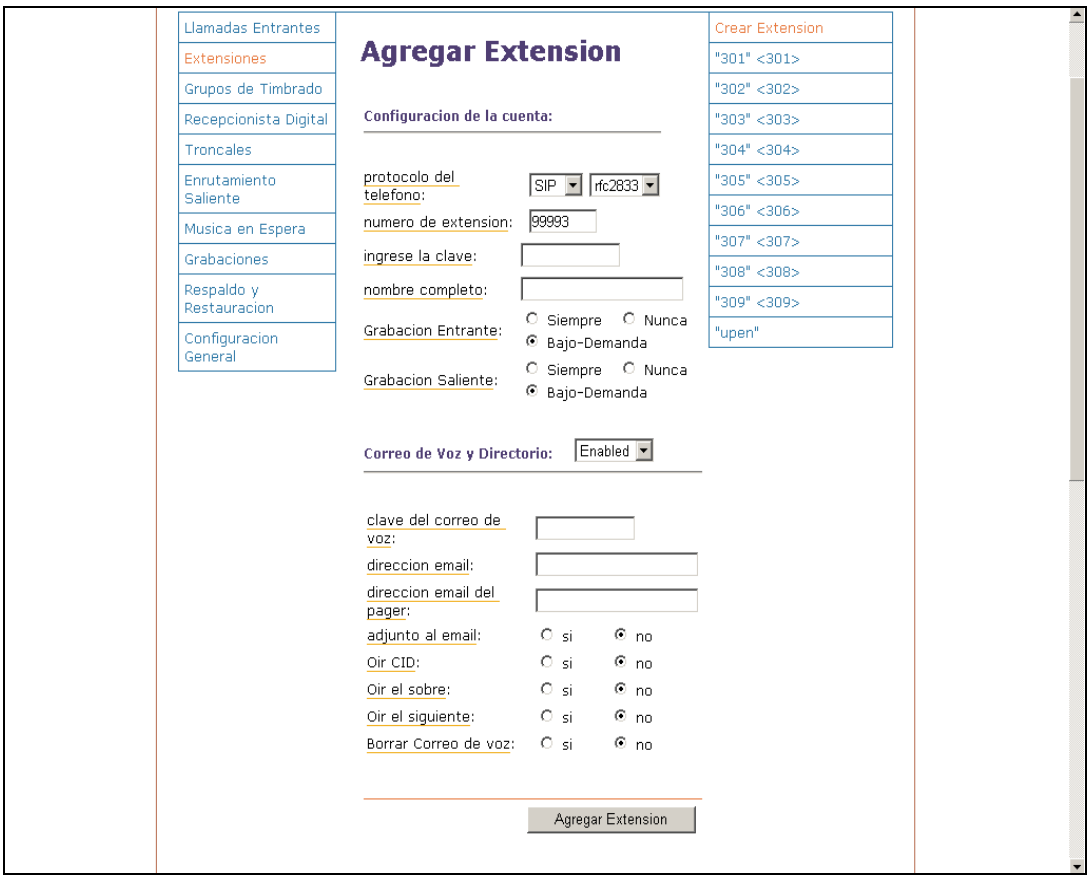


Figura 19. *Agregar una extensión al IPBX.*

Luego se va hasta la sección de correo de voz y se digita la clave para el correo de voz. Luego se digita una clave que se pueda ingresar en el teclado de un teléfono digital convencional como ‘1234’. Luego se ingresa la dirección de correo electrónico a la que se desea que se envíen los correos de voz y luego se realiza un click en agregar extensión. Luego se

reliza un click en la barra roja de **Aplicar** en la parte superior de la pantalla.

Configurando una troncal para llamadas salientes y entrantes.

1) Utilizando AMP se selecciona Configuración y luego Troncales.
2) Se realiza un click en el tipo de troncal que se desea crear.

Figure 20. *Agregar una Troncal.*

Configurando un enrutamiento de salida.

A continuación se necesita una ruta para permitir que las llamadas desde los teléfonos de la oficina puedan salir por una troncal. Si la institución

posee más de una troncal, se debe configurar reglas de marcado para determinar como se escoge una troncal para cada caso. Aquí se configuran todas las llamadas que salgan por una troncal.

Utilizando AMP se selecciona configuración, luego enrutamiento de salida, y se ingresa el nombre para la ruta.

Luego se ingresa lo siguiente en el recuadro para patrón de marcado.

1NXXNXXXXXX

NXXNXXXXXX

NXXXXXX

Esto hará que todas las llamadas usen esta ruta.

A continuación se va a la sección de secuencia de troncal.

En el menú desplegable se selecciona la troncal que se configuró anteriormente, y se presiona agregar.

Eso es todo, luego presionamos un click en enviar cambios y luego hacemos click en la barra roja de **Aplicar** en la parte superior de la pantalla.

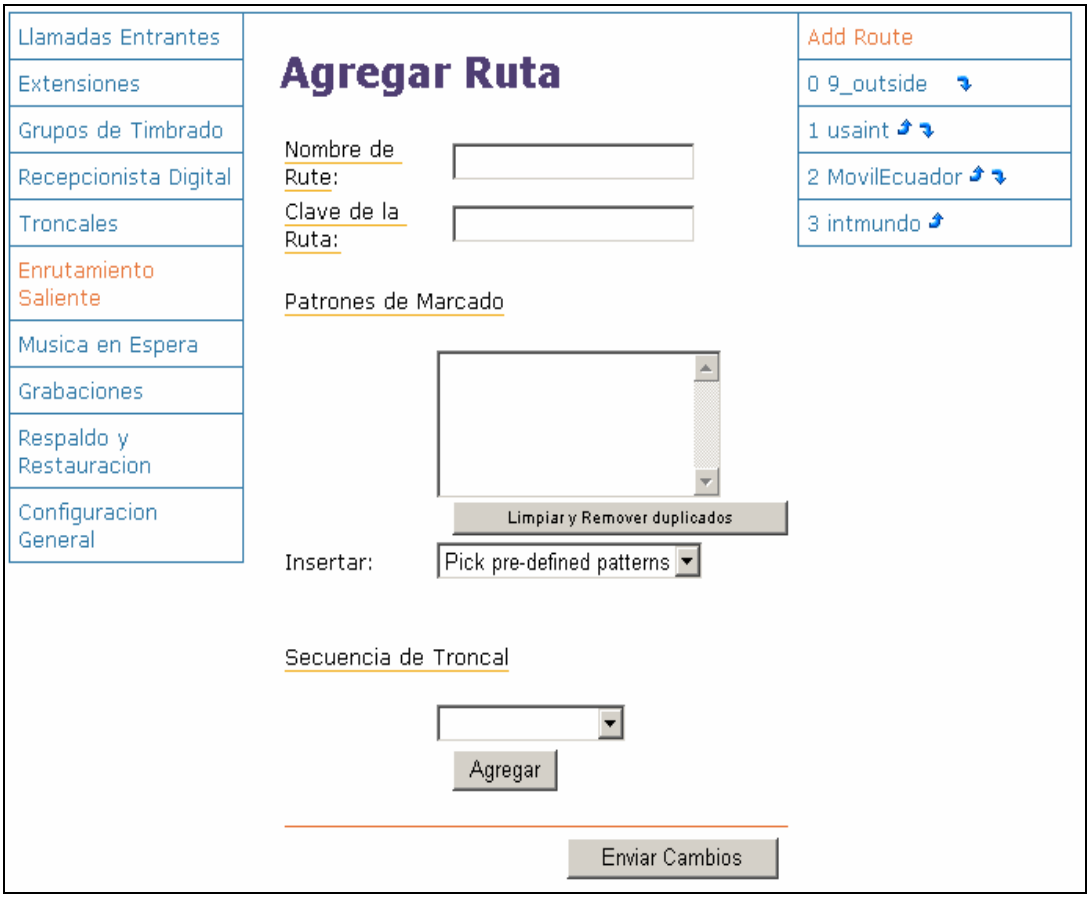


Figura 21. *Agregar una ruta.*

Configurando las llamadas entrantes.

A continuación se necesita una ruta para permitir que las llamas entrantes timbren a algún lado. Utilizando AMP seleccionamos configuración y luego llamadas entrantes. Bajo la leyenda enviar las llamadas entrantes de la PSTN a: se utiliza el menú desplegable y se selecciona la extensión 200 que se creó anteriormente.

Luego se presiona un click en enviar cambios y luego se presiona un click en la barra roja de **Aplicar** en la parte superior de la pantalla.

Para realizar una prueba se puede llamar desde un teléfono celular a la línea PSTN y después escoger la extensión 200 del teléfono SIP y éste debe sonar.

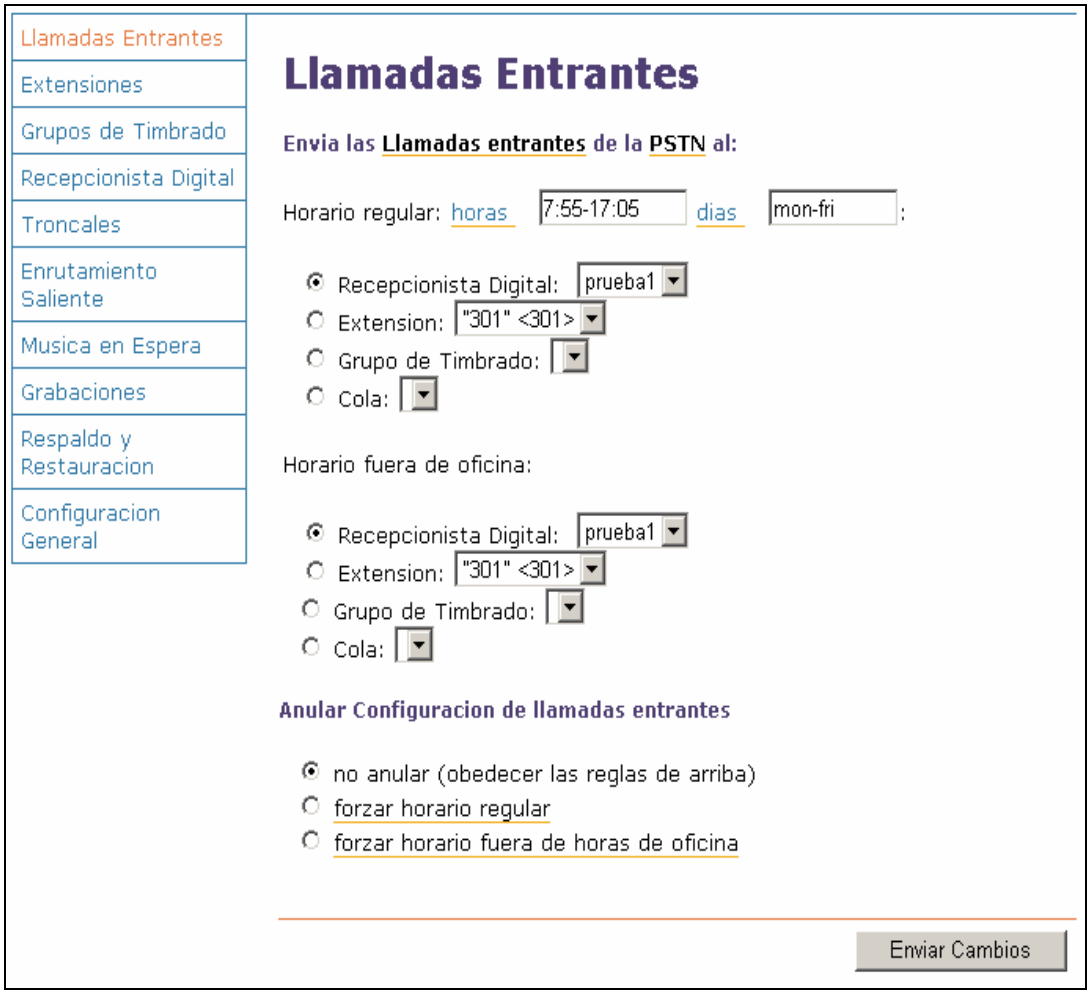


Figura 22. *Llamadas entrantes*

Configurando la recepcionista digital.

Se selecciona la opcion para ir a la página de configuración de la recepcionista digital. Se digita el número de la extensión (ejemplo. El número de la extensión del teléfono de donde va a grabar el mensaje para la recepcionista).

Se asigna un nombre para el menú (ejemplo. Horas_de_atencion) y luego se escribe el texto del mensaje en el cuadro de texto inferior, de tal forma que cuando este grabando solo se tendrá que leer el texto.

Figura 23. *Opción Recepcionista digital.*

Por defecto se puede asignar el símbolo # como una opción de acceso al directorio o marcar la extensión con la que se desea comunicar.

Un buen ejemplo sería:

"Bienvenido a ACME!... Gracias por su llamada... marque él número de la extensión con la que desea comunicarse o marque 1 para hablar con administrativo, 2 para la zona técnica, 3 para la tienda, 4 para hablar con un operador o 5 para dejar un mensaje en nuestro casillero de voz... Marque # para acceder al directorio... Gracias"

Notar lo siguiente: Se puede tener 2 recepcionistas digitales, una para las horas de trabajo y otra para las horas nocturnas, ejemplo:

"Bienvenido a ACME!... Gracias por su llamada... Nuestras oficinas están abiertas desde las 8 am hasta las 5 pm, Marque 1 para hablar con el Soporte 24/24, 5 para dejar un mensaje en nuestro casillero de voz. Gracias"

Se digita *77 y se graba el mensaje.

Se puede escuchar el mensaje recién grabado marcando *99.

Conferencias.

Cada extensión que se crea tiene su propio cuarto de conferencias. Por ejemplo la extensión 200 tiene un cuarto de conferencias en 8200. Se marca 8200 para ingresar al cuarto de conferencias. Una vez iniciada la conferencia ésta puede ser manejada desde la página del WebMeetme.

Solo se debe ingresar la extensión para la conferencia.

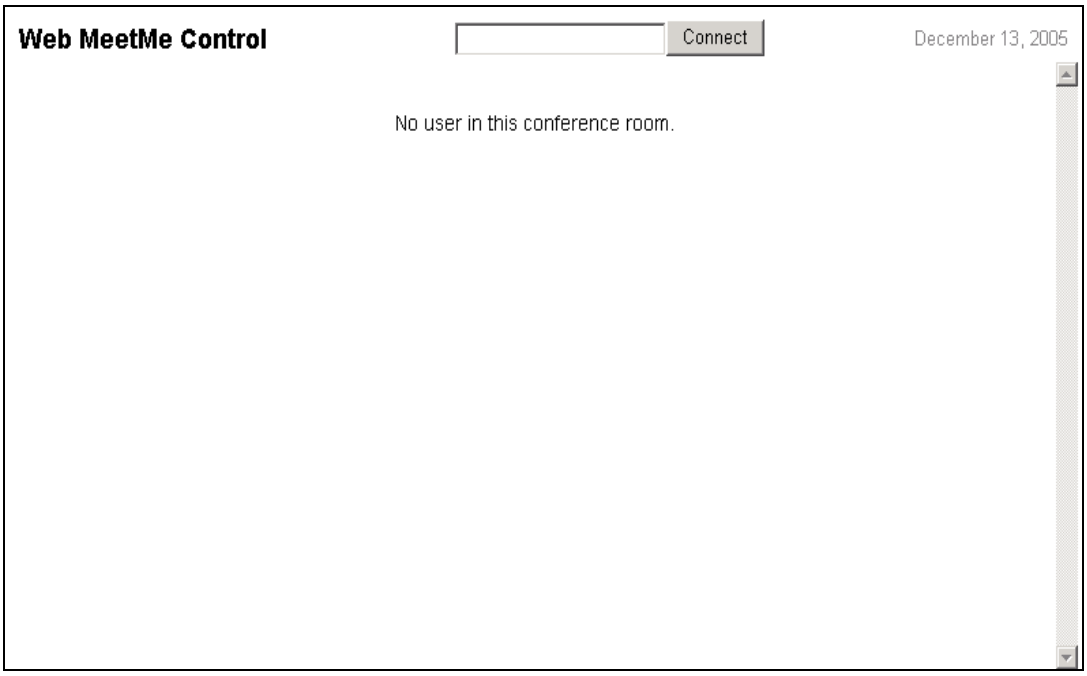


Figura 24. *Opción de configuración para conferencias.*

Los accesos externos para las conferencias están deshabilitados por razones de seguridad. No se desea muchas veces que las personas fuera del sistema tengan acceso a las conferencias.

Acceso externo a los cuartos de conferencias.

Si se desea marcar un acceso externo al cuarto de conferencias se necesita agregar una opción en uno de los auto attendants para éste propósito. Luego se edita el archivo extensions_custom.conf. Dentro de la línea de comando del Linux se ingresa a etc/asterisk (cd /etc/asterisk). Con un editor (por ejemplo vi nombre_del_archivo.conf) se ingresa al archivo extensions_custom.conf y se agrega lo siguiente, luego guardamos los cambios:

---

custom-meetme

include => ext-meetme

exten => s,1,BackGround(enter-conf-call-number)

exten => h,1,Hangup()

Luego se graba un mensaje con opción.

Se debe seleccionar una acción para éste ítem del menú. Damos click en aplicación personal y escribimos lo siguiente:

custom-meetmes,1

luego se presiona un click en continuar.

Esto debe ser todo, se marca hacia el servicio y se selecciona 1 del menú principal y nos preguntará por el número de conferencia a la que desea ingresar, se puede marcar 7777 desde una extensión para simular una llamada entrante.

Flash Operator Panel.

Flash Operator Panel es una interfase basada en web de tiempo real para el IP-PBX. Se puede monitorear todas las extensiones, troncales, y las conferencias que se están realizando. El archivo de op_buttons.conf tiene todas las configuraciones para cada uno de los botones en el Flash Operator Panel.

Desde el navegador web se dirije al FOP. En éste momento se puede hacer click sobre el ícono del candado, se ingresa la contraseña para desbloquearlo. Ahora de esta forma se puede manipular los ítems que están en la pantalla del panel.

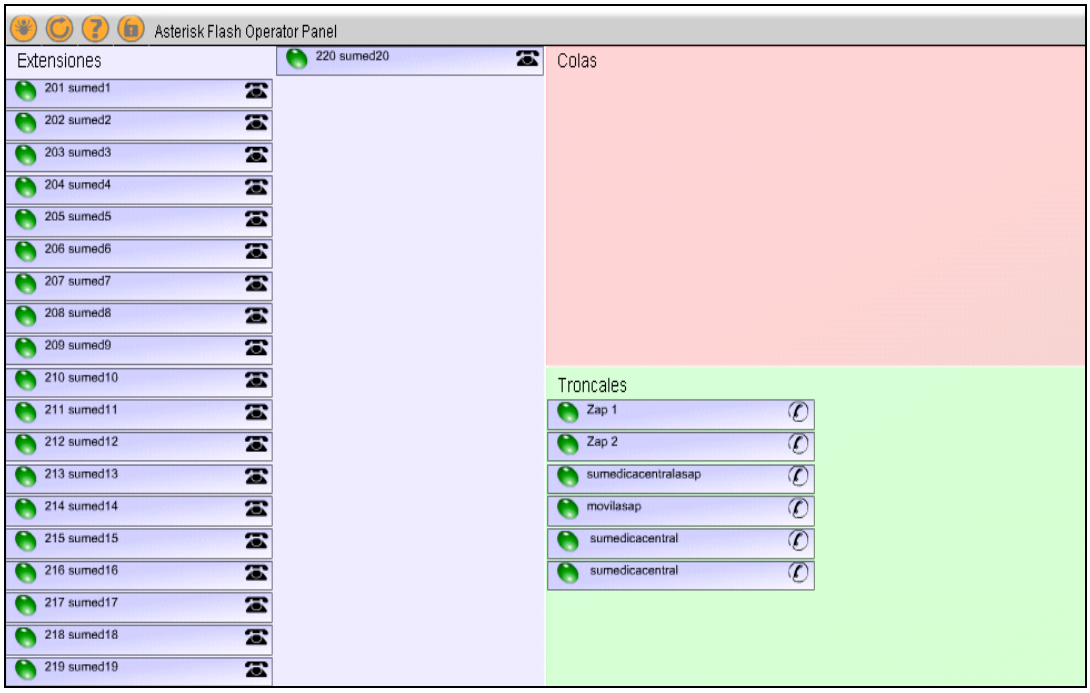


Figura 25. *Opción Flash Operator Panel.*

Cortar una llamada.

Se puede terminar (cortar) una llamada presionado un click en el botón rojo.

Transferir una llamada.

Simplemente se arrastra el icono del teléfono hacia la nueva extensión y se transferirá la llamada.

Iniciar una llamada.

Se puede agarrar y soltar el ícono del teléfono en el flash operator panel hacia otra extensión. Inicialmente comenzará sonando su teléfono, luego de levantarlo comenzará a sonar la otra extensión involucrada. Una vez que el teléfono destino este levantado se iniciará la llamada.

Cómo crear una conferencia.

Antes de crear una conferencia, se necesita tener una llamada entre dos extensiones. Luego se puede agregar otra extensión a la conversación (de esta forma creando una conferencia) al agarrar una extensión en el flash operador panel y colocándola en la conversación se conectará con la conferencia.

Música en espera (mpg123).

El IP-PBX ahora tiene usos de mpg123 para música de espera. Se toma una llamada en espera y así la persona que llama no escucha estática en el aire.

Correo de voz.

Para habilitar el mensaje de voz sobre una extensión simplemente "habilitar" esto en la configuración de las extensiones desde la guía del AMP.

3.6. Modelando la Base de Datos

Modelo Conceptual

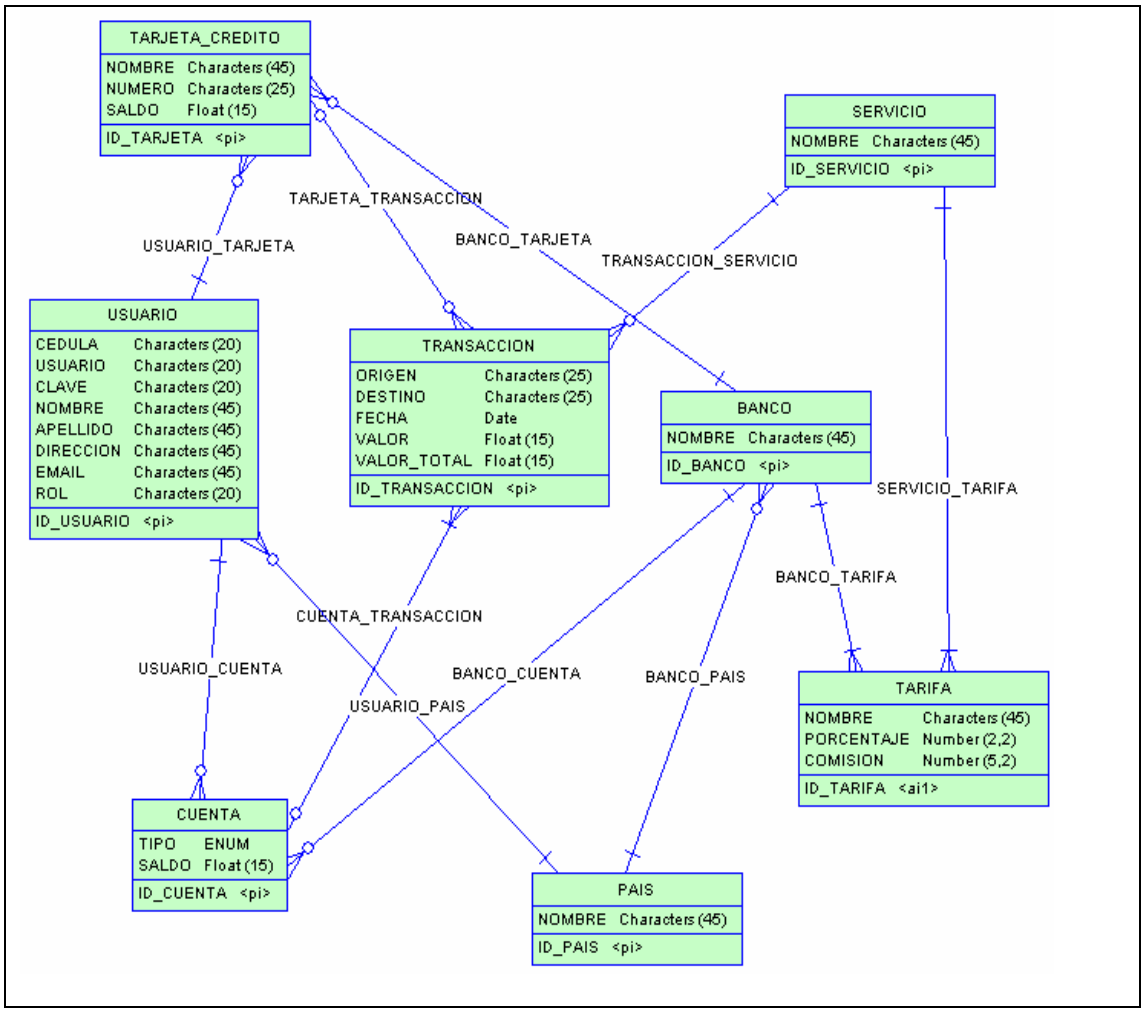


Figura 26. *Diseño conceptual de base de datos.*

Diccionario de datos.

Tabla: Usuario

Id_usuario: código de usuario/ VARCHAR[20]

Cedula: cédula del usuario / VARCHAR[20]

Usuario: usuario / VARCHAR[20]

Clave: clave del usuario / VARCHAR[20]

Nombre: nombre del usuario / VARCHAR[45]

Apellido: apellido del usuario / VARCHAR[45]

Dirección: dirección del usuario / VARCHAR[45]

e_mail: correo del usuario / VARCHAR[50]

rol : rol del usuario en el sistema (tipo de acceso) / ENUM ‘admin,user’

Tabla: País

Id _ país: código del país / INTEGER

país: nombre del país / VARCHAR[45]

Tabla: Cuenta

Id_cuenta : código de la cuenta / INT

tipo: tipo de la cuenta/ ENUN ‘ahorro,corriente’

saldo: valor que tiene en la cuenta/ DOUBLE

Tabla: Transacción

Id _ transacción : código de la transacción / INT

valor: valor que se cobra por la transacción / DOUBLE

origen: lugar origen de donde se hace la transacción/ VARCHAR [25]

destino: lugar destino a donde se hace la transacción / VARCHAR [25]

fecha: lugar destino a donde se hace la transacción / TIMESTAMP

valor_total: valor total que se cobra al usuario/ DOUBLE

Tabla: Tarifa

Id_tarifa : código de la tarifa / INT

nombre: nombre de la tarifa/ VARCHAR[45]

porcentaje: porcentaje que se cobra a la transacción/ DOUBLE

comisión: valor que indica la comisión a ganar / DOUBLE

Tabla: Banco

Id_banco: código del banco / INTEGER

nombre: nombre del banco / VARCHAR[45]

Tabla: Tarjeta de crédito

Id_tarjeta: código de la tarjeta de crédito/ INTEGER

nombre: nombre de la tarjeta / VARCHAR[45]

número : nombre de la tarjeta / VARCHAR[45]

saldo : saldo existente en la tarjeta / FLOAT

id_banco: clave foránea de la tabla banco / INT

Modelo lógico de datos.

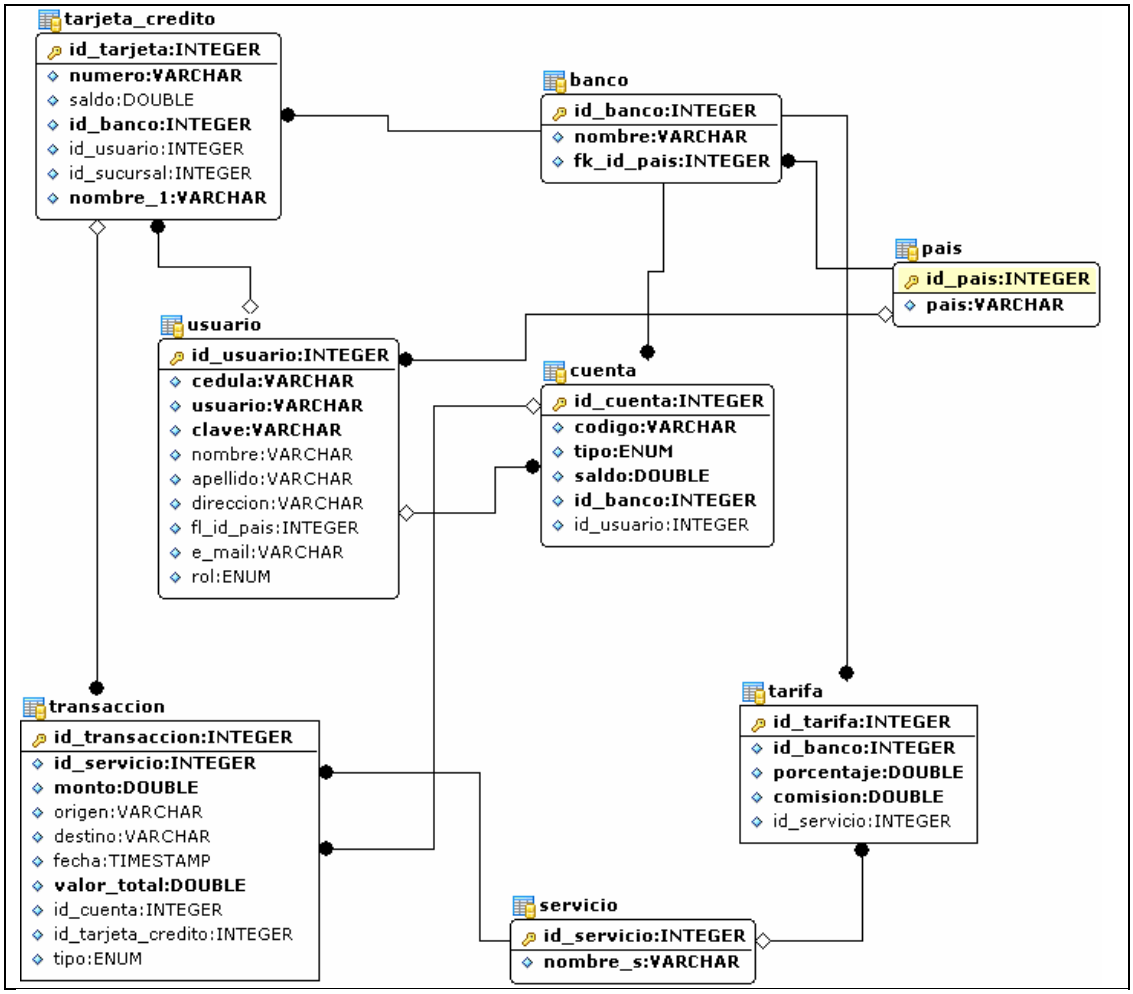


Figura 27. *Modelo Lógico de base de datos.*

3.7. Implementación del sistema vía Web.

El sistema vía web consta de la sección usuario y de la sección administrativa, las cuales se manejan independientemente y ambas poseen sus niveles de acceso y capacidades de operaciones en el sistema.

El modulo de administración consta de las siguientes opciones:

Bancos:

- Listar bancos registrados en el sistema.
- Modificar y eliminar un banco registrado en el sistema.
- Registrar un nuevo banco el sistema.

Tarifas:

- Listar tarifas de comisiones por cada uno de los servicios bancarios.
- Modificar y eliminar una tarifa registrada en el sistema.
- Agregar una nueva tarifa en el sistema.

Servicios:

- Listar servicios bancarios permitidos en el sistema
- Modificar y eliminar un servicio bancario registrado en el sistema.
- Agregar un nuevo servicio bancario en el sistema.

Usuarios:

- Buscar usuarios registrados en el sistema.
- Modificar y eliminar un usuario registrado en el sistema.
- Registrar un nuevo usuario en el sistema.

El modulo de usuario consta de las siguientes opciones:

Mis Cuentas:

- Realizar transacciones sobre cuentas bancarias.
- Listar las cuentas disponibles del cliente.

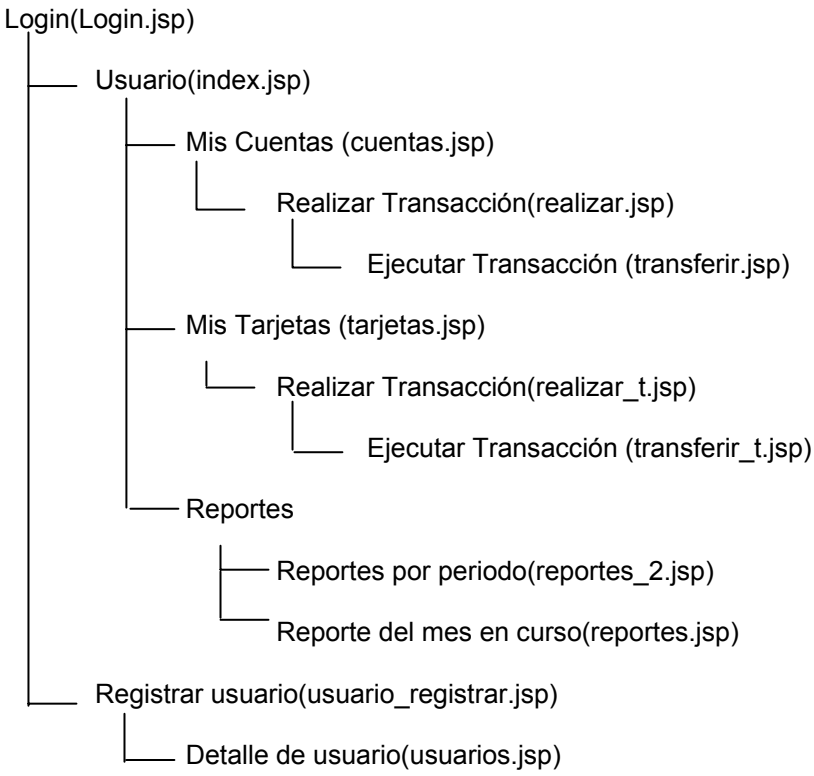
Mis Tarjetas:

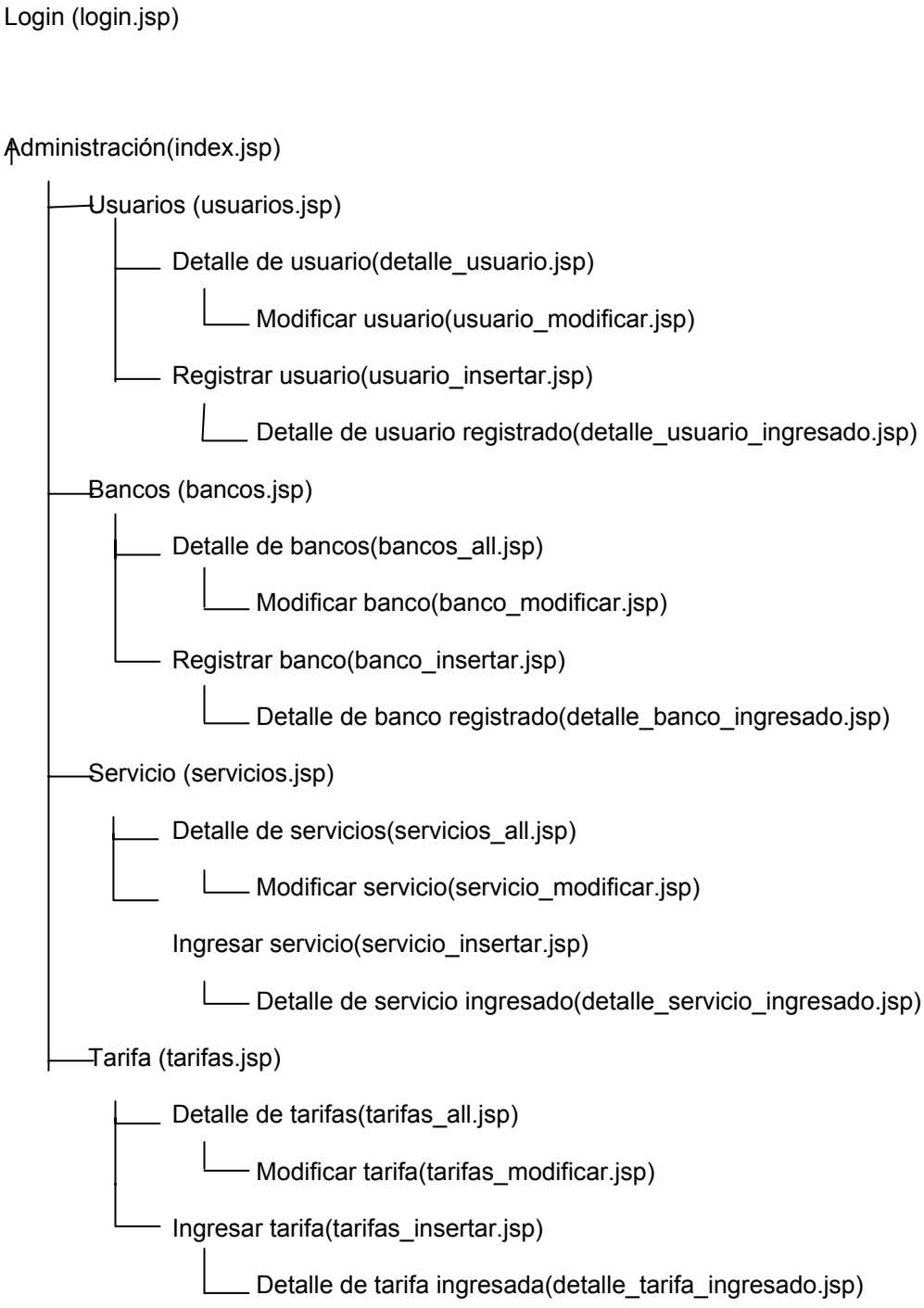
- Realizar transacciones sobre tarjetas de crédito
- Listar las tarjetas disponibles del cliente.

Reportes:

- Visualizar reportes de transacciones por periodo dado.
- Visualizar reporte de transacciones del mes en curso.

Diagrama de flujo de navegación:




Login (login.jsp)
Administración(index.jsp)
Usuarios (usuarios.jsp)
Detalle de usuario(detalle_usuario.jsp)
Modificar usuario(usuario_modificar.jsp)
Registrar usuario(usuario_insertar.jsp)
Detalle de usuario registrado(detalle_usuario_ingresado.jsp)
Bancos (bancos.jsp)
Detalle de bancos(bancos_all.jsp)
Modificar banco(banco_modificar.jsp)
Registrar banco(banco_insertar.jsp)
Detalle de banco registrado(detalle_banco_ingresado.jsp)
Servicio (servicios.jsp)
Detalle de servicios(servicios_all.jsp)
Modificar servicio(servicio_modificar.jsp)
Ingresar servicio(servicio_insertar.jsp)
Detalle de servicio ingresado(detalle_servicio_ingresado.jsp)
Tarifa (tarifas.jsp)
Detalle de tarifas(tarifas_all.jsp)
Modificar tarifa(tarifas_modificar.jsp)
Ingresar tarifa(tarifas_insertar.jsp)
Detalle de tarifa ingresada(detalle_tarifa_ingresado.jsp)

Para poder visualizar la codificación del sistema vía Web revisar los Apéndices A.

3.8. Implementación de transacciones usando IVR.

La implementación de las transacciones usando IVR se las realiza utilizando Asterisk@Home previamente instalado en el sistema operativo Linux, la programación se la realiza en el archivo extension.conf, en donde se invoca a la base de datos dependiendo de los códigos marcados por el teléfono.

El procedimiento a seguir para realizar una transacción usando el sistema IVR es el siguiente:

1) **Mensaje 1:** “Bienvenidos al sistema telefónico transaccional de la empresa ABC, si usted es un usuario registrado favor ingrese su código secreto”.
2) Ingresar código secreto del usuario.
3) Verificar si código ingresado se encuentra registrado en el sistema.
4) Si código ingresado está incorrecto, ir a la línea 5, caso contrario ir a línea 6.

5) **Mensaje 2:** “Estimado usuario el código ingresado está incorrecto, favor verifique y vuelva a ingresarlo”. Ir a línea 2.
6) **Mensaje 3:** “Bienvenido NNN”.
7) **Mensaje 4:** “Usded desea opción 1 enviar dinero u opción 2 recibir dinero”.
8) Si escogió opción enviar ir a la línea 9, caso contrario ir a la línea 36.
9) **Mensaje 5:** “beep, beep”
10) **Mensaje 6:** “Con que tarjeta ud. desea realizar la transacción, opción 1 VISA, opción 2 Diners, opción 3 American Express, opción 4 Mastercard”
11) Escoge con que tarjeta desea realizar la transacción y verifica si la tarjeta seleccionada se encuentra registrada en el sistema. Si no se encuentra registrada en el sistema ir a la línea 12, caso contrario ir a la línea 13.
12) **Mensaje 7:** “Usted no posee esta tarjeta”. Ir a la línea 10.
13) **Mensaje 8:** “Ingrese el código de su tarjeta de crédito”
14) Verifica si el código ingresado es correcto, si no está correcto ir a la línea 15, caso contrario ir a la línea 16.
15) **Mensaje 9:** “Código ingresado está incorrecto”. Ir a la línea 13.
16) **Mensaje 10:** “Ingrese la clave de su tarjeta de crédito”
17) **Mensaje 11:** “Ingrese la fecha de expiración de su tarjeta en formato día/mes/año”.

18) Verifica si la clave y la fecha de expiración ingresada es correcta, si no está correcta ir a la línea 19, caso contrario ir a la línea 20.

19) **Mensaje 12:** “Clave y fecha de expiración están incorrectos”. Ir a la línea 16.

20) **Mensaje 13:** “Ingrese el monto a enviar”.

21) Verifica si existe saldo suficiente para debitar el monto.

22) Si no existe saldo suficiente ir a la línea 23 caso contrario ir a la línea 24.

23) **Mensaje 14:** “Saldo insuficiente para realizar transacción”. Ir a línea 20.

24) **Mensaje 15:** “Ingrese el código del cliente al cual desea enviar dinero”

25) Ingresa el código del cliente y verifica si existe en el sistema. Si no existe en el sistema ir a la línea 26, caso contrario ir a línea 27.

26) **Mensaje 16:** “Código ingresado no se encuentra registrado en nuestro sistema”. Ir a la línea 24.

27) **Mensaje 17:** “El cliente al cual va enviar dinero se llama PPP”

28) **Mensaje 18:** “Ingrese el nombre de la tarjeta para depositar, opción 1 VISA, opción 2 Diners, opción 3 American Express, opción 4 Mastercard”

29) Escoge tipo de tarjeta y verifica si usuario posee tarjeta seleccionada. Si no posee tarjeta seleccionada ir a opción 30, caso contrario ir a opción 31.

30) **Mensaje 19:** “Tarjeta seleccionada no está registrada para el usuario”. Ir a línea 28.
31) **Mensaje 20:** “Ingrese el código de la tarjeta de crédito”
32) Verifica si el código ingresado es correcto, si no está correcto ir a la línea 33, caso contrario ir a la línea 34.
33) **Mensaje 21:** “Código ingresado está incorrecto”. Ir a la línea 31.
34) **Mensaje 22:** “Está seguro de realizar transacción 1 SI, 2 NO”
35) Si no está seguro ir a la línea 40, caso contrario ejecutar transacción con todos los parámetros ingresados e ir a línea 40.
36) Verifica si existe algún depósito en una de sus tarjetas de crédito.
37) Si no tiene ningún depósito en sus tarjetas de crédito, Ir a la línea 38, caso contrario ir a la línea 39.
38) **Mensaje 23:** “Usted no tiene giro de dinero en sus tarjetas de crédito”
39) **Mensaje 24:** “Usted tiene un giro de $$$$ realizado por NNN, depositado en su tarjeta de crédito número ###”
40) **Mensaje 25:** “Muchas Gracias por usar nuestro servicio”

El código de programación para esta implementación en asterisk la podemos encontrar en el Apéndice B.

CAPÍTULO 4.

4. FACTIBILIDAD DEL PROYECTO

4.1 Análisis técnico:

Para la institución ABC el proyecto técnicamente es factible debido a que poseen infraestructura para comunicación entre sucursales con una muy buena calidad de comunicación, además la empresa que les provee el servicio tiene la capacidad suficiente para poder ampliar el ancho de banda que se necesite.

Para la integración con la central telefónica convencional no existirá ningún inconveniente por que en todas sus sucursales poseen centrales telefónicas digital compatible con el sistema IPBX.

Respecto a los Servidores IVR e IPBX, los equipos que poseen como firewall se los podría reutilizar, por que poseen características óptimas para la implementación del sistema. Lo que si se recomienda es la adquisición de un PC en donde se almacenaría la Base de Datos y otro para la aplicación web que estaría pública en el Internet.

Para la institucion ABC en cada sucursal constara con Tarjetas FXO de 4 puertos para recibir las llamadas entrantes de los clientes y estas a su vez permitan interactuar con el IVR para poder realizar consultas y transacciones a la base de datos.

En cada sucursal tambien se implementara la segmentacion de ancho de banda para canales de voz, ganando asi fiabilidad en la comunicación de voz.

4.2 Análisis de costos:

FASE 1: Optimización de red privada de datos del cliente.

CANT	DESCRIPCION	PU ($)	PT ($)
3	PC PENTIUM IV 2,8 Ghz, HDD 80 Gbyte, 256 RAM.	700,00	2100,00
3	Licencia LINUX Centos 4,2.	1500,00	4500,00
3	Instalación y configuración de servidores Linux.	200,00	600,00
3	Configuración de segmentación de ancho de banda para voz y datos.	150,00	450,00
	TOTAL		**7650,00**

Tabla 8: *Costos optimización red privada de datos*

FASE 2: Implementación del servicio de VoIP en la red del cliente.

CANT	DESCRIPCION	PU ($)	PT ($)
3	Equipos terminales Sipura SPA 841	250,00	750,00
3	Tarjetas FXO 4 puertos	550,00	1650,00
3	Licencias para G729, 4 canales c/u	70,00	210,00
3	Gramstream modelo GXP-2000	150,00	450,00
3	Instalación y configuración de servicio VoIP	300,00	900,00
	TOTAL		**3960,00**

Tabla 9: *Costos implementación del servicio VoIP*

FASE 3: Implementación del IPBX.

CANT	DESCRIPCION	PU ($)	PT ($)
3	Instalación y configuración de IPBX locales deacuerdo a necesidad de cada sucursal.	350,00	1050,00
	TOTAL		**1050,00**

Tabla 10: *Costos implementación IPBX*

FASE 4: Instalación de sistema de transacciones de envio y retiro de dinero vía Web.

CANT	DESCRIPCION	PU ($)	PT ($)
1	Venta e instalación de sistema de transacciones de envio y retiro de dinero vía Web con licencia para 1000 conecciones concurrentes.	8000,00	8400,00
	TOTAL		**8400,00**

Tabla 11: *Costos de instalación del sistema de transacciones*

FASE 5: Implementación de transacciones usando IVR.

CANT	DESCRIPCION	PU ($)	PT ($)
3	Venta, instalación y configuración de sistema de transacciones de envio y retiro de dinero usando IVR para 32 conecciones simultáneas.	3000,00	3000,00
	TOTAL		**3000,00**

Tabla 12: *Costos de implementación transacciones usando IVR*

Nota: Todos estos precios no incluyen impuestos.

CONCLUSIONES Y RECOMENDACIONES

- La tecnología IVR fácilmente se la puede implementar para el pago de servicios básicos, tales como agua, luz, teléfono. Se lo podría también implementar para el pago de impuestos prediales, SRI, tarjetas de compra, etc.

- El proyecto así como tiene sus ventajas, también tiene sus desventajas, al momento solo se puede brindar seguridad con la clave de acceso al sistema, es decir si alguien llegase a conocerla, podrá realizar transacciones.

- La implementación de la central telefónica con una PC tiene su desventaja respecto a una central telefónica convencial, con la estabilidad del funcionamiento, una PC pueden ocurrir fallas en el sistema.

- La tecnología VoIP permite abaratar costos de implementación si el proyecto se lo realiza con una central telefónica moderna convencional que soporte IVR.

- Cada vez resulta más sencillo programar centrales telefónicas IPs con el lenguaje Asterisk, esto estrecha más la posibilidad de que proyectos de telefonía IP con integración de software y bases de datos se las realice utilizando esta tecnología.

- La implementación a través del internet del proyecto capta un alto porcentaje de usuarios que desearían realizar éste tipo de transacciones por el Internet.

- El proyecto, tambien está orientado a poder realizar cualquier tipo de transacción bancaria entre diferentes bancos locales o internacionales, bien sea utilizando el sistema IVR o vía web.

APENDICES

Apéndice A: Código JAVA para la aplicación vía web.

/* Código AdministracionServlet*/

```
package money.servlet;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;
import javax.naming.*;
import javax.sql.*;

import money.bean.UsuarioBean;

public class AdministracionServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;

	// Connection dbCon;
	DataSource ds;

	HttpSession session;

	/* Initialize servlet. Use JNDI to look up a DataSource */
	public void init() {
		try {
			Context initCtx = new InitialContext();
			Context envCtx = (Context) initCtx.lookup("java:comp/env");
			ds = (DataSource) envCtx.lookup("jdbc/MoneyDB");
		} catch (javax.naming.NamingException e) {
			e.printStackTrace();
		}
	}

	public void doPost(HttpServletRequest _req, HttpServletResponse _res)
			throws ServletException, IOException {
		/* Refresh session attributes */
		session = _req.getSession();
		session.removeAttribute("loginError");
		session.removeAttribute("submitError");

		String action = _req.getParameter("action");
		/* Authenticate user if request comes from login page */
		if (action.equals("login")) {
			String uid = _req.getParameter("UID");
			String pwd = _req.getParameter("PWD");
			if (authenticate(uid, pwd)) {
				// Si el usuario y clave son validos entonces creo una session
				UsuarioBean usuario = new UsuarioBean();
				usuario.setApellido("Money&Money");
				usuario.setNombre("Administración");
				session.setAttribute("usuario", usuario);
				session.setAttribute("validUser", "y");
				session.setAttribute("uid", uid);
				gotoPage("index.jsp", _req, _res);
			} else {
				// Si el usuario y clave son invalidos regreso a la
				// pagina de login indicando que está erroneo la informacion
				loginError(_req, _res);
			}
		} else {
			loginError(_req, _res);
```

```
                }

        } /* Send request to a different page */

        private void gotoPage(String _page, HttpServletRequest _req,
                        HttpServletResponse _res) throws IOException, ServletException {
                _res.sendRedirect(_req.getContextPath() + "/administracion/" + _page);
        }

        /* Set error attributes in session and return to Login page */
        private void loginError(HttpServletRequest _req, HttpServletResponse _res)
                        throws IOException, ServletException {
                session.setAttribute("validUser", "n");
                session.setAttribute("loginError", "y");
                gotoPage("login.jsp", _req, _res);
        }

        /* Check if the user is valid */
        private boolean authenticate(String _uid, String _pwd) {
                Connection dbCon = null;
                ResultSet rs = null;
                try {
                        dbCon = ds.getConnection();
                        Statement s = dbCon.createStatement();
                        rs = s
                                        .executeQuery("select * from usuario where usuario = '"
                                                        + _uid + "' and clave = '" + _pwd + "'" + " and rol =
'admin'");
                        return (rs.next());
                } catch (java.sql.SQLException e) {
                        e.printStackTrace();
                } finally {
                        try {
                                dbCon.close();
                        } catch (SQLException e) {
                                e.printStackTrace();
                        }
                }
                return false;
        }

        public void destroy() {
        }
}
```

/*Código UsuarioServlet*/

```
package money.servlet;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;
import javax.naming.*;
import javax.sql.*;

import money.bean.UsuarioBean;

public class UsuarioServlet extends HttpServlet {
        private static final long serialVersionUID = 1L;

        // Connection dbCon;
        DataSource ds;
```

```
HttpSession session;

UsuarioBean usuario;

/* Initialize servlet. Use JNDI to look up a DataSource */
public void init() {
	try {
		Context initCtx = new InitialContext();
		Context envCtx = (Context) initCtx.lookup("java:comp/env");
		ds = (DataSource) envCtx.lookup("jdbc/MoneyDB");
	} catch (javax.naming.NamingException e) {
		e.printStackTrace();
	}
}

public void doPost(HttpServletRequest _req, HttpServletResponse _res)
		throws ServletException, IOException {
	/* Refresh session attributes */
	session = _req.getSession();
	session.removeAttribute("loginError");
	session.removeAttribute("submitError");

	String action = _req.getParameter("action");
	/* Authenticate user if request comes from login page */
	if (action.equals("login")) {
		String uid = _req.getParameter("UID");
		String pwd = _req.getParameter("PWD");
		if (authenticate(uid, pwd)) {
			// Si el usuario y clave son validos entonces creo una session
			session.setAttribute("usuario", usuario);
			session.setAttribute("validUser", "y");
			session.setAttribute("uid", uid);
			gotoPage("index.jsp", _req, _res);
		} else {
			// Si el usuario y clave son invalidos regreso a la
			// pagina de login indicando que está erroneo la informacion
			loginError(_req, _res);
		}
	} else {
		loginError(_req, _res);
	}

} /* Send request to a different page */

private void gotoPage(String _page, HttpServletRequest _req,
		HttpServletResponse _res) throws IOException, ServletException {
	_res.sendRedirect(_req.getContextPath() + "/" + _page);
}

/* Set error attributes in session and return to Login page */
private void loginError(HttpServletRequest _req, HttpServletResponse _res)
		throws IOException, ServletException {
	session.setAttribute("validUser", "n");
	session.setAttribute("loginError", "y");
	gotoPage("login.jsp", _req, _res);
}

/* Check if the user is valid */
private boolean authenticate(String _uid, String _pwd) {
	Connection dbCon = null;
	ResultSet rs = null;
	try {
		dbCon = ds.getConnection();
		Statement s = dbCon.createStatement();
		rs = s
```

```
                                                .executeQuery("select * from usuario where usuario = '"
                                                        + _uid + "' and clave = '" + _pwd + "'" + " and rol =
'user'");

                        if(rs.next()){
                                usuario = new UsuarioBean();
                                usuario.buscar(dbCon,rs.getInt("id_usuario") + "");
                                return true;
                        } else {
                                return false;
                        }
                } catch (java.sql.SQLException e) {
                        e.printStackTrace();
                } finally {
                        try {
                                dbCon.close();
                        } catch (SQLException e) {
                                e.printStackTrace();
                        }
                }
                return false;
        }

        public void destroy() {
        }
}
```

/*Código BancoBean */

```
package money.bean;

import java.sql.Connection;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.http.HttpServletRequest;

public class BancoBean implements java.io.Serializable {
        private static final long serialVersionUID = 1L;

        private String nombre, fl_id_pais, pais;

        private int id;

        public BancoBean() {
        }

        public String getFl_id_pais() {
                return fl_id_pais;
        }

        public void setFl_id_pais(String fl_id_pais) {
                this.fl_id_pais = fl_id_pais;
        }

        public int getId() {
                return id;
        }

        public String getPais() {
                return pais;
        }
```

```
public void setPais(String pais) {
	this.pais = pais;
}

public void setId(int id) {
	this.id = id;
}

public String getNombre() {
	return nombre;
}

public void setNombre(String nombre) {
	this.nombre = nombre;
}

public void capturarParametros(HttpServletRequest _req) {
	// Populate bean properties from request parameters
	setId(Integer.parseInt(_req.getParameter("id")));
	setNombre(_req.getParameter("nombre"));
	setFl_id_pais(_req.getParameter("fl_id_pais"));
}

public boolean insertar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);

	try {
		s = _dbCon.createStatement();

		// Insert record
		sql.append("call ingresar_banco('" + nombre + "', " + fl_id_pais
				+ ")");

		s.executeUpdate(sql.toString());
		s.close();
	} catch (SQLException e) {
		e.printStackTrace();
		return false;
	}
	return true;
}

public boolean modificar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);

	try {
		s = _dbCon.createStatement();

		// Update record
		sql.append("call modificar_banco('" + nombre + "'," + id + ")");

		System.out.println(sql.toString());
		s.executeUpdate(sql.toString());
		s.close();
	} catch (SQLException e) {
		e.printStackTrace();
		return false;
	}
	return true;
}

public boolean eliminar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);
```

```
		try {
			s = _dbCon.createStatement();

			// Update record
			sql.append("call eliminar_banco(" + id + ")");

			System.out.println(sql.toString());
			s.executeUpdate(sql.toString());
			s.close();
		} catch (SQLException e) {
			e.printStackTrace();
			return false;
		}
		return true;
	}

	public boolean pais(Connection _dbCon) {
		Statement s = null;
		ResultSet rs = null;
		StringBuffer sql = new StringBuffer(256);

		try {
			s = _dbCon.createStatement();

			sql.append("call buscar_pais(" + fl_id_pais + ")");

			rs = s.executeQuery(sql.toString());
			if (rs.next()) {
				setPais(rs.getString("pais"));
			}
			rs.close();
			s.close();
		} catch (SQLException e) {
			e.printStackTrace();
			return false;
		}
		return true;
	}

	public ResultSet listarBancos(Connection _dbCon){
		Statement s = null;
		ResultSet rs = null;
		StringBuffer sql = new StringBuffer(256);
		try {
			s = _dbCon.createStatement();

			sql.append("call listar_bancos()");

			rs = s.executeQuery(sql.toString());
			//s.close();
		} catch (Exception e) {
			e.printStackTrace();
		}
		return rs;
	}
}
```

/* Código TarifaBean */

```
package money.bean;

import java.sql.Connection;
```

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.http.HttpServletRequest;

public class TarifaBean implements java.io.Serializable {
	private static final long serialVersionUID = 1L;

	private String id_servicio, id_banco, banco, servicio;

	private double comision, porcentaje;

	private int id;

	public TarifaBean() {
	}

	public int getId() {
		return id;
	}

	public void setId(int id) {
		this.id = id;
	}

	public String getBanco() {
		return banco;
	}

	public void setBanco(String banco) {
		this.banco = banco;
	}

	public double getComision() {
		return comision;
	}

	public String getServicio() {
		return servicio;
	}

	public void setServicio(String servicio) {
		this.servicio = servicio;
	}

	public void setComision(double comision) {
		this.comision = comision;
	}

	public String getId_banco() {
		return id_banco;
	}

	public void setId_banco(String id_banco) {
		this.id_banco = id_banco;
	}

	public String getId_servicio() {
		return id_servicio;
	}

	public void setId_servicio(String id_servicio) {
		this.id_servicio = id_servicio;
	}
```

```
public double getPorcentaje() {
	return porcentaje;
}

public void setPorcentaje(double porcentaje) {
	this.porcentaje = porcentaje;
}

public void capturarParametros(HttpServletRequest _req) {
	// Populate bean properties from request parameters
	setId(Integer.parseInt(_req.getParameter("id")));
	setId_servicio(_req.getParameter("id_servicio"));
	setId_banco(_req.getParameter("id_banco"));
	setPorcentaje(Double.parseDouble(_req.getParameter("porcentaje")));
	setComision(Double.parseDouble(_req.getParameter("comision")));
}

public boolean insertar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);

	try {
		s = _dbCon.createStatement();

		// Insert record
		sql.append("call ingresar_tarifa(" + id_servicio + ", " + id_banco
				+ "," + porcentaje + "," + comision + ")");

		s.executeUpdate(sql.toString());
		s.close();
	} catch (SQLException e) {
		e.printStackTrace();
		return false;
	}
	return true;
}

public boolean modificar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);

	try {
		s = _dbCon.createStatement();

		// Update record
		sql.append("call modificar_tarifa(" + id + "," + id_banco + ","
				+ porcentaje + "," + comision + ")");

		System.out.println(sql.toString());
		s.executeUpdate(sql.toString());
		s.close();
	} catch (SQLException e) {
		e.printStackTrace();
		return false;
	}
	return true;
}

public boolean eliminar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);

	try {
		s = _dbCon.createStatement();
```

```
                        // Update record
                        // sql.append("call eliminar_banco(" + id + ")");

                        System.out.println(sql.toString());
                        s.executeUpdate(sql.toString());
                        s.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                        return false;
                }
                return true;
        }

        public boolean banco(Connection _dbCon, String id_banco) {
                Statement s = null;
                ResultSet rs = null;
                StringBuffer sql = new StringBuffer(256);

                try {
                        s = _dbCon.createStatement();

                        sql.append("call consulta_datos_banco(" + id_banco + ")");

                        rs = s.executeQuery(sql.toString());
                        if (rs.next()) {
                                setBanco(rs.getString("banco.nombre"));
                        }
                        rs.close();
                        s.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                        return false;
                }
                return true;
        }

        public boolean servicio(Connection _dbCon, String id_banco) {
                Statement s = null;
                ResultSet rs = null;
                StringBuffer sql = new StringBuffer(256);

                try {
                        s = _dbCon.createStatement();

                        sql.append("call consulta_datos_servicio(" + id_servicio + ")");

                        rs = s.executeQuery(sql.toString());
                        if (rs.next()) {
                                setServicio(rs.getString("servicio.nombre_s"));
                        }
                        rs.close();
                        s.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                        return false;
                }
                return true;
        }

        public ResultSet listarTarifas(Connection _dbCon){
                Statement s = null;
                ResultSet rs = null;
                StringBuffer sql = new StringBuffer(256);
                try {
                        s = _dbCon.createStatement();
```

```
                                sql.append("call listar_tarifas()");

                                rs = s.executeQuery(sql.toString());
                                //s.close();
                        } catch (Exception e) {
                                e.printStackTrace();
                        }
                        return rs;
                }
}
```

/*Código Cuenta Bean */

```
package money.bean;

import java.sql.Connection;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.http.HttpServletRequest;

public class CuentaBean implements java.io.Serializable {
        private static final long serialVersionUID = 1L;

        private String código, tipo, id_banco, id_sucursal, id_usuario, pais,
                        banco;

        private double saldo;

        private int id;

        public CuentaBean() {
        }

        public int getId() {
                return id;
        }

        public String getPais() {
                return pais;
        }

        public void setPais(String pais) {
                this.pais = pais;
        }

        public void setId(int id) {
                this.id = id;
        }

        public String getBanco() {
                return banco;
        }

        public void setBanco(String banco) {
                this.banco = banco;
        }

        public String getCodigo() {
                return código;
        }

        public void setCódigo(String código) {
```

```
		this.código = código;
}

public String getId_banco() {
		return id_banco;
}

public void setId_banco(String id_banco) {
		this.id_banco = id_banco;
}

public String getId_sucursal() {
		return id_sucursal;
}

public void setId_sucursal(String id_sucursal) {
		this.id_sucursal = id_sucursal;
}

public String getId_usuario() {
		return id_usuario;
}

public void setId_usuario(String id_usuario) {
		this.id_usuario = id_usuario;
}

public double getSaldo() {
		return saldo;
}

public void setSaldo(double saldo) {
		this.saldo = saldo;
}

public String getTipo() {
		return tipo;
}

public void setTipo(String tipo) {
		this.tipo = tipo;
}

public void capturarParametros(HttpServletRequest _req) {
		// Populate bean properties from request parameters
		setId(Integer.parseInt(_req.getParameter("id")));
		/*
		 * setNombre(_req.getParameter("nombre"));
		 * setFl_id_pais(_req.getParameter("fl_id_pais"));
		 */
}

public boolean insertar(Connection _dbCon) {
		Statement s = null;
		StringBuffer sql = new StringBuffer(256);

		try {
				s = _dbCon.createStatement();

				// Insert record
				//sql.append("call ingresar_banco('" + nombre + "', " + fl_id_pais
				//				+ ")");

				s.executeUpdate(sql.toString());
				s.close();
		} catch (SQLException e) {
```

```
                    e.printStackTrace();
                    return false;
            }
            return true;
}

public boolean modificar(Connection _dbCon) {
            Statement s = null;
            StringBuffer sql = new StringBuffer(256);

            try {
                    s = _dbCon.createStatement();

                    // Update record
                    //sql.append("call modificar_banco('" + nombre + "'," + id + ")");

                    System.out.println(sql.toString());
                    s.executeUpdate(sql.toString());
                    s.close();
            } catch (SQLException e) {
                    e.printStackTrace();
                    return false;
            }
            return true;
}

public boolean eliminar(Connection _dbCon) {
            Statement s = null;
            StringBuffer sql = new StringBuffer(256);

            try {
                    s = _dbCon.createStatement();

                    // Update record
                    //sql.append("call eliminar_banco(" + id + ")");

                    System.out.println(sql.toString());
                    s.executeUpdate(sql.toString());
                    s.close();
            } catch (SQLException e) {
                    e.printStackTrace();
                    return false;
            }
            return true;
}

public boolean buscarCuenta(Connection _dbCon, String id_cuenta) {
            Statement s = null;
            ResultSet rs = null;
            StringBuffer sql = new StringBuffer(256);

            try {
                    s = _dbCon.createStatement();

                    sql.append("call select_cuenta_id(" + id_cuenta + ")");

                    rs = s.executeQuery(sql.toString());
                    if (rs.next()) {
                            setId(rs.getInt("id_cuenta"));
                            setId_banco(rs.getString("id_banco"));
                            setBanco(rs.getString("nombre"));
                            setPais(rs.getString("pais"));
                            setCódigo(rs.getString("código"));
                            setTipo(rs.getString("tipo"));
                            setSaldo(rs.getDouble("saldo"));
                    }
```

```
                        rs.close();
                        s.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                        return false;
                }
                return true;
        }

        public boolean buscarCuentaCódigo(Connection _dbCon, String código) {
                Statement s = null;
                ResultSet rs = null;
                StringBuffer sql = new StringBuffer(256);
                boolean ban = false;

                try {
                        s = _dbCon.createStatement();

                        sql.append("call select_cuenta('" + código + "')");

                        rs = s.executeQuery(sql.toString());
                        if (rs.next()) {
                                ban = true;
                                setId(rs.getInt("id_cuenta"));
                        }
                        //rs.close();
                        //s.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                        return false;
                }
                return ban;
        }
}
```

/* Código TarjetaBean*/

```
package money.bean;

import java.sql.Connection;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.http.HttpServletRequest;

public class TarjetaBean implements java.io.Serializable {
        private static final long serialVersionUID = 1L;

        private String código, id_banco, id_usuario, pais,
                        banco,nombre_1;

        private double saldo;

        private int id;

        public TarjetaBean() {
        }

        public int getId() {
                return id;
        }
```

```
public String getPais() {
		return pais;
}

public void setPais(String pais) {
		this.pais = pais;
}

public void setId(int id) {
		this.id = id;
}

public String getBanco() {
		return banco;
}

public String getNombre_1() {
		return nombre_1;
}

public void setNombre_1(String nombre_1) {
		this.nombre_1 = nombre_1;
}

public void setBanco(String banco) {
		this.banco = banco;
}

public String getCódigo() {
		return código;
}

public void setCódigo(String código) {
		this.código = código;
}

public String getId_banco() {
		return id_banco;
}

public void setId_banco(String id_banco) {
		this.id_banco = id_banco;
}

public String getId_usuario() {
		return id_usuario;
}

public void setId_usuario(String id_usuario) {
		this.id_usuario = id_usuario;
}

public double getSaldo() {
		return saldo;
}

public void setSaldo(double saldo) {
		this.saldo = saldo;
}

public void capturarParametros(HttpServletRequest _req) {
		// Populate bean properties from request parameters
		setId(Integer.parseInt(_req.getParameter("id")));
		/*
		 * setNombre(_req.getParameter("nombre"));
		 * setFl_id_pais(_req.getParameter("fl_id_pais"));
```

```
		 */
}

public boolean insertar(Connection _dbCon) {
		Statement s = null;
		StringBuffer sql = new StringBuffer(256);

		try {
			s = _dbCon.createStatement();

			// Insert record
			//sql.append("call ingresar_banco('" + nombre + "', " + fl_id_pais
			//			+ ")");

			s.executeUpdate(sql.toString());
			s.close();
		} catch (SQLException e) {
			e.printStackTrace();
			return false;
		}
		return true;
}

public boolean modificar(Connection _dbCon) {
		Statement s = null;
		StringBuffer sql = new StringBuffer(256);

		try {
			s = _dbCon.createStatement();

			// Update record
			//sql.append("call modificar_banco('" + nombre + "'," + id + ")");

			System.out.println(sql.toString());
			s.executeUpdate(sql.toString());
			s.close();
		} catch (SQLException e) {
			e.printStackTrace();
			return false;
		}
		return true;
}

public boolean eliminar(Connection _dbCon) {
		Statement s = null;
		StringBuffer sql = new StringBuffer(256);

		try {
			s = _dbCon.createStatement();

			// Update record
			//sql.append("call eliminar_banco(" + id + ")");

			System.out.println(sql.toString());
			s.executeUpdate(sql.toString());
			s.close();
		} catch (SQLException e) {
			e.printStackTrace();
			return false;
		}
		return true;
}

public boolean buscarTarjeta(Connection _dbCon, String id_tarjeta) {
		Statement s = null;
		ResultSet rs = null;
```

```
		StringBuffer sql = new StringBuffer(256);

		try {
			s = _dbCon.createStatement();

			sql.append("call select_tarjeta_id(" + id_tarjeta + ")");

			rs = s.executeQuery(sql.toString());
			if (rs.next()) {
				setId(rs.getInt("id_tarjeta"));
				setId_banco(rs.getString("id_banco"));
				setBanco(rs.getString("nombre"));
				setNombre_1(rs.getString("nombre_1"));
				setPais(rs.getString("pais"));
				setCódigo(rs.getString("número"));
				setSaldo(rs.getDouble("saldo"));
			}
			rs.close();
			s.close();
		} catch (SQLException e) {
			e.printStackTrace();
			return false;
		}
		return true;
	}

	public boolean buscarTarjetaCódigo(Connection _dbCon, String código) {
		Statement s = null;
		ResultSet rs = null;
		StringBuffer sql = new StringBuffer(256);
		boolean ban = false;

		try {
			s = _dbCon.createStatement();

			sql.append("call select_tarjeta('" + código + "')");

			rs = s.executeQuery(sql.toString());
			if (rs.next()) {
				ban = true;
				setId(rs.getInt("id_tarjeta"));
			}
			rs.close();
			s.close();
		} catch (SQLException e) {
			e.printStackTrace();
			return false;
		}
		return ban;
	}
}
```

/* Código servicioBean */

```
package money.bean;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.http.HttpServletRequest;

public class servicioBean implements java.io.Serializable {
```

```
private static final long serialVersionUID = 1L;

private String nombre;

private int id;

public ServicioBean() {
}

public int getId() {
	return id;
}

public void setId(int id) {
	this.id = id;
}

public String getNombre() {
	return nombre;
}

public void setNombre(String nombre) {
	this.nombre = nombre;
}

public void capturarParametros(HttpServletRequest _req) {
	// Populate bean properties from request parameters
	setId(Integer.parseInt(_req.getParameter("id")));
	setNombre(_req.getParameter("nombre"));
}

public boolean insertar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);

	try {
		s = _dbCon.createStatement();

		// Insert record
		sql.append("call ingresar_servicio('" + nombre + "'" + ")");

		s.executeUpdate(sql.toString());
		s.close();
	} catch (SQLException e) {
		e.printStackTrace();
		return false;
	}
	return true;
}

public boolean modificar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);

	try {
		s = _dbCon.createStatement();

		// Update record
		sql.append("call modificar_servicio('" + nombre + "'," + id + ")");

		System.out.println(sql.toString());
		s.executeUpdate(sql.toString());
		s.close();
	} catch (SQLException e) {
		e.printStackTrace();
		return false;
```

```
		}
		return true;
	}

	public boolean eliminar(Connection _dbCon) {
		Statement s = null;
		StringBuffer sql = new StringBuffer(256);

		try {
			s = _dbCon.createStatement();

			// Update record
			sql.append("call eliminar_servicio(" + id + ")");

			System.out.println(sql.toString());
			s.executeUpdate(sql.toString());
			s.close();
		} catch (SQLException e) {
			e.printStackTrace();
			return false;
		}
		return true;
	}

	public ResultSet listarServicios(Connection _dbCon){
		Statement s = null;
		ResultSet rs = null;
		StringBuffer sql = new StringBuffer(256);
		try {
			s = _dbCon.createStatement();

			sql.append("call listar_servicios()");

			rs = s.executeQuery(sql.toString());
			//s.close();
		} catch (Exception e) {
			e.printStackTrace();
		}
		return rs;
	}
}
```

/* Código UsuarioBean */

```
package money.bean;

import java.sql.Connection;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.http.HttpServletRequest;

import money.ServicesCuentasUsuario;

public class UsuarioBean implements java.io.Serializable {
	private static final long serialVersionUID = 1L;

	// Variables del Hotel
	private String identificacion, usuario, clave, nombre, apellido, direccion,
			fl_id_pais, email;

	private int id;
```

```
public UsuarioBean() {
}

public String getApellido() {
	return apellido;
}

public void setApellido(String apellido) {
	this.apellido = apellido;
}

public String getClave() {
	return clave;
}

public void setClave(String clave) {
	this.clave = clave;
}

public String getDireccion() {
	return direccion;
}

public void setDireccion(String direccion) {
	this.direccion = direccion;
}

public String getEmail() {
	return email;
}

public void setEmail(String email) {
	this.email = email;
}

public String getFl_id_pais() {
	return fl_id_pais;
}

public void setFl_id_pais(String fl_id_pais) {
	this.fl_id_pais = fl_id_pais;
}

public int getId() {
	return id;
}

public void setId(int id) {
	this.id = id;
}

public String getIdentificacion() {
	return identificacion;
}

public void setIdentificacion(String identificacion) {
	this.identificacion = identificacion;
}

public String getNombre() {
	return nombre;
}

public void setNombre(String nombre) {
	this.nombre = nombre;
```

```
}

public String getUsuario() {
	return usuario;
}

public void setUsuario(String usuario) {
	this.usuario = usuario;
}

public void capturarParametros(HttpServletRequest _req) {
	// Populate bean properties from request parameters
	setId(Integer.parseInt(_req.getParameter("id")));
	setIdentificacion(_req.getParameter("identificacion"));
	setUsuario(_req.getParameter("usuario"));
	setClave(_req.getParameter("clave"));
	setNombre(_req.getParameter("nombre"));
	setApellido(_req.getParameter("apellido"));
	setDireccion(_req.getParameter("direccion"));
	setFl_id_pais(_req.getParameter("fl_id_pais"));
	setEmail(_req.getParameter("e_mail"));
}

public boolean insertar(Connection _dbCon) {
	Statement s = null;
	StringBuffer sql = new StringBuffer(256);
	ResultSet rs = null;

	try {
		s = _dbCon.createStatement();
		_dbCon.setAutoCommit(false);
		// Insert record
		sql.append("call ingresar_usuario('" + identificacion + "', '"
				+ usuario + "', '" + clave + "', '" + nombre + "', '"
				+ apellido + "', '" + direccion + "', " + fl_id_pais
				+ ", '" + email + "')");

		s.executeUpdate(sql.toString());

		rs = s.executeQuery("call consulta_datos_user_cedula('"
				+ identificacion + "')");

		if (rs.next())
			id = rs.getInt("id_usuario");
		else
			return false;

		if (ServicesCuentasUsuario.crearCuentasUsuario(_dbCon,
				identificacion))
			System.out.println("Bien");
		else
			System.out.println("Mal");
		_dbCon.commit();
		s.close();
		_dbCon.setAutoCommit(true);
	} catch (SQLException e) {
		e.printStackTrace();
		try {
			_dbCon.rollback();
			_dbCon.setAutoCommit(true);
		} catch (SQLException e1) {
			e1.printStackTrace();
		}
		return false;
	}
	return true;
```

```
	}

	public boolean modificar(Connection _dbCon) {
		Statement s = null;
		StringBuffer sql = new StringBuffer(256);

		try {
			s = _dbCon.createStatement();
			_dbCon.setAutoCommit(false);
			// Update record
			sql.append("call modificar_usuario('" + identificacion + "', '"
					+ nombre + "', '" + apellido + "', '" + direccion + "', '"
					+ email + "'," + id + ")");
			System.out.println(sql.toString());
			s.executeUpdate(sql.toString());
			_dbCon.commit();
			s.close();
			_dbCon.setAutoCommit(true);
		} catch (SQLException e) {
			e.printStackTrace();
			try {
				_dbCon.rollback();
				_dbCon.setAutoCommit(true);
			} catch (SQLException e1) {
				e1.printStackTrace();
			}
			return false;
		}
		return true;
	}

	public boolean eliminar(Connection _dbCon) {
		Statement s = null;
		StringBuffer sql = new StringBuffer(256);

		try {
			s = _dbCon.createStatement();
			_dbCon.setAutoCommit(false);
			// Update record
			sql.append("call eliminar_usuario(" + id + ")");
			System.out.println(sql.toString());
			s.executeUpdate(sql.toString());
			_dbCon.commit();
			s.close();
			_dbCon.setAutoCommit(true);
		} catch (SQLException e) {
			e.printStackTrace();
			try {
				_dbCon.rollback();
				_dbCon.setAutoCommit(true);
			} catch (SQLException e1) {
				e1.printStackTrace();
			}
			return false;
		}
		return true;
	}

	public String getSaludo(String pre) {

		StringBuffer saludo = new StringBuffer();

		saludo
				.append("<td width=\"70%\"><font class=\"SmallFont\"> Usted es
 ");
```

```
                saludo
                        .append(this.getNombre().trim() + " "
                                + this.getApellido().trim());

                saludo.append("</font></td>" + "<td valign=\"top\" width=\"30%\">"
                        + "<a class=\"LinkFont\"href=\"" + pre + "salir.jsp\">"
                        + "<div align=\"right\">");

                saludo.append("Cerrar Sesi&oacute;n");

                saludo.append("</div></a></td>");

                return saludo.toString();
        }

        public boolean buscar(Connection _dbCon, String id_usuario) {
                Statement s = null;
                ResultSet rs = null;
                StringBuffer sql = new StringBuffer(256);

                try {
                        s = _dbCon.createStatement();

                        sql.append("call consulta_datos_user(" + id_usuario + ")");

                        rs = s.executeQuery(sql.toString());
                        if (rs.next()) {
                                setId(Integer.parseInt(id_usuario));
                                setNombre(rs.getString("nombre"));
                                setApellido(rs.getString("apellido"));
                                setIdentificacion(rs.getString("cedula"));
                                setDireccion(rs.getString("direccion"));
                                setEmail(rs.getString("e_mail"));
                                setUsuario(rs.getString("usuario"));
                        }
                        rs.close();
                        s.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                        return false;
                }
                return true;
        }

        public ResultSet listarUsuarios(Connection _dbCon){
                Statement s = null;
                ResultSet rs = null;
                StringBuffer sql = new StringBuffer(256);
                try {
                        s = _dbCon.createStatement();

                        sql.append("call listar_usuarios()");

                        rs = s.executeQuery(sql.toString());
                        //s.close();
                } catch (Exception e) {
                        e.printStackTrace();
                }
                return rs;
        }
}
```

Apéndice B: Código de implementación IVR.

```
[globals]
SetGlobalVar(codcli_en=0)
SetGlobalVar(codcli=0)
SetGlobalVar(codcli_re=0)
SetGlobalVar(TIME_OUT=0)
SetGlobalVar(TIME_OUT_MAX=0)

[general]
;
; The "General" category is for certain variables.
;

static=yes                    ; These two lines prevent the command-line interface
writeprotect=yes              ; from overwriting the config file. Leave them here.
language=es

[bogon-calls]
;
; The "bogon-calls" category is for bogus calls
;

;
; Take unknown callers that may have found
; our system, and send them to a re-order tone.
; The string "_." matches any dialed sequence, so all
; calls will result in the Congestion tone application
; being called.  They'll get bored and hang up eventually.
;
exten => _.,1,Congestion
;-----------------------------------------------------------------------------------------------------------------------end

[Asap-payment-menu]
;-----------------------------------------------------
; ejecucion del menu y transferencia a las opciones
exten => s,1,SetMusicOnHold,default
exten => s,2,DigitTimeout,30
exten => s,3,ResponseTimeout,30
exten => s,4,system(echo "Bienvenidos al sistema virtual de transferencia de dinero asappayment")
exten => s,5,system(echo "Escoja la opcion 1 para enviar dinero o la opcion 3 para recibir dinero")
exten => 1000,1, SetVar(opc="0")
exten => 1000,2,Read(opc,,1)
exten => 1000,4,system(echo "Ingrese su número de código")
exten => 1000,5,SetGlobalVar(codcli=0)
exten => 1000,6,Read(codcli,,2)
exten => 1000,7,System(/bin/echo -e "'El código ingresado es ==>  ${codcli}'")
exten => 1000,8,SayNumber(${codcli},f)
exten => 1000,9,system(echo "Si está correcto, presione 1 para continuar o 3 para corregir")
exten => 1000,10,SetVar(opt="0")
exten => 1000,11,Read(opt,,1)
exten => 1000,12,system(echo ${opt})
exten => 1000,13,GotoIf($["${opt}" = "3"]?4:14)
exten => 1000,14,MYSQL(Connect connid localhost dba sql cdr)
exten => 1000,15,MYSQL(Query resultid ${connid} SELECT\ `nombre`\ FROM\ `clientes`\ WHERE\
`id_cliente`=\'${codcli}\')
exten => 1000,16, GotoIf($["${resultid}" = "0"]?17:21)
exten => 1000,17,system(echo "Cliente no está registrado en nuestra base de datos, escoja opcion 1 para volver a
digitar el código del cliente o 3 para salir")
exten => 1000,18,SetVar(opt="0")
exten => 1000,19, Read(opt,,1)
exten => 1000,20,GotoIf($["${opt}" = "3"]?27:4)
exten => 1000,21,MYSQL(Fetch foundRow ${resultid} nombre)
```

```
exten => 1000,22,system(echo "Hola ==>"  ${nombre})
exten => 1000,23,SayAlpha(${nombre})
exten => 1000,24, GotoIf($["${opc}"="3"]?26:25)
exten => 1000,25,Goto(Enviar|100|1)
exten => 1000,26,Goto(Recibir|500|1)
exten => 1000,27,MYSQL(Clear ${resultid})
exten => 1000,28,MYSQL(Disconnect ${connid})
exten => 1000,29, Hangup


[Enviar]

exten => 100,1,DigitTimeout(10)
exten => 100,2,ResponseTimeout(10)
exten => 100,3,system(echo "Ingrese Código de cliente al cual va Enviar dinero")
exten => 100,4,Read(codcli_en,,2)
exten => 100,6,System(/bin/echo -e "'El código ingresado del cliente es ==>  ${codcli_en}'")
exten => 100,5,SayNumber(${codcli_en},f)
exten => 100,7,system(echo "Si está correcto, presione 1 para continuar o 3 para corregir")
exten => 100,8,SetVar(opt="0")
exten => 100,9,Read(opt,,1)
exten => 100,10,system(echo ${opt})
exten => 100,11,GotoIf($["${opt}" = "3"]?1:12)
exten => 100,12,MYSQL(Connect connid localhost dba sql cdr)
exten => 100,13,MYSQL(Query resultid ${connid} SELECT\ `nombre`\ FROM\ `clientes`\ WHERE\
`id_cliente`=\'${codcli_en}\')
exten => 100,14, GotoIf($["${resultid}" = "0"]?15:20)
exten => 100,15,system(echo "Cliente no está registrado en nuestra base de datos, escoja opcion 1 para volver a
digitar el código del cliente o 3 para salir")
exten => 100,16,SetVar(opt="0")
exten => 100,17, Read(opt,,1)
exten => 100,19,GotoIf($["${opt}" = "3"]?32:1)
exten => 100,20,MYSQL(Fetch foundRow ${resultid} nombre)
exten => 100,21,system(echo "El cliente al cual usted va Enviar dinero se llama ==>"  ${nombre})
exten => 100,22,SayAlpha(${nombre})
exten => 100,23,system(echo "Si está correcto, presione 1 para continuar o 3 para corregir")
exten => 100,24,SetVar(opt="0")
exten => 100,25,Read(opt,,1)
exten => 100,26,system(echo ${opt})
exten => 100,27,GotoIf($["${opt}" = "3"]?1:28)
exten => 100,28,MYSQL(Clear ${resultid})
exten => 100,29,MYSQL(Disconnect ${connid})
exten => 100,30,Hangup
exten => 100,31,Goto(IngresarTarjeta|200|1)
exten => 100,32,MYSQL(Clear ${resultid})
exten => 100,33,MYSQL(Disconnect ${connid})
exten => 100,34, Hangup
exten => t,1,SetGlobalVar(TIME_OUT=$[1 + ${TIME_OUT}]);
exten => t,2,GotoIf($[${TIME_OUT} = ${TIME_OUT_MAX}]?3:6)
exten => t,3,SetGlobalVar(TIME_OUT=0)
;exten => t,4,Playback(vm-goodbye)
exten => t,5,Hangup
exten => t,6,Goto(100|1)


[Recibir]
exten => 500,1,DigitTimeout(10)
exten => 500,2,ResponseTimeout(10)
exten => 500,3,MYSQL(Connect connid localhost dba sql cdr)
exten => 500,4,MYSQL(Query resultid ${connid} SELECT\ `UP`\ FROM\ `clientes`\ WHERE\ `id_cliente`=\'${codcli}\')
exten => 500,5,MYSQL(Fetch foundRow ${resultid} UP)
exten => 500,6,GotoIf($["${UP}" = "1"]?7:8)
exten => 500,7,MYSQL(Query resultid ${connid} SELECT\ `saldo`\ FROM\ `tarjetas`\ WHERE\
`id_cliente`=\'${codcli}\')
exten => 500,8,MYSQL(Fetch foundRow ${resultid} saldo)
```

```
[IngresarTarjeta]

exten => 200,1,DigitTimeout(10)
exten => 200,2,ResponseTimeout(10)
exten => 200,3,system(echo "Ingrese Código de su tarjeta de credito")
exten => 200,4,Read(codtarj,,2)
exten => 200,6,System(/bin/echo -e "'El código ingresado de su tarjeta de credito es ==>  ${codtarj}'")
exten => 200,5,SayNumber(${codtarj},f)
exten => 200,7,system(echo "Si está correcto, presione 1 para continuar o 3 para corregir")
exten => 200,8,SetVar(opt="0")
exten => 200,9,Read(opt,,1)
exten => 200,10,system(echo ${opt})
exten => 200,11,GotoIf($["${opt}" = "3"]?1:12)
exten => 200,12,MYSQL(Connect connid localhost dba sql cdr)
exten => 200,13,MYSQL(Query resultid ${connid} SELECT\ `saldo`\ FROM\ `tarjetas`\ WHERE\
`código_tarjeta`=\'${codtarj}\')
exten => 200,14, GotoIf($["${resultid}" = "0"]?15:20)
exten => 200,15,system(echo "Cliente no se encuentra registrado en nuestro sistema , escoja opcion 1 para volver a
digitar el código del cliente o 3 para salir")
exten => 200,16,SetVar(opt="0")
exten => 200,17, Read(opt,,1)
exten => 200,18,GotoIf($["${opt}" = "3"]?32:1)
exten => 200,19,MYSQL(Fetch foundRow ${resultid} saldo)
exten => 200,20,system(echo "El saldo de su cuenta es ==>"  ${saldo})
exten => 200,21,SayNumber(${saldo})
exten => 200,22,System(echo "Ingrese la cantidad de dinero a Enviar")
exten => 200,23,Read(cantdin,,3)
exten => 200,24,System(/bin/echo -e "'La cantidad de dinero a Enviar es ==>  ${cantdin}'")
exten => 200,25,SayNumber(${cantdin},f)
exten => 200,26,system(echo "Si está correcto, presione 1 para continuar o 3 para corregir")
exten => 200,27,SetVar(opt="0")
exten => 200,28,Read(opt,,1)
exten => 200,29,system(echo ${opt})
exten => 200,30,GotoIf($["${opt}" = "3"]?1:31)
exten => 200,31,MYSQL(Query resultid ${connid} UPDATE\ tarjetas\ set\ `saldo`\= \`${saldo}\`-\`${cantdin}\` \
WHERE\ `código_tarjeta`=\'${codtarj}\')
exten => 200,32,MYSQL(Clear ${resultid})
exten => 200,33,MYSQL(Disconnect ${connid})
exten => 200,34, Hangup
exten => t,1,SetGlobalVar(TIME_OUT=$[1 + ${TIME_OUT}])
exten => t,2,GotoIf($[${TIME_OUT} = ${TIME_OUT_MAX}]?3:6)
exten => t,3,SetGlobalVar(TIME_OUT=0)
;exten => t,4,Playback(vm-goodbye)
exten => t,5,Hangup
exten => t,6,Goto(200|1)

[Fin]
exten => 1000,1,MYSQL(Clear ${resultid})
exten => 1000,2,MYSQL(Disconnect ${connid})
exten => 1000,3,Hangup


[Consulta]
exten => 888,1,Background,beep ;Menu de voz ingresar código cliente
exten => 888,2,Read(codcli,,3)
exten => 888,3,MYSQL(Connect connid localhost dba sql cdr)
exten => 888,4,MYSQL(Query resultid ${connid} SELECT\ `nombre`\ FROM\ `clientes`\ WHERE\
`id_cliente`=\'${codcli}\')
exten => 888,5,MYSQL(Fetch foundRow ${resultid} nombre) ; fetch row
exten => 888,6,system(echo "${nombre}" >> /root/prueba.txt)
exten => 888,7,GotoIf($["${foundRow}" = "1"]?10:8) ; leave loop if no row found
exten => 888,8,NoOp(${nombre})
exten => 888,9,Goto(1) ; continue loop if row found
exten => 888,10,MYSQL(Clear ${resultid})
```

```
exten => 888,11,MYSQL(Disconnect ${connid})

[Agregar]
exten => 999,1,Background,beep ;Agregar
exten => 999,2,MYSQL(Connect connid localhost dba sql cdr)
exten => 999,3,MYSQL(Query resultid ${connid} INSERT\ INTO\ clientes\ set\ \(id_cliente=1512\,nombre=\\\"Jhon\
Smith\\\"\,direccion=\\\"Calle\ New\ York\ y\ Bahamas\\\"\,telefono=0115689996\,pais=\\\"USA\\\"\))
exten => 999,4,MYSQL(Clear ${resultid})
exten => 999,5,MYSQL(Disconnect ${connid})

[Actualizar]
exten => 777,1,Background,beep ;Agregar
exten => 777,2,MYSQL(Connect connid localhost dba sql cdr)
exten => 777,3,MYSQL(Query resultid ${connid} UPDATE\ tarjetas\ set\ saldo=156666\ WHERE\ id_tarjeta=3)
exten => 777,4,MYSQL(Clear ${resultid})
exten => 777,5,MYSQL(Disconnect ${connid})

[Eliminar]
exten => 333,1,Background,beep ;Agregar
exten => 333,2,MYSQL(Connect connid localhost dba sql cdr)
exten => 333,3,MYSQL(Query resultid ${connid} DELETE\ FROM\ clientes\ where\ id_cliente=1212)
exten => 333,4,MYSQL(Clear ${resultid})
exten => 333,5,MYSQL(Disconnect ${connid})
```

Apéndice C: Breve glosario de acrónimos VoIP

ATM Asynchronous Transfer Mode (Modo de Transferencia Asíncrona)

CCITT Consultative Committee for International Telegraph and Telephone (Comité Consultivo Internacional de Telefonía y Telegrafía)

CPE Customer Premises Equipment (Equipo en Instalaciones de Cliente)

CTI Computer Telephony Integration (Integración Ordenador- Telefonía)

DiffServ Differentiated Services Internet QoS model (modelo de Calidad de servicio en Internet basado en Servicios Diferenciados)

DNS Domain Name System (Sistema de Nombres de Dominio)

E.164 Recomendación de la ITU-T para la numeración telefónica internacional, eespecialmente para ISDN, BISDN y SMDS.

ENUM Telephone Number Mapping (Integración de Números de Teléfono en DNS)

FDM Frequency Division Multiplexing (Multiplexado por División de Frecuencia)

FoIP Fax over IP (Fax sobre IP)

H.323 Estándar de la ITU-T para voz y videoconferencia interactiva en tiempo real en redes de área local, LAN, e Internet.

IETF Internet Engineering Task Force (Grupo de Trabajo de Ingeniería de Internet)

IGMP Internet Group Management Protocol (Protocolo de Gestión de Grupos en Internet)

IN Intelligent Network (Red Inteligente)

IntServ Integrated Services Internet QoS model (modelo de Calidad de Servicio en Servicios Integrados de Internet)

IP Internet Protocol (Protocolo Internet)

IP Multicast Extensión del Protocolo Internet para dar soporte a comunicaciones multidifusión

IPBX Internet Protocol Private Branch Exchange (Centralita Privada basada en IP)

IPSec IP Security (Protocolo de Seguridad IP)

ISDN Integrated Services Data Network (Red Digital de Servicios Integrados, RDSI)

ISP Internet Service Provider (Proveedor de Servicios Internet, PSI)

ITSP Internet Telephony Service Provider (Proveedor de Servicios de Telefonía Internet, PSTI)

ITU-T International Telecommunications Union - Telecommunications (Unión Internacional de Telecomunicaciones - Telecomunicaciones)

LDP Label Distribution Protocol (Protocolo de Distribución de Etiquetas)

LSR Label Switching Router (Encaminador de Conmutación de Etiquetas)

MBONE Multicast Backbone (Red Troncal de Multidifusión)

MCU Multipoint Control Unit (Unidad de Control Multipunto)

MEGACO Media Gateway Control (Control de Pasarela de Medios)

MGCP Media Gateway Control Protocol (Protocolo de Control de Pasarela de Medios)

MOS Mean Opinion Score (Nota Media de Resultado de Opinión)

MPLS Multiprotocol Label Switching (Conmutación de Etiquetas Multiprotocolo)

OLR Overall Loudness Rating (Índice de Sonoridad Global)

PBX Private Branch Exchange (Centralita Telefónica Privada)

PHB Per Hop Behaviour (Comportamiento por Salto)

PoP Point of Presence (Punto de Presencia)

POTS Plain Old Telephone Service (Servicio Telefónico Tradicional)

PPP Point to Point Protocol (Protocolo Punto a Punto)

PSTN Public Switched Telephone Network (Red de Telefonía Conmutada Pública)

QoS Quality of Service (Calidad de Servicio)

RAS Registration, Authentication and Status (Registro, Autentificación y Estado)

RSVP Reservation Protocol (Protocolo de Reserva)

RTCP Real Time Control Protocol (Protocolo de Control de Tiempo Real)

RTP Real Time Protocol (Protocolo de Tiempo Real)

SAP Session Annunciation Protocol (Protocolo de Anuncio de Sesión)

SCN Switched Circuit Network (Red de Circuitos Conmutados)

SDP Session Description Protocol (Protocolo de Descripción de Sesión)

SIP Session Initiation Protocol (Protocolo de Inicio de Sesión)

SLA Service Level Agreement (Acuerdo de Nivel de Servicio)

SS7 Signalling System Number 7 (Sistemas de Señales número 7)

STMR Side Tone Masking Rating (Índice de Enmascaramiento para el Efecto Local)

TCP Transmission Control Protocol (Protocolo de Control de Transmisión)

TDM Time Division Multiplexing (Multiplexado por División de Tiempo)

TIPHON Telecommunications and Internet Protocol Harmonization Over Networks (Armonización de Protocolos de

Redes de Telecomunicación e Internet)

UDP User Datagram Protocol (Protocolo de Datagramas de Usuario)

UMTS Universal Mobile Telephone System (Sistema Universal de Telecomunicaciones Móviles)

VLAN Virtual Local Area Network (Red de Área Local Virtual)

VPN Virtual Private Network (Red Privada Virtual)

xDSL Cualquiera de las tecnologías de Líneas de SuscripciónDigital (por ejemplo, ADSL)

Apéndice D: Breve glosario de términos técnicos.

circuit switching (conmutación de circuitos). Técnica de comunicación en la que se establece un canal (o circuito dedicado) durante toda la duración de la comunicación. La red de conmutación de circuitos más ubicua es la red telefónica, que asigna recursos de comunicaciones (sean segmentos de cable, «ranuras» de tiempo o frecuencias) dedicados para cada llamada telefónica.

codec (codec). Algoritmo software usado para comprimir/ descomprimir señales de voz o audio. Se caracterizan por varios parámetros como la cantidad de bits, el tamaño de la trama (frame), los retardos de proceso, etc. Algunos ejemplos de codecs típicos son G.711, G.723.1, G.729 o G.726.

extranet (extranet). Red que permite a una empresa compartir información contenida en su Intranet con otras empresas y con sus clientes. Las extranets transmiten información a través de Internet y por ello incorporan mecanismos de seguridad para proteger los datos.

gatekeeper (portero). Entidad de red H.323 que proporciona traducción de direcciones y controla el acceso a la red de los terminales, pasarelas y MCUs H.323. Puede proporcionar otros servicios como la localización de pasarelas.

gateway (pasarela). Dispositivo empleado para conectar redes que usan diferentes protocolos de comunicación de forma que la información puede pasar de una a otra. En VoIP existen dos tipos principales de pasarelas: la Pasarela de Medios (Media Gateways), para la conversión de datos (voz), y la Pasarela de Señalización (Signalling Gateway), para convertir información de señalización.

impairments (defectos). Efectos que degradan la calidad de la voz cuando se transmite a través de una red. Los defectos típicos los causan el ruido, el retardo el eco o la pérdida de paquetes.

intranet (intranet). Red propia de una organización, diseñada y desarrollada siguiendo los protocolos propios de Internet, en particular el protocolo TCP/IP. Puede tratarse de una red aislada, es decir no conectada a Internet.

IP Telephony (Telefonía Internet). Ver «Voice over IP»

jitter (variación de retardo). Es un término que se refiere al nivel de variación de retado que introduce una red. Una red con variación 0 tarda exactamente lo mismo en transferir cada paquete de información, mientras que una red con variación de retardo alta tarda mucho más tiempo en entregar algunos paquetes que en entregar otros. La variación de retardo es importante

cuando se envía audio o video, que deben llegar a intervalos regulares si se quieren evitar desajustes o sonidos inintelegibles.

packet switching (conmutación de paquetes). Técnica de conmutación en la cual los mensajes se dividen en paquetes antes de su envío. A continuación, cada paquete se transmite de forma individual y puede incluso seguir rutas diferentes hasta su destino. Una vez que los paquetes llegan a éste se agrupan para reconstruir el mensaje original.

router (encaminador, enrutador). Dispositivo que distribuye tráfico entre redes. La decisión sobre a donde enviar los datos se realiza en base a información de nivel de red y tablas de direccionamiento. Es el nodo básico de una red IP.

softswitch (conmutación por software). Programa que realiza las funciones de un conmutador telefónico y sustituye a éste al emular muchas de sus funciones de dirigir el tráfico de voz, pero además añade la flexibilidad y las prestaciones propias del tráfico de paquetes.

VoIP, Voice over IP (Voz sobre IP). Método de envío de voz por redes de conmutación de paquetes utilizando TCP/IP, tales como Internet.

BIBLIOGRAFÍA

[Kumar, Corp. And Sengodan, 2001] Vineet Kumar, Markku Corp. And Senthil Sengodan, IP Telephony with H.323; Architectures for Unified Networks and integrated Services, Wilwy Computer Publishing, USA, 2001

[Black, 2002] Uyless Black, Voice over IP Second Edition, Prentice Hall, New Yersey 2002

[khasnabish, 2003] Bumip khasnabish, Implementing Voice over IP, Wiley, 2003

[Ohrtman, 2004] Frank Orthman, Voice over 802,11, Artec House, Noewood, 2004

[Flannang, 2001] Michael E. Flannagan, Adminestiring CISCO QoS in IP Networks, Syngress, 2001

[Flannang, 2002] Michael E. Flannagan, Configuring CISCO QoS in IP Networks, Syngress, 2001

[Cisco Press, 2000] Voice over IP Fundamentals, Cisco Press 2000

Introducción a voz sobre IP

http://www.monografias.com/trabajos2/voip/voip.shtml

Proyecto Open H.323

http://www.openh323.org/

Recursos VoIP

http://www.recursosvoip.com/b2/noticias.php?m=200201

Que es SIP?

http://www.tsares.net/VoIP/FAQ_VoIP.htm#queessip

Multimedia sobre IP

http://www.rediris.es/mmedia/

VoIP terminará siendo tan natural cmo el correo electrónico

http://www.el-mundo.es/navegante/2004/05/14/entrevistas/1084522811.html

Resumen de los mecanismos de Qos y como interoperan

http://www.microsoft.com/latam/technet/articulos/windows2k/qosmech/

Información sobre Codecs

http://www.ilbcfreeware.org/

VoIP GSM Gateway

http://voip-info.org/wiki-GSM+Codec

Speex, el código libre de compresión de voz

http://www.speex.org/

Introducción a los Codecs

http://compare.ozvoip.com/codecs.php

Información sobre Digium, el sponsor de Asterisk

http://www.linux-support.net

Página principal de Asterisk

http://www.asterisk.org