

PROYECTO FINAL DE INGENIERÍA

COMUNICACIONES INTEGRADAS

De Julián Rousselot – LU 109192

Ingeniería en Informática

Tutor/es:

García, Matías Alejo, UADE

Colaborador:

Lain, Lucas

Octubre 20, 2008



UNIVERSIDAD ARGENTINA DE LA EMPRESA
FACULTAD DE INGENIERÍA Y CIENCIAS EXACTAS



Agradecimientos

Agradecimientos agradecimientos Agradecimientos agradecimientos Agradecimientos
agradecimientos Agradecimientos agradecimientos Agradecimientos agradecimientos
Agradecimientos agradecimientos Agradecimientos agradecimientos Agradecimientos
agradecimientos Agradecimientos agradecimientos Agradecimientos agradecimientos
Agradecimientos agradecimientos Agradecimientos agradecimientos Agradecimientos
agradecimientos Agradecimientos agradecimientos Agradecimientos agradecimientos
Agradecimientos agradecimientos Agradecimientos agradecimientos Agradecimientos
agradecimientos Agradecimientos agradecimientos Agradecimientos agradecimientos
Agradecimientos agradecimientos Agradecimientos agradecimientos Agradecimientos
agradecimientos Agradecimientos agradecimientos Agradecimientos agradecimientos



Resumen

En este trabajo se describe el desarrollo de una plataforma de comunicaciones telefónicas accionada vía web, la cual implementa un sistema de llamadas telefónicas gratuitas para comunicarse en forma rápida y sin costo. La plataforma esta soportada por un modelo de negocios innovador en el cual las llamadas son costeadas mediante avisos publicitarios interlocutores.

Para la implementación de la plataforma se utilizó la tecnología Java Enterprise Edition junto con la instalación completa del servidor de comunicaciones Asterisk PBX encargado del establecimiento y administración de las llamadas.

Se incluye en el trabajo un estudio económico del modelo propuesto el cuál expresa la viabilidad o no de la implementación del sistema en el mercado actual.

Este trabajo comienza con un breve estudio de la importancia y evolución de las comunicaciones telefónicas en la vida cotidiana, diferencia entre la telefonía IP y la telefonía tradicional y luego la descripción de la arquitectura tecnológica para dar este tipo de servicios.

En el transcurso del documento se exponen todas las herramientas usadas para llevar a cabo este proyecto, estas herramientas son de código abierto, el autor mencionará la importancia que tiene este tipo de “filosofía” en el diseño, desarrollo, implementación y prueba de los sistemas informáticos.

Por último se expondrá el sistema desarrollado para la solución propuesta ilustrada en una arquitectura moderna de una aplicación web, su tecnología.



Abstract



Contenidos

Introducción.....	8
1. Integración de aplicaciones en redes telefónicas IP.....	9
1.1 Las comunicaciones a lo largo del tiempo.....	9
1.2 Redes telefónicas IP.....	11
1.3 Voz Sobre IP.....	12
1.4 Arquitectura de VOIP.....	14
2. El Mercado.....	17
2.1 Casos de negocio exitosos que ofrecen servicios VOIP.....	18
2.2 Costo publicitario en Argentina.....	21
2.2.1 Costo de la publicidad en medios gráficos.....	22
2.2.2 Costo de la publicidad en TV.....	22
2.2.3 Costo de la publicidad en radio.....	23
2.2.4 Costo de la publicidad en HableGratis.NET.....	23
3. Construcción del Prototipo.....	25
3.1 Construcción de la aplicación de comunicaciones integradas con tecnología VOIP orientadas a la web.....	25
3.2 Interfaz de usuario final.....	26
3.3 Caso de uso del usuario final.....	28
3.4 Interfaz administrativa.....	29
3.5 Página para el usuario administrador.....	31
3.6 Caso de uso del usuario administrador.....	32
3.7 Arquitectura del sistema de comunicaciones integradas.....	32
4. Aplicación.....	34
4.1 Diagrama de Transición de Estados (DTE).....	35
4.2 Diagrama de Clases.....	36



4.3 Descripción de las clases mas importantes del sistema.....	37
4.4 Función del Session Bean que establece la llamada.....	40
4.5 Plataforma para gestionar las comunicaciones.....	42
4.6 Instalación y configuración del servidor Asterisk.....	43
4.7 Configuración de TrixBox.....	46
4.8 Configuración del archivo manager.conf.....	51
4.9 Configuración de los contextos.....	51
4.10 Base de datos.....	52
4.11 Integración de la aplicación Web con el servidor de comunicaciones Asterisk.....	53
4.12 Descripción de la integración.....	54
4.13 Trabajo a Futuro.....	54
5. Conclusiones.....	56
Bibliografía.....	58
Anexo A, Herramientas utilizadas.....	59
La importancia del uso de herramientas de código abierto.....	61
Anexo B, Código Java de la aplicación.....	63
OriginateBean.....	63
FachadaBean.....	69
Publicidad.....	70
AdministradorPublicidadBean.....	72
AdministradorLlamadasBean.....	75
Anexo C, Archivos configurados del Asterisk PBX.....	77
zapata.conf.....	77
sip.conf.....	78
sip_additional.conf.....	79
sip_nat.conf.....	80



rtp.conf.....	81
extension_custom.conf.....	81
asterisk.conf.....	82
Consola de administración remota del servidor Asterisk.....	83



Introducción

Este proyecto final de Ingeniería trata sobre la implementación de un sistema que permitirá que dos personas puedan comunicarse telefónicamente accionando la llamada desde Internet, de una manera gratis y sencilla.

El mercado de las telecomunicaciones esta en fuerte crecimiento, las distintas tecnologías como dispositivos celulares están y estarán cada vez mas presentes en nuestras vidas demostrando la necesidad del ser humano de estar en constante comunicación con sus semejantes, de aquí parte la idea de hacer este proyecto que sera implementado mediante el nombre **HableGratis.NET**.

El trabajo trata sobre la integración de la tecnología de voz sobre IP en aplicaciones web convencionales utilizando como servidor back-end de comunicaciones a Asterisk¹ y construyendo la aplicación web mediante el lenguaje Java EE.

Primero se expondrá brevemente un resumen histórico sobre la evolución de las comunicaciones en el mundo y la importancia que estas tienen en la sociedad actual, luego se desarrollarán las diferencias entre la telefonía convencional y la telefonía IP introduciendo varios conceptos como protocolos de comunicación y codecs de comprensión de voz, se mencionarán los casos de negocio exitosos que ofrecen soluciones VoIP, posteriormente se expondrá el análisis, diseño y desarrollo de la solución propuesta para las comunicaciones integradas, se hará la descripción del servidor de comunicaciones que utilizaremos como back-end

En el anexo A se mencionan las herramientas utilizadas a lo largo del trabajo, en el Anexo B se encuentra todo el código de la aplicación y en el anexo C se encuentran todos los archivos de configuración que hicieron falta modificar para la correcta configuración del servidor Asterisk.

¹ Asterisk: Es una PBX opensource



Capítulo I

1. Integración de aplicaciones en redes telefónicas IP

1.1 Las comunicaciones a lo largo del tiempo

El individuo aislado no puede existir. El ser humano siempre aparece relacionándose con otros seres humanos. Los modos de convivencia pueden adoptar innumerables modalidades; por ejemplo, ser pacíficos o conflictivos, interesados o altruistas, afectivos o racionales, etc.; pero, en cualquier caso, la sociedad y los individuos humanos no pueden ser concebidos sino en relación intrínseca entre ellos. Resulta imposible imaginar la sociedad sin pensar en los seres humanos y, a la inversa, tampoco se puede pensar en los individuos sin considerar la sociedad. Allí donde encontremos seres humanos los hallaremos viviendo en sociedad, esto implica necesariamente la palabra comunicación que según la Real Academia Española significa “*Acción y efecto de comunicar o comunicarse*”, esta necesidad se materializó de diferentes maneras a lo largo de la historia de la humanidad.

Desde la antigüedad existieron sistemas de comunicación a distancia, mas o menos perfeccionados pero recién a principios del siglo XIX a partir de la revolución Francesa en donde surge el telégrafo óptico se pudo observar un avance significativo. Este evento puede indicarse como el comienzo de la telecomunicación como sistema organizado, brindando medios de comunicación a los gobiernos.

En 1857 se inicia la telegrafía eléctrica por compañías privadas en la Argentina, en la década de los años sesenta del siglo XIX se empezaron a tender cables submarinos intercontinentales .

Cuando en 1876 aparece el teléfono, la sociedad tenía satisfechas sus necesidades de comunicación por el telégrafo, por lo cual fue considerado al principio como un juguete o símbolo de lujo, no obstante pronto se le encontró aplicación como medio fácil de comunicación dado que para su funcionamiento no se precisaba de un especialista en



comunicaciones; las primeras redes fueron inauguradas por sociedades municipales o compañías privadas en Londres y París en 1879.

Ya en la década de los años sesenta la tecnología había logrado aportar prácticamente todo lo que la sociedad demandaba; gracias a las microondas las posibilidades de circuitos telefónicos eran enormes y estas posibilidades de la tecnología habían ampliado el alcance de las comunicaciones y por lo tanto su ámbito geográfico, adquiriendo de esta manera un carácter internacional.

Pasando por la red conmutada automática, la transmisión de datos y el fax llegamos a los años 1969, en donde Estados Unidos por medio del Departamento de Defensa, a través de ARPA crea una red experimental de conmutación de paquetes utilizando las líneas telefónicas, con el tiempo esta red sería el primer paso a Internet.

Casi al mismo tiempo que se producía todo este proceso, la tecnología del software o de la informática, es decir la tecnología digital era aplicada a la red telefónica principalmente por carácter económico para reducir costos, comenzó la conmutación, la razón principal fue el menor espacio ocupado por los equipos para atender a un mismo número de abonados; su implementación no fue fácil, por una parte era necesario convertir las señales de analógicas a digitales y viceversa, a la entrada y salida de las centrales, ya que los medios de transmisión seguían siendo analógicos.

El Departamento de Defensa de los Estados Unidos encomendó a un equipo de universitarios californianos el diseño de una red experimental que conectara los centros de investigación que trabajaban para el Pentágono. La primera demostración pública de ARPAnet se realizó en 1972 conectando 40 ordenadores, y al parecer fue recibida con escepticismo por algunos de los representantes de compañías comerciales presentes en la histórica ocasión.

A finales de 1981, en Estados Unidos, coincidieron dos circunstancias que habrían de ser cruciales para el fenómeno de Internet, por un lado la rescisión de las actividades militares de la red ARPAnet, dejando en manos de la National Science Foundation el desarrollo de una red de interés puramente académico. Por otro, la adopción del protocolo



TCP/IP (Transmission Control Protocol/ Internet Protocol) – diseñado para hacer posible la interconexión universal de distintas plataformas de ordenadores (desde los grandes equipos hasta los recientes PCs). Con ello se pretendía que las diferentes redes académicas de investigación pudieran trabajar en común, formando una red virtual que dio en llamarse internetwork o, más sencillamente, Internet.

El protocolo TCP/IP es una peculiar aplicación del concepto de conmutación de paquetes, originado en los años 60, para optimizar la utilización de las líneas, aprovechando los tiempos muertos, consiste en dividir la masa de datos a enviar en pequeños “paquetes” que pueden tomar diferentes rutas hasta su destino, donde se recomponen en su formato original.

En 1981 el número de ordenadores conectados a la incipiente Internet había subido a 213, gracias a la autonomía adquirida por las aplicaciones civiles – correo electrónico, foros de discusión, intercambio de información, transferencia de archivos - y en los dos años siguientes llegaría a 1.000, que se convertirán en 100.000 en 1989, en 1 millón en 1992 y en 56 millones antes de acabar el siglo XX.

En la actualidad contamos con 850 millones de computadoras personales en todo el mundo, mientras que los teléfonos de línea fija llegan a 1.300 millones a nivel mundial a este último grupo debemos sumarle 3.500 millones de líneas activas de celular en el mundo, podemos observar con estas cifras lo que significa estar comunicado.

1.2 Redes telefónicas IP

La telefonía IP² es un servicio que permite realizar comunicaciones de voz sobre redes basadas en el protocolo Internet. La principal diferencia con la telefonía tradicional, que funciona mediante conmutación de circuitos es que la telefonía IP hace uso de la conmutación de paquetes, esto es, transmite conversaciones como paquetes de datos. Se trata de transportar la voz, previamente convertida a datos, entre dos puntos distantes.

Esto hace posible utilizar las redes de datos para efectuar las llamadas telefónicas y desarrollar una única red convergente que se encargue de cursar todo tipo de

2 Internet Protocol



comunicación, ya sea voz, datos, vídeo o cualquier tipo de información. Además de reemplazar la funcionalidad de la telefonía tradicional a un menor costo, compite con las redes de telefonía móvil ofreciendo nuevas posibilidades de acceso a través de WiFi³ y WiMax⁴.

La infraestructura IT esta sufriendo cambios. Por ejemplo, los productos de software libre compiten cada vez con mejor solvencia contra productos propietarios. Internet se ha vuelto a convertir en un mercado atractivo para los inversores y se crean aplicaciones centradas en los usuarios que interactúan con otros usuarios o con aplicaciones de terceros entre si a través de servicios web.

Por último han aparecido empresas que actúan como integradores de aplicaciones que hacen uso de la red telefónica. En conjunto todos estos cambios han modificado drásticamente el entorno de los operadores telefónicos tradicionales.

Por todo ello los operadores se enfrentan al reto de migrar sus redes y servicios a IP. Hacerlo les dará nuevas posibilidades de beneficio, nuevos servicios, un menor tiempo desde que los productos son desarrollados hasta que llegan al mercado y enlazar a los operadores telefónicos con las comunidades de desarrollo IT. La tecnología para hacerlo posible se basa en un conjunto de estándares abiertos que forman un conjunto de APIs⁵ que permitirán la creación de los servicios de nueva generación.

1.3 Voz Sobre IP

La Voz sobre IP es una tecnología que permite la transmisión de la voz a través de redes IP en forma de paquetes de datos.

La Telefonía IP es una aplicación inmediata de esta tecnología, de forma que permita la realización de llamadas telefónicas ordinarias sobre redes IP u otras redes de

3 WiFi es un sistema de envío de datos sobre redes computacionales que utiliza ondas de radio en lugar de cables

4 WiMAX, acrónimo de Worldwide Interoperability for Microwave Access

5 Application Programming Interface



paquetes utilizando un PC, gateways⁶ y teléfonos estándares. En general, servicios de comunicación - voz, fax, aplicaciones de mensajes de voz - que son transportadas vía redes IP, Internet normalmente, en lugar de ser transportados vía la red telefónica convencional.

La voz sobre IP, no es en sí mismo 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 PSTN⁷. La telefonía IP no utiliza circuitos para la conversación, sino que envía múltiples comunicaciones codificadas a través de un único canal en paquetes y flujos independientes. Cuando se produce un silencio en una conversación no se transmite ningún paquete de ese flujo pero se aprovecha para transmitir paquetes de datos de otras conversaciones. Esto implica un uso más eficiente de la red de telecomunicaciones.

Las ventajas son que hay ahorro de costos debido a utilizar una sola red que puede ser mantenida centralizadamente para transmitir voz y datos, especialmente cuando los usuarios tienen redes que pueden usar para transportar la voz sobre IP sin un costo adicional.

Las llamadas de VoIP a VoIP internas hacen uso de la red de comunicaciones de datos y por tanto son gratuitas, con lo que se puede ahorrar en las llamadas entre departamentos o a trabajadores móviles. Solamente las conexiones entre la red telefónica tradicional y la red VoIP pueden llevar asociado un costo.

Por otro lado las desventajas parten de que es complicado diseñar una red que pueda asegurar fiabilidad y calidad de servicio para la telefonía porque el protocolo de Internet no fue diseñado para comunicaciones en tiempo real y siempre existe la posibilidad de que los paquetes se pierdan en la red y nunca lleguen al destino o bien que lleguen demasiado tarde.

En una implementación de VoIP básica no es posible determinar el tiempo que lleva el paquete dentro de la red ni su contenido, ni asegurar su integridad y es posible interceptar, pinchar e interferir comunicaciones. Los delincuentes también pueden suplantar la

⁶ Gateway: Equipo para interconectar redes

⁷ PSTN: Public switched telephone network



identidad de la persona que llama para cometer fraudes.

Por otro lado la voz sobre IP es una tecnología no regulada. Aunque se está empezando a legislar al respecto, los usuarios aún están expuestos a algunas vulnerabilidades y al riesgo de sufrir ciertos engaños especializados.

1.4 Arquitectura de VOIP

Necesariamente tenemos que distinguir entre lo que son los protocolos de señalización, los protocolos de transporte y los protocolos de datos transmitidos que están codificados con un codec⁸ de audio/video estandarizado por los distintos organismos (por ejemplo G711, G723, G729, GSM) o propietarios como el que utiliza skype en donde los usuarios interesados en usarlos deben bajarse una aplicación desde el sitio oficial correspondiente.

Existen varios protocolos de señalización comúnmente usados para VOIP, estos protocolos definen la manera en que por ejemplo los codecs se conectan entre sí y hacia otras redes usando VoIP. Estos también incluyen especificaciones para codecs de audio.

- **Protocolo H.323:** Es una recomendación del ITU-T⁹, que define los protocolos para proveer sesiones de comunicación audiovisual sobre paquetes de red.
- **Protocolo SIP¹⁰:** Es un protocolo desarrollado por el IETF MMUSIC Working Group con la intención de ser el estándar para la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como el video, voz, mensajería instantánea, juegos online y realidad virtual.

⁸ Garantizan la codificación y compresión del audio o del video

⁹ ITU: International Telecommunication Union

¹⁰ SIP: Session Initiation Protocol



- **IAX2¹¹**: Es uno de los protocolos utilizado por Asterisk, un servidor PBX de código abierto patrocinado por Digium. Es utilizado para manejar conexiones VoIP entre servidores Asterisk, y entre servidores y clientes que también utilizan protocolo IAX. 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.

El protocolo de transporte RTP¹² Es un protocolo de nivel de transporte utilizado para la transmisión de información en tiempo real, como por ejemplo audio y vídeo en una video-conferencia. Está desarrollado por el grupo de trabajo de transporte de Audio y Video del IETF¹³, publicado por primera vez como estándar en 1996 como la RFC 1889, y actualizado posteriormente en 2003 en la RFC 3550.

Hay otros protocolos en el mercado como por ejemplo el de **Skype**¹⁴ el cual es el servicio de VoIP mas usado en el mundo permitiendo que las personas puedan establecer las llamadas en forma gratuita entre ordenadores y haciendo un pago del servicio si desean comunicarse con un usuario que este conectado a la red de telefonía convencional PSTN.

La comunicación de voz es analógica, mientras que la red de datos es digital. El proceso de convertir ondas analógicas a información digital se hace con un codificador-decodificador (el codec). Hay muchas maneras de transformar una señal de voz analógica, todas ellas gobernadas por varios estándares. El proceso de la conversión es complejo. Es suficiente decir que la mayoría de las conversiones se basan en la modulación codificada mediante pulsos (PCM¹⁵) o variaciones.

Además de la ejecución de la conversión de analógico a digital, el CODEC comprime la secuencia de datos, y proporciona la cancelación del eco. La compresión de la forma de onda representada puede permitir el ahorro del ancho de banda. Esto es especialmente interesante en los enlaces de poca capacidad y permite tener un mayor número

¹¹ IAX: Inter-Asterisk eXchange version 2

¹² Real-time Transport Protocol

¹³ Internet Engineering Task Force

¹⁴ Es un software para realizar llamadas sobre Internet

¹⁵ Pulse Code Modulation



de conexiones de VoIP simultáneamente. Otra manera de ahorrar ancho de banda es el uso de la supresión del silencio, que es el proceso de no enviar los paquetes de la voz entre silencios en conversaciones humanas.

A continuación se muestra una tabla resumen con los códecs más utilizados actualmente:

Codec	Bandwidth (kbps)	Nominal Bandwidth (kbps)
G.711	64 Kbps	87.2 Kbps
G.729	8 Kbps	31.2 Kbps
G.723.1	6.3 Kbps	21.9 Kbps
G.723.1	5.3 Kbps	20.8 Kbps
G.726	32 Kbps	55.2 Kbps
G.726	24 Kbps	47.2 Kbps
G.728	16 Kbps	31.5 Kbps
GSM	7 or 13 kbps	low
ILBC	15 Kbps	27.7 Kbps

Cuadro comparativo de codecs de compresión

Para el proyecto se instalaron dos licencias de codec G.729, estas licencias se adquirieron de Digium¹⁶, esta empresa ofrece una implementación del código G.729 que es compatible con el servidor Asterisk usado para el proyecto, de esta manera se logro una optimización del ancho de banda con una calidad aceptable.

¹⁶ <http://www.digium.com>



Capítulo II

2. El Mercado

El mercado de la VoIP es distinto en las diferentes partes del mundo. La principal diferencia es que en Europa las operadoras tradicionales están jugando un papel importante en el desarrollo de la VoIP mientras que en América las operadoras están tratando de pararlo.

En 2006 las operadoras europeas perdieron 10 millones de líneas fijas. Se crearon 14 millones de líneas VoIP de las cuales 3 millones fueron de las operadoras. En Estados Unidos se perdieron 4.5 millones de líneas fijas y se crearon 4.4 millones de líneas VoIP de las cuales solamente 300000 pertenecen a las operadoras tradicionales.

Además en Estados Unidos la VoIP tiene la misma apariencia de cara al usuario que el teléfono fijo. Los mayores proveedores son los operadores de cable y no tienen incentivos para competir con las empresas de telecomunicaciones. Solamente tienen que hacer de la VoIP un servicio igual de fiable y ligeramente más barato. En Europa la VoIP exprime todas sus posibilidades. Se ofrecen paquetes de llamadas gratuitas a paquetes de países o se ofrecen servicios de libretas de direcciones online.

La legislación en Europa permite a los proveedores de Internet de banda ancha pagar precios competitivos para usar las líneas de las operadoras dominantes para ofrecer sus propios servicios. En Estados Unidos no existe esta legislación porque el gobierno lo ve como un obstáculo para que las operadoras inviertan en sus despliegues de fibra hasta los hogares.

Lo que si es común en todo el mundo es que cada vez hay más interés en la convergencia de las aplicaciones VoIP. La opinión general es que la VoIP en las empresas aumenta la productividad y hace que sean más competitivas en un mercado habitualmente saturado.

VoIP es un gran negocio para las empresas pero dónde existe un mayor crecimiento es en los hogares. Hay que recordar que SIP no es el protocolo VoIP dominante.



Skype, una empresa que ofrece gratuitamente los servicios de VoIP a todo el mundo, cuenta con cerca del 45 % del tráfico VoIP en estados unidos. Skype tiene 18 millones de suscriptores y los usuarios de Skype hicieron 5000 millones de minutos de llamadas en solo pocos meses. Por eso no son las operadoras tradicionales las más importantes en el sector VoIP sino que el crecimiento real de la VoIP viene de la mano de comunidades online como Skype y Google talk.

Un factor importante es que son servicios sencillos de instalar y baratos, pero además las comunidades online son virales. Esto quiere decir que cuando un usuario contrata una linea SIP, es un proceso transparente y el único efecto visible es que ese usuario reduce su factura telefónica. Sin embargo cuando un usuario se inscribe en servicios como Skype o Google Talk tiende a enviar mensajes para que sus amigos también se suscriban para ahorrar dinero.

2.1 Casos de negocio exitosos que ofrecen servicios VOIP

Hay varias compañías que lograron con éxito brindar servicios VoIP a los usuarios del mundo, normalmente unifican los siguientes servicios: VoIP en líneas múltiples y conferencias web, conferencias a números de la red telefónica tradicional, soporte de uno o varios protocolos de mensajería instantánea, envío de SMS, de archivos y de correo electrónico.

- **Iplan¹⁷**



Figura 1.1: Logotipo de IPLAN

¹⁷ [Http://www.iplan.com.ar](http://www.iplan.com.ar)



Iplan provee un servicio llamado VoIPack, es un servicio de Telefonía que permite la conexión a la PSTN con numeración pública asignada por la CNC¹⁸.

El usuario interesado puede contratar el plan y accediendo mediante una interfaz Ethernet y usando el protocolo SIP puede acceder a cualquier número telefónico de la red PSTN¹⁹.

- **Skype**²⁰



Figura 1.2: Logotipo de Skype

Es un software para realizar llamadas sobre Internet (VoIP), fundada en 2003 por los suecos Niklas Zennström y Janus Friis, los creadores de Kazaa. El código y protocolo de Skype permanecen cerrados y propietarios, pero los usuarios interesados pueden descargar gratuitamente la aplicación del sitio oficial. Los usuarios de Skype pueden hablar entre ellos gratuitamente si la comunicación es entre los ordenadores, pero si la comunicación tiene como destino un teléfono de la red publica PSTN tienen que pagar un servicio.

- **Jajah**²¹



Figura 1.3: Logotipo de Jajah

Es el nombre de un Software propietario de ayuda al establecimiento de llamadas telefónicas utilizando el sistema VoIP y de la empresa propietaria de dicho Software.

18 Comisión Nacional de Comunicaciones

19 Public Switched Telephone Network

20 <http://www.skype.com>

21 <http://www.jajah.com>



Esta herramienta establece la comunicación telefónica entre líneas fijas normales y teléfonos móviles sin tener que descargar ningún software, sin instalación y sin auriculares.

Las llamadas se activan desde el ordenador personal conectado a Internet, pero todo el flujo de la comunicación se hace entre teléfonos "normales". Una vez establecida la comunicación entre los teléfonos la conexión a Internet no es necesaria.

- **QuteCom²²**



Figura 1.4: Logotipo de QuteCom

Es el nuevo nombre de WhengoPhone, clon de skype que brinda un cliente softphone de código abierto.

- **Gizmo²³**



Figura 1.5: Logotipo de Gizmo

Clon de Skype pero usando los estándares abiertos P2P SIP y Jingle para la señalización y gestión de llamadas, utiliza codecs de audio propietarios y como Skype el cliente Gizmo es propietario.

²² <http://www.qutecom.org/>

²³ <http://gizmo5.com/>

- **Google Talk²⁴**



Figura 1.6: Logotipo de GoogleTalk

Es un programa cliente de mensajería instantánea y VoIP de protocolo Jabber²⁵, desarrollado por Google. La versión beta de Google Talk fue lanzada el 24 de agosto de 2005.

Google Talk se sustenta bajo el protocolo de interoperabilidad de Jabber y XMPP, siendo configurable en programas como Psi, Miranda IM, iChat y Pidgin (anteriormente llamado Gaim), entre otros. Para que un cliente Jabber se pueda conectar necesita cifrado TLS y autenticación SASL PLAIN a través del puerto 5222.

El servicio está disponible para los usuarios de Gmail. Actualmente el registro es abierto.

2.2 Costo publicitario en Argentina

En un mercado competitivo, una de las herramientas con la que cuentan las empresas para posicionarse es hacer una buena campaña de prensa. A continuación se describen los distintos costos, medios y estrategias con la que cuentan las PYME a la hora de decidir en que medio contratar las campañas.

A la hora de armar una campaña publicitaria es muy importante poder definir cual es el tipo de público al cuál se desea llegar, de dirigirla al público adecuado dependerá el éxito de la misma.

La televisión y los diarios son los medios más eficaces, pero según la empresa que necesita contratar el servicio publicitario puede elegir un medio gráfico ya sea en la vía pública o en un diario o revista para poder llegar más eficazmente al nicho de público que le interesa; para saber entonces donde invertir los especialistas indican que el cálculo debe pasar

²⁴ <http://www.google.es/talk>

²⁵ Jabber es un protocolo libre para mensajería instantánea, basado en el estándar XML y gestionado por XMPP Standards Foundation.



por el costo por contacto, es decir cual es el costo asociado por persona que recibe la publicidad..

2.2.1 Costo de la publicidad en medios gráficos

Las opciones son muchas si se quiere contratar un espacio por ejemplo en una revista dominical como La Nación Revista, los costos dependen del lugar de publicación, ya sea en la tapa que llaman cuatríptica zigzag que ocupa 4 páginas en donde el costo es de \$279.000, en una página interior el costo es de \$39.500 o por ejemplo en la contratapa el costo de publicación es de \$68.900. Esta revista vende promedio 370.000 ejemplares los días domingo.

2.2.2 Costo de la publicidad en TV

Con niveles de audiencia imposibles de conseguir por otros medios, la televisión es el espacio preferido de los grandes anunciantes. Por eso estar allí implica un desembolso de recursos importante. Los costos de las tandas son hasta 35 veces más caros que la radio.

Para entender el cuadro que expresa la figura 2.2, decimos que un punto de rating representa el 1 % (uno por ciento) del target elegido. Teniendo en cuenta los datos poblacionales del 2007 y considerando que el total de hogares de Capital y GBA es de 3.070.600 hogares, 1 punto de rating en Capital y GBA = 30.706 hogares. Considerando que el total de individuos de 4 a 99 años de Capital y GBA es 9.678.200 individuos, 1 punto de rating en Capital y GBA = 96.782 individuos. La cantidad de personas que representan uno o más puntos de rating varía de acuerdo al target considerado.



Programa	Costo por segundo en \$	Rating	Llegada en cantidad de personas	Costo por contacto x segundo
Showmatch	\$4.400	25,5	2.467.941	\$0,001782
CQC	\$4.500	13	1.258.166	\$0,003576
Susana Giménez	\$3.000	23,5	2.274.377	\$0,001319
Son de Fierro	\$3.300	21,2	2.051.778	\$0,001608
Por amor a vos	\$3.300	21,9	2.119.525	\$0,001556

Figura 2.2: Tabla rating de TV según IBOPE Argentina

2.2.3 Costo de la publicidad en radio

En radio cada punto de rating representa el 1 % (uno por ciento) de la población relevada. Considerando que el total individuos de 12 a 74 años es de 7.958.574 individuos, 1 punto de rating en Capital y GBA equivale a 79.586 individuos. La cantidad de personas que representan uno o más puntos de rating varía de acuerdo al target considerado.

Programa	Costo por segundo en \$	Rating	Llegada en cantidad de personas	Costo por contacto x segundo
Rock & Pop	\$125	1,38	108.449	\$0,001152
Metro	\$76	0,95	75.606	\$0,001005
Mega	\$80	1,26	100.278	\$0,000797

Figura 2.3: Tabla rating de radio según IBOPE Argentina

2.2.4 Costo de la publicidad en HableGratis.NET

El costo de una llamada telefónica según las compañías son: Telefónica cobra por llamadas locales 22 centavos por minuto mientras que Telecom las factura 18 centavos el minuto.

Una línea telefónica VoIP contratada por **HableGratis.NET** tiene una reducción de hasta un 90% al costo de una llamada que tiene una línea telefónica



convencional, estos valores se hacen mucho mas evidentes si las llamadas son de larga distancia..

Publicar en el servicio **HableGratis.NET** tendrá un costo publicitario de \$0,0286²⁶ por minuto de llamada, cabe destacar que esta publicidad será recibida por 2 personas que son las que intervienen en la comunicación, en conclusión el costo publicitario de publicar en **HableGratis.NET** será de \$0,0143 por minuto y por persona.

Medio	Duración e la Publicidad	Costo por segundo por contacto	Costo por 5 segundos por contacto
Televisión	5 segundos	\$0,00196	\$0,009841
Radio	5 segundos	\$0,000984	\$0,00492
Gráfica(Página interna)	1 Página	\$0,1837	\$0,1837
HableGratis.NET	5 segundos	---	\$0,0143

Figura 2.4: Cuadro comparativo promedio entre los distintos medios estudiados

Si bien el costo de publicar en **HableGratis.NET** es levemente superior a publicar en los demás medios, exceptuando el medio gráfico(Ver cuadro 2.4) , hay que destacar que la ventaja importante de publicar en **HableGratis.NET** es que las personas escucharan el mensaje publicitario si o si dado que no pueden hacer “zapping²⁷” mientras se establece la comunicación, esto es algo difícil que se de en otros medios publicitarios.

²⁶ Calculo compuesto por el costo de una llamada VoIP que ronda los \$0,022 multiplicado por un 130%

²⁷ Acción de cambiar de canal de radio o TV



Capítulo III

3. Construcción del Prototipo

3.1 Construcción de la aplicación de comunicaciones integradas con tecnología VOIP orientadas a la web.

El objetivo de la aplicación es hacer una integración entre la web y la telefonía, permitiendo que dos personas puedan establecer una comunicación sin la necesidad de instalar ninguna aplicación en sus computadoras ni realizar ningún tipo de configuración, esta implementación puede aplicarse también para una empresa que quiera comunicar a su vendedor con su cliente, esta última opción quedara fuera del prototipo y pasará a una oportunidad de mejora .

El sistema esta compuesto por dos interfaces, una es la interfaz en la cual el usuario final desea establecer la llamada y la otra es la interfaz administrativa la cual le permite al administrador del sistema observar los registros de las llamadas que fueron realizadas.

El funcionamiento para el usuario final es el siguiente:

1. La persona interesada en establecer la comunicación ingresa a **HableGratis.NET**
2. Ingresa su número telefónico y el número telefónico de la persona a la cual desea llamar.
3. Cuando la persona presiona en el botón llamar el servidor establece la comunicación entre los dos teléfonos, previamente envía un mensaje publicitario mediante el cual la empresa que implementa el servicio gratuito hará frente a los costos de la llamada en los cuales esta pasa por la PSTN o compañía celular.

El sistema desarrollado elimina la necesidad del uso de teléfonos IP, solo se necesita tener acceso a Internet para poder completar el formulario de ingreso de los dos teléfonos que se desean comunicar.



3.2 Interfaz de usuario final

HableGratis.NET | Una nueva forma de comunicarse

Llamar **Solución**
Soporte
Servicios **Contacto**

Establecer Llamada **Forma de llamar** **Trabajo Final**

Su Número de Teléfono:

Número de Teléfono destino:

[Borrar](#) [Llamar](#)

Forma de llamar

Siga las siguientes instrucciones para establecer la llamada

- Ingrese su número telefónico
- Ingrese el número telefónico de la persona a la cual desea comunicarse
- Presione el boton "Llamar"
- Su telefono comenzara a sonar
- Luego de atender escuchara el anuncio del servicio + una breve publicidad
- El teléfono de la persona con quien desea comunicarse comenzara a sonar
- Luego de que la persona atienda la llamada escuchara un anuncio + una breve publicidad
- Pueden comenzar a hablar gratuitamente!

Trabajo Final de Ingeniería

- Este es un trabajo final para la carrera Ingeniería en Informática de la UADE
- El proyecto cuenta con un servidor Asterisk configurado con una linea zap y dos lineas VOIP SIP. Este servidor se es utilizado como backend por el sistema desarrollado.
- EL sistema fue desarrollado en Java EE utilizando la API Asterisk-Java para la comunicación entre el sistema que controla la llamada y el servidor de comunicaciones Asterisk
- La base de datos utilizada para el CDR fue Mysql version 5.1.23rc-1
- A lo largo del trabajo se utilizaron herramientas como JBossIDE for Eclipse, GIMP, UML

Copyright © 2008, Julián Rousselot, [Privacy Policy](#)

Figura 3.1: Página del sitio HableGratis.NET



Para el diseño de la interfaz del usuario final se utilizó HTML y CSS para el estilo de la página, también se construyó una versión hecha en flash para darle un mejor impacto visual.

La aplicación es compatible con los navegadores modernos que soportan estas tecnologías.

La interfaz de usuario incluye lo siguiente:

- Una botonera de navegación por las distintas secciones como son:
- **Llamar:** Luego de ingresar el número de teléfono origen y el número de teléfono destino se presiona el botón llamar y se establece la comunicación (Ver figura 3.2).

Figura 3.2: Formulario de ingreso de datos para la llamada

- **Servicios:** Esta sección del menú describe los distintos tipos de servicios que provee **HableGratis.NET**.
- **Soporte:** Aquí se ofrecen indicaciones de como utilizar los servicios.
- **Solución:** En esta opción del menú el usuario podrá consultar sobre soluciones a medida para los distintos tipos de servicio e implementaciones de VoIP.

- **Contacto:** Haciendo click en esta opción el usuario puede comunicarse dejando un mensaje con **HableGratis.NET**.



Figura 3.3: Opciones de la página del usuario final

3.3 Caso de uso del usuario final

El siguiente caso de uso1 muestra como el usuario establece la llamada

Objetivo	Establecer la llamada
Acción	Un usuario necesita hacer una llamada telefónica
Actor Principal	El usuario interesado en hacer una llamada

Paso	Acción
1	El usuario ingresa a la sección del menú Llamar
2	El usuario ingresa su número telefónico
3	El usuario ingresa el número telefónico de la persona a la cual desea llamar
4	El usuario presiona el botón Llamar



3.4 Interfaz administrativa

Al igual que la interfaz del usuario final, para el diseño de la interfaz del administrador se utilizó HTML y CSS para el estilo de la página.

La aplicación es compatible con los navegadores modernos que soportan estas tecnologías.

Ingresando a esta página el usuario administrador podrá ver todos los registros que se crearon a raíz de las llamadas realizadas.

La interfaz de usuario administrador incluye lo siguiente:

- **Datos referidos a la publicidad anunciada en la llamada:** El administrador podrá observar que publicidad fue anunciada, el género de la misma, la dirección en donde esta guardada y la duración de la misma (Ver figura 3.4).

Reporte CDR							
Nro.	Fecha	Origen	Destino	Estado	Anuncio	Publicidad	Durac. Publ.
1	2008-04-18 20:48:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
2	2008-04-18 20:49:52	1551231234	1551231232	NO ANSWERED	Bienvenido	Linux OS	5s
3	2008-04-18 20:50:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
4	2008-04-18 20:51:52	1551231234	1551231232	NO ANSWERED	Bienvenido	Linux OS	5s
5	2008-04-18 20:52:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
6	2008-04-18 20:53:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
7	2008-04-18 20:54:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
8	2008-04-18 20:55:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
9	2008-04-18 20:56:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
10	2008-04-18 20:57:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
11	2008-04-18 20:58:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
12	2008-04-18 20:59:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
13	2008-04-18 21:00:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
14	2008-04-18 22:01:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
15	2008-04-18 21:02:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
16	2008-04-18 21:03:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
17	2008-04-18 21:04:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
18	2008-04-18 21:05:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s

Figura 3.4: Registro de las llamadas

- **Datos referidos al registro de las llamadas:** El administrador podrá observar el origen e la llamada, el destino, el resultado ya sea exitoso o con error de la comunicación, el anuncio dado a los interlocutores, la duración de la publicidad, la



fecha y la publicidad enviada(Ver figura 3.4).

- **Exportación de datos:** El administrador podrá exportar los datos a formato CSV o XML para su posterior manipulación(Ver figura 3.5).

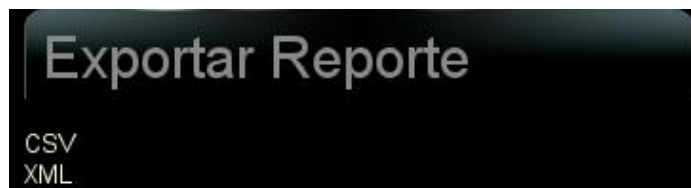


Figura 3.5: Opciones de exportación



3.5 Página para el usuario administrador

HableGratis.NET
Administrator

Buscar

Home Reporte CDR Contactos

Reporte CDR

Nro.	Fecha	Origen	Destino	Estado	Anuncio	Publicidad	Durac. Publ.
1	2008-04-18 20:48:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
2	2008-04-18 20:49:52	1551231234	1551231232	NO ANSWERED	Bienvenido	Linux OS	5s
3	2008-04-18 20:50:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
4	2008-04-18 20:51:52	1551231234	1551231232	NO ANSWERED	Bienvenido	Linux OS	5s
5	2008-04-18 20:52:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
6	2008-04-18 20:53:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
7	2008-04-18 20:54:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
8	2008-04-18 20:55:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
9	2008-04-18 20:56:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
10	2008-04-18 20:57:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
11	2008-04-18 20:58:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
12	2008-04-18 20:59:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
13	2008-04-18 21:00:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
14	2008-04-18 22:01:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
15	2008-04-18 21:02:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
16	2008-04-18 21:03:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
17	2008-04-18 21:04:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s
18	2008-04-18 21:05:52	1551231234	1551231232	ANSWERED	Bienvenido	Linux OS	5s

Exportar Reporte

CSV
XML

Home | Reporte CDR

© 2008 Julián Rousselot. All Rights Reserved.

Figura 3.6: Página administrativa del sitio HableGratis.NET



3.6 Caso de uso del usuario administrador

El siguiente caso de uso²⁸ muestra como el usuario administrador exporta un registro de Llamadas

Objetivo	Exportar registro de llamadas
Acción	El usuario administrador necesita exportar a un formato CSV o XML el registro de llamadas
Actor Principal	El usuario administrador

Paso	Acción
1	El usuario administrador selecciona la fecha en la cual esta interesada de obtener un reporte del registro de llamadas.
2	El usuario administrador presiona el icono de exportar registro
3	El usuario administrador selecciona el destino del archivo de registro de llamadas

3.7 Arquitectura del sistema de comunicaciones integradas

La arquitectura para la aplicación web y el servicio de telecomunicaciones que establece la llamada esta dada por tres componentes:

1. **Una aplicación o servicio:** Esta aplicación se desarrollo utilizando el lenguaje de programación java (JEE), junto con la API Asterisk-Java que permite comunicarse a alto nivel con el servidor de comunicaciones, en nuestro caso Asterisk.
2. **Una plataforma que gestiona las comunicaciones:** Se utilizo Asterisk PBX para gestionar las comunicaciones telefónicas que demanda el componente web.

²⁸ El formato de casos de uso sigue el estilo descrito en el libro Writing Effective Use Cases

3. **La base de datos:** Se utilizó la base de datos MYSQL para la persistencia de los entities.

La comunicación entre ambos componentes se realizó utilizando la API Asterisk-Java que logra comunicar vía TCP/IP a la aplicación con el servidor de comunicaciones Asterisk. (Ver figura 3.7)

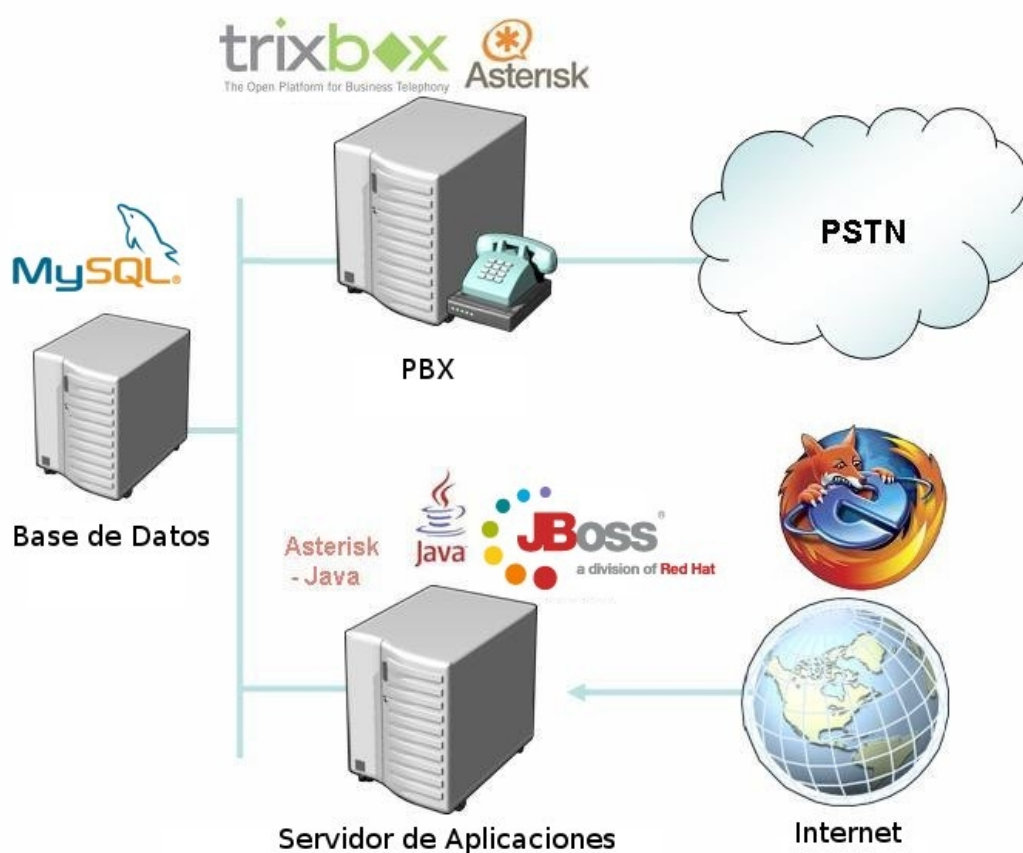


Figura 3.7: Esquema de componentes del servicio HableGratis.NET

Capítulo IV

4. Aplicación

La aplicación esta construida utilizando la tecnología JEE, utilizamos el modelo 2 de la arquitectura JSP(Ver figura 3.8), es una aproximación híbrida para servir contenido dinámico, combinando el uso de Servlets con páginas JSP tomando lo mejor de las dos tecnologías, usando las páginas JSP para generar la capa de presentación y los Servlets para realizar las tareas de procesamiento intensivo.

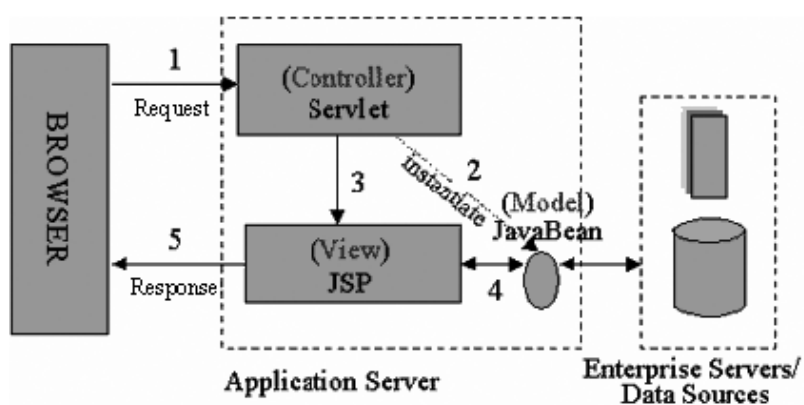


Figura 3.8: Esquema de la arquitectura JSP modelo 2

Se utilizó como servidor de aplicaciones a **JBOSS**, es un servidor de código abierto implementado en Java puro. Al estar basado en Java, JBoss puede ser utilizado en cualquier sistema operativo que lo soporte, en nuestro caso el sistema operativo usado fue GNU/Linux en su distribución Debian.

Se utilizó la API Asterisk-Java²⁹ para comunicar la aplicación web con el servidor de comunicaciones Asterisk, esta API provee todo el framework necesario para poder

²⁹ <http://asterisk-java.org/>

interactuar con el **Asterisk Manager**³⁰ y de esta manera poder tener todo el control necesario para establecer las llamadas.

La versión utilizada de Asterisk-java es la 1.0.0-m1, El API Asterisk Manager se compone de tres conceptos: las **acciones**, **respuestas** y los **eventos**. Las acciones pueden ser enviadas al Asterisk para que establezca una llamada. Por ejemplo, la aplicación puede enviar una Acción para el Asterisk que consista en marcar un número telefónico, en respuesta el servidor Asterisk envía los resultados de la operación realizada.

Para habilitar el Administrador de la API en Asterisk se debe editar el archivo `manager.conf`, la edición de este archivo se expresara en la sección en donde se describe el servidor de comunicaciones.

4.1 Diagrama de Transición de Estados (DTE)

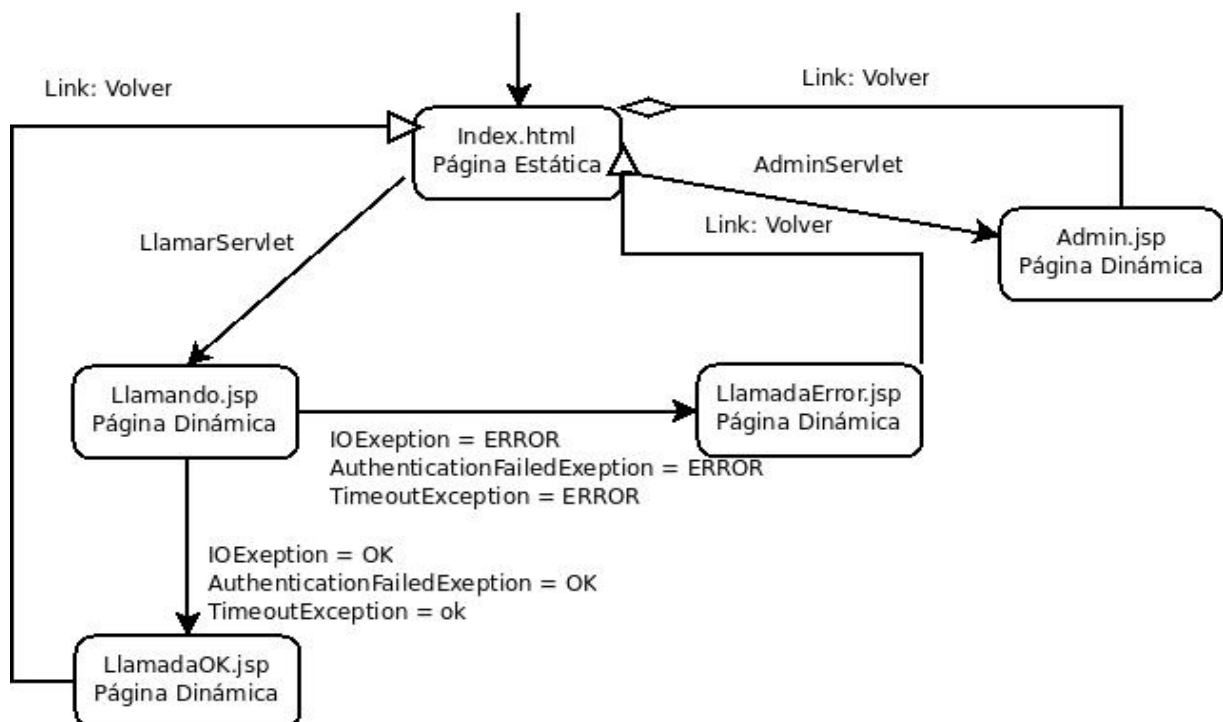


Figura 3.9: Diagrama de transición de estados

³⁰ Servicio provisto por el servidor Asterisk utilizando el puerto 5038, permite controlar el Asterisk vía TCP

4.2 Diagrama de Clases

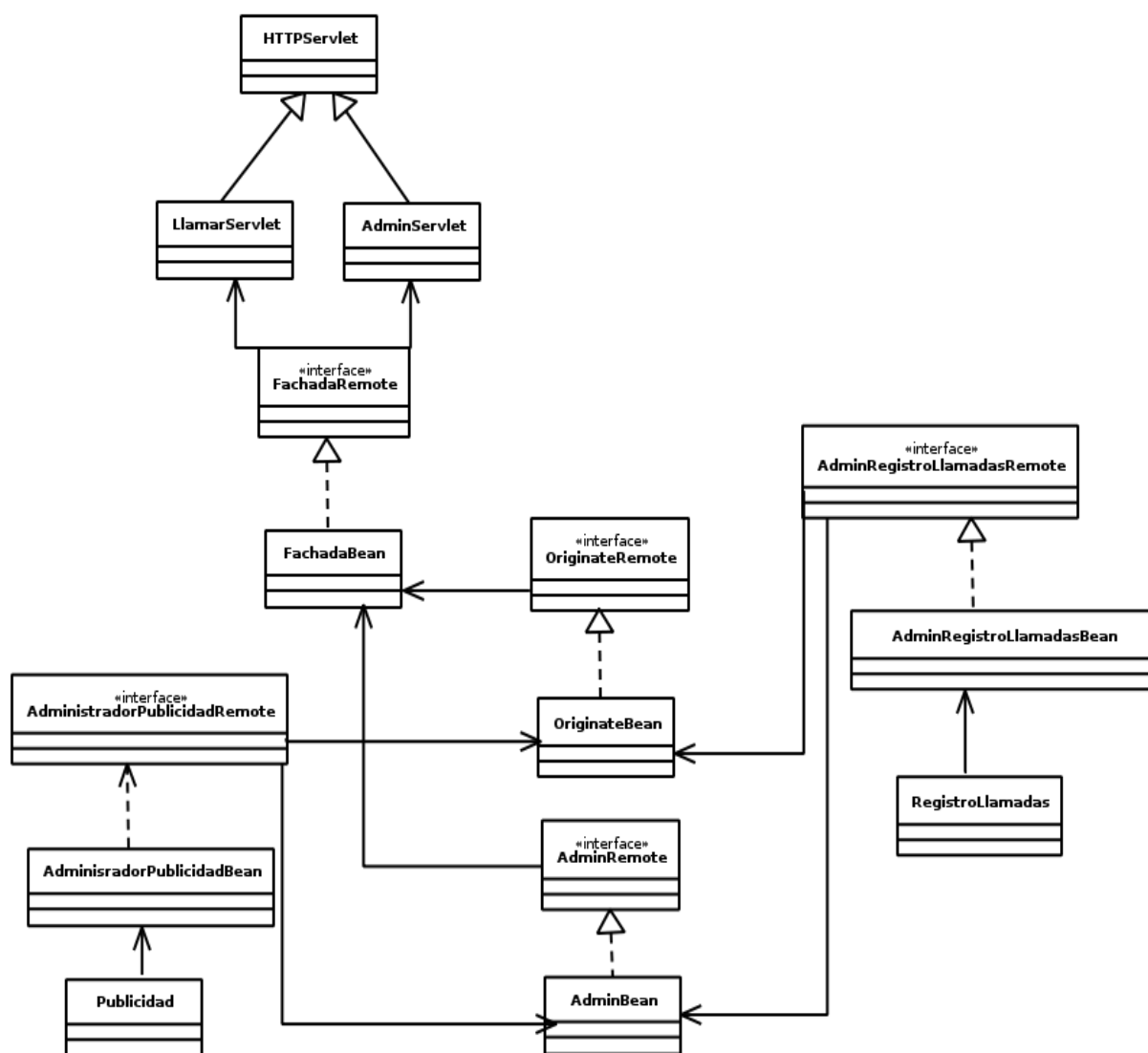


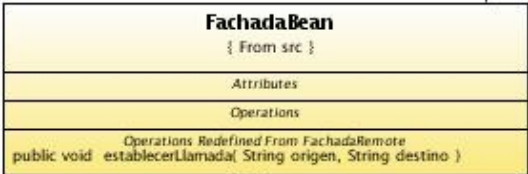



Figura 3.10: Diagrama de clases



4.3 Descripción de las clases mas importantes del sistema

Clase	Descripción
 <p>Figura 3.11: Clase LlamarServlet</p>	Servlet que recibe el HttpServletRequest para establecer la llamada.
 <p>Figura 3.12: Clase AdminServlet</p>	Servlet que recibe el HttpServletRequest para ver los registros de las llamadas.
 <p>Figura 3.13: Clase FachadaBean</p>	Es un SessionBean sin estado que tiene las acciones para establecer las llamadas y generar el reporte de los registros de las llamadas
 <p>Figura 3.14: Clase OriginateBean</p>	Es un SessionBean sin estado encargado de establecer las comunicaciones correspondientes.



AdminBean { From sessionBeans }
Attributes
private ManagerConnection managerConnection private String origen = null private String destino = null private Calendar calendario package ManagerConnectionFactory factory = new ManagerConnectionFactory("julianrousselot.dyndns.org", "tesis", "pepito77")
Operations
public AdminBean() public void run()
Operations Redefined From OriginateRemote public void establecerLlamada(String origen, String destino)

Figura 3.15: Clase AdminBean

Es un SessionBean sin estado encargado de resolver lo referente a los reportes de las llamadas.

AdministradorPublicidadBean { From administradores }
Attributes
private EntityManager entityManager
Operations
Operations Redefined From AdministradorPublicidadRemote public boolean existePublicidad(int id) public void guardarPublicidad(Publicidad publicidad) public Object obtenerPorNombre(String nombre) public Publicidad obtenerPorId(int id) public Collection obtenerRangoDePrecios(float precioDesde, float precioHasta) public Collection obtenerTodos() public void test()

Figura 3.16: Clase
AdministradorPublicidadBean

Es un SessionBean sin estado encargado de resolver todo lo referente al entity Publicidad.

AdminRegistroLlamadas Bean { From administradores }
Attributes
private EntityManager entityManager
Operations
Operations Redefined From AdminRegistroLlamadasRemote public boolean existeRegistroLlamadas(int id) public void guardarRegistroLlamadas(RegistroLlamadas registroLlamadas) public Collection obtenerPorDescripcion(String descripcion) public Publicidad obtenerPorId(int id) public Collection obtenerTodos() public void test()

Figura 3.17: Clase
AdminRegistroLlamadasBean

Es un SessionBean sin estado encargado de resolver todo lo referente al entity RegistroLlamadas.

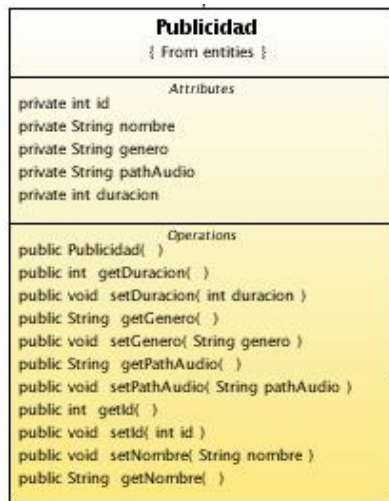


Figura 3.18: Clase Publicidad

Es un entity que contiene todos los atributos y métodos necesarios para una publicidad.

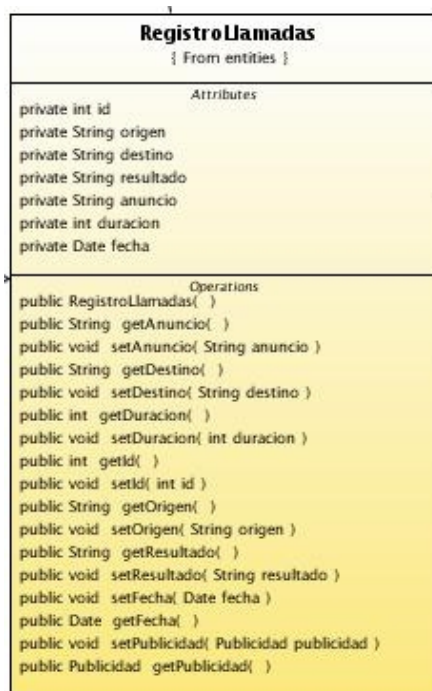


Figura 3.19: Clase RegistroLlamadas

Es un entity que contiene todos los atributos y métodos necesarios para un registro de llamada.



4.4 Función del Session Bean que establece la llamada

```
public void run() throws IOException, AuthenticationFailedException, TimeoutException{
```

```
//          Busco el entity publicidad
```

```
publicidad = (Publicidad) adminPublicidad.obtenerPorNombre("garbarino");
```

```
calendario = Calendar.getInstance();
```

```
//Creo un RegistroLlamadas
```

```
registroLlamadas = new RegistroLlamadas();
```

```
registroLlamadas.setPublicidad(publicidad);
```

```
registroLlamadas.setAnuncio("anuncio");
```

```
registroLlamadas.setDestino(destino);
```

```
registroLlamadas.setOrigen(origen);
```

```
registroLlamadas.setDuracion(publicidad.getDuracion());
```

```
registroLlamadas.setFecha(new Date());
```

```
registroLlamadas.setOrigen(origen);
```

```
try {
```

```
    OriginateAction originateAction = new OriginateAction();
```

```
    SipShowPeerAction sipShowpeerAction = new SipShowPeerAction();
```

```
    ManagerResponse originateResponse;
```

```
// DEFINITIVO
```

```
    originateAction = new OriginateAction();
```

```
    originateAction.setChannel("SIP/" + origen + concat("@CyberVOIP"));
```

```
    originateAction.setContext("hableGratis-custom");
```

```
    originateAction.setExten("foo");
```

```
    originateAction.setPriority(new Integer(1));
```

```
    originateAction.setTimeout(new Integer(30000));
```

```
    originateAction.setVariable("DESTINO", destino );
```

```
    originateAction.setVariable("PUBLICIDAD", "garbarino");
```

```
    originateAction.setVariable("ANUNCIO", "anuncio");
```

Se crea el objeto OriginateAction que es el que contiene los parametros de la



```
//Me conecto al Asterisk
managerConnection.login();

//Le mando el originate
originateResponse = managerConnection.sendAction(originateAction, 30000);
System.out.println(originateResponse.getResponse());

//Si se establecio la llamada
if(originateResponse.getResponse().equals("Success"))
    registroLlamadas.setResultado("ANSWERED");
else
    registroLlamadas.setResultado("NO ANSWERED");
adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);
managerConnection.logoff();
} catch(IOException e){
    e.printStackTrace();
    registroLlamadas.setResultado("NO ANSWERED");
    adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);

} catch (IllegalArgumentException e) {
    e.printStackTrace();
    registroLlamadas.setResultado("NO ANSWERED");
    adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);
} catch (IllegalStateException e) {
    registroLlamadas.setResultado("NO ANSWERED");
    adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);
    e.printStackTrace();
} catch (org.asteriskjava.manager.TimeoutException e) {
    e.printStackTrace();
    registroLlamadas.setResultado("NO ANSWERED");
    adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);
}
}
```



4.5 Plataforma para gestionar las comunicaciones

El sistema operativo instalado en el servidor fue GNU/Linux en su distribución llamada Debian dado que es el mejor soportado por el sistema de comunicaciones usado para el trabajo.

Como plataforma para gestionar las comunicaciones se utilizó el servidor Asterisk; es una aplicación de software libre (bajo licencia GPL) que proporciona funcionalidades de una central telefónica. Como cualquier PBX, se puede conectar un número determinado de teléfonos para hacer llamadas entre sí e incluso conectar a un proveedor de VoIP, en nuestro caso se contrataron dos líneas telefónicas VoIP que utilizan el protocolo SIP.

Asterisk fue creado por 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.

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 dialplan en lenguaje de script de Asterisk o añadiendo módulos escritos en lenguaje C o en cualquier otro lenguaje de programación soportado por Linux.

Para conectar teléfonos estándar analógicos son necesarias tarjetas electrónicas telefónicas FXS o FXO fabricadas por Digium (como la TDM400P instalada especialmente para el proyecto) 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 soporta 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.



Asterisk se empieza a adoptar en algunos entornos corporativos como una gran solución de bajo costo junto con SER³¹ (Sip Express Router).

Se eligió instalar el paquete TrixBos el cual integra el sistema operativo Linux y varios paquetes importantes para el manejo y administración del Asterisk Server.

4.6 Instalación y configuración del servidor Asterisk

Se instalo la distribución que nuclea TrixBos en un servidor Pentium III de 500 Mhz, esta distribución viene con el sistema operativo Linux, la versión instalada de TrixBos fue la 2.6.1.2.(Ver figura 4.1)

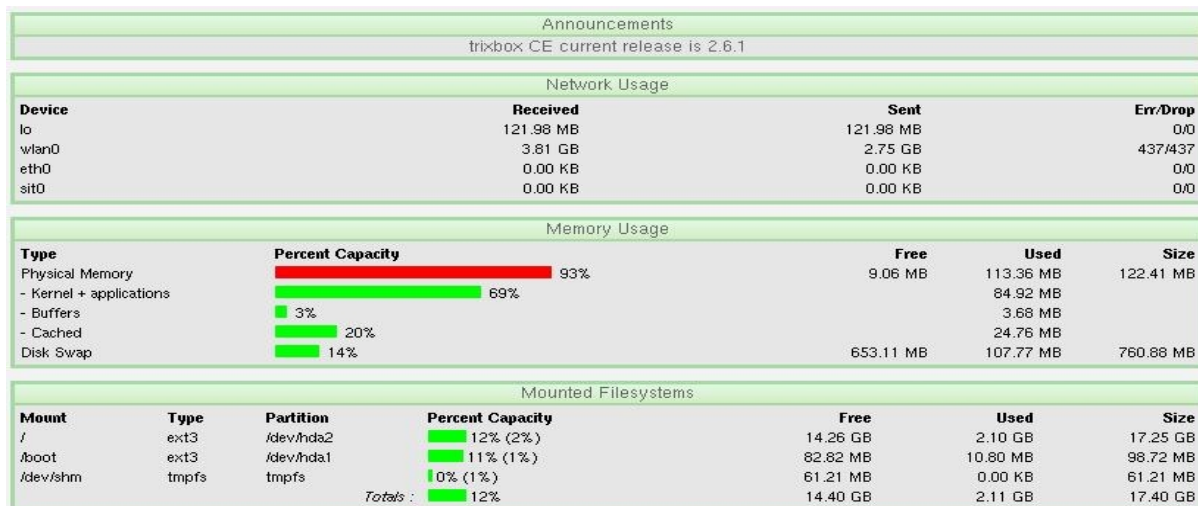


Figura 4.1: TrixBos

31 Sip Express Router



System Status

Notices

show all

Statistics

Total active calls	0
Internal calls	0
External calls	0
Total active channels	0

Connections

IP Phones Online	0
IP Trunks Online	2
IP Trunk Registrations	2

Uptime

System Uptime: 2 weeks, 4 days, 21 hours, 42 minutes
Asterisk Uptime: 1 day, 5 hours, 3 minutes
Last Reload: 1 day, 2 hours, 11 minutes



FreePBX is a registered trademark of Atengo, LLC.
FreePBX 2.4.0 is licensed under GPL

System Statistics

Processor

Load Average 0.40

CPU 1%

Memory

App Memory 70%

Swap 14%

Disks

/ 12%

/boot 10%

/dev/shm 0%

Networks

wlan0 receive 2.62 KB/s

wlan0 transmit 1.02 KB/s

eth0 receive 0.00 KB/s

eth0 transmit 0.00 KB/s

Server Status

Asterisk OK

Op Panel OK

MySQL OK

Web Server OK

SSH Server OK

Figura 4.2: Estado del servidor instalado

Se le instaló al servidor una placa analógica TDM400P con 2 módulos FXO para ser conectados a dos líneas telefónicas(Ver figura 4.3).

Si bien se instaló esta placa para ser conectada con dos líneas telefónicas convencionales se eligió contratar dos líneas VoIP para resolver las llamadas dado que fueron mas económicas.



Figura 4.3: Placa TDM400P con 2 módulos FXS y 2 módulos FXO

Se configuró la placa de la siguiente manera:

Span 1: WCTDM/0 "Wildcard TDM400P REV I Board 1" (MASTER)

fxoks=1

fxsks=2

Span 2: WCFXO/0 "Wildcard X100P Board 1" RED

fxsks=5

Global data

loadzone = us

defaultzone = us



4.7 Configuración de TrixBox

Se procedió a instalar los siguientes módulos necesarios para el buen funcionamiento del servidor de comunicaciones Asterisk: **SACAR ESPACIOS?**

	dialog	glibc-headers	ircd-hybrid	libattr
GConf2	diffutils	gmp	isdn4k-utils	libbonobo
MAKEDEV	distcache	gnome-keyring	ire	libbonoboui
ORBit2	dmidecode	gnome-mime-data	kbd	libc-client
OpenIPMI	dmraid	gnome-mount	kernel	libc-client2006
OpenIPMI-libs	dos2unix	gnome-python2	kernel-devel	libcap
SysVinit	e2fsprogs	gnome-python2-bonobo	kernel-headers	libdaemon
alsa-lib	e2fsprogs-devel	gnome-python2-canvas	keyutils	libevent
alsa-utils	e2fsprogs-libs	gnome-python2-gnomevfs	keyutils-libs	libgcc
anacron	ed	gnome-vfs2	keyutils-libs-devel	libgcrypt
app_flite	elfutils-libelf	gnu-efi	kpartx	libgcrypt-devel
apr	esound	gnutls	krb5-devel	libglade2
apr-util	ethtool	gnutls-devel	krb5-libs	libgnome
aspell	expat	gpm	kudzu	libgnomecanvas
aspell-en	ez-ipupdate	grep	lame	libgnomeui
asterisk	file	groff	less	libgomp
asterisk-addons	filesystem	grub	lha	libgpg-error
asterisk-perl	findutils	gsm	libICE	libgpg-error-devel
asterisk-sounds	flex	gtk2	libIDL	libgssapi
atftp	flite	gzip	libSM	libhugetlbfs
atftp-server	fontconfig	hal	libX11	libioverbs
atk	freetype	hdparm	libXau	libidn
audiofile	ftp	hesiod	libXcursor	libidn-devel
audiofile-devel	gamin	hicolor-icon-theme	libXdmcp	libjpeg
audit-libs	gawk	httpd	libXext	libmthca
audit-libs-python	gcc	hwdata	libXfixes	libnotify
authconfig	gcc-c++	iksemel	libXft	libogg
avahi	gd	info	libXi	libpcap
avahi-glib	gdbm	initscripts	libXinerama	libpng
basesystem	gettext	iproute	libXrandr	libpri
bash	glib2	iptables	libXrender	librdmacm
beecrypt	glibc	iptables-ipv6	libXres	libsdp
bind	glibc-common	iptraf	libacl	libselineux
bind-libs	glibc-devel	iputils	libart_lgpl	libselineux-devel

Figura 4.4: Módulos instalados en el servidor Asterisk



libselinux	make	tbm-dhcp	perl-Term-ReadKey
libselinux-devel	man	tbm-endpointcfg	perl-Time-HiRes-Value
libselinux-python	man-pages	tbm-phpsysinfo	perl-Tk
libsemanage	mcstrans	tbm-regtool	perl-URI
libsepol	mingetty	tbm-sysmaint	perl-XML-Parser
libsepol-devel	mkinitrd	tbm-tcpip	perl-XML-Simple
libstdc++	mktemp	tcl	perl-libwww-perl
libstdc++-devel	mlocate	tcp_wrappers	perl-suidperl
libsysfs	mod_ssl	termcap	php
libtermcap	module-init-tools	tzdata	php-cli
libtermcap-devel	mpg123	udev	php-common
libtiff	mysql	unixODBC	php-gd
libtiff-devel	mysql-server	unzip	php-imap
libtool-ltdl	nano	usermode	php-mbstring
libtool-ltdl-devel	nash	util-linux	php-mysql
libusb	ncurses	vconfig	php-pdo
libuser	neon	vim-minimal	php-pear
libutempter	net-snmp	vixie-cron	php-pear-DB
libvolume_id	net-snmp-libs	vsftpd	php-snmp
libvorbis	net-tools	web-meetme	pkgconfig
libwnck	nfs-utils	wget	pm-utils
libwvstreams	nfs-utils-lib	which	polycoreutils
libxml2	ngrep	wireless-tools	popt
libxml2-python	nmap	wvdial	portmap
libxslt	notification-daemon	xinetd	postfix
lksctp-tools	nspr	xmlsec1	postgresql-libs
lm_sensors	nss	xmlsec1-openssl	ppp
logrotate	ntp	xorg-x11-filesystem	prelink
lvm2	openib	yum	procmail
m2crypto	openldap	yum-fastestmirror	pycairo
m4	openssh	yum-metadata-parser	pygobject2
mailcap	openssh-clients	zaptel	pygtk2
mailx	openssh-server	zaptel-modules	pygtk2-libglade
make		zlib	pyorbit
man			python

Figura 4.5: Módulos instalados en el servidor Asterisk



Se configuraron cuentas SIP e IAX2 para hacer las pruebas de las llamadas.

Admin Reports Panel Recordings Help

Add an Extension

Please select your Device below then click Submit

Device

Device Generic SIP Device

Submit

English

Add Extension

Matias <5000>

Julian <6000>

Engineering Providers (52565432) <7000>

Antonio Stark <8000>

Tesis <9000>

FreePBX® Freedom to Connect®

FreePBX is a registered trademark of Atengo, LLC.
FreePBX 2.4.0 is licensed under GPL

Device Options

This device uses iax2 technology.

secret

xxxxxx

nottransfer

yes

context

from-internal

host

dynamic

type

friend

port

4569

qualify

yes

disallow

all

allow

gsm

dial

IAX2/7000

accountcode

mailbox

7000@device

Device Options

This device uses sip technology.

secret

xxxxxx

dtmfmode

rfc2833

canreinvite

no

context

from-internal

host

dynamic

type

friend

nat

yes

port

5060

qualify

yes

callgroup

pickupgroup

disallow

allow

gsm&alaw&ulaw

dial

SIP/6000

accountcode

mailbox

6000@default

Figura 4.6: Configuración de cuentas SIP en TrixBos mediante FreePBX



Para establecer las comunicaciones telefónicas se contrataron dos líneas SIP con los proveedores CyberVOIP y hablemos y se configuraron de la siguiente manera:

Admin

Reports

Panel

Recordings

Help

Add a Trunk

Add ZAP Trunk

Add IAX2 Trunk

Add SIP Trunk

Add ENUM Trunk

Add Custom Trunk

Add DUNDI Trunk

English

Add Trunk

Trunk ZAP/g0

Trunk SIP/hablemos

Trunk SIP/CyberVOIP2

Outgoing Settings

Trunk Name: CyberVOIP2

PEER Details:

allow=gsm&ulaw&alaw

canreinvite=yes

context=from-pstn

fromdomain=gw.cybervoip.com.ar

port=5060

fromuser=100207

host=gw.cybervoip.com.ar

nat=yes

secret=XXXXXX

type=peer

username=XXXXXX

Outgoing Settings

Trunk Name: hablemos

PEER Details:

allow=gsm&ulaw&alaw

canreinvite=yes

context=from-pstn

fromdomain=voip.hablemos.com.ar

port=5060

fromuser=XXXXXXX

host=voip.hablemos.com.ar

nat=yes

secret=XXXXXXXXX

type=peer

username=7070100822

Figura 4.7: Configuración de las líneas SIP contratadas

Se configuraron las líneas de salida para ser usadas en las llamadas:

Edit Route

Delete Route EngineeringProviders













Route Name:	EngineeringProviders	Rename												
Route Password:	<input type="text"/>													
Emergency Dialing:	<input type="checkbox"/>													
Intra Company Route:	<input type="checkbox"/>													
Music On Hold?	default ▼													
Dial Patterns	<div><input type="text" value="0 ."/></div> <div>Clean & Remove duplicates</div>													
Dial patterns wizards:	(pick one) ▼													
Trunk Sequence	<table><tr><td>0</td><td>SIP/CyberVOIP2 ▼</td><td></td><td></td></tr><tr><td>1</td><td>SIP/hablemos ▼</td><td></td><td></td></tr><tr><td></td><td><input type="text"/></td><td></td><td></td></tr></table> <div>Add</div>		0	SIP/CyberVOIP2 ▼			1	SIP/hablemos ▼				<input type="text"/>		
0	SIP/CyberVOIP2 ▼													
1	SIP/hablemos ▼													
	<input type="text"/>													

Figura 4.8: Configuración de las rutas de salidas para las llamadas

En esta configuración se puso el patrón de que cualquier llamada en la cual se interponga un 0 tomara una de las dos líneas SIP, la que se encuentre desocupada.



4.8 Configuración del archivo manager.conf

Para que la aplicación JEE pueda comunicarse con el servidor de comunicaciones Asterisk se debe configurar el archivo manager.conf, indicando el puerto, usuario, clave y restricciones.

; Asterisk Call Management support

;

[general]

enabled = yes

port = 5038

bindaddr = 0.0.0.0

[tesis]

secret = XXXXXX

read = system,call,log,verbose,command,agent,user

write = system,call,log,verbose,command,agent,user

#include manager_additional.conf

#include manager_custom.conf

4.9 Configuración de los contextos

Un contexto es un grupo de extensiones, cada extensión debe existir en un contexto, se agregaron estos contextos en el archivo extensions_custom.conf para lograr la reproducción de los audios de uso del sistema gratuito y de las publicidades.

[hableGratis-custom] Contexto 1

exten => foo,1,Answer

exten => foo,n,Playback(\${ANUNCIO})

exten => foo,n,Playback(\${PUBLICIDAD})

exten => foo,n,Dial(SIP/\${DESTINO}@CyberVOIP2,,M(bar))



[macro-bar]

Contexto 2

exten => s,1,Playback(\${PUBLICIDAD})

exten => s,n,Playback(garbarino)

4.10 Base de datos

Como servidor de base de datos se utilizo Mysql Server versión 5.0.51a-15, esta base de datos es la encargada de guardar todos los registros referentes a las llamadas telefónicas que hacen los usuarios de **HableGratis.NET**.



Figura 4.9: Logo de la base de datos MySQL

4.11 Integración de la aplicación Web con el servidor de comunicaciones

Asterisk

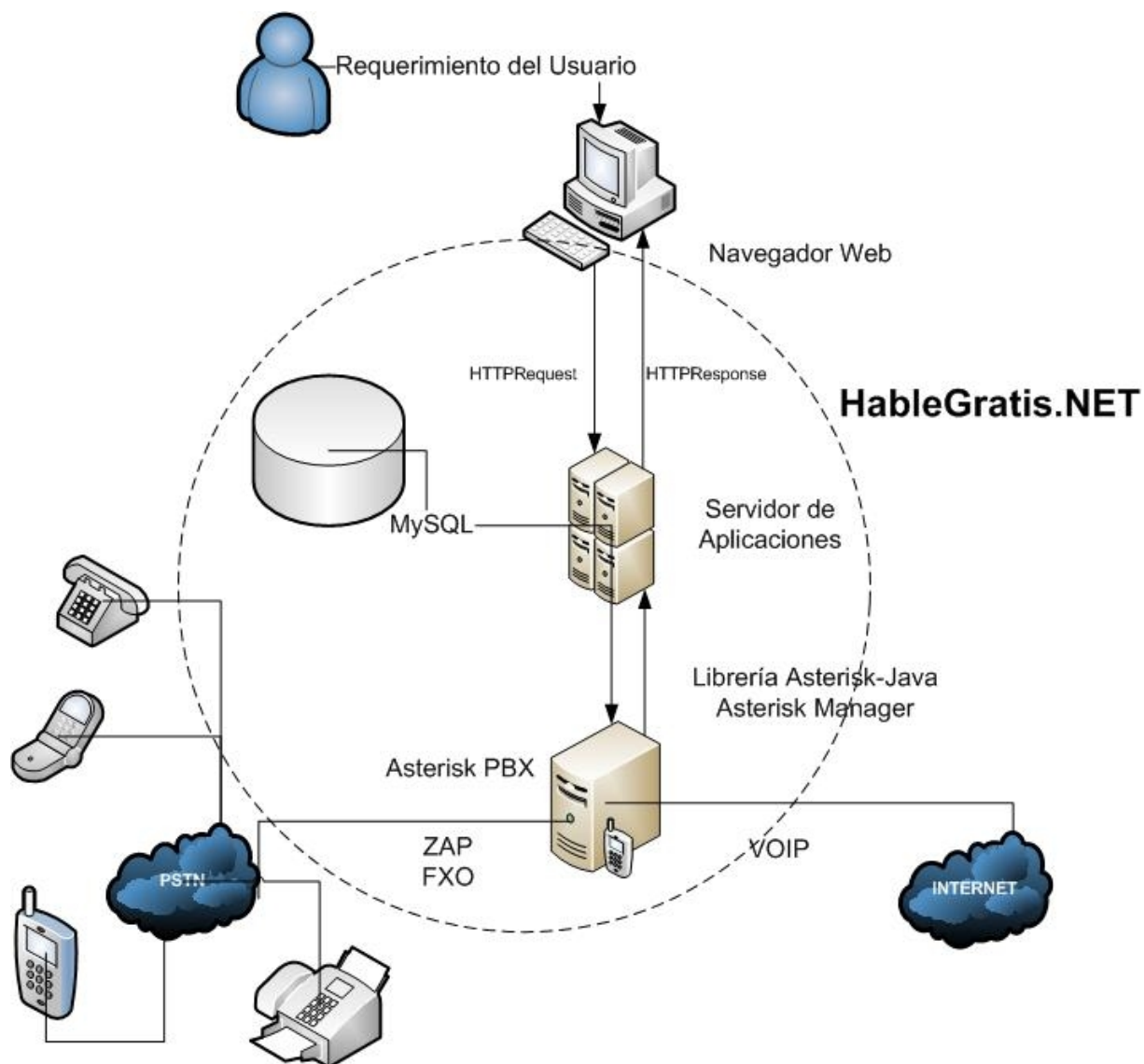


Figura 4.10: Integración del sistema



4.12 Descripción de la integración

1. El usuario interesado en establecer la llamada se dirige con su browser a **HableGratis.NET**, ingresa su número telefónico, el número telefónico al cual desea comunicarse y presiona el botón llamar.
2. El requerimiento http de establecer la llamada llega hasta el servidor de aplicaciones el cual mediante el servlet LlamarServlet se comunica con el modelo para establecer la llamada.
3. EL modelo se comunica vía TCP/IP con el Asterisk Manager usando la API Asterisk-Java, le pasa los parámetros de la llamada junto con el tipo de publicidad que sera enviada a los interlocutores.
4. El servidor Asterisk toma las lineas disponibles SIP o Zap, disca los números telefónicos origen y destino y establece la comunicación.
5. El servidor Asterisk apenas los usuarios atienden la llamada les envía la publicidad correspondiente.
6. El servidor Asterisk le envía la respuesta al modelo de el estado de la comunicación, si se pudo concretar o no.
7. El modelo guarda en la base de datos las variables y estados de la comunicación que se realizo.

4.13 Trabajo a Futuro

En el siguiente listado se encuentran las oportunidades de mejora que pueden agregarse en el futuro.

- **Establecer las llamadas discando un 0800:** El usuario puede encontrarse en un lugar en donde no tiene acceso a Internet, esta mejora le permitirá utilizar el sistema discando un número gratuito 0800 e introducir los números de teléfono origen y destino.



- **Establecer la llamada enviando un SMS:** Esta opción le permitiría al usuario que mediante un mensaje de texto SMS pueda enviar su número telefónico y el número de teléfono destino para que el servidor que los recibe pueda establecer la llamada.
- **IVR:** Un IVR avanzado puede darle la opción al usuario interesado en el servicio que si presiona tal número del teléfono puede tipear la opción de recibir promociones vía telefónica.
- **Servicio corporativo:** Se puede implementar el sistema a nivel corporativo brindándole la posibilidad a las empresas que no tienen o no quieren una estructura de PBX en su organización a que puedan comunicarse directamente con sus clientes. Este sistema se activaría vía web y comunicara al cliente con el vendedor de la empresa que contrata el servicio.



Capítulo V

5. Conclusiones

El presente trabajo se realizó a lo largo de todo el año en el cual a medida que se desarrollaba también iban surgiendo problemas debido a que el sistema que se quería desarrollar era algo nuevo y no se contaban con muchos datos disponibles.

Fueron muy importantes las soluciones que surgieron mediante el planteo de problemas a la comunidad OpenSource, especialmente las referidas a TrixBot, Asterisk y Asterisk-Java; el trabajo en equipo logra simplificar los problemas.

Del análisis de mercado surgió que publicar en **HableGratis.NET** es ligeramente mas caro que publicar en los distintos medios analizados como son la televisión, la radio y los medios gráficos, pero este leve incremento en el costo publicitario se ve compensado con las ventajas que tiene enviar la publicidad previamente a la llamada en donde los interlocutores no pueden cambiar de canal y deben estar atentos al curso de la de la misma.

Para la primera versión del prototipo se lograron cumplir todas las expectativas que tenia acerca de la implementación de este trabajo, cabe destacar que las pruebas que se hicieron fueron a una pequeña escala debido a las limitaciones de hardware que se encontraban disponibles a mi alcance, queda para el futuro la prueba en servidores más robustos los cuáles podrán brindar una buena calidad de servicio; Se probó que el sistema no solo es innovador sino que puede llegar a brindarse como servicio de comunicaciones masivas para una zona determinada como por ejemplo la Ciudad de Buenos Aires, esto dependerá de donde provengan las líneas VoIP contratadas las cuales son usadas por el servidor de comunicaciones.

Si bien el presente trabajo llega hasta un prototipo de la implementación de una primera versión quedaron sentadas las bases para una ampliación a futuro dado que el planteo del negocio de **HableGratis.NET** es innovador y hasta el momento no se encontraron datos



de otra empresa que este ofreciendo este tipo de servicios

El modelo según los costos actuales en la Argentina es rentable y de implementarse podría ser una alternativa eficiente para competir con los demás medios tradicionales de publicidad..



Bibliografía

Barrie Dempster, Kerry Garrison, *TrixBos Made Easy*, Packt Publishing, 2006.

David Gomillion, *Barrie Dempster, Building Telephony Systems With Asterisk*, Packt Publishing, 2006.

Jared Smith, Jim Van Meggelen, Leif Madsen, *Asterisk: The Future of Telephony*, Oreilly, 2005.




Panda, Rahman, Lane, *EJB In Action*, Manning, 2007.





Anexo A, Herramientas utilizadas

 <p>Figura 5.1: Logotipo de subversion</p>	<p>Se utilizó el servidor SVN provisto por google https://hablegratis.googlecode.com/svn/trunk/HableGratis.net</p>
 <p>Figura 5.2: JbossIDE for Eclipse</p>	<p>Como entorno de desarrollo se utilizó JbossIDE for Eclipse 2.0.0.Beta2</p>
 <p>Figura 5.3: Logotipo Dia</p>	<p>Se utilizó esta herramienta para el modelado UML</p>
 <p>Figura 5.4: Logotipo de Ant</p>	<p>Para facilitar las tareas repetitivas y el despliegue de los servlets.</p>
	<p>Se utilizó Jboss como servidor de aplicaciones</p>



<p>Figura 5.5: Logotipo de JBoss</p>	
 <p>Figura 5.6: JUnit en proceso</p>	<p>JUnit, es un conjunto de librerías que son utilizadas en programación para hacer pruebas unitarias de aplicaciones Java.</p>
 <p>Figura 5.7: Logotipo de Mysql</p>	<p>Se utilizo MySQL como motor de base de datos.</p>
 <p>Figura 5.8: Logotipo de TriBox</p>	<p>Se utilizó la distribución de GNU/Linux TriBox como servidor de comunicaciones.</p>
 <p>Figura 5.9: Logotipo de Gimp</p>	<p>Para editar los gráficos se utilizo el <i>GNU Image Manipulation Program</i></p>
 <p>Figura 5.10: Logotipo de Nvu</p>	<p>Se utilizó NVU para el diseño de las paginas HTML</p>



 <p>Figura 5.11: Logotipo de ArgoUML</p>	Se utilizó ArgoUML para el diseño del diagrama de clases
 <p>Figura 5.12: Logotipo de OpenOffice</p>	Se utilizó OpenOffice para armar todo el documento

La importancia del uso de herramientas de código abierto

Muchas veces se entiende que la ventaja de un programa de código abierto es que es gratis y eso es un error, es por eso que se explica el por qué hay gente que no lo ve como una ventaja suficiente, mas aun cuando es muy extendido el caso de no estar pagando por los programas utilizados porque se usen copias “ilegales”.

Si bien la tendencia principal de los programas de código abierto es que sean gratis esta no es la ventaja principal, hay muchas otras mas como las que se expresan a continuación:

- **Libertad de uso y redistribución:** Las licencias de software de fuentes abiertas existentes permiten la instalación del software tantas veces y en tantas máquinas como el usuario desee.
- **Independencia tecnológica:** El acceso al código fuente permite el desarrollo de nuevos productos sin la necesidad de desarrollar todo el proceso partiendo de cero.



- **Fomento de la libre competencia al basarse en servicios y no licencias:** Uno de los modelos de negocio que genera el software de fuentes abiertas es la contratación de servicios de atención al cliente. Este sistema permite que las compañías que den el servicio compitan en igualdad de condiciones al no poseer la exclusividad del producto del cual dan el servicio. Esto, además, produce un cambio que redundará en una mayor atención al cliente y contratación de empleados, en contraposición a sistemas mayoritariamente sostenidos por la venta de licencias.
- **Estándares abiertos:** Los estándares abiertos permiten una interoperatividad más alta entre sistemas, evitando incompatibilidades. Los estándares de facto son válidos en ocasiones para lograr una alta interoperatividad si se omite el hecho de que estos exigen el permiso del propietario y, en su caso, el pago de royalties.
- **Sistemas sin puertas traseras y más seguros:** El acceso al código fuente permite que tanto expertos como empresas de seguridad de todo el mundo puedan auditar los programas, por lo que la existencia de puertas traseras es ilógica, ya que se pondría en evidencia de manera casi inmediata.
- **Corrección mas rápida y eficiente de fallos:** La disponibilidad del código fuente ha demostrado solucionar mas rápidamente los fallos de seguridad en el software de fuentes abiertas, posibilidad que no se da en el caso del software propietario.

Entender estos conceptos permitirá entender que los programas de código abierto tienen mucho más potencial que un simple programa gratuito y que quienes lo usen tienen una serie de ventajas que van más allá de no pagar por licencias.



Anexo B, Código Java de la aplicación

OriginateBean

```
package sessionBeans;

import java.io.IOException;
import java.util.Calendar;
import java.util.Date;

import javax.ejb.EJB;
import javax.ejb.Stateless;

import org.asteriskjava.manager.AuthenticationFailedException;
import org.asteriskjava.manager.ManagerConnection;
import org.asteriskjava.manager.ManagerConnectionFactory;
import org.asteriskjava.manager.action.OriginateAction;
import org.asteriskjava.manager.action.SipShowPeerAction;
import org.asteriskjava.manager.response.ManagerResponse;

import administradores.AdminRegistroLlamadasRemote;
import administradores.AdministradorPublicidadRemote;

import com.sun.corba.se.impl.orbutil.threadpool.TimeoutException;

import entities.Publicidad;
import entities.RegistroLlamadas;
```



```
@Stateless
public class OriginateBean implements OriginateRemote {
//    Injection del Session Bean AdminRegistroLlamadas
    @EJB AdminRegistroLlamadasRemote adminRegistroLlamadas;
//    Injection del Session Bean AdminPublicidad
    @EJB AdministradorPublicidadRemote adminPublicidad;

    private ManagerConnection managerConnection;
    private String origen = null;
    private String destino = null;
    private Publicidad publicidad;
    private RegistroLlamadas registroLlamadas;
    private Calendar calendario;
    ManagerConnectionFactory factory = new
ManagerConnectionFactory("julianrousselot.dyndns.org", "tesis", "XXXXXXX");

    public OriginateBean() throws IOException{

this.managerConnection = factory.createManagerConnection();
managerConnection.getState();
    }

//Establece la llamada
    public void establecerLlamada(String origen, String destino){
        this.origen = origen;
        this.destino = destino;
    }
}
```




```
System.out.println("ORIGINATE BEAN");
System.out.println("ORIGEN: " + origen);
System.out.println("DESTINO: " + destino);

try {
    this.run();
} catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (AuthenticationFailedException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (TimeoutException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
}

public void run() throws IOException, AuthenticationFailedException,
TimeoutException{
//    Busco el entity publicidad
publicidad = (Publicidad) adminPublicidad.obtenerPorNombre("garbarino");
calendario = Calendar.getInstance();
//Creo un RegistroLlamadas
registroLlamadas = new RegistroLlamadas();
registroLlamadas.setPublicidad(publicidad);
registroLlamadas.setAnuncio("anuncio");
registroLlamadas.setDestino(destino);
```



```
registroLlamadas.setOrigen(origen);
registroLlamadas.setDuracion(publicidad.getDuracion());
registroLlamadas.setFecha(new Date());
registroLlamadas.setOrigen(origen);

try {

    OriginateAction originateAction = new OriginateAction();
    SipShowPeerAction sipShowpeerAction = new SipShowPeerAction();

    ManagerResponse originateResponse;
    // DEFINITIVO
    originateAction = new OriginateAction();
    originateAction.setChannel("SIP/".concat(origen).concat("@CyberVOIP"));
    originateAction.setContext("hableGratis-custom");
    originateAction.setExten("foo");
    originateAction.setPriority(new Integer(1));
    originateAction.setTimeout(new Integer(30000));
    originateAction.setVariable("DESTINO", destino );
    originateAction.setVariable("PUBLICIDAD", "garbarino");
    originateAction.setVariable("ANUNCIO", "anuncio");

    /*PRUEBA
    originateAction = new OriginateAction();
    originateAction.setChannel("SIP/6000");
    //originateAction.setContext("from-inernal");
```



```
originateAction.setExten("7000");
originateAction.setPriority(new Integer(1));
originateAction.setTimeout(new Integer(30000));
*/

//Me conecto al Asterisk
managerConnection.login();

//Le mando el originate
originateResponse = managerConnection.sendAction(originateAction, 30000);

System.out.println(originateResponse.getResponse());

System.out.println("Attributes: " + originateResponse.getAttributes());
System.out.println("Message: " + originateResponse.getMessage());
System.out.println("ActionID: " + originateResponse.getActionId());
System.out.println("Response: " + originateResponse.getResponse());
System.out.println("DateReceived: " + originateResponse.getDateReceived());
System.out.println("Channel: " + originateAction.getChannel());

//Le mando una accion al Asterisk Manager
// originateResponse = managerConnection.sendAction(sipShowpeerAction);

// System.out.println("SIP ShowPeer: " + originateResponse.getAttributes());

//Si se establecio la llamada
if(originateResponse.getResponse().equals("Success"))
```



```
registroLlamadas.setResultado("ANSWERED");
else
    registroLlamadas.setResultado("NO ANSWERED");

adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);

managerConnection.logoff();
/*
} catch (InterruptedException ex) {
    Logger.getLogger(OriginateBean.class.getName()).log(Level.SEVERE, null, ex);
}
*/ catch (IOException e) {
    e.printStackTrace();
    registroLlamadas.setResultado("NO ANSWERED");
    adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);

} catch (IllegalArgumentException e) {
    e.printStackTrace();
    registroLlamadas.setResultado("NO ANSWERED");
    adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);
} catch (IllegalStateException e) {
    registroLlamadas.setResultado("NO ANSWERED");
    adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);
    e.printStackTrace();
} catch (org.asteriskjava.manager.TimeoutException e) {
    e.printStackTrace();
    registroLlamadas.setResultado("NO ANSWERED");
```



```
        adminRegistroLlamadas.guardarRegistroLlamadas(registroLlamadas);  
    }  
}  
  
}
```

FachadaBean

```
package src;
```

```
import java.io.IOException;
```

```
import javax.ejb.EJB;
```

```
import javax.ejb.Stateless;
```

```
import org.asteriskjava.manager.AuthenticationFailedException;
```

```
import com.sun.corba.se.impl.orbutil.threadpool.TimeoutException;
```

```
import sessionBeans.OriginateRemote;
```

```
@Stateless
```

```
public class FachadaBean implements FachadaRemote{
```

```
@EJB OriginateRemote originate;
```

```
    public void establecerLlamada(String origen, String destino) {  
        originate.establecerLlamada(origen, destino);  
    }  
}
```



```
}
```

Publicidad

```
package entities;
```

```
import java.io.Serializable;
```

```
import javax.persistence.Column;
```

```
import javax.persistence.Entity;
```

```
import javax.persistence.GeneratedValue;
```

```
import javax.persistence.Id;
```

```
import javax.persistence.Table;
```

```
@Entity
```

```
@Table(name="Publicidad")
```

```
public class Publicidad implements Serializable{
```

```
    private int id;
```

```
    private String nombre;
```

```
    private String genero;
```

```
    private String pathAudio;
```

```
    private int duracion;
```

```
    public Publicidad(){
```

```
        super();
```



```
}  
  
@Column  
public int getDuracion() {  
    return duracion;  
}  
  
public void setDuracion(int duracion) {  
    this.duracion = duracion;  
}  
  
@Column  
public String getGenero() {  
    return genero;  
}  
  
public void setGenero(String genero) {  
    this.genero = genero;  
}  
  
@Column  
public String getPathAudio() {  
    return pathAudio;  
}  
  
public void setPathAudio(String pathAudio) {  
    this.pathAudio = pathAudio;  
}  
  
@Id  
@GeneratedValue
```



```
public int getId() {  
    return id;  
}  
  
public void setId(int id) {  
    this.id = id;  
}  
public void setNombre(String nombre) {  
    this.nombre = nombre;  
}  
public String getNombre() {  
    return nombre;  
}  
}
```

AdministradorPublicidadBean

```
package administradores;  
  
import java.util.ArrayList;  
import java.util.Collection;  
  
import javax.ejb.Stateless;  
import javax.persistence.EntityManager;  
import javax.persistence.PersistenceContext;  
import javax.persistence.Query;
```




```
import entities.Publicidad;

public @Stateless class AdministradorPublicidadBean implements
AdministradorPublicidadRemote {
    @PersistenceContext(name="hableGratis.NET")
    private EntityManager entityManager;

    public boolean existePublicidad(int id) {
        // TODO Auto-generated method stub
        return false;
    }

    //Guarda una publicidad en la base de datos
    public void guardarPublicidad(Publicidad publicidad) {
        entityManager.merge(publicidad);
    }

    public Object obtenerPorNombre(String nombre) {
        Publicidad publicidad;
        ArrayList publicidades;

        Query query = entityManager.createQuery("SELECT p FROM Publicidad p
WHERE p.nombre LIKE :nombre");
        query.setParameter("nombre", nombre);

        publicidades = (ArrayList <Publicidad>)query.getResultList();
        publicidad = (Publicidad) publicidades.get(0);
    }
}
```



```
        return publicidad;
    }

    public Publicidad obtenerPorId(int id) {
        // TODO Auto-generated method stub
        return null;
    }

    public Collection obtenerRangoDePrecios(float precioDesde, float precioHasta) {
        // TODO Auto-generated method stub
        return null;
    }

    public Collection obtenerTodos() {
        // TODO Auto-generated method stub
        return null;
    }

    public void test() {
        // TODO Auto-generated method stub

    }

}
```



AdministradorLlamadasBean

```
package administradores;

import java.util.Collection;

import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import entities.Publicidad;
import entities.RegistroLlamadas;

public @Stateless class AdminRegistroLlamadasBean implements
AdminRegistroLlamadasRemote {
    @PersistenceContext(name="hableGratis.NET")
    private EntityManager entityManager;

    public boolean existeRegistroLlamadas(int id) {
        // TODO Auto-generated method stub
        return false;
    }

    public void guardarRegistroLlamadas(RegistroLlamadas registroLlamadas) {
        entityManager.merge(registroLlamadas);
    }
}
```



```
public Collection obtenerPorDescripcion(String descripcion) {  
    // TODO Auto-generated method stub  
    return null;  
}  
  
public Publicidad obtenerPorId(int id) {  
    // TODO Auto-generated method stub  
    return null;  
}  
  
public Collection obtenerTodos() {  
    // TODO Auto-generated method stub  
    return null;  
}  
  
public void test() {  
    // TODO Auto-generated method stub  
}  
}
```



Anexo C, Archivos configurados del Asterisk PBX

zapata.conf

```
; Zapata telephony interface
; Configuration file
[trunkgroups]

[channels]
language=es
context=from-zaptel
signalling=fxs_ks
rxwink=300 ; Atlas seems to use long (250ms) winks
;
; Whether or not to do distinctive ring detection on FXO lines
;usedistinctiveringdetection=yes
usecallerid=yes
hidecallerid=no
callwaiting=yes
usecallingpres=yes
callwaitingcallerid=yes
threewaycalling=yes
transfer=yes
cancallforward=yes
callreturn=yes
echocancel=yes
echocancelwhenbridged=no
;echotraining=800
rxgain=0.0
txgain=0.0
group=0
callgroup=1
```



```
pickupgroup=1
immediate=no
;faxdetect=both
faxdetect=incoming
;faxdetect=outgoing
;faxdetect=no

;Include genzaptelconf configs
#include zapata-auto.conf
group=1
;Include AMP configs
#include zapata_additional.conf
```

sip.conf

```
[general]
;
; enable and force the sip jitterbuffer. If these settings are desired
; they should be set in the sip_general_custom.conf file as this file
; will get overwritten during reloads and upgrades.
;
; jbenable=yes
; jbforce=yes

; These will all be included in the [general] context
;
#include sip_general_additional.conf
#include sip_general_custom.conf
#include sip_nat.conf
#include sip_registrations_custom.conf
#include sip_registrations.conf
```



; These should all be expected to come after the [general] context

;

#include sip_custom.conf

#include sip_additional.conf

#include sip_custom_post.conf

sip_additional.conf

[CyberVOIP2]

allow=gsm

allow=ulaw

allow=alaw

canreinvite=yes

context=from-pstn

fromdomain=gw.cybervoip.com.ar

port=5060

fromuser=XXXXXXXX

host=gw.cybervoip.com.ar

nat=yes

secret=XXXXXXXX

type=peer

username=XXXXXXXX

qualify=yes

insecure=port,invite

[CyberVOIP]

allow=gsm

allow=ulaw

allow=alaw

canreinvite=yes

context=from-pstn

fromdomain=gw.cybervoip.com.ar

port=5060



```
fromuser=XXXXXXXXXX
host=gw.cybervoip.com.ar
nat=yes
secret=XXXXXXXXXX
type=peer
username=XXXXXXXXXX
qualify=yes
insecure=port,invite
```

```
[hablemos]
allow=gsm
allow=ulaw
allow=alaw
canreinvite=yes
context=from-pstn
fromdomain=voip.hablemos.com.ar
port=5060
fromuser=XXXXXXX
host=voip.hablemos.com.ar
nat=yes
secret=XXXXXXX
type=peer
username=XXXXXXX
qualify=yes
insecure=port,invite
```

sip_nat.conf

```
externhost=julianrousselot.dyndns.org ;replace with your public ip address
localnet=192.168.0.0/255.255.255.0 ;edit to reflect your network
nat=yes
```




rtp.conf

```
;
; RTP Configuration
;
[general]
;
; RTP start and RTP end configure start and end addresses
;
rtpstart=10000
rtpend=20000
```

extension_custom.conf

```
[test-custom]
exten => _X.,1,Answer()
exten => _X.,n,Playback(garbarino)
;exten => _X.,n,Goto(from-internal,${EXTEN},1)
;exten => _X.,n,Goto(test-custom2,${EXTEN},1)

[test-custom2]
exten => _X.,1,Answer()
exten => _X.,n,Playback(garbarino)
exten => _X.,n,Dial(iax2/7000)

[Agi-custom]
;exten => s,1,Answer()
;exten => s,n,Agi(agi://192.168.0.33/hello.agi)

;exten => 7000,1,Answer()
;exten => 7000,n,Agi(agi://192.168.0.33/hello.agi)
;exten => s,1,Agi(agi://192.168.0.33/hello.agi) este solo
[test-custom3]
```



```
exten => foo,1,Answer
exten => foo,n,Playback(garbarino)
exten => foo,n,Dial(SIP/1559049720@CyberVOIP2,,M(bar))
```

[hableGratis-custom]

```
exten => foo,1,Answer
exten => foo,n,Playback(${ANUNCIO})
exten => foo,n,Playback(${PUBLICIDAD})
exten => foo,n,Dial(SIP/\${DESTINO}@CyberVOIP2,,M(bar))
```

[macro-bar]

```
exten => s,1,Playback(anuncio)
exten => s,n,Playback(garbarino)
```

asterisk.conf

```
[directories]
astetcdir => /etc/asterisk
astmoddir => /usr/lib/asterisk/modules
astvarlibdir => /var/lib/asterisk
astagidir => /var/lib/asterisk/agi-bin
astspooldir => /var/spool/asterisk
astrundir => /var/run/asterisk
astlogdir => /var/log/asterisk
```

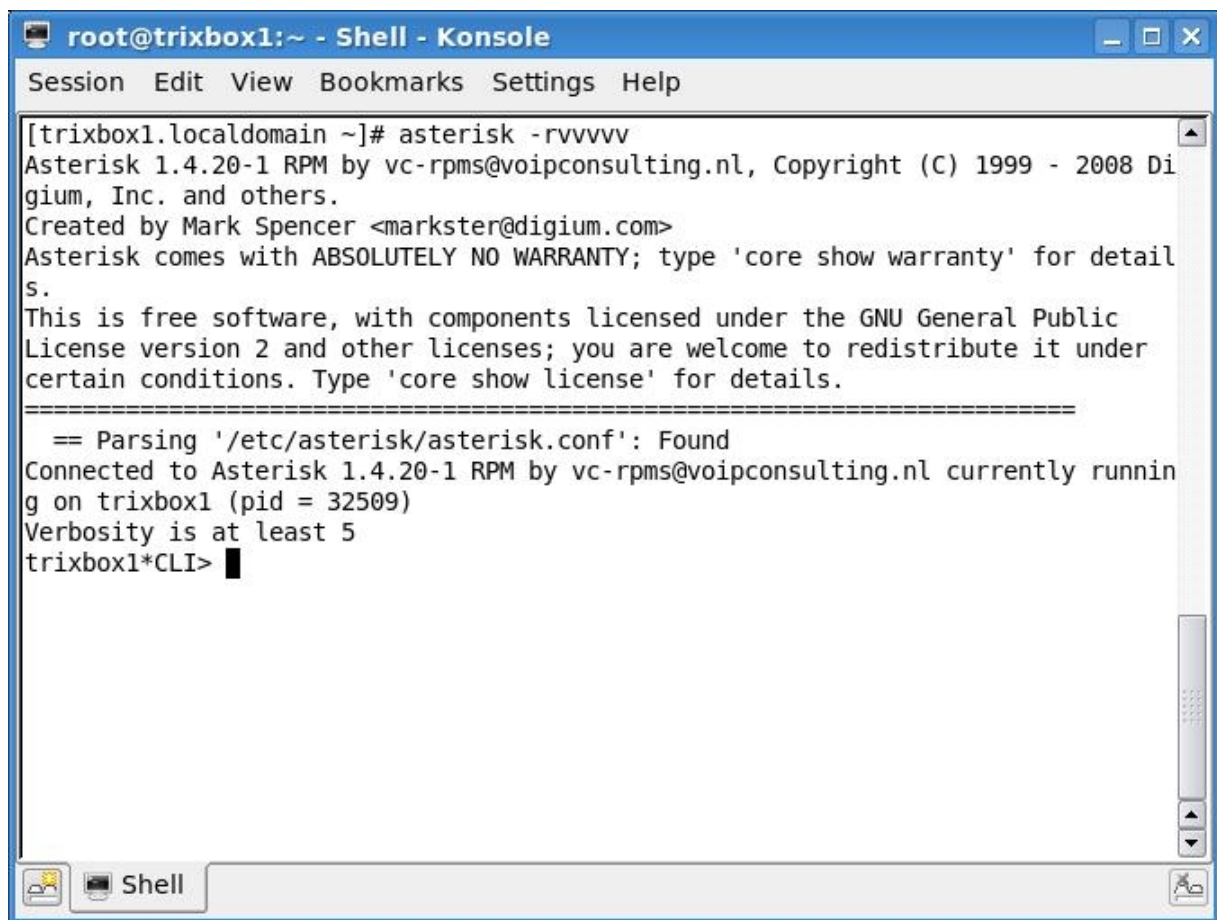
```
[options]
transmit_silence_during_record = yes
```



[general]

languageprefix=yes

Consola de administración remota del servidor Asterisk



```
root@trixbox1:~ - Shell - Konsole
Session Edit View Bookmarks Settings Help

[trixbox1.localdomain ~]# asterisk -rvvvvv
Asterisk 1.4.20-1 RPM by vc-rpms@voipconsulting.nl, Copyright (C) 1999 - 2008 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public License version 2 and other licenses; you are welcome to redistribute it under certain conditions. Type 'core show license' for details.
=====
== Parsing '/etc/asterisk/asterisk.conf': Found
Connected to Asterisk 1.4.20-1 RPM by vc-rpms@voipconsulting.nl currently running on trixbox1 (pid = 32509)
Verbosity is at least 5
trixbox1*CLI>
```

Figura 6.1: Consola de comando para interactuar con el servidor Asterisk



```
root@trixbox1:~ - Shell - Konsole
Session Edit View Bookmarks Settings Help

[trixbox1.localdomain ~]# asterisk -rvvvvv
Asterisk 1.4.20-1 RPM by vc-rpms@voipconsulting.nl, Copyright (C) 1999 - 2008 Digium, Inc
. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=====
== Parsing '/etc/asterisk/asterisk.conf': Found
Connected to Asterisk 1.4.20-1 RPM by vc-rpms@voipconsulting.nl currently running on trix
box1 (pid = 32509)
Verbosity is at least 5
trixbox1*CLI> sip show peers
Name/username      Host              Dyn Nat ACL Port      Status
hablemos/7070100822 200.69.196.222    N      5060    OK (65 ms)
CyberVOIP2/100207   201.216.214.61    N      5060    OK (64 ms)
CyberVOIP/100501    201.216.214.61    N      5060    OK (64 ms)
9000                (Unspecified)     D      N      0        UNKNOWN
8000                (Unspecified)     D      N      0        UNKNOWN
6000/6000           (Unspecified)     D      N      0        UNKNOWN
5000                (Unspecified)     D      N      0        UNKNOWN
7 sip peers [Monitored: 3 online, 4 offline Unmonitored: 0 online, 0 offline]
trixbox1*CLI>
```

Figura 6.2: Consola de comando ejecutando el comando sip show peers el cual muestra los peers configurados en el servidor Asterisk de HableGratis.NET