Universidad de las Ciencias Informáticas

Facultad 2

Implementación de un módulo para el envío de mensajes a celulares con el componente SAM.

Trabajo de Diploma para optar por el título de Ingeniero en

Ciencias Informáticas.

Autores: María del Pilar García Martínez.

Daniel Crespo Vázquez.

Tutor: Ing. Yunier J. Trujillo Hernández.

Co-Tutor: Ing. Arianna Pérez Carmenates.

Ciudad de La Habana, Junio 2010

Declaración de Autoría

Declaro ser autor de la presente tesis, reconociendo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de manera exclusiva.

Para que así conste, firmo la presente a los ___ días del mes de _____ de _____

_____ _____

María del Pilar García Martínez

Firma Autor

Daniel Crespo Vázquez

Firma Autor

_____

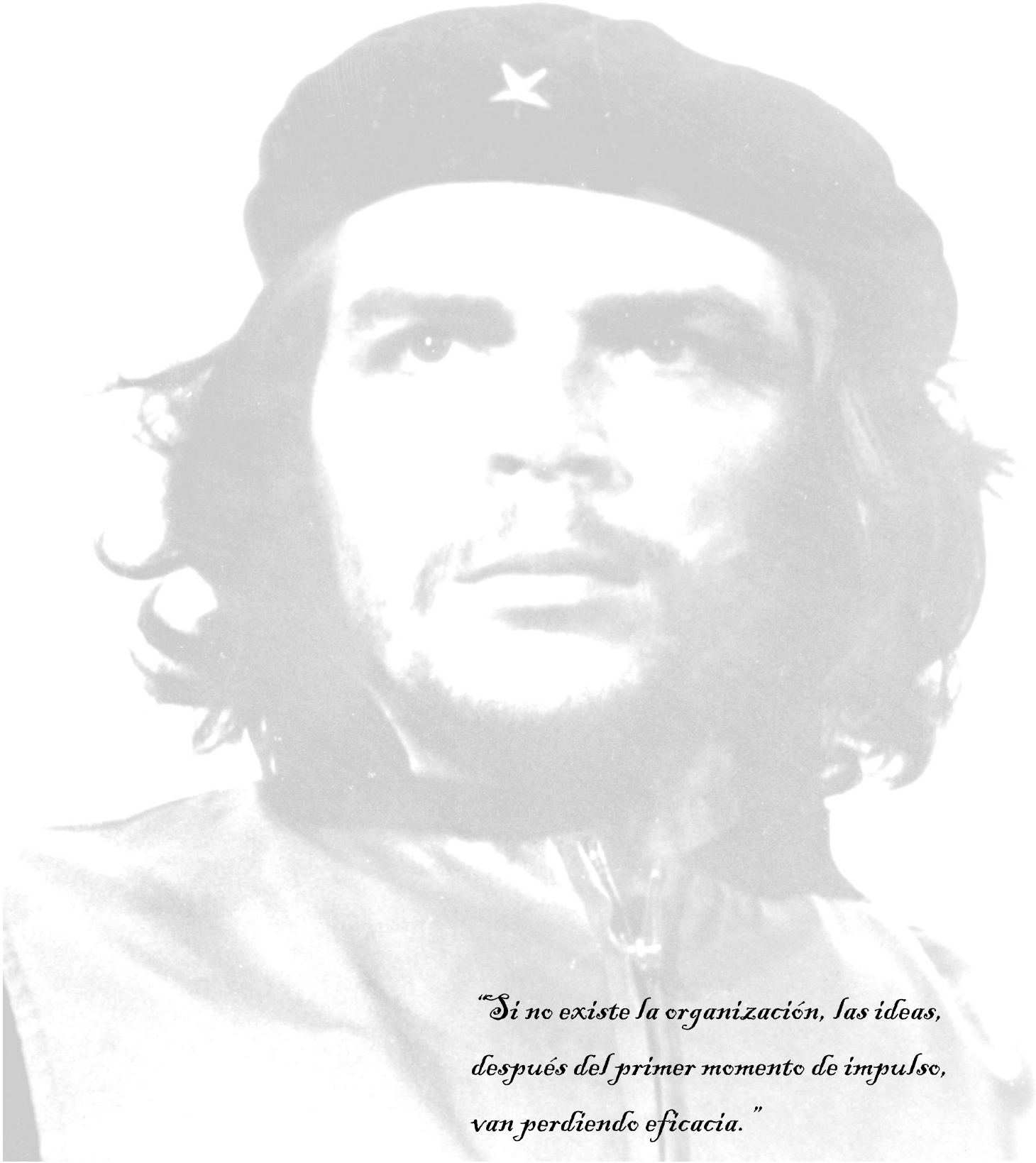
Firma del Tutor

Ing. Yunier J. Trujillo

_____

Firma del Co-Tutor

Ing. Ariana Carmenates



"Si no existe la organización, las ideas,
después del primer momento de impulso,
van perdiendo eficacia."

Dedicatoria

... a mis abuelos, principio de lo que he sido, soy y seré.

... a mis padres, presentes en cada ocasión.

... a mis profes, amigos y amigas de todos estos años de estudio.

Agradecimientos

"Cumple con la gratitud del peregrino, no olvidar nunca la fuente que apagó su sed, la palmera que le brindó frescor y sombra, y el dulce oasis donde vio abrirse un horizonte a su esperanza"

Dedico esta tesis, colofón de mis estudios de pregrado, a todas aquellas personas e instituciones que han aportado a la realización de la misma; en mi formación como Revolucionario, como ingeniero y como persona, haciendo posibles, llevaderos y profundos todos estos años de estudio.

Agradezco A la UCI, a sus profes, a su gente, por ser la casa donde vivimos y nutrimos la mente, por darnos la oportunidad de probarnos como hombres de bien.

Agradezco a mis abuelos paternos Orlando y María por inculcarme el estudio, por darme su cariño y ejemplo para con la vida.

Agradezco a mis padres Rosa María y Emilio, por confiarme las riendas, por saber escucharme y hablarme, por estar siempre presentes en los momentos que yo siempre lo he necesitado.

Agradezco a Wendy, por consentirme y ser mi hermana.

Agradezco a mis amigos de todos estos años, a los de verdad, a los que han quedado, presentes siempre, a los del pre, a los de la UCI, a Luis Ramón Rodríguez, Yordanis Furet Álvarez, Adrian Hernández Ayeja porque me han ayudado a que esta tesis se realizara.

Agradezco a mi tutor Yunier por el interés y el apoyo, y a mi cotutora Arianna porque ha sido como mi tutora, porque para la realización completa de esta tesis ella me ha dado mucha fuerza y le agradezco por la paciencia que ha tenido conmigo.

María.

Resumen

La telefonía móvil o celular se conoce desde el siglo pasado, ya en los años 70 es donde comienza a popularizarse. En la actualidad las tecnologías de mensajería inalámbrica tienen un amplio alcance en el mercado mundial, generando su utilización un gran impacto en el mismo. La permanente investigación y adelantos tecnológicos permiten enmarcar etapas bien definidas dentro de las comunicaciones móviles, denominadas generaciones, de manera que un teléfono móvil de última generación permite no solo realizar llamadas, sino reproducir música y videos, tomar fotos y pequeñas películas, navegar por internet, enviar y recibir mensajes de texto y otras opciones.

Inmersos en esta revolución económica, los sistemas de gestión de contenidos para dispositivos móviles aportan a los operadores móviles mayor capacidad al nivel de aplicación y de servicio, cubriendo las necesidades requeridas por los usuarios.

La telefonía móvil cubana también se enfrenta a los retos de la revolución tecnológica y para ello cuenta con La Universidad de las Ciencias Informáticas (UCI) que juega un papel fundamental en este sector. Para el desarrollo en este ámbito está el presente trabajo de diploma que documenta la implementación de un módulo que envíe mensajes a celulares desde un ordenador utilizando un API. Dicho módulo surge a partir de la necesidad de crearle una interfaz al componente Service API for Mobile Services (SAM), para el envío de mensajes a celulares.

Aunque existen aplicaciones con funcionalidades similares a las expuestas, tal es el caso de Hola Tu! com, creasms.com, digitel.com.ve o movilnet.com.ve es muy difícil encontrar alguna que integre el API.

Palabras Claves.

Celulares, SAM, envío, mensajes.

Introducción ..... - 12 -
Capítulo1: Fundamentación Teórica..... - 15 -
1 Introducción..... - 15 -
1.1Conceptos vinculados al campo de acción. ..... - 15 -
1.2Terminal móvil. ..... - 15 -
1.3 Multimedia Messaging System o Sistema de mensajería multimedia, (MMS). ..... - 15 -
1.4Global System for Mobile Communications o Sistema Global para las Comunicaciones Móviles (GSM). ..... - 16 -
1.5 Mensajería instantánea, (IM)..... - 16 -
1.6Servicios. ..... - 17 -
1.6.1Short Messages Standard o Servicios de mensajes cortos (SMS). ..... - 17 -
1.6.2 Sistema Universal de Telecomunicaciones Móviles (UMTS)..... - 17 -
1.7Herramientas a utilizar. ..... - 18 -
1.7.1Lenguaje de programación PHP. ..... - 18 -
1.7.2Lenguaje de programación Java. ..... - 19 -
Independencia de la plataforma. ..... - 20 -
1.8 Sistemas de Gestores de Base Datos. ..... - 20 -
1.8.1Sistema Gestor de Base de Datos propuesto. ..... - 21 -
Características de PostgreSQL ..... - 22 -
1.9Herramientas de Desarrollo. ..... - 23 -
1.9.1 Dreamweaver8. ..... - 23 -
1.9.2 Editor de Código PHP Zend Studio. ..... - 23 -
1.9.3 Framework ..... - 25 -
1.10 Metodología de desarrollo. ..... - 26 -
1.10.1 Extreme Programming (XP) ..... - 26 -
1.10.2 Por qué se escogió Extreme Programming. ..... - 28 -
1.11. Servidor WEB. ..... - 28 -
1.11.2 AppServ..... - 29 -
1.12 Conclusiones. ..... - 29 -
Capítulo 2. Exploración y Planificación. ..... - 30 -
2.1 Introducción..... - 30 -
2.2 Propuesta del Sistema. ..... - 30 -

2.3Flujo de procesos del sistema propuesto. - 30 -
2.4 Personas relacionadas con el sistema. - 30 -
Tabla 1 Personas relacionadas con el sistema. - 30 -
2.5 Fase de exploración. - 31 -
2.5.1 Historias de usuarios (HU). - 31 -
2.5.1.1Las UH son clasificadas de la siguiente forma: - 31 -
Tabla 2 Enviar Mensajes. - 32 -
Tabla 3 Eliminar Mensajes - 33 -
Tabla 4 Reenviar Mensajes - 34 -
Tabla 5 Autenticar. - 35 -
Tabla 6 Adicionar usuario. - 36 -
Tabla 7 Eliminar usuario. - 37 -
Tabla 8 Mostrar Reportes - 38 -
Tabla 9 Adicionar Operadora. - 40 -
Tabla 10 Eliminar Operadora. - 41 -
Tabla 11 Actualizar Operadora. - 42 -
2.6 Fase de Planificación. - 43 -
2.6.1 Estimación de esfuerzo por Historias de Usuario. - 43 -
Tabla 12 Estimación de esfuerzo por HU. - 43 -
2.6.2 Plan de Iteraciones. - 44 -
2.6.3 Plan de duración de las iteraciones. - 45 -
Tabla 13 Plan de duración de iteraciones. - 45 -
2.6.4 Plan de entregas. - 45 -
Tabla 14 Plan de Entregas. - 45 -
2.7 Conclusiones. - 46 -
Capitulo 3.Diseño, implementación y Prueba. - 46 -
3.1 Introducción. - 46 -
3.2 Patrón de Arquitectura Modelo Vista Controlador. - 46 -
3.3 Fase de Diseño. - 47 -
3.3.1 Diseño de Diagrama de clase del Sistema. - 48 -
3.3.2 Diseño de base de datos. - 49 -
3.4 Fase de Implementación. - 50 -

3.4.1 Primera iteración. - 51 -
Tabla 15 Historias de Usuario desarrolladas en la primera iteración. - 51 -
3.4.1.1 Tareas de las historias de usuario desarrolladas en la primera iteración. - 51 -
Tabla 16 Tarea 1 de la historia de usuario Enviar Mensajes. - 51 -
Tabla 17 Tarea 2 de la historia de usuario Eliminar Mensaje. - 51 -
Tabla 18 Tarea 3 de la historia de usuario Reenviar Mensaje. - 52 -
3.4.2 Segunda iteración. - 52 -
Tabla 19 Historias de Usuario desarrolladas en la segunda iteración. - 52 -
3.4.2.1 Tareas de las historias de usuario desarrolladas en la segunda iteración. - 53 -
Tabla 20 Tarea 4 de la historia de usuario Autenticar. - 53 -
Tabla 21 Tarea 4 de la historia de usuario Adicionar Usuario. - 53 -
Tabla 22 Tarea 4 de la historia de usuario Eliminar Usuario. - 53 -
3.4.3 Tercera iteración. - 54 -
Tabla 23 Historias de Usuario desarrolladas en la tercera iteración. - 54 -
3.4.3.1 Tareas de las historias de usuario desarrolladas en la tercera iteración. - 54 -
Tabla 24 Tarea 5 de la historia de usuario Mostrar Reportes. - 54 -
Tabla 25 Tarea 7 de la historia de usuario Adicionar Operadora. - 55 -
Tabla 26 Tarea 8 de la historia de usuario Eliminar Operadora. - 55 -
Tabla 27 Tarea 9 de la historia de usuario Actualizar Operadora. - 55 -
3.5 Fase de Prueba. - 56 -
3.5.1 Pruebas Unitarias. - 57 -
3.5.2 Pruebas de Aceptación. - 57 -
Tabla 28 Pruebas de aceptación para la historia de usuario Enviar Mensaje. - 58 -
Tabla 29 Pruebas de aceptación para la historia de usuario Reenviar Mensajes. - 59 -
Tabla 30 Pruebas de aceptación para la historia de usuario Eliminar Mensajes. - 59 -
3.8 Conclusiones. - 60 -
CAPÍTULO 4 ESTUDIO DE FACTIBILIDAD. - 61 -
4.1Introducción. - 61 -
4.2 Modelo matemático COCOMO II. - 61 -
4.3Características del Proyecto. - 62 -
4.3.1 Entradas externas. - 62 -
Tabla 31 Entradas externas. - 62 -

4.3.2 Salidas Externas. - 62 -
Tabla 32 Salidas externas. - 63 -
4.3.3 Consultas Externas. - 63 -
Tabla 33 Consultas externas. - 63 -
4.3.4 Archivos Lógicos Internos. - 63 -
Tabla 34 Fichero lógico interno. - 63 -
4.3.5 Archivos de Interfaz Externos. - 64 -
Tabla 35 Interfaces externas - 64 -
4.3.6 Puntos de Función Desajustados. - 64 -
Tabla 36 Puntos de función desajustados. - 64 -
4.4 Cálculos de instrucciones fuentes, esfuerzo, tiempo de desarrollo, cantidad de hombres y costo. - 65 -
4.4.1 Cálculo de instrucciones fuentes - 65 -
Tabla37 Características del sistema. - 65 -
4.4.2 Cálculo de esfuerzo nominal. - 66 -
4.4.2.1 Precedentes (PREC). - 67 -
4.4.2.2 Flexibilidad de desarrollo (FLEX). - 67 -
4.4.2.3 Cohesión del equipo (TEAM). - 67 -
4.4.2.4 Solución de riesgos (RESL). - 67 -
4.4.2.5 Madurez del proceso (PMAT). - 67 -
Tabla 38 Factores de escala. - 67 -
4.5 Ajuste del esfuerzo nominal. - 68 -
4.5.1Clasificados en categorías - 68 -
Tabla 39 Multiplicadores de esfuerzo. - 69 -
4.5.2 Calculo del tiempo de desarrollo del software. - 70 -
4.5.3 Resultados. - 72 -
Tabla 40 Resultados. - 72 -
4.6 Análisis de costo. - 72 -
4.7 Conclusiones. - 72 -
Conclusiones Generales. - 73 -
Recomendaciones. - 74 -
BIBLIOGRAFÍAS. - 75 -

Referencia Bibliográfica. ..... - 77 -
Anexo ..... - 78 -
Anexo I. Pruebas de Aceptación. ..... - 78 -
Anexo III. Tablas de Cocomo II. ..... - 82 -

Introducción

Desde su surgimiento las tecnologías de telefonía móvil se manifestaron como un innegable paso de avance en la esfera de las comunicaciones, acogido su uso por un gran número de usuarios y respaldado su desarrollo por varias empresas existentes y otras surgidas con este propósito.

En las comunicaciones, la telefonía móvil ha sido una de las tecnologías de gran desarrollo en la última década por las facilidades que brinda. En estos días se continúa incrementando sus potencialidades, llegando incluso estas a diferir un tanto del objetivo principal de su creación: la transmisión de voz, pero manteniendo el concepto de servir como medio de comunicación. Entre estas potencialidades cuenta hoy con la posibilidad de brindar servicios de amplio acceso a las redes de computadoras, incluyendo la red de Internet, lo cual implica una gran cadena de productores de contenidos, proveedores de servicio y fabricantes de equipos, así como una nueva percepción de la rentabilidad apoyada en la gestión e inversión en la infraestructura, obligando a seguir una serie de estándares de calidad y seguridad que derivan en una mejor relación productor-consumidor.

De esta forma, los servicios que se brindan han llegado a convertirse en una prestación básica para los usuarios, a consecuencia de las mejoras que brinda en términos de conectividad, movilidad, disminución de costo, rapidez, facilidad de despliegue, aumento de los servicios y calidad de los mismos.

Todos estos servicios que se brindan tienen como resultado que las líneas móviles hayan ido suplantando las líneas fijas tradicionales, a la vez que los productores de terminales van incorporando a los teléfonos celulares tecnologías que permiten la transmisión de datos, dando a su vez la posibilidad del uso de móviles para brindar nuevos servicios.

Cuba a pesar de ser un país bloqueado, con limitaciones económicas y tecnológicas trata de desarrollarse en este sector que es tan importante y tan competitivo a nivel internacional y para ello cuenta con varias instituciones, un ejemplo de estas instituciones es La Universidad de las Ciencias Informáticas (UCI).

La UCI juega un papel fundamental en este campo ya que promueve la utilización de las tecnologías de informática y telecomunicaciones, para desarrollar procesos de formación, investigación y producción. Cada año la universidad fomenta nuevos proyectos para darle solución a las necesidades que afronta la rama de las telecomunicaciones. En el centro de telemática de la facultad 2, se desarrollan proyectos para desarrollar el perfil de esta facultad. Platel Comunicaciones Unificadas es un ejemplo, el cual integra mensajería instantánea, fax y correo.

Entre los diversos productos para la mejora de la calidad de los servicios de la telefonía celular, se encuentra Service API for Mobile Services (SAM), un componente para el recibo y envío de mensajes desde móviles. El componente SAM tiene la limitante de no tener una interfaz para el envío de mensajes a celulares desde un ordenador. Por tanto a partir de la necesidad de la UCI de crear un software que resuelva la situación imperante surge la exigencia de integrar el componente SAM con una interfaz web.

A partir de la situación problémica anteriormente planteada, se deriva el siguiente **Problema científico:** ¿Cómo garantizar el envío de mensajes a celulares desde un ordenador utilizando SAM?

El **Objeto de estudio** del presente trabajo es el proceso de envío de mensajes a celulares desde un ordenador orientando a, **Campo de acción** al proceso automatizado de envío de mensajes a celulares desde un ordenador usando el componente SAM en la Universidad de las Ciencias Informáticas.

Por tanto se trazó como **Objetivo general** la implementación de un sistema que posibilite el envío de mensajes a celulares desde un ordenador utilizando el componente SAM, dando lugar a los siguientes **objetivos específicos:**

- ✓ Implementación de una aplicación de tipo servidora que maneje el flujo de actividades relacionado con el envío de mensajes a celulares por parte de los usuarios.
- ✓ Implementación de una interfaz Web que posibilite llevar a cabo el proceso de envío de mensajes a celulares
- ✓ Desarrollo de una interfaz Web amigable que permita consultar la gestión de los mensajes, gestión de las operadoras y gestión de los usuarios, por parte de los usuarios del sistema.

La **Idea a defender** es que con la creación del módulo se permitirá enviar mensajes desde un ordenador a un celular con la integración del componente SAM.

Para dar cumplimiento a los objetivos propuestos se definen las siguientes **Tareas de investigación**.

- ✓ Investigar el funcionamiento de los sistemas de mensajería de celulares, para tener una base para la creación del módulo futuro.
- ✓ Analizar las metodologías de investigación.
- ✓ Estudiar el arte sobre las metodologías de desarrollo para la creación de software, tecnologías y herramientas actuales referentes al objetivo general.
- ✓ Identificar las principales características asociadas a las aplicaciones web.

- ✓ Estudiar la arquitectura del componente SAM
- ✓ Analizar la factibilidad del proyecto.
- ✓ Estudiar los patrones de arquitectura, en especial el patrón Modelo Vista Controlador.
- ✓ Estudiar las metodologías de desarrollo de software, en especial XP.
- ✓ Estudiar los Sistemas Gestores de Bases de Datos más utilizados en la actualidad, en especial Postgres.

Con el cumplimiento de todas estas tareas se tendrá como resultado el diseño de un módulo que permitirá enviar mensajes desde un ordenador a un celular utilizando el componente SAM.

Este trabajo consta de cuatro capítulos en los cuales se describe todo el proceso por el que transita la investigación.

Capítulo 1 Fundamentación Teórica: Contiene una descripción del proceso que será automatizado y un breve estudio de los principales conceptos tratados en el trabajo. Se exponen las principales características de las tecnologías y herramientas seleccionadas para la solución del problema actual.

Capítulo 2 Exploración y Planificación: Se enfatiza en las características del sistema a desarrollar, haciendo un análisis de los flujos de trabajo involucrados en el campo de acción y el objeto de automatización de la propuesta. Se detallan los artefactos generados durante las fases de exploración y planificación del proyecto.

Capítulo 3. Diseño, Implementación y Pruebas: Este capítulo está referido a las fases de diseño, implementación y prueba propias de la metodología de desarrollo utilizada.

Capítulo 4. Estudio de Factibilidad: Se lleva a cabo un estudio de factibilidad del sistema.

Capítulo1: Fundamentación Teórica.

1 Introducción

En este capítulo se profundizan algunos conceptos fundamentales para el desarrollo de un módulo que permita el envío de mensajes a celulares desde un ordenador con el componente SAM.También se exponen las principales características de las tecnologías y herramientas propuestas para la solución del problema actual, así como la justificación de la selección de las mismas.

1.1Conceptos vinculados al campo de acción.

Con el estudio realizado se han encontrado algunos conceptos relacionados con el campo de acción, que son de vital importancia para un mejor entendimiento del contenido de la investigación. A continuación se presentan los más importantes.

1.2Terminal móvil.

Es un dispositivo inalámbrico electrónico que permite tener acceso a la red de telefonía celular o móvil. Existen redes telefónicas móviles satelitales: su principal característica es su portabilidad, pues permite la comunicación desde cualquier lugar. Aunque su principal función es la comunicación de voz, su rápido desarrollo ha incorporado otras funciones como son cámara fotográfica, agenda, acceso a Internet, reproducción de vídeo e incluso correo electrónico.

La terminal móvil tiene presente una red de telefonía móvil que está formada por dos grandes partes: una red de comunicaciones (red de telefonía móvil) y los terminales (teléfonos móviles) que permiten el acceso a dicha red.

1.3 Multimedia Messaging System o Sistema de mensajería multimedia, (MMS).

Es un estándar de mensajería que le permite a los teléfonos móviles enviar y recibir contenidos multimedia, incorporando sonido, video, fotos o cualquier otro contenido disponible en el futuro. La mensajería multimedia permite el envío de estos contenidos a cuentas de correo electrónico, ampliando las posibilidades de la comunicación móvil, pudiendo publicar fotografías digitales. El límite de cada mensaje multimedia suele ser de 100 o 300 KB, dependiendo de cada móvil, si bien ese límite lo definen el operador o las características del terminal.

Los mensajes MMS constituyen una nueva extensión de la mensajería móvil para incluir, en un principio, imágenes, sonidos reales, posteriormente vídeos y en el futuro virtualmente cualquier

tipo de contenido que pueda mostrarse en un teléfono móvil. En el estándar no se define, además, una limitación de tamaño máximo, gran diferencia con respecto a los mensajes SMS (que admiten un máximo de 160 caracteres de 7 bits por mensaje), si bien muchos operadores limitan su tamaño máximo por cuestiones prácticas.

En principio, el estándar MMS es independiente del tipo de transporte y red móvil empleada y sólo define un sistema de sincronización del contenido, el Lenguaje de integración multimedia sincronizada, Multimedia Messaging System (SMIL) y un modo de encapsulado de los mensajes. Dependiendo del tipo de red, se emplea un método de aviso al terminal de usuario cuando hay mensajes nuevos y un camino de transporte sobre la red radio, que no tienen por que ser el mismo en todas las redes. Así se permite que usuarios móviles de distintos tipos de redes puedan enviarse mensajes multimedia sin problemas, ya que el estándar final empleado para el transporte da igual.

1.4Global System for Mobile Communications o Sistema Global para las Comunicaciones Móviles (GSM).

Es un sistema estándar, completamente definido para la comunicación mediante teléfonos móviles que incorporan tecnología digital. Por ser digital cualquier cliente de GSM puede conectarse a través de su teléfono con su computador y puede enviar y recibir mensajes por e-mail, fax, navegar por Internet y tener acceso seguro a la red informática de una compañía (LAN/Intranet), así como utilizar otras funciones digitales de transmisión de datos, incluyendo el Servicio de Mensajes Cortos o Short Message Service (SMS).

Este estándar GSM ha sido una ventaja en Cuba, que se introdujo en el año 2001 y en el que se trabaja por ser un estándar abierto que no necesita pago de licencias. Dicho país ofrece servicios de telefonía celular con cobertura nacional en las normas del estándar GSM. La misma opera en las frecuencias 850/900 MHZ en la tecnología GSM. La banda de los 850MHZ (GSM).

1.5 Mensajería instantánea, (IM).

Es una forma de comunicación en tiempo real entre dos o más personas basada en texto. El texto es enviado a través de dispositivos conectados a una red como Internet.

La mensajería instantánea requiere el uso de un cliente de mensajería instantánea que realiza el servicio y se diferencia del correo electrónico en que las conversaciones se realizan en tiempo

real. La mayoría de los servicios ofrecen el aviso de presencia, indicando cuando el cliente de una persona en la lista de contactos se conecta, el estado de la misma o si está disponible para tener una conversación. En los primeros programas de mensajería instantánea, cada letra era enviada según se escribía y así las correcciones también se veían en tiempo real, esto daba más la sensación de una conversación telefónica que un intercambio de texto.

En los programas actuales como Hola Tu! COM, creasms.com, digitel.com.ve y movilnet.com.ve, habitualmente, se envía cada frase de texto al terminarse de escribir. Además, en algunos, también se permite dejar mensajes aunque la otra parte no esté conectada al estilo de un contestador automático, pero estas páginas no utilizan el Sam como se utiliza en esta interfaz.

Hay programas de mensajería instantánea que utilizan el protocolo abierto Jabber, con un conjunto descentralizado de servidores.

Los clientes de mensajería instantánea más utilizados son ICQ, Yahoo! Messenger, Windows Live Messenger, Pidgin, AIM (AOL Instant Messenger) y Google Talk (que usa el protocolo abierto Jabber). Estos servicios han heredado algunas ideas del viejo, aunque aún popular, sistema de conversación. Cada uno de estos mensajeros permite enviar y recibir mensajes de otros usuarios usando los mismos software clientes, sin embargo, últimamente han aparecido algunos clientes de mensajería que ofrecen la posibilidad de conectarse a varias redes al mismo tiempo aunque necesitan registrar un usuario distinto en cada una de ellas. También existen programas que ofrecen la posibilidad de conectarte a varias cuentas de usuario a la vez.

1.6Servicios.

1.6.1Short Messages Standard o Servicios de mensajes cortos (SMS).

Short Message Service es un servicio disponible en redes digitales, teléfonos móviles, teléfonos fijos y otros dispositivos de manos, permitiendo enviar y recibir mensajes de texto hasta 160 caracteres. Fue diseñado originalmente como parte del estándar de telefonía móvil digital de GSM, en la actualidad está disponible en una amplia variedad de redes, incluyendo las redes de Sistema Universal de Telecomunicaciones Móviles.

1.6.2 Sistema Universal de Telecomunicaciones Móviles (UMTS).

UMTS se caracterizó por la convergencia de la voz y datos con acceso inalámbrico a Internet, aplicaciones multimedia y altas transmisiones de datos. Los protocolos empleados en estos sistemas soportan altas velocidades de información enfocados para aplicaciones más allá de la

voz, tales como audio Mp3, video en movimiento, video conferencia y acceso rápido a Internet, sólo por nombrar algunos. Las redes de 3G empezaron a operar en el 2001 en Japón, Europa y parte de Asia en el 2002, posteriormente en Estados Unidos y otros países.
Los sistemas 3G alcanzan velocidades de hasta 384 Kbps permitiendo una movilidad total a usuarios viajando a 120 Km/h en ambientes exteriores y consiguen una velocidad máxima de 2 Mbps permitiendo una movilidad limitada a usuarios caminando a menos de 10 Km/h en ambientes estacionarios de corto alcance o en interiores. Entre las tecnologías contendientes de la tercera generación se encuentran UMTS, CDMA2000, IMT-2000, HSPA, entre otras.

1.7Herramientas a utilizar.

1.7.1Lenguaje de programación PHP.

Este lenguaje es usado principalmente en interpretación del lado del servidor, pero actualmente puede ser utilizado desde una interfaz de línea de comandos, o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica.

Fue creado originalmente por Rasmus Lerdorf en 1994; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal.

Se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. PHP se encuentra instalado en muchos sitios web y también en servidores.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite.

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como UNIX (y de ese tipo, como Linux o Mac OS X) y Windows, y puede interactuar con los servidores Web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

El gran parecido que posee PHP con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy corta. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones.

Aunque todo en su diseño está orientado a facilitar la creación de páginas Web, es posible crear aplicaciones con una interfaz gráfica para el usuario, utilizando la extensión PHP-Qt o PHP-GTK. También puede ser usado desde la línea de órdenes, de la misma manera como Perl o Python pueden hacerlo, a esta versión de PHP se la llama PHP CLI (Command Line Interface).

Debido a la serie de ventajas enunciadas anteriormente y además por petición del cliente se utilizó PHP en la elaboración del módulo.

1.7.2Lenguaje de programación Java.

El lenguaje Java se creó con cinco objetivos principales:

- ✓ Usar la metodología de la programación orientada a objetos.
- ✓ Permitir la ejecución de un mismo programa en múltiples sistemas operativos.
- ✓ Incluir por defecto soporte para trabajo en red.
- ✓ Diseño para ejecutar código en sistemas remotos de forma segura.
- ✓ Ser fácil uso y tomar lo mejor de otros lenguajes orientados a objetos, como C++.

A continuación se mencionan algunas características del lenguaje.

Orientado a Objetos.

La programación orientado a objetos (“OO”), se refiere a un método de programación y al diseño del lenguaje. El objetivo es hacer que grandes proyectos sean fáciles de gestionar y manejar, mejorando como consecuencia su calidad y reduciendo el número de proyectos fallidos. Otra de las grandes promesas de la programación orientada a objetos es la creación de entidades más genéricas (objetos) que permitan la reutilización del software entre proyectos, una de las premisas fundamentales de la Ingeniería del Software. Un objeto genérico “cliente”, por ejemplo, debería en teoría tener el mismo conjunto de comportamientos en diferentes proyectos, sobre todo cuando estos coinciden en cierta medida, algo que suele suceder en las grandes organizaciones.

En este sentido, los objetos podrían verse como piezas reutilizables que pueden emplearse en múltiples proyectos, posibilitando así a la industria del software construir proyectos de envergadura empleando componentes ya existentes y de comprobada calidad; conduciendo esto finalmente a una reducción drástica del tiempo de desarrollo.

La reutilización del software ha experimentado distintos resultados, encontrando dos dificultades principales: el diseño de objetos realmente genéricos es pobremente comprendido, y falta una

metodología para la amplia comunicación de oportunidades de reutilización. Algunas comunidades de “código abierto” (open source) quieren ayudar en este problema dando medios a los desarrolladores para diseminar la información sobre el uso y versatilidad de objetos reutilizables y bibliotecas de objetos.

Independencia de la plataforma.

Independencia de la plataforma, significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware. Este es el significado de ser capaz de escribir un programa una vez y que pueda ejecutarse en cualquier dispositivo.

Las primeras implementaciones del lenguaje usaban una máquina virtual interpretada para conseguir la portabilidad. Sin embargo, el resultado eran programas que se ejecutaban comparativamente más lentos que aquellos escritos en C o C++. Esto hizo que Java se ganase una reputación de lento en rendimiento. Las implementaciones recientes del lenguaje dan lugar a programas que se ejecutan considerablemente más rápido que las versiones antiguas, empleando diversas técnicas, aunque sigue siendo mucho más lento que otros lenguajes. La primera de estas técnicas es simplemente compilar directamente en código nativo como hacen los compiladores tradicionales.

La re compilación dinámica puede lograr mayor grado de optimización que la compilación tradicional (o estática), ya que puede basar su trabajo en el conocimiento que de primera mano tiene sobre el entorno de ejecución y el conjunto de clases cargadas en memoria. La compilación JIT (Just In Time, o ‘”compilación al vuelo”’), y la re compilación dinámica permiten a los programas Java aprovechar la velocidad de ejecución del código nativo sin perder la ventaja de la portabilidad en ambos. (0)

Para el desarrollo de este módulo se utiliza el componente SAM, y el lenguaje de programación Java, ya que este componente esta implementado en este lenguaje y es muy importante para la parte de la integración.

1.8 Sistemas de Gestores de Base Datos.

Existen varias formas o maneras de como poder guardar información necesaria y de vital importancia, una de las más conocidas y seguras son las bases de datos.

Una BD es un conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que una BD puede considerarse una colección de datos variables en el tiempo.

El software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez, se denomina SGBD (sistema Gestor base de datos). (1)

En general los objetivos fundamentales de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora y manejar de manera clara, sencilla y ordenada, un conjunto de datos que posteriormente se convertirán en información relevante para una organización.

Los SGBD poseen ventajas, a continuación se mencionan algunas:

- ✓ Organizan los datos con un impacto mínimo en el código de los programas.
- ✓ Simplifican la programación de equipos de consistencia.
- ✓ Usualmente, proveen interfaces y lenguajes de consulta que simplifican manejo en general de los datos.

Entre los SGBD comúnmente utilizados en el mundo se tienen Oracle, MySQL, PostgreSQL, entre otros (2)

1.8.1Sistema Gestor de Base de Datos propuesto.

PostgreSQL es un gestor de base de datos de código abierto que posee una gran escalabilidad, es capaz de ajustarse al número de procesadores y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta. Implementa el uso de subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz, y ofreciendo soluciones en campos en las que MySQL no podría. Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle. (4)

Está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL

declarativas, control de concurrencia multi-versión, soporte multi-usuario, transacciones, optimización de consultas, herencia, y arreglos.

Posee alta concurrencia permitiendo que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.

Características de PostgreSQL

A continuación se ofrecen algunas características:

- ✓ Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
- ✓ Soporta gran cantidad de tipos de datos para las columnas.
- ✓ Gran portabilidad entre sistemas.
- ✓ Soporta hasta 32 índices por tabla.
- ✓ Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.

•Lenguajes Procedurales.

PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido. (4)

•Cliente/Servidor.

PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL. (5)

Se propone PostgreSQL, ya que es un SGBD libre, el cual posee una gran escalabilidad y soporta una mayor cantidad de peticiones simultáneas de manera segura. Posee además una alta concurrencia pues mediante un sistema denominado MVCC (Acceso concurrente multiversión) permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso

a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.(5)

1.9Herramientas de Desarrollo.

Las herramientas de programación sirven para soportar los diferentes lenguajes de programación para los cuales estén destinadas. Unas presentan interfaz gráfica, otras no las tienen, otras son para realizar aplicaciones desktop, otras para hacerlas Web, etc.

Se tomaran como herramientas de desarrollo el Dreamweaver en su versión 8 y Zend Studio en su versión 7.1.0. A continuación se exponen algunas de las características de estas herramientas:

1.9.1 Dreamweaver8.

Es la opción profesional para crear sitios Web y aplicaciones, dado que proporciona una potente combinación de herramientas visuales de diseño, funciones de desarrollo de aplicaciones y soporte para la edición del código, características que permiten a los desarrolladores y diseñadores con más o menos experiencia crear rápidamente sitios Web y aplicaciones basados en estándares. Desde el avanzado soporte de diseño basado en CSS a las funciones de codificación manual, Dreamweaver proporciona las herramientas profesionales que requiere un entorno integrado y agilizado. Los desarrolladores pueden utilizar Dreamweaver con su tecnología de servidor preferida para crear potentes aplicaciones en Internet destinadas a conectar a los usuarios a las bases de datos, las fuentes de datos dinámicos y los sistemas heredados.

Se ha creado a partir de las versiones anteriores, pero se ha ampliado su funcionalidad en ámbitos esenciales como XML y CSS. Dreamweaver es compatible con todas las principales tecnologías de servidor como, por ejemplo, ColdFusion, PHP, ASP, ASP.NET y JSP, para que los desarrolladores puedan dar vida a sus diseños. (6)

1.9.2 Editor de Código PHP Zend Studio.

Debido al incremento de la necesidad del uso de aplicaciones Web y el ascenso de la complejidad de estas, se hace necesaria la utilización de leguajes de programación que las doten de nuevas potencialidades. En el caso de la aplicación que se propone el lenguaje de programación que se utilizará será PHP, y aunque este lenguaje puede ser fácilmente utilizado desde editores de texto se hace necesaria la utilización de un editor especializado en dicho lenguaje, en este caso Zend Studio. Zend Studio es una poderosa herramienta para el trabajo con aplicaciones Web usando el lenguaje de programación PHP. Además de servir de editor de texto para páginas PHP,

proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código.

Zend Studio o Zend Development Environment es un completo entorno de desarrollo integrado para el lenguaje de programación PHP. Está escrito en Java, y está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux.

Junto con su contraparte Zend Platform, son la propuesta de Zend Technologies para el desarrollo de aplicaciones Web utilizando PHP, actuando Zend Studio como la parte cliente y Zend Platform como la parte servidora. Se trata en ambos casos de software comercial, lo cual contrasta con el hecho de que PHP es software libre. (7)

Dentro de sus características posee:

- ✓ No requiere la instalación previa de PHP ni del entorno de ejecución de Java.
- ✓ Soporte para PHP 4 y PHP 5.
- ✓ Resaltado de sintaxis, autocompletado de código, ayuda de código y lista de parámetros de funciones y métodos de clase.

PhpDoc integrado.

- ✓ Inserción automática de paréntesis y corchetes de cierre.
- ✓ Detección de errores de sintaxis en tiempo real.
- ✓ Funciones de depuración: Botón de ejecución y traza, marcadores, puntos de parada (breakpoints), seguimiento de variables y mensajes de error del intérprete de PHP. (7)
- ✓ Soporte para gestión de grandes proyectos de desarrollo.

El programa en conjunto está implementado en Java, lo que a veces supone que no funcione tan rápido como otras aplicaciones de uso diario. Sin embargo, esto ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para Windows, Linux y MacOS, aunque el desarrollo de las versiones de este último sistema se retrase un poco más.

Lo más notable es que contiene una ayuda contextual con todas las librerías de funciones del lenguaje que asiste en todo momento, ofreciendo nombres de las funciones y parámetros que deben recibir. Aunque esta ayuda contextual no sólo se queda en las funciones definidas en el lenguaje, sino que también reporta ayudas con las funciones que se vayan creando, incluso en páginas que estén incluidas con la función include, otras ayudas que ofrece a la hora de escribir

son las típicas en editores avanzados, como permitir editar varios archivos, y moverse fácilmente entre ellos, marcar a qué elementos corresponden los inicios y cierres de las etiquetas, paréntesis o llaves, moverse al principio o al final de una función, identificación automática del código. (7)

Las mejores cualidades de Zend Studio no se encuentran en la parte de edición sino en las posibilidades de gestión de proyectos y depuración. La barra de la izquierda, que permite navegar los archivos, también dispone de herramientas para gestionar los proyectos, muy útiles para mejorar la productividad en la programación. Los proyectos permiten guardar mucha más información al programa sobre los archivos, discos, servidores, etc. que se gestionen en aplicaciones PHP.

Una vez que los archivos se han añadido al proyecto se pueden guardar señales como puntos de ruptura en las depuraciones, asimismo, cuando se ejecuta Zend Studio, se vuelven a abrir los archivos que estuvieran abiertos la última vez que el programa se cerró y las herramientas de completar código mejoran sus comportamientos, asumiendo toda la información de los archivos relacionados con el proyecto.

Zend Studio implementa además unas interesantes opciones para trabajar en grupo, al integrar el sistema de trabajo conocido como Control Versión System (CVS). Sin duda, más de una vez los programadores de PHP se han visto en un duro problema por no encontrar un error en algún script que está dando resultados inesperados, para facilitar el trabajo, Zend Studio dispone de una herramienta muy interesante de debug o depuración. Gracias a ella se pueden ejecutar páginas y conocer en todo momento el contenido de las variables de la aplicación y las variables del entorno como las cookies, las recibidas por formulario o en la sesión. Se pueden colocar puntos de parada de los scripts y realizar las acciones típicas de depuración.

Además de la ventana para visualizar el contenido de las variables, dispone de otras donde muestra la salida del script según se va generando, así como las alertas y errores. Las posibilidades se completan con distintos tipos de depuración, en local, en remoto o a partir de una URL. (7)

Debido a las numerosas facilidades que ofrece este editor se decidió utilizarlo como editor de código para la implementación de la aplicación propuesta.

1.9.3 Framework

Codeigniter es un framework PHP de gran alcance, construido para programadores de PHP que necesitan una guía sencilla y elegante para crear aplicaciones web con todas las funciones.

Codeigniter es un entorno de desarrollo abierto que permite crear webs dinámicas con PHP. Su principal objetivo es ayudar a que los desarrolladores puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero. El uso de Codeigniter se basa en el patrón Modelo-Vista-Controlador (MVC), lo que facilita mucho estructurar la aplicación y su mantenimiento. El manejo y abstracción de base de datos es realmente bueno, e implementa el patrón Active Record. La división en modelos, vistas, controladores, librerías y configuración, es muy práctica y comprensible. Los mecanismos de ruteo de Codeigniter favorecen a éste y otras aplicaciones. Es un framework con gran flexibilidad. No fuerza al usuario a utilizar una Base de Datos para un desarrollo. Además su separación de código PHP y código HTML no está basada en un template en particular, puede usarse incluso sin este. Es un framework Multiplataforma.

1.10 Metodología de desarrollo.

1.10.1 Extreme Programming (XP)

En el desarrollo de aplicaciones, es sin dudas de vital importancia la selección de una metodología de desarrollo. Actualmente existe un sin número de éstas, siendo imposible determinar una que satisfaga en su totalidad las necesidades de un proyecto de desarrollo en específico; de esta forma, la selección de una metodología de desarrollo debe llevarse a cabo teniendo en cuenta diferentes puntos que influyen en el desarrollo del sistema, entre estos: recursos técnicos, humanos y tiempo de desarrollo esperado o disponible.
Básicamente consiste en trabajar estrechamente con el cliente, haciendo pequeñas iteraciones (mini-entregas), cada dos semanas, donde no existe más documentación que el código en si; cada versión contiene las modificaciones necesarias según el cliente vaya retroalimentando el sistema (por eso es necesaria la disponibilidad del cliente durante todo el desarrollo).

Para suplir la falta de requisitos, casos de uso, y demás herramientas; XP utiliza historias de usuarios, la historia de usuario es una frase corta que representa alguna función que realizara el sistema. Cada historia de usuario no puede demorar en desarrollarse más de una semana, si así lo requiera, debe segmentarse. (8)

Es requisito para XP definir un estándar en el tipo de codificación, esto hace que los programadores tengan definido ya el estilo de programación y no que cada uno programe a su estilo.

Los programadores trabajan en parejas (dos por computadora) intercambiándose el tiempo de trabajo en la máquina, esta interesante forma de trabajar tiene ventajas como:

- ✓ Detecta más fácilmente los errores de programación (el programador libre controla al que tipea);
- ✓ El programador poco experimentado aprende del que más lo está.
- ✓ Si una pareja consigue desarrollar algún trozo de código reutilizable, se comunica más fácilmente a los otros programadores.
- ✓ El testing en cada iteración es más que importante; de eso se trata este paradigma de programación, corregir mientras se programa. De esta forma se van cubriendo todos los baches que cada versión padezca.
- ✓ El código no es de nadie, todo el equipo puede manipular el código que existe, de esta forma cada pareja puede mejorar cada sección de código que utiliza, esto requiere de un testing del mismo y la re-implementación en el sistema general.

¿Qué es lo que propone XP?

- ✓ Empieza en pequeño, o sea, de forma simple y sin mucha complejidad y añade funcionalidad con retroalimentación continua.
- ✓ El manejo del cambio se convierte en parte sustantiva del proceso.
- ✓ El costo del cambio no depende de la fase o etapa.
- ✓ No introduce funcionalidades antes que sean necesarias.
- ✓ El cliente o el usuario se convierte en miembro del equipo.

Cada cierto tiempo se entrega una versión al cliente, que lo verifica, realiza el feedback y se continúa el desarrollo; este ciclo continua hasta que el sistema cumpla con las expectativas del cliente.

No existe documentación del proyecto lo que más se acerca a la documentación son las historias de usuario, pero al concluir el proyecto se descartan. Inclusive se recomienda hacer dos secciones, una con todas las historias de usuario que faltan desarrollar, y otra donde se archiven las concluidas, esto aproximara el estado de avance del proyecto. (9)

1.10.2 Por qué se escogió Extreme Programming.

Se decidió utilizar XP debido a que se adapta en gran medida tanto al tipo de proyecto a desarrollar como a las condiciones de trabajo. A continuación se exponen varias de las razones que llevaron al uso de esta metodología.

El proyecto es pequeño. XP está concebida para ser utilizada dentro de proyectos pequeños.

No existe un contrato previo especificando tiempo, recursos y alcance. Para el desarrollo del sistema no se dispone de un contrato con un presupuesto ni un alcance previamente definidos, puesto que es un proyecto para el uso interno de la UCI y será llevado a cabo por programadores pertenecientes a la misma.

Los requisitos del sistema cambian frecuentemente. Con la aceptación de nuevos tipos de proyectos, con estructura y requerimientos disímiles, el sistema debe cambiar y ampliar sus funcionalidades de forma que sea capaz de adaptarse a cada nueva situación. Uno de los principios básicos de XP es que el cambio frecuente de los requerimientos es algo normal en el proceso de desarrollo. Esta metodología se adapta perfectamente a los proyectos cuyos requerimientos cambian a menudo (10)

El cliente forma parte del equipo de desarrollo. Mediante la aplicación de XP se puede lograr una retroalimentación mayor y lograr un producto que satisfaga sus necesidades.

Debido al corto tiempo de entrega planteado y a los continuos cambios de requerimientos, XP está diseñada a mitigar los riesgos en proyectos con estas características (10)

1.11. Servidor WEB.

1.11.1. Apache Tomcat.

Para el desarrollo del sistema se usará apache-tomcat-6.0.26. Tomcat es un servidor Web con soporte de servlets, el cual incluye el compilador Jasper, que compila convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor Apache. El Tomcat puede funcionar como servidor Web por sí mismo, es usado como servidor Web autónomo en entornos con alto nivel de tráfico y alta disponibilidad. Dado que Tomcat fue desarrollado en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java. Este servidor es utilizado para elaborar el war del SAM.

1.11.2 AppServ

Es una herramienta Open Source para Windows que facilita la instalación de PHP en la cual estas aplicaciones se configuran en forma automática. Esta herramienta se utilizada para publicar el sitio web, la interfaz. (11)

1.12 Conclusiones.

Como se pudo apreciar en este capítulo se definieron los principales conceptos a tratar en el resto del documento. También se abordaron las características referentes a la metodología de desarrollo escogida y se realizó un estudio de las tecnologías existentes en el mundo actual y por último se justificó la selección de cada una de las herramientas para el desarrollo del software.

Capítulo 2. Exploración y Planificación.

2.1 Introducción.

El presente capítulo permitirá conocer el contexto en el que se expone la solución propuesta. Se describirán los procesos existentes u observados con el objetivo de comprenderlos y se especificarán aquellos que soportará el sistema. Además guiará el desarrollo del software hacia el sistema adecuado, mediante la captura y descripción de las condiciones o capacidades que el sistema debe cumplir. También se hace alusión a la fase de exploración y planificación propias de la metodología de desarrollo utilizada para la implementación del sistema que se propone.

2.2 Propuesta del Sistema.

La solución que se propone estará diseñada para La Universidad de las Ciencias Informáticas y para cualquier empresa u organismo que quiera brindar este servicio a través del SAM. Se propone la puesta en marcha de un sistema que envíe mensajes a celulares con el componente SAM. El módulo administrará una base de datos en la cual se guardarán los mensajes, los números de teléfono de los destinatarios y la operadora por la que se ha enviado dichos mensajes, una vez guardados estos mensajes el Sam, se ocupará de enviarlos hacia los destinatarios correspondientes.

2.3Flujo de procesos del sistema propuesto.

El Usuario accede a la página principal del módulo que le permite realizar una serie de acciones como son: escoger la operadora por la cual el va a mandar su mensaje, introducir el número de teléfono, escribir su mensaje y enviarlo. El módulo Web se encarga de procesar esta información originada guardando todo estos datos introducidos por el usuario, en una Base de Datos incluyendo así la fecha en que se envían estos mensajes. Luego estos datos serán cogidos por el componente y enviados a sus destinatarios.

2.4 Personas relacionadas con el sistema.

Se define como persona relacionada al sistema toda aquella que obtiene un resultado del valor de uno o varios procesos que se ejecutan en el mismo.

Tabla 1 Personas relacionadas con el sistema.

Personas Relacionada con el Sistema.	Justificación.

Usuario anónimo.	Es la persona que navega en el sistema, envía sus mensajes sin tener que autenticarse, ni crearse una cuenta.
Administrador.	Es la persona encargada de ver los reportes del sistema y de administrar todo lo referente del módulo.

2.5 Fase de exploración.

La metodología de desarrollo Extreme Programming comienza con la fase de exploración. En esta fase se define el alcance general del proyecto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto, también se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

En esta fase, el cliente define lo que necesita mediante la redacción de sencillas “historias de usuarios”. Los programadores estiman los tiempos de desarrollo en base a esta información. Debe quedar claro que las estimaciones realizadas en esta fase son primarias, ya que estarán basadas en datos de muy alto nivel, y podrían variar cuando se analicen más en detalle en cada iteración.

2.5.1 Historias de usuarios (HU).

La identificación de las historias de usuario (HU) es la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

2.5.1.1Las UH son clasificadas de la siguiente forma:

Teniendo en cuenta la Escala Nominal de Prioridad en el Negocio:

Alta: Se le otorga a las Historias de Usuario que resultan funcionalidades fundamentales en el desarrollo del proyecto, a las que el cliente define como principales para el control integral de proyectos.

Media: Se le otorga a las Historias de Usuario que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación directa sobre el proyecto que se esté desarrollando.

Baja: Se le otorga a las Historias de Usuario que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura y no tienen nada que ver con el proyecto en desarrollo.

Teniendo en cuenta la Escala Nominal de Riesgo en Desarrollo:

Alta: Cuando la implementación de las historias de usuario se consideran las posible existencia de errores que lleven la inoperatividad del código.

Media: Cuando pueden aparecer errores en la implementación de la UH que puedan retrasar la entrega de la versión.

Baja: Cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan prejuicios para el desarrollo del proyecto.

A continuación se muestran las UH que serán utilizadas en el desarrollo de la aplicación propuesta y que reflejan la información brindada por el cliente.

Tabla 2 Enviar Mensajes.

Historia de Usuario	
Número:*1*	**Nombre Historia de Usuario:** Enviar Mensajes.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:** 1
Prioridad en Negocio: Alta.	**Puntos Estimados:** 3/7
Riesgo en Desarrollo: Alto.	**Puntos Reales:**1
Descripción: El usuario inserta el número de teléfono al que va enviar su mensaje y lo envía.	

Prototipo de interfaz:

Tabla 3 Eliminar Mensajes

Historia de Usuario	
Número:*2*	**Nombre Historia de Usuario:** Eliminar Mensajes.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:** 1
Prioridad en Negocio: Alta.	**Puntos Estimados:** 4/7
Riesgo en Desarrollo: Baja.	**Puntos Reales:** 1

Descripción: El administrador elimina de BD los mensajes que están guardados desde hace un mes.

Prototipo de interfaz:

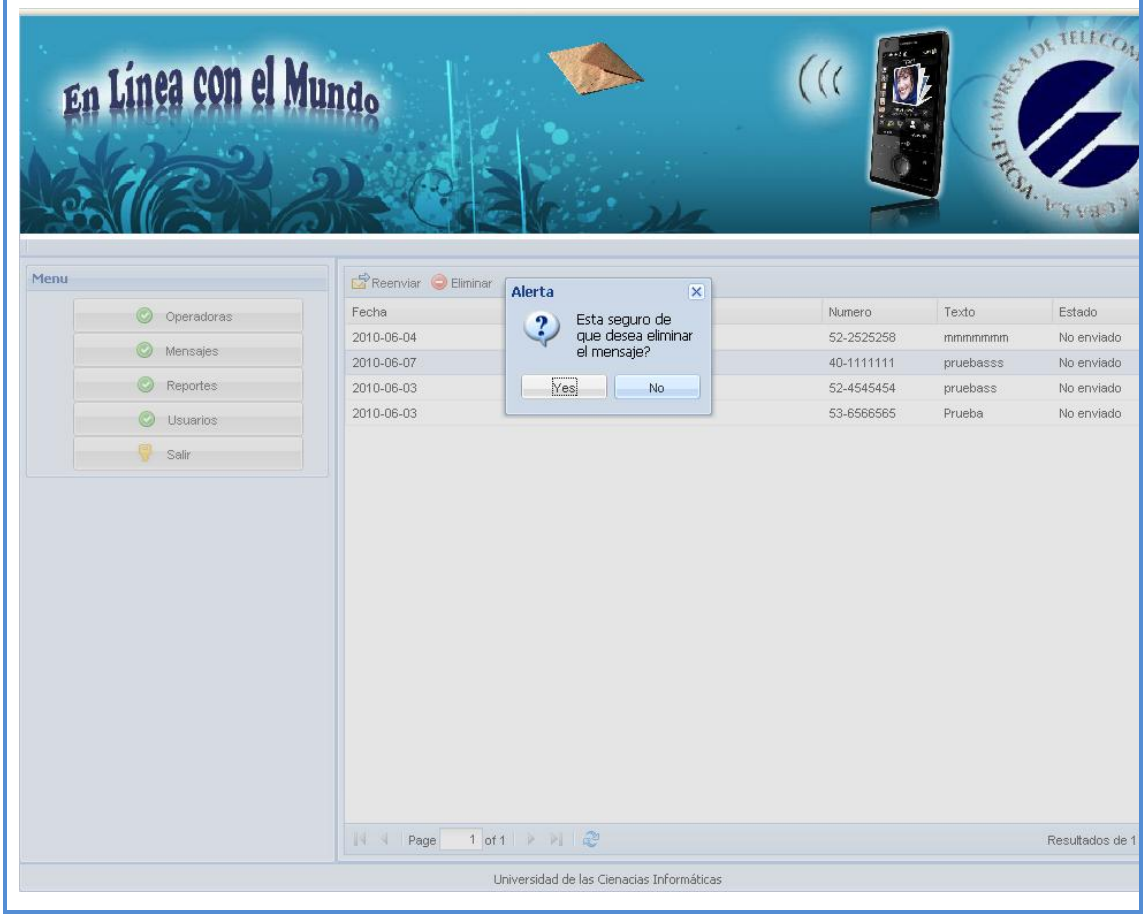


Tabla 4 Reenviar Mensajes

Historia de Usuario	
Número:*3*	**Nombre Historia de Usuario:** Reenviar Mensajes.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:** 1
Prioridad en Negocio: Alta.	**Puntos Estimados:** 1

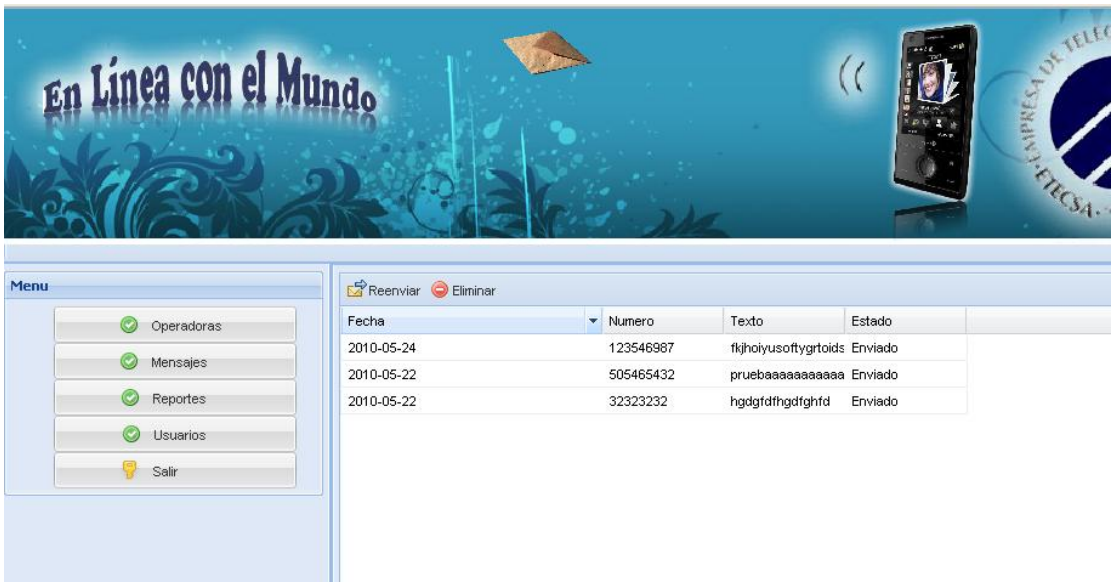
Riesgo en Desarrollo: Baja.	**Puntos Reales:** 1
Descripción: El usuario envía su mensaje, el administrador al ver que ese mensaje no se ha ido reenvía ese mensaje y si no se va lo elimina de la base de datos.	
Prototipo de Interfaz: 	

Tabla 5 Autenticar.

Historia de Usuario	
Número:4	**Nombre Historia de Usuario:** Autenticar
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:**1
Prioridad en Negocio: Media	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alta.	**Puntos Reales:** 1
Descripción: El administrador es el usuario que se va autenticar para ver todo relacionado con el sistema.	

Prototipo de Interfaz:

Tabla 6 Adicionar usuario.

Historia de Usuario	
Número:4	**Nombre Historia de Usuario:** Adicionar Usuario
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:**1

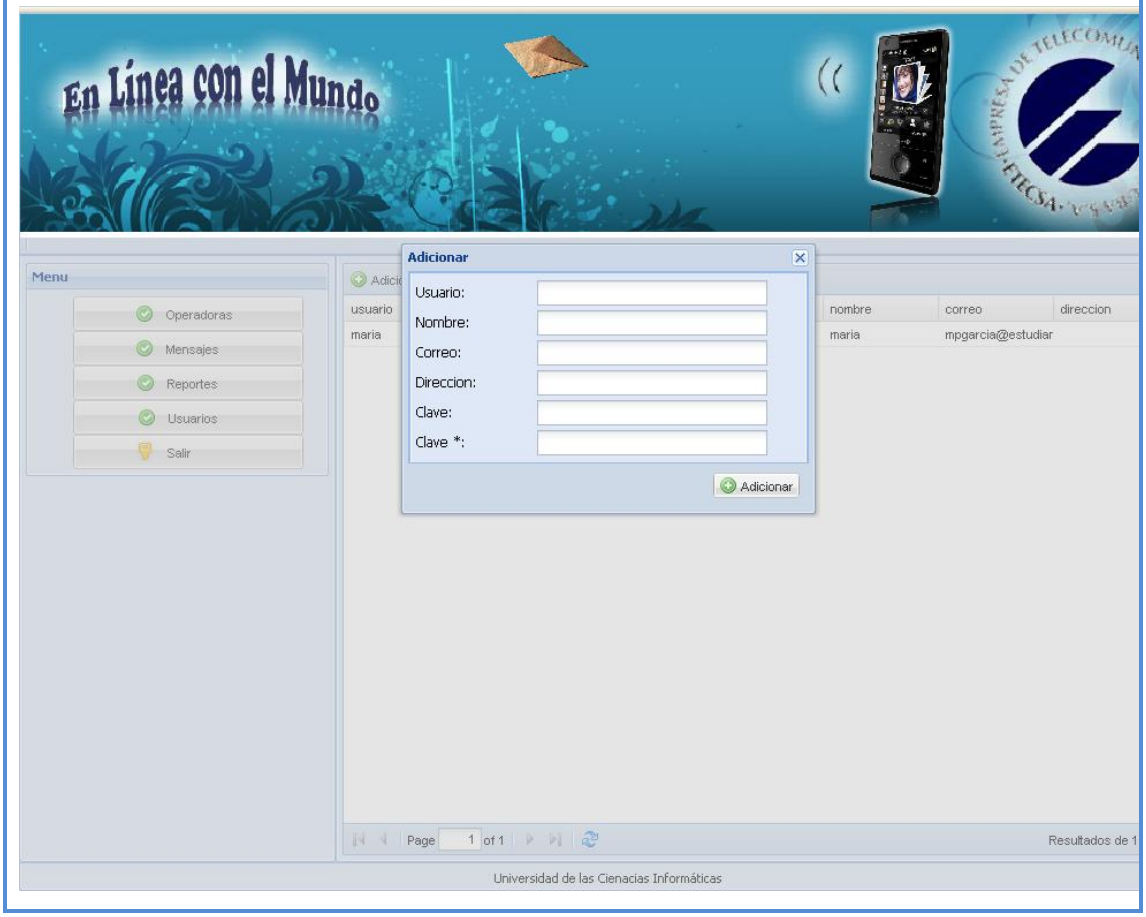
Prioridad en Negocio: Media	**Puntos Estimados:** 4/7
Riesgo en Desarrollo: Alta.	**Puntos Reales:** 1
Descripción: El administrador puede adicionar cualquier usuario para que sea administrador del sistema.	
Prototipo de Interfaz: 	

Tabla 7 Eliminar usuario.

Historia de Usuario	
Número:4	**Nombre Historia de Usuario:** Eliminar Usuario
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:**1

Prioridad en Negocio: Media	**Puntos Estimados:** 3/7
Riesgo en Desarrollo: Alta.	**Puntos Reales:** 1
Descripción: El administrador puede eliminar a cualquier usuario que este insertado en el sistema.	
Prototipo de Interfaz:	

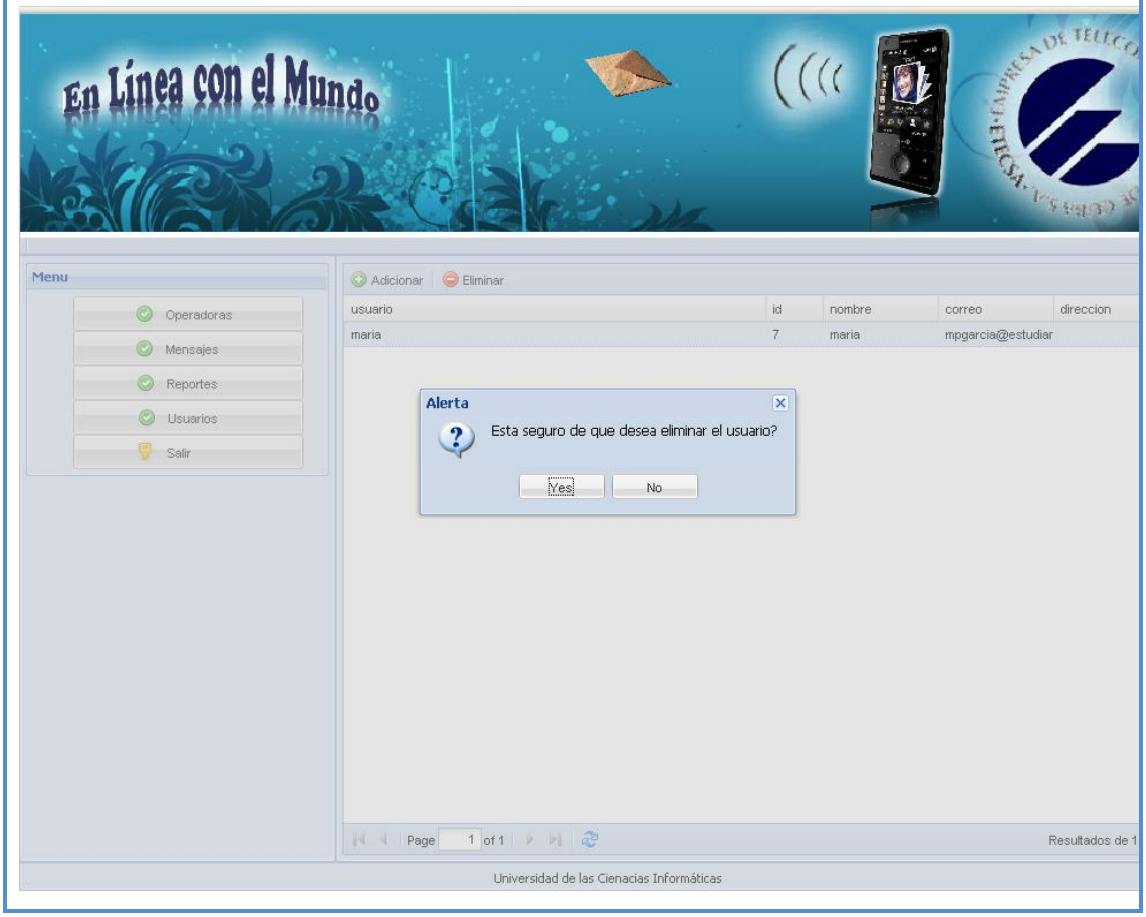


Tabla 8 Mostrar Reportes

Historia de Usuario	
Número:5	**Nombre Historia de Usuario:** Mostrar Reportes
Modificación de Historia de Usuario Número: Ninguna.	

Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:**2
Prioridad en Negocio: Alta.	**Puntos Estimados:** 1
Riesgo en Desarrollo: Baja.	**Puntos Reales:** 1
Descripción: El administrador puede ver los mensajes que se han enviado en las 24horas como en los 15 días.	
Prototipo de Interfaz:	

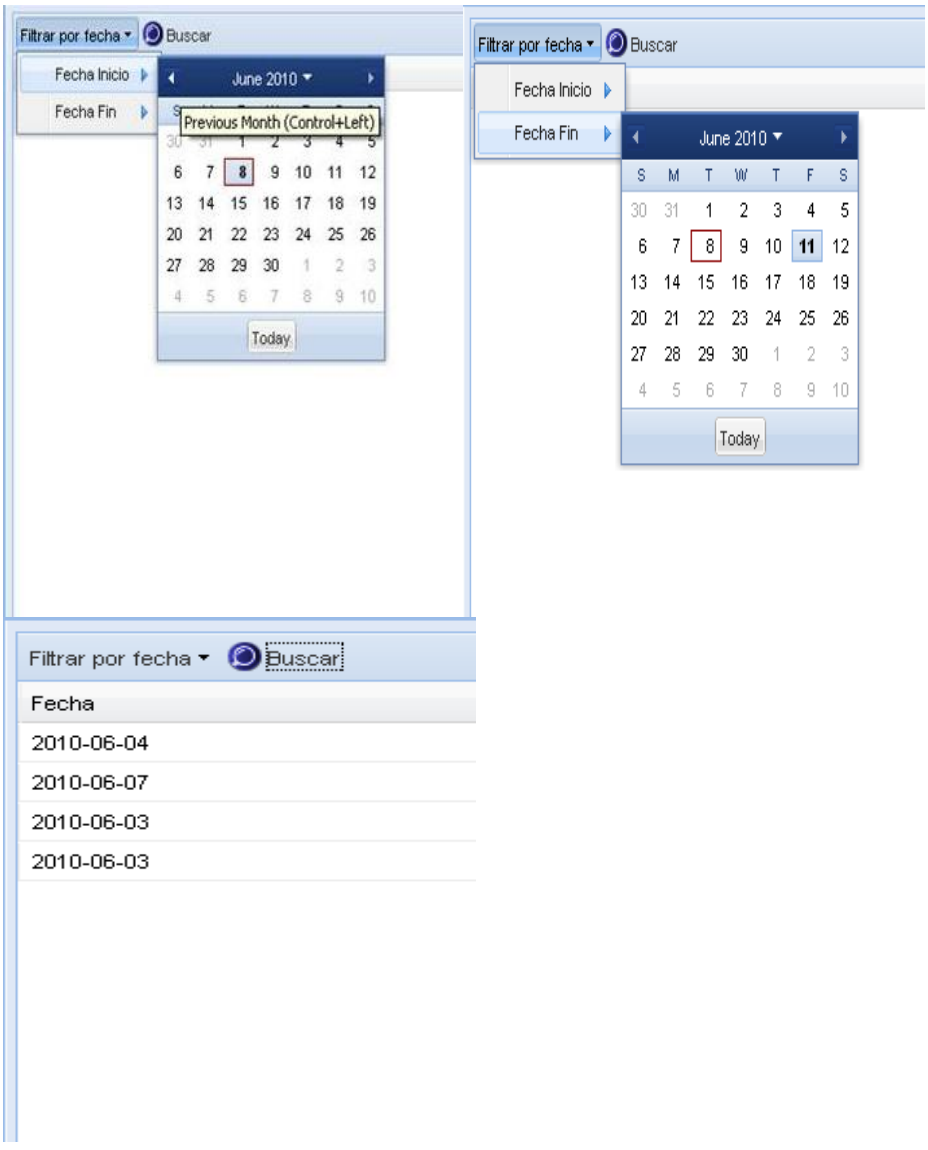


Tabla 9 Adicionar Operadora.

Historia de Usuario	
Número:*6*	**Nombre Historia de Usuario:** Adicionar Operadora.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:** 1
Prioridad en Negocio: Alta.	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alto.	**Puntos Reales:**1
Descripción: El Administrador puede adicionar cualquier operadora, que no esté presente en la lista de operadoras.	
Prototipo de Interfaz:	

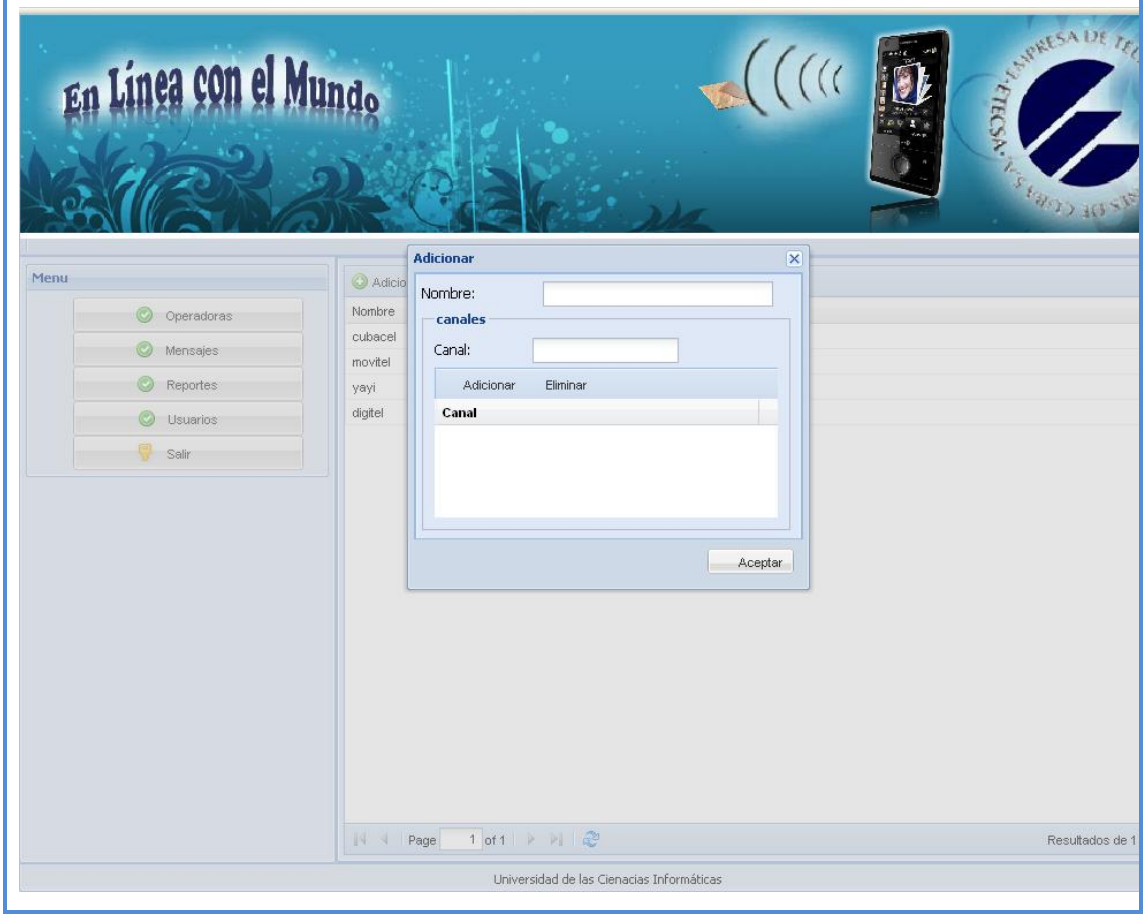


Tabla 10 Eliminar Operadora.

Historia de Usuario	
Número: 7	**Nombre Historia de Usuario:** Eliminar Operadora.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:** 1
Prioridad en Negocio: Alta.	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alto.	**Puntos Reales:** 1
Descripción: El Administrador puede eliminar cualquier operadora, que esté presente en la lista de operadoras.	
Prototipo de Interfaz:	

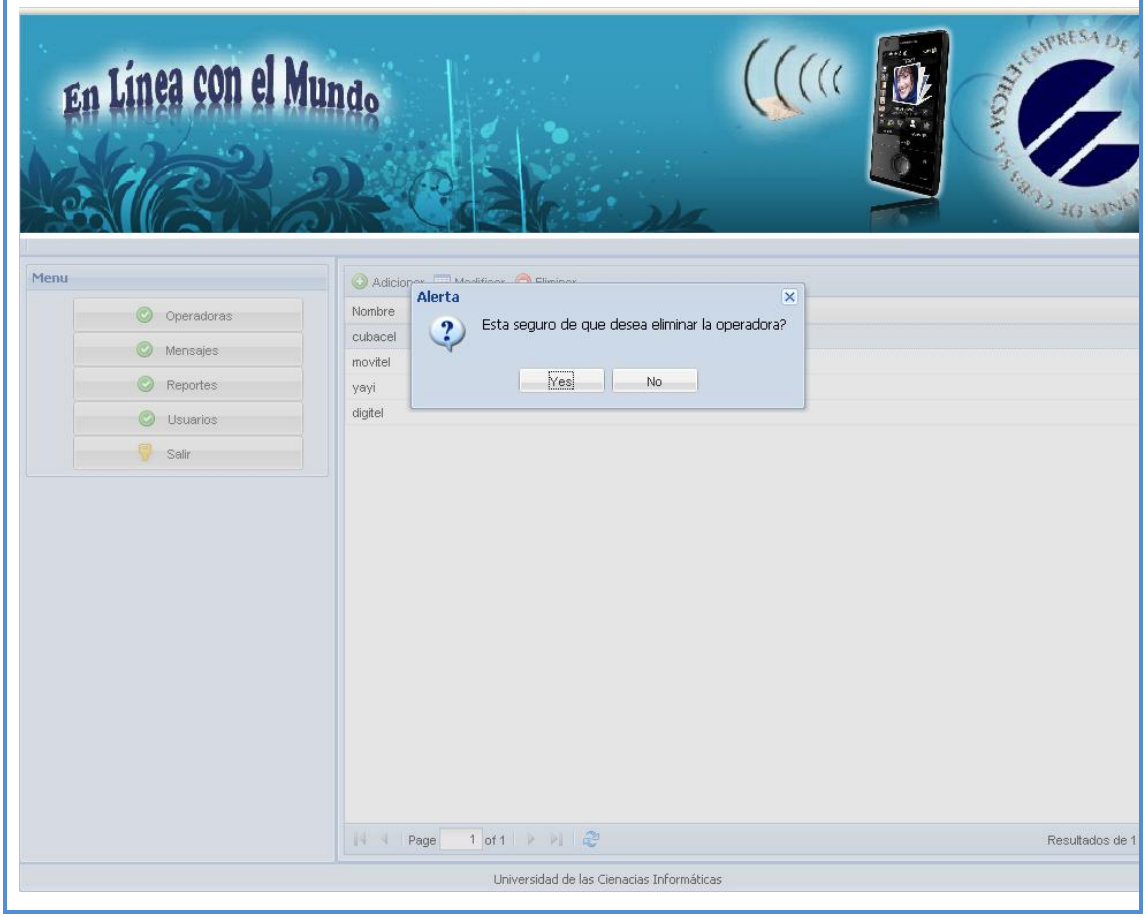
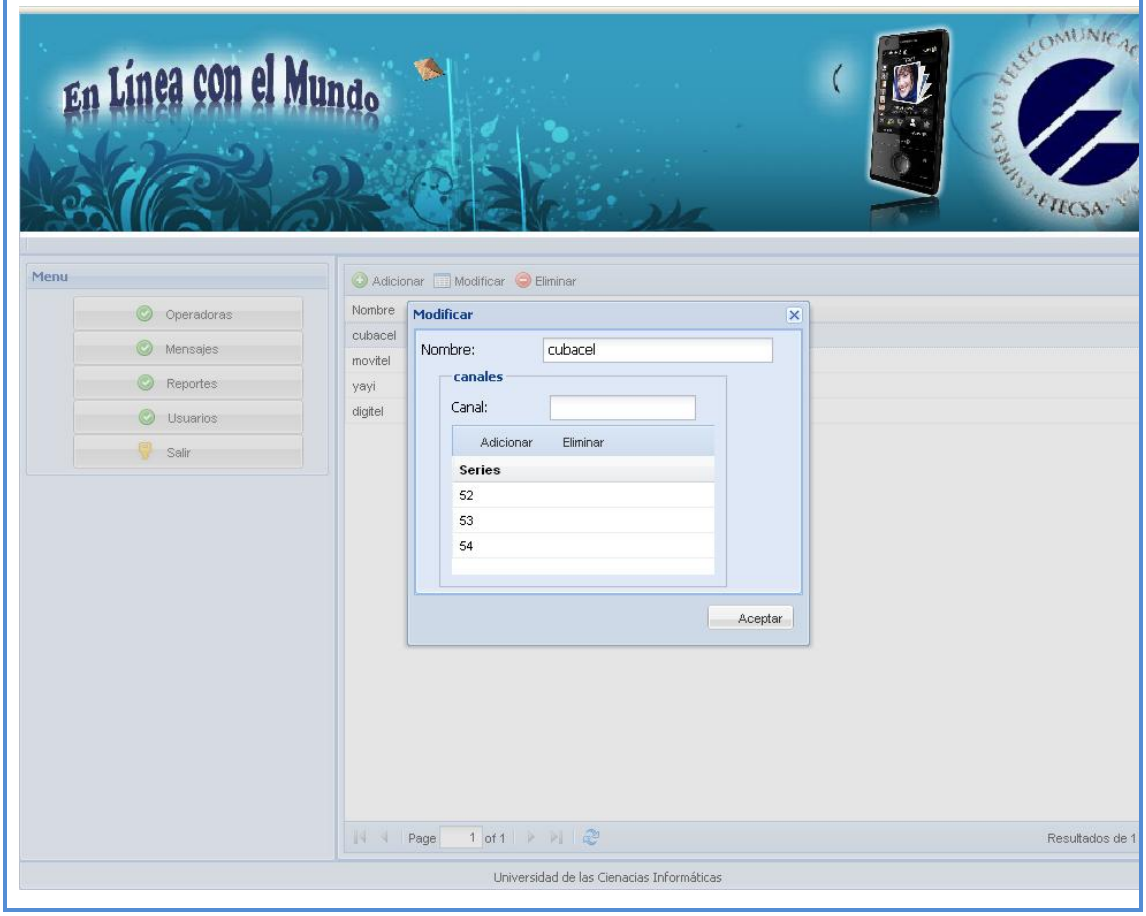


Tabla 11 Actualizar Operadora.

Historia de Usuario	
Número:*8*	**Nombre Historia de Usuario:** Actualizar Operadora.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: María del Pilar García Daniel Crespo Vázquez.	**Iteración Asignada:** 1
Prioridad en Negocio: Alta.	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alto.	**Puntos Reales:**1
Descripción: El Administrador puede cambiarle el nombre a una operadora.	
Prototipo de Interfaz: 	

2.6 Fase de Planificación.

XP plantea la planificación como un permanente diálogo entre la parte empresarial y técnica del proyecto, en la que los primeros decidirán el alcance, ¿qué es lo realmente necesario del proyecto?, la prioridad, qué debe ser hecho en primer lugar, la composición de las versiones, qué debería incluir cada una de ellas y la fecha de las mismas. En cuanto a los técnicos, son los responsables de estimar la duración requerida para implementar las funcionalidades deseadas por el cliente, de informar sobre las consecuencias de determinadas decisiones, de organizar la cultura de trabajo y, finalmente, de realizar la planificación detallada dentro de cada versión.

Durante la fase de planificación se realiza una estimación del esfuerzo que costará implementar cada historia de usuario. En XP las métricas son libres, pudiendo utilizarse cualquier criterio para medir el desarrollo del proyecto. Una métrica popular es la que utiliza como medida el punto. Un punto se considera como una semana ideal de trabajo, donde los miembros de los equipos de desarrollo trabajan el tiempo planeado sin ningún tipo de interrupción. La estimación incluye todo el esfuerzo asociado a la implementación de la historia de usuario.

2.6.1 Estimación de esfuerzo por Historias de Usuario.

Para el buen desarrollo del sistema propuesto, se realizó una estimación de esfuerzo para cada una de las historias de usuario identificadas.

Tabla 12 Estimación de esfuerzo por HU.

Historias de Usuario.	Puntos de Estimación.
Enviar Mensajes.	3/7
Eliminar Mensajes.	4/7
Reenviar Mensajes.	1
Autenticar.	1

Adicionar Usuario	4/7
Eliminar Usuario	3/7
Mostrar Reportes.	1
Adicionar Operadoras.	1
Eliminar Operadoras.	1
Actualizar Operadoras.	1

2.6.2 Plan de Iteraciones.

Una vez identificadas las historias de usuario del sistema y estimado el esfuerzo dedicado a la realización de cada una de estas se procede a la planificación de la etapa de implementación del proyecto. En base a lo antes planteado se decide realizar esta en tres iteraciones, las cuales se detallan a continuación.

Iteración 1.

En esta iteración se implementarán las historias de usuario que por la importancia que tienen para el cliente tienen prioridad con respecto a las otras. Al final de esta se contará con una primera versión de prueba, la cual será mostrada al cliente con el objetivo de obtener una retroalimentación para el grupo de trabajo.

Iteración 2.

El objetivo de esta es la implementación de las historias de usuario de prioridad media. Al finalizar se contará con una versión de prueba que será mostrada al cliente con el objetivo de realizar cambios necesarios en base a la opinión del mismo.

Iteración 3.

Durante el transcurso de esta se implementaron las historias de usuario de baja prioridad. Al finalizar la misma se constará de la versión 1.0 del producto final y como resultado de esta, el sistema será puesto en funcionamiento durante un período de tiempo para evaluar su desempeño.

2.6.3 Plan de duración de las iteraciones.

A continuación se presenta el plan de duración de iteraciones. Este plan tiene como finalidad mostrar la duración de cada iteración, así como el orden en que serán implementadas las historias de usuario en cada una de las mismas.

Tabla 13 Plan de duración de iteraciones.

Iteraciones.	Orden de las Historias de Usuarios a Implementar.	Duración de las Iteraciones. **(**Semanas**)**
Iteración 1.	Enviar Mensajes. Eliminar Mensajes. Reenviar Mensajes.	2
Iteración 2.	Autenticar. Adicionar Usuario. Eliminar Usuario.	2
Iteración 3.	Mostrar Reportes. Adicionar Operadora. Eliminar Operadora. Actualizar Operadora.	4

2.6.4 Plan de entregas.

A continuación se presenta el plan de entregas ideado para la fase de implementación. Como producto del mismo se harán entregas del sistema al finalizar cada iteración en la fecha aproximada que se indica en la siguiente tabla.

Tabla 14 Plan de Entregas.

Módulo.	**Final de Iteración 1**	**Final de Iteración 2**	**Final de Iteración 3**

	Primeras Semanas de Marzo.	Ultimas Semanas de Marzo.	Mes de abril.
EN-M-SAM	0.1	0.2	1.0

2.7 Conclusiones.

Como parte del presente capítulo se abordó todo lo referente a las fases de exploración y planificación del proyecto. La descripción de las historias de usuario, como paso fundamental en el ciclo de desarrollo, ha contribuido a la identificación de los requerimientos del sistema, los cuales sirvieron de base para la generación de una serie de artefactos que serán necesarios en la siguiente fase. Con la obtención del plan de entregas y la planificación de releases se ha logrado delimitar el ciclo de desarrollo del producto.

Capitulo 3.Diseño, implementación y Prueba.

3.1 Introducción.

En el presente capítulo se describen las fases de diseño, implementación y pruebas propias de la metodología de desarrollo XP. Esta metodología plantea que la implementación de un software debe realizarse de forma iterativa, obteniendo al culminar cada iteración un producto funcional que debe ser probado y mostrado al cliente para retroalimentar a los desarrolladores con la opinión de este. También se detallan las tres iteraciones realizadas durante la etapa de codificación del proyecto, así como las tareas de programación desarrolladas en cada una de estas y se hace referencia a las pruebas de aceptación realizadas sobre el sistema.

3.2 Patrón de Arquitectura Modelo Vista Controlador.

En la medida que el tamaño de los sistemas crece, los algoritmos y las estructuras dejan de convertirse en el mayor de los problemas. El nuevo reto lo constituye diseñar y especificar la estructura global del sistema. Para esto en la actualidad se plantea el seguimiento de patrones de

arquitectura para la construcción del software. Para el desarrollo de este módulo el patrón de arquitectura que se seguirá es el Modelo Vista Controlador (MVC).

El Modelo Vista Controlador es aplicable al desarrollo de cualquier aplicación independientemente del lenguaje de programación elegido. MVC consiste en dividir el código de una aplicación en capas, específicamente 3:

- ✓ Modelo
- ✓ Vista
- ✓ Controlador

Cada una de estas capas puede ser sustituida en cualquier momento sin afectar a las otras, puede tener diferentes vistas para un mismo modelo. Esta división del código hace más fácil la portabilidad y la adaptación a los requerimientos del usuario.

Modelo es la representación específica de la información con la cual el sistema opera, se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas. Típicamente el modelo de clases contendrá funciones para consultar, insertar y actualizar información de la base de datos.

Vista en una aplicación Web, es el HTML y lo necesario para convertir datos en HTML. O sea, muestra la información del modelo al usuario. Tienen un registro de su controlador asociado (normalmente porque además lo instancia). Pueden dar el servicio de "Actualización ()", para que sea invocado por el controlador o por el modelo. Tener la vista separada del controlador permite cambiar la aplicación para que genere, en lugar de HTML, algo distinto (por ejemplo, WML), sin tener que tocar más que una parte completamente delimitada del código.

Controlador es lo que une la vista y el modelo. Por ejemplo, son las funciones que toman los valores de un formulario, consultan la base de datos (a través el modelo) y producen valores, que la vista tomará y convertirá en HTML. En resumen gestiona las entradas del usuario.

3.3 Fase de Diseño.

Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto. La complejidad innecesaria y el código extra deben ser evitados. En todo momento, el diseño adecuado para el software es aquel que supera con éxito todas las

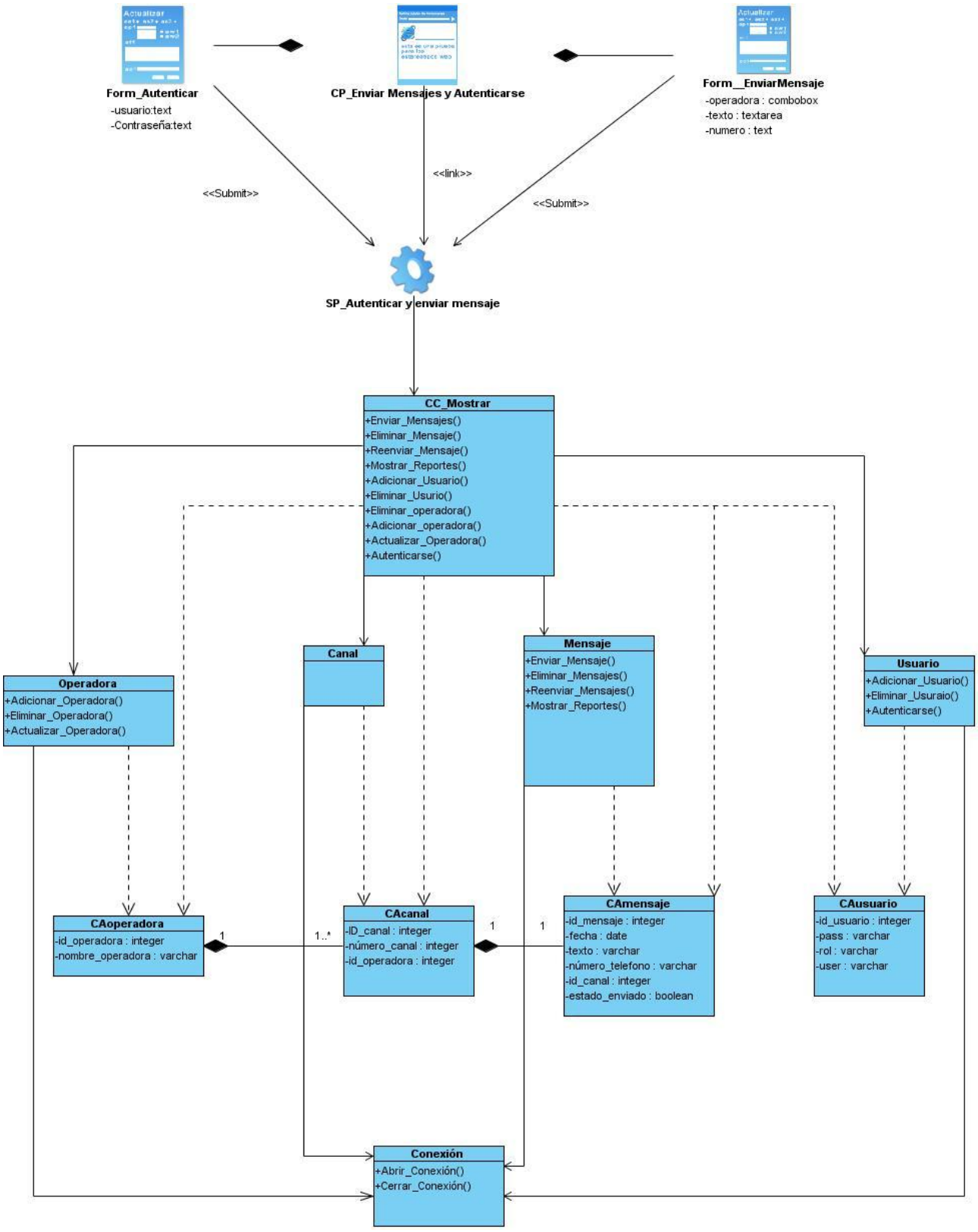
pruebas, no tiene lógica duplicada, refleja claramente la intención de implementación de los programadores y tiene el menor número posible de clases y métodos (Fowler, 2004).

Para el diseño de las aplicaciones, la metodología XP no requiere la representación del sistema mediante diagramas de clases utilizando notación UML. En su lugar se usan otras técnicas como las tarjetas CRC13 como una extensión informal a UML. La técnica de las tarjetas CRC se puede usar para guiar el sistema a través de análisis guiados por la responsabilidad. Las clases se examinan, se filtran y se refinan en base a sus responsabilidades con respecto al sistema, y las clases con las que necesitan colaborar para completar sus responsabilidades (Wake, 1999). No obstante el uso de los diagramas de clases UML puede aplicarse siempre y cuando mejore la comunicación, no sea un peso su mantenimiento, no sean extensos y se enfoquen en la información importante (Fowler, 2004).

3.3.1 Diseño de Diagrama de clase del Sistema.

Para la facilitar la implementación del sistema se realizó, además, el diseño del diagrama de clase del sistema en la cual se registran los datos necesarios para el funcionamiento de la aplicación.

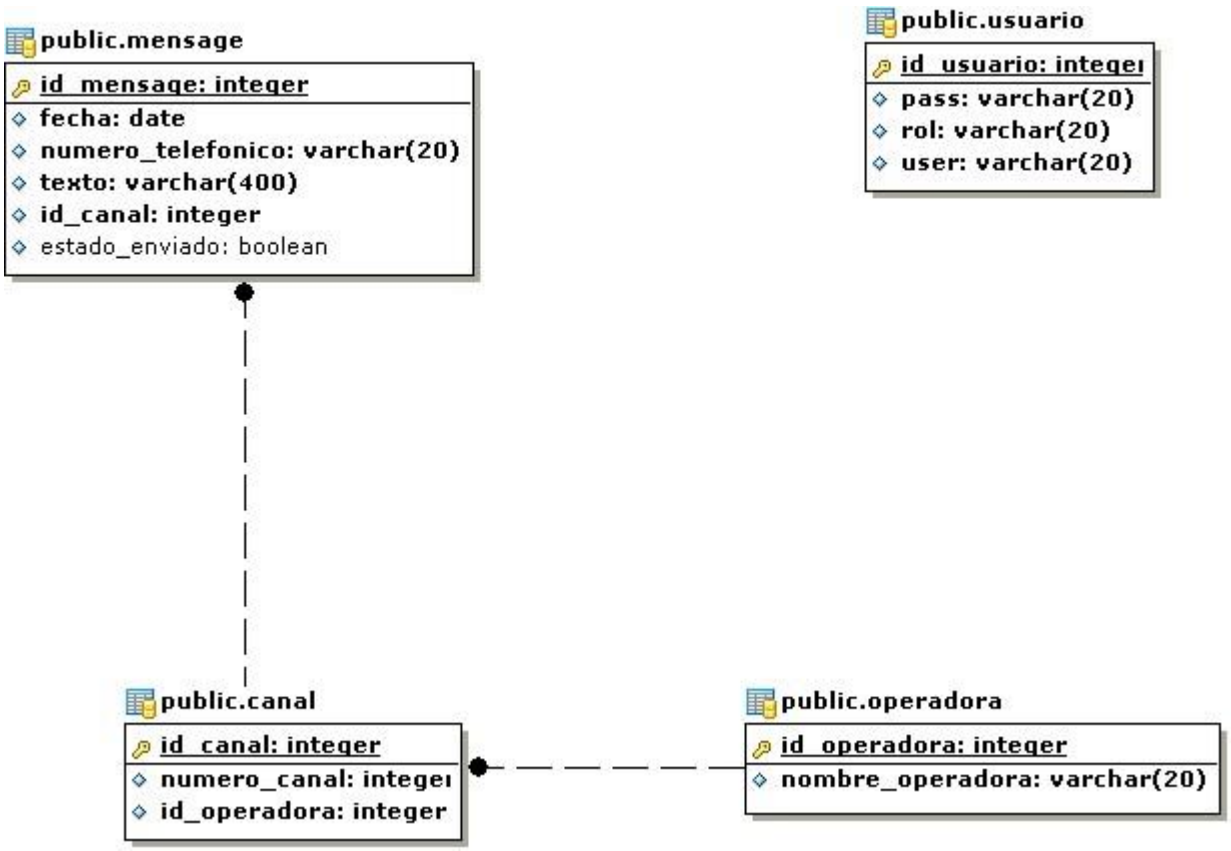
Diagrama de clase del Sistema.



3.3.2 Diseño de base de datos.

Para la facilitar la implementación del sistema se realizó, además, el diseño de una base de datos en la cual se registrarán todos los datos necesarios para el correcto funcionamiento de la aplicación.

Diagrama de clase Persistentes.



3.4 Fase de Implementación.

Durante el inicio de cada iteración, se lleva a cabo una revisión del plan de iteraciones y se modifica de ser necesario (Fowler, 1999).Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable. Estas tareas, pueden escribirse utilizando un lenguaje técnico y no necesariamente deben ser entendibles para el cliente. (Beck, 1999).

A continuación se detalla cada una de las iteraciones del sistema, desarrolladas según la planificación propuesta.

3.4.1 Primera iteración.

Durante el transcurso de esta iteración se implementaron las historias de usuario de mayor prioridad en el sistema, construyéndose la base de la arquitectura. Se logró crear un prototipo del sistema capaz de gestionar el envío de mensajes.

Tabla 15 Historias de Usuario desarrolladas en la primera iteración.

Historia de Usuario.	Estimación.	Real.
Enviar Mensajes.	3/7	3/7
Eliminar Mensajes.	4/7	4/7
Reenviar Mensajes.	1	1
Total.	2	2

3.4.1.1 Tareas de las historias de usuario desarrolladas en la primera iteración.

Tabla 16 Tarea 1 de la historia de usuario Enviar Mensajes.

Tarea	
Número tarea:1	**Número historia:1**
Nombre tarea: Enviar Mensajes.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 3/7
Fecha de inicio: 1/03/10	**Fecha de fin:** 3/03/10
Programador responsable: María del Pilar García.	
Descripción: Se crea un mensaje, que es guardado en la base de datos luego que el usuario escoge operadora, número de teléfono, e introduce su mensaje.	

Tabla 17 Tarea 2 de la historia de usuario Eliminar Mensaje.

Tarea

Número tarea:2	**Número historia:**2
Nombre tarea: Eliminar Mensajes.	
Tipo de tarea: Desarrollo.	**Puntos estimados**: 4/7
Fecha de inicio: 4/03/10	**Fecha de fin:** 8/03/10
Programador responsable: María del Pilar García.	
Descripción: Se implementará la funcionalidad de eliminar mensaje ,cuando desee el administrador eliminar cualquier mensaje	

Tabla 18 Tarea 3 de la historia de usuario Reenviar Mensaje.

Tarea	
Número tarea:3	**Número historia:**3
Nombre tarea: Reenviar Mensajes.	
Tipo de tarea: Desarrollo.	**Puntos estimados:**1
Fecha de inicio: 9/03/10.	**Fecha de fin:** 16/03/10.
Programador responsable: María del Pilar García.	
Descripción: Cada determinado tiempo, se verifica si hay mensajes no enviados en la base de datos, si hay se reenvían, si retorna el mensaje una vez más, es eliminado.	

3.4.2 Segunda iteración.

Durante el transcurso de esta iteración se implementó la historia de usuario de prioridad media en el sistema, Se logró crear una primera vista de la interfaz Web donde el administrador puede autenticarse en el servicio.

Tabla 19 Historias de Usuario desarrolladas en la segunda iteración.

Historia de Usuario.	Estimación.	Real.
Autenticar.	1	1
Adicionar Usuario.	4/7	4/7

Eliminar Usuario.	3/7	3/7
Total.	2	2

3.4.2.1 Tareas de las historias de usuario desarrolladas en la segunda iteración.

Tabla 20 Tarea 4 de la historia de usuario Autenticar.

Tarea	
Número tarea:4	**Número historia:4**
Nombre tarea: Autenticar.	
Tipo de tarea: Desarrollo.	**Puntos estimados:**1
Fecha de inicio:17/03/10	**Fecha de fin:** 24/03/10
Programador responsable: Daniel Crespo Vázquez.	
Descripción: Se implementará la funcionalidad autenticarse para que el administrador se autentique y pueda ver todo lo relacionado con el sistema.	

Tabla 21 Tarea 4 de la historia de usuario Adicionar Usuario.

Tarea	
Número tarea:4	**Número historia:4**
Nombre tarea: Autenticar.	
Tipo de tarea: Desarrollo.	**Puntos estimados:**1
Fecha de inicio:25/03/10	**Fecha de fin:**29/03/10
Programador responsable: Daniel Crespo Vázquez.	
Descripción: Se implementará la funcionalidad adicionar usuario. El administrador es el que agregará el usuario que pueda ver el sistema, como un administrador más.	

Tabla 22 Tarea 4 de la historia de usuario Eliminar Usuario.

Tarea	
Número tarea:4	**Número historia:4**
Nombre tarea: Autenticar.	
Tipo de tarea: Desarrollo.	**Puntos estimados:**1

Fecha de inicio:30/03/10	**Fecha de fin:**2/04/10
Programador responsable: Daniel Crespo Vázquez.	
Descripción: Se implementará la funcionalidad eliminar usuario. El administrador es el elimina al usuario que se autentica.	

3.4.3 Tercera iteración.

Durante el transcurso de esta iteración se implementaron las historias de usuarios de baja prioridad. Qué proporcionará al administrador mostrarle los reportes de los mensajes enviados en los últimos 15 días y 24horas, así también como las últimas funcionalidades del módulo. Al finalizar esta fase, se obtendrá una primera versión del producto.

Tabla 23 Historias de Usuario desarrolladas en la tercera iteración.

Historia de Usuario.	Estimación.	Real.
Mostrar Reportes.	1	1
Adicionar Operadora.	1	1
Eliminar Operadora.	1	1
Actualizar Operadora.	1	1
Total.	4	4

3.4.3.1 Tareas de las historias de usuario desarrolladas en la tercera iteración.

Tabla 24 Tarea 5 de la historia de usuario Mostrar Reportes.

Tarea	
Número tarea:5	**Número historia:5**
Nombre tarea: Mostrar Reportes.	
Tipo de tarea: Desarrollo.	**Puntos estimados:**2
Fecha de inicio:3/04/10	**Fecha de fin:**10/04/10
Programador responsable: Daniel Crespo Vázquez	

Descripción: Esta funcionalidad se implementará para mostrarle al administrador los reportes que quiera percibir, como la cantidad de mensajes que se han enviados en los 15 días o en las 24 horas pasando las fechas que el estime conveniente.

Tabla 25 Tarea 7 de la historia de usuario Adicionar Operadora.

Tarea	
Número tarea:6	**Número historia:6**
Nombre tarea: Adicionar Operadora.	
Tipo de tarea: Desarrollo.	**Puntos estimados:**2
Fecha de inicio:11/04/10	**Fecha de fin:**18/04/10
Programador responsable: Daniel Crespo Vázquez.	
Descripción: Esta funcionalidad se implementará para que el administrador adicione la operadora que desee.	

Tabla 26 Tarea 8 de la historia de usuario Eliminar Operadora.

Tarea	
Número tarea:7	**Número historia:7**
Nombre tarea: Eliminar Operadora.	
Tipo de tarea: Desarrollo.	**Puntos estimados:**2
Fecha de inicio:19/04/10	**Fecha de fin:**26/04/10
Programador responsable Daniel Crespo Vázquez.	
Descripción: Esta funcionalidad se implementará para que el administrador elimine la operadora que desee.	

Tabla 27 Tarea 9 de la historia de usuario Actualizar Operadora.

Tarea	
Número tarea:8	**Número historia:8**
Nombre tarea: Actualizar Operadora.	
Tipo de tarea: Desarrollo.	**Puntos estimados:**2
Fecha de inicio:27/04/10	**Fecha de fin:**4/05/10

Programador responsable: María del Pilar García.
Descripción: Esta funcionalidad se implementará para que el administrador actualice las operadoras.

3.5 Fase de Prueba.

Uno de los pilares de la extreme Programming es el proceso de pruebas. XP anima a probar constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones.

XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñadas por el cliente final.

Las pruebas del sistema tienen como objetivo: verificar la funcionalidad del sistema a través de sus interfaces externas, comprobando que sea la esperada, en función de sus requisitos. Generalmente las pruebas del sistema son desarrolladas por los programadores para verificar que su sistema se comporta de la manera esperada, por lo que se podría enmarcar dentro de la definición de pruebas unitarias que propone XP.

Sin embargo, las pruebas del sistema tienen como objetivo verificar que el sistema cumple los requisitos establecidos por el usuario por lo que también pueden encajar dentro de la categoría de pruebas de aceptación.

Cada vez que se le da cumplimiento a una o varias historias de usuario, se presenta el software debidamente probado al cliente, para realizar las pruebas de aceptación de la funcionalidad implementada. Cuando se reúnen un número suficiente de funcionalidades que representan una versión útil y parcialmente completa de la aplicación se produce una liberación, lo cual es una versión funcional de la aplicación que aporta valor al negocio y que debe ser mantenida a la par que se desarrollan las siguientes funcionalidades.

A continuación se explica detalladamente las pruebas unitarias y las de aceptación.

3.5.1 Pruebas Unitarias.

En el transcurso de la implementación de un sistema, cada uno de los desarrolladores tiene que ir probando constantemente lo que va obteniendo para garantizar que las funcionalidades exigidas por el cliente estén siendo implementadas correctamente. A estas pruebas se le llama Pruebas Unitarias. Aunque estas no generan artefactos y no son directamente palpables para el cliente, son de vital importancia para el desarrollo de un proyecto.

3.5.2 Pruebas de Aceptación.

Las pruebas de aceptación son las realizadas por el cliente y usuarios finales de la aplicación. En estas serán probadas las funcionalidades exigidas por el cliente y descritas en las historias de usuario, además de los aspectos de seguridad requeridos. Luego de haber superado las pruebas de aceptación podrá considerarse que la aplicación es apta para el uso y despliegue dentro del proyecto.

Como resultado de las pruebas de aceptación se obtendrán artefactos descritos en tablas, estas contarán con los siguientes campos:

- ✓ Código: servirá como identificador de la prueba realizada, a su vez será sugerente al nombre de la prueba a la que hace referencia.
- ✓ UH: tendrá el nombre de la historia de usuario a la que hace referencia la prueba a realizar.
- ✓ Nombre: nombre que se le da a la prueba a realizar.
- ✓ Descripción: se describe la funcionalidad que se desea probar.
- ✓ Condiciones de Ejecución: mostrará las condiciones que deben cumplirse para poder llevar a cabo el caso de prueba.
- ✓ Entradas / Pasos de Ejecución: se hará la descripción de cada uno de los pasos seguidos durante el desarrollo de la prueba, se tendrá en cuenta cada una de las entradas que hace el usuario con el objetivo de ver si se obtiene el resultado esperado.
- ✓ Resultado esperado: se hará una breve descripción del resultado que se espera obtener con la prueba realizada.

- ✓ Evaluación de la prueba: acorde al resultado de la prueba realizada se emitirá una evaluación sobre la misma. Esta evaluación tendrá uno de los tres resultados que a continuación se describen:

1. Bien: cuando el resultado de la prueba es exactamente el esperado por el usuario.
2. Parcialmente bien: cuando el resultado no es completamente el esperado por el cliente o usuario de la aplicación y muestra resultados erróneos o fuera de contexto.
3. Mal: cuando el resultado de la prueba realizada genera un error de codificación en la aplicación o muestra como resultado elementos no deseados o fuera de contexto, trayendo como consecuencia que la funcionalidad requerida por el cliente no tenga resultado, lo que invalida también la UH.

Las pruebas de aceptación se llevarán a cabo siguiendo los pasos que se muestran a continuación:

1. Se redactarán los casos de prueba teniendo en cuenta el orden de las historias de usuarios y los niveles de prioridad dados a las funcionalidades.

2. Se hará la planificación con el cliente de cuándo y cuáles pruebas serán llevadas a cabo.

3. Se reunirán los miembros del proyecto seleccionados para realizar las pruebas.

4. Se completarán cada uno de los campos de las tablas de las pruebas de aceptación con el resultado de la prueba.

Las pruebas de aceptación son más importantes que las pruebas unitarias dado que significan la satisfacción del cliente con el producto desarrollado y el final de una iteración y el comienzo de la siguiente por esto, el cliente es la persona adecuada para diseñar las pruebas de aceptación.

A continuación se muestran las pruebas de aceptación de mayor prioridad del sistema y las otras se pueden ver en el anexo I.

Tabla 28 Pruebas de aceptación para la historia de usuario Enviar Mensaje.

Caso de Prueba de Aceptación.	
Código:1	**Historia de Usuario:**1
Nombre: Enviar Mensaje.	
Descripción: Prueba para verificar la funcionalidad de enviar un mensaje.	

Condiciones de Ejecución: Cualquier usuario pueda acceder y mandar un mensaje.
Entrada/Pasos de ejecución: Se intente enviar un mensaje.
Resultado Esperado: Se envió correctamente un mensaje.
Evaluación de la Prueba: Prueba satisfactoria.

Tabla 29 Pruebas de aceptación para la historia de usuario Reenviar Mensajes.

Caso de Prueba de Aceptación.	
Código:2	**Historia de Usuario:**2
Nombre: Reenviar Mensaje	
Descripción: Prueba para comprobar la funcionalidad de reenviar un mensaje.	
Condiciones de Ejecución: El sistema debe ser levantado y logueado por un administrador.	
Entrada/Pasos de ejecución: Se intenta reenviar un mensaje.	
Resultado Esperado: Se logra reenviar un mensaje.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 30 Pruebas de aceptación para la historia de usuario Eliminar Mensajes.

Caso de Prueba de Aceptación.	
Código:3	**Historia de Usuario:**3
Nombre: Eliminar Mensaje.	
Descripción: Prueba para comprobar la funcionalidad de eliminar un mensaje.	
Condiciones de Ejecución: El sistema debe ser levantado y logueado por un administrador.	
Entrada/Pasos de ejecución: Se intenta eliminar un mensaje.	
Resultado Esperado: Se logra eliminar un mensaje sin errores.	

Evaluación de la Prueba: Prueba satisfactoria.

3.8 Conclusiones.

En este capítulo se describieron las fases de diseño, implementación y prueba. Se realizó una breve descripción del diseño del sistema, el cual se realizó basado en la simplicidad. Se detallaron cada una de las tareas que se realizaron en las tres iteraciones del sistema. Se abordó acerca de la importancia de las pruebas en la metodología XP, en especial las pruebas de aceptación, ya que éstas miden la satisfacción del cliente con el producto desarrollado.

CAPÍTULO 4 ESTUDIO DE FACTIBILIDAD.

4.1Introducción.

En la planificación del proceso de desarrollo de software es de vital importancia la estimación, la cual consiste en determinar con cierto grado de certeza los recursos necesarios para el desarrollo del mismo, ya sean recursos de hardware, software, esfuerzo, tiempo y costo. En el presente capítulo se realizará un estudio de factibilidad para la realización del sistema propuesto mediante una estimación de tamaño, esfuerzo y planificación necesaria para llevar a cabo el mismo.

4.2 Modelo matemático COCOMO II.

COCOMO, propuesto y desarrollado por Barry Boehm, es uno de los modelos de estimación de costos mejor documentados y utilizados. El modelo permite determinar el esfuerzo y tiempo que se requiere en un proyecto de software a partir de una medida del tamaño del mismo expresada en el número de líneas de código que se estimen generar para la creación del producto software.

Consiste básicamente en la aplicación de ecuaciones matemáticas sobre los Puntos de Función sin ajustar o la cantidad de líneas de código (SLOC, Source Lines of Code) estimados para un proyecto. Estas ecuaciones se encuentran ponderadas por ciertos factores de costo (cost drivers) que influyen en el esfuerzo requerido para el desarrollo del software.

Está compuesto por tres modelos que se adaptan tanto a las necesidades de los diferentes sectores descritos, como al tipo y cantidad de información disponible en cada etapa del ciclo de vida de desarrollo, lo que se conoce por granularidad de la información.

El estudio de factibilidad de este sistema se realizó mediante el modelo de diseño temprano que se utiliza en las primeras etapas del desarrollo, en las cuales se evalúan las alternativas de hardware y software de un proyecto. En estas etapas se tiene poca información, se conoce muy poco del tamaño del producto a ser desarrollado, de la naturaleza de la plataforma, del personal a ser incorporado al proyecto o aspectos específicos del proceso a utilizar. Ese nivel de detalle en este modelo es consistente con el nivel general de información disponible y con el nivel general de estimación detallada que es necesaria en estas etapas, lo que concuerda con el uso de Puntos de Función. Para estimar tamaño usa Puntos de Función No Ajustados como métrica de medida y un número reducido de factores de costo.

4.3Características del Proyecto.

El primer paso a llevar a cabo para la estimación del proyecto consiste en la obtención de los Puntos de Función desajustados, los cuales están dados por la suma de cada una de las entradas, las salidas y las consultas externas del sistema, así como los archivos lógicos internos y de interfaz externos. A continuación se muestran cada una de estas características aplicadas al software en cuestión.

4.3.1 Entradas externas.

Se definen como un proceso elemental mediante el cual ciertos datos cruzan la frontera del sistema desde afuera hacia adentro. En el caso particular de la aplicación propuesta se cuenta con tres entradas externas, especificadas en la siguiente tabla:

Tabla 31 Entradas externas.

Nombre de la entrada externa.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)
Enviar Mensajes.	1	3	Simple
Autenticarse.	1	2	Simple
Mostrar Reportes.	1	2	Simple
Eliminar Mensajes.	1	1	Simple
Reenviar Mensajes.	1	1	Simple
Adicionar Operadora.	1	2	Simple
Eliminar Operadora.	1	1	Simple
Actualizar Operadora.	1	2	Simple
Total		14	Simple

4.3.2 Salidas Externas.

Se definen como un proceso elemental con componentes de entrada y de salida mediante el cual datos simples y datos derivados (esto es, datos que se calculan a partir de otros datos) cruzan las

fronteras del sistema desde adentro hacia afuera. Las salidas externas vinculadas al proyecto se describen a continuación.

Tabla 32 Salidas externas.

Nombre de la salida externa.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)
Todos los mensajes enviados.	1	3	Simple
Total		**3**	

4.3.3 Consultas Externas.

Se definen como un proceso elemental con componentes de entrada y de salida donde un Actor del sistema rescata datos de uno o más Archivos Lógicos Internos o Archivos de Interfaz Externos. Los datos de entrada no actualizan ni mantienen ningún archivo (lógico interno o de interfaz externo) y los datos de salida no contienen datos derivados (es decir, los datos de salida son básicamente los mismos que se obtienen de los archivos).En este caso no fueron identificados.

Tabla 33 Consultas externas.

Nombre de la Petición.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)
Total		**0**	

4.3.4 Archivos Lógicos Internos.

Constituyen un grupo de datos relacionados lógicamente e identificables por el usuario, que residen enteramente dentro de los límites del sistema y se mantienen a través de entradas externas.

Tabla 34 Fichero lógico interno.

Nombre de la petición.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)

Base Batos: Celular	1	4(tablas)	Simple.
Total		**4**	

4.3.5 Archivos de Interfaz Externos.

Son un grupo de datos relacionados lógicamente e identificables por el usuario, que se utilizan solamente para fines de referencia. Los datos residen enteramente fuera de los límites del sistema y se mantienen por las Entradas Externas de otras aplicaciones, es decir, cada Archivo de Interfaz Externo es un Archivo Lógico Interno de otra aplicación. En este caso fue identificado un Archivos de interfaz externa.

Tabla 35 Interfaces externas

Nombre de la interfaz externa.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)
Service API for Mobile Services(SAM)	1	3	Simple.
Total		**3**	

4.3.6 Puntos de Función Desajustados.

La siguiente tabla está basada en las características del sistema anteriormente expuestas, el producto de la cantidad existente de cada una de ellas y el peso correspondiente a las mismas dan como resultado final, los puntos de función desajustados pertenecientes al proyecto.

Tabla 36 Puntos de función desajustados.

Elementos.	Simple.		Medio.		Complejo.		Subtotal.
	No.	Peso.	No.	Peso.	No.	Peso.	
Entradas externas.	14	3	0	4	0	6	39
Salidas externas.	3	4	0	5	0	7	12
Consultas externas.	0	3	0	4	0	6	0
Fichero lógico interno.	4	7	0	10	0	15	28

Fichero interfaz interno.	3	5	0	7	0	10	15
Total (UFP):							**94**

Se encuentra la tabla de peso del factor de complejidad. [Anexo III]

4.4 Cálculos de instrucciones fuentes, esfuerzo, tiempo de desarrollo, cantidad de hombres y costo.

4.4.1 Cálculo de instrucciones fuentes.

Una vez obtenida la cantidad total de puntos de función desajustados pertenecientes al proyecto, se procede a calcular la cantidad de instrucciones fuentes del mismo, para lo cual se utiliza la ecuación que se muestra seguidamente:

$$SLOC = UFP \; x \; ratio$$

Donde:

SLOC: Cantidad de instrucciones fuente (5546).

UFP: Puntos de función desajustados. (94)

Ratio : Conversión de puntos de función desajustados a líneas de código para el lenguaje PHP. (59)

Partiendo de la ecuación anterior se obtiene el siguiente resultado:

SLOC $=^{94x5}$ SLOC$=5546$ = 5.5 KSLOC

KSLOC: Tamaño del software a desarrollar.

Las características del sistema y valores obtenidos anteriormente han sido plasmadas en la siguiente tabla:

Tabla37 Características del sistema.

Características	Valor
Puntos de función desajustados.	94

Lenguaje (PHP).	44
Instrucciones fuentes por puntos de función.	4136 SLOC
Instrucciones fuentes.	4.1 KSLOC

4.4.2 Cálculo de esfuerzo nominal.

Posteriormente se procede al cálculo del esfuerzo nominal, ecuación que se toma como base tanto en el método de diseño preliminar, al cual se hace alusión anteriormente, como en el modelo Post arquitectura, ambos definidos por COCOMO II.

$$PM_{Nominal} = Ax(Size)^{B}$$

$$PM_{Nominal} = 2.94\ x(4.41)^{0.91}$$

$$PM_{Nominal} = 11.3\ meses/hombre$$

Donde:

$PM_{Nominal}$: Esfuerzo nominal requerido en meses – hombres.

$Size$: Tamaño estimado del software en Puntos de Función sin Ajustar. (4.1 KSLOC)

A: Constante que se utiliza para capturar los efectos multiplicativos en el esfuerzo requerido de acuerdo al crecimiento de tamaño del software. (2.94).

B: Constante denominada Factor escalar y su valor está dado por la resultante de los aspectos positivos sobre los negativos que presenta el proyecto.

$$B = 0.91 + 0.01\ x \sum(W_i)$$

B =0.91 + 0.01 x 9.15

B =0.91

Donde:

W_i: Variables escalares que indican las características que el proyecto presenta en lo que a su complejidad y entorno de desarrollo se refiere.

4.4.2.1 Precedentes (PREC).

El factor de precedencia (PREC) toma en cuenta el grado de experiencia previa en relación al producto a desarrollar, tanto en aspectos organizacionales como en el conocimiento del software y hardware a utilizar.

4.4.2.2 Flexibilidad de desarrollo (FLEX).

El factor de flexibilidad (FLEX) considera el nivel de exigencia en el cumplimiento de los requerimientos preestablecidos, plazos de tiempos y especificaciones de interfaz.

4.4.2.3 Cohesión del equipo (TEAM).

El factor de escala denominado Cohesión del Equipo tiene en cuenta las dificultades de sincronización entre los participantes del proyecto: usuarios, clientes y desarrolladores. Estas dificultades pueden surgir por diferencias culturales, dificultad en la conciliación de objetivos, falta de experiencia y familiaridad con el trabajo en equipo.

4.4.2.4 Solución de riesgos (RESL).

Este factor involucra aspectos relacionados al conocimiento de los ítems de riesgo crítico y al modo de abordarlos dentro del proyecto.

4.4.2.5 Madurez del proceso (PMAT).

El procedimiento para determinar el PMAT es establecer el porcentaje de cumplimiento de cada una de las Áreas evaluando el grado de cumplimiento de las metas correspondientes.

La siguiente tabla muestra los valores asignados a cada una de estas variables:

Tabla 38 Factores de escala.

Nombre.	Valor.	Justificación.
PREC	1.24	Existen varios proyectos similares a nivel

		internacional, pero no se encontró ninguno a nivel nacional.
FLEX	1.01	Cuenta con alta flexibilidad en cuanto a los requerimientos establecidos inicialmente.
TEAM	1.10	El equipo de desarrollo presenta una alta cohesión.
RESL	4.24	No se identificaron grandes riesgos.
PMAT	1.56	Se cuenta con la experiencia necesaria como para que el software cumpla con las funcionalidades requeridas.
Total(SF)	9.15	

Se encuentra la tabla de factores de escala. [Anexo III]

4.5 Ajuste del esfuerzo nominal.

El esfuerzo calculado anteriormente es un valor nominal y debe ser ajustado en base a las características del proyecto para lo cual se tiene un conjunto de Multiplicadores de Esfuerzo (MEi) que representan las características del proyecto y expresan su impacto en el desarrollo total del producto de software.

4.5.1Clasificados en categorías

Los 7 Multiplicadores de Esfuerzo son:

Del Producto.

RCPX: Confiabilidad y Complejidad del producto

RUSE: Reusabilidad Requerida

De la Plataforma.

PDIF: Dificultad de la Plataforma

Del Personal.

PERS: Aptitud del Personal

PREX: Experiencia del Personal

Del Proyecto.

FCIL: Facilidades

SCED: Cronograma de Desarrollo Requerido

Tabla 39 Multiplicadores de esfuerzo.

Nombre.	Valor.	Justificación.
RCPX	1.74	La complejidad del sistema es alta.
RUSE	1.00	Pretende reutilizarse una parte del código.
PDIF	1.00	Uso de memoria y almacenamiento normal, plataforma estable.
PREX	1.12	Baja experiencia en el personal en cuanto a la utilización del lenguaje y herramientas.
PERS	1.00	La capacidad del personal es normal.
FCIL	1.00	La utilización de entornos de desarrollo integrados facilita en gran medida el trabajo.
SCED	1.00	El sistema se desarrollo en el tiempo establecido.

Total(EM)	1.95	

Se encuentra la tabla de multiplicadores de esfuerzo del modelo del diseño temprano. [Anexo III]

$$PM_{ajustado} = PM_{Nominal} x \prod (ME_t)$$

$PM_{ajustado}$ = 11.34 meses/hombre x 1.95.

$PM_{ajustado}$= 22.11meses/hombre.

4.5.2 Calculo del tiempo de desarrollo del software.

El tiempo requerido para el desarrollo del proyecto está dado por la siguiente ecuación:

$$TDEV = C\ x\ (PM_{ajustado})^F$$

$$TDEV = 3.64\ x\ 22.11^{0.26}$$

$$TDEV = 3.64$$

Donde:

C: Constante con valor 3.64.

$PM_{ajustado}$ = 22.11meses/hombre.

F: 0.26.

CALCULAR

$$F = D + 0.2\ x\ 0.01x \sum SF$$

F =0.24 + 0.2 x 0.01 x 9.15.

F =0.26

Donde:

D : Constante cuyo valor es 0.24.

SF : Valor de los factores de escala.

4.3.5 Cálculo del costo total del proyecto.

Para el cálculo del costo total correspondiente al proyecto en cuestión, COCOMO II propone la siguiente ecuación:

$$C = CHM \; x \; PM$$

$C =$ 138 x 11.3.

$C =$ 1559.4

CALCULAR

Donde:

C:Costo total.

CHM**:** Costo teniendo en cuenta salario de todos los obreros, el cual se calcula por la siguiente ecuación.

$$CHM = CH \; x \; sal$$

CHM =1.38 x 100.

CHM =138.

CALCULAR

Donde:

sal : Salario medio por cada trabajador.

CH : Cantidad de personas destinadas al proyecto, lo que es calculado a través de la ecuación:

$$CH = {}^{PM}/_{TDEV}$$

CH =11.3 / 8.14.

CH =1.38.

CALCULAR

4.5.3 Resultados.

En la siguiente tabla se muestran los resultados obtenidos luego de haber efectuado todos los cálculos para determinar el costo y esfuerzo requeridos por el proyecto.

Tabla 40 Resultados.

Cálculo de:	Valor
Esfuerzo.	11.3 meses/hombre.
Tiempo de desarrollo.	6 meses.
Cantidad de hombres.	2 hombres.
Salario medio.	100 pesos.
Costo.	1559.4 pesos.

4.6 Análisis de costo.

El desarrollo de un producto siempre tiene un costo de producción, el cual debe ser justificado en base a los beneficios reportados por el mismo. El sistema que se propone en este trabajo no conlleva a grandes gastos, puesto que solo es influyente el salario de los desarrolladores, por lo cual se concluye que su implementación es factible. Esto se debe en gran medida a la utilización del componente api, y a las herramientas libres que no requieren el pago de alguna licencia en cambio se prevén beneficios pues resolvería la problemática de no tener una interfaz de la cual mandar mensajes a celulares de territorio nacional e internacional, además, debido a sus características se podría comercializar e instalar en algún país interesado.

4.7 Conclusiones.

En el presente capítulo se realizó un análisis de factibilidad de la solución propuesta, arribando a la conclusión de que es viable su desarrollo comparando los costos de producción con los beneficios reportados por su puesta en funcionamiento.

Conclusiones Generales.

Con el desarrollo de este trabajo se profundizó en el conocimiento de la mensajería móvil. Se aplicó la metodología XP para guiar el proceso de desarrollo de software, mediante el cual se implementó un servicio de envío de mensajes a celulares desde una interfaz integrada al componente SAM, con las funcionalidades previstas para el primer ciclo de desarrollo del software. El diseño y la implementación se rigieron por algunas herramientas como PHP, la plataforma Java y Postgres, aplicando el patrón de diseño modelo vista controlador (MVC). Se puede concluir que se ha cumplido satisfactoriamente el objetivo trazado para este trabajo, enfatizando en los siguientes puntos: Se implementó una interfaz Web que posibilita llevar a cabo el proceso de envío de mensajes a los usuarios, además es amigable y permite enviar los mensajes por parte de los usuarios.

Recomendaciones.

A continuación son listadas las recomendaciones que en opinión de los autores deben ser tenidas en cuenta con el objetivo de realizar un seguimiento de este trabajo:

1. Integrar el sistema a la Plataforma de Servicios de Cubacel

2. Utilizar este sistema como base para la implementación de otros servicios de valor agregado en la plataforma de servicios de Cubacel.

3. Ponerlo al alcance de los usuarios de la UCI.

4. Agregarle funcionalidades nuevas que permitan la mejor comodidad de los usuarios.

5. Estudio para la Implementación en pequeñas empresas de Venezuela.

BIBLIOGRAFÍAS.

AppServ. appserv. [En línea] [Citado el: 24 de febrero de 2010.] http://www.appservnetwork.com.

AppServ. AppServ. [Online] [Cited: febrero 24, 2010.] http://www.tallerwebmaster.com/tutorial/instalacion-de-appserv-php-apache-mysql/67/.

AppServ. AppServ. [Online] [Cited: febrero 24, 2010.] http://www.thestromboliproject.com/tutoriales/extras/appserv-apache-+-mysql-2.htm.

Extreme Programming (XP). Extreme Programming (XP). [En línea] [Citado el: 28 de Febrero de 2010.] http://elezeta.net/2004/08/27/extreme-programming-xp/.

Fernández, Gerardo Escribano. Introducción a Extreme Programming. 2002.

Framework. Wikipedia enciclopedia libre. [En línea] 2010. [Citado el: 6 de Marzo de 2010.] http://es.wikipedia.org/wiki/Framework.

Gestor de Base de Datos: MySQL, PostgreSQL, SQLite. Gestor de Base de Datos: MySQL, PostgreSQL, SQLite. [En línea] [Citado el: 28 de febrero de 2010.] http://www.eaprende.com/gestor-de-basededatos-mysql-postresql-sqlite.html.

Interfaz de programación de aplicaciones. Wikipedia enciclopedia libre. [En línea] 2010. [Citado el: 14 de Marzo de 2010.] http://es.wikipedia.org/wiki/Application_Programming_Interface.

Las principales características de MySQL. Las principales características de MySQL. [En línea] [Citado el: 12 de febrero de 2010.] http://dev.mysql.com/doc/refman/5.0/es/features.html.

Las principales características de MySQL. Las principales características de MySQL. [En línea] [Citado el: 24 de febrero de 2010.] http://dev.mysql.com/doc/refman/5.0/es/features.html.

Matos García, Rosa María. Diseño de Bases de Datos. Octubre 1999.

Nuevas funciones y ventajas de Dreamweaver [En línea] [Citado el: 28 de febrero de 2010.] http://www.adobe.com/es/devnet/dreamweaver/articles/dw8_newfeatures.html.

PostgreSQL.2008. PostgreSQL.2008. [En línea] [Citado el: 25 de febrero de 2010.] http://www.postgresql.org/about/..

PostgreSQL. . PostgreSQL. . [En línea] 20 de Marzo de 2008. [Citado el: 26 de febrero de 2010.] http://gokuh12.go.funpic.org/weblog.php?w=1&category=Software%20Libre.

Sistemas de gestión de bases de datos. Sistemas de gestión de bases de datos. [En línea] [Citado el: 10 de febrero de 2010.] http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos.

Short Message. Search Tech Target. [En línea] 2 de Mayo de 2005. [Citado el: 15 de Febrero de 20010.] http://searchmobilecomputing.techtarget.com/sDefinition/0,290660,sid40_gci773769,00.html.

Tomcat. tomcat. [Online] [Cited: febrero 22, 2010.] http://es.wikipedia.org/wiki/Tomcat.

Tomcat. Tomcat. [Online] [Cited: febrero 22, 2010.] http://tomcat.apache.org/.

Tomcat. Tomcat. [Online] [Cited: febrero 22, 2010.] http://mundogeek.net/archivos/2006/04/04/apache-y-tomcat-en-windows/.

TechTarget. 2005. Multimedia Message Service. Search Tech Target. [En línea] 2 de Mayo de 2005. [Citado el: 21 de Febrero de 2010.] http://searchmobilecomputing.techtarget.com/sDefinition/0,290660,sid40_gci943702,00.html.

Telefónica, Movistar. Telefónica Movistar. E-mocion Movistar. [En línea] 2010. [Citado el: 23 de Febrero de 2010.] http://www.movistar.es/accesible/movistar-emocion.html.

Zend Studio. Zend Studio. [En línea] [Citado el: 29 de febrero de 2010.] http://es.wikipedia.org/wiki/Zend_Studio.

Referencia Bibliográfica.

0. Las principales características de Java. Las principales características de Java. [En línea] [Citado el: 12 de febrero de 2010.] http://dev.java.com/doc/refman/5.0/es/features.html.

1. Extreme Programming (XP). Extreme Programming (XP). [En línea] [Citado el: 28 de Febrero de 2010.] http://elezeta.net/2004/08/27/extreme-programming-xp/.

2. Fernández, Gerardo Escribano. Introducción a Extreme Programming. 2002.

3. Gestor de Base de Datos: MySQL, PostgreSQL, SQLite. Gestor de Base de Datos: MySQL, PostgreSQL, SQLite. [En línea] [Citado el: 28 de febrero de 2010.] http://www.eaprende.com/gestor-de-basededatos-mysql-postresql-sqlite.html.

4. Las principales características de MySQL. Las principales características de MySQL. [En línea] [Citado el: 12 de febrero de 2010.] http://dev.mysql.com/doc/refman/5.0/es/features.html.

5. Las principales características de MySQL. Las principales características de MySQL. [En línea] [Citado el: 24 de febrero de 2010.] http://dev.mysql.com/doc/refman/5.0/es/features.html.

6. Matos García, Rosa María. Diseño de Bases de Datos. Octubre 1999.

7. Nuevas funciones y ventajas de Dreamweaver

8. [En línea] [Citado el: 28 de febrero de 2010.] http://www.adobe.com/es/devnet/dreamweaver/articles/dw8_newfeatures.html.

8. PostgreSQL. . PostgreSQL. . [En línea] 20 de Marzo de 2008. [Citado el: 26 de febrero de 2010.] http://gokuh12.go.funpic.org/weblog.php?w=1&category=Software%20Libre.

9. Sistemas de gestión de bases de datos. Sistemas de gestión de bases de datos. [En línea] [Citado el: 10 de febrero de 2010.] http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos.

10. Zend Studio. Zend Studio. [En línea] [Citado el: 29 de febrero de 2010.] http://es.wikipedia.org/wiki/Zend_Studio.

11. AppServ. appserv. [En línea] [Citado el: 24 de febrero de 2010.] http://www.appservnetwork.com.

Anexo

Anexo I. Pruebas de Aceptación.

Tabla 41 Pruebas de aceptación para la historia de usuario Autenticar.

Caso de Prueba de Aceptación.	
Código:4	**Historia de Usuario:**4
Nombre: Autenticar.	
Descripción: Prueba para comprobar la funcionalidad de un usuario al autenticarse.	
Condiciones de Ejecución: El sistema debe estar levantado y logueado por un usuario.	
Entrada/Pasos de ejecución: Se intenta autenticar un usuario.	
Resultado Esperado: Se autentica sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 42 Pruebas de aceptación para la historia de usuario Adicionar Usuario.

Caso de Prueba de Aceptación.	
Código:5	**Historia de Usuario:**5
Nombre: Adicionar Usuario.	
Descripción: Prueba para comprobar la funcionalidad que un usuario se adiciona para la administración del sistema.	
Condiciones de Ejecución: El sistema debe estar levantado y logueado por un usuario para adicionar un usuario.	
Entrada/Pasos de ejecución: Se intenta adicionar un usuario.	
Resultado Esperado: Se adiciona sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 43 Pruebas de aceptación para la historia de usuario Eliminar Usuario.

Caso de Prueba de Aceptación.	
Código:6	**Historia de Usuario:**6
Nombre: Eliminar Usuario.	
Descripción: Prueba para comprobar la funcionalidad que se elimine un usuario del sistema.	
Condiciones de Ejecución: El sistema debe estar levantado y logueado por un usuario para eliminar un usuario.	
Entrada/Pasos de ejecución: Se intenta eliminar un usuario.	
Resultado Esperado: Se elimina sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 44 Pruebas de aceptación para la historia de usuario Mostrar Reportes.

Caso de Prueba de Aceptación.	
Código:7	**Historia de Usuario:**7
Nombre: Mostrar Reportes.	
Descripción: Prueba para comprobar la validez de los reportes mostrados	
Condiciones de Ejecución: El sistema debe ser levantado y logueado por un administrador.	
Entrada/Pasos de ejecución: Se intenta mostrar un reporte con la cantidad de mensajes enviados.	
Resultado Esperado: La cantidad del reporte coincide con la cantidad real de mensajes enviados.	

Evaluación de la Prueba: Prueba satisfactoria.

Tabla 45 Pruebas de aceptación para la historia de usuario Adicionar Operadoras.

Caso de Prueba de Aceptación.	
Código:8	**Historia de Usuario:**8
Nombre: Adicionar Operadoras.	
Descripción: Prueba para comprobar la funcionalidad de adicionar una nueva operadora.	
Condiciones de Ejecución: El sistema debe ser levantado y logueado por un administrador.	
Entrada/Pasos de ejecución: Se intenta adicionar una nueva operadora.	
Resultado Esperado: La operadora se inserta sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 46 Pruebas de aceptación para la historia de usuario Eliminar Operadoras.

Caso de Prueba de Aceptación.	
Código:9	**Historia de Usuario:**9
Nombre: Eliminar Operadoras.	
Descripción: Prueba para comprobar la funcionalidad de eliminar una operadora.	
Condiciones de Ejecución: El sistema debe ser levantado y logueado por un administrador.	
Entrada/Pasos de ejecución: Se intenta eliminar una operadora.	
Resultado Esperado: Se logra eliminar una operadora sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 47 Pruebas de aceptación para la historia de usuario Actualizar Operadoras.

Caso de Prueba de Aceptación.	
Código:10	**Historia de Usuario:**10
Nombre: Actualizar Operadoras.	
Descripción: Prueba para comprobar la funcionalidad de actualizar las operadoras.	
Condiciones de Ejecución: El sistema debe ser levantado y logueado por un administrador.	
Entrada/Pasos de ejecución: Se intenta actualizar las operadoras del sistema**.**	
Resultado Esperado: Se realiza la actualización de las operadoras.	
Evaluación de la Prueba: Prueba satisfactoria.	

Anexo III. Tablas de Cocomo II.

Tipo de función	Peso del Factor de Complejidad		
	Bajo	Promedio	Alto
Entradas Externas (Inputs)	3	4	6
Salidas Externas (Outputs)	4	5	7
Archivo Lógicos Internos (Archivos)	7	10	15
Archivos Externos de Interfase (Interfases)	5	7	10
Consultas Externas (Queries)	3	4	6

Tabla 10: Peso del Factor de Complejidad. [Boehm 1995/2]

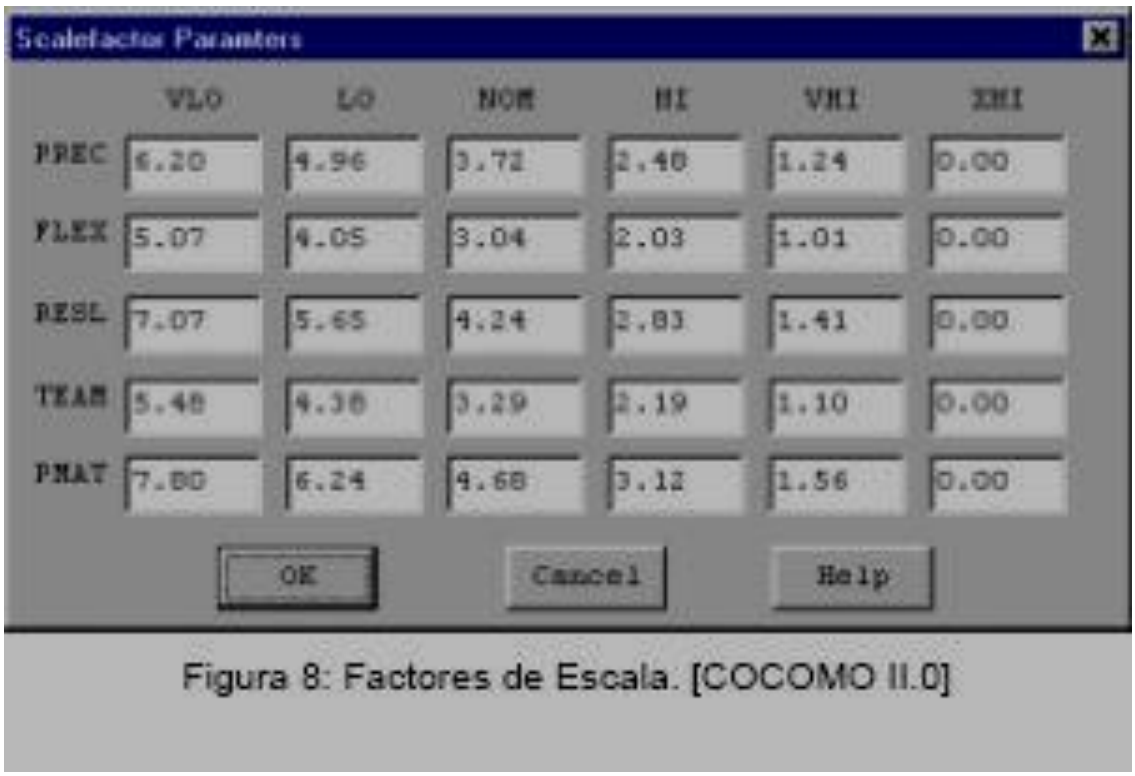


Figura 8: Factores de Escala. [COCOMO II.0]

Figura 4: Multiplicadores de Esfuerzo del Modelo de Diseño Temprano. [COCOMO II.0]