PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE ADMINISTRACIÓN DE CALENDARIOS ONLINE CON SINCRONIZACIÓN MÓVIL

Tesis para optar el Título de Ingeniero de las Telecomunicaciones, que presenta el bachiller:

CESAR STUARDO LUCHO ROMERO

ASESOR: ING. ARTURO DÍAZ ROSEMBERG

Lima, 17 de febrero del 2012

Resumen

El presente proyecto de Tesis consiste en el diseño e implementación de un sistema de administración de calendarios online de código abierto, para ser usado por los alumnos de la Pontificia Universidad Católica del Perú como alternativa al servicio de agendas presente en el campus virtual, con la posibilidad de sincronizar eventos programados vía un dispositivo móvil.

Se ha dividido el proyecto en 5 etapas. El primer capítulo está centrado en la exposición de las tecnologías y software a usarse en todo el proyecto. El segundo capítulo presenta un análisis de sistemas de calendario online, tanto gratuitos como de código libre, con el fin de realizar una comparativa de requerimientos y funcionalidades, que sirvan para la elección de dos sistemas de calendario, uno web y uno móvil, como parte de la arquitectura del sistema final. El tercer capítulo detalla el manejo de los usuarios y de los eventos de cada sistema seleccionado previamente en función de sus campos a nivel de base de datos. El cuarto capítulo muestra el diseño de la arquitectura del sistema que se logra mediante la sincronización de las bases de datos del sistema de calendarios online y del móvil. En un primer lugar se muestra la interconexión de los usuarios y posteriormente la interconexión y sincronización de los eventos de ambos sistemas. Así mismo se muestran las interfaces y módulos que se adicionan con la finalidad de lograr un único sistema de calendarios online con soporte para la sincronización de eventos vía un dispositivo móvil. El quinto capítulo presenta el procedimiento seguido para lograr la implementación del sistema diseñado en un entorno real, usando un servidor y un nombre de dominio activo en la PUCP. Finalmente en el sexto capítulo, se muestran las pruebas realizadas al sistema en producción.

Por último, se presentan las conclusiones y recomendaciones del presente trabajo, además de proponer algunos trabajos futuros que permitan optimizar el diseño de la arquitectura del sistema para una posible implementación en un entorno externo al de la Pontificia Universidad Católica del Perú.

Dedicatoria

A mis padres Mauro y Magda

A mis tíos Hugo y Richard

A mis hermanos Eduardo y Grezzia

A Diana

A mis amigos.

Agradecimientos

En primer lugar agradecer a Dios por darme fuerza, salud e inteligencia para poder llegar hasta este momento tan importante en mi vida.

En segundo lugar agradecer a mis padres por su constante apoyo y preocupación por que alcance mis metas, sueños y objetivos desde que empecé el colegio, pasando por la universidad y por todo lo que me tiene preparado el futuro.

Agradecer a mi tío Hugo por la humildad, empeño y perseverancia que me enseñó con el ejemplo, desde que era un niño y a mi tío Richard por insertarme en el mundo de la ingeniería y tecnología desde muy pequeño.

Agradecer al profesor Enrique Larios, por asesorarme y ayudarme durante los momentos más difíciles de mi carrera así como también por cultivarme siempre la perseverancia por todo aquello que requería mayor dedicación. Además de brindarme sus conocimientos para poder terminar mi tesis.

Agradecer a mi asesor, el Ing. Arturo Díaz por la confianza que depositó en mí, por su apoyo y asesoría para llevar todo hacia delante de la mejor manera y por todos los conocimientos que me transmitió desde la primera vez que lo conocí.

Agradecer a mi enamorada y mejor amiga Diana, por su cariño, compañía y consejo desde el comienzo de mi Tesis.

Agradecer también al profesor Ángelo Velarde por su amistad y porque siempre me insistió en que me titule.

Finalmente agradecer a mis amigos y profesores, pues gracias a ellos la vida universitaria ha sido tan emocionante y amena.

Índice

Lista de Figuras

Lista de Tablas

Introducción

La Pontificia Universidad Católica del Perú realiza diversos eventos en todo el campus universitario y locales de la institución como el centro cultural, CCPUCP, la escuela de música, entre otros. En varias ocasiones por falta de difusión o por el poco interés, el alumno pierde conocimiento de todos estos eventos.

Por otro lado, la Universidad proporciona un sistema de agendas a todos sus miembros por medio del campus virtual, por el cual se pueden observar los eventos personales de cada usuario pero posee un nivel bajo de personalización y no permite la sincronización con ningún dispositivo móvil.

Ante los problemas observados, en la presente Tesis se propone la implementación de un único sistema de calendarios online, haciendo uso y sincronizando dos sistemas existentes, como es el WebCalendar y el Funambol, ambos software libre de código abierto.

El sistema de calendarios online se puede visualizar desde cualquier navegador web, con el fin de proveer a la comunidad universitaria una agenda virtual con un alto grado de personalización y de centralización de eventos creados por el usuario.

Así mismo, el sistema brinda la capacidad al usuario de poder sincronizar los eventos de su calendario online con un dispositivo móvil, haciendo uso de un software brindado de forma gratuita de parte del sistema.

Finalmente, el sistema es implementado bajo ciertos requerimientos y posteriormente sometido a pruebas un entorno de desarrollo real y de pre-producción.

Capítulo 1

Conceptos básicos y estudio de la tecnología

En el siguiente capítulo se explicarán los conceptos básicos usados en toda la Tesis, así como también se realizará un estudio a la tecnología a usarse para poner en contexto las definiciones usadas en el desarrollo de los siguientes capítulos.

1.1 Calendarios online

Se le llama calendario online al calendario personal o empresarial que se encuentra dispuesto en la red (Internet), para poder verlo, modificarlo y eliminarlo desde cualquier parte en que uno se encuentre, siempre y cuando tenga conexión a Internet. Así también, un calendario online puede ser observado por medio de distintos dispositivos electrónicos, como una computadora, un PDA, un celular, entre otros.

Entonces, existen dos maneras de observar los eventos presentes en un calendario. Una es por medio del navegador web de un dispositivo electrónico; mientras que la otra es tener todos los eventos del calendario en el dispositivo electrónico, lo cual se logra por medio de una sincronización de eventos usando un software específico para el dispositivo usado.

1.1.1 Estándares y protocolos en estudio

Los calendarios online poseen distintas maneras de intercambiar información y sincronizarse ya sea con otros calendarios, un cliente móvil o un servidor. Entre éstas maneras, para el desarrollo de la presente Tesis, se analizará el protocolo de intercambio de información SyncML, así como también el estándar ICal para representación de información de un evento y por último el formato SIF (Sync4j Interchange Format) como representación de la PIM (Personal Information Manager).

1.1.1.1 SyncML

Con sus siglas en ingles “Synchronization Markup Languaje”, es un protocolo para la sincronización de información, como contactos, eventos, alertas, entre otros, todo esto conocido en el mundo web como PIM (Personal Information Maganer), a través de múltiples redes, plataformas y equipos. Es promovido y desarrollado por la OMA (Open Mobile Alliance) y apoyado en su desarrollo por un gran número de empresas.

Debido a que cada proveedor de ofrecía su propio software para sincronizar la PIM, con su propia manera de estructurar los datos y de encapsularlos, la OMA propone un protocolo de sincronización común llamado “SyncML”, el cual consta de tres postulados y cinco características principales, las cuales son detalladas en el anexo 1. De esta manera, un protocolo común consiste en sincronizar dispositivos como PDA’s, teléfonos móviles, PCs y laptops con un servidor sin importar el medio de transmisión ni la tecnología. Según se observa en la figura 1-1, se necesita instalar un software en el dispositivo del usuario, para que éste envíe los datos usando el protocolo SyncML.

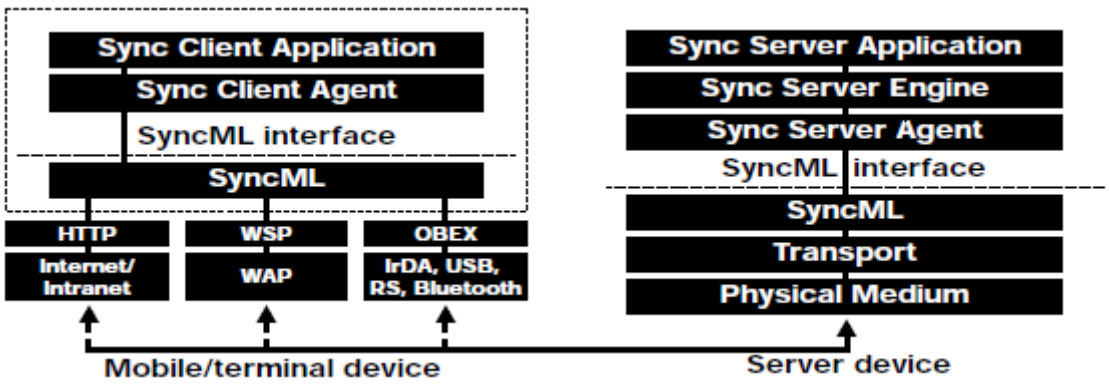

FIGURA 1-1: ARQUITECTURA DEL PROTOCOLO SYNCML

Fuente: “SyncML White Paper” [SYN2010]

1.1.1.2 ICalendar

Es un estándar definido en el RFC 5545 [ICA2009], usado para el intercambio de información de calendarios. Es también conocido como "ICal", debido al nombre del programa de Apple Computer, que fue la primera aplicación en implementarlo. Es utilizado en una gran variedad de programas incluyendo ICal de Apple, Mozilla Calendar (incluyendo Mozilla Sunbird), Google Calendar, Lotus de IBM, Evolution de Novell, Simple Groupware, Windows Live Calendar, Microsoft Outlook, entre otros.

Los datos contenidos mediante el formato ICalendar pueden ser enviados por diferentes medios de transmisión, tal como el correo electrónico y otros medios, pues ICalendar fue diseñado para ser independiente del protocolo de transmisión. Así mismo, un archivo que contenga información de calendario o agendas, que posea eventos, tareas, publicación o distribución del tiempo ocupado y disponible, al ser renombrado con la extensión “ics" y tener la estructura adecuada, adopta el formato de ICalendar. Su estructura y detalle de los campos referenciados en la presente Tesis se encuentran en el Anexo 2.

1.1.1.3 SIF

Con sus siglas en ingles, Sync4j Information Format, es una manera de representar la información personal del usuario que proviene de diferentes clientes en una estructura común para de esta manera realizar el intercambio de información entre el cliente y el servidor de una manera más transparente.

El formato SIF se basa en XML y usa su estructura para representar la información del usuario. Así mismo, existen cuatro sub formas de almacenar la información con formato SIF:

- SIF-C para contactos
- SIF-E para eventos
- SIF-T para tareas
- SIF-N para notas

Cada uno de estos formatos que toma SIF es usado por el Funambol, un software al cual en capítulos posteriores se hará referencia. De esta manera, el formato y valores que adopta SIF-E es de especial interés para la presente Tesis, puesto que su

estructura y los valores que ésta puede tener deben ser traducidos al formato iCal por medio de una base de datos.

Se adjunta la estructura del formato SIF-E y el detalle de sus campos en el anexo 3.

1.2 Software en uso

El software a usarse en la presente Tesis es muy diverso; sin embargo para la solución propuesta y el análisis de los diversos calendarios, se hace uso de dos metodologías, software libre y software de código abierto.

En primer lugar es importante resaltar la diferencia entre el software libre y el software de código abierto o “open source” (nomenclatura en ingles), debido a que ambos conceptos son muy usados en el mundo actual, pero que en situaciones particulares pueden expresar la misma filosofía con ideales distintos.

Luego, un término muy usado en el mundo del software es “licencia” y se entiende como un contrato entre el proveedor que ofrece el software y el usuario que acepta o consume el software, de manera que éste acepta una serie de términos y condiciones establecidas para su uso, empleo y distribución del mismo impuestas por el proveedor.

1.2.1 Software libre

Promovido por la “Free software Foundation” [FSF2010], indica que un software es considerado libre cuando cumple con las siguientes libertades [SWL2010]:

- La libertad de usar el programa con cualquier propósito, esto es para nivel institucional, empresarial, personal o para el uso que cada usuario desee darle al programa, sin necesidad de reportar los cambios o realizar algún pago extra.
- La libertad de estudiar cómo funciona el programa y adaptarlo a las necesidades de cada usuario. Ante esta premisa se entiende que el acceso al código fuente es una condición previa a esto.
- La libertad de distribuir copias de sus versiones modificadas a terceros. Si lo hace puede dar a toda la comunidad una oportunidad de beneficiarse de sus cambios. Nuevamente el acceso al código fuente es una condición necesaria.

Entonces, se entiende que para la segunda y tercera libertad es necesario el acceso al código fuente, por lo que en la filosofía de software libre el acceso al código fuente es un derecho del usuario, que siempre debe ser capaz de acceder a él sin importar el tipo de programa que use, pues forma parte de la filosofía de software libre.

Sin embargo “free software” o software libre no significa software que el software pueda ser comercializado, razón por la cual uno puede realizar modificaciones al código fuente de un programa determinado, para posteriormente venderlo, pero siempre dándole las tres libertades de software libre al usuario final.

1.2.2 Software Open source

En comparación con software libre, el software open source o código abierto es una terminología adaptada por la “Open Source Initiative (OSI)” [OSI2010] y en ésta se expone que para que un software tenga la denominación de código abierto o “open source”, éste debe reunir criterios presentes en el Anexo 4.

Tal y como se presentan en los diez puntos de dicho anexo, el software de código abierto indica especificaciones algo más pragmáticas; mientras que software libre indica ideales, formas de tratar y entender un software bajo ciertas libertades.

Así mismo, para los diferentes programas de código abierto y software libre, existen licencias, tales como [LIC2010]: Licencia GNU GPL, Licencia GNU LGPL, Licencia Pública de Mozilla (MPL), Licencia de la Fundación Apache, entre otras.

Aún con sus diferencias, viéndolo desde el punto de vista del usuario, la mayoría de software libre es de código abierto y a su vez el software de código abierto es en su mayoría libre; por lo cual, ambas tendencias comparten en muchos casos las mismas licencias y han optado por usar un término en común, llamado FOSS (Free Open Source Software), el cual junta las libertades del software libre, con las especificaciones de software de código abierto.

1.3 Tecnologías

Tanto para su desarrollo como para la implementación, la Tesis está basada netamente en software libre y open source. De esta manera las tecnologías a usar son

en su totalidad virtuales, es decir, no es necesario el uso de algún hardware o equipo de telecomunicaciones en específico; más que una PC para el desarrollo.

Entonces, para el desarrollo del sistema se va a hacer uso de 2 lenguajes de programación: PHP y Java; mientras que para la interconexión del sistema se usará el gestor de base de datos MySQL.

1.3.1 Lenguajes de programación

En general se clasifican en lenguajes de programación de bajo nivel y de alto nivel.
En los lenguajes de programación de bajo de nivel las sentencias que se usan están vinculadas íntimamente con el hardware.
Por otro lado, un lenguaje de programación de alto nivel se caracterizan por expresar algoritmos de una manera adecuada a la capacidad cognitiva humana, tales como: “System.out.println(variable)”, la cual es una sentencia de Java usada para imprimir un texto en consola en función de una “variable”.
Por ende, en la presente Tesis se va a hacer uso de dos lenguajes de programación de programación de alto nivel: PHP y Java.

1.3.1.1 Lenguaje PHP

El lenguaje PHP, con sus siglas “PHP Hypertext Pre-processor” [PHP2010], es un lenguaje de programación usado principalmente para el desarrollo de entornos web..
Es llamado por gran parte de la comunidad desarrolladora, como un “HTML dinámico”, pues lo que un usuario observa al final en su PC es un entorno HTML pero que previamente ha sido procesado en el servidor como PHP.
Es entonces previsible entender que éste lenguaje al trabajar en entornos web, se ejecuta sobre una arquitectura cliente servidor, en la cual el cliente envía peticiones por medio de una interfaz web y el servidor procesa éstas peticiones, recibiendo los parámetros necesarios y generando de manera dinámica diversos resultados que son observados por parte del cliente como un solo entorno web con código HTML.PHP también permite conectarse con bases de datos de diversos fabricantes, tales como: MySQL, Postgres, Oracle, ODBC, SQLite.

Para ser usando en el lado del cliente, una página desarrollada en PHP no requiere más que un explorar de Internet (Internet Explorer, Mozilla Firefox, Google Chrome,

entre otros); mientras que para ser ejecutado y procesado en el lado del servidor es necesario tener instalado el módulo de PHP5 (o alguna otra versión que se desee) y en caso se requiera conectividad con algún otro servicio, tal como una base de datos, se instalará el modulo correspondiente en función al sistema operativo. Lo cual trae como último apunte resaltar que “PHP puede ser ejecutado en la mayoría de los sistemas operativos tales como Mac OS (Apple), Windows, Windows server (Microsoft), CentOS, Ubuntu, Red hat, entre otras distribuciones de Linux” [PHS2010].

1.3.1.2 Lenguaje Java

Java es un lenguaje de programación orientado a objetos creado por Sun Microsystems [SUN2010] en el año 1991, para poder funcionar en distintos tipos de procesadores.

El código Java, una vez compilado, puede llevarse sin modificación alguna sobre cualquier máquina, y ejecutarlo. Esto se debe a que el código se ejecuta sobre una máquina virtual, la “Java Virtual Machine” (JVM) [JVM2010], que se encarga de interpretar el código (ficheros compilados .class) y convertirlo a código particular de la CPU en donde esté corriendo (siempre que se soporte dicha máquina virtual).
Sun Microsystems define tres plataformas en un intento por cubrir distintos entornos de aplicación. Así, ha distribuido muchas de sus APIs de forma que pertenezcan a cada una de las plataformas:

- Java ME (Java Platform, Micro Edition) o J2ME, orientada a entornos de limitados recursos, como teléfonos móviles, PDAs, etc.
- Java SE (Java Platform, Standard Edition) o J2SE, para entornos de gama media y estaciones de trabajo. Aquí se sitúa al usuario de un PC de escritorio.
- Java EE (Java Platform, Enterprise Edition) o J2EE, orientada a entornos distribuidos empresariales o de Internet.

1.3.2 Base de datos Mysql

Desde el punto de vista informático, una base de datos es un sistema de información formado por un conjunto de datos almacenados en discos duros y un programa encargado de manipular éste conjunto de datos. Cada base de datos está conformada

por una o más tablas y a su vez cada tabla tiene una o más filas y columnas, en la cuales se almacena la información de manera organizada.

De esta manera, al software encargado de la manipulación de los datos se le conoce como un “sistema de gestión de base de datos”, el cual sirve de interfaz entre el usuario y la información almacena en los discos.

Entre los diferentes gestores de base de datos están: MySQL, Oracle, PostgreSql, Access, Microsoft SQL Server, entre muchos otros. Para el desarrollo de la presente Tesis se ha optado por MySQL, pues estaba basada en licencia GLP y es soportada por ambos sistemas como parte de la implementación.

MySQL [SQL2010] es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. Por un lado se ofrece bajo la licencia GNU GPL como software libre para cualquier uso de parte de los usuarios, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a MySQL una licencia corporativa específica que les permita este uso.

Actualmente, muchos lenguajes de programación pueden hacer consultas a una base de datos MySQL, pues es soportada de manera eficiente y en el caso particular de la presente Tesis, por PHP.

Debido al alto uso de las bases de datos, MySQL posee distintos motores de almacenamiento, los cuales se encargan de distribuir y manejar la data almacenada en la base de datos de una manera particular, entre las cuales sobresalen dos: MyISAM e InnoDB. Una de las mayores diferencias entre MyISAM e InnoDB es que ésta última soporta transacciones e integridad referencial, referenciada como ACID (Atomicity, Consistency, Isolation and Durability) lo cual nos garantiza la integridad de las tablas ante un desperfecto. En los últimos años InnoDB está desplazando a MyISAM como motor de almacenamiento MySQL por defecto.

1.3.2.1 Trigger en MySQL

Un trigger o disparador es una rutina autónoma asociada a una tabla o vista que se activa de manera automática cuando se realiza una sentencia de INSERT, UDPATE o DELETE sobre la tabla o vista en cuestión. Los triggers no pueden ser activados por

cambios en las tablas hechos por APIs externos a la base de datos que modifican sus valores de manera directa. Pueden usarse en MySQL desde la versión 5.0.

Los triggers se pueden ejecutar antes (BEFORE) y/o después (AFTER) de que sean modificados los datos en la fila correspondiente. Por ejemplo, se puede tener un trigger que se active antes que un registro sea borrado o después que sea actualizado.

Hacen uso de dos sentencias claves en sus procedimientos, OLD y NEW que se refieren a los valores que tienen las columnas antes y después de la modificación. Cuando se realiza un INSERT, un eventual trigger relacionado a éste permite asociar los nuevos valores por medio de la sentencia NEW; mientras que, cuando se realiza un DELETE sólo se pueden manejar los datos a ser borrados o datos antiguos por medio de OLD; para la sentencia UPDATE, se pueden usar ambos bajo la misma lógico anteriormente mencionada.

1.3.2.2 Procedimientos almacenados

Un procedimiento almacenado es un conjunto de comandos SQL que pueden almacenarse en el servidor. Una vez que se hace, los clientes no necesitan realizar los comandos individuales uno por uno; sino, en su lugar pueden referirse al procedimiento almacenado.

Los procedimientos almacenados mejoran el rendimiento del servidor de base de datos y el servidor que realiza la consulta, por ejemplo, si para un mismo cliente se tienen que realizar un gran números de consultas de base datos todas ellas relacionadas entre sí, cada consulta generaría una conexión con el servidor de base de datos y por ende un incremento de procesamiento y de tráfico (de estar el servidor de base de datos remotamente). Sin embargo, si solo se envía la llamada a un procedimiento almacenado con los parámetros de entrada requeridos, todas las consultas las realiza el servidor de base de datos internamente y posteriormente se envía el resultado según las variables asignadas, con lo cual se evita el exceso de tráfico y aumenta la performance del sistema.

En MySQL un procedimiento almacenado se invoca por medio de la sentencia "*CALL*" seguido del nombre del procedimiento, las variables de entrada (IN), las de salida (OUT) y las que pueden funcionar como entrada y salida (INOUT).

1.3.2.3 Event scheduler

A partir de la versión 5.1, MySQL soporta la creación de eventos, los cuales son tareas que se ejecutan de acuerdo a un horario programado, razón por la cual suelen referirse al mismo como "*planificador de eventos*".

Al crearse un evento con un nombre determinado, en la base de datos se crea un objeto con el mismo nombre, el cual contiene una o más sentencias SQL (pueden también ser procedimiento almacenados) que serán ejecutados en un intervalo de tiempo determinado o con un comienzo y fin específico. Conceptualmente la idea de un evento en MySQL es la misma idea de un "crontab" en Unix o un planificador de tareas en Windows, los cuales ejecutan ciertas acciones en determinados momentos definidos por el usuario.

Por defecto el planificador de tareas (event scheduler) viene desactivado en la base de datos MySQL, por lo cual debe ser activado para su uso en la presente tesis, ya sea por medio del archivo de configuración, agregando la línea:

```
event-scheduler = 1
```

O por medio de la interfaz de línea de comandos de MySQL:

```
SET GLOBAL event_scheduler = ON;
```

1.4 Arquitectura cliente servidor

La arquitectura cliente/servidor es un modelo para el desarrollo de sistemas de información en el que las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos. Se denomina cliente al proceso que inicia el diálogo o solicita los recursos y servidor al proceso que responde a las solicitudes. En este modelo las aplicaciones se dividen de forma que el servidor contiene la parte que debe ser compartida por varios usuarios, y en el cliente permanece sólo lo resultante a la petición de cada usuario.

Los clientes realizan generalmente funciones tales como: manejo de la interfaz de usuario, generación de consultas, informe sobre las bases de datos, entre otros; mientras que, por su parte los servidores realizan funciones tales como: gestión de

periféricos compartidos, control de acceso a bases de datos compartidas, enlaces de comunicaciones con otras redes de área local, entre otros.

Así mismo un servidor puede ser del tipo: de archivos, de Bases de Datos (SQL, MySQL, Oracle, otros), de Impresión, de Aplicaciones, entre otros.

En una arquitectura cliente servidor, los usuarios del sistema pueden ser diversos, razón por la cual se ha dividido la arquitectura en 2 modelos, cliente web/servidor y cliente móvil/servidor, las cuales pueden darse tanto en la red cableada como en la red inalámbrica, es decir, es transparente para el acceso al medio.

1.4.1 Cliente web - servidor

En este tipo de arquitectura, el cliente que usa el servicio es un cliente web, es decir, se realizan peticiones y transacciones mediante un navegador, como Internet Explorer, Mozilla Firefox, Google Chrome, entre otros.

Uno de los sistemas a usar en la presente tesis está implementado en PHP y con una base de datos MySQL, razón por la cual se muestra en la figura 1-2 una arquitectura cliente-servidor con PHP.

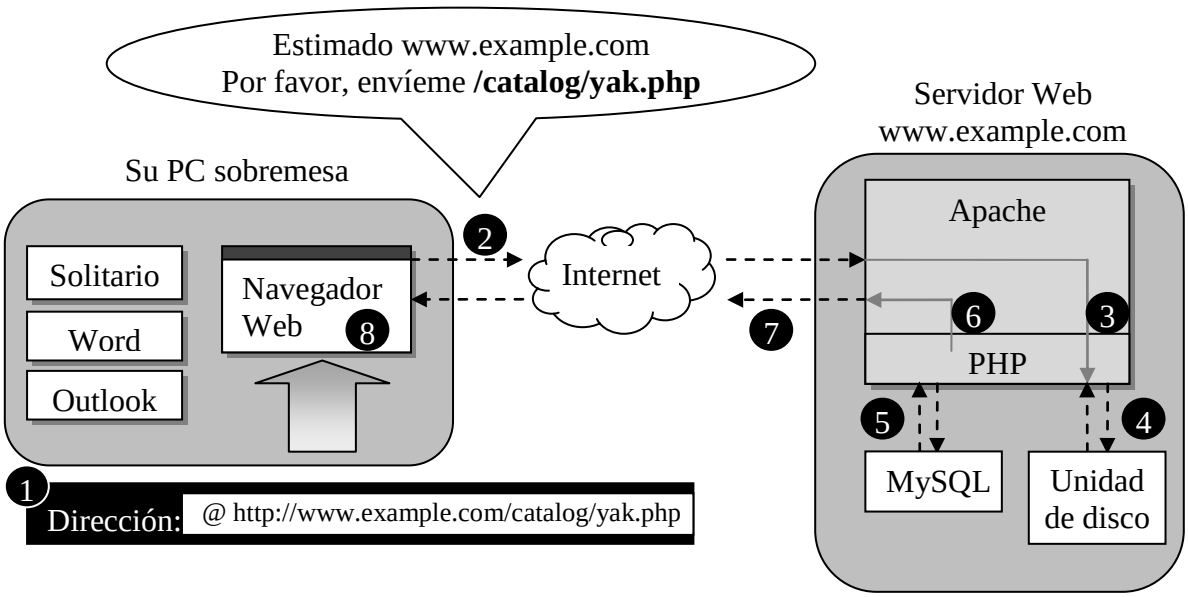

FIGURA 1-2: ESQUEMA DE CLIENTE – SERVIDOR CON PHP

Fuente: "Introducción a PHP 5" [IPH2010]

Según lo observado en la figura 1-2, generalmente son 8 pasos los que se siguen para realizar una comunicación entre el cliente y el servidor que contiene el intérprete PHP:

1. El cliente escribe la dirección en su explorador web, para este caso del ejemplo es: www.example.com/catalog/yak.php
2. El explorador web envía un mensaje por medio del protocolo IP hacia la Internet al servidor con domino “example” y le solicita la página “catalog/yak.php”
3. En el servidor, el Apache recibe el mensaje y le pregunta al intérprete PHP, otro programa que se ejecuta en el servidor (el mismo del Apache), cómo es la pagina “/catalog/yak.php”
4. El intérprete PHP lee el archivo “yak.php” de su lista de archivos que se encuentran en la carpeta “/catalog/” y que ésta a su vez se encuentra en la carpeta principal del servidor Apache.
5. El intérprete PHP ejecuta los comandos en “yak.php”, posiblemente intercambiando datos con un programa de base de datos como MySQL.
6. El intérprete PHP recibe la salida del programa “yak.php” y se lo devuelve al servidor Apache como respuesta a la petición de la página “/catalog/yak.php”.
7. El servidor Apache devuelve el contenido de la página que recibe del intérprete PHP al ordenador del cliente a través de la Internet como respuesta a la petición del explorador web.
8. El explorador web muestras la página en pantalla, siguiendo las indicaciones de las etiquetas HTML en la página.

Se puede entender de manera sencilla el funcionamiento de una petición web a un servidor PHP mediante los ocho pasos descritos anteriormente, donde todas las pantallas que ve el usuario son las recibidas por el intérprete PHP, el cual ha ejecutado las sentencias contenidas en el sistema correspondiente y le muestra al usuario un resultado final con etiquetas HTML.

1.4.2 Cliente móvil - servidor

En una arquitectura cliente móvil y servidor, el cliente es un dispositivo que no se encuentra conectado directamente a una red LAN o una red fija.

En este sentido un cliente móvil es un terminal que se conecta el internet por medio de un acceso inalámbrico, como Wifi, Wimax, 3G, 4G, Bluetooth, entre otros. Como

ejemplo puede ser: un dispositivo celular, una tablet o una computadora personal, como una laptop, notebook o netbook.

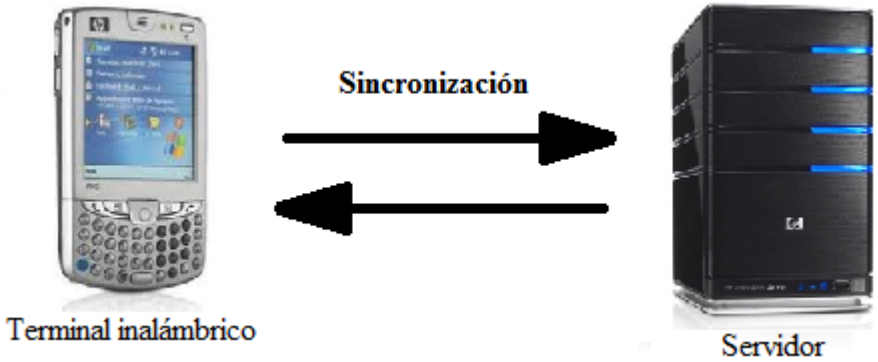

FIGURA 1-3: ARQUITECTURA CLIENTE MOVIL – SERVIDOR

Fuente propia

En la gráfica se observa que se hace uso de la palabra sincronización en medio de la comunicación entre el cliente móvil (un terminal inalámbrico) y el servidor. El termino sincronización hace referencia al poder tener en dos lugares distintos la misma información y poder enviar en ambos sentidos actualizaciones de distintas índoles, pero manteniendo una integridad de sistema.

Entonces, un dispositivo móvil que posee la capacidad de trabajar con un calendario para almacenar citas y visualizar las mismas, puede sincronizar todos sus eventos vía la red inalámbrica hacia un servidor que lo soporte, el cual con la información recibida podría mostrar los eventos en una página web y así tener tanto en el dispositivo móvil como en el explorador web, la misma información. De esta manera, el usuario podrá también actualizar o crear un nuevo evento y ésta modificación se le notificará al servidor y él mismo realizará los cambios necesarios para visualizarlo en la página web.

Capítulo 2

Investigación y análisis del proyecto

En el presente capítulo se hará una investigación sobre los distintos calendarios online disponibles en la web, divididos en dos fases, los que tengan licencias propietarias pero que pueden ser usados por el público en general con toda libertad y los que tengan licencias GNU [GNU2010] ya sea de código abierto o software libre, disponible igualmente para el público y en ambos casos se analizará si soportan sincronización con dispositivos móviles.

Finalmente se elegirán los calendarios necesarios para poder implementar el sistema, el cual debe permitir acceder a éste desde una interfaz web amigable y configurable para el usuario, así como también que soportar la sincronización de eventos en doble sentido (cliente-servidor) para la mayor cantidad de dispositivos móviles disponibles.

2.1 Investigación de calendarios online

Como primer punto se procederá a investigar los distintos calendarios online disponibles en el mercado para el uso del cliente.

Se analizarán calendarios se puedan ser accedidos mediante un explorar web, sin importar el tipo de dispositivo electrónico que se use para acceder al mismo.

Según lo expresado en el punto anterior, quedan por ende descartados todos los calendarios que pueden ser manejados de manera local en sus dos modelos:

- Con licencia propietario, como el calendario de “Microsoft Office Outlook”, el calendario del “Windows live mail” [WIN2010], entre otros.
- Con licencia GNU, como el Mozilla Lightning [MOL2010] o el Mozilla Sunbird.

En la misma medida quedan descartados los sistemas de calendario en los cuales se tenga que realizar un pago para poder instalarlo en un servidor local para su posterior uso como: IceWarp, eGroupWare y Kerio Mail Server.

2.1.1 Análisis de sistemas de calendarios online con licencia propietario existentes en el mercado

El análisis de los sistemas de calendario con licencia propietario que pueden ser accedidos por medio de un explorador web, serán expuestos en los siguientes puntos.
Se analizarán tres sistemas de calendarios:

1. Google calendar [GOO2010]
2. Yahoo calendar [YAH2010]
3. Windows Live Calendar [WLC2010]

Teniendo en consideración tres factores:

- Funcionalidades.
- Nivel de libertad para el usuario de configurar la interfaz del sistema.
- Posibilidad para sincronizarse con un dispositivo móvil.

2.1.1.1 Google Calendar

Es un calendario online, brindado de forma gratuita a los usuarios por Google Corporation, como uno de sus muchos servicios.

Para poder usar Google calendar es necesario tener una cuenta de correo Gmail o Google Apps, sin embargo los eventos e invitaciones que se pueden realizar por este medio son independientes del gestor de calendario, es decir, el usuario puede crear un evento e invitar por correo electrónico a una persona que posea una cuenta de correo

distinta a la de Gmail y la misma podría aceptar la invitación y agregar el eventos a su calendario personal.

Al momento de iniciar sesión éste permite cargar la página web en una versión HTML para conexiones lentas y la versión normal, en AJAX, con una interfaz gráfica más interactiva y atractiva para el usuario. Una vez dentro es posible organizar las vistas, ya sea por día, semana, mes o los próximos cuatro días o también se pueden visualizar los eventos a manera de lista para poder verlos de una manera más general.

Una de sus características más preferidas entre los usuarios, es la capacidad de poder compartir sus calendarios en diferentes niveles, tales como:

- Solo a mis amigos, personas puntuales a las que uno desea compartirle su calendario, con la capacidad de brindarles privilegios para:
    - Solamente ver el calendario
    - Ver y agregar eventos al calendario que se le ha compartido
    - Ver, agregar, editar y eliminar eventos al calendario
- Público, de manera que todo el mundo puede ver el calendario. Útil cuando se trata de una empresa o congregación que desea dar a conocer sus eventos a la comunidad de usuarios sin necesidad de acceder al sistema.

Así pues, se pueden manejar diversos calendarios que se le han compartido a uno, en un mismo entorno con la ventaja de poder colocarle un color determinado a cada calendario y así poder distinguirlo más fácilmente. Así también, uno mismo tiene la capacidad de crear más calendarios como por ejemplo, calendario de la universidad, calendario del trabajo o calendario de las tareas de la casa y poder tenerlos todos por separados, es decir, cada uno individualmente pero con la posibilidad de poder visualizar todos en un mismo calendario si así se desea y poder distinguirlos.

Entre sus otras características tenemos:

- Importar y exportar eventos en formato ".ics" o ".csv"
- Configurar las notificaciones de los eventos para ser enviadas por correo, por medio de una ventana emergente en el mismo Gmail o por mensajes de texto (con un recargo de parte del operador telefónico).
- Envío de invitaciones para eventos, con la capacidad que el invitado puede responder con tres opciones: si, no, quizás.

- Se puede también adjuntar la localización donde se va a realizar el evento cuando se invita a un usuario, de manera que éste pueda visualizar en el Google Maps (otra aplicación gratuita de Google).
- Capacidad de poder observar el calendario por medio del navegador web de los dispositivos móviles en una versión reducida y simple, pero de gran utilidad.
- Permite agregar notas o tareas, las cuales son eventos sin un tiempo determinado, las cuales se ubican al lado derecho del calendario y pueden ocultarse si el usuario así lo desea.

Luego, también permite al usuario (previa activación del servicio) colocar una imagen de fondo a la cuadrícula donde se pueden visualizar los eventos.

En el lado de sincronización móvil, Google incorpora “Google Sync” [GOS2010], el cual es una aplicación para los dispositivos móviles que le brinda la capacidad de sincronizar los calendarios disponibles en Google Calendar con el dispositivo móvil en doble sentido, ya sea actualizando el cliente desde el servidor y viceversa. Entre los fabricantes soportados más populares están: BlackBerry, IPhone, Nokia S60 y Windows Mobile.

2.1.1.2 Yahoo Calendar

Servicio ofrecido por Yahoo! [YAH2010], uno de los pioneros en sacar el servicio de calendario online de manera gratuita para los usuarios, permanecía desactualizado desde 1998, sin embargo en los dos últimos años (2008-2010) se ha actualizado considerablemente al punto que su versión que durante este periodo fue beta, ahora ya ha sido relanzada al mercado (gratuito igualmente).

En cuestión de rendimiento, con una interfaz visual basada gran parte en AJAX, no dispone de una vista alterna, con lo cual de no tener una conexión a Internet con una velocidad de descarga adecuada, podría demorarse más de lo esperado para visualizar la página. Así también a esto se le suma que incorpora tantos efecto gráfico que hace el tiempo de visualización aún mayor que el del Google Calendar y el del Windows Live Calendar. Incluso al agregar un evento, es necesario recargar la página web (se realiza automáticamente), a diferencia del Google Calendar que lo maneja en su totalidad por medio de AJAX, lo cual implica la no necesidad de refrescar.

Como característica principal posee la capacidad de agregar tareas o notas, las cuales son eventos sin un tiempo determinado pero que le recuerdan a uno que se tienen cosas pendientes por hacer. Para mejorar esta característica, se posee un panel lateral derecho dedicado exclusivamente para el manejo de las mismas; sin embargo, es un poco pobre en su implementación pues solo permite agregar notas o tareas sin la posibilidad de asignarles horarios específicos (en caso requiera el usuario.

Entre sus otras características se encuentran:

- Compartir calendario con otros usuarios, brindado privilegio para agregar, editar y eliminar eventos.
- Permite tener múltiples calendarios organizados por colores o agregarles "stickers" para diferenciarlos.
- Envío de invitaciones a otros usuarios para un evento en específico con la capacidad de aceptar o declinar el evento.
- Sincronizar los días del mes con las imágenes en el flicker.
- Suscribirse a calendarios remotos y poder ver el calendario de manera pública
- Envío de recordatorios vía correo o mensajes de texto (con un recargo de parte del operador telefónico).

En lo que respecta a su configuración gráfica de la interfaz, el usuario no dispone de ninguna posibilidad de editar o configurar la misma, con lo cual está limitado por el momento a lo que le ofrece Yahoo Calendar.

En cuanto a la sincronización móvil, únicamente soporta iPhone, otros equipos y sistemas operativos serán soportados más adelante según afirma el equipo de Yahoo: "Integration with other mobile phones is coming!" [YAN2010].

2.1.1.3 Windows Live Calendar

Es un sistema de calendario brindado como parte del servicio de Windows Live al usuario que posee una cuenta de correo Windows Live MSN.
Basado en AJAX, el tiempo de descarga del portal web es comparable con el de Google Calendar, por ende mucho más rápido que Yahoo Calenda; así como también nos brinda la posibilidad de visualizar los eventos por día, semana, mes o a manera de lista, un evento tras otro con sus respectivos detalles.

Entre sus características principales está la capacidad de tener una lista de tareas o notas llamadas "To-do List" o "lista de cosas por hacer", en la cual a comparación de Yahoo Calendar y Google Calendar, tiene más funcionalidades, tales como:

- Agregar tareas a calendarios en específicos.
- Agregar prioridad a las tareas.
- Establecer una hora determinada para la tarea.
- Envío del recordatorio de la tarea vía mail (en caso posea fecha y hora) .
- Agregarle una descripción a la tarea.

Tiene a su vez la capacidad de crear múltiples calendarios y poder organizarlos en una barra lateral de manera visual y muy intuitiva, para poder activar o desactivar los que se desean visualizar; así como también agregarle colores a cada uno y poder visualizarlos todos en una sola interfaz visual.
Entre sus características tenemos:

- Compartir calendarios con otros usuarios con una cuenta Windows live Msn, pudiendo otorgar permisos para agregar, editar y eliminar.
- Eventos festivos del país donde uno reside se agregan al calendario si el usuario lo desea para saber en qué fecha se celebra los mismos.
- Los aniversarios y cumpleaños de los contactos de Hotmail se auto-agregan para disponer de éstos de manera automática.
- Integración total con los grupos de Windows Live Msn, con lo cual, para cada grupo se crea un calendario por defecto.
- No dispone de un portal web para ser visitado desde un dispositivo móvil de manera básica y sencilla.
- Importar eventos vía ".ics"
- Capacidad para sincronizarse con el calendario de "Windows Live Mail" y con "Microsoft Office Outlook".
- Envío de recordatorios vía correo electrónico o mensajes de texto (con un recargo de parte del operador telefónico).

En la parte de configuración visual, no implementa ningún servicio para darle privilegios al usuario de manipular el entorno a su gusto, con lo que el usuario debe usar lo que le brinda por defecto Windows Live Calendar.

En términos de sincronización con dispositivos móviles, Windows Live Calendar no dispone hasta la actualidad con un software para sincronizar sus eventos con ningún dispositivo móvil.

2.1.2 Investigación de calendarios online open-source

Como parte del segundo análisis, se analizará los calendarios online de código abierto o software libre disponibles en el mercado. De esta manera, los sistemas de calendarios expuestos a continuación, a excepción de uno, no cuentan con un servicio para el cliente, en donde uno con una cuenta puede hacer uso del calendario; sino más bien, el usuario que desea usarlos se debe descargar el instalar e instalarlo ya sea de manera local o en un servidor público para nivel organizacional.

Los sistemas de calendario a exponerse serán:

1. SOGo
2. WebCalendar
3. Bedework
4. Funambol

2.1.2.1 SOGo

Es un “groupware server” [SOG2010] o en otras palabras, un servidor de muchas aplicaciones concentradas en una sola, llamado SOGo. Brinda los servicios de calendario, agenda de contactos y servidor de correos, todo como un solo programa.

Posee una interfaz web basada en AJAX, a un nivel muy interactivo, pues incluso el botón derecho del clic tiene otras funcionalidades distintas a las normales, tales como crear eventos o editar los mismos. Aún cuando implementa grandes funcionalidades su carga inicial de la página es relativamente rápida pero el desarrollo una vez dentro del programa vía la interfaz web es lento y toma un poco acostumbrarse. Su interfaz se asemeja mucho a un programa instalado en una PC, con una vista rápida en la parte superior y una división para observar el calendario en la parte inferior.

En cuanto a sus características, se tienen las siguientes:

- Lista de notas o tareas en una barra lateral izquierda de fácil acceso y con todas las características iguales a las del Windows Live Calendar (punto

2.1.1.3), por lo cual es un componente bien completo y que se integra de gran manera al calendario, con la posibilidad de adjuntar archivos a las notas.

- Crear calendarios para distintos ocasiones o eventos.
- Compartir calendarios con otros usuarios que posean una cuenta en el mismo sistema SOGo; sin embargo, solo se puede ver el calendario compartido más no dar privilegios de agregar, editar o borrar eventos si uno así lo quisiera.
- Posibilidad de importar eventos para cada calendario independientemente.
- Agregar a cada calendario un color distinto para poder diferenciarlo.
- Importar eventos con extensión ".ics"
- No dispone de un portal web para ser visitado desde un dispositivo móvil de manera básica y sencilla.

En la parte de configuración visual de la interfaz, no implementa ningún servicio para que el usuario pueda configurarla a su gusto, de manera que uno queda limitado a lo que se le ofrece desde el producto; sin embargo, al ser código abierto, es posible editar la hoja de estilos donde reside el calendario.

En términos de sincronización con dispositivos móviles, SOGo como sistema en sí no soporta la sincronización con dispositivos móviles; sin embargo, se puede configurar localmente con el Funambol, otra plataforma de sistemas de calendarios que sí permite sincronizar eventos con dispositivos móviles.

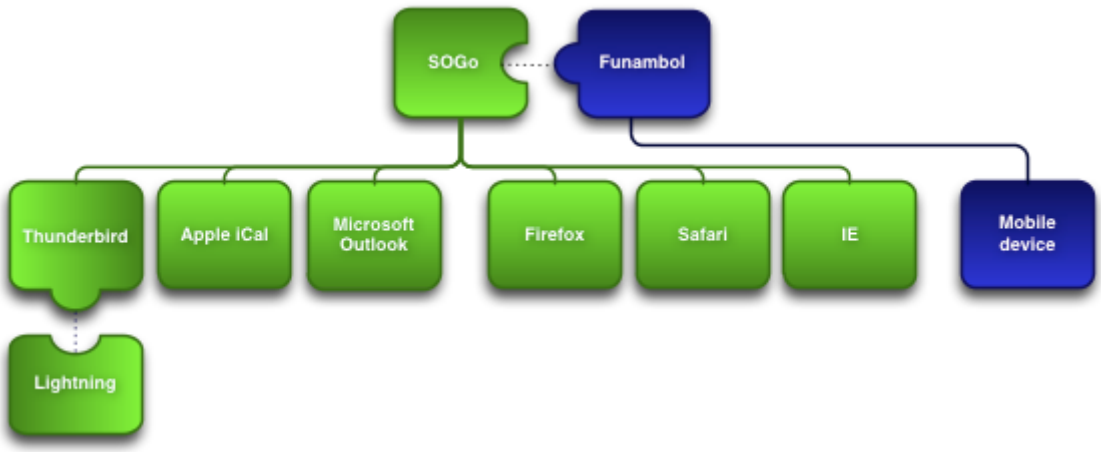

FIGURA 2-1: ARQUITECTURA DE TRABAJO DE SOGO

Fuente: "SOGo overview" [SOV2010]

De esta manera, se puede observar que SOGo realiza la sincronización de contactos y calendarios por medio del Funambol, pues ha logrado una integración con el mismo a nivel de framework, logrando una lógica de analogía entre el protocolo SyncML que utiliza el Funambol para la sincronización y el protocolo CalDAV que utiliza el SOGo.

2.1.2.2 WebCalendar

A diferencia de todos los sistemas expuestos anteriormente, en donde eran clientes de un servicio de correo y un servicio de calendarios (Gmail, Yahoo, Windows Live, SOGo), WebCalendar es una aplicación que brinda un sistema de calendario único, pues no tiene soporte para servidor de correos o libreta de contactos, con lo cual su desarrollo e implementación es netamente sobre un servidor de calendarios online.

Implementado en PHP, con una interfaz en HTML a diferencia de todos los sistemas expuestos anteriormente, posee una carga inicial de la interfaz muy rápida pues no inicia animaciones, ni AJAX, ni servicios adicionales, razón por la cual puede ser considerado el sistema de calendarios más rápido en acceder.

En cuanto a sus características, no posee soporte para crear más de un calendario por usuario, con lo cual se limita a un único calendario pero que trabaja con categorías para poder distinguir entre eventos de distintos índoles u ocasiones.

Entre sus características están: [WEB2010]

- Capacidad de crear eventos e invitar a otros usuarios que posean una cuenta en el servicio de WebCalendar, con la posibilidad de ver si están libres u ocupados durante el periodo en que se planea realizar el evento.
- Envío de recordatorio vía correo electrónico.
- Posibilidad de importar eventos vía un ".ics"
- No posee la capacidad de compartir calendarios con otros usuarios, pero sí de poder ver los calendarios de otros usuarios, siempre y cuando éstos sean de modo público, agregándolos como una "vista", en la cual uno define como desea verlo, ya sea por día, semana o mes y poder asignarle un nombre.
- Ver calendarios juntos con el de uno por medio de "capas" en las cuales se simula como si se estuviera teniendo un solo calendario.
- Capacidad de mostrar los eventos vía un RSS.

- Posibilidad de crear un único calendario público y poder crear eventos en éste con ciertos privilegios.
- Exportar eventos para ICalendar, vCalendar o Palm.
- Configurar el sistema en más de 30 lenguajes.
- Capacidad de tener un usuario administrador para gestionar la totalidad de calendarios.

En cuanto a la interfaz, es completamente configurable para el usuario, desde que color elegir para los bordes de los días, las barras, el fondo del calendario (con algún conocimiento de hoja de estilos), agregar estilos personalizados y elegir elementos para mostrar o no en el sistema, configurado personalmente por cada usuario.

Su sincronización con dispositivos móviles es nula en cualquier aspecto; sin embargo, al estar implementada en PHP y mostrarse en HTML, es posible ver el mismo calendario desde un terminal móvil de manera básica y sencilla.

2.1.2.3 Bedework

Es un sistema de calendario online de código abierto denominado “Enterprise” [BED2010], pues está enfocado a un nivel más corporativo. Escrito en JAVA, tiene la posibilidad de ser embebido en páginas web desarrolladas en HTML o en otras aplicaciones como PHP, ASP, entre otros.

Soporta distintos protocolos e interoperabilidad con los mismos, entre los cuales resaltan: ICalendar, Itip, CalDAV y VVenue. Así mismo, su sistema de calendario funciona en base al protocolo CalDAV, el cual es capaz de operar con una gran gama de clientes locales como ICal, Mozilla Lightning, entre otros.

Entre sus características posee:

- Poder ver los eventos por día, semana o mes.
- Dos niveles de calendarios, público y personal; sin embargo, posee un administrador que se encarga de gestionar los eventos públicos por medio de un portal web dedicado exclusivamente.
- El calendario público puede ser visto sin necesidad de haber ingresado al sistema.

- Los eventos públicos son administrados de manera muy cautelosa, pues solo usuarios con nivel de administrador pueden publicarlos; mientras que los usuarios que no lo son, solo pueden sugerir eventos públicos que serán publicados como tales una vez aprobados por el administrador principal.
- Capacidad de importar eventos en ".ics".
- Capacidad de exportar eventos en ".ics"
- Se puede tener más de un calendario por usuario y de ésta manera decidir en caso se desee cuál de éstos hacer público enteramente.
- Mostrar los eventos como un RSS.
- Capacidad de compartir calendarios con privilegios de crear, editar o eliminar eventos a los usuarios que se les brinde esta capacidad.

En cuanto a la interfaz gráfica, Bedework permite al usuario usar diferentes "skins" los cuales son temas que uno puede elegir, según mejor le parezca a su agrado.

En relación a la sincronización con dispositivos móviles, Bedework no implementa sincronización alguna para algún terminal móvil; ni tampoco presenta la opción de poder visualizarlo vía un explorar móvil, debido a que lo que se observa se nota muy distorsionado y todo el orden se pierde vía un browser móvil.

2.1.2.4 Funambol

Es un servidor de aplicaciones que soporta calendario, agendas de contactos e email usando el protocolo de sincronización SyncML, pero exclusivamente para los dispositivos móviles. Esto quiere decir que el Funambol, al ser instalado en un servidor no brinda una interfaz web para un usuario poder ver sus aplicaciones; sino que por el contrario le permite al usuario brindarle el soporte para éste sincronizar sus eventos desde su equipo móvil hacia el servidor y una vez teniéndolos almacenados ahí, sincronizarlos con algún otro gestor de aplicaciones que permita visualizar estos datos por medio de la web, si así éste lo deseara.

Para un usuario poder hacer uso del servicio Funambol instalado en un servidor, debe de contar con aplicativo en su dispositivo, razón por la cual se muestra la figura 2-2 los diversos terminales soportados:

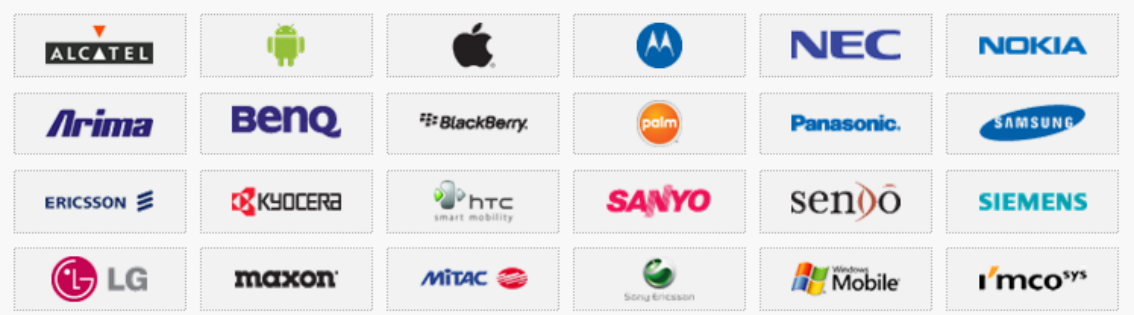

FIGURA 2-2: TERMINALES SOPORTADOS POR FUNAMBOL

Fuente: "Supported devices" [FSD2010]

Tal como se puede observar, Funambol soporta una gran gama de fabricantes y otros sistemas de calendario como Gmail o Yahoo; sin embargo depende mucho qué sistema operativo corre en cada dispositivo para poder saber si es soportado o no, pues los soportados por Funambol son los siguientes: [FUN2010]

- Windows Mobile.- email, contactos, calendario
- Nokia S60.- email, contactos, calendario
- Android.- contactos, calendario
- Symbian.- contacto, calendario
- iPhone.- contactos, calendario

2.2 Selección del sistema de calendarios

Basándonos en los sistemas expuestos anteriormente, se puede llegar a una conclusión: no existe un software open-source que soporte por sí solo: un portal web con lo que se en exige en esta Tesis y a su vez que soporte sincronización móvil.

Debido a esto es que en base a los sistemas de calendarios open-source expuestos anteriormente, se elegirán dos, de los cuales uno será escogido por las funcionalidades, nivel de configuración y libertad que le brinda el usuario para manejar su calendario; mientras que el otro sistema será el que soporte de mejor manera la sincronización con dispositivos móviles.

Una referencia de los calendarios es mostrada en la tabla 2-1:

TABLA 2-1: COMPARACIÓN DE LOS SISTEMAS DE CALENDARIOS

Fuente propia

	Interfaz web	Compartir calendario	Calendario público	Lista de tareas	Varios calendarios	Importar/ exportar eventos	Sincronización móvil
Google Calendar	√	√	√	—	√	√	Gran gama de dispositivos
Yahoo Calendar	√	√	√	√	√	√	Solo iphone
Windows Live Calendar	√	√	—	√	√	√	—
SOGo	√	√	—	√	√	√	vía el Funambol
WebCalendar	√	—	√	—	—	√	—
Bedework	√	√	√	—	√	√	—
Funambol	—	—	—	—	—	—	Gran gama de dispositivos

2.2.1 Análisis y selección del sistema de calendario según su portal web

Teniendo como base los sistemas online open-source descritos en puntos anteriores, según su portal web tenemos al: SOGo, WebCalendar y Bedework.

Entre los 3 disponibles, analizando la factibilidad de implementarse en un servidor para ser usado por la comunidad PUCP, todos ellos soportan conectividad con una amplia gama de bases de datos, dos de ellos están escritos en Java, como son SOGo y Bedework, de los cuales se dispone su código fuente, mientras que el WebCalendar está desarrollado íntegramente en PHP.

La ventaja de tener el WebCalendar en PHP es que se vuelve un entorno más flexible para desarrollar modificaciones, a comparación del código Java. Esto debido a que el código PHP no se compila y se tiene acceso a él directamente; mientras que el código Java, si bien es cierto está disponible para ser editado, se deben configurar los módulos que se desean modificar, para luego tener que compilarlos y subirlos nuevamente al servidor.

En cuanto a funcionalidades, tanto el Bedework como el SOGo presentan mejores prestaciones que el WebCalendar; sin embargo, para las opciones y la aplicaciones que se le desea dar al sistema de calendarios en la presente Tesis, funciones como notas o tener un administrador de los eventos públicos, son prescindibles.

El WebCalendar, Bedework y SOGo tienen la ventaja de poder correr en un servidor local y no tener la necesidad de acceder a Internet para poder hacer uso de ellos, evitando así tráfico saliente en la red y teniendo la ventaja de poder acceder al sistema con una velocidad mucho más alta pues nunca se sale al exterior.

La facilidad con que se puede configurar la interfaz gráfica del WebCalendar, dándole completa libertad al usuario de acondicionarla a su gusto y configurar el estilo, hace que éste, en conjunto con las razones expuestas anteriormente, se convierta en la opción más adecuada para el sistema como parte de la arquitectura cliente servidor.

Cabe mencionar que el SOGo aún teniendo soporte para sincronizar PIM por medio del Funambol, es un sistema mucho más completo y más pesado en términos de consumo de recursos y capacidad, por ende en la presente Tesis se desea implementar únicamente un sistema de calendarios, razón por la cual no se tiene la necesidad de sincronizar otros alcances como correo y contactos.

2.2.2 Análisis y selección del sistema de calendario con soporte para sincronización a dispositivos móviles

En lo que refiere a la selección del sistema de calendario con soporte para sincronización con dispositivos móviles, el único que tiene esta capacidad es el Funambol, razón por la cual es la opción elegida para el sistema en lo que se refiere a la arquitectura cliente servidor, en específico, cliente móvil-servidor.

De esta manera, el Funambol, en conjunto con sus clientes para sincronizar con dispositivos móviles será el sistema elegido para enviar y recibir los eventos ya sea en el dispositivo móvil como en el servidor y éste a su vez enviarlo al WebCalendar de una manera transparente al usuario.

Capítulo 3

Arquitectura de los sistemas

En el siguiente capítulo se explicará la primera parte de la solución para la interconexión de los sistemas, la cual incluye, el modelo de la arquitectura y posteriormente el análisis detallado de los sistemas así como sus respectivos esquemas de base de datos y sus tablas (presentes en el anexo 6 y 7) que van a ser materia de estudio, para ser usadas en la arquitectura del capítulo 4.

Las arquitecturas que se van a analizar son dos: el WebCalendar y el Funambol, ambos sistemas fueron elegidos en el capítulo 2 para implementar la solución. Tal como se mencionó en el capítulo 2.2.1 y 2.2.2, el WebCalendar será el sistema de calendario online web, mientras que el Funambol será usado para la sincronización de los eventos con dispositivos móviles.

La arquitectura planteada para la solución integra ambos sistemas y se disponen tal como se observa en la figura 3-1.

FIGURA 3-1: ARQUITECTURA DEL SISTEMA

Fuente propia

Según el gráfico mostrado en la figura 3-1 el tráfico que se genera entre las peticiones del explorador web (manipulado por el cliente) y el servidor usan el formato html que viaja sobre el protocolo http; mientras que la comunicación que se genera entre el cliente móvil y el servidor, usa el protocolo de sincronización SyncML, el cual es un protocolo que se utiliza indistinto al medio de acceso, como en este caso, la red inalámbrica (wi-fi) o la red celular 3g usada por el dispositivo móvil.

3.1 Arquitectura del WebCalendar

En la arquitectura del WebCalendar se analizarán los siguientes puntos:

- El esquema de la base de datos del mismo.
- El manejo de los usuarios.
- La creación, modificación y eliminación de eventos.

3.1.1 Esquema de la base de datos

El esquema de la base de datos del WebCalendar no se brinda en su página oficial [WEB2010] ni tampoco en algún link buscado en Internet. Sin embargo fue obtenido usando ingeniería inversa por medio del MySQL Workbench en su versión 5.2.

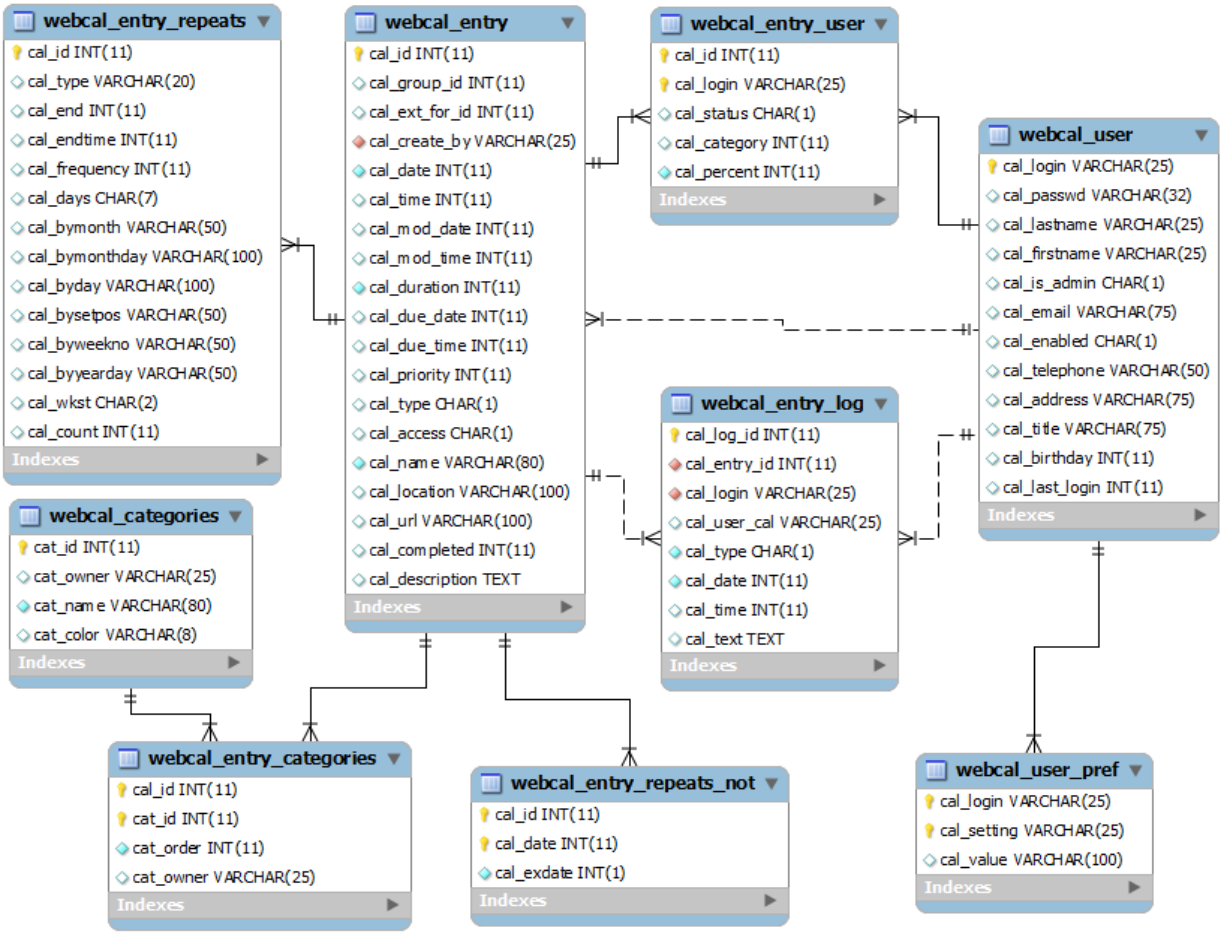

FIGURA 3-2: ESQUEMA SIMPLIFICADO DEL WEBCALENDAR

Fuente: "Tablas del WebCalendar" [WEB2010]

Se observa en la figura 3-2 el esquema de base de datos simplificado, es decir, sólo están presentes las tablas que tienen relevancia para la interconexión del sistema con el Funambol, tanto para los eventos como para los usuarios.

El detalle de cada tabla se encuentra en el anexo 6, siendo de gran importancia su revisión para comprender la función que realiza cada campo de las tablas mostradas en la figura 3-2, como parte de la interconexión de los sistemas.

3.1.2 Manejo de usuarios

La tabla donde se almacena la información de los usuarios del sistema tiene por nombre *'webcal_user'*, donde la contraseña es almacenada usando el algoritmo de encriptación md5.

Según lo predeterminado por el sistema, el único que puede agregar otros usuarios es el administrador general, por medio de la interfaz mostrada en la figura 3-3.

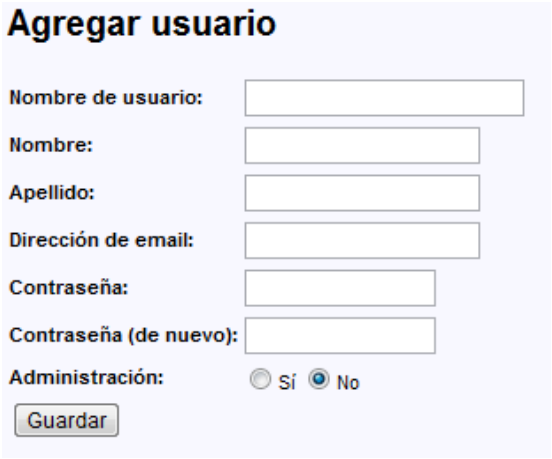

FIGURA 3-3: FORMULARIO DE REGISTRO DE NUEVOS USUARIOS

Fuente propia

De esta forma el usuario es agregado al sistema y listo para usar el mismo; sin embargo, resulta impráctico que cada usuario que desee usar el sistema tenga que solicitárselo al administrador y esperar a que éste le cree una nueva cuenta.

3.1.3 Manejo de eventos

Los eventos en el WebCalendar son de tres tipos:

- **Nuevo evento:** su información es almacenada en la tabla webcal_entry y según la configuración del mismo por parte del usuario, el evento puede tener:
  - Repetición.
  - Recordatorio.
  - Participantes del evento.
  - Categoría a la cual pertenece el evento.

Mediante la figura 3-4 se explica el flujo de creación de un evento:

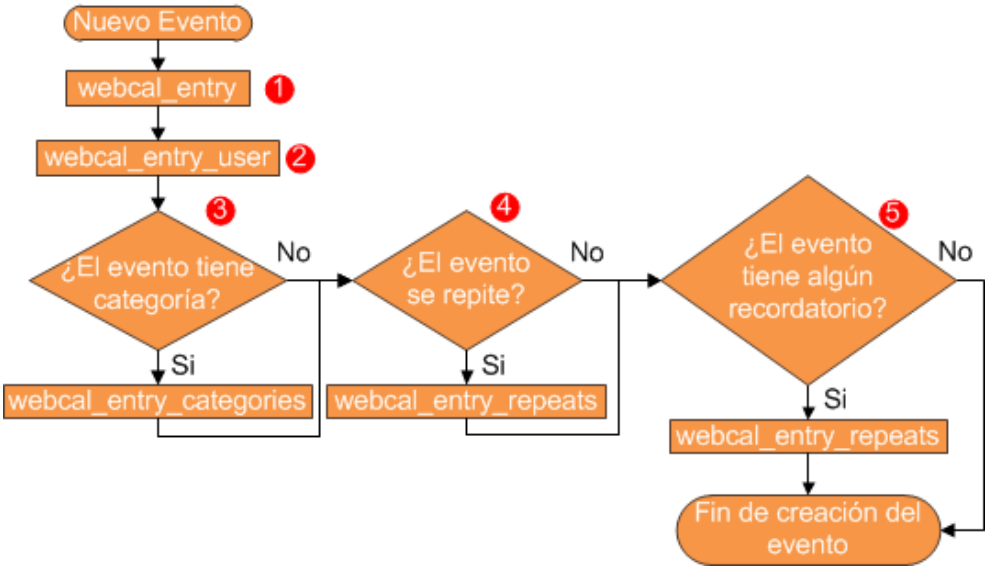

FIGURA 3-4: FLUJO DE CREACIÓN DE UN NUEVO EVENTO

Fuente propia

1. Al crearse un nuevo evento mediante la interfaz web, se genera una entrada en la tabla '*webcal_entry'* con todos los detalles del mismo.
2. Se genera una nueva entrada en la tabla *'webcal_entry_user'* relacionando el usuario que creó el evento (que se encuentra en la tabla '*webcal_user*') y el evento en sí. Ademas aquí se relaciona al evento con los invitados al mismo.
3. Si el evento creado tiene una categoría asociada, la relación evento-categoría se crea como una nueva entrada en la tabla '*webcal_entry_categories*'. Si la categoría ingresada es nueva, previo a la relación evento-categoría, se crea una nueva entrada en la tabla '*webcal_categories'* con la categoría nueva.
4. Si el evento se repite, se crea una nueva entrada en la tabla '*webcal_entry_repeats*'.
5. Por último si el evento posee un recordatorio, el cual le enviará un correo a una hora determinada, se crea una nueva entrada en la tabla '*webcal_reminders'*.

El flujo mostrado en la figura 3-4 es parte de la interconexión de los sistemas, con excepción del punto 5, pues los recordatorios no son parte del desarrollo de la presente Tesis.

- **Edición de eventos:** el WebCalendar no actualiza campo por campo sus eventos haciendo uso de la sentencia sql UPDATE; si no, sigue el siguiente flujo:

1. Guarda temporalmente el identificador único del evento (ID) y el creador del evento.
2. Elimina la entrada correspondiente al evento en la tabla '*webcal_entry'*.
3. Inserta en la tabla '*webcal_entry'* el evento modificado tal como si fuera un nuevo evento, con la diferencia que el ID del evento y su creador son los que fueron almacenados temporalmente en el paso 1.

- **Eliminación de eventos:** se actualiza el campo cal_status de la tabla '*webcal_entry_user'* con el valor de 'D' que implica *"deleted"* o borrado.

3.2 Arquitectura del Funambol

El Funambol sigue una arquitectura cliente-servidor móvil de dos vías. Cuando el cliente (software instalado en el dispositivo móvil) inicia la transferencia de datos, éstos se almacenan en el servidor y a su vez el sistema consulta si existe algún cambio o evento nuevo que notificar al cliente móvil, de ser así, el servidor le envía éstos cambios al cliente móvil usando la misma sesión. Este proceso se conoce como sincronización.

Sin embargo, surge un eventual problema, qué sucede si tanto el servidor como el cliente móvil han modificado el mismo evento y están enviando actualizaciones. Para este caso el administrador del sistema tiene la opción de configurar cuál de los dos tiene la prioridad y por ende cual actualización es aceptada por el servidor y cual es descartada.

3.2.1 Esquema de la base de datos del Funambol

El funambol al ser un groupware, posee una gran cantidad de acciones realizadas sobre la sincronización, para ello requiere un total de 45 tablas, las cuales permiten almacenar:

- Los contactos del dispositivo móvil del usuario.
- Los equipos que se han sincronizado con el sistema.
- Los correos que se han sincronizado.
- Eventos de calendarios creados por los usuarios.
- Notas de los usuarios sincronizados desde el dispositivo móvil.
- Rol del usuario, si es un cliente o un administrador, el cual puede realizar funciones en el servidor.

- Entre otros servicios que no son tema de análisis de la presente tesis.

De las funciones mencionadas anteriormente, sólo se va a trabajar sobre la sincronización de eventos.

El esquema de base de datos completo del Funambol se encuentra en el anexo 5 debido a la gran cantidad de tablas que relaciona y utiliza. Sin embargo, para el manejo de los usuarios el Funambol usa dos tablas:

- fnbl_user.
- fnbl_user_role.

Y para los eventos utiliza otras dos tablas:

- fnbl_pim_calendar.
- fnbl_id.

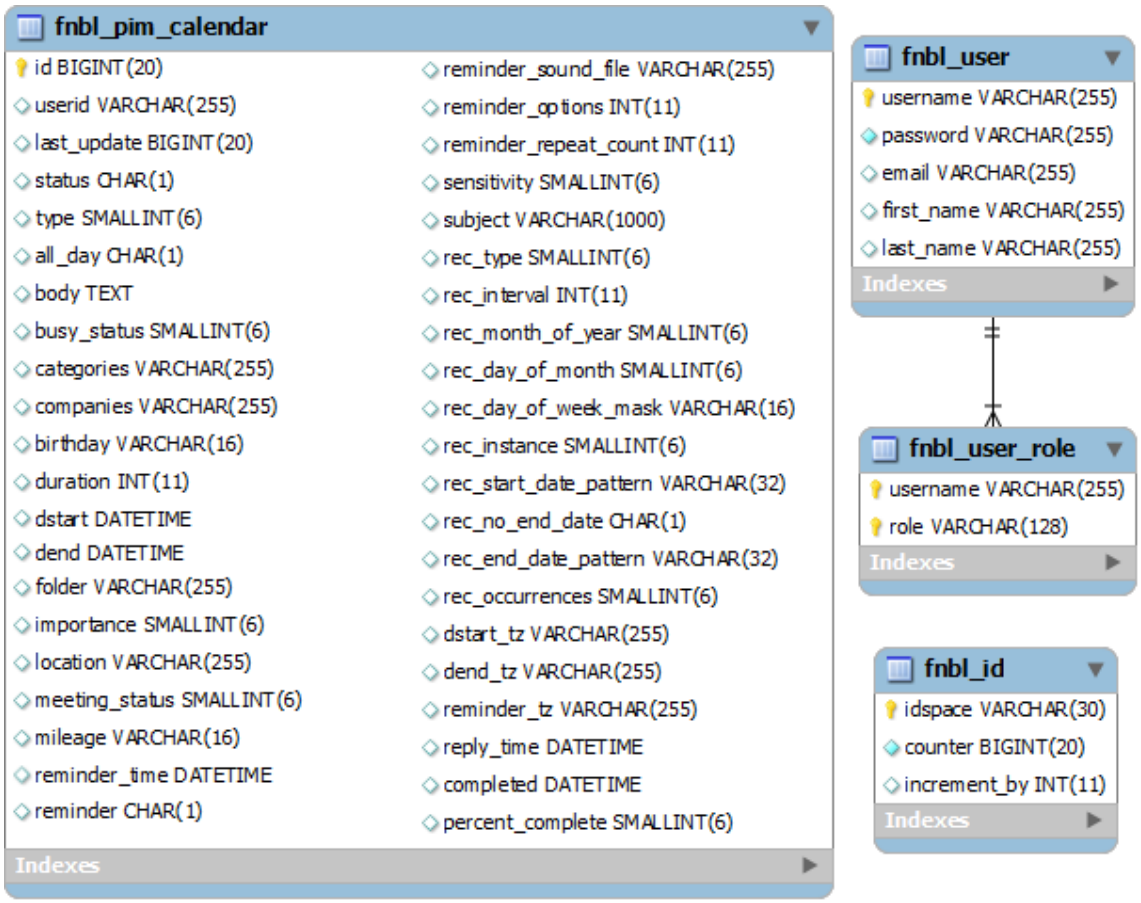

FIGURA 3-5: ESQUEMA SIMPLIFICADO DEL FUNAMBOL

Fuente propia

En la figura 3-5 se muestra el diagrama entidad-relación resumido del esquema de base de datos del Funambol, teniendo en consideración que sólo las cuatro tablas mostradas son estudio de la presente tesis.

El detalle de cada tabla del Funambol mostrada en la figura 3-5, se encuentra en el anexo 7, siendo de gran importancia su revisión para comprender la función que realiza cada campo de las tablas, como parte de la interconexión de los sistemas.

Capítulo 4

Diseño de la arquitectura de la solución

En el siguiente capítulo se explicará el diseño de la arquitectura de la solución, para lograr la integración de ambos sistemas (Funambol y WebCalendar) por medio de sus bases de datos, analizadas en el capítulo 3.
En primer lugar se mostrará la interconexión del manejo de los usuarios para ambos sistemas. Posteriormente se mostrará la arquitectura e interconexión de los eventos creados en ambos esquemas de base de datos, haciendo uso de eventos SQL, procedimientos almacenados y triggers.

Para lograr la interconexión, tanto de los usuarios y los eventos, se ha creado una nueva base de datos llamada 'interconexion' compuesta de dos tablas:

- usuarios registrados.- Usada para la interconexión de los usuarios.
- correspondencia_id.- Usada para la interconexión de los eventos.

El detalle y uso de cada una de las tablas mencionadas se muestra durante el desarrollo del presente capítulo y su código SQL está presente en el CD de anexos.
Es importante mencionar, que ambos sistemas van a ser ejecutados sobre el mismo servidor, por ende sus esquemas de bases de datos residen en la misma ubicación.

4.1 Interconexión de los usuarios

Se va a usar un único portal por el cual cualquier usuario podrá registrarse al sistema, luego se le enviará un correo de confirmación y una vez que confirme su cuenta, estará activo y podrá hacer uso del sistema web y móvil con su mismo nombre de usuario. El código de estas interfaces adicionales está disponible en el CD de anexos.

El flujo desde la página web que sigue el usuario para registrarse en el sistema se muestra en la figura 4-1 y 4-2.

FIGURA 4-1: INTERFAZ DE INGRESO AL WEBCALENDAR

Fuente propia

El usuario al hacer clic en 'aquí' es enviado a la siguiente pantalla (figura 4-2).

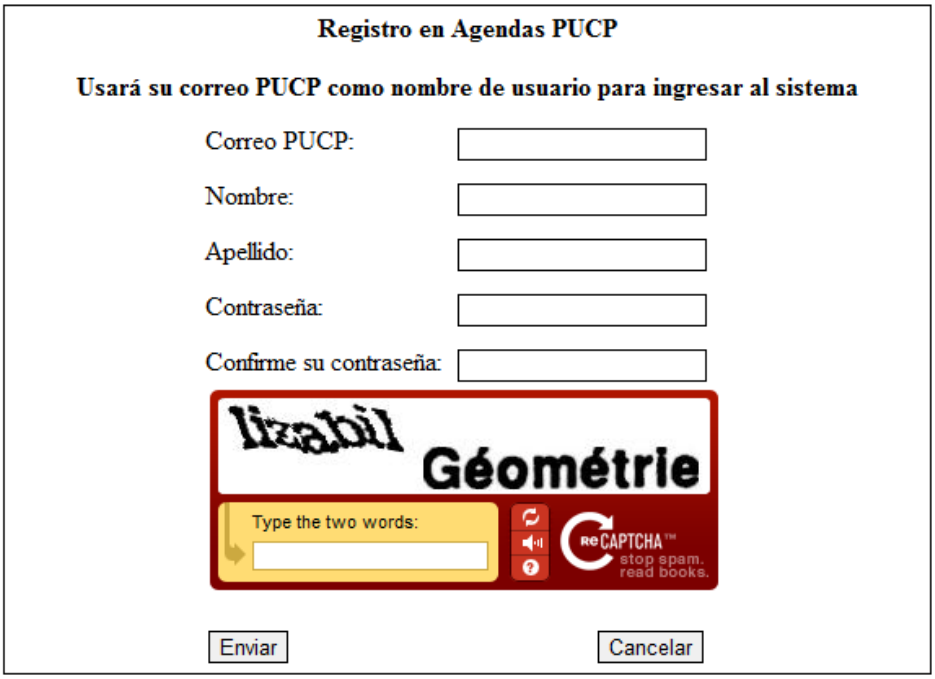

FIGURA 4-2: INTERFAZ DE REGISTRO DE USUARIOS

Fuente propia

La interfaz 4-2 fue creada en PHP con el fin de que el usuario pueda registrarse el sistema de calendarios online y hacer uso del sistema web y el móvil con el mismo nombre de usuario, en este caso, el correo electrónico. La interfaz cuenta además con un captcha brindado por Google [GRL2010] para evitar el spam y el registro de usuarios por medio de bots.

Una vez que el usuario termina de llenar el registro, se le envía un correo de confirmación de cuenta y hasta ese momento el usuario queda para el sistema como registrado pero inactivo. Para lograr la interconexión, los datos que llena el usuario en el formulario son almacenados en la tabla 'usuarios_registrados' de la base de datos interconexión, con el fin que cuando el usuario confirme su cuenta, desde esta tabla internamente el sistema va a crear el usuario correspondiente en el WebCalendar y así mismo el usuario correspondiente en el Funambol.

El esquema de la tabla 'usuarios_registrados' se muestra en la figura 4-3.

TABLA 4-1: TABLA USUARIOS_REGISTRADOS

Fuente propia

Campo	Tipo	Descripción
username	Varchar	Llave primaria. Identificador único de usuario.
last_name	Varchar	Apellidos del usuario.
first_name	Varchar	Nombres del usuario.
passwd_webcal	Varchar	Password del usuario en formato WebCalendar (md5).
passwd_funam	Varchar	Password del usuario en formato Funambol (3DES).
email	Varchar	Correo del usuario.
estado	Char	Estado del usuario para el sistema: • 'A' = usuario activo, confirmó su cuenta. • 'I' = usuario inactivo, no confirmó su cuenta.

Cuando el usuario confirma su cuenta por medio del correo que se le envió, el sistema internamente realiza tres acciones:

1. Cambia el el valor del campo 'estado' de la tabla 'usuarios_registrados' de 'I' a 'A', es decir, el usuario pasa a estar activo.
2. Se inserta en la tabla 'webcal_user' de la base de datos del WebCalendar los valores correspodientes almacenados en la tabla 'usuarios_registrados', tal como se muestra en la figura 4-3. Se debe llenar un solo campo adicional, 'cal_is_admin', con el valor 'N' pues el usuario no es administrador. Se entiende que la contraseña almacenada en el campo 'passwd_webcal' ya esta

encriptada en md5, por lo que no se realiza mayor operación al insertar en la tabla webcal_user.

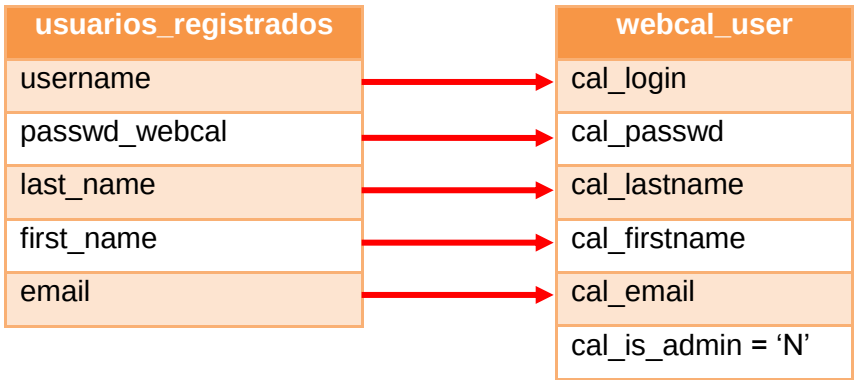

FIGURA 4-3: REGISTRO DE UN USUARIO WEBCALENDAR

Fuente propia

3. Se inserta en la tabla 'fnbl_user' de la base de datos del Funambol los valores correspondientes almacenados en la tabla 'usuarios_registrados', tal como se muestra en la figura 4-4. De la misma manera que para el punto 2, el password del Funambol ya se encuentra encriptado en 3DES y no es necesario ningún procesamiento para su insetar en la tabla destino.
   También es necesario crear una nueva entrada con el usuario registrado en la tabla 'fnbl_user_role' para indicar el rol del usuario en el Funambol, que por defecto va a ser 'sync_user', es decir, un usuario con privilegios solamente de uso del sistema.

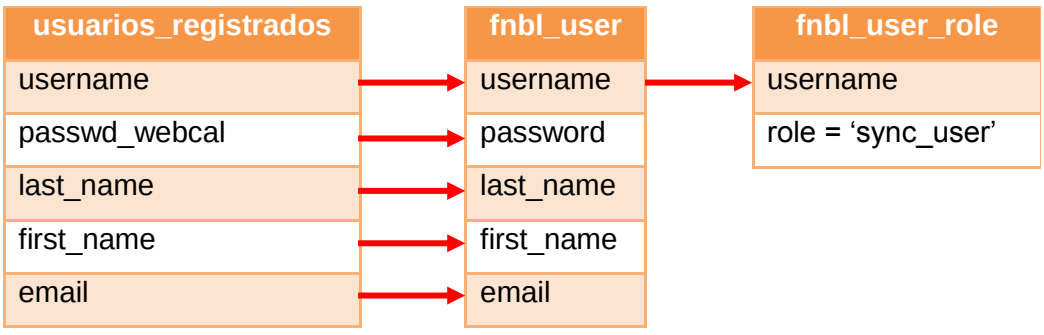

FIGURA 4-4: REGISTRO DE UN USUARIO FUNAMBOL

Fuente propia

4.1.1 Interfaces y módulos adicionales

Como parte del desarrollo del registro de usuarios, también se crearon dos interfaces y un módulo adicional para el sistema.

4.1.1.1 Módulo de envío de correos

Tal como se mencionó en el diseño de la interfaz de registro, es necesario enviar un correo al usuario de manera que éste pueda activar su cuenta. En este sentido, PHP posee funciones capaces de realizar el envío de correo; sin embargo, la configuración de estas librerías internas de PHP suelen traer consigo problemas como:

- Incompatibilidad de caracteres especiales como “ñ”, tildes y diéresis.
- Error en el envío de correo desde el servidor.
- Falta de información en el correo recibido por el usuario.

Por esta razón se optó por usar una librería externa para el envío de correos llamada PHPMailer. [PHM2010]
Entre las ventajas de usar PHPMailer se tiene:

- Enviar correos de una manera muy intuitiva.
- Enviar correos con ficheros adjuntos.
- Enviar correo con formato HTML.

Éste módulo de envío de correos es usado tanto para el registro de usuarios como para la recuperación de contraseña.

4.1.1.2 Interfaz de recuperación de contraseña

Se ha creado una interfaz de recuperación de contraseña desde la pantalla principal de logeo (figura 4-1), con la finalidad de permitirle al usuario poder obtener una nueva contraseña en caso haya perdido u olvidado la suya.

En la figura 4-5 se muestra la lógica para la recuperación de contraseña de parte del usuario; mientras que en la figura 4-6 se puede observar la interfaz web que se ha diseñado. Nuevamente, se hace uso del captcha [GRL2010] para evitar los bots y spam. El dirección que se envía en el correo electrónico al usuario es válida una sola vez, es decir, luego que el usuario recupere su contraseña, posteriormente dejara de funcionar por motivos de seguridad.

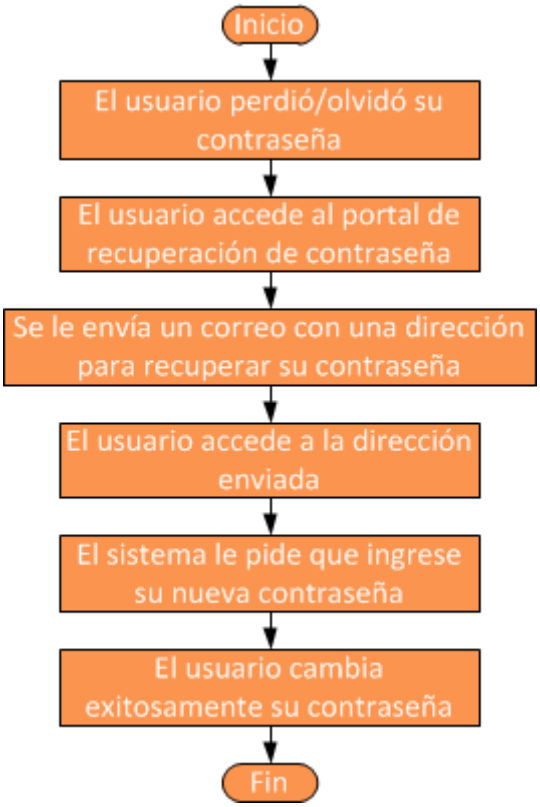

FIGURA 4-5: LOGICA PARA LA RECUPERACIÓN DE CONTRASEÑA

Fuente propia

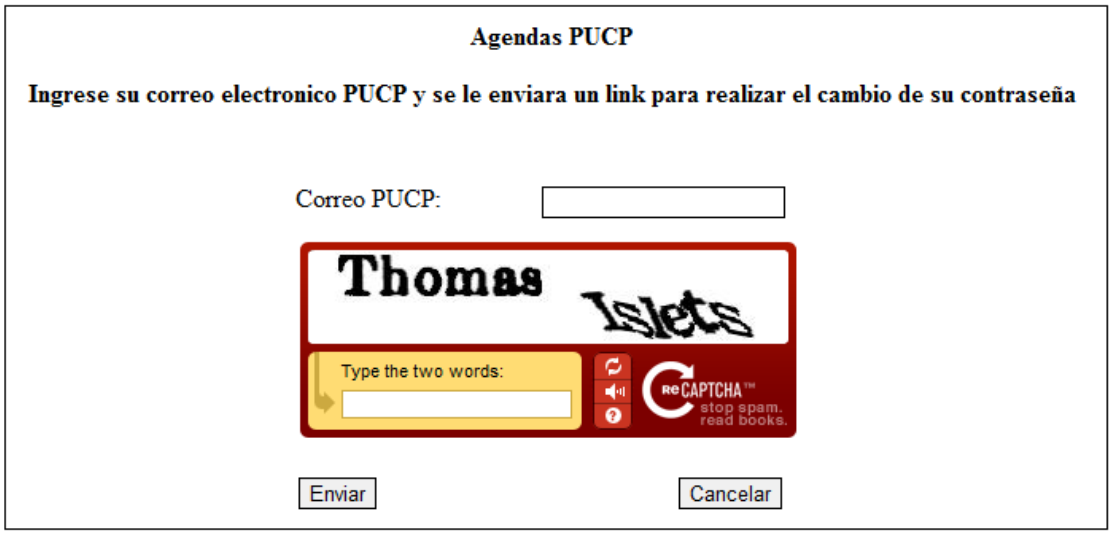

FIGURA 4-6: INTERFAZ PARA LA RECUPERACIÓN DE CONTRASEÑA

Fuente propia

4.1.1.3 Interfaz de mantenimiento de usuarios

Como parte de la interconexión de usuarios, se ha creado una interfaz de mantenimiento de los usuarios registrados en el sistema, estén activos o inactivos. La ventaja de esta interfaz es que permite tener un control centralizado de los usuarios de ambos sistemas (Funambol y WebCalendar) y cualquier modificación se realizará en sus respectivas bases de datos.

Para ingresar a la interfaz de administración, se ha protegido la carpeta por medio de una combinación de archivos '.htaccess' y '.htpasswd', para lograr la protección de la carpeta de archivos así como también permitir el logeo de un único usuario administrador.

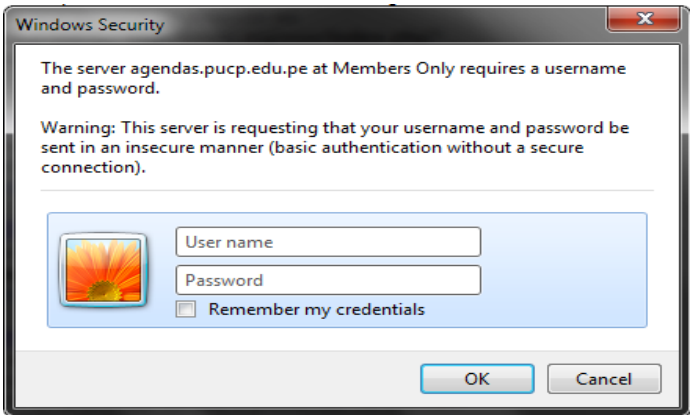

FIGURA 4-7: INGRESO AL MANTENIMIENTO DE USUARIOS

Fuente propia

Así mismo una vez dentro se observa en la figura 4-8 la interfaz de mantenimiento de usuarios.

FIGURA 4-8: INTERFAZ DE ADMINISTRADOR DE USUARIOS

Fuente propia

La interfaz cuenta con un buscador de usuarios, así como una barra con información acerca de los usuarios totales en el sistema, los activos, los inactivos y además una

opción que le permite al usuario administrador, reenviar el correo de activación de cuenta y de recuperación de contraseña en caso el envió hubiera fallado o si el usuario se contactó porque nunca le llegó el correo. Así mismo le permite al usuario administrador cambiar el correo del usuario por si éste así lo solicita.

4.2 Interconexión de los eventos

La interconexión de los eventos se realiza por medio de la arquitectura mostrada en la figura 4-9.

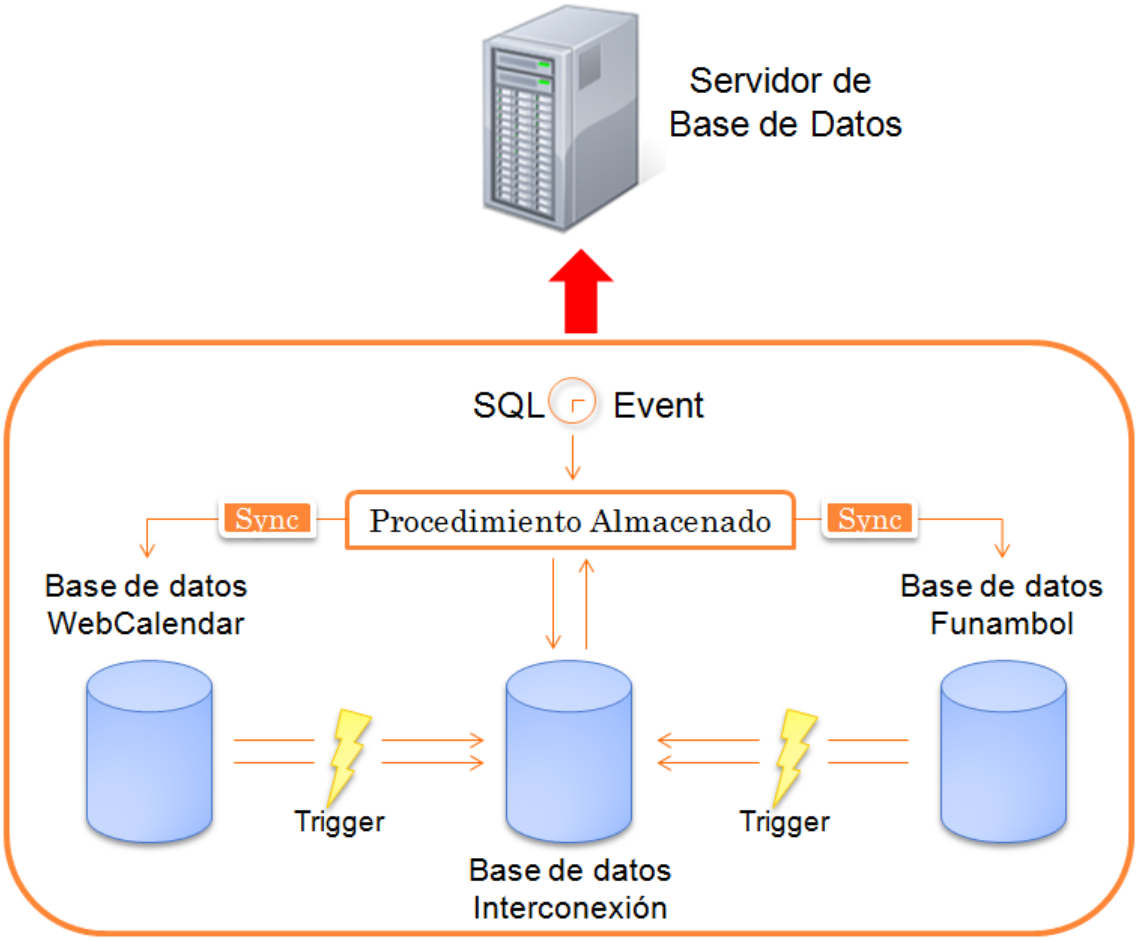

FIGURA 4-9: ARQUITECTURA DE LA INTERCONEXIÓN DE EVENTOS

Fuente Propia

Entonces, la solución radica básicamente en dos puntos principales. El primero son los triggers de interconexión, los cuales van a almacenar en la tabla 'correspondecia_id' de la base de datos 'interconexión', los IDs generados en ambos sistemas. El segundo punto es un evento SQL el cual se va a encargar de ejecutar un procedimiento almacenado el cual va a realizar todo el procesamiento de los eventos y

posteriormente la sincronización de los mismo y así lograr la interconexión de los sistemas.

Para realizar la interconexión, se hace uso de la tabla mencionada anteriormente 'correspondencia_id' la cual tiene el esquema mostrado en la tabla 4-2.

TABLA 4-2: TABLA CORRESPONDENCIA_ID

Fuente propia

Campo	Tipo	Descripción
web_ID	Int	ID del evento de la tabla 'webcal_entry' de la base de datos del WebCalendar.
fun_ID	Bigint	ID del evento de la tabla 'fnbl_pim_calendar' de la base de datos del Funambol.
accion	Varchar	Acción a realizar cuando se ejecute el evento SQL. • NEW.- Crea el evento. • UPD.- Actualiza el evento. • DEL.- Borra el evento. Para cada caso se apuntará a la base de datos especificada en el campo 'sync_to'.
sync_to	Varchar	Esquema de base de datos hacia donde se va a realizar la sincronización.
estado	Varchar	Estado de la sincronización. • NOT_SYNC.- el evento no ha sido sincronizado. • SYNC.- el evento ya ha sido sincronizado.

La razón por la cual se tiene en una misma tabla el ID del Funambol y el ID del WebCalendar es porque para cada evento creado en el Funambol, existe su igual creado en el WebCalendar y viceversa, esto como parte de la interconexión. Entonces esta tabla hace referencia tiene una correspondencia entre un evento creado en un sistema y su par o igual en el otro sistema. El uso y detalle de los demás campos se realizará en los dos siguientes subcapítulos (4.2.1 y 4.2.2).

4.2.1 Triggers de Interconexión

Los triggers son la primera parte esencial del desarrollo de la interconexión de los sistemas, debido a que son usados en la presente tesis para mantener actualizada la tabla 'correspondencia_id' ante la creación, edición o borrado de un evento. Las sentencias SQL que describen los triggers se encuentra en el anexo 8.

La tabla ‘correspondencia_id’ mostrada en el tabla 4-2, tiene los siguientes campos que van a ser manipulados por los triggers de la siguiente manera:

- Web_ID.- Hace referencia al ID del evento en el WebCalendar.
- Fun_ID.- Hace referencia al ID del evento en el Funambol.
- Acción.- Se refiere a la acción que se va a ejecutar cuando se sincronicen los eventos de una base de datos hacia la otra. Tiene tres posibilidades:
    - NEW.- el evento es nuevo y debe crearse en la base de datos indicada en el campo ‘sync_to.
    - UPD.- se debe actualizar el evento hacia la base de datos indicada en el campo ‘sync_to’. Es decir, si en el campo ‘accion’ su valor es ‘UPD’ y en el campo ‘sync_to’ el valor es ‘FUN’, se actualizará el evento con la información del WebCalendar en la base de datos del Funambol.
    - DEL.- se debe realizar un borrado lógico en la base de datos indicada en el campo ‘sync_to’.
- Sync_to.- se refiere a la base de datos hacia donde se va a realizar la sincronización. Tiene dos posibles valores:
    - ‘FUN’.- Hace referencia que la sincronización del evento mediante la acción (‘NEW’,’UPD’ o ‘DEL’) se va a realizar hacia el Funambol.
    - ‘WEB’.- Hace referencia que la sincronización del evento se va a realizar hacia el WebCalendar.

  El campo ‘sync_to’ funciona siempre en conjunto con el campo ‘accion’ mencionado en el punto anterior.
- Estado.- Se refiere al estado de la sincronización, en otras palabras, si es que lo indicado en el conjunto ‘accion’-‘sync_to’ ya se ha realizado o todavía se está a la espera de su sincronización. Tiene dos opciones:
    - ‘NOT_SYNC’.- Hace referencia a que la acción (‘accion’) no ha sido realizada aún hacia la base de datos destino (‘sync_to’).

Tanto el WebCalendar como el Funambol poseen cada uno dos triggers para manejar los tres tipos de acciones que se presentan en el sistema (creación, edición y eliminación de un evento).

4.2.1.1 Triggers en el WebCalendar

Son tres opciones las que deben disparar el trigger asociado para informar a la tabla ‘correspondencia_id’ sobre el nuevo suceso. Estas son:

1. Creación de un evento.
2. Edición de un evento.
3. Borrado de un evento.

El WebCalendar crea un nuevo evento insertando una nueva fila en la tabla 'webcal_entry' y relacionando el evento creado con su usuario creador en la tabla 'webcal_entry_user'. Para actualizar un evento el WebCalendar no realiza una sentencia SQL UPDATE, si no que almacena el ID del evento que se desea actualizar, borrar toda la fila correspondiente a ese ID y vuelve a insertar la fila, con el ID previamente guardado pero con los nuevos campos que son en realidad, los actualizados.

Entonces podemos agrupar los triggers del WebCalendar en dos grupos:

1. **Trigger de creación y actualización de eventos.-** Cuando se crea un nuevo evento desde la interfaz web, se inserta una nueva entrada en la tabla 'webcal_entry' y en la tabla 'webcal_entry_user' donde se relaciona el usuario creador del evento y el evento en sí. Por otro lado, cuando se actualiza un evento, se almacena temporalmente el ID de ese evento, se borra la entrada correspondiente a ese ID y finalmente inserta una nueva entrada usando el ID que fue guardado previamente, con la diferencia que esta nueva entrada tiene los campos actualizados.

   Entonces, se ha asociado un trigger a la sentencia INSERT en MySQL sobre la tabla 'webcal_entry_user', el cual se encarga de dos operaciones:

   a. Insertar una entrada en la tabla 'correspondencia_id' con el ID del nuevo evento del WebCalendar en el campo 'web_ID', el ID del evento Funambol (fun_ID) en nulo pues todavía no se ha creado en la otra base de datos, tal como se muestra en la tabla 4-3. La frase **{nuevo id}** se refiere al ID de nuevo evento.

TABLA 4-3: NUEVO EVENTO EN EL WEBCALENDAR

Fuente propia

web_ID	fun_ID	accion	sync_to	estado
{nuevo id}	null	NEW	FUN	NOT_SYNC

   b. Actualizar la entrada correspondiente al ID en cuestión, es decir, si el trigger se dispara y el id del nuevo evento ya existe en la tabla

'correspondencia_id', entonces sólo se actualiza el campo 'accion', 'sync_to' y 'estado' tal como se observa en la tabla 4-4.

TABLA 4-4: ACTUALIZACIÓN DE EVENTO EN EL WEBCALENDAR

Fuente propia

web_ID	fun_ID	accion	sync_to	estado
{ID}	null	UPD	FUN	NOT_SYNC

Se debe mencionar que en la tabla 4-4 el campo 'fun_ID' continua estado con el valor de nulo, lo cual es y no cierto dependiendo del última sincronización del sistema. Por ejemplo, un evento recién creado se observa tal y como está en la tabla 4-3, posteriormente existen dos posibilidades. La primera es que el sistema se sincronizó, entonces el campo 'fun_ID' va a tener un valor, pues en la base de datos del Funambol ya se ha recreado el evento desde el WebCalendar y existe por consiguiente el ID correspondiente. Entonces al hacerse un update del evento el campo 'fun_ID' ya estaría lleno. Sin embargo la otra posibilidad es que el usuario actualice el evento recién creado, entonces puede ser que el sistema todavía no se haya sincronizado y por consiguiente el campo 'fun_ID' seguiría estando con el valor de nulo.
En ambos casos la sincronización de eventos continúa funcionando.

2. **Trigger de eliminación de eventos.-** Cuando se elimina un evento en el WebCalendar, se realiza un borrado lógico en su base de datos, actualizando el campo 'cal_status' de la tabla 'webcal_entry_user' con el valor 'D'. El trigger asociado a esta tabla se dispara ante una sentencia SQL UPDATE y actualiza la entrada correspondiente en la tabla 'correspondencia_id' tal como se muestra en la tabla 4-5. La frase **{ID}** hace referencia el ID del evento del WebCalendar que acaba de ser borrado. Cabe resaltar nuevamente que el valor del campo 'fun_ID' con nulo depende si el evento se ha sincronizado previamente o no, pues de ser así, el valor del campo sería el ID del evento correspondiente en el Funambol.

TABLA 4-5: ELIMINACIÓN DE EVENTO EN EL WEBCALENDAR

Fuente propia

web_ID	fun_ID	Acción	sync_to	estado
{ID}	Null	DEL	FUN	NOT_SYNC

4.2.1.2 Triggers en el Funambol

Al igual que para el WebCalendar, son tres las opciones que deben disparar el trigger asociado para informar a la tabla 'correspondencia_id' sobre el nuevo suceso.

1. Creación de un evento.
2. Edición de un evento.
3. Borrado de un evento.

El Funambol crea un evento insertando un una nueva fila en la tabla 'fnbl_pin_calendar', mientras que para realizar la actualización y borrado de un evento, realiza una sentencia SQL UPDATE sobre la misma tabla pero alterando un campo en específico ('status').

Al igual que el WebCalendar, podemos agrupar los triggers del Funambol en dos grupos:

1. **Trigger de creación de eventos.-** Cuando se crea un nuevo evento desde el dispositivo móvil, éste al sincronizarlo con el servidor Funambol inserta en su base de datos una nueva entrada para la tabla 'fnbl_pim_calendar'. Entonces el trigger asociado a ésta tabla se disparará ante una sentencia SQL INSERT e insertará en la tabla 'correspondencia_id' una fila con el nuevo ID del evento Funambol y los demás parámetros tal como se muestran en la tabla 4-6, donde la frase **{nuevo id}** es el ID del evento recién creado en la base de datos del Funambol.

TABLA 4-6: NUEVO EVENTO EN EL FUNAMBOL

Fuente propia

web_ID	fun_ID	accion	sync_to	estado
null	**{nuevo id}**	NEW	WEB	NOT_SYNC

2. **Trigger de actualización y eliminación de eventos.-** Cuando el Funambol realiza la actualización o el borrado de un evento, en ambos casos realiza una actualización sobre la tabla 'fnbl_pim_calendar'. Ante esto, se ha asociado un trigger a la tabla 'fnbl_pim_calendar' que se dispara ante una sentencia SQL UPDATE. El trigger verifica si el campo 'estado' ha cambiado su valor a 'D', de ser así, el evento se ha eliminado y se actualiza la entrada correspondiente en la tabla

'correspondencia_id', con los valores mostrados en la tabla 4-7. La frase **{ID}** hace referencia al ID del evento del Funambol creado previamente.

TABLA 4-7: ELIMINACIÓN DE EVENTO EN EL FUNAMBOL

Fuente propia

web_ID	fun_ID	accion	sync_to	estado
null	**{ID}**	DEL	WEB	NOT_SYNC

De la misma forma, si el trigger asociado verifica que el campo 'estado' no se ha modificado (conserva su valor anterior), entonces el evento ha sido solamente actualizado desde el dispositivo móvil y procede a actualizar la tabla 'correspondencia_id' con los valores mostrados en la tabla 4-8.

TABLA 4-8: ACTUALIZACIÓN DE EVENTO EN EL FUNAMBOL

Fuente propia

web_ID	fun_ID	accion	sync_to	estado
null	**{ID}**	UPD	WEB	NOT_SYNC

4.2.2 Evento SQL

El core de la tesis es la interconexión de los sistemas, con el fin de brindar un único servicio, lo cual es logrado por medio de los triggers (expuestos en el punto 4.2.1) en conjunto con un evento SQL, el cual va a ejecutar un procedimiento almacenado que es el encargado de realizar la completa sincronización de eventos de una base de datos hacia otra y viceversa.

Entonces, se ha programado un evento SQL con el nombre de 'Sync_Calendars' que se ejecuta **cada minuto** sin un tiempo de fin, con el objetivo de que siempre se esté ejecutando el evento SQL sobre la base de datos 'interconexion' y sincronizando las bases de datos del Funambol y del WebCalendar.

La única operación que realiza el evento SQL es llamar al procedimiento almacenado con nombre 'Sync_Events'. Recordando el capitulo 1.3.2.3, para que el planificados de eventos funcione en MySQL y así poder crear eventos SQL, la variable global 'event_scheduler' debe estar 'ON' en el archivo de configuración MySQL correspondiente. El detalle de la sentencia del evento SQL se encuentra en el anexo 9.

4.2.3 Procedimiento almacenado para la sincronización

El SQL event 'Sync_Calendars' ejecuta el procedimiento almacenado encargado de la sincronización de los eventos llamado 'Sync_Events'. El código SQL de todos los procedimientos almacenados detallados en la presente tesis se encuentran en el CD de anexos; sin embargo, en el anexo 10 se muestran un pequeño resumen de todos los procedimientos y subprocedimientos almacenados utilizados en la presente tesis.
El flujo seguido para este procedimiento almacenado es el mostrado en la figura 4-10.

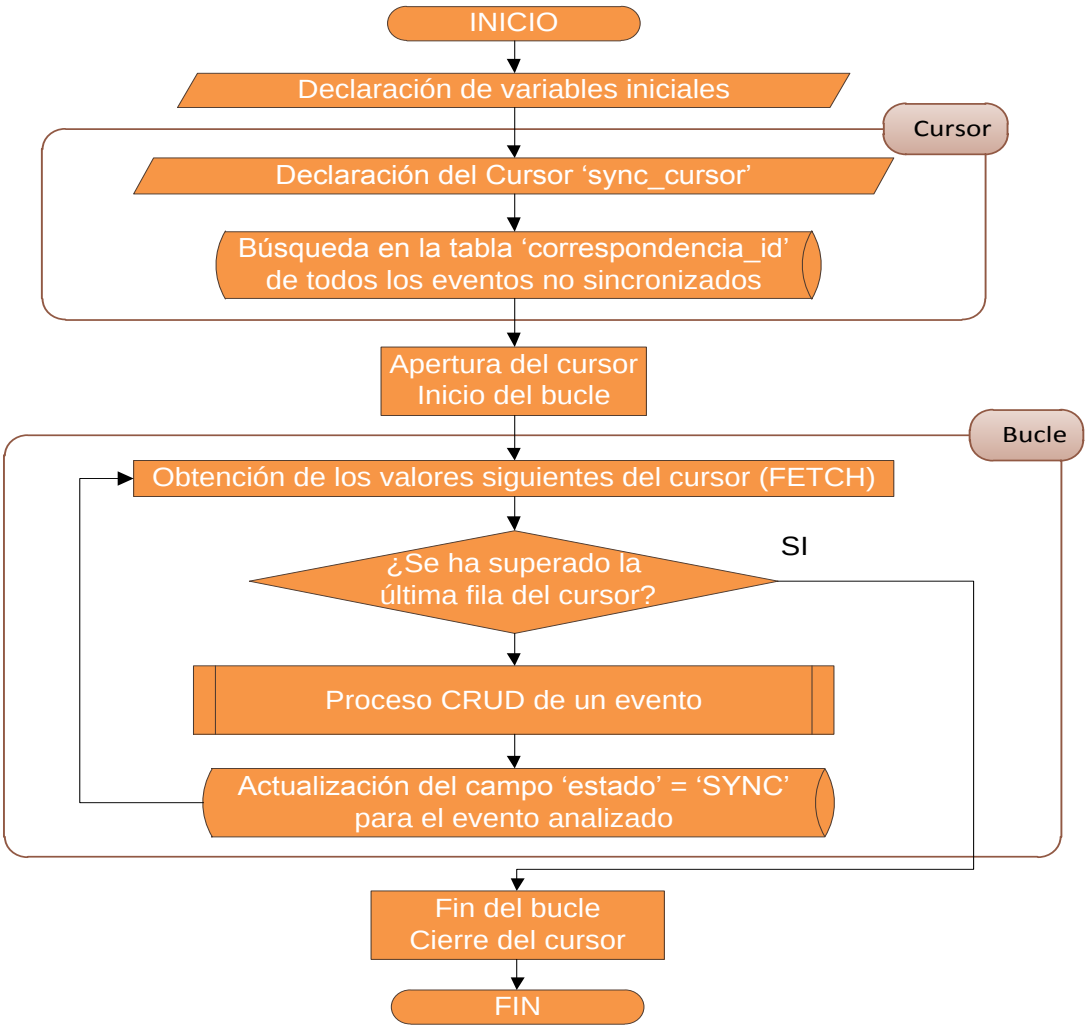

FIGURA 4-10: PROCEDIMIENTO ALMACENADO SYNC_EVENTS

Fuente Propia

Como inicio del procedimiento almacenado, se declaran 6 variables, las cuatro primeras van a tener el valor del resultado de cada registro del cursor, la variable siguiente se utilizará para manejar el fin del cursor a declararse y la última variable es el cursor en sí. Las cinco primeras variables tienen como valor inicial NULL.

- **web_id_val**.- valor del ID del evento en el WebCalendar, campo 'web_ID'.
- **fun_id_val**.- valor del ID del evento en el Funambol, campo 'fun_ID'.
- **acción_val**.- valor del campo 'accion'.
- **sync_to_val**.- valor del campo 'sync_to'.
- **last_row**.- variable usada para manejar el fin del cursor.

Posteriormente a la declaración de las variables iniciales, se declara la variable del cursor con nombre '**sync_cursor'**. El cursor va a almacenar la búsqueda sobre la tabla 'correspondencia_id' de la base de datos 'interconexion' que cumple con la condición de seleccionar todos los registros que tengan el campo 'estado' igual a 'NOT_SYNC'. Así pues, el cursor va a poseer todos los eventos de un sistema o del otro que no están sincronizados y deben sincronizarse.

Se procede a aperturar el cursor y a dar inicio al bucle. Para cada iteración se realiza la sentencia SQL 'FETCH' para obtener el valor de todos los campos de un solo registro (o fila). La asignación de campos en variables se muestra en la figura 4-11.

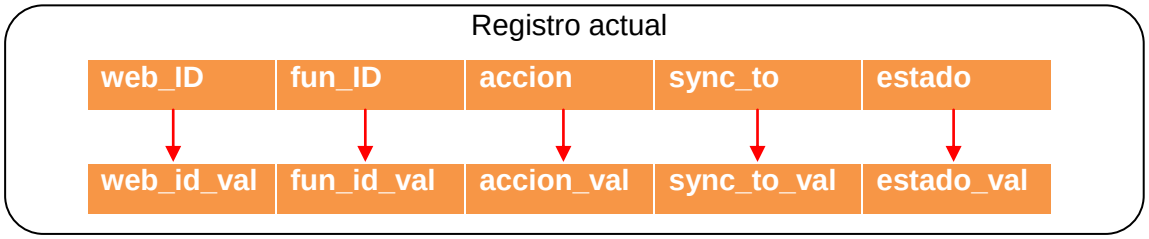

FIGURA 4-11: ASIGNACIÓN DEL REGISTRO ACTUAL EN VARIABLES

Fuente Propia

En el proceso de obtención del registro, cabe la posibilidad que el cursor haya llegado a la siguiente posición de su último registro, por lo cual no hay más valores disponibles que obtener y dispara la condición '02000' o 'NOT FOUND', la cual setea la variable 'last_row' (definida previamente) en 1. Es por esta razón que cada vez que se inicia una iteración se debe verificar que el valor de la variable 'last_row' sea igual a 0, pues si su valor es 1, el cursor ha llegado a su fin y se debe proceder a salir del bucle.

Si la variable 'last_row' es igual a 0, según la figura 4-10, se realiza el *'Proceso CRUD de un evento*', el cual es encargado de realizar las operaciones necesarias para la sincronización de eventos entre las bases de datos. El detalle del mismo se realiza en el capítulo siguiente (4.2.4) así como también todos sus subprocesos.

4.2.4 Proceso crud de un evento

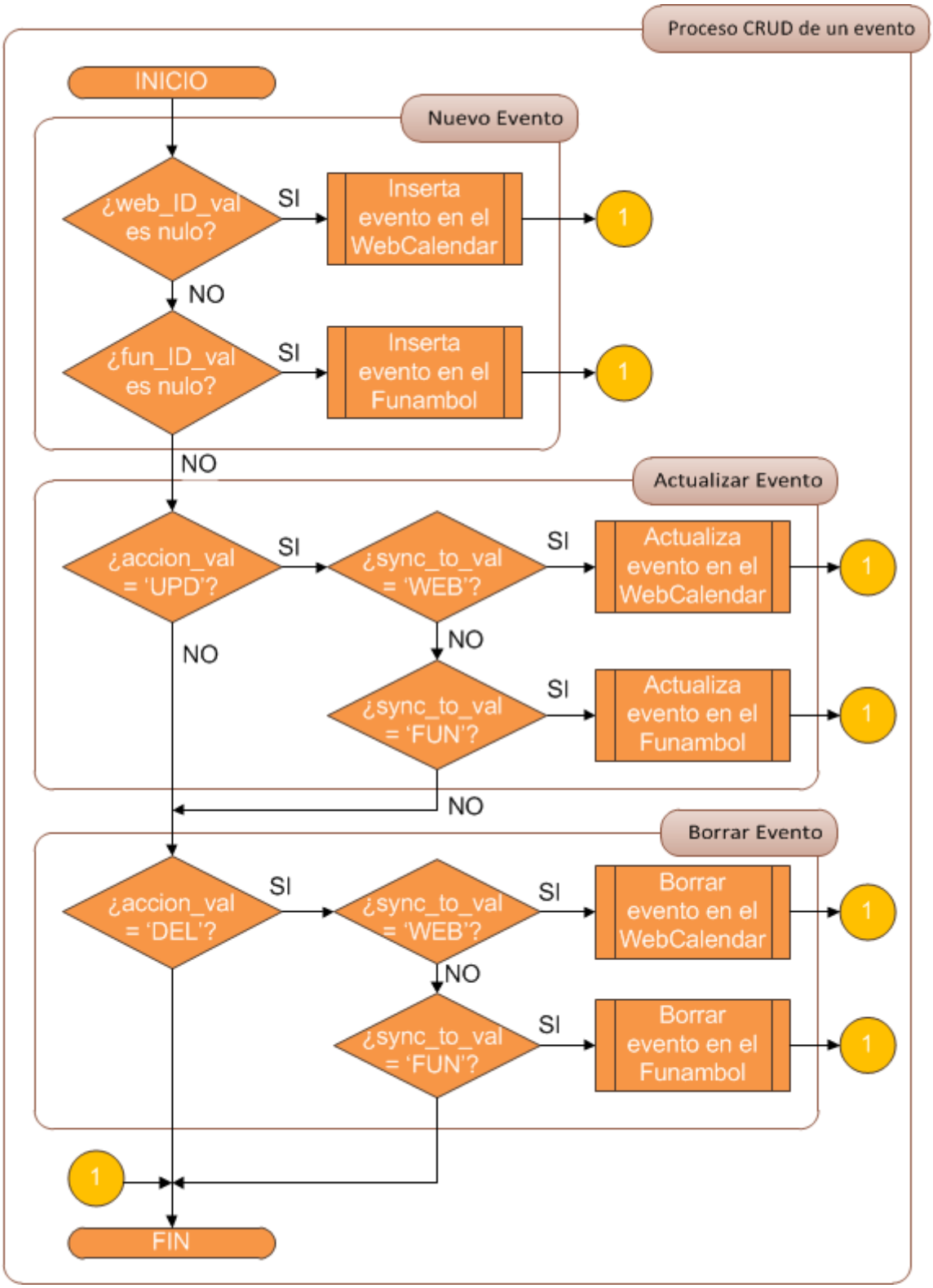

FIGURA 4-12: PROCESO DE CREACIÓN, ACTUALIZACIÓN Y BORRADO DE EVENTO

Fuente Propia

El nombre CRUD del proceso viene del acrónimo en ingles ‘Create, Read, Update and Delete’ usado para referencias sobre bases de datos para ‘Crear, Leer, Actualización y borrar’. Para el proceso CRUD mostrado en la figura 4-12, se realiza una Lectura de los datos de los eventos y sobre los cuales se ejecuta una creación, actualización o borrado en las tablas pertinentes.

Cuando se inician las validaciones de creación, actualización o borrado de eventos, en primer lugar siempre se analiza si el valor las variables ‘web_ID_val’ o ‘fun_ID_val’ tienen el valor de nulo como parte de la validación de un nuevo evento. El motivo de realizar ésta validación y no verificar solamente si el campo ‘accion’ es igual a ‘NEW’, es porque si un evento recién creado, es actualizado o incluso borrado en ese mismo instante, el ID del sistema contrario todavía seguiría estado en nulo causando un error. En la figura 4-13, se simula la creación de un evento en el WebCalendar con ID ‘*123*’ que es sincronizado de su base de datos hacia la tabla ‘correspondencia_id’ por medio de los triggers, en conjunto con la ejecución del evento SQL programado cada minuto.

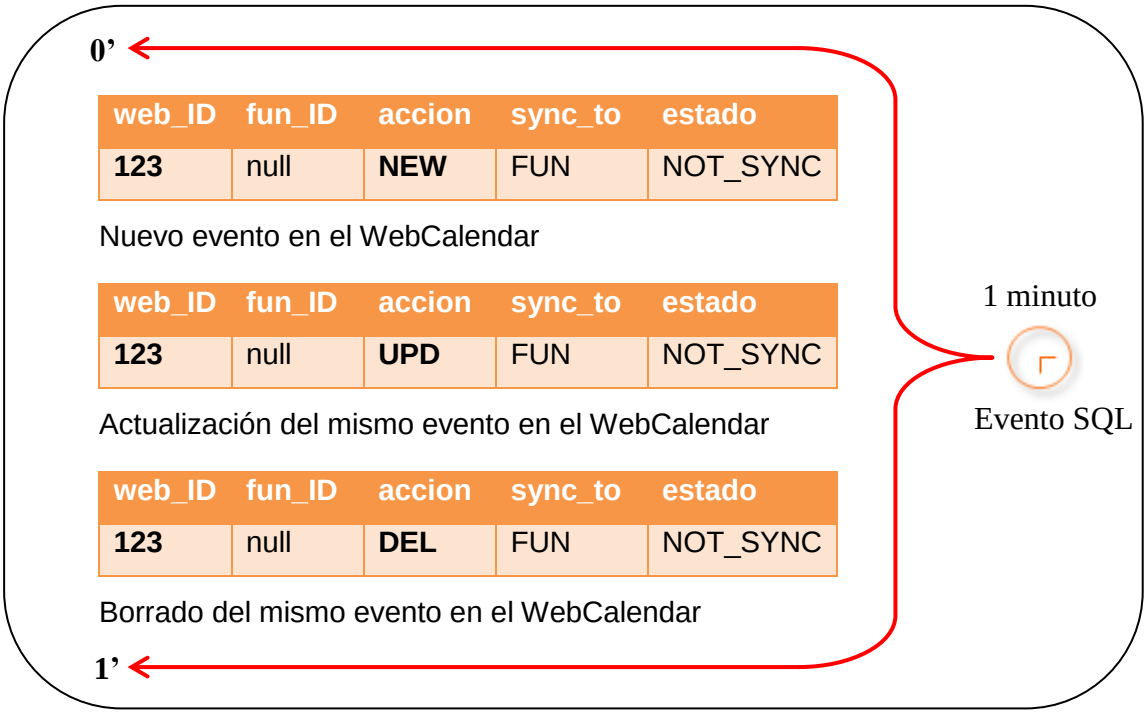

FIGURA 4-13: SIMULACIÓN DE CREACIÓN DE UN EVENTO

Fuente Propia

Se puede observar que un evento se ha creado, luego ha sido actualizado y finalmente borrado, todo este proceso entre el minuto de espaciamiento para la ejecución del evento SQL. Si no se comprobaría si las variables ‘web_id_val’ o ‘fun_id_val’ son nulas

antes de realizar cualquier sincronización, el procedimiento almacenado validaría la última acción declara en la tabla, que para este caso es 'DEL' o borrado e intentaría borrar el evento en el Funambol con un ID nulo, lo cual causaría un error SQL y terminaría la ejecución del procedimiento almacenado. Lo mismo sucedería en caso de una actualización sin ID Funambol definido.
Entonces, es por esta razón que se ha optado por sincronizar todos los eventos disponibles, verificando siempre inicialmente las variables 'web_id_val' o 'fun_id_val' son nulas para proceder a la creación de su evento correspondiente.

Para comenzar (en base a la figura 4-12), se analiza si un evento es nuevo por medio de la verificación del valor nulo en los campos 'web_id_val' o 'fun_id_val', tal como se acaba de explicar. Luego se verifica si la variable 'accion_val' es igual a 'UPD', que de ser así, se procede a analizar el valor de la variable 'sync_to_val' para verificar el sistema hacia donde se debe realizar la sincronización del evento. Finalmente, se verifica si la variable 'accion_val' es igual a 'DEL', para lo cual en conjunto con 'sync_to_val' se comprueba el sistema hacia donde se debe borrar el evento.

Con este flujo se concluye el '*Proceso CRUD de un evento'* mostrado en la figura 4-12, pero que a su vez, es un subproceso del procedimiento almacenado mostrado en la figura 4-10, en donde se observa que posterior a éste, se realiza la actualización del registro en la tabla 'correspondencia_id' con el valor 'SYNC' en el campo 'estado', para indicar que el registro actual se ha sido sincronizado correctamente.

Luego de realizar esta actualización, se repite el bucle hasta que el puntero del cursor termine de recorrerlo, con lo cual se procede a salir del bucle, cerrar el cursor y terminar el procedimiento almacenado, resultando así ambos sistemas completamente sincronizados, pues todos los eventos existen tanto en el WebCalendar como en el Funambol. De esta manera se **cumple con el objetivo principal de la tesis**.

En los siguientes subcapítulos se detallarán los subprocesos que realiza el '*Procedo CRUD de un evento'* para la creación, actualización y borrado de un evento, desde un sistema hacia el otro, siguiendo el flujo de la figura 4-12. Se empezará por los eventos que deben sincronizarse hacia el WebCalendar y luego por los que deben sincronizarse hacia el Funambol.

4.2.4.1 Creación de un evento en el WebCalendar

El flujo que sigue el procedimiento almacenado para la creación de un evento en el WebCalendar es el mostrado en la figura 4-14.

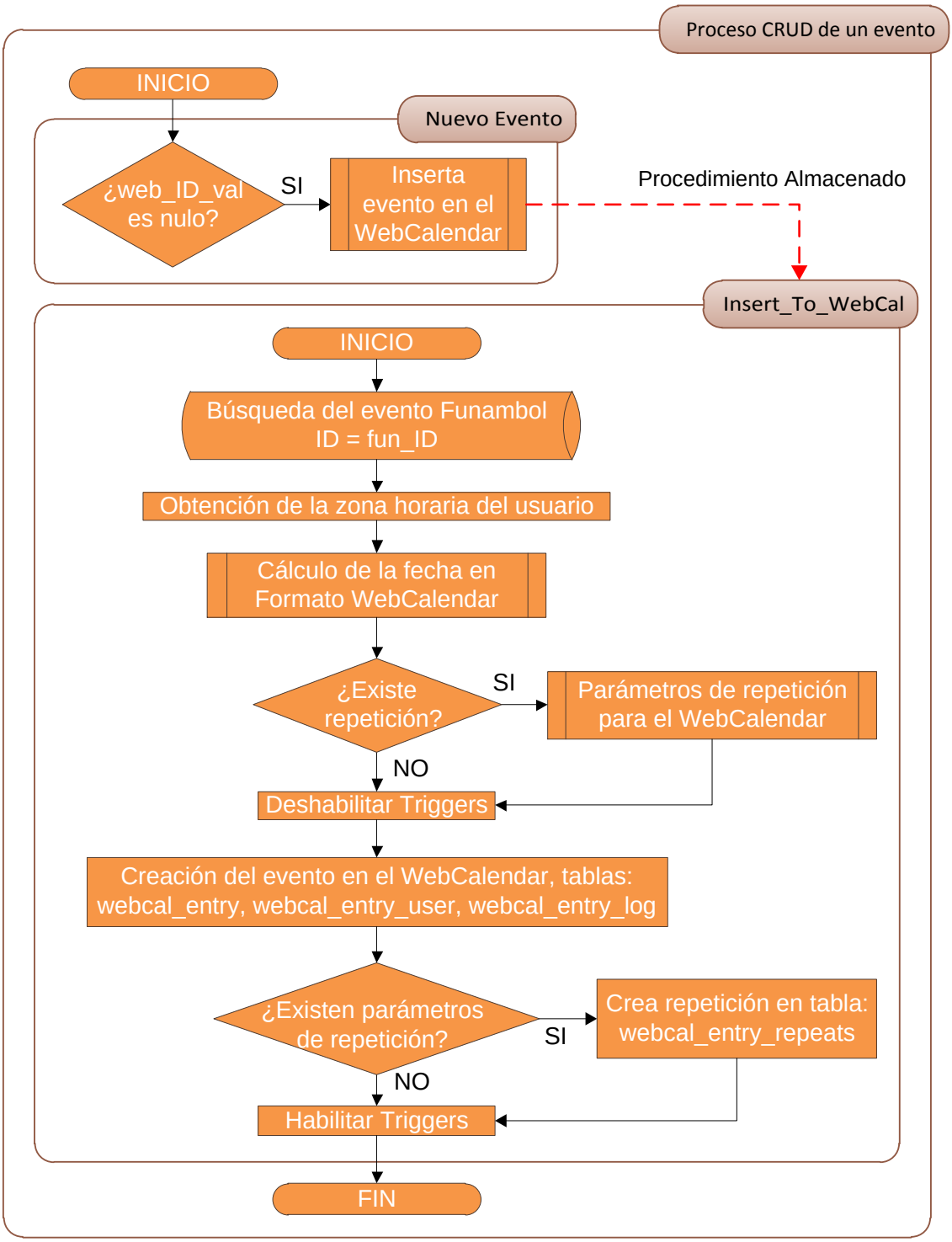

FIGURA 4-14: CREACIÓN DE UN EVENTO WEBCALENDAR

Fuente Propia

Teniendo en consideración lo explicado para el '*proceso CRUD de un evento*', en particular para la creación de un nuevo evento, al empezar el procedimiento almacenado se verifica si el valor de la variable 'web_id_val' es igual a NULL, de ser así, se da inicio al subproceso denominado '*Inserta evento en el WebCalendar'*.

Este subproceso es un procedimiento almacenado de nombre *'Insert_To_WebCal',* el cual se encarga de hacer la conversión de un evento existente en el Funambol y procesarlo para ser insertado en el WebCalendar. Cuando es llamado recibe como parámetro de entrada el ID del Funambol ('fun_id_val') y devuelve como parámetro de salida el nuevo ID generado en la base de datos del WebCalendar, 'web_id_val'.

En primer lugar, se realiza la búsqueda en la tabla 'fnbl_pim_calendar' del Funambol, con el identificador único del evento (ID) igual al recibido como parámetro de entrada 'fun_id_val'. Luego se procede a obtener la zona horaria de usuario WebCalendar, pues según definición del sistema web, éste contempla la configuración de un offset debido la zona horaria (pe. America/Los Angeles). Para obtener este valor se realiza una consulta sobre la tabla 'webcal_user_pref' del WebCalendar con el campo 'cal_setting' igual a 'TIMEZONE' y el id del usuario correspondiente.

Para el procesamiento de la zona horaria, se debe tener en consideración que los timezones definidos en la base de datos del WebCalendar siguen el formato 'Named Time Zones' el cual define las zonas horarias basadas en nombres. En MySQL se va a usar la sentencia SQL 'CONVERT_TZ', que recibe como parámetro de entrada el 'Named Time Zone' y devuelve el timezone gmt (pe. +05:00, es decir gmt+5). Para que esta función esté disponible en MySQL y funcione correctamente, es necesario contar con unas tablas adicionales en el propio esquema de MySQL que se deben instalar y configurar respectivamente y cuya información detallada se encuentra en el anexo 11.

A continuación se inicia el proceso *'Cálculo de la fecha en Formato WebCalendar'*, el cual es un procedimiento almacenado SQL de nombre 'Fecha_En_Formato_WebCal' (definido en el anexo 10), que recibe como parámetros de entrada la fecha y hora de inicio y fin del evento Funambol, el valor del campo 'all_day' y la diferencia horaria previamente calculada. Como parámetros de salida se obtienen la fecha y hora de inicio, la duración del evento y la fecha y hora de modificación del evento, todo en formato WebCalendar, tal como se observa en la figura 4-15.

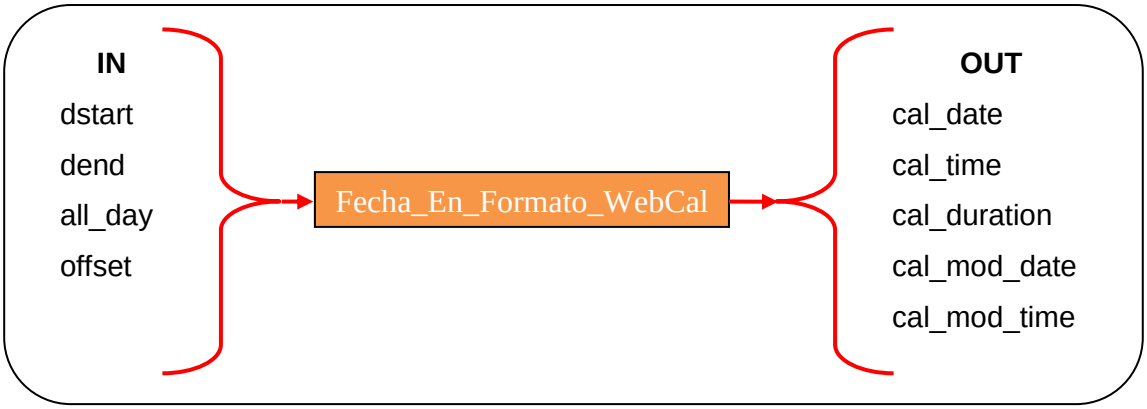

FIGURA 4-15: PARAMETROS IN,OUT DE FECHA_EN_FORMATO_WEBCAL

Fuente Propia

Paso siguiente, se valida si el evento tiene repetición. De ser así se ejecuta el subproceso '*Parámetros de repetición para el WebCalendar'*, el cual es un procedimiento almacenado de nombre *'Repeticion_En_Formato_WebCal'* que recibe como parámetros de entrada los valores de repetición del evento Funambol y como parámetros de salida los valores de repetición del evento en formato WebCalendar.

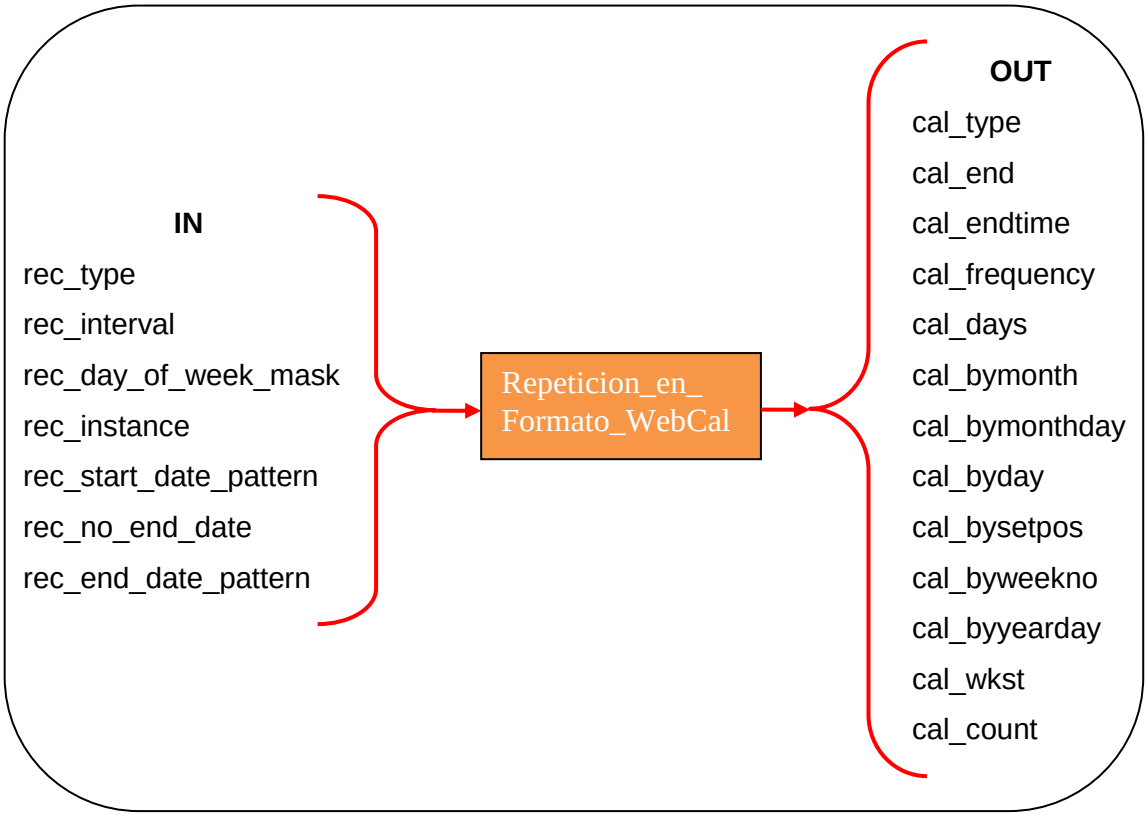

FIGURA 4-16: PARAMETROS IN,OUT DE REPETICION_EN_FORMATO_WEBCAL

Fuente Propia

Luego de realizar los procesos de conversión de los valores del evento Funambol hacia valores del WebCalendar, se procede a calcular el ID del evento que se creará en el WebCalendar.

Como siguiente paso se deshabilitan los triggers asociados a la tabla webcal_entry_user, por medio de la sentencia SET @DISABLE_TRIGGER = 1, lo que provoca que cuando se inserte una nueva entrada en la tabla webcal_entry_user, el trigger se dispara pero verifica el valor de la variable global 'DISABLE_TRIGGER' y al encontrar que su valor esta seteado en 1, no va a realizar ningún tipo de acción y termina el trigger.

Con el trigger deshabilitado, se procede a la creación del nuevo evento en la tabla 'webcal_entry' y 'webcal_entry_user', con todos los parámetros calculados en el proceso *'Cálculo de la fecha en Formato WebCalendar'.* Así mismo, se verifica si existen los parámetros de repetición del evento calculados en el proceso y '*Parámetros de repetición para el WebCalendar'*, mostrado en la figura 4-16. De existir repetición, se inserta en la tabla 'webcal_entry_repeats' una nueva entrada con el mismo ID del nuevo evento creado para la tabla 'webcal_entry'; caso contario, no se realiza ninguna inserción.

Luego se reactivan los triggers por medio de la sentencia SET @DISABLE_TRIGGER = null, para que el sistema continúe llenando la tabla 'correspondencia_id' por medio de los triggers en caso de algún evento nuevo, actualizado o borrado.

Finalmente el procedimiento almacenado 'Insert_To_WebCal' retorna el ID del nuevo evento generado en el WebCalendar con nombre 'web_id_val', con el cual se actualiza la tabla 'correspondencia_id' para la entrada correspondiente al 'fun_id_val', además de actualizar el campo 'estado' con el valor de 'SYNC', para indicar que el registro actual se ha sido sincronizado correctamente. De esta manera se obtendrá el evento Funambol con su correspondiente evento en el WebCalendar, en otras palabras, se ha sincronizado el evento en ambas base de datos.

Como parte del flujo principal descrito en el punto 4.2.3, luego de la actualización de la tabla 'correspondencia_id', se inicia una nueva iteración sobre el bucle que recorre el cursor.

4.2.4.2 Actualización de un evento en el WebCalendar

Cuando un evento se actualiza en el Funambol y debe ser sincronizado en el WebCalendar, se sigue el flujo mostrado en la figura 4-17.

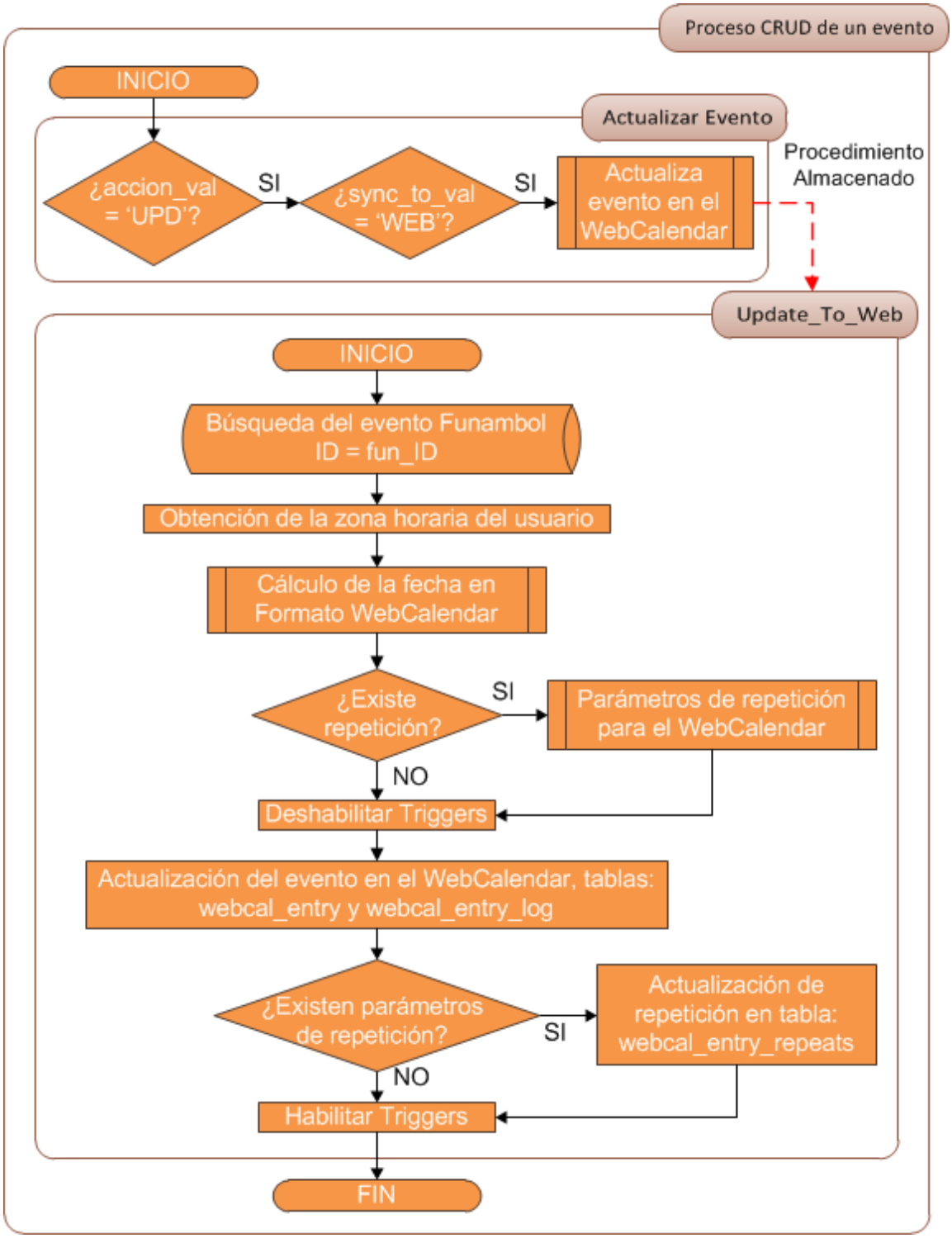

FIGURA 4-17: ACTUALIZACIÓN DE UN EVENTO WEBCALENDAR

Fuente Propia

Como parte del 'Proceso CRUD de un evento' (4.2.4), se verifica si la variable 'accion_val' es igual a 'UPD' y luego si la variable 'sync_to_val' es igual a 'WEB'. Si ambos requisitos se cumplen entonces se realiza la actualización del evento desde el Funambol hacia el WebCalendar por medio del subproceso '*Actualiza evento en el WebCalendar'* que es un procedimiento almacenado de nombre 'Update_To_Web'.

El proceso que sigue el procedimiento almacenado 'Update_To_Web' para actualiza un evento es exactamente igual al procedimiento almacenado de creación de un evento (4.2.4.1), con dos diferencias:

1. El procedimiento almacenado recibe dos parámetros de entrada a diferencia de 'Insert_To_WebCal' que tenía un parámetro de entrada y otro de salida. Estos dos parámetros de entrada son el ID del evento Funambol 'fun_id_val' y el ID del evento WebCalendar 'web_id_val', pues como se explicó en el capítulo 4.2.4.1, un evento sólo puede actualizarse si ha sido creado previamente en ambos sistemas, por ende existe el ID del evento Funambol y WebCalendar.
2. Una vez deshabilitado los triggers, no se realiza el proceso de creación de un evento sino por el contrario se realiza la actualización del evento con los parámetros obtenidos por los procedimientos almacenados '*Fecha_En_Formato_WebCal'* y en caso de existir repetición '*Repeticion_En_Formato_WebCal'* (Ambos procedimientos almacenados son los mismos descritos en el capítulo 4.2.4.1 para la creación de un evento).
   La actualización se realiza en las tablas 'webcal_entry', 'webcal_entry_log' y en caso de existir repetición en la tabla 'webcal_entry_repeats'.

Con los procesos descritos anteriormente se realiza la actualización de un evento del Funambol hacia el WebCalendar; sin embargo, se debe considerar un caso en particular. Un evento en el Funambol que inicialmente tenía un evento con repetición y que fue sincronizado por medio del procedimiento almacenado 'Insert_To_WebCal', si luego a este mismo evento se le borra la configuración de repetición, al momento de realizar la actualización del evento por medio de 'Update_To_Web', el registro de la tabla 'webcal_entry_repeats' no es borrado ni puesto como inactivo pues no tiene esa opción; sino que, el registro principal del evento que se encuentra en la tabla 'webcal_entry', cuando se actualiza varía el valor de su campo 'cal_type' de 'M' (el cual indica que se repite) al valor de 'E', indicando así que ese evento ya no tiene repetición.

4.2.4.3 Borrado de un evento en el WebCalendar

Cuando un evento se borra en el Funambol y debe ser sincronizado en el WebCalendar, se sigue el flujo mostrado en la figura 4-18.

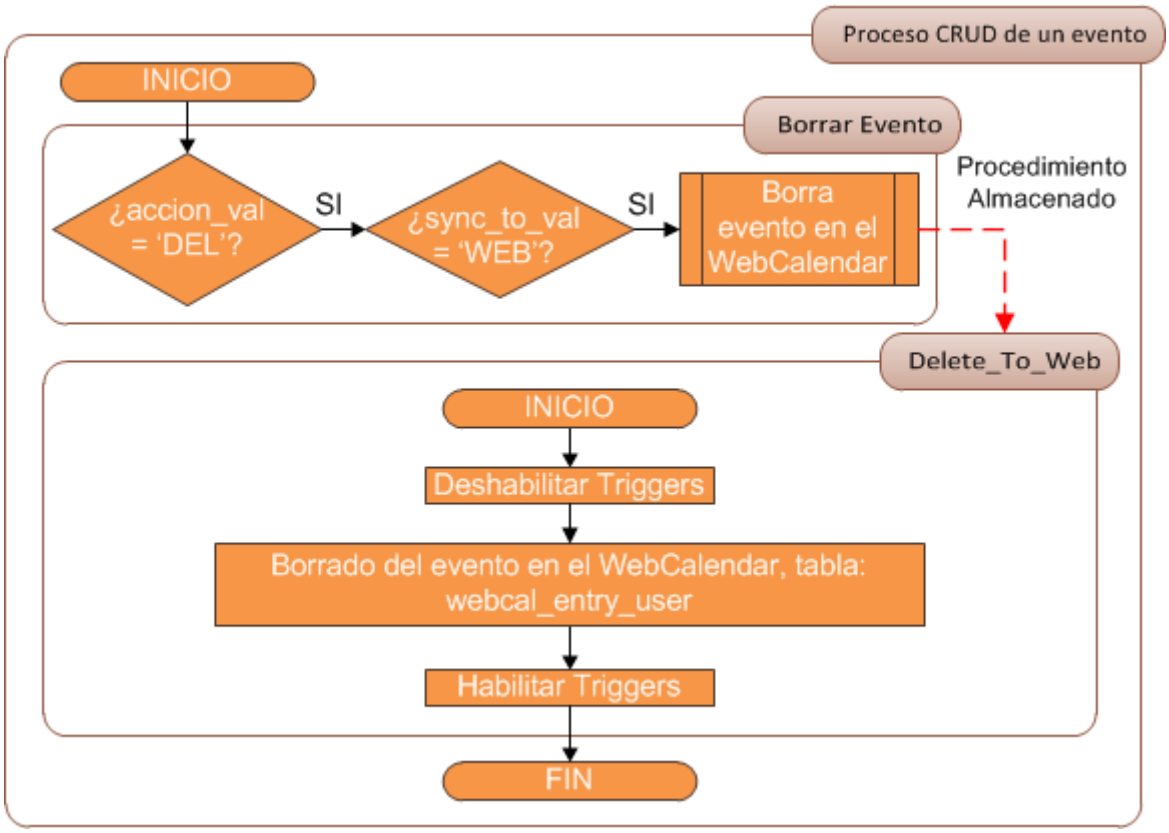

FIGURA 4-18: BORRADO DE UN EVENTO WEBCALENDAR

Fuente Propia

Como punto final del WebCalendar, se debe poder sincronizar el borrado de un evento del Funambol hacia el WebCalendar. En el 'Proceso CRUD de un evento' (4.2.4), se verifica se verifica si la variable 'accion_val' es igual a 'DEL' y luego si la variable 'sync_to_val' es igual a 'WEB', que de cumplirse la condición se llama al proceso '*Borra evento en el WebCalendar'*. Este proceso es procedimiento almacenado de nombre 'Delete_To_Web', el cual recibe un único parámetro de entrada que es el ID del evento WebCalendar correspondiente al evento Funambol que se ha borrado. El procedimiento almacenado deshabilita el trigger y realiza una sentencia SQL udpate sobre la tabla 'webcal_entry_user' cambiando el valor del campo 'cal_status' a 'D' para el evento WebCalendar con ID igual al parámetro de entrada recibido 'web_id_val'. Finalmente se habilitan nuevamente los triggers y termina el proceso de borrado de un evento del Funambol hacia al WebCalendar.

4.2.4.4 Creación de un evento en el Funambol

El flujo que sigue el procedimiento almacenado para la creación de un evento en el Funambol es el mostrado en la figura 4-19.

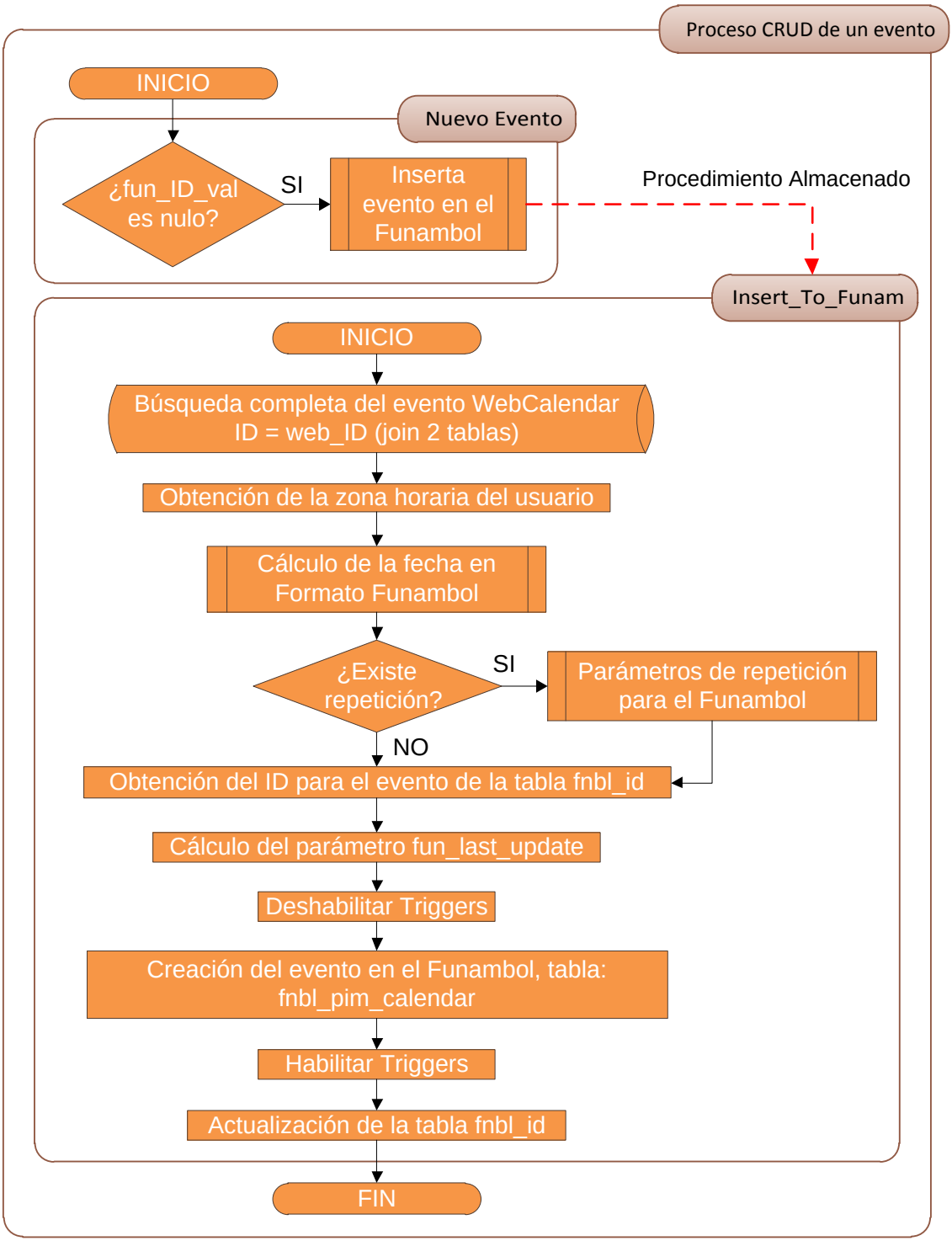

FIGURA 4-19: CREACIÓN DE UN EVENTO FUNAMBOL

Fuente Propia

Para la sincronización un evento nuevo desde el WebCalendar hacia el Funambol (siguiendo el '*Proceso CRUD de un evento*'), se valida si la variable 'fun_id_val' tiene el valor de nulo. De ser así, se inicia el subproceso '*Inserta evento en el Funambol'*.

Este subproceso es un procedimiento almacenado de nombre *'Insert_To_Funam*', el cual se encarga de hacer la conversión de un evento existente en el WebCalendar y procesarlo para ser insertado en el Funambol. Cuando se llama a éste procedimiento almacenado, se tiene como parámetro de entrada el ID del evento WebCalendar ('web_id_val') y como parámetro de salida el nuevo ID del evento generado en la base de datos del Funambol, 'fun_id_val'.

En primer lugar, se realiza la búsqueda completa del evento, es decir:

- Toda la información general del evento, obtenida mediante la tabla 'webcal_entry' (tabla 1 del anexo 6), para el identificador único del evento 'cal_id' igual al parámetro recibido 'web_id_val'.
- Toda la información de la repetición del evento, obtenida mediante la tabla 'webcal_entry_repeats' (tabla 3 del anexo 6) en caso lo tuviera para el identificador 'cal_id' igual a 'web_id_val'.

Para realizar una correcta búsqueda en la base de datos, se hace uso de la sentencia SQL 'LEFT JOIN', entre la tabla 'webcal_entry' y la tabla 'webcal_entry_repeats', de manera que siempre se obtenga la información general del evento como parte del resultado y también la información de la repetición del evento en caso existiera. Caso contario, los campos de repetición tendrían el valor de nulo.

Luego se procede a obtener la zona horaria del usuario configurada en el WebCalendar, por medio de una consulta sobre la tabla 'webcal_user_pref', con el campo 'cal_setting' igual a 'TIMEZONE' y el ID del usuario correspondiente.

A continuación se inicia el proceso *'Cálculo de la fecha en Formato Funambol'*, el cual es un procedimiento almacenado de nombre 'Fecha_En_Formato_Funam' detallado en el anexo 10. Este procedimiento recibe como parámetros de entrada la fecha y hora del evento WebCalendar, su duración, y la diferencia horaria previamente calculada. Como parámetros de salida se obtienen la fecha y hora de inicio y fin en formato Funambol y una variable que indica si el evento dura todo el día. Las variables que se envían y se reciben del procedimiento almacenado son mostradas en la figura 4-20.

FIGURA 4-20: PARAMETROS IN,OUT DE FECHA_EN_FORMATO_FUNAM

Fuente Propia

El paso siguiente es validar si el evento tiene repetición. De ser así se ejecuta el subproceso *'Parámetros de repetición para el Funambol'* el cual llama al procedimiento almacenado *'Repeticion_En_Formato_Funam'*, mostrado en la figura 4-21.

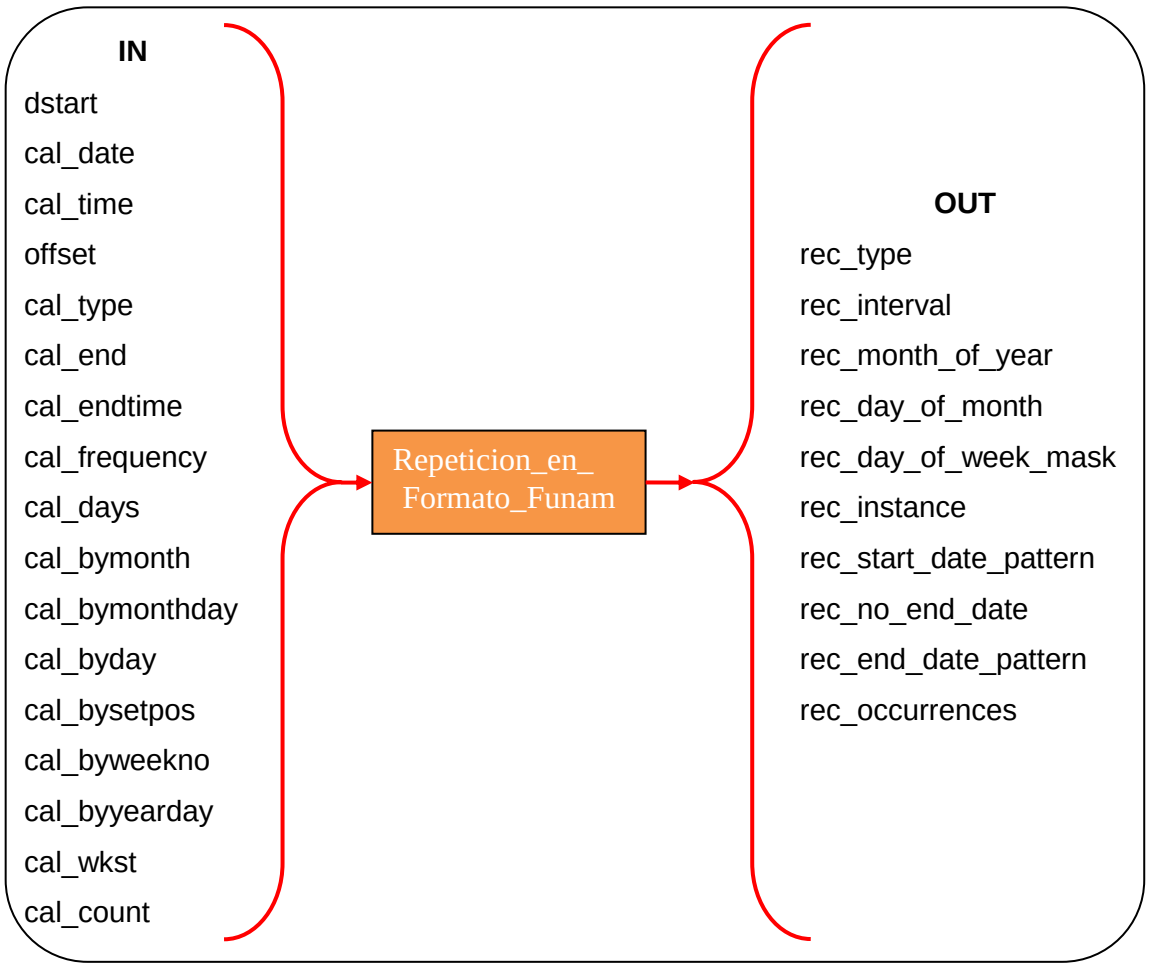

FIGURA 4-21: PARAMETROS IN,OUT DE REPETICION_EN_FORMATO_FUNAM

Fuente Propia

La figura 4-21 muestra el procedimiento almacenado *'Repeticion_En_Formato_Funam'* el cual tiene como parámetros de entrada los valores de repetición del evento WebCalendar y como parámetros de salida los valores de repetición del evento en formato Funambol. Su detalle en el anexo 10.

Se obtiene el ID que tendrá el nuevo evento en la variable 'fun_pim_id' por medio de la tabla fnbl_id, cumpliendo la condición en el SQL de 'idspace' igual a 'pim_id'.

A continuación se debe tener presente lo siguiente. Cuando el Funambol sincroniza eventos entre su base de datos y los dispositivos móviles asociados, graba la hora de la última sincronización realizada, de manera que la siguiente vez que se solicite sincronizar los eventos, sólo sincroniza aquellos cuyo valor del campo 'last_update' de la tabla 'fnbl_pim_calendar' sea mayor al valor de la última sincronización. Es por esta razón que se obtiene la hora actual en formato UNIX en la variable 'fun_last_update' y se inserta junto con la información del evento, lo cual le indicará al Funambol que éste evento es nuevo y debe ser sincronizado con el dispositivo móvil que se lo solicite.

Se procede a deshabilitar los triggers asociados a la tabla fnbl_pim_calendar, por medio de la sentencia SET @DISABLE_TRIGGER = 1.
Con el trigger deshabilitado, se procede a la creación del nuevo evento en la tabla 'fnbl_pim_calendar', con todos los parámetros del evento con formato Funambol, calculados en los procesos descritos anteriormente.
Una vez terminada la creación del evento se reactivan los triggers por medio de la sentencia SET @DISABLE_TRIGGER = null.

Finalmente se actualiza la tabla fnbl_id (campo 'pim_id') con el valor que tendrá el próximo evento a ser creado. Así mismo el procedimiento almacenado 'Insert_To_Funam' retorna el nuevo ID generado en el Funambol con nombre 'fun_id_val', con el cual se actualiza la tabla 'correspondencia_id' para la entrada correspondiente al 'web_id_val', además de actualizar el campo 'estado' con el valor de 'SYNC', para indicar que el registro actual se ha sido sincronizado correctamente.

Como parte del flujo principal descrito en el punto 4.2.3, luego de la actualización de la tabla 'correspondencia_id', se inicia una nueva iteración sobre el bucle que recorre el cursor.

4.2.4.5 Actualización de un evento en el Funambol

El flujo que sigue el procedimiento almacenado para la actualización de un evento en el Funambol es el mostrado en la figura 4-22.

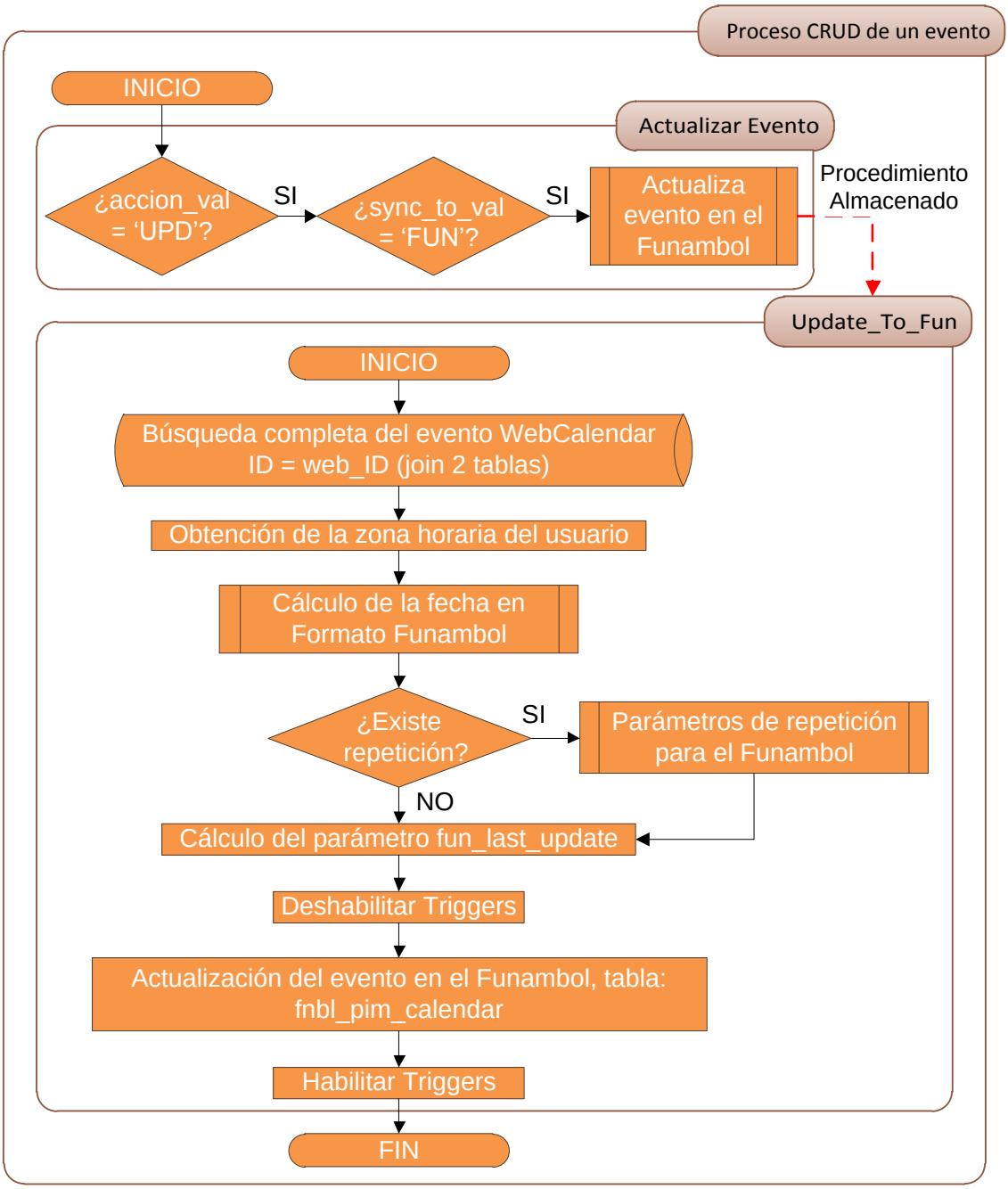

FIGURA 4-22: ACTUALIZACIÓN DE UN EVENTO FUNAMBOL

Fuente Propia

Siguiendo el flujo principal del '*Proceso CRUD de un evento'* (4.2.4), ahora se verifica si la variable 'accion_val' es igual a 'UPD' y si la variable 'sync_to_val' es igual a 'FUN'.

Si ambos requisitos se cumplen entonces se procede a realizar la actualización del evento desde el WebCalendar hacia el Funambol por medio del subproceso '*Actualiza evento en el Funambol',* el cual es un procedimiento almacenado de nombre 'Update_To_Fun'.

El flujo seguido por el procedimiento almacenado 'Update_To_Web' para la actualización de un evento desde el WebCalendar hacia el Funambol, es exactamente igual al flujo seguido por el procedimiento almacenado de creación de un evento del WebCalendar hacia el Funambol (4.2.4.4), con dos diferencias:

1. El procedimiento almacenado recibe dos parámetros de entrada a diferencia de 'Insert_To_Funam' que tenía un parámetro de entrada y otro de salida. Estos dos parámetros de entrada son el ID del evento WebCalendar 'web_id_val' y el ID del evento Funambol 'fun_id_val', ambos obtenidos de la búsqueda sobre la tabla 'correspondencia_id' almacenada en el cursor.
2. Una vez deshabilitado los triggers, no se realiza el proceso de creación de un evento sino por el contrario se realiza la actualización del evento con los parámetros obtenidos por los procedimientos almacenados '*Fecha_En_Formato_Funam'* y en caso de existir repetición '*Repeticion_En_Formato_Funam'* (Ambos procedimientos almacenados son los mismos descritos en el capítulo 4.2.4.4 para la creación de un evento).
   La actualización se realiza únicamente sobre la tabla 'fnbl_pim_calendar'.

Con los procesos descritos anteriormente se realiza la actualización del evento de la base de datos del WebCalendar hacia la base de datos del Funambol.

El procedimiento almacenado de actualización de un evento no retorna ninguna variable pues no tiene definido parámetros de salida. Su detalle se encuentra en el anexo 10.

Finalmente, se actualiza el campo 'estado' con el valor de 'SYNC' de la tabla 'correspondencia_id' para la entrada correspondiente al 'web_id_val', para indicar que el registro actual se ha sido sincronizado correctamente.
Como parte del flujo principal descrito en el punto 4.2.3, luego de la actualización de la tabla 'correspondencia_id', se inicia una nueva iteración sobre el bucle que recorre el cursor.

4.2.4.6 Borrado de un evento en el Funambol

Cuando se borra un evento en el WebCalendar y debe ser sincronizado al Funambol, se sigue el flujo mostrado en la figura 4-23.

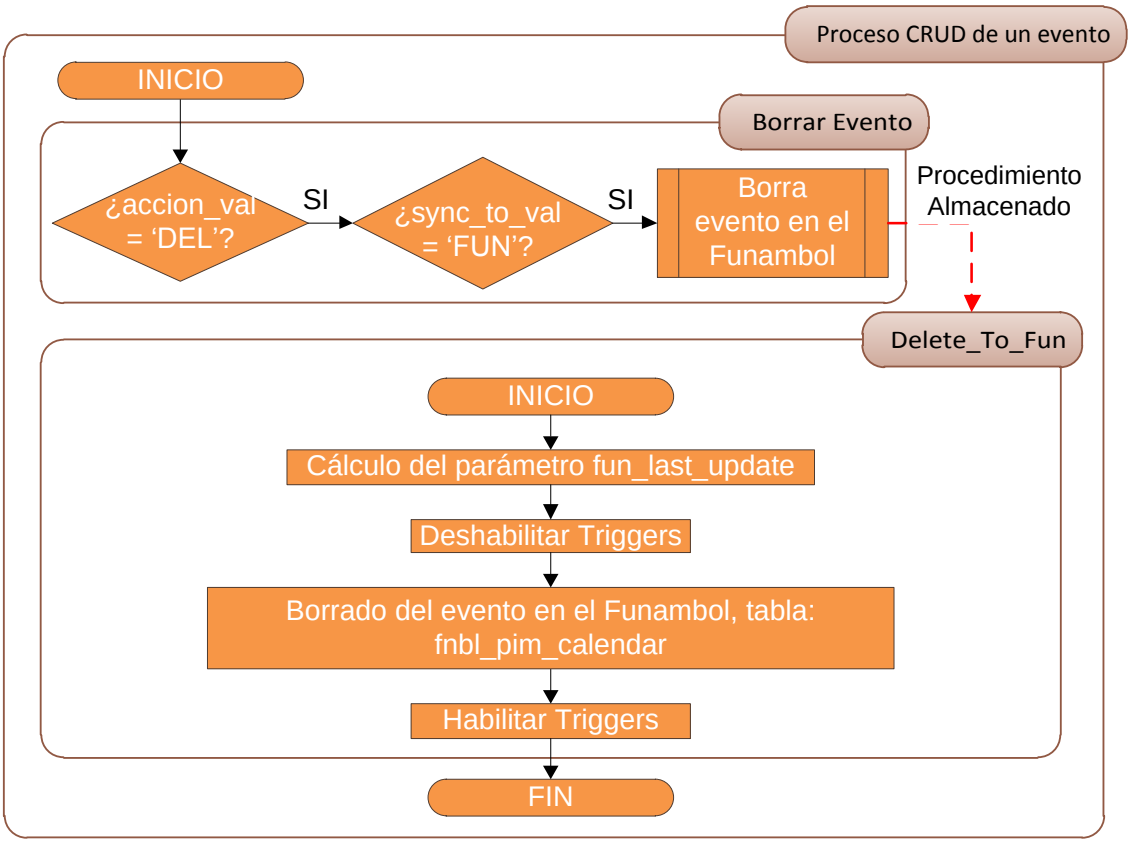

FIGURA 4-23: BORRADO DE UN EVENTO FUNAMBOL

Fuente Propia

Como parte final del '*Proceso CRUD de un evento'* (4.2.4), se debe poder sincronizar el borrado de un evento del WebCalendar hacia el Funambol.

Para comenzar se verifica si la variable 'accion_val' es igual a 'DEL' y si la variable 'sync_to_val' es igual a 'FUN'. De cumplirse ambas condiciones se llama al proceso *'Borra evento en el Funambol'* que es un procedimiento almacenado de nombre 'Delete_To_Fun'.

Este procedimiento recibe un único parámetro de entrada que es el ID del evento Funambol ('fun_id_val') equivalente al evento WebCalendar que se ha borrado. En primer lugar se obtiene el parámetro 'fun_last_update', con el fin de notificar al sistema

Funambol que el evento debe ser sincronizado si un dispositivo móvil así lo solicita (esto fue detallado en el punto 4.2.4.4).

Posteriormente se deshabilita el trigger y se realiza una sentencia SQL udpate sobre la tabla 'fnbl_pim_calendar' para el ID del evento que se recibió como parámetro de entrada, actualizando el valor del campo 'status' a 'D' y el valor del campo 'last_update' con el parámetro 'fun_last_update' calculado previamente.

Luego se habilitan nuevamente los triggers y termina el proceso de borrado de un evento del WebCalendar hacia el Funambol.

Finalmente, se actualiza el campo 'estado' con el valor de 'SYNC' de la tabla 'correspondencia_id' para la entrada correspondiente al 'web_id_val', para indicar que el registro actual se ha sido sincronizado correctamente.

Como parte del flujo principal descrito en el punto 4.2.3, luego de la actualización de la tabla 'correspondencia_id', se inicia una nueva iteración sobre el bucle que recorre el cursor.

4.3 Interfaz de descarga del software para la sincronización

El Funambol para realizar la sincronización entre su base de datos y los dispositivos móviles que usen el sistema, necesita de un software capaz de enviar la data usando el protocolo SyncML y que no viene instalado por defecto en ningún dispositivo actual en el mercado.

Este software es brindado de manera gratuita para los usuarios del Funambol, pero se encuentra en la página web de la misma y le tomaría tiempo al usuario buscar el software que se acomode a sus necesidades entre las muchas alternativas y versiones que se poseen.

Es por esta razón que se crea una interfaz web para ser visitada desde el WebCalendar, en la cual se muestra el software que se debe descargar el usuario, necesario para la sincronización de eventos. Como parte del software disponible se encuentra versiones para:

- Dispositivos con sistema operativo Android.

- Blackberry OS.
- Iphone.
- Dispositivos con sistema operativo symbian.
- Dispositivos con sistema operativo Windows Mobile.

Ademas del instalador, se cuenta con una guía del usuario para poder realizar la correcta instalación del software en el dispositivo.

La interfaz para la descarga del software se muestra en la figura 4-24.

FIGURA 4-24: INTERFAZ DE DESCARGA DEL SOFTWARE PARA LA SINCRONIZACIÓN

Fuente propia

Para poder acceder a esta interfaz es necesario hacer clic en la palabra “Sincronízate!” que se encuentra en la barra superior de la interfaz web.

El código fuente para implementación de la interfaz de descarga del software para la sincronización en clientes móviles se encuentra presente en el CD de anexos.

Capítulo 5

Implementación del sistema

Como resultado de la interconexión del sistema de calendarios online web (WebCalendar) con el sistema de calendarios con soporte para la sincronización móvil (Funambol), se procede a indicar los requerimientos necesarios para poder instalar el sistema interconectado completo. Posteriormente se indicará un método para instalar el SQL event, los procedimientos almacenados y los triggers, todo por medio de un solo script SQL. Finalmente deben ser configurados ciertos aspectos claves para que tanto los usuarios regulares como los administradores puedan hacer uso correcto y efectivo de todas las capacidades del sistema.

5.1 Requerimientos físicos del sistema

En lo que respecta a los requerimientos físicos del sistema, en primer lugar especificar que ambos sistemas van a ser instalados en el mismo servidor, es decir, tanto el WebCalendar, el Funambol, la base de datos respectiva de cada uno de ellos y la base de datos interconexion.

En materia de procesamiento, el Funambol recomienda como mínimo un host (PC o servidor) con los siguientes parámetros: [FAG2010]

- Pentium 4, corriendo a 1.8GHz
- No hay restricción de sistema operativo para Linux; sin embargo, para Windows es soportado a partir del 2000, XP profesional, Vista y 7(todas sus versiones).
- 200 MB de espacio en el disco
- 512 MB de memoria RAM

Esto solo para levantar el servicio del Funambol; sin embargo en capacidad de almacenamiento, se recomienda un total de 1.5Mb por usuario (con lo que se permite un promedio de 1000 contactos y eventos por usuario). Si se tiene en cuenta una capacidad aproximada de 5000 usuarios, es necesario un disco de almacenamiento con capacidad mínima de 7,5 GB sólo como storage de la data del sistema Funambol.

Por otro lado, el WebCalendar no especifica requerimientos mínimos de capacidad de procesamiento para su sistema, por lo cual, podríamos considerar la misma capacidad de procesamiento para ambos sistemas dado que siempre que un evento se cree en un sistema, éste debe ser recreado en el otro, con lo cual se duplicaría la carga de procesamiento. En temas de almacenamiento, también se va a optar por una capacidad de 7,5 GB para la data del WebCalendar.

En lo que respecta al software a ser instalado, en primer lugar, se necesita instalar el gestor de base de datos MySQL. Esto debido a que aún cuando ambos sistemas pueden funcionar con distintas bases de datos como Oracle, PostgreSql, Interbase, entre muchas otras, el sistema se ha desarrollado en base al funcionamiento y facilidades que brinda la base de datos MySQL, explicado en el capítulo 1. Así mismo la versión mínima de MySQL debe ser la 5.0.2 pues a partir de esta versión se incorporó el soporte básico para los triggers, base de la interconexión de los sistemas de la presente Tesis.

En segundo lugar, es necesario tener un servidor Apache 2.0, en el cual se encuentre instalado como módulo el servicio PHP. De igual forma, la versión para PHP mínima necesaria por el WebCalendar es la 5.0, además de tener instalados los siguientes módulos para su correcto funcionamiento:

- Módulo de soporte MySQL para PHP.
- Módulo de soporte mcrypt para PHP.
- Módulo de soporte para cURL en PHP.

5.2 Instalación del WebCalendar y Funambol

Para la instalación de cada uno de los subsistemas, en ambos casos se encuentra online sus archivos respectivos necesarios para la instalación en cada una de sus páginas webs.

En el caso del Funambol, se encuentra actualmente la versión 10.0 y puede ser encontrado en la siguiente página web:

https://www.forge.funambol.org/download/#server

En la cual se puede elegir el instalador ya sea para un host en Windows o uno en alguna distribución de Linux.

Para el caso del WebCalendar, se encuentra en la versión 1.2.3 y se puede obtener el comprimido con el sistema en la siguiente dirección:

http://www.k5n.us/webcalendar.php?topic=Download

Al descomprimir el sistema puedes instalarse en un host en Windows o Linux, pues es indistinto del sistema operativo, siempre y cuando corra sobre el Apache WebServer y teniendo el módulo de PHP instalado y habilitado.

En ambos casos las guías y/o manuales para la instalación paso a paso en el host o servidor se encuentran en sus respectivas páginas web:

WebCalendar: http://webcalendar.cvs.sourceforge.net/viewvc/webcalendar/webcalendar/docs/WebCalendar-SysAdmin.html

Funambol: http://download.forge.objectweb.org/sync4j/Funambol-installation-and-administration-guide-v8.7.pdf

Un punto a tener en cuenta es que el Funambol por defecto instala se instala con conexión a una base de datos PostgreSql, sin embargo se deben realizar las configuraciones necesarias para levantarlo con la base de datos MySQL, lo cual se encuentra en la página 71 del manual referido anteriormente para su instalación.

5.3 Script SQL

Luego de realizar la instalación del sistema, es necesario insertar el evento SQL, los procedimientos almacenados y los triggers en la base de datos MySQL de ambos esquemas (WebCalendar y Funambol) así como también instalar la base de datos interconexión y sus tablas correspondientes.

El script completo de todo lo implementado en SQL, el cual reúne la base de datos interconexión y sus tablas, los procedimientos almacenados descritos en los anexos, el evento SQL y los triggers, se encuentra distribuido en los anexos 8,9, y 10. Para un correcto funcionamiento del sistema, los scripts deben ejecutarse en el siguiente orden:

1. Creación de base de datos interconexión y sus tablas correspondiente.
2. Inserción de todos los procedimientos almacenados sobre la base de datos interconexión.
3. Inserción de los triggers en cada uno de los sistemas.

5.4 Configuración de parámetros en el WebCalendar

Se deben configurar un parámetro en el sistema WebCalendar y uno en el servidor CentOS para que el sistema funcione correctamente para los usuarios.

El primero es configurar en el servidor el envío de recordatorios por medio de correos en caso el usuario decida que se le recuerde de un evento en particular por correo electrónico. Para activar esta funcionalidad en el WebCalendar es necesario configurar un cron en el servidor (de ser Linux). El cron es un administrador regular de procesos de segundo plano, que se encarga de ejecutar las sentencias contenidas en su estado de registro según intervalos de tiempo definidos por el administrador del sistema.

Para modificar el cron se usa la sentencia *crontab –e (se debe estar en el modo superusuario),* para una vez dentro insertar la siguiente línea de código:

*1 * * * * cd /some/directory/webcalendar/tools; ./send_reminders.php*

La línea escrita le indica al cron que debe ejecutar el archivo send_remindes.php el primer minuto de cada hora de todos los días.
Cabe resaltar que *"/some/directory"* es la ubicación en el servidor Apache en donde se encuentra el directorio del sistema WebCalendar.
También es posible setear un cron en el host con Windows. La guía para el mismo se encuentra en el manual de instalación del punto 5.2.

En segundo lugar es necesario activar el acceso público de manera que los usuarios puedan crear eventos y éstos a su vez sean puestos de manera pública siempre y cuando el administrador o moderador haya dado el visto bueno aprobándolos.
La configuración es la siguiente:

1. Entrar al WebCalendar como administrador.
2. Clic en el link “Opciones” en el menú superior de la página.
3. Clic en “System Setting”.
4. Clic en la pestaña “Public Access”.
5. Seleccionar “Yes” para “Allow public Access”.
6. Luego seleccionar “Yes” para “Public Access new events require approval”

De esta manera se asegurará que los eventos públicos deban ser aprobados por un administrador.

5.5 Configuración de parámetros en el Funambol

Para el caso del Funambol, posterior a su instalación tal y como se indica en el punto 5.2, restaría indicar el IP público por el cual está saliendo el servidor con el servicio Funambol. Los pasos son los siguientes:

1. Entrar a la herramienta de administración del Funambol (Administration Tool)
2. Logearse con el usuario por defecto, “admin” y password “sa” (sin comillas).
3. Expandir el árbol lateral izquierdo y hacer doble clic en “Server Settings”.
4. En el campo “Server URI property” ingresar el IP o nombre de dominio y el puerto donde está ejecutándose el servicio Funambol, siguiendo el modelo:

   http://<server name or IP address>:<server port>/funambol/ds
5. Finalmente hacer clic en “Save” para guardar los cambios respectivos.

Para que el cambio haga efecto es necesario realizar un reinicio al servicio Funambol.

5.6 Modificaciones al código del WebCalendar

Todas las modificaciones y comentarios adicionales al WebCalendar para lograr que funcione según los requerimientos de la presente Tesis, se encuentran detallados en el anexo 13 y además para evitar errores de codificación entre la base de datos del WebCalendar y del Funambol, revisar el anexo 12.

Capítulo 6

Pruebas de uso del sistema

Para la implementación se poseía un servidor virtual ubicado en la sala de servidores de la Dirección de Informática Académica (DIA) de la Pontificia Universidad Católica del Perú. Este servidor virtual tenía el sistema operativo CentOS versión 5 de 64bits, con 2GB de memoria RAM, un único núcleo y 20 GB de disco de almacenamiento.

El servidor poseía IP pública y fue registrado con el dominio agendas.pucp.edu.pe, sobre el cual se montó un servidor Apache 2 con su respectivo módulo de PHP versión 5.1.2 y MySQL en la versión 5.1.

Luego de la instalación de ambos sistemas y la ejecución del Script SQL de sincronización, se realizó pruebas de funcionamiento del sistema así como la correcta sincronización de un sistema hacia el otro.

6.1 Funcionamiento del sistema

En primer lugar el usuario al ingresar al sistema es recibido por una interfaz en la cual tiene tres opciones: entrar al sistema como usuario PUCP, registrarse o recuperar su contraseña, tal como se muestra en la figura 6-1.

FIGURA 6-1: INTERFAZ DE INGRESO AL SISTEMA

Fuente propia

Si el usuario es nuevo debe registrarse para poder hacer uso del sistema. La interfaz por donde se realiza el proceso de registro se muestra en el capítulo 4.1, figura 4-2.

Luego de registrarse, se envía un correo de manera automática para confirmar la cuenta. Un ejemplo del correo recibido se muestra en la figura 6-2.

FIGURA 6-2: CORREO ENVIADO PARA LA CONFIRMACIÓN

Fuente propia

Una vez que el usuario hace clic en el link mostrado, se procede automáticamente a activar la cuenta.

Luego el usuario ingresa al sistema WebCalendar, mostrado en la figura 6-3.

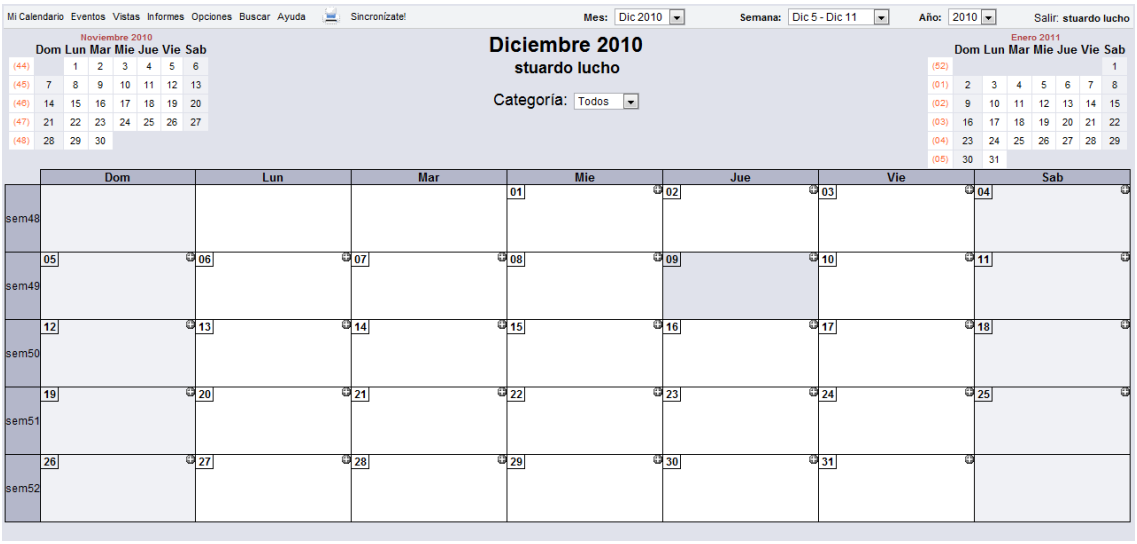

FIGURA 6-3: INTERFAZ PRINCIPAL DEL WEBCALENDAR

Fuente propia

Una vez dentro del sistema es posible revisar eventos así como también editar y crear nuevos eventos. Como parte de las pruebas, se creó un evento en el WebCalendar y se tomó una captura de la entrada correspondiente a éste evento en su base de datos (en la tabla webcal_entry). El grafico se encuentra en la figura 6-4.

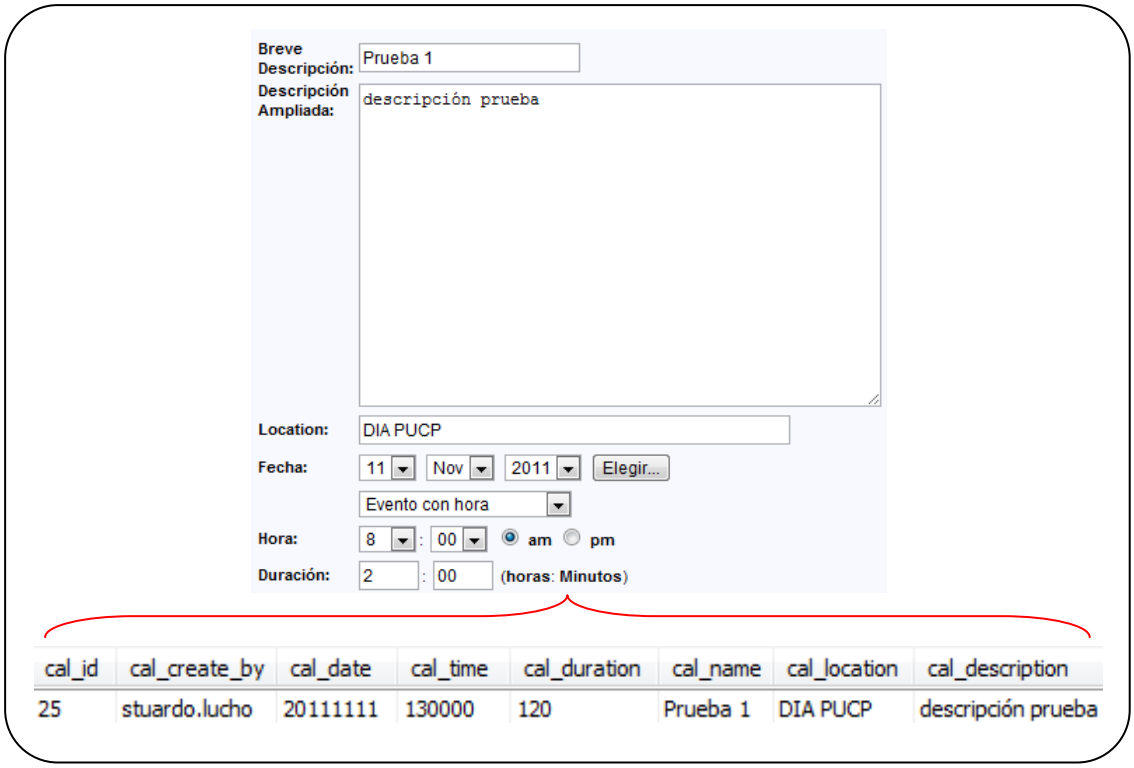

FIGURA 6-4: EVENTO DE PRUEBA EN EL WEBCALENDAR

Fuente propia

A continuación se muestra en la figura 6-5, la tabla ‘correspondencia_id’ con la nueva entrada generada por medio del trigger asociado a la tabla ‘webcal_entry_user’.

web_ID	fun_ID	accion	sync_to	estado
25	NULL	NEW	FUN	NOT_SYNC

FIGURA 6-5: EVENTO DE PRUEBA – TABLA CORRESPONDENCIA

Fuente propia

Posteriormente, el evento SQL que se ya está ejecutando internamente cada minuto, realiza la sincronización del evento del WebCalendar hacia el Funambol y actualiza la tabla ‘correspondencia_id’ con el nuevo valor del ID del evento Funambol generado.

web_ID	fun_ID	accion	sync_to	estado
25	160	NEW	FUN	SYNC

FIGURA 6-6: EVENTO DE PRUEBA – TABLA CORRESPONDENCIA ACTUALIZADA

Fuente propia

Se puede observar que el ID del evento generado en el Funambol es ‘166’. Se procede entonces a mostrar en la figura 6-7 la tabla ‘fnbl_pim_calendar’ sincronizada.

id	userid	last_update	status	all_day	body	dstart	dend	location	reminder	subject
160	stuardo.lucho	1320806777000	N	0	descripción prueba	2011-11-11 08:00:00	2011-11-11 10:00:00	DIA PUCP	0	Prueba 1

FIGURA 6-7: EVENTO DE PRUEBA – TABLA FNBL_PIM_CALENDAR

Fuente propia

Finalmente se realizó la captura de pantalla de un celular Blackberry con el software Funambol instalado, en el cual con las credenciales adecuadas se procedió a sincronizar este evento creado. Se puede observar en la figura 6-8.

FIGURA 6-8: PRUEBA DEL PROCESO DE SINCRONIZACIÓN

Fuente propia

6.2 Pruebas de rendimiento

Luego de implementar la solución y realizar las pruebas pertinentes, se realizó una última prueba para comprobar el rendimiento del sistema con y sin la solución implementada. La razón de la misma es observar dos factores:

1. Si la cantidad de datos transmitidos entre el servidor y el dispositivo móvil al realizarse una sincronización de eventos aumenta.
2. Si existe mayor retardo de llegada de los paquetes al dispositivo móvil al realizarse una sincronización.

Para estas pruebas se utilizó el analizador de protocolos Wireshark, además de tener el sistema implementado en un servidor local. El dispositivo móvil cuenta con acceso a la red wi-fi con dirección ip: 192.168.1.39; mientras que el servidor que se encuentra en la misma red local que el dispositivo móvil, cuente con una dirección ip: 192.168.1.33.

En primer lugar se realizó la prueba con un dispositivo móvil Blackberry 9700, con el software Funambol en su versión 10.0.7. Esta primera prueba se realizó sin la solución implementada. En la figura 6-9 se puede observar la sincronización de 9 eventos que están siendo enviados desde el dispositivo móvil (usando la red wi-fi) hacia el servidor.

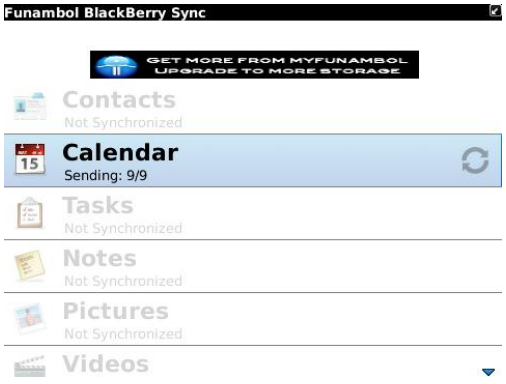

FIGURA 6-9: PRUEBA DE SINCRONIZACIÓN - CLIENTE

Fuente propia

Posteriormente, en el lado del servidor, estaba corriendo el wireshark, capturando tramas tal como se muestra en la figura 6-10, todas las tramas de la comunicación entre el cliente y el servidor.

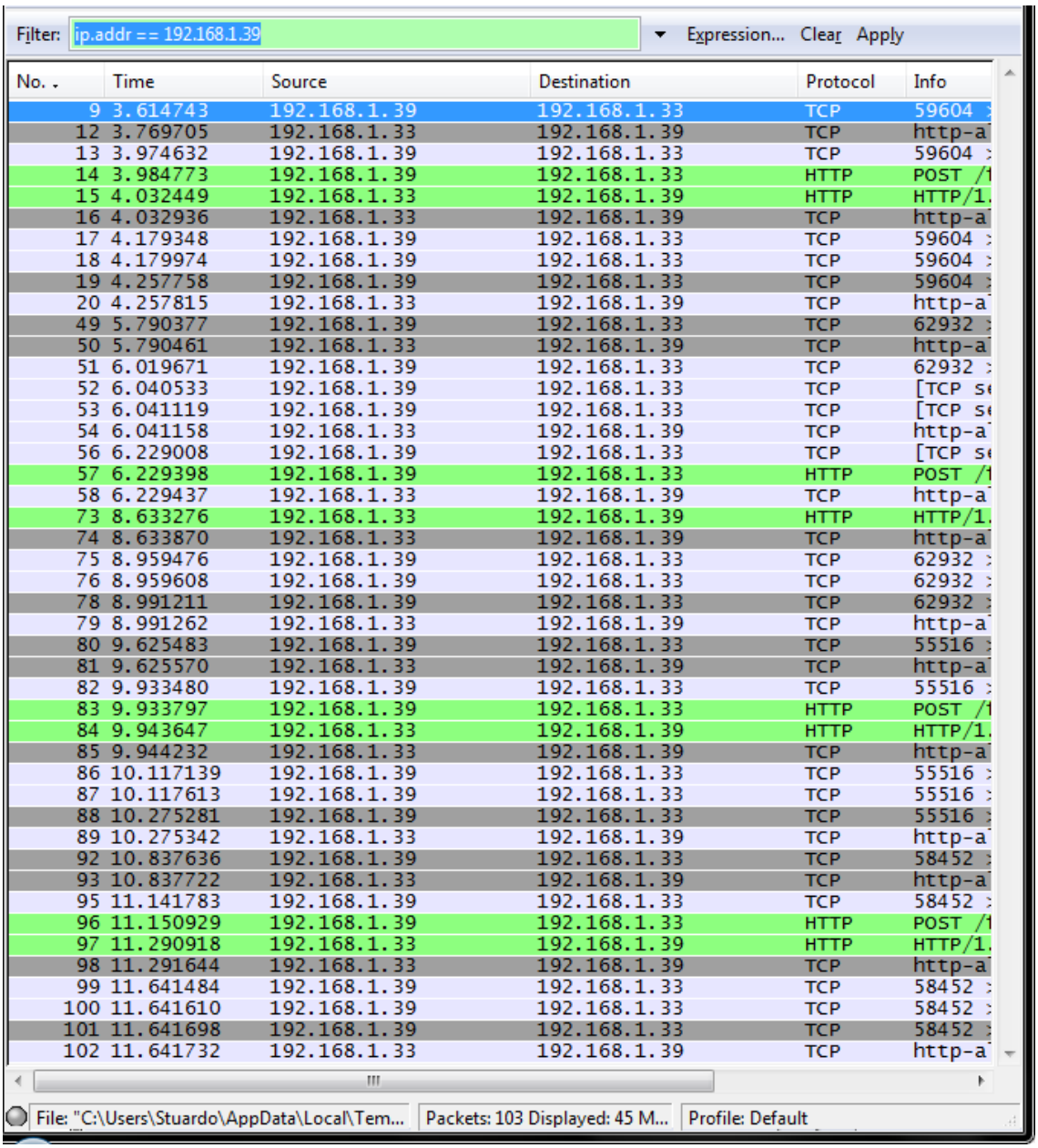

FIGURA 6-10: TRAMAS CAPTURADAS – SERVIDOR, PRUEBA 1

Fuente propia

En base a las tramas capturadas se obtuvo las estadísticas filtradas con ip.addr == 192.168.1.39 y se obtuvo que la comunicación entre el cliente y servidor duró un aproximado de 8 segundos y 14 KBytes de datos transferidos. El cuadro estadístico se puede observar en la figura 6-11.

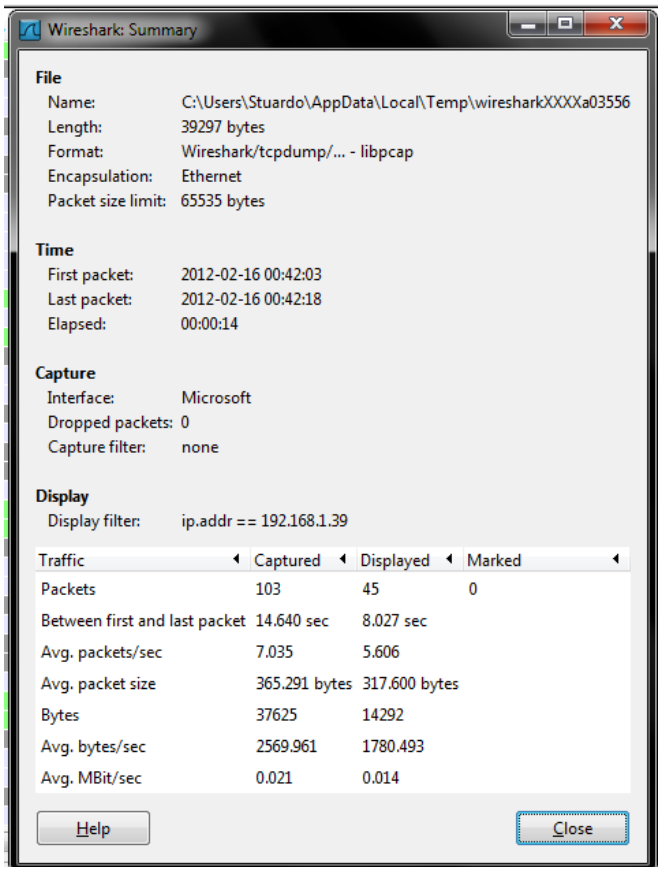

FIGURA 6-11: CUADRO ESTADÍSTICO, PRUEBA 1

Fuente propia

Con estos primeros datos se procede a realizar nuevamente la prueba pero esta vez teniendo la solución implementada. El ambiente, configuraciones y dispositivos son los mismos.

Tal como se muestra en la figura 6-9, nuevamente se sincronizaron 9 eventos desde el dispositivo móvil hacia el servidor. En el lado del servidor, al igual que la primera prueba se utilizó el Wireshark para capturar las tramas, mostrado en la figura 6-12 y analizar sus estadísticas, las mismas que se muestran en la figura 6-13.

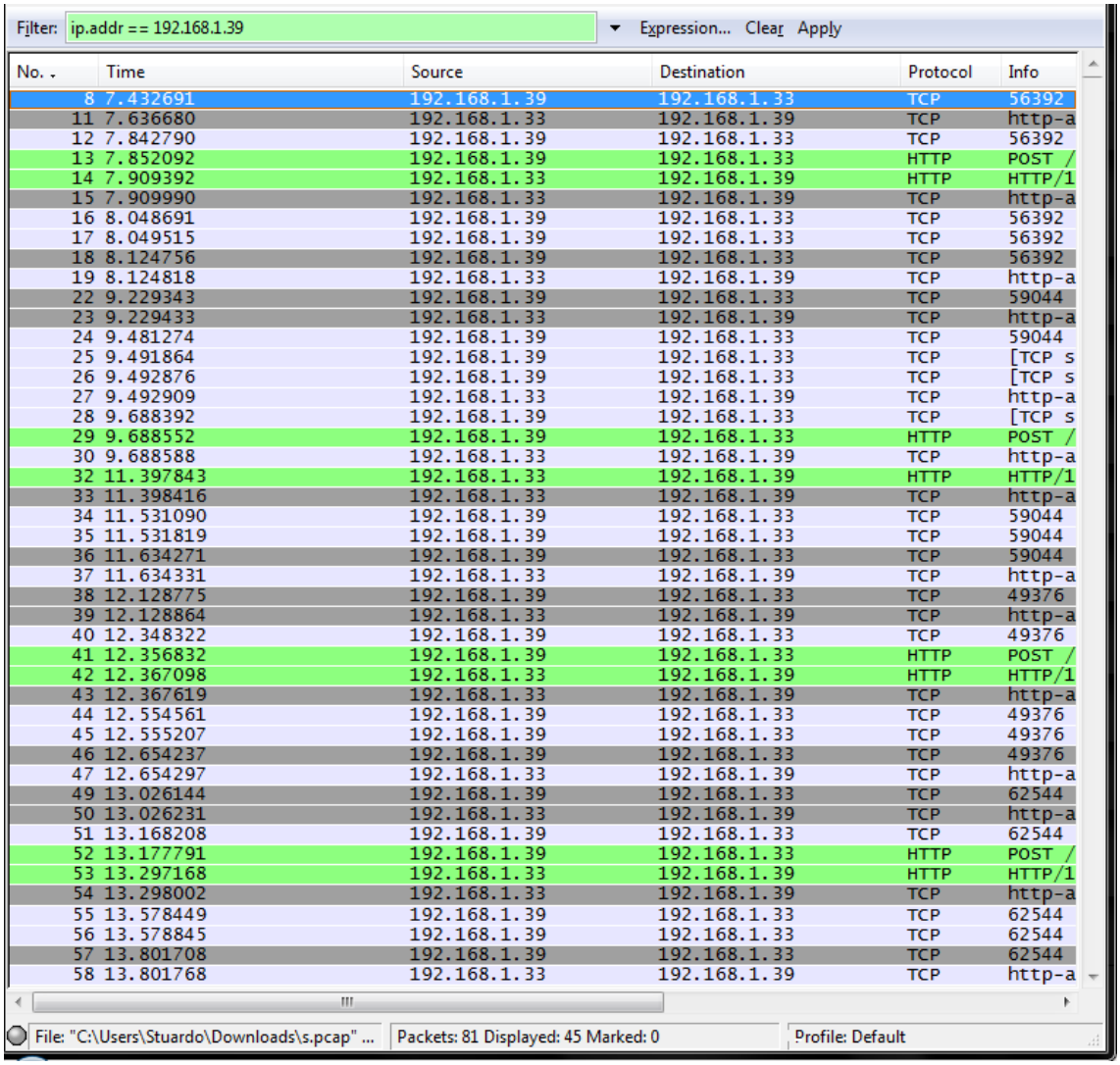

FIGURA 6-12: TRAMAS CAPTURADAS – SERVIDOR, PRUEBA 2

Fuente propia

De la misma manera se obtuvieron sus estadísticas, mostradas en la figura 6-13, de donde se obtuvo la información que, la comunicación entre el cliente y servidor duró un aproximado de 6 segundos y 14 Kbytes de datos transferidos.

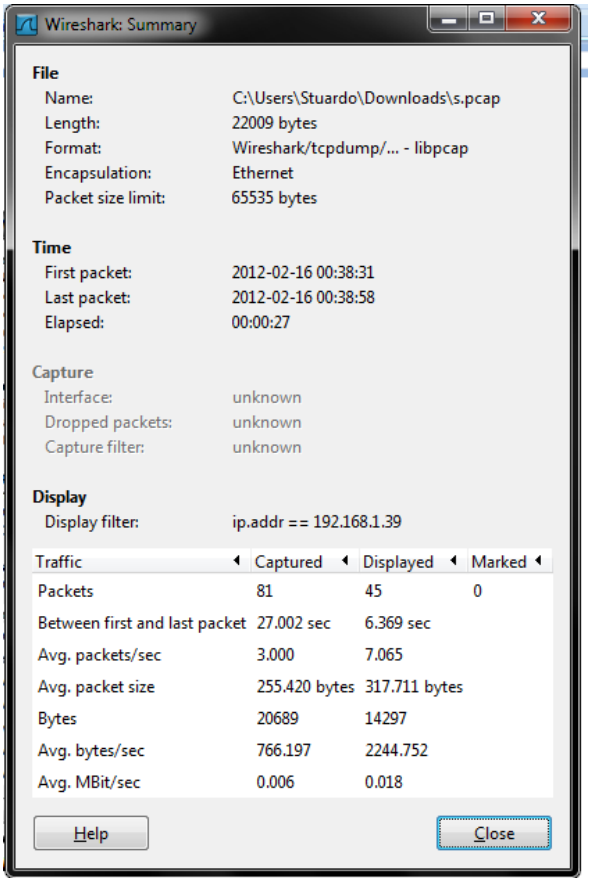

FIGURA 6-13: CUADRO ESTADÍSTICO, PRUEBA 2

Fuente propia

Vale decir que no sólo se tomaron 1 prueba de cada caso (son y sin solución implementada), sino que se realizaron 7 pruebas para cada caso, todas con las mismas variables:

- 9 eventos sincronizados.
- Usando una red Wi-fi en un entorno local.

Los resultados mostrados son los más regulares, es decir, los que más se repitieron en la mayoría de las pruebas.

En conclusión a la hipótesis inicial, se puede observar en ambos cuadros estadísticos, que el tiempo que demoró la comunicación entre cliente y servidor, con y sin solución implementada, es muy parecido en ambos casos y además la cantidad de datos transferidos presenta alta similitud, con diferencias del orden de los bytes.

Antes estos resultados, se puede afirmar que la solución implementada no afecta al rendimiento inicial que tenía el sistema.

Conclusiones

Luego de lograr la interconexión de los sistemas y su implementación se ha llegado a las siguientes conclusiones:

- Se logró la implementación de un sistema de administración de calendarios mediante la integración transparente de aplicaciones basadas en software libre

- La interconexión de sistemas basados en distintos paradigmas de funcionamiento es completamente posible y funcional por medio de sus bases de datos, en particular usando MySQL.

- Al tener a disposición el código fuente de los sub-sistemas usados para el diseño del sistema final, esto permite que ante una eventual evolución de alguno de los sub-sistemas se puedan realizar las modificaciones necesarias para seguir manteniendo la integración de manera óptima.

- Como parte de la solución usando el paradigma de triggers simples, en combinación con un evento SQL, programado cada minuto, se obtuvo un incremento de almacenamiento de información del sistema en la base de datos, de 0.009 segundos, tiempo aceptable, considerando que previo a la implementación de la solución, el sistema demoraba en almacenar en la base de datos 0.005 segundos.

- Se comprobó que el tiempo y los datos transmitidos entre el cliente móvil y el servidor no aumentan con la implementación de la solución, sino que mas bien se mantienen constantes y no afecta al rendimiento original del sistema (Funambol).

Recomendaciones y trabajos futuros

- Al trabajar con bases de datos, se debe tener especial cuidado con el juego de caracteres que se usa en ambos sistemas, pues mientras uno hace uso de iso-8891-1 el otro usa utf-8, lo cual trae serias complicaciones para el manejo de caracteres especiales como la “ñ”. Por esta razón es aconsejable trabajar todo el sistema con una sola codificación.

- Se recomienda, como trabajo futuro, integrar las bases de datos de ambos sistemas, de esta manera se podría eliminar la repetición del mismo evento en dos bases de datos distintas.

- Debido a que para implementar el servicio es necesario instalar varios sistemas y posteriormente insertar lo triggers, el evento SQL y los procedimientos almacenados, todo esto sin contar las modificaciones puntuales al código fuente del sistema web. Un trabajo a futuro es la creación de un instalador, el cual posea ambos sistemas, todas las sentencias SQL y las modificaciones del código fuente, de manera que el usuario que desea implementar el sistema sólo ejecuta el instalador y tras una serie de pasos queda instalado y configurado de manera automática y transparente.

- Puesto que el sistema es implementado para la Pontificia Universidad Católica del Perú, un trabajo a futuro es desarrollar el sistema con un grado mayor de personalización y flexibilidad para quien desee hacer uso del mismo, brindado la posibilidad de no limitarlo a una institución; sino, que pueda ser configurado para un entorno externo como el empresarial.

Bibliografía

[ARQ2010] Arquitectura cliente-servidor.
URL:http://www.di.uniovi.es/~labra/cursos/Web20/images/WWW.png
Consultada el 3 de junio del 2010.

[BED2010] Bedework. URL: http://www.bedework.org/bedework/setup.do
Consultada el 4 de julio del 2010.

[CLI2010] Client-Server Architecture. Computer Science Program, The University of Texas, Dallas. URL: http://www.utd.edu/~chung/SA/2client.pdf
Consultada el 2 de junio del 2010.

[CMS2010] Arquitectura cliente móvil-servidor.
URL: http://www.mailxmail.com/curso-tecnicas-desarrollo-computacion-movil-orientado-pda/procesos-desarrollo-aplicaciones-moviles
Consultada el 3 de junio del 2010.

[EVS2010] Event Scheduler Overview.
URL: http://dev.mysql.com/doc/refman/5.1/en/events-overview.html

[FAG2010] Guía de administración del Funambol.
URL: http://download.forge.objectweb.org/sync4j/Funambol-installation-and-administration-guide-v8.7.pdf
Consultada el 24 de noviembre del 2010.

[FSD2010] Funambol soppoted devices.
URL: http://www.funambol.com/solutions/devices.php
Consultada el 4 de julio del 2010

[FSF2010] Free software foundation. URL: http://www.fsf.org/
Consultada el 24 de mayo del 2010

[FUN2010] Funambol. URL: http://www.funambol.com/
Consultada el 4 de julio del 2010

[GOO2010] Google Calendar. URL: www.google.com/calendar/
Consultada el 3 de julio del 2010

[GOS2010] Google Sync. URL: http://www.google.com/mobile/sync/
Consultada el 4 de julio del 2010

[GRL2010] Código fuente de Recaptchalib. URL:http://www.google.com/recaptcha
Consultada el 19 de agosto del 2010

[ICA2009] RFC 5545. Internet Calendaring and Scheduling Core Object Specification (ICalendar). URL: http://tools.ietf.org/html/rfc5545
Consultada el 20 de mayo del 2010

[IPH2010] SKLAR, David. “Introducción a PHP 5”. Primera edición. España, 2005: Anaya Multimedia.

[JVM2010] Página principal Java. URL: http://www.java.com/es/about/
Consultada el 28 de mayo del 2010

[LIC2010] Licencia libres.
URL: http://www.maestrosdelweb.com/editorial/licencias-libres-de-software-ii/
Consultada el 12 de julio del 2010

[MYS2010] Jay, Randy. Reese, George. King, Tom. “MySQL and mSQL”. Primera edición. EE.UU, 1999: O’Reilly & Associates.

[NET2010] Netbeans IDE. URL: http://www.netbeans.org
Consultada el 22 de junio del 2010

[OMA2010] Open Mobile Alliance. URL: http://www.openmobilealliance.org
Consultada el 20 de mayo del 2010

[OSD2010] Definición de Open source. URL: http://www.opensource.org/docs/osd
Consultada el 25 de mayo del 2010

[PHM2010] Página principal PHPMailer.
URL: http://phpmailer.worxware.com/index.php
Consultada el 22 de setiembre del 2010

[PHP2010] Página principal PHP. URL: http://php.net/index.php
Consultada el 27 de mayo del 2010

[PHS2010] Requerimientos para PHP. URL: http://www.php.net/manual/es/tutorial.requirements.php
Consultada el 22 de junio del 2010

[PMA2010] PhpMyAdmin. URL: http://www.phpmyadmin.net/home_page/support.php
Consultada el 22 de junio del 2010

[SIF2010] Guía para los desarrolladores del Funambol versión 8.5. URL: http://download.forge.objectweb.org/sync4j/funambol-developers-guide.pdf
Consultada el 14 de noviembre del 2010

[SOG2010] Página principal SOGo. URL: http://www.sogo.nu/
Consultada el 5 de junio del 2010

[SOV2010] Resumen SOGo. URL: http://www.sogo.nu/about/overview.html
Consultada el 4 de julio del 2010

[SQL2010] Página principal MySQL. URL: http://www.mysql.com
Consultada el 31 de mayo del 2010

[SUN2010] Sun MicroSystems. URL: http://es.sun.com/
Consultada el 28 de mayo del 2010

[SWL2010] Definición de software libre.
URL: http://www.gnu.org/philosophy/free-sw.es.html
Consultada el 24 de mayo del 2010

[SYN2010] SyncML White Paper. Open mobile Alliance.
URL:http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_whitepaper.html
Consultada el 20 de mayo del 2010

[WEB2010] Página principal de WebCalendar.
URL: http://www.k5n.us/webcalendar.php
Consultada el 7 de julio del 2010

[WLC2010] Windows Live Calendar. URL: http://windowslive.com/Online/Calendar
Consultada el 3 de julio del 2010

[YAH2010] Yahoo Calendar. URL: http://calendar.yahoo.com/
Consultada el 3 de julio del 2010

[YAN2010] Yahoo! Help.
URL: http://help.yahoo.com/l/us/yahoo/calendar/yahoocalendar/sync/sync-07.html
Consultada el 4 de julio del 2010

Anexos