**ESCUELA SUPERIOR POLITECNICA DEL LITORAL**


**Facultad de Ingeniería en Electricidad y Computación**

# "Diseño e Implementación de un Sistema de Gestión Telefónica Automática para Negocios Hoteleros"

**INFORME DE MATERIA DE GRADUACIÓN**

Previa a la obtención del Título de:

**INGENIERO EN TELEMÁTICA**

Presentada por:

**LENIN ISAÍAS ESCOBAR MENDOZA**

**FREDDY JAVIER GARNICA ARROBA**

GUAYAQUIL-ECUADOR

AÑO

2011

# AGRADECIMIENTO

En primer lugar a Dios por haber hecho realidad este sueño tan anhelado por nosotros y poder de esta manera dar una inmensa alegría a nuestras respectivas familias.

A nuestras familias por ser un soporte constante, brindándonos ese amor incondicional y la motivación para seguir adelante todos los días.

Agradecemos a nuestros profesores por todas las enseñanzas que nos han brindado, algo primordial para nuestro crecimiento intelectual.

# DEDICATORIA

A Dios, por ser mi guía a lo largo de toda mi carrera. A mis padres, mi esposa e hija y mis hermanas por ser mi motivación para seguir adelante.

***Freddy Garnica Arroba***

A Dios, a mis padres, hermanos y amigos por estar junto a mí y ser mi fuerza para seguir adelante, ya que sin ellos no hubiese podido alcanzar esta meta.

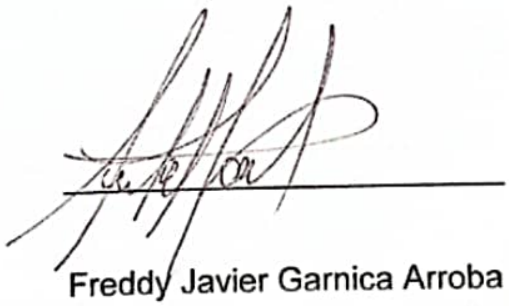
***Lenin Escobar Mendoza***

# DECLARACIÓN EXPRESA

"La responsabilidad del contenido de este proyecto de grado, nos corresponde exclusivamente; y el patrimonio intelectual de la misma a la Escuela Superior Politécnica del Litoral".

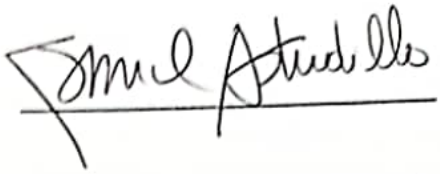
(Reglamento de Graduación de la ESPOL)

Lenin Isaías Escobar Mendoza



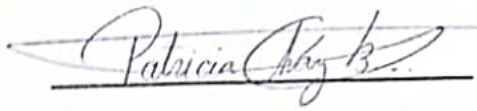
Freddy Javier Garnica Arroba

# TRIBUNAL DE SUSTENTACION



Ing. Gabriel Astudillo Brocel

PROFESOR DE LA MATERIA DE GRADUACION



Ing. Patricia Chávez Burbano

PROFESOR DELEGADO POR EL DECANO DE LA FACULTAD

# RESUMEN

Este proyecto consistió en la implementación de una pequeña central telefónica basada en Asterisk capaz de gestionar a través de llamadas telefónicas las actividades relacionadas al sector hotelero. El cual permitió hacer reservaciones telefónicas, programación de despertadores a las habitaciones y consultar el estado del clima en tiempo real.

Con el desarrollo de este proyecto se buscó administrar de una manera más eficiente los medios tecnológicos con los que consta el sector hotelero.

Con la realización de este proyecto se buscó alcanzar los siguientes objetivos:

a) Gestionar de manera ágil las reservaciones.
b) Control sobre el flujo de llamadas.
c) Brindar un servicio automatizado de reservas, eliminando el costo un operador.
d) Brindar servicios diferentes como son despertadores y consulta del clima.

# INDICE DE CONTENIDO

# INDICE DE FIGURAS

# INDICE DE TABLAS

# ABREVIATURAS

| | |
|---|---|
| FXO | Interfaz de Central Externa |
| FXS | Interfaz de Abonado Externo |
| GPL | Licencia Pública General |
| HTTP | Protocolo de Transferencia de Hipertexto |
| IAX | Inter-Asterisk eXchange Protocol |
| IETF | Grupo de Tareas de Ingeniería en Internet |
| IP | Protocolo de Internet |
| IVR | Respuesta de Voz Interactiva |
| MGCP | Protocolo de Control de Dispositivos de entrada |
| PSTN | Red de Telefonía Pública Conmutada |
| RDSI | Red Digital de Servicios Integrados |
| RTB | Red Telefónica Básica |
| SCCP | Protocolo de Control de Cliente Skinny |
| SDP | Protocolo de Descripción de Sesión |
| SER | Media de Error de Símbolos |
| SIP | Protocolo de Inicio de Sesión |
| SMTP | Protocolo de Transferencia Simple de Correo |
| UAC | Control de Cuentas de Usuario |
| UAS | Servidor de Agentes de Usuario |

# INTRODUCCION

En el mundo actual el constante movimiento de las personas entre puntos geográficamente lejanos hace la necesidad del uso de hoteles para poder hospedarse.

Estos hoteles necesitan sistemas autónomos cada vez mejores para brindar un servicio de calidad a sus usuarios desde el momento de la reserva hasta cuando se realiza el registro de salida del mismo.

Uno de los sistemas a mejorarse es el sistema de reservas telefónicas mediante la implementación de centralitas telefónicas VoIP.

No obstante, un gran obstáculo para la implementación de estas soluciones ha sido su costo económico ya que la mayoría de estas son propietarias.

La centralita telefónica Open Source Asterisk, proporciona un método eficaz para combatir problemas de costos de implementación a la vez que nos permite un control del flujo de llamadas.

Además posee una alta escalabilidad, ya que puede ofrecer servicios para pequeñas empresas con pocos usuarios, hasta empresas que tengan diferentes sedes.

Asterisk incorpora la mayoría de los estándares telefónicos, lo que le permite conectarse sin ningún problema a la red pública de telefonía y se complementa con los diferentes lenguajes de programación (PHP, C, Pascal) para realizar tareas de más alta complejidad.

# CAPÍTULO 1

# ANTECEDENTES Y JUSTIFICACIÓN

## 1.1 **ANTECEDENTES**

Los servicios tecnológicos de hoy en día y su correcto aprovechamiento pueden marcar la diferencia entre el triunfo y fracaso de una empresa ya que estos dan un valor agregado al producto o servicio que estas ofrecen. Los usuarios buscan satisfacer sus necesidades de la forma más rápida posible y quien se lo ofrezca tendrá su preferencia.

Es por este motivo que los empresarios vanguardistas deben tener especial preocupación por el desarrollo tecnológico en sus empresas para poder así entrar o mantenerse en la competencia por los clientes.

En la actualidad con el desarrollo de la tecnología y la masificación del código abierto un nuevo software ha tomado fuerza, este software llamado Asterisk el cual nos entrega un servicio de PBX se ha popularizado debido a sus excelentes prestaciones además de los beneficios que nos ofrece por ser software libre, como el no depender de PBX's propietarias.

Una de las ventajas que nos ofrece Asterisk es su compatibilidad con otros servicios como bases de datos a través de MySQL y de la ejecución de scripts utilizando php características muy importantes para la realización de nuestro proyecto.

## 1.2 JUSTIFICACIÓN

Como es de conocimiento general, todas las empresas ya sean medianas o grandes necesitan estar siempre ofreciendo un servicio adicional, pero eso representa que ellos tengan que desembolsar grandes cantidades de dinero para obtener estos servicios. En este caso en particular este servicio va dirigido a la industria hotelera.

Se propone una opción de servicio muy ágil, automatizado y que sea fácil de usar para cualquier tipo de huésped. Todos estos servicios son posibles gracias a las prestaciones que se obtienen de la tecnología de voz sobre IP, que sin duda lograra cubrir muchas expectativas de los clientes a un bajo costo en relación a los servicios que se pueden ofrecer con un software propietario.

## 1.3 DESCRIPCIÓN DEL PROYECTO

La solución tecnológica que este proyecto plantea se basa en los siguientes objetivos:

### 1.3.1 OBJETIVO GENERAL

- Implementar un sistema que permita gestionar a través de llamadas telefónicas las actividades relacionadas al sector hotelero.

### 1.3.2 OBJETIVOS ESPECÍFICOS

- Crear un sistema de reservas telefónicas automatizado para usuarios registrados previamente en la base de datos del hotel.
- Establecer un sistema que pueda realizar la programación de alarmas en cada una de las habitaciones del hotel.
- Optimizar los servicios que puedan brindarse a través de un medio telefónico.

## 1.4 METODOLOGÍA

Se va a instalar Asterisk sobre un servidor el cual tendrá como sistema operativo CENTOS (Linux), también lo utilizaremos como servidor web para el cual se instalará Apache y php los cuales harán intercambio de información con una base de datos que estará corriendo en MySQL.

El servidor también contará con una tarjeta con puertos FXS y FXO para poder receptar y conmutar las llamadas que provengan desde la red PSTN.

## 1.5 PERFIL DE LA TESIS

Con el desarrollo de la tesis se tiene como objetivo principal optimizar los recursos tecnológicos con los que cuenta un hotel y así poder ofrecer nuevos servicios a sus clientes.

En el capítulo 2, se revisarán los fundamentos teóricos, sus características, mecanismos de implementación, aplicaciones y servicios que brinda actualmente, además de las proyecciones que tiene la tecnología para su futuro.

En el capítulo 3 se detallará las especificaciones técnicas de la solución, el análisis, el diseño y la implementación del proyecto.

Posteriormente en el capítulo 4, se realizarán las pruebas de conexión, el establecimiento de las llamadas, la concurrencia, la base de datos, y escalabilidad.

# CAPITULO 2

# ASTERISK Y PROTOCOLO SIP

## 2.1 ASTERISK

Asterisk 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 como se menciona en [1].

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 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. La conexión entre el servidor y las líneas analógicas o digitales están de acuerdo al diseño de la figura 2.1.

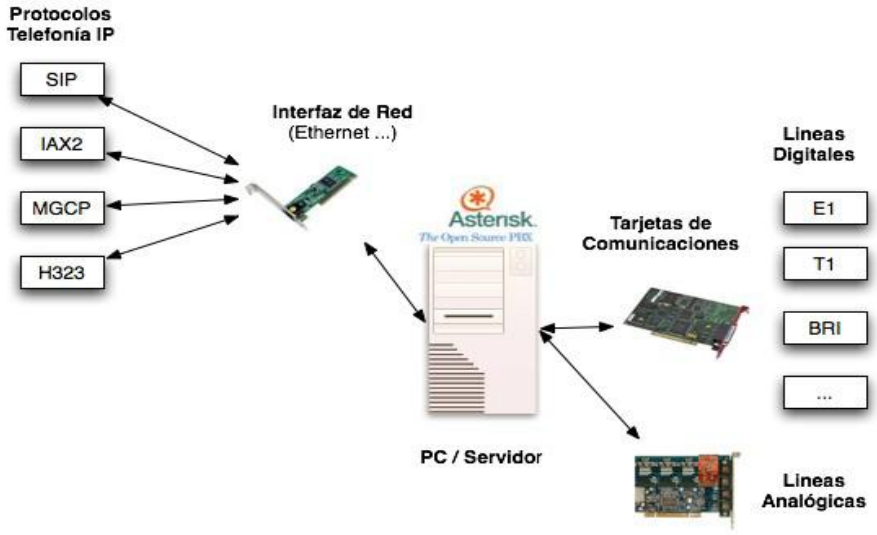



FIGURA 2.1 Esquema conceptual de una red basada en Asterisk.

Una de las características más interesantes 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 puerta de enlace entre ambos.

Porque la telefonía es un servicio crítico y un reemplazo masivo no siempre es bienvenido ni recomendable. En muchos casos, la transición de tecnología debe hacerse con mucho cuidado.

Asterisk puede integrarse como un puente transparente hacia la tecnología Voz sobre IP sin necesidad de modificar o actuar en la infraestructura telefónica ya desplegada y en producción.

Gracias a su condición de software libre, Asterisk está al alcance de pequeñas, medianas y grandes empresas, ya que su implementación es de muy bajo costo con respecto a las PBX's propietarias, ofreciendo además una gama de funcionalidades que se ajustan a los requerimientos de cualquier empresa.

Cabe recalcar que Asterisk podrá ajustarse a las redes telefónicas públicas a través de unas tarjetas que permiten la comunicación entre estas dos tecnologías como se ilustra en la figura 2.2. Las tarjetas que se van a utilizar son las FXS/FXO, que son muy accesibles para cualquier tipo de empresa de acuerdo a los requerimientos necesarios. En el mercado existen diferentes tipos de tarjetas que van a ir variando de acuerdo a los servicios que se vayan a brindar.

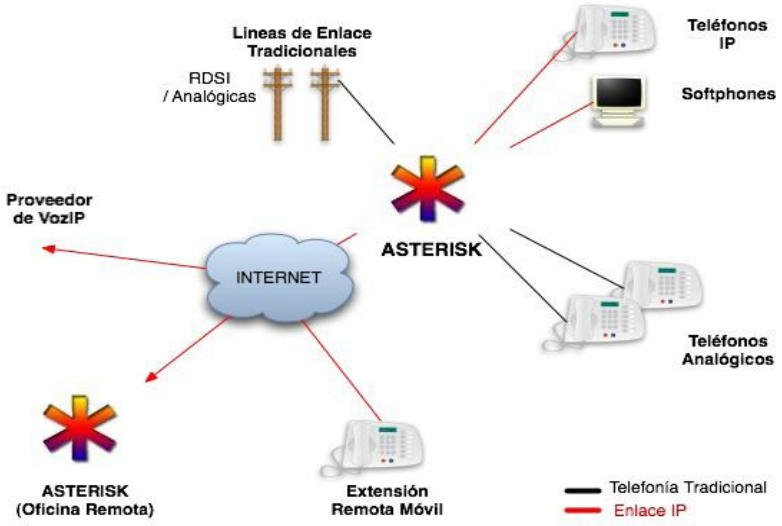



FIGURA 2.2 Interconexión entre redes de comunicación.

El servicio de telefonía de voz sobre IP basado en Asterisk provee de innumerables características.

### 2.1.1 FUNCIONALIDADES DE UTILIZAR ASTERISK.

Asterisk presenta varias funcionalidades que representan un valor agregado para las empresas que se deciden a implementarlo. Entre estas funcionalidades básicas podemos mencionar:

- Transferencia Ciega
- Música en espera
- Timbres Distintos

- Contestador automático de llamadas
- Llamada en espera
- Conferencias
- Buzón de Voz
- Colas de llamadas
- Colas con Prioridad
- Identificador de llamadas en espera

De igual manera, Asterisk posee funciones avanzadas como:

- Soporte de todos los protocolos estándar:
    - SIP (Session Initiation Protocol).
    - H.323
    - MGCP (Media Gateway Control Protocol).
    - IAX2 (Inter-Asterisk eXchange).
    - SCCP (Cisco Skinny).
- Soporta puenteo entre tecnologías distintas.
- Soporta transcodificación.

## 2.2 FXS/FXO

FXO es un dispositivo de computador que permite conectar éste a la RTB, y mediante un software especial, realizar y recibir llamadas de teléfono. Sirve sobre todo para implementar centralitas telefónicas (PBX) con un ordenador. Los dispositivos para conectar un teléfono a un ordenador son las llamadas FXS. Existen dispositivos que se denominan FXO y son usados en las puertas de enlace de VoIP, así como en tarjetas de ordenadores con funciones de centralitas telefónicas. Un claro ejemplo de FXO es un típico módem.

FXS es el conector en una central telefónica o en la pared de nuestro hogar, que permite conectar un teléfono analógico estándar.

Hay unas tarjetas que sirven para conectar teléfonos analógicos normales a un ordenador y, mediante un software especial, realizar y recibir llamadas hacia el exterior o hacia otras interfaces FXS. Las tarjetas para conectar un ordenador a la Red Telefónica Conmutada son las FXO. El esquema de conexión se muestra en la figura 2.3.


FIGURA 2.3 Esquema de conexión FXS/FXO sin PBX.

FXS y FXO son los nombres de los puertos usados por las líneas telefónicas analógicas de acuerdo a [2] (también denominados POTS - Servicio Telefónico Básico y Antiguo)

FXS, 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 mientras que FXO, 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, el teléfono se conecta directamente al puerto FXS que brinda la empresa telefónica.

Si tiene centralita, 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), como se muestra en la figura 2.4.


FIGURA 2.4 Esquema de conexión FXS/FXO con PBX.

## 2.3 PROTOCOLO SIP

SIP es un protocolo desarrollado por el grupo de trabajo MMUSIC del IETF 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 en línea y realidad virtual como se menciona en [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 Internet.

### 2.3.1 RELACION ENTRE SIP Y ASTERISK

El módulo de canal SIP de Asterisk permite comunicarse a través de VOIP con SIP y teléfonos de los intercambios. Asterisk es capaz de actuar como

- Un cliente SIP: Esto significa que los registros de Asterisk como un cliente a otro servidor SIP y recibe y realiza llamadas a este

servidor. Las llamadas entrantes se encaminan a una extensión de Asterisk.

- Un servidor SIP: Asterisk puede ser configurado para que los clientes SIP (teléfonos, los clientes de software) el registro en el servidor Asterisk y establecer sesiones con el servidor SIP, es decir, las llamadas y responde a las llamadas entrantes. Dicho esto, Asterisk no es una completa función de servidor SIP como SIP Express Router u OpenSER. Si usted va a tener miles de teléfonos SIP, usted debe utilizar SER o OpenSER y desviar llamadas a Asterisk de voz o el acceso PSTN.
- Una puerta de enlace SIP: Actos de Asterisk como puerta de enlace de los medios de comunicación entre la SIP, IAX, MGCP, H.323 y PSTN conexiones. A modo de ejemplo, un servidor de Asterisk se puede conectar a la RDSI para dar a su conectividad de clientes SIP a la red telefónica conmutada

### 2.3.2 FLUJO DE INICIO DE SESION

El flujo habitual del establecimiento de una sesión mediante el protocolo SIP es el siguiente (en este ejemplo todos los servidores actúan como proxy):

Un usuario ingresa la dirección lógica de la persona con la que quiere comunicarse, puede indicar al terminal también las características de las sesión que quiere establecer (voz, voz y video, etc.), o estas pueden estar implícitas por el tipo de terminal del que se trate. El agente de usuario SIP que reside en el terminal, actuando como UAC envía la petición (en este caso con el método INVITE) al servidor que tiene configurado. Este servidor se vale del sistema DNS para determinar la dirección del servidor SIP del dominio del destinatario. El dominio lo conoce pues es parte de la dirección lógica del destinatario. Una vez obtenida la dirección del servidor del dominio destino, encamina hacia allí la petición. El servidor del dominio destino establece que la petición es para un usuario de su dominio y entonces se vale de la información de registro de dicho usuario para establecer su ubicación física. Si la encuentra, entonces encamina la petición hacia dicha dirección. El agente de usuario destino si se encuentra desocupado comenzará a alertar al usuario destino y envía una respuesta hacia el usuario origen con un código de estado que indica esta situación (180 en este caso). La respuesta sigue el camino inverso hacia el usuario origen. Cuando el usuario destino finalmente acepta la invitación, se

genera una respuesta con un código de estado (el 200) que indica que la petición fue aceptada. La recepción de la respuesta final es confirmada por el UAC origen mediante una petición con el método ACK (de Acknowledgement/Acuse de recibo), esta petición no genera respuestas y completa la transacción de establecimiento de la sesión.

Normalmente la petición con el método INVITE lleva un cuerpo donde viaja una descripción de la sesión que quiere establecer, esta descripción es realizada con el protocolo SDP.6 En ella se indica el tipo de contenido a intercambiar (voz, video, etc.) y sus características (códecs, direcciones, puertos donde se espera recibirlos, velocidades de transmisión, etc.). Esto se conoce como "oferta de sesión SDP". La respuesta a esta oferta viaja, en este caso, en el cuerpo de la respuesta definitiva a la petición con el método INVITE. La misma contiene la descripción de la sesión desde el punto de vista del destinatario. Si las descripciones fueran incompatibles, 7 la sesión debe terminarse (mediante una petición con el método BYE).

Al terminar la sesión, que lo puede hacer cualquiera de las partes, el agente de usuario de la parte que terminó la sesión, actuando como UAC, envía hacia la otra una petición con el método BYE. Cuando lo recibe el UAS genera la respuesta con el código de estado correspondiente.

Si bien se ha descrito el caso de una sesión bipartita, el protocolo permite el establecimiento de sesiones multipartitas. También permite que un usuario esté registrado en diferentes ubicaciones pudiendo realizar la búsqueda en paralelo o secuencial entre todas ellas.

### 2.3.3 CARACTERISTICAS PRINCIPALES DE SIP

Aspectos importantes referentes al protocolo SIP:

- El control de llamadas es sin estado (stateless), y proporciona escalabilidad entre los dispositivos telefónicos y los servidores.

- SIP necesita menos ciclos de CPU para generar mensajes de señalización de forma que un servidor podrá manejar más transacciones.
- Una llamada SIP es independiente de la existencia de una conexión en la capa de transporte.
- SIP soporta autentificación de llamante y llamado mediante mecanismos HTTP.
- Autenticación, criptográfica y cifrado son soportados salto a salto por SSL/TSL pero SIP puede usar cualquier capa de transporte o cualquier mecanismo de seguridad de HTTP, como SSH o S-HTTP.
- Un proxy SIP puede controlar la señalización de la llamada y puede bifurcar a cualquier número de dispositivos simultáneamente.

Las características de este protocolo las analizaremos en el capítulo de implementación a medida que se vaya desarrollando el tema.

# CAPITULO 3

# IMPLEMENTACION

## 3.1 INTRODUCCION

La finalidad de este proyecto se basa en brindar soluciones telefónicas a muy bajo costo, haciendo uso del software libre como herramienta primordial para la elaboración del mismo.

Para esta solución utilizamos como software principal Asterisk, que permite implementar centrales telefónicas a pequeña y gran escala dando muy buenos resultados, sin necesidad de recurrir a otros tipos de software propietarios que demandarían una mayor inversión.

Asterisk aumenta notablemente sus alcances al trabajar conjuntamente con otras herramientas como son: MySql, PHP entre otras. Las antes mencionadas son que ayudan a cumplir con todas las funcionalidades propuestas en el proyecto.

Las centrales telefónicas han sido reemplazadas por computadoras que conjuntamente con programas simulan su funcionamiento, dando pie a que este tipo de servicios sean cada vez más accesibles para las personas y así poder mejorar los servicios que ofrecen sus empresas.

## 3.2 HARDWARE

El proceso para la selección del hardware a utilizar para implementar las soluciones es muy delicado y en el cual se deben tener en cuenta muchos factores que pueden influir en el funcionamiento de la central telefónica.

Se debe precisar los tipos de servicios a brindar y los datos que estos van a procesar, y con esto definir las características del hardware apropiado.

### 3.2.1 SERVIDOR

Se ha trabajado con tres servidores: un servidor web, un servidor de bases de datos y el servidor que funciona como central telefónica, todos estos instalados sobre una misma computadora. Cuyas características se detallan en la tabla I.

| | |
|---|---|
| CPU | AMD SEMPRON de 2.1 GHz |
| RAM | 1 GB. |
| Disco Duro | 20 GB. |

**Tabla I. Características del servidor**

### 3.2.2 TELEFONO IP

Para nuestras pruebas realizadas en el laboratorio utilizamos un teléfono IP de la marca GrandStream modelo GXP2000 el cual se muestra en la figura 3.1.

FIGURA 3.1. Teléfono IP GPX2000

## 3.3 SOFTWARE

### 3.3.1 SERVIDOR PBX

En la tabla II se detalla los componentes utilizados en nuestro servidor.

| | |
|---|---|
| Sistema Operativo | Centos 5.5 |
| Software PBX | Asterisk versión 1.8 |

**Tabla II. Servidor Asterisk**

Para que Asterisk funcione correctamente se deben instalar las librerías que se detallan en el capítulo posterior.

### 3.3.2 SERVIDOR WEB

La tabla III muestra los detalles del servidor apache.

| | |
|---|---|
| Sistema Operativo | Centos 5.5 |
| Software Servidor Web | Apache2.2.3 |
| Add-On | PHPmyadmin |

**Tabla III Servidor Web**

### 3.3.3 BASE DE DATOS

Se utilizó MySql como servidor de base de datos el cual almacena los datos necesarios para el desarrollo del proyecto como se muestra en la tabla IV.

| | |
|---|---|
| Sistema Operativo | Centos 5.5 |
| Motor de base de datos | My Sql server 5.0.77 |

**Tabla IV Servidor Base de Datos**

### 3.3.4 SOFTPHONES

Los softphones son simuladores de teléfonos IP los cuales trabajan con la dirección IP de la máquina sobre la cual está instalado y nos han sido de mucha ayuda para nuestras pruebas ya que son aplicaciones que se las pueden descargar gratuitamente desde internet mientras que los teléfonos IP reales son relativamente costosos.

Para este caso utilizamos el softphone X-Lite.

## 3.4 INSTALACIÓN

### 3.4.1 INSTALACIÓN DE LIBRERÍAS BASE

Para la instalación de las librerías base, se utiliza el comando `yum` el cual permite hacer descargas, actualizaciones e instalaciones este comando se lo escribe en el intérprete de líneas de comando de Centos de la siguiente manera.

```
# yum install – y "nombre_librería"
```

Las librerías a ser instaladas son las siguientes:

| | |
|---|---|
| bison | openssl |
| bison-devel | openssl-devel |
| ncurses | gnutls-devel |
| ncurses-devel | gcc |
| libtermcap | gcc-c++ |
| libtermcap-devel | newt |
| zlib | newt-devel |
| zlib-devel | libtool |

### 3.4.2 INSTALACIÓN DE ASTERISK

Para la instalación se debe descargar el archivo comprimido desde la página principal de Asterisk.

Estos paquetes son automáticamente guardados en la ruta: `etc/usr/src` aquí también deberán ser desempaquetados.

Luego se obtiene el código fuente que complementa a Asterisk por medio de línea de comandos de la siguiente forma:

Asterisk add-ons:

```
#wget
http://downloads.asterisk.org/pub/telephony/ast
erisk /asterisk-addons-1.6.2.3.tar.gz
```

Dahdi:

```
#wget
http://downloads.asterisk.org/pub/telephony/dah
di-linux-complete/releases/dahdi-linux-
complete-2.4.1.2+2.4.1.tar.gz
```

Libpri:

```
#wget
http://downloads.asterisk.org/pub/telephony/lib
pri/ releases/libpri-1.4.11.5.tar.gz
```

Una vez descargados los paquetes necesarios se procede a desempaquetarlos de la siguiente forma:

```
#cd /usr/src/
```

```
#tar zxvf asterisk-1.8.3.2.tar.gz

#tar zxvf asterisk-addons-1.6.2.3.tar.gz

#tar zxvf dahdi-linux-complete-
2.4.1.2+2.4.1.tar.gz

#tar zxvf libpri-1.4.11.5.tar.gz
```

Las versiones de cada uno de los paquetes no necesariamente tienen que ser las mismas, pues cada uno de ellas se manejan como un proyecto diferente.

Se desinstala cualquier versión previa de Asterisk

```
# cd /usr/src/asterisk-1.”version
anterior”

# make uninstall

# make uninstall-all
```

Se desinstala cualquier versión previa de Dahdi

```
# cd /usr/src/dahdi-linux-1.”version
anterior”

# make uninstall
```

Finalmente se borran los directorios de las versiones anteriores

Los pasos concernientes a la desinstalación de versiones anteriores no son necesarios si no se posee una versión anterior instalada.

El orden de instalación es el siguiente:

- Libpri
- Dahdi
- Asterisk
- Asterisk add-ons

Al instalarlos en este orden nos aseguramos de que cualquier dependencia para libpri, dahdi o asterisk fueron instaladas antes de correr los scripts de configuración.

La instalación de Libpri es opcional ya que es una librería que agrega soporte para ISDN.

```
# cd /usr/src/libpri-version

# make clean

# make

# make install
```

Proceso de instalación de Dahdi

```
cd /usr/src/dahdi-linux-complete-version

# make clean

# make

# make install

# make config
```

Para la instalación de Asterisk debemos seguir los siguientes pasos.

```
# cd /usr/src/asterisk-version

# make clean

# ./configure
```

```
# make menuselect

# make install

# make samples

# make config
```

Al ejecutar el comando `make menuselect`, se abre una ventana en la cual podremos elegiremos las opciones que estén relacionadas con MySql, para poder habilitar la comunicación entre Asterisk y la base de datos.

Por ultimo instalaremos los add-ons

```
# cd /usr/src/asterisk-addons-version

# make clean

# ./configure

# make menuselect

# make install

# make samples
```

Una vez hecho todo esto, en el directorio /etc/asterisk/ se encuentran todos los archivos de configuración que posteriormente van a ser modificados.

### 3.4.3 INSTALACION DE SERVICIOS ADICIONALES

En esta parte se instalaron las diferentes herramientas que se van a utilizar para llevar a cabo este proyecto.

Motor de Base de Datos MySql

```
yum install mysql mysql-server
```

Servidor Web Apache2

```
yum install httpd
```

Finalmente instalamos PHP

```
yum install php
```

### 3.4.4 INSTALACION DEL SOFTPHONE X-LITE

Para instalar este softphone debemos descargarlo desde la dirección que se encuentra en [4] y luego solo ejecutar el archivo de instalación, después sólo seguir las instrucciones del asistente de instalación.

## 3.5 CONFIGURACION DE LOS ARCHIVOS DE ASTERISK

### 3.5.1 CONFIGURACION DEL ARCHIVO “SIP.CONF”

El sip.conf es el archivo de configuración de canal correspondiente al protocolo SIP (Protocolo de Inicio de Sesión/Session Initiation Protocol) el cual es el protocolo de señalización más popular en la actualidad como se menciona en [5]. Este archivo contiene todas las configuraciones acerca de los usuarios que van a utilizar el protocolo SIP y además van a estar registrados dentro de nuestra PBX, y se encuentra en el directorio “etc/asterisk/sip.conf”. Nosotros hemos definido tres usuarios o “peers” cuya descripción se la puede encontrar en el anexo A.

### 3.5.2 CONFIGURACION DEL ARCHIVO “EXTENSIONS.CONF”

El principal archivo de configuración del Asterisk es el extensions.conf. En él se plasma toda la lógica de funcionamiento del sistema mejor conocida como dialplan o plan de discado. Ver anexo B. En el extensions.conf se controlan todas las conexiones al asterisk, tanto entrantes como salientes, por tanto en él definimos todo el comportamiento de nuestra central telefónica IP según [6].

#### 3.5.2.1 DEFINICION DEL CONTEXTO GENERAL

Es la primera sección que nos encontramos al abrir el archivo, contiene varias opciones, entre las más importantes tenemos:

*static:* esta opción controla la operación del comando "dialplan save"(Asterisk 1.6) o "save dialplan"(Asterisk 1.4). Su valor predeterminado es no, es decir, no puedes ejecutar los comandos.

*writeprotect:* Si writeprotect=no y static=yes, se puede guardar el dialplan actual desde la CLI (Command Line Interface) con el comando "dialplan save". Las

definiciones de variables globales en la sesión [globals] no se ve afectada. Su valor predeterminado es no.

*autofallthrough:* Si colocamos *yes* una vez que la lógica de programación de una extensión termine, Asterisk terminará la llamada con BUSY, CONGESTION o HANGUP, que es la conducta deseada, caso contrario Asterisk esperará que marques otra extensión

*clearglobalvars:* Si colocamos yes el valor de las variables globales serán borradas cada vez que ocurra una recarga del dialplan, caso contrario, el valor de la variable global persistirá aunque eliminemos la variable del archivo.

## 3.6 CONFIGURACION DE LOS SCRIPTS

### 3.6.1 CONSULTAR UN CLIENTE EXISTENTE

Mediante el script “Consulta-1.agi” se verifica si un cliente se encuentra registrado en la base de datos del hotel, donde los parámetros para la búsqueda del mismo serán la cedula y la contraseña. El código fuente de este archivo se encuentra en el Anexo C.

La consulta del cliente se la realiza de acuerdo al diagrama de flujo como se muestra en la figura 3.2.

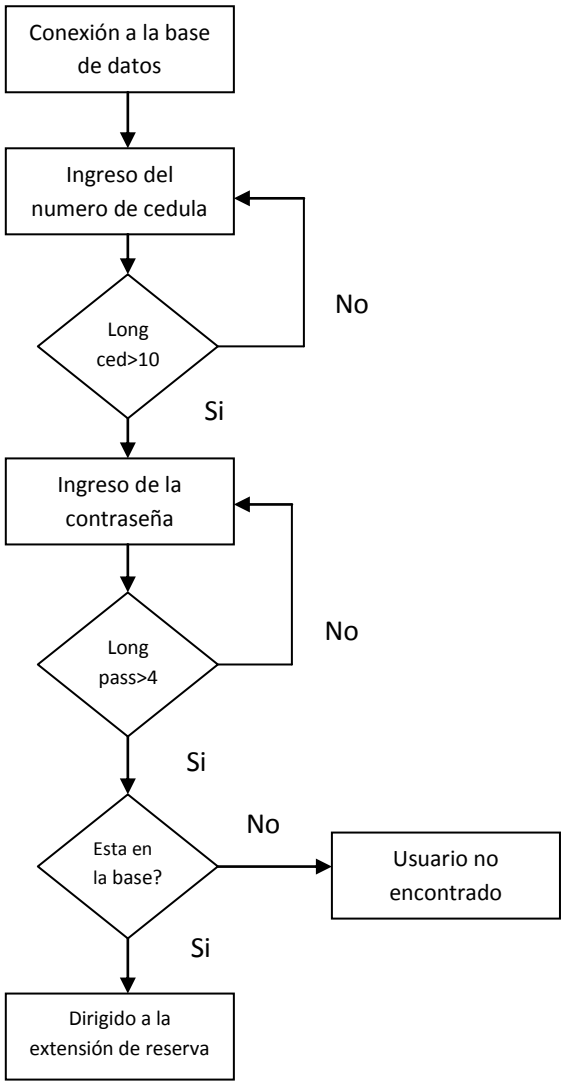



FIGURA 3.2 Proceso para consultar un cliente existente en la base de datos.

### 3.6.2 REALIZAR UNA RESERVA

En el script “Reserva.agi” se procede a realizar la reserva de una habitación para un usuario existente en la base de datos, de acuerdo a la disponibilidad de fecha y tipo de habitación requerida. Su código fuente se encuentra en el Anexo D.

La reserva se lleva a cabo según la figura 3.3.

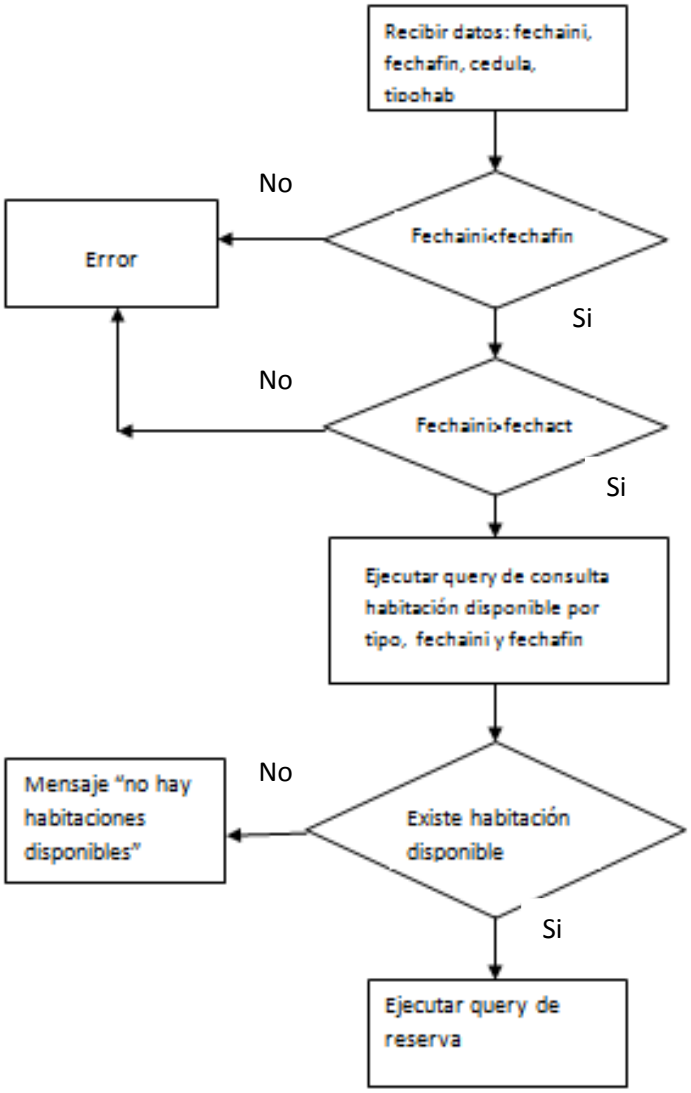



FIGURA 3.3 Proceso de la reserva telefónica.

### 3.6.3 CONSULTAR EL ESTADO DEL CLIMA EN TIEMPO REAL

La consulta del estado del clima se realiza mediante la ejecución del script “Wer.php”, el cual toma los datos de la temperatura junto con los valores mínimos y máximos para el día actual. Su código se encuentra en el Anexo E.

La consulta del clima se realizara de acuerdo a la figura 3.4.

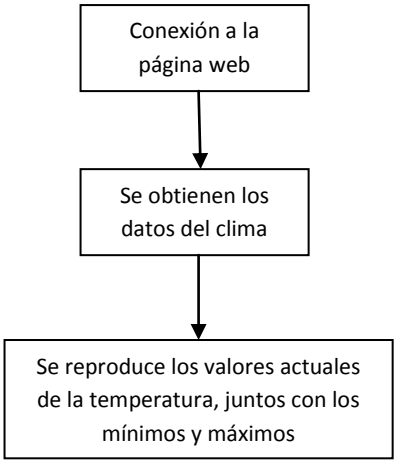



FIGURA 3.4 Consulta del clima en tiempo real.

### 3.6.4 PROGRAMACION DE DESPERTADORES AUTOMATICOS

Los despertadores automáticos se realizan mediante la ejecución del script “Wakeup.php” según lo obtenido en [6], en el cual el huésped ingresara la hora deseada para su despertador. El código fuente se encuentra en el Anexo F.

Esta opción sigue el diagrama de flujo de la figura 3.5.

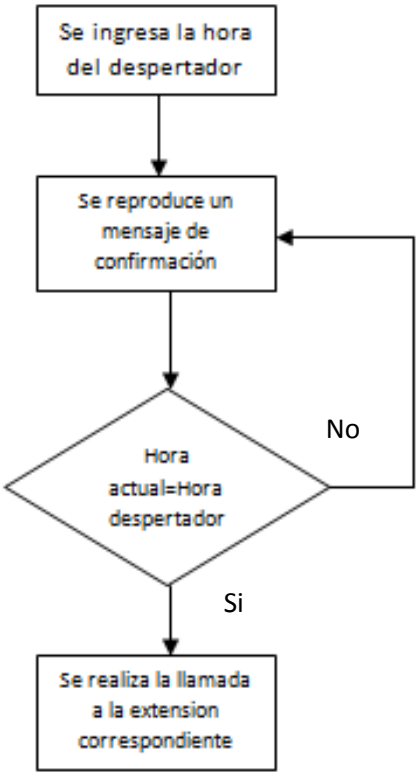



FIGURA 3.5 Programación de un despertador.

### 3.6.5 PAGINA WEB (FORMULARIO)

La página web mediante la cual los clientes ingresan los datos y que quedan registrados en la base de datos de clientes del hotel se muestra mediante la ejecución del archivo "registro.php" cuyo código se especifica en el Anexo G.

La funcionalidad de la página web se basa en el diagrama de flujo de la figura 3.6.

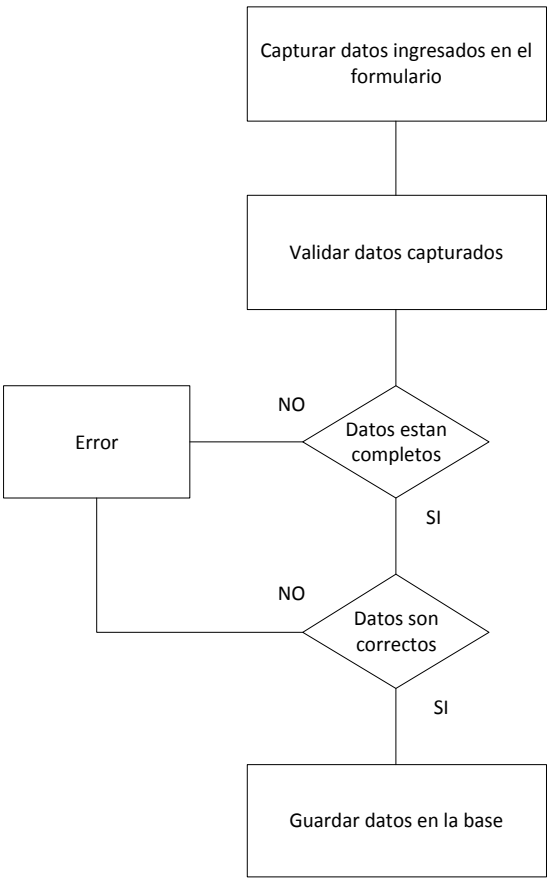



FIGURA 3.6 Proceso que cumple la página web.

# CAPITULO 4

# FUNCIONAMIENTO Y PRUEBAS

## Introducción

Después de haber concluido satisfactoriamente con la instalación de Asterisk, nuestro motor de base de datos MySQL, el servidor web Apache y todas las librerías necesarias para el correcto funcionamiento de nuestra PBX, además de haber modificado nuestros archivos de configuración y habiendo creado nuestros propios scripts para la realización de las consultas nos disponemos a presentar detalladamente cada una de las pruebas realizadas.

## 4.1 Inicializando Asterisk

A continuación mostraremos un conjunto de comandos que nos permiten iniciar o detener los servicios de Asterisk todos estos comandos son ingresados mediante el intérprete de líneas de comando.

| | |
|---|---|
| `service asterisk start` | Iniciar servicio |
| `service asterisk stop` | Detener servicio |
| `service asterisk restart` | Reiniciar servicio |
| `service asterisk status` | Consultar estado del servicio |

| | |
|---|---|
| `asterisk -c` | Iniciar Asterisk<br>Y abrir la consola remota |
| `asterisk -r` | Ingresar a la consola remota |
| `asterisk -rx'comando'` | Ejecutar comando sin ingresar a la consola |

Algunos comandos muy usados dentro de la consola de Asterisk

| | |
|---|---|
| `stop now` | Detener el servicio de Asterisk desde la consola |
| `exit` | Salir de la consola sin detener el servicio |
| `sip show peers` | Muestra el estado de los usuarios SIP |
| `reload` | Recarga las configuraciones de Asterisk |

`restart` Reinicia servicio

## 4.2 Configuración del Softphone X-lite

De acuerdo al archivo de configuración SIP.conf, se ha registrado un usuario con el nombre “fgarnik”, el cual va a ser registrado en el servidor donde está instalado Asterisk, como se muestra en la figura 4.1.


FIGURA 4.1 Usuario registrado.

En la figura 4.2 se muestra como se debe configurar el softphone X-Lite aquí vemos los campos en donde se pone la dirección IP del servidor Asterisk y también se coloca el usuario y contraseña con los que se va a

autenticar en el servidor, estos deben ser los mismos que se encuentran en el archivo sip.conf para el usuario respectivo para este ejemplo “fgarnik”.

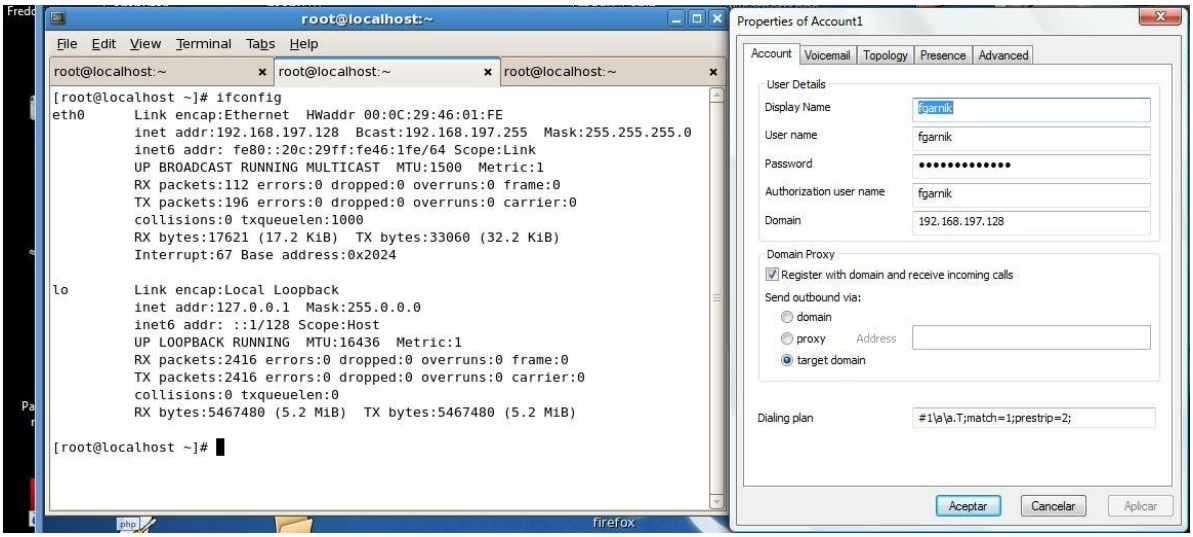



FIGURA 4.2 Configuración del usuario SIP.

## 4.3 Consulta de la temperatura en tiempo real

Como se detalló en el capítulo tres mediante un diagrama de flujo, procederemos a mostrar el proceso para la consulta del clima.

Marcamos las extensión “0” desde nuestro softphone, luego del cual se reproduce un mensaje de bienvenida ofreciéndonos dos opciones, como se muestra en la figura 4.3.

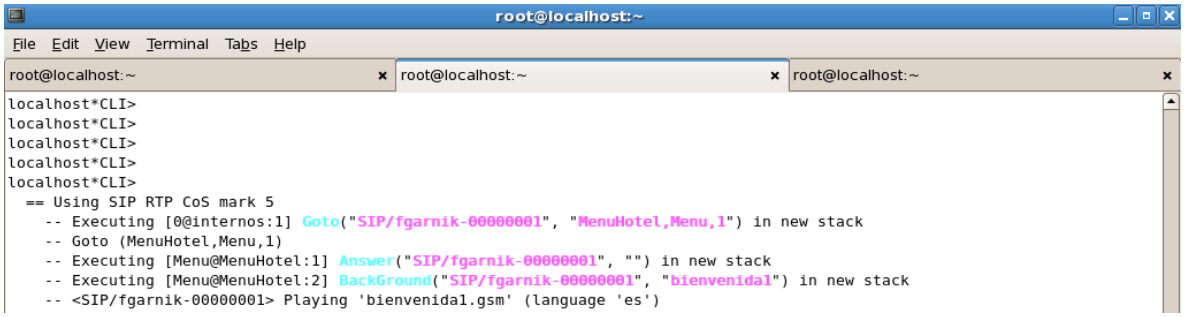



FIGURA 4.3 Ingreso al menú del hotel

Luego de esto al usuario se le permite elegir entre dos opciones, para esta prueba se ingresará la opción dos, la cual reproduce la temperatura actual junto con la temperatura máxima y mínima para ese mismo día.

Una vez ingresado el número dos, Asterisk ejecuta el archivo “Wer.agi” que contiene las instrucciones para poder conectarse a la página web y mediante los comandos de PHP poder obtener los datos de la temperatura. Los archivos de audio son reproducidos uno a la vez junto con los valores obtenidos desde la web como se puede apreciar en la figura 4.4.

```
    -- Executing [2@MenuHotel:1] Goto("SIP/fgarnik-00000001", "internos,clima,1") in new stack
    -- Goto (internos,clima,1)
    -- Executing [clima@internos:1] Answer("SIP/fgarnik-00000001", "") in new stack
    -- Executing [clima@internos:2] AGI("SIP/fgarnik-00000001", "Wer.agi") in new stack
    -- Launched AGI Script /var/lib/asterisk/agi-bin/Wer.agi
    -- Playing 'temperature' (escape_digits=) (sample_offset 0)
    -- Playing 'currently' (escape_digits=) (sample_offset 0)
    -- <SIP/fgarnik-00000001> Playing 'digits/27.ulaw' (language 'es')
    -- Playing 'degrees' (escape_digits=) (sample_offset 0)
    -- Playing 'celsius' (escape_digits=) (sample_offset 0)
    -- Playing 'temperature' (escape_digits=) (sample_offset 0)
    -- Playing 'minimum' (escape_digits=) (sample_offset 0)
    -- <SIP/fgarnik-00000001> Playing 'digits/21.ulaw' (language 'es')
    -- Playing 'degrees' (escape_digits=) (sample_offset 0)
    -- Playing 'temperature' (escape_digits=) (sample_offset 0)
    -- Playing 'maximum' (escape_digits=) (sample_offset 0)
    -- <SIP/fgarnik-00000001> Playing 'digits/28.ulaw' (language 'es')
    -- Playing 'degrees' (escape_digits=) (sample_offset 0)
    -- Playing 'vm-goodbye' (escape_digits=) (sample_offset 0)
[Aug 19 13:52:05] ERROR[4926]:        :                    : write() returned error: Broken pipe
    -- <SIP/fgarnik-00000001>AGI Script Wer.agi completed, returning 0
    -- Executing [clima@internos:3] Hangup("SIP/fgarnik-00000001", "") in new stack
  == Spawn extension (internos, clima, 3) exited non-zero on 'SIP/fgarnik-00000001'
localhost*CLI>
```

FIGURA 4.4 Ejecución del script “Wer.agi”.

Para comprobar los datos, se consulta instantáneamente los valores que se encuentran en la página web como se muestra en la figura 4.5, y se verifica que son los mismos que han sido reproducidos por Asterisk.

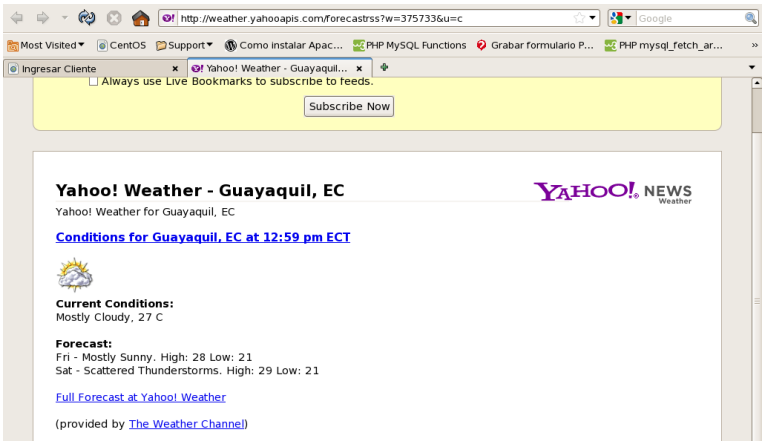



FIGURA 4.5 Condiciones del clima.

## 4.4 Ingreso de un nuevo cliente a la base de datos del hotel.

En esta parte se realiza el ingreso de un nuevo cliente en el hotel.

Una vez comprobado que la base de datos está corriendo se procede a ingresar al cliente, abrir el navegador e ingresar la página del hotel cuya dirección es http://localhost/registro.php, como resultado se obtiene lo que se muestra en la figura 4.6.

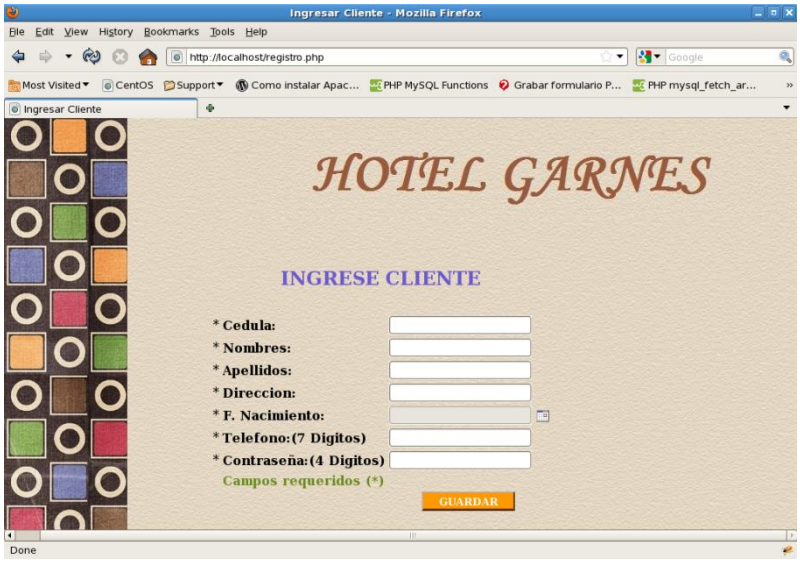



FIGURA 4.6 Página web del hotel.

Se ingresan los datos del nuevo cliente como se aprecia en la figura 4.7, en este caso todos los datos son obligatorios y todos los campos tienen las validaciones correspondientes para que el usuario no ingrese datos erróneos.

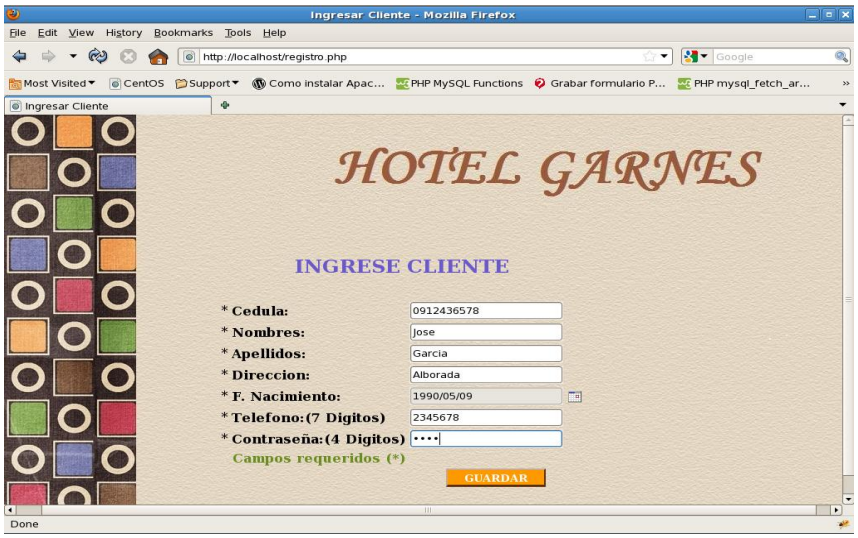



FIGURA 4.7. Ingreso de los datos.

Si el cliente ingresa algún dato que no es correcto, o datos incompletos el sistema no dejara que sean ingresados, caso contrario los datos son ingresados de forma satisfactoria como se muestra en la figura 4.8.

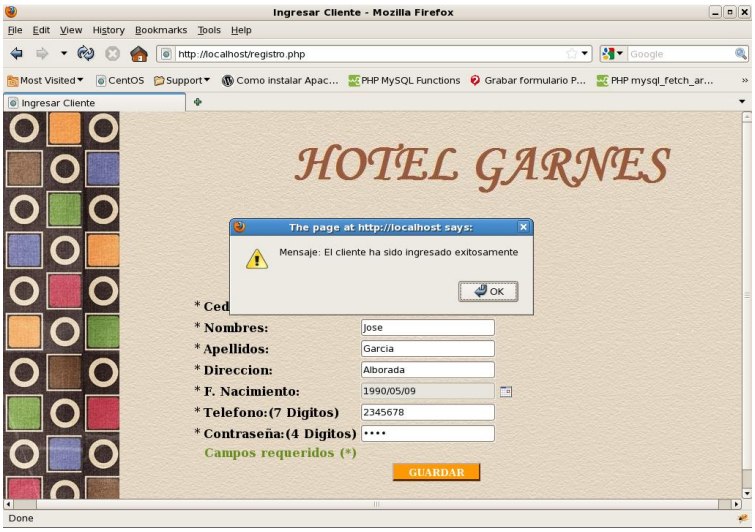



FIGURA 4.8 Datos ingresados correctamente.

Como se puede apreciar en la figura 4.9 el cliente no se encuentra registrado previamente en la base de datos.

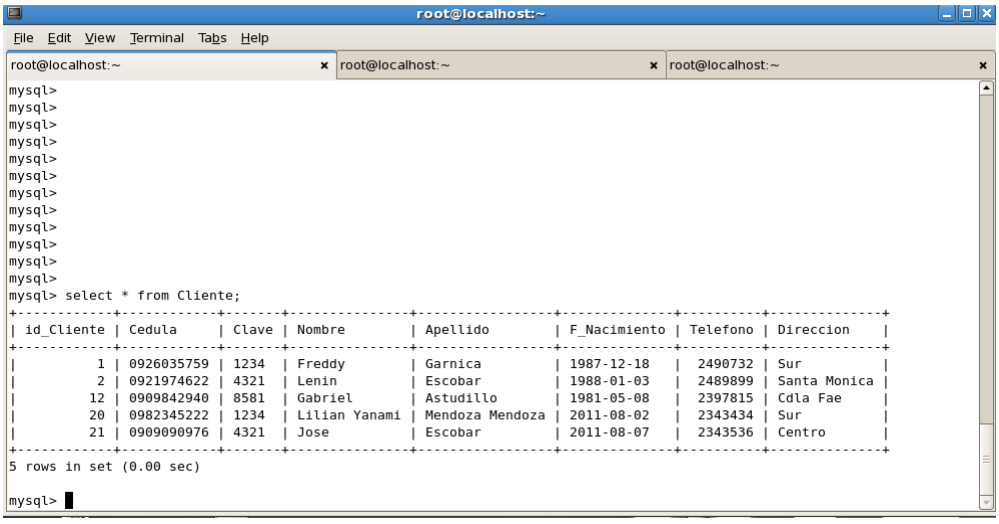



FIGURA 4.9 Base de datos sin el cliente.

Después de haber hecho el ingreso satisfactorio de los datos, el cliente ya aparece en la base de datos de acuerdo a la figura 4.10, con toda la información descrita previamente.

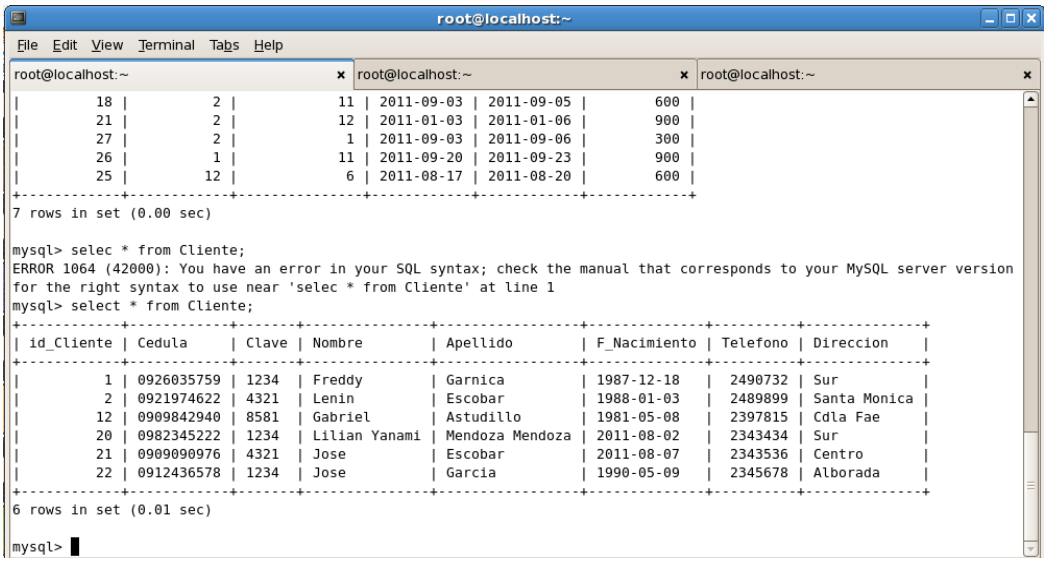



FIGURA 4.10 Base de datos ya con el cliente.

## 4.5 Programación de despertador

En este punto se revisa el proceso para la programación de un despertador desde una extensión la cual se encuentra dentro del hotel.

Marcar la extensión 0 para escuchar el menú principal, luego escoger la opción 1 que es la opción para la programación de despertadores, se

ingresa la hora deseada para el despertador y en ese momento el despertador ya se encuentra programado como lo muestra la figura 4.11.

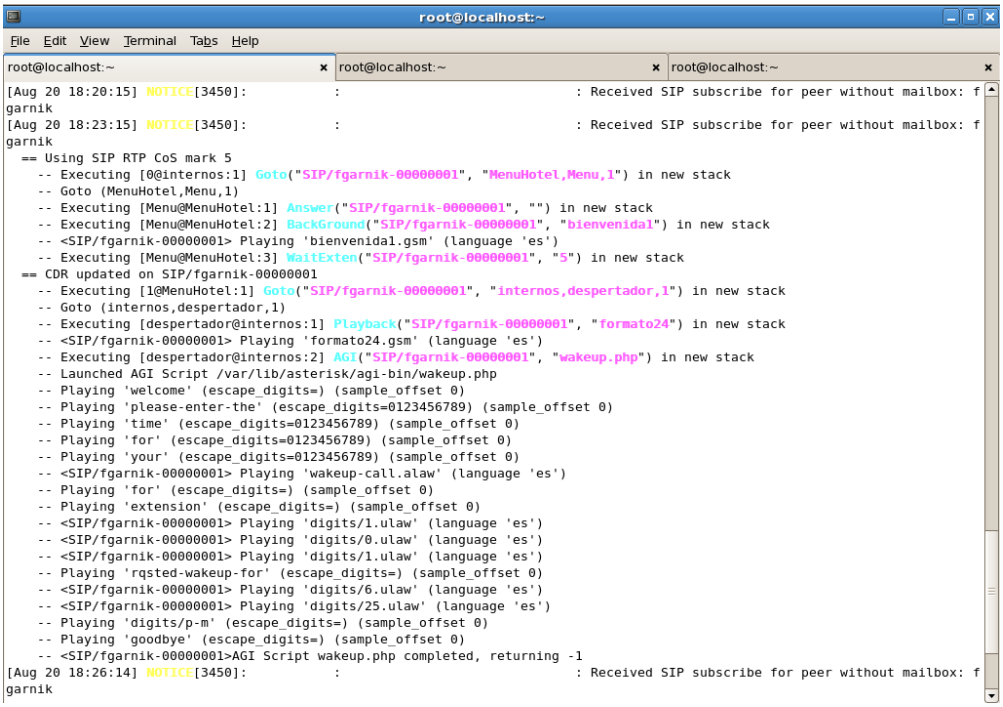



FIGURA 4.11 Ingreso del despertador para las 18:25.

Cuando sea la hora a la que se programó el despertador, el servidor genera una llamada hacia la extensión como se muestra en la figura 4.12a, y al alzar el auricular se reproduce una melodía similar al de un tono de alarma como se muestra en la figura 4.12b.

```
[Aug 20 18:26:14] NOTICE[3450]:           :                              : Received SIP subscribe for peer without mailbox: f
garnik
    -- Attempting call on Local/101@internos for application MusicOnHold() (Retry 1)
    -- Executing [101@internos:1] Dial("Local/101@internos-e214;2", "SIP/fgarnik") in new stack
  == Using SIP RTP CoS mark 5
    -- Called fgarnik
    -- SIP/fgarnik-00000002 is ringing
    -- SIP/fgarnik-00000002 answered Local/101@internos-e214;2
    -- Started music on hold, class 'default', on Local/101@internos-e214;1
  == Spawn extension (internos, 101, 1) exited non-zero on 'Local/101@internos-e214;2'
    -- Stopped music on hold on SIP/fgarnik-00000002
[Aug 20 18:27:16] NOTICE[4179]:           :                 : Call completed to Local/101@internos
localhost*CLI>
```

FIGURA 4.12a Consola de Asterisk, generación de la llamada


FIGURA 4.12b. Llamada entrante del despertador.

## 4.6 Programación de una reserva

En este punto se analiza la programación de una reserva para un cliente el cual ya debe estar previamente ingresado en la base de datos del hotel con su respectivo usuario y contraseña.

Debe llamar a la extensión indicada para realizar la reserva, luego ingresar los datos de usuario (cedula) y contraseña (ver figura 4.13), luego de ser validados los datos se le indica que ingrese el tipo de habitación que desea reservar 1 “sencilla”, 2 “doble” o 3 “triple”, después debe ingresar las fechas tanto de llegada al hotel como de salida del mismo, si existen habitaciones disponibles para esas fechas se le indica los datos ingresados por el cliente y el costo total de la reserva, si el cliente está de acuerdo presionará 1 caso contrario 2; Si el cliente elije la opción 1 la reservación se realiza como se observa en las figuras 4.14 y 4.15.

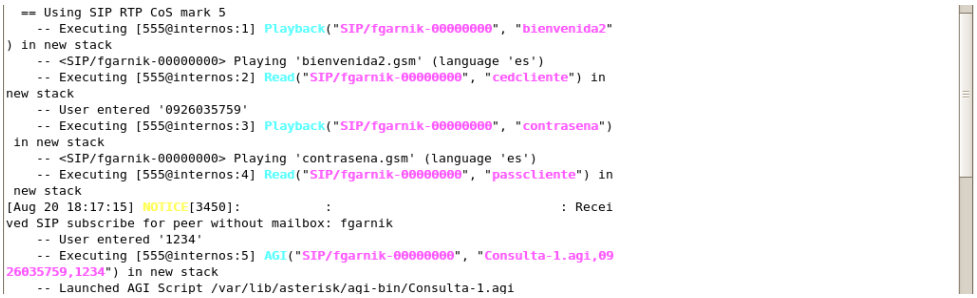



FIGURA 4.13 Ingreso de usuario y contraseña.

```
    -- AGI Script Executing Application: (GoTO) Options: (reserva,1)
    -- Goto (internos,reserva,1)
    -- <SIP/fgarnik-00000000>AGI Script Consulta-1.agi completed, returning 0
    -- Executing [reserva@internos:1] Playback("SIP/fgarnik-00000000", "tipohabi
tacion") in new stack
    -- <SIP/fgarnik-00000000> Playing 'tipohabitacion.gsm' (language 'es')
    -- Executing [reserva@internos:2] Read("SIP/fgarnik-00000000", "tipohab") in
 new stack
    -- User entered '1'
    -- Executing [reserva@internos:3] GotoIf("SIP/fgarnik-00000000", "0?badHabi,
1:goodHabi") in new stack
    -- Goto (internos,reserva,4)
    -- Executing [reserva@internos:4] Playback("SIP/fgarnik-00000000", "diaini")
 in new stack
    -- <SIP/fgarnik-00000000> Playing 'diaini.gsm' (language 'es')
    -- Executing [reserva@internos:5] Read("SIP/fgarnik-00000000", "diaIni") in
new stack
    -- User entered '23'
    -- Executing [reserva@internos:6] GotoIf("SIP/fgarnik-00000000", "0?badDay,1
:goodDay") in new stack
    -- Goto (internos,reserva,7)
    -- Executing [reserva@internos:7] Playback("SIP/fgarnik-00000000", "mesini")
 in new stack
    -- <SIP/fgarnik-00000000> Playing 'mesini.gsm' (language 'es')
    -- Executing [reserva@internos:8] Read("SIP/fgarnik-00000000", "mesIni") in
new stack
    -- User entered '8'
    -- Executing [reserva@internos:9] GotoIf("SIP/fgarnik-00000000", "0?badMonth
,1:goodMonth") in new stack
    -- Goto (internos,reserva,10)
    -- Executing [reserva@internos:10] Playback("SIP/fgarnik-00000000", "diafin"
) in new stack
    -- <SIP/fgarnik-00000000> Playing 'diafin.gsm' (language 'es')
    -- Executing [reserva@internos:11] Read("SIP/fgarnik-00000000", "diaFin") in
 new stack
    -- User entered '25'
    -- Executing [reserva@internos:12] GotoIf("SIP/fgarnik-00000000", "0?badDayF
in,1:goodDayFin") in new stack
    -- Goto (internos,reserva,13)
    -- Executing [reserva@internos:13] Playback("SIP/fgarnik-00000000", "mesfin"
) in new stack
    -- <SIP/fgarnik-00000000> Playing 'mesfin.gsm' (language 'es')
    -- Executing [reserva@internos:14] Read("SIP/fgarnik-00000000", "mesFin") in
 new stack
    -- User entered '8'
    -- Executing [reserva@internos:15] GotoIf("SIP/fgarnik-00000000", "0?badMont
hFin,1:goodMonthFin") in new stack
```

FIGURA 4.14 Momento en que el usuario ingresa los datos.

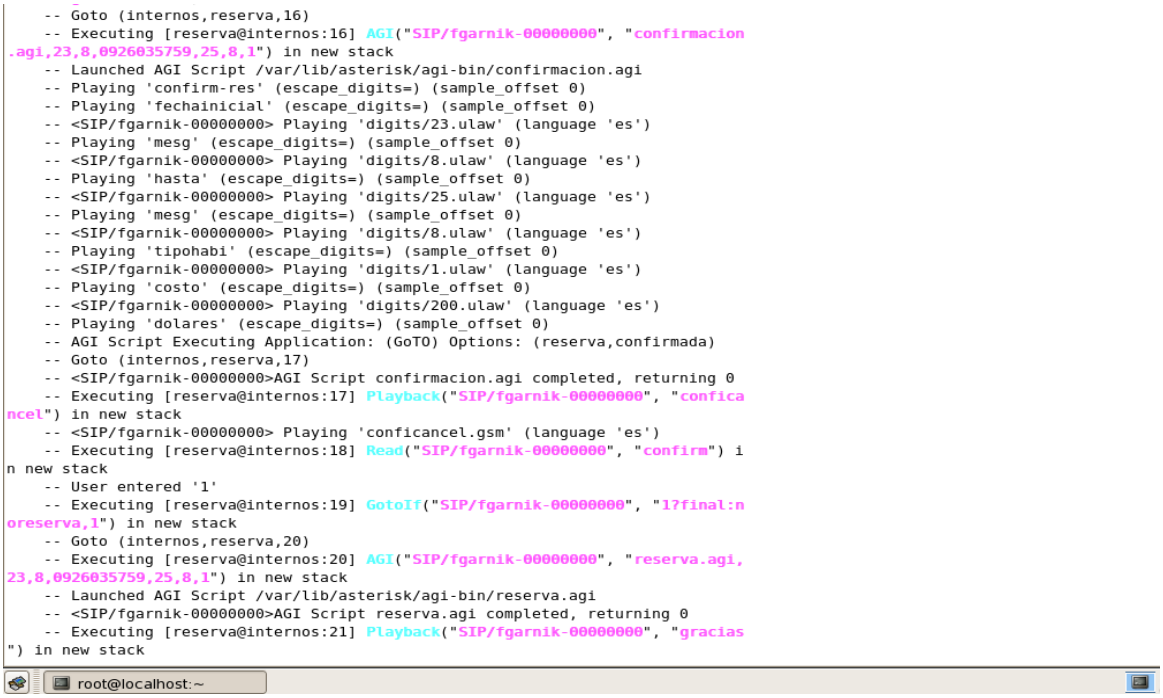



FIGURA 4.15 Reproducción del mensaje de confirmación, y creación de la reserva.

# CONCLUSIONES Y RECOMENDACIONES

## CONCLUSIONES:

1. El desarrollo de este tipo de aplicaciones automatizadas está creciendo a un ritmo cada vez mayor, y el uso de herramientas de este tipo son de mucha utilidad en estos días, donde las empresas están destinadas a estar a la vanguardia de la tecnología, en nuestro caso al implementar un sistema de reservas telefónico que si bien es cierto, es básico a la hora de realizar la reserva pero con posteriores modificaciones llegaría a ser un software muy útil para cualquier tipo de hotel.

2. El uso del software ha sido muy beneficioso para nosotros, porque podemos demostrar que mediante el uso del mismo se pueden lograr aplicaciones muy novedosas sin necesidad de recurrir a software privativos que pueden alcanzar grandes valores monetarios, y además de eso, en el internet se puede encontrar un sinnúmero de tutoriales y foros en los cuales encontrar información acerca de Asterisk y sus aplicaciones que en otros países se han desarrollado con anterioridad.

3. Mediante el uso de lenguaje de programación PHP, y el motor de base de datos MySQL logramos interactuar entre nuestra PBX y los datos almacenados. Utilizando estas herramientas conseguimos la información con respecto a las condiciones actuales del clima, realizar nuevas reservas para clientes registrados previamente en nuestra base de datos y la programación automática de despertadores de una manera sencilla, con esto llegamos a la conclusión que es de gran importancia aprender un poco más acerca de estos lenguajes, ya que Asterisk se complementa con los mismos.

4. Pudimos darnos cuenta del alcance de Asterisk y así entender el porqué esta aplicación ha tomado tanta fuerza y su uso va en aumento cada día más. Asterisk es una herramienta muy poderosa la cual se puede aprovechar conforme se conozca más de ella.

5. Se logró crear un sistema de reserva automatizado y de fácil acceso para todos los usuarios que ya se encuentran en los registros del hotel.

6. Se estableció un sistema de despertador capaz de realizar llamadas a todas las extensiones del hotel de acuerdo a la hora ingresada por los diferentes usuarios.

7. Se optimizo el servicio de reservas, en su nivel más básico, al no necesitar de una persona que se encuentre tomando los datos del cliente.

## RECOMENDACIONES:

1. Aprender los lenguajes de programación PHP y SQL, pues Asterisk aumenta su funcionalidad haciendo uso de estos lenguajes.
2. Usar software libre para este tipo de soluciones ya que mediante el uso del mismo logramos abaratar costos de forma muy considerable.
3. Determinar las características del hardware basados en el número de usuarios y tipo de servicios que el sistema va a brindar.
4. Verificar que la página que nos va servir como fuente de información del clima se encuentre actualizada y vigente ya que si esta página llegase a sufrir alguna modificación nuestro servicio de consulta del clima se vería afectado y deberíamos realizar las modificaciones necesarias para que nuestro servicio continúe funcionando de forma correcta.
5. Aprender Linux es de vital importancia ya que muchas aplicaciones y en particular Asterisk por lo general se encuentran instaladas sobre equipos con Linux. En sus distintas distribuciones.

# GLOSARIO DE TERMINOS

**VOIP:** Voz sobre protocolo de internet, también llamado Voz IP, VozIP, VoIP (por sus siglas en inglés), es un grupo de recursos que hacen posible que la señal de voz viaje a través de Internet empleando un protocolo IP (Internet Protocol).

**Dialplan:** O plan de marcado, es la configuración que permite determinar el tratamiento que debe darse a un número discado.

Open Source: Código abierto (en inglés Open source) es el término con el que se conoce al software distribuido y desarrollado libremente.

**PBX:** Es cualquier central telefónica conectada directamente a la red pública de telefonía por medio de líneas troncales para gestionar, además de las llamadas internas, las entrantes y/o salientes con autonomía sobre cualquier otra central telefónica.

**DTMF:** (Dual Tone Multifrecuency) Multifrecuencia de doble tono. Tonos en diferentes hertz que utiliza una telefonía para marcar números. Cada número u opción de teléfono tiene su tono que es identificado en la telefonía.

**AGI:** El AGI (Interfaz de puerta de enlace de Asterisk/Asterisk Gateway interface) permite extender las funcionalidades de Asterisk mediante el uso

de lenguajes de programación. El AGI sirve de enlace éntrelas aplicaciones externas y el núcleo de Asterisk.

**SOFTWARE PROPIETARIO:** Se refiere a cualquier programa informático en el que los usuarios tienen limitadas posibilidades de usarlo, modificarlo o redistribuirlo, o cuyo código fuente no está disponible o el acceso a este se encuentra restringido.

**GATEWAY:** Puerta de enlace, acceso, pasarela. Nodo en una red informática que sirve de punto de acceso a otra red.

**MySQL:** Es un sistema de gestión de bases de datos relacional, multihilo y multiusuario.

**H.323:** Recomendación de la Unión Internacional de Telecomunicaciones (ITU), que define los protocolos para proveer sesiones de comunicación audiovisual sobre paquetes de red. No garantiza una calidad de servicio, además es independiente de la topología de la red y admite pasarelas, permitiendo usar más de un canal de cada tipo (voz, video, datos) al mismo tiempo.

# BIBLIOGRAFIA

[1] Wikipedia, Definicion de Asterisk, http://es.wikipedia.org/wiki/Asterisk, revisado en Junio del 2011.

[2]3CX Ltda., Definicion de FXS/FXO, http://www.3cx.es/voip-sip/fxs-fxo.php, revisado en Junio del 2011.

[3] Wikipedia, Definicion del protocolo SIP, http://es.wikipedia.org/wiki/Session_Initiation_Protocol, revisado en Junio del 2011.

[4] Ramón Jiménez, Definición del archivo SIP.conf, http://jimenezra.blogspot.com/2011/02/telefonia-ip-asterisk-sipconf.html, revisado en Julio del 2011.

[5] Ramón Jiménez, Definición del archivo EXTENSIONS.conf, http://jimenezra.blogspot.com/2011/02/telefonia-ip-asterisk-extensionsconf.html , revisado en Agosto del 2011.

[6] VoIp-info.org, Descarga del archivo “Wakeup.php”, http://www.voip-info.org/wiki/view/Asterisk+tips+Wake-Up+Call+PHP, revisado en Junio del 2011.

[7] VoIp-info.org, Definiciones varias, http://www.voip-info.org, revisado en Agosto del 2011.

# ANEXOS

## Anexo A

En este anexo mostraremos el código del archivo sip.conf

```
[general]
context=noautenticadas
allowguest=no
srvlookup=yes
udpbindaddr=0.0.0.0
tcpenable=no
qualify=yes
autofallthrough=yes
clearglobalvars=no
language=es

[lescobar]
type=friend
callerid="lenin"<100>
context=internos
host=dynamic
nat=yes
secret=12345
dtmfmode=auto
disallow=all
allow=gsm
allow=ulaw
allow=alaw

[fgarnik]
type=friend
callerid="freddy"<101>
context=internos
host=dynamic
nat=yes
secret=clavesecreta1
dtmfmode=auto
disallow=all
allow=ulaw
allow=gsm
allow=alaw
```

## Anexo B

En este anexo mostraremos el código del archivo extensions.conf donde se muestra el plan de marcado y como se van a comportar las llamadas.

```
[general]
autofallthrough=yes
clearglobalvars=no

[internos]

exten=> 0,1,Goto(MenuHotel,Menu,1)

;*************************************************
;Extension para programar despertadores
exten=> despertador,1,Playback(formato24)
exten=> despertador,n,agi(wakeup.php)
exten=> despertador,n,Hangup()

;*************************************************
;Extension para consulta del clima
exten=> clima,1,Answer()
exten=> clima,n,AGI(Wer.agi)
exten=> clima,n,Hangup()

;*************************************************
exten=> 100,1,Dial(SIP/lescobar)
exten=> 100,n,Hangup()

exten=> 101,1,Dial(SIP/fgarnik)
exten=> 101,2,Hangup()

;*************************************************
;Extension Para hacer una reserva
;Si el idCliente es encontrado, va a la extension
;caso contrario ira a la extension
;*************************************************

exten=> 555,1,Playback(bienvenida2)
exten=> 555,n,Read(cedcliente)
exten=> 555,n,Playback(contrasena)
exten=> 555,n,Read(passcliente)
exten=> 555,n,AGI(Consulta-1.agi,${cedcliente},${passcliente})

exten=> reserva,1,Playback(tipohabitacion)
exten=> reserva,n,Read(tipohab)
exten=> reserva,n,GoToIf($[${tipohab}>3]?badHabi,1:goodHabi)
exten=> reserva,n(goodHabi),Playback(diaini)
exten=> reserva,n,Read(diaIni)
exten=> reserva,n,GoToIf($[${diaIni}>30]?badDay,1:goodDay)
```

```
exten=> reserva,n(goodDay),Playback(mesini)
exten=> reserva,n,Read(mesIni)
exten=> reserva,n,GoToIf($[${mesIni}>12]?badMonth,1:goodMonth)
exten=> reserva,n(goodMonth),Playback(diafin)
exten=> reserva,n,Read(diaFin)
exten=> reserva,n,GoToIf($[${diaFin}>30]?badDayFin,1:goodDayFin)
exten=> reserva,n(goodDayFin),Playback(mesfin)
exten=> reserva,n,Read(mesFin)
exten=> reserva,n,GoToIf($[${mesFin}>12]?badMonthFin,1:goodMonthFin)
exten=>
reserva,n(goodMonthFin),AGI(confirmacion.agi,${diaIni},${mesIni},${c
edcliente},${diaFin},${mesFin},${tipohab})
exten=> reserva,n(confirmada),Playback(conficancel)
exten=> reserva,n,Read(confirm)
exten=> reserva,n,GoToIf($[${confirm}=1]?final:noreserva,1)
exten=>
reserva,n(final),AGI(reserva.agi,${diaIni},${mesIni},${cedcliente},$
{diaFin},${mesFin},${tipohab})
exten=> reserva,n,Playback(gracias)
exten=> reserva,n,Hangup()

exten=> noreserva,1,Playback(cancelada)
exten=> noreserva,n,Hangup()

exten=> fechaerror,1,PLayback(errorfecha)
exten=> fechaerror,n,Hangup()

exten=> nohab,1,PLayback(sinhabitacion)
exten=> nohab,n,Hangup()

exten=> baduser,1,PLayback(errorusuario)
exten=> baduser,n,Hangup()

exten=> badHabi,1,Answer()
exten=> badHabi,n,Playback(errorhabitacion)
exten=> badHabi,n,Goto(reserva,1)

exten=> badDay,1,Answer()
exten=> badDay,n,Playback(errordia)
exten=> badDay,n,Goto(reserva,goodHabi)

exten=> badMonth,1,Answer()
exten=> badMonth,n,Playback(errormes)
exten=> badMonth,n,Goto(reserva,goodDay)

exten=> badDayFin,1,Answer()
exten=> badDayFin,n,Playback(errordia)
exten=> badDayFin,n,Goto(reserva,goodMonth)

exten=> badMonthFin,1,Answer()
exten=> badMonthFin,n,Playback(errormes)
exten=> badMonthFin,n,Goto(reserva,goodDayFin)
```

```
[MenuHotel]

exten=> Menu,1,Answer()
exten=> Menu,n,Background(bienvenida1)
exten=> Menu,n,WaitExten(5)

exten=> 1,1,Goto(internos,despertador,1)

exten=> 2,1,Goto(internos,clima,1)

exten=> i,1,PlayBack(pbx-invalid)
exten=> i,n,Goto(MenuHotel,Menu,1)

exten=> t,1,PlayBack(vm-goodbye)
exten=> t,n,Hangup()
```

## Anexo C

En este anexo mostraremos el código del archivo Consulta-1.agi en el cual se realiza la validación del cliente si existe o no en la base de datos del hotel.

```
#! /usr/bin/php -q

<?php

ob_implicit_flush(true);
set_time_limit(6);
$in = fopen("php://stdin","r");

$stdlog = fopen("/var/log/asterisk/acadi.log", "w");

// Definicion de funciones antes del programa principal

function read()
{
global $in, $debug;
$input = str_replace("\n", "", fgets($in, 4096));
return $input;
}

function errlog($line) {
global $err;
echo "VERBOSE \"$line\"\n";
}

function write($line) {
global $debug;
echo $line."\n";
}

function connect_db()
{
$db_connection = mysql_connect ("localhost", "root", "admin") or die
(mysql_error());
$db_select = mysql_select_db("Hotel") or die (mysql_error());
}


while ($env=read())
{
$env = str_replace("\"","",$env);
$s = split(": ",$env);
$agi[str_replace("agi_","",$s[0])] = trim($s[1]);
if ($env == "")
{
break;
}
}
```

```
$cedcliente = $argv[1];
$passcliente = $argv[2];

connect_db();

$query1="SELECT id_Cliente FROM Cliente WHERE Cedula = '$cedcliente'
and Clave = '$passcliente'";
$query_result1 = @mysql_query($query1);
$row_count= mysql_num_rows($query_result1);


//$con = mysql_connect("localhost","root","admin");


if ($row_count !=0)
  {
      write ("SET CONTEXT internos");
      write ("EXEC GoTO reserva,1"); // extension para clientes
antiguos
  }
Else { // registro no existe en nuestra base, entonces es un cliente
nuevo

write ("SET CONTEXT internos");
write ("EXEC GoTO baduser,1"); //extension para clientes nuevos

}

// clean up file handlers etc.
fclose($in);
fclose($stdlog);

exit;
?>
```

## Anexo D

En este anexo mostraremos el código del archivo reserva.agi en el cual se realiza la reserva de la habitación con los datos ingresados por el cliente

```
. #! /usr/bin/php -q

<?php
ob_implicit_flush(true);
set_time_limit(6);
$in = fopen("php://stdin","r");
$stdlog = fopen("/var/log/asterisk/acadi.log", "w");

// Definicion de funciones antes del programa principal

function read()
{
      global $in, $debug;
      $input = str_replace("\n", "", fgets($in, 4096));
      return $input;
}

function errlog($line) {
      global $err;
      echo "VERBOSE \"$line\"\n";
}

function write($line) {
      global $debug;
      echo $line."\n";
}

function connect_db()
{
      $db_connection = mysql_connect ("localhost", "root", "admin")
or die (mysql_error());
      $db_select = mysql_select_db("Hotel") or die (mysql_error());
}

while ($env=read())
{
      $env = str_replace("\"","",$env);
      $s = split(": ",$env);
      $agi[str_replace("agi_","",$s[0])] = trim($s[1]);
      if ($env == "")
      {
            break;
      }
}
```

```
$dia = $argv[1];
$mes = $argv[2];
$cedcliente = $argv[3];
$diaF = $argv[4];
$mesF = $argv[5];
if(strlen($mes)==1)$mes= "0".$mes;
if(strlen($dia)==1)$dia= "0".$dia;
if(strlen($mesF)==1)$mesF= "0".$mesF;
if(strlen($diaF)==1)$diaF= "0".$diaF;
$fecha = "2011"."-".$mes."-".$dia;
$fechaFin = "2011"."-".$mesF."-".$diaF;
$tipo_Habitacion= $argv[6];
$var_fecha_actual = "20".DATE("y-m-d");


connect_db();

if($fecha<$fechaFin && $fecha>=$var_fecha_actual){

switch ($tipo_Habitacion){
      case 1:
            $tipoH = "Sencilla";
            $query_costo = "select DATEDIFF('$fechaFin','$fecha') *
100 as CostoTotal";
            break;
      case 2:
            $tipoH = "Doble";
            $query_costo = "select DATEDIFF('$fechaFin','$fecha') *
200 as CostoTotal";
            break;
      case 3:
            $tipoH = "Triple";
            $query_costo = "select DATEDIFF('$fechaFin','$fecha') *
300 as CostoTotal";
            break;
      }

//Seleccionamos el id_cliente para el cual se va a realizar la
reserva
$query1="SELECT id_Cliente FROM Cliente WHERE Cedula =
'$cedcliente'";

//Verificamos que existan habitaciones disponibles,segun el tipo de
habitacion y fecha
$query2="select id_Habitacion  from Habitacion  where id_Habitacion
NOT IN (select          H.id_Habitacion from DetalleReserva AS
D,Habitacion AS H where ((D.F_inicio BETWEEN '$fecha' AND
'$fechaFin') OR (D.F_Fin BETWEEN '$fecha' AND '$fechaFin') OR (
'$fecha' BETWEEN D.F_inicio AND D.F_Fin) OR ('$fechaFin' BETWEEN
D.F_inicio AND D.F_Fin)) AND (D.id_Habitacion=H.id_Habitacion)) AND
Descripcion='$tipoH'";
$query_result1 = @mysql_query($query1);
$query_result2 = @mysql_query($query2);
$row_count= mysql_num_rows($query_result2);
```

```
$rows = @mysql_fetch_array($query_result2);
$rows2 = @mysql_fetch_array($query_result1);

//query costo total Sencilla**********************
$query_result_costo = @mysql_query($query_costo);
$price = @mysql_fetch_array($query_result_costo);
//fin*****************************************

$res_cliente = $rows2[0]; //obtenemos el id del cliente
$res_habitacion = $rows[0]; //obtenemos el id de la habitacion
disponible
$valor_total_habitacion = $price[0]; //obtenemos el costo total de
la reserva

if ($row_count !=0)  {        //verificamos si existen habitaciones
disponibles

      $query_reserva = "INSERT INTO
DetalleReserva(id_Cliente,id_Habitacion,F_inicio,F_Fin,CostoTotal)
values
('$res_cliente','$res_habitacion','$fecha','$fechaFin','$valor_total
_habitacion')";
      $query_result3 = @mysql_query($query_reserva);
}
Else {
      write ("SET CONTEXT internos");
      write ("EXEC GoTO nohab,1"); //Extension cuando no hay
habitaciones disponibles
      }
}//Fin del If para validar las fechas
Else {
      write ("SET CONTEXT internos");
      write ("EXEC GoTO fechaerror,1"); //extension cuando la fecha
final es menor q la inicial
}

fclose($in);
fclose($out);
fclose($stdlog);

exit;
?>
```

## Anexo E

En este anexo mostraremos el código del archivo wer.agi en el cual se realiza la consulta del clima de la ciudad de Guayaquil.

```
#! /usr/bin/php -q
<?php
      $xml =
@file_get_contents("http://weather.yahooapis.com/forecastrss?w=37573
3&u=c");
      if (!defined('STDIN'))
          {
            define('STDIN', fopen('php://stdin', 'r'));
          }
          if (!defined('STDOUT'))
          {
             define('STDOUT', fopen('php://stdout', 'w'));
          }
          if (!defined('STDERR'))
          {
             define('STDERR', fopen('php://stderr', 'w'));
          }

     # retrieve all AGI variables from Asterisk
    while (!feof(STDIN))
    {
        $temp = trim(fgets(STDIN,4096));
        if (($temp == "") || ($temp == "\n"))
        {
            break;
        }
        $s = split(":",$temp);
        $name = str_replace("agi_","",$s[0]);
        $agi[$name] = trim($s[1]);
    }

      if($xml){
            $dom = new DOMDocument;
            $dom->preserveWhiteSpace = false;
            $dom->LoadXML($xml);
            $domCondition = $dom->getElementsByTagName("condition");

            foreach ($domCondition as $domConditionChild)
            {
                  if($domConditionChild->hasChildNodes()){
                        $children = $domConditionChild->childNodes;
                        foreach ($children as $child)
                        {
                              $tmp_dom = new DOMDocument();
```

```
                                    $tmp_dom->appendChild($tmp_dom-
>importNode($child, true));
                                    $innerHTML.=trim($tmp_dom->saveHTML());
                                    echo($innerHTML);
                              }
                        }else{
                              echo(" ");
                        }
                        echo("</br>");
                        if($domConditionChild->hasAttributes()) {
                              $attributes = $domConditionChild->attributes;
                              if(!is_null($attributes))  {
                                    foreach ($attributes as $attr){

                                          if($attr->name=="temp"){

                                          echo($attr->value);

                                                fwrite(STDOUT,"STREAM FILE
temperature \"\"\n");
                                            fflush(STDOUT);

                                            $result =
trim(fgets(STDIN,4096));

                                          fwrite(STDOUT,"STREAM FILE
temperature \"\"\n");
                                            fflush(STDOUT);

                                            $result =
trim(fgets(STDIN,4096));

                                          fwrite(STDOUT,"STREAM FILE
currently \"\"\n");
                                            fflush(STDOUT);

                                            $result =
trim(fgets(STDIN,4096));

                                            fwrite(STDOUT,"say number
$attr->value \"\"\n");
                                            fflush(STDOUT);
                                            $result =
trim(fgets(STDIN,4096));

                                            fwrite(STDOUT,"STREAM FILE
degrees \"\"\n");
                                            fflush(STDOUT);
                                            $result =
trim(fgets(STDIN,4096));

                                            fwrite(STDOUT,"STREAM FILE
celsius \"\"\n");
                                            fflush(STDOUT);
```

```
                                        $result =
trim(fgets(STDIN,4096));

                                    }

                                }
                            }
                        }

                }
                $domCondition = $dom->getElementsByTagName("forecast");
                $bandera=true;
                foreach ($domCondition as $domConditionChild)
                {
                      if($domConditionChild->hasChildNodes()){
                            $children = $domConditionChild->childNodes;
                            foreach ($children as $child)
                            {
                                  $tmp_dom = new DOMDocument();
                                  $tmp_dom->appendChild($tmp_dom-
>importNode($child, true));
                                  $innerHTML.=trim($tmp_dom->saveHTML());
                                  echo($innerHTML);
                            }
                      }else{
                            echo(" ");
                      }

                      echo("</br>");
                      if($domConditionChild->hasAttributes()) {
                            $attributes = $domConditionChild->attributes;
                            if(!is_null($attributes))  {
                                  foreach ($attributes as $attr) {
                                        if($bandera){
                                              if($attr->name=="low"){
                                                    echo($attr->value);
                                        fwrite(STDOUT,"STREAM FILE
temperature \"\"\n");
                                          fflush(STDOUT);

                                          $result =
trim(fgets(STDIN,4096));

                                        fwrite(STDOUT,"STREAM FILE
temperature \"\"\n");
                                          fflush(STDOUT);

                                          $result =
trim(fgets(STDIN,4096));

                                        fwrite(STDOUT,"STREAM FILE
minimum \"\"\n");
                                          fflush(STDOUT);
```

```
trim(fgets(STDIN,4096));


$attr->value \"\"\n");

trim(fgets(STDIN,4096));

degrees \"\"\n");

trim(fgets(STDIN,4096));




temperature \"\"\n");


trim(fgets(STDIN,4096));

temperature \"\"\n");


trim(fgets(STDIN,4096));

maximum \"\"\n");


trim(fgets(STDIN,4096));

$attr->value \"\"\n");

trim(fgets(STDIN,4096));

degrees \"\"\n");

trim(fgets(STDIN,4096));

goodbye \"\"\n");
```

```
  $result =


  fwrite(STDOUT,"say number

  fflush(STDOUT);
  $result =

  fwrite(STDOUT,"STREAM FILE

  fflush(STDOUT);
  $result =

            }
      echo(" ");
      if($attr->name=="high"){
           echo($attr->value);
fwrite(STDOUT,"STREAM FILE

  fflush(STDOUT);

  $result =


fwrite(STDOUT,"STREAM FILE

  fflush(STDOUT);

  $result =


fwrite(STDOUT,"STREAM FILE

  fflush(STDOUT);

  $result =


  fwrite(STDOUT,"say number

  fflush(STDOUT);
  $result =


  fwrite(STDOUT,"STREAM FILE

  fflush(STDOUT);
  $result =


fwrite(STDOUT,"STREAM FILE vm-
```

```
                                        fflush(STDOUT);
                                        $result =
trim(fgets(STDIN,4096));
                                            }

                                        }

                                }
                                $bandera=false;
                        }
                }

            }
      }
      else{
            echo("no se encontro");
      }
?>
```

## Anexo F

En este anexo mostraremos el código del archivo wakeup.agi en el cual se realiza la programación de despertadores.

```
#!/usr/bin/php -q
<?php
{

      // Si la aplicacion tiene problemas puede buscarlos en este
archivo
      $parm_error_log = '/tmp/wakeup.log';

      // Coloca 1 para tener archivo de log
      $parm_debug_on = 1;

      // Aqui es donde las llamadas de despertador seran creadas
      $parm_temp_dir = '/tmp';

      $parm_call_dir = '/var/spool/asterisk/outgoing';

      // Numero maximo de intento de llamadas
      $parm_maxretries = 3;

      // Tiempo que el telefono va a estar sonando
      $parm_waittime = 60;

      // Numero en segundos entre intentos
      $parm_retrytime = 60;

      // Caller ID
      $parm_wakeupcallerid = '"WakeUp" <77>';

      $parm_chan_ext = 0;

      $parm_short_entry = 1;

      // Aplicacion que se ejecutara cuando el usuario conteste el
despertador
      $parm_application = 'MusicOnHold';
      $parm_data = '';

      // -- Use this for the ANNOY application
      //$parm_application = 'AGI';
      //$parm_data = 'wakeconfirm.agi';
      // ---------------------------------------------------

      // 1 hora militar, 0 am/pm
      $parm_prompt_ampm = 1;
```

```
	$parm_operator_mode = 1;

	$parm_operator_extensions= array(  0, 15  );

	$parm_operator_ext_len = 4;

	//-------------------
	// Fin de los parametros de configuracion
	//-------------------

	GLOBAL	$stdin, $stdout, $stdlog, $result, $parm_debug_on,
$parm_test_mode;

	// Estos parametros se detallan en la pagina http://www.voip-
info.org
	ob_implicit_flush(false);
	set_time_limit(30);
	error_reporting(0);

	$stdin = fopen( 'php://stdin', 'r' );
	$stdout = fopen( 'php://stdout', 'w' );

	if ($parm_debug_on)
	{
		$stdlog = fopen( $parm_error_log, 'w' );
		fputs( $stdlog, "---Start---\n" );
	}


	// Obteniendo las variables de Asterisk se puede saber mas de
ellas en:
	// http://www.voip-info.org/tiki-
index.php?page=Asterisk%20AGI%20php
	while ( !feof($stdin) )
	{
		$temp = fgets( $stdin );

		if ($parm_debug_on)
			fputs( $stdlog, $temp );

		$temp = str_replace( "\n", "", $temp );

		$s = explode( ":", $temp );
		$agivar[$s[0]] = trim( $s[1] );
		if ( ( $temp == "") || ($temp == "\n") )
		{
			break;
		}
	}


	// Obtenemos el canal y el numero de extension del telefono
	$channel = $agivar['agi_channel'];
```

```
	if ( $parm_debug_on )
		fputs( $stdlog, "Channel: $channel\n" );

	if (preg_match('~^(.*)/(.*)-([0-9a-zA-Z].*)$~', $channel,
$match) )
	{
		$sta = trim($match[2]);
		$chan = trim($match[1]);

		if ( $parm_debug_on )
			fputs( $stdlog, "Channel SPLIT-STA: $sta - CHAN:
$chan\n" );

	}


	// Obtenemos el callerid
	$callerid = $agivar['agi_callerid'];

	// Obetenemos el numero de extension
	if (preg_match('/<([ 0-9]+)>/', $callerid, $match) )
	{
		$cidn = trim($match[1]);
	}
	else
	{
		if (preg_match('/([0-9]+)/', $callerid, $match) )
		{
			$cidn = trim($match[1]);
		}
		else
			$cidn = -1;
	}


	//=============================================================
	// En esta parte interactuamos con el cliente
	//=============================================================


	$rc = execute_agi( "ANSWER ");

	sleep(1);

	if ( !$rc['result'] )
		$rc = execute_agi( "STREAM FILE welcome \"\" ");


	// Revisamos si el cliente ya tiene programado un despertador
	if ( $parm_chan_ext )
		$dir_check = "$chan.$sta.call";
	else
		$dir_check = "ext.$cidn.call";
```

```
      if ( $parm_operator_mode )
      {
            if ( $parm_chan_ext )
            {
                  if ( $parm_debug_on )
                        fputs( $stdlog, 'ERROR: Modo operador solo
trabaja cuando $parm_chan_ext es 0'.
                                          "Esto es porque nosotros no
conocemos si la extension es SIP, ZAP, IAX....\n");

                  $rc = execute_agi( "STREAM FILE something-terribly-
wrong \"\" ");
                  if ( !$rc['result'] )
                        $rc = execute_agi( "STREAM FILE goodbye \"\"
");
                  if ( !$rc['result'] )
                        $rc = execute_agi( "HANGUP");
                  exit;
            }
            else
            {

                  $ext_ok = 0;
                  foreach( $parm_operator_extensions AS $oe )
                        if ( $oe == $cidn )
                              $ext_ok = 1;

                  $pound = 0;
                  $loop = 0;

                  if ( $ext_ok )
                  {
                        while ( !$rc['result'] && !$pound )
                        {

                              $rc = execute_agi( "STREAM FILE please-
enter-the \"0123456789\" ");

                              if ( !$rc['result'] )
                                    $rc = execute_agi( "STREAM FILE
extension \"0123456789\" ");

                              if ( !$rc['result'] )
                                    $rc = execute_agi( "STREAM FILE
for \"0123456789\" ");

                              if ( !$rc['result'] )
                                    $rc = execute_agi( "STREAM FILE
your \"0123456789\" ");
                              // si llegamos aqui aun no han
presionado nada
                              if ( !$rc['result'] )
                              {
```

```
                                    $rc = execute_agi( "GET DATA
wakeup-call 15000 $parm_operator_ext_len");

                                    if ( isset($rc['timeout']) )
                                          $rc['result'] = '';
                                    else if ( $rc['result'] != -1 )
                                    {
                                          $pound = 1;
                                          $num = $rc['result'];
                                    }
                                    else if ( strlen( $rc['result'] )
!= $parm_operator_ext_len )
                                          $rc['result'] = '';
                              }

                              if ( $rc['result'] != -1 )
                              {


                                    if ($parm_debug_on)
                                          fputs( $stdlog, "We have
digits: len=" . strlen( $rc['result']) . " val=$rc[result] \r\n\r\n"
);


                                    if ( strlen( $rc['result'] ) == 2
&& !$pound )
                                    {
                                          $num= $rc['result']-48;
                                          while ( strlen($num) <
$parm_operator_ext_len && $rc['result'] > 0 && !$pound)
                                          {
                                                $rc = execute_agi(
"WAIT FOR DIGIT 15000");
                                                if ($parm_debug_on)
                                                      fputs( $stdlog,
"We have a digit:" . strlen( $rc['result']) . "- $rc[result] \r\n"
);


                                                if ( $rc['result'] >=
48 && $rc['result'] <= 57 )
                                                      $num .=
$rc['result'] - 48;
                                                else if (
$rc['result'] == '35' )
                                                      $pound = 1;
                                                else
                                                      $rc['result'] =
0;

                                          }
                                          $rc['result'] = $num;
                                    }
```

```
                                    if ($parm_debug_on)
                                          fputs( $stdlog, "Checking
Results [$rc[result]] \r\n\r\n" );


                                    if ( strlen( $num ) <= 0 )
                                    {
                                          $pound = 0;
                                          $loop++;

                                          if ( $loop >= 3 )
                                          {
                                                $rc = execute_agi(
"STREAM FILE goodbye \"\" ");
                                                if ( !$rc['result'] )
                                                      $rc =
execute_agi( "HANGUP");
                                                exit;

                                          }
                                    }
                                    else
                                    {
                                          $rc = execute_agi( "STREAM
FILE you-entered \"\" ");

                                          if ( !$rc['result'] )
                                                $rc = execute_agi(
"SAY DIGITS $num \"\" ");

                                          $cidn = $num;
                                          $dir_check =
"ext.$num.call";
                                          $pound = 1;
                                    }

                              }
                        }
                  }
            }
      }

      if ($parm_debug_on)
            fputs( $stdlog, "Checking Directory [$parm_call_dir]
Check=[$dir_check]\n" );

      $outc=0;
      $dir_handle = opendir( $parm_call_dir );
      while( $file = readdir($dir_handle ) )
      {
            if ($parm_debug_on)
                  fputs( $stdlog, "File=$file\n" );
```

```
            if (strstr( $file, $dir_check ) )
                  $out[$outc++] = $file;

      }
      closedir( $dir_handle );



      if ( $outc )
      {
            $i = 0;
            while ( $out[$i] )
            {

                  $wtime = strtok( $out[$i], '.' );

                  if ($parm_debug_on)
                        fputs( $stdlog, "wakeup found=$out[0] saying
time $wtime\n" );


                  say_wakeup( $wtime );
                  $i++;

            }

      }


      if ( $outc )
      {
            // Si el usuario quiere o no cancelar una llamada de
despertador
            while ( !$rc['result'] )
            {
                  if ( !$rc['result'] )
                        $rc = execute_agi( "STREAM FILE for-wakeup-
call \"12\" ");
                  if ( !$rc['result'] )
                        $rc = execute_agi( "STREAM FILE press-1
\"12\" ");
                  if ( !$rc['result'] )
                        $rc = execute_agi( "STREAM FILE to-cancel-
wakeup \"12\" ");
                  if ( !$rc['result'] )
                        $rc = execute_agi( "STREAM FILE press-2
\"12\" ");
                  if ( !$rc['result'] )
                  {
                        $rc = execute_agi( "WAIT FOR DIGIT 15000");
                  }
                  if ( $rc['result'] != -1 )
                  {
```

```
                        if ( $rc['result'] == 49 || $rc['result'] ==
50 )
                        {
                              ;
                        }
                        else
                        {
                              // Si presiona otro numero que no sea 1
o 2
                              $rc['result'] = 0;
                              $rc = execute_agi( "STREAM FILE you-
dialed-wrong-number \"\" ");
                              $rc = execute_agi( "STREAM FILE wrong-
try-again-smarty \"\" ");
                        }

                  }
            }
      }
      else  // Crear el despertador
            $rc['result'] = '49';

      // Procesar digito ingresado

      switch( $rc['result'] )
      {
            case '49':  // Presiono 1  - para crear un nuevo
despertador
            {
                  $rc['result'] = 0;
                  while ( !$rc['result'] )
                  {
                        $rc = execute_agi( "STREAM FILE please-enter-
the \"0123456789\" ");

                        if ( !$rc['result'] )
                              $rc = execute_agi( "STREAM FILE time
\"0123456789\" ");
                        if ( !$rc['result'] )
                              $rc = execute_agi( "STREAM FILE for
\"0123456789\" ");
                        if ( !$rc['result'] )
                              $rc = execute_agi( "STREAM FILE your
\"0123456789\" ");

                        if ( !$rc['result'] )
                        {
                              $rc = execute_agi( "GET DATA wakeup-
call 15000 4");

                              if ( isset($rc['timeout']) )
                                    $rc['result'] = '';
                              else if ( $rc['result'] != -1  &&
$parm_short_entry )
```

```
                                    $rc['result'] = str_pad(
$rc['result'], 4, '0', STR_PAD_LEFT );
                              else if ( strlen( $rc['result'] ) != 4
)
                                    $rc['result'] = '';
                        }

                        if ( $rc['result'] != -1 )
                        {


                              if ($parm_debug_on)
                                    fputs( $stdlog, "We have digits:
len=" . strlen( $rc['result']) . " val=$rc[result] \r\n\r\n" );


                              if ( strlen( $rc['result'] ) == 2 )
                              {
                                    $num= $rc['result']-48;
                                    while ( strlen($num) < 4 &&
$rc['result'] > 0 )
                                    {
                                          $rc = execute_agi( "WAIT
FOR DIGIT 15000");
                                          if ($parm_debug_on)
                                                fputs( $stdlog, "We
have a digit:" . strlen( $rc['result']) . "- $rc[result] \r\n" );


                                          if ( $rc['result'] >= 48 &&
$rc['result'] <= 57 )
                                                $num .= $rc['result']
- 48;
                                          else if ( $rc['result'] ==
'35' && $parm_short_entry )
                                                $num = str_pad($num,
4, '0', STR_PAD_LEFT);
                                          else
                                                $rc['result'] = 0;

                                    }
                                    if (strlen($num) == 4 )
                                          $rc['result'] = $num;
                              }

                              if ($parm_debug_on)
                                    fputs( $stdlog, "Checking Results
[$rc[result]] \r\n\r\n" );


                              $overTime =  ( $parm_prompt_ampm == 2 )
? 1259 : 2359;
```

```
                              if ( $rc['result'] > $overTime ||
strlen( $rc['result']) < 4 || substr($rc['result'],2,2) > 59 ||
$rc['result'] < 0)
                              {
                                    $rc['result'] = 0;
                                    $rc = execute_agi( "STREAM FILE
please-try-again \"\" ");
                              }

                              if (strlen( $rc['result'] ) == 4 &&
$rc['result'] == 0 )
                                    $rc['result'] = -2;

                        }
                  }

                  if ( $rc['result'] == -2 )
                        $rc['result'] = '0000';
                  else if ( $rc['result'] == -1 )
                        exit;


                  // Almacena la hora ingresada
                  $wtime = $rc['result'];


                  // We don't know who the user is, so if its less
than 1300 it could be AM or PM, so prompt
                  // them for am pm
                  if ( $wtime != -1 && $wtime < 1300 &&
$parm_prompt_ampm != 1)
                  {
                        $rc['result'] = 0;
                        while ( !$rc['result'] )
                        {

                              if ( !$rc[result] )
                                    $rc = execute_agi( "GET DATA 1-
for-am-2-for-pm 15000 1");

                        }

                        switch( $rc['result'] )
                        {
                              case '1':               // Set to AM
should be under 1159 or less
                                    if ( $wtime > 1159 )
                                          $wtime -= 1200;
                                    $rc['result'] = 0;
                                    break;

                              case '2':               // Set to PM
should be equal or over 1200
                                    if ( $wtime < 1159 )
```

```
                                        $wtime += 1200;
                                    $rc['result'] = 0;
                                    break;
                        }
                  }


                  // At this point we have a millitary time in the
$wtime variable
                  if ( $parm_chan_ext )
                  {
                        $wakefile =
"$parm_temp_dir/$wtime.$chan.$sta.call";
                        $callfile =
"$parm_call_dir/$wtime.$chan.$sta.call";
                  }
                  else
                  {
                        $wakefile =
"$parm_temp_dir/$wtime.ext.$cidn.call";
                        $callfile =
"$parm_call_dir/$wtime.ext.$cidn.call";
                  }

                  if ($parm_debug_on)
                        fputs( $stdlog, "Wakeup File [$wakefile]\n"
);

                  // Open up a wakeup file to write it out.
                  $wuc = fopen( $wakefile, 'w');

                  if ( $wuc )
                  {
                        // Delete any old Wakeup call files this one
will override
                        for ($i=0; $i < $outc; $i++ )
                        {
                              if( file_exists(
"$parm_call_dir/$out[$i]" ) )
                                    {
                                          if ($parm_debug_on)
                                                fputs( $stdlog, "Unlinking
Old File [$parm_call_dir/$out[$i]]\n" );

                                          unlink( "$parm_call_dir/$out[$i]"
);
                                    }
                        }

                        // I've noticed that the other WAKEUP example
has a different format.  This worked for me
                        // Here is where we either make the call to
the Extension or the Channel.  Extension
```

```
                        // is the better way to go, but required the
caller ID information.  Where Channel
                        // should always get you back to where you
were called from, provided its on your system
                        if ( $parm_chan_ext )
                              fputs( $wuc, "channel: $chan/$sta\n" );
                        else
                              fputs( $wuc, "channel:
Local/$cidn@$agivar[agi_context]\n" );

                        fputs( $wuc, "maxretries:
$parm_maxretries\n");
                        fputs( $wuc, "retrytime: $parm_retrytime\n");
                        fputs( $wuc, "waittime: $parm_waittime\n");
                        fputs( $wuc, "callerid:
$parm_wakeupcallerid\n");

                        fputs( $wuc, "application:
$parm_application\n");
                        fputs( $wuc, "data: $parm_data\n");

                        fclose( $wuc );

                        $w = getdate();

                        $w['hours']   = substr( $wtime, 0, 2 );
                        $w['minutes'] = substr( $wtime, 2, 2 );

                        $time_wakeup = mktime( substr( $wtime, 0, 2
), substr( $wtime, 2, 2 ), 0, $w['mon'], $w['mday'], $w['year'] );

                        $time_now = time( );

                        if ($parm_debug_on)
                              fputs( $stdlog, 'time_wakeup=' .
date('l dS \of F Y h:i:s A', $time_wakeup) . " ($time_wakeup) |
time_now=" . date('l dS \of F Y h:i:s A',$time_now) . "
($time_now)\n" );

                        if ( $time_wakeup <= $time_now )
                              $time_wakeup += 86400; // Add One Day
on

                        if ($parm_debug_on)
                              fputs( $stdlog, 'Setting WAKEUP file to
=' . date('l dS \of F Y h:i:s A', $time_wakeup) . " -
$time_wakeup\n" );


                        touch( $wakefile, $time_wakeup, $time_wakeup
);

                        rename( $wakefile, $callfile );
```

```
                }
                else
                {
                        // Couldn't open the file.  Make sure you
created the /var/lib/asterisk/wakeups directory
                        if ($parm_debug_on)
                              fputs( $stdlog, "Error opening file
[$wakefile]\n" );

                        $rc = execute_agi( "STREAM FILE something-
terribly-wrong \"\" ");
                        if ( !$rc['result'] )
                              $rc = execute_agi( "STREAM FILE goodbye
\"\" ");
                        if ( !$rc['result'] )
                              $rc = execute_agi( "HANGUP");
                        exit;
                }

                //  If we have a caller ID  and waking up by
extension say the extension number
                if ( $cidn && $parm_chan_ext == 0 )
                {
                        $rc = execute_agi( "STREAM FILE for \"\" ");

                        if ( !$rc['result'] )
                              $rc = execute_agi( "STREAM FILE
extension \"\" ");


                        if ( !$rc['result'] )
                              $rc = execute_agi( "SAY DIGITS $cidn
\"\" ");

                }

                say_wakeup( $wtime );
                $rc['result'] = 0;

            }
                  break;

                  // This is the END Of a new wakeup call


            case '50':  // Pressed 2 - Delete old wakeup calls
            {
                  // Go through the list of old files and
unlink/delete them
                  for ($i=0; $i < $outc; $i++ )
                  {
                        if( file_exists( "$parm_call_dir/$out[$i]" )
)
                        {
```

```
                              if ($parm_debug_on)
                                    fputs( $stdlog, "Unlinking Wakeup
File [$parm_call_dir/$out[$i]]\n" );

                              unlink( "$parm_call_dir/$out[$i]" );
                        }
                  }

                  // If Caller ID and recording by Extension then say
the extension
                  if ( $cidn && $parm_chan_ext == 0 )
                  {
                        $rc = execute_agi( "STREAM FILE for \"\" ");

                        if ( !$rc['result'] )
                              $rc = execute_agi( "STREAM FILE
extension \"\" ");


                        $L = strlen( $cidn );

                        for( $i = 0; $i < $L && !$rc['result']; $i++
)
                        {
                              $cid_digits = substr( $cidn, $i, 1 );

                              if ( !$rc['result'] )
                                    $rc = execute_agi( "SAY NUMBER
$cid_digits \"\" ");
                        }
                  }

                  $rc = execute_agi( "STREAM FILE wakeup-call-
cancelled \"\" ");


            }
                  break;

      }

      if ( !$rc['result'] )
            $rc = execute_agi( "STREAM FILE goodbye \"\" ");
      if ( !$rc['result'] )
            $rc = execute_agi( "HANGUP");
      if ($parm_debug_on)
            fclose($stdlog);
      exit;
}


// ---------------------------------------------
// This will say military time in AM/PM format
// ---------------------------------------------
```

```
function say_wakeup( $wtime )
{
      GLOBAL      $stdin, $stdout, $stdlog, $parm_debug_on;

      $pm = 0;

      if ($wtime > 1159 )
      {
            $wtime -=1200;
            $pm = 1;
      }

      if ($wtime <= 59 )
            $wtime += 1200;

      if ( strlen( $wtime ) == 3 )
            $wtime = '0' . $wtime;


      $h = substr( $wtime, 0, 2 );
      $h1 = substr( $wtime, 0, 1 );
      $h2 = substr( $wtime, 1, 1 );
      $m = substr( $wtime, 2, 2 );
      $m1 = substr( $wtime, 2, 1);
      $m2 = substr( $wtime, 3, 1);


      if ($parm_debug_on)
            fputs( $stdlog, "Wakeup time is set to $wtime\n" );

      $rc = execute_agi( "STREAM FILE rqsted-wakeup-for \"\" ");

      if ( !$rc['result'] )
      {
            if ( $h1 == 0 )
                  $rc = execute_agi( "SAY NUMBER $h2 \"\"");
            else
                  $rc = execute_agi( "SAY NUMBER $h \"\"");

            if ( !$rc['result'] )
            {
                  if ($m == 0 )
                        $rc = execute_agi( "STREAM FILE digits/oclock
\"\" ");
                  else
                  {
                        if ( $m1 == 0 )
                        {
                              $rc = execute_agi( "STREAM FILE
digits/0 \"\" ");
                              $rc = execute_agi( "SAY NUMBER $m2 \"\"
");
                        }
                        else
```

```
                              $rc = execute_agi( "SAY NUMBER $m
\"\"");
                  }
                  if ( !$rc['result'] )
                  {
                        if ( $pm )
                              $rc = execute_agi( "STREAM FILE
digits/p-m \"\" ");
                        else
                              $rc = execute_agi( "STREAM FILE
digits/a-m \"\" ");
                  }
            }
      }
}


//-------------------------------------------------
// This function will send out the command and get
//    the response back
//-------------------------------------------------
function execute_agi( $command )
{
      GLOBAL      $stdin, $stdout, $stdlog, $parm_debug_on;

      fputs( $stdout, $command . "\n" );
      fflush( $stdout );
      if ($parm_debug_on)
            fputs( $stdlog, $command . "\n" );

      $resp = fgets( $stdin, 4096 );

      if ($parm_debug_on)
            fputs( $stdlog, "      READ=$resp" );

      if ( preg_match("/^([0-9]{1,3}) (.*)/", $resp, $matches) )
      {
            if (preg_match('/result=([-0-9a-zA-Z]*)(.*)/',
$matches[2], $match))
            {
                  $arr['code'] = $matches[1];
                  $arr['result'] = $match[1];
                  if (isset($match[3]) && $match[3])
                        $arr['data'] = $match[3];

                  if ( trim( $match[2] ) == '(timeout)' )
                  {
                        //$arr['result'] = -1;
                        $arr['timeout'] = TRUE;
                  }

                  if ($parm_debug_on)
                        fputs( $stdlog, "1=$match[1], 2=$match[2],
3=$match[3], 4=$match[4]\n" );
```

```
                  return $arr;
            }
            else
            {
                  if ($parm_debug_on)
                        fputs( $stdlog, "Couldn't figure out returned
string, Returning code=$matches[1] result=0\n" );
                  $arr['code'] = $matches[1];
                  $arr['result'] = 0;
                  return $arr;
            }
      }
      else
      {
            if ($parm_debug_on)
                  fputs( $stdlog, "Could not process string,
Returning -1\n" );
            $arr['code'] = -1;
            $arr['result'] = -1;
            return $arr;
      }
}
?>
```

## Anexo G

En este anexo mostraremos el código del archivo registro.php en el cual el cliente ingresa sus datos.

```
<html>
    <head>
        <title> Ingresar Cliente </title>
        <script src="js/jquery.js"></script>
        <script src="js/validacion.js"></script>
      <script src="js/calendar_us.js"></script>
        <!--<script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.4/jquery.min.js"
type="text/javascript"></script>-->
        <script src="style1/js/jquery.easing.1.3.js"
type="text/javascript"></script>
        <link rel="stylesheet" href="style1/css/style.css"
type="text/css" media="screen"/>
      <link rel="stylesheet" href="style1/css/calendar.css"
type="text/css" media="screen"/>
    </head>
    <body>
        <h1 class="title" align="center">Ingresar</h1>
        <h2 align="center">INGRESE CLIENTE</h2>
        <div class="formulario" align="center">
            <form action="#" id="Formulario" name="Formulario">
                <table>
                    <tbody>
                        <tr><td>*</td><td><b>Cedula:</b></td>
                        <td><input type="text" maxlength="10"
name="cedula" id="cedula"/></td></tr>
                        <tr><td>*</td><td><b>Nombres:</b></td>
                        <td><input type="text" maxlength="50"
name="nombre" id="nombre"/></td></tr>
                        <tr><td>*</td><td><b>Apellidos:</b></td>
                        <td><input type="text" maxlength="50"
name="apellido" id="apellido"/></td></tr>
                        <tr><td>*</td><td><b>Direccion:</b></td>
                        <td><input type="text" maxlength="50"
name="direccion" id="direccion"/></td></tr>
                        <tr><td>*</td><td><b>F. Nacimiento:</b></td>
                        <td><input type="date" maxlength="10"
name="fecha" readonly="readonly" id="fecha"/>
      <script language="JavaScript">
      new tcal ({
            // form name
            'formname': 'Formulario',
            // input name
            'controlname': 'fecha'
      });
```

```
      </script></td></tr>
                        <tr><td>*</td><td><b>Telefono:(7
Digitos)</b></td>
                        <td><input type="text" maxlength="7"
name="telefono" id="telefono"/></td></tr>
                        <tr><td>*</td><td><b>ContraseÃ±a:(4
Digitos)</b></td>
                        <td><input type="password" maxlength="4"
name="password" id="password" /></td></tr>
                        <tr><td></td><td align="center"
style="color:#6B8E23; font-weight: bold;">Campos requeridos
(*)</td></tr>
                        <tr><td></td><td></td><td
align="center"><input type="button" class="boton" name="btnEnviar"
value="GUARDAR"  onCLick="validar();"/></td></tr>
                    </tbody>
                </table>
            </form>
        </div>
       </body>
</html>
```