Universidad de las Ciencias Informáticas

Sistema de Mensajería Instantánea para Dispositivos Móviles.

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Darilkis Ramírez Columbié

Yania Crespo Zurita

Tutor: Ing. José Antonio Plá Rodríguez

Ciudad de La Habana, junio 2010.

Declaración de Autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas, para que haga el uso que estime pertinente con este trabajo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2010.

Autores:

Yania Crespo Zurita.

_____

Firma

Darilkis Ramírez Columbié.

_____

Firma

Tutor:

Ing. José Antonio Plá Rodríguez.

_____

Firma

En este momento quiero agradecer de manera especial a mi mamá Dania y a mi papá Adael, que gracias a su esfuerzo y sacrificio hoy estoy aquí. Quiero también agradecerles a los 2 por su amor y apoyo incondicional en los momentos difíciles y también por guiarme siempre por el camino correcto. A mi hermana por confiar en mí. Les quiero agradecer a toda mi familia por darme las fuerzas y ánimos para seguir adelante.

A Diegui por hacerme sentir especial y comprenderme. También para toda su familia que me ha acogido con mucho cariño.

A todos mis amigos por hacerme sentir bien y saber que puedo contar con ellos siempre, al igual que a mis compañeras de cuarto, que la extrañaré muchísimo. Un agradecimiento especial para mi compañera de tesis y amiga.

Darilkis

Quiero agradecerles mucho a mis padres Raquel y Jorge por apoyarme siempre y darme las fuerzas para seguir adelante. A mi hermano Yanier y a mi hermana Yaima por su cariño y amor incondicional. A mi abuela que es una persona muy especial en mi vida. A toda mi familia por confiar en mí y darme ánimos.

A Hectico por comprenderme y quererme durante todo este tiempo, además de apoyarme.

A todos mi amigos que de una manera u otra me han hecho sentir bien durante estos 5 años y por saber que siempre puedo contar con ellos. A mi compañera de tesis por su amistad durante estos 5 años.

Yania

Le agradecemos a nuestro tutor José Antonio Plá por su ayuda incondicional. A nuestro tribunal porque sin sus críticas y sugerencias no hubiésemos llegado hasta aquí. A todos los profesores que han contribuido en nuestra formación durante estos 5 años, a nuestros amigos. A la Revolución, Fidel y la UCI por esta oportunidad en nuestras vidas.

Dedicatoria 2010

A mis padres Dania y Adael que son mi razón de ser.

A mi hermana querida.

A toda mi familia y amigos.

Darilkis

A mis padres Raquel y Jorge, que los quiero muchísimo.

A mis 2 hermanos.

A toda mi familia y amigos.

Yania

Resumen

Hoy en día los teléfonos celulares se han convertido en una herramienta muy importante en la vida de las personas y entre los factores que propician su alta demanda se encuentran la reproducción de música, video, navegación por Internet y la mensajería instantánea. Una de las ventajas que posee este último frente al correo electrónico y a los SMS es que las conversaciones se realizan en tiempo real, aunque una de las dificultades que más afecta a los usuarios es que sólo está disponible para una cantidad muy reducida de móviles.

El objetivo general de este trabajo es implementar un cliente de mensajería instantánea que sea compatible con una amplia gama de dispositivos a partir de la generación 2.5, que garantice entre otras funcionalidades el intercambio de mensajes y así facilitar la comunicación entre los usuarios de este servicio en Cuba.

En este documento se recoge además los resultados del trabajo realizado. Se estudian las características esenciales de los sistemas de mensajería instantánea y de los protocolos existentes para realizar el servicio, también se hace un análisis de las tecnologías y herramientas usadas. Además se muestran los resultados del análisis, diseño e implementación de la propuesta y se recomiendan algunos aspectos para el mejoramiento futuro de la aplicación.

Introducción

La necesidad de comunicarse en cualquier momento y lugar, sin importar la distancia, el tiempo y las condiciones ha llevado al hombre a desarrollar una serie de herramientas en su beneficio que a lo largo de los años se han ido perfeccionando y desarrollando de manera muy significativa. Un ejemplo de ello fue la aparición de los primeros sistemas de telefonía móvil civil que empezaron a desarrollarse a finales de los 40 en EE.UU.

Hoy en día los teléfonos celulares se han convertido en una herramienta muy importante en la vida de las personas. Su gran desarrollo ha permitido la disminución en tamaño y peso, así como la incorporación de pantallas de colores, software amigable al usuario, que cada vez es más exigente. También se debe tener presente que otros de los factores que propician su alta demanda en la actualidad por parte del público joven son los juegos, agendas electrónicas, reproducción de música y video, navegación por Internet y la mensajería instantánea.

Los celulares no contaban con este último hasta los inicios de los años 90, sino con uno similar conocido por *SMS* o *sistema de mensajes cortos*, que como muchas otras tecnologías ha perdido considerablemente interés por parte de los usuarios debido al tiempo que demoran los mensajes en ser recibidos y por la restricción en cuanto a cantidad de caracteres, llegando algunos móviles a 160 caracteres. La figura (tomada del artículo *VeriSign Mobile Messaging Q2 2008*) muestra un crecimiento notable desde el año 2006 – 2010 de la mensajería instantánea (*MI*).

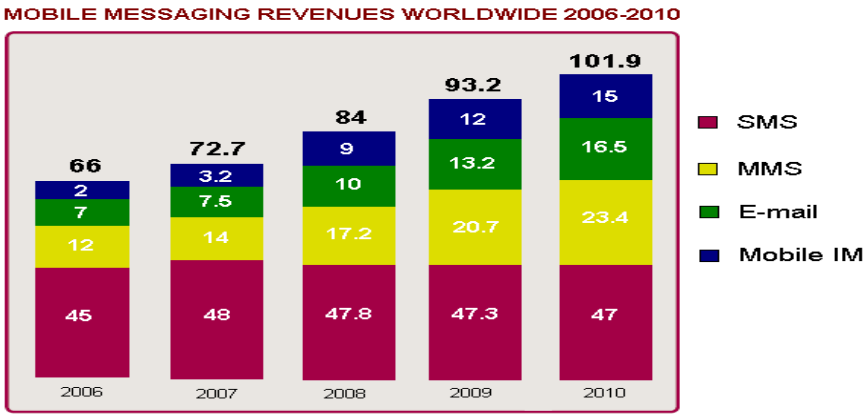

Ilustración 1: Crecimiento de la MI desde el 2006 – 2010.

La mensajería instantánea requiere el uso de un cliente 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 contacto existente en la lista de un usuario se conecta o en qué estado se encuentra, si está disponible para tener una conversación. **(Saavedra, 2009)**

En la actualidad para tener el servicio (*MI*), se necesitan descargar un cliente compatible con su sistema operativo e instalarlo en su teléfono. Pero esto solo está disponible para una pequeña gama de móviles debido a la cantidad de versiones que existen y las variadas características que los mismos poseen.

En nuestro país también se ha ido desarrollando considerablemente la telefonía móvil, aspirando a terminar el 2010 con más de 1 millón de usuarios de este servicio **(Lafuente, 2010)**. Entre las prestaciones que ofrecen, no se encuentra disponible la mensajería instantánea debido a que no existe un cliente orientado a los dispositivos móviles.

Por toda esta problemática que existe en el mundo de las comunicaciones es que se ha propuesto desarrollar un cliente de mensajería instantánea para móviles que soporten Java (MIDP 2.0 y CLDC 1.0) para que sea usado por una mayor cantidad de usuarios en móviles a partir de la generación 2.5.

Teniendo en cuenta todo lo antes expuesto, el **problema científico de la investigación** queda formulado con la siguiente interrogante:

¿Cómo realizar el envío y recepción de mensajes instantáneos entre dispositivos móviles?

El **objeto de estudio** de la presente investigación lo constituye todo el proceso de intercambio de mensajes de manera inmediata entre dispositivos móviles, donde el **campo de acción** está enfocado a los eventos de enviar y recibir mensajes en dispositivos móviles a partir de la generación 2.5 en Cuba.

Partiendo del análisis realizado con la problemática, se planteó que el **objetivo general** es desarrollar un cliente de mensajería instantánea (chat) para dispositivos móviles.

Para llevar un buen desarrollo y guiar la investigación se plantea la siguiente **idea a defender.**

La implementación del cliente de mensajería instantánea para móviles facilitará la comunicación entre los usuarios de este servicio en Cuba.

Para darle cumplimiento al objetivo general del presente trabajo se pretende darle seguimiento a las siguientes **tareas de investigación**:

- Análisis del estado a nivel mundial de la mensajería instantánea en los dispositivos móviles.
- Análisis de los protocolos más utilizados para la creación de servicios MI.
- Análisis de las características de clientes de mensajería instantánea.
- Selección de la metodología, herramientas y lenguaje de programación.

Para la ejecución de estas tareas científicas se utilizan los siguientes **métodos teóricos**:

- Análisis Histórico – Lógico: se estudia la evolución y desarrollo de la mensajería instantánea tanto en internet como en teléfonos móviles, además de estudiar los cambios que la misma ha tenido desde su surgimiento para un buen avance de la aplicación.

Dentro de los **métodos empíricos** se emplean:

- Observación: se realiza un estudio de la situación de la mensajería instantánea teniendo en cuenta los objetivos planteados para observar cómo funciona un cliente de mensajería instantánea, que herramientas usan, así como su importancia en la vida moderna.
- Entrevista: se realizaron conversaciones de intercambio con personas con conocimiento del tema a desarrollar.

Este documento está estructurado por cinco capítulos. A continuación se expone una breve descripción de lo que abordará cada uno de ellos.

Capítulo 1: Se aborda todo lo relacionado con la fundamentación teórica que sustenta la presente investigación acerca del estudio del arte, así como las tendencias, tecnologías y herramientas a utilizar para el desarrollo del sistema.

Capítulo 2: Se identifican y se describen los procesos del negocio al igual que las características del sistema a través de los requisitos funcionales y no funcionales.

Capítulo 3: Se presenta el análisis y diseño de la solución propuesta, lo cual incluye la definición del modelo de clases del análisis, modelo de clases del diseño y los diagramas de colaboración.

Capítulo 4: Muestra cómo va a estar estructurada la implementación del sistema. Contiene los diagramas de componentes y de despliegue.

Capítulo 5: Se realiza el estudio de la duración y costo del sistema.

Capítulo 1. Fundamentación teórica

Introducción

En este capítulo se desarrolla un estudio de los sistemas de mensajería instantánea que existen hoy en el mundo de las comunicaciones, específicamente en los que están disponibles para teléfonos móviles así como el estado del arte a nivel mundial y las tendencias actuales de los protocolos que son usados para el desarrollo de estos clientes. También se expone en este capítulo las herramientas seleccionadas, las características fundamentales de las tecnologías a utilizar para dar solución a la situación problemática, las metodologías y software utilizados que servirán de apoyo para el desarrollo de la aplicación, así como el lenguaje de programación en el que se implementará dicho sistema.

1.1 Antecedentes y estado del arte

Una primera forma de mensajería instantánea fue la implementación en el sistema PLATO usado al principio de la década de 1970. Más tarde, el sistema *talk* implementado en UNIX/LINUX comenzó a ser ampliamente usado por ingenieros y académicos en las décadas de 1980 y 1990 para comunicarse a través de internet. **(Saavedra, 2009)**

ICQ fue el primer sistema de mensajería instantánea para ordenadores con sistema operativo distinto de UNIX/LINUX. Surgió en noviembre de 1996 y fue utilizado de manera masiva en Internet, además fungió como pionero para el desarrollo de diversos mensajeros como: MSN Messenger, Yahoo Messenger, Google Talk y AIM (de AOL). A partir de su aparición, un gran número de clientes de mensajería instantánea han surgido y desarrollado, cada aplicación teniendo su propio protocolo. Esto ha llevado a los usuarios a tener que usar un cliente para cada servicio simultáneamente para estar conectado a cada red de mensajería. **(Saavedra, 2009)**

Características de la mensajería instantánea

Los sistemas de mensajería tienen funciones muy importantes para los usuarios independientemente de las básicas (mostrar los usuarios que hay conectados y chatear) que las personas conocen, a continuación se muestran estas características:

Contactos:

- Mostrar varios estados: Disponible, sin actividad, invisible, no conectado.
- Mostrar un mensaje de estado: Es una palabra o frase que aparece en las listas de contactos de los amigos junto al Nick.
- Registrar y borrar usuarios de la lista de contactos.
- Permite agrupar los contactos: Familia, Trabajo, Facultad, etc.
- Usar un avatar: una imagen que identifique al usuario.

Conversación:

- Puede haber varios tipos de mensajes: Normal, Mensajes de difusión, Chat, GroupChat.
- Muestra cuando otro contacto está escribiendo un mensaje.
- Se puede usar emoticonos.
- Enviar y recibir ficheros: algunos clientes de mensajería instantánea permiten el envío de archivos, generalmente usando protocolos punto a punto (P2P).

1.1.1 Mensajería en el móvil

La mensajería en el móvil ha tenido muchos avances en la actualidad, aunque unido a ello no han dejado de estar presente algunas limitaciones en estos servicios **(Huawei, 2008)**, como por ejemplo:

SMS: Es uno de los servicios más utilizados por los usuarios aunque a estos les gustaría:

- Añadir características personalizadas a los mensajes en el contenido (colores, tipos de letras, etc.).
- Añadir firmas o emoticonos.

Otras limitaciones:

- La cantidad de caracteres es muy pequeña, es decir, los más largos solamente llegan hasta 160.
- Los mensajes enviados o recibidos no se guardan en un buzón de correo externo, por lo que una vez que sean borrados del móvil por la limitación en cuanto a cantidad de memoria, no pueden volver a verse.

- Cuando se envían los mensajes no se puede saber si se envió con éxito, debido a que puede darse el caso que el receptor este apagado o fuera del área de cobertura.

MMS ofrece mejores servicios que los SMS:

- Teniendo en cuenta contenido permite enviar música, imágenes y videos.
- Una limitante es la variedad existente de marcas y modelos de móviles, no siendo compatible este servicio con muchos de estos modelos.

El **correo electrónico móvil** se refiere al envío de correos a través del móvil.

- Se hace uso de una dirección de correo de Internet en vez de un número de teléfono móvil.
- El correo apareció en los móviles en el 2005 pensándose que reemplazaría a los SMS, pero la cantidad de suscriptores y la cantidad de tráfico han sido menores, por la falta de normas internacionales.

La **MI** en el móvil ha ido creciendo considerablemente, aunque se ha visto frenada porque:

- Los clientes de este servicio no son compatibles con todos los modelos y marcas existentes.
- Se cobra al igual que el SMS, siendo esta una de las limitantes para el público joven que es el que más uso hace de ella para la diversión y el entretenimiento.

Todos estos servicios antes mencionados han tenido acogida por los usuarios, pero como se puede ver cada uno de ellos ha tenido dificultades, por lo que los usuarios los catalogan como incompletos o defectuosos y es por ello que este tema sigue siendo una tarea de investigación para sus proveedores. **(Huawei, 2008)**

1.1.2 Ejemplos de clientes de mensajería instantánea

Existen en la actualidad una gran variedad de clientes de mensajería instantánea orientados a los dispositivos móviles. A continuación se exponen las características más importantes de algunos de ellos.

1.1.2.1 MSN Messenger Touch

Es un cliente que posee prácticamente todas las opciones y funciones que el MSN Messenger que se encuentra en las PCs. Entre los que se tienen: **(Fashion, 2009)**

Características generales:

- Compatible para dos terminales Nokia, el Nokia N97 y para el Nokia 5800, que es táctil.
- Presenta una versión en inglés y una en español.
- Permite establecer conversaciones en tiempo real.
- Permite revisar el correo.
- Disponible para la conectividad con MSN Messenger.
- Compatible con redes EDGE, GPRS, y Wifi.
- Permite bloquear los contactos no deseados.
- Ventanas de chat con pestañas.

1.1.2.2 Fring

Fring es un cliente de mensajería instantánea para móviles con gran aceptación por parte del público, encontrándose disponible la conectividad con Skype, MSN Messenger, Google Talk, ICQ, SIP, Twitter, Yahoo! y AIM, con soporte para VoIP, cuya finalidad es la de utilizar Fring como cliente de llamadas de voz en Skype y Gizmo, entre otras redes basadas en el protocolo SIP. **(Fashion, 2009)**

Fring se limita a dar compatibilidad con terminales cuyo sistema operativo sea Symbian 8, Symbian 9.1, Symbian 9.2, Windows Mobile 5, Windows Mobile 6 y Symbian UIQ.

Características generales:

- Llamadas de VoIP sobre una red Wifi utilizando el protocolo SIP.
- Conexiones a servicios de chats.
- Integración dinámica de contactos en tiempo real.
- Conexiones múltiples.

Móviles compatibles:

- Nokia 5230
- Nokia 5530 XpressMusic
- Nokia 5800 XpressMusic
- Nokia N97
- Nokia N97 Mini
- Samsung i8910 Omnia HD
- Sony Ericsson Satio

1.1.2.3 Nimbuzz

Cliente de mensajería para móviles similar a los anteriormente mencionados.

Características generales:

- Se puede conectar a 15 protocolos diferentes, entre ellos: AIM, Gtalk, GaduGadu, ICQ, MySpace, MSN, Twiter, XMPP y Skype. **(Zona Windows, 2009)**
- Presenta pocas opciones, siendo sencillo y agradable.
- Cada contacto incluye un ícono, que permite distinguirlo de los demás.
- Permite organizar los contactos por grupos.
- Se encuentra disponible en inglés y español, así como una versión para PCs.
- Compatible con teléfonos LG de pantalla táctil, Samsung SPH – M540, Samsung Star, Samsung Jet y Nokia 6600.
- Permite a los usuarios hacer uso de salas de chat.

1.1.2.4 eBuddy

Es una versión para dispositivos móviles el cual es muy similar en cuanto a aspecto a la versión de las PCs. **(Softonic, 2008)**

Características generales:

- Con este cliente de mensajería instantánea móvil los usuarios se pueden conectar a Yahoo!, AIM, Google Talk, MSN, Yahoo! e incluso a Facebook.
- Permite recibir mensajes *offline* de los contactos y mandar mensajes a contactos no conectados.
- Para utilizar ebbudy se necesita sistema operativo Symbian OS.
- Interfaz dividida en pestañas.
- No presenta soporte para el chat de Skype, ni servicio de VoIP.

Series compatibles:

- Symbian 9.1 Series 60 3rd Ed.
- Symbian 9.2 Series 60 3rd Ed.
- Symbian 9.4 Series 60 5th Ed.

Es compatible con una amplia gama de teléfonos por ejemplo: Alcatel, HTC, i-Mate, LG, Motorola, Nokia, O2, Orange, PocketPC, Qtek, Sagem, Samsung, Sanyo, Sony Ericsson, Sharp, Siemens y T-Mobile. **(Softonic, 2008)**

1.1.2.5 Palringo

Cliente de mensajería instantánea multiservicio para teléfonos móviles basado en servidores para varias plataformas. **(Softonic, 2009)**

Características generales:

- Permite chatear y enviar mensajes de voz.

- Soporta varios protocolos como Yahoo!, Google Talk, MSN, ICQ y XMPP.
- Los mensajes se despliegan automáticamente y es posible realizar una llamada vía VoIP.
- No presenta soporte para Skype.
- Permite enviar fotos a los contactos por el chat.
- Permite a los usuarios entrar en salas de chat.
- No salva los mensajes de forma cronológica.

Para utilizar Palringo se necesita de los sistemas operativos: Windows Mobile, BlackBerry / RIM, Symbian y Android. **(Softonic, 2009)**

Series compatibles:

- BlackBerry 4.5 y 4.7.
- Windows Mobile 2003 y 2003.2.
- Windows Mobile 6 Std.

Ejemplos de móviles BlackBerry con los que es compatible Palringo:

- BlackBerry 8100 Pearl.
- BlackBerry Bold 9000.
- BlackBerry Curve 8520.
- BlackBerry 9500 Storm.

1.1.2.6 IM^{+}

IM^{+} es un cliente de mensajería que entre sus características más significativas tiene: **(Softonic, 2007)**

- Permite conectar con Messenger, ICQ, AOL, Google Talk y XMPP.

- Compatible con redes Wifi, Bluetooh, GPRS y UMTS.
- Permite ver los contactos en línea, enviar y recibir mensajes de texto y voz.
- Guarda y visualiza los historiales de conversación.
- Controla múltiples conversaciones a la vez, gestionar contactos y modificar estados.
- Permite la transferencia de archivos.

Es compatible con los sistemas operativos BlackBerry / RIM y Symbian OS.

Series compatibles:

- Symbian 9.4 Series 60 5th Ed.
- Symbian 7 UIQ 2.1.
- Symbian 7 Series 60 2.0 y Symbian 7 Series 60 2.1.
- Symbian 9.1 Series 60 3rd Ed.
- BlackBerry 4.0 – 4.5.
- BlackBerry 3.2 – 3.8.

Ejemplos de móviles con lo que es compatible IM^{+}:

- BlackBerry 3.2 – 3.8.
- BlackBerry 4.0 – 4.5.
- Nokia 6260, 6620, 7610, 3230.
- Panasonic X700, X701, X800.
- Samsung SGH – D710, SGH – D720, SGH – D728 y SGH – D730.
- Sony Ericsson P900, P910, P910a, P910c, P910i.

1.1.3 Protocolos para la mensajería instantánea

En la actualidad existe una gran variedad de protocolos que son usados en la mensajería instantánea, algunos de ellos son propietarios y por tal motivo no ofrecen documentación ni dan acceso a sus fuentes, por otra parte, hay otros que son libres y están muy bien documentados a disposición de todos los usuarios. Estos protocolos son usados por los clientes anteriormente mencionados.

1.1.3.1 Skype y SIP

El protocolo y código de Skype son cerrados y propietarios aunque la aplicación se puede descargar gratuita en el sitio oficial. Conecta a los usuarios vía texto, voz o video. El programa fue desarrollado en Pascal y más tarde exportado a GNU/Linux. Skype utiliza AES (*Advanced Encryption Standard*) para cifrar la voz, la transferencia de datos o mensajes instantáneos y utiliza una llave asimétrica para evitar ataques. Es compatible con Windows XP, Windows Mobile, iPhone OS, Symbian s60 5th Edition, entre otros. **(Díaz García, 2008)**

Otro de los protocolos usados por algunos clientes de mensajería es SIP (*Session Initiation Protocol)* que permite el control y señalización, mayoritariamente usado en los sistemas de telefonía IP.

Entre las funciones que SIP posee se tienen las siguientes:

- Localización de usuarios (proporciona soporte para la movilidad).
- Disponibilidad del usuario.
- Establecimiento y mantenimiento de una sesión.
- Autenticación y encriptación que son soportados por SSL/TLS.

1.1.3.2 IRC (*Internet Relay Chat*)

IRC es un protocolo de comunicación en tiempo real basado en texto, que permite la conferencia entre 2 o más personas y que está clasificado dentro de la mensajería instantánea. Las conversaciones se desarrollan en los llamados canales de IRC, que pueden ser locales al servidor al que se conectan los clientes o no. **(Díaz García, 2008)**

Los usuarios del IRC utilizan una aplicación cliente para conectarse con un servidor en el que funciona una aplicación IRCd (IRC Daemon o servidor IRC), que gestiona los canales y las conversaciones. Para adicionarle seguridad al protocolo se puede utilizar SSL opcionalmente. También se puede decir que un servidor se conecta con otros servidores para expandir la red IRC e intercambian todo su tráfico, de forma tal que todos los servidores de una red tienen copia de todos los mensajes de las salas de chat y así se crea la ilusión del lado de los clientes de que están conectados a un gran servidor. La manera en que los usuarios acceden a las redes IRC es conectando un cliente al servidor. **(Díaz García, 2008)**

Existen diversas implementaciones de clientes (mIRC o X - Chat) IRC así como de servidores. La mayoría de los servidores no necesitan que los usuarios se registren, aunque se necesita que los usuarios establezcan un nick (*alias*) antes de conectarse. Este protocolo se basa en la arquitectura cliente – servidor y es adecuado para funcionar en varias máquinas de un modo distribuido.

Las redes más grandes de IRC conocidas son: DALnet, EFnet (la primera entre las más grandes con 60 servidores), IRCnet.org e IRCnet.com. Una de las redes más grandes se encuentra en España conocida como IRC – Hispano.

1.1.3.3 OSCAR

El protocolo OSCAR (*Open System for Communication in Realtime*), oficial del programa de mensajería de AOL, AIM (sistema de MI de AOL) y también usado por ICQ. Es un protocolo de desarrollo propietario y no ofrece documentación ni código. Para conocer su forma de actuar y adaptar sus programas existen desarrolladores que han recurrido a la ingeniería inversa. OSCAR funciona con 3 características importantes, primero realiza la autenticación del usuario, luego el envío y recepción de datos en servidores centrales BOS (*Basic OSCAR Service*), y finalmente el ChatNav (*navegador del chat*) donde se crean las salas de charlas. **(Díaz García, 2008)**

Este protocolo utiliza paquetes binarios de longitud variable, de forma que permite una amplia variedad de servicios (chat, directorio, gestión, localización, etc.). Los clientes no se conectan directamente entre sí, lo hacen a través de servidores, que se responsabilizan de la entrega de los mensajes a sus destinatarios.

La red consiste en múltiples servidores centrales BOS y un servidor de autorizaciones, que antes de que los clientes se puedan conectar a los servidores BOS o a otros, éste debe autorizarlo primero. Este

proveerá al cliente de una “cookie” que le permitirá conectarse al resto de los servidores de manera automática, para poder utilizar el resto de los servicios disponibles en la red de MI.

El servidor de autorizaciones también redirige el cliente a un servidor BOS predeterminado según las preferencias de conexión del cliente, pudiendo ser redirigido para cualquier otro y así equilibrar la carga de trabajo entre servidores.

1.1.3.4 XMPP

La especificación base de Jabber (más tarde XMPP) surgió en 1998 por Jeremie Miller, conocido como el primero de carácter abierto y tomado como protocolo por la comunidad Open Source en 1999, donde ha ido creciendo y evolucionando hasta la actualidad. Ver figura 8 para más detalles sobre su historia.

XMPP (*Extensible Messaging and Presence Protocol*), es un protocolo abierto y extensible, con él queda establecida una plataforma para el intercambio de datos XML, que puede ser usado entre aplicaciones de Internet para mensajería instantánea, aunque originalmente fue ideado para la misma. Posee muchas implementaciones abiertas de servidores, clientes y librerías para las más diversas plataformas y lenguajes. Este protocolo en su funcionamiento topológico se basa en la clásica arquitectura cliente – servidor, aunque no fuerza a hacerlo así y mediante TLS permite cifrar los mensajes empleando diferentes algoritmos como *RSA* y *DSS*. **(Díaz García, 2008)**

Hasta la fecha XMPP ha tenido buena aceptación como alternativa libre con respecto al MSN Messenger de Microsoft, al AIM de AOL, que como se explicó anteriormente son propietarios. Hay que tener en cuenta que aunque es un protocolo joven, su uso ha ido creciendo considerablemente. Un ejemplo de ello es Google Talk, un cliente de mensajería instantánea que utiliza el protocolo y entre otras funcionalidades permite la transferencia de archivos, intercambio mediante voz y texto, así como el aviso de presencia de los usuarios y su estado de disponibilidad.

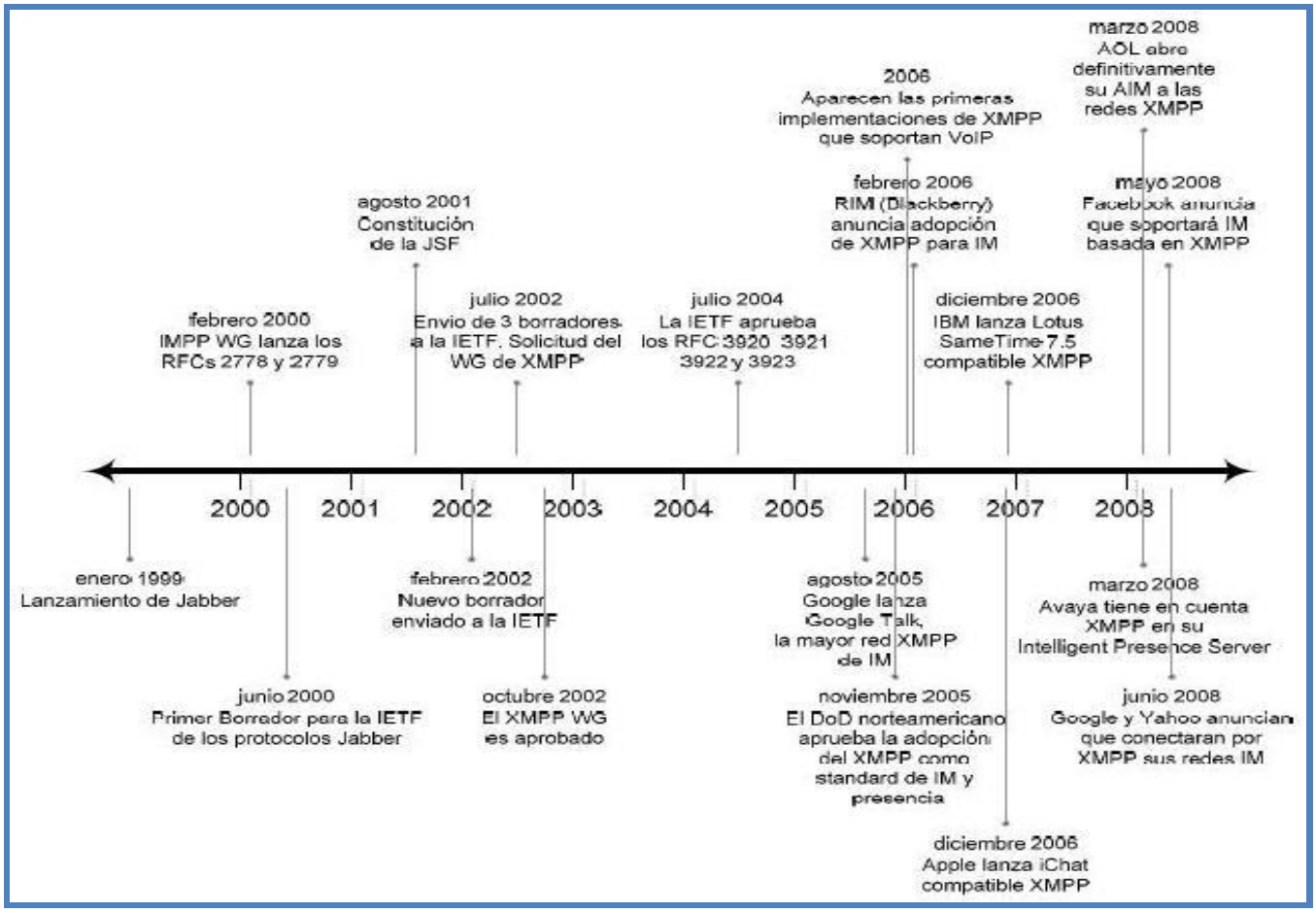

Ilustración 2: Historia del XMPP hasta el 2008.

Entre las características más significativas que posee el protocolo en cuanto a adaptabilidad y sencillez se tienen las siguientes:

- **Abierto, público y libre**

  El protocolo XMPP está ampliamente documentado y puesto a disposición del público por la Jabber Foundation, en el sitio web de la misma. Además, cualquier individuo, organización o empresa pueden hacer uso de él para sus tecnologías de comunicación sin tener que pagar derechos de autor o licenciarlo gracias a que es abierto y multiplataforma. **(Jabber, 2008)**

- **Estándar**

  XMPP está avalado por la organización imparcial IETF que se encarga de su registro y conservación. Al ser un estándar que ha seguido un riguroso proceso de regularización, no es concebible que cambie de la noche a la mañana sin previo aviso, por ejemplo, por los intereses comerciales de una empresa.

- **Probado**

  Numerosas organizaciones privadas, grupos de usuarios y universidades usan la tecnología XMPP como sistema de comunicación, siendo notable la Universidad Jaume I de Castellón, que alberga uno de los servidores Jabber más grandes de España. **(JabberES, 2007)**

- **Descentralizado**

  Su topología es distribuida. Cualquier individuo u organización pueden fácilmente montar un servidor para dar servicio a una comunidad de usuarios en régimen público o privado.

- **Seguro**

  Las comunicaciones punto a punto, pueden ir aseguradas bajo una capa segura SSL. De este modo, se cifra la información que se intercambia. Adicionalmente, es habitual el uso de una firma o cifrado GPG (o PGP) de cada mensaje para autentificar la autoría del mismo por parte de los clientes. **(JabberES, 2007)**

- **Extensible**

  XMPP se apoya en una potente característica de XML, los espacios de nombres. Al comportarse un servidor de XMPP básicamente como un router de XML, basta con definir nuevos espacios de nombres sobre los ya existentes para extender las funcionalidades de Jabber sin modificar en ningún momento los servidores existentes. El "peso" del protocolo recae en los clientes, que son los que interpretan los datos XML que reciben. **(Jabber, 2008)**

- **Flexible**

  XMPP es un potente y versátil protocolo que permite a dos aplicaciones (cualesquiera) comunicarse eficientemente usando XML. Se conocen usos de XMPP para comunicar bots, autómatas, etc.

Entre otra de sus posibilidades de uso, ofrece servicios tales como:

- Un directorio de usuarios.
- Salas de charla pública o puentes a otras mensajerías (como el email o el MSN).

Un cliente con este protocolo se conecta a uno o varios servidores simultáneamente, de este modo puede ponerse en contacto con otros clientes conectados a esos servidores, esto se puede lograr con pasarelas o puertos de enlaces (*gateways*) que traducen de XMPP a otros protocolos de mensajería de diferentes redes. De todos modos, si se quiere entrar en contacto con clientes conectados a otros servidores o viceversa, basta con especificar la dirección de los mismos y los servidores implicados se pondrán en contacto entre sí de forma transparente al usuario.

XMPP está orientado a los mensajes, que pueden ser de tipos diferentes entre los que se encuentran:

- Normal: mensajes parecidos a los de correo electrónico.
- Chat: los mensajes utilizados entre 2 personas en una conversación.
- Groupchat: mensajes enviados a un grupo de personas.
- Error: para los mensajes de error.
- Jabber:x:oob: para las conexiones directas entre clientes para el envío de archivos.

Se debe tener presente que los usuarios usan los mensajes de tipo *chat* para enviarse entre ellos mensajes instantáneos, los mismos suelen ser cortos, además normalmente son mostrados en una interfaz de tipo chat en la que el usuario puede ir viendo lo que escribe él y lo que escribe su contacto.

Direccionamiento de los mensajes

Para el envío de mensajes el protocolo lo hace de una manera muy sencilla, los paquetes son enviados por el usuario a un receptor. Por defecto, no se reciben confirmaciones de que los mensajes han sido recibidos para reducir el tráfico en el servidor, además si el receptor no se encuentra disponible, el servidor guarda los mensajes hasta que se conecte. A este comportamiento particular se le denomina *store and forward.* El destinatario puede estar conectado al mismo servidor o no, en este último caso el remitente necesita establecer conexión con el otro servidor y entregar el mensaje, que tendrá que ser enrutado hacia el servidor destino.

Servidores XMPP

Son muy variadas las implementaciones de software de servidores entre las que se encuentran más de 10 que son propietarias/comerciales como son Jabber XCP, TIMP.NET, Jerry Messenger, Antepo OPN, etc. También de este tipo de implementaciones existen alrededor de 15 Open Source, dentro de las que se destacan:

- Citadel: Desarrollado en lenguaje C. Ofrece correo electrónico, calendario, libreta de direcciones, mensajería instantánea y otras funcionalidades en un mismo paquete integrado. Disponible en http://www.citadel.org/.
- DJabberd: Programa en lenguaje Perl. Disponible en http://danga.com/djabberd/.
- Jabberd14: Programa en lenguaje C/ C++. Disponible en http://jabberd.org/.
- Ejabberd: Programado en Erlang, mantenido por la empresa ProcessOne y por una comunidad de desarrolladores cuya sede web se encuentra en http://www.ejabberd.im/.

De estos servidores se hará uso para la aplicación de ejabberd que es multiplataforma, de código abierto, distribuido, tolerante a fallos y basado en estándares abiertos para lograr la comunicación en tiempo real (XMPP). Entre otras de sus características es de fácil instalación, es decir, no es necesario configurar una base de datos externa o un servidor externo, etc. porque ya todo está listo en un mismo paquete. Además, este protocolo al igual que sus servidores y librerías están ampliamente documentado (dígase videos y tutoriales) en varios sitios web, por lo que será usado en la implementación del cliente de mensajería.

1.2 Metodología, lenguaje y herramientas de desarrollo

Para desarrollar el sistema de mensajería instantánea se hace necesario analizar las características fundamentales de la metodología, lenguajes tanto para el modelado como para la codificación, herramientas de desarrollo y tecnologías de punta. A continuación se caracterizaran detalladamente cada uno de los elementos antes mencionados.

1.2.1 Metodología de desarrollo RUP

"Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. Sin embargo, el Proceso Unificado es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto". **(Ivar Jacobson, 2000)**

RUP es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado, constituye una metodología estándar muy utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

Provee un enfoque disciplinado en la asignación de tareas y responsabilidades dentro de una organización de desarrollo. Su meta es asegurar la producción de software que satisfaga las necesidades de los usuarios finales, dentro de un calendario y presupuesto predecible. RUP divide el proceso en cuatro fases (Inicio, Elaboración, Construcción y Transición), dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en las distintas actividades. Su ciclo de vida se caracteriza por estar dirigido por casos de uso, iterativo e incremental y centrado en la arquitectura.

Beneficios de un proceso iterativo controlado:

- Reduce el costo del riesgo a los costos de un solo incremento. Si los desarrolladores tienen que repetir la iteración, la organización solo pierde el esfuerzo mal empleado de la iteración, no el valor del producto entero.

- La iteración controlada acelera el ritmo del esfuerzo de desarrollo en su totalidad, debido a que los desarrolladores trabajan de manera más eficiente para obtener resultados claros a corto plazo, en lugar de tener un calendario largo que se prolongue eternamente.
- Reconoce una realidad que a menudo se ignora, que las necesidades del usuario y sus correspondientes requisitos no pueden definirse completamente al principio. Típicamente, se refinan en iteraciones sucesivas, esta forma hace más fácil la adaptación a los requisitos cambiantes.

Otras ventajas de la aplicación de esta metodología son las siguientes:

- Distribuye la carga de trabajo a lo largo del tiempo del proyecto ya que todas las disciplinas colaboran en cada iteración.
- Genera un volumen considerable de documentación, posibilitando que los cambios realizados en los miembros del equipo no resulten un problema para el avance del proyecto.

1.2.2 Lenguaje de modelado UML

Lenguaje de Modelado Unificado (*Unified Modeling Language*) es un lenguaje gráfico para especificar, construir, visualizar y documentar todas las disciplinas de un proyecto informático: desde el análisis con los casos de uso, hasta la implementación y configuración mediante los diagramas de despliegue. **(James Rambaugh, 1998)**

UML permite:

- Especificar cuáles son las características de un sistema antes de su construcción.
- Construir sistemas diseñados a partir de modelos especificados.
- Visualizar gráficamente un sistema de manera que otros puedan entenderlo.
- Documentar los elementos gráficos del sistema desarrollado para futuras revisiones.
- Verificar y validar el modelo realizado.
- Generar código a partir de los modelos y viceversa.

1.2.3 Lenguaje de programación Java

Es un lenguaje orientado a objetos, eso implica que su concepción es muy próxima a la forma de pensar humana. Desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. **(López, 1997)**

También posee otras características muy importantes como son:

- Aprovecha características de la mayoría de los lenguajes modernos evitando sus inconvenientes (en particular los del C++).
- Tiene una gran funcionalidad gracias a sus librerías.
- Seguro: La máquina virtual, al ejecutar el código java, realiza comprobaciones de seguridad.
- Multiplataforma: El mismo código java que funciona en un sistema operativo, funcionará en cualquier otro sistema operativo que tenga instalada la máquina virtual java.
- Indiferente a la arquitectura: Soporta aplicaciones que serán ejecutadas en los más variados entornos de red. Para ello, el compilador de Java genera *bytecodes*: un formato intermedio indiferente a la arquitectura, diseñado para transportar el código eficientemente a múltiples plataformas de hardware y software. **(López, 1997)**
- En este lenguaje se pueden programar aplicaciones independientes, como con cualquier otro lenguaje de propósito general y applets (pequeñas aplicaciones que se ejecutan en un documento HTML, siempre y cuando el navegador soporte Java). **(Flanagan, 1999)**

1.2.4 Plataforma de desarrollo JME

La plataforma JME (*Java Micro Edition*) es una familia de especificaciones que definen varias versiones minimizadas de la plataforma Java 2 para entornos más limitados; estas versiones minimizadas pueden ser usadas para programar en dispositivos electrónicos; desde teléfonos celulares, en PDAs y hasta en tarjetas inteligentes. Es una versión orientada a los dispositivos móviles, debido a que estos tienen una

potencia de cálculo baja e interfaces de usuario pobres además no disponen de abundante memoria ni mucha potencia en el procesamiento. **(Diego Blanco Moreno, 2009)**

Para una ejecución de Java para JME que cumpla con un rango amplio de dispositivos es necesario que se componga de configuraciones, perfiles y paquetes opcionales. **(JavaME, 2010)**

Basadas en JME fueron así definidas dos configuraciones: la CDC (Connected Device Configuration) y la CLDC (Connected Limited Device Configuration), las cuales describen las características mínimas en cuanto a hardware y software. Como el nombre deja ver, la segunda corresponde a una aplicación del MIPD (define las características más específicas de hardware y software y las APIs) para dispositivos de coste más bajo y con menos recursos, tales como teléfonos móviles. **(JavaME, 2010)**

CLDC

Esta configuración está diseñada para dispositivos con conexiones de red intermitentes, procesadores lentos y memoria limitada como son teléfonos móviles, asistentes personales (PDAs), etc. Está orientado a dispositivos que cumplan las siguientes características:

- Procesador: 16 bits/16 MHz o más.
- Memoria: 160-512 KB de memoria total disponible para la plataforma Java.
- Alimentación limitada, a menudo basada en batería.
- Trabajo en red: Conectividad a algún tipo de red, con ancho de banda limitado habitualmente.

CLDC 1.1 es una revisión de la especificación CLDC 1.0 e incluye nuevas características como son punto flotante o soporte a referencias débiles, junto con otras mejoras. CLDC 1.1 es compatible con versiones anteriores y sigue soportando dispositivos pequeños o con recursos limitados.

MIDP

Este perfil se combina con la configuración CLDC para proporcionar un entorno de ejecución para dispositivos móviles (teléfonos, PDAs, etc.), que a su vez incrementa la capacidad de los dispositivos y reduce el consumo de memoria y energía. Las MIDP permiten tener aplicaciones intuitivas y gráficas,

además se pueden instalar y ejecutar en local, trabajar en red o de forma desconectada y pueden almacenar y gestionar de forma segura datos en local.

Para poder ejecutar aplicaciones MIDP 2.0, los dispositivos deberán tener las siguientes características mínimas:

- Tamaño de Pantalla de 96 x 54 píxeles con un bit de profundidad de color.
- Entrada por teclado o pantalla táctil.
- Memoria de 256 KB no volátil para la aplicación MIDP, 8 KB no volátil para datos persistentes y 128 KB volátil para el entorno de ejecución Java.
- Conexión a redes bidireccionales con acceso inalámbrico posiblemente intermitente con ancho de banda limitado.
- Capacidad para reproducir sonidos.

MIDP 2.0 es la versión revisada y mejorada de la especificación MIDP 1.0.

1.2.5 Herramienta case Visual Paradigm

Visual Paradigm para UML (VP-UML) es una herramienta Case multiplataforma (Windows/Linux/Mac OS X) que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y genera documentación. **(UML Tool, 2009)**

Esta herramienta está diseñada para una amplia gama de usuarios. VP-UML soporta los últimos estándares de la notación UML.

VP-UML 6.4 incluye mejoras como:

- Mayor velocidad en el proyecto abierto.
- Reducción de consumo de recursos para el proyecto.
- Buena integración con IDEs.
- Incluye versiones en español.

- Muy personalizable.
- Soporta múltiples lenguajes de programación.
- Aumenta la producción automática de código (Java), bases de datos y generación de informes.
- Interoperabilidad con modelos UML2 (metamodelos UML 2.x para plataforma Eclipse) a través de XMI.
- Multiplataforma.

1.2.6 IDE Eclipse

Eclipse es un entorno de desarrollo integrado (*IDE, Integrated Development Environment*) de código abierto y multiplataforma (GNU/Linux, Solaris, Mac OSX), desarrollado originalmente por IBM, actualmente propiedad de la Fundación Eclipse. La compilación es en tiempo real. Posee integración con Ant, asistentes (wizards) para creación de proyectos, clases, tests y refactorización. **(Eclipse, 2008)**

De manera general, presenta las siguientes características:

- Es soportado para distintas arquitecturas (x86, 64 bits,..).
- Posee una estructura de plug-in que hace sencillo añadir nuevas características y funcionalidades.
- Dispone de un control de versiones integrado a Subversión.
- Incluye muchas utilidades de edición que ayudan al programador a desarrollar el producto con rapidez, algunas son: resaltado de sintaxis, autocompletado de código, tabulador de un bloque de código seleccionado y formateado automático de código, tributando a la obtención de código de mayor calidad. **(EclipseME, 2009)**

Conclusiones

En este capítulo se realizó una descripción panorámica general de las características de la mensajería instantánea y algunos ejemplos de clientes para este servicio. También se hizo referencia a los protocolos más usados, y así se determinó el más adecuado para la implementación del sistema. Se trataron una

serie de conceptos importantes relacionados con el objetivo del trabajo y quedó reflejado de forma resumida la metodología de desarrollo, herramientas y tecnologías que serán utilizadas. Se estableció Eclipse como entorno de desarrollo, Java como lenguaje de programación con la plataforma JME, así como el protocolo XMPP para el intercambio de mensajes entre los usuarios, metodología de desarrollo RUP y UML como lenguaje de modelado.

Capítulo 2. Características del sistema

Introducción

En el presente capítulo se tiene como objetivo realizar una valoración detallada de las principales características del producto a desarrollar, complementando las necesidades que dieron origen al mismo. Se pretende describir las funcionalidades que serán implementadas y la especificación de los requerimientos funcionales y no funcionales del sistema a implementar, así como la arquitectura del sistema.

2.1 Arquitectura cliente – servidor

La arquitectura de software o también llamada arquitectura lógica de un programa, es una vista del sistema que incluye los componentes principales del mismo, así como la conducta de esos componentes según se percibe desde el resto del sistema y las formas en que estos componentes interactúan y se coordinan para alcanzar la misión del sistema. **(ISW 1, 2009)**

En un sistema cliente – servidor, la interfaz de usuario siempre se ejecuta en el cliente y la gestión de datos siempre es proporcionada por un servidor compartido. La funcionalidad de la aplicación puede ser implementada en el cliente o en el servidor. Estos sistemas permiten compartir recursos, son abiertos, escalables, concurrentes, tolerantes a defectos y transparentes. **(Sommerville, 2005)**

Cliente-servidor se recomienda, en particular, para redes que proporcionen un alto grado de fiabilidad. Sus principales ventajas son:

- Sus recursos son centralizados debido a que el servidor es el centro de la red y puede administrar los recursos que son comunes a todos los usuarios.
- Seguridad mejorada ya que la cantidad de puntos de entrada que permite el acceso a los datos no es importante.
- Administración a nivel de servidor ya que los clientes no desempeñan un papel importante en este modelo, requiere menos administración.

- Gracias a esta arquitectura, es posible quitar o agregar clientes sin afectar el funcionamiento de la red y sin necesidad de realizar mayores modificaciones.

Esta arquitectura a pesar de las ventajas que posee tiene como desventaja fundamental que es fácil la congestión del tráfico cuando muchos clientes envían simultáneamente gran cantidad de peticiones al mismo servidor pudiéndoles causar problemas a éste.

2.2 Propuesta del sistema

Se propone el desarrollo de un sistema que permita el intercambio de mensajes entre dispositivos móviles.

El sistema estará basado en la arquitectura cliente – servidor. La lógica del sistema se implementará en el cliente utilizando la plataforma JME, donde se realizarán las funcionalidades de enviar y recibir mensajes, añadir y eliminar de la lista de contactos, así como la selección por parte del usuario del estado de disponibilidad que por defecto estará en línea. Como política de seguridad se tendrá la autenticación por parte del usuario que también se desarrollará en el cliente, donde a través de la interfaz tendrá que llenar los campos que serán de carácter obligatorio para poder acceder al sistema.

La aplicación será capaz de mostrar una interfaz donde se mostrará la conversación con un determinado contacto, en caso de que otro envíe un mensaje será mostrado en la misma interfaz. Para enviar un mensaje al último usuario que escribió, podrá elegir el contacto en la lista y podrá ver todos los mensajes, incluido el de otro usuario que escriba.

El cliente implementado se conectará al servidor Ejabberd 2.1.3 (se puede usar otro servidor) donde estarán disponibles todas las cuentas de usuarios, el servidor se encargará de recoger las últimas actualizaciones de las mismas. Las cuentas no podrán ser gestionadas a menos que sea por un administrador, que será el único autorizado a modificar datos en el servidor.

2.3 Modelo de dominio

Se llegó a la conclusión de que no se modelaría el negocio a través de un diagrama de casos de uso, sino que se implementará un modelo de dominio, el cual es un subconjunto del modelo de objeto del negocio donde se capturan los tipos más importantes de objetos en el contexto del sistema o los eventos que suceden en el entorno en el que trabaja el sistema. **(Ivar Jacobson, 2000)**

En el modelo de dominio los objetos o los eventos se representan como clases y con la cardinalidad existentes entre ellas.

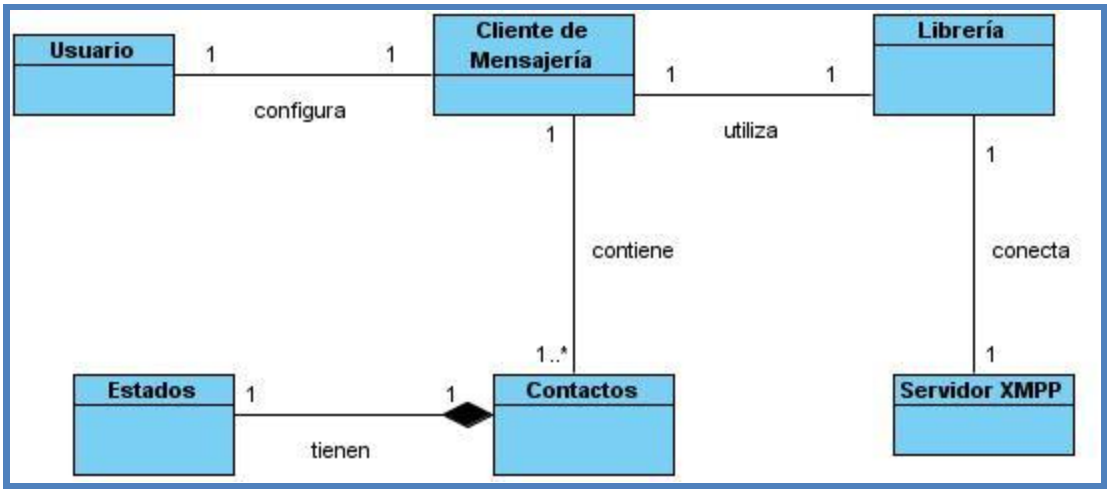

Ilustración 3: Modelo de dominio del Sistema de Mensajería Instantánea.

Descripción de los conceptos del dominio

A continuación se describe brevemente los conceptos representados en el modelo de dominio, para una mejor comprensión de las relaciones que se establecen en el mismo.

Usuario: Es la persona que va a trabajar con el sistema directamente, es decir, quien lo instalará y configurará en su móvil.

Cliente de Mensajería Instantánea: Sistema mediante el cual, una vez instalado en el móvil el usuario podrá conectarse al servidor y sostener un diálogo con los usuarios de su lista, ver sus estados y realizar otras operaciones.

Servidor XMPP: Es donde estarán almacenados todos los contactos y permitirá la conexión.

Contactos: Personas existentes en la lista del cliente de mensajería, con los cuales el usuario podrá sostener conversaciones en tiempo real.

Librería: Librería usada donde se implementa el protocolo XMPP.

Estados: Es la disponibilidad que tienen los contactos para establecer el diálogo, por defecto cuando se accede al sistema aparece *online*.

2.3.1 Descripción del modelo de dominio

El usuario que posee un dispositivo móvil es la persona que va interactuar directamente con el cliente de mensajería instantánea, una vez que lo instale y lo configure en su teléfono. Cuando se ejecute la aplicación y el usuario se autentique, el sistema se conectará al servidor XMPP haciendo uso de la librería que implementa dicho protocolo, para establecer comunicación con los contactos existentes en su lista de acuerdo con el estado de disponibilidad que los mismos posean, y así realizar los eventos enviar y recibir mensajes, añadir y eliminar contactos.

2.4 Relación de los requerimientos funcionales y no funcionales del sistema

A continuación se detallan los requerimientos funcionales y los no funcionales, que el sistema debe tener para la aceptación de todos los usuarios.

2.4.1 Requerimientos funcionales

R1: Autenticar.

R2: Modificar estado de disponibilidad.

R2.1: Mostrar ícono de estado.

R2.2: Mostrar opciones para seleccionar el estado deseado.

R3: Chatear.

R3.1: Seleccionar el contacto deseado.

R3.2: Mostrar opción para enviar mensaje a un contacto determinado.

R3.3: Mostrar mensajes recibidos.

R3.4: Mostrar el diálogo existente.

R4: Gestionar contacto.

R4.1: Adicionar usuario a la lista.

R4.1.1: Mostrar opción para añadir un contacto.

R4.2: Eliminar usuario de la lista.

R4.2.1: Mostrar opción para eliminar determinado contacto.

2.4.2 Requerimientos no funcionales

"Los requerimientos no funcionales especifican propiedades del sistema, como restricciones del entorno o de la implementación, rendimiento, facilidad de mantenimiento, extensibilidad y fiabilidad." **(Ivar Jacobson, 2000)**

Apariencia e interfaz interna:

RNF1: El sistema interactúa mediante una interfaz amigable y fácil de utilizar, dirigida al usuario.

RNF2: El sistema cuenta con un código que posee clases con nombres sugerentes al igual que los métodos de acuerdo con la funcionalidad que implementan, también poseen comentarios para facilitar la comprensión del mismo por desarrolladores, en caso de que se desee hacerle futuras versiones a la aplicación.

Usabilidad:

RNF3: Diseño sencillo, permitiendo al usuario que no sea necesario pasar un entrenamiento para poder hacer uso de la aplicación.

RNF4: Debe ser escalable para poder agregar nuevas funcionalidades sin tener que afectar las que ya están funcionando.

Eficiencia:

RNF5: La eficiencia estará determinada fundamentalmente por la velocidad que se logre entre la conexión del móvil y el servidor.

Portabilidad:

RNF6: El sistema será compatible con móviles que soporten Java (MIDP 2.0 y CLDC 1.0).

Seguridad:

RNF7: Se garantizará que el usuario tenga que autenticarse al ejecutar el sistema.

Software:

RNF8: Se utilizará como lenguaje de programación Java, con la plataforma de desarrollo Java ME.

Hardware:

RNF9: Requiere un móvil con tamaño de pantalla mínima de 96 x 54 píxeles.

RNF10: Memoria de 160 - 512 KB para el entorno de ejecución Java (128).

RNF11: Entrada por teclado o pantalla táctil.

RNF12: Conectividad algún tipo de red.

RNF13: Alimentación limitada (batería).

2.5 Modelo de casos de uso del sistema

Mediante el modelo de Casos de Uso del Sistema se representan las funcionalidades deseadas y el entorno de la aplicación mediante actores y casos de uso. Sirve además para sentar las bases necesarias para el desarrollo del análisis y diseño. **(Ivar Jacobson, 2000)**

2.5.1 Definición de los actores del sistema

Los actores del sistema son aquellas personas que interactúan con el sistema una vez terminado.

Tabla 1: Descripción de los actores del sistema.

Actor	Justificación
Usuario	Es el encargado de instalar y configurar la aplicación en el móvil.

2.5.2 Descripción de los casos de uso

Los casos de uso son artefactos que describen en forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Su importancia además de formar parte de la documentación de la aplicación, está en que es la guía para que el actor pueda usar el sistema correctamente. Las siguientes tablas que se exponen a continuación presentan los casos de uso determinados para satisfacer los requisitos funcionales del sistema. **(ISW, 2009)**

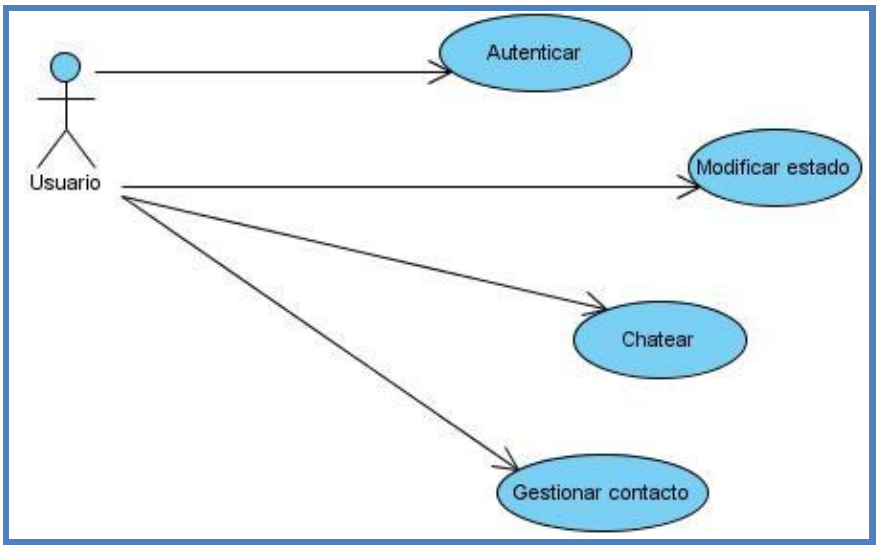

Ilustración 4: Modelo de Casos de Uso del Sistema.

Tabla 2: Descripción del caso de uso, Autenticar.

Caso de Uso	Autenticar
Actor	Usuario
Resumen	El caso de uso se inicia cuando el usuario solicita acceder a las acciones del sistema mediante la selección del check box “Jabber”. El sistema brinda una interfaz para insertar el usuario, contraseña, servidor y puerto.
Precondiciones	Que todos los campos estén llenos.
Referencias	RF1
Prioridad	Crítico
Flujo normal de eventos	

Acción del actor	Respuesta del sistema
1. El usuario desea acceder al sistema y selecciona el check box “Jabber” en el menú principal.	2. Muestra la interfaz de autenticación Menú.
3. Introduce usuario, contraseña, servidor, puerto y oprime el comando “OK”.	4. Comprueba que el usuario y la contraseña sean válidos.
	5. Da acceso al sistema.

Prototipo de interfaz

Flujos alternos 4.a: Usuario y contraseña incorrectos.

Acción del actor	Respuesta del sistema
	4a.1 Indica que el usuario o la contraseña son incorrectos. Ir a la acción 2.

Flujo alterno a: Back

Acción del actor	Respuesta del sistema

a.1 Oprime el comando “Atrás”.	a.2 Va hacia el menú principal.
Poscondiciones	No aplica.

Tabla 3: Descripción del caso de uso, Modificar estado.

Caso de Uso	Modificar estado de disponibilidad.
Actor	Usuario
Resumen	El caso de uso se inicia cuando el usuario selecciona la lista desplegable de estados y selecciona el deseado. Luego oprime el comando Menú y selecciona la acción Cambiar Estado.
Precondiciones	El usuario conectado por defecto aparece “*online*”.
Referencias	RF2
Prioridad	Secundario
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. Cuando el usuario accede al sistema, aparece por defecto “*online*”.	2. Muestra la interfaz que contiene por defecto el estado “*online*”.
3. Si el usuario decide cambiar el estado, selecciona en el menú desplegable el estado deseado y oprime el comando “Menú” la opción Cambiar Estado, para guardar el cambio.	4. Muestra las opciones del comando “Menú” y guarda los cambios.
Prototipo de interfaz	

Flujos alternos 1.a: Dejar por defecto el estado *online*.

Acción del actor	Respuesta del sistema
1a.1 El actor deja por defecto el estado *online*.	1a.2 Muestra la interfaz con el estado por defecto.

Prototipo de interfaz

Flujo alterno a: Desconectar	
Acción del actor	**Respuesta del sistema**
a.1 Oprime el comando “Desconectar”.	a.2 Se desconecta del servidor.
Poscondiciones	No aplica

Tabla 4: Descripción del caso de uso, Chatear.

Caso de Uso	Chatear.
Actor	Usuario
Resumen	El caso de uso se inicia cuando el usuario selecciona en su lista de contactos conectados al que desea enviarle un mensaje y luego selecciona la opción chatear para empezar el diálogo.
Precondiciones	Seleccionar un contacto conectado.
Referencias	RF3
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El usuario selecciona el contacto deseado y escoge la opción chatear después de oprimir el comando “Menú”.	2. Muestra la interfaz que contiene la lista de contactos y las opciones del “Menú”.
3. El usuario escribe el texto deseado y oprime el comando “Enviar”.	4. Muestra la interfaz para establecer el diálogo.
Prototipo de interfaz	

Flujos alternos 4.a: Desea salir del diálogo.	
Acción del actor	**Respuesta del sistema**
4a.1 Presiona el comando “Atrás” y sale del diálogo.	1a.2 Muestra la interfaz con la lista de contacto y el estado actual.
Poscondiciones	No aplica

Tabla 5: Descripción del caso de uso, Añadir contacto.

Caso de Uso	Añadir contacto.
Actor	Usuario
Resumen	El caso de uso se inicia cuando el usuario selecciona la opción Añadir contacto en el “Menú” de la interfaz “Jabber”, para especificar el contacto que desea incluir en su lista.
Precondiciones	Que exista la cuenta en el servidor.
Referencias	RF4
Prioridad	Crítico
Flujo Normal de Eventos	

Acción del actor	Respuesta del sistema
1. El usuario oprime el comando "Menú" para seleccionar la opción "Añadir contacto".	2. Muestra la interfaz que contiene las opciones del "Menú".
3. El usuario escribe el usuario que desea añadir y oprime "OK" para enviar la solicitud.	4. Muestra la interfaz para especificar el usuario deseado y envía la solicitud.
	5. Muestra la interfaz con actualizada con el usuario añadido. En caso de que el usuario añadido no haya aceptado la solicitud, el sistema lo muestra conectado pero sin el recurso disponible (Pandion, Móvil, Gajim, etc.)

Prototipo de interfaz

Flujos alternos 4.a: Desea cancelar la acción.

Acción del actor	Respuesta del sistema

4a.1 Presiona el comando “Atrás” y va hacia la interfaz “Jabber”, para ejecutar otra acción.	1a.2 Muestra la interfaz con la lista de contacto y el estado actual.
Poscondiciones	No aplica

Tabla 6: Descripción del caso de uso, Eliminar contacto.

Caso de Uso	Eliminar contacto.
Actor	Usuario
Resumen	El caso de uso se inicia cuando el usuario selecciona la opción Eliminar contacto en el “Menú” de la interfaz “Jabber”, para especificar el contacto que desea eliminar de su lista.
Precondiciones	El contacto a eliminar debe de estar en la lista.
Referencias	RF5
Prioridad	Secundario
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El usuario selecciona el contacto que desea eliminar y en “Menú” selecciona la opción “Eliminar contacto”, para que se ejecute dicha acción.	2. Muestra la interfaz que contiene las opciones del “Menú”.
	3. Muestra la lista de contactos actualizada.
Prototipo de interfaz	

Flujos alternos 4.a: Desea cancelar la acción.	
Acción del actor	**Respuesta del sistema**
4a.1 Presiona el comando “Atrás” y va hacia la interfaz “Jabber”, para ejecutar otra acción.	1a.2 Muestra la interfaz con la lista de contacto y el estado actual.
Poscondiciones	No aplica

Conclusiones

En este capítulo se desarrolló la propuesta del sistema y se obtuvieron las funcionalidades que debe tener la aplicación, las cuales fueron representadas mediante un diagrama de casos de uso del sistema. Mediante el modelo de dominio se representaron las principales clases que estarán presentes en el sistema y finalmente fueron descritas todas las funcionalidades.

A partir de aquí se puede empezar a construir el sistema, cumpliendo con todos los requerimientos y las funcionalidades que se consideraron en el capítulo.

Capítulo 3. Análisis y diseño

Introducción

Este capítulo tiene como objetivo principal realizar el modelo de análisis y el modelo de diseño del sistema de mensajería instantánea para dispositivos móviles. Se definen las entidades y sus relaciones con los actores para efectuar los diagramas de clases del análisis y del diseño, además de los diagramas de interacción (dígase colaboración y secuencia). Este flujo de trabajo es de gran importancia en el ciclo de desarrollo, pues con el se traduce los requisitos definidos a funcionalidades que debe realizar el producto.

3.1 Modelo de análisis

Durante el análisis, se analizan los requisitos que fueron descritos en el capítulo anterior, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requerimientos y una descripción de los mismos que sea fácil de mantener y que ayude a estructurar todo el sistema, incluyendo su arquitectura. **(ISW 1, 2009)**

El objetivo fundamental del análisis es comprender los requisitos del software y no precisar cómo se implementará la solución. Este puede considerarse como una primera aproximación al modelo de diseño.

3.1.1 Diagramas de clases del análisis

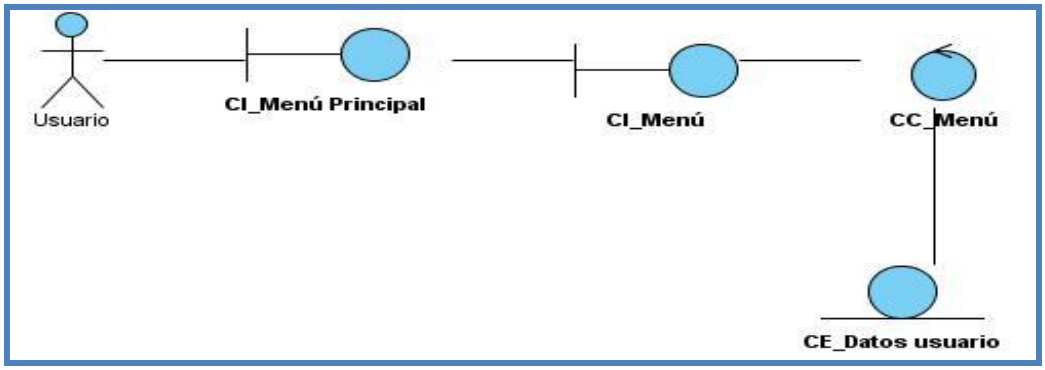

Ilustración 5: DCA_Autenticar.

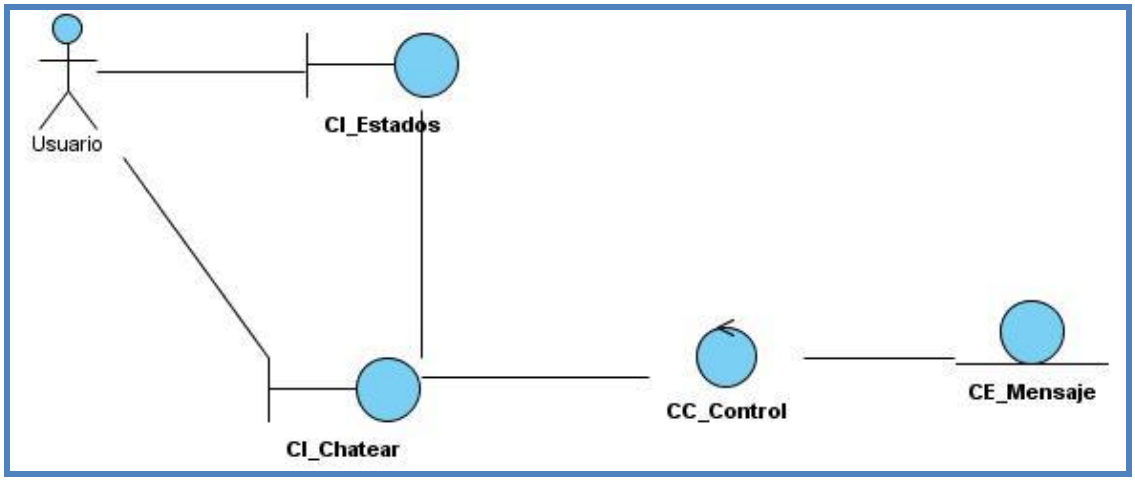

Ilustración 6: DCA_Chatear.

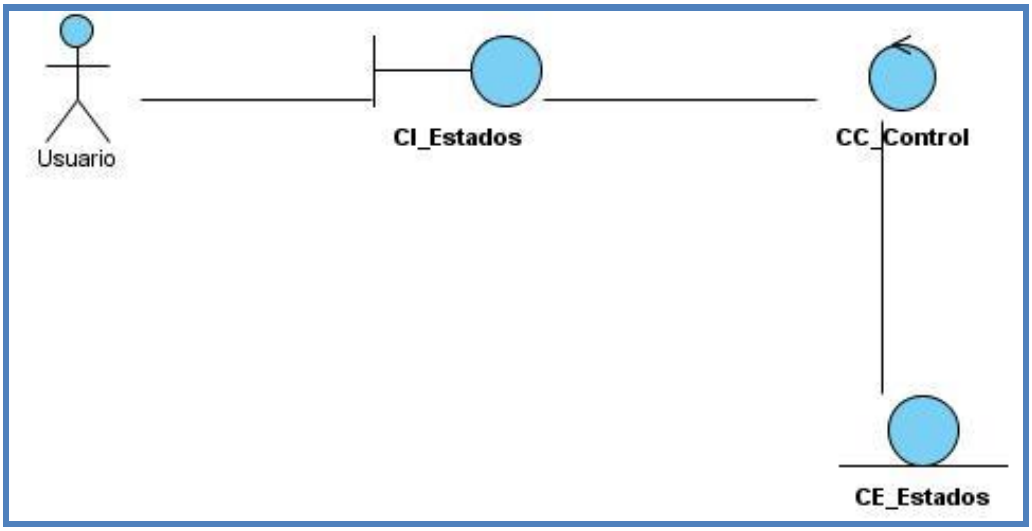

Ilustración 7: DCA_Modificar estado.

Ilustración 8: DCA_ Gestionar usuario (añadir).

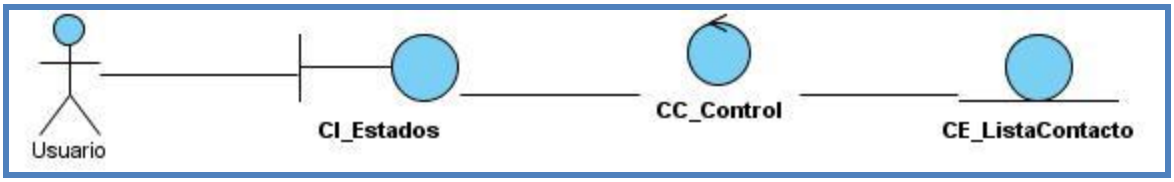

Ilustración 9: DCA_ Gestionar usuario (eliminar).

3.2 Modelo de diseño

El modelo de diseño es un modelo de objetos que describe la realización de los casos de uso, y sirve como una abstracción del modelo de implementación y el código fuente. **(ISW 1, 2009)**

3.2.1 Diagrama de clases del diseño

El diagrama de clases es una descripción de los modelos de objetos, contiene clases y las relaciones estructurales entre ellas. Este se obtiene como resultado del refinamiento del modelo conceptual y diagramas de secuencia. La definición de clase incluye definiciones para atributos y responsabilidades. **(ISW 1, 2009)**

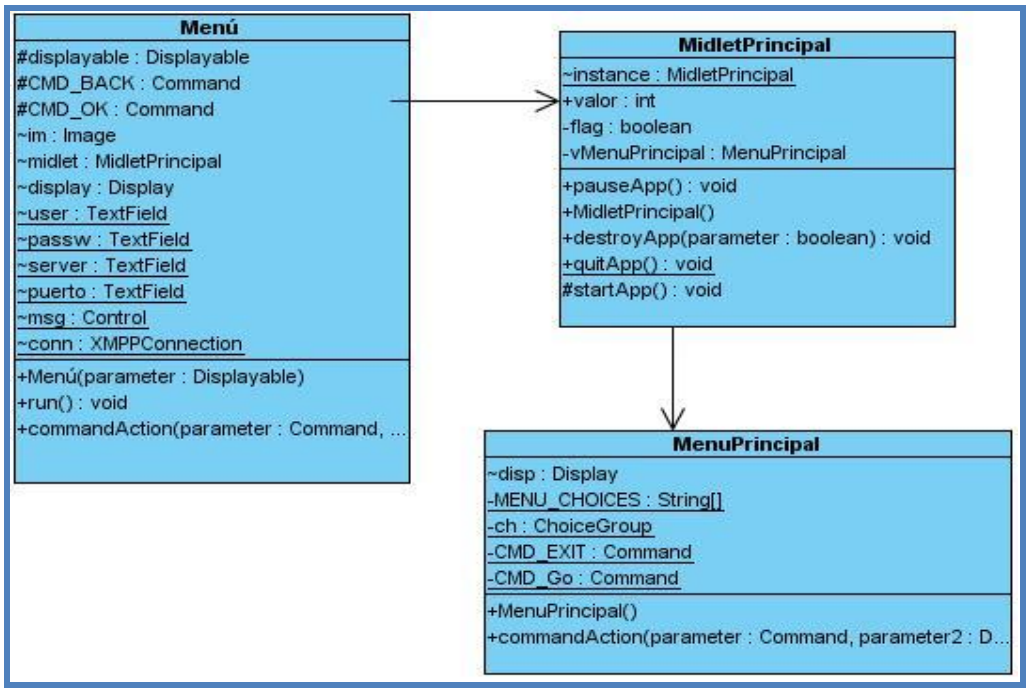

Ilustración 10: DCD_Autenticar.

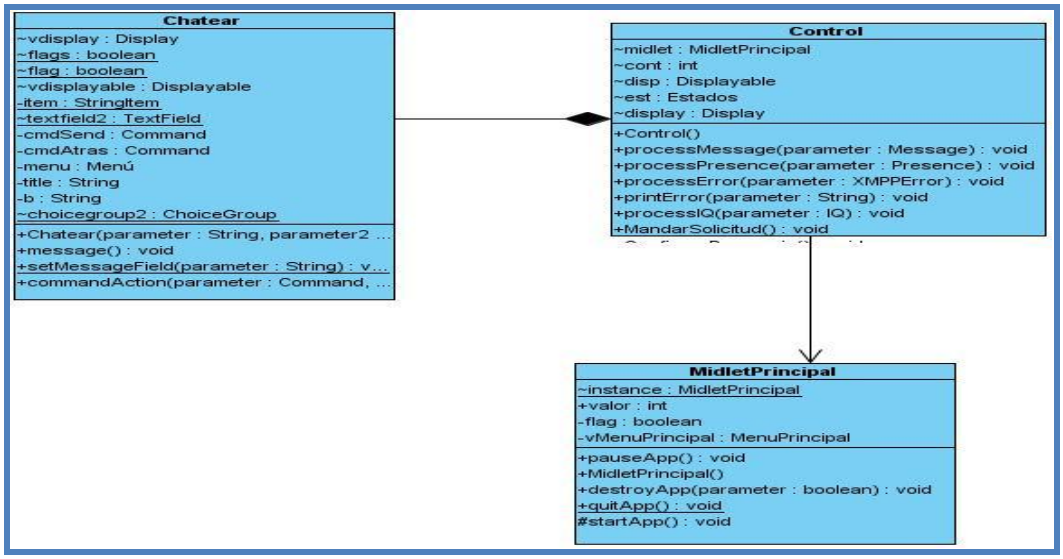

Ilustración 11: DCD_Chatear.

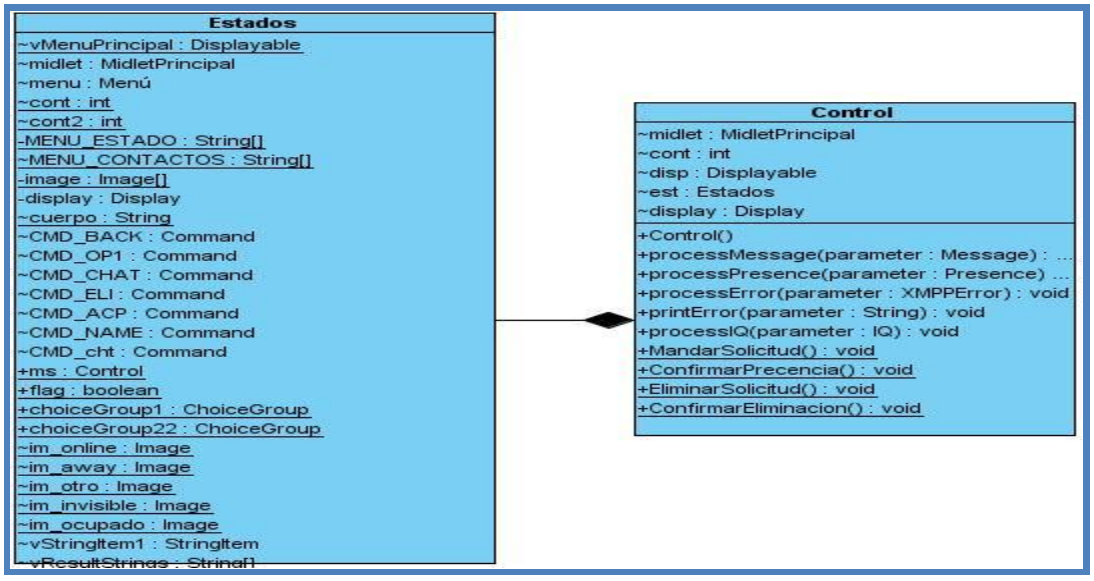

Ilustración 12: DCD_Modificar estado.

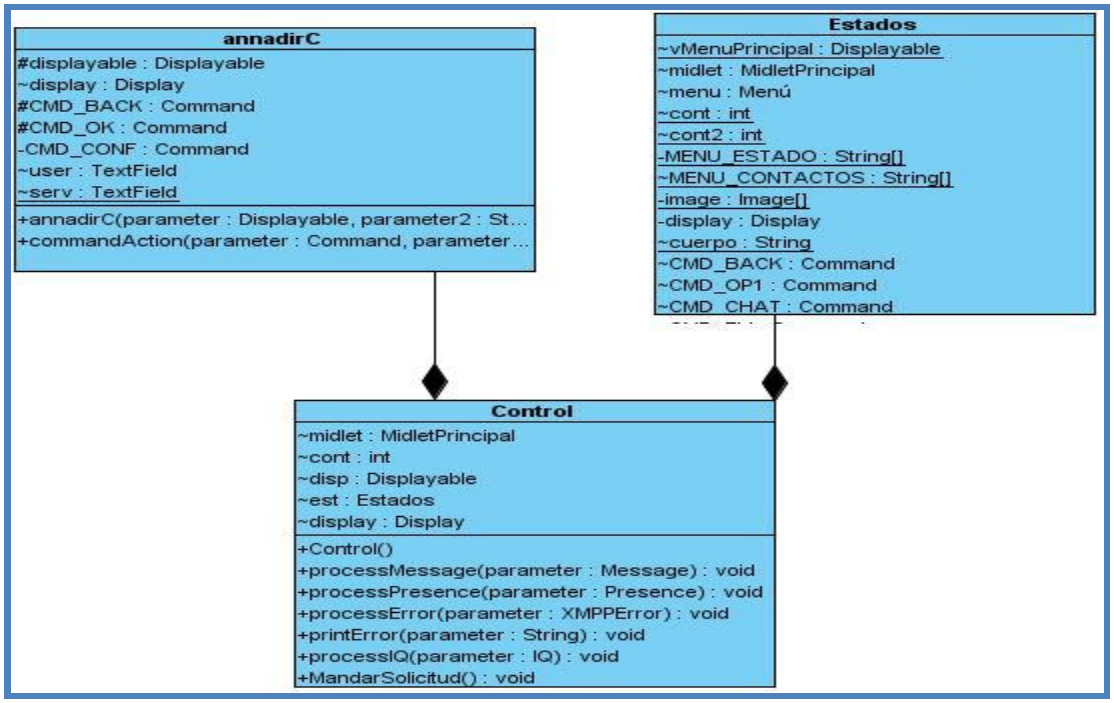

Ilustración 13: DCD_Gestionar usuario.

Para una mejor comprensión del sistema se muestra en un diagrama de clases del diseño general, que incluye a cada una de las clases utilizadas en el diseño. Ver Anexo 1.

3.2.2 Patrones de diseño

Los patrones de diseño son aquellos que expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que se construyen sistemas de software. Estos identifican: clases, instancias, roles, colaboraciones y la distribución de responsabilidades. **(ISW II, 2009)**

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. GRASP contiene una lista de cinco patrones básicos que se refieren a cuestiones y aspectos fundamentales del diseño:

- **Experto**

  Asigna responsabilidades al experto en información, o sea, las clases que tienen la información necesaria para cumplir con la responsabilidad.

- **Creador**

Guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. La creación de objetos es una de las actividades más frecuentes en un sistema orientado a objetos. En consecuencia, conviene contar con un principio general para asignar las responsabilidades concernientes a ella. El diseño, bien asignado, puede soportar un bajo acoplamiento, una mayor claridad, el encapsulamiento y la reutilización.

- **Bajo Acoplamiento**

El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras, mientras que una clase con alto (o fuerte) acoplamiento recurre a muchas otras.

- **Alta Cohesión**

Es asignar una responsabilidad de modo que la cohesión siga siendo alta. En la perspectiva del diseño orientado a objetos, la cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas y que no realicen un trabajo enorme.

- **Controlador**

Un Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema y define además el método de su operación. Un evento del sistema es un evento de alto nivel generado por un actor externo; es un evento de entrada externa. Se asocia a operaciones del sistema: las que emite en respuesta a los eventos del sistema.

3.3 Diagramas de interacción

Los diagramas de interacción describen la forma en que cada operación detectada en los diagramas de secuencia lleva a cabo sus responsabilidades y modifica el estado del sistema, mostrando el modo en que los objetos interactúan a través de mensajes. **(ISW II, 2009)**

En UML los diagramas de interacción pueden representarse a través de los diagramas de colaboración y/o de los diagramas de secuencia. Los diagramas de secuencia muestran interacciones entre objetos basadas en el tiempo.

Ver Anexo 2 Diagramas de secuencia.

Conclusiones

En este capítulo se desarrollaron los flujos de trabajo de Análisis y Diseño que propone la Metodología RUP y se modelaron los diagramas fundamentales de estos flujos. Se expusieron además los patrones de diseño utilizados para dar solución a problemas comunes.

Capítulo 4. Implementación

Introducción

En este capítulo se describe cómo los elementos del modelo del diseño se implementan en términos de componentes es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares, además de cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue.

4.1 Diagrama de componentes

Un componente es una parte modular de un sistema, desplegable y reemplazable que encapsula implementación y un conjunto de interfaces y proporciona la realización de los mismos. Un componente típicamente contiene clases y puede ser implementado por uno o más artefactos. **(EVA, 2010)**

Los diagramas de componentes son usadas para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. El uso más importante de estos diagramas es mostrar la estructura de alto nivel del modelo de implementación, especificando:

- Los subsistemas de implementación y sus dependencias a la hora de importar código.
- Organizar los subsistemas de implementación en capas. **(EVA, 2010)**

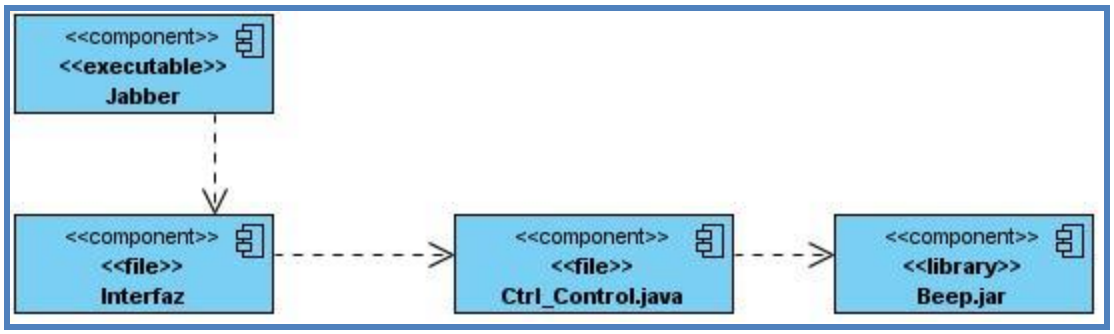

Ilustración 14: Diagrama de componentes.

4.1.1 Librería Beep v1.0.

La librería Beep es uno de los componentes usados para la implementación del sistema. Es una librería de código abierto que permite la comunicación con servidores XMPP para proporcionar funcionalidades de mensajería instantánea y presencia. Con este componente se pueden crear clientes completos de XMPP o simples integraciones con otras aplicaciones para brindar estos servicios.

Este es una versión del Smack para aplicaciones JME, que entre sus principales ventajas están:

- Existen otras versiones para J2SE.
- Es fácil de usar.
- Es probado y de código libre.

Para una mejor comprensión en cuanto a la estructura de las clases. Ver Anexo 3.

4.2 Diagrama de despliegue

El modelo de despliegue muestra la configuración de los nodos de procesamiento en tiempo de ejecución y los links de comunicación entre ellos. El propósito del despliegue es capturar la configuración de los elementos, y las conexiones entre estos elementos en el sistema.

El modelo consiste en unos o más nodos (elementos de procesamiento con al menos un procesador, memoria, y posiblemente otro dispositivo), dispositivos (nodos estereotipados con una capacidad de procesamiento en el nivel modelado de abstracción), y conectores, la conexión se establece entre nodos o entre dispositivos y nodos. **(EVA, 2010)**

Ilustración 15: Diagrama de despliegue.

Conclusiones

En este capítulo se desarrolló el flujo de trabajo de implementación que propone RUP, donde se modelaron los diagramas correspondientes, así como los propósitos de ambos. También se hace referencia a la librería que se usó para el completo funcionamiento de la aplicación y un resumen de sus principales ventajas.

Capítulo 5. Estimación y costo

Introducción

En este capítulo se realiza el desarrollo del método de estimación Puntos por Casos de Uso, para estimar el tiempo de duración del proyecto, el esfuerzo y el costo, para determinar si es viable continuar con el desarrollo del sistema.

5.1 Aplicar Método de Estimación Puntos por Casos de Uso

Se trata de un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores. **(EVA, 2008)**

Con esta técnica existe la posibilidad de predecir el tamaño de un sistema a partir de las características de sus requisitos, expresados en los casos de uso.

- **Cálculo de Puntos de Casos de Uso sin ajustar:**

Se calcula a partir de la siguiente ecuación: **UUCP = UAW + UUCW**

UUCP: Puntos de Casos de Uso sin ajustar.

UAW: Factor de Peso de los Actores sin ajustar.

UUCW: Factor de Peso de los Casos de Uso sin ajustar.

Tabla 7: Para calcular el Factor de Peso de los Actores sin ajustar (UAW).

Tipo de actor	Descripción	Factor de peso	Actores * peso
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API).	1	
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz	2	1 * 2

	basada en texto.		
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3	1 * 3
UAW=∑(actores*Peso)			**5**

Tabla 8: Para calcular el Factor de Peso de los Caso de Uso sin ajustar (UUCW).

Tipo de CU	Descripción	Peso	CU * Peso
Simple	El CU contiene de 1 a 3 transacciones.	5	2 * 5
Medio	El CU contiene de 4 a 7 transacciones.	10	2 * 10
Complejo	El CU contiene más de 8 transacciones.	15	
UUCW = ∑ CU*Peso			**30**

UUCP = 5 + 30 = 35 (Puntos de Casos de Uso sin ajustar**)**

- **Cálculo de Puntos de Casos de Uso ajustados:**

Se calcula a partir de la siguiente ecuación: **UCP = UUCP x TCF x EF**

UCP: Puntos de Casos de Uso ajustados.

UUCP: Puntos de Casos de Uso sin ajustar.

TCF: Factor de Complejidad Técnica.

EF: Factor de ambiente.

Nota: Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante.

Tabla 9: Para calcular Factor de Complejidad Técnica (TCF).

Factor	Descripción	Peso	Valor	Σ(Pesoi x Valor i)
T1	Sistema distribuido.	2	3	6
T2	Objetivos de performance o tiempo de respuesta.	1	5	5
T3	Eficiencia del usuario final.	1	4	4
T4	Procesamiento interno complejo.	1	4	4
T5	El código debe ser reutilizable.	1	4	4
T6	Facilidad de instalación.	0.5	5	2.5
T7	Facilidad de uso.	0.5	5	2.5
T8	Portabilidad.	2	4	8
T9	Facilidad de cambio.	1	2	2
T10	Concurrencia.	1	0	0
T11	Incluye objetivos especiales de seguridad.	1	4	4
T12	Provee acceso directo a terceras partes.	1	0	0
T13	Se requieren facilidades especiales de entrenamiento a usuarios.	1	3	3
Total				**45**

Para calcular TCF:

TCF = 0.6 + 0.01 x Σ (Pesoi x Valor asignadoi)

TCF = 0.6 + 0.01 * 45

TCF = 1.05

Nota: Para los factores E1 al E4, un valor asignado de 0 significa sin experiencia, 3 experiencia media y 5 amplia experiencia (experto).

- Para el factor E5, 0 significa sin motivación para el proyecto, 3 motivación media y 5 alta motivación.
- Para el factor E6, 0 significa requerimientos extremadamente inestables, 3 estabilidad media y 5 requerimientos estables sin posibilidad de cambios.
- Para el factor E7, 0 significa que no hay personal part-time (es decir todos son full-time), 3 significa mitad y mitad, y 5 significa que todo el personal es part-time (nadie es full-time).
- Para el factor E8, 0 significa que el lenguaje de programación es fácil de usar, 3 medio y 5 que el lenguaje es extremadamente difícil.

Tabla 10: Para calcular el Factor Ambiente (EF).

Factor	Descripcion	Peso	Valor	Σ(Pesoi x Valor i)
E1	Familiaridad con el modelo de proyecto utilizado	1.5	0	0
E2	Experiencia en la aplicación	0.5	0	0
E3	Experiencia en orientación a objetos	1	5	5
E4	Capacidad del analista líder	0.5	3	1.5
E5	Motivación	1	4	4
E6	Estabilidad de los requerimientos	2	4	8

E7	Personal part-time	-1	5	-5
E8	Dificultad del lenguaje de programación	-1	3	-3
Total				**10.5**

Para calcular EF:

EF = 1.4 - 0.03 x Σ (Pesoi x Valor asignadoi)

EF = 1.4 – 0.03 * 10.5

EF = 1.08 (Factor ambiente**)**

Luego UCP:

UCP = UUCP x TCF x EF

UCP = 35 * 1.05 * 1.08

UCP = 39.69 (Puntos de CU ajustados**)**

5.2 Cálculo del esfuerzo

E = UCP x CF

E: esfuerzo estimado en horas-hombre.

UCP: Puntos de Casos de Uso ajustados.

CF: factor de conversión

Para calcular CF:

CF = 20 horas – hombre (si total de EF ≤ 2**)**

CF = 28 horas – hombre (si total EF = 3 o total EF = 4**)**

CF = abandonar o cambiar el proyecto (si total de EF ≥ 5**)**

Total EF = Cant EF < 3 (entre E1 - E6) + Cant EF > 3 (entre E7, E8)

Total EF = 2 + 1 = 3

CF = 28 horas – hombre porque EF = 3

E = 39.69 * 28

E = 1111.32 (esfuerzo estimado en horas-hombre**)**

Flujo de trabajo de implementación.

Tabla 11: Estimación del esfuerzo del proyecto.

Actividad	% Esfuerzo	Valor esfuerzo
Modelamiento del Negocio	15	370.44
Análisis	15	370.44
Diseño	20	493.92
Implementación	45	1111.32
Sobrecarga	5	123.48
Total	**100**	**2469.6**

2469.6/ 8 = 308.7

308.7 / 48 = 6.431 ~ 6 meses y 4 días.

El proyecto requiere de 2469.6 horas-hombre para su desarrollo. Trabajando 8 horas diarias se obtienen aproximadamente 308.7 días para el desarrollo del proyecto, que, para un equipo de 2 personas trabajando 24 días al mes, equivale a aproximadamente 6 meses y 4 días.

5.3 Cálculo del costo

Se tiene la siguiente ecuación:

CHM = CH * SH

CHM = $ 200 / mes

Donde:

CHM: Costo de horas mes.

SH: Salario mensual por hombre = $ 100.

CH: Cantidad de hombres = 2.

Tiempo total del proyecto

ET = 2469.6 horas – hombre.

ET = 2469.6 / 192 = 12.86 mes – hombre.

TD = ET / CH

TD = 12.86 / 2

TD = 6.43 (6 meses y 4 días**)**

Costo total:

CT = ET * CHM / CH

CT = 12.86 * 200 / 2

CT = $ 1286

CT = $ 1286

Atendiendo a los resultados obtenidos se puede concluir que con 2 hombres trabajando en el proyecto se puede tener la propuesta en 6 meses y 4 días con un costo estimado de $ 1286.

Conclusiones

En este capítulo se desarrolló el estudio de la factibilidad correspondiente al sistema, así como la estimación, el esfuerzo de cada una de las etapas por las que pasa el software y el costo al ser implementado.

Conclusiones

El presente trabajo se culminó dando cumplimiento al objetivo general planteado, el cual consiste en la implementación de un cliente de mensajería instantánea, que permite el intercambio de mensajes entre dispositivos móviles de manera inmediata.

Con el desarrollo del trabajo se cumplieron las tareas de investigación satisfactoriamente, realizando un estudio del avance de la telefonía celular, así como de los protocolos usados para la mensajería instantánea y las principales características de clientes para este servicio.

Durante el desarrollo de los capítulos se hizo un seguimiento de las etapas por las que pasó el software, haciendo uso de la metodología *RUP*. Mediante el modelado usando *UML* y como herramienta case *Visual Paradigm* se generaron los diagramas correspondientes a cada flujo de trabajo y con el uso de Eclipse se desarrolló el sistema en lenguaje Java con la plataforma *JME*, herramientas y tecnologías que se seleccionaron en el capítulo 1 siendo óptimas para la aplicación.

Para un buen entendimiento del sistema se desarrolló el modelo de análisis y diseño, se determinó la arquitectura cliente – servidor y se utilizaron patrones de diseño para la solución de problemas comunes. También con el uso de tablas se arrojaron resultados de duración y costo del sistema, así como las descripciones generales de los requerimientos funcionales.

De manera general siguiendo la idea a defender planteada en la introducción del trabajo, se realizó el ciclo de vida del software hasta la fase de implementación, tributando a la construcción de un sistema capaz de enviar y recibir mensajes, como una nueva vía de comunicación para los usuarios portadores de teléfonos móviles en nuestro país.

Recomendaciones

Incluirle otras funcionalidades al sistema como:

- Renombrar a los contactos de la lista con el nombre que el usuario desee.
- Incluir emoticonos en el envío de los mensajes.
- Implementar un método para el cifrado de los mensajes.

Referencias bibliográficas

Díaz García, José Carlos. 2008. *Estudio del protocolo XMPP de mensajería instantánea.* Madrid, España : s.n., 2008.

Diego Blanco Moreno, Juan Pavón. 2009. J2ME Grasia. [En línea] Universidad Complutense de Madrid, 2009. http://grasia.fdi.ucm.es/j2me/_J2METech/CLDC.html.

Eclipse. 2008. Eclipse. [En línea] agosto de 2008. http://help.eclipse.org/ganymede/index.jsp.

EclipseME. 2009. Eclipse. [En línea] 2009. http://www.eclipse.org/documentation/ .

EVA. 2008. Clase Teórica-Práctica 2.Tecnicas de Estimacion ISW1 tema 2. Habana. *ISW 1 Tema 2.* Habana : s.n., 2008.

—. 2010. Conferencia 6: Fase de Construcción. FT Implementación. Modelo de implementación. *Tema 2 ISW2.* Habana : s.n., 2010.

Fashion, Moviles. 2009. Moviles Fashion. [En línea] 2009. http://www.movilesfashion.com/tag/msn-messenger-touch/.

Flanagan, David. 1999. *Java en Pocas Palabras.* s.l. : 2da Edición, 1999.

Huawei. 2008. Huawei. [En línea] 2008. http://www.huawei.com/publications/view.do?id=2865&cid=5264&pid=61.

II, Dpto. ISW. 2009. Conferencia 2: arquitectura y patrones de diseño. *Tema 2 ISW 2.* Habana : s.n., 2009.

II, ISW. 2009. Clase Práctica: Desarrollo del Modelo de Diseño. *Tema 1 ISW 2.* Habana : s.n., 2009.

ISW 1, Dpto. 2009. Fase de Elaboración. Flujo de trabajo de Análisis y Diseño. *Conferencia 7 ISW 1.* Habana : s.n., 2009.

ISW, Dpto. 2009. Tema 2: Fase de Inicio. Negocio y Requerimiento. *ISW 1.* Habana : s.n., 2009.

Ivar Jacobson, Grady Booch, James Rambaugh. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid, España : s.n., 2000.

Jabber. 2008. Jabber. [En línea] abril de 2008. http://www.jabber.org/.

JabberES. 2007. JabberES. [En línea] 2007. http://www.jabberes.org/dientes.

James Rambaugh, Ivar Jacobson, Grady Booch. 1998. *El Lenguaje Unificado de Modelado. Manual de referencia.* Madrid : s.n., 1998.

JavaME. 2010. Java. [En línea] febrero de 2010. http://java.sun.com/javame/index.jsp.

Lafuente, Máximo. 2010. Telefonía móvil en Cuba. *Juventud Rebelde.* http://www.juventudrebelde.cu/suplementos/informatica/2010-04-21/rebajaran-tarifas-para-llamadas-de-telefonia-movil-en-cuba/ , Habana, 21 de abril de 2010.

López, Ángel. 1997. *Java la programación del futuro .* Uruguay : s.n., 1997.

Saavedra, Carlos Andres Sanchez. 2009. Universidad ICESI. [En línea] agosto de 2009. http://www.icesi.edu.co/blogs_estudiantes/emicasanchez/2009/08/09/la-mensajeria-instantanea/.

Softonic. 2008. Softonic. [En línea] mayo de 2008. http://ebuddy-mobile-messenger.softonic.com/symbian.

—. 2009. Softonic. [En línea] mayo de 2009. http://palringo-bb.softonic.com/blackberry.

—. 2007. Softonic. [En línea] diciembre de 2007. http://im-all-in-one-mobile-messenger.softonic.com/symbian.

Sommerville, Iam. 2005. *Ingeniería del Software. Parte III Diseño 7ma Edición.* Madrid : s.n., 2005.

UML Tool. 2009. Visual Paradigm. [En línea] febrero de 2009. http://www.visual-paradigm.com/support/vpuml/releasenotes/642.jsp..

Zona Windows. 2009. Zona Windows. [En línea] 06 de diciembre de 2009. http://www.zonawindows.com.ar/nimbuzz-cliente-de-mensajeria-gratuito-para-telefonos-moviles/.

Bibliografía

Aplicaciones Empresariales con Jabber. 2010. Aplicaciones Empresariales. [En línea] 2010. http://www.aplicacionesempresariales.com/mensajeria-instantanea-en-las-empresas-con-jabber.html.

Consumer EROSKI. 2010. Consumer EROSKI. [En línea] 2010. http://www.consumer.es/web/es/tecnologia/internet/2009/09/03/187715.php.

Díaz García, José Carlos. 2008. *Estudio del protocolo XMPP de mensajería instantánea.* Madrid, España : s.n., 2008.

Diego Blanco Moreno, Juan Pavón. 2009. J2ME Grasia. [En línea] Universidad Complutense de Madrid, 2009. http://grasia.fdi.ucm.es/j2me/_J2METech/CLDC.html.

DOSBIT. 2008. DOSBIT. [En línea] agosto de 2008. http://www.dosbit.com/moviles/ebuddy-gran-cliente-de-mensajera-instantnea-para-dispositivos-mviles.

Eclipse. 2008. Eclipse. [En línea] agosto de 2008. http://help.eclipse.org/ganymede/index.jsp.

EclipseME. 2009. Eclipse. [En línea] 2009. http://www.eclipse.org/documentation/ .

EVA. 2008. Clase Teórica-Práctica 2.Tecnicas de Estimacion ISW1 tema 2. Habana. *ISW 1 Tema 2.* Habana : s.n., 2008.

—. 2010. Conferencia 6: Fase de Construcción. FT Implementación. Modelo de implementación. *Tema 2 ISW2.* Habana : s.n., 2010.

—. 2008. EVA. [En línea] 2008. http://eva.uci.cu/file.php/452/Bibliografia_Basica/documentacion_Java.rar.

Fashion, Moviles. 2009. Moviles Fashion. [En línea] 2009. http://www.movilesfashion.com/tag/msn-messenger-touch/.

Flanagan, David. 1999. *Java en Pocas Palabras.* s.l. : 2da Edición, 1999.

Huawei. 2008. Huawei. [En línea] 2008. http://www.huawei.com/publications/view.do?id=2865&cid=5264&pid=61.

IBM. 2007. IBM. [En línea] 2007. http://www.ibm.com/developerworks/opensource/library/os-eclipse.html.

II, Dpto. ISW. 2009. Conferencia 2: arquitectura y patrones de diseño. *Tema 2 ISW 2.* Habana : s.n., 2009.

II, ISW. 2009. Clase Práctica: Desarrollo del Modelo de Diseño. *Tema 1 ISW 2.* Habana : s.n., 2009.

ISW 1, Dpto. 2009. Fase de Elaboración. Flujo de trabajo de Análisis y Diseño. *Conferencia 7 ISW 1.* Habana : s.n., 2009.

ISW, Dpto. 2009. Tema 2: Fase de Inicio. Negocio y Requerimiento. *ISW 1.* Habana : s.n., 2009.

Ivar Jacobson, Grady Booch, James Rambaugh. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid, España : s.n., 2000.

Jabber. 2008. Jabber. [En línea] abril de 2008. http://www.jabber.org/.

JabberES. 2007. JabberES. [En línea] 2007. http://www.jabberes.org/clientes.

James Rambaugh, Ivar Jacobson, Grady Booch. 1998. *El Lenguaje Unificado de Modelado. Manual de referencia.* Madrid : s.n., 1998.

JavaME. 2010. Java. [En línea] febrero de 2010. http://java.sun.com/javame/index.jsp.

Jon Watson. 2009. Iniciacion al servidor groupware Citadel. [En línea] 2009. http://www.linux-magazine.es/issue/24/035-038_CitadelLM24.crop.pdf.

Lafuente, Máximo. 2010. Telefonía móvil en Cuba. *Juventud Rebelde.* http://www.juventudrebelde.cu/suplementos/informatica/2010-04-21/rebajaran-tarifas-para-llamadas-de-telefonia-movil-en-cuba/ , Habana, 21 de abril de 2010.

López, Ángel. 1997. *Java la programación del futuro .* Uruguay : s.n., 1997.

Marketing Charts. 2008. Marketing Charts. [En línea] 9 de julio de 2008. http://www.marketingcharts.com/direct/mobile-photo-messaging-up-60-in-us-among-top-mobile-data-apps-5205/..

OnSoftware. 2008. OnSoftware. [En línea] 2008. http://es.onsoftware.com/p/programas-mensajeria-movil.

Open Source. 2009. Jabber/XMPP daemon. [En línea] 2009. http://www.ejabberd.im.

Polinux. 2009. Polinux. [En línea] 2009. http://www.polinux.upv.es/drupal/node/170.

Saavedra, Carlos Andres Sanchez. 2009. Universidad ICESI. [En línea] agosto de 2009. http://www.icesi.edu.co/blogs_estudiantes/emicasanchez/2009/08/09/la-mensajeria-instantanea/.

Softonic. 2008. Softonic. [En línea] mayo de 2008. http://ebuddy-mobile-messenger.softonic.com/symbian.

—. 2009. Softonic. [En línea] mayo de 2009. http://palringo-bb.softonic.com/blackberry.

—. 2007. Softonic. [En línea] diciembre de 2007. http://im-all-in-one-mobile-messenger.softonic.com/symbian.

Sommerville, Iam. 2005. *Ingeniería del Software. Parte III Diseño 7ma Edición.* Madrid : s.n., 2005.

UML Tool. 2009. Visual Paradigm. [En línea] febrero de 2009. http://www.visual-paradigm.com/support/vpuml/releasenotes/642.jsp..

UNIX. 2009. UNIX. [En línea] 2009. http://unix.freshmeat.net/projects/vp-umlce..

VeriSign. 2008. VeriSign. [En línea] 2008. http://www.verisign.com/static/044155.pdf.

Zona Windows. 2009. Zona Windows. [En línea] 06 de diciembre de 2009. http://www.zonawindows.com.ar/nimbuzz-cliente-de-mensajeria-gratuito-para-telefonos-moviles/.

Glosario de términos

AOL: America Online.

AIM: America On Line Instant Messenger.

BOS: Basic OSCAR Service.

Bots (programas informáticos que realizan diversas funciones, imitando el comportamiento humano).

CLDC: Connected Limited Device Configuration.

Cliente-servidor: arquitectura en la que confluyen una serie de aplicaciones basadas en dos categorías que cumplen funciones diferentes. El cliente envía un mensaje solicitando un determinado servicio a un servidor y este envía uno o varios mensajes con la respuesta.

Cliente de mensajería instantánea: es una aplicación que permite al usuario hacer uso de la mensajería instantánea.

DCA: Diagrama de clases del análisis.

DCD: Diagrama de clases del diseño.

DS: Diagrama de secuencia.

DSS: Digital Signature Standard.

GRASP (General Responsibility Assignment Software Patterns): Patrones generales de software para la asignación de responsabilidades.

ICQ (I seek you): Es un cliente de mensajería instantánea y el primero de su tipo en ser ampliamente utilizado en Internet.

IRC: Internet Relay Chat.

IDE: Integrated Development Environment.

JME: Java Micro Edition.

MIDP: Mobile Information Device Profile.

MMS (Multimedia Messaging System): sistema de mensajería multimedia.

MI: Mensajería instantánea (conocida también en inglés como IM): Es una forma de comunicación en tiempo real entre dos o más personas basada en texto.

MSN (MicroSoft Network): servicios de internet ofrecidos por Microsoft.

OSCAR (Open System for Communication in Realtime): Sistema abierto para la comunicación en tiempo real.

P2P: Peer to Peer.

PLATO: Programmed Logic Automated Teaching Operations.

RSA (Rivest, Shamir, Adlman): Sistema criptográfico de clave pública.

RUP (Rational Unified Process): Proceso Unificado de Rational.

SMS (Short Message Service): sistema de mensajes cortos.

SIP (Session Initiation Protocol): Protocolo de inicio de sesiones.

SSL (Secure Sockets Layer): Protocolo de capa de conexión segura.

Sistema en tiempo real (STR): es aquel sistema digital que interactúa activamente con un entorno con dinámica conocida en relación con sus entradas, salidas y restricciones temporales.

TLS (Transport Layer Security): Seguridad de la capa de transporte.

UIQ (User Interface Quartz): Es una plataforma para terminales móviles desarrollada por UIQ Technology basada en el sistema operativo Symbian. Se caracteriza por agregar soporte para pantallas táctiles.

UML (Unified Modeling Language): Lenguaje Unificado de Modelado.

VoIP (Voice Over Internet Protocol): Permiten la conmutación de voz vía Internet.

VP- UML: Visual Paradigm for Unified Modeling Language.

XMPP (Extensible Messaging and Presence Protocol): estándar de mensajería instantánea y comunicación de presencia.

XML: Extensible Markup Language.