Ostfalia
Hochschule für angewandte Wissenschaften

Faculty of Electrical Engineering

Resumen del Proyecto Final

"Conception and prototypical Implementation of a CTI application exemplified on Zarafa Groupware and SIPFoundry's SipXecs"

Author: Daniel Peinado López

Tutor: Diederich Wermser and Daniel Hartmann

Cotutor: Iván Vidal Fernández

Wolfenbüttel, February 27th, 2012

Resumen

Resumen: Este proyecto consiste en la integración de la telefonía en los ordenadores, conocido con las siglas CTI. El objetivo principal es integrar un groupware con un sistema PBX Voice-IP y desarrollar una aplicación "click to dial" para Zarafa Collaboration Platform (ZCP), que es un software colaborativo de código abierto. Tendré un cliente de correo (Zarafa Webaccess) y que crearé un plugin para hacer posible las llamadas entre contactos con un sólo click. Usaré un servidor SipXecs, que es un servidor de telefonía IP de código abierto, el cual funciona como centralita (PBX) y es el responsable de gestionar las llamadas entre los usuarios. Cada usuario contará con un telefono IP y todo esto se probará en una red local. De esta manera podré conectar múltiples usuarios en una red de manera gratuita, como por ejemplo conectar varias oficinas con centralitas y telefonos IP dentro de una misma compañía.

Palabras Clave: Zarafa Collaboration Platform, Zarafa Webaccess, PBX, CTI, Representational State Transfer (REST), voice IP, SipXecs, PHP, Groupware.

Nota: Todos los capítulos contienen información adicional en la primera versión del proyecto en inglés. Este documento es un resumen de dicho proyecto. Si desea ver alguno de los capítulos de forma más exhautiva, dirijase al mismo en la versión en inglés.

Indice

INDEX OF IMAGES

1.INTRODUCCIÓN Y OBJETIVOS

1.1 Introducción

El papel cada vez más fuerte de Internet en los negocios y la vida cotidiana, y las actuales líneas de desarrollo tecnológico en las telecomunicaciones, donde hay una fuerte tendencia hace el "todo IP", hace que el desarrollo de tecnologías basadas en IP permitan estas comunicaciones y servicios de voz.

Este proyecto consiste en la integración de la telefonía en los ordenadores (Computer Telephony Integration) y este nombre (CTI) es utilizado para cualquier tecnología que permita integrar un telefono en una computadora y puedan éstos puedan ser coordinados e interactuar.

La tecnología CTI intent integrar todos los canales de comunicación de una empresa, como por ejemplo el telefono, el correo electrónico, la web, chat, el fax, los SMS, etc.

La idea de este proyecto es integrar un Groupware (Grupo de usuarios) con una centralita IP (PBX), y para mostrar esta integración utilizaremos un ejemplo en particular. Integraré el "Zarafa Groupware" con una centralita "SipXecs-PBX" y desarrollaré un software para realizer una demostración de esta integración.

El software sera una aplicación "click to call", esto es una aplicación con la cual los usuarios pueden realizar llamadas telefónicas a través de su cliente de correo (en este caso Zarafa Webaccess) haciendo un simple click en el contacto al que desean llamar. Registraremos a nuestros usuarios en una base de datos MySQL y configuraremos el servidor de Zarafa.

Un Groupware puede ser utilizado para comunicar, cooperar y coordinar. Algunos beneficios que proporciona esta implementación son:

- El Groupware estimula la cooperación dentro de una organización y ayuda a la comunicación entre las personas y la colaboración en proyectos comunes.
- Groupware ayuda a definir el flujo de documentos y definir el trabajo por hacer para completar un proyecto.
- El Groupware proporciona a los usuarios una forma única de compartir información.

Con este proyecto realizaré un prototipo para que dos usuarios que utilicen el Zarafa Groupware puedan realizar llamadas sin tener que marcar un número de teléfono; simplemente haciendo click en el contacto al que desean llamar.

Click to call, también conocido como click-to-talk, click-to-chat, click-to-text o click-to-dial, es una forma de comunicación basada en la Web, en la que la persona hace click en un objeto (botón, imagen o texto) para solicitar una conexión inmediata con otra persona en tiempo real, ya sea por llamada telefónica, voz sobre ip o texto. Una de las grandes ventajas del VoIP es que las comunicaciones internas son más rápidas y gratuitas.

1.2 Objetivos

Las empresas se vuelven cada vez más y más competitivas y adoptan más estrategias para asegurarse el éxito. Con proyectos como este, nos aseguraremos de que la telefonía es una de las partes del proceso de optimización y ahorro de las empresas.

El objetivo principal del proyecto es ahorrar y optimizar los recursos para los empleados de las empresas, consiguiendo reducir el tiempo que un empleado necesita para realizar una llamada; de esta forma no necesita recordar y marcar un número de telefono, simplemente selecciona la persona a la que desea llamar y todo sucede automáticamente.

Otros objetivos secundarios serían:

- Cambiar la forma en que las centralitas de las empresas se relacionan con los clientes.
- Hacer la vida más fácil para el departamento de ventas de una empresa, de manera que tenga conexión rápida y fácil con todos sus clientes.
- Aumentar las ventas de una empresa.

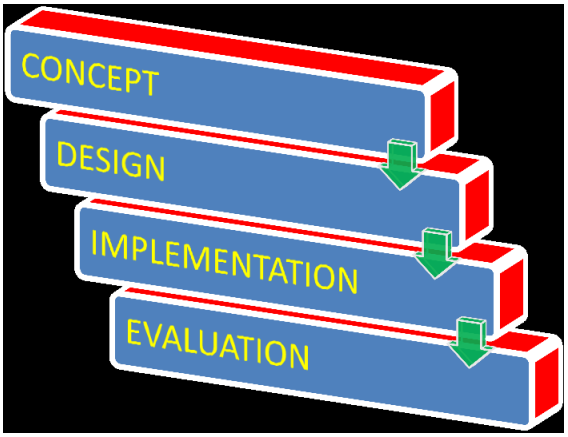
Actualmente la mayoría de las aplicaciones CTI que se diseñan, van dirigidas a los "Call Centres" o centros de llamadas. Entendemos por un "Call Centre" un centro de atención al cliente al que se accede en un principio de manera telefónica. Otro caso podría ser que el "centro de llamadas" realice llamadas a clientes para ofrecer ciertos servicios o información, en lugar de ser un centro de atención al cliente que reciba llamadas.

Cada vez más empresas ofrecen servicios de telefonía, o centran su actividad en el uso del teléfono. Podemos considerar distintos puntos de vista.

- Usuarios que quieren una atención rápida.
- El agente que quiere la forma más cómoda de trabajar.
- La empresa, que quiere tener el máximo rendimiento posible.

Los usuarios ahorran tiempo de espera, y su llamada se procesa desde el primer momento al agente adecuado, y en muchos casos, ni siquiera se requiere la intervención del agente, lo que permite el acceso interactivo a bases de datos.

1.3 Fases del proyecto



1. Conceptos
    a. Conceptos teóricos.

2. Diseño
    a. Configurar nuestro sistema y montar el escenario (Hardware).
    b. Diagramas de flujo de nuestro software

3. Implementación
    a. Instalación y configuración del sistema software.
    b. Comprobar que todo funciona.
    c. Desarrollar el código del plugin.
    d. Depuración de errores.

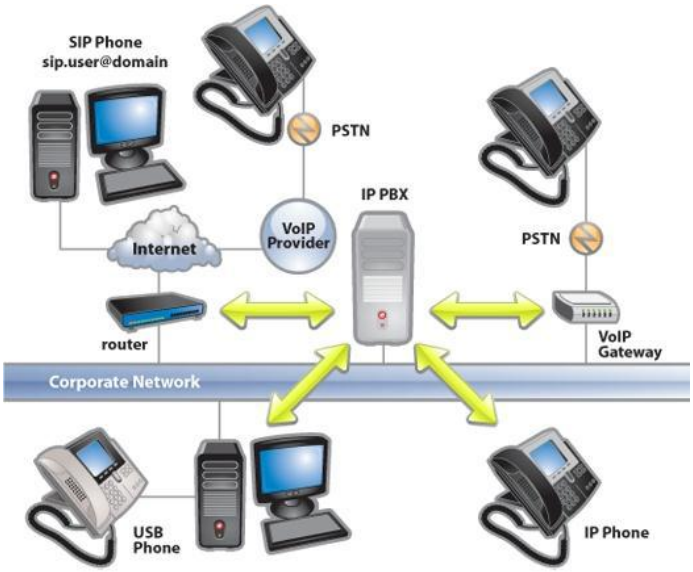
4. Evaluación
    a. Comprobar que todo funciona y cumple las especificaciones

2. Sistemas PBX

2.1 Descripción *

Un 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 salientes con autonomía sobre cualquier otra central telefónica.

PBX hace conexiones entre los telefonos internos de una organización privada (normalmente una empresa) y los telefonos de la red pública conmutada (PSTN) a través de líneas troncales. Una de las ventajas de usar estas centralitas, es el ahorro de costes en llamadas telefónicas internas.



2.2 Historia

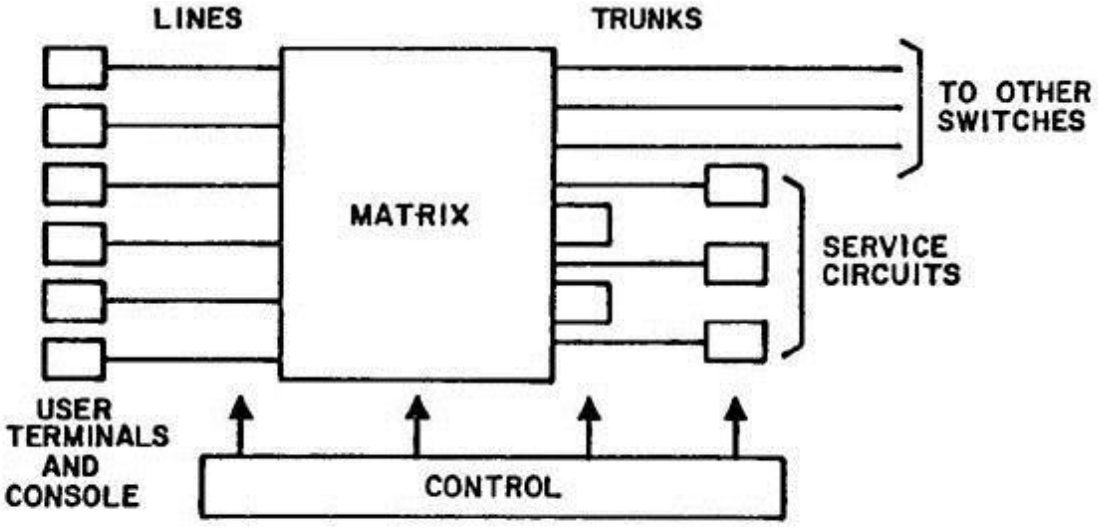
Ver este apartado en el proyecto completo.

2.3 Componentes del sistema

- Red de conmitación interna de la central
- Microcontrolador o microprocesador
- Tarjetas lógicas
- Estaciones
- Telco trunks de salida
- Panel de control
- Sistema de alimentación ininterrumpida
- Conductores de interconexión

2.4 Funcionamiento

Un sistema PBX se compone de cuatro partes básicas: Matriz de conmutación, de control, terminales de usuario y trunks. La propia PBX incluye el control y la conmutación.



basic elements of a PBX.

No importa lo moderna y compleja que sea una PBX, porque al final el funcionamiento a los ojos de un usuario será el mismo.

- Establecimiento de conexiones.
- Mantenimiento de estas conexiones.
- Desconexión.
- Proporciona información, como medición de tiempo de llamada.

2.5 Tendencias actuales

En la actualidad, se están desarrollando programas de software libre, que configuran y mejoran las funciones de las PBX. Es el caso de Asterisk o SipXecs, software que utilizaremos en parte de este proyecto. Con estos sistemas podremos integrar muchas funciones como telefonía, internet, fax, etc, en un único equipo. Asterisk o Sipxecs podrían sustituir completamente a una PBX, ya que este software realizan todas las funciones y mucho más, además sin costes de licencia asociados.

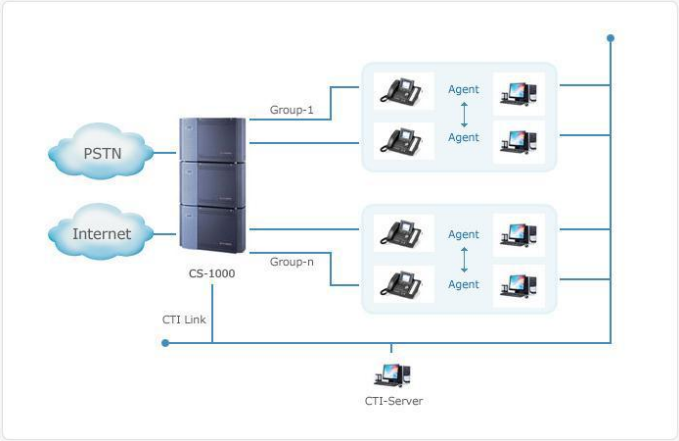
2.6 Ventajas

Aquí tenemos cinco ventajas importantes
1. Permite a los empleados compartir lineas telefónicas del sistema, reduciendo los gastos de telecomunicación en general.
2. El sistema PBX es totalmente programmable y puede soportar complejas instalaciones.
3. Hay una amplia variedad de estandares.
4. El sistema PBX se puede ampliar fácilmente en caso de que su empresa crezca.
5. No se require un espacio físico muy grande para instalar el hub PBX

3. CTI – Computer Telephony Integration

3.1 Definicion

Integración de Telefonía Informática implica la integración de un sistema informático con los recursos de telefonía para aumentar las capacidades de una organización de comunicaciones.



3.2 Componentes de CTI

Los componentes de hardware mas comunes son:

- PBXs
- Modems
- Ordenadores

3.3 Modelos de implementación

- Third Party model: En este modelo, el servidor de telefonía se comunica con los dispositivos electrónicos. Cada ordenador puede conectarse con cada dispositivo si tiene permisos.

- First Party model: Cada ordenador se puede conectar con uno o varios dispositivos y puede acceder solamente a los dispositivos conectados a él.

3.4 Tipos de conexiones

Conexiones telefónicas: Digitales, Analógicas, VoIP

Conexiones de Datos

3.5 Aplicaciones de CTI

Puede ser desde algo sencillo como enviar fax desde un ordenador, o algo más complicado como aplicaciones de reconocimiento de voz. Se pueden realizar llamadas desde el ordenador, utilizando un telefono para transmitir la voz. Por ejemplo, en este proyecto integraré en el cliente de correo de Zarafa la opción de realizar llamadas de telefono y utilizaremos los telefonos para transmitir la voz.

Un ejemplo contrario podría ser que cuando un usuario reciba una llamada y descuelgue el teléfono, se abra una ventana con una base de datos para introducir los datos de un cliente.

Podríamos poner infinidad de ejemplos, ya que existe una larga lista de aplicaciones CTI fáciles de implementar, y que dan un valor mayor a las comunicaciones de nuestra empresa.

4. Voz sobre IP

4.1 Definicion

Voz sobre IP, o VoIP es una familia de tecnologías, metodologías, protocolos de comunicaciones, y técnicas de transmisión para la transmisión de comunicaciones de voz y sesiones multimedia usando redes usando el protocolo IP, como por ejemplo Internet.

4.2 Telefonía IP vs Telefonía convencional

En la telefonía convencional se establece un circuito entre los usuarios, sin embargo en IP se envían paquetes de datos que pueden seguir distintos caminos.

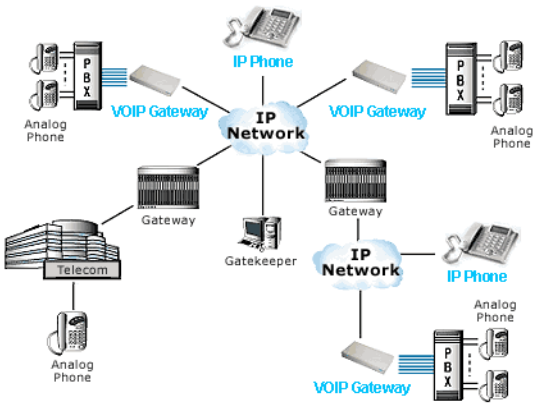
El objetivo de la telefonía IP es traer avances tecnológicos a los usuarios y a las empresas interesadas en reducir costes utilizando internet como plataforma de comunicaciones. El precio de las llamadas por voz IP, es inferior al precio de las llamadas convencionales.

4.3 Elementos

Cliente: Establece y origina las llamadas.

Servidores: Administran y gestionan las operaciones con la base de datos en tiempo real. Colectan información, enrutan, controlan, etc.

Gateways: Pasarelas. Proporcionan conexion entre usuarios. Se utilizan para "terminar" las llamadas. El cliente la origina y la pasarela la finaliza.



4.4 Protocolos

El VoIP/H.323 consiste en una serie de estándares y soporta varios protocolos distintos.

- Direccionamiento: RAS, DNS
- Señalización: Q.931, H.225, H.245
- Compresión de voz: G.711, G.723, G.728, G.729 y G.722
- Transmisión de voz: UDP, RTP
- Control de Transmisión: RTCP

<table>
<tr><td colspan="6">CALL SETUP AND CONTROL</td></tr>
<tr><td colspan="2"></td><td colspan="2">Presentation</td><td colspan="2"></td></tr>
<tr><td colspan="2">Direccionamiento</td><td>Audio Compression G.711 or G.723</td><td>DTMF</td><td>Señalización</td><td>Direcciona-miento</td></tr>
<tr><td>RAS(H.225)</td><td>DNS</td><td>RTP/RTCP</td><td>H.245</td><td>Q.931 (H.225)</td><td>DNS</td></tr>
<tr><td colspan="3">UDP Transport</td><td colspan="3">TCP Transport</td></tr>
<tr><td colspan="6">IP NETWORK</td></tr>
<tr><td colspan="6">DATA LINK</td></tr>
<tr><td colspan="6">PHYSICAL</td></tr>
</table>

5. Representational State Transfer (REST)

5.1 Definición

La Transferencia de Estado Representacional o REST es una técnica de arquitectura de software para sistemas hipermedia distribuidos como la World Wide Web.

El término REST se utiliza en el sentido más amplio para describir cualquier interfaz web simple que utiliza XML y HTTP.

5.2 Diferentes arquitecturas o limitaciones

Cliente-Servidor.

Stateless.

Cacheable.

Layered system.

Code on demand.

Uniform interface.

5.3 Principios de la interfaz

Los sistemas que siguen los principios REST se suelen llamar RESTful. REST dice que la disfruta de escalabilidad gracias a una serie de diseños fundamentales:

- Cliente-Servidor protocolo stateless: Cada mensaje HTTP contiene toda la información necesaria para entender cada petición. Por lo tanto, los servidores no necesitan recordar ningún estado anterior.
- Un conjunto de operaciones bien definidas aplicables a los recursos de información: HTTP define unas operaciones propias (POST, GET, PUT, DELETE) sencillas.
- Una sintaxis universal para identificar recursos: Cada recurso es accesible solamente a través de su URI (Uniform Resource Identifier).
- El uso de hipermedia, tanto para información de la aplicación como para transiciones de estado de la aplicación: Las representaciones de este estado en un sistema REST son típicamente HTML o XML. Como resultado es posible navegar de un recurso REST a otro.

5.4 Objetivos Principales

Los objetivos principales de REST incluyen:

- Escalabilidad de las interacciones de los componentes
- Generalidad de las interfaces
- Despliegue independiente de los componentes
- Componentes intermedios para reducir la latencia, reforzar la seguridad y encapsular los sistemas heredados.

5.5 Elementos de la arquitectura REST

REST DATA ELEMENTS	
DATE ELEMENT	**MODERN WEB EXAMPLES**
Recurso	El objetivo conceptual deseado de una referencia de hipertexto
Identificador de Recurso	URL, URN
Representación	HTML document, JPEG image
Representación de Metadatos	media type, last-modified time
Recurso de Metadatos	source link, alternates, vary
Datos de Control	if-modified-since, cache-control

5.6 Principio Fundamental

La motivación de REST fue capturas las características de la Web, las cuales hicieron que la Web tuviera éxito. Posteriormente, estas características serían utilizadas en la propia evolución de la Web.

Un concepto importante en REST es la existencia de recursos, cada uno de los cuales es referenciado con un identificador global (por ejemplo, URI en HTTP). Para manipular estos recursos, los componentes de la red se comunican a través de una interfaz estandarizada (por ejemplo HTTP) e intercambian representaciones de estos recursos.

Cualquier número de conectores (por ejemplo, clientes, servidores, cachés, tuneles, etc) puede mediar en la petición, pero cada uno lo hace sin ver más allá de su propia solicitud. Por lo tanto, una aplicación puede interactuar con un recurso por saber dos cosas: el identificador y la acción requerida. No tiene por qué saber si existen cachés, proxies, gateways, o cualquier cosa entre #el y el servidor.

5.7 Ventajas y Desventajas

VENTAJAS	DESVENTAJAS
Soporte para redireccionar las diferentes representaciones. Fácil de implementar. No necesitamos herramientas especiales. Escalabilidad probada. Soporte universal y simple desde cualquier lenguaje y plataforma. Integración real de comunicaciones B2B. Funciona con XML, pero también con otros formatos.	Se pierde el tipado fuerte. REST solo conoce el protocol HTTP.

5.8 Formato de comandos REST

El format de un comando WLP REST es:

<protocol>://<host>:<port>/<webapp>/bea/wlp/api/<type>/<action>/<label>?<params>

Command Part	Description
<protocol>	El protocolo de transporte. Tipicamente HTTP.
<host>	La IP del host name, por ejemplo localhost.
<port>	El número de puerto, por ejemplo 7001
<webapp>	El nombre de la aplicacion web donde se hospedan los servicios, como por ejemplo myWebbApp. Tenga en cuenta que todos los comandos REST también requieren de un parámetro webapp. El parámetro webapp especifica la aplicación web específica para operar con el comando. La aplicación web especificada en el parámetro puede ser diferente de la aplicación web que usted publique, siempre que se implemente en el mismo archivo EAR.
bea/wlp/api	Ruta de espacio para nombres estándar que se utiliza para todos los comandos REST
<type>	El tipo de portal con el que el comando opera, como el escritorio, libro, página, shell, menu, diseño, etc.
<action>	La acción específica a realizar, como listar, borrar, añadir, y obtener.
<label>	La etiqueta única del objeto.
command-specific parameters	Una lista de los parámetros de la URL de comandos específicos. Consultar la API de REST para ver una lista completa de cada comando.

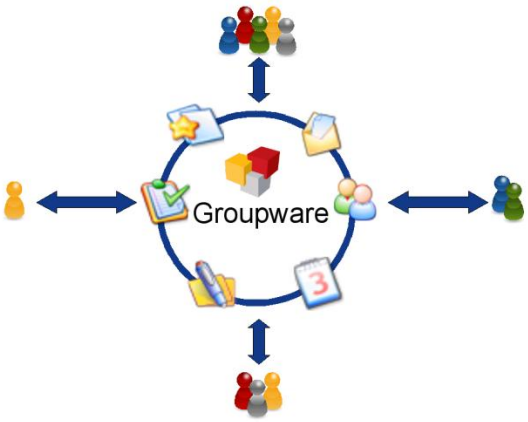
Por ejemplo, el siguiente comando es el que uso en mi proyecto para realizar una llamada de un telefono a otro:

PUT https://myUser:pass@localhost:PORT/sipxconfig/rest/call/number
PUT https://206:0000@test.local:8443/sipxconfig/rest/call/207

6. Groupware

6.1 Definición

El software colaborativo son programas diseñados para ayudar a personas implicadas en tareas comunes a lograr objetivos. Una de las primeras definiciones de Groupware es: Procesos intencionales de grupo, además de programas para apoyarlos.

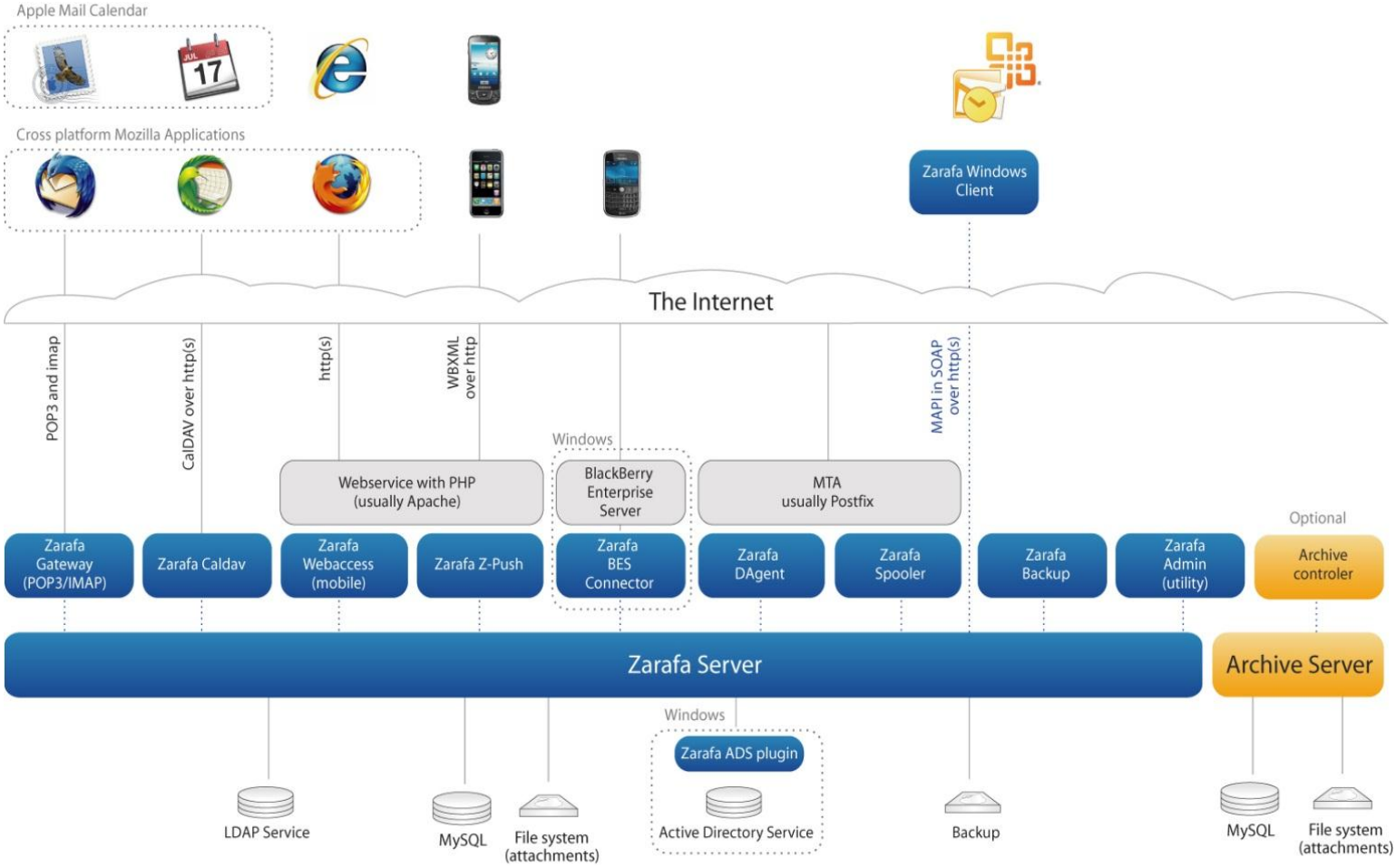


La intención del groupware es transformar la manera de compartir documentos y medios de comunicación para hacer el trabajo en equipo más eficaz.

6.2 Zarafa Collaboration Platform (ZCP)

Zarafa es una solución con todas las funciones de correo electrónico y groupware, enfocada hacia clientes que utilicen el estándar MAPI. Es un software europeo colaborativo. El groupware de Zarafa ofrece almacenamiento de correo en el servidor y trae su propio cliente basado en Ajax denominado WebAccess. En estos momentos está disponible para Linux solamente y es un servidor de correo electrónico de alto rendimiento con avanzadas características. Los usuarios que estén acostumbrados a utilizar MS Outlook serán capaces de utilizar Zarafa WebAccess sin ningún problema.

ARQUITECTURA DEL SERVIDOR ZARAFA



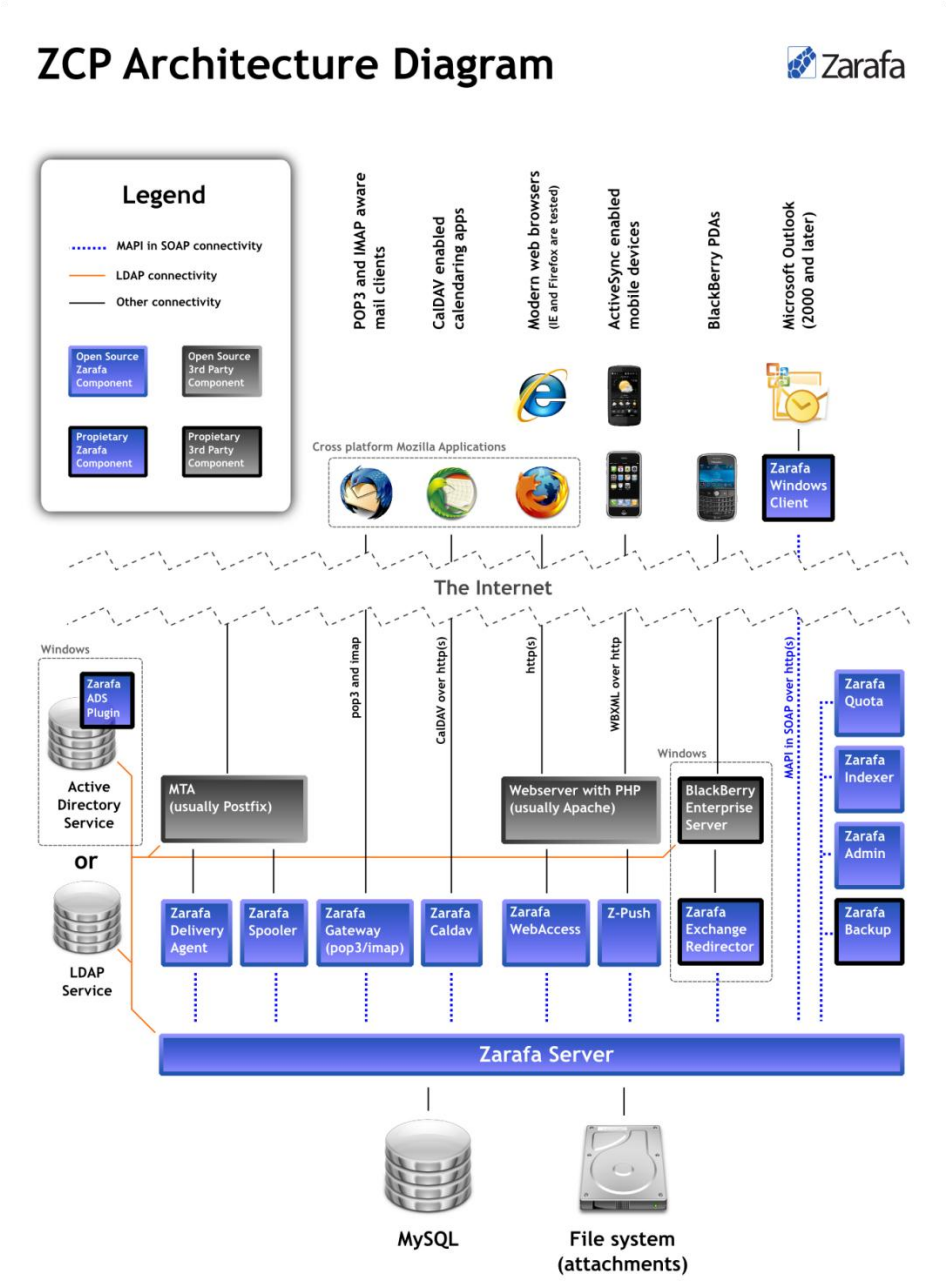
6.2.1 Componentes de Zarafa

- Zarafa Server
- Zarafa License Manager
- Zarafa Windows Client
- Zarafa WebAccess
- Zarafa Delivery Agent y Zarafa Spooler
- Zarafa Admin
- Zarafa Gateway
- Zarafa Monitor
- Zarafa Caldav
- Zarafa Backup Tools
- Zarafa Indexer
- Apache
- PHP
- PHP-MAPI extension
- Python-MAPI extension

El paquete de Zarafa contiene unos componentes y trabajando con estos podemos configurar todos los parametros para utilizar Zarafa en nuestro SO Linux. La mayoría de estos componentes tienen archivos de configuración, los cuales podemos modificar para conseguir distintas configuraciones. Para configurar estos componentes podemos consultar el Manual para desarrolladores de Zarafa, que se adjunta en el CD-Rom del proyecto.

6.2.2 Protocolos y conexiones

Todas las aplicaciones conectadas directamente al servidor de Zarafa usan MAPI en SOAP. Incluso WebAccess utiliza MAPI en SOAP para conectarse al servidor Zarafa. El Zarafa Windows Client es un estándar de MS Windows compatible con proveedores MAPI. Se conecta al servidor mediante el protocolo HTTP(s).



7. Instalando nuestro sistema

7.1 Primeros pasos de instalación

1) Primero descargamos "Ubuntu 10.04 LTS 32 bit". Quemamos la imagen e instalamos nuestro sistema operativo

2) Ahora podemos descargar Zarafa (ZCP). Debemos elegir la version correcta compatible con nuestro SO Ubuntu. He utilizado el siguiente paquete, pero podemos actualizarlo despues de la instalación.

   http://download.zarafa.com/community/final/7.0/7.0.1-28479/zcp-7.0.1-28479-ubuntu-10.04-i386-free.tar.gz

3) Necesitamos algún software adicional para que nuestro Zarafa funcione correctamente. Por lo tanto instalamos previamente MySQL, Apache 2.2 y un servidor SMTP (utilizare Postfix).

4) Ahora ya podremos instalar nuestro paquete de archivos de Zarafa.

5) Tendremos que modificar algunos archivos de configuración que explico en la versión completa del proyecto para que todo quede configurado correctamente.

6) Finalmente tenemos el siguiente escenario todo instalado y configurado:

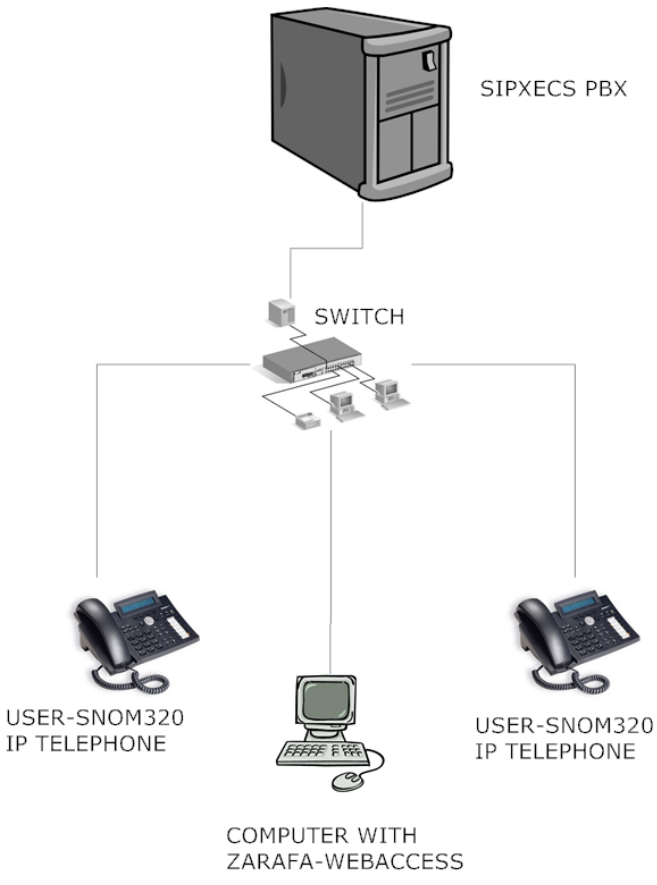
Crearé dos usuarios que serán los que registraré con cada teléfono. Estos usuarios se crean con el componente Zarafa-Admin. Para crear un usuario escribo lo siguiente:

zarafa-admin –c 207 –p test –f 'user1' –e 207@sipxecs.example.com

Estos serían los datos de mi usuario:

Name User1: 207 Pass: test
Mail: 207@sipxecs.example.com

Y así sucesivamente con todos los usuarios que quiera crear.



7.2 Configuración del servidor SipXecs

Tendré que configurar los siguientes parámetros utilizando el manual de SipXecs que adjunto en el CD-Rom del proyecto:

- **User name**
- **Voice-mail PIN**
- **SIP password**
- **User alias**
- **Email Address**
- **User group**
- **Phone serial number**
- **Phone model**

8. WebAccess y arquitectura del Plugin desarrollado

8.1 Introducción

Mediante Zarafa WebAccess se puede acceder al servidor de Zarafa a través del navegador de internet. Zarafa Webaccess contiene características como:

- Compartir email de forma sencilla
    - Adjuntar archivos
    - Crear carpetas públicas
    - Posibilidad de abrir el calendario o el correo de otros usuarios

- User-Friendly
    - Multiples navegadores: Firefox e Internet Explorer
    - Libre/ocupado en un horario
    - Calendario multi usuario

- Búsqueda avanzada de cualquier item
    - Indice Lucene integrado
    - Herramienta de búsqueda avanzada

8.2 Arquitectura del Plugin

Esto permite a los desarrolladores personalizar características, integrar aplicaciones Third Party y cooperar en la comunidad para desarrollar nuevas funcionalidades. El administrador del servidor tiene permiso para añadir plugins a la interfaz WebAccess simplemente añadiendo los archivos del programa desarrollado en un directorio concreto denominado “Plugins”.

8.2.1 Anatomía del Plugin

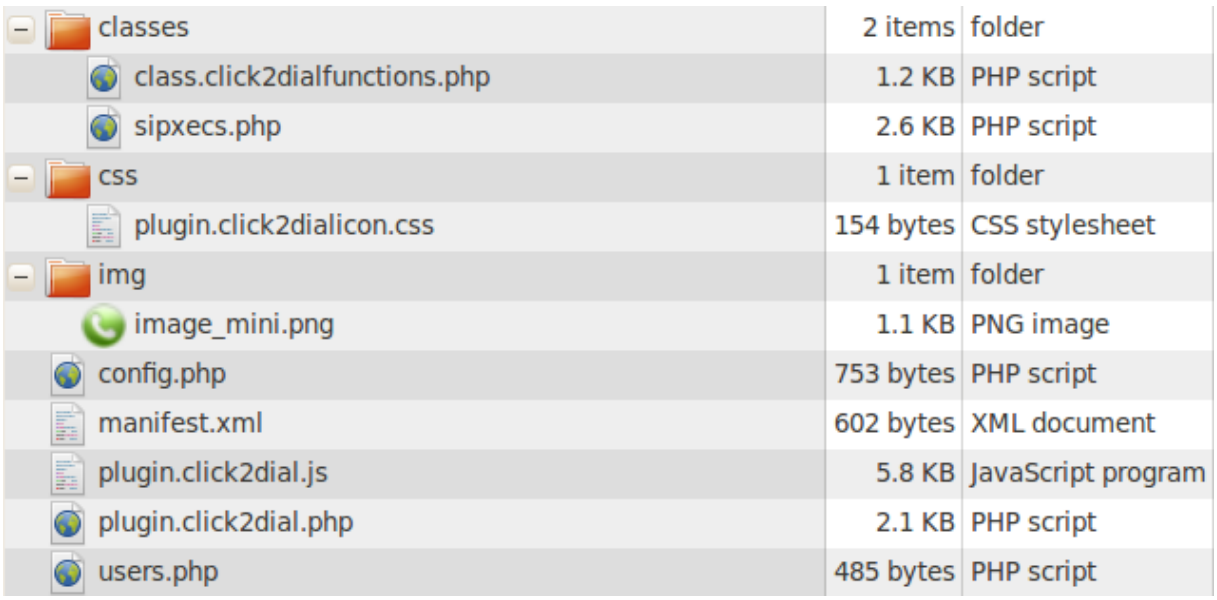
Un plugin puede interactuar con la interfaz WebAccess de varias maneras:

- **Añadiendo nuevos componentes:** Un plugin puede crear sus propios módulos, vistas, widgets y conjuntos de CSS y añadirlos a WebAccess a través de un simple plugin. Esto permite al desarrollador crear sus propias aplicaciones e integrarlas en la interfaz.
- **Extender componentes existentes (solo en el lado del cliente):** Un plugin puede extender componentes y añadir o cambiar sus funcionalidades.
- **Hooks:** Un plugin puede enganchar en ciertos puntos del codigo y parar el código nativo de WebAccess. Por lo tanto se pueden modificar objetos y datos antes de que el código nativo termine de ejecutarse. Se pueden implementar tus propios hooks, aunque en este proyecto no será necesario.

8.2.2 Estructura de un Plugin

El plugin se define en un archivo denominado manifest.xml, donde está la configuración de dicho plugin en formato XML. Cada componente que añada será listado en este archivo.

El plugin desarrollado se pega en la carpeta "Plugins" de Zarafa-Webaccess. Esta sería la estructura del plugin que he desarrollado para mi proyecto.



9. Desarrollo del Plugin para Zarafa-Webaccess

9.1 Configuración

Modificando las opciones en el archivo "config.php" se definen 3 características:

- La ruta donde se encuentra el plugin se puede personalizar
- Podemos activar o desactivar que se carguen todos los plugins
- Podemos desactivar los plugins que deseemos dejando otros activos

9.2 Manifest

Este archivo en XML describe que modulos se utilizaran en el Plugin.

```
−<plugin version="1">
 −<info>
    <version>0.1</version>
    <name>click2dial</name>
    <title>Zarafa Click to Call</title>
    <author>Daniel Peinado Lopez</author>
    <authorURL>http://www.zarafa.com</authorURL>
    <description>Zarafa Click to Call</description>
  </info>
 −<resources>
  −<server>
     <serverfile>config.php</serverfile>
     <serverfile>users.php</serverfile>
     <serverfile>plugin.click2dial.php</serverfile>
    </server>
  −<client>
     <clientfile type="js">plugin.click2dial.js</clientfile>
     <clientfile type="css">css/plugin.click2dialicon.css</clientfile>
    </client>
  </resources>
 </plugin>
```

Tendremos una parte que se ejecuta del lado del cliente, y otra del lado del servidor. Cada una de estas partes sigue un patrón genérico y después le añadimos lo que queremos que en realidad haga esa aplicación.

9.3 Plugin en el lado del cliente

Podemos añadir componentes completamente nuevos a la interfaz Webaccess. Estos componentes tendrán el mismo acceso y permisos que los componentes nativos de la aplicación. Para extender componentes existentes podemos añadir un archivo JavaScript que extienda los objectos nativos. Siempre seguiremos una estructura como la siguiente para crear un Plugin:

```
/**
 * Example plugin class
 */

Pluginexample.prototype = new Plugin;
Pluginexample.prototype.constructor = Pluginexample;
Pluginexample.superclass = Plugin.prototype;

function Pluginexample(){}

Pluginexample.prototype.init = function(){
      this.registerHook("main.hierarchymodule.sharedFoldersPane.buildup");
}

Pluginexample.prototype.execute = function(eventID, data){
      switch(eventID){
            case "main.hierarchymodule.sharedFoldersPane.buildup":
                  this.doExampleStuff(data);
                  break;
      }
}

Pluginexample.prototype.doExampleStuff = function(data){
      // Do stuff
}
```

Este es el código que debemos escribir siempre que queramos crear un Plugin para el lado del cliente.

```
Pluginexample.prototype = new Plugin;
Pluginexample.prototype.constructor = Pluginexample;
Pluginexample.superclass = Plugin.prototype;
```

Estas lineas son necesarias para todas las clases de los componentes de Webaccess. Es este caso la clase se llama “Pluginexample” y extiende la superclase “Plugin”

```
function Pluginexample(){}
```

Este sería el constructor, que es necesario para registrar los hooks.

```
Pluginexample.prototype.init = function(){
     this.registerHook("main.hierarchymodule.sharedFoldersPane.buildup");
}
```

Las funciones “init” es llamada para inicializar la clase del plugin. En este punto se inicializan los “hooks” que vayamos a utilizar en nuestro plugin. Cada “hook” se identifica por un String. El “hook” de este ejemplo (main.hierarchymodule.sharedFoldersPane.buildup) nos permitiría colocar enlaces en la lista de jerarquía de Webaccess. Por ejemplo, el que yo tengo para mi programa, me permite añadir objetos al menú desplegable cuando se hace click derecho. Lo utilizo para añadir un botón que sea “Llamar al usuario”.

```
Pluginexample.prototype.execute = function(eventID, data){
     switch(eventID){
           case "main.hierarchymodule.sharedFoldersPane.buildup":
                 this.doExampleStuff(data);
                 break;
     }
}
```

Esta función se llama cuando se han registrado uno o más “hooks” y ese “hook” se inicia con el código nativo de Webaccess.

```
Pluginexample.prototype.doExampleStuff = function(data){
      // Do stuff
}
```

Em esta parte del código se programa todo lo que queremos que haga nuestro programa cuando suceda el "hook" que hemos dicho antes. Es decir, lo que pasaría cuando se pulse el botón "Llamar al usuario".

9.4 Plugin en el lado del servidor

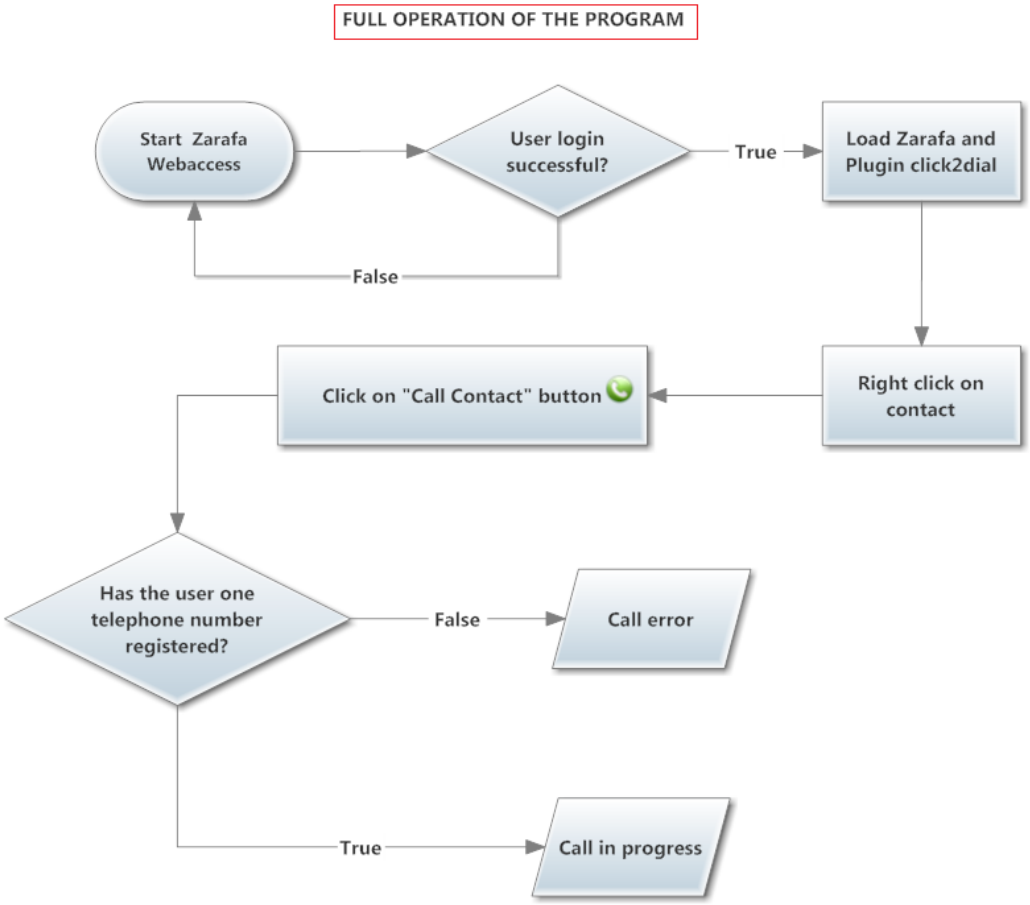
Esta estructura es casi idéntica a la del lado del cliente, cambiando algún pequeño detalle en el código base que he explicado anteriormente.

```
<?php
/**
 * Example Plugin class
 */
class Pluginexample extends Plugin {
      function Pluginexample(){}
      function init(){
            $this->registerHook('dialogs.general.buildMenu');
      }
      function execute($eventID, &$data){
            switch($eventID){
                  case 'dialogs.general.buildMenu':
                        $this->addDialogMenuItems($data);
                        break;
            }
      }
      function addDialogMenuItems(&$data){
            // Do stuff
      }
}
```

Tendremos el código PHP en lugar de JavaScript, pero viene a definir lo mismo que en el lado del cliente.

9.5 Diagramas de flujo del plugin

Estos diagramas los podemos ver más en detalle en la versión completa del proyecto escrita en inglés. Aquí se muestra el diagrama de cómo funcionaria el programa de modo general y no mostraré cada función de forma detallada.



En este diagrama vemos de forma sencilla como funcionaría el programa a grandes rasgos. Tras registrarnos correctamente, hacemos click derecho en el contacto al que deseamos llamar, se abre el menu desplegable donde aparecerá nuestro botón “Llamar al usuario”. Tras pulsar este botón el plugin hará una serie de encrinptaciones, comprobaciones y realizará la llamada o mostrará una expcepción de qué es lo que falla.

10. Conclusión y mejoras futuras

La tecnología CTI es el resultado de unir la telefonía y los ordenadores para proporcionar servicios. Este proyecto ha conseguido su objetivo principal que era realizar un software con el que realizar llamadas a los usuarios de un cliente de correo. He diseñado un programa que nos permita añadir usuarios que posean un telefono IP y puedan realizar llamadas entre ellos tras registrarse en el servidor de Zarafa.

Además he programado de forma adicional, un archivo específico en el que cambiando ciertos parámetros podríamos adaptar nuestro programa a diferentes PBX. De esta forma modificariamos un archivo de configuración y no tendriamos que escribir un nuevo programa para cada centralita.

Para que funcione correctamente tenemos que registrar a los usuarios con el mismo nombre que el numero de telefono. Es decir, si mi número de telefono es 206, mi nombre de usuario sería 206. Sería bueno mejorar esto en el futuro utilizando para ello una base de datos con nombres y telefonos.

Los telefonos IP tienen una contraseña (que no tiene que ver con la contraseña de usuario). He utilizado la misma contraseña para todos, ya que sino no funcionaría el programa. Esto sería otra opción a mejorar creando la opción de cambiar la contraseña desde la propia interfaz de Zarafa Webaccess.

Si quisieramos utilizar esta aplicación con grandes redes de telefonía para llamar a usuarios que están en distintos lugares geográficos deberíamos programar el plugin de manera que añadiese automáticamente un prefijo a nuestros números de teléfono dependiendo del lugar geográfico donde estuviésemos.

He encontrado algunas dificultades especialmente al principio, ya que no tenía muy claro cómo empezar con los primeros pasos. También encontré alguna dificultad en la parte de programar, ya que no soy experto en programación y nunca había trabajado antes con PHP ni JavaScript, pero finalmente he conseguido que mi proyecto funcionase correctamente cumpliendo las especificaciones e incluso añadiendo alguna pequeña característica más.

Finalmente pienso que he aprendido muchas cosas desarrollando este proyecto, ya que he aprendido a enfrentarme a los problemas que he encontrado, y tengo más conocimientos sobre telefonía IP, SIP, la tecnología Click2Call y las centralitas PBX SipXecs.