Universidad de las Ciencias Informáticas

Facultad 2

Título: Desarrollo del Sistema de Registro Detallado de Llamadas y Facturación de la plataforma Platel.

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores:

Yunior Reyes Pedraza

Lenier Pérez León

Tutor: Ing. Amed Vázquez Pérez

Ciudad de La Habana, 2010.

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Yunior Reyes Pedraza

_____

Firma del autor

Lenier Pérez León

_____

Firma del autor

Ing. Amed Vázquez Pérez

_____

Firma del tutor

RESUMEN

El presente trabajo comprende la elaboración de un sistema de administración que brinde la posibilidad de agilizar el proceso de generación de reportes y facturación de una PBX basada en Asterisk para la plataforma telefónica PlaTel. La automatización de dicho proceso se hace necesaria debido a que en este momento tanto la configuración como la obtención de información para los reportes y la facturación se hace de forma manual en ficheros de texto que se encuentran en el servidor. La elaboración de reportes y facturas, es un eslabón esencial dentro de los servicios que brinda cualquier PBX, y si a esto se le agrega la posibilidad de hacerlo de forma visual y remota mediante una aplicación Web, se garantiza un mejor funcionamiento de la Plataforma. En Cuba no ha sido elaborado un sistema con estas características, y aquellos que existen a nivel mundial son vendidos por grandes sumas de dinero o en ocasiones no satisfacen las necesidades. Es por ello que el presente sistema está diseñado para optimizar dicho proceso y cumplir con la meta de soberanía tecnológica, aspecto importante dentro del desarrollo de nuestra sociedad.

PALABRAS CLAVE

PBX, Asterisk, PlaTel, VoIP, aplicación web.

TABLA DE CONTENIDOS

Tabla de contenidos

Tabla de contenidos

Introducción

La humanidad hoy en día se encuentra indisolublemente ligada con el desarrollo vertiginoso de las tecnologías y las comunicaciones, esto explica que estos dos factores logren regir su comportamiento y se conviertan en partes indispensables de sus actividades cotidianas.

Un punto importante a tener en cuenta dentro de la informatización de la sociedad es el tema de las telecomunicaciones. Actualmente con la gran evolución de esta rama, la comunicación universal por Internet ha crecido de manera explosiva. La transmisión de voz en paquetes sobre redes ha experimentado grandes avances en los últimos años, tanto por el desarrollo de estándares como por la aparición de productos basados en tecnología IP.

VoIP por sus siglas en inglés (Voz sobe IP) es una tecnología avanzada que ha sobrepasado a las compañías del teléfono y a compañías móviles. Su renombre está creciendo enormemente con un aumento en su cobertura de la red. Ha demostrado ser una invención asombrosa la cual ha creado un nuevo modo de la telecomunicación usando Internet. También con la introducción de esta tecnología la fabricación de llamadas ha llegado a ser mucho más fácil y barata. VoIP cuenta con varios servicios de valor añadido ofrecidos por sus abastecedores. Por ejemplo, existe facilidad del correo de voz, conferencia, expedición de llamada, anillo simultáneo e identificación de llamador entre muchos otros. Todos estos servicios están sobre todo libres de coste en su plan del servicio básico.

Ligado estrechamente a este desarrollo han surgido un sin número de plataformas telefónicas, y un ejemplo de esto es Asterisk.

Asterisk PBX es de las más utilizadas por ser Libre, y sobre todo por la gran comunidad con que cuenta a nivel mundial. Esta plataforma a base de software brinda todas las funcionalidades de aquellas desarrolladas por las más reconocidas empresas a nivel mundial.

Entre las funcionalidades más notables de Asterisk PBX se encuentra un completo sistema de reportes y facturación, con el objetivo de dar a los usuarios la posibilidad de controlar el tráfico de llamadas dentro de la plataforma, y realizar el cobro del servicio de telefonía.

Para llevar a cabo la administración de esta plataforma telefónica existen varios softwares de administración dentro de los que se pueden destacar Elastix y TrixBox, este último se encuentra desplegado en las Fuerzas Armadas Revolucionarias (FAR).

Cuba y principalmente las FAR se encuentran inmersas en la automatización del país con el objetivo principal de situarse a la vanguardia de la sociedad.

Actualmente en la FAR se encuentra en desarrollo la plataforma telefónica PlaTel, un producto totalmente nacional e independiente con capacidad autónoma de soporte técnico. PlaTel emplea como elemento fundamental de su núcleo la PBX Open Source Asterisk y como sistema de administración AdminPlaTel.

Existen diferentes softwares de administración para la PBX Asterisk, destaca entre ellos FreePBX, dentro de las principales funcionalidades con que cuenta se puede destacar un Registro Detallado de Llamadas(CDR) y la opción de generar reportes dependiendo de criterios. Pero este sistema no implementa la facturación de las llamadas y no cuenta con gráficas que ilustren su comportamiento, además está concebido para usuarios con un alto nivel de conocimientos en temas de informática y telecomunicaciones. Al mismo tiempo, su arquitectura es complicada, y por tanto difícil de modificar. También FreePBX v3 se encuentra bajo la Licencia Pública de Mozilla (MPL), mientras Asterisk por su parte es cubierto por la Licencia Pública General (GPL), lo cual dificulta considerablemente su utilización, debido a que no se puede, legalmente, enlazar un módulo GPL con un módulo MPL.

Debido a esta situación se plantea como **problema científico**: ¿Cómo gestionar el Registro Detallado de Llamadas y Facturación en el Sistema de Administración de la plataforma PlaTel de forma personalizada?

Por tanto, el presente trabajo centra su **objeto de estudio** en los diferentes sistemas de administración de plataformas telefónicas, derivándose de este como **campo de acción** el Registro Detallado de Llamadas y Facturaciones en el Sistema de Administración de la plataforma PlaTel.

Para dar solución al problema existente, se ha tomado como **objetivo general**: Desarrollar los módulos de CDR y Facturación en el Sistema de Administración de la plataforma PlaTel.

Objetivos específicos:

1. Describir el funcionamiento del Sistema de Administración de la plataforma PlaTel.

    1.1. Realizar el análisis de los procesos necesarios para el funcionamiento del CDR en el Sistema de administración de la plataforma PlaTel.

    1.2. Realizar el análisis de los procesos necesarios para realizar la Facturación en el Sistema de Administración de la plataforma PlaTel.

2. Realizar el diseño de los módulos necesarios para la gestión del CDR en el Sistema de Administración de la plataforma PlaTel.

3. Realizar el diseño de los módulos necesarios para la gestión de la Facturación en el Sistema de Administración de la plataforma PlaTel.

4. Implementar los módulos definidos para el CDR en el Sistema de Administración de la plataforma PlaTel.

5. Implementar los módulos definidos para la Facturación en el Sistema de Administración de la plataforma PlaTel.

Idea a defender:

Si se implementan lo módulos de CDR y Facturación satisfactoriamente y cumpliendo todos los pasos definidos en el Proceso de Desarrollo de Software del centro UCID entonces se logrará brindar un servicio satisfactorio y personalizado en el Sistema de Administración de la plataforma PlaTel.

Capítulo 1: Fundamentación Teórica

1. Introducción.

En este capítulo se abarcará el estado actual de los Sistemas de Reportes y Facturación en las PBX basadas en Asterisk a nivel mundial y nacional. Se presentarán las herramientas, tecnologías, y la metodología utilizada para el desarrollo.

1.1. Asterisk.

La telefonía VoIP es una tecnología que ha revolucionado las comunicaciones, para su implementación existen soluciones hardware y software. Por sus limitaciones las primeras cada día están más en desuso, poca flexibilidad y un elevado costo destacan dentro de los inconvenientes que presentan. Por el contrario, las soluciones software ofrecen una gran flexibilidad a un coste más factible y justificado siendo incluso algunas de ellas gratuitas.

Asterisk destaca dentro de las soluciones software por ser una aplicación de software libre bajo licencia GPL, capaz no solo de proveer las funciones básicas de una central telefónica de bajo coste, sino que incluye muchas características anteriormente sólo disponibles en caros sistemas propietarios PBX: buzón de voz, conferencias, distribución automática de llamadas, y otras muchas más. Los usuarios pueden crear nuevas funcionalidades escribiendo un dialplan en el lenguaje de script de Asterisk o añadiendo módulos escritos en lenguaje C o en cualquier otro lenguaje de programación soportado por Linux. (1)

Además, al ser software brinda una mayor escalabilidad y flexibilidad que el resto de las soluciones disponibles en el mercado. Entre sus funcionalidades cuenta con un completo sistema de reportes y facturación, con el objetivo de dar a los usuarios la posibilidad de controlar el tráfico de llamadas dentro de la plataforma, y realizar el cobro del servicio de telefonía.
Existen varios software de administración para la PBX Asterisk, dentro de los que destacan FreePBX, Elastix y A2billing, los cuales se analizarán a continuación.

1.2. Sistemas de Reportes y Facturación en el mundo.

En el mundo existen sistemas basados en Asterisk como: FreePBX, Elastix, A2billing. Estos sistemas proveen la posibilidad de consultar reportes detallados de las llamadas o brindan la posibilidad de poder configurar su facturación; por lo general cuentan con interfaces amigables para el usuario, aunque no disponen de una sólida reportería y facturación en conjunto. Los mencionados sistemas cuentan con ventajas y desventajas que serán expuestas a continuación.

1.2.1. Free PBX.

FreePBX es una herramienta dedicada a la administración de PBX Asterisk. Es muy utilizado por ser un software de telefonía con funciones completas de aplicación web. Cuenta con una interfaz de usuario amigable que permite tener un Asterisk completamente funcional casi de inmediato, evitando tener que editar los ficheros de configuración manualmente. Entre sus principales funcionalidades cuenta con un Registro Detallado de Llamadas y la opción de generar reportes dependiendo de criterios. Este sistema no implementa la factura de las llamadas y no cuenta con gráficas que ilustren el comportamiento de las mismas. (2)

Asimismo está concebido para usuarios con un alto nivel de conocimientos en temas de informática y telecomunicaciones. Al mismo tiempo, su arquitectura es complicada, y por tanto difícil de modificar Además el cambio de su licencia GPL por MPL se ha convertido en su principal desventaja, debido a que no se puede, legalmente, enlazar un módulo cubierto por la GPL con un módulo MPL.

1.2.2. Elastix.

Elastix es un software aplicativo que integra las mejores herramientas disponibles para PBXs basados en Asterisk en una interfaz simple y fácil de usar. (3). Integra varias herramientas disponibles para centrales telefónicas basadas en PBX Asterisk. Entre otras funcionalidades presenta un módulo para generar varios reportes aunque estos reportes no brindan suficiente información al no ser reportes comparativos, ni evolutivos; conjuntamente implementa la facturación de llamadas. El sistema está realizado sobre plataforma Libre, pero al integrar varios módulos de otros sistemas, dificulta el soporte, que tiene que ser a través de cursos pagados que se imparten de forma online.

1.2.3. A2billing.

A2billing es un completo sistema de facturación para Asterisk, es decir, que muestra el importe a cobrar de acuerdo con el tiempo de duración y destino de una llamada. El administrador puede modificar y ejecutar tareas que controlen el sistema a través de una interfaz web. Permite obtener datos estadísticos de la actividad y minutos consumidos por destino. Este sistema está realizado sobre plataforma Libre y posee gran auge gracias a su sencillez, su documentación es abundante aunque con respecto a los reportes no posee gran fortaleza, ya que se centra principalmente en la factura. Otra de las desventajas que presenta es que no permite la integración con base de datos como SQLite y Oracle; además de que muchas de las configuraciones necesarias para su trabajo se hacen sobre ficheros. (4)

1.3. Sistema de Reportes y Facturación en Cuba y en la Universidad de las Ciencias Informáticas.

Actualmente consultar las llamadas y la facturación de una PBX basada en Asterisk en instituciones cubanas es un proceso muy complejo y tedioso, debido a que no se cuenta con una interfaz que muestre los datos almacenados en ficheros de configuración localizados en la PC servidor, haciendo necesaria su búsqueda manual, para posteriormente realizar su consulta de una forma poco eficaz, pues en el fichero se encuentran las estadísticas de todas las llamadas gestionadas por el servidor, y no es posible efectuar una búsqueda rápida de las mismas por criterios, con el objetivo de obtener reportes detallados.

Luego de elaborar un análisis que comprende las aplicaciones anteriormente expuestas, surge la necesidad de la confección de un sistema de administración que reúna las mejores características de los existentes hasta el momento, agregando otras funcionalidades que necesite el cliente, garantizando así que cumpla con las exigencias del mismo. De esta manera, también se contribuye con la ardua lucha por alcanzar la soberanía tecnológica que lleva a cabo nuestro país.

1.4. Framework.

El concepto framework se emplea en muchos ámbitos del desarrollo de sistemas software, no solo en el ámbito de aplicaciones Web. Se pueden encontrar para el desarrollo de aplicaciones médicas, de visión por computador, para el desarrollo de juegos, entre otros.

Un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Se ha convertido en la piedra angular de la moderna ingeniería del software. Asimismo es una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, es una aplicación genérica incompleta y configurable a la que se le puede añadir las últimas piezas para construir una aplicación concreta. (5)

1.4.1. Sauxe Framework.

Sauxe es el marco de trabajo que se utiliza en el centro UCID para desarrollar aplicaciones web. Presenta una arquitectura en capas (presentación, negocio, servicio, dominio y acceso a datos, datos). Además, utiliza el estilo Modelo Vista Controlador para integrar la capa de presentación con las capas de negocio y/o servicio. Asimismo emplea varias tecnologías y herramientas, como es el caso de:

- Zend Framework.
- Doctrine.
- ExtJS.
- Lenguaje de programación PHP.
- Gestor de base de datos: PostgreSQL.

1.4.1.1. Estilo Modelo-Vista-Controlador (MVC).

El estilo Modelo-Vista-Controlador está formado por tres niveles: el modelo, la vista y el controlador.

El modelo: representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.

La vista: transforma el modelo en una página web que permite al usuario interactuar con ella.

El controlador: se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. Si por ejemplo una misma aplicación

debe ejecutarse tanto en un navegador estándar como un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email, etc.). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación. (6)

Cuenta con una gran cantidad de **ventajas**, entre las que destacan:

- Hay una clara separación entre los componentes de un programa; lo cual permite implementarlos por separado.
- Tiene un API (Application Programming Interface o Interfaz de Programación de Aplicaciones) muy bien definido; cualquiera que use el API, podrá reemplazar el Modelo, la Vista o el Controlador, sin aparente dificultad.
- La conexión entre el Modelo y sus Vistas es dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.

1.4.1.2. Zend Framework.

Zend Framework es un framework de código abierto para el desarrollo de aplicaciones y servicios web con PHP 5. Está implementado utilizando código 100% orientado a objetos. Los componentes tienen poco acoplamiento entre ellos lo cual permite a los desarrolladores utilizarlos separadamente. Ofrece una implementación MVC robusta y de alto rendimiento, abstracción para interactuar con bases de datos, un componente que implementa el renderizado de formularios HTML, validación y filtrado.

Presenta entre otras las siguientes **ventajas**:

- Es desarrollado por Zend que es la empresa que respalda comercialmente a PHP.
- Trabaja con MVC.
- Cuenta con módulos para manejar archivos PDF, canales RSS, Web Services (Amazon, Flickr, Yahoo).

- Simplifica la gestión de archivos de configuración.
- Soporte avanzado para i18n (internacionalización).
- Un buscador compatible con Lucene.
- Clientes para servicios web, incluidos Google Data APIs y StrikeIron.

1.4.1.3. ExtJS.

ExtJS es un framework de JavaScript que permite construir aplicaciones complejas en Internet. Utiliza AJAX, DHTML, CSS y DOM. Cuenta con componentes que le permiten crear y manipular DataGrids, los cuales le proporcionan cierta ventaja sobre otros frameworks. Es posible crear "ventanas" con Barras de Herramientas y Menús con estilo de aplicaciones de Escritorio, Diálogos modales y eventos. Además, tiene una API fácil de usar, es rápido y posee widgets personalizables. Presenta licencia open source y comercial. Funciona en la mayoría de los navegadores. Al utilizar ExtJS existe un balance entre Cliente–Servidor. La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo. (7)

1.4.1.3.1. DHTML.

DHTML son las siglas en inglés de Lenguaje de Marcado de Hipertexto Dinámico, o simplemente HTML Dinámico. No se trata de un único lenguaje sino de un término genérico para referirse a los últimos lenguajes de creación de páginas web que permiten aportar interactividad y animación a las mismas. Hace referencia a páginas desarrolladas usando una combinación de HTML, JavaScript, hojas de estilo en cascada (CSS), incluyendo la extensión para posicionamiento (CSSP), uso de capas (layers) y divisiones de página (DIV). (8)

1.4.1.3.2. JavaScript.

Es un lenguaje de programación, -tipo script- que se utiliza en las páginas web, permitiendo crear efectos especiales, interactuar con el visitante y demás funciones interesantes. Aunque sea interpretado por el navegador (y no interactúe con ningún servidor) JavaScript es bastante poderoso. Con él se pueden controlar los elementos de la página (validación de formularios, entre otros). Otra función muy importante es la utilización de cookies, que permite reconocer a usuarios que ya visitaron la página. (9)

Es multiplataforma y parcialmente orientado a objetos, desarrollado para incrementar las funcionalidades del lenguaje HTML.

Ventajas:

- Permite elaborar aplicaciones Web que simulen características de aplicaciones escritorios.
- Es un código "interpretado" por el cliente.
- Es un código orientado a objetos.
- Es un código integrado a HTML.
- Reutilización de código de programación.
- El lenguaje de scripting es seguro y fiable.
- El código es visible y puede ser leído por cualquiera, incluso si está protegido con las leyes del copyright.

Unos de los inconvenientes que tiene este lenguaje es que tiene que estar activado en los navegadores para que su uso sea posible por estos. Por otro lado, no le proporciona al programador control total de la página web. En ocasiones los desarrolladores deben realizar diferentes implementaciones de código JavaScript para sus aplicaciones, debido a que no todos los navegadores interpretan de la misma manera el código.

1.4.1.3.3. CSS.

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación; y es imprescindible para crear páginas web complejas. Dicho lenguaje se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista.

Ventajas:

- Obliga a crear documentos HTML/XHTML bien definidos y con significado completo.
- Mejora la accesibilidad del documento.
- Reduce la complejidad de su mantenimiento.
- Permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Desventajas:

Incompatibilidad entre diferentes navegadores, por lo que si algo se ve excelente en un navegador es posible que en otro no se vea igual, o que varíe la visualización incluso en las diferentes versiones de un mismo navegador.

1.4.1.3.4. AJAX.

AJAX, acrónimo de JavaScript asíncrono y XML, no es una tecnología en sí misma; es una técnica de desarrollo web para crear aplicaciones interactivas. En realidad, se trata en la unión de varias tecnologías independientes. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios, mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma, es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad, usabilidad en las aplicaciones y mejor respuesta a las acciones del usuario. Es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores, dado que está basado en estándares abiertos como HTML o, XHTML, JavaScript, DOM, XML, CCS, XSLT, y el objeto XMLHttpRequest. (10)

1.4.1.4. Doctrine.

Doctrine es un ORM (por sus siglas en inglés Object Relational Mapping) para PHP 5.2.3+ que se encuentra en la parte superior de una poderosa capa de abstracción de bases de datos (DBAL), proporciona a los desarrolladores una poderosa alternativa SQL que mantiene la flexibilidad, sin necesidad de la duplicación de código innecesaria. Consiste en una serie de objetos que permiten acceder a los datos y que contienen en su interior cierta lógica de negocio. Las bases de datos siguen una estructura relacional. PHP 5 y los framework por el contrario son orientados a objetos. Por este motivo, es necesaria una interfaz que traduzca la lógica de los objetos a la lógica relacional para

acceder a la base de datos como si fuera orientada a objetos, y es específicamente esto lo que hace Doctrine. (11)

Una de las ventajas de utilizar estas capas de abstracción de objetos/relacional es que evita utilizar una sintaxis específica de un sistema de bases de datos concreto. Esta capa transforma automáticamente las llamadas a los objetos en consultas SQL optimizadas para el sistema gestor de bases de datos que se está utilizando en cada momento.

1.4.1.5. Lenguaje de Programación: PHP.

PHP es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, similar al ASP de Microsoft o el JSP de Sun, embebido en páginas HTML y ejecutado en el servidor. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. No es un lenguaje de marcas como podría ser HTML, XML o WML, está más cercano a JavaScript o a C. (12)

Uno de sus principales problemas consistía en que la orientación a objeto no estaba bien lograda. Pero con PHP 5 se incorpora una verdadera orientación a objeto. Se agregan las palabras reservadas public, protected y prívate a la definición de las propiedades y métodos de los objetos, se permite una verdadera encapsulación. Además del considerable avance con respecto a los objetos, PHP 5 incorpora un control de errores muy mejorado, al estilo de los lenguajes de programación más avanzados.

Al ser un lenguaje libre dispone de una gran cantidad de **características** que lo convierten en la herramienta ideal para la creación de páginas web dinámicas: (12)

- Soporte para una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, Sybase mSQL, Informix, entre otras.
- Se caracteriza por ser un lenguaje muy rápido.
- Ofrece una solución simple y universal para las paginaciones dinámicas del Web de fácil programación.

- Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Soportado por una gran comunidad de desarrolladores, como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente.
- Incorpora un nuevo modelo de objetos que permite crear clases y métodos privados, protegidos y públicos, clases abstractas e interfaces.
- Es un lenguaje multiplataforma: Linux, Windows, entre otros.
- Con PHP se puede hacer cualquier cosa que podemos realizar con un script CGI, como el procesamiento de información en formularios, foros de discusión, manipulación de cookies y páginas dinámicas.
- PHP es un potente lenguaje y el intérprete, tanto incluido en el servidor Web como módulo o ejecutado como un binario CGI, puede acceder a ficheros, ejecutar comandos y abrir comunicaciones de red en el servidor. Todas estas características hacen que lo que se ejecute en el servidor Web sea seguro por defecto.
- No requiere definición de tipos de variables ni manejo detallado del bajo nivel.

Teniendo en cuenta todo lo anteriormente mencionado, y conociendo que el marco de trabajo que se utilizará para el desarrollo fue elaborado con PHP 5, se selecciona para el desarrollo de la aplicación. Aprovechando de esta manera la orientación a objeto con que cuenta dicho lenguaje.

1.4.1.6. Gestor de Base de Datos.

Es importante diferenciar los términos BD (base de datos) y SGBD (Sistema de Gestión de Bases de Datos). El primero no es más que un conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que una BD puede considerarse una colección de datos variables en el tiempo.

El software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez, se denomina sistema de gestión de bases de datos (SGBD).

El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado.

1.4.1.7. PostgreSQL.

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS). El proyecto PostgreSQL sigue actualmente un activo proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuidores de código abierto. Es ampliamente considerado como una de las alternativas de sistema de bases de datos de código abierto. Entre sus características destacan que, permite el paso entre dos estados consistentes manteniendo la integridad de los datos, tiene múltiples tipos de datos predefinidos, soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario. Además, incorpora una estructura de datos Array y presenta conectividad TCP/IP, JDBC y ODBC. Cuenta también con un amplio conjunto de enlaces con diversos lenguajes de programación, tales como C, C++, Java, Delphi, Python, Perl, PHP, Bash.

Ventajas: (13)

- Instalación ilimitada, puesto que no hay costo asociado a la licencia del software.
- Ahorros considerables en costos de operación, ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características, estabilidad y rendimiento.
- Estabilidad y confiabilidad legendaria.
- Extensible, el código fuente esta disponible para todos sin costo.
- Multiplataforma, esta disponible en casi cualquier Unix.
- Diseñado para ambientes de alto volumen, usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mejor respuesta.
- Existen varias herramientas gráficas de alta calidad para administrar las bases de datos (pgAdmin, pgAccess) y para hacer diseño de bases de datos (Tora, Data Architect).

- Implementa un subconjunto extendido de los estándares SQL92 y SQL99.
- Ofrece varios modos de bloqueo para controlar el acceso concurrente a los datos en tablas.

Se selecciona el gestor PostgreSQL, por todas las características y ventajas que cuenta, destacando entre ellas que es código abierto. Además, el marco de trabajo propuesto lo utiliza.

1.5. Modelo Cliente-Servidor.

Desde el punto de vista funcional, se puede definir el modelo Cliente/Servidor como una arquitectura distribuida que permite a los usuarios finales obtener acceso a la información en forma transparente aún en entornos multiplataforma. Las comunicaciones entre el cliente y el servidor son generalmente bidireccionales, el cliente envía un mensaje solicitando un determinado servicio (hace una petición) a un servidor y este envía uno o varios mensajes con la respuesta (provee el servicio). (14)

La arquitectura Cliente-Servidor es una extensión de programación modular en la que la base fundamental es separar una gran pieza de software en módulos con el fin de proveer usabilidad, flexibilidad, interoperabilidad y escalabilidad en las comunicaciones, hacer más fácil el desarrollo y mejorar su mantenimiento, cuenta con tres elementos fundamentales; cliente, servidor y red de comunicación.

Respondiendo a este modelo se lleva a cabo la utilización de las diferentes aplicaciones o servicios de Internet, que es una forma de especializar terminales y programas para que las actividades y tareas se ejecuten con la mayor eficiencia posible. Mediante esta arquitectura el usuario puede acceder a la información sin tener en cuenta su ubicación física y donde pueda estar alojada la misma.

1.6. Servidor web Apache.

Apache es un servidor de páginas web de código abierto multiplataforma y modular. Es considerado el servidor HTTP de código abierto más utilizado del mundo. Presenta entre otras características, mensajes de errores altamente configurables, bases de datos de autentificación y negociado de contenido. Flexible, rápido y eficiente. Continuamente actualizado y adaptado a los nuevos protocolos (HTTP 1.1). Puede ser adaptado a diferentes entornos y necesidades.

Características principales: (15)

- Trabaja sobre múltiples plataformas (Unix, Linux, MacOSX, Vms, Win32, OS2).
- Incluye módulos que se cargan de forma dinámica.
- Soporta CGI, Perl, PHP.
- Soporte para Bases de datos.
- Soporte SSL para transacciones seguras.
- Incluye soporte para hosts virtuales.
- Soporta HTTP 1.1.
- Código Abierto.
- Rápido y eficiente.

Teniendo en cuenta todas las características y ventajas que puede brindar al ser un servidor de código abierto se selecciona para publicar la aplicación.

1.7. Fundamentación de la Metodología.

Para desarrollar un software es necesario seleccionar la metodología adecuada, teniendo en cuenta parámetros como los requerimientos, calidad, límite de tiempo, entre otros. Actualmente existen muchas tendencias de metodologías que brindan diferentes marcos que los desarrolladores pueden emplear para realizar su trabajo.

1.7.1. Proceso de Desarrollo de Software del centro UCID.

La producción eficaz y eficiente de un producto software que reúna y satisfaga los requisitos del cliente con una planificación y estimación de recursos predecibles es el propósito fundamental de este proceso de desarrollo. Propone un modelo de desarrollo de software que describe la secuencia de actividades de alto nivel para la construcción y desarrollo de soluciones. La combinación entre los modelos basado en Componentes, el Iterativo y el Incremental es la forma de lograr este modelo de desarrollo, el cual define una secuencia de actividades a llevar a cabo para tener una solución más efectiva. El ciclo de vida del proceso de desarrollo cuenta con cinco fases: Inicio, Modelación,

Construcción, Explotación Experimental y Despliegue, las cuales permiten mejorar el planeamiento, ejecución y control del proyecto. (16)

Por todas las características antes mencionadas se selecciona el Proceso de Desarrollo de Software del centro UCID.

1.8. Lenguaje de Modelado (UML).

UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra cierta cantidad de software, permite la modelación de sistemas con tecnología orientada a objetos y recomienda la utilización de diagramas para representar las distintas vistas de un sistema.

Se ha convertido en la actualidad en uno de los más utilizados por lo expresivo, claro y uniforme que resulta para el diseño orientado a objeto y además por permitir una fuerte integración entre las herramientas, los procesos y los dominios.

1.9. Herramienta CASE.

En las últimas décadas se ha trabajado en el área de desarrollo de sistemas para encontrar técnicas que permitan incrementar la productividad y el control de calidad en cualquier proceso de elaboración de software.

CASE es una sigla que corresponde a las iniciales de: Computer Aided Software Engineering; y en su traducción al Español significa Ingeniería de Software Asistida por Computación.

Las Herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar el diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación y detección de errores. (17)

Algunos de los componentes de las herramientas CASE permiten: (17)

- Confeccionar la definición de requerimientos de los usuarios.

- Mejorar el diseño de los sistemas.
- Mejorar la eficiencia en la programación (por su generación automática de códigos).
- Otorgar a la administración un mejor soporte en la documentación.

1.9.1. Visual Paradigm for UML.

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. (18)

Principales características:

- Entorno de creación de diagramas para UML.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad de integrarse en los principales IDEs.
- Disponibilidad en múltiples plataformas.

1.10. Conclusiones.

Durante el desarrollo de este capítulo se ha hecho un estudio de la situación actual de los sistemas de Reportes y Facturación basados en la PBX Asterisk a nivel nacional e internacional. Además, se analizaron las herramientas, lenguajes y metodologías que pueden dar solución al sistema. Seleccionándose como metodología el Proceso de Desarrollo de Software del centro UCID, tomándose la decisión de desarrollar la aplicación web utilizando PHP 5 como lenguaje de programación, Sauxe como framework y aprovechando las potencialidades que brinda PostgreSQL como gestor de base de datos. Una correcta utilización de la tecnología y las herramientas seleccionadas brindará la posibilidad de lograr un sistema de máxima calidad que cumpla con los requisitos propuestos y satisfaga los intereses del cliente.

Capítulo 2: Características del sistema.

2. Introducción.

En este capítulo se realizará un análisis de las características del sistema a desarrollar, se hará énfasis en el objeto de automatización, los requerimientos no funcionales y los funcionales, así como las interfaces de usuario que integraran el mismo.

2.1. Problema y situación problemática.

Llevar a cabo la administración de Asterisk es un proceso complicado, pues se realiza escribiendo comandos en la consola incluida (CLI). Al efectuar una llamada a través de Asterisk PBX se genera los informes de contabilidad de llamadas, también conocidos como CDR (Call Detail Record). Los registros son almacenados en un archivo de texto separado por comas, también conocido como CSV (comma separated value), en el directorio /var/log/asterisk/cdr-csv. Este fichero brinda toda la información necesaria para confeccionar los reportes, así como los datos para facturar las llamadas. La factura se lleva a cabo luego de conocer el destino de la llamada, el tiempo de duración y el lugar de origen. Pero para realizar la configuración del fichero mencionado es necesario hacerlo de forma manual, razón por la cual se requiere acceso físico al ordenador donde esté hospedada la PBX y generalmente se encuentra en servidores dedicados, sin mouse ni teclado, con el hardware y software necesario para un rendimiento óptimo; por lo que se hace necesario el desarrollo de un sistema que permita administrar los procesos y la configuración de la plataforma telefónica PlaTel de forma personalizada.

2.2. Objeto de automatización.

El proceso de configuración del sistema de reporte y facturación en el sistema de administración de la Plataforma telefónica PlaTel, así como la creación de reportes dependiendo de diferentes criterios y además la facturación de todas las llamadas efectuadas en la plataforma.

2.3. Información que se maneja.

La información que se maneja es la brindada por el servidor Asterisk, el mismo genera los informes de contabilidad de llamadas, también conocidos como CDR (Call Detail Records o registro detallado de llamadas), para cada llamada que se realiza. Se ha empleado el libro Asterisk PBX, Guía de la configuración para obtener los principales campos CDR. (Ver Anexo No.1).

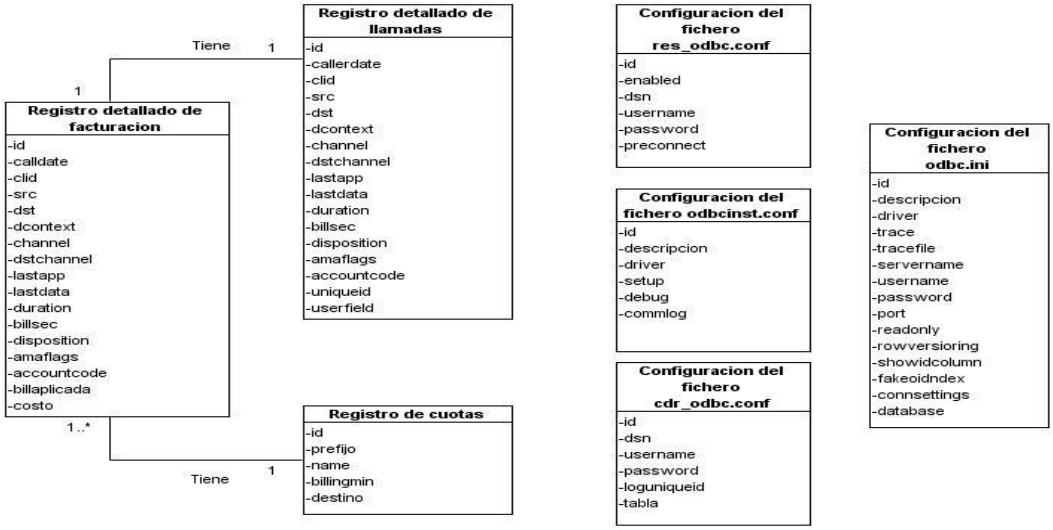
2.4. Propuesta de sistema.

El sistema tendrá como principal objetivo brindarle al usuario toda la información que necesite con respecto a cada llamada efectuada en la plataforma, así como la opción de facturar las mismas. Dentro de los reportes que se obtendrán se encuentra graficar el comportamiento de las llamadas: el cual mostrará gráficas estadísticas de la evolución de las llamadas, teniendo en cuenta la cantidad de llamadas por meses y los horarios de las mismas. Otros de los reportes son listar llamadas realizadas: estos mostrarán una completa información de todas las llamadas realizadas, además de permitir realizar una búsqueda avanzada, la cual mostrará una completa información de todas las llamadas realizadas dependiendo de varios criterios de búsqueda seleccionados por el usuario, datos de última llamada: estos mostrarán una completa información de la última llamada realizada, y datos último llamante: estos mostrarán una completa información del último llamante.

El sistema también se encargará de facilitar la configuración de la facturación dentro del sistema de administración PlaTel, permitiendo gestionar cuotas: este se encarga de la creación, modificación y eliminación de cuotas y estado de cuotas: el cual se encarga de mostrar el estado de las cuotas establecidas.

Al comparar la propuesta de sistema con algunos ya existentes, como es el caso de FreePBX y Elastix, que son de los más reconocidos y usados a nivel mundial, se llega a la conclusión que incluye aspectos que lo diferencia de estos, logrando una superioridad. Debido a que incluye gráficas que ilustran el comportamiento de las llamadas e incluye la facturación de las mismas. Será capaz de generar reportes comparativos y evolutivos. Asimismo estará concebido para usuarios con conocimientos básicos de informática y telecomunicaciones. Adquiriendo de esta forma una sólida reportería y facturación en conjunto.

2.5. Modelo Conceptual.



2.6. Especificación de requisitos de software.

2.6.1. Requerimientos Funcionales del sistema.

- RF1 Mostrar reportes detallados de llamadas realizadas.
    - RF1.1 Listar llamadas realizadas.
    - RF1.2 Mostrar datos de la última llamada realizada.
- RF3. Mostrar datos de último llamante.
- RF4. Gestionar cuotas de llamadas.
    - RF4.1. Adicionar cuota.
    - RF4.2. Modificar cuota.
    - RF5.3. Eliminar cuota.
- RF5 Mostrar estado de cuota.
- RF6 Graficar el comportamiento de las llamadas.

- RF7 Configurar CDR.

2.6.2. Requerimientos no Funcionales del sistema.

2.6.2.1. Interfaz.

La aplicación propuesta podrá ser utilizada por usuarios con conocimientos básicos en temas de informáticas y telecomunicaciones, pues estará bien detallada y organizada la información que se mostrará en la interfaz. Será amigable y fácil de usar, de forma tal que no se convierta en un problema para los usuarios el trabajo con la misma.

2.6.2.2. Usabilidad.

A los usuarios del sistema se les dará un adiestramiento básico en el uso de la aplicación en cada uno de sus niveles, mientras que los administradores deberán obtener un conocimiento avanzado para ser capaces de dar respuesta a cualquier incidente que ocurra con el sistema.

2.6.2.3. Rendimiento.

Luego de haber realizado un estudio del hardware y la red que se va a utilizar, es preciso configurar la aplicación para lograr un rendimiento óptimo de la misma. La eficiencia del producto estará condicionada en gran medida por el aprovechamiento de los recursos con que se dispongan. Teniendo una fuerte influencia la rapidez de las consultas a la base de datos y la velocidad del servidor Asterisk.

2.6.2.4. Portabilidad.

La herramienta es compatible con los navegadores web y los sistemas operativos más utilizados en todo el mundo, desde cualquier versión de Windows NT en adelante hasta cualquier distribución de Linux. Tiene la capacidad de integración con los SGBD más usados y brinda la posibilidad de alojarse en cualquier servidor web, aunque se recomienda PostgreSQL y Apache respectivamente.

2.6.2.5. Seguridad.

La información manejada por el sistema debe estar protegida de acceso no autorizado y divulgación, a partir de los diferentes roles de los usuarios que empleen el sistema. Es por esa razón que el usuario debe autenticarse antes de entrar al sistema. Además, se presentarán las interfaces para

cada usuario dependiendo del nivel de acceso a la información. La información manejada por el sistema será objeto de cuidadosa protección contra corrupción y estados inconsistentes, de igual manera el origen y autoridad de los datos. Asimismo la aplicación deberá estar disponible en todo momento para aquellas personas con acceso a la información, y los mecanismos utilizados para lograr la seguridad, no deben ser un obstáculo.

2.6.2.6. Hardware.

En el cliente se requiere una máquina con 256 MB de RAM como mínimo, un microprocesador Pentium 4 a más de 1.6 GHz y un disco duro de más de 10 GB. El servidor web junto con el servidor de base de datos requieren como mínimo 512 MB de RAM y 20 GB de disco duro. Todos los nodos involucrados en la funcionalidad de la aplicación deben estar conectados a una red que requiere como mínimo 100 Mbps de velocidad.

2.6.2.7. Software.

Para la interacción con la herramienta el usuario sólo debe tener instalado un navegador web y cualquier sistema operativo. Del lado del servidor se requiere un servidor web (Apache), una PBX Asterisk (1.4 o superior) y un servidor de base de datos (PostgreSQL).

2.6.2.8. Políticos Culturales.

Debido a que algunos de los parámetros que se guardan en el servidor Asterisk no se le encuentra una traducción aceptable al español, se ha determinado para el desarrollo de la herramienta utilizar los idiomas inglés y español.

2.6.3. Descripción de Requisitos Funcionales.

A continuación se muestra la especificación del requisito Listar llamadas realizadas, para ver las especificaciones de los demás requisitos remitirse al expediente de proyecto.

2.6.3.1. Especificación del requisito Listar llamadas realizadas

Conceptos tratados	Conceptos	Atributos
	Registro detallado de llamadas.	id, callerdate, clid, src, dst, dcontext,cannel, dstchannel, lastapp, lastdata, duration, billsec,

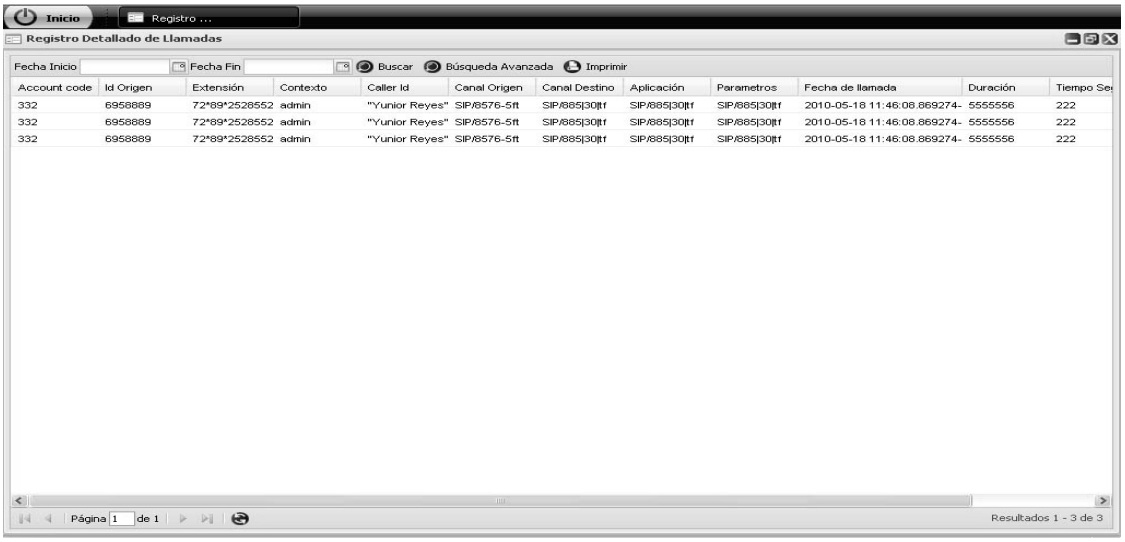
		disposition, amaflags, accountcode, uniqueid, userfield.
Precondiciones	**Precondiciones**	**Pre-requisito**
	Se ha autenticado el usuario.	Autenticar usuario.
Descripción	1. Se selecciona el menú Inicio. 2. Se selecciona la opción Platel. 3. Se selecciona la opción CDR. 4. Se selecciona la opción Registro Detallado de Llamadas. 5. Se muestra la interfaz Registro Detallado de Llamadas y en un grid todas las llamadas realizadas. 6. Se selecciona Fecha Inicio. 7. Se selecciona Fecha Fin. 8. Se oprime el botón Buscar. 6.1. En caso de error al seleccionar correctamente las fechas, se le informa al usuario y se le permite corregirlas. 9. Se muestra en el grid de la interfaz Registro Detallado de Llamadas una lista de todas las llamadas que se realizaron en el rango de fecha seleccionado. 10. En caso de que el usuario oprima el botón Búsqueda avanzada se muestra la interfaz Búsqueda avanzada. 11. Se llenan los campos por los cuales se desea realizar la búsqueda. 11.1 En caso de error al introducir los datos se le informa al usuario y se le permite corregirlos. 12. Se oprime el botón Buscar. 13. Se cierra la interfaz Búsqueda avanzada, y se muestra en el grid de la interfaz Registro Detallado de Llamadas una lista de las llamadas que cumplen con los campos por los cuales se realizó la búsqueda	

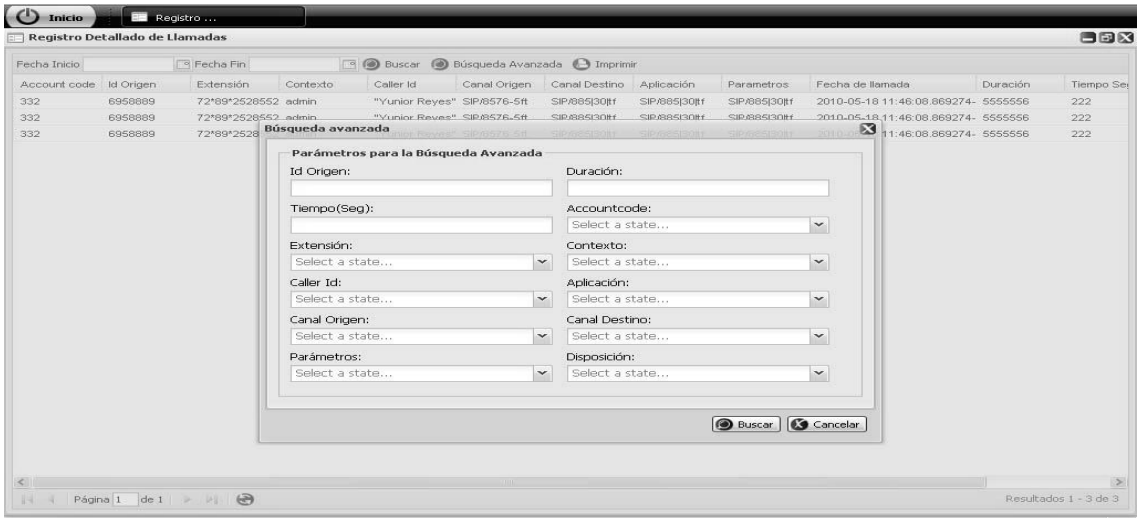
	avanzada. 13.1 En caso de que se oprima el botón Imprimir, se genera un documento en formato PDF con los datos que se encuentran el grid de la interfaz.
Validaciones	• La Fecha inicio debe ser menor a la Fecha fin. • El sistema valida los datos según lo descrito en el Modelo Conceptual: Modelo conceptual.doc.
Post-condiciones	Se ha mostrado una lista de las llamadas que están en el rango de fecha seleccionado o que cumplen con los parámetros especificado por el usuario en caso de utilizar la opción “Búsqueda avanzada”.
Post-requisito	No procede.

2.6.4. Prototipos de Interfaz de usuarios.

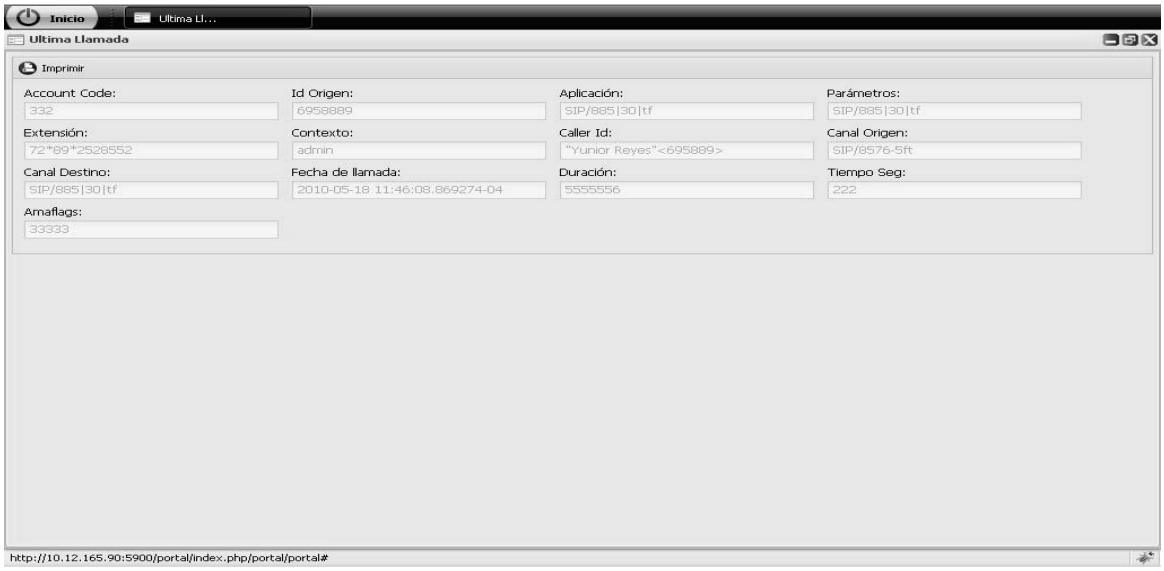
2.6.4.1. Mostrar reportes detallados de llamadas realizadas.

2.6.4.1.1. Listar llamadas realizadas.

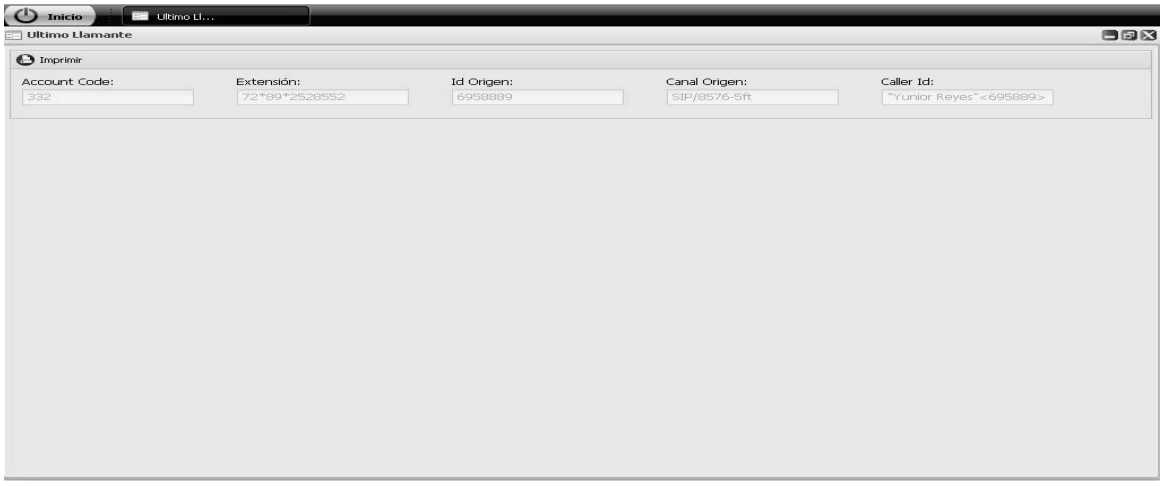




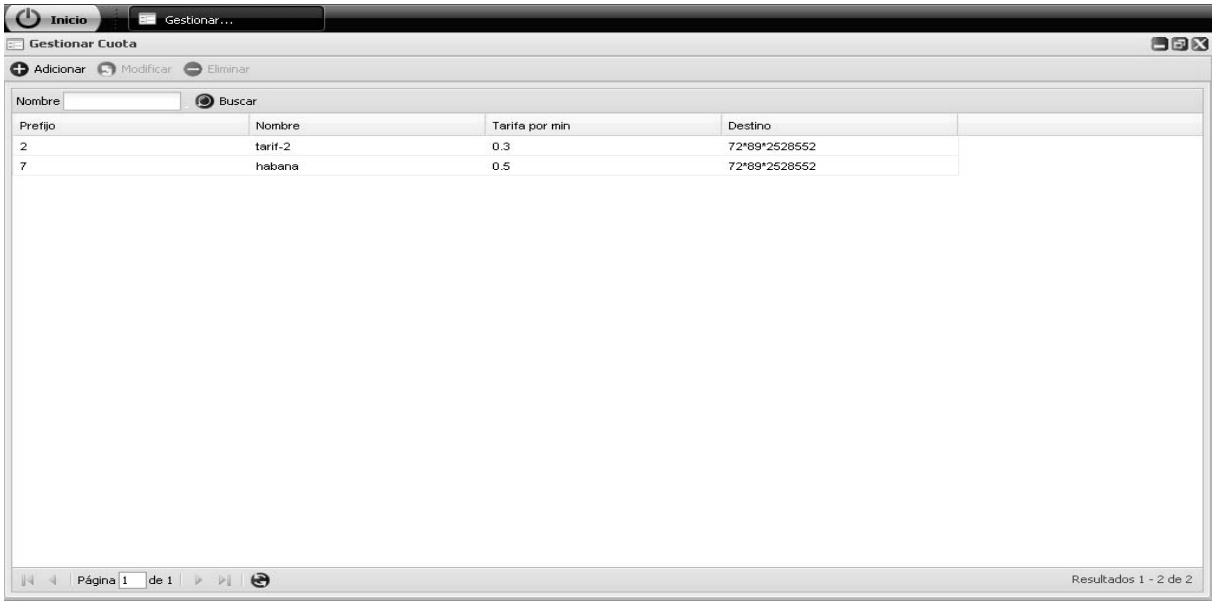
2.6.4.1.2. Mostrar datos de la última llamada realizada.



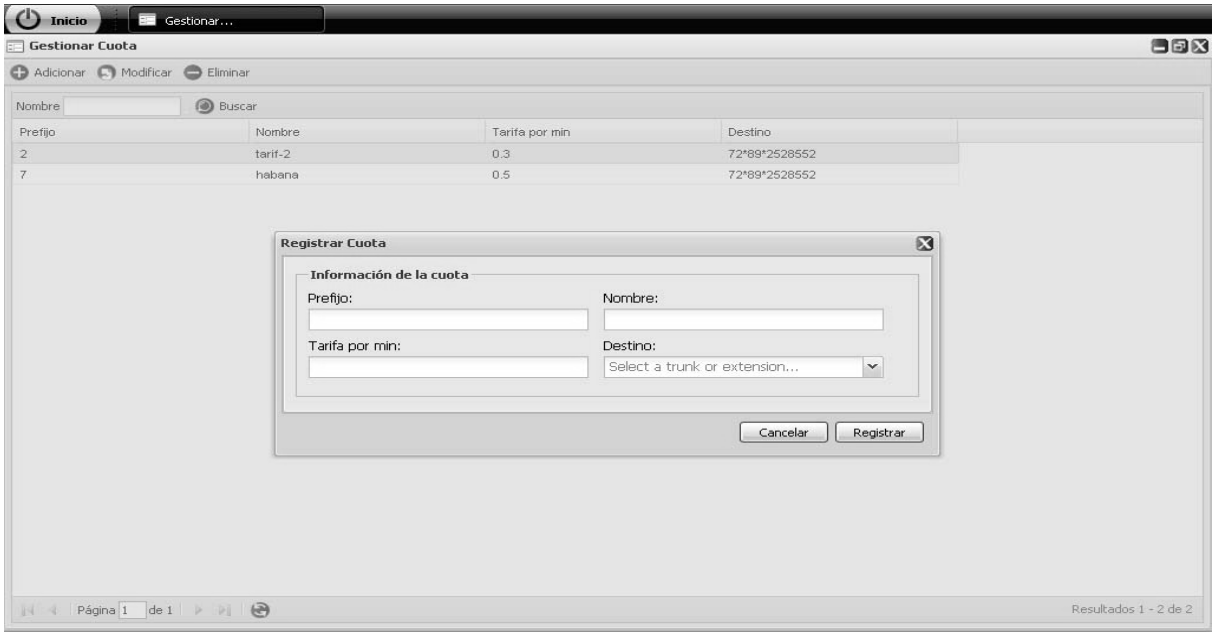
2.6.4.2. Mostrar datos de último llamante.



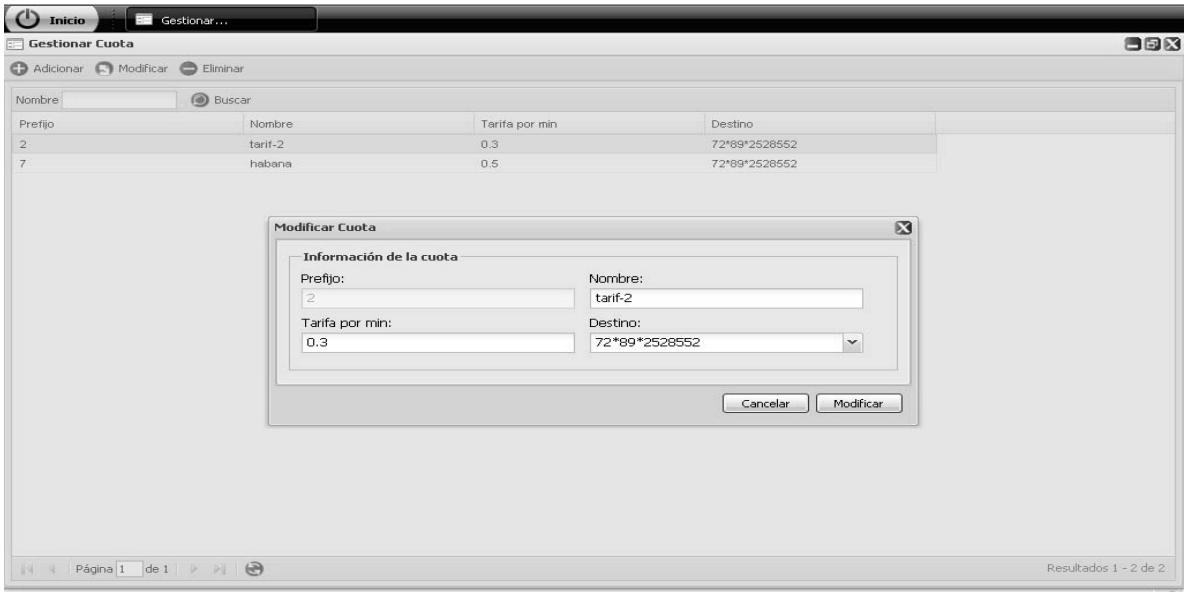
2.6.4.3. Gestionar cuotas de llamadas.



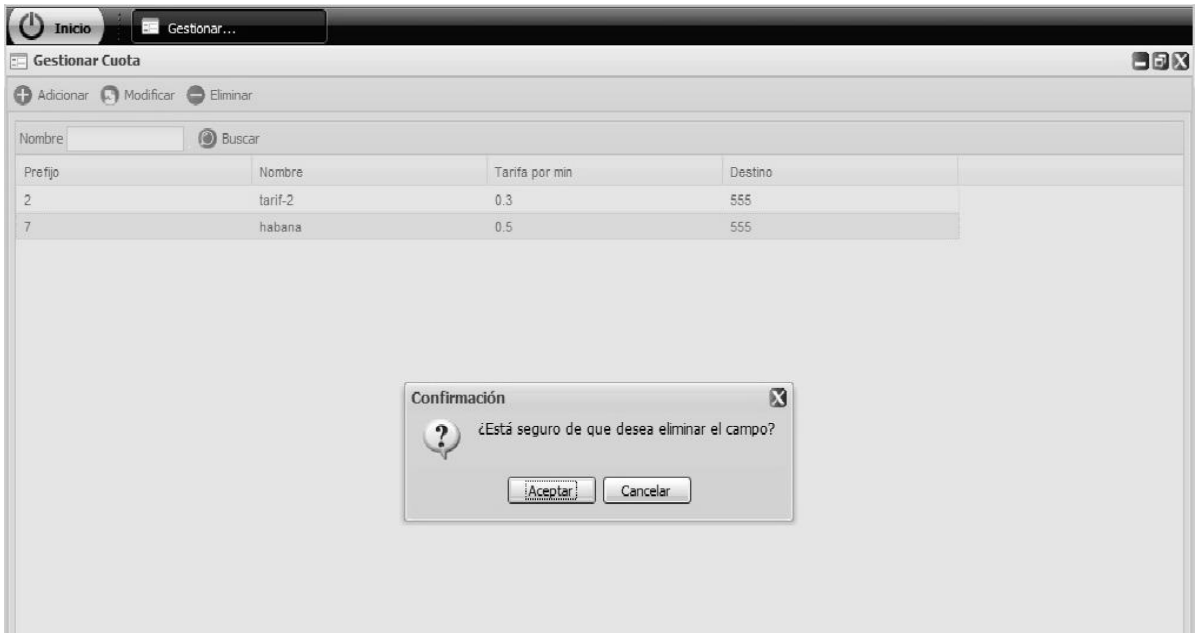
2.6.4.3.1. Adicionar cuota.



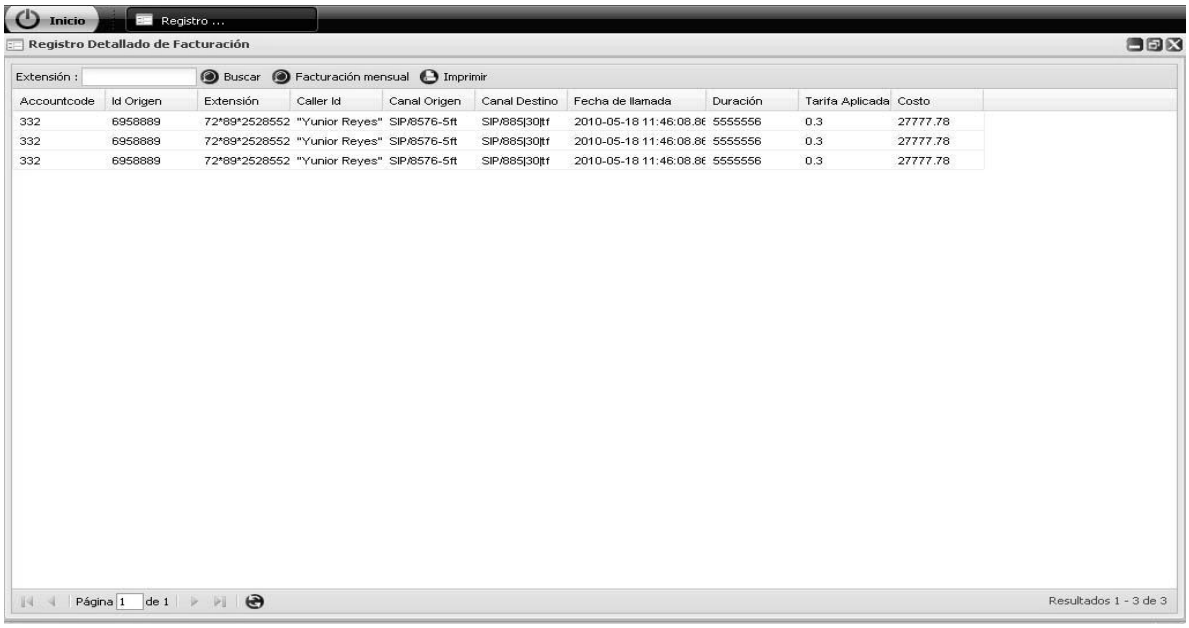
2.6.4.3.2. Modificar cuota.



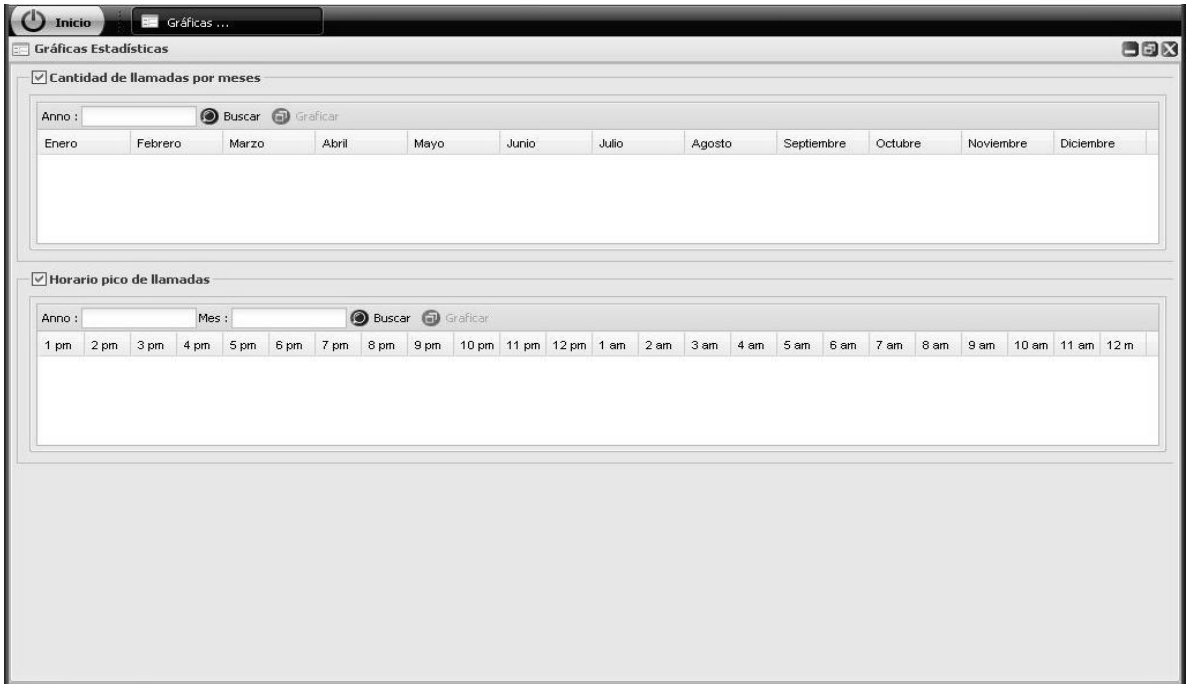
2.6.4.3.3. Eliminar cuota.



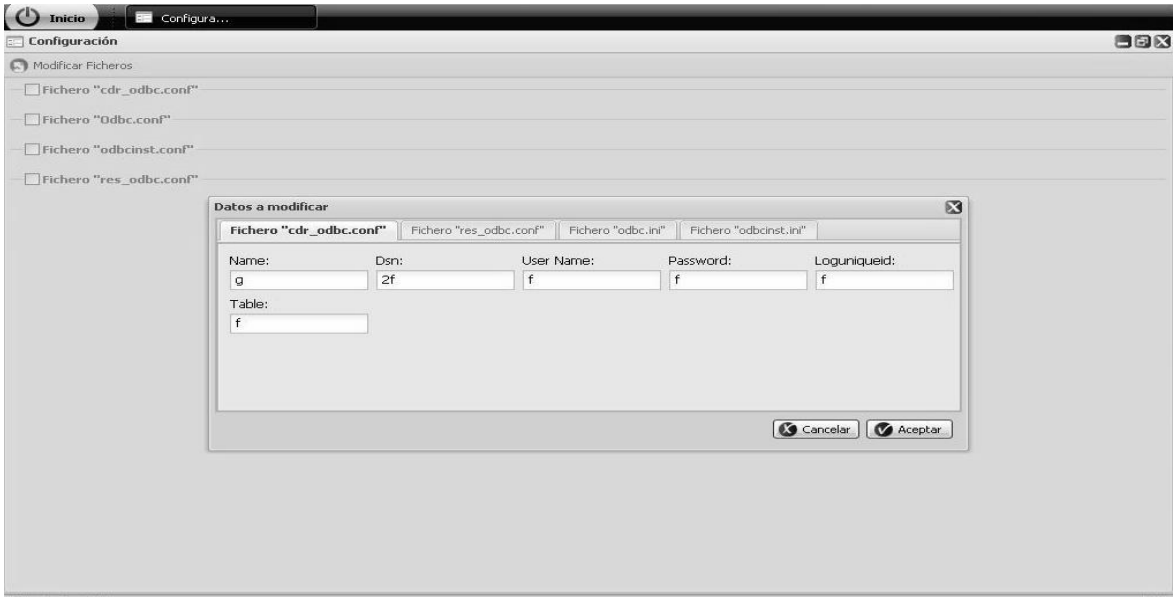
2.6.4.4. Mostar estado de cuota.



2.6.4.5. Graficar el comportamientos de las llamadas.



2.6.4.6. Configurar CDR.



2.7. Conclusiones.

En este capítulo se realizó una descripción de las características del sistema a desarrollar. Se dieron a conocer las interfaces. Se definieron los requerimientos no funcionales y los requerimientos funcionales, sirviendo de base para el comienzo del desarrollo del sistema. Todo en su conjunto dará como resultado un sistema amigable, adaptable y fácil de usar.

Capítulo 3: Diseño del sistema.

3. Introducción.

En este capitulo se realizará el diseño del sistema, definiendo las diferentes clases que intervendrán en el mismo y el flujo entre ellas en cada uno de los escenarios de los requisitos así como los patrones de diseño a utilizar. Además, se va a realizar un análisis de la seguridad del sistema.

3.1. Mecanismo de diseño.

Al igual que los desarrolladores desean siempre reutilizar código, los diseñadores de hoy en día tienen la preocupación de cómo reutilizar modelos ya elaborados. El término de Mecanismos de diseño fue introducido en Documentación del Proceso Unificado, 2003. Ha sido utilizado con el objetivo de simplificar los diagramas de clases.

Los mecanismos de diseño aportan algunos beneficios, dentro de los que destacan de manera significativa que:

- Mantienen la homogeneidad en el diseño.
- Permiten reutilizar soluciones anteriormente probadas.
- Permiten reutilizar documentación.

Cada diseñador establece sus propios mecanismos de diseño y es el total responsable de sus modelos y su forma, siempre y cuando no viole los patrones y estilos seleccionados.

El centro UCID no se encuentra exento de esto y cuenta con un mecanismo de diseño, en el cual se evidencia de forma genérica el funcionamiento del marco de trabajo utilizado para el desarrollo de las aplicaciones web. El mismo se ha utilizado para la realización del diseño del sistema propuesto.

3.2. Diagramas de clases y de secuencia.

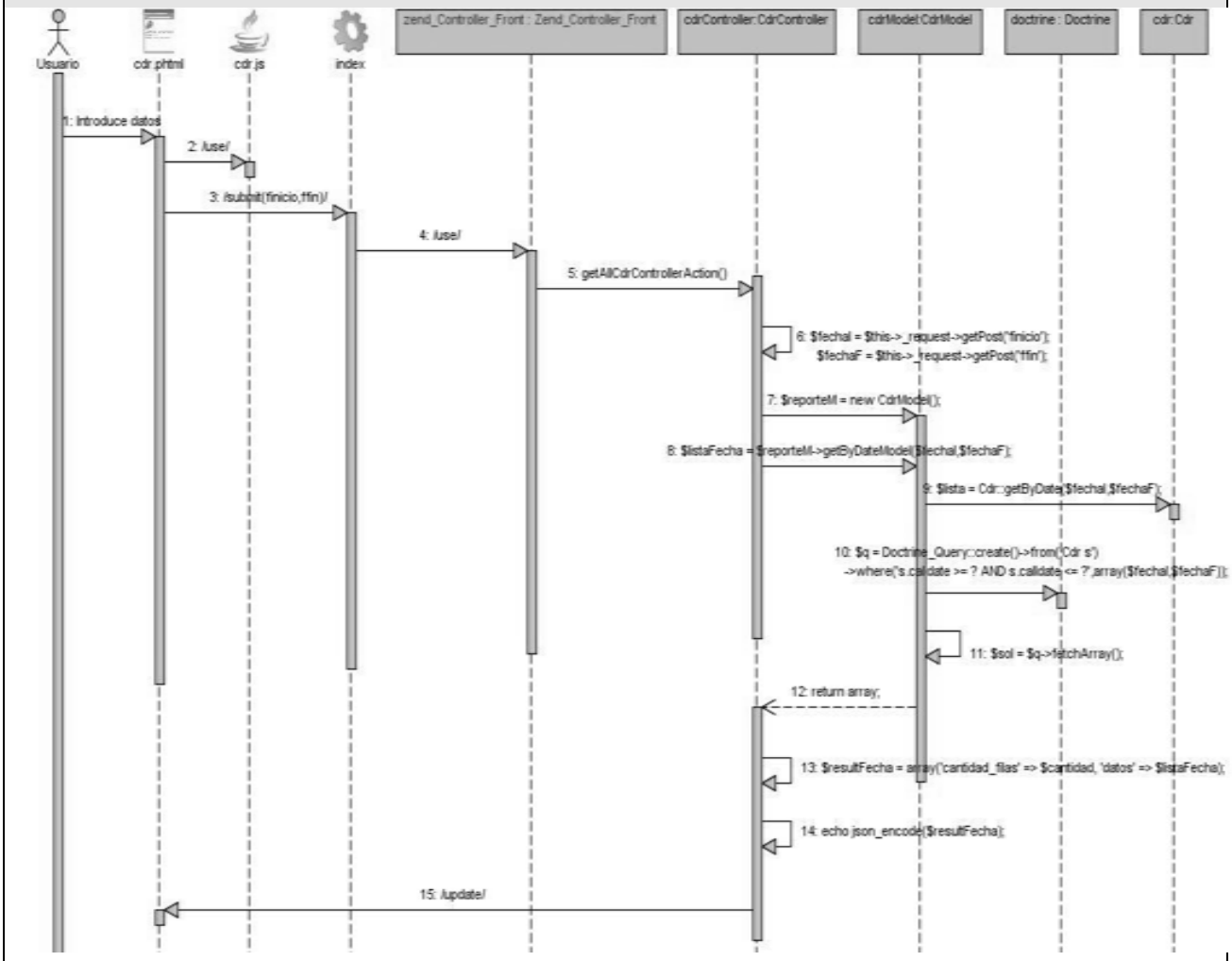
Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Si bien los diagramas de clases muestran las relaciones estáticas entre las diferentes clases, los diagramas de secuencia muestran la parte dinámica de las relaciones entre las instancias de dichas clases. A continuación se muestra el diagrama de secuencia y de clases del requisito listar llamadas realizadas.

Remitirse al expediente de proyecto para ver los restantes.

3.3. Diagramas de interacción.

3.3.1. Diagrama de secuencia Listar llamadas realizadas.

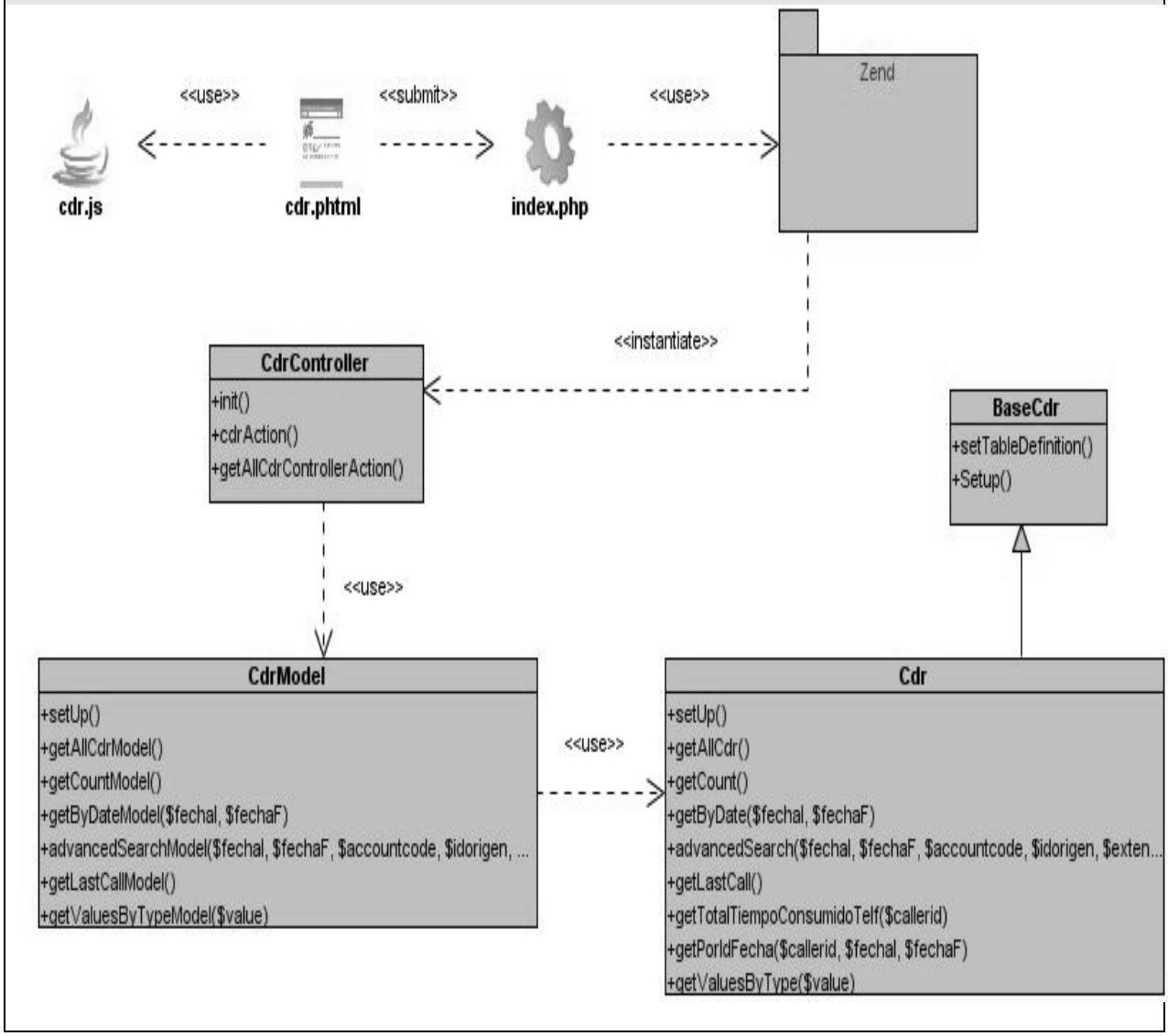
Diagrama de secuencia Listar llamadas realizadas



3.4. Diagramas de clases.

3.4.1. Diagrama de clases Listar llamadas realizadas.

Diagrama de clases Listar llamadas realizadas



3.5. Descripción de las clases

3.5.1. Clases Controladoras

Nombre: CdrController	
Tipo de clase (controladora)	
Para cada responsabilidad:	
Nombre:	init()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	cdrAction()
Descripción:	Permite cargar la página.
Nombre:	getAllCdrControllerAction()
Descripción:	Permite obtener todas las llamadas.

Nombre: CdrultimallamadaController	
Tipo de clase (controladora)	
Para cada responsabilidad:	
Nombre:	Init()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	cdrultimallamadaAction()
Descripción:	Permite cargar la página.
Nombre:	getLastCallControllerAction()

Descripción:	Permite obtener la última llamada realizada.

Nombre: CdrultimollamanteController	
Tipo de clase (controladora)	
Para cada responsabilidad:	
Nombre:	init()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	cdrultimollamanteAction()
Descripción:	Permite cargar la página.
Nombre:	getLastCallerControllerAction()
Descripción:	Permite obtener los datos del último llamante.

Nombre: CdrsetupController	
Tipo de clase (controladora)	
Para cada responsabilidad:	
Nombre:	init()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	cdrsetupAction()
Descripción:	Permite cargar la página.
Nombre:	getDataControllerAction()

Descripción:	Permite obtener los datos del fichero de configuración
Nombre:	getDataResOdbcControllerAction()
Descripción:	Permite obtener los datos del fichero de configuración
Nombre:	getDataOdbcControllerAction()
Descripción:	Permite obtener los datos del fichero de configuración
Nombre:	getDataOdbcInstControllerAction()
Descripción:	Permite obtener los datos del fichero de configuración
Nombre:	updateSetupControllerAction()
Descripción:	Permite modificar la configuración de los ficheros de configuración
Nombre:	catchDatosSetupAction()
Descripción:	Permite obtener los datos de los ficheros de configuración de la interfaz

Nombre: BillingcdrController	
Tipo de clase (controladora)	
Para cada responsabilidad:	
Nombre:	init()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	billingcdrAction()
Descripción:	Permite cargar la página.

Nombre:	getAllFacturationControllerAction()
Descripción:	Permite obtener una lista de todas las facturas
Nombre:	makeFacturacionByMothControllerAction()
Descripción:	Permite obtener una lista de todas las facturas realizadas en el rango de fecha seleccionado.

Nombre: BillinggestionarcuotaController	
Tipo de clase (controladora)	
Para cada responsabilidad:	
Nombre:	init()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	billinggestionarcuotaAction()
Descripción:	Permite cargar la página.
Nombre:	getAllTrunkControllerAction()
Descripción:	Permite obtener una lista de todos los troncales
Nombre:	getAllRateControllerAction()
Descripción:	Permite obtener una lista de todas las cuotas
Nombre:	addRateControllerAction()
Descripción:	Permite adicionar una nueva cuota
Nombre:	updateRateControllerAction()

Descripción:	Permite modificar una cuota
Nombre:	deleteRateControllerAction()
Descripción:	Permite eliminar una cuota
Nombre:	catchDataControllerCuotaAction()
Descripción:	Permite obtener los datos de la cuota de la interfaz

3.5.2. Clases Model

Nombre: CdrModel	
Tipo de clase (Modelo)	
Para cada responsabilidad:	
Nombre:	SetUp()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	getAllCdrModel()
Descripción:	Permite obtener una lista de todos los reportes.
Nombre:	getCountModel()
Descripción:	Permite obtener el total de reportes.
Nombre:	getByDateModel($fechaI,$fechaF)
Descripción:	Permite obtener una lista de los reportes en un rango de fecha.
Nombre:	advancedSearchModel($fechaI,$fechaF,$accountcode,$idorigen,$extension,$contexto,$callerid,$canalorigen,$canaldestino,$aplicacion,$parametros,$atendido,$dur

	acion,$tiemposeg,$disposiocion,$amaflags)
Descripción:	Permite obtener una lista de los reportes a partir de una búsqueda avanzada por los campos específicos.
Nombre:	getLastCallModel()
Descripción:	Permite obtener los datos de la última llamada.
Nombre:	getValuesByTypeModel($value)
Descripción:	Permite obtener una lista de los valores dependiendo del parámetro.

Nombre: CdrsetupModel	
Tipo de clase (Modelo)	
Para cada responsabilidad:	
Nombre:	setUp()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	getDataModel()
Descripción:	Permite obtener los datos de los ficheros de configuración
Nombre:	updateSetupModel($id,$dsn,$username,$password,$loguniqueid,$tabla,$descripcionodbc,$driverodbc,$traceodbc,$tracefileodbc,$servernameodbc,$usernameodbc, $passwordodbc,$portodbc,$readonlyodbc,$rowversioringodbc,$showoidcolumnodbc,$fakeoidindexodbc, $connsettingsodbc,$databaseodbc,$enabledresodbc,$dsnresodbc,$usernameresodbc,$passwordresodbc,$preconnect,$descripcionodbcinst,$driverodbcinst,$set

	upodbcinst,$debugodbcinst,$commlogodbcinst)
Descripción:	Permite modificar los ficheros de configuración

Nombre: FacturacionModel	
Tipo de clase (Modelo)	
Para cada responsabilidad:	
Nombre:	setUp()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	getAllFacturationModel()
Descripción:	Permite obtener una lista de todas las facturaciones
Nombre:	getFacturationByIdModel($dst)
Descripción:	Permite obtener una lista de las facturas realizadas por el id
Nombre:	makeFacturacionByMothModel($fechaI, $fechaF)
Descripción:	Permite obtener una lista de todas las facturas realizadas en el rango de fecha seleccionado.

Nombre: CuotaModel	
Tipo de clase (Modelo)	
Para cada responsabilidad:	
Nombre:	setUp()

Descripción:	Permite inicializar el padre de la clase.
Nombre:	getAllRateModel()
Descripción:	Permite obtener una lista de todas las cuotas
Nombre:	getAllTrunkModel()
Descripción:	Permite obtener una lista de todos los troncales
Nombre:	addRateModel($cuota)
Descripción:	Permite adicionar una cuota
Nombre:	updateRateModel($name,$billingmin,$destino,$id)
Descripción:	Permite modificar una cuota
Nombre:	deleteRateModel($prefijo)
Descripción:	Permite eliminar una cuota
Nombre:	getRateByIdModel($prefijo)
Descripción:	Permite obtener una cuota a partir del id pasado

3.5.3. Clases Entidad

Nombre: Cdr	
Tipo de clase (entidad)	
Para cada responsabilidad:	
Nombre:	SetUp()
Descripción:	Permite inicializar el padre de la clase.

Nombre:	getAllCdr()
Descripción:	Devuelve todos los reportes de la tabla
Nombre:	getCount()
Descripción:	Devuelve el total de reportes de la tabla
Nombre:	getByDate($fechaI,$fechaF)
Descripción:	Devuelve todos los reportes que se encuentren en el rango de fecha pasado
Nombre:	advancedSearch($fechaI,$fechaF,$accountcode,$idorigen,$extension,$contexto, $callerid,$canalorigen,$canaldestino,$aplicacion,$parametros,$atendido,$duracion,$tiemposeg,$disposiocion,$amaflags)
Descripción:	Realiza una búsqueda avanzada usando los parámetros pasados
Nombre:	getLastCall()
Descripción:	Devuelve los datos de la ultima llamada realizada
Nombre:	getTotalTiempoConsumidoTelf($callerid)
Descripción:	Devuelve el total de tiempo consumido por un determinado teléfono
Nombre:	getPorIdFecha($callerid,$fechaI,$fechaF)
Descripción:	Devuelve todos los reportes que tengan el mismo id y además se encuentren en el rango de fecha introducido.
Nombre:	getValuesByType($value)
Descripción:	Permite obtener una lista de todas las llamadas realizadas que tengan los valores especificados.

Nombre: CdrSetup	
Tipo de clase (entidad)	
Para cada responsabilidad:	
Nombre:	SetUp()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	getData()
Descripción:	Devuelve los datos de los ficheros de configuración
Nombre:	updateSetup($id,$dsn,$username,$password,$loguniqueid,$tabla)
Descripción:	Modifica los datos de los ficheros de configuración

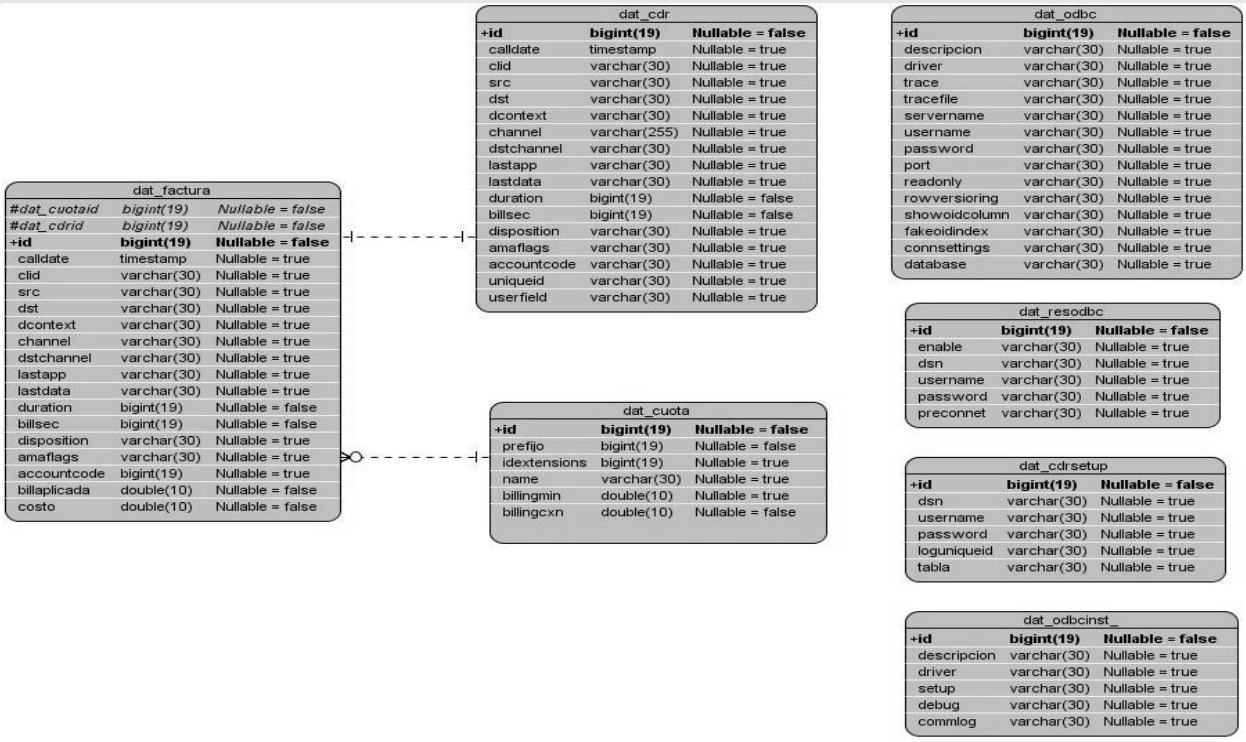
Nombre: Facturacion	
Tipo de clase (entidad)	
Para cada responsabilidad:	
Nombre:	setUp()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	getAllFacturation()
Descripción:	Devuelve todas las facturas
Nombre:	getDateLastCall()
Descripción:	Devuelve la última llamada

Nombre: Cuota	
Tipo de clase (entidad)	
Para cada responsabilidad:	
Nombre:	setUp()
Descripción:	Permite inicializar el padre de la clase.
Nombre:	getAllRate()
Descripción:	Devuelve todas las cuotas
Nombre:	getAllTrunk()
Descripción:	Devuelve todos los troncales
Nombre:	updateRate($name,$billingmin,$destino,$id)
Descripción:	Modifica una cuota
Nombre:	deleteRate($idcallerid)
Descripción:	Elimina una cuota
Nombre:	getRateById($callerid)
Descripción:	Devuelve la cuota por el id
Nombre:	dameTarifaById($callerid)
Descripción:	Devuelve la tarifa por el id

3.6. Diseño de la base de datos.

3.6.1. Diagrama Entidad Relación de la BD.

DIAGRAMA ENTIDAD RELACION



3.6.2. Descripción de las tablas.

Nombre: dat_cdr		
Descripción: Almacena los datos correspondientes a las llamadas.		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador.
calldate	timestamptz	Fecha de la llamada.
clid	varchar(80)	Texto de identificación de la llamada.

src	varchar(80)	Identificación de origen.
dst	varchar(80)	Extensión de destino.
dcontext	varchar(80)	Contexto de destino.
channel	varchar(80)	Canal usado.
dstchannel	varchar(80)	Canal de destino.
lastapp	varchar(80)	Última aplicación.
lastdata	varchar(80)	Últimos parámetros de la aplicación.
duration	integer	Tiempo total desde el discado en segundos.
billsec	integer	Tiempo total, contabilizado en segundos.
disposition	varchar(45)	ANSWERED (atendida), NO ANSWER (no atendida), BUSY (ocupada), FAILED (falló)
amaflags	integer	Indicador. DOCUMENTATION(documentar), BILLING (Contabilizar),OMMIT (omitir), IGNORE (ignorar)
accountcode	varchar(20)	Número de la cuenta usado.
uniqueid	varchar(150)	Identificador único del canal
userfield	varchar(255)	Campo definido por el usuario.

Nombre: dat_cuota		
Descripción: Almacena los datos correspondientes a las cuotas de las llamadas.		
Atributo	**Tipo**	**Descripción**

id	bigint	Identificador.
prefijo	bigint	Prefijo de la cuota.
name	varchar(30)	Nombre de la cuota.
billingmin	double precision	Tarifa por minuto
destino	varchar(30)	Extensión o Troncal a la que se le establece la cuota.

Nombre: dat_factura		
Descripción: Almacena los datos correspondientes a las facturas		
Atributo	**Tipo**	**Descripción**
id	bigint	Identificador de la factura.
calldate	timestamptz	Fecha de la llamada.
clid	varchar(30)	Texto de identificación de la llamada.
src	varchar(30)	Identificación de origen.
dst	varchar(30)	Extensión de destino.
dcontext	varchar(30)	Contexto de destino.
channel	varchar(30)	Canal usado.
dstchannel	varchar(30)	Canal de destino.
lastapp	varchar(30)	Última aplicación.
lastdata	varchar(30)	Últimos parámetros de la aplicación.

duration	bigint	Tiempo total desde el discado en segundos.
billsec	bigint	Tiempo total, contabilizado en segundos.
disposition	varchar(30)	ANSWERED (atendida), NO ANSWER (no atendida), BUSY (ocupada), FAILED (falló)
amaflags	varchar(30)	Indicador. DOCUMENTATION(documentar), BILLING (Contabilizar),OMMIT (omitir), IGNORE (ignorar)
accountcode	varchar(30)	Número de la cuenta usado.
billaplicada	double precision	Tarifa aplicada a la llamada.
costo	double precision	Costo total de la llamada.

Nombre: dat_odbc		
Descripción: Almacena los datos correspondientes al fichero odbc		
Atributo	**Tipo**	**Descripción**
id	bigint	Identificador
descripcion	varchar(30)	Breve descripción
driver	varchar(30)	Manejador.
trace	varchar(30)	Si va a trabajar con un archivo (Si ó No).
tracefile	varchar(30)	Localización física del archivo.
servername	varchar(30)	Nombre del servidor de base datos al que se va a conectar.
username	varchar(30)	Nombre de usuario.

password	varchar(30)	Contraseña.
port	varchar(30)	Puerto.
readonly	varchar(30)	Solo lectura.
rowversioring	varchar(30)	Filas.
showoidcolumn	varchar(30)	Mostrar el identificador de columna.
fakeoidindex	varchar(30)	Falso identificador del índice.
connsettings	varchar(30)	Ajustes de la conexión.
database	varchar(30)	Nombre de la base de datos.

3.7. Patrones de diseño.

A principios de los 90 los patrones de diseño obtuvieron un gran éxito en el mundo de la informática, a partir de la publicación del libro Design Patterns, en el que se recogían 23 patrones de diseño comunes, escrito por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides, conocidos como "la pandilla de los cuatro".

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. Se deben tener presente los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios).

En las diferentes capas del framework utilizado para el desarrollo de la aplicación se utilizan varios patrones, ellos son:

Singleton: Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.

Factory Method: Define una interfaz para crear un objeto, pero deja que sean las subclases quienes decidan qué clase instanciar. Permite que una clase delegue en sus subclases la creación de objetos.

Front Controller: implica que todas las solicitudes son dirigidas a un único script PHP que se encarga de instanciar al controlador frontal y redirigir las llamadas.

Decorator: Añade funcionalidad a un objeto de manera dinámica.

Fachada: Provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema. Este patrón se emplea para brindar una interfaz que abstrae completamente al usuario del proceso de comunicación con la PBX Asterisk.

Inversión de Control (IoC): permite un menor acoplamiento entre componentes de una aplicación y fomenta así la reutilización de los mismos.

Data Access Object (DAO): La idea de este patrón es ocultar la fuente de datos y la complejidad del uso de JDBC a la capa de presentación o de negocio.

Experto: Este se encarga de asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

3.8. Tratamiento de errores

Cuando se realizan los procesos de entrada, búsqueda y eliminación de datos pueden ocurrir errores que atenten contra la estabilidad y el correcto funcionamiento del sistema. Con el objetivo de mitigar los mismos se realizan acciones concretas. Del lado del cliente se validan los datos a partir de expresiones regulares. Del lado del servidor se implementa la clase Exception, para la gestión de excepciones.

3.9. Seguridad

AdminPlaTel utiliza para el desarrollo el marco de trabajo Sauxe, el mismo cuenta con un módulo de seguridad, el cual garantiza el acceso y uso de la aplicación con la creación de usuarios y la asignación de roles y permisos a los mismos; además de permitir la configuración del sistema para cada uno de estos usuarios. Conjunto a esto el sistema se centraliza, garantizando que el acceso a las clases controladoras se realice usando el controlador frontal el cual se encarga de redireccionarlos a las mismas.

3.10. Conclusiones

En este capítulo se llevó a cabo el diseño de los módulos de Reportes y Facturación para la plataforma PlaTel en el cual se definieron los flujos de los escenarios de cada requisito, las diferentes clases que intervienen, así como los patrones de diseño a emplear para el futuro desarrollo de dicho sistema. Además, se tuvieron en cuenta algunos elementos que son fundamentales a la hora de diseñar un sistema como es el caso de la seguridad, la misma es garantizada por un módulo creado especialmente para gestionar los usuarios, roles y permisos, logrando de esta manera una mayor fortaleza y fiabilidad del sistema.

Capítulo 4: Implementación y prueba.

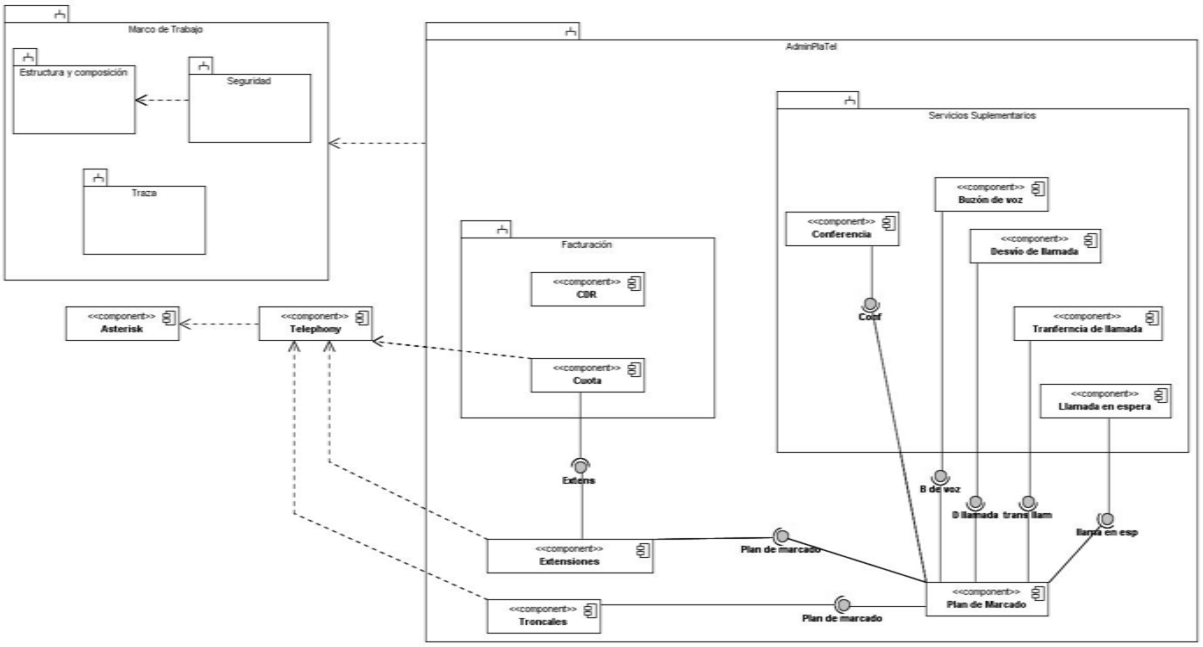
4. Introducción

El principal objetivo de este capítulo es realizar la implementación de los módulos de Reportes y Facturación para la plataforma PlaTel, teniendo en cuenta varios aspectos como es la definición de los componentes que van a integrar el sistema así como la realización de pruebas al mismo para garantizar su robustez y correcto funcionamiento.

4.1. Diagrama de componentes.

Un diagrama de componentes representa cómo un sistema software es dividido en componentes y muestra las dependencias lógicas entre estos. Los diagramas de Componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema, es decir, para describir la vista de implementación estática de un sistema.

4.1.1. Diagrama de componentes AdminPlaTel



4.2. Matriz de integración de componentes internos

Componentes Internos	Extensiones	Telephony
Cuota	Lista de extensiones	Telephony

4.3. Pruebas de software

La correcta realización de las pruebas permite detectar y corregir la mayor cantidad de errores posibles antes de la entrega al cliente del software desarrollado. Logrando de esta manera mejorar la percepción de calidad del usuario final y su satisfacción. Aunque se debe destacar que las pruebas de software no garantizan que un sistema esté libre de errores, sino que se detecten la mayor cantidad de defectos posibles para su debida corrección.

En el caso del sistema de Registro Detallado de Llamadas y Facturación de la plataforma PlaTel el método de prueba utilizado es el de **caja negra,** el cual es aquel que se realiza sobre la interfaz del software, teniendo en cuenta las entradas que recibe y las salidas o respuestas que produce, siendo completamente indiferente el funcionamiento interno de la aplicación.

Asimismo se emplea **Partición de Equivalencia** como técnica de caja negra, la cual permite examinar los valores válidos e inválidos de las entradas existentes en el software y detectar de forma rápida una clase de errores que de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. Por lo que su objetivo principal es reducir el posible conjunto de casos de prueba en uno más pequeño, un conjunto manejable que evalúe bien el software.

Los errores tienden a darse más en los límites del campo de entrada que en el centro, por ello se utiliza el **análisis de valores límites (AVL),** la cual es una técnica de diseño de caso de prueba que completa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el AVL lleva a la elección de casos de prueba en los extremos de la clase.

Los resultados obtenidos fueron satisfactorios, y permitieron corregir los errores encontrados.

4.4. Descripción diseño caso de prueba

La descripción de los diseños de casos de pruebas (DCP) se realiza para cada uno de los requisitos funcionales, con el objetivo principal de obtener una documentación por la cual guiarse a la hora de realizarle pruebas al sistema desarrollado. Este trabajo no es la excepción, por lo que cada requisito funcional cuenta con el DCP correspondiente. Sin embargo, a continuación se colocará el DCP del RF Adicionar cuota, ya que colocarlos todos sería engorroso y opacaría la calidad del trabajo.

Para ver los restantes DCP remitirse al expediente de proyecto.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
Adicionar cuota	El objetivo de este requisito es adicionar una cuota.	EP 1.1: Adicionar cuota correctamente.	1. Se selecciona el menú Inicio. 2. Se selecciona la opción Platel 3. Se selecciona la opción Facturación. 4. Se selecciona la opción Gestionar Cuota. 5. Se oprime el botón Adicionar. 6. Se muestra la interfaz Registrar Cuota. 7. Se introducen el prefijo, el nombre y la tarifa por minutos, y se selecciona el destino. 8. Se oprime el botón Registrar, se comprueban los datos, se cierra la interfaz Registrar Cuota y se actualiza el grid de la interfaz Gestionar Cuota.

		EP 1.2: Adicionar cuota incorrectamente.	1. Se selecciona el menú Inicio. 2. Se selecciona la opción Platel 3. Se selecciona la opción Facturación. 4. Se selecciona la opción Gestionar Cuota. 5. Se oprime el botón Adicionar. 6. Se muestra la interfaz Registrar Cuota. 7. Se introducen el prefijo, el nombre y la tarifa por minutos, y se selecciona el destino. 8. El usuario llena los campos con datos incorrectos o deja campos en blanco y oprime el botón Registrar. 9. El sistema pone los campos en rojo y muestra el mensaje: *Por favor verifique nuevamente que hay campos con valores incorrectos.*

		EP 1.3: Cancelar operación.	1. Se selecciona el menú Inicio. 2. Se selecciona la opción Platel 3. Se selecciona la opción Facturación. 4. Se selecciona la opción Gestionar Cuota. 5. Se oprime el botón Adicionar. 6. Se muestra la interfaz Registrar Cuota. 7. Se introducen el prefijo, el nombre y la tarifa por minutos, y se selecciona el destino. 8. Se oprime el botón Cancelar. 9. El sistema cancela las operaciones y se cierra la ventana.

Id del escenario	Escenario	Prefijo	Nombre	Tarifa por min	Destino	Respuesta del sistema	Resultado de la prueba
EP 1.1:	Adicionar cuota correctamente.	V(22)	V(vcl)	V(0.9)	V(telef)	Se adiciona la cuota al grid y se muestra un mensaje de confirmación.	
EP 1.2:	Adicionar cuota incorrectamente.	I(vacío)	I(vacío)	I(vacío)	I(vacío)	Se muestra el campo en color rojo y se muestra un mensaje indicando que los campos tienen valores incorrectos.	
		V(22)	I(vacío)	I(vacío)	I(vacío)		
		V(22)	V(vcl)	I(vacío)	I(vacío)		
		V(22)	V(vcl)	V(0.9)	I(vacío)		

EP 1.3:	Cancelar operación.	NA	NA	NA	NA	Se cancela la operación y se cierra la ventana.	

4.5. Conclusiones

En el presente capítulo se abordaron las etapas de implementación y prueba del software. Se definieron los componentes que integran el sistema y se implementó el mismo. Asimismo se realizaron los diseños de casos de prueba correspondiente. De esta manera, el sistema está listo para ser evaluado a través de pruebas de aceptación.

Conclusiones Generales:

Como resultado de la realización de este trabajo, se puede concluir que se cumplieron cada uno de los objetivos planteados, desarrollándose el análisis, diseño e implementación del módulo de Reportes y Facturación del sistema de administración AdminPlaTel. Logrando de esta manera cumplir el objetivo general satisfactoriamente, surgiendo así una nueva herramienta que permite autonomía en su aplicación como parte del proceso de soberanía tecnológica del país, lo que se avala por las siguientes consideraciones:

- El sistema presentado es el resultado del estudio y evaluación de varios sistemas de facturación y reportes existentes, de los que se tomaron sus principales ventajas.

- Se alcanzó satisfactoriamente los objetivos propuesto: Facilitar la administración de reportes y facturación en Asterisk PBX.

Recomendaciones

Se recomienda para futuras versiones del sistema desarrollado:

- Generalizar el sistema de facturación de forma que pueda ser aplicable a cualquier PBX.
- Realizar entrevistas con los clientes para agregar nuevos reportes.

Bibliografía

1. **Julio Gómez López, Francisco Gil Montoya.** *Asterisk 1.4 y FreePBX 2.3.* s.l. : Universidad de Almería, 2007.

2. **Nicolás Montero Puñales, Luis Eduardo Acosta Aparicio.** *Sistema de Reportes y Facturación de PBX Asterisk.* 2009. pág. 90.

3. **Palosanto Solutions.** *Manual del Usuario en Español (Elastix).*

4. A2Billing. [En línea] [Citado el: 5 de febrero de 2010.] http://www.asterisk2billing.org.

5. *Grails: Framework para el desarrollo de aplicaciones web (1era parte).* **López, Esteban Saavedra.** 2009.

6. Librosweb.es. [En línea] [Citado el: 5 de febrero de 2010.] http://www.librosweb.es/symfony/capitulo2/el_patron_mvc.html.

7. **García, Mayer Horna.** *Ext JS Introducción.* [Flash] 2010.

8. Nocion Digital. [En línea] [Citado el: 8 de febrero de 2010.] http://www.nociondigital.com/webmaster/html-tutorial-introduccion-a-dhtml-detalle-285.html.

9. **Jarinton, Nicolás Escobar.** Alexandria. [En línea] 2007. [Citado el: 10 de febrero de 2010.] http://www.alexandria.com.mx/tecnologias.php.

10. **Pérez, Javier Eguíluz.** *Introducción a AJAX.* 2008. pág. 251. Disponible en: http://www.librosweb.es/ajax/index.html.

11. *Doctrine ORM for PHP (Guide to Doctrine for PHP).* 2009. pág. 407.

12. **Hinostroza, Raúl Rodas.** Linux Centro. [En línea] 22 de febrero de 2007. [Citado el: 17 de febrero de 2010.] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP.

13. **Espinoza, Humberto.** *PostgreSQL Una alternativa de DBMS Open Source.* 2005. pág. 26. Disponible en : http://www.lgs.com.ve/pres/PresentacionES_PSQL.pdf.

14. Buenas Tareas. [En línea] [Citado el: 23 de febrero de 2010.] http://www.buenastareas.com/ensayos/Arquitectura-Cliente-Servidor/115409.html.

15. ABCDatos. [En línea] [Citado el: 4 de marzo de 2010.] http://www.abcdatos.com/webmasters/programa/z2820.html.

16. **Unidad de Compatibilización Integración y Desarrollo de Software para la Defensa.** *Proceso de Desarrollo y Gestion de Proyectos de Software.*

17. **Adairis Galván Rodríguez, Yanelys Olivera Osorio.** *Sistema de gestión integral de la planificación y control del servicio de comedores UCI: análisis y diseño del módulo "Administración, análisis económico y gestión de aseguramientos".* 2008.

18. Free Download Manager. [En línea] [Citado el: 18 de marzo de 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(Iglesia_Anglicana)_%5B Mac_OS_X_cuenta_14717_p/.

19. **Goncalves, Flavio E.** *Asterisk PBX Guía de la configuración.* Primera edición. 2007. pág. 362.

20. **Jim Van Meggelen, Leif Madsen, and Jared Smith.** *Asterisk: The Future of Telephony.* Segunda Edición. 2007. pág. 604.

21. **Benjamin Jackson, Champ Clark.** *Asterisk Hacking.* pág. 274. ISBN: 978-1-59749-151-8.

22. **Fernández, Alfonso.** *Guía Rápida Asterisk.* 2007.

23. **Sharif, Ben.** *Trixbox Without Tears.* 2006. pág. 209.

24. **Eduardo Viegas, Facundo Correa.** *Asterisk Desconsolado.* pág. 101, Manual. versión 2.0.

25. **Luis Felipe Martínez, Wilson Teran T.** *Manual de instalación y configuración de un servidor Asterisk.* 2007. pág. 51.

26. Ciberaula. *Una introducción a Apache.* [En línea] [Citado el: 14 de marzo de 2010.] http://linux.ciberaula.com/articulo/linux_apache_intro.

27. **Pecos, Daniel.** PostgreSQL vs MySQL. [En línea] http://www.netpecos.org/docs/mysql_postgres/x15.html.

28. ¿Qué es un Sistema Gestor de Bases de Datos o SGBD? [En línea] [Citado el: 18 de marzo de 2010.] http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.

29. **José Márquez Díaz, Leonardo Sampedro, Félix Vargas.** *Instalación y configuración de Apache,un servidor Web gratis.* pág. 14.

30. **Rosa, Fabián Sellés.** *Manual de Asterisk y otras hierbas.* s.l. : Universidad de Cádiz. pág. 61.

Referencias bibliográficas

1. **Julio Gómez López, Francisco Gil Montoya.** *Asterisk 1.4 y FreePBX 2.3.* s.l. : Universidad de Almería, 2007.

2. **Nicolás Montero Puñales, Luis Eduardo Acosta Aparicio.** *Sistema de Reportes y Facturación de PBX Asterisk.* 2009. pág. 90.

3. **Palosanto Solutions.** *Manual del Usuario en Español (Elastix).*

4. A2Billing. [En línea] [Citado el: 5 de febrero de 2010.] http://www.asterisk2billing.org.

5. *Grails: Framework para el desarrollo de aplicaciones web (1era parte).* **López, Esteban Saavedra.** 2009.

6. Librosweb.es. [En línea] [Citado el: 5 de febrero de 2010.] http://www.librosweb.es/symfony/capitulo2/el_patron_mvc.html.

7. **García, Mayer Horna.** *Ext JS Introducción.* [Flash] 2010.

8. Noción Digital. [En línea] [Citado el: 8 de febrero de 2010.] http://www.nociondigital.com/webmaster/html-tutorial-introduccion-a-dhtml-detalle-285.html.

9. **Jarinton, Nicolás Escobar.** Alexandria. [En línea] 2007. [Citado el: 10 de febrero de 2010.] http://www.alexandria.com.mx/tecnologias.php.

10. **Pérez, Javier Eguíluz.** *Introducción a AJAX.* 2008. pág. 251. Disponible en: http://www.librosweb.es/ajax/index.html.

11. *Doctrine ORM for PHP (Guide to Doctrine for PHP).* 2009. pág. 407.

12. **Hinostroza, Raúl Rodas.** Linux Centro. [En línea] 22 de febrero de 2007. [Citado el: 17 de febrero de 2010.] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP.

13. **Espinoza, Humberto.** *PostgreSQL Una alternativa de DBMS Open Source.* 2005. pág. 26. Disponible en : http://www.lgs.com.ve/pres/PresentacionES_PSQL.pdf.

14. Buenas Tareas. [En línea] [Citado el: 23 de febrero de 2010.] http://www.buenastareas.com/ensayos/Arquitectura-Cliente-Servidor/115409.html.

15. ABCDatos. [En línea] [Citado el: 4 de marzo de 2010.] http://www.abcdatos.com/webmasters/programa/z2820.html.

16. **Unidad de Compatibilización Integración y Desarrollo de Software para la Defensa.** *Proceso de Desarrollo y Gestion de Proyectos de Software.*

17. **Adairis Galván Rodríguez, Yanelys Olivera Osorio.** *Sistema de gestión integral de la planificación y control del servicio de comedores UCI: análisis y diseño del módulo "Administración, análisis económico y gestión de aseguramientos".* 2008.

18. Free Download Manager. [En línea] [Citado el: 18 de marzo de 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(Iglesia_Anglicana)_%5BMac_OS_X_cuenta_14717_p/.

Anexos

Anexo No.1 Diccionario de datos.

Nombre de la entidad	Registro detallado de facturación.					
Descripción de la entidad	Registra toda la información sobre las facturas establecidas.					
Nombre del atributo	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clases válidas	**Clases no válidas**
id	Identificador de la factura	bigint	No	Sí	Números	Letras y caracteres extraños
calldate	Fecha de la llamada.	timestamp	No	Sí	Números y caracteres extraños	Letras
clid	Texto de identificación de la llamada.	varchar	No	Sí	Números	Letras y caracteres extraños
src	Identificación de origen.	varchar	No	Sí	Números y letras	Caracteres extraños
dst	Extensión de destino.	varchar	No	Sí	Letras, números y caracteres extraños	
dcontext	Contexto de destino.	varchar	No	Sí	Números	

					, letras y caracteres extraños	
channel	Canal usado.	varchar	No	Sí	Números, letras y caracteres extraños	
dstchannel	Canal de destino	varchar	No	Sí	Números, letras y caracteres extraños	
lastapp	Última aplicación.	varchar	No	Sí	Letras	Números y caracteres extraños
lastdata	Últimos parámetros de la aplicación	varchar	No	Sí	Números, letras y caracteres extraños	
duration	Tiempo total desde el discado en segundos	bigint	No	Sí	Números	Letras y caracteres extraños
billsec	Tiempo total, contabilizado en segundos.	bigint	No	Sí	Números	Letras y caracteres extraños
disposition	ANSWERED (atendida), NO ANSWER (no atendida), BUSY (ocupada), FAILED (falló)	varchar	No	Sí	Letras	Números y caracteres extraños
amaflags	Indicador. DOCUMENTATION(documentar), BILLING	varchar	No	Sí	Letras	Números y caractere

	(Contabilizar),OMMIT (omitir), IGNORE (ignorar)					s extraños
accountcode	Número de la cuenta usado	varchar	No	Sí	Números	Letras y caracteres extraños
billaplicada	Tarifa aplicada a la llamada.	doubleprecision	No	Sí	Números y el caracter extraño (.)	Letras y caracteres extraños
costo	Costo total de la llamada.	doubleprecision	No	Sí	Números y el caracter extraño (.)	Letras y caracteres extraños

Nombre de la entidad	Registro de cuotas					
Descripción de la entidad	Registra toda la información de las cuotas establecidas.					
Nombre del atributo	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clase válidas	**Clases no válidas**
id	Identificador	integer	No	Sí	Números	Letras y caracteres extraños
prefijo	Prefijo de la cuota	bigint	No	Sí	Números	Letras y caracteres extraños
name	Nombre de	varchar	No	Sí	Letras	Números y

	la cuota					caracteres extraños
billingmin	Tarifa por minuto	doubleprecision	No	Sí	Números y el carácter extraño (.)	Letras y caracteres extraños
destino	Extensión o Troncal a la que se le establece la cuota.	varchar	No	Sí	Letras, números y caracteres extraños	

Nombre de la entidad	Registro detallado de llamadas					
Descripción de la entidad	Registra toda la información de las llamadas realizadas.					
Nombre del atributo	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clase válidas	**Clases no válidas**
id	Identificador	bigint	No	Sí	Números	Letras y caracteres extraños
callerdate	Fecha de la llamada.	timestampt	No	Sí	Números y el carácter extraño (/)	
clid	Texto de identificación de la llamada	varchar	No	Sí	Números	Letras y caractere

						s extraños
src	Identificación de origen	varchar	No	Sí	Números y letras	Caracteres extraños
dst	Extensión de destino	varchar	No	Sí	Letras, números y caracteres extraños	
dcontext	Contexto de destino	varchar	No	Sí	Números, letras y caracteres extraños	
channel	Canal usado	varchar	No	Sí	Números, letras y caracteres extraños	
dstchannel	Canal de destino	varchar	No	Sí	Números, letras y caracteres extraños	
lastapp	Última aplicación	varchar	No	Sí	Letras	Números y caracteres

						extraños
lastdata	Últimos parámetros de la aplicación	varchar	No	Sí	Números, letras y caracteres extraños	
duration	Tiempo total desde el discado en segundos	integer	No	Sí	Números	Letras y caracteres extraños
billsec	Tiempo total, contabilizado en segundos	integer	No	Sí	Números	Letras y caracteres extraños
disposition	ANSWERED (atendida), NO ANSWER (no atendida), BUSY (ocupada), FAILED (falló)	varchar	No	Sí	Letras	Números y caracteres extraños
amaflags	Indicador. DOCUMENTATION(documentar), BILLING (Contabilizar),OMMIT (omitir), IGNORE (ignorar)	integer	No	Sí	Letras	Números y caracteres extraños
accountcode	Número de la cuenta usado	varchar	No	Sí	Números	Letras y caracteres extraños
uniqueid	Identificador único del canal	varchar	No	Sí	Números, letras y caractere	

					s extraños	
userfield	Campo definido por el usuario	varchar	Si	Sí	Números, letras y caracteres extraños	

Nombre de la entidad	**Configuración del fichero cdr_odbc.conf**					
Descripción de la entidad	Registra la información del fichero cdr_odbc.conf					
Nombre del atributo	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clase válidas	**Clases no válidas**
id	Identificador	bigint	No	Sí	Números	Letras y caracteres extraños
dsn	Nombre del servidor.	varchar	No	Sí	Letras, números y caracteres extraños	
username	Nombre de usuario.	varchar	No	Sí	Letras, números y caracteres extraños	
password	Contraseña.	varchar	No	Sí	Letras, números y caracteres extraños	

loguniqueid	Identificador único de entrada.	varchar	No	Sí	Letras, números y caracteres extraños	
tabla	Nombre de la tabla de la base de datos en la cual Asterisk va a almacenar los CDR.	varchar	No	Sí	Letras, números y caracteres extraños	

Glosario de términos

CDR: Call Detail Record. Fichero donde Asterisk almacena los datos relacionados a cada llamada.

CPU: Unidad de Proceso Central.

GPL: General Public License. Es una licencia creada por la Free Software Foundation y orientada principalmente a los términos de distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es Software Libre.

HTTP: HyperText Transfer Protocol.

IP: Internet Protocol. Es un número que identifica de manera lógica y jerárquica a una interfaz de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP.

Libre: Se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software.

Opensource: Código abierto es el término con el que se conoce al software distribuido y desarrollado libremente.

PBX: Private Branch Exchange. Es una central telefónica propiedad de una empresa privada, en contraposición con la central que es propiedad de un operador de telecomunicaciones o de una empresa de telefonía.

VoIP: Voz sobre IP. Término usado en telefonía IP para definir los servicios que se usan para transmitir voz usando el protocolo IP. Es el enrutamiento de conversaciones de voz a través de Internet o cualquier otra red basada en IP.

i18n: Internationalization. Es la abreviatura de la palabra internationalization, la primera letra que es la i + 18 caracteres que tiene la palabra + la n que es la última letra.

PlaTel: software para centros de contactos basado en VoIP.

Widgets: Controles. Pequeña aplicación o programa, usualmente presentado en archivos o ficheros pequeños que son ejecutados por un motor de widgets o Widget Engine.

SSL: Secure Sockets Layer o Protocolo de Capa de Conexión Segura. Proporciona comunicación segura sobre la red.

CGI: Common Gateway Interface. Permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web.

API: Application Programming Interface. Es el conjunto de funciones y procedimientos (o métodos en la programación orientada a objeto) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.