UNIVERSIDAD SAN FRANCISCO DE QUITO

DESARROLLO DE LA INTERFASE DE ADMINISTRACIÓN DEL SISTEMA DE COBRO DE MENSAJES PARA MOVISTAR

Marcelo Dávalos

Tesis de grado presentada como requisito para la obtención del título de Ingeniero en Sistemas

Quito

22 de Junio del 2006

Universidad San Francisco de Quito
Colegio Politécnico

HOJA DE APROBACIÓN DE TESIS

DESARROLLO DE LA INTERFASE DE ADMINISTRACIÓN DEL SISTEMA DE COBRO DE MENSAJES PARA MOVISTAR

Edmundo Marcelo Dávalos Ríos

Lorena Balseca, Master. _____
Directora de Tesis

Vinicio Carrera, DSC. _____
Profesor de Ingeniería de Sistemas

Fausto Pasmay, MBA. _____
Decano de Ingeniería de Sistemas

Fernando Romo _____
Decano del Colegio Politécnico

Quito, 22 de Junio del 2006

Resumen

El siguiente proyecto tiene como objetivo la creación de una interfase de administración para el sistema de cobro de mensajes de la Empresa Movistar.

Esta interfase permitirá administrar la información de los clientes y planes tarifarios del sistema de mensajería, como también monitorear el correcto funcionamiento del sistema mediante reportes y manipulación de procesos relacionados al sistema.

Para la realización del proyecto se ha escogido el modelo de caída de cascada que permite definir los requerimientos al inicio, luego diseñar y construir la interfase sin cambios mayores en los requerimientos iniciales. De esta manera el tiempo de desarrollo y presupuesto planificado pueden ser cumplidos de la mejor manera.

Como herramienta de desarrollo se utiliza el lenguaje C# que ofrece beneficios en sus características entre las que están: ser un lenguaje estable, orientado a objetos y de gran acogida en el mercado.

La arquitectura del sistema se define en tres capas, una del diseño de la interfase, otra que define las reglas del negocio y finalmente la capa de acceso a los datos que se conecta con la aplicación de cobro de mensajes para acceder a la información de la base de datos de éste.

Al terminar el desarrollo de la interfase, se prueba su funcionalidad y desempeño para verificar que se cumplan los requerimientos establecidos. Finalmente, se documenta el sistema, creando un manual de usuarios y nuevos requerimientos para una futura fase de mantenimiento.

Abstract

The following project's goal is the creation of an administrative interface for the messaging charge system of the Movistar enterprise.

This interface would permit to manage the client's information, the monthly plans from the instant messaging system and the correct behavior of the system by been able to manipulate the system's process.

To develop the interface, the water fall modeling strategy has been chosen. In this way the initial requirements are defined at the initial phase of the development with no changes later on. Then the work is continued with the design and codification of the interface. The results of this strategy are the accomplishment of the expected time and budged.

As a development tool it has been chosen the C# language which offers advantages with its special characteristics: stable language, object oriented and with a well acceptance in the market.

The architecture of the system is defined in three stages: the first one is the design of the interface; the following is the codification of the rules of the business and the last one is the access to the data which is done by the connection to the messaging charge system.

At the end of the development, the interface's functionality and performance are tested and verified to it accomplish the initial requirements. Finally, the user manual is created as well the conclusions and recommendations for the future use of the interface.

Agradecimiento

Gracias a las personas que me ayudaron para la conclusión de esta tesis y mis estudios universitarios, mi padre Robert Dávalos, mi madre Nancy Ríos, mi tío Fabio Dávalos, mi tía Janell Dávalos, mi novia Liliana Montenegro y mis compañeros que siempre me brindaron el apoyo para concluir cada año de estudio de la mejor manera.

Contenidos

Lista de Figuras

Lista de Tablas

Capitulo I Introducción

1.1 Antecedentes

MOVISTAR es una compañía Española de telefonía celular que se instauró en el Ecuador en el 2002, mediante la compra de acciones de Bellsouth Filial Ecuador. Esta empresa desarrolló una extensa e intensa campaña publicitaria que le ha permitido un posicionamiento en el mercado del pueblo ecuatoriano al captar un crecimiento de clientes potenciales y permanentes. Con una estrategia de negocio relacionada con servicios en telefonía de voz celular, mensajería escrita, y en un menor grado transferencia de datos con la mayor cobertura en la geografía ecuatoriana, Movistar se ha convertido en la más grande compañía de telefonía celular del País.

Con la finalidad de prestar una mejor cobertura de servicios en todo el país, Movistar posee celdas y centrales telefónicas en cada una de las provincias, pero el control técnico del servicio y los equipos de cobro se encuentran en la matriz de Quito ubicada en la Avenida República del Salvador.

Siendo MOVISTAR una empresa del Grupo Telefónica, ha formulado su estrategia de negocio sobre la base de declarar su visión en los siguientes términos:

- El Grupo Telefónica tiene el objetivo de convertirse en el mejor y mayor Grupo integrado de telecomunicaciones del mundo. Cuando Telefónica se refiere al mejor, alude al mejor en términos de orientación al cliente, innovación, excelencia operativa y profesionales. El mayor, en cuanto a rentabilidad para sus accionistas, crecimiento y creación de valor.

- Esta visión se basa en un programa de transformación denominado "Acelerar para ser más líderes" que se asienta en cinco pilares fundamentales: i) orientación al cliente, ii) innovación, iii) excelencia operativa, iv) liderazgo y compromiso con las personas, y v) identidad corporativa y comunicación.

Los cinco objetivos estratégicos a cumplir a su vez se describen como:

- ORIENTACION AL CLIENTE: La estrategia de la Compañía es crear un grupo con una visión orientada al cliente, considerando que se trata de un cliente "multiservicio", por tanto es conciente de la necesidad de posicionarse en el Mercado con una visión integral y uniforme que comprenda la oferta comercial de una serie de soluciones demandadas por el cliente y propuestas por el Grupo. De esta manera la empresa se orienta a crecer incrementando su base de clientes actual con ofertas innovadoras y a la medida de cualquier tipo de cliente.

- INNOVACIÓN: La empresa considera a la Innovación como uno de los pilares en los que se asienta su transformación y dedica plenamente el esfuerzo de sus empleados a la innovación tecnológica, comercial y de procesos.

- EXCELENCIA OPERATIVA: La empresa desarrolla y lleva a la práctica proyectos como: "nuevo centro de trabajo de Telefónica", que considera la tecnología para fomentar la colaboración y la movilidad; la "racionalización de la cadena de distribución", cuyo objetivo radica en la rotación y disminución de existencias, la "compra electrónica" y la consolidación de sistemas de planificación de recursos.

Este último objetivo, a su vez tiene a cargo proyectos de “mejora de la calidad de la banda ancha”, “gestión global de terminales”, “estándares de servicio revisados en contact centers” y “gestión de operaciones regionales”.

- LIDERAZGO Y COMPROMISO CON LAS PERSONAS: El liderazgo lo establecen por el compromiso y empoderamiento de cada uno de los 201.357 empleados del grupo en todo el mundo a septiembre de 2005. Para lo cual la empresa refuerza el valor de sus profesionales en pertenecer a la empresa y de crear una compañía basada en las personas con relaciones de confianza; y sobre esta base premia el desempeño personal seguido de oportunidades de desarrollo profesional.

- IDENTIDAD CORPORATIVA Y COMUNICACIÓN: Para la empresa, la Identidad Corporativa y la Comunicación son piezas clave en la proyección de su visión, posicionamiento y valores a todos aquellos con los que se relaciona: clientes, empleados, accionistas, autoridades y la sociedad en su conjunto.

1.2 Organización

La estructura organizativa de MOVISTAR, comprende los siguientes niveles:

- PRESIDENCIA
- CORDINACIÓN Y DESARROLLO DEL NEGOCIO
- FINANZAS Y DESARROLLO CORPORATIVO

Dentro de Coordinación y Desarrollo del Negocio tenemos la subdivisión de:

- ESTRATEGIA, PRESUPUESTACIÓN Y CONTROL
- COMERCIAL

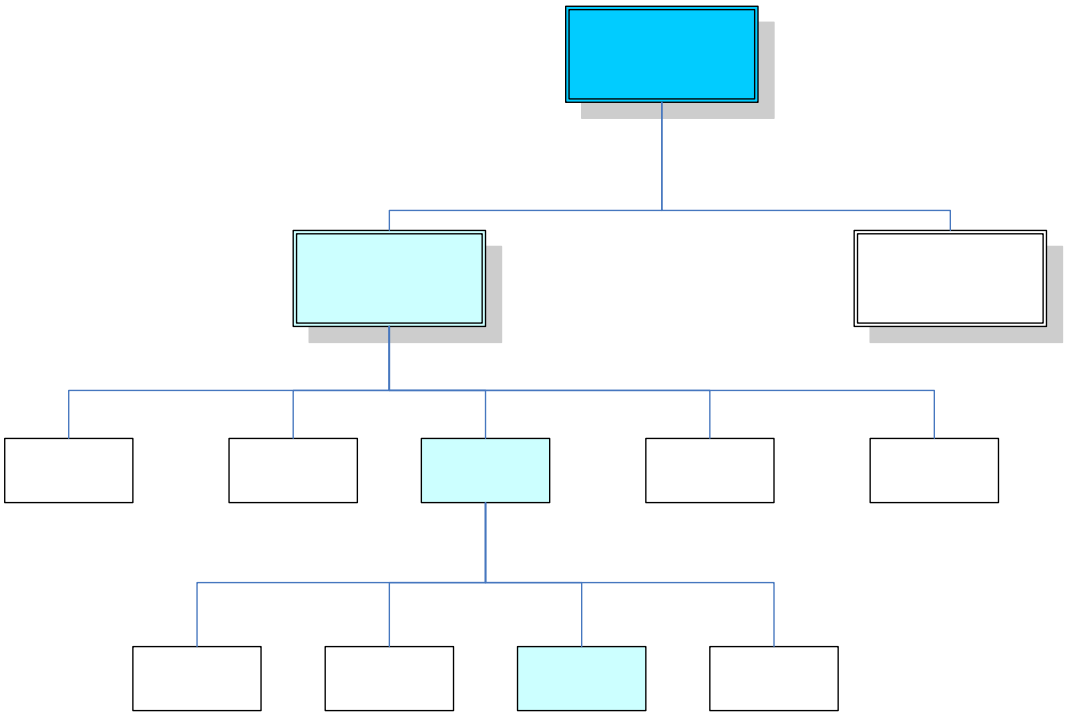
- INFRAESTRUCTURA Y SISTEMAS DE INFORMACIÓN
- COMPRAS
- RECURSOS HUMANOS

Específicamente con relación al Departamento de Infraestructura y Sistemas de Información que es el fundamento de estudio, engloba a su vez la siguiente distribución de organización:

- Infraestructura
- Desarrollo de Aplicaciones
- Administración de Aplicaciones
- Redes

Dentro de esta estructura, la Sección de Administración de Aplicaciones, es la encargada de la administración y control de las aplicaciones utilizadas en la empresa, que será la materia de estudio en la presente tesis, con relación a la aplicación de cobro por mensaje.

Por tanto, de acuerdo con el gráfico que se presenta la organización de la empresa es la siguiente:



Organigrama de Movistar

Figura 1.1

1.3 Sistema de Cobro por Mensaje

El sistema de cobro por mensaje utiliza una aplicación llamada Socket_Server, desarrollada por la empresa Novix Ltda., para realizar los cobros necesarios del servicio de mensajería.

El Socket_Server se encarga de identificar los mensajes que los usuarios envían, clasificarlos de acuerdo al plan que corresponden y realizar los cobros correspondientes a cada usuario. Para esto utiliza dos bases de datos donde se encuentra la información del cliente y el plan del cliente con sus características. Además, con la actualización de las base de datos, se puede conectar o desconectar el servicio de mensajería.

Esta aplicación se encuentra instalada en un servidor de la Empresa Movistar, y para iniciar su funcionamiento y verificarlo es necesario ingresar en el servidor y activar al Socket_Server. También, para controlar que la información se esté procesando correctamente, se ingresan individualmente en las bases de datos y se compara los datos.

1.4 Problemática del Sistema de Cobro de Mensajes

A pesar de que el sistema tiene manuales de usuario, siempre existen contrariedades y eventualidades que deben ser resueltas en forma inmediata, para lo cual la Empresa ha encargado esta función a un único administrador de la plataforma. Dicho Administrador se encarga en activar, reactivar o desactivar al Socket_Server cuando es necesario o si existe un error. También, el Administrador ingresa individualmente a cada base de datos para verificar el correcto funcionamiento y obtener información como estadísticas de uso del sistema de mensajería por parte de los clientes. Sin embargo, esta metodología de administración del Socket_Server es insegura e ineficiente puesto que se requiere crear tanto permisos al servidor como a las bases de datos para el Administrador y el personal de respaldo involucrado con el Socket_Server. Adicionalmente, para crear los reportes de uso del sistema de mensajería por los clientes, es necesario ingresar a la base de datos y crear la sentencia SQL correspondiente.

Para optimizar la administración del Socket_Server se requieren de una solución que permita el control de este sistema significativamente importante y vital para la empresa, pues aunque el mismo se encuentre operando en condiciones normales, no hay manera rápida y eficiente de reaccionar ante los continuos problemas que se presentan. De esta manera, en esta tesis se plantea la creación de una interfase que pueda ser instalada en

cualquier estación Windows dentro de la intranet de la empresa para verificar, controlar y manipular el funcionamiento del Socket_Server. Esta interfase se comunicará al servidor por medio de un puerto y enviará mensajes XML para realizar las peticiones de cambio de información o consultas.

1.5 Objetivos

Con la ampliación del Socket_Server e inclusión de la interfase se pretende alcanzar un mejor rendimiento en el servicio de mensajería, tanto a nivel de servicio al cliente como en la operación y monitoreo del sistema de cobro de mensajes. Por lo que los objetivos previstos a cumplir serían los siguientes:

- Facilitar la administración del Socket_Server sin la necesidad de ingresar al servidor u base de datos para verificar su funcionamiento.
- Limitar el acceso al servidor y bases de datos a los empleados de la empresa y que solo los usuarios autorizados del Socket_Server pueden tener acceso a la información por medio de la interfase.
- Realizar reportes con opciones rápidas para evitar la construcción de sentencias SQL.
- Incluir nuevas funcionalidades en el Socket_Server como la creación, modificación y borrado de planes de mensajes; provisionamiento y desprovisionamiento a los usuarios como tambien la reactivación de los mismos. Todas estas funciones sin la necesidad de ingresar directamente a la base de datos y manipular la información.
- Agilitar el trabajo del administrador de la aplicación para que este pueda ofrecer un servicio más efectivo y rápido a los clientes.
- Controlar los procesos relacionados con el Socket_Server de tal manera que se pueda levantar, bajar y modificar los mismos desde la interfase.

- Iniciar una nueva visión en el sistema de cobro de mensajería que luego podrá incluir otros servicios.

1.6 Importancia y Justificación

El negocio de la mensajería ha tenido una expansión en el mundo de las comunicaciones. Es uno de los servicios más utilizados por la gente común, en especial por los jóvenes. Esta revolucionaria forma de comunicación ha representado un potencial negocio para las empresas de telefonía celular.

Según los reportes de MOVISTAR, diariamente las ventas en mensajería alcanzan aproximadamente a los 20 mil dólares que representa un porcentaje significativo en las utilidades, dado que sus costos operativos son relativamente bajos y que utiliza los mismos recursos ya construidos para la comunicación por voz.

De esta manera, considerando que el servicio de mensajería produce una alta rentabilidad, se hace indispensable garantizar el correcto funcionamiento de cada uno de los componentes que hacen el sistema.

En el área de cobro de mensajes, el Socket_Server se ha desenvuelto bajo condiciones aceptables, pero abocado a múltiples eventualidades de problemas. Además, la Empresa ha experimentado un notable crecimiento en el servicio por efecto de un aumento en la clientela, lo cual ha sido determinante para incorporar nuevas funcionalidades al Socket_Server. En consecuencia se ha hecho indispensable tener una respuesta más rápida y efectiva en la realización de las tareas del administrador de la aplicación para agilitar el servicio al cliente como por ejemplo en el cambio de planes de mensajes, reactivaciones, reportes por usuario entre otros. Por tanto, en conjunto tanto los problemas que afronta continuamente el Socket_Server como por el efecto del

incremento de clientes y variedad de servicios que ofrece, se considera que requiere de la inclusión de la interfase de administración como una solución para la problemática en especial por el crecimiento de usuarios SMS. De esta manera lo que se logrará con el desarrollo de la misma será obtener reportes rápidos y agilitar el provisionamiento y control de servicio por cada usuario. Esta nueva innovación es la continuación del crecimiento de un negocio millonario que necesita ser confiable y garantizado.

Capitulo II Fundamentos de la Telefonía Celular

2.1 Historia de la Telefonía Celular

Un teléfono celular es un teléfono inalámbrico que la gente utiliza para comunicarse en estos días. Sin embargo, no fue sino hasta hace pocos años que la gente estaba limitada a comunicaciones cableadas. En 1843 un hombre llamado Michael Faraday estudió si se podía transmitir electricidad por el espacio. Faraday inició lo que más tarde se convertiría en la telefonía inalámbrica. En el año de 1865 el Doctor Mahlon Loomis fue la primera persona en comunicarse por medio de la atmósfera; él tuvo la idea de transmitir y recibir mensajes por medio de la atmósfera, utilizando la misma como un conductor. Luego, 32 años más tarde el italiano Guglielmo Marconi terminara el desarrollo de un sistema de telegrafía y recibirá la primera patente a nivel mundial en las comunicaciones sin cable. Así se inicia una nueva era en las comunicaciones sin cables.

Pero es Martín Copper conocido como el padre de la telefonía celular, quien contratado por Motorola en 1954 entra en la investigación de los celulares. Para el 3 de abril de 1973, Cooper realiza la primera llamada a su rival en los laboratorios de Bell en AT&T.

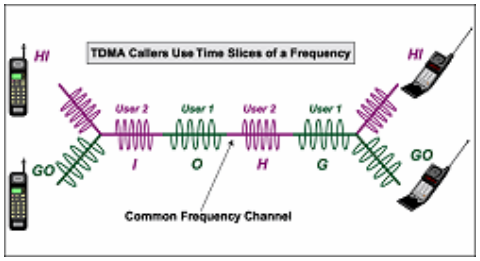
El primer teléfono celular de Motorola fue llamado Dyna-Tac que tenía las siguientes mediadas: 9 x 5 x 1.75 pulgadas y un peso de 2.5 libras. No tenía una pantalla, constaba de 30 tarjetas de circuitos, 35 minutos de tiempo para hablar, se recargaba en 10 horas y se podía hablar, escuchar y marcar la llamada.

En 1983 Motorola introduce una versión Dyna-Tac de 16 onzas para el mercado a un precio de 3500 dólares. Hoy en día hay más clientes de telefonía celular que de telefonía fija y con teléfonos de 3 onzas. (Marples 1)

2.2 Tecnologías Celulares

El negocio de las telecomunicaciones ha te tenido tal efecto en el mercado que se promocionó el desarrollo de varias tecnologías o protocolos para la transmisión de la voz y datos. Entre las mas comunes y utilizadas en nuestro medio están: TDMA (Time Division Multiple Access), GSM (Global System for Mobile) y CDMA (Code Division Multiple Access).

TDMA (Time Division Multiple Access) es un estándar digital que funciona bajo la idea de compartir el mismo medio de comunicación entre varios usuarios. La metodología se basa en asignar a cada uno de los usuarios un pequeño tiempo para que transmita su información o parte de la misma. En la Figura 2.1 se observa un ejemplo de la transmisión en TDMA donde una señal inicial HI es segmentada, enviada y reconstruida.

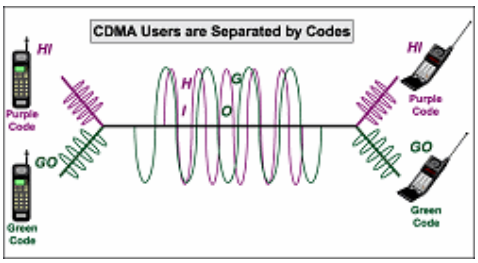


Funcionamiento de TDMA

Figura 2.1

Luego aparece GSM (Global System for Mobile) que es una tecnología desarrollada en base a TDMA, pero con la inclusión de GPRS (General Packet Radio Service) que permite la transmisión de información digital en los canales que no sean utilizados por TDMA y dedica el medio de transmisión solo a los usuarios que lo requieren.

Finalmente, tenemos a CDMA (Code Division Multiple Access) que trabaja en un principio utilizado en satélites militares donde cada conversación es asignada un código respectivo y enviada por el mismo medio (Figura 2.2). Esto se expresa como si una persona se encuentra en una multitud donde se hablan varias lenguas, y la persona percibirá solo aquellas conversaciones de su lenguaje natal. CDMA permite 4.4 trillones de códigos para asegurar que diferentes usuarios puedan acceder al medio al mismo tiempo. Además, CDMA utiliza tecnología de espectro extendido, lo que significa que las conversaciones son dispersadas en amplios segmentos en las transmisiones. Esto ayuda a evitar problemas en áreas de gran concentración y de geografía montañosa donde la señal puede ser reflejada.



Funcionamiento de CDMA

Figura 2.2

Sin embargo, la codificación que se realiza en la transmisión de la señal crea la necesidad en un mayor ancho de banda ya que se necesita enviar la señal codificada (más larga que decodificada) con su respectiva llave. Pero aún así CDMA permite tener

de 10 veces más llamadas al mismo tiempo que en TDMA. Otra ventaja de CDMA es la casi completa eliminación de ruido de fondo gracias a la codificación, y oportunidad de recibir señales a niveles de 1/25 a 1/1000 veces menos que en TDMA, permitiendo tener teléfonos con una vida mas larga de batería. Este tipo de protocolo es además un avance para la mejor transmisión de datos, fax móvil e identificador de llamadas.

2.3 Historia de los Short Messaging Service (SMS)

El servicio de mensajes cortos denominado "Short Messaging Service" (SMS) fue un éxito accidental que tomó a todos por sorpresa en la industria de las telecomunicaciones. Pocas personas predijeron que este servicio difícil de usar realmente funcionaría. Inicialmente, solo fue un servicio que las compañías de telefonía celular ofrecían gratuitamente sin importancia. Después, los SMS se convirtieron en un negocio millonario que revolucionaría desde las comunicaciones a los negocios y publicidad.

Este gran éxito de los SMS se debe a la identificación de los mismos con la juventud. Debido a su inicial servicio sin costo y difícil manejo para los adultos, se convirtió en la forma más atractiva de comunicación juvenil. Así, la moda de los SMS se expandió por todo el mundo convirtiéndose en una necesidad para el joven de hoy y en poco tiempo millones de mensajes eran enviados.

Luego, las compañías de telefonía móvil se dieron cuenta del gran negocio que se encontraba frente a ellos y comenzaron el cobro del servicio de los SMS. Sin embargo, los operadores de red estaban técnicamente limitados para cobrar a los consumidores por el uso de los SMS. Este problema se daba por la carencia de un enlace entre el sistema de cobro y el sistema de transmisión de mensajes. Así, los operadores de red en conjunto con los proveedores de las plataformas trabajaron para tratar de filtrar este

último evento. Finalmente, se inició el desarrollo de una aplicación de cobro de los SMS para los consumidores de prepago.

Al iniciarse el cobro de los mensajes se dio una caída inmediata del uso de los mismos. Pero algo interesante sucedió, el volumen de los mensajes SMS comenzó gradualmente a subir y pronto alcanzó una vez más los niveles de uso iniciales. Desde entonces el crecimiento ha continuado en ascenso con la ayuda de promociones y la suscripción de nuevos clientes.

Este accidental agujero inicial de la tecnología permitió que se creara una de las más grandes campañas publicitarias para ofrecer el servicio de mensajería, llegándose a convertir en un modo de vida de los jóvenes.

2.4 Funcionamiento del Sistema de Mensajería

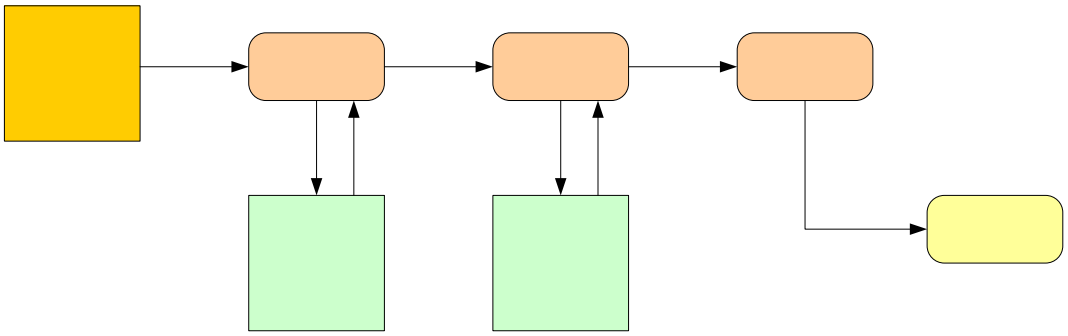
La transmisión de mensajes escritos es un proceso de guardar y reenviar mensajes entre teléfonos móviles. El mensaje (sólo texto) es enviado desde una estación móvil (celular), hacia una central proveedora de servicio en donde es guardado y luego es reenviado a su destino final. Para esto, en los casos en que el receptor no se halle disponible, podrá recibir el mensaje en intervalos de tiempo de espera, esto es 10 minutos más tarde si no se logró en el primer intento; si en esta ocasión no se logra, reenviará en una hora, y así esperará hasta 24 horas después. De no haber receptor, el mensaje será eliminado. Cada mensaje puede tener hasta un máximo de 150 caracteres. Estos caracteres pueden ser texto (alfanumérico) o binarios. En países como Rusia y China donde se tiene caracteres especiales, la capacidad es limitada a 70 caracteres.

Los mensajes SMS pueden ser enviados/recibidos simultáneamente con voz/datos/fax debido a que los canales utilizados para la comunicación son los no utilizados para la

transmisión por voz. De la misma manera que los otros servicios, los mensajes SMS soportan roaming internacional, esto significa que es posible mandar mensajes a otras partes del mundo. Por esta gran facilidad, varias de las tecnologías celulares como GSM, CDMA y TDMA soportan SMS.

En el caso de envío a través de GSM y CDMA, el proceso es el siguiente: El mensaje llega inicialmente al SMS_center (Short Message Service Center) que es un servidor localizado en la central, que guarda y reenvía los mensajes desde y para las estaciones móviles. El SMS_center analiza los mensajes y realiza una petición al HLR (Home Location Register), que es una base de datos en la central con toda la información sobre el usuario, para obtener la información de ruteo. El HLR contiene la información de los subscriptores y en que área se encuentran localizados. Así, se puede determinar a que MSC (Mobile Switching Center), descritos como switches computarizados localizados en cada región, se pasa el mensaje.

El MSC contiene un VLR (Visitor Location Register), una base de datos local a la región, que contiene información temporal sobre teléfonos móviles y en que células o grupo de células están ubicados. De esta manera usando la información del VLR, el MSC puede enviar el mensaje al correcto BSS (Base Station System) que transmite la información al celular destino. El BSS consiste de transmisores/receptores que envían y reciben información por el aire y desde o para la estación móvil. Esta información es pasada por los canales de señales de forma que se puede recibir mensajes aún cuando se esté realizando una llamada de voz. Finalmente, el MSC notifica al SMS_center que la operación se realizó. (Figura 2.3).



Transmisión de un Mensaje

Figura 2.3

2.4.1 Componentes de un mensaje SMS

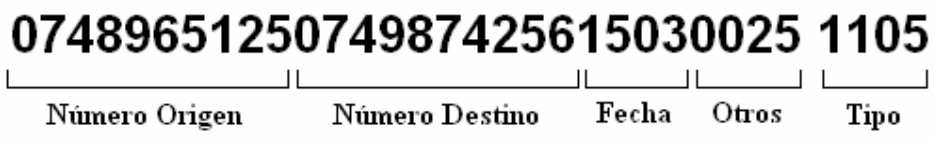
Para realizar todo este proceso de transmisión del mensaje de un móvil a otro es necesario añadir al mensaje la información necesaria sobre como debe ser tratado el mensaje por la estación SMS_center. Esta información se llama el encabezamiento del mensaje y lleva los siguientes parámetros:

- Encabezado de identificación del tipo de mensaje
- Service Center Timestamp: tiempo de la estación central que esta contenido en el teléfono.
- Dirección de origen: número de teléfono origen
- Dirección de destino: número de teléfono destino
- Identificador de protocolo
- Esquema de codificación
- Longitud de datos del usuario: dice que tan largo es el mensaje

Finalmente, a continuación del encabezado va el mensaje de texto. (Hutchison 1).

2.4.2 Incorporación del sistema de cobro al sistema de SMS

Luego de que el sistema de mensajería se convirtió en un negocio, se añadió a este sistema inicial el control de la mensajería. El primer paso fue lograr que se registre el tráfico de mensajería en archivos planos. Para esto fue necesario modificar al sistema del SMS_center, ubicado en el servidor SMS para que guarde ciertos parámetros del encabezado de cada mensaje. El resto del mensaje es ignorado para el control de mensajería ya que el volumen de mensajes es muy extenso. Entre los parámetros registrados están el número de origen, destino, fecha y campos adicionales. A esta trama se le ha nombrado como un CDRS y tiene un formato similar al siguiente ejemplo:



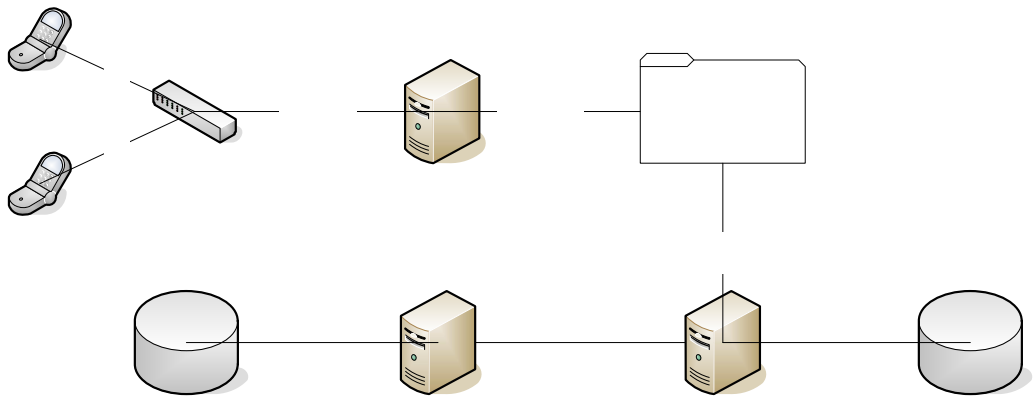
Esta información es la suficiente y necesaria para iniciar el proceso de cobro. El teléfono de origen será al que se le descontará de su saldo, la fecha nos indica a que mensualidad pertenece y el tipo que se clasifica en prepago, pospago o hibrido nos dice como tratar al cliente, si se descuenta instantáneamente, o si se le factura. Por ahora los otros campos no son utilizados, pero eventualmente con el gran incremento del volumen, se deja el espacio abierto para poder cobrar por el tipo de prioridad en caso de requerirse.

Cuando se alcanza un número de 500 CDRS guardados en un archivo plano, el Sistema SMS_center envía dicho paquete de información vía FTP a un segundo servidor, seguido de una señal de confirmación para notificar el final de la transmisión. Este

segundo servidor llamado CobroSMS, ubicado en la ciudad de Guayaquil, contiene toda la aplicación del sistema de cobro de mensajes que procesará los archivos de CDRS recibidos.

Adicionalmente, el Servidor CobroSMS se encuentra conectado a dos bases de datos. La primera será la base de datos SMSDatos donde se registrará las transacciones que realice la aplicación, los planes de mensajería, y los números celulares suscritos al servicio de mensajería. La otra base de datos es la central de clientes donde se tiene el saldo de los clientes y la descripción detallada de cada uno de ellos, esta última es accedida a través del Servidor Central de la compañía.

En la figura 2.4 se puede observar el sistema de cobro de mensajes; donde los celulares, switch central y Servidor SMS representan la transmisión del mensaje vista en la Figura 2.3. Luego tenemos el archivo generado por el servidor SMS y enviado al servidor CobroSMS que contiene al Socket_Server, el mismo que se encuentra conectado a las bases de datos y al Servidor Central.



Sistema de Cobro de Mensajes

Figura 2.4

2.4.3 Aplicación de Cobro de Mensajes

En el servidor CobroSMS es donde se realiza el resto del procesamiento de mensajes, control de mensajería, actualización de datos y provisionamiento de servicios. La aplicación de cobro de Mensajes está dividida en módulos que realizan funciones diferentes. Los módulos son: Clasificador, Prepago, Híbrido, Pospago, Socket_Server, Provisionador y Reloj. Estos módulos se llaman entre ellos una vez que realizan su operación designada.

El primer módulo, *Clasificador*, contiene dos procesos por los que pasan los archivos. El primero es un Spliter que divide los archivos que llegan en grupos de 500 CDRS (un CDRS es un mensaje), para ser procesados. Luego van hacia un *procesador,* que está conectado con la base de datos central y verifica si el celular de origen está provisionado con el servicio. Si no lo está desprovisiona el servicio de mensajería. Pero si está provisionado, clasifica al CDRS en una de las categorías que se manejan en la empresas, prepago, pospago, hibrido o error. Cabe resaltar que debido a la gran cantidad de volumen que existe, se crean varias copias de esta aplicación para que procese varios archivos al mismo tiempo.

Una vez clasificado el CDRS el módulo anterior llama a uno de los siguientes módulos para continuar con el proceso, y en caso de un CDRS con error notifica para registrarlo y que mas tarde sea revisado por el Administrador. Este sistema tiene tres módulos que representan las categorías de clientes de la compañía: prepago, híbrido y pospago. Cada uno de estos, actualiza los contadores de los planes correspondientes, y si el número de mensajes sobrepasa el número que se contiene dentro del plan, cada mensaje adicional es cobrado al cliente. Si el CDRS va a prepago, se verifica que el contador sea menor al número de mensajes del paquete de mensajería que tiene el cliente. Si lo es, se actualiza

los contadores. Si no lo es, se conecta a la base de datos central y se realiza un débito a la tarjeta o saldo restante del cliente. En caso que no exista saldo notifica para que se realice la correspondiente desconexión.

El módulo de híbridos es similar, ya que es un plan que tiene una mezcla de prepago y pospago. En híbridos, se actualiza el contador, si este es menor al número de mensajes dentro del plan no se hace nada, pero si el contador es superior, se descuenta del saldo del cliente. Y si no existe saldo se procede a la desconexión.

Pospago es un poco diferente a los demás, aquí sólo se actualiza el contador. Al final del mes se realiza una cuenta de los mensajes enviados, se resta de los mensajes del plan, y si existen mensajes de exceso, se añade a la facturación del cliente.

Estos cuatro módulos, desde el clasificador para realizar las desconexiones y notificación de error, hasta los módulos de híbridos, prepago y pospago para concluir su trabajo, utilizan un módulo adicional llamado Socket_Server que cumple varias funciones de acuerdo a peticiones que se realizan desde diferentes usuarios. El Socket_Server se encuentra directamente conectado a otros servidores y aplicaciones, entre los cuales se tiene la base de datos SMSDatos y la base de datos central, además, de las aplicaciones para cobro y conexión/desconexión de la central de telefonía (generalmente en otro servidor).

El módulo Socket_Server cumple varias funciones, como crear componentes nuevos (paquetes de mensajería con sus características), desactivar los componentes, provisionar a los números con un componente especifico, desprovisionar, reactivar un número, desconectar, y actualizar datos de cobros en la base de datos SMSDatos y la central.

Para todas estas funciones, el Socket_Server recibe peticiones por mensajes XML desde los módulos Prepago, Híbrido, Pospago e incluso Clasificador; procesa estas peticiones y las ejecuta. Por ejemplo, si el módulo de prepago, detecta una trama, analiza el mensaje y manda a actualizar los contadores en la base de datos. Para esto crea una trama XML con la correspondiente petición y la envía al Socket_Server. Esta última se comunica con las bases de datos y realiza la transacción. Pero si el módulo de prepago detecta que ya no existe saldo en el celular origen, crea una petición XML con destino al Socket_Server para proceder con la desconexión del servicio de mensajería.

Pero el proceso no termina en el Socket_Server. Para las conexiones y desconexiones, como cobros, el Socket_Server se conecta a los servidores centrales y realiza las peticiones de cobros a los mismos, o al servidor de conexiones para conectar a un usuario. En estos servidores centrales existen otras aplicaciones que reciben las peticiones y ejecutan devolviendo un resultado o confirmación.

Adicionalmente existen otros dos módulos que realizan funciones fuera del cobro directo de los mensajes receptados. El primero es el *Provisionador,* que es llamado por el servidor central para provisionar del servicio a los clientes. Por ejemplo si un cliente ingresa una nueva tarjeta, el provisionador recepta el pedido y se conecta con el Socket_Server para proceder con el proceso de activación de un plan de mensajería al respectivo cliente.

Finalmente, está el módulo de reloj que se encarga de determinar cuando el tiempo mensual de un plan ha concluido, y si el cliente está provisionado con saldo, reactiva el plan de mensajes, reseteando los contadores a cero.

En el siguiente diagrama se muestra como el sistema está conformado y se puede observar el trabajo de cada uno módulos descritos. (Figura 2.5). (Manzano).

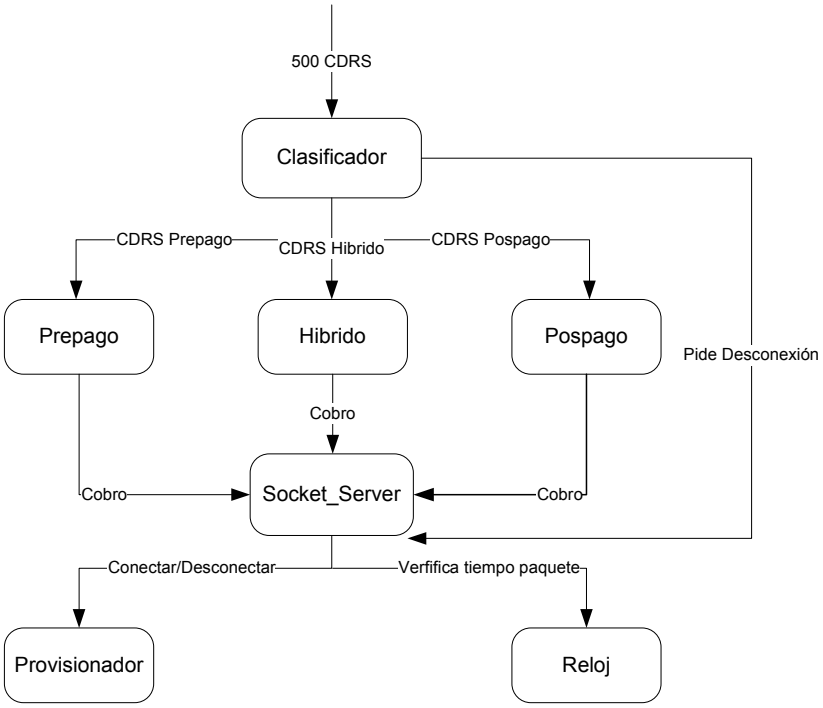


Diagrama de Funcionamiento del Sistema de Cobro de Mensajes

Figura 2.5

2.5 Una nueva innovación al Sistema de Cobro de Mensajería

Observando el funcionamiento del Sistema de Cobro de Mensajes, se puede establecer la carencia de una interfase de administración. El sistema funciona correctamente y produce los resultados esperados. Sin embargo, no es un sistema libre de fallas y presenta grandes limitaciones para el administrador de la plataforma. El siguiente paso a desarrollarse en este sistema es el desarrollo de la interfase gráfica de administración que ofrece posibilidades de interacción con el sistema mismo, las bases de datos y los procesos que corren en el servidor relacionados con esta plataforma. Esta interfase se

comunicará vía mensajes XML al Socket_Server y la misma responderá con mensajes XML que contendrán la respuesta correspondiente a la petición inicial. De esta manera se limita el ingreso del Administrador al sistema operativo del servidor o a la base de datos SMSDatos. Para incorporar estas nuevas innovaciones es necesario realizar el diseño de la interfase, desarrollo de la misma y modificar al Socket_Server para incorporar nuevas características.

Capitulo III Metodología y Herramientas

Como todo sistema a realizarse, el desarrollo de esta interfase requiere el uso de metodologías y herramientas. Estas metodologías y herramientas son escogidas en base a las necesidades del cliente, tiempo en realizarse y sus objetivos.

El primer paso es escoger una metodología que nos lleve a nuestros objetivos en forma clara y en el tiempo planeado.

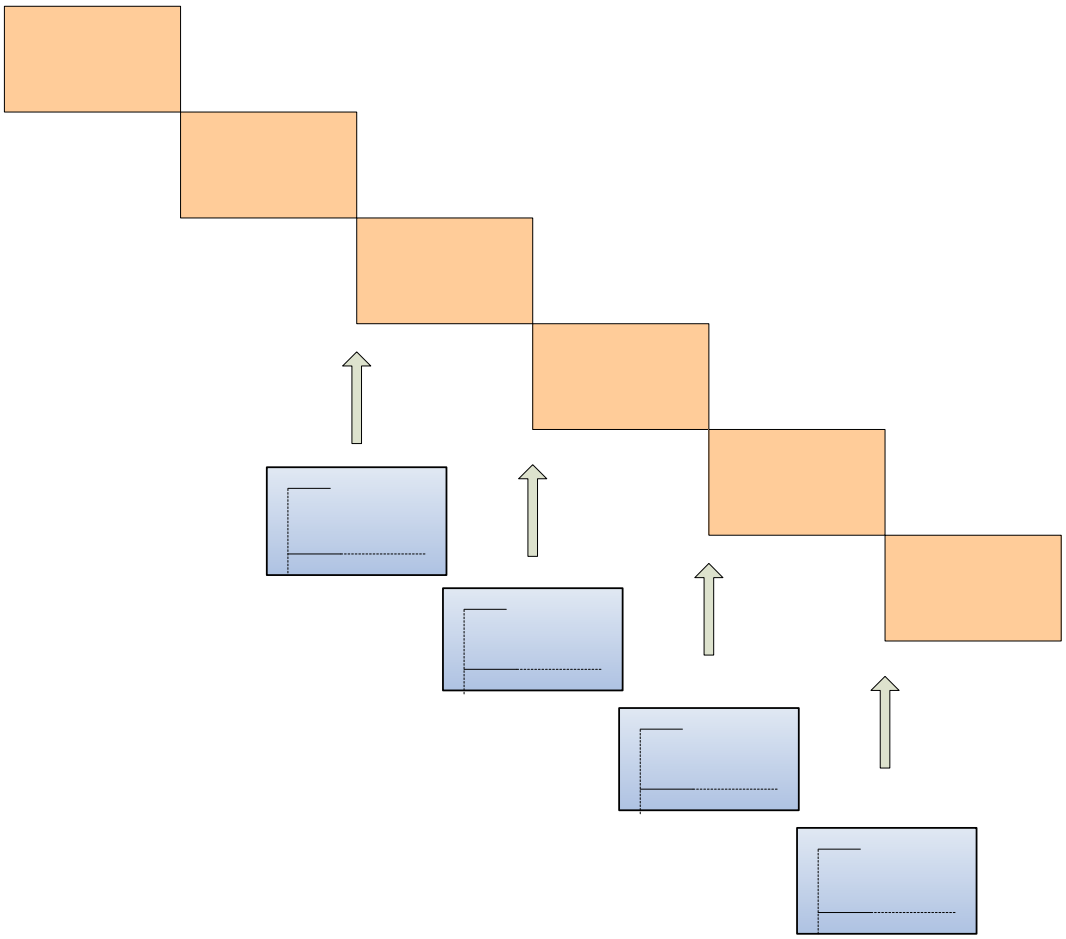
3.1 Metodología

Para el desarrollo de software se utiliza la ingeniería de software que es la práctica de usar técnicas selectas para mejorar la calidad del desarrollo. Esto se basa en debates y experiencias que nos llevan a concluir que el uso de una metodología trae mejores resultados en el desarrollo de programas, con menores defectos, tiempo y costo. Además, se trata de evitar pasos innecesarios que incrementan el tiempo y disminuyen la productividad de los desarrolladores.

Para determinar la metodología en el desarrollo de la interfase gráfica del sistema de cobro de mensajes se debe considerar los objetivos del proyecto, tiempo de realización, personal involucrado y requerimientos establecidos. En este caso los requerimientos del sistema ya han definidos y documentados. Además, el tiempo es limitado ya que se espera que el producto final esté en producción en dos meses y medio. Tomando en cuenta esta información se ha decidido optar por la metodología de Caída de Cascada para el desarrollo del proyecto, ya que esta se enfoca en el tiempo de desarrollo y considera que los requerimientos son aclarados y limitados a cambios desde la fase inicial.

3.1.1 Modelo de Caída de Cascada

Todos los proyectos pueden ser manejados de una mejor manera cuando son segmentados en forma jerárquica. Cada segmento es llamado fase, estado, actividad, evento o paso. En el desarrollo de software, la manera más simple de segmentación es la metodología de caída de cascada que se muestra en la figura 3.1.



Modelo de Caída en Cascada

Figura 3.1

Viendo este gráfico, es posible notar que los requerimientos han sido definidos en forma precisa y clara al inicio. Esto debe asegurarse para evitar errores en las etapas siguientes.

También se observan principios básicos de una buena metodología:

- El trabajo es dividido en secciones para facilitar su desarrollo.
- Las revisiones se realizan al final de cada sección
- Además, las revisiones representan el control de calidad y puntos de decisiones para pasar a la siguiente sección.

Consideradas dichas características descritas de este modelo de desarrollo se puede identificar como los elementos del modelo de Caída de Cascada se acoplan a las necesidades del desarrollo de la interfase puesto que las primeras fases ya han sido definidas y el desarrollo se encuentra listo para iniciarse en su etapa de diseño.

3.1.2 Actividades Básicas para el Desarrollo de Software

Como se dijo anteriormente el desarrollo de software está compuesto de varias actividades que tienen que cumplirse. A continuación se detalla las características de cada una de los segmentos del modelo de Caída de Cascada.

3.1.2.1 Inicialización

En la primera sección del modelo de Caída de Cascada se define la especificación del sistema describiendo las funciones, requerimientos del sistema informático y los

principios que gobiernan su desarrollo. Estas especificaciones sirven como el fundamento para la organización y planificación de hardware, software, base de datos y recursos humanos a utilizarse. Una vez obtenida esta información, se organiza y documenta en el Plan del proyecto.

En el desarrollo de la interfase para el cobro por mensaje se resume esta sección con la formulación de la propuesta del proyecto, aprobación del contrato del proyecto y definición del personal involucrado.

3.1.2.2 Análisis y Especificación de Requerimientos

Las especificaciones de los requerimientos juegan un papel importante en el proceso de desarrollo. Son utilizadas para acumular información necesaria para la introducción en el producto, su funcionalidad, diseño, procedimiento de implementación y pruebas del sistema. (Zammit 1).

El análisis de requerimientos se inicia estudiando lo que se espera del sistema y como este encaja en el plan del proyecto escogido en la etapa previa. Así, se puede ver el software en un contexto global en el negocio y que posición ocupará dentro de la empresa. Por ejemplo, se define que información se necesita desplegar en la interfase, como también que se puede modificar y que no. Para terminar, se documenta el procedimiento y especificaciones técnicas de dicho sistema describiendo el alcance del proyecto, que se hará y que no.

Para el desarrollo de la interfase, se tomaron los requerimientos ya listados por el administrador del Socket_Server de acuerdo a sus necesidades diarias. Además, se consideró las peticiones del supervisor del área y se propuso recomendaciones para mejorar la funcionalidad del sistema. El detalle de estos requerimientos se puede observar en la primera sección del Capítulo 4.

3.1.2.3 Diseño

Una vez que el documento de especificaciones está terminado, los desarrolladores de software tienen una idea clara de las funciones que el sistema tiene que realizar. La información en el documento de especificación de requerimientos es usada para descomponer el sistema en módulos y describir que debe hacer cada uno de estos, además de como están relacionados cada uno de estos módulos. Así se comienza a crear la arquitectura del sistema.

El diseño de la interfase de Administración considera la creación de la interfase y además la modificación del Socket_Server con el que se comunicará. Primeramente, se diseñará el GUI de la interfase, sus componentes y navegación. Luego se pasará a planificar la funcionalidad de los elementos de la interfase para que cumplan con las reglas del negocio y den el formato apropiado. Finalmente, para la comunicación con el Socket_Server se determinará el formato de las tramas XML a transmitirse para poder comunicar a la Interfase con el Socket_Server. Estos elementos se detallan en el capitulo 4.

3.1.2.4 Implementación

Durante la implementación se construye y prueba la interfase. Toda la documentación es producida en una forma casi final, incluyendo la información interna del código. Se debe revisar y verificar el código para que este sea preciso y completo, que implemente lo que se planeó en el diseño y que se aplique buenas prácticas de programación definidas anteriormente con referencia a los respectivos estándares. Los documentos deben ser inspeccionados para verificar que estén completos y que puedan ser utilizados como documentación de respaldo. Además se crea el manual de usuario del Sistema.

3.1.2.5 Integración y Pruebas

La fase de integración y pruebas consta en verificar el funcionamiento final del sistema, para lo cual se realizan distintas pruebas dependiendo de los objetivos iniciales. Entre las pruebas a utilizarse están la de de funcionalidad que consta en verificar cada una de las opciones del sistema comprobar su funcionamiento. Otra prueba utilizada es la de desempeño que se utiliza para ver el tiempo de respuesta del sistema.

Una vez verificado el funcionamiento, se pasa a poner el sistema en producción, para lo que se respalda la información inicial y se pone el sistema a disposición de los usuarios finales. En esta etapa se monitorea el correcto funcionamiento y se capacita al personal para que pueda utilizar el producto final.

3.1.2.6 Mantenimiento

El mantenimiento de un sistema consta en monitorear su correcto funcionamiento y realizar cambios en el código en caso que se produzcan errores. Los cambios en software pueden variar en lo que abarcan; puede ir desde una simple corrección ortográfica o de colores de interfase hasta una corrección que exigirá un proceso de ciclo de vida como una reingeniería.

Siguiendo cada uno de los pasos descritos, se garantiza que el producto final cumpla con la calidad necesaria y los objetivos especificados inicialmente. Tambien, se realiza el desarrollo en un tiempo aproximado al planeado y con la limitación de cambios en los requerimientos.

3.2 Herramientas a Utilizarse

En el mercado de desarrollo de software existe una variedad de herramientas para realizar el proyecto deseado. Muchas ofrecen diferentes facilidades para el usuario, permitiendo agilitar su trabajo. Para la realización de este proyecto se escogió Visual Studio C# (C-Sharp). Este nuevo lenguaje presenta varias facilidades gráficas y formas preconstruidas que lo catalogan como una solución rápida y efectiva.

3.2.1 Ventajas de C#

La migración a la plataforma .NET ofrece nuevas posibilidades a escoger. C# (C – Sharp) es un nuevo lenguaje de programación que unifica las diferentes facilidades de

C++, Java y Visual Basic. Estas características, más la inclusión del framework de Microsoft permiten una funcionalidad avanzada y de fácil uso para la creación de interfases con extensas funcionalidades. Entre las características de utilidad para el proyecto de C# están:

3.2.1.1 Lenguaje Modernizado

En el mundo de la programación el lenguaje C es muy conocido y utilizado para el desarrollo de las aplicaciones de distintas empresas. Luego de C y basado en este, salió al mercado C++ pero como un lenguaje orientado a objetos. Finalmente, el nuevo descendiente de esta familia es C# que trae las funcionalidades de C++ pero adicionalmente la facilidad de Visual Basic. Este nuevo lenguaje toma ventaja de las características del Framwork de .NET que permite acceso a generadores de formas automáticas.

La gran aceptación de C# en el mercado garantiza un mantenimiento fácil y la posibilidad de ampliación de los sistemas realizados. Por ejemplo, para la interfase gráfica se requiere un lenguaje popular y que se mantenga en el mercado para contar con el respaldo de varios programadores que puedan dar un futuro mantenimiento.

3.2.1.2 Seguridad de Tipos

La seguridad de tipo es una característica especial de C# que se encarga de alertar el acceso a espacios de memoria no asignados. Por ejemplo en C# el compilador notifica

que una variable no a sido inicializada o que se intenta obtener el valor de un elemento que no se encuentra dentro de un arreglo como el 4 elemento de un arreglo de 3.

3.2.1.3 Orientación a Objetos

Al igual que otros lenguajes en el mercado C# utiliza la programación orientada a objetos, sin embargo, lleva este concepto un paso mas allá. En C# todos los tipos de datos tienen métodos asignados incluso los datos primitivos.

3.2.1.4 Simplificación de Sintaxis

Por la complejidad de los programas de hoy en día C# simplifica la sintaxis para ser mas consistente y lógico, eliminando algunas de las más difíciles características de los lenguajes anteriores. Por ejemplo, C# no utiliza el uso de punteros de tal manera que no se usa directamente la manipulación de memoria.

Otro caso de reducción de código son los operadores de referencia. En C# solo se tiene un operador de referencia para llamar los métodos de un objeto. Este operador es el punto (.) que permite llamar a las funciones relacionadas con el objeto.

Finalmente, el problema de manejo de memoria queda a cargo del framework .NET, y la utilización de un colector de basura. De esta manera se va marcando lo que ya no es necesario y liberando memoria mientras se va necesitando.

3.2.1.5 Fuera de Microsoft

Un nuevo cambio que se realizó en C# es la salida de las manos de Microsoft, y pasó a ECMA (Industria Internacional de Estándares de Información y Comunicación), donde se publicó como un estándar. Además de esto, existen intentos para pasar todo el framework a una versión de código abierto, permitiendo su uso en Linux y otros sistemas operativos.

3.2.1.6 Componentes

La creación de componentes es directa, referenciado estos al código. Los nombres de espacio en C# resuelven muchos de los problemas del mundo COM en el cual se tenía que realizar búsquedas de registros y crear ejemplos de objetos. Usando C# simplemente se importa el nombre del espacio y se puede comenzar a utilizar las clases en esos componentes, sin búsquedas de registros. (Albahari 1).

Considerando todas estas características se concluye que C# es una solución práctica y efectiva para la creación de la Interfase que se desarrollará en esta tesis. La ventaja de tener herramientas gráficas para el diseño y respaldadas de un lenguaje estable garantiza un producto de alta calidad.

3.3 Desarrollo de Tres Capas

Luego de conocer la metodología para el desarrollo del proyecto y la herramienta a utilizarse, se escoge la arquitectura del funcionamiento del programa. De esta manera

se modulariza el trabajo y ven los componentes que forman el producto final. Para este proyecto se utilizará el desarrollo en tres capas que se describe a continuación.

Se llama desarrollo en tres capas debido a que toda la aplicación es dividida en tres secciones especializadas en campos determinados. Estas secciones o más bien dicho capas son llamadas: capa de presentación, capa de negocios y capa de acceso de datos.

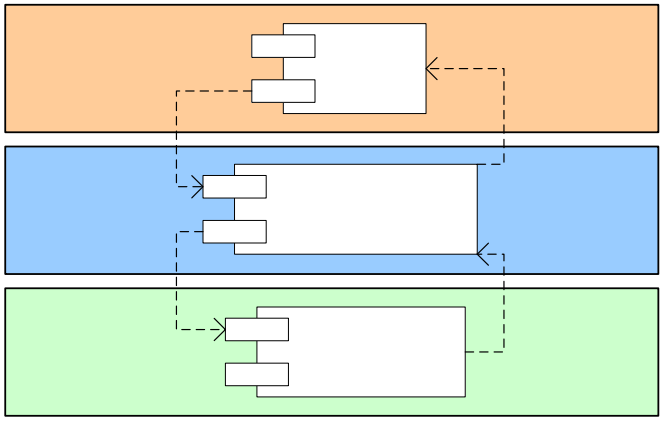
La capa de presentación está formada por los Componentes de IU, y los componentes de proceso de IU. Aquí se desarrollará la parte gráfica de la interfase, con la inclusión de botones, espacios de texto, ventanas, menús y otros componentes. También, se crea reportes de la información requerida y se orienta los esfuerzos del desarrollador para crear una interfase de fácil navegación y manejo.

En la siguiente capa, de negocio, se encapsula la lógica del negocio. Para el caso de la interfase de Administración existe la necesidad de cambiar de formato, realizar cálculos, transformar los datos para su procesamiento y manipular otros procesos del servidor_SMS. Este trabajo se lo realiza con funciones especializadas y desarrolladas basadas tanto en los requerimientos del usuario, como en las necesidades de la base de datos y el Socket_Server. Aquí no solo se realiza funciones o clases en la interfase si no también se realizan cambios en el Socket_Server.

La tercera y última capa es la de acceso a datos. Generalmente, es aquí donde se definen las clases y sentencias de consulta para realizar las peticiones correspondientes a la base de datos. Sin embargo, la interfase de Administración a desarrollarse no tiene una conexión directa con la base de datos si no que crea tramas de XML con las peticiones que serán evaluadas y realizadas por el Socket_Server; el mismo que se comunica con la base de datos para concluir con el proceso de solicitud de información.

De igual manera el trabajo de acceso a datos requiere que se intervenga el código del Socket_Server como en el de la interfase.

A continuación se observa las tres capas divididas y sus componentes internos en un ámbito general. (Figura 3.2).



Arquitectura en Tres Capas

Figura 3.2

3.3.1 Desarrollo de Tres Capas en C#

Entendiendo la arquitectura de tres capas es fácil identificarla con el desarrollo en C#. Primero tenemos la capa de aplicación que es ejecutada fácilmente con la interfase gráfica que ofrece Visual Studio. Los componentes como botones y menús son alineados en una barra de herramientas. Para añadir los componentes se escoge los íconos de los mismos con el ratón y se los lleva a la forma de diseño. Una vez que están ubicados los componentes respectivos podemos acceder a las propiedades de los

Capa

mismos para especificar sus características de color, tamaño, tipo de letra, orden de navegación formato de los contenidos.

Una vez terminada la primera capa pasamos al desarrollo de la capa de de Negocio donde se especifican las reglas del negocio y funcionalidad de la aplicación. Aquí programamos cada clase y creamos las funciones necesarias para procesar la información recibida del usuario como la información recibida de la base de datos. Tambien damos formato a la información para que sea legible por ambas partes. En otras palabras en esta capa es donde se realiza todo el procesamiento de datos y se regula los mismos de acuerdo a las reglas del negocio previamente establecidas.

Finalmente, está la capa de Acceso a Datos. Esta capa es un conjunto de clases que se encargan de la comunicación entre la base de datos y la aplicación. Para este acceso se puede realizar de diferentes maneras, ya sea utilizando las librerías de C# con funciones predefinidas (SQLHelper) o creando nuestras propias clases para la comunicación con la información.

Dicha arquitectura utilizada, Tres Capas, en conjunto con el lenguaje C# dividen el trabajo a realizarse en forma ordenada y precisa. Así, el desarrollo del sistema puede ser modularizado y corregido con mayor facilidad por distintos desarrolladores.

3.4 Comunicación por XML

XML (Extensible Markup Language) es un texto simple y flexible que sigue un formato específico con la finalidad de definir lenguajes. Los elementos que lo componen pueden

dar información sobre lo que contienen, no necesariamente sobre su estructura física o presentación. Este formato fue internacionalizado como una metodología de nombrar elementos de una estructura de datos y transmitir la información a un destino de tal manera que el mismo pueda entender y diferenciar cada uno de los datos enviados. Así, gracias a XML es posible definir una estructura y entender su significado.

Se puede suponer de este modo que XML constituye la capa más baja dentro del nivel de aplicación, sobre el que se puede montar cualquier estructura de tratamiento de documentos, hasta llegar a la presentación. Y así podemos ver la compartición de documentos entre dos aplicaciones como intercambio de datos a ese nivel.

3.4.1 Estructura del XML

El metalenguaje XML consta de cuatro especificaciones (el propio XML sienta las bases sintácticas y el alcance de su implementación):

DTD (*Document Type Definition*): Definición del tipo de documento. Es en general un archivo que encierra una definición formal de un tipo de documento y a la vez especifica la estructura lógica de cada documento.

XSL (*eXtensible Stylesheet Language*): Define o implementa el lenguaje de estilo de los documentos escritos para XML. Permite modificar el aspecto de un documento. Se puede lograr múltiple columnas, texto girado, orden de visualización de los datos de una tabla, múltiples tipos de letra con amplia variedad en los tamaños.

XLL (*eXtensible Linking Language*): Define el modo de enlace entre diferentes enlaces. Se considera que es un subconjunto de HyTime (*Hipermedia/Timed-based structuring Language* o Lenguaje de estructuración hipermedia/basado en el tiempo, ISO 10744) y sigue algunas especificaciones del TEI (*Text Encoding Initiative* o Iniciativa de codificación de texto).

XUA (*XML User Agent*): Estandarización de navegadores XML. Todavía está en proceso de creación de borradores de trabajo. Se aplicará a los navegadores para que compartan todos las especificaciones XML. (Garcia 1).

En este sistema se utiliza XML tipo DTD para formar una estructura de datos que contienen las peticiones y respuestas entre la interfase y el Socket_Server. De esta manera estas dos partes se comunican en forma clara y con un formato específico. La estructura de los textos XML puede ser observada en el Anexo II.

Capitulo IV Desarrollo de la Interfase Gráfica

Este capítulo describe el desarrollo de la interfase gráfica de administración para la plataforma de cobro de mensajes de la empresa Movistar. La interfase se comunicará con el módulo Socket_Server que es el corazón de la plataforma de cobro y administrará los procesos levantados como también podrá manipular la información de los clientes de mensajería.

Para desarrollar este sistema se utilizará la metodología de caída en cascada, primero definiendo los requerimientos; luego diseñando el sistema y la interfase; después codificando en el lenguaje C# con su respectivo control de calidad y pruebas de funcionamiento y finalmente instalando el producto final.

A continuación se describe cada uno de los pasos nombrados para el desarrollo de la interfase de administración.

4.1 Levantamiento de Requerimientos

La finalidad de esta interfase es la administración de la aplicación desde una estación remota y sin la necesidad de acceder al servidor, garantizando la seguridad de la información en el mismo.

Además, la interfase debe contener las suficientes funcionalidades y vistas, de la información de los usuarios del servicio de mensajería, para el administrador, de tal manera que no sea necesario ingresar directamente en el servidor. Estas visas y

funcionalidades se basan en el trabajo diario del administrador y de acuerdo a sus necesidades.

Todos estos requerimientos ya han sido definidos y aprobados por la empresa para luego ser entregados para el desarrollo de esta tesis.

A continuación se describe los requerimientos definidos para la interfase administrativa:

Manipulación de planes de mensajería que especifican el número de mensajes que un usuario puede mandar a un precio establecido:

- Creación de un nuevo plan.
- Eliminación del plan.
- Edición del plan.
- Búsqueda del plan.

Provisionamiento del servicio al usuario.

- Provisionamiento al usuario con un plan de mensajería.
- Desprovisionamiento al usuario.
- Edición del provisionamiento.
- Búsqueda del usuario con su respectivo plan de mensajería asignado.
- Reactivación del Usuario con el servicio.

Reportes de mensajería

- Reporte por número con detalle de los mensajes enviados y a que categoría pertenecen los mensajes (Movistar – Movistar, Movistar – otros, internacionales, comerciales).
- Reporte total del uso del sistema de mensajería con detalle de ganancias y pérdidas en las fechas establecidas.

Manipulación de procesos utilizados en el sistema (daemons)

- Consulta de procesos registrados.
- Consulta de la configuración de los procesos.
- Edición de la configuración de los procesos.
- Consulta del estado del proceso.
- Arranque del proceso.
- Parada del proceso.

Información del Socket_Server

- Estado del Socket_Server, puerto utilizado y puerto disponible para la conexión de la interfase.

Conexión al Socket_Server

- Comunicación al Socket_Server por medio de sockets definidos.
- Reconexión automática después del timeout.
- Creación y edición de un archivo de configuración con los parámetros de la conexión.

Registro de operaciones realizadas

- Creación dinámica de una bitácora en la máquina local donde se encuentra la interfase con las transacciones realizadas por el usuario.

Administración de Usuario

- Verificación del Usuario de la interfase.
- Edición del nombre y clave del Usuario.
- Log in y Log out de la interfase.

Estos requerimientos fueron los inicialmente definidos y se tomaron como base para el desarrollo del desarrollo de la interfase.

4.2 Diseño del Sistema

4.2.1 Descripción General

El sistema de cobro por mensaje se basa en la utilización de la aplicación Socket_Server. Esta permite realizar las transacciones de cuenta de mensajes enviados, descuento del saldo del cliente, administración de los diferentes tipos de componentes como clientes e ingreso como modificación de la información en la base de datos. De esta manera se asegura un control rígido y eficaz del servicio de mensajería provisto por las compañías de telefonía móvil. El Socket_Server está compuesto por varias clases que se entrelazan para conformar la aplicación. Cada una de estas clases esta diseñada para realizar las peticiones requeridas. Uno de los componentes a realizarse es la Interfase del administrador, que se conecta al Socket_Server por medio de puertos específicos y envía respuestas para obtener información y procesar para un control de las funciones del sistema de cobro por mensajes. Esta interfase maneja toda la

información de las base de datos por lo que es destinada solo para el uso del administrador de la plataforma. Sin embargo, para la utilización de la Interfase también es necesario modificar el Socket_Server y sus componentes mediante una revisión de sus clases principales.

4.2.2 Diagrama de Casos de Uso

El siguiente diagrama de casos de uso simboliza la relación entre los distintos sistemas y usuarios involucrados en la administración de la aplicación del cobro de mensajes. Además muestra los eventos que simbolizan las funcionalidades de la interfase y como estos se dan entrelazados con los actores de la interfase de administración.

Para un entendimiento claro, se ha dividido cada evento como un proceso distinto que involucra un grupo de actores y un conjunto de flujos de funcionamiento del evento.

A continuación se presenta el diagrama de casos de uso y la descripción de cada uno de sus eventos. (Figura 4.1).

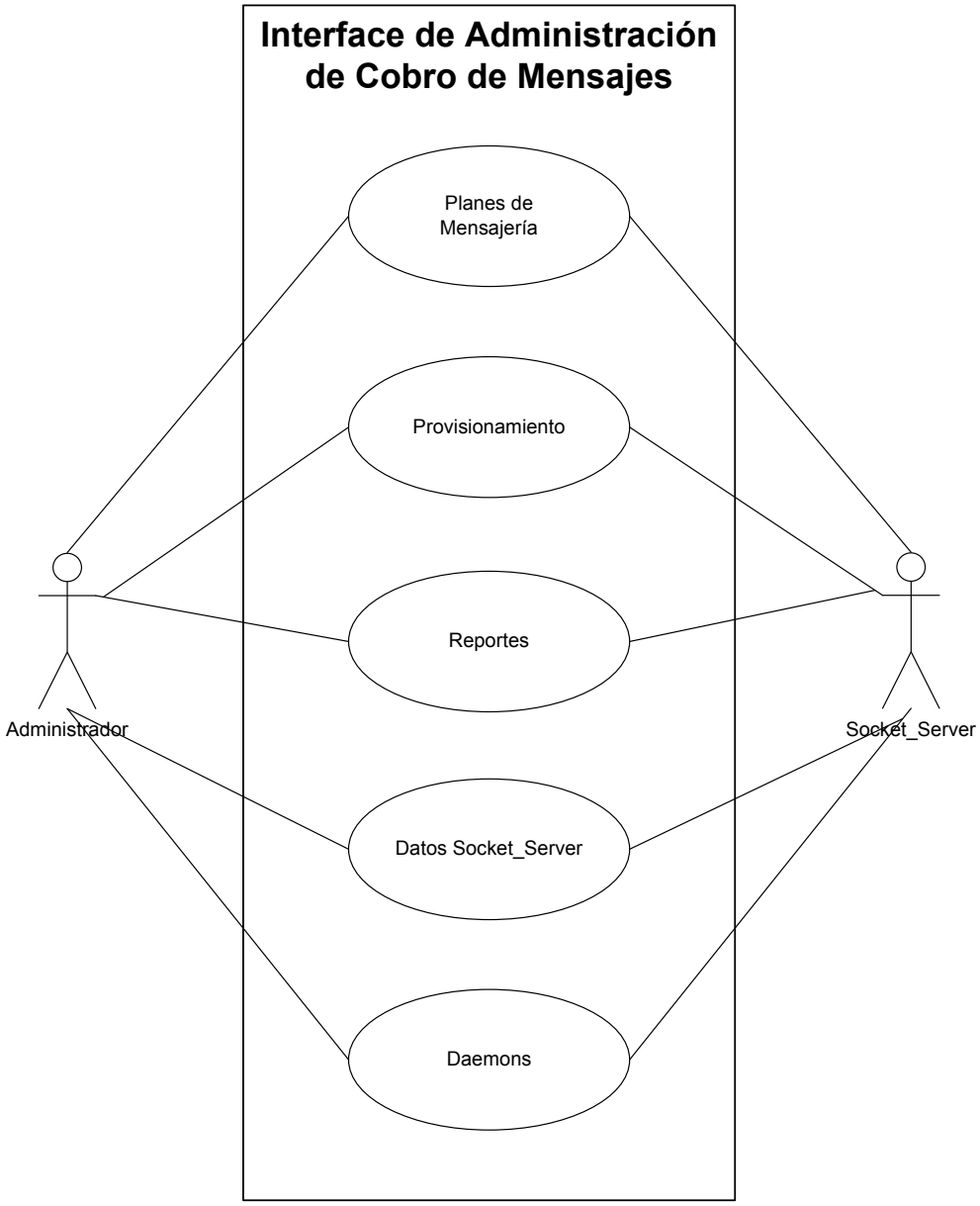


Diagrama de Casos de Uso

Figura 4.1

Casos de Uso	**Eventos de Plan de Mensajería, Provisionamiento y Reportes**
Actores	Administrador: realiza la petición especifica de creación, consulta, edición o borrado de un componente específico. Socket_Server: procesa las solicitudes y realiza los procesos establecidos por el mismo. Base_Datos: ingresa nuevos registros o devuelve consultas.
Tipo	Generalización
Propósito	Creación, Edición y Búsqueda de Planes de Mensajería, Provisionamiento y Consulta de Reportes.
Resumen	Proceso de manipulación y consulta de la información de los planes de Mensajería y clientes del sistema.
Precondiciones	Administrador registrado y conexión con el Socket_Server.
Flujo Principal	1 Se realiza una nueva petición (creación, consulta, edición o borrado). 2 La petición es verificada para eliminar errores por parte del administrador. 3 La aplicación genera un archivo XML que envía al servidor. 4 Se verifica que el XML este correcto y bien estructurado. 5 El Socket_Server transforma el XML y lo analiza para comenzar con el proceso. 6 La base de datos es accesada y se realiza los procedimientos establecidos. 7 Se retorna el resultado a el Socket_Server 8 El resultado es verificado y se crea una respuesta en XML. 9 La interfase recibe la respuesta, verifica si los datos son

	correctos, los analiza y crea una respuesta en forma gráfica. 10 Termina el proceso.
Excepciones	1 Se realiza una nueva petición (creación, consulta, edición o borrado). 2 La petición es verificada para eliminar errores por parte del administrador. 3 La aplicación genera un archivo XML que envía al servidor. 4 Se verifica el XML el cual no es correcto 5 El Socket_Server crea una trama XML de error y envía. 6 La interfase recepta la trama, la evalúa y reporta el error. 7 Termina el proceso.

Casos de Uso	**Descripción del evento Datos Socket_Server**
Actores	Administrador: realiza la petición especifica de creación, consulta, edición o borrado de un componente específico. Socket_Server: procesa las solicitudes y realiza los procesos establecidos por el mismo.
Tipo	Flujo Básico
Propósito	Consulta del estado de los sockets del Socket_Server
Resumen	Consulta del estado de los sockets disponibles para el uso de la interfase.

Precondiciones	Administrador registrado y conexión con el Socket_Server.
Flujo Principal	1 Se realiza la petición por parte del administrador. 2 La aplicación genera un XML que envía al servidor. 3 Se verifica que el XML este correcto y bien estructurado. 4 El Socket_Server se encarga transforma el XML y lo analiza para comenzar con el proceso. 5 El resultado es verificado y se crea una respuesta en XML. 6 La interfase recibe la respuesta, verifica si los datos son correctos, los analiza y crea una respuesta en forma gráfica. 7 Termina el proceso.
Excepciones	1 Se realiza la petición por parte del administrador. 2 La aplicación genera un XML que envía al servidor. 3 Se verifica el XML el cual no es correcto 4 El Socket_Server crea una trama XML de error y envía. 5 La interfase recepta la trama, la evalúa y reporta el error. 6 Termina el proceso.

Casos de uso	**Descripción del evento Daemons**
Actores	Administrador: realiza la petición especifica de creación, consulta, edición o borrado de un componente específico. Socket_Server: procesa las solicitudes y realiza los procesos establecidos por el mismo.

	Daemon: son procesos que tiene un archivo de configuración, pueden ser levantados o bajados.
Tipo	Flujo Básico
Propósito	Administración y manipulación de los daemons registrados en el Socket_Server
Resumen	Consulta de daemons, manipulación del archivo de configuración y iniciado y bajado de los daemons.
Precondiciones	Administrador registrado y conexión con el Socket_Server.
Flujos Principales	**P1: Consulta de daemons y archivo de configuración** 1 El administrador realiza la solicitud para consulta de los daemons registrados. 2 La interfaz genera el XML correspondiente y lo envía 3 El Socket_Server recibe la trama, verifica por errores y la transforma para analizar la recuesta. 4 Se analiza el archivo de configuración del Socket_Server donde se encuentran los daemons registrados. 5 La respuesta es creada con una lista de daemons. 6 Se envía el resultado en una trama XML. 7 La interfase recibe la trama, y presenta la información. 8 El Usuario escoge el daemon y realiza la petición para ver el archivo de configuración. 9 La interfase crea la trama XML con el daemon escogido y envía al Socket_Server. 10 El Socket_Server recibe la trama, verifica por errores y la transforma para analizar la recuesta.

	11 Se ingresa al archivo de configuración del daemon específico y se modifica su configuración. 12 La operación es exitosa, el Socket_Server envía una notificación de éxito. 13 La aplicación recibe la notificación y reporta al usuario. 14 Termina el proceso. **P2: Iniciado y bajado de los daemons** 1 El administrador realiza la solicitud para consulta de los daemons y su estado. 2 La interfase genera el XML correspondiente y lo envía 3 El Socket_Server recibe la trama, verifica por errores y la transforma para analizar la recuesta. 4 Se analiza el archivo de configuración del Socket_Server donde se encuentran los daemons registrados. 5 La respuesta es creada con una lista de daemons. 6 Se envía el resultado en una trama XML. 7 La interfaz recibe la trama, y presenta la información. 8 El Usuario escoge el daemon y opta por iniciarlo o bajarlo dependiendo de su estado 9 La interfase crea la trama XML con el daemon escogido y envía al Socket_Server. 10 Socket_Server recibe la trama, verifica por errores y la transforma para analizar la recuesta. 11 Se arranca o baja al daemon. 12 La operación es exitosa, el Socket_Server envía una

	notificación de éxito. 13 La aplicación recibe la notificación y reporta al usuario. 14 Termina el proceso.
Excepciones	**E1: Consulta de daemons y archivo de configuración** **E1.1:** 1 El administrador realiza la solicitud para consulta de los daemons registrados. 2 La interfaz genera el XML correspondiente y lo envía 3 Se verifica el XML el cual no es correcto 4 El Socket_Server crea una trama XML de error y envía. 5 La interfase recepta la trama, la evalúa y reporta el error. 6 Termina el proceso. **E1.2:** 1 El administrador realiza la solicitud para consulta de los daemons registrados. 2 La interfaz genera el XML correspondiente y lo envía 3 El Socket_Server recibe la trama, verifica por errores y la transforma para analizar la recuesta. 4 Se analiza el archivo de configuración del Socket_Server donde se encuentran los daemons registrados. 5 La respuesta es creada con una lista de daemons. 6 Se envía el resultado en una trama XML. 7 La interfase recibe la trama, y presenta la información. 8 9 El Usuario escoge el daemon y realiza la petición para ver el

	archivo de configuración. 10 La interfase crea la trama XML con el daemon escogido y envía al Socket_Server. 11 El Socket_Server recibe la trama, verifica por errores y la transforma para analizar la recuesta. 12 Se ingresa al archivo de configuración del daemon específico y se modifica su configuración. 13 La operación de edición no es exitosa, el Socket_Server envía un error. 14 La aplicación recibe la notificación y reporta al usuario. 15 Termina el proceso. **E2: Iniciado y bajado de los daemons** **E2.1:** 1 El administrador realiza la solicitud para consulta de los daemons y su estado. 2 La interfase genera el XML correspondiente y lo envía 3 Se verifica el XML el cual no es correcto. 4 El Socket_Server crea una trama XML de error y envía. 5 La interfase recepta la trama, la evalúa y reporta el error. 6 Termina el proceso. **E2.2:** 1 El administrador realiza la solicitud para consulta de los daemons y su estado. 2 La interfase genera el XML correspondiente y lo envía. 3 El Socket_Server recibe la trama, verifica por errores y la

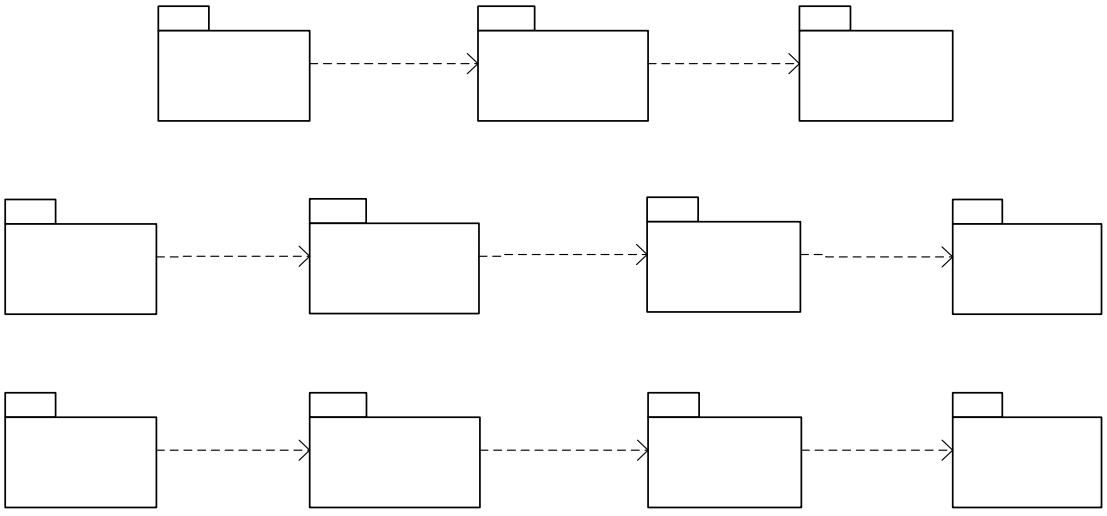
		transforma para analizar la recuesta.
	4	Se analiza el archivo de configuración del Socket_Server donde se encuentran los daemons registrados.
	5	La respuesta es creada con una lista de daemons.
	6	Se envía el resultado en una trama XML.
	7	La interfaz recibe la trama, y presenta la información.
	8	El Usuario escoge el daemon y opta por iniciarlo o bajarlo dependiendo de su estado.
	9	La interfase crea la trama XML con el daemon escogido y envía al Socket_Server...
	10	Socket_Server recibe la trama, verifica por errores y la transforma para analizar la recuesta.
	11	Se arranca o baja al daemon.
	12	La operación de edición no es exitosa, el Socket_Server envía un error.
	13	La aplicación recibe la notificación y reporta al usuario.
	14	Termina el proceso.

4.2.3 Modelo de Diseño

En el modelo de diseño se especifica cada uno de los componentes del sistema y estos en conjunto con las funcionalidades de la interfase.

4.2.3.1 Diagrama de Componentes

El diagrama de componentes especifica cada una de las partes que intervienen en el funcionamiento de la interfase de administración para realizar las operaciones necesarias. Para dichas operaciones se requiere de la participación de 4 componentes que interactúan de forma independiente para cada acción. En el primer caso se tiene la interacción de información del Socket_Server en el que se encuentran involucrados el Administrador, la interfase y el Socket_Server. El segundo caso de operaciones depende del acceso a la información del usuario y la empresa, por lo que se involucra al Administrador, interfase, Socket_Server y la base de datos. Finalmente, el tercer caso se compone en la manipulación de los daemons por lo que los componentes involucrados son el Administrador, interfase, Socket_Sever y los daemons. A continuación se observa en la figura 4.2 los tres casos de componentes descritos y el flujo de comunicación.



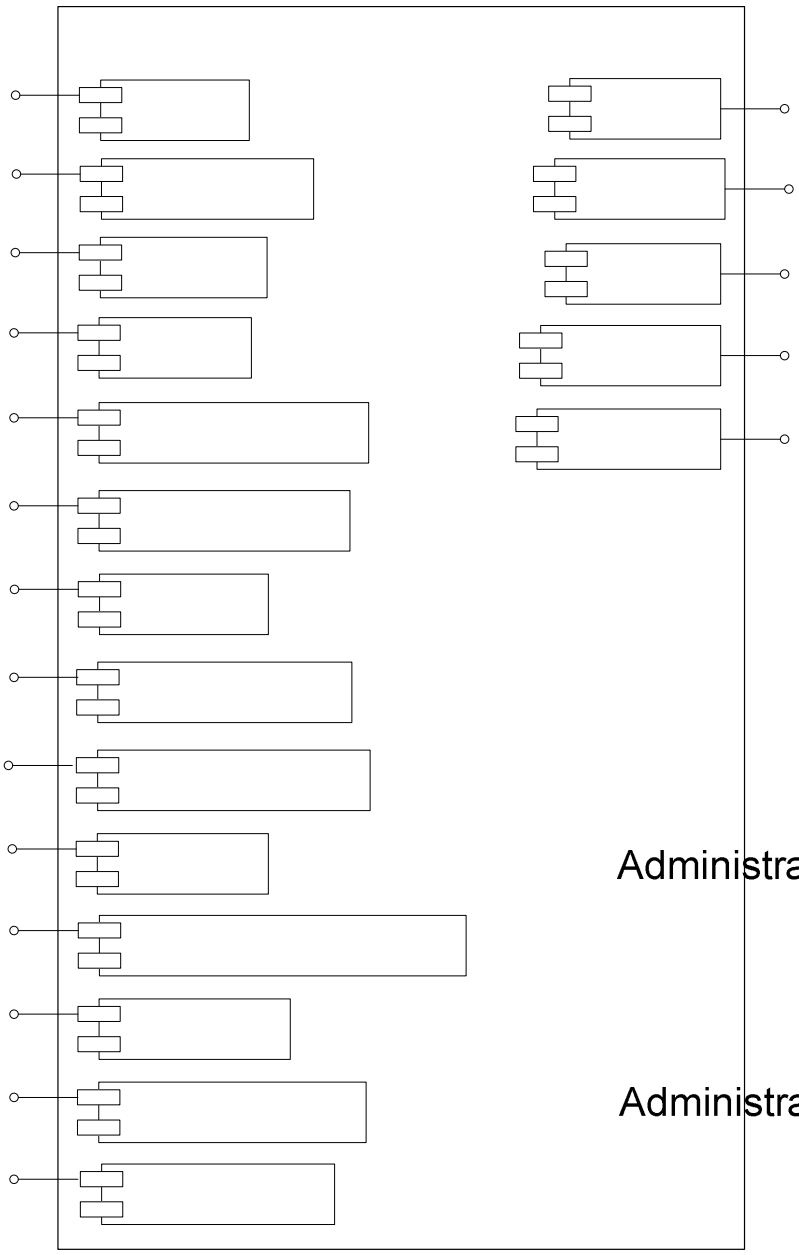
Componentes

Figura 4.2

4.2.3.2 Interfase de Administración de Cobro de Mensajes

La segunda parte del modelo de diseño es la especificación de la interfase de Administración de Cobro de Mensajes y sus actores inmediatos.

La interfase de Administración de Cobro de Mensajes se compone de varias funcionalidades que requieren de diferentes recursos (actores) para su funcionamiento. En la siguiente figura 4.3 se muestra un diagrama de la interfase con sus funcionalidades y los actores directos que intervienen.



Funcionalidades y Actores de la interfase

Figura 4.3

Administrador

Las funcionalidades especificadas en la figura 4.3 representan cada una de las partes que forman la interfase y se caracterizan por realizar una función específica.

a. Creación de Plan: Esta es una ventana vista por el administrador donde se ingresan los datos necesarios para la creación de un nuevo plan de mensajería. Una vez obtenida la información, se genera el XML respectivo y se pasa al Gestionador de XML.
b. Consulta y edición de Plan: En una sola ventana se ha creado la posibilidad de buscar un plan específico ingresando cierta información sobre el plan (ver Anexo II) y como resultado se tiene todos los datos referentes al plan. Además una vez obtenida la información de respuesta se puede editar los campos con excepción del identificador del plan y enviar los nuevos cambios al Socket_Server. Para esto se utiliza el Gestionador de XML
c. Eliminación del Plan: En una ventana se ingresa la información básica sobre el Plan y se envía por medio del Gestionador de XML para solicitar su eliminación.
d. Provisionador: Esta es una ventana donde se ingresa un plan existente (datos que lo identifican) y los datos sobre el cliente para provisionarlo del servicio de mensajería.
e. Consulta y Edición del Provisionador: Con el número de celular y el Plan de mensajería al que corresponde, se ingresa dicha información en una nueva ventana para consultar los datos del cliente y modificar los mismos si se requiere.

f. Eliminación de Provisionamiento: Para eliminar el provisionamiento se ingresa en la ventana de eliminación los datos del celular y el Plan al que corresponde y la interfase genera el XML respectivo.

g. Reprovisionamiento: Esta ventana sirve para reactivar a los clientes que fueron desconectados del servicio por falta de saldo. Para realizar esta operación se requiere del número de celular respectivo.

h. Reporte de Volumen de Mensajes: Este reporte toma como datos iniciales un rango de fechas y retorna el detalle de los mensajes totales de todos los clientes que se enviaron durante el rango establecido de fechas.

i. Reporte de Volumen de Mensajes por Min: Toma un rango de fechas y un número de celular para retornar un reporte detallado de los mensajes enviados de ese celular en el rango establecido de fechas.

j. Control de Daemons: Es un conjunto de funcionalidades que permite observar los daemons de la aplicación de cobro por mensaje, su tiempo de operación si están en funcionamiento y subirlos o bajarlos.

k. Edición del archivo de configuración de los daemons: El Administrador tiene acceso a una ventana con una lista de todos los daemons y puede escoger uno de ellos para editar su archivo de configuración.

l. Servidor Socket_Server: En esta consulta, se observa características básicas del Socket_Server con respecto a la interfase, el puerto utilizado y el puerto disponible.

m. Edición de los Datos del Administrador: Esto es un conjunto de ventanas encargadas de mostrar los datos del administrador y editar los mismo, como su nombre de usuario y clave de acceso

n. Edición de los Datos de Conexión: Es una ventana que define permite al administrador definir el puerto y dirección IP del Socket_Server para realizar la conexión.

o. Gestionador XML: Este elemento de la interfase se encarga de recibir los XML de las funcionalidades ‘a’ a la ‘l’ para enviarlo al Socket_Server y espera por una respuesta que se enviará a la funcionalidad que realizó la petición.

p. Gestionador errores (XML): Cuando existen errores ya sea en la trama de respuesta o el Socket_Server retorna un error, el gestionador se encarga de traducir el error y notificar al Administrador.

q. Gestionador Logs: Cada acción que se realiza en la Interfase de Administración es guardada en un archivo de texto local (bitácora), con la fecha, hora y nombre del Administrador que realizó.

r. Gestionador Usuarios: Esta funcionalidad se relaciona con la de Datos del Administrador y se encarga de guardar, obtener y modificar la información del Usuario.

s. Gestionador de Conexión: Esta funcionalidad se comunica con la de Datos de Conexión para guardar, obtener y modificar la información referente a la conexión con el Socket_Server.

Una vez definida las funcionalidades de la Interfase de Administración, se define los actores de la misma.

a. Socket_Server

Es la aplicación ubicada en el servidor que se encarga de realizar el cobro por mensajes. Recepta mensajes XML y realiza consultas en la base de datos.

b. Archivo de Bitácora

Está ubicado en la máquina donde corre la interfase y es un archivo de texto que contiene todas las transacciones realizadas por el Administrador.

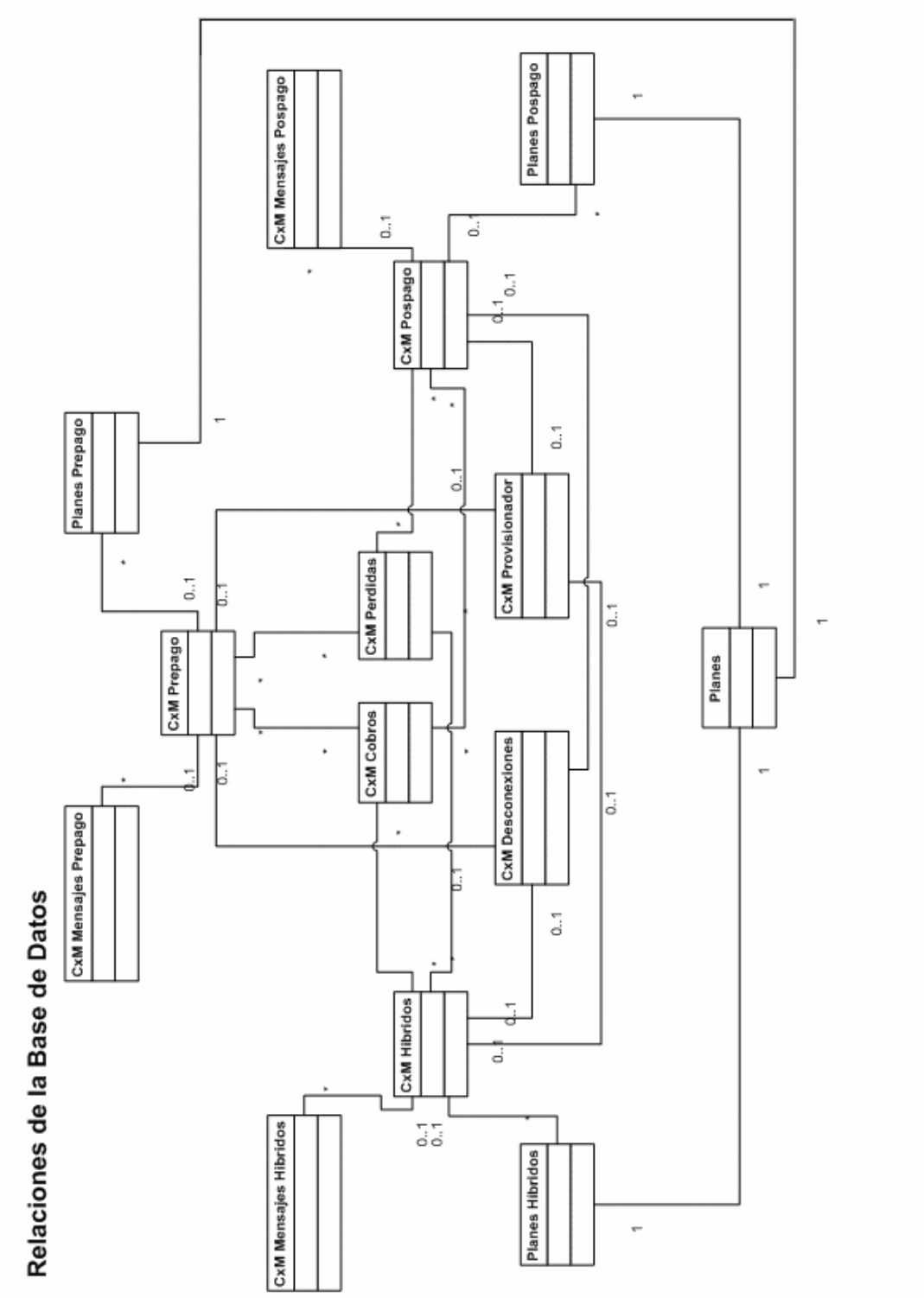
c. Archivo de Configuración

Es un archivo .conf localizado en la maquina local y guarda los parámetros de la conexión.

d. Base de datos

La base de datos está compuesta por numerosas tablas utilizadas por el Socket_Server y otras aplicaciones para realizar el cobro por mensajes. Estas tablas están divididas de acuerdo a los diferentes tipos de usuarios: Prepago, Híbridos y Pospago.

A continuación se describe las tablas con sus respectivas relaciones entre ellas en la figura 4.4 y el detalle de las principales tablas se puede observar en el Apéndice II.



Relaciones de la Base de Datos

Figura 4.4

4.2.4 Definición del Protocolo de Comunicación

Para la interacción de la interfase con el Socket_Server es necesario definir un protocolo de comunicación que permita que los dos sistemas se entiendan. Este protocolo se describe con detalle en el Anexo II.

4.2.5 Diseño de la Interfase

La interfase gráfica realizada consta de varias ventanas y vistas para consultas y comandos que se necesitan hacer. Estas ventanas son agrupadas en un menú en común de acuerdo al objeto al que se refieran. (Figura 4.5).

Ventana de la Interfase

Figura 4.5

La vista inicial consta de una ventana con menú en el cual se especifica cada una de las operaciones. El primer objeto en el menú es Archivo, que contiene los submenús de conexión, desconexión, cambio de conexión, administración de usuarios con las opciones de cambio de nombre y clave del usuario, y la función de salida.

El siguiente menú es de operaciones, aquí se encuentran todas las opciones para la administración de componentes, consulta, creación, modificación y eliminación de los mismos.

A continuación está el menú de provisionamiento el cual tiene las opciones de provisionamiento a los usuarios con un componente. Aquí se provisionará, consultará, desprovisionará y reactivará a los usuarios con el servicio de mensajería.

La cuarta opción del menú principal son los reportes. Dentro de este tenemos al reporte por número individual y al reporte total de todo el sistema de mensajería.

Procesos viene después de reportes y aquí tenemos dos posibilidades observar los procesos del sistema de mensajería que se encuentran corriendo, su tiempo y la posibilidad de subirlos o bajarlos. La segunda posibilidad del submenú Procesos es la consulta del archivo de configuración de los procesos. Se despliega el archivo de configuración en una tabla y se puede modificar, agregar y borrar campos del mismo. Luego se puede enviar al servidor a guardar este nuevo archivo de configuración para el proceso elegido.

El quinto puesto del menú es para el servidor API, que es al cual se conecta la interfase. Al escoger este submenú se debe desplegar una ventana con las estadísticas del servidor, con los puertos utilizados y los desocupados.

El siguiente submenú es el de ayuda, que abre una ventana con los datos de la compañía que desarrollo el software y forma de contacto.

Finalmente tenemos el menú de login/logout que sirve para deshabilitar la aplicación sin cerrarla y sin perder la conexión con el servidor.

En le anexo III se tiene el manual de Usuario donde se presenta cada una de las vistas de la interfase. Aquí se puede observar los menús y submenús descritos.

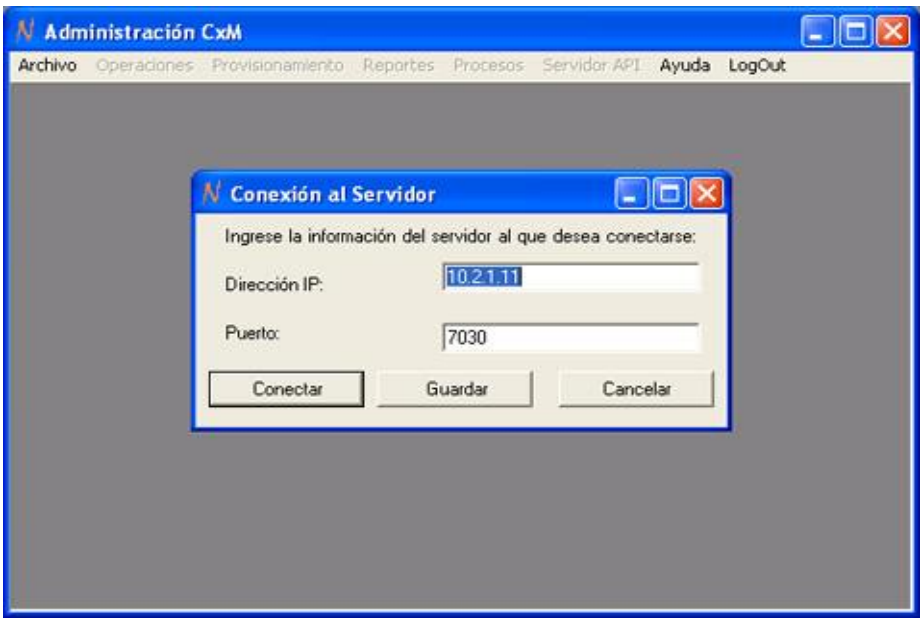
4.2.5.1 Ventanas

Las ventanas de consulta llevan el siguiente formato: en la parte superior se coloca los casilleros con los datos iniciales para la petición, luego vienen los botones de enviar o cancelar y al final de la ventana se coloca los casilleros de texto para los datos de respuesta. Los datos de respuesta van encerados en groupBoxes separando los datos comunes o los datos de respuesta de los de entrada inicial que van en la parte superior. Para la clasificación de la información se dividirá en prepago, pospago híbridos, mensajes wis y mensajes internacionales.

4.2.5.2 Formatos de las formas

Para los componentes visuales se decidió que la aplicación sea sencilla de acceso directo que no entretenga al usuario. El color de contorno de la aplicación es basado en las características definidas por el sistema operativo. Los bordes son simples sin contornos. Las letras de los reportes no llevan sombra, tienen el tamaño predeterminado

y se alinean a la izquierda. Los campos de texto deberán tener una altura de 20 píxeles y la longitud varia de acuerdo a la información. Si existen varios campos de texto se debe tratar de mantener la misma longitud para todos. Los botones tienen una altura de 24 píxeles con longitud variable de acuerdo a la ventana y se alinean al centro. Las etiquetas tienen una altura de 16 píxeles e igual que los anteriores con longitud variables. (Figura 4.6).



Diseño de Ventanas

Figura 4.6

Estas son las normas base en las que se desarrolla la interfase, por la variedad de la información se incorporará otros objetos como tablas pero su formato se definirá con pruebas para encajar la información de la mejor manera.

Para observar los resultados del diseño de la aplicación se puede ver en el Anexo III en el manual de usuario cada una de las vistas de las ventanas.

4.3 Desarrollo del Sistema

4.3.1 Estándares de Programación

La programación entre varias personas puede ser un trabajo complicado y con dificultades de comunicación. Así, como se sigue un formato en el diseño de los documentos, o en las comunicaciones formales, es necesario tener un formato específico en la manera de cómo programar. Estas normas facilitan que el trabajo pueda pasar de una persona a otra en caso que el programador inicial no pueda concluir el trabajo. Las reglas del formato pueden ser vistas con detalle en el Anexo I.

Todas estas normas serán impuestas para garantizar un producto legible y con calidad. Además facilitan el control y corrección en la etapa de mantenimiento.

4.3.2 Ampliación del Sistema

El sistema original desarrollado en base a los requerimientos del cliente presentó carencias de algunas características básicas de seguridad. Primero inicialmente no se consideró el manejo de Usuarios, y se tuvo que incluir un sistema de control de usuarios básico controlado localmente. Esto se realizó debido a las limitaciones de tiempo y presupuesto. El sistema de control de usuarios consta de un archivo de texto generado automáticamente al iniciar la aplicación. Este archivo contiene un nombre de usuario y

clave grabadas en el código, que una vez que el usuario ingresa a la aplicación se es requerido cambiar dicho nombre de usuario y clave. Esta información es encriptada y guardada localmente junto con otra información que produce la aplicación en archivos de texto. Sin embargo, como parte de las recomendaciones se documentó que en la segunda versión de la interfase se controle el usuario y clave desde el Socket_Server garantizando un mejor nivel de seguridad.

Además, se cambió el reporte de los procesos (daemons) que corren en el servidor. El reporte inicial determinaba el tiempo que los daemons se encontraban funcionando, pero debido a que no era posible determinar dicho tiempo por limitaciones en el sistema operativo se cambió el reporte a colocar el tiempo en minutos u horas de funcionamiento y una vez que pasen un día se registra como un proceso corriendo mas de un día.

4.4 Pruebas y Respuestas del Sistema

Para la realización de pruebas de la interfase de Administración se utilizó un servidor diferente al de producción, incluyendo una copia del Socket_Server con sus debidas modificaciones y extensiones de funcionalidades. Además, se realizaron copias de las base de datos de tal manera que la información real no sea modificada.

Una vez preparado el ambiente de pruebas se realizaron dos tipos de análisis del funcionamiento del sistema. El primero fue la pruebas de funcionalidad que se utilizó para verificar que cada componente realice las operaciones requeridas; y la segunda

pruebas es la de desempeño que determina el tiempo de respuesta para las operaciones que realiza la interfase.

4.4.1 Prueba de Funcionalidad

La prueba de funcionalidad consta en verificar cada uno de los componentes de la interfase. Se realiza accediendo individualmente a los menús de le interfase y sus submenús para observar que las opciones funcionen correctamente.

A continuación en la tabla 4.2 se presenta un cuadro con cada una de las pruebas de funcionalidad realizadas a la interfase y su correspondiente resultado

Funcionalidad a Probarse	Respuesta	Comentarios
Instalación		
Wizard de Instalación de la interfase	Correcta	La interfase permite escoger la carpeta destino donde se instalará
Control de Usuarios		
Control inicial de Usuarios	Correcta	Al iniciarse la interfase pide un usuario y contraseña para poder proceder
Bloque automático	Correcta	Si el usuario no acerta al cuarto intento con el usuario y contraseña la interfase se cierra
Cambio de nombre de usuario	Correcta	Utilizando las opciones del menú archivo, se puede cambiar en nombre de usuario.

Funcionalidad a Probarse	Respuesta	Comentarios
Cambio de contraseña	Correcta	Utilizando las opciones del menú archivo, se puede cambiar en contraseña de usuario.
Logout	Correcta	El menú logout permite bloquear la interfase hasta que el administrador ingrese el nombre de usuario y contraseña.
Conexión al Socket_Server		
Cambiar Conexión al Socket_Server	Correcta	En el menú archivo, la opción conexión permite establecer los parámetros de conexión y conectarse al Socket_Server
Conectar	Correcta	Si los parámetros de conexión ya fueron establecidos anteriormente, esta opción permite realizar una conexión directa
Reconexión automática	Correcta	Si el tiempo de conexión en el Socket_Server se a caducado y se desconecto, la interfase se reconecta automáticamente para seguir con su operación.
Desconexión	Correcta	Con la opción desconectar se desconecta la interfase del Socket_Server

Funcionalidad a Probarse	Respuesta	Comentarios
Manipulación de Planes(Componentes)		
Crear componente	Correcta	Se puede crear un nuevo componente
Eliminar componente	Correcta	Se puede eliminar componentes individualmente
Buscar componente	Correcta	Se puede buscar cualquier componente activo.
Editar componente	Correcta	Se puede editar los componentes individualmente
Provisionamiento		
Crear provisionamiento	Correcta	Se puede provisionar un Celular con un componente
Consultar provisionamiento	Correcta	Se puede consultar un celular provisionado
Eliminar provisionamiento	Correcta	Se puede eliminar el provisionamiento a un celular.
Reactivación	Correcta	Se puede reactivar a un celular desactivado.
Reportes		
Número de mensajes por Min	Correcta	Se genera correctamente un reporte de un celular específico en un rango de fechas.
Volumen de mensajes	Correcta	Se genera un reporte del sistema de mensajería en un rango de fechas

Funcionalidad a Probarse	Respuesta	Comentarios
Volumen de mensajes	Correcta	Se genera un reporte del sistema de mensajería en un rango de fechas
Procesos		
Consulta de estado	Correcta	Se tiene un listado de los procesos y se puede activarlos o desactivarlos
Consulta de configuración	Correcta	Se tiene una lista de los procesos y se puede escoger a cada uno individualmente para modificar su configuración
Servidor API	Correcta	Se tiene un reporte del puerto utilizado y si hay un puerto disponible del Socket_Server.
Ayuda	Correcta	Se tiene una ventana con la información sobre la empresa que realizó la interfase.
Archivo de Bitácora	Correcta	En la carpeta donde se instaló la aplicación se genera un archivo .log con las operaciones realizadas.
Respuesta de error	Correcta	Si el Socket_Server retorna un error a una petición la interfase notifica del mismo.

Prueba de Funcionalidad

Tabla 4.2

Este es el resultado de la última prueba de funcionalidad realizada. Anterior a esta se realizaron similares donde se fue detectando errores en la interfase y además nuevos requerimientos y limitaciones. Dichos requerimientos se describen en la primera parte de este capítulo como requerimientos recomendados. Entre estos tenemos la aplicación de un control de usuarios y vistas de todos los planes o paquetes de mensajería disponibles.

4.4.2 Prueba de Desempeño

El desempeño del sistema es un punto importante considerado en las pruebas, ya que la cantidad de información es extensa, las consultas a la base de datos pueden tomar largos períodos de tiempo dependiendo de lo que se desee obtener y uno de los objetivos de la interfase es agilitar y reducir el tiempo de trabajo del Administrador.

A continuación se muestra una tabla con la operación realizada y los tiempos de ejecución. Este tiempo incluye tambien el tiempo de procesamiento de la información.

	Operación	Tiempo de Respuesta
1	Creación de un nuevo componente	Instantáneo
2	Eliminación del componente	Instantáneo
3	Edición del componente	Instantáneo
4	Búsqueda del componente	Instantáneo
5	Provisionamiento al usuario con un componente	Instantáneo

	Operación	Tiempo de Respuesta
6	Desprovisionamiento al usuario	Instantáneo
7	Edición de provisionamiento	Instantáneo
8	Búsqueda del usuario con el respectivo componente	Instantáneo
9	Reactivación del Usuario con el servicio	Instantáneo
10	Reporte por número con detalle de los mensajes enviados y a que categoría pertenecen	De 10 segundos a 10 minutos
11	Reporte total del uso del sistema de mensajería con detalle de ganancias y pérdidas en las fechas establecidas	De 10 segundos a 30 minutos
12	Consulta de procesos registrados	Instantáneo
13	Consulta de la configuración de los procesos	Instantáneo
14	Edición de la configuración de los procesos	Instantáneo
15	Consulta del estado del proceso	Instantáneo
16	Arranque del proceso	Instantáneo
17	Parada del proceso	Instantáneo
18	Estado del servidor API, puerto utilizado y puerto disponible	Instantáneo
19	Comunicación al Socket_Server por medio de sockets definidos	Instantáneo
20	Reconexión automática después del timeout	Instantáneo

	Operación	Tiempo de Respuesta
21	Creación y edición de un archivo de configuración con los parámetros de la conexión	Instantáneo
22	Creación dinámica de una bitácora con las transacciones realizadas por el usuario	Instantáneo

Prueba de Desempeño

Tabla 4.3

Como se puede observar en la tabla 4.3 la mayoría de las operaciones se realizan en una forma casi instantánea o sea en un tiempo menor a 5 segundos, con la excepción del reporte por número con detalle de mensajes enviados y el reporte de Volumen de mensajes que puede tomar hasta 30 minutos. Esta limitación se encuentra fuera del alcance de la interfase ya que el tiempo de la consulta se da en la base de datos y no en el procesamiento de la información. Sin embargo, el administrador ya estaba al tanto de este suceso por lo que se ha considerado como un inconveniente aceptable.

4.4.3 Pruebas de Administración de Procesos.

Como se mencionó en los objetivos de la interfase, se pretende limitar el acceso al servidor que contiene al Socket_Server; para lo cual es necesario poder bajar y levantar los procesos relacionados con el mismo. Con la interfase de administración se posee la funcionalidad de modificar los archivos de configuración de los daemons, levantarlos, bajarlos y ver su tiempo de operación.

El proceso de modificación del archivo de configuración se realiza de forma instantánea, pudiendo agregar campos y valores al archivo, como modificar los existentes o borrarlos.

Similarmente, la activación y desactivación de los daemons es instantáneo y podemos observar si los mismo se encuentran funcionando o no. Esto ahorra el tiempo de hacer telnet al servidor, ingresar los comandos al sistema operativo y ver daemon por daemon si están funcionando. Con esta funcionalidad de administración de los daemons se estima reducir el tiempo de administración de daemons a 1/(número de daemons) ya que se puede revisar todos los daemons al mismo tiempo.

4.4.4 Confidencialidad del Servidor

Una vez implementada la interfase se restringió el acceso al servidor, en especial a la base de datos. Sin embargo al poseer la interfase de Administración estas limitaciones no han presentado un problema ya que la mayoría de las funciones diarias pueden hacerse por medio de la misma. Adicionalmente, se producen menos errores al ingresar información a la base de datos ya que se verifica la información y cumple un formato establecido en la interfase.

Después de estas pruebas de uso, se verificó el éxito de la interfase, de igual manera se añadieron nuevos requerimientos para una futura remodelación de la interfase para ofrecer nuevas funcionalidades y en el futuro contar cada vez con una herramienta más sólida.

Capitulo V Conclusiones y Recomendaciones

5.1 Conclusiones:

1.- Una vez implementada la interfase, se ha limitado el acceso al servidor donde reside el Socket_Server y al servidor que contiene la base de datos. En consecuencia se reduce el riesgo de errores humanos en la manipulación de información y agilita el proceso de administración del Socket_Server al poder verificar su funcionamiento observando el estado de los procesos daemons y la actualización en la base de datos por medio de los reportes de la interfase.

Adicionalmente, con el acceso restringido a los servidores, estos son menos propensos a ser accedidos por personal no autorizado y sufrir ataques o pérdidas de información por dichos eventos.

2.- Los reportes que ofrece la interfase de Administración entre los que tenemos Número de mensajes por número de celular, Volumen de mensajes, búsqueda de provisionamiento y búsqueda de componentes agilitan el proceso de consulta al solicitar solo los datos iniciales y evitan la necesidad de construir sentencias SQL para obtener la información.

3.- Las nuevas funcionalidades que incluyen creación de componentes (planes tarifarios), edición de componentes, eliminación de componentes, provisionamiento, desprovisionamiento y reactivación de servicio permite un control sobre el Socket_Server para la corrección de errores y agilita el mismo proceso.

4.- El control de los procesos daemons por medio de la interfase permite agilitar la administración de procesos sin la necesidad de ingresar directamente al servidor para ejecutar comandos del sistema operativo. Con la interfase es posible levantar o bajar los procesos como editar su archivo de configuración.

5.- El servicio del administrador con el cliente final presenta una mejora al disminuir el tiempo de respuesta en un 40% basado en el tiempo que se tomaba anteriormente para verificar el provisionamiento del servicio y provisionarlo o reactivarlo.

6.- Los tiempos de respuesta en la mayoría de las peticiones fueron exitosos. Las actividades del Administrador como por ejemplo la manipulación de planes o activación y reactivación de usuarios se realizaron de forma inmediata sin la necesidad de verificar en la base de datos o ingresar al servidor de la misma, por lo que permitió al mismo realizar su trabajo efectivamente y en menor tiempo.

7.- Adicionalmente, se abrió un nuevo camino en el desarrollo de la interfase gráfica donde se reestablecerán nuevos cambios y funcionalidades. En consecuencia, luego de cada etapa de mantenimiento podrá ser un sistema más estable y necesario para la administración de uno de los servicios más importantes, el sistema de cobro por mensaje.

5.2 Recomendaciones:

1.- El acceso a los servidores de base de datos y Socket_Server debe ser restringido a un menor número de personas y como solución al acceso a la información se ofrece la distribución de la interfase al personal autorizado que la requiera.

2.- La generación de reportes de la interfase puede ser utilizada para evaluar el funcionamiento del Socket_Server, comportamiento de los clientes y plantear planes estratégicos para la creación de nuevos proyectos que incrementarán la productividad del sistema de Mensajería.

3.- Las nuevas funcionalidades de creación de componentes y provisionamientos pueden presentar una ayuda no sólo al administrador si no a otros usuarios internos de la institución por lo que se debe analizar la posibilidad de distribuir la interfase en otras áreas como en servicio al cliente.

4.- Para la siguiente etapa de mantenimiento se recomienda la inclusión de threads que ayudará en el procesamiento de reportes. Esta acción no fue realizada anteriormente ya que fue propuesta en la etapa de pruebas e involucraba un retraso en la entrega de la interfase.

5.- La última consideración a realizarse es mejorar la seguridad de la aplicación y ampliar el sistema de control de usuarios. Para realizar esto se requiere verificar los usuarios en el servidor y en la interfase.

Bibliografía

Albahari, Ben. "A comparative overview of C#". http://genamics.com/developer/csharp_comparative.htm. (20 jun. 2005).

Chapman, James R. "Software Development Methology". www.hyperhot.com/pm_sdm.htm. (15 jun. 2005).

Garcia, María I. "Curso XML Introducción". http://geneura.ugr.es/~maribel/xml/introduccion/index.shtml#1. (3 jun. 2006).

"History of cell phones". http://library.thinkquest.org/04oct/00047/historycell.htm. (15 jun. 2005).

Hutchison, Ryan. "How does SMS works". http://classes.design.ucla.edu/Spring05/152BC/projects/hutchison/researchFinal/smsWorks.html. (25 nov. 2005).

Manzano, Christian. Entrevista personal. 8 junio 2005.

Marples, Gareth. "The history of cell phones – a vision realized".

http://technology.preferredconsumer.com/cellular_phone/history_of_cell_phone.html. (15 jun. 2005).

Mössenböck, H. "Introduction to C#".www.ssw.uni-linz.ac.at/Teaching/Lectures/CSharp/Tutorial/Part1.pdf. University of Lynks, Austria. (20 jun. 2005).

Mössenböck, H. "Advanced C#".www.ssw.uni-linz.ac.at/Teaching/Lectures/CSharp/Tutorial/Part2.pdf. University of Lynks, Austria. (20 jun. 2005).

"Nobel Lectures, Physics", "Marconi", Elsevier Publishing Company, Ámsterdam. 1967.

"SMS history". www.funsms.net/sms_history.htm. (15 jun. 2005).

Zammit, Johan. "How do we Build Correct Systems?". www.cis.um.edu.mt/%7Ejzam/building.html. (15 jun. 2005).

Anexo I Reglas para la programación

Para mantener un código ordenado y de fácil mantenimiento es necesario definir estándares de desarrollo que se describen a continuación:

a.- Primeramente aislaremos los procesos generales de los específicos. Los procesos generales son los utilizados por varias clases y se repiten en varias partes de la aplicación. Estas funciones serán agrupadas en una clase *traductor.cs*. De esta manera si es necesario cambiar el proceso, solo se realiza un cambio y no hay que buscar en cada una de las clases que lo contenga para modificarlo.

Esta estrategia ahorrara tiempo en el desarrollo inicial, como en el desarrollo avanzado evitando la repetición de código.

b.- Debido a que Visual Studio crea clases individuales para cada forma, mantendremos esta modalidad. La clase llevara el nombre de la forma. En la clase de la forma se encontrarán las funciones de los componentes de las mismas como funciones de áreas de texto y/o botones.

c.- Los nombres de las clases inician como letra Mayúscula. En caso que el nombre sea una composición de más de una palabra seguida, cada palabra comenzará con mayúscula para poder dar un mejor sentido al nombre.

d.- Las funciones se definen con minúsculas. Esta regla tiene limitaciones debido a Visual Studio ya que el lenguaje genera automáticamente los nombres de muchas

funciones y en momentos varia su utilización de mayúsculas y minúsculas. Pero para las funciones que sean creadas por el programador deberán cumplir esta regla. Si las funciones se componen de dos o más palabras, la primera comenzará con minúscula y a partir de la segunda empezarán con mayúsculas.

e.- De la misma manera que las funciones, las variables tambien se definen en minúsculas, y si contiene más de una palabra se sigue la misma estrategia. Esta regla es para todas las variables, aún cuando estas sean instancias de otras clases.

f.- Los nombres a utilizarse, tanto para clases, funciones y variables deben ser significativos de lo que representan. Sin embargo, se aceptará nombres simples como "x" para variables de lazos for donde no es necesario explicar su significado.

g.- Los comentarios se colocarán antes de cada método, especificando que realiza el método o función, que variables toma y que resultado se debe esperar. En caso de ser necesario se puede añadir comentarios dentro de los métodos para especificar una funcionalidad o característica específica.

h.- La sangría será respetada. El namespace no lleva sangría, después de esto cada método tendrá su sangría correspondiente para indicar su profundidad dentro del código. Con los elementos de control como if, while, etc. se colocará una sangría adicional a sus contenidos dentro de las llaves que encierran su contenido.

i.- Las elementos de control if, while, for, switch, siempre llevarán llaves para sus contenidos, sin importar si es solo una línea.

j.- El logo de la empresa será colocado en cada una de las formas de la interfase. Es necesario revisar que todas las ventanas de presentación contengan este logo.

k.- En las clases desarrolladas se añadirá en la parte superior el nombre, apellido y forma de contacto como email o teléfono de la persona que desarrollo la clase. Esta norma se utiliza en caso que ocurran cambios de personal y se pueda localizar a los desarrolladores.

Anexo II Manual Técnico

Soluciones de Código Abierto

Movistar
Interfase del Sistema de Cobro de Mensajes
Junio del 2005

Manual de Técnico

Fecha última modificación: **Junio, 20 del 2005**

Investigación y Desarrollo
Novix
Quito, Ecuador

Contenidos

1. Descripción General

La interfase de cobro de mensajes es un sistema de administración diseñado para conectarse remotamente con la aplicación Socket_Server, encargada del cobro automático de mensajes, y facilitar el control de su funcionamiento y flujo de información.

Gracias a esta interfase es posible realizar reportes prediseñados para estudiar los parámetros que verifican el desempeño del sistema de mensajería. Además, es posible ingresar y editar usuarios y planes tarifarios.

A continuación en este manual se detalla la información a ser considerada por el desarrollador de la interfase o personal encargado de su mantenimiento, para resolver posibles errores o problemas que puedan suscitarse, como también para extender las funcionalidades de la interfase.

2. Requerimientos del Sistema

Para la realización del sistema se requirió la utilización del lenguaje de Programación Visual C#. Este lenguaje ya descrito requiere la utilización de Windows 98 o mayor con el último service pack e Internet Information Service. Además de Internet Explorer 5.01 o más.

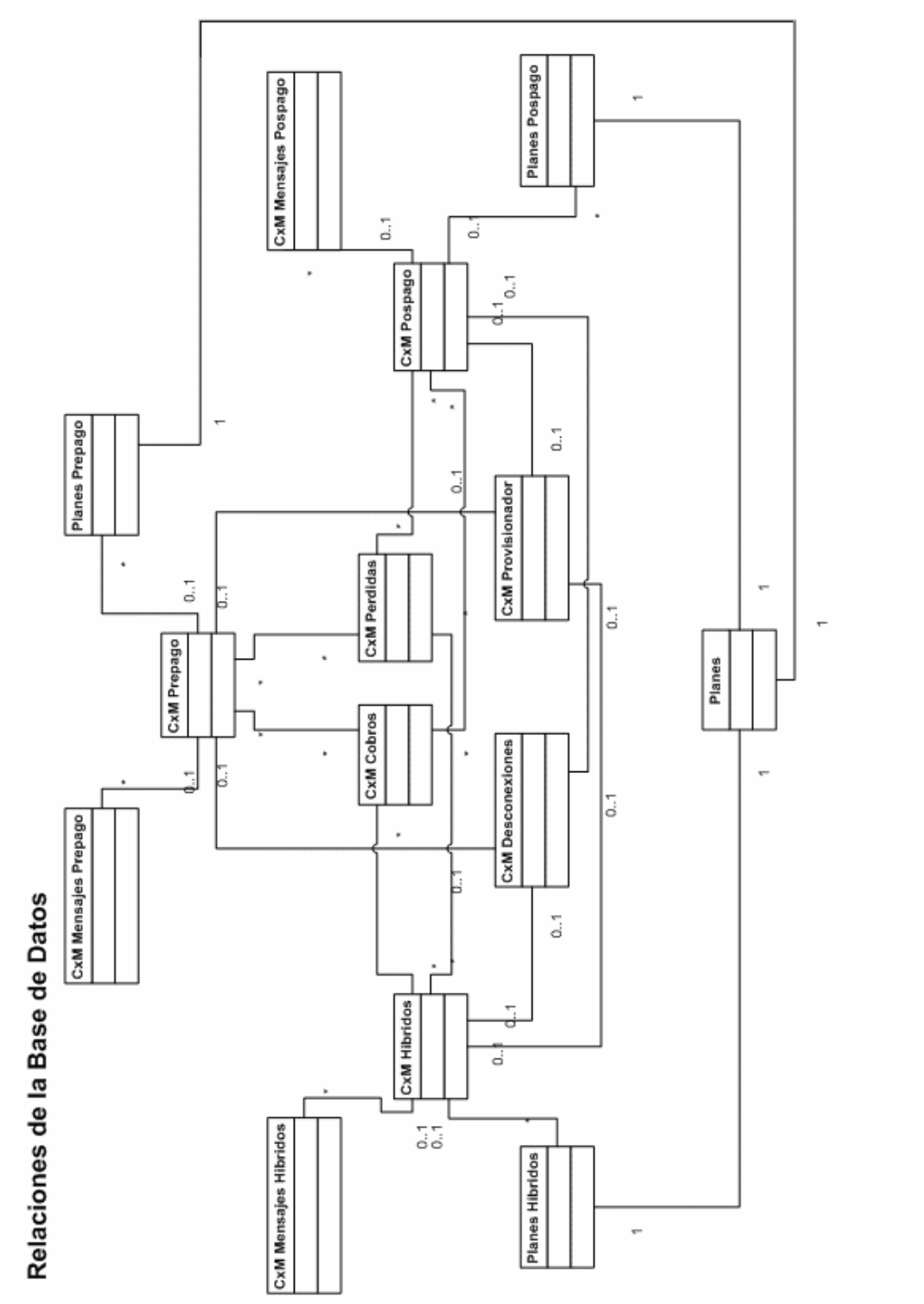
El harware mínimo necesario es un computador PC con un procesador de 600 megahertz con 128 MB de memoria RAM, 800MB de disco duro para la plataforma .NET, display de 800 x 600 y 256 colores y con tarjeta de red.

Además, para probar la conexión y realizar las pruebas es necesario tener acceso a la red donde se encuentra el servidor que contiene el Socket_Server y poder acceder al mismo por medio de Telnet u otro método de administración remota para realizar cambios en el Socket_Server de ser necesario, y manipular los puertos de acceso a la interfase.

3. Base de Datos

La información que se manipula en la interfase de cobro de mensajes se encuentra en una base de datos llamada SMSDatos. Esta base de datos contiene una variedad de tablas relacionadas entre si que en conjunto acumulan toda la información necesaria para mantener un registro de los planes de mensajería, los usuarios suscritos a los planes, los mensajes enviados, los cobros realizados y las pérdidas dadas por la falta de provisonamiento del servicio.

A continuación en la figura 1 se presenta la estructura de la base de datos y seguida a la misma el diccionario de datos que detalla las principales tablas a manipularse (tabla 1).



SMSDatos

Figura 1

3.1 Diccionario de Datos

En el siguiente diccionario de datos se describe la estructura de las tablas más utilizadas por la interfase para la manipulación de la información.

Planes

Esta tabla define los datos comunes de todos los planes en general

Nombre	**Tipo**	**Null**	**Valor Único**
PLAN	VARCHAR2(10)	Not Null	Si (Key)
MS_INCLUIDOS	NUMBER	Null	No
COSTO_MS_ADD	VARCHAR2(10)	Null	No

Planes Híbridos

Planes Híbridos se deriva de la tabla Planes con el complemento de la información de los planes Híbridos

Nombre	**Tipo**	**Null**	**Valor Único**
PLAN	VARCHAR2(10)	Not Null	Si (Key)
PLAN_DESCONEXION	VARCHAR2(10)	Not Null	No
COSTO_PAQUETE	NUMBER	Null	No
MENSAJES_INCLUIDOS	NUMBER	Null	No
COSTO_ADICIONAL	NUMBER	Null	No
COSTO_INTERNACIONAL	NUMBER	Null	No
DESCRIPCION	VARCHAR2(10)	Null	No
FECHA_ACTIVACION	DATE	Not Null	No
FECHA_DESACTIVACION	DATE	Null	No

Planes Pospago

Planes Pospago se deriva de la tabla Planes con el complemento de la información necesaria para los planes Pospago

Nombre	**Tipo**	**Null**	**Valor Único**
PLAN	VARCHAR2(10)	Not Null	Si (Key)
PLAN_DESCONEXION	VARCHAR2(10)	Not Null	No
MENSAJES_INCLUIDOS	NUMBER	Null	No
DESCRIPCION	VARCHAR2(10)	Null	No
FECHA_ACTIVACION	DATE	Not Null	No
FECHA_DESACTIVACION	DATE	Null	No

Plan Prepago

Planes Prepago se deriva de la tabla Planes con el complemento de la información sobre los planes Prepago.

Nombre	**Tipo**	**Null**	**Valor Único**
PLAN	VARCHAR2(10)	Not Null	Si (Key)
PLAN_DESCONEXION	VARCHAR2(10)	Not Null	No
COSTO_PAQUETE	NUMBER	Null	No
MENSAJES_INCLUIDOS	NUMBER	Null	No
PERIODO	NUMBER	Not Null	No
COSTO_ADICIONAL	NUMBER	Null	No
COSTO_INTERNACIONAL	NUMBER	Null	No
DESCRIPCION	VARCHAR2(10)	Null	No
FECHA_ACTIVACION	DATE	Not Null	No
FECHA_DESACTIVACION	DATE	Null	No

CxM Híbridos

CxM Híbridos es la tabla que contiene el número de celular (MIN), el plan al que pertenece y la información del uso del servicio de este usuario.

NOMBRE	**TIPO**	**Null**	**Valor Único**
MIN	VARCHAR2(10)	Not Null	No (Key)
PLAN	VARCHAR2(10)	Not Null	No (Key)
CONTADOR	NUMBER	Null	No
CONTADOR_GRATUITOS	NUMBER	Null	No
CONTADOR_PLAN	NUMBER	Null	No
CONTADOR_COBRO	NUMBER	Null	No
CONTADOR_INTERNACIONALES	NUMBER	Null	No
CONTADOR_WIS	NUMBER	Null	No
GRATUITOS_ACTIVADOS	NUMBER	Not Null	No
FECHA_EXPIRACION	DATE	Not Null	No
ESTADO	NUMBER	Not Null	No
ULTIMO_CAMBIO	DATE	Null	No
ULTIMO_COBRO	DATE	Null	No
USUARIO	VARCHAR2(50)	Null	No
FECHA_ACTIVACION	DATE	Not Null	No
FECHA_DESACTIVACION	DATE	Null	No

CxM Prepago

CxM Prepago contiene los usuarios prepago, su número de celular (MIN), el plan Prepago y los datos de uso del usuario.

NOMBRE	TIPO	Null	Valor Único
MIN	VARCHAR2(20)	Not Null	No (Key)
PLAN	VARCHAR2(20)	Not Null	No (Key)
CONTADOR	NUMBER	Null	No
CONTADOR_GRATUITOS	NUMBER	Null	No
CONTADOR_PLAN	NUMBER	Null	No
CONTADOR_COBRO	NUMBER	Null	No
CONTADOR_INTERNACIONALES	NUMBER	Null	No
CONTADOR_WIS	NUMBER	Null	No
GRATUITOS_ACTIVADOS	NUMBER	Not Null	No
FECHA_EXPIRACION	DATE	Not Null	No
ESTADO	NUMBER	Not Null	No
ULTIMO_CAMBIO	DATE	Null	No
ULTIMO_COBRO	DATE	Null	No
USUARIO	VARCHAR2(50)	Null	No
FECHA_ACTIVACION	DATE	Not Null	No
FECHA_DESACTIVACION	DATE	Null	No

CxM Pospago

CxM Pospago contiene los usuarios Pospago, con su número de teléfono (MIN), el plan Pospago y la información del uso del servicio de mensajería.

NOMBRE	TIPO	Null	Valor Único
MIN	VARCHAR2(20)	Not Null	No (Key)
PLAN	VARCHAR2(20)	Not Null	No (Key)
CONTADOR	NUMBER	Null	No
CONTADOR_INTERNACIONALES	NUMBER	Null	No
FECHA_EXPIRACION	DATE	Not Null	No
ULTIMO_CAMBIO	DATE	Null	No
USUARIO	VARCHAR2(50)	Null	No
FECHA_ACTIVACION	DATE	Not Null	No
FECHA_DESACTIVACION	DATE	Null	No

Diccionario de Datos

Tablas 1

4. Procedimientos Importantes

En la interfase de administración existen dos elementos críticos y necesarios de conocer para estar en la posibilidad de dar un correcto mantenimiento o modificación al sistema. Estos elementos son la comunicación por tramas XML y la descripción de las clases que componen la interfase.

4.1 Comunicación por Tramas XML

La comunicación por tramas de XML comprende en tramas que se transmiten entre la interfase y el Socket_Server. Estas son la base para extraer como manipular la información y comportamiento del Socket_Server.

A continuación se describe cada uno de los procesos que realiza la interfase, y su respectiva trama XML utilizada para cumplir el proceso especificado. Cabe recalcar que la comunicación es de dos vías por lo que para cada acción existe una trama de petición y una de respuesta. Además al final se detalla los tipos de errores que se pueden dar con su respectivo significado.

4.1.1 Manipulación de Planes

En el primer grupo de necesidades está la manipulación de los planes de mensajería que involucra la creación, eliminación y consulta del plan.

4.1.1.1 Creación de un Plan

Los Parámetros son:

Plan

Char[10]

Es el número del Plan por ejemplo ‘366’

Plan de desconexión

Char[10]

Es el número del Plan de desconexión por ejemplo ‘367’

Plataforma

int

Es el parámetro para saber a que palataforma aplica, puede ser PPP, HYB, POS y su traducción es 2001, 2002 y 2003 correspondientemente.

Tipo (Free, normal)

int

Ahora se manejan Planes promocionales, y normales. El número 2011 es normal y 2012 promocional.

Periodo

int

Numero de días del periodo del paquete de mensajería

Precio

float

Precio del paquete de mensajería

Mensajes

int

Numero de mensajes incluidos en el paquete de mensajería

Precio adicional

Float

Costo del mensaje adicional

Descripción

Char[40]

Una descripción breve del Plan

Las respuestas posibles para esta operación serán las siguientes:

0 para una operación exitosa o uno de los siguientes errores:

PLATAFORMA_INCORRECTA

REGISTRO_EXISTENTE

TIPO_INCORRECTO

ERROR

El protocolo de XML a seguir y su formato se describe a continuación:

Solicitud Cliente:

```
<?XML version="1.0" encoding="UTF-8"?>
<solicitud>
<transaccion nombre="creaPlan">
<parametro nombre="Plan">valor</parametro>
<parametro nombre="PlanDesconexion">valor</parametro>
<parametro nombre="plataforma">valor</parametro>
<parametro nombre="tipo">valor</parametro>
<parametro nombre="periodo">valor</parametro>
<parametro nombre="precio">valor</parametro>
<parametro nombre="mensajes">valor</parametro>
<parametro nombre="precioAdicional">valor</parametro>
<parametro nombre="precioInternacional">valor</parametro>
<parametro nombre="descripcion">valor</parametro>
</transaccion>
</solicitud>
```

Respuesta Servidor

```
<?XML version="1.0" encoding="UTF-8"?>
<respuesta>
<transaccion nombre="creaPlan">
<resultado nombre="" descripción="">valor</resultado>
<error descripcion="">valor</error>
```

</transaccion>

</respuesta>

4.1.1.2 Eliminación de un Plan

Esta función permite eliminar un Plan existente en la plataforma.

Los parámetros son:

Plan

Char[10]

Es el número del Plan por ejemplo ‘366’

Plataforma

int

Es el parámetro para saber a que palataforma aplica, puede ser PPP, HYB, POS

Tipo (Free, normal)

int

Ahora se manejan Planes promocionales, y normales

Las respuestas son:

0 Eliminado exitosamente

PALAFORMA_INCORRECTA

REGISTRO_INEXISTENTE

TIPO_INCORRECTO

ERROR

El protocolo protocolo para realizar la operacion es:

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="eliminaPlan">

<parametro nombre="Plan">valor</parametro>

<parametro nombre="plataforma">valor</parametro>

<parametro nombre="tipo">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="eliminaPlan">

<resultado nombre="" descripción="">valor</resultado>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.1.3 Consulta de información de Plan

Esta función permite ver la información de un Plan configurado en la plataforma

Parámetros

Plan

Char[10]

Es el número del Plan por ejemplo ‘366’

Plataforma

int

Es el parámetro para saber a que palataforma aplica, puede ser PPP, HYB, POS

Tipo

int

Es el tipo de Plan, normal o promocional

Las respuestas son:

0 mas información que es una petición exitosa

PLATAFORMA_INCORRECTA

PLAN_INEXISTENTE

ERROR

Parámetros de retorno si la petición es exitosa:

Plan

Char[10]

Es el número del Plan por ejemplo ‘366’

Plan de Desconexion

Char[10]

Es el número del Plan de desconexión por ejemplo ‘367’

Plataforma

int

Es el parámetro para saber a que palataforma aplica, puede ser PPP, HYB, POS

Tipo (Free, normal)

int

Ahora se manejan Planes promocionales, y normales

Periodo

Int

Número de dias de perióro del paquete de mensajería

Precio

Float

Precio del paquete de mensajería

Mensajes

Int

Número de mensajes incluidos en el paquete de mensajería

Precio adicional

Float

Costo del mensaje adicional

Descripción

Char[40]

Una descripción breve del Plan

El protocolo a seguir es:

Solicitud Cliente

<?XML version="1.0" encoding=" UTF-8"?>

<solicitud>

<transaccion nombre="consultaPlan">

<parametro nombre="Plan">valor</parametro>

<parametro nombre="plataforma">valor</parametro>

<parametro nombre="tipo">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="consultaPlan">

<resultado nombre="Plan">valor</parametro>

<resultado nombre="PlanDesconexion">valor</parametro>

<resultado nombre="plataforma">valor</parametro>

<resultado nombre="tipo">valor</parametro>

<resultado nombre="periodo">valor</parametro>

<resultado nombre="precio">valor</parametro>

<resultado nombre="mensajes">valor</parametro>

<resultado nombre="precioAdicional">valor</parametro>

<resultado nombre="precioInternacional">valor</parametro>

<resultado nombre="descripcion">valor</parametro>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.2 Provisionamiento del Servicio

Una vez que se tienen Planes ingresados en la base de datos se puede realizar la siguiente operación que es el provisionamiento de un Plan a un número telefónico. En otras palabras asignar un plan de mensajes a un cliente específico, como quitar el plan a un cliente, consultar el plan de un cliente o reactivar el plan de un cliente. Cabe resaltar que un cliente solo puede tener un plan de mensajes.

4.1.2.1 Inserción de Provisionamiento

Esta función permite provisionar un teléfono en la plataforma

Parámetros

MIN

Char[10]

Número telefonico de abonado

Plan

Char[10]

Es el número del Plan por ejemplo ‘366’

Plataforma

int

Es el parámetro para saber a que palataforma aplica, puede ser PPP, HYB, POS

Usuario

Char[20]

Una descripción del usuario que inserta el min

Tipo (Free, normal)

int

Ahora se manejan Planes promocionales, y normales

Dia de corte

Int

Fecha de corte para híbridos y pospago

Las respuestas son:

0 Creado exitosamente

PLATAFORMA_INCORRECTA

PLAN_INEXISTENTE

REGISTRO_EXISTENTE

TIPO_INCORRECTO

MIN_SIN_PLAN

ERROR

El protocolo a seguir es:

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="provisionaTelefono">

<parametro nombre=" min">valor</parametro>

<parametro nombre="Plan">valor</parametro>

<parametro nombre="plataforma">valor</parametro>

<parametro nombre="usuario">valor</parametro>

<parametro nombre="tipo">valor</parametro>

<parametro nombre="diaCorte">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="provisionaTelefono">

<resultado nombre="" descripción="">valor</resultado>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.2.2 Eliminación del provisionamiento

Esta funcion permite desprovisionar un teléfono en la plataforma.

Parámetros

MIN

Char[10]

Número telefónico de abonado

Las respuestas son:

0 Eliminación exitosa

REGISTRO_INEXISTENTE

ERROR

El protocolo a seguir es:

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="eliminaTelefono">

<parametro nombre=" min">valor</parametro>

<parametro nombre="Plan">valor</parametro>

<parametro nombre="usuario">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="eliminaTelefono">

<resultado nombre="" descripción="">valor</resultado>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.2.3 Consulta de información de Teléfono

Esta función permite ver la información de un teléfono configurado en la plataforma

Parámetros

MIN

Char[10]

Es el número de teléfono

Las respuestas son:

0 con la informacion en caso de éxito

MIN_INEXISTENTE

ERROR

Parámetros de retorno

MIN

Char[10]

Número telefónico de abonado

Plan

Char[10]

Es el número del Plan por ejemplo ‘366’

Plataforma

int

Es el parámetro para saber a que palataforma aplica, puede ser PPP, HYB, POS

Mensajes Totales Enviados

Int

Número de mensajes enviados

Mensajes Gratuitos Enviados

Int

Número de mensajes gratutos enviados

Mensajes Enviados dentro del plan

Int

Número de mensajes enviados dentro del plan

Mensajes Gratuitos Activados

Int

Número de mensajes gratutos activados

Fecha de caducidad

Char[10]

Fecha de caducidad del paquete

Estado

Int

Estado actual del suscriptor

Mensajes internacionales Enviados

Int

Número de mensajes internacionales enviados

Mensajes wis Enviados

Int

Número de mensajes wis enviados

El protocolo a utilizarse es:

Solicitud Cliente

```
<?XML version="1.0" encoding="UTF-8"?>
<solicitud>
<transaccion nombre="consultaTelefono">
<parametro nombre="min">valor</parametro>
</transaccion>
</solicitud>
```

Respuesta Servidor

```
<?XML version="1.0" encoding="UTF-8"?>
<respuesta>
```

<transaccion nombre="consultaTelefono">

<resultado nombre="min">valor</parametro>

<resultado nombre="Plan">valor</parametro>

<resultado nombre="plataforma">valor</parametro>

<resultado nombre="mensajesTotalesEnviados">valor</parametro>

<resultado nombre="mensajesGratuitosEnviados">valor</parametro>

<resultado nombre="mensajesEnPlanEnviados">valor</parametro>

<resultado nombre="mensajesGratuitosActivados">valor</parametro>

<resultado nombre="fechaCaducidad">valor</parametro>

<resultado nombre="mensajesInternacionalesEnviados">valor</parametro>

<resultado nombre="estado">valor</parametro>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.2.4 Reactivación del Provisionamiento

Esta función sera llamada cuando se ingrese una tarjeta a un teléfono

Parámetros

MIN

Char[10]

Número telefónico de abonado

Las respuestas son:

0 Procesado exitosamente

MIN_INEXISTENTE

ERROR

Protocolo

Solicitud Cliente

<?XML version="1.0" encoding=" UTF-8"?>

<solicitud>

<transaccion nombre="reactivaTelefono">

<parametro nombre=" min">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="reactivaTelefono ">

<resultado nombre="" descripcion="">valor</resultado>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.3 Reportes

La siguiente funcionalidad de la interfase es la generación de dos tipos de reportes sobre los mensajes y costos que se han dado en un rango de tiempo. El primero es un reporte determinado por un rango de tiempo y un número celular específico y el segundo reportes se determina por un rango de tiempo y muestra todos los mensajes y costos totales en este tiempo.

4.1.3.1 Reporte por número telefónico

Reporte de mensajes enviados por un min

Parametros

MIN

Char[10]

Número telefónico del abonado

Fecha de Inicio

Char[10]

Fecha inicial de reporte

Formato dd-MES-aa

Fecha de fin

Char[10]

Fecha final de reporte

Formato dd-MES-aa

Las respuestas son:

0 y la información si la consulta es exitosa

PLATAFORMA_INCORRECTA

MIN_INEXISTENTE

ERROR

Parámetros de retorno

MIN

Char[10]

Número telefónico de abonado

Mensajes Totales Enviados

Int

Número de mensajes enviados

Mensajes operadora Enviados

Int

Número de mensajes normales enviados

Mensajes internacional Enviados

Int

Número de mensajes internacionales enviados

Mensajes wis Enviados

Int

Número de mensajes wis enviados

Nombre del archivo

Char[50]

Nombre del archivo con el reporte en el servidor

Protocolo

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="reportePorNumero">

<parametro nombre="min">valor</parametro>

<parametro nombre="fechaInicio">valor</parametro>

<parametro nombre="fechaFin">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="reportePorNumero">

<resultado nombre="min">valor</parametro>

<resultado nombre="mensajesTotalesEnviados">valor</parametro>

<resultado nombre="mensajesOperadoraEnviados ">valor</parametro>

<resultado nombre="mensajesInternacionalesEnviados">valor</parametro>

<resultado nombre="mensajesWisEnviados">valor</parametro>

<resultado nombre="nombreArchivo">valor</parametro>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.3.2 Volumen de mensajes totales

Este reporte muestra todos los mensajes y costos involucrados en un rango de tiempo dado por el Administrador.

Parametros

Fecha de Inicio

Char[10]

Fecha inicial de reporte

Formato dd-MES-aa

Fecha de fin

Char[10]

Fecha final de reporte

Formato dd-MES-aa

Las respuestas son:

0 y la información si la consulta es exitosa

ERROR

Parámetros de retorno

Número de mensajes PPP

Int

Número de mensajers HYB

Int

Número de mensajes POS

Int

Número de mensajes internacionales PPP

Int

Número de mensajers internacionales HYB

Int

Número de mensajes internacionales POS

Int

Número de mensajes wis PPP

Int

Número de mensajers wis HYB

Int

Número de mensajes wis POS

Int

Cantidad Cobros

Int

Monto Cobros Provisionamiento

Float

Monto Cobros adicionales

Float

Monto Cobros Internacionales

Float

Monto Cobros wis

Float

Monto Cobros Total

Float

Monto Pérdidas

Flota

Cantidad Desconexiones

Int

Cantidad Pérdidas

Int

El protocolo a utilizarse es:

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="reporteVolumenMensajes">

<parametro nombre="fechaInicio">valor</parametro>

<parametro nombre="fechaFin">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="reporteVolumenMensajes">

<resultado nombre="mensajesPPP">valor</parametro>

<resultado nombre="mensajesHYB">valor</parametro>

<resultado nombre="mensajesPOS">valor</parametro>

<resultado nombre="mensajesInternacionalesPPP">valor</parametro>

<resultado nombre="mensajesInternacionalesHYB">valor</parametro>

<resultado nombre="mensajesInternacionalesPOS">valor</parametro>

<resultado nombre="mensajesWisPPP">valor</parametro>

<resultado nombre="mensajesWisHYB">valor</parametro>

<resultado nombre="mensajesWisPOS">valor</parametro>

<resultado nombre="cantidadCobros">valor</parametro>

<resultado nombre="montoCobrosProvisionamiento">valor</parametro>

<resultado nombre="montoCobrosAdicionales">valor</parametro>

<resultado nombre="montoCobrosInternacionales">valor</parametro>

<resultado nombre="montoCobrosWis">valor</parametro>

<resultado nombre="montoCobrosTotal">valor</parametro>

<resultado nombre="montoPerdidas">valor</parametro>

<resultado nombre="cantidadDesconexiones">valor</parametro>

<resultado nombre="cantidadPerdidas">valor</parametro>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.4 Manipulación de Procesos

Además de la manipulación de información, se requiere administrar procesos o daemons que se utilizan para la funcionalidad de toda la aplicación de CxM.

Dentro de estas funcionalidades de manipulación de procesos está: consulta de procesos registrados, consulta de configuración de proceso, configuración de proceso, consulta del estado de un proceso, arranque de un proceso y parada de un proceso.

4.1.4.1 Consulta procesos registrados

Esta función permite obtener una lista de los daemons registrados para ser administrados por el servidor de APIs

Las respuestas son:

0 e información si es exitoso.

ERROR

Parámetros de retorno

Nombre[ ]

Char[100]

Es el nombre del daemon

El protocolo utilizado:

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="consultaDaemonsRegistrados"/>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="consultaDaemonsRegistrados">

<resultado nombre="nombreDaemon">valor</parametro>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.4.2 Consulta configuración de proceso

Esta función permite obtener todos los parámetros del archivo de configuración de un daemon

Parámetros

Nombre Daemon

Char[100]

Es el nombre del daemon

Las respuestas son:

0 mas la información si es exitoso

DAEMON NO REGISTRADO

ERROR

Parámetros de retorno

Nombre Campo[ ]

Char[100]

Es el nombre del campo dentro del archivo de configuración

Valor Campo[ ]

Char[100]

Es el valor correspondiente al campo dentro del archivo de configuración

El protocolo utilizado:

Solicitud Cliente

```
<?XML version="1.0" encoding="UTF-8"?>
<solicitud>
<transaccion nombre="consultaConfiguracionDaemon">
<parametro nombre="nombreDaemon">valor</parametro>
</transaccion>
</solicitud>
```

Respuesta Servidor

```
<?XML version="1.0" encoding="UTF-8"?>
<respuesta>
<transaccion nombre="consultaConfiguracionDaemon ">
<parametro nombre="nombreCampo">valor</parametro>
<error descripcion="">valor</error>
</transaccion>
</respuesta>
```

4.1.4.3 Configuración del proceso

Esta función permite crear un nuevo archivo de configuración para el daemon determinado

Parámetros

Nombre Daemon

Char[100]

Es el nombre del daemon

Nombre Campo[ ]

Char[100]

Es el nombre del campo dentro del archivo de configuración

Valor Campo[ ]

Char[100]

Es el valor correspondiente al campo dentro del archivo de configuración

Los resultados son:

0 si es exitoso

ERROR

Protocolo

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="configuraDaemon">

<parametro nombre="nombreDaemon">valor</parametro>

<parametro nombre="*nombreCampo*">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="configuraDaemon">

<resultado nombre="" descripcion="">valor</resultado>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.4.4 Consulta del estado de un proceso

Esta función permite consultar el estado (ejecutándose o no) y el tiempo de ejecución de un daemon

Parámetros

Nombre Daemon

Char[100]

Es el nombre del daemon

Las respuestas son:

0 mas la información si es exitoso

ERROR

Parámetros de retorno

Estado

Int

1 si está ejecutándose ó 0 si no está ejecutándose

Tiempo de Ejecución

Char[10]

Indica el tiempo que lleva el daemon ejecutándose. Formato hh:mm:ss

Protocolo

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="consultaEstadoDaemon">

<parametro nombre="nombreDaemon">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="consultaEstadoDaemon">

<resultado nombre="ejecutandose">valor</resultado>

<resultado nombre="tiempoEjecucion">valor</resultado>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.4.5 Arranque de un proceso

Esta función permite arrancar un daemon

Parámetros

Nombre Daemon

Char[100]

Es el nombre del daemon

Las respuestas son:

0 si es exitoso

FALLO_ARRANQUE

Protocolo

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="arrancaDaemon">

<parametro nombre="nombreDaemon">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="arrancaDaemon ">

<resultado nombre="" descripcion="">valor</resultado>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.4.6 Parada de proceso

Esta función permite detener un daemon

Parámetros

Nombre Daemon

Char[100]

Es el nombre del daemon

Las respuestas son:

0 si se detiene exitosamente

FALLO_PARADA

Protocolo

Solicitud Cliente

<?XML version="1.0" encoding="UTF-8"?>

<solicitud>

<transaccion nombre="detieneDaemon">

<parametro nombre="detieneDaemon">valor</parametro>

</transaccion>

</solicitud>

Respuesta Servidor

<?XML version="1.0" encoding="UTF-8"?>

<respuesta>

<transaccion nombre="detieneDaemon">

<resultado nombre="" descripcion="">valor</resultado>

<error descripcion="">valor</error>

</transaccion>

</respuesta>

4.1.5 Consulta estado servidor APIs

Esta función permite consultar algunos parámetros del estado del servidor de APIs que son: puerto activo, puerto inactivo y tiempo de ejecución dado en minutos. Y si es superior a un día muestra que es mayor a un día.

Las respuestas son:

0 si es exitoso

ERROR

Parámetros de retorno

Puerto activo

Int

Número de puerto

Puerto inactivo

Int

Número de puerto

Tiempo de Ejecución

Char[10]

Indica el tiempo que lleva el servidor ejecutándose. Formato hh:mm:ss

Protocolo

Solicitud Cliente

```
<?XML version="1.0" encoding="UTF-8"?>
<solicitud>
<transaccion nombre="consultaEstadoSockserver"/>
</solicitud>
```

Respuesta Servidor

```
<?XML version="1.0" encoding="UTF-8"?>
<respuesta>
<transaccion nombre="consultaEstadoSockserver ">
<parametro nombre="puertoActivo">valor</parametro>
<parametro nombre="puertoInactivo">valor</parametro>
<parametro nombre="tiempoEjecucion">valor</parametro>
</transaccion>
</respuesta>
```

4.1.6 Lista de Errores

Finelamente está la descripción de los errores que se pueden producir. Estos errores se pueden dar una las diferentes consultas y se utilizan para notificar al Administrador de que no se pudo concluir su petición y la razón.

PLATAFORMA_INCORRECTA	10001
REGISTRO_INEXISTENTE	10002
REGISTRO_EXISTENTE	10003
TIPO_INCORRECTO	10004
ERROR_INSERCION	10005
ERROR_REMOCION	10006
MIN_EXISTENTE	10007
MIN_SIN_PLAN	10008
MIN_INEXISTENTE	10009
ERROR_TIPO_TRAMA	10010
ERROR_TRAMA	10011
ERROR_NO_PARAMETROS	10012
ERROR_CREACION_XML	10013
ERROR_REACTIVACION	10014
ERROR	10015
FALLO_ARRANQUE_DAEMON	10016
FALLO_PARADA_DAEMON	10017
PROCESO_NO_REGISTRADO	10018
ERROR_GUARDAR_ARCHIVO.CONF	10019

4.2 Descripción de Clases de la Interfase.

La interfase del Sistema de Cobro de Mensajes se encuentra compuesta por varias clases con sus respectivos métodos que se relacionan entre si para cumplir las diferentes funciones.

A continuación en la siguiente tabla se lisa las clases con su respectiva descripción.

Clase	Descripción
Acerca	Muestra una ventana con la información básica de cómo contactar el servicio de soporte de la interfase.
CambioNombre	Es una ventana que permite definir un nuevo nombre de Usuario.
CambioContraseña	Es una ventana que permite definir un nuevo contraseña del Usuario.
Configuración	Esta clase permite extraer y definir los parámetros de la conexión de la interfase.
ConsultaConfiguracionProceso	Genera una ventana que contiene la lista de los procesos y permite escoger uno de ellos para extraer su archivo de configuración y editarlo.
ConsultaEstadoProceso	Genera una ventana que lista los procesos, con su estado de activo/inactivo y el tiempo de ejecución. También permite activar o desactivar al proceso señalado.

ConsultaComponente	Es una ventana donde se realiza la petición de la consulta o edición de un plan de mensajería.
ConsultaProvisionamiento	Es una ventana donde se puede realizar la consulta de un número de celular para observar su plan de mensajes y su consumo actual.
ConsultaServidorApi	Genera una ventana con la información del Socket_Server, puerto activo, puerto inactivo y tiempo de ejecución.
CrearComponente	Es una ventana que acepta los parámetros de un nuevo plan y crea el plan especificado.
CrearProvisionamiento	Es una ventana que acepta los parámetros de un nuevo provisionamiento del servicio de mensajería a un celular específico y crea el provisionamiento.
CreateLog	Esta clase se utilizar para registrar las acciones de la interfase en un archivo de texto (bitácora) ubicado en el directorio donde se encuentra la Interfase.
DesplegarComponente	Esta clase es llamada para dar formato y desplegar y editar un plan de mensajería consultado.
DesplegarProvisionamiento	Esta clase es llamada para dar formato y desplegar la información de un celular específico con su respectivo plan.
EliminarComponente	Genera una ventana que acepta el plan de mensajería, plataforma y tipo y si el componente existe lo elimina.
EliminarProvisionamiento	Genera una ventana que acepta un número de celular y si lo encuentra retira el provisionamiento al mismo.

LogIn	Es una ventana que se despliega para ingresar el nombre de usuario y contraseña, revisa si son correctos y permite el acceso a la interfase.
LogOn	Genera una ventana para ingresar la dirección IP del Socket_Server y el puerto. Permite guardar en el archivo de configuración de la interfase y conectar al Socket_Server.
mainForm	Es la clase padre desde donde se llama a las otras clases. Contiene el menú del usuario y las diferentes opciones de las consultas. También aquí se encuentra los objetos de conexión y transmisión de las tramas XML.
NumeroMenajesMin	Genera una ventana que acepta un rango de fechas y genera un reporte con los mensajes consumidos en ese rango de fechas.
Reactivacion	Genera una ventana donde se ingresa un número de celular y se reactiva al servicio de mensajería del mismo con el último plan relacionado a este.
Traductor	Esta clase contiene una variedad de métodos comunes que se utilizan por las otras clases de la interfase.
VolumenMensajes	Genera una ventana con un reporte detallado del consumo de mensajes en un rango de fechas de todo el sistema.

5. Estructura de Archivos

En el momento de la instalación de la interfase, el usuario define la carpeta destino donde se instalará la interfase y sus archivos de configuración. El wizard escoge la raíz donde se encuentra el archivo de setup y define una carpeta llamada AdministracionCxM donde estará la interfase. En dicha carpeta se escribe los siguientes archivos:

AdministracionCxM.exe: Es el archivo ejecutable de la interfase

Novix.ico: Archivo de tipo icono que tiene el logo de la empresa.

Usuario.txt: Archivo de configuración de usuario.

Interfaz.conf: Archivo de configuración de la conexión de la interfase.

Log.txt: Archivo donde se guarda las transacciones realizadas por la interfase.

Es importante recordar donde se instala la interfase ya que pueden darse circunstancias donde se requiere cambiar o manipular sus componentes, como también para crear accesos directos a la interfase desde otros lugares.

Anexo III Manual de Usuario

Soluciones de Código Abierto

Movistar
Interfase del Sistema de Cobro de Mensajes
Junio del 2005

Manual de Usuario

Fecha última modificación: **Junio, 20 del 2005**

Investigación y Desarrollo
Novix
Quito, Ecuador

Contenidos

1. Descripción General

La interfase de cobro de mensajes es un sistema de administración diseñado para facilitar el acceso y control de toda la aplicación Socket_Server encargada de los planes de mensajería y cobro de mensajes.

Esta interfase ofrece varias funcionalidades, desde un control de usuarios para protección del administrador, manipulación de la información de sistema de mensajería y administración de procesos (daemons) relacionados con el Socket_Server.

Con el siguiente manual de usuario se pretende ofrecer una explicación clara y concreta del funcionamiento de la interfase al Administrador del sistema de cobro de mensajes.

A continuación observamos cada una de las funcionalidades del la interfase de cobro por mensaje.

2. Instalación de la Interfase

Para la instalación de la interfase solo se requiere hacer doble clic en el icono setup distribuido, luego seguir las instrucciones del wizard de instalación y en el wizard colocar la carpeta destino donde se instalarán los componentes de la Interfase de Administración CxM (Administración Cobro de Menaje). Luego de terminar el wizard de instalación y se tendrá acceso a la interfase ya sea por el menú inicio o por un icono localizado en la carpeta donde se instaló la Interfase.

3. Ingreso del Usuario.

Al ingresar por primera vez a la interfase, el sistema notificará que se recomienda cambiar el nombre de usuario y clave. Para obtener el usuario y clave por defecto para el primer ingreso se debe solicitar a la persona encargada del departamento de cobro de mensajes.

Se ingresa el usuario y clave y el menú de la interfase se activa.

Para el cambio de usuario y Clave se va al menú Archivo, Administrador, Cambiar Nombre o Cambiar Contraseña respectivamente; como se ve a continuación:

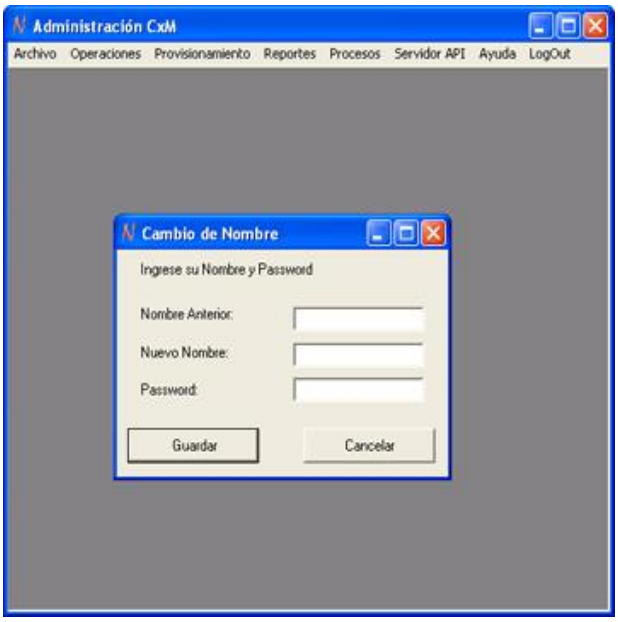


Figura A3.1

Ingreso de usuario

4. Conexión al Socket_Server.

Para realizar la conexión al Socket_Server se realiza de dos maneras: Archivo, conectar y Archivo, cambiar conexión. La opción cambiar conexión (Figura A3.2) se utiliza para especificar el servidor y socket donde se encuentra el Socket_Server y se realiza en el primer ingreso a la interfase, o para cambiar los parámetros de la conexión. La opción conectar, utiliza los parámetros ya establecidos anteriormente en la última sesión de la interfase y se conecta directamente.

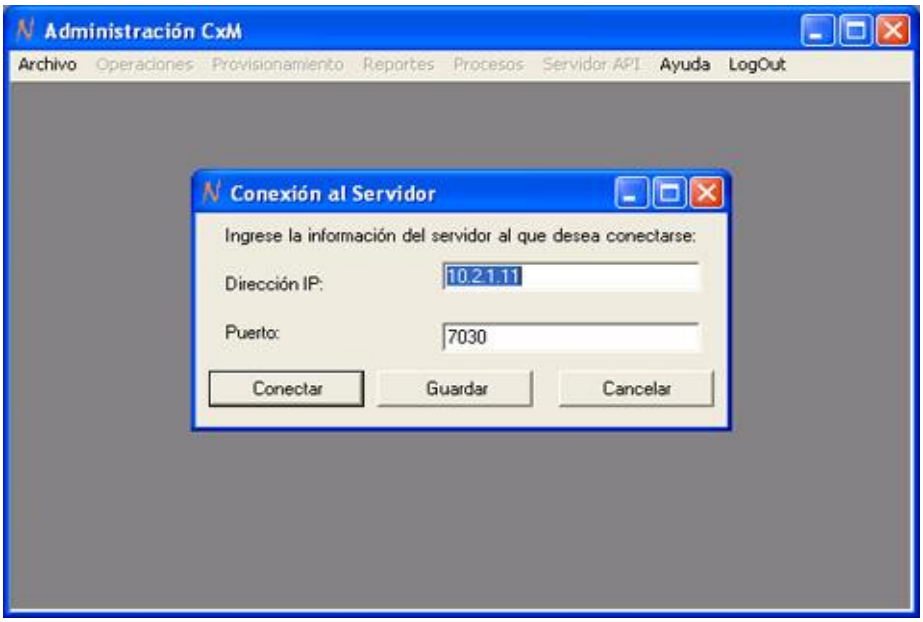


Figura A3.2

Cambiar Conexión

5. Manipulación de Planes de Mensajería

En la interfase de Administración se llama a los planes de mensajería como componentes, por lo que en el menú Operaciones tenemos todas las operaciones con el nombre de Crear Componente, Eliminar Componente y Consultar/Editar Componente.

Para la creación de un componente se escoge el menú Operaciones, Crear Componente. A continuación una ventana aparece con la información necesaria para crear un componente. (Figura A3.3)

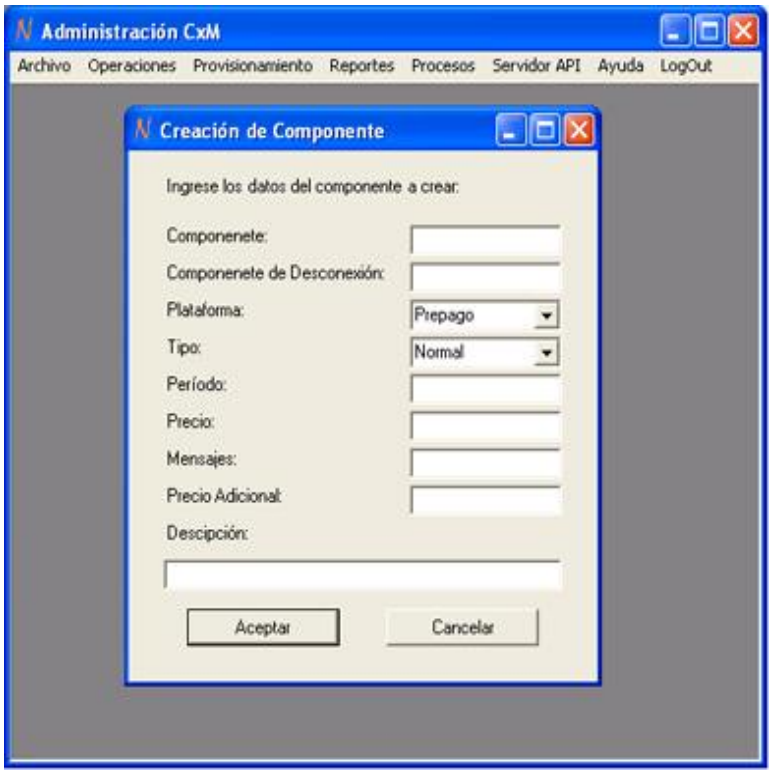


Figura A3.3

Creación de Componente

En el campo Componente se coloca el número de componente, en el componente de Desconexión, el número de componente de desconexión. Luego se escoge la plataforma Prepago, Pospago o Híbrido. El tipo puedes ser normal o promocional. El período define el tiempo de duración del componente. Finalmente tenemos el precio, mensajes que es el número de mensajes incluidos y el precio adicional después de cumplidos el número de mensajes. La descripción es un campo no necesario donde se puede dar una pequeña descripción del componente.

Para la eliminación del componente (Figura A3.4) se tiene una nueva ventana donde se pide el número de componente, la plataforma a la que pertenece y el tipo. Si la eliminación se realiza exitosamente la interfase notifica del éxito o del error en caso de que no se elimine.

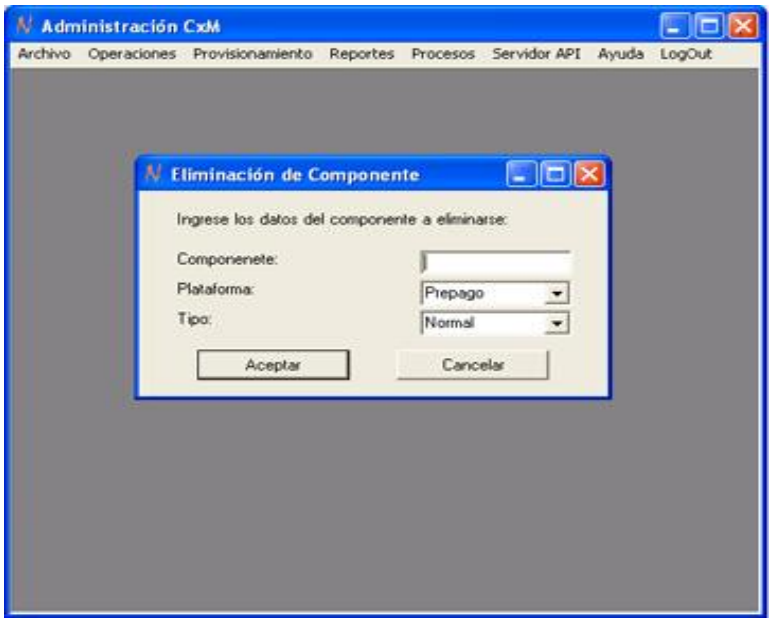


Figura A3.4

Eliminación de Componente

La consulta y edición de componente presenta ventanas similares a las Figuras A3.4 y A3.3. Primer se tiene una ventana solicitando los datos del componente a consultar, y las opciones de solo consultar o de editar. Como respuesta se presenta una ventana similar a la de creación de componente donde se puede editar todos los campos menos el de componente, plataforma y tipo.

6. Provisionamiento de Servicio

El Provisionamiento del servicio consta en suscribir a un teléfono celular con un componente específico de tal manera que dicho teléfono puede mandar la cantidad de mensajes establecidos en el componente al precio del componente.

La primera opción del provisionamiento es la creación de componente donde se ingresa el número de celular (MIN), plataforma, tipo, usuario y en el caso de plataformas híbridas y pospago se ingresa el día de corte. (Figura A3.5)

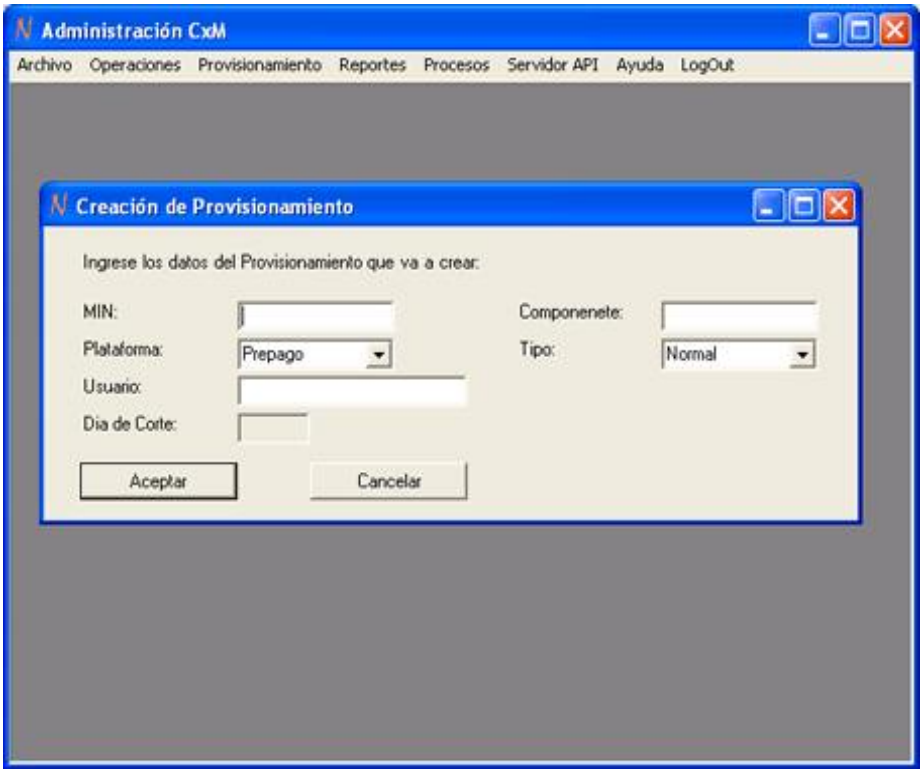


Figura A3.5

Creación de Provisionamiento

La opción Provisionamiento, Eliminar Provisionamiento nos permite retirar el servicio de mensajería a un celular específico Figura A3.6. La interfase notificará si tuve éxito en la operación o si se dio un error

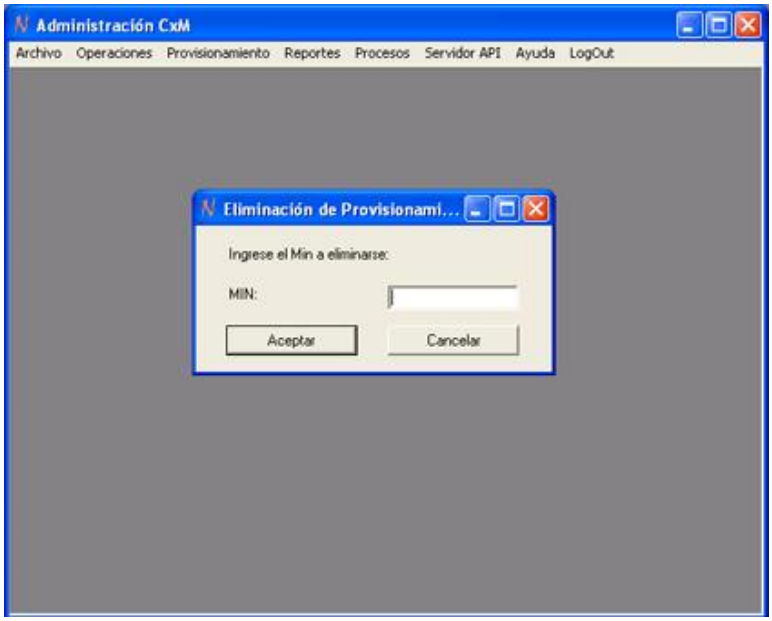


Figura A3.6

Eliminar Provisionamiento

Para consultar el provisionamiento se escoge el menú Provisionamiento, Consultar Provisionamiento. En la nueva ventana similar a la de eliminación de provisionamiento se ingresa el número de celular y como respuesta se despliega una nueva ventana Figura A3.7 con la información sobre el celular

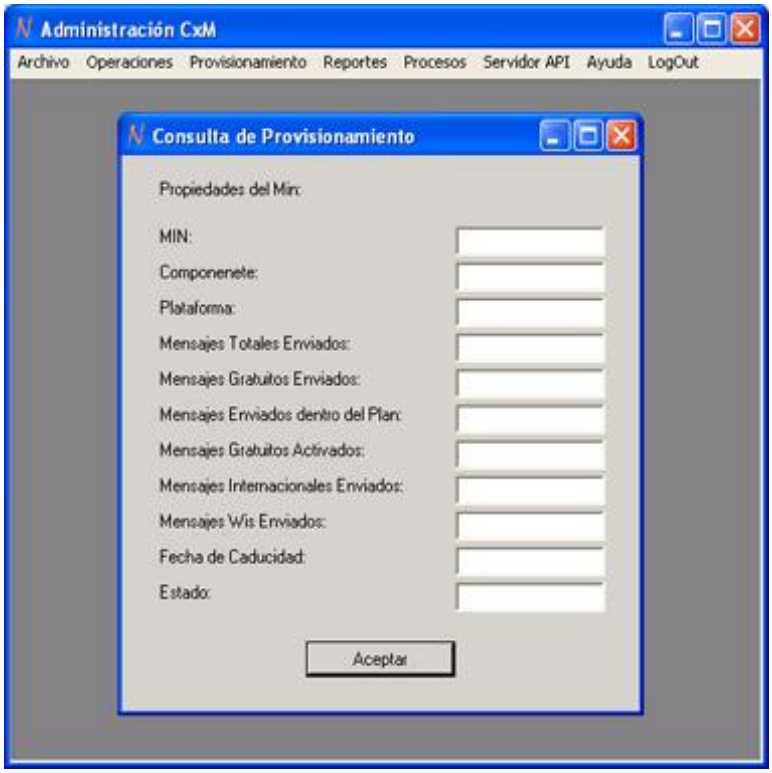


Figura A3.7

Datos del Provisionamiento

La última opción del menú Componentes es la reactivación que sirve para volver a provisionar del servicio a celulares que han sido suspendidos del servicio por diferentes motivos como la falta de saldo. (Figura A3.8).

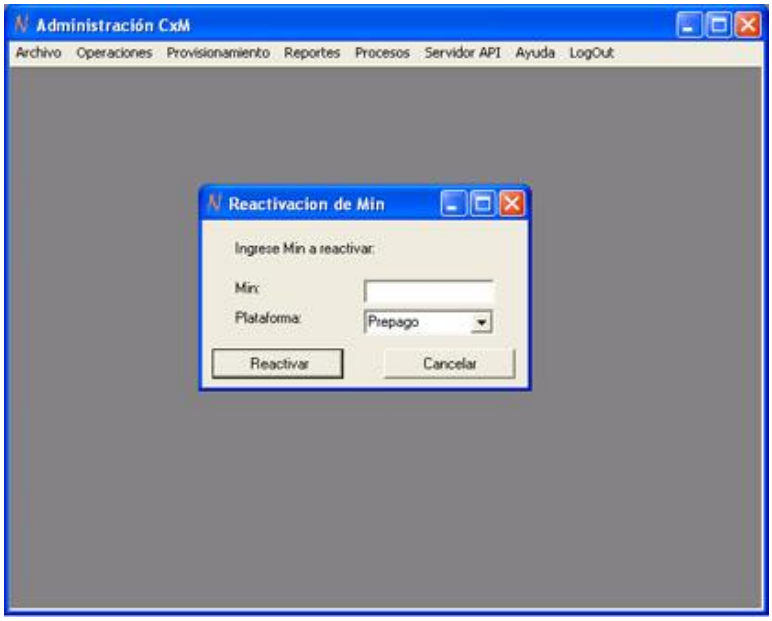


Figura A3.8

Reactivación de Provisionamiento

7. Reportes

Para monitorear el correcto funcionamiento se necesita consultar reportes que presenten estadísticas del funcionamiento del sistema.

La interfase de administración tiene dos tipos de reportes, uno individual por número de celular y otro general de todo el sistema.

El primer reporte se lo encuentra en el menú Reportes, Número de Mensajes por Min. Este reporte requiere el número de teléfono y un rango de fechas a ser analizadas. (Figura A3.9).

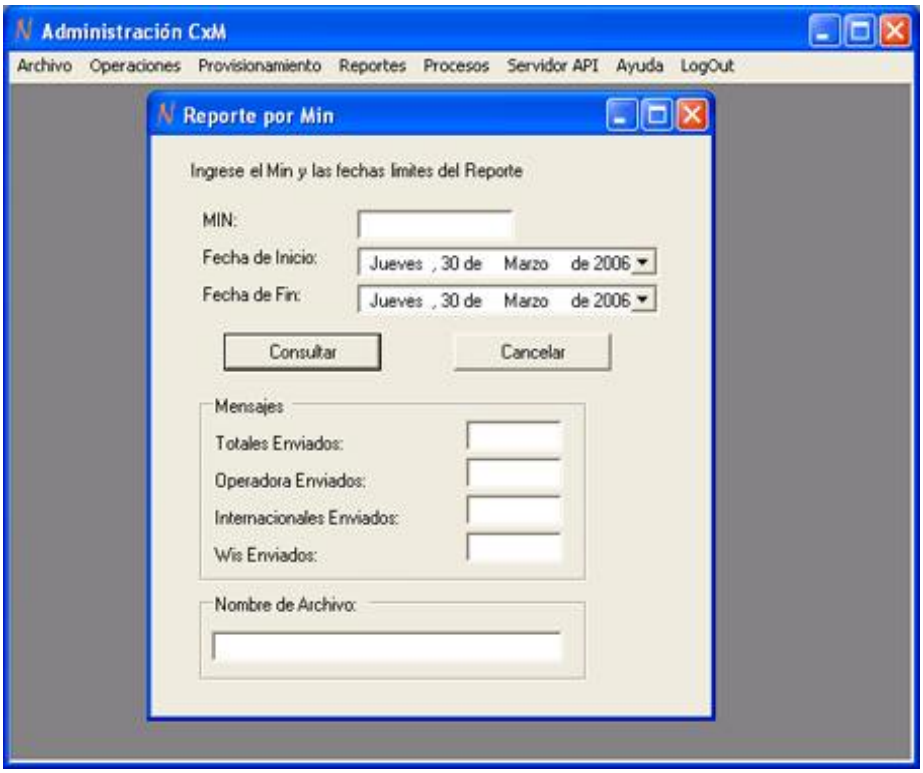


Figura A3.9

Reporte por Min

En este reporte se observa el número total de mensajes enviados por el celular en el período especificado, los mensajes que fueron dentro de la operadora, mensajes internacionales y mensajes Wis (Mensajes comerciales).

La siguiente consulta que se encuentra en Reportes es el Volumen de Mensajes. Esta consulta lista la cantidad total de mensajes enviados en un rango de fechas por todos los clientes de mensajería. (Figura A3.10).

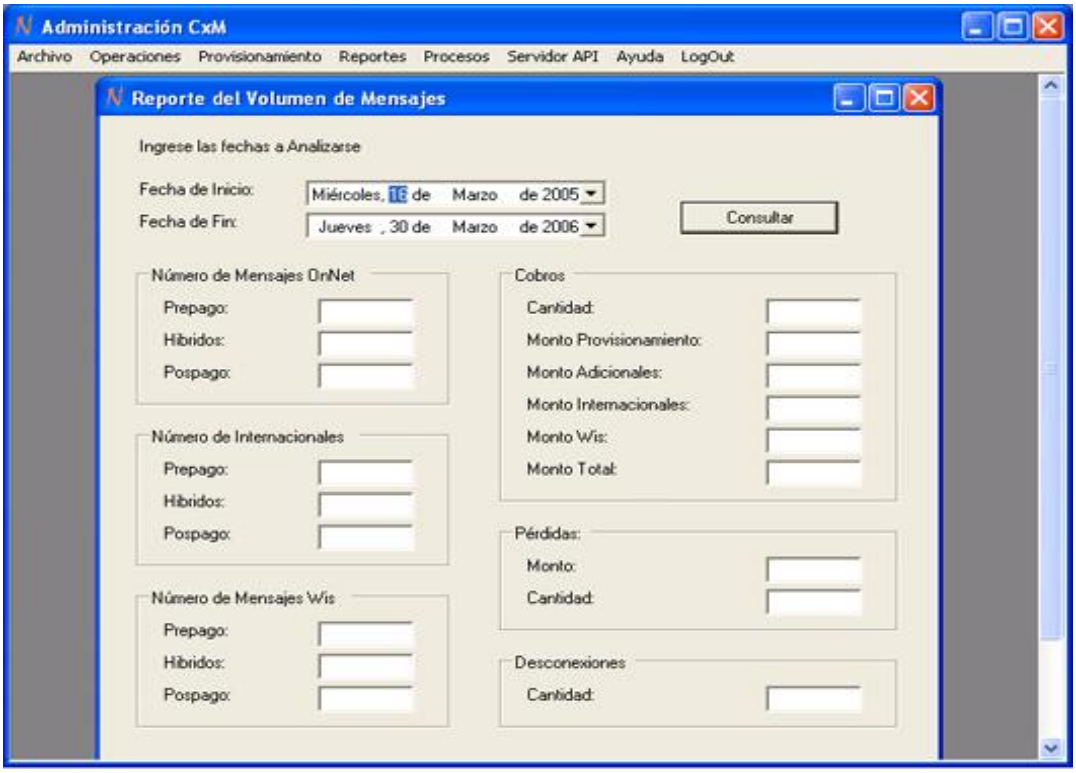


Figura A3.10

Reporte de Volumen de Mensajes

La información obtenida se divide en mensajes de la red celular nacional, mensajes internacionales, mensajes Wis, Cobros por los mensajes enviados, Pérdidas por falta de provisionamiento de servicio y desconexiones realizadas.

Este reporte es útil para determinar la productividad del sistema de mensajería.

8. Administración de Procesos

Los procesos (daemons) son pequeños programas con funciones específicas que procesan información hacia el Socke_Server. Estos procesos constan de un archivo de configuración donde se determina las características del mismo.

La interfase de Administración permite sacar un listado de los procesos relacionados con el Socke_Server. Subirlos y Bajarlos y editar el archivo de configuración.

En el menú Procesos, Consulta de Estado se lista todos los procesos relacionados con el Socket_Server y se permite Arrancar o Parar los mismos. (Figura A3.11).

Figura A3.11

Consulta de Proceso, Arranque y Parada

La otra consulta relacionada con los proceso está en Proceso, Consulta de la configuración. Esta consulta permite seleccionar un proceso de una lista y editar su archivo de configuración, ya sea añadiendo campos, manipulado los existentes o eliminando campos. (Figura A3.12).,

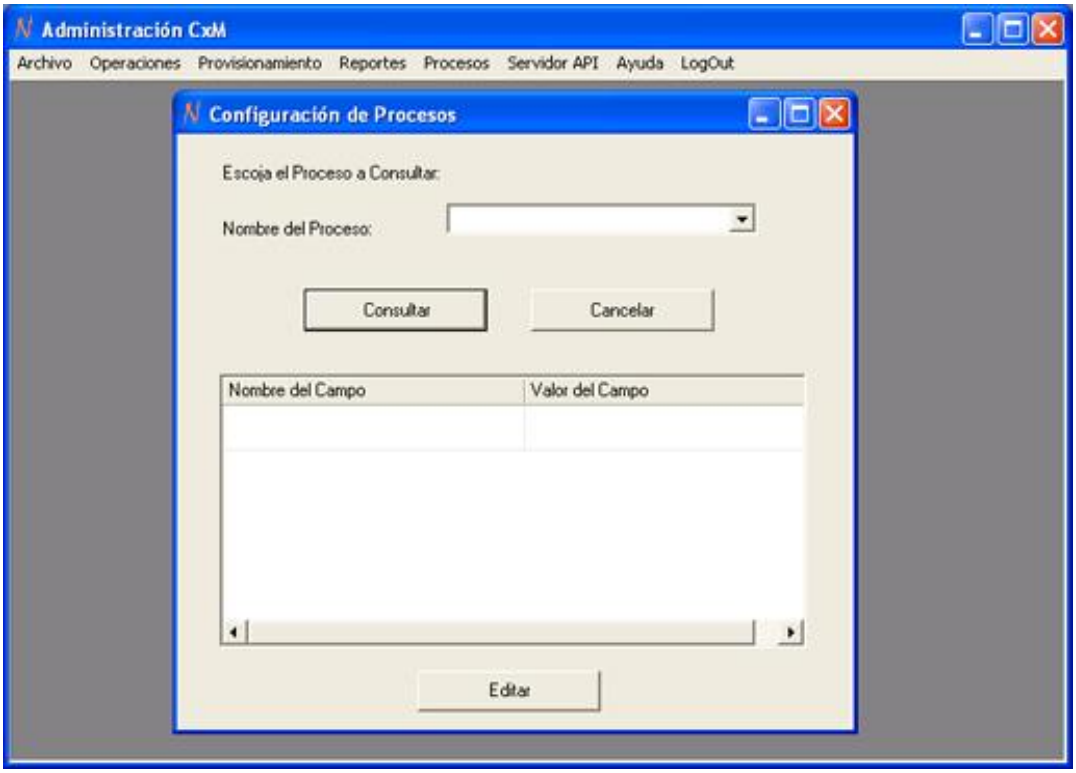


Figura A3.13

Configuración de Procesos

9. Servidor API

La última funcionalidad de la interfase es la consulta del estado del servidor Socket_Server que se encuentra en el menú Servidor API.

Esta consulta muestra el socket utilizado y si existe un socket inactivo que puede ser utilizado, como tambien el tiempo de operación.

Nota: Para manipular la configuración del Socket_Server se lo puede realizar con la consulta anterior, pero hay que notar que si se baja al Socket_Server se desactiva el servicio de Administración.

Para recibir soporte adicional sobre la interfase se tiene la opción de ayuda en el menú que proporciona los métodos de comunicación de los desarrolladores de la interfase para un soporte técnico.