ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"SISTEMA TELEFÓNICO AUTOMÁTICO PARA CONSULTAS DE DEUDAS Y FECHAS DE PAGO"

INFORME DE MATERIA DE GRADUACIÓN

Previa a la obtención del Título de:

INGENIERO EN TELEMÁTICA

Presentada por:

ANDREA SOLANGE FREIRE MORÁN

EDUARDO ARTURO LÓPEZ YAGUANA

GUAYAQUIL – ECUADOR

2010

AGRADECIMIENTO

A Dios que me ha dado inteligencia, me ha guiado y cuidado hasta el día de hoy, a mi familia por darme el soporte necesario para avanzar, a mis profesores y a mis amigos que me han ayudado a crecer en todos los aspectos.

A Dios Todopoderoso por fortalecerme y guiarme a lo largo de mi vida y mi carrera, a mi familia y seres queridos, en especial mi padre que gracias a su esfuerzo he llegado hasta aquí. También agradezco a mis amigos y profesores que han contribuido con mi desarrollo humano y profesional.

DEDICATORIA

A Dios, a mis padres, a mis hermanas y a todas las personas que alguna vez me ayudaron para seguir avanzando para alcanzar mis objetivos.

Andrea Freire Morán

Dedico este trabajo a Dios, a mi madre, a mi hermana, seres queridos y especialmente a la memoria de mi padre.

Eduardo López Yaguana

DECLARACIÓN EXPRESA

"La responsabilidad por los hechos, ideas y doctrinas expuestas en este trabajo, nos corresponden exclusivamente; y, el patrimonio intelectual de la misma, a la Escuela Superior Politécnica del Litoral"

(Reglamento de exámenes y títulos profesionales de la ESPOL)

_____

Andrea Solange Freire Morán

_____

Eduardo Arturo López Yaguana

TRIBUNAL DE SUSTENTACIÓN

_____

Ing. Gabriel Astudillo Brocel

PROFESOR DE LA MATERIA DE GRADUACIÓN

_____

Ing. Patricia Chávez Burbano

PROFESOR DELEGADO POR EL DECANO DE LA FACULTAD

RESUMEN

El proyecto expuesto en esta tesis es una de las muchas soluciones para darle a los usuarios de una institución comercial o educativa información de sus movimientos en este caso de sus deudas de una forma ágil, rápida y sencilla con una llamada telefónica y usando un campo clave para identificarse en este caso el numero de cédula.

Cuando un usuario realice una llamada a la central telefónica este escuchará un menú y tendrá que ingresar su identificador; de acuerdo a lo que el presione se pueden dar varios casos, los cuales se explican en la descripción del proyecto.

Esta es una solución a muy bajo costo para las instituciones puesto que todo lo que se utiliza para su implementación es software libre.

Se utiliza para la implementación de esta solución como servidor de Voz sobre IP Asterisk, para hacer el código que permite usar la opción Interfaz de Enlace de Asterisk (AGI) que tiene Asterisk se utiliza el lenguaje php, se usa una librería de php “php-agi” que maneja todas las opciones de Asterisk para que puedan ser usadas más fácilmente en el código php y para almacenar los datos se usa el motor para la base de datos Mysql.

INDICE

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCIÓN

Las tecnologías avanzan a una velocidad muy acelerada y cambian constantemente, podemos ver el cambio que están teniendo actualmente los sistemas analógicos a sistemas digitales, tomando como tema particular las centrales telefónicas y una de las tecnologías emergentes Voz sobre IP sobre la cual se pueden adaptar todos los servicios de las centrales telefónicas analógicas.

Por ejemplo, una institución comercial o de educación la cual requiere una forma más eficaz para que sus usuarios consulten su información en la empresa en este caso sus deudas; este proyecto consiste en la implementación de un sistema de consultas de deudas y fechas de pago por teléfono.

Tomando en cuenta la arquitectura de red de la empresa podemos adecuar un servicio de Voz sobre IP de varias formas: una Arquitectura Distribuida, Centralizada o Mixta; para la implementación de esta solución usaremos la arquitectura centralizada que consiste en un servidor que tiene toda la información de la red de Voz sobre IP de la institución.

Se asume que en la empresa hay instalado un servidor de voz sobre IP asterisk, sino lo hay, se lo pondrá en funcionamiento y para implementar la solución se utilizara un código en php usando la librería php-agi y base de datos mysql que son de código abierto.

CAPITULO I

ANTECEDENTES Y JUSTIFICACION

1.1. ANTECEDENTES

Las comunicaciones han sido una parte importante para el desarrollo y crecimiento de las instituciones. Dentro de una institución todo lo que represente rapidez y bajos costos es fácilmente adaptado para su uso dentro de la misma. Hace no muchos años el crecimiento de las redes de datos, la llegada del internet y las nuevas tecnologías facilito la acción de comunicarnos y el acceso a la información.

El cambio tecnológico a permitido la migración de tecnologías, las centrales telefónicas analógicas que han sido usadas por muchos años para la comunicación de instituciones tienen costos bastante elevados de implementación, estás están siendo cambiadas por centrales PBX de telefonía IP que presenta grandes ventajas como los costos por implementación y equipos. Esto es beneficioso para las instituciones que adoptan esta tecnología por que existen diversas opciones para implementarla y bastantes programas para administrar una red de este tipo.

En los últimos años se han desarrollado programas en software libre que realizan las funciones de una central PBX.

Uno de estos programas es Asterisk que es un sistema completo de comunicaciones sobre redes de datos que integra la telefonía, internet, fax y otros elementos. Tiene funciones bastante útiles que vienen incluidas en el programa, uno de los recursos es la Interfaz de Enlace de

Asterisk (AGI: Asterisk Gateway Interface) que permite programar en cualquier lenguaje.

1.2. JUSTIFICACIONES

Una institución requiere varias formas de enviar y recibir información de sus usuarios sobre sus servicios. Por ejemplo: por correo electrónico, notificaciones escritas, o por teléfono; en esta última se pueden utilizar centrales de servicio al cliente para recibir consultas por parte de los usuarios.

Esto puede ser automatizado de tal forma que no represente un gasto mayor por la contratación de personal; la implementación explicada en este documento es una opción bastante económica para cualquier institución que posea una red de datos implementada. Otra de las ventajas es que el usuario es libre de revisar cuantas veces quiera su información y se puede controlar si la información llego al usuario por este medio.

1.3. DESCRIPCIÓN DEL PROYECTO

El proyecto consiste en la implementación de un sistema de consultas de deudas y fechas de pago en un servidor de Voz sobre IP como asterisk.

Este sistema funciona de la siguiente manera:

El usuario llama a la central y cuando le contestan escuchara un menú pidiéndole que digite su número de cédula para identificarse o que presione las teclas 999 seguidas de la tecla numeral para salir del sistema; si el usuario no presiona nada volverá a reproducirse la grabación hasta que se ejecute una de las dos acciones.

Una vez ingresado el numero de cédula este puede tener tres tipos de respuesta. El primero es un mensaje indicándole que su cédula no está registrada si el numero digitado no se encuentra en la base de datos. El segundo es un mensaje indicándole que no tiene deudas si el numero ingresado no registra ninguna deuda en la base de datos. El tercero, cuando la cédula ingresada tenga deudas, se escuchara el valor de la deuda y la fecha de pago seguida de un mensaje pidiéndole que presione la tecla 1 para confirmar que el usuario ha escuchado el mensaje de la deuda y así sucesivamente si el usuario tiene más de una deuda

1.3.1. Objetivos generales

- Implementar una pequeña central telefónica que provea el servicio de poder consultar deudas y fechas de pago a los usuarios registrados de una manera automatizada, que sea fácil de usar y cumpla con los requerimientos de los clientes al realizar sus consultas.

1.3.2. Objetivos específicos

- Demostrar el uso de AGI para el desarrollo del sistema.
- Demostrar el uso de la librería “php-agi” para el desarrollo del script que procesa la llamada.
- Implementar un sistema de consultas usando una base de datos que permita independizar la fuente de datos de Asterisk.
- Implementar una aplicación que permita el ingreso de datos para el uso del sistema.

1.4. METODOLOGÍA

Los pasos para la realización de este proyecto son:

1. Instalación de la base de datos mysql.
2. Iniciar el servicio mysql con los respectivos permisos para su uso.
3. Instalación de php 5.
4. Instalación del programa Asterisk sobre una distribución Linux para el proyecto se utilizo la distribución CentOS 5.
5. Configuración de teléfonos IP con usuarios SIP para pruebas.
6. Instalación y configuración de softphone Zoiper.
7. Diseño e implementación del sistema usando php-agi

1.5. PERFIL DE LA TESIS

En el capítulo 2 se presentan las bases teóricas para mayor comprensión de los conceptos tratados para la realización del proyecto.

En el capítulo 3 se presentan las especificaciones técnicas, la instalación del hardware y las configuraciones respectivas para el funcionamiento del proyecto.

En el capítulo 4 se presentan las pruebas realizadas con los diferentes casos que se pueden presentar en el proyecto.

CAPITULO II

ASTERISK Y PHP AGI

2.1. ASTERISK

Asterisk es un PBX por software de código abierto, creado por Digium, Inc. y una base de usuarios y desarrolladores en continuo crecimiento. Digium desarrolla código fuente y hardware de telefonía de bajo costo que funciona con asterisk. Asterisk funciona en Linux y otras plataformas Unix con o sin hardware que conecte tu servidor con la red global tradicional de telefonía, la PSTN.

Para la interconexión con equipo digital y analógico de telefonía, Asterisk soporta un número de dispositivos de hardware, más notablemente de todos el hardware fabricado por los patrocinadores de Asterisk's, Digium. Digium tiene placas T1 y E1 de 1, 2 y 4 puertos para la interconexión a líneas PRI y bancos de canal. Además, están disponibles tarjetas analógicas FXO y/o FXS de 1 a 4 puertos y son muy populares para instalaciones pequeñas. Las tarjetas de otros vendedores se pueden utilizar para BRI (ISDN2) o tarjetas compatibles de cuatro y ocho puertos BRI basadas en tarjetas compatibles con CAPI o tarjetas de chipset HFC.

2.1.1. Características

Asterisk incluye muchas características encontradas solamente en los sistemas de mensajería unificados de última generación, como:

- Música en espera para los clientes que esperan en cola, soportando transmisión en directo así como música MP3.
- Integración de sistemas Texto-a-Voz (los software de síntesis de la voz como el “Festival” o el “Cepstral” pueden ser integrados).
- Generación de datos de llamada (CDR) para la integración con los sistemas de facturación.
- Integración con sistemas de reconocimiento de voz (tales como el software abierto de reconocimiento de voz “Sphinx”).
- La capacidad de interconectarse con las líneas telefónicas normales, ISDN de tarifa básica y las interfaces PRI.

2.1.2. Canales

Un canal es una conexión que trae una llamada a un PBX asterisk. Un canal podría ser una conexión a un teléfono ordinario fijado a mano o a una línea telefónica ordinaria, o a una llamada lógica (como una llamada telefónica del Internet). Asterisk no hace ninguna distinción entre los canales del estilo de “FXO” y de “FXS”, es decir, no distingue entre las

líneas telefónicas y los teléfonos. Cada llamada se pone o se recibe en un canal distinto.

2.2. AGI(ASTERISK GATEWAY INTERFACE)

Es una interfaz que agrega funcionalidad a Asterisk para que interactúe con diversos lenguajes de programación como: Perl, PHP, C, PASCAL, Bourne Shell, etc., y así poder hacer sistemas más complejos y de mayor utilidad.

- AGI puede controlar el plan de discado, llamado en extensions.conf.
- EAGI da a la aplicación la posibilidad para tener acceso y para controlar los canales de sonidos además de la interacción con el plan del discado.
- FastAGI se puede utilizar para hacer el procesamiento en una máquina remota por conexión de red.
- El deadagi da el acceso a un canal muerto después de colgar, deprecated.

Las variables pasadas por asterisk son:

agi_request – El nombre de archivo del código.

agi_channel – El canal de origen (su teléfono).

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

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

agi_uniqueid - El ID único para la llamada.

agi_version - La versión de Asterisk (desde Asterisk 1.6).

agi_callerid - El número del identificador de llamada (o "unknown" en caso de que sea desconocido).

agi_calleridname – El nombre del identificador de llamada (o "unknown" en caso de que sea desconocido).

agi_callingpres – La presentación para el identificador de llamada en un canal ZAP .

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

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

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

agi_dnid – El id del número marcado (o "unknown" en caso de que sea desconocido) .

agi_rdnis – El número referente a DNIS (o "unknown" en caso de que sea desconocido).

agi_context – contexto origen en extensions.conf.

agi_extension – El número llamado.

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

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

agi_accountcode – Código de la cuenta del canal de origen.

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

La variable que se va a usar más en el proyecto es agi_extension encapsulada dentro de las funciones de la librería php-agi.

2.3. CLASE PHP-AGI

Existe una clase con varias funciones previamente definidas que hacen más fácil la tarea de elaborar códigos PHP para Asterisk. [1] Existen dos

versiones de esta clase, la más reciente es la 2.14. Lo único que hay que hacer es descargarla, descomprimirla e instalarla (copiarla) dentro del directorio **/var/lib/asterisk/agi-bin/**. Ahora, para poder utilizar las funciones que en esta clase se definen, se debe incluir la clase en el código mediante la sentencia:

```
require "/var/lib/asterisk/agi-bin/phpagi-1.12/phpagi.php";
```

Con estas funciones es posible construir aplicaciones más complejas de forma sencilla. Se recomienda revisar las funciones disponibles dentro de cada clase en la documentación correspondiente dentro de su sitio oficial.

CAPITULO III

IMPLEMENTACIÓN

3.1. INTRODUCCIÓN

El objetivo del proyecto de graduación es poner en práctica los conocimientos adquiridos a lo largo de la carrera y utilizarlos juntamente con los sistemas operativos de libre distribución para poder implementar sistemas muy utilizados sin costo alguno.

Este sistema está implementado con las herramientas que el Software Libre nos proporciona que interactúan con otros lenguajes de programación y un sistema de bases de datos básico.

El sistema implementado es muy utilizado en las compañías donde sus clientes necesiten saber sus deudas y fechas de pago. Ejemplo de esto son las compañías de telefonía móvil en donde sus clientes frecuentemente están interesados por conocer su saldo y hasta cuándo está en vigencia su saldo.

3.1.1. Hardware

Este sistema cuenta con un servidor y un teléfono que soporta voz sobre IP, cuyas características detallamos a continuación:

3.1.2. Servidor

Es donde se encuentra el sistema, y por ende, es donde se van a procesar las llamadas de los clientes. El servidor que se utiliza es una computadora con las características mostradas en la tabla I

Tabla I Características del Servidor

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

3.1.3. Teléfono IP

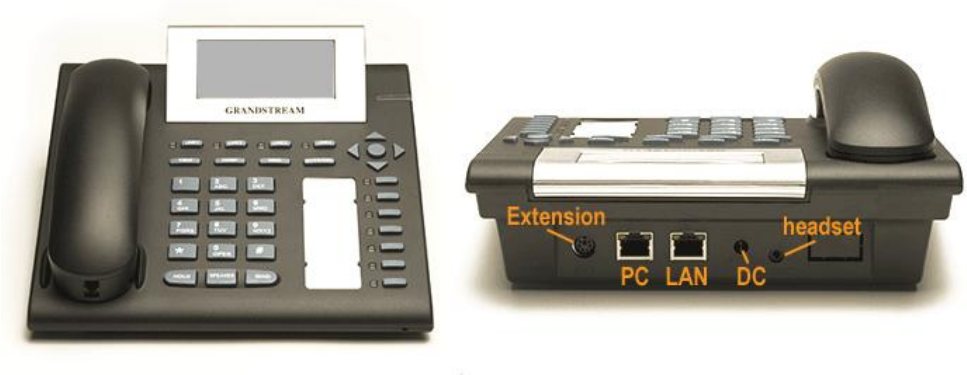

Fig. 3.1 Teléfono IP Grandstream GXP2000

Utilizamos el teléfono de voz sobre IP Grandstream GXP2000, del cual utilizamos sus conectores RJ-45 para conectar el teléfono a la computadora y a la red. (Véase figura 3.1)

3.2. SOFTWARE

3.2.1. Servidor

En el servidor se instalaron los sistemas operativos y programas mostradas en la tabla II

Tabla II Características del servidor – software

Sistema Operativo	Linux
Distribución	Centos 5.2
Arquitectura	x86
Software IP PBX	Asterisk 1.4
Base de datos	MySQL
Software adicional	Java Development Kit 1.5

3.2.2. Instalación de Asterisk

Para instalar Asterisk en el servidor se requieren instalar algunas dependencias previamente, a continuación se detalla todo este proceso:

1. Descargar el paquete estable de Asterisk 1.4

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

2. Se recomienda tener todos los paquetes actualizados:

   - **yum update o yum upgrade –y**

3. Actualizar y activar los siguientes paquetes:

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

   openssl: librerías de desarrollo para soporte RSA y MD5.

   ncurses: para el CLI y para que otro aplicativo?

   bison, ncurses, openssl-devel y las dependencias relacionadas.

Fuentes del kernel de Linux (librerías del sistema)

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

4. Para instalar las dependencias y soporte para DB:
    - **yum -y install mysql-server mysql-devel newt-devel unixODBC unixODBC-devel libtool-ltdl libtool-ltdl-devel mysql-connector-odbc**

En el caso de tener problemas de “sources for kernel”:

yum –y install kernel-devel

yum –y update kernel

uname -r

5. Descargar los paquetes en /usr/src
    - **wget -c http://downloads.digium.com/pub/asterisk/asterisk-1.4-current.tar.gz**
    - **wget -c http://downloads.digium.com/pub/asterisk/asterisk-addons-1.4-current.tar.gz**
    - **wget -c http://downloads.digium.com/pub/telephony/dahdi-tools/dahdi-tools-current.tar.gz**
    - **wget -c http://downloads.digium.com/pub/telephony/dahdi-linux/dahdi-linux-current.tar.gz**
    - **wget -c http://downloads.digium.com/pub/libpri/libpri-1.4-current.tar.gz**

6. Siguiendo el LSB (Linux Standard Base) Asterisk deberá ser descomprimido en /usr/src/

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

7. Ejecutar el siguiente grupo de comandos:

- cd dahdi-linux-current, make, install
- cd dahdi-tools –current, ./configure, make, make install, make config
- cd ../libpri-1.4.7, make, make install
- cd ../asterisk-1.4.22, make clean, ./configure, make menuconfig
- cd ../asterisk-addons-1.4.7, make clean, ./configure

3.2.3. Instalación de mysql

A continuación se detalla el procedimiento para instalar mysql y php en Asterisk:

1. Verificar e instalar MySQL:

   - **yum install mysql* unixODBC-devel -y**

   o sino:

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

2. Levantar mysql y asignar password a root:

- **service mysqld start mysqladmin -u <root> password <contraseña_de_root>**

3. Ingresar al directorio donde se encuentran las fuentes de asterisk add-ons e instalar el soporte para mysql:
   - **cd /usr/src/asterisk-addons-**
   - **./configure**
   - **make menuselect**
   - **make**
   - **make install**

3.2.4. Instalación de php

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

1. Para agregar php para trabajar con mysql, ejecute el comando:
   - **yum install php***
2. Para que se pueda ejecutar un código, se debería cambiar los permisos de ejecución mediante el comando:
   - **chmod 777 <ruta_absoluta_o_nombre_archivo.php>**
3. Para correr un código php en consola se debe ubicar en la ruta donde se encuentra el archivo o especificar la ruta absoluta en el comando y ejecutar el comando:
   - **! /usr/bin/php -q <archivo.php>**

4. Adicionalmente se debe descargar y colocar la librería phpagi versión 2.14 en la ruta /var/lib/asterisk/agi-bin, para poder ejecutar los scripts creados en php.

3.3. BASE DE DATOS DEL SISTEMA

Este sistema cuenta con una base de datos almacenada en MySQL. Esta base de datos consiste en dos tablas:

1. Clientes
2. Deudas

Por tanto que cada cliente está interesado en conocer cuantas deudas tiene y las fechas de pago de cada deuda, los datos se encuentran almacenados en estas dos tablas.

Un cliente puede estar relacionado con ninguna o muchas deudas, mientras que una deuda siempre se relaciona con un cliente.

3.3.1. Lenguaje de Descripción de Datos del Sistema

A continuación se detalla la creación de la base de datos de este sistema, todos estos comandos son ejecutados desde la línea de comandos del terminal de MySQL.

3.3.2. Ingreso a mysql

La contraseña del usuario root está en blanco para ingresar a MySQL

mysql –u root -p

3.3.3. Creación de la base de datos

La base de datos tiene por nombre “SistemaConsultas” y se deberá emplear el comando:

create database SistemaConsultas;

3.3.4. Creación de las tablas

Tabla Clientes

create table Clientes

(

IDCliente int not null auto_increment,

Nombres varchar(40) not null,

Apellidos varchar(40) not null,

telefono varchar(40),

celular varchar(40),

edad int,

sexo varchar(40),

domicilio varchar(200),

cédula varchar(40) not null,

estado int not null,

primary key (IDCliente)

)ENGINE=INNODB;

Tabla Deudas

create table Deudas

(

IDDeuda int not null auto_increment,

IDCliente int not null,

monto double(7,2),

FechaPago date,

descripcion varchar(200),

cancelado int not null,

estado int not null,

Confirmacion int,

primary key (IDDeuda),

foreign key (IDCliente) references Clientes (IDCliente) on delete cascade

)ENGINE=INNODB;

Como podemos ver el sistema tiene la restricción de que la deuda de un cliente está en el rango entre 0 a 99999.99 dólares, y cada deuda tiene

una referencia a un cliente, si un cliente se elimina de la base de datos, automáticamente se eliminan del sistema las deudas de ese cliente.

3.4. CONFIGURACIONES EN ASTERISK

3.4.1. Configuración del archivo sip.conf

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

sip.conf

```
[general]
rtcachefriends=yes
rtupdate=no
rtautoclear=yes
srvlookup=yes
disallow=all
allow=alaw
allow=ulaw
allow=gsm
```

language=es

callwaiting=no

qualify=yes

calltransfer=no

callforwarding=no

[901]

type=friend

secret=901

qualify=yes

nat=yes

host=dynamic

canreinvite=no

context=incoming

En esta configuración se ha establecido un usuario SIP: 901

Cabe recalcar que este usuario SIP se lo creó con el propósito de poder establecer una conexión de prueba con el Sistema, es decir, el sistema no utiliza usuarios SIP, todos los usuarios están almacenados en la base de datos del sistema. Este usuario SIP sólo lo puede utilizar el administrador. En el contexto se puede también cambiar “incoming” por

“internal” si se desea trabajar con una red interna que no requiera conectarse con una red externa.

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

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

A continuación la configuración para los mismos sería así:

chan_dahdi.conf

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

context=internal

signaling=fxs_ks

channel => 4

system.conf

fxsks=4

echocanceller=mg2,4

loadzone=us

defaultzone=us

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

- **dahdi_genconf**

3.4.3. Configuración del archivo extensions.conf

Se encuentra en la ruta /etc/asterisk y consiste en el plan de marcado de este sistema.

La función principal de este archivo es de invocar al código donde se encuentra almacenado el programa principal.

; contexto general sirve para establecer los parámetros generales

; del plan de marcado.

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

; contexto “internal” sirve para especificar el procedimiento a ;realizarse
para comunicarse con alguna de las extensiones en el interior de la PBX.
[internal]
exten => 789,1,AGI(proyecto.php)
```

3.5. PROGRAMA PRINCIPAL

3.5.1. Descripción general

El sistema recibe una llamada, para lo cual si la extensión marcada es “789” el archivo extensions.conf invoca al código “proyecto.php” donde se almacena el programa principal y se ejecuta el código.

El sistema contesta la llamada y se reproduce un mensaje de bienvenida donde se le indica al cliente que ingrese su número de cédula o si quiere salir del sistema que digite “999”. El sistema espera a que el cliente ingrese su cédula, captura el número de cédula ingresado y con ese

número ejecuta una consulta a la base de datos para verificar si el cliente está registrado y si tiene deudas. Una vez que la base de datos retorna los datos de las deudas y fechas de pago del cliente, se reproducen unas grabaciones que le indican al cliente la cantidad de deudas que tiene, el monto y la fecha de pago de cada deuda. Por cada deuda se reproduce una grabación que le indica al cliente que digite la tecla “1” para confirmar que el cliente escuchó la grabación su deuda. Si digitó “1” se graba en la base de datos, en la tabla de Deudas, en el campo Confirmación un “1”, que indica que el cliente si escuchó la grabación, de lo contrario el campo Confirmación permanece con un valor de “0”, que quiere decir que el cliente no ha confirmado que ha escuchado una de sus deudas. Si el cliente desea salir del sistema puede digitar “999” y con esto se reproduce un mensaje de despedida y se termina la llamada.

Después de que el cliente escuchó sus deudas, el sistema reproduce la grabación inicial y si el cliente desea puede volver a escuchar sus deudas o de lo contrario salir del sistema.

Las grabaciones fueron realizadas en Asterisk y se encuentran almacenadas en la ruta /var/lib/asterisk/sounds.

Las voces que vienen pregrabadas en Asterisk y que están en español también se encuentran en la ruta /var/lib/asterisk/sounds.

Los archivos proyecto.php y phpagi.php se encuentran en la ruta /var/lib/asterisk/agi-bin y se los hace ejecutables para poder utilizarlos.

3.5.2. Casos del sistema

Al momento en que llegue una llamada al sistema, se pueden producir cuatro distintos casos:

1. El cliente no digite ningún número: El sistema esperará siete segundos para que el cliente ingrese su cédula, sino ingresa ningún número, el sistema volverá a reproducir la grabación inicial pidiendo por un número de cédula.

2. El cliente ingresa una cédula no registrada en la base de datos: El sistema hará la consulta a la base de datos, pero al no encontrarse registrada su cédula se reproducirá una grabación que le indique que el cliente no se encuentra registrado en el sistema. Luego se reproduce la grabación inicial.

3. El cliente ingresa su cédula que sí se encuentra registrada en el sistema pero no tiene deudas: El sistema hará la consulta a la base de datos, pero ésta no devolverá los datos de las deudas por

lo que el cliente no posee deudas, entonces el sistema reproduce una grabación que le indica al cliente que no posee deudas. Luego se reproduce la grabación inicial.

4. El cliente ingresa su cédula que sí está registrada y también posee deudas: Mediante grabaciones, el sistema le indica al cliente la cantidad de deudas que posee, luego le indica deuda por deuda, el monto de la deuda y la fecha a pagar. Por cada deuda el sistema le pide al cliente que digite un “1” para confirmar que escuchó la deuda, ésta confirmación se la registra en la base de datos, pero si el cliente no escuchó la deuda, no se altera la base de datos. Luego de que el cliente escuchó sus deudas, se reproduce la grabación inicial.

3.5.3. Código fuente

```
#! /usr/bin/php -q
<?php
set_time_limit(30);
error_reporting(E_ALL);
require ('phpagi-2.14/phpagi.php');
/*autor: Andrea*/
   //creo mi conexion
$con = mysql_connect ('localhost', 'root', 'labtelecom09')          or die
   (mysql_error());

      //selecciono mi bd
      mysql_select_db('SistemaConsultas',$con)or die (
   mysql_error());

$agi = new AGI();
```

```
$agi->answer();
$agi->exec(Playback,"Bienvenida");

do{
   $result = $agi->get_data("Menu", 7000,10);
   $nummarcado = $result['result'];

   //Verifico si el número digitado es la tecla de escape para salir del
sistema

   if($nummarcado!='999' && $nummarcado!= null)
   {

   $sql="SELECT * FROM Clientes WHERE cédula='$nummarcado';";

            $res=mysql_query($sql,$con) or die ('ERROR SQL');
            $r1=mysql_fetch_array($res);

            if($r1['cédula']==$nummarcado)
            {

                  $sql="SELECT * FROM Deudas WHERE
                  IDCliente='".$r1['IDCliente']."'; and cancelado=0";

                  $res2=mysql_query($sql,$con)or die (mysql_error());

                  if(mysql_num_rows($res2)== 0)
                  {
                        $agi->exec(Playback,"SinDeudas");
                  }

//si llega a este if es porque si esta la CI y tiene deuda

                  else
                  {

                        $agi->exec(Playback,"UstedTiene");

                        $CantDeudas = mysql_num_rows($res2);
                        $agi->say_number($CantDeudas);
                        $agi->exec(Playback,"Deudas");


                        while($r2=mysql_fetch_array($res2))
```

```
                {
                        $IDDeuda=$r2['IDDeuda'];

                        $agi->exec(Playback,"UstedDebe");
                        $monto=$r2['monto'];
                        $fechap=$r2['FechaPago'];

                        $montos=explode(".",$monto);
                        $fechas=explode("-",$fechap);

                        $agi->say_number($montos[0]);
                        $agi->exec(Playback,"DolaresCon");


                        if($montos[1]=='00')
                        {
                                $agi->say_number(0);
                        }

                        else
                        {
                                $agi->say_number($montos[1]);

                        }

                        $agi->exec(Playback,"Centavos");

                        $agi-
>exec(Playback,"ydebepagarantesde");

                        $agi->say_number($fechas[2]);
                        $agi->exec(Playback,"de");

                        switch($fechas[1])
                        {
                                case 1: $cad="Enero"; break;
                                case 2: $cad="Febrero"; break;
                                case 3: $cad="Marzo"; break;
                                case 4: $cad="Abril"; break;
                                case 5: $cad="Mayo"; break;
                                case 6: $cad="Junio"; break;
                                case 7: $cad="Julio"; break;
                                case 8: $cad="Agosto"; break;
                                case 9: $cad="Septiembre"; break;
```

```
                                        case 10: $cad="Octubre"; break;
                                        case 11: $cad="Noviembre"; break;
                                        case 12: $cad="Diciembre"; break;

                                    }

                                    $agi->exec(Playback, $cad);
                                    $agi->exec(Playback,"del");
                                    $agi->say_number($fechas[0]);


        //Mensaje de Confirmacion

    $confirmacion = $agi->get_data("MensajeConfirmacion", 7000,1);

    $num = $confirmacion['result'];


                    if($num==1)
                    {

                    $sql2="update Deudas set Confirmacion=1 WHERE
                            IDDeuda=$IDDeuda;";

                    $res=mysql_query($sql2,$con) or die ('ERROR SQL');

                    }


                    }

            }

        //Si no existe la cédula digitada en la base se reproduce una
                grabación que no está registrado y regresa a menú

    }

    else{ $agi->exec(Playback,"CédulaNoReg");}

    }
}
```

```
while($nummarcado!='999');
$agi->exec(Playback, "Despedida");
$agi->hangup();
?>
```

3.5.4. Descripción de las funciones utilizadas

La librería phpagi nos permite realizar distintas operaciones sobre el plan de marcado y sobre bases de datos. En este proyecto utilizamos distintas funciones que nos provee ésta librería las cuales detallamos a continuación:

3.5.5. Funciones sobre la base de datos

mysql_connect ('localhost', 'root', 'labtelecom09')

Función que nos permite conectar con la base de datos, el primer parámetro indica la ubicación de la base de datos, el segundo parámetro indica el nombre del usuario al cual pertenece la base de datos y el tercer parámetro indica la clave de ese usuario.

mysql_select_db('SistemaConsultas',$con)

Función que selecciona la base de datos con la cual vamos a trabajar, el primer parámetro indica el nombre de la base de datos y el segundo parámetro indica el nombre del puntero que tiene la conexión hacia la base de datos.

$res=mysql_query($sql,$con)

Función que ejecuta una petición, el primer parámetro es una cadena de caracteres que contiene la petición a ser ejecutada y el segundo parámetro indica el nombre del puntero que tiene la conexión hacia la base de datos.

$r1=mysql_fetch_array($res)

Función que devuelve un arreglo de claves de cada una de las columnas de la base de datos la petición ejecutada.

3.5.6. Funciones sobre el plan de marcado

error_reporting(E_ALL)

Crea un log con todos los errores ocurridos durante la ejecución del código.

$agi = new AGI()

Crea una nueva instancia de la clase agi.

require ('phpagi-2.14/phpagi.php')

Permite que las funciones en la librería phpagi sean usadas en el código principal.

$agi->answer()

Contesta la llamada.

$agi->exec(Playback,"ArchivodeAudio")

Reproduce una grabación.

$agi->get_data("ArchivodeAudio", tiempoEspera,MaxDigitos);

Captura la extensión marcada luego de reproducirse un archivo de audio dentro de un tiempo de espera, el tercer parámetro indica la cantidad máxima de dígitos que se pueden ingresar.

agi->say_number($Numero)

Reproduce en audio el número que se le ha enviado.

$arreglo=explode("Caracter",$cadena)

Se le envía una cadena y retorna un arreglo de los elementos de la cadena separados por el carácter que se envía como primer parámetro.

$agi->hangup()

Termina la llamada.

3.5.7. Descripción de archivos de audio

Las grabaciones fueron realizadas utilizando Asterisk, con la función “Record” y llamando a la extensión 999. La función “Playback” nos permite escuchar lo que acabamos de grabar. Estos archivos están almacenados en la ruta /var/lib/asterisk/sounds y fueron grabados de la siguiente manera:

```
exten => 999,1,Record(NombredelArchivo.gsm)
exten => 999,2,Playback(${RECORDED_FILE})
exten => 999,3,Hangup()
```

Los siguientes son los archivos de audio grabados utilizando Asterisk, todos tienen el formato .gsm:

- Bienvenida
- SinDeudas
- UstedTiene
- Deudas
- UstedDebe
- DolaresCon
- Centavos
- ydebepagarantesde
- de

- del
- MensajeConfirmacion
- CédulaNoReg
- Despedida

Otros archivos de audio que se emplean en este proyecto vienen pregrabados en Asterisk, y los utiliza la librería “phpagi.php” junto con la función “text-to-sound” para reproducir texto a audio.

3.5.8. Sistema de administración de la base de datos

Se creó un sistema que administra la base de datos donde se encuentra almacenada toda la información concerniente a los clientes y sus deudas. Este sistema puede ingresar, modificar, consultar y eliminar los registros de los clientes y sus deudas. Este sistema fue hecho en Java utilizando la librería “mysql-connector-java5.1.5-bin.jar” y lo pueden utilizar los administradores del sistema.

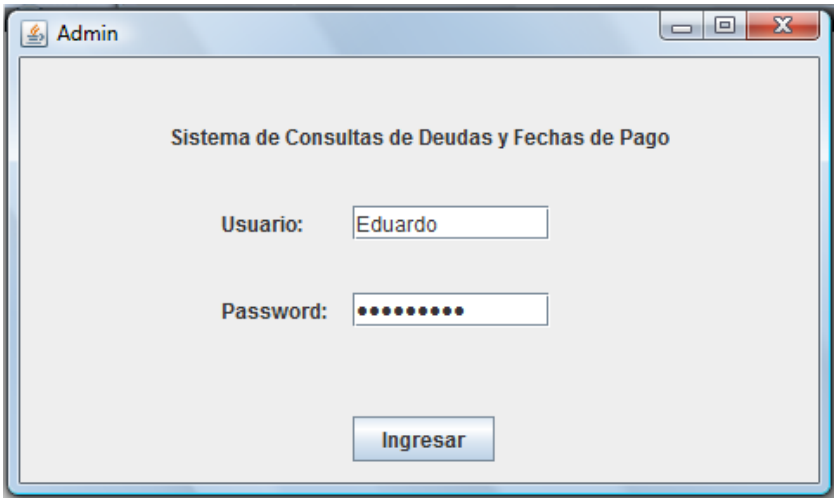

Fig. 3.2 Pantalla de login del sistema de administración

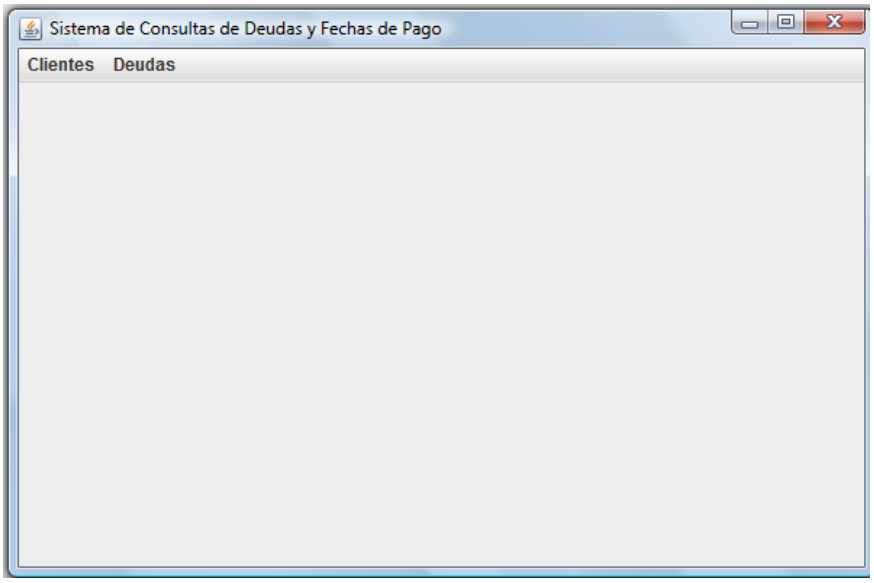

Fig.3.3 Pantalla principal del sistema de administración

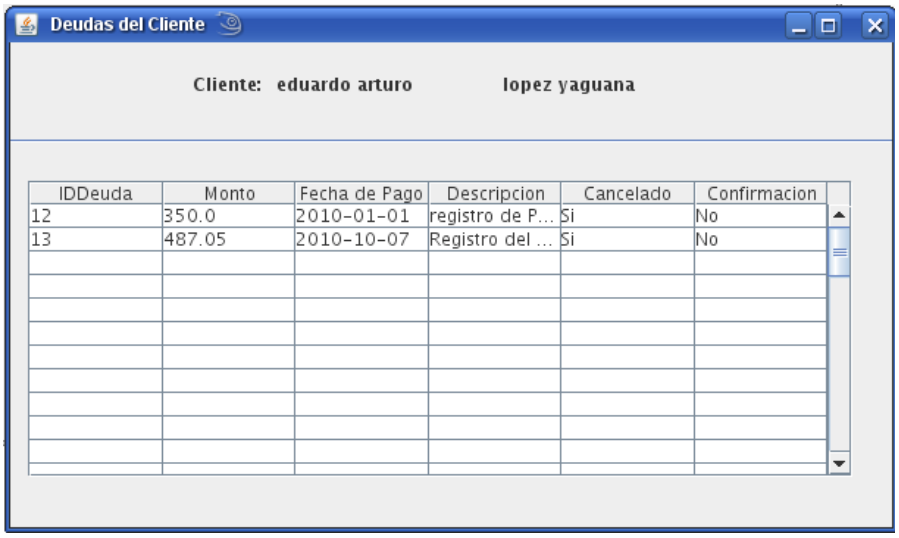

Fig. 3.4 Consulta de deudas de un cliente

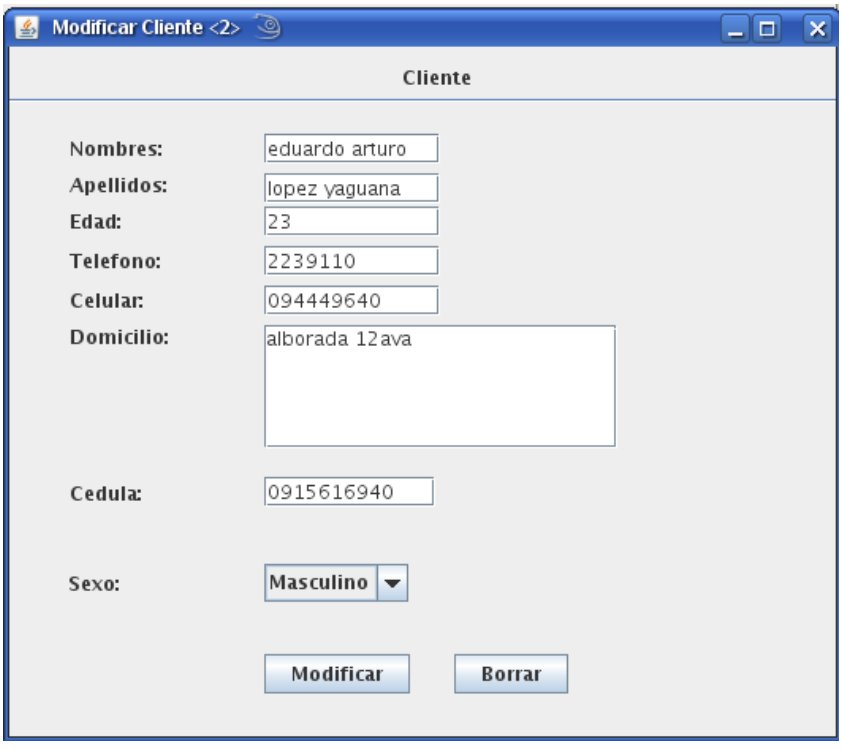

Fig. 3.5 Modificación de los datos de un cliente

Fig. 3.6 Modificación de una deuda de un cliente

3.5.9. Configuración del teléfono IP Grandstream GXP2000

Para poder configurar este teléfono con los usuarios SIP para poder realizar las llamadas, ingresamos a cualquier navegador de internet con la dirección IP del teléfono, luego ingresamos la clave de administrador y el usuario, y procedemos a configurar el teléfono con los usuarios SIP previamente creados en el archivo sip.conf. A continuación se detalla la configuración:

Account Name:901

SIP server: 901

Outbound proxy: direccion_ip_del_servidor_asterisk

SIP user ID: 901

Authenticate ID: 901

Authenticate password: 901

Name: 901

Local Sip port: 5060

CAPÍTULO IV

FUNCIONAMIENTO Y PRUEBAS

4.1. ACTIVACIÓN DEL SERVICIO DE MYSQL

Para que el sistema pueda hacer las consultas pertinentes, MySQL debe estar activado

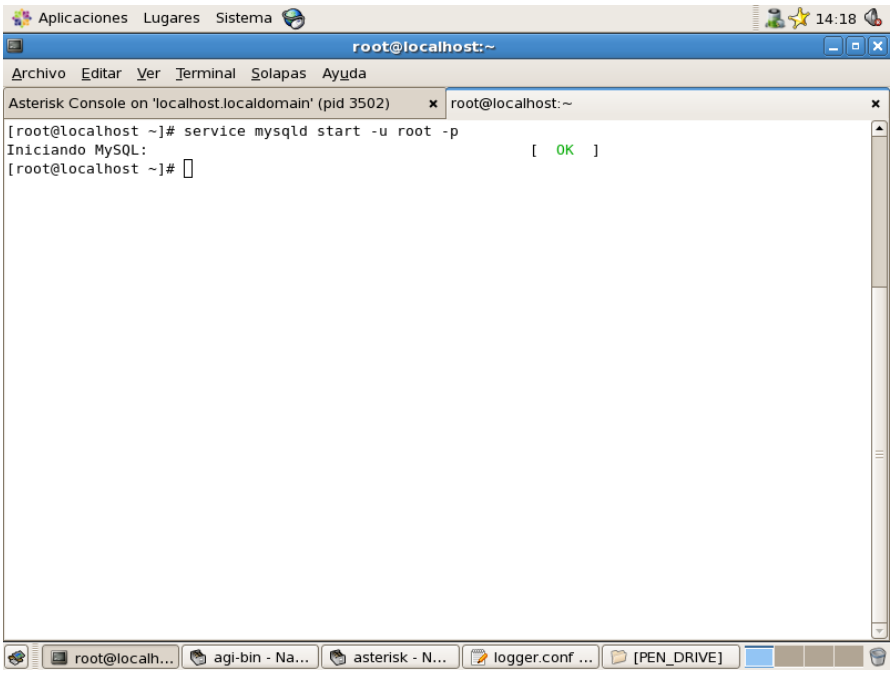

Fig. 4.1 Activación del servicio de MySQL

4.2. INGRESO DE DOS USUARIOS

Antes de utilizar el sistema, ingresamos dos usuarios a la base de datos desde el programa que administra la base de datos del sistema, para lo cual tenemos que ingresar al sistema con el nombre de uno de los administradores y su clave:

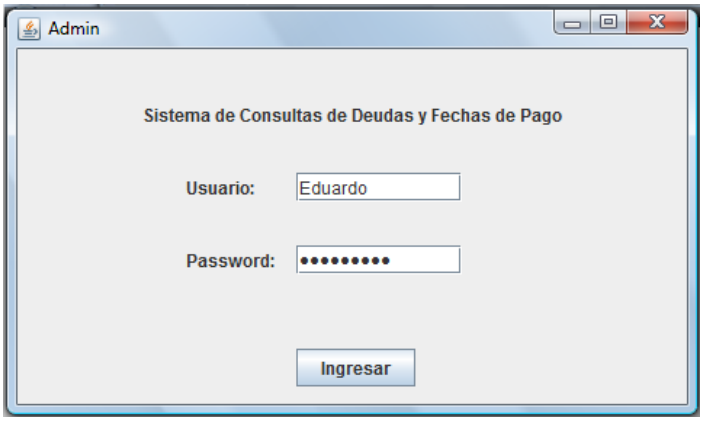

Fig. 4.2 Login del administrador

Luego procedemos a ingresar usuarios en la base de datos:

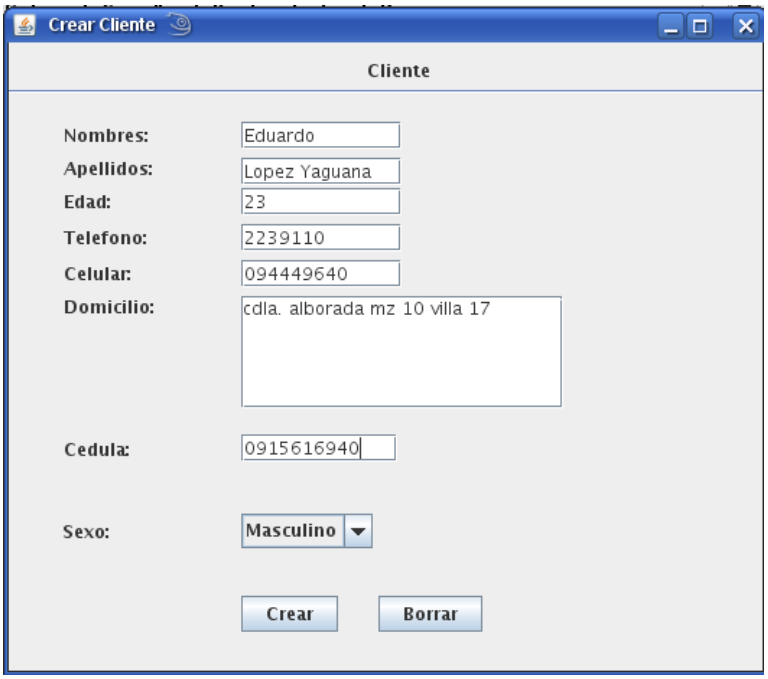

Fig. 4.3 Ingreso del primer usuario a la base de datos

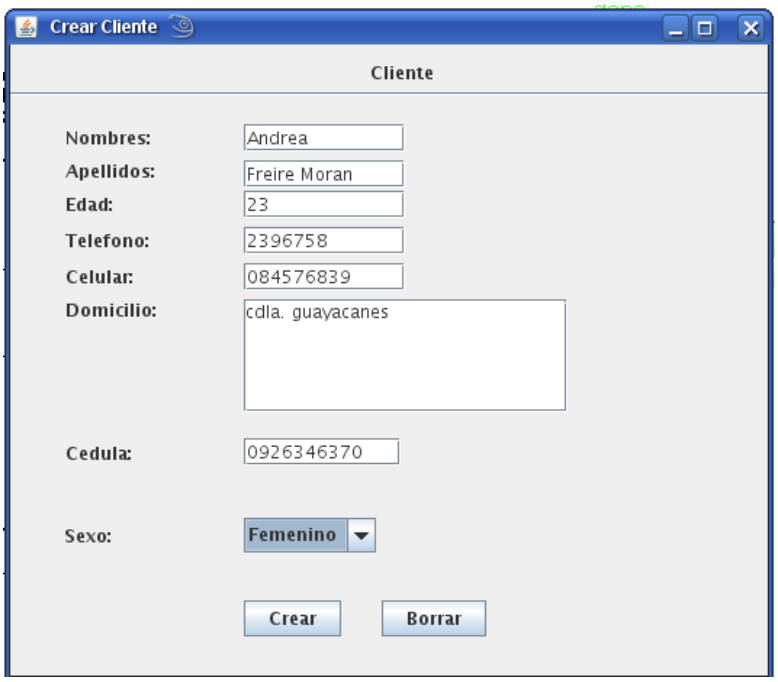

Fig. 4.4 Ingreso del segundo usuario a la base de datos

4.3. INGRESO DE DEUDAS A UN USUARIO

Para este ejemplo se van a ingresar dos deudas a un usuario, y el otro usuario quedará registrado en la base de datos pero sin deudas.

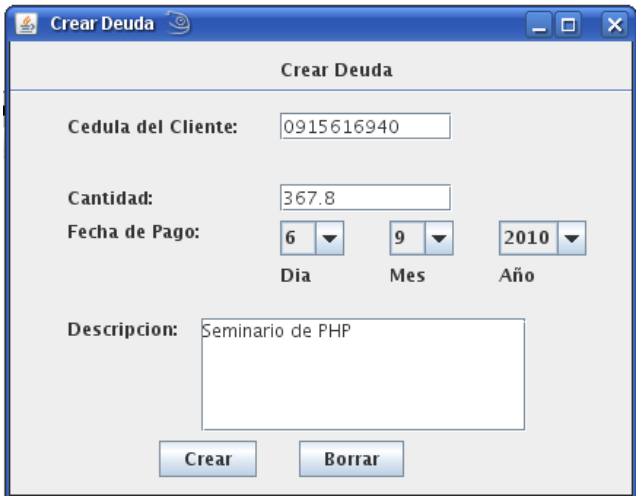

Fig. 4.5 Ingreso de la primera deuda a un usuario

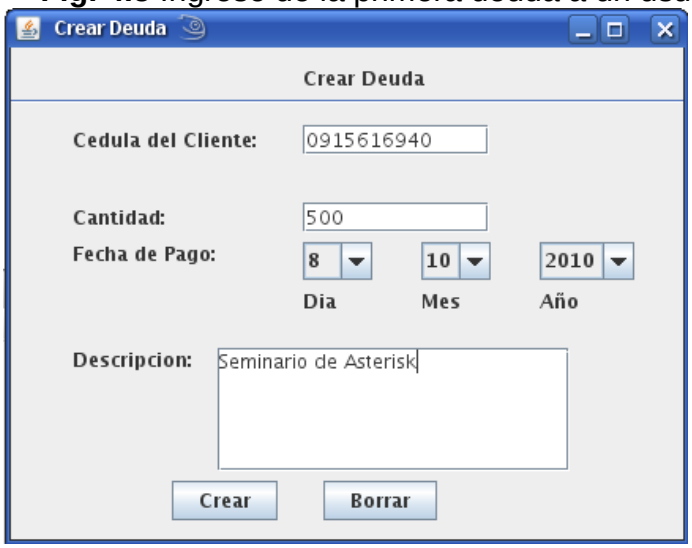

Fig. 4.6 Ingreso de la segunda deuda a un usuario

Con ambos usuarios registrados y con las deudas registradas para uno de los usuarios ahora procedemos a probar el sistema.

4.4. ACTIVACIÓN DE ASTERISK

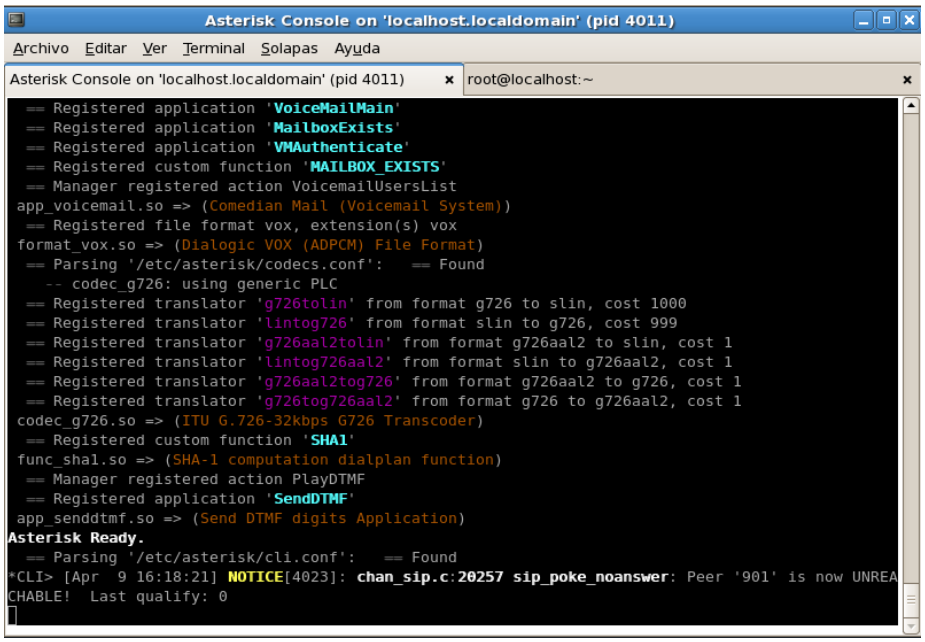

Fig. 4.7 Activación de Asterisk

4.5. LLAMADA AL SISTEMA CONSULTAS

Se marca la extensión 789 y el sistema contestará la llamada. Luego el sistema reproduce el mensaje de bienvenida y espera siete segundos para que el usuario digite su cédula.

El sistema reproduce grabaciones de audio a los distintos casos de ingreso de cédulas que se den, a continuación se mostrará el reporte que emite Asterisk a cada uno de estos casos.

4.5.1. Respuesta del sistema al ingreso de una cédula registrada con deudas

Una vez que hemos llamado al sistema e ingresado una cédula registrada, el sistema busca las deudas del cliente y reproduce las grabaciones pertinentes de acuerdo al resultado.

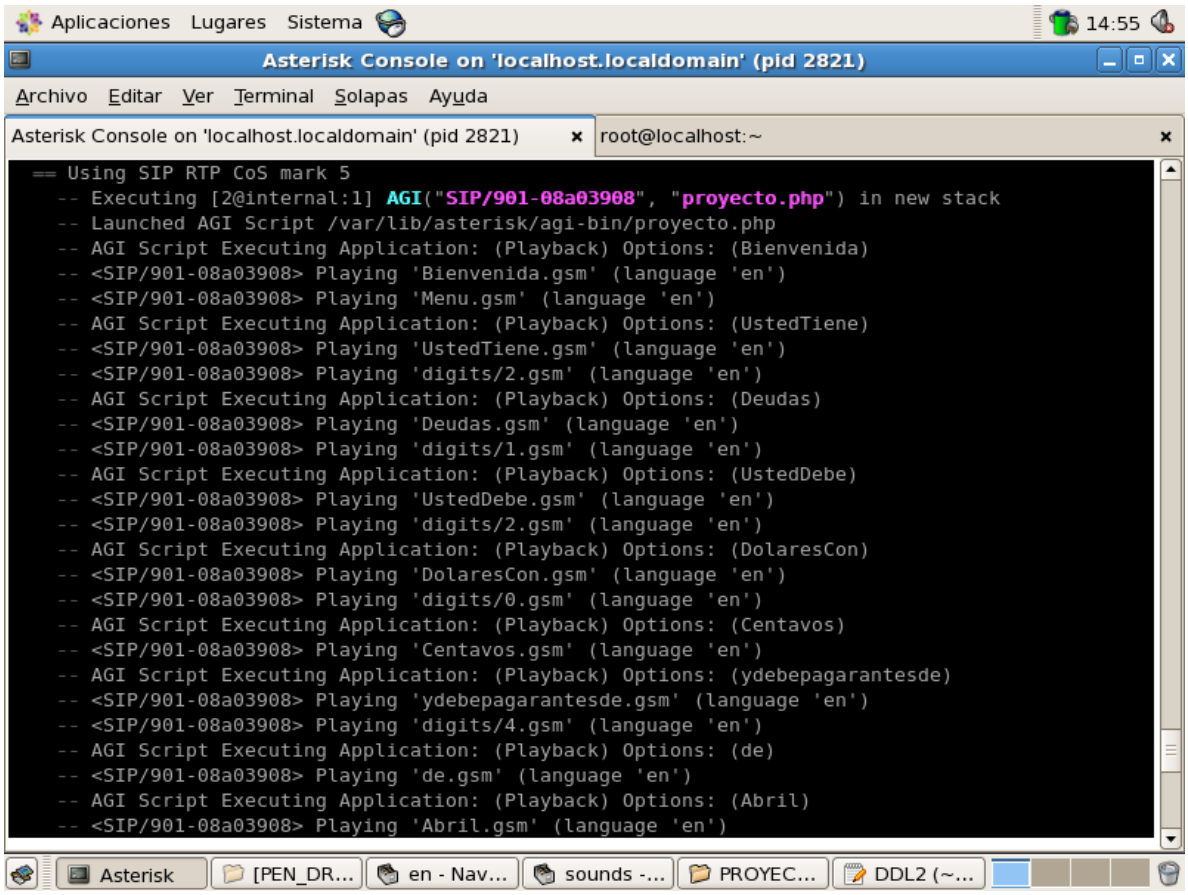

Fig. 4.8 Respuesta del sistema al ingreso de una cédula en el servidor

4.5.2. Confirmación del usuario

Cuando el usuario ha escuchado la deuda, el sistema reproduce una grabación indicándole que digite uno si escuchó la grabación. Cuando el usuario confirma que escuchó la deuda, se graba en la base de datos que esa deuda sí fue escuchada.

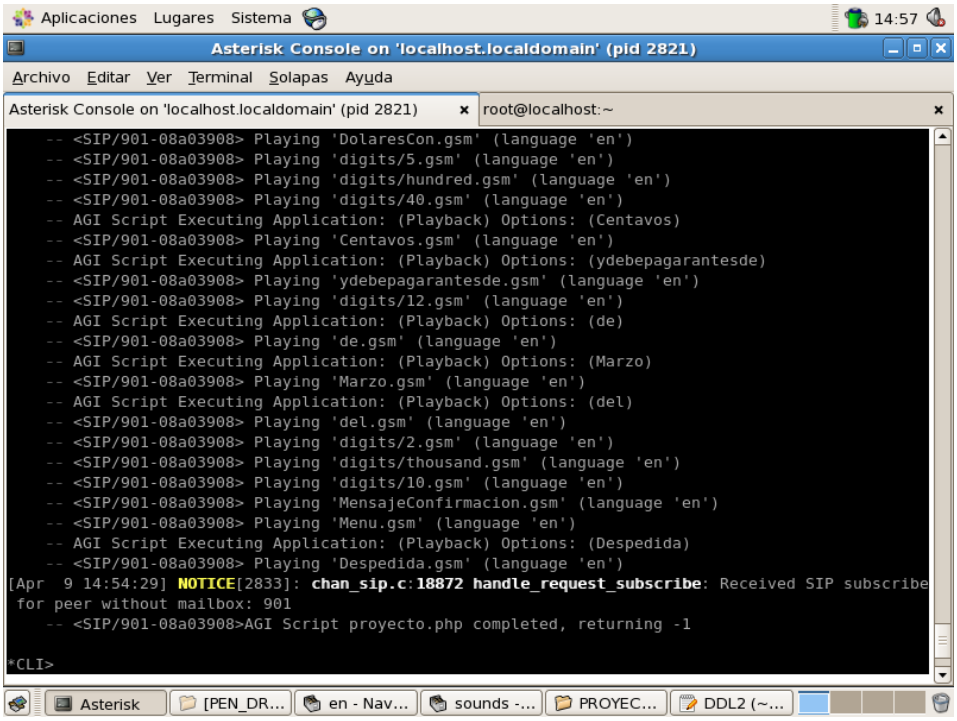

Fig. 4.9 Confirmación del usuario

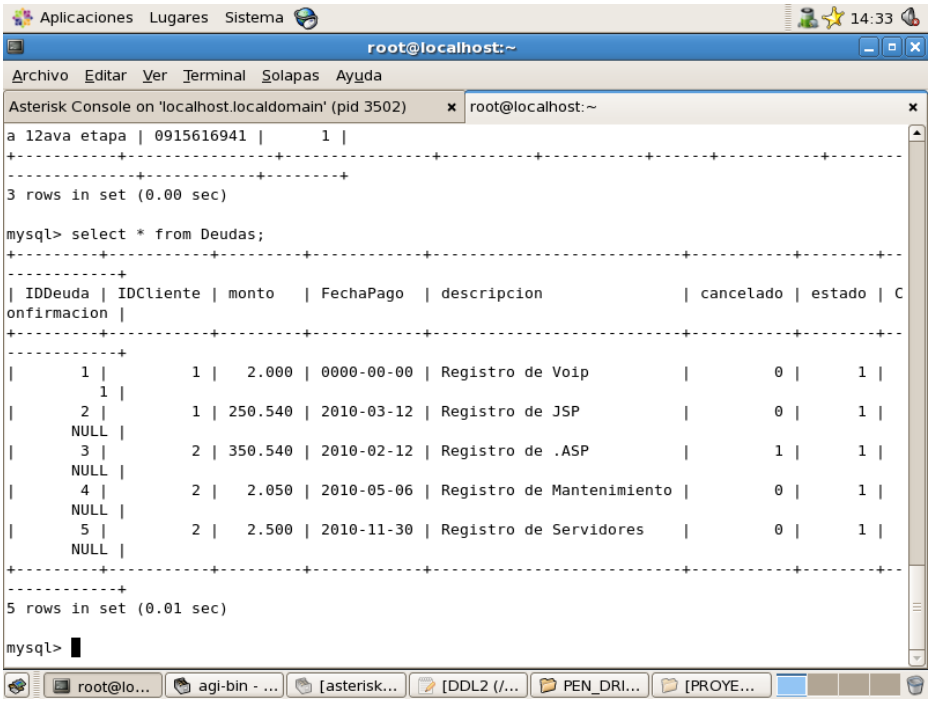

Fig. 4.10 Confirmación del usuario en la base de datos

4.5.3. Respuesta del sistema a una cédula registrada sin deudas

El sistema reproducirá una grabación indicando que el usuario no posee deudas pero si está registrado.

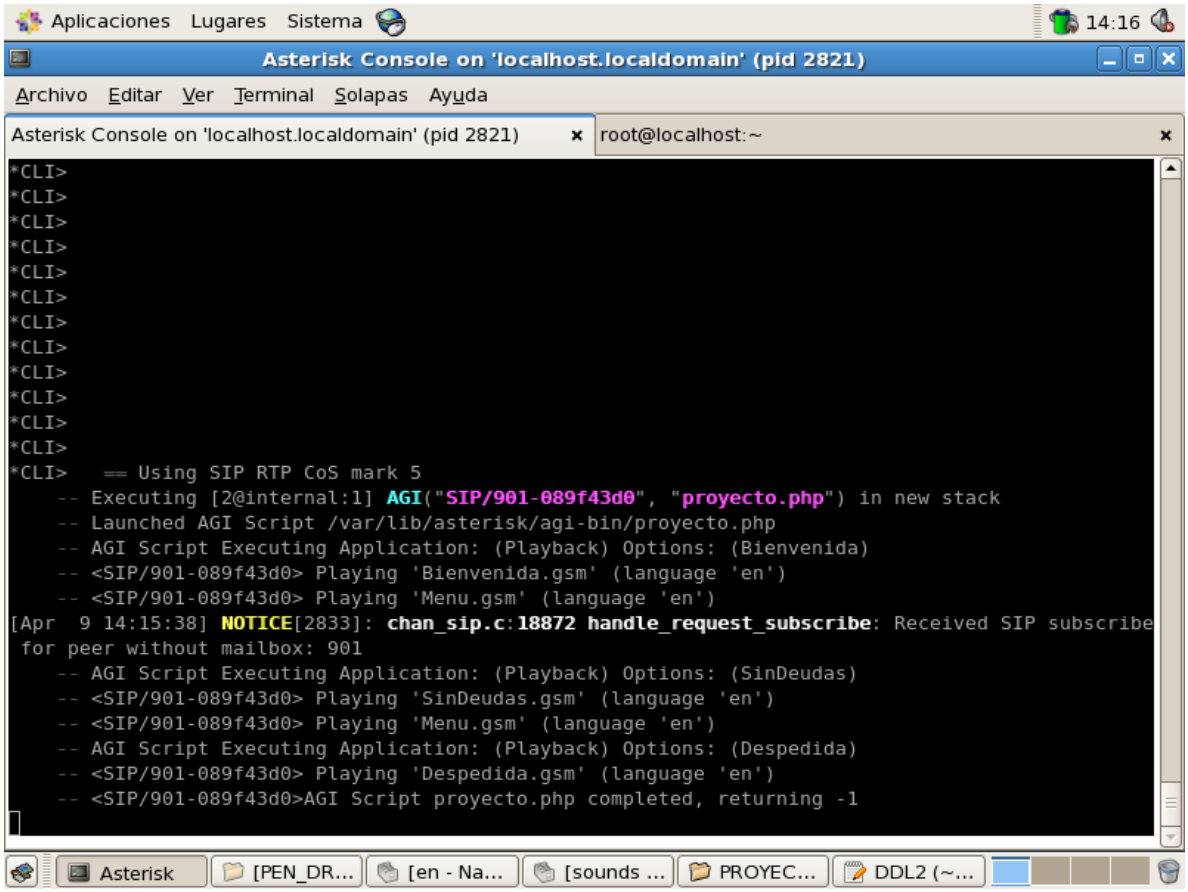

Fig. 4.11 Respuesta del sistema a una cédula registrada sin deudas

4.5.4. Respuesta del sistema a una cédula no registrada

El sistema reproducirá una grabación indicando que el usuario no posee deudas pero si está registrado.

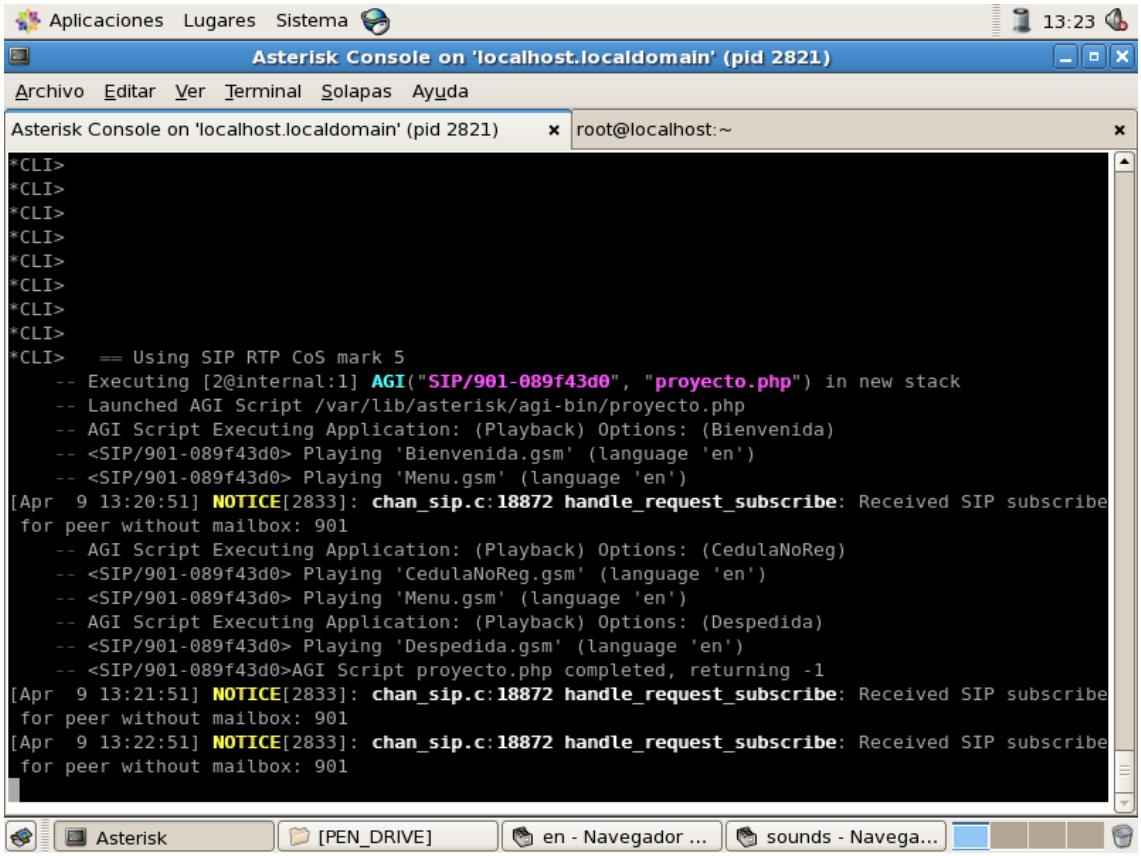

Fig. 4.12 Respuesta del sistema a una cédula no registrada

4.5.5. Fin de la llamada

La llamada termina cuando el usuario digita 999 o cuelga el teléfono.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

1. La implementación de un sistema telefónico automatizado de consultas de uso sencillo que satisfaga las necesidades del usuario, es fácil de realizar con las funciones para crear IVR's que nos ofrece asterisk.
2. Se pueden desarrollar funciones a medida de una institución usando la interfaz de desarrollo de asterisk AGI, con el uso de la librería phpagi se facilita en gran manera el desarrollo de aplicaciones complejas sobre Asterisk, permitiéndonos trabajar con un motor de base de datos tan utilizado como lo es MySQL.
3. El uso del software libre nos proporciona una gran variedad de herramientas y librerías, tiempos menores de desarrollo lo que nos permite la implementación de sistemas útiles, eficientes, robustos, adaptables a la institución, escalables y a un bajo costo.
4. Para administrar la base de datos que maneja el sistema de consultas se necesita desarrollar una aplicación que permita poder ejecutar todas las operaciones sobre ella.

Recomendaciones

1. Para utilizar este sistema con una red externa es necesario poner la extensión del sistema en el contexto “incoming” y en el archivo chan_dahdi.conf especificar que se trabaja con el contexto “incoming”.
2. Utilizar el sistema operativo CentOS puesto que es bastante estable y es dedicado a los servidores.
3. Levantar los servicios que se utilizan con este sistema, tales como MySQL, como administrador para poder acceder a los datos.
4. Descargar las voces en español que trabajan con este sistema y colocarlas en la ruta var/lib/asterisk/sounds para su correcto funcionamiento.
5. Renombrar la carpeta que contiene los archivos de audio de las voces en ingles.
6. Ubicar la ruta donde se encuentre almacenado el compilador de php.

Trabajo a futuro

Este sistema puede ser modificado en un corto o mediano plazo según como vayan cambiando los requerimientos de los usuarios para hacer distintas consultas. Por ejemplo, se podría añadir al sistema la opción de consultar los seminarios que están próximos a ser dictados, consultar cuándo se van a dictar y su costo y hasta cuándo se puede pagar. También se puede añadir la opción de reservar un registro de un seminario al digitar una tecla e inmediatamente se le indicaría al usuario hasta cuando está vigente su cupo.

APÉNDICE

APÉNDICE A

Sistema administrador de la base de datos

Este sistema fue hecho con tres distintas capas para su funcionamiento: la interfaz gráfica, el modelo de negocios y la capa que trabaja sobre la base de datos.

La interfaz gráfica hace un llamado a una función del modelo de negocios y el modelo de negocios a su vez llama a la capa que trabaja sobre la base de datos.

Con esta distribución entre capas podemos organizar mejor el sistema y hacerlo escalable, puesto que todas las capas son independientes entre sí.

El sistema tiene dos entidades: Cliente y Deuda, los cuales se encuentran en el modelo de negocios.

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package BM;

/**
 *
 * @author edu
 */
public class Cliente {

    private int IDCliente,edad,estado;
    private String Nombres,
Apellidos,telefono,celular,domicilio,sexo,cédula;

    public Cliente(){}
```

```
    public Cliente(int id,String Nombres,String Apellidos,int
edad,String sexo,String telefono,
                   String celular,String dir,String cedula,int
estado){

        this.IDCliente=id;
        this.Nombres=Nombres;
        this.Apellidos=Apellidos;
        this.edad=edad;
        this.sexo=sexo;
        this.telefono=telefono;
        this.celular=celular;
        this.domicilio=dir;
        this.estado=estado;
        this.cédula=cedula;
    }
    public void setCampos(int id,String Nombres,String
Apellidos,int edad,String sexo,String telefono,
                   String celular,String dir,String cedula,int
estado){

        this.IDCliente=id;
        this.Nombres=Nombres;
        this.Apellidos=Apellidos;
        this.edad=edad;
        this.sexo=sexo;
        this.telefono=telefono;
        this.celular=celular;
        this.domicilio=dir;
        this.estado=estado;
        this.cédula=cedula;
    }
    public int getId(){
        return this.IDCliente;
    }

    public String getNoms(){
        return this.Nombres;
    }
    public String getApe(){
        return this.Apellidos;
    }
    public int getEdad(){
        return this.edad;
    }
    public String getSexo(){
        return this.sexo;
    }
    public String getTelf(){
        return this.telefono;
    }
```

```
    public String getCel(){
        return this.celular;
    }

    public String getDir(){
        return this.domicilio;
    }
    public int getEstado(){
        return this.estado;
    }
    public String getCedula(){
        return this.cedula;
    }


}

/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package BM;
import java.util.*;

/**
 *
 * @author edu
 */
public class Deuda {

    private int IDDeuda,IDCliente,cancelado,estado,confirmacion;
    private double monto;
    private String descripcion;
    private Date FechaPago;

    public Deuda(){}

    public Deuda(int IDDeuda,int IDCliente,int cancelado,int
estado,double monto,String descripcion,
            Date FechaPago,int confirmacion){

        this.IDDeuda=IDDeuda;
        this.IDCliente=IDCliente;
        this.cancelado=cancelado;
        this.estado=estado;
        this.monto=monto;
        this.descripcion=descripcion;
        this.FechaPago=FechaPago;
        this.confirmacion=confirmacion;
```

```
    }
    public void setCampos(int IDDeuda,int IDCliente,int
cancelado,int estado,double monto, Date FechaPago,int
confirmacion,String desc){

        this.IDDeuda=IDDeuda;
        this.IDCliente=IDCliente;
        this.cancelado=cancelado;
        this.estado=estado;
        this.monto=monto;
        this.FechaPago=FechaPago;
        this.confirmacion=confirmacion;
        this.descripcion=desc;

    }
    public int getIDDeuda(){ return this.IDDeuda;}
    public int getIDCliente(){ return this.IDCliente;}
    public double getMonto(){ return this.monto;}
    public String getDescripcion(){ return this.descripcion;}
    public int getCancelado(){ return this.cancelado;}
    public int getConfirmacion(){ return this.confirmacion;}
    public Date getFechaPago(){ return this.FechaPago;}
  }
```

A continuación se muestran los pasos que sigue el sistema para registrar una deuda a un cliente.

1. Interfaz gráfica: Nombre de la ventana: VentanaCrearDeuda

```
/*
 * VentanaCrearDeuda.java
 *
 * Created on 16 de enero de 2010, 18:41
 */

package UI;
import BM.*;
import java.util.*;
import javax.swing.JOptionPane;

/**
 *
 * @author  edu
 */
```

```
public class VentanaCrearDeuda extends javax.swing.JFrame {

    /** Creates new form VentanaCrearDeuda */
    public VentanaCrearDeuda() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this
method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        jSeparator1 = new javax.swing.JSeparator();
        jLabel2 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        Cedula = new javax.swing.JTextField();
        Cantidad = new javax.swing.JTextField();
        jScrollPane1 = new javax.swing.JScrollPane();
        Descripcion = new javax.swing.JTextArea();
        Crear = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jLabel9 = new javax.swing.JLabel();
        C_Dia = new javax.swing.JComboBox();
        C_Mes = new javax.swing.JComboBox();
        C_Año = new javax.swing.JComboBox();

        setTitle("Crear Deuda");
        setBounds(new java.awt.Rectangle(280, 260, 0, 0));

        jLabel1.setText("Crear Deuda");

        jLabel2.setText("Cédula del Cliente: ");

        jLabel4.setText("Cantidad: ");

        jLabel5.setText("Fecha de Pago: ");

        jLabel6.setText("Descripcion: ");

        Descripcion.setColumns(20);
```

```
        Descripcion.setRows(5);
        jScrollPane1.setViewportView(Descripcion);

        Crear.setText("Crear");
        Crear.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                CrearActionPerformed(evt);
            }
        });

        jButton2.setText("Borrar");
        jButton2.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        jLabel7.setText("Dia");

        jLabel8.setText("Mes");

        jLabel9.setText("Año");

        C_Dia.setModel(new javax.swing.DefaultComboBoxModel(new
String[] { "1", "2", "3", "4", "5", "6", "7", "8", "9", "10",
"11", "12", "13", "14", "15", "16", "17", "18", "19", "20", "21",
"22", "23", "24", "25", "26", "27", "28", "29", "30", "31" }));

        C_Mes.setModel(new javax.swing.DefaultComboBoxModel(new
String[] { "1", "2", "3", "4", "5", "6", "7", "8", "9", "10",
"11", "12" }));

        C_Año.setModel(new javax.swing.DefaultComboBoxModel(new
String[] { "2010", "2011", "2012", "2013", "2014", "2015",
"2016", "2017", "2018", "2019", "2020" }));

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addComponent(jSeparator1,
javax.swing.GroupLayout.DEFAULT_SIZE, 400, Short.MAX_VALUE)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
```

```
                .addContainerGap(169, Short.MAX_VALUE)
                .addComponent(jLabel1)
                .addGap(164, 164, 164))
            .addGroup(layout.createSequentialGroup()
                .addGap(39, 39, 39)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)
                            .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED, 85, javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(Crear)
                            .addGap(29, 29, 29)
                            .addComponent(jButton2)
                            .addGap(84, 84, 84)))
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                            .addComponent(jLabel4)
                            .addComponent(jLabel5)
                            .addComponent(jLabel2))
                        .addGap(25, 25, 25)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                            .addComponent(Cedula,
javax.swing.GroupLayout.PREFERRED_SIZE, 118,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(Cantidad,
javax.swing.GroupLayout.PREFERRED_SIZE, 118,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(layout.createSequentialGroup()
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                                    .addComponent(jLabel7)
                                    .addComponent(C_Dia,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGap(31, 31, 31)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                                    .addComponent(jLabel8)
                                    .addComponent(C_Mes,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGap(31, 31, 31)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                                    .addComponent(C_Año,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(jLabel9))))))
                .addContainerGap(31, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)
                .addComponent(jSeparator1,
javax.swing.GroupLayout.PREFERRED_SIZE, 10,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.BASELINE)
                    .addComponent(jLabel2)
                    .addComponent(Cedula,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(30, 30, 30)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.BASELINE)
                    .addComponent(jLabel4)
                    .addComponent(Cantidad,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                    .addComponent(jLabel5)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.BASELINE)
                        .addComponent(C_Dia,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(C_Mes,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(C_Año,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                    .addComponent(jLabel7)
                    .addComponent(jLabel9)
                    .addComponent(jLabel8))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED, 22, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                    .addComponent(jLabel6)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.BASELINE)
                    .addComponent(Crear)
                    .addComponent(jButton2))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

private void jButton2ActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_jButton2ActionPerformed
// TODO add your handling code here:
}//GEN-LAST:event_jButton2ActionPerformed

private void CrearActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_CrearActionPerformed
// TODO add your handling code here:

    if(TodosCamposCorrectos())
    {
        BMClientes bmc = new BMClientes();
        BMDeudas bmd = new BMDeudas();
        Date fechaPago =
bmd.parseFecha((String)C_Dia.getSelectedItem(),(String)C_Mes.getS
electedItem()

,(String)C_Año.getSelectedItem());
        try
        {
            Cliente cli = bmc.BMconsultarxCI(Cédula.getText());

            //el IDDeuda se genera en la base, cancelado=0 es
decir no se ha pagado, y el estado es 1
            Deuda d = new
Deuda(1,cli.getId(),0,1,Double.parseDouble(Cantidad.getText()),De
scripcion.getText(),fechaPago,0);

            try
            {
                bmd.Registrar(d);
                JOptionPane.showMessageDialog(this,"Deuda
registrada exitosamente");
                this.setVisible(false);
            }
```

```
            catch(Exception e2)
            {
JOptionPane.showMessageDialog(this,e2.getMessage());}

        }//si entra a este catch es xq el cliente no esta
registrado o hubo algun error en la base
        catch(Exception e1)
        {  JOptionPane.showMessageDialog(this,"Cliente no
registrado");  }

    }


}

public boolean TodosCamposCorrectos(){

    int errores=0;


        if( (Validaciones.IngresoVacio(Cedula.getText())) ||
(!Validaciones.ingresoIntcorrecto(Cédula.getText()))
                || Cédula.getText().length()>10)
        {
            JOptionPane.showMessageDialog(this,"Campo cédula mal
ingresado");
            Cédula.selectAll();
            errores++;
        }
        if( (Validaciones.IngresoVacio(Cantidad.getText())) ||
(!Validaciones.DoubleCorrecto(Cantidad.getText())) )
        {
            JOptionPane.showMessageDialog(this,"Cantidad mal
ingresada, se permiten maximo dos decimales y cinco enteros");
            Cantidad.selectAll();
            errores++;
        }
        if( (Validaciones.IngresoVacio(Descripcion.getText())) )
        {
            JOptionPane.showMessageDialog(this,"Descripcion mal
ingresada");
            Descripcion.selectAll();
            errores++;
        }
        if(
!Validaciones.DiaCorrecto((String)C_Dia.getSelectedItem(),(String
)C_Mes.getSelectedItem() ))
        {
            JOptionPane.showMessageDialog(this,"Mal ingreso de la
fecha");
            errores++;
```

```
        }
    if (errores>0) return false;
    else return true;


}//GEN-LAST:event_CrearActionPerformed

    /**
    * @param args the command line arguments
    */
    /*public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new VentanaCrearDeuda().setVisible(true);
            }
        });
    }*/

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JComboBox C_Año;
    private javax.swing.JComboBox C_Dia;
    private javax.swing.JComboBox C_Mes;
    private javax.swing.JTextField Cantidad;
    private javax.swing.JTextField Cedula;
    private javax.swing.JButton Crear;
    private javax.swing.JTextArea Descripcion;
    private javax.swing.JButton jButton2;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JSeparator jSeparator1;
    // End of variables declaration//GEN-END:variables

}
```

Para poder registrar una deuda vemos que primero se valida si todos los campos ingresado fueron correctos mediante la función TodosCamposCorrectos, si esto no es válido entonces se muestra un mensaje de error que indica el campo que fue mal ingresado, de lo contrario se procede a verificar si la cédula ingresada está registrada en el sistema,

sino, se muestra un mensaje de error que indica que la cédula no está registrada, de lo contrario se procede a registrar la deuda con el identificador del cliente que se ingresó su cédula. Esta ventana hace un llamado a la función Registrar del modelo de negocios. En caso de que se haya producido alguna excepción con el sistema, se muestra en pantalla.

2. Modelo de negocios: Registrar

```
public void Registrar(Deuda d) throws Exception
    {
        DBDeuda dbd = new DBDeuda();
        try{ dbd.Registrar(d);}
        catch(Exception e){ throw new Exception(e.getMessage());}

    }
```

Esta función crea una instancia de la capa que trabaja sobre la base de datos y utiliza una de sus funciones, en este caso, la función Registrar.

3. Capa de la base de datos

```
public void Registrar(Deuda d)throws Exception
    {
        Conexion cnx = new Conexion();
        PreparedStatement ps=null;
        SimpleDateFormat formato = new SimpleDateFormat("yyyy-MM-
dd");

        try{con=cnx.conectar();}
        catch(Exception e){throw new Exception(e.getMessage());}

        if(con!=null)
        {

            try
            {
                        String query="insert into Deudas
(IDCliente,monto,FechaPago,descripcion,cancelado,estado,Confirmac
ion) values
("+d.getIDCliente()+","+d.getMonto()+",'"+formato.format(d.getFec
haPago())+"','"+d.getDescripcion()+"',0,1,0)";
                        ps = con.prepareStatement(query);
                        ps.executeUpdate();
```

```
                ps.close();
            }
            catch(SQLException o){
                throw new Exception(o.getSQLState());

            }
            finally{ con.close();}
        }
    }
```

Este procedimiento registra una deuda en la base de datos con toda la información que viene desde la interfaz gráfica que fue ingresada por el administrador del sistema. Vemos que este procedimiento puede arrojar una excepción la cual se propaga hasta la interfaz gráfica y le muestra al usuario el mensaje de error que se produjo o un mensaje de éxito.

GLOSARIO

AGI.- AGI es una interfaz de gateway para el Asterisk similar al CGI usado por los servidores Web. Este permite el uso de diferentes lenguajes de programación como el Perl, PHP, C a su elección. AGI es llamado a partir del plan de discado definido en extensions.conf. La aplicación principal para el AGI es URA con acceso a banco de datos.

Asterisk.- Es software código abierto en su totalidad, liberado bajo licencia GPL. Soporta todas las funcionalidades de las centralitas tradicionales/IP.

Analógico.- Se refiere a las magnitudes o valores que varían con el tiempo en forma continua (distancia, temperatura, velocidad, voltaje, frecuencia, amplitud, etc.) y pueden representarse en forma de ondas. Las computadoras emplean lo digital y, por lo tanto, si entra información analógica, se debe convertir; este es el caso de la conexión a Internet por Dial up, donde un módem convierte la señal analógica (el sonido) en digital.

CDMA.- (Code didivison Multiple Access) Acceso Múltiple de División de Código. Norma de transferencia de información por teléfonos inalámbricos.

CLI.- (Línea de comandos, Intérprete de comandos, Terminal, Consola, Shell, CLI, Command line interface). Tipo de interfaz para manipular un programa o sistema operativo con instrucciones escritas. Cada instrucción es escrita en una línea de texto y suelen ejecutarse al presionar ENTER. También se suele permitir archivos scripts para la ejecución automática de varias líneas de comandos que cumplen alguna función. Si bien se siguen usando en algunos ámbitos, este tipo de interfaz ha evolucionado hacia los GUI, que son interfaces gráficas. Sistemas operativos de la familia DOS son ejemplos de uso de interfaz CLI

CentOS.- Community ENTerprise Operating System. CentOS es una distribución de Linux gratuita que está basada en la distribución Red Hat Enterprise Linux (RHEL). CentOS es muy similar al RHEL, pero gratuito, aunque no es mantenido por Red Hat.

Digital.- Cualquier señal o modo de transmisión que utiliza valores discretos en lugar de un espectro continúo de valores (como las señales analógicas). Los valores pueden medir voltaje, frecuencia, amplitud, ubicación, etc. En informática se suele utilizar el sistema digital de unos y ceros (sistema binario) para transmitir, procesar o almacenar información. Por ejemplo, el reloj del microprocesador trabaja en dos voltajes distintos, cada uno

representa un uno o un cero. Con la combinación de unos y ceros se puede procesar todo tipo de información.

Digium.- Digium, empresa que promueve el Asterisk, invierte en ambos aspectos, el desenvolvimiento de código fuente y en hardware de telefonía de bajo costo que funciona con Asterisk.

E1.- E1 es un canal especial de alta velocidad de comunicación. Es de 32 canales: 2 canales de control y 30 canales de transmisión, tiene 64 Kbps por canal.E1=2.048 Mbps.

FXO.- Los dispositivos FXO permiten manejar una línea telefónica externa, los dispositivos FXO solo "reciben" tono de marcado, es decir, son lo mismo que los teléfonos análogos, necesitan de una línea (FXS) que les provea tono para funcionar. En los jerga de los PBX se conocen como las entradas de las troncales.

FXS.- los dispositivos FXS los cuales permiten simular el comportamiento de una línea telefónica (voltaje, corriente, timbres…) a estos dispositivos se conectan dispositivos FXO como los teléfonos convencionales o maquinas de fax. En la jerga de los PBX están son las salidas de extensión.

Hardware.- En computación, término inglés que hace referencia a cualquier componente físico tecnológico, que trabaja o interactúa de algún modo con la computadora. No sólo incluye elementos internos como el disco duro, CD-ROM, disquetera, sino que también hace referencia al cableado, circuitos, gabinete, etc. E incluso hace referencia a elementos externos como la impresora, el mouse, el teclado, el monitor y demás periféricos. El hardware contrasta con el software, que es intangible y le da lógica al hardware (además de ejecutarse dentro de éste). El hardware no es frecuentemente cambiado, en tanto el software puede ser creado, borrado y modificado sencillamente. (Excepto el firmware, que es un tipo de software que raramente es alterado).

Internet.- Conocida como la red de redes, pues se trata de una de las redes más grandes con un estimado de mil cien millones de usuarios (2007). Para funcionar utiliza el conjunto de protocolos TCP/IP. Desde que fue creada la WWW, el número de usuarios no paró de crecer; pero ese no es el único servicio de internet: podemos acceder remotamente a otras máquinas (telnet y SSH), transferir archivos (FTP), conversar con personas (chat y mensajeros), servicio de correo electrónico (email), grupos de noticias, etc.

IP.- Protocolo para la comunicación en una red a través de paquetes conmutados, es principalmente usado en Internet. Los datos se envían en bloques conocidos como paquetes (datagramas) de un determinado tamaño (MTU). El envío es no fiable (conocido también como best effort o mejor esfuerzo); se llama así porque el protocolo IP no garantiza si un paquete alcanza o no su destino correctamente. Un paquete puede llegar dañado, repetido, en otro orden o no llegar. Para la fiabilidad se utiliza el protocolo TCP de la capa de transporte. Los paquetes poseen una cabecera con información sobre la máquina de origen y la de destino (sus direcciones IP), con esta información los enrutadores determinan por dónde enviar la información. Cada paquete de un mismo archivo puede enviarse por diferentes rutas dependiendo de la congestión del momento.

Java.- Lenguaje de programación orientado a objetos. Fue desarrollado por James Gosling y sus compañeros de Sun Microsystems al principio de la década de los 90.La programación en Java es compilada en bytecode, el cuál es ejecutado por la máquina virtual Java. Usualmente se usa un compilador JIT. El lenguaje es parecido a C y C++, aunque su modelo de objetos es más sencillo, y fue influenciado también por Smalltalk, y Eiffel.

Kernel.- Núcleo. Parte esencial de un sistema operativo que provee los servicios más básicos del sistema. Se encarga de gestionar los recursos como el acceso seguro al hardware de la computadora. Se encarga también del multiplexado, determinando qué programa accederá a un determinado hardware si dos o más quieren usarlo al mismo tiempo. El kernel también ofrece una serie de abstracciones del hardware para que los programadores no tengan que acceder directamente al hardware, proceso que puede ser complicado.

Linux.- Sistema operativo que posee un núcleo del mismo nombre. El código fuente es abierto, por lo tanto, está disponible para que cualquier persona pueda estudiarlo, usarlo, modificarlo y redistribuirlo. El término Linux se utiliza para describir al sistema operativo tipo Unix que utiliza filosofías y metodologías libres y que está constituido por la combinación del núcleo Linux con las bibliotecas y herramientas del proyecto GNU, además de otros proyectos libres y no libres. El término Linux también hace referencia al kernel que utilizan múltiples sistemas operativos. Es ampliamente popular en el mercado de servidores.

Mail.- El correo electrónico (también conocido como e-mail, un término inglés derivado de electronic mail) es un servicio que permite el intercambio de mensajes a través de sistemas de comunicación electrónicos. El concepto se utiliza principalmente para denominar al sistema que brinda este servicio vía Internet mediante el protocolo SMTP (Simple Mail Transfer Protocol), pero también permite nombrar a otros sistemas similares que utilicen distintas tecnologías. Los mensajes de correo electrónico posibilitan el envío, además de texto, de cualquier tipo de documento digital (imágenes, videos, audios, etc.).

MD5.- (MD que significa Message Digest; en castellano, Resumen de mensaje). Desarrollado por Rivest en 1991, el MD5 crea, a partir de un texto cuyo tamaño es elegido al azar, una huella digital de 128 bits procesándola en bloques de 512 bits. Es común observar documentos descargados de Internet que vienen acompañados por archivos MD5: este es el hash del documento que hace posible verificar su integridad.

MP3.- Formato de audio que combina gran calidad de sonido y poco tamaño. Desarrollado en Alemania por Brandenburg, Popp y Grill, tres científicos del instituto tecnológico de Fraunhofer en Ilemenau en el año 1986. Luego en 1992 la Moving Picture Experts Group (MPEG) aprobó oficialmente la tecnología. El formato MP3 redujo el tamaño de los archivos de música conocidos hasta diez veces casi sin perder calidad por la compresión. Su nombre técnico es ISO MPEG Audio Layer 3.

Mysql.- MySQL es un sistema de gestión de bases de datos (SGBD) multiusuario, multiplataforma y de código abierto. MySQL pertenece a la compañía sueca MySQL AB, a la que le pertenece casi todos los derechos del código fuente. La compañía desarrolla y mantiene el sistema, vendiendo soporte y servicios, como también las licencias para usar MySQL.

ODBC.- En informática, el ODBC (Open Database Connectivity) es un estándar de acceso a bases de datos, que permite mantener independencia entre los lenguajes de programación, los sistemas de bases de datos (las bases de datos y su software gestor), y los sistemas operativos. ODBC inserta una "capa" entre la aplicación y el SGBD (sistema gestor de base de datos). Esta capa es llamada "manejador de base de datos". El objetivo de la capa es traducir las consultas a la base de datos (u otras acciones) por parte de la aplicación a una consulta que el SGBD comprenda. Para que esto sea

posible, tanto la aplicación como el SGBD deben ser compatibles con ODBC.ODBC es desarrollado por Microsoft Corporation.

Open source.- Denominación para aquellas aplicaciones que tienen su código fuente liberado. En general, los programas de código abierto suele ser libres. Aunque existen aplicaciones de código abierto que no son libres. Open Source es utilizado también para hacer referencia a un nuevo movimiento de software, la Open Source Initative.

PBX.- (Private Branch Exchange) es una central telefónica privada. Permite interconectar los teléfonos internos de una empresa. Selecciona la línea saliente de forma (más o menos) Automática. Algunos permiten transferir llamadas, realizar conferencias, llevar un control de los números marcados. Es usado por empresas grandes y pequeñas para reducir costos. Hoy en día muchos PBX soportan el protocolo IP.

PHP.- (PHP Hypertext Pre-processor). Lenguaje de programación usado generalmente en la creación de contenidos para sitios web. Es un lenguaje interpretado especialmente usado para crear contenido dinámico web y aplicaciones para servidores, aunque también es posible crear aplicaciones gráficas utilizando la biblioteca GTK+. Generalmente los scripts en PHP se embeben en otros códigos como HTML, ampliando las posibilidades del diseñador de páginas web enormemente. La interpretación y ejecución de los scripts PHP se hacen en el servidor, el cliente (un navegador que pide una página web) sólo recibe el resultado de la ejecución y jamás ve el código PHP. Permite la conexión a todo tipo de servidores de base de datos como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite. PHP es una alternativa a otros sistemas como el ASP.NET/C#/VB.NET de Microsoft o a ColdFusion de Macromedia, a JSP/Java de Sun Microsystems, y a CGI/Perl. La ventaja con los de Microsoft o Macromedia es que es totalmente gratuito, no hay que pagar licencias.

PHP-AGI.- Librería que encapsula las funciones de asterisk para que sea mas fácil desarrollar programas en asterisk para php.

PSTN.- (Public switched telephone network o PSTN). Red telefónica para la transferencia de voz y datos. No es efectiva para la transmisión de datos.

RSA.- Tipo de certificado digital para su uso en transmisiones seguras por SSL, especialmente para la protección de sitios con acceso por HTTPS. Es emitida por VeriSign.

Script.- Grupo de lenguajes de programación que son típicamente interpretados y pueden ser tipiados directamente desde el teclado. Los scripts son un conjunto de instrucciones generalmente almacenadas en un archivo de texto que deben ser interpretados línea a línea en tiempo real para su ejecución, se distinguen de los programas, pues deben ser convertidos a un archivo binario ejecutable para correrlos. Los scripts pueden estar embebidos en otro lenguaje para aumentar las funcionalidades de este, como es el caso los scripts PHP o Javascript en código HTML.

Servidor.- En redes, computadora central en un sistema de red que provee servicios a otras computadoras. En internet, los servidores son los proveedores de todos sus servicios, incluyendo la WWW (las páginas web), el FTP, el correo electrónico, los grupos de noticias, etc. Básicamente, una computadora conectada a internet emplea una dirección (dirección web, dirección IP, dirección FTP, etc.) para poder comunicarse con el servidor al que le corresponde. La computadora envía (utilizando el protocolo adecuado) las distintas solicitudes al servidor, y el servidor responde (empleando el protocolo adecuado) las solicitudes. El servidor también puede solicitar datos de la computadora, y la computadora le responde.

Softphone.- software que realiza una simulación de teléfono en una computadora, permitiendo así la comunicación con otras computadoras que posean este mismo software, usando un VSP (VoIP Service Provider Proveedor de Servicios de VoIP, carrier voip, etc). Lo más comunes son: Skype, WengoPhone, Gizmo, XLite, el Aim, SjPhone, y últimamente Zoiper.

T1.- (T-carrier, T-CX). En telecomunicaciones, la portadora-T es la designación de un sistema genérico de telecomunicaciones para los sistemas digitales multiplexados. Fueron originalmente desarrollados por los Laboratorios Bell y utilizados en Estados Unidos y Japón. La unidad básica del sistema de portadoras-T es el DS0 que tiene una velocidad de transmisión de 64 kbit/s y es normalmente usado para un circuito de voz. El sistema de Portadoras–E, o sistema europeo de portadoras, es incompatible con las Portadoras-T y se utiliza en el todo el mundo excepto en Japón y los Estados Unidos.

Telefonía IP.- Hace referencia a comunicaciones telefónicas realizadas a través de redes TCP/IP. A diferencia de PSTN, que se compone de señales analógicas y digitales a través de una red con conmutación de circuitos, la

telefonía IP utiliza conmutación de paquetes. Toda la información que se va a transmitir a través de la red se divide en paquetes de datos.

Voz sobre IP.- Dice que es la tecnología que posibilita el uso de redes IPs como medio de transmisión de voz. El concepto es simple y consiste en convertir los paquetes de voz, analógicos, en paquetes digitales y hacerlos transitar por internet. Con un relativo ancho de banda (128Kbps ya se garantiza una buena calidad de voz, siendo de 256Kbps en adelante mejor) es posible usar la capacidad de transporte del protocolo IP para transmitir voz.

X86.- Nombre dado al grupo de microprocesadores de la familia de Intel y a la arquitectura que comparten estos procesadores. Existen X86 tanto de 16 bits como de 32 bits, estos últimos llamados IA-32 (x86-32). Los microprocesadores x86 fueron 8086, 80286, 80386, 80486. Luego continuó con el Pentium. Su sucesor inmediato es el x86-64, creado por AMD (con el nombre de AMD64), y luego Intel creó su propia versión llamada Intel 64. Ambas implementan los 64 bits y son muy similares entre sí.

Zoiper.- Véase softphone.

BIBLIOGRAFÍA

1. **Matthew Asham,**Pagina oficial de la clase phpagi, http://phpagi.sourceforge.net, 2004.
2. **asteriskguide.com**, Integración de Asterisk usando AGI y AMI, http://www.asteriskguide.com/mediawiki/index.php/Integraci%C3%B3n_de_Asterisk_usando_AGI_y_ AMI, 2010.
3. **Loris Santamaria**, Introducción a Asterisk y la telefonía IP, http://www.solucionesit.com.ve/tisol/files/PresentacionAsterisk.pdf, 2010.
4. **Portal de informática**, internet, tecnologías y web, http://www.alegsa.com.ar/, 2009.
5. **asteriskguide.com**, Introducción al Asterisk, http://www.asteriskguide.com/ mediawiki/index.php/Introducc%C3%ADon_al_Asterisk, 2009.
6. **Universidad de Quevedo**, Sistemas de Telecomunicaciones, http://www.uteq.edu.ec/facultades/empresariales/informatica/tutoriales/siscomunicacionesii2004/materia/e1yt1.htm, 2004.
7. **Proyecto de grado profesional de un estudiante de la UNAD**, Diferencia entre puertos FXS y FXO, http://proyectopica.wordpress.com/2008/11/27/diferencia-entre-puertos-fxs-y-fxo/, 2008.
8. **Kioskea**, Firmas electrónicas, http://es.kioskea.net/contents/crypto/signature.php3, 2010.
9. **Entender VoIP (Voz sobre IP**), Conceptos de Voz sobre IP, http://www.informatica-hoy.com.ar/voz-ip-voip/Entender-VoIP-Voz-sobre-IP.php, 2010.
10. **VOIP-Info.org**, Asterisk AGI, http://www.voip-info.org/wiki/view/Asterisk+AGI, 2003.
11. **Wikipedia**, Asterisk, http://es.wikipedia.org/wiki/Asterisk, 2010.
12. **asterisk_agi[VoIP en español**], Asterisk AGI, http://voip.megawan.com.ar/doku.php/asterisk_agi, 2006.
13. **Van Meggelen J., Smith J. y Madsen L.,** Conceptos de Asterisk, Asterisk The Future of Telephony, Editorial O'Relly Media, 2005.