ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

INFORME DE MATERIA DE GRADUACIÓN

"SISTEMAS VOIP USANDO SOFTWARE LIBRE"

"Implementación de un Sistema Callback usando archivos .call"

Previa a la obtención del Título de:

INGENIERO EN TELEMÁTICA

Presentada por:

MERCY DENISSE ANCHUNDIA RUIZ

ALBERTO JAVIER SANTOS FLORES

GUAYAQUIL – ECUADOR

2010

A G R A D E C I M I E N T O S

Mercy Anchundia Ruiz

Agradezco a mi Dios todopoderoso que me ha permitido tener esta maravillosa oportunidad la ayuda incondicional y constante de mis padres: Luis Anchundia Ormeño y Mariana Ruiz Caiche. Además me siento agradecida con todas aquellas personas: familiares, profesores y amigos que han contribuido a mi desarrollo personal y profesional.

Alberto Santos Flores

Agradezco a Dios por haberme dado esta oportunidad de poder realizarme como profesional. A mis padres: Alberto Santos Joniaux y Maria Esther Flores Valdivieso que sin su apoyo no habría podido llegar hasta este punto. A mi abuelita Ligia Valdivieso Valdivieso y a mi abuelo Luis Eduardo Flores Saenz, que en paz descanse, por haberme recibido en su hogar. Y a mis profesores por el conocimiento que impartieron en mí.

D E D I C A T O R I A

Mercy Anchundia Ruiz

Dedico este trabajo a mis seres queridos, a la ESPOL, a la FIEC por todo el apoyo brindado durante mi carrera profesional.

Alberto Santos Flores

Dedico este trabajo a mis padres, a mi hermana, y mis abuelos y a todas las personas que me han apoyado.

D E C L A R A C I Ó N   E X P R E S A

“La responsabilidad por los hechos, ideas y doctrinas expuestas en esta tesis, nos corresponden exclusivamente; y, el patrimonio intelectual de la misma, a la Escuela Superior Politécnica del Litoral”

(Reglamento de Graduación de la ESPOL)

Mercy Denisse Anchundia Ruiz

Alberto Javier Santos Flores

T R I B U N A L   D E   S U S T E N T A C I O N

Ing. Gabriel Astudillo Brocel

PROFESOR DIRECTOR DE LA MATERIA DE GRADUACIÓN

Ing. Patricia Chávez Burbano

PROFESOR DELEGADO POR EL DECANO DE LA FACULTAD

R E S U M E N

Hoy en día el uso de una central privada de intercambio (PBX) es fundamental para cualquier tipo de organización y en este ambiente suele suceder que cuando se llama por teléfono a un usuario, cuya línea se encuentra ocupada, existe la necesidad de insistir en tratar de llevar a cabo la comunicación, intentando nuevamente la llamada después de un tiempo prudencial de espera, pero muchas veces no se dispone del tiempo o por alguna razón se puede olvidar reintentar la llamada, que podría ser muy importante para el llamante. De ahí que el objetivo de este proyecto es demostrar que con Asterisk se puede dotar de funcionalidades tan específicas como la descrita en este proyecto y que permiten competir con sistemas de telefonía propietarios, de manera que ésta es una de las diversas aplicaciones que se pueden implementar como solución al problema descrito anteriormente y muy útil para usuarios que hoy en día usan la telefonía ip como su medio de comunicación a través de un servicio de código abierto como lo es Asterisk.

En el primer capítulo se detallan los antecedentes sobre los que se plantearon los objetivos y la justificación para la realización del presente trabajo. En el capítulo segundo, los conceptos esenciales que se necesitaron conocer para la implementación del proyecto. En el tercer capítulo se detallan las especificaciones técnicas y se describe detalladamente el desarrollo del proyecto. Finalmente, en el capítulo cuarto se especifica cómo hacer funcionar el proyecto y las pruebas efectuadas al sistema propuesto.

I N D I C E   G E N E R A L

INDICE DE GRÁFICOS

INTRODUCCION

El Proyecto trata de un sistema de callback o retrollamada en ocupado usando archivos .call, los mismos que se emplean para generar llamadas salientes en la PBX de código abierto: Asterisk. Para su implementación se han empleado algunas características propias de Asterisk a fin de que este sistema de retrollamada funcione entre peers sip e iax, que son las tecnologías para canales de comunicaciones de voz sobre ip ampliamente requeridas; además de evitar más de una llamada entrante en los teléfonos mediante un mecanismo creado con grupos y categorías en Asterisk, el cual se detalla posteriormente. El callback implementado también funciona para llamadas externas a la organización, empleando para ello una tarjeta analógica Digium para interconexión con la red de telefonía pública.

Los dispositivos finales son los softphones, como zoiper y x-lite, los teléfonos ip Grandstream y cualquier tipo de teléfono convencional desde el cual se realice la llamada a alguna extensión dentro de la organización.

C A P I T U L O 1

ANTECEDENTES Y JUSTIFICACIÓN

1.1 Antecedentes

En el mundo actual, las telecomunicaciones representan un factor muy importante entre personas y organizaciones, razón por la cual es fundamental contar con un buen sistema de este tipo, en el cual se refleje la eficiencia y eficacia de las mismas. Anteriormente, acceder a este servicio con diversas funcionalidades era privilegio de unos pocos, pero hoy en día se ha difundido tanto que es imprescindible. La mayoría de las veces, estos servicios suelen ser costosos y son brindados por empresas específicas públicas o privadas, pero gracias a la tecnología de código abierto y al desarrollo del internet se puede acceder al mundo de las telecomunicaciones mediante telefonía de voz sobre ip, todo esto a un bajo costo con una gama de opciones que no nos ofrece la telefonía tradicional. De manera que un revolucionario programa de código abierto llamado Asterisk, desarrollado por la compañía Digium, es el precursor de lo que hoy en día se conoce como la telefonía de voz sobre ip. Asterisk es conocido como una PBX de código abierto, en la cual es posible implementar características similares a las que nos brinda la telefonía tradicional y agregar más funcionalidades, que en ésta última requieren costos adicionales para ser establecidas o simplemente no existen. Una de estas funcionalidades adicionales es el sistema de retrollamada que se expondrá a lo largo de este documento, el cual ha sido programado gracias a este software de código abierto, lo que nos ha permitido también ajustarlo a las necesidades reales de las empresas.

1.2 Descripción del Proyecto

El callback o retrollamada se ejecuta según el usuario llamante a la línea ocupada la active mediante el número 6 del teclado de su teléfono, para ello se le notifica sobre esta opción en caso de que la extensión solicitada esté ocupada. Si el usuario activó la retrollamada, se le indica que debe esperar unos segundos para programarla y a continuación se le notifica que puede colgar. De manera que el callback realiza la llamada a la extensión solicitada, que está inicialmente ocupada, con un tiempo de espera programado para empezar a ejecutarla, con un número máximo de intentos y un tiempo determinado entre cada intento; estos valores son establecidos automáticamente por este sistema y serán expuestos más adelante. Cuando el usuario llamado conteste al callback, inmediatamente será rellamado el usuario llamante para establecer la comunicación entre ambos.

Este sistema callback está diseñado para que funcione con extensiones del tipo sip, iax dentro de la empresa y con las líneas de la telefonía convencional. El funcionamiento del proyecto se puede resumir en el diagrama de la figura 1.1 que se muestra a continuación de los objetivos.

El presente sistema callback implementado pretende alcanzar los siguientes objetivos:

1.2.1 Objetivos Generales

- Ofrecer una alternativa al usuario, sin importar su ubicación geográfica, para la comunicación inmediata con una extensión ocupada de la PBX de la empresa en cuanto esté disponible.
- Facilitar y extender el área de comunicación de la empresa u organización.

1.2.2 Objetivos Específicos

- Mostrar la aplicación de un sistema de retrollamada en Asterisk mediante el uso de archivos .call.
- Implementar un procedimiento para el retorno de llamada tipo Call-Through (Realizar llamadas) con Asterisk.
- Establecer un canal de comunicación que permita una conexión con la PBX de la organización desde fuera de ésta.
- Aplicar el uso de las tecnologías de voz sobre ip más comunes para obtener un proyecto genérico, útil para cualquier tipo de organización.

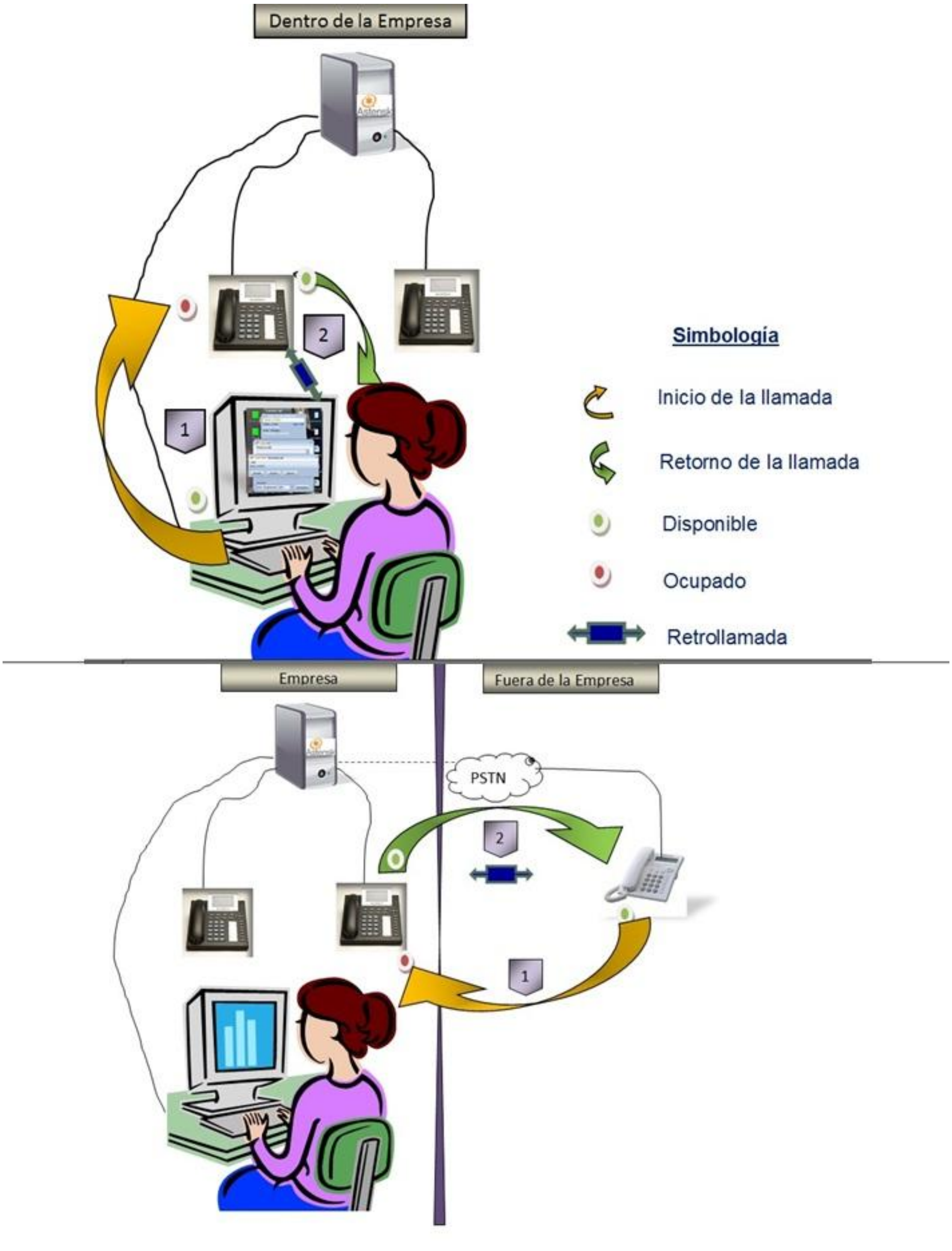

Figura 1.1 Diagrama de Funcionamiento del proyecto

1.3 Justificación

En diversas ocasiones, cuando un usuario necesita comunicarse con alguna extensión de la PBX de su empresa, y ésta se encuentra ocupada, surge la necesidad de insistir en la llamada o intentarlo nuevamente después de un tiempo prudencial, en el cual se corre el riesgo, incluso de olvidarlo. Pero esa llamada podría ser muy importante, pudiendo causar pérdidas para la organización. Este es un problema cotidiano, no sólo en las empresas sino a nivel personal incluso, razón por la cual, este proyecto sería una solución tecnológica de bajo costo, eficaz y eficiente; puesto que para su desarrollo se ha empleado tecnología de código abierto y uso del acceso a internet.

Este sistema de retrollamada no sólo resuelve el problema anteriormente expuesto a nivel interno de la empresa, sino también para los usuarios externos a la misma.

1.4 Metodología

Para la implementación del sistema de retrollamada se han empleado los siguientes componentes:

1. Instalación del servicio Asterisk en una máquina con el sistema operativo Centos 5. Esta máquina será nuestro servidor PBX.
2. Instalación del servicio de base de datos mysql.
3. Instalación del compilador para el lenguaje de programación php 5.
4. Instalación de la Tarjeta Digium Tdm410p Pci con 4 Puertos fxs/fxo en el servidor.
5. Conexión y configuración de teléfonos IP marca Grandstream con usuarios sip.
6. Instalación y configuración del softphone zoiper con un usuario iax.

1.5 Perfil de la tesis

En el capítulo 2 se abarcan las diferentes generalidades de los componentes que se emplearon para implementación del proyecto, entre ellos se encuentran: Asterisk, protocolos SIP,IAX, el uso de macros, grupos y categorías, archivos .call, agi, base de datos de Asterisk y los teléfonos VoIP Grandstream.

En el capítulo 3 se describen las especificaciones técnicas de este sistema paso a paso a fin de que sea fácilmente implementado.

En el capítulo 4 se detallan las pruebas realizadas, con imágenes, para mostrar el funcionamiento del proyecto.

C A P I T U L O 2

FUNDAMENTOS TEÓRICOS

2.1 Generalidades de una retrollamada

Una retrollamada es exactamente el retorno de una llamada a donde se originó, también conocida como callback, el propósito de la misma es el ahorro de tiempo y dinero cuando se requiere de la comunicación con alguien, que por diversos motivos no se encuentra disponible, ya sea porque el usuario se encuentra usando la línea o porque está ausente. De manera que este sistema da la opción al llamante de que se le devuelva la llamada en cuanto el usuario se encuentre disponible, así por ejemplo, en el caso de que éste se encuentre usando la línea y por lo tanto está ocupada y apenas la desocupe el sistema se encarga de llamar a este usuario y de ahí al llamante para permitir la comunicación entre ambos. En este sentido, este sistema de retrollamada implementado en Asterisk es de gran utilidad a nivel personal y organizacional.

2.2 Asterisk

Figura 2.1 Logotipo de Asterisk¡Error! Marcador no definido.

Asterisk, cuyo logotipo se muestra en la figura 2.1, es un programa de software libre (bajo licencia GPL) que proporciona funcionalidades de una central telefónica (PBX). Como cualquier PBX, se puede conectar un número determinado de teléfonos para hacer llamadas entre sí e incluso conectar a un proveedor de VoIP o bien a una RDSI tanto básicos como primarios. Mark Spencer, de Digium, inicialmente creó Asterisk y actualmente es su principal desarrollador, junto con otros programadores que han contribuido a corregir errores y añadir novedades y funcionalidades. Originalmente desarrollado para el sistema operativo GNU/Linux, Asterisk actualmente también se distribuye en versiones para los sistemas operativos BSD, MacOSX, Solaris y Microsoft Windows, aunque la plataforma nativa (GNU/Linux) es la que cuenta con mejor soporte de todas.

Asterisk incluye muchas características anteriormente sólo disponibles en costosos sistemas propietarios PBX como buzón de voz, conferencias, IVR, distribución automática de llamadas, y otras muchas más. Los usuarios pueden crear nuevas funcionalidades escribiendo un plan de llamadas en el lenguaje de script de Asterisk o añadiendo módulos escritos en lenguaje C o en cualquier otro lenguaje de programación reconocido por Linux.

Para conectar teléfonos estándar analógicos son necesarias tarjetas electrónicas telefónicas FXS o FXO fabricadas por Digium u otros proveedores, ya que para conectar el servidor a una línea externa no basta con un simple módem. Quizá lo más interesante de Asterisk es que reconoce muchos protocolos VoIP como pueden ser SIP, H.323, IAX y MGCP. Asterisk puede interoperar con terminales IP actuando como un registrador y como gateway entre ambos.

La versión estable de Asterisk está compuesta por los módulos siguientes:

- Asterisk: Ficheros base del proyecto.
- DAHDI: Soporte para hardware. Drivers de tarjetas. (Anteriormente ZAPTEL)
- Addons: Complementos y añadidos del paquete Asterisk. Opcional.
- Libpri: Soporte para conexiones digitales. Opcional.
- Sounds: Aporta sonidos y frases en diferentes idiomas. (Incluidos en el paquete Asterisk)

Cada módulo cuenta con una versión estable y una versión de desarrollo. La forma de identificar las versiones se realiza mediante la utilización de tres números separados por un punto. Teniendo desde el inicio como primer número el uno, el segundo número indica la versión, mientras que el tercero muestra la revisión liberada. En las revisiones se llevan a cabo correcciones, pero no se incluyen nuevas funcionalidades. En las versiones de desarrollo el tercer valor siempre es un cero, seguido de la palabra "beta" y un número, para indicar la revisión (1). Actualmente este software se encuentra en su versión 1.6; sin embargo, por su estabilidad, en el proyecto usamos la versión 1.4.

2.2.1 Funcionalidades De Asterisk

Las principales funcionalidades de Asterisk son:

- Enrutamiento de llamadas según tarifa.
- Enrutamiento de llamadas según tráfico.
- Sistema de colas de distribución de llamadas.
- Plan DDI.
- Recepción y envío de fax.
- Música en espera.
- Transferencia de llamadas ciega y asistida.
- Transferencia de llamadas a extensiones locales y remotas.
- Operadora Virtual para atención automática de las llamadas.
- Mensaje de bienvenida.
- Programación por horario.
- Interface guía de usuario para su gestión.
- Buzones de voz con funcionalidad VOICE MAIL accesible desde teléfono o cuenta de correo.

- Identificación del nombre y número del llamante en llamadas internas.
- Identificación del número en llamadas externas e internas.
- Configuración de los nombres de los dueños de las extensiones, para que aparezca cuando se llaman a otras.
- Mensaje de apertura y cierre de llamadas.
- Auto-aprovisionamiento de los teléfonos automáticamente.
- Multiconferencia entre 4 y 5 personas.
- Listas negras tanto con llamadas salientes como entrantes.
- Posibilidad de visualizar las estadísticas CDR de todas las llamadas.
- Multiprotocolo, le permite tener operadores IP y tradicionales (2).

2.2.2 Compatibilidad

Los requisitos mínimos del sistema son los siguientes:

- Procesador Dual Intel Xeon 1.8 Ghz
- 256 Mb Ram
- 100Mb de disco
- Sistema operativo:
  - GNU/Linux 2.x
  - MacOSX 10.x
  - Open Solaris
  - BSD

- MS Windows

2.3 Protocolo SIP

SIP (Session Initiation Protocol) es un protocolo de señalización para conferencia, telefonía, presencia, notificación de eventos y mensajería instantánea a través de Internet. Fue desarrollado incialmente en el grupo de trabajo IETF MMUSIC (Multiparty Multimedia Session Control) y, a partir de Septiembre de 1999, pasó al grupo de trabajo IETF SIP (3).

La sintaxis de sus operaciones se asemeja a las de HTTP y SMTP, los protocolos utilizados en los servicios de páginas Web y de distribución de e-mails respectivamente. Esta similitud es natural ya que SIP fue diseñado para que la telefonía se vuelva un servicio más en la Internet.

En Noviembre del año 2000, SIP fue aceptado como el protocolo de señalización de 3GPP y elemento permanente de la arquitectura IMS (IP Multimedia Subsystem). SIP es uno de los protocolos de señalización para voz sobre IP. Este protocolo usa el puerto UDP 5060 y su documentación se encuentra en la RFC 2543.

Funcionamiento

El protocolo SIP permite el establecimiento de sesiones multimedia entre dos o más usuarios. Para hacerlo se vale del intercambio de mensajes entre las partes que quieren comunicarse.

Componentes del protocolo:

- Agentes de Usuario
- Servidores de Registro o Registrar
- Servidores Proxy y de Redirección

Agentes de Usuario

Los usuarios, que pueden ser seres humanos o aplicaciones de software, utilizan para establecer sesiones lo que el protocolo SIP denomina "Agentes de usuario". Estos son puntos extremos del protocolo, es decir son los que emiten y consumen los mensajes del protocolo SIP. Un videoteléfono, un teléfono, un cliente de software (softphone) y cualquier otro dispositivo similar es para el protocolo SIP un agente de usuario. Los agentes de usuario se comportan como clientes (UAC: User Agent Clients) y como servidores (UAS: User Agent Servers). Son UAC cuando realizan una petición y son UAS cuando la reciben. Por esto los agentes de usuario deben implementar un UAC y un UAS.

Servidores de Registro o Registrar

El protocolo SIP permite establecer la ubicación física de un usuario determinado, esto es en qué punto de la red está conectado. Para ello se vale del mecanismo de registro. Este mecanismo funciona como sigue:

Cada usuario tiene una dirección lógica que es invariable respecto de la ubicación física del usuario. Una dirección lógica del protocolo SIP es de la forma usuario@dominio es decir tiene la misma forma que una dirección de correo electrónico. La dirección física (denominada "dirección de contacto") es dependiente del lugar en donde el usuario está conectado (de su dirección IP). Cuando un usuario inicializa su terminal (por ejemplo conectando su teléfono o abriendo su software de telefonía SIP) el agente de usuario SIP que reside en dicho terminal envía una petición con el método REGISTER a un Servidor de Registro, informando a qué dirección física debe asociarse la dirección lógica del usuario. El servidor de registro realiza entonces dicha asociación (denominada binding). Esta asociación tiene un período de vigencia y si no es renovada, caduca. También puede terminarse mediante un desregistro.

Servidores Proxy y de Redirección

Para encaminar un mensaje entre un agente de usuario cliente y un agente de usuario servidor normalmente se recurre a los servidores. Estos servidores pueden actuar de dos maneras:

Como Proxy, encaminando el mensaje hacia destino,

Como Redirector (Redirect), generando una respuesta que indica al origen la dirección del destino o de otro servidor que lo acerque al destino.

La principal diferencia es que el servidor proxy queda formando parte del camino entre el UAC y el (o los) UAS, mientras que el servidor de redirección una vez que indica al UAC cómo encaminar el mensaje ya no interviene más. Un mismo servidor puede actuar como Redirector o como Proxy dependiendo de la situación (4).

2.4 Protocolo IAX

IAX (Inter-Asterisk eXchange) es uno de los protocolos utilizado por Asterisk, creado por Mark Spencer, para el control de llamadas de voz sobre ip, documentado en la RFC 5456, el tráfico usa el puerto udp 4569. Inicialmente fue desarrollado para la PBX Asterisk y originalmente se mantuvo para intercambio de tráfico entre servidores Asterisk. Hoy en día este protocolo es usado en diversas plataformas de voz sobre ip. IAX fue diseñado para reemplazar tempranamente a los protocolos de control de llamada: H.323 and SIP. IAX emplea de manera eficiente el ancho de banda comparado a otros protocolos de su tipo, habilitándolo para soportar un mayor número de llamadas concurrentes de voz sobre ip, sobre la misma cantidad de ancho de banda, soporta autenticación mediante claves públicas con RSA con el algoritmo de message diggest SHA-1 para firmas digitales(5). El protocolo IAX ahora se refiere generalmente al IAX2, la segunda versión del protocolo IAX. El protocolo original ha quedado obsoleto en favor de IAX2.

Propiedades Básicas

IAX2 es robusto, lleno de novedades y muy simple en comparación con otros protocolos. Permite manejar una gran cantidad de códecs y un gran número de transmisiones conocidas como streams, lo que significa que puede ser utilizado para transportar virtualmente cualquier tipo de dato. Esta capacidad lo hace muy útil para realizar videoconferencias o realizar presentaciones remotas; utiliza un único puerto UDP, generalmente el 4569, para comunicaciones entre puntos finales (terminales VoIP) para señalización y datos. El tráfico de voz es transmitido dentro de banda (in-band), lo que hace a IAX2 un protocolo casi transparente a los cortafuegos y realmente eficaz para trabajar dentro de redes internas. En esto se diferencia de SIP, que utiliza una cadena RTP fuera de banda (out-of-band) para entregar la información.

IAX2 soporta Trunking (red), donde un simple enlace permite enviar datos y señalización por múltiples canales. Los componentes de la señalización iax son muy parecidos a los del protocolo SIP.

Objetivos de IAX

El principal objetivo de IAX ha sido minimizar el ancho de banda utilizado en la transmisión de voz y vídeo a través de la red IP, con particular atención al control y a las llamadas de voz y proveyendo un soporte nativo para ser transparente a la traducción de direcciones de red (NAT). La estructura básica de IAX se fundamenta en la multiplexación de la señalización y del flujo de datos sobre un simple puerto UDP entre dos sistemas. El ancho de banda para algunas aplicaciones se sacrifica en favor del ancho de banda para VoIP (6).

2.5 Interfaz de puerta de enlace de Asterisk

La interfaz de puerta de enlace de Asterisk (AGI) es una interfaz para añadir funcionalidad a Asterisk con diferentes lenguajes de programación como: Perl, PHP, C, Pascal, Bourne Shell.

AGI puede controlar el plan de marcado, llamado en extensions.conf.

A continuación se mencionan sus variantes:

Async AGI.- introducido en Asterisk 1.6, permite un asincrónico scripting de AGI.

EAGI.- brinda a la aplicación la posibilidad de acceder y controlar el canal de sonido además de la interacción con el plan de marcado.

FastAGI.- puede ser usado para hacer el procesamiento en una máquina remota vía conexión a la red.

DeadAGI.- brinda acceso a canales deshabilitados después de ser colgados, esta variante está descartada a partir de Asterisk 1.6.

Ambiente de ejecución

Cuando Asterisk empieza un script AGI, este alimenta las variables de canal para el script en entrada estándar. Los nombres de las variables tienen el prefijo “agi_” y están separadas de sus valores por dos puntos y un espacio. Aunque las variables de canal deben estar en mayúsculas, los nombre pasados a un script agi deben estar en minúsculas. Un script agi puede retornar cualquiera de los siguientes estados en la variable "AGISTATUS": SUCCESS, FAILURE, HANGUP.

Las variables pasadas por Asterisk son:

agi_request – El nombre de archivo del script.

agi_channel – El canal de origen (su teléfono).

agi_language – El lenguaje del código (e.j. "en") .

agi_type – El tipo de canal de origen (e.j. "SIP" or “IAX2").

agi_uniqueid - El ID único para la llamada.

agi_version - La version de Asterisk (desd Asterisk 1.6).

agi_callerid - El número del caller ID (o "unknown").

agi_calleridname – El nombre del caller ID (o "unknown").

agi_callingpres – La presentación para el callerid en un canal ZAP .

agi_callingani2 – El número que está definido en ANI2. Revisar en la lista detallada de variables (sólo para canales PRI).

agi_callington – El tipo de número usado en canales PRI. Revisar en la lista detallada de variables.

agi_callingtns – un número de 4 digitos opcional(Selector de Tránsito de Red) usado en canales PRI . Revisar en la lista detallada de variables.

agi_dnid – El id del número marcado (o "unknown") .

agi_rdnis – El número referente a DNIS (o "unknown") .

agi_context – contexto origen en extensions.conf.

agi_extension – El número llamado.

agi_priority – La prioridad desde la cual fue ejecutado en el plan de marcado.

agi_enhanced – El valor de la bandera es 1.0 si empezó como un script EAGI , 0.0 de otro modo.

agi_accountcode – Código de la cuenta del canal de origen.

agi_threadid – ID del hilo del script agi (desde Asterisk 1.6)

Un script agi también puede pasar un número de variables de ambiente, que apunta a los directorios del sistema de archivos que contiene los archivos de Asterisk, estas son (7):

AST_CONFIG_DIR	AST_DATA_DIR
AST_CONFIG_FILE	AST_LOG_DIR
AST_MODULE_DIR	AST_AGI_DIR
AST_SPOOL_DIR	AST_KEY_DIR
AST_MONITOR_DIR	AST_RUN_DIR
AST_VAR_DIR	

Existen una gran cantidad de comandos para poder acceder, modificar variables de Asterisk y entre otras funcionalidades muy útiles cuando se emplea agi, para lo cual es recomendable revisar la documentación respectiva.

2.6 Archivos Call

Como su nombre lo indica, son archivos cuya extensión es call y los cuales Asterisk procesa para generar llamadas salientes según los parámetros indicados en el mismo archivo, indicando básicamente:

1. Cómo realizar la llamada, similar a la aplicación Dial().
2. Qué hacer cuando la llamada sea respondida.

Con los archivos call se puede especificar toda esta información simplemente creando el archivo con la sintaxis requerida y colocarlo en el directorio /var/spool/Asterisk/outgoing/ (esta ruta se puede modificar en el archivo Asterisk.conf).

El módulo PBX_spool examina cada segundo el contenido de dicho directorio, creando una nueva llamada para cada archivo call que encuentre. No se debe escribir o crear este archivo en la ruta antes mencionada o Asterisk la leerá parcialmente; por esto, el archivo debe crearse en otra ruta y luego moverlo a la que realmente permitirá su ejecución.

<u>Sintaxis</u>

El archivo call consiste de pares <clave>:<valor>, uno por línea.

Los comentarios se indican mediante el carácter '**#**' que debe antecederles; los comentarios también pueden ser indicados con '**;**'. Pero no se permiten comentarios multilínea de la forma: (;-- --;) usado en los archivos de configuración de Asterisk. Los ';' pueden ser escapados con el símbolo: '\'.

Los siguientes pares claves-valor son usados para configurar la llamada:

Channel: <channel>

El canal sobre el cual se debe realizar la llamada, usa la misma sintaxis que el Dial().

Callerid: <callerid>

El caller ID a ser usado para la llamada.

WaitTime: <number>

Número de segundos que el sistema debe esperar para que la llamada sea respondida. Si no se especifica, por default es de 45 segundos.

MaxRetries: <number>

Máximo número de reintentos de marcado (si un intento falla porque el dispositivo está ocupado o es inalcanzable). Si no está especificado, el valor por default es 0 y sólo un intento será realizado.

RetryTime: <number>

Número de segundos para esperar antes de realizar el siguiente intento. Si no se especifica, el valor por default es de 300 segundos.

Account: <account>

El código de la cuenta para el CDR.

Context: <context>

El contexto destino.

Extension: <exten>

La extension destino, en el que la ejecución del plan de marcada o empieza si el dispositivo es respondido.

Priority: <priority>

La prioridad destino. Si no se especifica, el valor por default es1.

Setvar: <var=value>

Sirve para setear una variable

Archive: <yes|no>

De manera predeterminada, los archivos call son eliminados inmediatamente después de su ejecución. Si Archive:yes está configurado, ellos son copiados al directorio: /var/spool/Asterisk/outgoing_done/ y Asterisk añade una línea al archivo que describe el resultado:

Status: <Expired|Completed|Failed>

Application: <appname>

La aplicación a utilizar, una vez que la llamada sea respondida.

Data: <args>

Opciones o argumentos de la aplicación (9).

2.7 Macros

Una macro es una clase de subrutina; esta puede contener complejos flujos de trabajos pero es llamada a través de una única entrada. Este reduce la repetición en el plan de marcado y hace que sea más impecable y más pequeño.

Sintaxis

Macro(name[,args(arg1[,arg2[,...]])])

Se ejecuta una macro usando como nombre contexto: macro-*nombre* , saltando a la extensión **s** de este contexto y ejecutando cada paso, entonces retorna al plan de marcado cuando los pasos hayan sido finalizados.

La extensión llamante, contexto y prioridad son almacenados en MACRO_EXTEN, MACRO_CONTEXT y MACRO_PRIORITY respectivamente. Los argumentos enviados se convierten en ARG1, ARG2,etc en el contexto macro.

Si MACRO_OFFSET es configurado en la terminación, el macro intentará continuar a la prioridad MACRO_OFFSET +N+1 si la misma existe y N+1 de otra manera (10).

2.8 Base de datos de Asterisk: AstDB

La base de datos de Asterisk, también conocida como AstDB, es un mecanismo muy poderoso que provee este software para el almacenamiento de valores. Esta base de datos brinda una manera simple de almacenar los datos para su uso en el plan de marcado.

La AstDB almacena los datos en grupos *familias*, con valores identificados por *claves*. Dentro de una familia, una clave puede ser usada una sola vez. Por ejemplo si se tiene la familia llamada test, se podría almacenar sólo un valor con la clave llamada count. Cada valor almacenado debe estar asociado con una familia.

Para **almacenar** datos en la AstDB se usa la aplicación Set(). Así por ejemplo para asignar la clave count a la familia test con el valor de 1:
exten => 456,1,Set(DB(test/count)=1); si la clave ya existe, ésta será sobrescrita. También es equivalente aplicar el comando en la CLI>database put *family* key *value*

Para **recuperar** datos de la AstDB y asignarlo a una variable, se usa la aplicación Set() nuevamente. Así por ejemplo para obtener el valor de count de la familia test y se le asigna a la variable COUNT:
exten => 456,1,Set(COUNT=${DB(test/count)})

Para **eliminar** datos de la AstDB, en el caso de que se desee eliminar una clave se usa la aplicación DB_DELETE(). Para eliminar una familia de claves entera se usa DBdeltree() (11).
exten => 457,1,Verbose(0, The value was ${DB_DELETE(test/count)})
exten => 457,1,DBdeltree(test)

2.9 Grupos y Categorías

Existen varios conceptos sobre grupos en Asterisk pero todos están relacionados. Existen:

- Grupos de Canal.- como su nombre lo indica, son usados para agrupar canales que están teniendo comunicación. Las principales operaciones son:

  -Hacer que el canal actual se una a un grupo dado (con identificador alfanumérico)

  -Contar la cantidad de canales que actualmente pertenecen al grupo dado.

Un canal puede pertenecer sólo a un grupo por categoría. Por lo tanto, si uno quiere hacer que un canal pertenezca a ambos grupos: Zap y SIP, uno tendría que usarse como Zap@in y SIP@out, creando las categorías in y out. Usando categorías se pueden establecer múltiples grupos en un solo canal activo; por ejemplo, permitirnos establecer la cantidad de llamadas en el canal destino y en el canal llamante.

Aquí las funciones para trabajar con este tipo de grupos a partir de Asterisk 1.2 en adelante:

GROUP() (replaces both SetGroup() and CheckGroup())

GROUP_COUNT()

GROUP_MATCH_COUNT()

GROUP_LIST()

- Grupos de llamadas y pickup.- esta clase de grupos es configurado para permitir capturar remotamente un tono de ring telefónico a través de*8(#) (por default, está denegado). El grupo de llamadas es a lo que una extensión pertenece, el grupo de pickup o captura es el cual los grupos de llamadas, la extensión puede remotamente capturar. Este mecanismo es desafortunadamente local a una tecnología/driver aunque parches pueden levantar esta limitación.
- Grupos troncales.- agrupando puertos zapata fxo dentro de grupos troncales que permiten la automática selección de un puerto deshabilitado para llamadas salientes.
- Grupos de Agentes.-Esta característica permite administrar agentes a través de grupos más que individualmente (12).

2.10 Buzón de voz

El buzón de voz de Asterisk es configurado en el archivo voicemail.conf, que se encuentra en la ruta /etc/Asterisk, permite al usuario grabar un mensaje y guardarlo en un archivo de audio dado el número de buzón. Para invocar el uso del buzón de voz en el plan de marcado se usa la aplicación

VoiceMail(*boxnumber[@context][&boxnumber2[@context]][&boxnumber*

3],[*flags*], y puede tener cualquiera de los siguientes parámetros como flags:

s.- reproduce una voz predeterminada que dice “Por favor deje su mensaje después del tono cuando esté listo cuelgue o presione la tecla numeral”.

u.-reproduce una voz predeterminada indicando que la extensión marcada no se encuentra disponible.

b.- reproduce la voz predeterminada indicando que la extensión marcada se encuentra ocupada.

Para acceder al contenido del buzón de voz y escuchar los mensajes, se puede acceder mediante la aplicación VoicemailMain en el plan de marcado (13).

2.11 FXS y FXO

FXS y FXO son los nombres de los puertos usados por las líneas telefónicas analógicas (también denominados POTS - Servicio Telefónico Básico y Antiguo). FXS es la interfaz de abonado externo es el puerto que efectivamente envía la línea analógica al abonado. En otras palabras, es el “enchufe de la pared” que envía tono de marcado, corriente para la batería y tensión de llamada. FXO es la Interfaz de central externa es el puerto que recibe la línea analógica. Es un enchufe del teléfono o aparato de fax, o el enchufe de su centralita telefónica analógica. Envía una indicación de colgado/descolgado (cierre de bucle). Como el puerto FXO está adjunto a un dispositivo, tal como un fax o teléfono, el dispositivo a menudo se denomina “dispositivo FXO”.

FXO y FXS son siempre pares, es decir, similar a un enchufe macho/hembra. Sin una centralita, como se muestra en la figura 2.2, el teléfono se conecta directamente al puerto FXS que brinda la empresa telefónica.

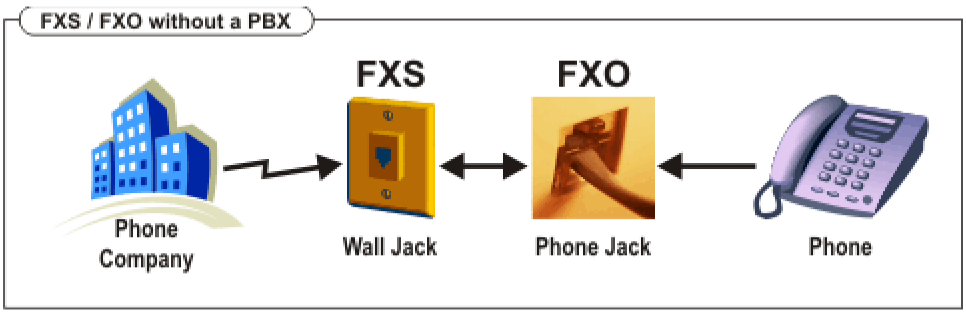

Figura 2.2 FXS/FXO sin centralita

Si tiene centralita, como se muestra en la figura 2.3, debe conectar las líneas que suministra la empresa telefónica a la centralita y luego los teléfonos a la centralita. Por lo tanto, la centralita debe tener puertos FXO (para conectarse a los puertos FXS que suministra la empresa telefónica) y puertos FXS (para conectar los dispositivos de teléfono o fax)

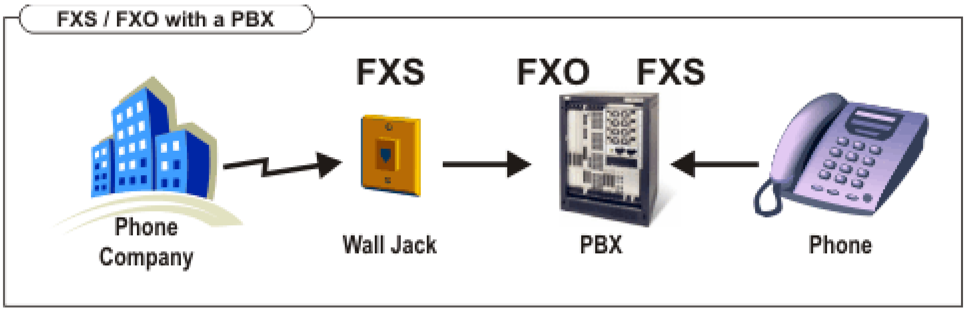

Figura 2.3 FXS/FXO con centralita

Pasarela FXO

Para conectar líneas telefónicas analógicas con una centralita IP, se necesita una pasarela FXO. Ello le permitirá conectar el puerto FXS con el puerto FXO de la pasarela, que luego convierte la línea telefónica analógica en una llamada VOIP, véase la figura 2.4 a continuación.

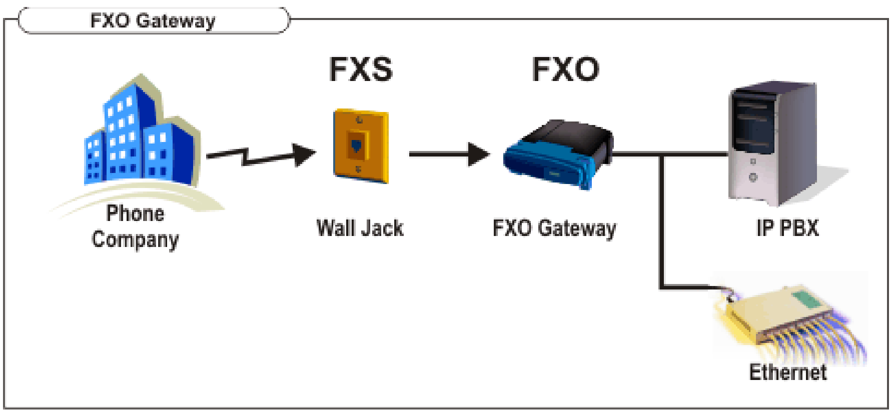

Figura 2.4 Pasarela FXO

Pasarela FXS

La pasarela FXS se usa para conectar una o más líneas de una centralita tradicional con una centralita o suministrador telefónico VOIP. Usted necesitará una pasarela FXS ya que usted desea conectar los puertos FXO (que normalmente se conectan a la empresa telefónica) a la Internet o centralita VOIP, véase la figura 2.5 para comprenderlo mejor.

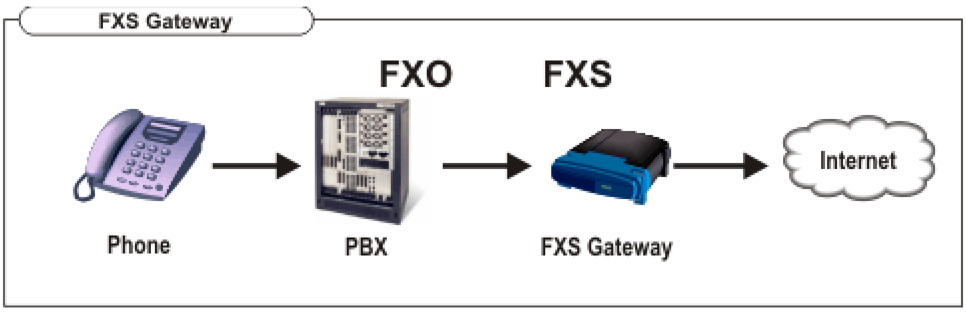

Figura 2.5 Pasarela FXS

Adaptador FXS, también denominado adaptador ATA

El adaptador FXS se usa para conectar un teléfono analógico o aparato de fax a un sistema telefónico VOIP o a un prestador VOIP. Usted lo necesitará para conectar el puerto FXO del teléfono/fax con el adaptador (14).

CAPITULO 3

ESPECIFICACIONES TÉCNICAS DEL SISTEMA

3.1 Hardware

Al momento de establecer las especificaciones del hardware se debe recordar que esto depende de las funcionalidades con que se desea dotar al sistema. No obstante, se puede establecer los requerimientos mínimos del mismo. A continuación:

3.1.1 Servidor

El servidor es una computadora en la que se desea colocar en funcionamiento algún tipo de servicio para determinada función. En nuestro caso, ese servicio es el de la PBX Asterisk, para lo cual, la computadora en la que se ha implementado este proyecto tiene las siguientes características:

Tabla I Características del servidor - hardware

Procesador	Intel Core Duo de 2.8 GHZ
Ram	2 GB
Tarjeta de Red	10/100 Mbps
Disco Duro	80 GB
Tarjeta analógica	Marca Digium TDM410p 4 puertos

3.1.2 Teléfono IP

Figura 3.1 Teléfono IP Grandstream GXP2000

La figura 3.1 muestra el teléfono ip que se emplea en el laboratorio de simulación de telecomunicaciones para las prácticas respectivas de voz sobre ip; su marca es Grandstream, modelo GXP2000, es un tipo de teléfono SIP empresarial de cuatro líneas, que es una característica fácil de usar; soporta PoE (power over Ethernet). ES un teléfono ideal para empresas de todo tipo, ofrece 2 puertos Ethernet 10/100Mbps, tiene una interface de usuario intuitiva, display lcd con soporte para diferentes lenguajes, seguridad y protección de privacidad.

3.2 Software

3.2.1 Servidor

El servidor que será la PBX Asterisk deberá tener instalado los siguientes elementos:

Tabla II Características del servidor – software

Sistema Operativo	Linux
Distribución	Centos 5.2
Arquitectura	x86
Software IP PBX	Asterisk 1.4

Para instalar Asterisk en el servidor se requieren instalar algunas dependencias, los procesos de instalación se detallan en los apéndices de esta tesis. Después de instalar todo lo necesario, se puede continuar con las configuraciones que se explican a continuación.

3.2.2 Configuración del archivo sip.conf

Este archivo de configuración se encuentra en la ruta /etc/Asterisk y sirve para colocar los usuarios sip a registrar en la PBX Asterisk, asi como para conectarse a un proveedor sip y establecer todo lo relacionado a este protocolo. A continuación la configuración que se ha implementado para la realización de este proyecto:

sip.conf

```
[general]
srvlookup=yes
disallow=all
allow=alaw
allow=ulaw
allow=gsm
language=es
callwaiting=no
qualify=yes
calltransfer=no
callforwarding=no
[501]
type=friend
secret=501
```

```
qualify=yes
nat=yes
host=dynamic
canreinvite=no
context=internal

[502]
type=friend
secret=502
qualify=yes
nat=yes
host=dynamic
canreinvite=no
context=internal
```

En esta configuración se han establecido 2 usuarios sip: 501 y 502.

3.2.3 Configuración del archivo iax.conf

Este archivo de configuración se encuentra en la ruta /etc/Asterisk y sirve para colocar los usuarios iax a registrar en la PBX Asterisk y establecer todo lo relacionado al protocolo iax. A continuación la configuración que se ha implementado para la realización de este proyecto:

iax.conf

```
[general]
disallow=all
allow=gsm
allow=alaw
allow=ulaw
jitterbuffer=yes
bindport=4569
language=es
callwaiting=no

[503]
type=friend
secret=503
host=dynamic
context=internal
callerid=503
requirecalltoken=no
```

En esta configuración se han establecido 1 usuario iax: 503.

3.2.4 Configuración de los archivos system.conf y chan_dahdi.conf

Estos archivos se usan para la configuración de los canales dahdi, que son los que se emplean para el uso de la tarjeta analógica tdm410p de 4 puertos. Para nuestro caso, se usa el puerto 4 como fxo con señalización fxs. El archivo system.conf se ubica en la ruta /etc/dahdi y el archivo chan_dahdi.conf se encuentra en /etc/Asterisk. A continuación la configuración para los mismos sería así:

chan_dahdi.conf

```
[channels]
usecallerid=yes
hidecallerid=no
callwaiting=no
threewaycalling=yes
transfer=yes
echocancel=yes
echotraining=yes
inmediate=no
group=1
context=incoming
```

signaling=fxs_ks

channel => 4

system.conf

fxsks=4

echocanceller=mg2,4

loadzone=us

defaultzone=us

Para activar la tarjeta analógica se debe emitir el siguiente comando en consola de root:

- **dahdi_genconf**

3.2.5 Configuración del archivo voicemail.conf

Este archivo de configuración se encuentra en la ruta /etc/Asterisk y sirve para especificar los números de buzón de voz y a quienes están asociados, dando incluso la posibilidad de enviar adjunto a un mensaje de correo electrónico el archivo de audio grabado, el mismo que se encuentra en la ruta /var/lib/Asterisk/sounds. Este archivo debe ser personalizado según los usuarios de las extensiones correspondientes dentro de la PBX. A continuación la configuración que se ha implementado para la realización de este proyecto, a manera de ejemplo:

Voicemail.conf

[default]

Language=es

501=>501,Orly Macias,micorreo@gmail.com,,tz=central|attach=yes

502=>502,Jose Cun,micorreo@gmail.com,,tz=central|attach=yes

503=>503,Juan Ramirez,micorreo@gmail.com,,tz=central|attach=yes

3.2.6 Configuración del archivo extensions.conf

Este archivo de configuración se encuentra en la ruta /etc/Asterisk y sirve para especificar el plan de marcado de la PBX Asterisk y en el cual se va realizar la programación de la retrollamada.

A continuación la configuración que se ha implementado para la realización de este proyecto:

extensions.conf

;Contexto general sirve para establecer los parámetros generales del ;plan de marcado.

[general]

```
autofallthrough=no
clearglobalvars=no

; Contexto internal sirve para especificar el procedimiento a realizarse
;para comunicarse con alguna de las extensiones en el interior de la
PBX.
[internal]
exten => _XXX,1,Answer()
exten => _XXX,2,Wait(1)

; Invoca al script agi que se encargará de crear una base de datos de
;usuarios sip e iax según los archivos de configuración sip.conf e iax.conf
;respectivamente.
exten => _XXX,n,AGI(actualiza_base.agi)

;Invoca al script agi que se encarga de verificar la existencia del número
;marcado en la base de datos creada con el script anterior para devolver
;en una variable el valor del canal en el que se encuentra registrado.
exten => _XXX,n,AGI(scrbasefinal.agi,${EXTEN})

;Rutina para limitar llamadas entrantes a 1 usando grupos y categorías.
exten => _XXX,n,Set(GROUP(${EXTEN})=OUTBOUND_GROUP)
```

exten =>

_XXX,n,Set(GROUP(${CALLERID(num)})=OUTBOUND_GROUP)

exten =>

_XXX,n,GotoIf($[${GROUP_COUNT(OUTBOUND_GROUP@${EXTEN})

} > 1]?busy)

;Bandera o indicador que se guarda en la base de datos AstDB definido

;como cero, su uso se explica más adelante.

exten => _XXX,n,Set(DB(channels/bandera)=0)

;Marca la extensión según el canal en el que se encuentra registrado,

;devuelto por el script: scrbasefinal.agi

exten => _XXX,n,Dial(${CHANN}/${EXTEN},30,wm)

; En caso de que el número marcado no se encuentre habilitado en la

; PBX se manda al buzón de voz al llamante.

exten => _XXX,n,Voicemail(${EXTEN}@default,u)

exten => _XXX,n,Hangup()

;En caso de que la extensión marcada se encuentre ocupada, se le da la

;opción al llamante de activar el callback marcando el número 6 a fin de

;que se le regrese la llamada en cuanto se encuentre disponible, pero

```
;sino lo activa se lo envía al buzón de voz indicándole que la extensión
;marcada se encuentra ocupada.
exten => _XXX,n(busy),Playback(rellamada)
exten => _XXX,n,Read(callbusy,,1,,1,5)
exten => _XXX,n,GotoIf($["${callbusy}" = "6"]?callfile)
exten => _XXX,n,VoiceMail(${EXTEN}@default,b)
exten => _XXX,n,Hangup()

;En caso de que se haya escogido la opción de callback se realiza esta
;rutina para guardar las variables de canal necesarias: canales de
;origen, destino, número del llamante y el número marcado en la base de
;datos AstDB.
exten =>
_XXX,n(callfile),Set(DB(channels/src)=${CHANNEL(channeltype)})
exten => _XXX,n,Set(DB(channels/id)=${CALLERID(num)})
exten => _XXX,n,Set(DB(channels/num)=${EXTEN})
exten => _XXX,n,AGI(scrbasefinal.agi,${DB(channels/num)})
exten => _XXX,n,Set(DB(channels/dst)=${CHANN})
exten => _XXX,n,Set(DB(channels/bandera)=1)
exten => _XXX,n,Playback(colgar)
exten => _XXX,n,Hangup()
```

```
; La bandera nos indica que el callback ha sido activado.
exten => h,1,GotoIf($[${DB(channels/bandera)}=1]?h,2:h,3)

; Se invoca al macro que se encargará de la programación y ejecución
;del callback, enviándole las variables almacenadas en la AStDB.
exten =>
h,2,Macro(internos,${DB(channels/dst)},${DB(channels/id)},${DB(channel
s/num)},${DB(channels/src)})
exten => h,3,Hangup()

;El contexto incoming se usa para especificar el tratamiento de las
;llamadas entrantes desde fuera de la PBX.
[incoming]
exten => s,1,Answer()
exten => s,2,Background(intro1)
exten => s,2,Playback(bienvenida)
exten => s,3,WaitExten()

;Se incluye el contexto internal para que se ejecute todo lo detallado en
;ese contexto.
include => internal
```

```
;Esta macro se encarga de la programación y ejecución del callback
;según las variables enviadas, creando el archivo .call y colocándole en
;el directorio respectivo.
[macro-internos]
exten => s,1,System(echo Channel:${ARG1}/${ARG3}>>
/tmp/callback${ARG3})
exten => s,n,System(echo Callerid:CallBack "<VozToVoice>" >>
/tmp/callback${ARG3})
exten => s,n,System(echo WaitTime:60 >> /tmp/callback${ARG3})
exten => s,n,System(echo Maxretries:10 >> /tmp/callback${ARG3})
exten => s,n,System(echo RetryTime:60 >> /tmp/callback${ARG3})
exten => s,n,System(echo Account: ${ARG2}>> /tmp/callback${ARG3})
exten => s,n,System(echo Application:Macro >> /tmp/callback${ARG3})
exten => s,n,System(echo Data: verify >> /tmp/callback${ARG3})
exten =>
s,n,GotoIf($[${GROUP_COUNT(OUTBOUND_GROUP@${ARG3})} >
1]?busy)
exten => s,n,System(mv /tmp/callback${ARG3}
/var/spool/Asterisk/outgoing)
exten => s,n(busy),Macro(internos,${ARG1},${ARG2},${ARG3},${ARG4})
```

```
;Esta macro se encarga de verificar si la extensión para la llamada de
;retorno se encuentra disponible para realizar el marcado respectivo, esta
;macro es invocada por el archivo .call.
[macro-verify]
exten => s,1,Set(GROUP(${DB(channels/id)})=OUTBOUND_GROUP)
exten =>
s,2,GotoIf($[${GROUP_COUNT(OUTBOUND_GROUP@${DB(channels/i
d)})} > 1]?busy)
exten => s,3,GotoIf($[${DB(channels/src}=DAHDI]?s,4:s,5)
exten => s,4,Set(DB(channels/src)=DAHDI/4)
exten => s,5,Dial(${DB(channels/src)}/${DB(channels/id)})
exten => s,n(busy),VoiceMail(${DB(channels/id)}@default,b)
```

Los archivos que se mencionan en la aplicación Playback fueron grabados con la aplicación Record del plan de marcado, son voces grabadas por nosotros mismos, cuyos mensajes textualmente son los siguientes:

- bienvenida.gsm :”Por favor digite el número de la extensión.”
- rellamada.gsm:”Lo sentimos, en este momento la extensión se encuentra ocupada. Si usted desea que le devolvamos la llamada presione la tecla 6.”
- colgar.gsm:”La acción fue programada, ahora puede colgar.”

Estos archivos deben estar situados en la ruta /var/lib/Asterisk/sounds, en forma predeterminada se almacenan allí.

3.2.7 Script de creación y actualización de base de datos usuarios

Con este script escrito en php y colocado en la ruta /var/lib/Asterisk/agi-bin se crea una base de datos con mysql que contiene los usuarios sip e iax que están configurados en los archivos correspondientes. Se debe reemplazar la contraseña de root (labtelecom09) por el propio del servidor Asterisk.

actualiza_base.agi

```
#!/usr/bin/php -q
<?php
require '/var/lib/Asterisk/agi-bin/phpagi-2.14/phpagi.php';
$agi=new AGI();
function connect_db()
{
$db_connection = mysql_connect ('localhost', 'root', 'labtelecom09') or die
(mysql_error());
$sql1 = "DROP DATABASE IF EXISTS Asterisk";
mysql_query($sql1);
$sql = "CREATE DATABASE Asterisk";
```

```
mysql_query($sql);
$sql2="GRANT INSERT ON Asterisk.* TO Asterisk@localhost
IDENTIFIED BY 'labtelecom09'";
mysql_query($sql2);
$db_select = mysql_select_db('Asterisk') or die (mysql_error());
$sql3="DROP  TABLE IF EXISTS `sip`";
mysql_query($sql3);
$sql4= "CREATE TABLE `sip` (`peer` varchar(80) NOT NULL default '')";
mysql_query($sql4);
$sql5="DROP TABLE IF EXISTS `iax`";
mysql_query($sql5);
$sql6="CREATE TABLE `iax` (`peer` varchar(80) NOT NULL default '')";
mysql_query($sql6);
$sql7="DROP TABLE IF EXISTS `dahdi`";
mysql_query($sql7);
$sql8="CREATE TABLE `dahdi` (`peer` varchar(80) NOT NULL default
'')";
mysql_query($sql8);
}
//Programa principal
connect_db();
```

```
$fichero = @fopen("/etc/Asterisk/sip.conf","r") or die("No se puede abrir el
archivo");
$query0="DELETE FROM sip";
mysql_query($query0) or die('Error, delete query failed');
while(!feof($fichero)){
        if(strcasecmp(fgetc($fichero),"[")==0){
                $temp=fgetc($fichero);
                $cadena="";
          while(strcasecmp($temp,"]")!=0){
               $cadena=$cadena.$temp;
                $temp=fgetc($fichero);
          }
                if(strcasecmp($cadena,"general")!=0){
                  $query = "INSERT INTO sip VALUES ('$cadena')";
                  mysql_query($query) or die('Error, insert query failed');
                }
        }
}
fclose($fichero);
$fichero = @fopen("/etc/Asterisk/iax.conf","r") or die("No se puede abrir el
archivo");
$query0="DELETE FROM iax";
```

```
mysql_query($query0) or die('Error, delete query failed');
while(!feof($fichero)){
        if(strcasecmp(fgetc($fichero),"[")==0){
          $temp=fgetc($fichero);
                $cadena="";
          while(strcasecmp($temp,"]")!=0){
        $cadena=$cadena.$temp;
        $temp=fgetc($fichero);
     }//fin de while de concatenacion
                if(strcasecmp($cadena,"general")!=0){$query = "INSERT INTO
iax VALUES ('$cadena')";
                  mysql_query($query) or die('Error, insert query failed');
                }//fin de if de inserción
   }//fin de if de [
}//fin de while feof
 fclose($fichero);
?>
```

3.2.8 Script para identificación del canal destino del número marcado

Con este script escrito en php y colocado en la ruta /var/lib/Asterisk/agi-bin se busca el número marcado o extensión para determinar el canal al que pertenece, es decir si se encuentra registrado como usuario sip o iax, consultando la base de datos creada anteriormente, de manera que se pueda establecer el canal de destino para marcar a dicha extensión en el plan de marcado.

scrbasefinal.agi

```
#!/usr/bin/php -q
<?php
require '/var/lib/Asterisk/agi-bin/phpagi-2.14/phpagi.php';
$agi=new AGI();
function connect_db()
{
$db_connection = mysql_connect ('localhost', 'root', 'labtelecom09') or die
(mysql_error());
$db_select = mysql_select_db('Asterisk') or die (mysql_error());
}
function esSIP($cli){
$query1 = "SELECT peer FROM sip WHERE peer = '$cli' ";
$query_result1 = @mysql_query($query1);
$row_count = mysql_num_rows($query_result1);
$row1 = @mysql_fetch_array ($query_result1);
return $row_count;
```

```
}

function esIAX($cli){
$query1 = "SELECT peer FROM iax WHERE peer = '$cli' ";
$query_result1 = @mysql_query($query1);
$row_count = mysql_num_rows($query_result1);
$row1 = @mysql_fetch_array ($query_result1);
return $row_count;
}
//programa inicial
$cli1 = $argv[1];
connect_db();
if(esSIP($cli1)) { //registro ya existe en SIP
$temp="SIP";
}else if(esIAX($cli1)) {
$temp="IAX2";
}else {
        $temp="";
}
$agi->set_variable("CHANN","1");
$agi->set_variable("CHANN","$temp");
mysql_close();
```

?>

3.2.9 Configuración softphone X-Lite

Este softphone al igual que el zoiper tiene soporte para usuarios sip, la diferencia entre ambos es que el zoiper también tiene soporte para usuarios iax. A continuación se describe como registrar un usuario sip en este softphone.

Display Name: 501

Username: 501

Domain: ip_servidor_Asterisk

Y colocar visto en la opción que dice: Registrar con dominio y recibir llamadas entrantes.

Asimismo se debe crear otro usuario con los siguientes datos:

Display Name: 502

Username: 502

Domain: ip_servidor_Asterisk

3.2.10 Configuración softphone Zoiper

Este softphone tiene soporte para usuarios sip e iax. A continuación se describe cómo registrar un usuario iax en este softphone, proceder de la misma manera en el caso de usuarios sip.

Esta es la interfaz del zoiper, se presiona sobre la llave para configurar cuentas de usuario.

Se abre una ventana y en el lado izquierdo aparecen los usuario sip e iax configurados. Para crear cuentas IAX2 con el softphone zoiper, es igual que en SIP, primero damos click en el boton options, aquel que es una llave, damos click en IAX accounts y luego seleccionamos add new iax account.

Se le da el nombre a la cuenta.

Finalmente se abre una especie de formulario para completar los datos de la cuenta. Procedemos a llenar esos campos com o se indica abajo.

Por ejemplo, se crea el usuario IAX 503

Server Hostname/IP: dirección_ip_del_servidor

Username: 503

Password: 503

Caller ID Name:503

3.2.11 Configuración teléfono ip Grandstream GXP2000

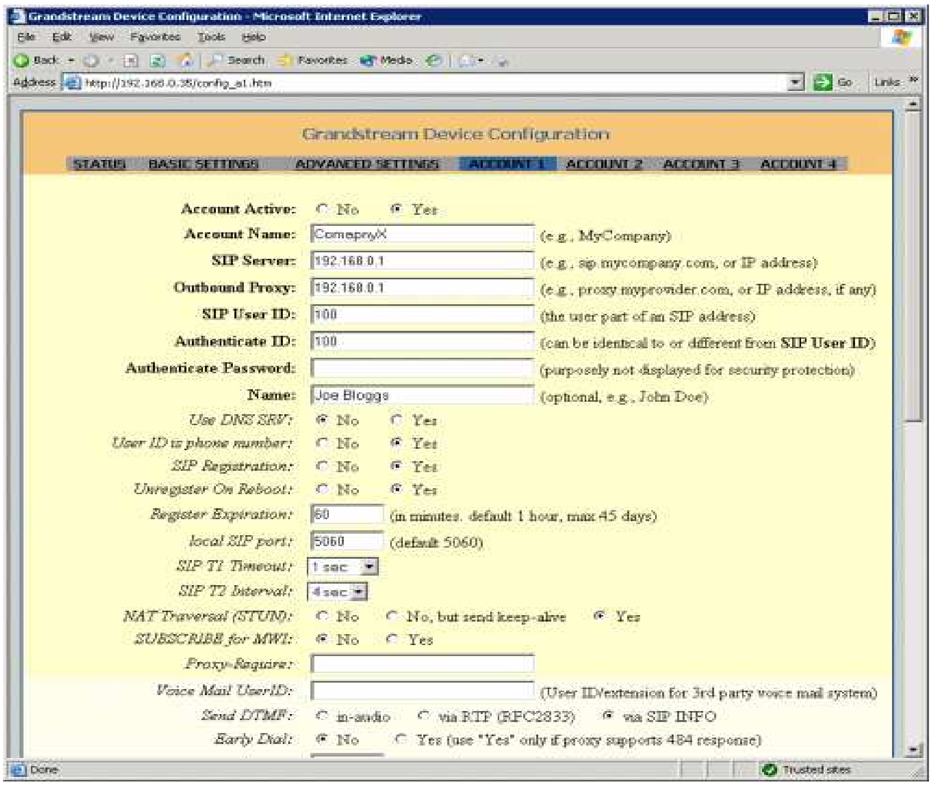

Figura 3.2 Interfaz web del teléfono ip Grandstream GXP2000

Como se muestra en la figura 3.2, la interfaz web de este teléfono ip se puede acceder mediante un browser, digitando como url: http://direccion_ip_del_telefono y autenticándose con el usuario y contraseña de fábrica, en los teléfonos que se están usando en este caso, por lo general el usuario y contraseña son los mismos: admin; pero por seguridad se recomienda cambiarlos. En este tipo de teléfonos sólo se pueden configurar usuarios sip. Por ejemplo, en lugar de usar x-lite para crear cuentas sip se puede usar este dispositivo, colocando los siguientes parámetros:

Account Name:501

SIP server:501

Outbound proxy:direccion_ip_del_servidor_Asterisk

SIP user ID:501

Authenticate ID:501

Authenticate password:501

Name: 501

Local Sip port: 5060

El resto de parámetros son opcionales, después de hacer las configuraciones respectivas, se debe reiniciar el teléfono a través del botón Reboot de la interfaz web.

CAPITULO 4

FUNCIONAMIENTO Y PRUEBAS DEL PROYECTO

4.1 Inicialización de Asterisk

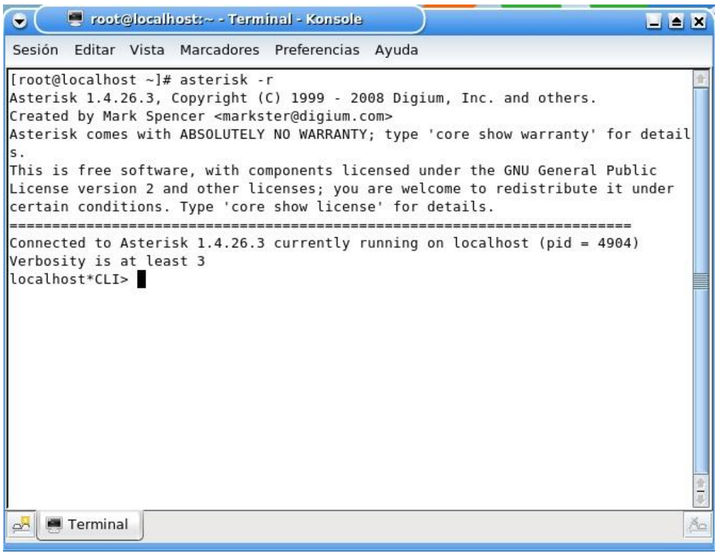

Figura 4.1 Inicialización de Asterisk

1. Como se muestra en la figura 4.1, lo primero es inicializar el servidor Asterisk, en esta imagen se muestra el prompt CLI>.
2. Se procede a registrar los clientes sip e iax mediante los softphone o el teléfono ip. tal y como se observa en la figura 4.2, así aparecen cuando ya están registrados en el servidor.

Figura 4.2 Cuentas sip e iax registradas

Se puede verificar que los clientes están registrados en la consola de Asterisk, emitiendo los comandos **sip show peers** e **iax show peers** respectivamente.

4.2 Realización de llamada entre 2 usuarios

Figura 4.3 Usuario sip 501 llama a 502

En la figura 4.3 se puede observar que un usuario sip está llamando a otro usuario sip, cuyas extensiones son respectivamente 501 y 502; mientras que en la figura 4.4 se observa la consola del servidor Asterisk en la que se muestran los procesos que están ejecutándose, invocando al plan de marcado en el momento que el usuario sip 501 marca la extensión 502.

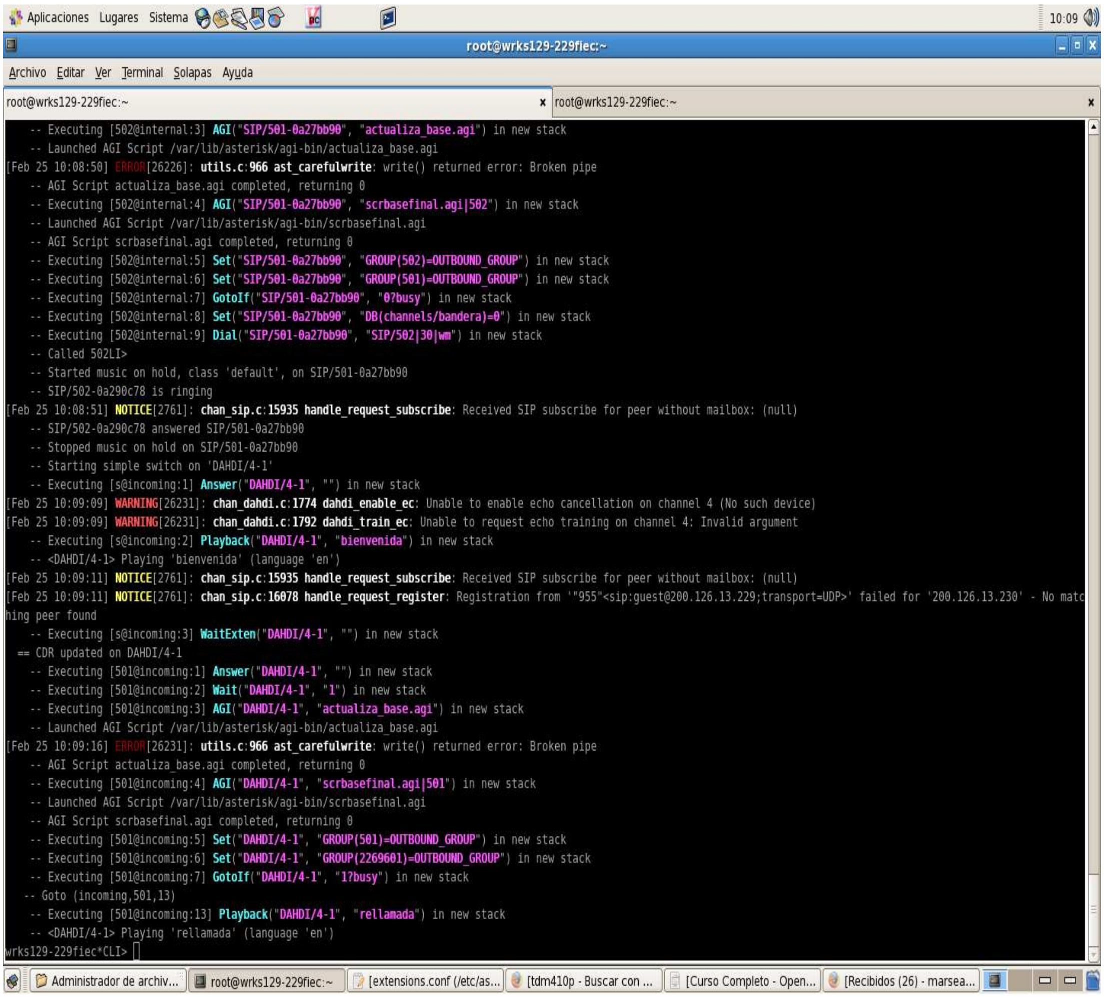

Figura 4.4 Ejecución de plan de marcado cuando usuario sip 501 llama a 502

4.3 Realización de llamada de un usuario a una línea ocupada.

En la figura 4.5 se puede observar la secuencia de dígitos marcados por el usuario llamante de la red de telefonía pública.

1. El usuario marca el teléfono de la organización, que en este caso es el 2269502.

2. Una grabación en la PBX le dice que digite el número de la extensión. En este caso el 501.
3. Una grabación en la PBX le dice que la extensión marcada se encuentra ocupada y le da la opción que se le devuelva la llamada en cuanto esté disponible, marcando el número 6. En este caso el usuario escoge esa opción.

Figura 4.5 Llamada fuera de la PBX a la extensión 501

4.4 Ejecución de la retrollamada

A continuación se muestra lo que ocurre en el servidor en el transcurso de los pasos anteriores hasta la ejecución de la retrollamada o callback.

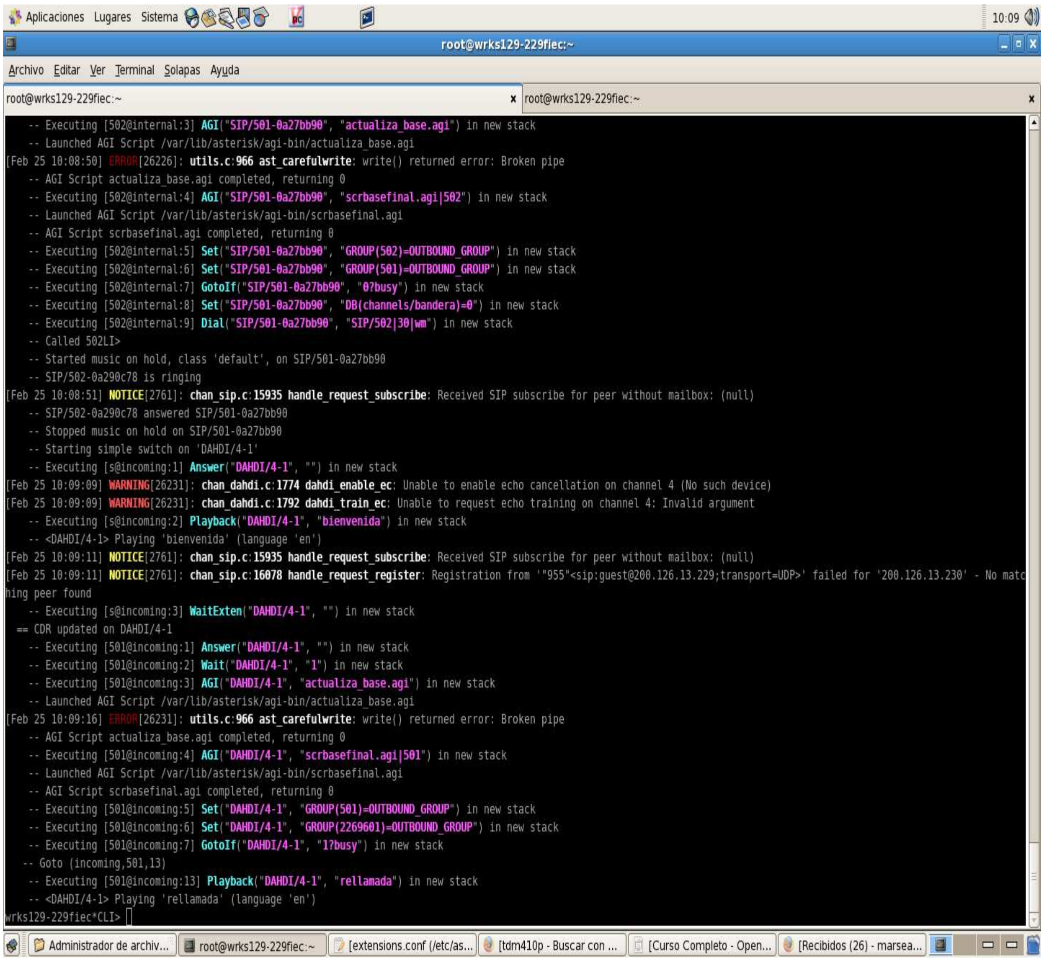

Figura 4.6 Ejecución del plan de marcado – parte 1

En la figura 4.6 se puede observar que se está generando una llamada de la extensión sip 501 a la 502 de la PBX de la organización y que posteriormente entra una llamada a través del canal Dahdi y asimismo se ejecuta el plan de marcado para contactar a la extensión 501, la misma que está ocupada en ese momento y entonces se reproduce la grabación para dar la opción de la rellamada o callback al llamante.

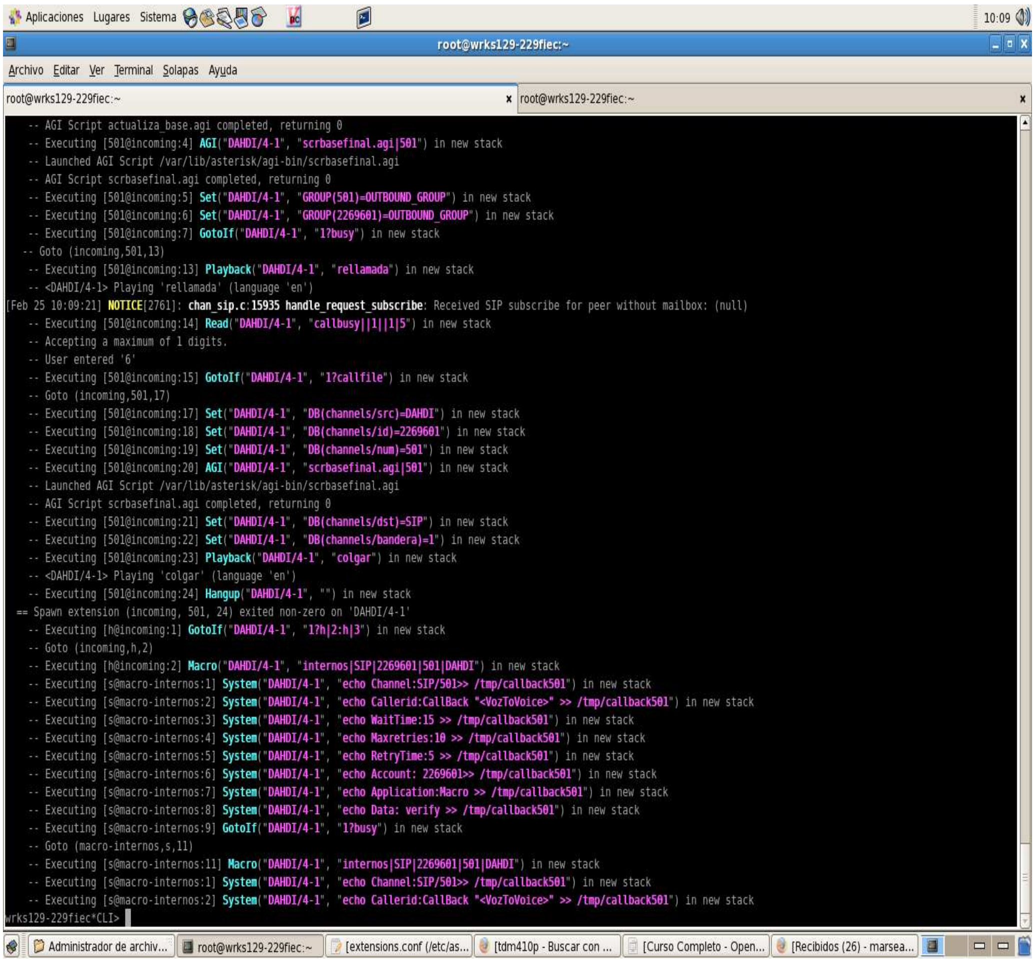

Figura 4.7 Ejecución del plan de marcado – parte 2

En la figura 4.7 se observa que el llamante de la línea externa a la PBX ha escogido la opción del callback y presionó el número 6 para activarlo, se muestra que se almacenan las variables de canal en la AstDB y manda a ejecutar el macro internos, el mismo que crea el archivo .call con todos sus parámetros.

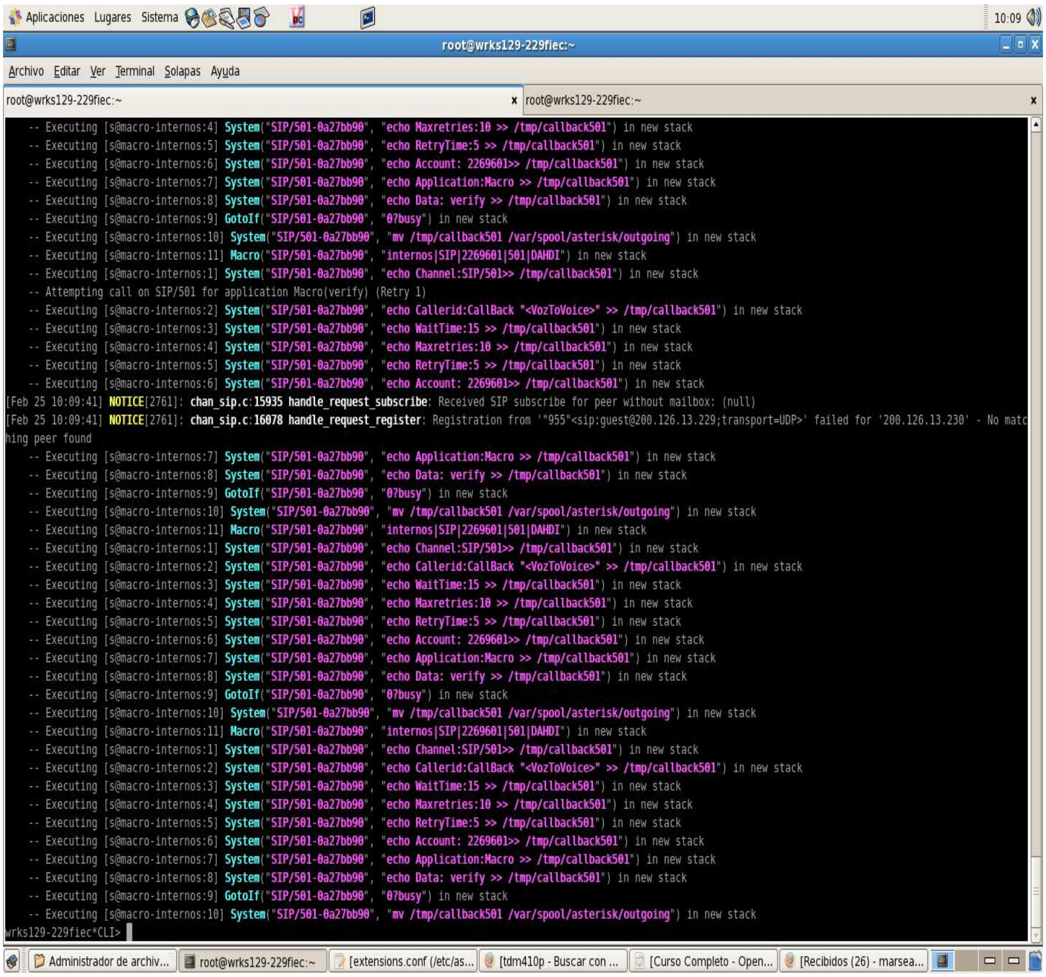

Figura 4.8 Ejecución del plan de marcado – parte 3

En la figura 4.8 se observa que el archivo .call es movido de la ruta temporal en la que se lo creó a /var/spool/Asterisk/outgoing para su ejecución, esto se realiza algunas veces seguidas, puesto que el usuario llamado aún sigue con su línea ocupada.

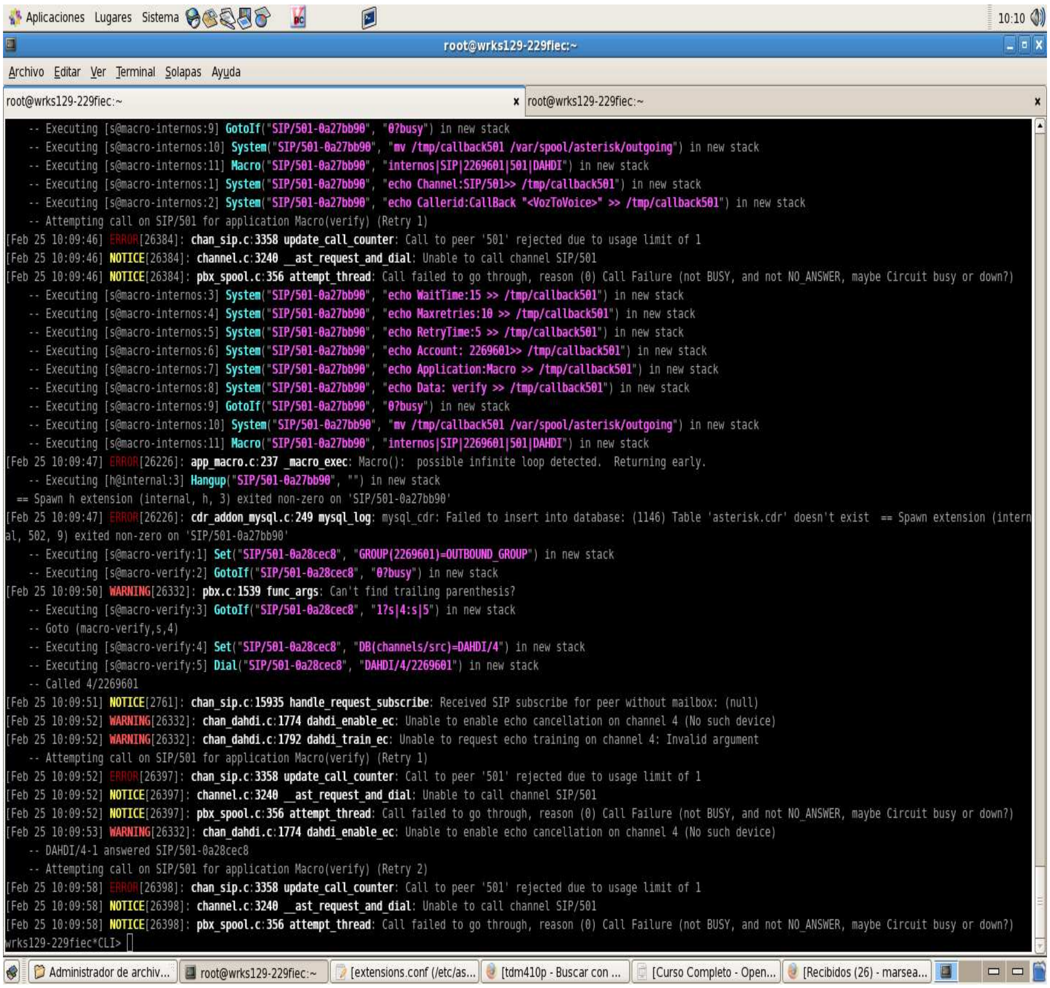

Figura 4.9 Ejecución del plan de marcado – parte 4

En la figura 4.9 se observa que se realiza la retrollamada o callback, regresando de esta manera la llamada al número telefónico del usuario externo a la PBX, cuyo número es 2269601 mediante el canal dahdi.

Figura 5.1 Pruebas de la ejecución del callback¡Error! Marcador no definido.¡Error! Marcador no definido.

Finalmente, en la figura 5.1 se puede observar como primera imagen la del retorno de la llamada al teléfono analógico, externo a la PBX Asterisk, mostrando en su display el número de la empresa, que en este caso es el 2269602. Las siguiente imagen corresponde a la ejecución del callback para un usuario iax registrado en el softphone zoiper y la última para el caso de un usuario sip registrado en el teléfono ip grandstream GXP2000.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

1. La implementación de esta solución tecnológica es una contribución a la importancia de las telecomunicaciones a nivel empresarial, puesto que en diversas ocasiones, una llamada telefónica puede ser tan importante, que se debe procurar no dejar de atenderla, capturándola mediante un sistema callback como el propuesto en esta tesis, siendo esta una alternativa más eficaz con respecto a lo que hoy en día se le conoce como el buzón de voz.

2. La utilización de la PBX open source conocida como Asterisk permite un alto nivel de personalización del sistema a un costo bajo y más ajustado a las necesidades propias de quienes lo requieran e incluso trascender sus expectativas.

3. La flexibilidad con la que se ha programado el sistema callback en Asterisk permite que pueda ser fácilmente aplicado en una organización con tecnologías sip, iax y dahdi para conexión con la red de telefonía pública, las cuales son ampliamente usadas en la comunicación mediante voz sobre ip.

4. La variedad de componentes que se pueden emplear al momento de desarrollar un sistema en Asterisk es abundante y sumado con la creatividad de programar scripts bajo Linux hacen que su implementación sea relativamente sencilla, asequible, escalable, segura, personalizable, con una inversión de bajo costo , acorde a las tecnologías y necesidades actuales gracias al uso del internet como canal de comunicaciones, lo cual la constituye como la mejor opción con respecto a los sistemas de comunicaciones tradicionales.

Recomendaciones

1. Los parámetros del archivo .call pueden ser configurados y ajustados a las necesidades de la empresa, es posible que se desee disminuir el número de intentos del callback o aumentar el tiempo de espera para su ejecución a fin de no sobrecargar el servidor con la realización de llamadas salientes, se recomienda adaptarlo a sus necesidades, la programación del callback se encuentra en el archivo extensions.conf en la macro internos, que está ubicado en la ruta /etc/Asterisk.

2. Asegurarse que los scripts que se colocan en la ruta /var/lib/Asterisk/agi-bin se encuentren con los permisos de ejecución

correspondientes, por ejemplo mediante el comando chmod 777 nombre_del_script.

3. Verificar en los softphones que se encuentren habilitados los codecs de audio gsm, allaw, ullaw, que son los que se suelen usar para las voces del buzón de voz.

4. Recuerde colocar la librería phpagi versión 2.14 en la ruta /var/lib/Asterisk/agi-bin, ya que se la requiere en el script scrbasefinal.agi, en este mismo directorio.

5. Asegurarse de usar el puerto correcto de la tarjeta Digium TDM410p para el cual ha sido configurada la conexión con la red de telefonía pública, cuya configuración se encuentra en el archivo chan_dahdi.conf, que se encuentra en la ruta /etc/Asterisk.

TRABAJO A FUTURO

Solución a costosas llamadas internacionales

Este sistema callback se puede ajustar para que sea activado de cualquiera de tres formas posibles:

1. Mediante una llamada telefónica.
2. Mediante un mensaje sms.
3. Mediante una página web.

En el primer caso, el funcionamiento es exactamente similar al que se detalla en esta tesis, únicamente se requeriría adicionar en el plan de marcado, en el contexto internal, el tratamiento de llamadas internacionales entrantes, así las personas que llaman a la empresa desde afuera se les devolverá la llamada y evitarían altos costos por la misma. En el segundo y tercer caso la diferencia sería el modo en cómo se activa el callback que sería mediante un mensaje sms o por un enlace en una página web. Otra enfoque para el callback es servir como método de arbitraje para realizar llamadas internacionales; así por ejemplo, para brindar este servicio como empresa, brindándole al usuario la alternativa de que cuando requiera hacer una llamada internacional, active el callback de cualquiera de las tres formas descritas anteriormente, para que se le devuelva la llamada y al contestar digite el número al cual desee llamar, sin incurrir por costo de llamadas internacionales salientes.

APÉNDICES

APÉNDICE A

INSTALACIÓN DE ASTERISK Y SUS DEPENDENCIAS

Se requiere descargar el paquete estable de Asterisk 1.4, se lo puede encontrar en el siguiente enlace:

http://downloads.digium.com/pub/Asterisk/Asterisk-1.4-current.tar.gz

1. Se recomienda tener todos los paquetes actualizados:

   - **yum update o yum upgrade –y**

2. Se necesita tener actualizado y funcionando los paquetes:

   gcc: es el compilador de C y todas sus dependencias relacionadas.

   openssl: librerias de desarrollo para soporte RSA y MD5.

   ncurses: para el CLI y las otras dependencias relacionadas.

   Fuentes del kernel de Linux (librerias del sistema)

   - **yum -y install gcc gcc-c++ kernel-devel bison openssl-devel libtermcap-devel ncurses-devel**

3. Para instalar las dependencias y soporte para DB:

   - **yum -y install mysql-server mysql-devel newt-devel unixODBC unixODBC-devel libtool-ltdl libtool-ltdl-devel mysql-connector-odbc**

4. En el caso de tener problemas de “sources for kernel”, emita los siguientes comandos

- **yum –y install kernel-devel**
- **yum –y update kernel**
- **uname -r**

De ser necesario reinicie verificando la correcta version de Linux!!

5. Descargue los paquetes en /usr/src

   - **wget -c http://downloads.digium.com/pub/Asterisk/Asterisk-1.4-current.tar.gz**
   - **wget -c http://downloads.digium.com/pub/Asterisk/Asterisk-addons-1.4-current.tar.gz**
   - **wget -c http://downloads.digium.com/pub/telephony/dahdi-tools/dahdi-tools-current.tar.gz**
   - **wget -c http://downloads.digium.com/pub/telephony/dahdi-linux/dahdi-linux-current.tar.gz**
   - **wget -c http://downloads.digium.com/pub/libpri/libpri-1.4-current.tar.gz**

6. Siguiendo el LSB (Linux Standard Base) de Asterisk estos paquetes deberán ser descomprimidos en /usr/src/

   - **tar -xvzf Asterisk-1.4-current.tar.gz**
   - **tar -xvzf Asterisk-addons-1.4-current.tar.gz**

   Descomprima los otros paquetes también.

7. Ejecutar el siguiente grupo de comandos:

Comandos para instalación de Asterisk

1. cd dahdi-linux-current make make install	4. cd ../Asterisk-1.4.22 make clean ./configure make menuconfig (opt) make install make samples make config
2. cd dahdi-tools-current ./configure make make install make config	5. cd ../Asterisk-addons-1.4.7 make clean ./configure make menuselect (opt) make install make samples
3. cd ../libpri-1.4.7 (opt) make make install	LISTO !!

Finalmente para comprobar que Asterisk se encuentra instalado se puede ingresar a su consola, emitiendo el comando **Asterisk –r** y se debería ver el prompt **CLI>** .

APÉNDICE B

INSTALACIÓN DE MYSQL

A continuación se detalla el procedimiento para instalar mysql:

1. Verificar e instalar MySQL:
   - **yum install mysql* unixODBC-devel -y**

   o sino:
   - **yum -y install mysql-server mysql-devel newt-devel unixODBC unixODBC-devel libtool-ltdl libtool-ltdl-devel mysql-connector-odbc**
2. Iniciar el servicio mysql como usuario root:
   - **service mysqld start mysqladmin -u <root> password <contraseña_de_root>**
3. Ingresar al directorio donde se encuentran las fuentes de Asterisk add-ons e instalar el soporte para mysql:
   - **cd /usr/src/Asterisk-addons-**
   - **./configure**
   - **make menuselect**
   - **make**
   - **make install**

APÉNDICE C

INSTALACIÓN DE PHP

Se requiere la instalación del compilador de php versión 5 para la ejecución de los scripts .agi escritos en lenguaje de programación php. A continuación se detalla el procedimiento de instalación:

1. Para agregar php para trabajar con mysql, ejecute el comando:
   - **yum install php***
2. Para que se pueda ejecutar un script, se debería cambiar los permisos de ejecución mediante el comando:
   - **chmod 777 <ruta_absoluta_o_nombre_archivo.php>**
3. Para correr un script php en consola se debe ubicar en la ruta donde se encuentra el archivo o especificar la ruta absoluta en el comando y ejecutar el comando:
   - **! /usr/bin/php -q <archivo.php>**
4. Adicionalmente se debe descargar y colocar la librería phpagi versión 2.14 en la ruta /var/lib/Asterisk/agi-bin, puesto que se la requiere para uno de los scripts escritos en php, que se emplean en este proyecto.

   Para hacer debugging de un script agi en Asterisk ejecute en el CLI>agi debug.

BIBLIOGRAFÍA

1.**Wikipedia**, definición de Asterisk
http://es.wikipedia.org/wiki/Asterisk, Enero 2010.

2.**Tecsible**, características de Asterisk
http://www.tecsible.com/Asterisk_funcionalidades.php, Enero 2010.

3.**Universidad Columbia**, definición del protocolo SIP
http://www.cs.columbia.edu/sip/, Enero 2010.

4.**CENIP**, información detallada del protocolo SIP
http://www.cenip.com.ar/index.php?option=com_content&task=view&id=18&Itemid=25, Enero 2010.

5.**TopBits**, Definición del protocolo IAX
http://www.topbits.com/iax.html, Enero 2010.

6.**Nextor Telecom**, información detallada del protocolo IAX
http://www.nextortelecom.com/nextor_voip/base-de-conocimientos/73-iax-protocolo, Enero 2010.

7.**Voip-info.org,** información detallada sobre AGI. http://www.voip-info.org/wiki/view/Asterisk+AGI, Enero 2010.

8.**Voz to Voice**, información detallada sobre archivos .call
http://www.voztovoice.org/tmp/callfiles.txt, Enero 2010.

9.**The Asterisk Book**, archivos .call
http://www.the-Asterisk-book.com/unstable/call-file.html, Enero 2010.

10.**Página oficial de Asterisk**, información detallada sobre Macros
http://www.Asterisk.org/docs/Asterisk/trunk/applications/macro, Enero 2010.

11. **Asterisk: el futuro de la telefonía**, base de datos de Asterisk AstDB http://astbook.Asteriskdocs.org/en/2nd_Edition/Asterisk-book-html-chunk/Asterisk-CHP-6-SECT-6.html, Enero 2010.
12. **Voip-info.org**, información detallada sobre grupos y categorías http://www.voip-info.org/wiki/view/Asterisk+groups,Enero 2010.
13. **Voip-info.org**, información detallada sobre buzón de voz en Asterisk http://www.voip-info.org/wiki/view/Asterisk+cmd+VoiceMail,Enero 2010
14. **Central Telefónica 3CX**, información detallada sobre fxs y fxo http://www.3cx.es/voip-sip/fxs-fxo.php, Enero 2010

GLOSARIO

Base de datos: Una base de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

Callback: es un proceso por el cual una persona que origina una llamada telefónica, es inmediatamente colgada y vuelta a llamar (tras unos segundos) sin que incurra en ningún tipo de coste en la realización de dicha llamada.

CentOS: Community ENTerprise Operating System, es un clon a nivel binario de la distribución Linux Red Hat Enterprise Linux RHEL, compilado por voluntarios a partir del código fuente liberado por Red Hat.

Codec: es una librería de software que contiene los algoritmos necesarios para convertir una señal analógica a una señal digital.

Dependencia: En el campo del software una dependencia es una aplicación o una biblioteca requerida por otro programa para poder funcionar correctamente. Por ello se dice que dicho programa depende de tal aplicación o biblioteca.

IP - Internet Protocol: La parte IP del protocolo de comunicaciones TCP/IP. Implementa el nivel de red (capa 3 de la pila de protocolos OSI), que contiene una dirección de red y se utiliza para enrutar un paquete hacia otra red o subred.

MySQL: es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones

PBX: es la sigla de “Private Branch eXchange”. Es el sistema que conecta llamadas dentro de la misma compañía. Comúnmente puede tener desde dos a diez mil extensiones y una conexión al sistema telefónico (PSTN) para llamadas hacia o desde el exterior de la compañía.

PHP: es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas.

Script: es un conjunto de instrucciones, sentencias de control, variables y demás elementos de programación generalmente almacenadas en un archivo de texto (pude considerarse como un archivo de instrucciones o como un programa).

Softphone: es un software que hace una simulación de teléfono convencional por computadora, es decir, permite usar la computadora para hacer llamadas a otros softphones o a otros teléfonos convencionales.

VoIP: es la sigla para "Voice Over Internet Protocol". Es lo que lleva llamadas telefónicas a través de redes de datos como redes corporativas o el internet.