Universidad de las Ciencias Informáticas
Facultad 2

Título: "SDSCenter asociado al proyecto SERWAP"

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor(es): Mailyn Cortada Corrales
Yamilka Ricardo Rodríguez

Tutor(es): Ing. Joan Martínez Herrera.
Ing. Yeilin Pérez Martínez.

Ciudad de la Habana, Junio 2011

PENSAMIENTO

"Si me ofreciesen la sabiduría con la condición de guardarla para mí sin comunicarla a nadie, no la querría."

Lucio Anneo Séneca

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos al Centro de Telemática de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Autores:

Mailyn Cortada Corrales
_____
Firma

Yamilka Ricardo Rodríguez
_____
Firma

Tutores:

Ing. Joan Martínez Herrera.
_____
Firma

Ing. Yeilin Pérez Martínez
_____
Firma

DATOS DE CONTACTO

Tutor: Ing. Joan Martínez Herrera (jherrera@uci.cu).
Profesión: Ingeniero en Ciencias Informáticas.
Años de graduado: 3.
Labor que desempeña: Analista Principal del Proyecto Servidor de Aplicaciones WAP (SERWAP) del Centro de Telemática, Facultad 2 de la UCI.

Tutor: Ing. Yeilin Pérez Martínez (ypmartinez@uci.cu).
Profesión: Ingeniero en Ciencias Informáticas.
Años de graduado: 3.
Labor que desempeña: Líder del Proyecto Servidor de Aplicaciones WAP (SERWAP) del Centro de Telemática, Facultad 2 de la UCI.

AGRADECIMIENTOS

Agradezco a mi mamá Marilyn por su amor y dedicación, a mi papá Pepito por ser tan comprensivo conmigo para las cosas buenas y malas, por quererme y preocuparse tanto y por ser un excelente padre. Muchas gracias a ambos por ser lo más grande que yo tengo, por darme todo el apoyo del mundo, por estar conmigo en cada momento ayudándome, educándome y aconsejándome, realmente les debo todo lo que soy.

A mi hermana Ivette y mi cuñado Alain por tenerme siempre presente y por confiar en mí, y a mis sobrinitos Allison y Alain que aunque son pequeños aún, sé que me quieren mucho como yo a ellos.

Al resto de mi familia, mis abuelos, tíos, primos, en fin…todos se preocuparon y me apoyaron hasta el final, muchas gracias de corazón.

A mis amistades…que son muchas…las de la UCI que vivieron conmigo aquí al diario, principalmente las chicas de mi apartamento, que pasamos muchísimas cosas juntas, buenas y malas. Y las amistades que no son de la UCI, que son desde el pre-universitario y hasta de la secundaria, que siempre me apoyaron y confiaron en mí. Les agradezco a todos por considerarme una amiga y tenerme siempre en cuenta.

A mi compañera de tesis Yamilka y a mis tutores Joan y Yeilin, que dieron lo mejor de sí en cada momento para que este trabajo de diploma saliera con los mejores resultados.

En fin a todas aquellas personas que se preocuparon por mí….

A todos ustedes Gracias

Mailyn

Agradezco todo lo que soy primeramente a mis padres, por haberme tocado como padres, no los quiero mejor, ellos han sabido vivir para sus hijas, y lo han dado todo por nosotras, a mi padre por ser incansable, emprendedor y no encontrar metas imposibles y sobre todo por amarnos sin fin, y saber trasmitirnos eso, a mi madre por la educación que me ha dado, y los valores que me ha sabido trasmitir, por ser paciente, resistente y devota a su familia, por amarnos sin fin. A mis abuelos, que han sido mis segundos padres, dándome su apoyo, confianza y amor cada vez que lo necesito, a mi hermana, que ha sido mi modelo a imitar desde que la conocí, siempre ahí, mi brazo derecho, apoyándome incondicionalmente. A mi sobrino, que fue mi ensayo materno, por adorar a su tía y su tía a él.

A mi novio por enseñarme a aprender, por ser mi guía profesional, mi maestro. Por apoyarme en todo y darme ánimo cuando las cosas no van bien, por amarme sin fin.

A mis tíos, tías, primas y primos, que han sido parte de mi vida y mi formación.

A mis amigos de Holguín, en especial a Yailin y Yadira, siempre las recuerdo.

A Dorita que ha sabido ser una amiga y que siempre lo será.

A mis amigas y amigos de la UCI, a Odaymis, en las buenas y en las malas siempre ahí.

A mis profesores a lo largo de toda mi vida de estudios.

A mis tutores por estar siempre ahí, y todos los demás que estuvieron dispuestos cuando los necesité.

A mi compañera de tesis, flexible y siempre de buen humor.

A todos Muchas Gracias

Yamilka

DEDICATORIA

Dedico este trabajo de diploma a mis padres que son lo más importante que tengo, a mis abuelos paternos ya fallecidos que estoy segura de que se hubieran sentido orgullosos de mí y a mis abuelos maternos que siempre los tengo presentes y los quiero con la vida.
Mailyn

Dedico este trabajo de diploma a mis padres, esto es más suyo que mío, a mis abuelos que lo merecen de sobra, a mi hermana por ser mi guía y a mi sobrino para que este orgulloso y siga mis pasos.
Yamilka

RESUMEN

En la Universidad de las Ciencias Informáticas se han desarrollado varios proyectos para la solución de problemas dentro del campo de la Informática y las Telecomunicaciones tanto nacionales como internacionales. Uno de estos proyectos es Servidor de Aplicaciones WAP (SERWAP), el cual consiste en un sistema de aplicaciones que automatizan los procesos que llevan a cabo diferentes agentes de radio profesional. El envío de mensajes WAP PUSH hacia la infraestructura NEBULA es una característica novedosa del sistema, sin embargo pueden ocurrir errores durante dicha operación y el mensaje puede no llegar a su destino ya que no existe un mecanismo que gestione dicho proceso. Durante el envío de un mensaje pueden ocurrir eventos no deseados por lo que la información referente a dicho comportamiento es de gran importancia para los administradores del sistema, pero SERWAP no contiene aún un sistema que registre y muestre esta información.

Para dar solución al mencionado problema, se desarrolló el sistema SDSCenter asociado a dicho proyecto, el cual es capaz de gestionar satisfactoriamente el envío de los mensajes WAP PUSH desde el Portal Web de SERWAP y permite visualizar la información referente al envío de los mismos a través del portal.

PALABRAS CLAVE

SDSCenter, Mensajes WAP PUSH, TETRA

ÍNDICE

INTRODUCCIÓN

La especie humana por ser de carácter social tiene la necesidad de comunicarse. Desde sus inicios unos de los primeros recursos utilizados fueron la voz, las señales de humo y los dibujos pictóricos. Así fue evolucionando la humanidad y las técnicas de comunicación se fueron perfeccionando, de esta forma en el siglo XIX aparece el telégrafo eléctrico, que permitió el enviar mensajes que contenían solo letras y números. Más tarde apareció el teléfono, con el que fue posible comunicarse utilizando la voz. [1][2]

Como consecuencia de estos y otros avances y el transcurso del tiempo surge el término *telecomunicación,* que fue definido por primera vez en la reunión conjunta de la XIII Conferencia de la UTI (Unión Telegráfica Internacional) y la III de la URI (Unión Radiotelegráfica Internacional) que se inició en Madrid el día 3 de septiembre de 1932. La definición entonces aprobada del término fue: "*Telecomunicación es toda transmisión, emisión o recepción, de signos, señales, escritos, imágenes, sonidos o informaciones de cualquier naturaleza por hilo, radioelectricidad, medios ópticos u otros sistemas electromagnéticos*". [1][2]

Actualmente existen varias tecnologías enmarcadas dentro de este concepto, todas destinadas a resolver problemas diferentes y específicos de cada caso. Entre estas se encuentra TETRA (del inglés: TErrestial Trunked Radio) un estándar definido por el ETSI (del inglés: European Telecomunication Standars Institute) teniendo entre sus objetivos, unificar diversas alternativas de interfaces de radio digitales para la comunicación entre los sectores profesionales como policías, bomberos, guardias forestales, ambulancias, etc. [3]

Otro estándar destacado en este ámbito es GSM (del francés: Groupe Spécial Mobile) el mismo es el sistema global para las comunicaciones móviles, que desde sus inicios se ha ido extendiendo y mejorando, permitiendo la comunicación mediante teléfonos móviles que incorporan tecnología digital. Unos de los principales servicios que provee este estándar es el envío de mensajes de texto o SMS (del inglés: Short Message Service). Para gestionar este servicio de mensajes existen los SMSC (del inglés: Short Message Service Center) dentro de estas infraestructuras de red, su función es la de recibir, responder, enviar y reenviar SMS, o sea, actúa como un “puente” entre los subscriptores que se encarga de realizar estas funciones. [4]

En las redes TETRA mencionadas anteriormente, se cuenta con el SDS (del inglés: Short Data Service), este es un servicio que permite el envío de datos cortos dentro de estas infraestructuras de red. Algunos sistemas de comunicación desarrollados para redes TETRA tienen la necesidad de gestionar el envío de

estos mensajes, en este caso, las acciones de gestión de envío las realiza un SDS Center, o sea que, permiten enviar, reenviar y responder mensajes. Si el suscriptor no está inmediatamente disponible, el mensaje es guardado en el SDS Center hasta que el terminal de destino esté disponible si no se ha vencido su tiempo de espera predefinido. Si el mensaje no llega por alguna razón entonces se envía una notificación al emisor informando de las posibles fallas. [5]

Existen grandes empresas a nivel mundial que se dedican a la producción de componentes y unidades para redes TETRA, una de estas es TELTRONIC la cual diseña y fabrica equipos y sistemas de radiocomunicación para uso profesional bajo este estándar de red.
Uno de los principales productos de TELTRONIC es la infraestructura NEBULA que proporciona servicios de comunicaciones digitales, para dicha infraestructura se desarrolló el proyecto SERWAP en la UCI (Universidad de las Ciencias Informáticas) por el TLM (Centro de Telemática) de la universidad, en conjunto con la empresa cubana ALBET. SERWAP consiste en un servidor de aplicaciones WEB y WAP, esta última funciona bajo el protocolo WAP (del inglés: Wireless Application Protocol). Estas aplicaciones automatizan los procesos que llevan a cabo diferentes agentes de radio profesional mencionados anteriormente. SERWAP cuenta con un Portal Web para gestionar la información referente a dichos agentes. Uno de los principales servicios que brinda el Portal Web es el envío de mensajes WAP PUSH hacia la infraestructura. Para garantizar el envío de mensajes desde el Portal Web se utiliza el API (del inglés: Application Programming Interface) N2AClient. Este API provee un conjunto de funcionalidades que permiten enviar los mensajes hacia la infraestructura y luego recoger la respuesta proveniente de NEBULA, indicando lo ocurrido durante el envío del mensaje.

Actualmente en el Portal Web de SERWAP no existe un mecanismo para gestionar el envío de los mensajes WAP PUSH. Los mensajes durante este proceso de envío – recibo se pueden perder o no llegar a su destino, ya que no se realiza ninguna acción como almacenamiento o reenvío de los mismos en caso de que esto suceda. Tampoco se brinda ninguna información al usuario acerca de los datos del envío, como la confirmación de recibo de los mensajes, hora de llegada y salida, etc.

A partir de la problemática planteada se deriva el siguiente **problema científico** ¿Cómo gestionar el envío de los mensajes WAP PUSH desde el Portal Web de SERWAP hacia la infraestructura NEBULA?

A partir del problema científico se puede definir como **objeto de estudio** a los sistemas de gestión de envío de mensajes SMS y SDS, delimitando el **campo de acción** al sistema de envío de mensajes WAP PUSH utilizado en SERWAP mediante el API N2AClient.

El **objetivo general** de la presente investigación es desarrollar un sistema que gestione el envío de los mensajes WAP PUSH a la infraestructura NEBULA y brinde información útil sobre el envío y estado de estos al usuario. Se derivan de este los siguientes **objetivos específicos**:

- Enviar mensajes hacia la infraestructura utilizando el API N2AClient.
- Mantener una cola de mensajes para poder reenviarlos si no pueden ser entregados inmediatamente por problemas con el terminal o la infraestructura.
- Consultar mediante una interfaz web la información referente a los mensajes enviados.
- Decodificar los mensajes para poder mostrar en la interfaz web los campos con los que fueron creados los mismos.
- Guardar en una base de datos el contenido y datos de envío de los mensajes para ser mostrados en una interfaz web.
- Permitir el intercambio de datos entre los servicios de Web Services o de RMI (del inglés: Remote Method Invocation) y el sistema de gestión de envío de mensajes mediante una conexión socket TCP (del inglés: Transfer Communication Protocol).

Para guiar el cumplimiento de los objetivos propuestos se definen como **tareas de la investigación**:

- Investigar el funcionamiento de los sistemas de envío de mensajes y sus funcionalidades.
- Investigar y determinar las acciones que debe realizar el sistema de gestión de envío de mensajes WAP PUSH.
- Investigar el funcionamiento y funcionalidades del API N2AClient para el envío de mensajes.
- Realizar un estudio de las aplicaciones web y el acceso a datos en el Framework Spring 2.5.
- Realizar un estudio acerca del funcionamiento del API TETRA PUSH encargada de codificar los mensajes.
- Realizar un análisis acerca de la información que será almacenada y como estará estructurado el modelo de datos.
- Investigar el funcionamiento de las conexiones socket en Java.

Se ha determinado establecer las siguientes **preguntas científicas**:

- ¿Existe algún SDS Center de código libre que pueda ser utilizado para la solución?
- ¿Qué funciones, similares a las de un SDS Center pueden ser implementadas dentro del sistema SERWAP para gestionar el envío de mensajes WAP PUSH?
- ¿Cómo utilizar las funcionalidades del API N2AClient para gestionar el envío de los mensajes?
- ¿Cómo extraer los datos de envío de los mensajes y registrarlos en una base de datos?
- ¿Cómo decodificar el contenido de los mensajes?
- ¿Qué tipo de socket es más factible utilizar para establecer conexión entre el Portal Web y el sistema de gestión de envío de mensajes?

Métodos de la Investigación

Durante la presente investigación se utilizaron **métodos teóricos** pudiendo mencionar el **analítico-sintético** que se utilizó para analizar la bibliografía y así identificar ideas claras que ayudaran a lograr una mejor comprensión del problema a partir de la extracción de los elementos más importantes. Así como el método **inductivo-deductivo** el cual se utilizó para lograr entender los casos particulares a partir de un estudio general de las diferentes temáticas. Otro método utilizado fue el **análisis histórico-lógico** para aclarar conceptos e incertidumbres a partir de la trayectoria histórica de estos.

También se utilizaron **métodos empíricos** como la **observación,** el cual fue muy útil a la hora de adquirir experiencia y conocimiento al guiarse por el equipo de trabajo.

El contenido está estructurado de la siguiente forma:

Capítulo I Fundamentación Teórica: Contiene un análisis a partir de la investigación realizada sobre las tecnologías inmersas en la solución. Además se fundamentan las metodologías, lenguajes, conceptos y herramientas utilizadas.

Capítulo II Características del Sistema: Contiene una descripción de los principales conceptos reflejados en un Modelo de Dominio, se recogen los requisitos funcionales y no funcionales, así como el diagrama de casos de usos del sistema y la descripción de los mismos.

Capítulo III Diseño del Sistema: Contiene los diagramas del diseño. Incluye una descripción de la arquitectura y patrones de diseño empleados así como el modelo de base de datos.

Capítulo IV Implementación y Prueba: Contiene las principales características de la implementación, representando el diagrama de despliegue y el diagrama de componentes, así como la realización y descripción de casos de prueba seleccionados.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se describen las tecnologías y conceptos vinculados al tema tratado en el presente trabajo, dígase sistemas de gestión de envío de mensajes tales como los SMS Center y SDS Center, redes TETRA, la infraestructura NEBULA y el API N2AClient. Además se fundamentan las principales herramientas, metodología y tecnologías que serán utilizadas.

1.2 SMS Center

Un SMS Center es un sistema cuya función principal es la de enviar y recibir mensajes SMS en las redes de telefonía móvil. Ofrece una amplia gama de servicios dentro de una misma plataforma, como puede ser la utilización de lista de contactos, que le permite identificar correctamente el usuario de origen de los mensajes como también mantener una trazabilidad de la conversación realizada con dicho usuario. [4]
Es posible utilizarlo como una herramienta de gestión para recibir, responder y enviar mensajes SMS como también para hacer la derivación de los mensajes recibidos y enviados a una plataforma existente en el cliente que puede obtener información desde un sistema central. [4]
Cuando un usuario envía un mensaje de texto a otro usuario, si el destinatario se encuentra en cobertura el mensaje será entregado inmediatamente, de lo contrario se guarda el mensaje en el SMS Center hasta que el destino esté disponible siempre que no exceda el tiempo máximo de espera, pues si en ese tiempo el destinatario no es localizado, el mensaje será desechado. [4]

Algunas de las características que pueden presentar los SMS Center son: [4]

- ✓ Emplear un teléfono móvil o un módem GSM conectado a un computador para enviar y recibir mensajes.
- ✓ Contener un servidor de correo integrado que enviará mensajes de correo electrónico en forma de SMS a uno o varios destinatarios.
- ✓ Ser utilizado como una herramienta de gestión para recibir, responder y enviar mensajes SMS, como también para hacer la derivación de los mensajes recibidos y enviados a una plataforma existente en el cliente.
- ✓ Permitir personalizar distintas acciones dependiendo del contenido del mensaje que envía el usuario, definiendo así, los posibles caminos a tomar.

- ✓ Permitir consultar en cualquier momento y horario los mensajes recibidos y enviados a través de un explorador de Internet.

Existen en la actualidad varias empresas que se dedican a desarrollar productos de este tipo, una de estas es San Diego SoftWorks, con sede en Uruguay, la cual cuenta ya con 10 años de trayectoria, especializada en el desarrollo de sistemas para plataformas móviles. Uno de los principales productos de esta empresa es SMS Center, que no es más que una solución integral de mensajería SMS bidireccional, la cual está basada en un modelo económicamente viable y de rápida implementación, que incorpora una completa funcionalidad, cubriendo todos los aspectos de una plataforma de mensajería. Mantiene el flujo de mensajes recibidos y enviados pudiendo realizar un seguimiento de la comunicación. Su herramienta de administración de contactos permite mantener la información de sus clientes de forma segura y centralizada pudiendo conocer los datos de éstos al momento de recibir y enviarles información. [4]

Su interfaz web brinda un acceso rápido al listado de mensajes recibidos para poder contestarlos automáticamente e ingresar comentarios sobre cada uno de ellos. [4]

Al utilizar la función de “Puente”, SMS Center se convierte en una plataforma de mensajería que realiza la recepción y envío de mensajes entre el usuario y la empresa, pudiendo integrar sistemas comerciales actuales de las empresas sin necesidad de costos extras. A su vez, con esta funcionalidad, es posible que la empresa determine los mensajes automáticos de respuesta que se desea dar a un usuario dependiendo del texto contenido en el SMS recibido. Por ejemplo consultar en una base de datos el saldo de una cuenta o solicitar hora para un especialista. [4]

Otras de las principales empresas que desarrollan sistemas para plataformas móviles son Motorola, Nokia y ATS (del inglés: Advance Technology Solutions). Estas empresas incluyen funcionalidades de los SMS Centers dentro de sus productos, o sea que, combinan estos sistemas con otros servicios que incluyen por ejemplo la gestión de multimedia y otros. [6][7][8]

1.3 SDS Center

Un SDS Center realiza acciones similares a un SMS Center con la diferencia de que en vez de trabajar con SMS lo hace con SDS, correspondiente a las redes TETRA. Realiza todo el proceso relacionado a la

gestión de envío de los mensajes SDS, tales como enviar, reenviar, recepcionar los mensajes para enviarlos en el momento definido, ya sea porque el terminal no esté disponible o por otras causas, todo depende de las funciones que debe realizar la central y para lo que fue creada, ya que todos estos sistemas son privativos; por lo que no hay recursos libres de este tipo a la vez que existe muy poca documentación liberada de estos. [5]

Una de las empresas que desarrolla productos similares es Tecnotree, esta tiene su sede en Espoo, Finlandia, y posee una fuerte presencia mundial. Uno de sus principales productos es SDS Centre, este ofrece una rica gama de servicios de mensajería de SDS que se puede personalizar fácilmente e integrarse con los servicios PMR (del inglés: Professional Mobile Radio). SDS Centre incluye almacenamiento y reenvío de SDSs, compositor SDS, SDS a correo electrónico y SDS a buscapersonas. La funcionalidad de almacenamiento y reenvío de SDS Centre asegura que los mensajes de texto se entreguen a los terminales incluso cuando no están inmediatamente disponibles en la red. El sistema guarda los mensajes y luego los reenvía cuando los destinatarios están disponibles otra vez.
SDS Centre también mejora la conectividad entre las redes TETRA y GSM, posee conversión de mensajes SDS a formato SMS y viceversa. Esto hace posible que usuarios de TETRA y usuarios de GSM puedan enviar mensajes de texto entre sí. [5]

1.4 TETRA

Los sistemas de radiocomunicaciones móviles son sistemas de radiotelefonía privada que ofrecen servicios a un grupo cerrado de usuarios, generalmente una flota de vehículos por ejemplo policía, bomberos, ambulancias, transporte público; con ámbito local y en algún caso regional, sin necesidad o posibilidad de conexión a la red pública telefónica. También existe la modalidad pública, similar a la privada, pero en la que una empresa presta el servicio a terceros, aportando la infraestructura y gestionando la red. [3][9]

TETRA define un sistema digital de radio móvil que aporta mayor privacidad y confidencialidad, más calidad de audio, mejora la velocidad de transmisión de datos, además de la capacidad de acceso a otras redes como Internet, red telefónica fija o móvil. En consecuencia, TETRA es un estándar global de comunicación de radio en la misma manera que el GSM es el estándar de telefonía móvil. TETRA ha sido desarrollado para satisfacer las necesidades de los más exigentes clientes que necesitan rápida conexión

punto a punto y punto a múltiples puntos vía radio. Los usuarios son típicamente de seguridad pública y de organismos de seguridad como policía, bomberos y fuerzas de rescate, servicios de ambulancia, en la frontera, los guardias y otros usuarios profesionales móviles, como compañías de transporte, servicios de mensajería, empresas de servicios públicos de energía, aeropuertos, etc. [3][9]

La red TETRA actualmente es una evolución de la radiotelefonía convencional, que desde un primer momento tuvo como prioridad la seguridad pública y los servicios de emergencia. El éxito de estas comunicaciones obligó a crear sistemas alternativos para poder hacer un uso más optimizado del espectro radioeléctrico. [3]

El estándar TETRA ha sido desarrollado para proporcionar un rendimiento óptimo en la gama de frecuencias de 300 a 1000 MHz, aunque en la práctica, los sistemas TETRA se concentran en unas pocas bandas de frecuencia, como resultado de la armonización europea y también de la armonización del espectro mundial. Este número reducido de frecuencias de radio ha contribuido en gran medida al desarrollo de la situación actual del mercado TETRA, con múltiples proveedores que fabrican productos interoperables en el mismo espectro de frecuencias de radio. [3]

Dentro de las prestaciones de TETRA se encuentran: [3]

- ✓ Seguridad en las comunicaciones.
- ✓ Comunicaciones dúplex de voz y datos o semi-dúplex de voz junto con datos por el mismo equipo.
- ✓ Diseño específico para la transmisión optimizada de datos por paquetes.
- ✓ Alta velocidad de transmisión de datos.
- ✓ Elevada calidad de señales de voz y datos.
- ✓ Flexibilidad de configuraciones, desde un sistema de monoemplazamiento para una aplicación local hasta complejas redes multiemplazamiento de ámbitos regional o nacional.
- ✓ Identificación y redireccionamiento de llamadas.
- ✓ Múltiples servicios suplementarios.
- ✓ Amplia gama de interfaces y cabeceras para interfuncionamiento con redes telefónicas externas.

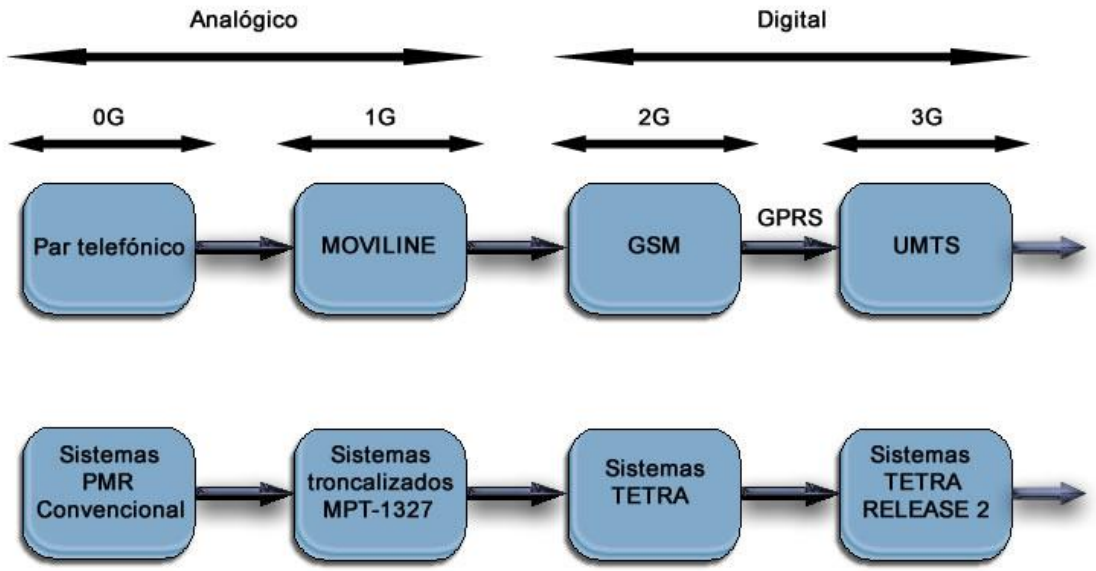

Figura #1. Evolución de los Sistemas de Comunicación. [10]

Todas estas prestaciones contribuyen a que TETRA sea un estándar de calidad y eficiencia muy adecuado para aplicaciones profesionales en servicios de seguridad y emergencia, de transporte público y de distribución. En la actualidad los fabricantes de TETRA trabajan para hacer que sus productos sean compatibles, permitiendo a los clientes comprar radios e infraestructuras en un mercado abierto y competitivo. En la actualidad, más de 15 fabricantes ofrecen redes TETRA, radios TETRA, o ambas cosas. [3]

Ventajas

1. Utilización de una banda de frecuencias más baja, lo que supone una menor necesidad de equipos repetidores para dar cobertura a una misma zona geográfica.
2. Infraestructura propia separada de las redes de telefonía móvil públicas.
3. Puede trabajar en modo terminal a terminal, en caso de fallo en las comunicaciones.
4. Es un sistema digital moderno, por lo que la calidad de audio es superior al implementar sistemas más modernos de compresión de voz.
5. Las capacidades de transmisión de datos están definidas en el propio estándar inicial y sólo son comparables al actual estándar GPRS (del inglés: General Packet Radio Service).

6. Mejor aprovechamiento del canal, ya que permite comunicaciones semi-dúplex, como la radio convencional; o full-dúplex, como el teléfono en casos necesarios, utilizando los canales no ocupados.
7. Menor grado de saturación, ya que el propio estándar garantiza una capacidad por defecto superior al doble de los canales convencionales en uso.
8. Permite comunicaciones uno a muchos, lo que mejora la gestión de grupos en caso de comunicaciones para coordinación de urgencias.
9. Dispone de terminales específicos para cada necesidad: portátiles (equiparables a teléfonos móviles), móviles (vehículos) y terminales para bases.

Desventajas

1. Requiere una menor densidad de usuarios que los servicios GSM, debido al tipo de modulación realizada.
2. Los terminales tienen un precio mucho mayor al estar dirigidos a sectores diferentes y no disponer de un mercado masivo de clientes.
3. Las transferencias de datos son más lentas, aunque se está mejorando en versiones más modernas.
4. Debido a la baja modulación de frecuencia, los terminales pueden interferir con dispositivos electrónicos sensibles, como marcapasos o desfibriladores.

La seguridad de las comunicaciones es un requisito fundamental para las organizaciones de seguridad pública. Las redes TETRA incorporan una serie de medidas de seguridad entre las que se incluyen: [11]

- ✓ Seguridad de la información: en los mercados de Misiones Críticas, las actividades operativas pueden quedar seriamente comprometidas si se interceptan los mensajes de voz o de datos; TETRA resuelve este problema.

- ✓ Acceso a la red: Este estándar resuelve también un posible fallo de seguridad en las redes y los dispositivos. La autenticación de la conexión entre el dispositivo y la red controla el tráfico para asegurar que las transmisiones provienen de usuarios autorizados. Además, si se pierde o se roba un terminal, puede ser deshabilitada inmediatamente.

- ✓ Intimidad: TETRA permite la partición de las redes, de esta forma, se asegura que las distintas organizaciones tienen acceso a comunicaciones privadas para sus operaciones rutinarias a través de su propia VPN (del inglés: Red Virtual Privada), canalizada de forma segura por el sistema.

1.5 Infraestructura NEBULA

Hoy en día, los sistemas de despliegue rápido son un elemento clave para las agencias de seguridad pública. En una situación de emergencia, las unidades de respuesta rápida tienen la obligación de actuar tanto si el área de actuación está dentro del área de cobertura como si no. Está comprobado que soluciones flexibles, fáciles de transportar, proporcionan la mejor solución para ese tipo de situaciones. NEBULA, es la infraestructura transportable de TELTRONIC. Lista para desplegarse rápidamente, incluye todos los módulos necesarios para trabajar de manera aislada o como una estación base móvil conectada a una red fija. NEBULA complementa su sistema con la cobertura necesaria para la respuesta a emergencias, ocurran donde ocurran. [12]

NEBULA ofrece un conjunto completo de servicios de red, soportando todo tipo de llamadas de voz y datos, gestión inteligente de llamadas de grupo, soluciones optimizadas de localización de vehículos (AVL). [12]

Las soluciones de la infraestructura NEBULA de TELTRONIC proporcionan servicios de comunicaciones digitales en multitud de países y en segmentos tan dispares como seguridad pública, compañías petrolíferas y de gas, medios de transporte u operadores públicos. Uno de los principales motivos de su éxito es su potente arquitectura basada completamente en IP (del inglés: Internet Protocol), que hace de NEBULA la plataforma ideal para comunicaciones fiables y seguras, tanto en sistemas pequeños como en grandes sistemas con cobertura regional o nacional. Su excepcional flexibilidad posibilita su migración futura a nuevos estándares de comunicaciones. Los usuarios de NEBULA tienen la seguridad de que su sistema podrá actualizarse a versiones más potentes de tecnología según las exigencias del mercado. [12]

1.6 N2AClient

Este es un API que implementa el protocolo TDP (del inglés: TETRA Dispatcher Protocol) para el envío de mensajes dentro de la infraestructura NEBULA de TELTRONIC. Además es el responsable de establecer la conexión entre el sistema y la infraestructura de red, así como el envío del mensaje encapsulando el WBXML (del inglés: Wireless Binary XML) en los mensajes SDS con destino a los terminales para su visualización.

Cuenta con un grupo de funcionalidades para realizar el envío de los SDS las cuales son:

- ✓ Establecer una conexión entre el cliente y el servidor.
- ✓ El envío de los mensajes SDS.
- ✓ La recepción de la respuesta ofrecida por NEBULA luego de realizar un envío.

1.7 Mensajes WAP PUSH

Una de las funcionalidades del Portal Web de SERWAP es el envío de mensajes WAP PUSH hacia la infraestructura NEBULA.

El modelo de envío de mensajes de tipo PUSH a través del protocolo WAP está basado en el modelo Cliente/Servidor, pero en el cual no existe una solicitud específica para la misma por parte del cliente antes del proceso de envío de información desde el servidor. Estos mensajes pueden ser codificados de manera que pueda ser más eficiente y optimizar el envío de los mismos a través de la infraestructura correspondiente y para lograr una mayor rapidez a la hora de ser enviados y recibidos, pues con los mensajes de una gran cantidad de caracteres pueden reducirse inclusive hasta la mitad de su tamaño, evitando con esto el engorroso proceso de segmentación, que puede incurrir en la pérdida de un mensaje en su totalidad al no poder ser reensamblado en el terminal. [14]

Existen 3 clasificaciones para estos mensajes que pueden ser: [14]

- ✓ SI (del inglés: Service Indication)
- ✓ SL (del inglés: Service Loading)
- ✓ CO (del inglés: Cache Operation)

Los mensajes de tipo *Service Indication* contienen un pequeño texto para el usuario del terminal, así como una URI (del inglés: Uniform Resource Identifier) mediante la cual se puede acceder a un servicio y es

responsabilidad del usuario decidir si visitar el servicio que se ha enviado en el momento que llega el mensaje o en otro instante. Seguidamente se muestra un ejemplo de este tipo de mensaje. [14]

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE si PUBLIC "-//WAPFORUM//DTD SI 1.0//EN" "http://www.wapforum.org/DTD/si.dtd">
<si>
<indication href="http://192.168.8.1:8080/Portal" created="2009-10-21T11:04:00Z" si-
expires="2009-11-02T11:04:00Z" action="signal-low">Soluciones TETRA</indication>
</si>
```

Ejemplo #1. XML correspondiente al mensaje SI

En contraste con los mensajes de tipo SL los *Service Loading* no requieren acciones por parte de los usuarios. Estos mensajes contienen una URI mediante la cual se puede acceder al servicio que puede ser cargado sin la confirmación del usuario final. El contenido puede ser ejecutado o cargado en la memoria caché del navegador utilizado en el dispositivo móvil. Seguidamente se muestra un ejemplo de este tipo de mensaje. [14]

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sl PUBLIC "-//WAPFORUM//DTD SL 1.0//EN"
"http://www.wapforum.org/DTD/sl.dtd">
<sl href="http://192.168.8.1:8080/Portal" action="execute-low"></sl>
```

Ejemplo #2. XML correspondiente al mensaje SL

Mediante un mensaje del tipo *Cache Operation* se provee de un medio para invalidar objetos específicos o todos los objetos que comparten el mismo prefijo de una URI dada, de la memoria caché del navegador del cliente. Esta característica es útil cuando el tiempo de expiración del contenido guardado en la caché no puede ser determinado con antelación (por ejemplo las vistas de un correo electrónico) y los contenidos cambian con más rapidez que lo que el cliente los accede. Seguidamente se muestra un ejemplo de este tipo de mensaje. [14]

```
<?xml version="1.0"?>
<!DOCTYPE co PUBLIC "-//WAPFORUM//DTD CO 1.0//EN"
"http://www.wapforum.org/DTD/co_1.0.dtd">
<co>
<invalidate-object uri="Portal.jsp"></invalidate-object>
```

```
<invalidate-service uri="http://192.168.8.1:8080/"></invalidate-service>
</co>
```

Ejemplo #3. XML correspondiente al mensaje CO

1.8 Proyecto SERWAP

La empresa española TELTRONIC, en correspondencia con su deseo de proveer productos tecnológicos al mercado internacional, ha brindado nuevos servicios a sus clientes del mercado de radio profesional, con una mejor calidad y mayores beneficios. Para lograr esto, se creó en la Universidad de las Ciencias Informáticas el proyecto Servidor de Aplicaciones WAP, más conocido por sus siglas SERWAP.

Dicho proyecto consiste en un sistema de aplicaciones que automatizan los procesos que llevan a cabo diferentes agentes de radio profesional, como policías, bomberos y agentes forestales. Las aplicaciones son diversas, las cuales necesitan acceder a fuentes de información para obtener los datos almacenados en bases de datos. Se garantiza que el acceso a dicha información se realice de forma segura equilibrando la seguridad con la eficiencia y rapidez del flujo de datos. Los programas y clientes implementados de diferentes formas y ubicados en diferentes lugares geográficos pueden acceder a dicha información. Además permite la interoperabilidad entre los portales WAP y WEB por medio de protocolos estándares y abiertos.

1.9 Metodología de desarrollo

Hacer un software resulta algo complejo, para ello se requiere de una guía o un plan de trabajo que el equipo de desarrollo se debe trazar o planificar, y así de tal forma que el producto o resultado del trabajo tenga éxito, o sea que se cumplan los objetivos y los requerimientos, se entregue en tiempo, forma y con la calidad que requiere o que el cliente exija. Para lograr cumplir todo esto con una mayor facilidad de trabajo, existen las metodologías de desarrollo.

Una metodología de desarrollo de software en ingeniería de software es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. [15]

Una metodología puede seguir uno o varios modelos de ciclo de vida, es decir, el ciclo de vida indica qué es lo que hay que obtener a lo largo del desarrollo del proyecto, pero no cómo hacerlo. La metodología

indica cómo hay que obtener los distintos productos parciales y finales. Debe ajustarse a los objetivos y cubrir el ciclo entero de desarrollo de software. [15]

Para ello la metodología ha de realizar las siguientes etapas:

- ✓ Investigación
- ✓ Análisis de requisitos
- ✓ Diseño
- ✓ Desarrollo
- ✓ Pruebas

La metodología de desarrollo de software debe: [15]

- ✓ Integrar las distintas fases del ciclo de desarrollo
- ✓ Incluir la realización de validaciones.
- ✓ Soportar la determinación de la exactitud del sistema a través del ciclo de desarrollo.
- ✓ Ser la base de una comunicación efectiva.
- ✓ Funcionar en un entorno dinámico orientado al usuario
- ✓ Especificar claramente los responsables de los resultados
- ✓ Poder emplearse en un entorno amplio de proyectos software
- ✓ Soportar la eventual evolución del sistema.
- ✓ Contener actividades que conducen a mejorar el proceso de desarrollo de software.

Cada aproximación al desarrollo de software está basada en unos objetivos. Por ello la metodología que se elija debe recoger el aspecto filosófico de la aproximación deseada, es decir que los objetivos generales del desarrollo deben estar implementados en la metodología de desarrollo.

Metodologías Agiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código.	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo.
Especialmente preparados para cambios durante el proyecto.	Cierta resistencia de los cambios.
Impuestas internamente (por el equipo de desarrollo)	Impuestas externamente
Proceso menos controlado, con pocos principios.	Proceso mucho más controlado, con numerosas políticas/normas.
No existe contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos	Más artefactos.
Pocos roles	Más roles.
Menos énfasis en la arquitectura del software.	La arquitectura del software es esencial y se expresa mediante modelos.

Tabla #1. Comparación entre las metodologías ágiles y las tradicionales. [16]

Entre las metodologías tradicionales o pesadas, la más común es RUP (del inglés: Rational Unified Process), un proceso iterativo e incremental. Esta metodología presenta desventajas para su aplicación. Algunas de estas son: la generación de una gran cantidad de documentación que no genera valor respecto a la calidad del desarrollo; es necesario incluir a más personas en el equipo de desarrollo, tales como especialistas en los diseños de los casos de usos, etc. [17]

Una de las cualidades de las metodologías ágiles es su sencillez, tanto en su aprendizaje como en su aplicación, reduciéndose así los costos de implantación en un equipo de desarrollo.

Entre las metodologías ágiles más populares sobresale XP (del inglés: eXtreme Programming). A pesar de sus ventajas, ésta no es adecuada para desarrollar esta aplicación, ya que está concebida para proyectos pequeños y el que se necesita es de alcance medio, también porque en esta metodología el cliente forma parte del equipo de desarrollo. Además en el desarrollo del software se prevé la distribución por capas del código donde un programador puede no saber implementar cambios en todo el código; mientras que, XP plantea una propiedad colectiva del código, donde todos los programadores pueden realizar cambios en cualquier parte del código en cualquier momento. [17]

Otra de las metodologías más usadas es Scrums. Se puede resumir que tiene dos características principales: una es la realización de reuniones a lo largo del proyecto, que para dicha metodología estas son las verdaderas protagonistas, especialmente la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración. Para proyectos de desarrollo donde los requisitos no cambian constantemente, esta gran cantidad de reuniones constituye como ineficiencia el desvío de tiempo de implementación para realizar reuniones. La otra característica importante es que el desarrollo de software se realiza mediante iteraciones, denominadas *sprints*, con una duración de 30 días. El resultado de cada *sprint* es un incremento ejecutable que se muestra al cliente. Esta metodología está indicada especialmente para proyectos con un cambio rápido de requisitos, por lo que tampoco es aplicable. [17]

1.9.1 Metodología FDD

FDD (del inglés: Feature Driven Development) se podría considerar a medio camino entre RUP y XP, aunque al seguir siendo un proceso ligero es más similar a este último. FDD está pensado para proyectos con tiempo de desarrollo relativamente cortos (menos de un año). Se basa en un proceso iterativo con iteraciones cortas (aproximadamente 2 semanas) que producen un software funcional que el cliente y la dirección de la empresa pueden ver y monitorizar. Las iteraciones se deciden en base a features (de ahí el nombre del proceso) o funcionalidades, que son pequeñas partes del software con significado para el cliente. También define métricas para seguir el proceso de desarrollo de la aplicación, útiles para el cliente y la dirección de la empresa, y que pueden ayudar, además para conocer el estado actual del desarrollo, a realizar mejores estimaciones en proyectos futuros. [17] Por estas razones esta metodología es la aplicada para el desarrollo del proyecto SERWAP y sus módulos asociados, además de ser la propuesta por el cliente.

Un proyecto que utilice la metodología FDD se divide en 5 fases: [17]

1. Desarrollo de un modelo general.
2. Construcción de la lista de funcionalidades.
3. Plan de releases (emisiones) en base a las funcionalidades a implementar.
4. Diseñar en base a las funcionalidades.
5. Implementar en base a las funcionalidades.

Las primeras tres fases ocupan parte del tiempo en las primeras iteraciones, siendo las dos últimas las que absorben la mayor parte del tiempo según va avanzando el proyecto, limitándose las primeras a un proceso de refinamiento. [17]

El trabajo (tanto de modelado como de desarrollo) se realiza en grupo, aunque siempre habrá un responsable último (arquitecto jefe), con mayor experiencia, que tendrá la última palabra en caso de no llegar a un acuerdo. Al hacerlo en grupo se consigue que todos formen parte del proyecto y que los inexpertos aprendan de las discusiones de los más experimentados. Las funcionalidades a implementar se dividen entre los distintos subgrupos del equipo, y se procede a implementarlas. [17]
En el proceso de implementar la funcionalidad también se contemplan como partes del mismo (en otros métodos se describen como actividades independientes) la preparación y ejecución de pruebas. [17]

De manera general se puede resumir que FDD: [17]

✓ Ayuda al equipo a producir resultados periódicos y tangibles.

✓ Esta metodología utiliza pequeños bloques llamados features (funcionalidades), que contienen la funcionalidad del sistema.

✓ Organiza los bloques que están relacionados entre sí, en una lista llamada feature set.

✓ Hace énfasis en la obtención de resultados cada dos semanas.

✓ Asegura en gran parte la calidad del software entregado.

✓ Es adaptativo, pues permite realizar cambios de último momento debido a nuevos requerimientos y a las necesidades del negocio.

1.10 Herramienta de Modelado: StarUML

StarUML es una herramienta de código abierto para modelar de forma rápida, flexible, extensible, práctica y de libre acceso a la plataforma UML (del inglés: Unified Modeling Language) /MDA (del inglés: Model-Driven Architecture) que funciona en Windows. Esta herramienta sirve como alternativa a las existentes como Rational Rose, Visual Paradigm, etc. [18]

Es una herramienta multilenguaje que se puede utilizar para modelar software programados en C/C++, Java, Visual Basic, Delphi, JScript, VBScript, C# y VB.NET. Tiene soporte para UML 2.0, ha sido implementado de forma tal que provee al usuario una interfaz amigable, siendo destacable sus características de diálogo rápido, manipulación utilizando el teclado y visión general de diagramas, entre otras. [19]

En esta herramienta de modelado se destacan las siguientes características: [20]

Diagramas UML que posee:

- ✓ Diagrama de Caso de Uso
- ✓ Diagrama de Clase
- ✓ Diagrama de Secuencia
- ✓ Diagrama de Colaboración
- ✓ Diagrama de Actividades
- ✓ Diagrama de Componentes
- ✓ Diagrama de Despliegue
- ✓ Diagrama de Composición de Estructura (UML 2.0).

Soporte de generador de código e ingeniería inversa para los lenguajes:

- ✓ Java
- ✓ C++
- ✓ C#

Debido a las características planteadas, esta herramienta es la seleccionada por el líder del proyecto SERWAP, por lo que se utilizará para el modelado de la aplicación que dará solución a la problemática planteada.

1.11 Framework Spring

Un Framework es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, en base a la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Para el desarrollo de software en ocasiones se utilizan Frameworks, debido a que constituyen estructuras de soporte definidas mediante la cual otros proyectos de software pueden ser organizados y desarrollados. Son diseñados con el intento de facilitar el desarrollo del producto, permitiendo a los diseñadores e implementadores concentrarse en aspectos de mayor nivel.

Spring es un Framework de código abierto de desarrollo de aplicaciones para la plataforma Java que ha ido revolucionando la manera de programar promoviendo buenas prácticas de diseño e implementación. El mismo se encarga de construir clases una vez que se desee, construyendo los objetos según sea necesario y de esta forma facilitar el desarrollo. [21]

A pesar de que Spring Framework no obliga a usar un modelo de programación en particular, se ha popularizado considerándosele una alternativa al modelo de Enterprise JavaBeans. Por su diseño el framework ofrece mucha libertad a los desarrolladores en Java y soluciones muy bien documentadas y fáciles de usar para las prácticas comunes en la industria. Las características de éste pueden emplearse en cualquier aplicación hecha en Java. Spring es el más popular y de los más ambiciosos de los Frameworks de peso ligero. Además está diseñado para facilitar una flexibilidad arquitectónica.

Ofrece varias opciones de configuración de su aplicación, la más utilizada radica en el uso de ficheros XML; generando notables ventajas, entre las que existe, que una vez diseñado y puesto en producción es posible extender el XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación, es sencillo de entender su estructura y procesarla. [21]

Una de las grandes ventajas que posee Spring es que está diseñado como una serie de módulos que pueden trabajar independientemente uno de otro. Además intenta mantener un mínimo acoplamiento entre la aplicación y el propio Framework de forma que podría ser desvinculada de él sin mucha dificultad. [22]

1.12 Lenguaje de Programación: Java

Java es un lenguaje de programación orientado a objetos. Es uno de los lenguajes de programación más conocidos. Es robusto porque maneja la memoria de la computadora por el usuario, pudiendo ejecutar varias líneas de código al mismo tiempo. Ofrece un entorno de ejecución seguro para programas con acceso a red. Logra ser un lenguaje que no depende de una arquitectura computacional definida ya que su código es interpretado por diferentes computadoras de igual manera teniendo solamente un intérprete. [23]

Java es toda una tecnología orientada al desarrollo de *software.* Hoy en día, esta tecnología ha cobrado mucha importancia en el ámbito de Internet gracias a su plataforma Java EE. [23]

La plataforma Java se estructura: [24]

- ✓ Como lenguaje de programación.
- ✓ La máquina virtual de Java que permite la portabilidad en ejecución.
- ✓ El API Java, una biblioteca estándar para el lenguaje.

Una de las principales características que ha favorecido el crecimiento y difusión del lenguaje Java es su capacidad de que el código funcione sobre cualquier plataforma de *software* y *hardware* que cuente con una implementación de la máquina virtual [24].

Java posee características y ventajas considerables, entre estas se encuentran: [25]

Seguridad: Ofrece un entorno de ejecución seguro para programas con acceso a red. La máquina virtual de Java lleva a cabo una verificación estricta del código antes de la ejecución, asegurando que éste no trate de saltarse las protecciones impuesta por el lenguaje, utilizar punteros que accedan directamente a memoria o usar el objeto equivocado.

Código reutilizable: Debido a la orientación a objetos, se consiguen características como la facilidad en el desarrollo, la reutilización y la mayor calidad del código.

Orientado a objetos: Lo cual lo hace muy útil para la representación de entidades tal y como las crean las personas.

Multihilo: Lo que permite la realización de muchas actividades simultáneas en un programa.

Debido a las características planteadas, así como por el hecho de ser de uso libre, la tecnología Java fue seleccionada por el equipo de trabajo del proyecto SERWAP para implementar el sistema requerido por el cliente, y es por esto que será utilizado para dar solución a la problemática planteada.

1.13 Entorno de desarrollo: Eclipse

El Eclipse es un IDE o entorno de desarrollo integrado de código abierto y multiplataforma. Se utiliza para integrar herramientas de desarrollo que corren sobre un amplio rango de sistemas operativos, con una arquitectura abierta y basada en plug-ins o módulos. [26]

Eclipse es una gran estructura formada por un núcleo y muchos plug-ins que van conformando la funcionalidad final.

La característica clave del Eclipse es su extensibilidad a través de los plug-ins que no son más que la mínima unidad de la plataforma que puede ser desarrollado por separado y que le aporta una nueva funcionalidad. La forma en que los plug-ins interactúan es mediante interfaces o puntos de extensión; así, las nuevas aportaciones se integran sin dificultad ni conflictos. Actualmente Eclipse dispone de un editor de texto, resaltado de sintaxis, compilación en tiempo real, pruebas unitarias con JUnit, control de versiones con CVS (del inglés Concurrent Versions System), asistentes para creación de proyectos, clases, pruebas, etc.

También tiene como ventaja que los errores de compilación se muestran en tiempo real subrayando el fragmento de código adecuado. Y además el entorno, automáticamente, compila los archivos salvados. [26]

Dicho IDE ha alcanzado un grado de madurez, así como más robustez y rendimiento. Posee herramientas para desarrollar aplicaciones de escritorio y Web además de Servicios Web utilizando diferentes servidores como el Apache Tomcat. [27]

Este entorno de desarrollo es el seleccionado para el desarrollo del proyecto SERWAP, debido a sus características, ventajas y facilidades de trabajo. Por lo tanto también es el indicado para la implementación y desarrollo de la aplicación que dará solución a la problemática planteada en el presente documento.

1.14 Servidor Web: Apache Tomcat

Apache Tomcat es un servidor web con soporte de servlets y JSPs (del inglés: Java Server Pages). Es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad. [28]

Dicho servidor web es fácil de obtener desde Internet debido en mayor medida a que es totalmente libre; ocasionando por tal motivo su amplia extensión para el uso de los desarrolladores de software. Los requisitos de software para la utilización de este servidor recaen en la necesidad de disponer de la Máquina Virtual de Java para su adecuado funcionamiento. Resulta realmente sencillo de instalar, con pocos requerimientos de capacidad en disco y compatible con las APIs más recientes de Java. [28]
Apache Tomcat resalta por su fiabilidad, debido a ello innumerables empresas lo utilizan aunque resulta imposible contabilizar su número exactamente. Cuenta con el trabajo de miles de desarrolladores que contribuyen con su código y ponen a disposición de toda la comunidad las últimas actualizaciones. Desplegar aplicaciones para Apache Tomcat se hace en relativamente poco tiempo, el resultado de probar la aplicación web con este servidor asegura de que también pueda desplegarse en otros servidores de aplicaciones web. [28]

1.15 Sistema gestor de Base de Datos: MySQL

Un sistema gestor de base de datos es un conjunto de programas que permiten crear y mantener una base de datos, asegurando su integridad, confidencialidad y seguridad.
MySQL se ha convertido en la base de datos de código abierto más popular debido a su alto rendimiento, alta fiabilidad, facilidad de uso, es relacional, multihilo y multiusuario. Se ahorra tiempo y dinero al usarlo en grandes volúmenes de sitios Web, los sistemas críticos de negocio y paquetes de software. Además en aplicaciones Web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en la lectura de datos, lo que hacen a MySQL ideal para este tipo de aplicaciones. [29]

MySQL se ejecuta en más de 20 plataformas, incluyendo Linux, Windows, Mac OS, Solaris, que le da el tipo de flexibilidad. Ofrece una amplia gama de herramientas de base de datos, asistencia técnica, capacitación y servicios de consultoría para aplicarlo con éxito. Lo que hace a este gestor multiplataforma, permite procedimientos almacenados, "*triggers"* (disparadores), soporte para SSL, soporte completo para Unicode, soporta el estándar SQL, etc.
Los clientes se pueden conectar al servidor MySQL usando sockets TCP/IP en cualquier plataforma. [29]
Este gestor de Base de Datos es el seleccionado en el proyecto SERWAP en su versión MySQL 5.01 debido a las características planteadas, por lo que también es el escogido para la aplicación que dará solución a la problemática presente.

1.16 Control de Versiones: SVN Subversion

El Subversion es un controlador de versiones de software libre que mantiene los registros de todos los cambios que se han realizado a los archivos de un software, lo que permite el trabajo de distintos desarrolladores en un mismo proyecto, esta herramienta es usada por los programadores de software. Es uno de los sistemas de control de versiones más modernos y utiliza un sistema con repositorio centralizado y fue diseñado como reemplazo del sistema más utilizado hasta la fecha, CVS (del inglés: Concurrent Versions System). Sus principales características están relacionadas con gestionar las modificaciones durante el desarrollo mediante el seguimiento del historial de los archivos y directorios a través de copias y renombrados, además sus modificaciones son automáticas por lo que permite que varias personas trabajen sobre los mismos ficheros enviando solo las diferencias en ambas direcciones. [30]

Para la interacción con el servidor de control de versiones del proyecto SERWAP el cliente seleccionado fue TortoiseSVN, ya que es extremadamente rápido, cómodo de utilizar, provee integración con el explorador de Windows y con el Eclipse que es el IDE utilizado además de presentar una interfaz muy amigable para el usuario. El estado de cada carpeta y fichero versionado se indica por pequeños iconos y de esta forma se puede ver fácilmente el estado en el que se encuentra la copia del trabajo [30].

1.17 Conclusiones

En este capítulo se realizó una descripción teórica sobre las redes TETRA, la infraestructura NEBULA, los SDS Center y SMS Center, el sistema SERWAP, entre otros conceptos importantes. También se realizó una fundamentación acerca de las tecnologías y herramientas que se utilizarán para el desarrollo de la aplicación que pretende dar solución al problema planteado.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En el presente capítulo se describen procesos y se define la problemática existente que da paso al nuevo sistema a desarrollar, exponiéndose así los objetos que requieren de automatización. Se expresan los principales conceptos y relaciones entre los mismos mediante un modelo de dominio. Además se presenta la estructura y funcionamiento de la nueva solución mediante la propuesta del sistema y se definen requerimientos, así como casos de uso y actores del sistema.

2.2 Definición del problema

El proyecto SERWAP consiste en un sistema de aplicaciones que automatizan los procesos que llevan a cabo diferentes agentes de radio profesional. SERWAP cuenta con un Portal Web para gestionar la información referente a los usuarios profesionales de radio como policías, bomberos y guardias forestales. El envío de mensajes WAP PUSH hacia la infraestructura es una característica novedosa del sistema, sin embargo pueden ocurrir errores durante dicha operación y el mensaje puede no llegar a su destino ya que no existe un mecanismo que gestione dicho proceso.

Durante el envío de un mensaje pueden ocurrir eventos no deseados, por lo que la información referente a dicho comportamiento es de gran importancia para los administradores del sistema. Mediante dicha información es posible conocer el estado de determinados mensajes, o si ocurrió algún error en el proceso de envío, entre otras utilidades. SERWAP no contiene un sistema que registre y muestre esta información.

2.3 Objeto de automatización

Se desea automatizar la gestión de envío de mensajes WAP PUSH lo que conducirá a la creación de un módulo que gestione el envío de los mismos, utilizará el API N2AClient para realizar el envío y registrará los mensajes en una base de datos.

Se procederá a la implementación de un nuevo módulo que será añadido al Portal Web de SERWAP donde se mostrará la información referente a los mensajes WAP PUSH enviados por el sistema.

Los datos a mostrar son:

- Identificador del mensaje.
- Hora y fecha de envío.
- Hora y fecha de entrega.

- Campos del mensaje.
- Respuesta del sistema.
- Estado (entregado o no entregado).

2.4 Propuesta del sistema

Se propone el desarrollo de un sistema que gestione el envío de mensajes WAP PUSH desde el Portal Web de SERWAP hacia la infraestructura NEBULA y muestre información a los administradores, referente a dichos mensajes enviados. Actualmente los mensajes se envían desde SERWAP hacia NEBULA sin la presencia de un mediador. Con el desarrollo de un SDSCenter ahora los mensajes pasarán a través de este donde se realizará el proceso de gestión de envío hacia NEBULA (Figura 2).

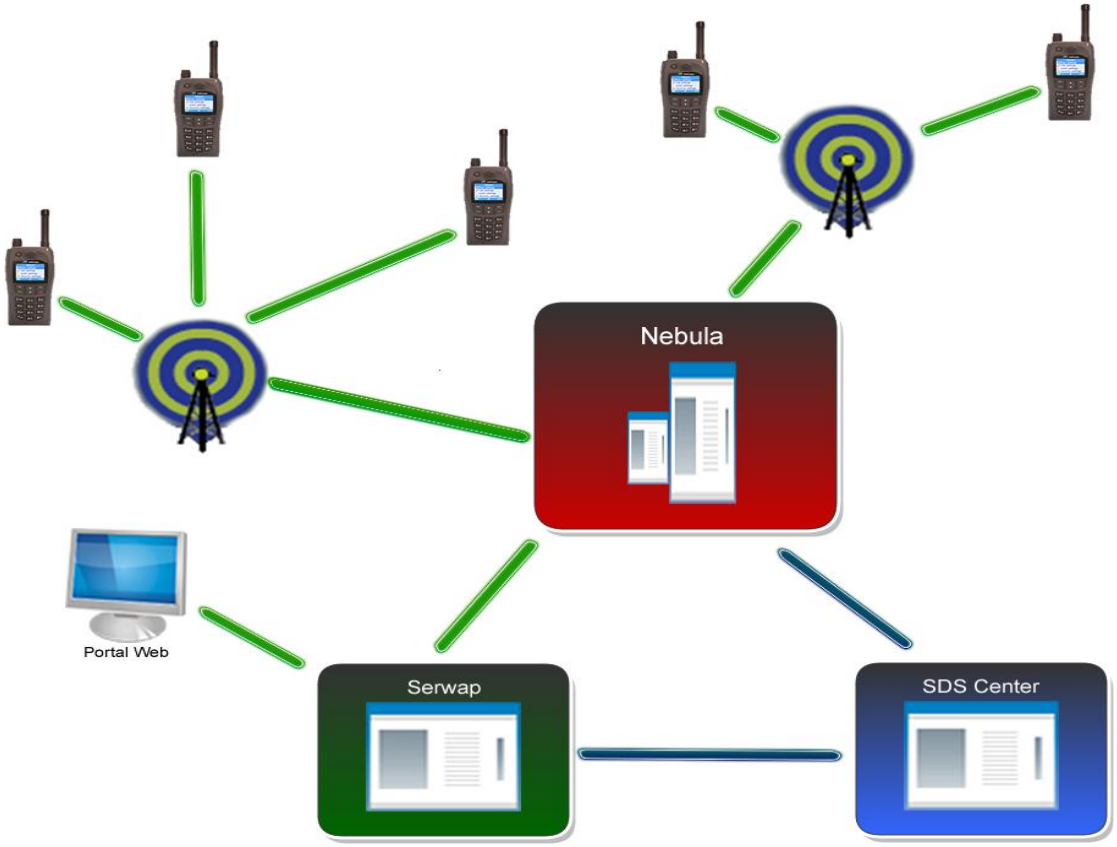

Figura #2. Propuesta del sistema.

El SDSCenter estará compuesto por dos módulos principales, los cuales son: el módulo Web que estará adjunto al Portal Web de SERWAP en el cual se visualizará la información referente a los mensajes WAP PUSH enviados, permitiendo el acceso a la base de datos. Al ser un módulo incluido en el Portal Web, deberá mantener su misma estructura y apariencia. También se contará con el módulo SDSCenter (refiriéndose al módulo de gestión de envío de los mensajes), donde se envían los mensajes hacia NEBULA mediante el API N2AClient. Además en dicho módulo se decodificarán los mensajes para registrarlos y guardar los campos de estos en la base de datos del sistema. También se utilizará un servidor de socket TCP contenido en el módulo mencionado, para recibir los mensajes que se envían desde el Portal Web a través del cliente de socket desde los servicios de Web Services o de RMI. (Figura 3).

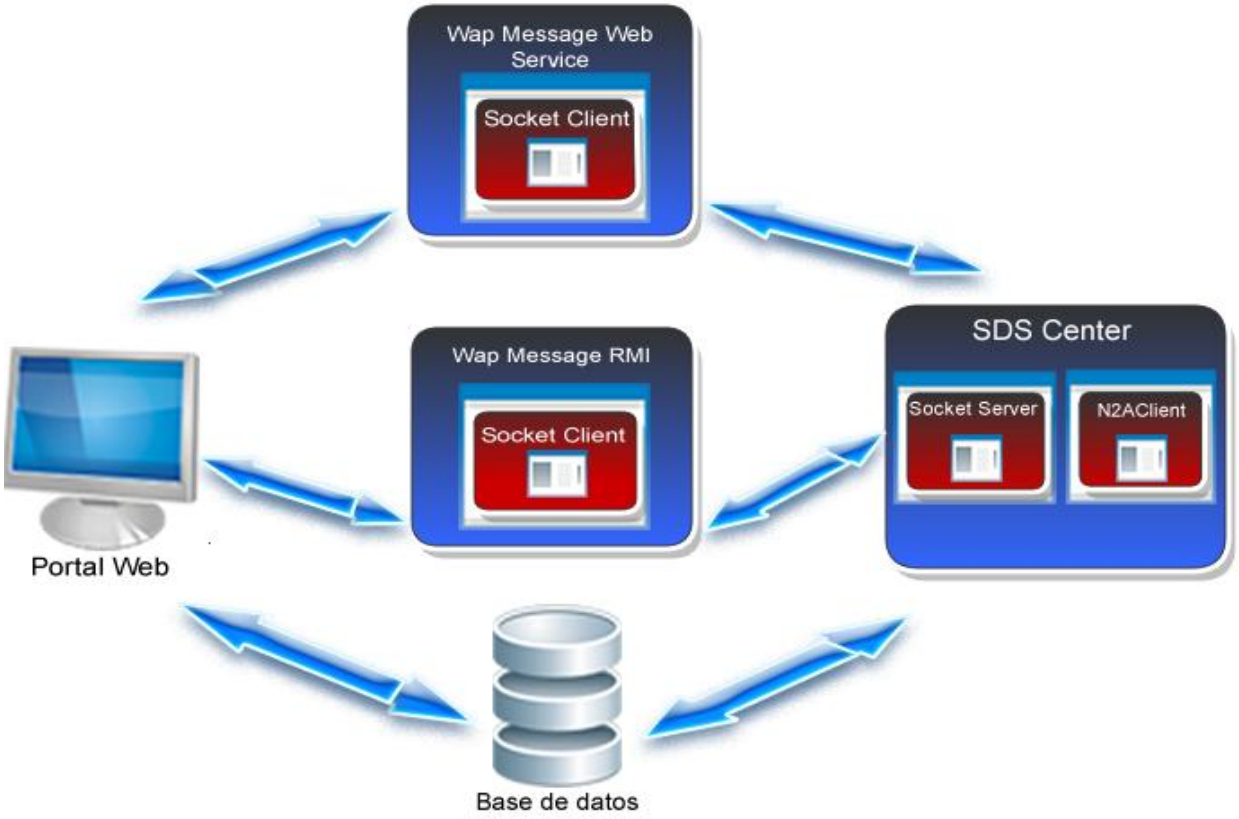

Figura #3. Nuevos módulos Web y SDSCenter.

2.5 Modelo de Dominio

Para representar los conceptos y relaciones involucrados en el entorno del sistema a desarrollar se realizó un modelo de dominio, ya que los procesos no se encuentran bien definidos. Para modelarlo se identificaron varios elementos enmarcados en la realidad física del problema, como eventos, transacciones, sistemas externos y objetos.

Conceptos:

- *Módulo Mostrar Mensajes:* Sistema donde se muestra la informacion de los mensajes enviados.
- *Módulo Enviar Mensajes:* Sistema donde se crean los mensajes a enviar.
- *SDSCenter:* Sistema que gestiona el envío de los mensajes.
- *Mensaje:* Objeto que representa a los mensajes.
- *Base de Datos:* Sistema donde se almacenan todos los mensajes.
- *NEBULA:* Infraestructura hacia donde se envían los mensajes.
- *Terminal:* Terminal hacia donde va destinado el mensaje.

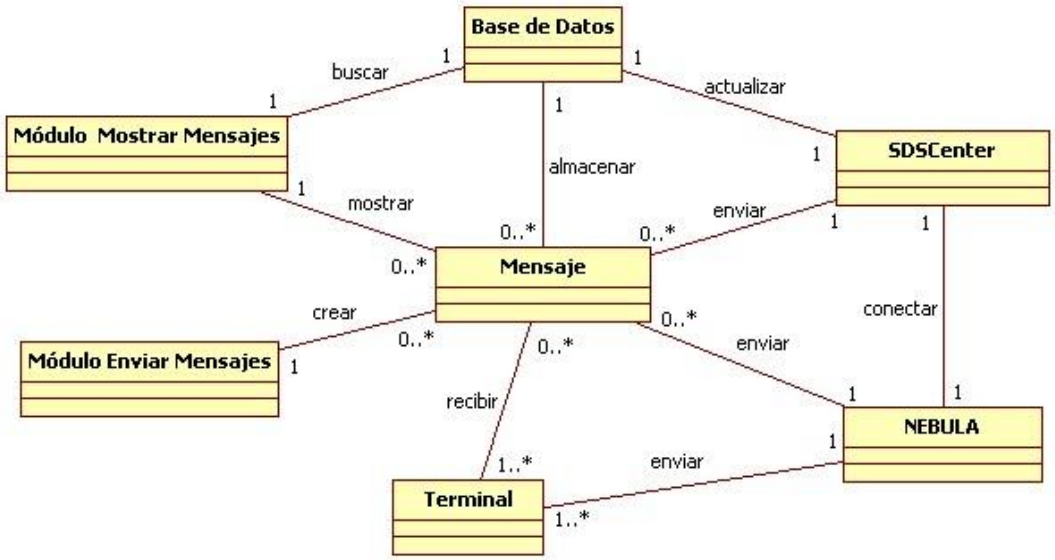

Figura #4. Modelo de Dominio

2.6 Especificación de los requisitos del Software

El IEEE Standard Glossary of Software Engineering Terminology (Glosario Estándar de la IEEE de Terminologías de Ingeniería del Software) define un requerimiento como condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo. [33]

Se ven como una condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente. Todas las ideas que los clientes, usuarios y miembros del equipo de proyecto tengan acerca de lo que debe hacer el sistema, deben ser analizadas como candidatas a requisitos. [34][35]

Los requisitos se pueden clasificar en: **funcionales y no funcionales.**

2.7 Requerimientos funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Estos se deben mantener invariables sin importar con qué propiedades o cualidades se relacionen**.** [34][35]

RF1. Listar mensajes enviados.

RF2. Eliminar mensajes enviados.

RF3. Buscar mensajes enviados.

Criterio de búsqueda:

- Fecha de inicio
- Fecha de fin

RF4. Iniciar servicios del SDSCenter.

RF5. Enviar mensajes WAP PUSH.

RF6. Actualizar cola de mensajes.

RF7. Eliminar mensajes de la cola.

RF8. Registrar mensajes en la cola.

- Identificador del mensaje
- Mensaje codificado
- Grupo
- Identidad del suscriptor
- Flota
- Red privada virtual
- Cantidad de envíos

RF9. Buscar mensajes en la cola.

RF10. Registrar mensajes enviados para mostrar.

- Identificador del mensaje
- Fecha de envío
- Fecha de entrega
- Hora de envío
- Hora de entrega
- Campos del mensaje
- Respuesta del sistema
- Estado

RF11. Decodificar mensajes WAP PUSH.

RF12. Registrar eventos y errores.

RF13. Registrar trazas de operaciones de los usuarios y el sistema.

2.8 Requerimientos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Normalmente están vinculados a requisitos funcionales.

Los requisitos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con toda la funcionalidad requerida, las propiedades no funcionales pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación. [34][35]

Software

- El sistema debe desplegarse en un servidor web Apache Tomcat v6.0.24.
- Los datos deben almacenarse dentro de un gestor de bases de datos MySQL v5.0.
- Se requiere de la Máquina Virtual de Java para ejecutar el sistema.
- El SDSCenter debe funcionar en los sistemas operativos Windows y Linux.

Hardware

El ordenador servidor donde se ejecutará el sistema debe tener requisitos como los de un ordenador Servidor PRO, modelo DELL PowerEdge 2950 con las siguientes prestaciones:

- Microprocesador Dual-Core Intel Xeon Processor 5100 3.0 GHz
- 2GB RAM
- 40 GB HDD
- RAID 5
- Extracción en caliente
- Ethernet 10/100 BASE T con tarjeta dual con fail – over y balanceo de carga.
- Tarjeta de video 1 monitor 8MB
- Fuente de alimentación con selector bitensión 110/220 voltios. Redundante. [36]

Estos requerimientos de hardware son los designados a petición del cliente, sin embargo vale destacar que el sistema podrá funcionar en cualquier otro hardware con características similares.

Restricciones en el diseño y la implementación

- El módulo web debe ajustarse a los patrones de diseño y arquitectura del Portal Web de SERWAP.

Apariencia o interfaz externa

- La apariencia debe ajustarse a la del Portal Web de SERWAP y seguir los estándares de este respetando los colores azul y blanco que representan a TELTRONIC.

Requisitos de Seguridad

- **Confidencialidad:** Solo tendrá acceso a la información de los mensajes WAP PUSH el administrador del Portal Web de SERWAP que antes debe estar autenticado y autorizado mediante el usuario y la contraseña.
- **Disponibilidad:** La información de los mensajes deberá estar disponible siempre que el administrador necesite consultarla, para ello será almacenada en una base de datos que permitirá el acceso a la información en cualquier momento.

2.9 Modelo del sistema

2.9.1 Definición de actores del sistema.

Actores	Descripción
Módulo de Envío de Mensajes WAP	Representa al módulo donde se crean los mensajes a enviar en el Portal Web de SERWAP.
Administrador	Representa al actor que tiene total acceso al sistema.

Tabla #2. Definición y descripción de los actores del sistema.

2.9.2 Diagrama de Casos de Uso

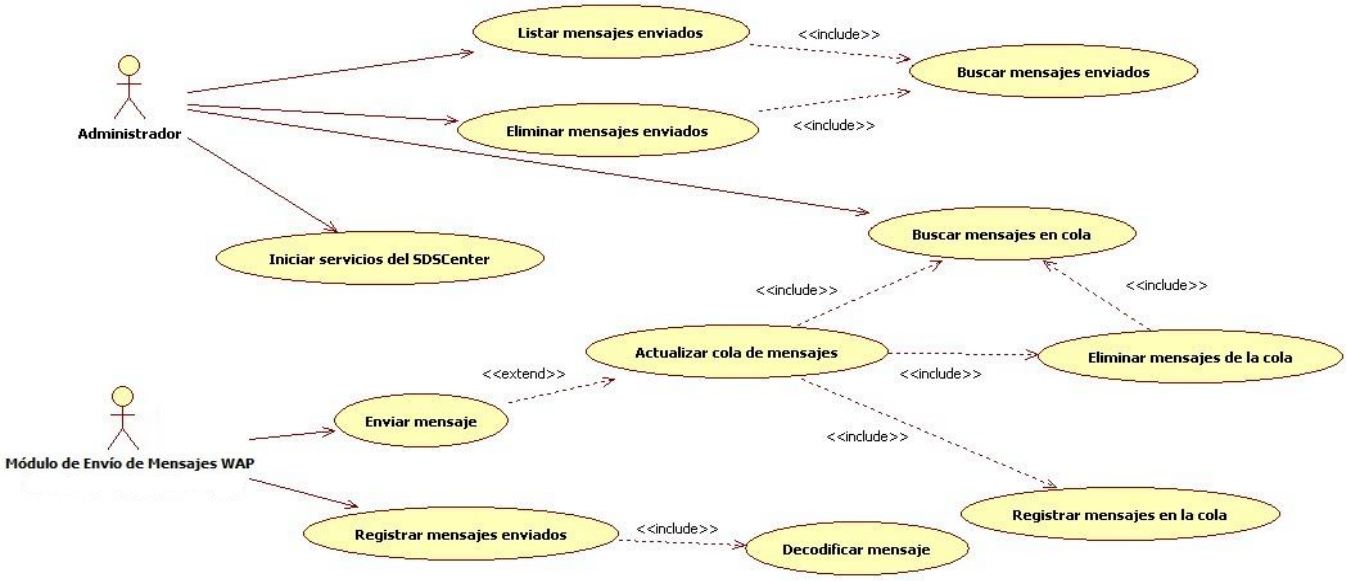

Figura #5. Diagrama de Casos de Uso.

2.9.3 Descripción de Casos de Uso

Caso de uso	
CU 1	Listar mensajes enviados.
Propósito	Mostrar un listado con los datos de los mensajes enviados que se encuentran registrados en la base de datos del sistema.
Actor: Administrador	

<table>
<tr><td colspan="3">Resumen: El caso de uso se inicia cuando el administrador accede a la funcionalidad “Listar Mensajes Enviados” contenida dentro del módulo “Mensajes WAP”. Selecciona mediante dos componentes de fecha un rango de tiempo, para consultar los mensajes que se han enviado en ese período. Se busca en la base de datos los mensajes correspondientes a las fechas seleccionadas y se muestran los datos de los mensajes en forma de listado en la interfaz web.</td></tr>
<tr><td>Referencias</td><td colspan="2">RF1, RF3, RF12, RF13</td></tr>
<tr><td>Precondición</td><td colspan="2">Usuario autenticado y autorizado para acceder a la funcionalidad “Listar Mensajes Enviados”.</td></tr>
<tr><td colspan="3">Flujo Normal de Eventos</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td colspan="2">1. El administrador accede al módulo “Mensajes WAP” y selecciona la opción “Listar Mensajes Enviados”.</td><td>2. El sistema busca en la base de datos todos los mensajes enviados y los muestra en un listado.</td></tr>
<tr><td colspan="2">3. El administrador decide buscar mensajes enviados durante un rango de fecha determinado.
Criterios:
▪ Fecha de inicio.
▪ Fecha de fin.</td><td>4. El sistema busca en la base de datos los mensajes que se han enviado en el rango de fechas seleccionado.</td></tr>
<tr><td colspan="2"></td><td>5. El sistema muestra en un listado los datos correspondientes a los mensajes:
▪ Identificador del mensaje.
▪ Hora y fecha de envío.
▪ Hora y fecha de entrega.
▪ Campos del mensaje.
▪ Respuesta del sistema.
▪ Estado (entregado/ no entregado).</td></tr>
<tr><td>Poscondición</td><td colspan="2">No aplica</td></tr>
</table>

Tabla #3. Descripción detallada del caso de uso: Listar mensajes enviados.

<table>
<tr><th colspan="2">Caso de uso</th></tr>
<tr><td>CU 2</td><td>Eliminar mensajes enviados.</td></tr>
<tr><td>Propósito</td><td>Eliminar los mensajes enviados, seleccionados por el administrador.</td></tr>
<tr><td colspan="2">Actor: Administrador</td></tr>
<tr><td colspan="2">Resumen: El caso de uso se inicia cuando el administrador accede a la funcionalidad “Eliminar Mensajes Enviados” contenida dentro del módulo “Mensajes WAP”. Selecciona mediante dos componentes de fecha un rango de días, para consultar los mensajes que se han enviado en el período deseado. El usuario marca los mensajes que desee eliminar y posteriormente ejecuta la acción “Eliminar”.</td></tr>
<tr><td>Referencias</td><td>RF2, RF3, RF12, RF13</td></tr>
<tr><td>Precondición</td><td>Usuario autenticado y autorizado para acceder a la funcionalidad “Eliminar mensajes enviados”.</td></tr>
<tr><th colspan="2">Flujo Normal de Eventos</th></tr>
<tr><td>Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td>1. El administrador accede al módulo “Mensajes WAP” y selecciona la opción “Eliminar Mensajes Enviados”.</td><td>2. El sistema muestra la vista correspondiente, la cual posee un formulario para la búsqueda de mensajes en un rango de fecha.</td></tr>
<tr><td>3. El administrador decide buscar mensajes enviados durante un rango de fecha determinado.
Criterios:
▪ Fecha de inicio.
▪ Fecha de fin.</td><td>4. El sistema busca en la base de datos todos los mensajes enviados en el rango de fechas seleccionado y los muestra en un listado.</td></tr>
<tr><td>5. El administrador selecciona los mensajes que desea eliminar.</td><td>6. El sistema elimina de la base de datos el mensaje seleccionado con sus respectivos datos:
▪ Identificador del mensaje.
▪ Hora y fecha de envío.
▪ Hora y fecha de entrega.</td></tr>
</table>

	▪ Campos del mensaje. ▪ Respuesta del sistema. ▪ Estado.
Poscondición	Se eliminaron mensajes enviados de la base de datos.

Tabla #4. Descripción detallada del caso de uso: Eliminar mensajes enviados.

Caso de uso	
CU 3	Buscar mensajes enviados.
Propósito	Buscar mensajes enviados en la base de datos.
Actor: Administrador	
Resumen: El caso de uso se inicia cuando se decide listar o eliminar mensajes enviados desde el módulo “Mensajes WAP” del Portal Web, ya que para ambas acciones se hace necesaria la búsqueda de mensajes.	
Referencias	RF1, RF2, RF3, RF11, RF12
Precondición	Usuario autenticado y autorizado para acceder al módulo “Mensajes WAP”.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El administrador selecciona la opción “Listar Mensajes Enviados” o “Eliminar Mensajes Enviados”.	2. El sistema realiza la búsqueda de los mensajes en la base de datos.
	3. El sistema devuelve hacia la clase controladora la lista de los mensajes buscados.
Poscondición	No aplica

Tabla #5. Descripción detallada del caso de uso: Buscar mensajes enviados.

Caso de uso	
CU 4	Iniciar servicios del SDSCenter.
Propósito	Establecer conexión entre el cliente y el servidor de socket para realizar el envío de los mensajes WAP PUSH e iniciar los servicios del SDSCenter.

Actor: Administrador	
Resumen: El caso de uso se inicia cuando el administrador decide iniciar el SDSCenter el cual contiene el servicio del servidor de socket y otros procesos, para lo cual debe ejecutar el jar que contiene la aplicación.	
Referencias	RF4, RF11, RF12
Precondición	No aplica
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El administrador ejecuta el jar que contiene al SDSCenter y a su vez al servidor de socket.	2. Se inician los servicios y se establece la conexión entre el cliente socket y el SDSCenter.
Poscondición	Se iniciaron los servicios del SDSCenter y se estableció la nueva conexión.

Tabla #6. Descripción detallada del caso de uso: Iniciar servicios del SDSCenter.

Caso de uso	
CU 5	Enviar mensaje.
Propósito	Enviar mensajes WAP PUSH desde el cliente de socket contenido en los servicios de Web Services o de RMI hacia el servidor de socket en el SDSCenter y luego hacia la infraestructura NEBULA.
Actor: Módulo de Envío de Mensajes WAP	
Resumen: El caso de uso se inicia cuando desde el Módulo de Envío de Mensajes WAP se crea un mensaje WAP PUSH y se envía hacia los servicios de Web Services o de RMI correspondientes. Una vez que llega el mensaje a uno de estos servicios (se especifica que puede ser cualquiera), el mismo es entregado al cliente de socket que se encuentra en ellos. Este cliente se encarga de enviarlo hacia el servidor de socket del SDSCenter mediante una conexión socket TCP. Una vez en el módulo SDSCenter el mensaje es registrado en la base de datos y luego enviado hacia NEBULA mediante el API N2AClient, que en caso de ocurrir algún fallo en el terminal o en la infraestructura, el mensaje es almacenado en una cola en la base de datos y a partir de este momento será reenviado una cantidad de veces predefinida.	
Referencias	RF4, RF5, RF6, RF7, RF8, RF9, RF10, RF11, RF12

Precondición	Se debe enviar un mensaje desde el Módulo de Envío de Mensajes WAP y deben estar iniciados los servicios del SDSCenter.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. Se envía un mensaje desde el "Módulo de Envío de Mensajes WAP".	2. Se captura el mensaje en el cliente de socket y se envía hacia el SDSCenter.
	3. El sistema envía el mensaje hacia NEBULA a través del API N2AClient.
Flujo Alterno 3ra Opción: No tuvo éxito la entrega del mensaje.	
	3ra1. El sistema almacena el mensaje en una cola y lo reenvía una cantidad de veces predefinida.
Poscondición	Se envió un mensaje WAP PUSH hacia la infraestructura.

Tabla #7. Descripción detallada del caso de uso: Enviar mensaje.

Caso de uso	
CU 6	Actualizar Cola de Mensajes.
Propósito	Mantener una cola actualizada con los mensajes que requieren ser reenviados.
Actor: Módulo de Envío de Mensajes WAP	
Resumen: El caso de uso se inicia cuando se envía un mensaje desde el Módulo de Envío de Mensajes WAP. Si el mismo no puede ser entregado después de su primer envío por problemas de conexión con la infraestructura NEBULA, o que el terminal esté fuera de servicio, entonces el sistema almacena el mensaje en la cola para ser reenviado una cantidad de veces predefinida. Se actualiza el campo donde se guarda la cantidad de envíos que ha tenido el mismo, si la cantidad de envíos iguala la cantidad de reenvíos predefinida o es entregado satisfactoriamente durante uno de los intentos de reenvío, el mensaje será eliminado de la cola.	
Referencias	RF5, RF6, RF7, RF8, RF11, RF12
Precondición	Mensaje no entregado por problemas con el terminal o la infraestructura.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**

1. Se envía un mensaje desde el “Módulo de Envío de Mensajes WAP”.	2. El sistema almacena el mensaje en la cola cuando no se pudo entregar satisfactoriamente después del primer envío.
	3. El sistema extrae el mensaje de la cola.
	4. El sistema reenvía el mensaje y actualiza el número de envíos.
Flujo Alterno 4ta Opción: Tuvo éxito la entrega del mensaje durante los intentos de reenvío.	
	4ta 1. El sistema elimina el mensaje de la cola.
Flujo Alterno 4ta Opción: No tuvo éxito la entrega del mensaje durante los intentos de reenvío.	
	4ta 2. El sistema elimina el mensaje de la cola
Poscondición	Cola de mensajes actualizada.

Tabla #8. Descripción detallada del caso de uso: Actualizar Cola de Mensajes.

Caso de uso	
CU 7	Eliminar mensajes de la cola.
Propósito	Eliminar mensajes que no requieren seguir almacenados en la cola.
Actor: Módulo de Envío de Mensajes WAP	
Resumen: El caso de uso se inicia cuando un mensaje ya no requiere seguir almacenado en la cola. Si durante los intentos de reenvío de un mensaje, la cantidad de envíos iguala a la cantidad de reenvíos predefinida o es entregado satisfactoriamente, el mensaje es eliminado de la cola.	
Referencias	RF5, RF6, RF7, RF11, RF12
Precondición	Mensaje que cumpla con los requisitos para ser eliminado.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
	1. El sistema reenvía el mensaje cada vez que, después de un intento de envío no pueda ser entregado. Esto lo hace una cantidad de veces predefinida. 2. El sistema elimina de la cola el mensaje siempre que

	ocurra una de las siguientes condiciones: ▪ Se supere la cantidad de veces que se reenvía el mensaje. ▪ El mensaje se logre enviar satisfactoriamente.
Poscondición	Se eliminó un mensaje de la cola.

Tabla #9. Descripción detallada del caso de uso: Eliminar mensajes de la cola.

Caso de uso	
CU 8	Registrar mensajes en cola.
Propósito	Registrar mensajes en la cola para reenviar.
Actor: Módulo de Envío de Mensajes WAP	
Resumen: El caso de uso se inicia cuando un mensaje después de su primer envío no pudo ser entregado satisfactoriamente por problemas con el terminal o la infraestructura. El mensaje es almacenado en la cola en la base de datos y se registra con cantidad de envíos en 0 para contar la cantidad de veces que el mismo será enviado.	
Referencias	RF5, RF6, RF8, RF12, RF13
Precondición	Mensaje no entregado después de su primer envío por problemas con el terminal o la infraestructura.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. Se envía un mensaje desde “Módulo de Envío de Mensajes WAP”.	2. El sistema trata de enviar el mensaje.
	3. El sistema recibe la respuesta del envío desde NEBULA que en este caso no es satisfactoria y registra el mensaje en la cola con sus datos correspondientes: ▪ Identificador ▪ Contenido del mensaje ▪ Grupo ▪ Identidad del suscriptor

	▪ Flota ▪ Red privada virtual ▪ Cantidad de envíos
Poscondición	Se adicionó un mensaje a la cola.

Tabla #10. Descripción detallada del caso de uso: Registrar mensajes en cola.

Caso de uso	
CU 9	Buscar mensajes en cola.
Propósito	Buscar mensajes en la cola para ser enviados o eliminados.
Actor: Administrador	
Resumen: El caso de uso se inicia cuando se ejecutan los servicios del SDSCenter, se levanta el proceso que busca sistemáticamente si hay mensajes en la cola para enviar. También cuando se va a eliminar un mensaje de la cola es necesario buscarlo antes.	
Referencias	RF5, RF6, RF7, RF9, RF12, RF13
Precondición	No aplica
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. El administrador inicia los servicios del SDSCenter.	2. El sistema levanta el proceso para buscar sistemáticamente si hay mensajes en la cola para enviar. 3. El sistema realiza la búsqueda de un mensaje en la cola una vez que necesite eliminarlo.
Poscondición	No Aplica

Tabla #11. Descripción detallada del caso de uso: Buscar mensajes en cola.

Caso de uso	
CU 10	Decodificar mensaje.
Propósito	Obtener el formato xml que poseía el mensaje antes de ser codificado para extraer los campos del mismo.
Actor: Módulo de Envío de Mensajes WAP	

Resumen: El caso de uso se inicia cuando se envía un mensaje WAP PUSH desde el Portal Web de SERWAP al SDSCenter. Independientemente de si el mensaje es entregado o no, el mismo es decodificado para mostrar los campos con los cuales fue creado y poder visualizarlos a la hora de listar los mensajes enviados.	
Referencias	RF5, RF6, RF10, RF11, RF12, RF13
Precondición	El Módulo de Envío de Mensajes WAP debe enviar un mensaje WAP PUSH al SDSCenter para ser decodificado.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. Se envía un mensaje desde el "Módulo de Envío de Mensajes WAP".	2. El sistema decodifica el mensaje mediante el WAPDecoder (decodificador que se encuentra en el SDSCenter).
	3. El sistema almacena los campos del mensaje decodificado en la base de datos.
Poscondición	Se obtuvo el mensaje decodificado para ser almacenado y visualizado.

Tabla #12. Descripción detallada del caso de uso: Decodificar mensaje.

Caso de uso	
CU 11	Registrar mensajes enviados.
Propósito	Registrar los mensajes enviados en la base de datos para poder visualizar los datos de envío de los mismos desde la aplicación web.
Actor: Módulo de Envío de Mensajes WAP	
Resumen: El caso de uso se inicia cuando se envía un mensaje WAP PUSH desde el Módulo de Envío de Mensajes al SDSCenter. Luego de terminar el proceso de envío de un mensaje independientemente de si se pudo entregar o no, se almacenan los datos de envío del mismo en la base de datos para poder ser listados en la aplicación web posteriormente.	
Referencias	RF1, RF3, RF5, RF6, RF11, RF12, RF13
Precondición	No aplica
Flujo Normal de Eventos	

Acción del actor	Respuesta del sistema
1. Se envía un mensaje desde el “Módulo de Envío de Mensajes WAP”.	2. El sistema recibe el mensaje y realiza el proceso de envío del mismo.
	3. El sistema almacena los datos del envío del mensaje en la base de datos para su posterior visualización.
Poscondición	Se registraron los datos de envío de un mensaje en la base de datos.

Tabla #13. Descripción detallada del caso de uso: Registrar mensajes enviados.

2.10 Conclusiones

En este segundo capítulo se detallaron y definieron los problemas existentes que dieron paso al desarrollo del nuevo sistema. Se presentaron conceptos y sus relaciones haciendo uso del modelo de dominio, se mostró la propuesta del sistema, así como los requerimientos del software los cuales se deben cumplir para lograr el resultado esperado. También se realizó una descripción detallada de los casos de uso, los cuales son representados en un diagrama de casos de uso del sistema.

CAPÍTULO 3: DISEÑO DEL SISTEMA

3.1 Introducción

El presente capítulo tiene como objetivo abordar la fase de Diseño. En la cual se confecciona un modelo de diseño que describe en forma de diagramas las clases a utilizar y los flujos de secuencia, así como un modelo de datos que representará las tablas de la base de datos. También se aborda sobre los patrones y la arquitectura que se aplicarán para dar un mejor funcionamiento al sistema a desarrollar.

3.2 Modelo de diseño

El modelo de diseño es un paso inicial, una primera aproximación conceptual, para una vez comprendidos los requisitos, aumentar el nivel de especificidad en aras de garantizar el correcto levantamiento de los requisitos funcionales y no funcionales, considerando además el entorno en el cual se realizará la futura implementación. [38]

3.2.1 Diagramas de paquetes

Los diagramas de paquetes permiten dividir el sistema, organizándolo en subsistemas y detallando sus relaciones. A continuación se muestran los diagramas de paquetes que representan los dos módulos por los que está compuesto el sistema SDSCenter.

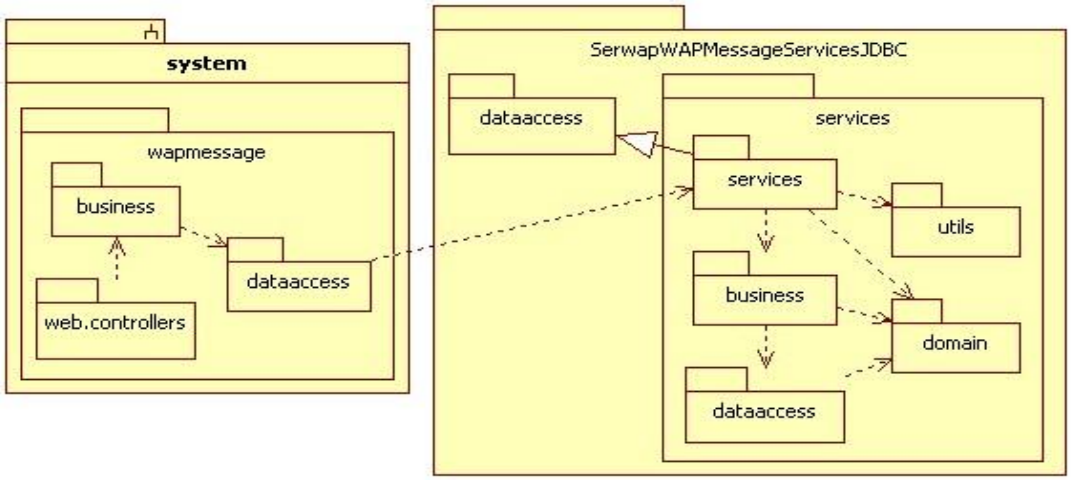

Figura #6. Diagrama de paquetes del Módulo Web.

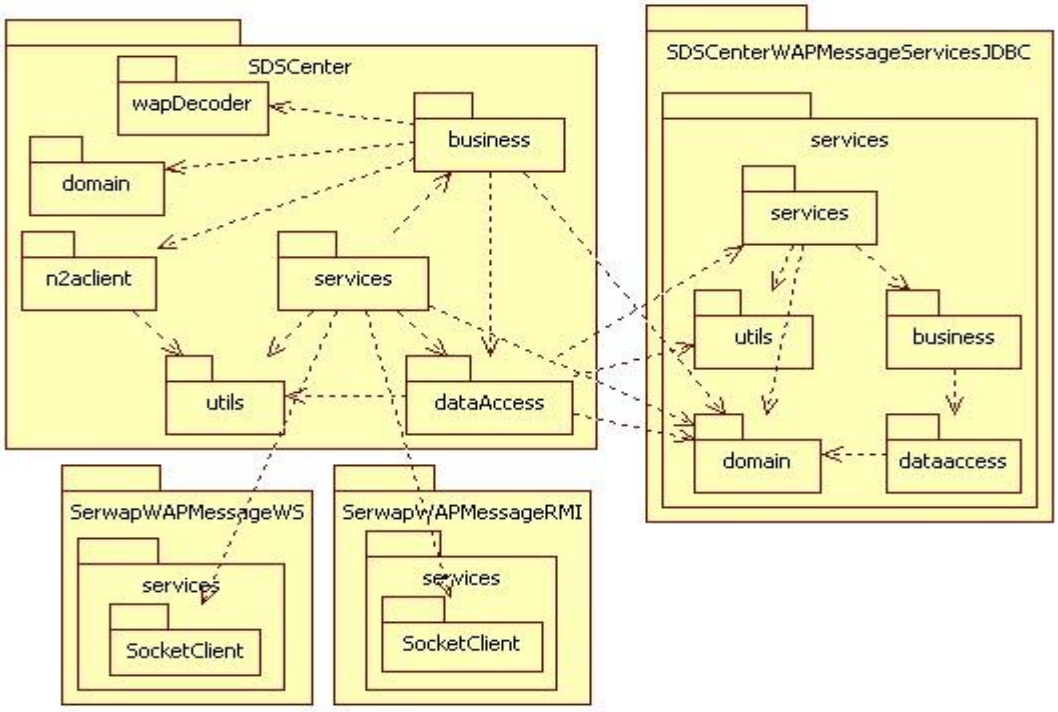

Figura #7. Diagrama de paquetes del Módulo SDSCenter.

Se pueden encontrar los diagramas de paquetes más especificados para cada módulo en:
Ver Anexo #1 y Anexo #2

3.2.2 Diagramas de clases del diseño

Los diagramas de clases del diseño permiten describir gráficamente las especificaciones de las clases del software. A continuación se muestran diagramas de clases correspondientes para cada módulo.

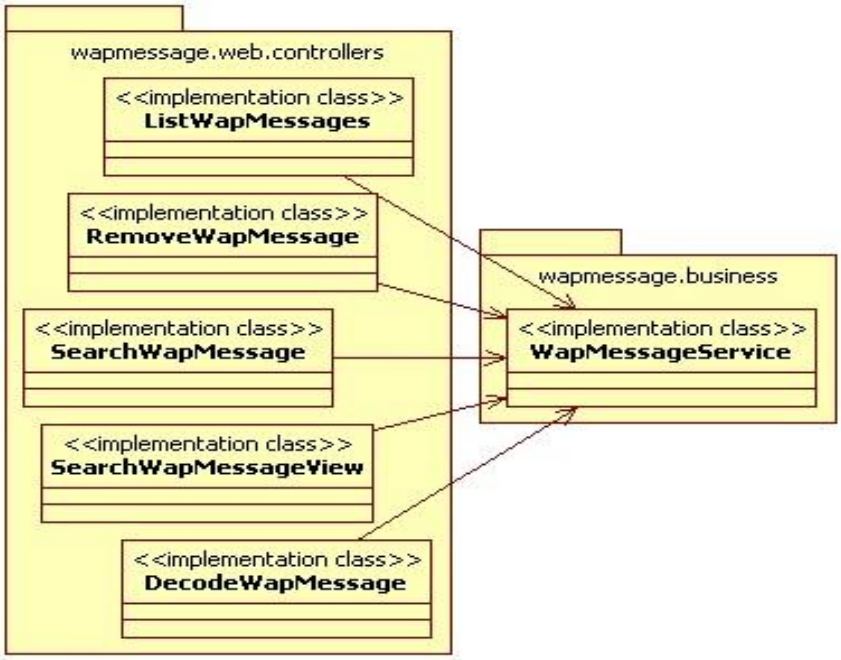

Figura #8. Diagrama de clases del paquete "web.controllers" de wapmessage del Módulo Web.

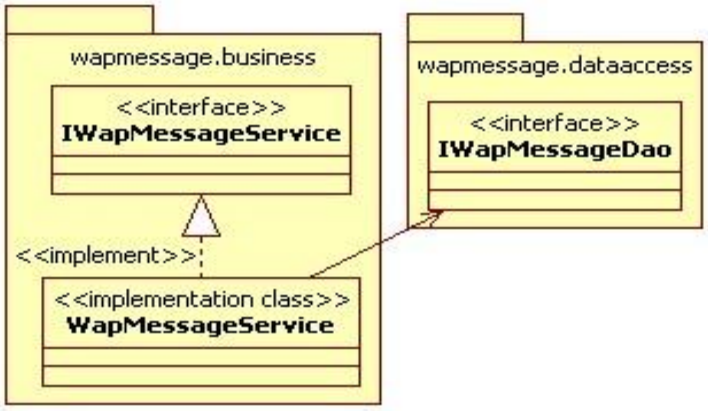

Figura #9. Diagrama de clases del paquete "business" de wapmessage del Módulo Web.

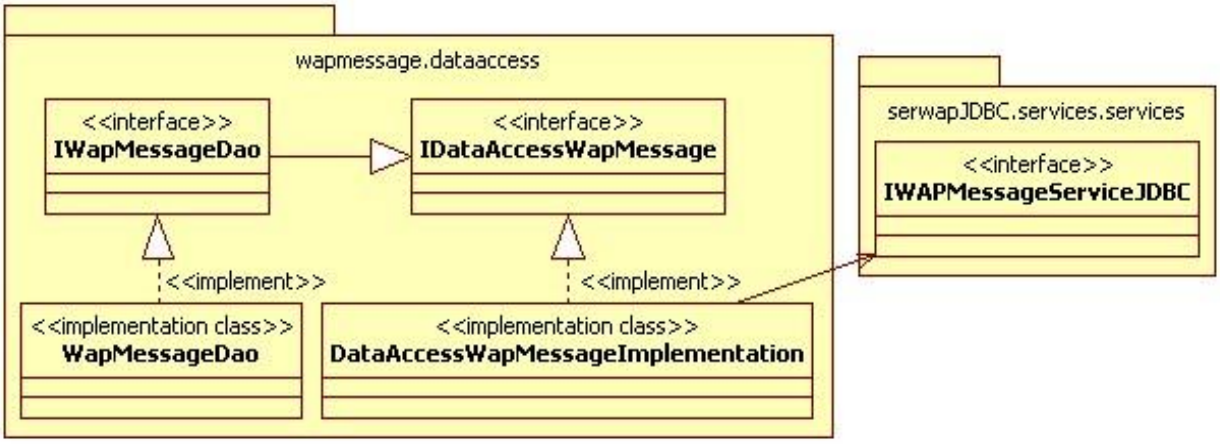

Figura #10. Diagrama de clases del paquete "dataaccess" de wapmessage del Módulo Web.

Los diagramas de clases por funcionalidades para el Módulo Web aparecen en:

Ver
Anexo #**3**. *Diagramas de clases del diseño Web.*

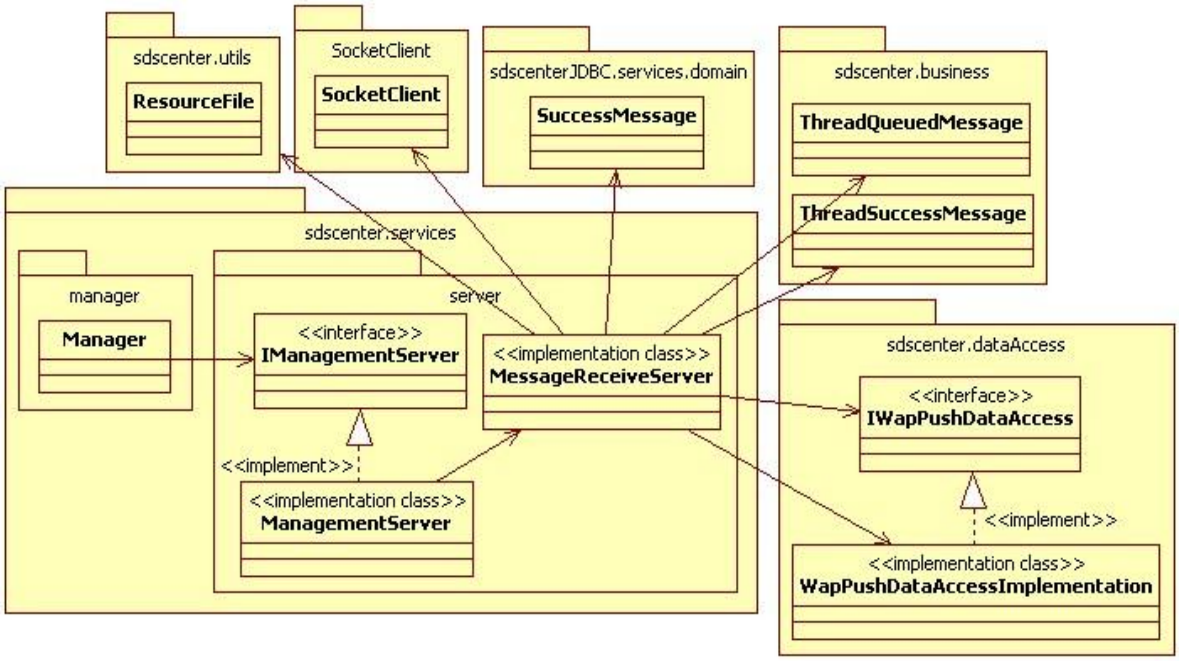

Figura #11. Diagrama de clases del paquete "services" del Módulo SDSCenter.

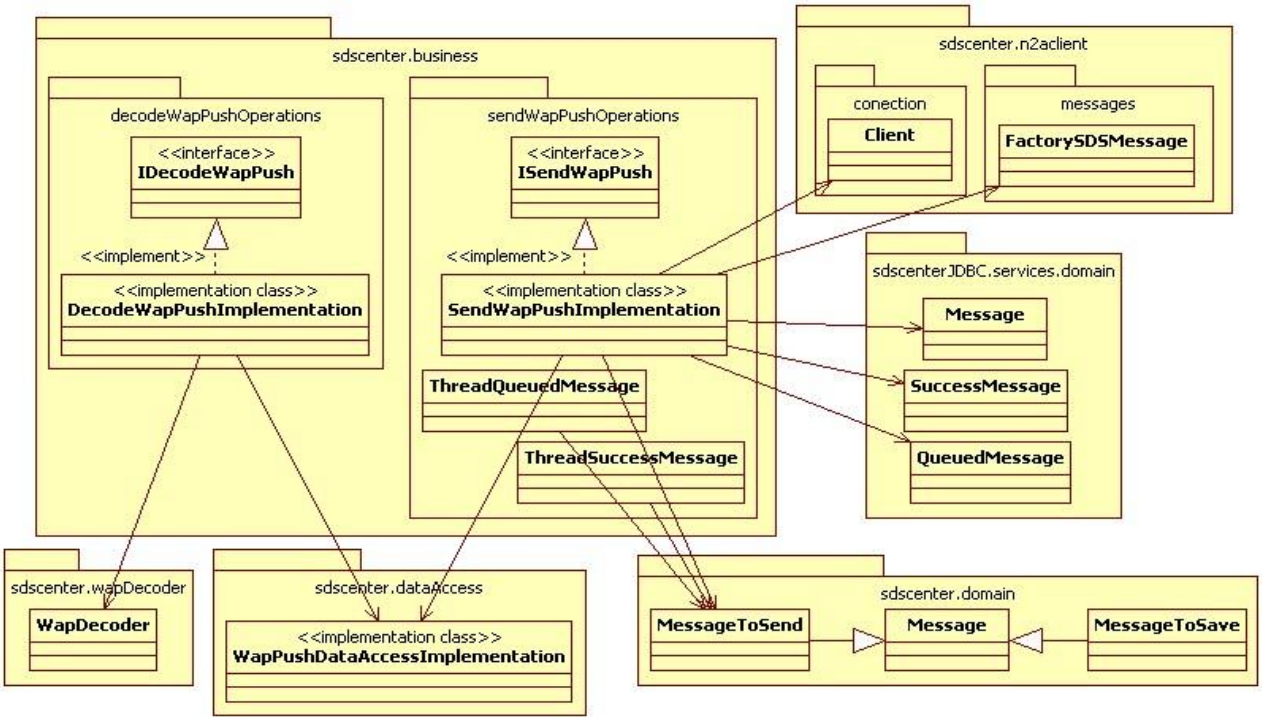

Figura #12. Diagrama de clases del paquete "business" del Módulo SDSCenter.

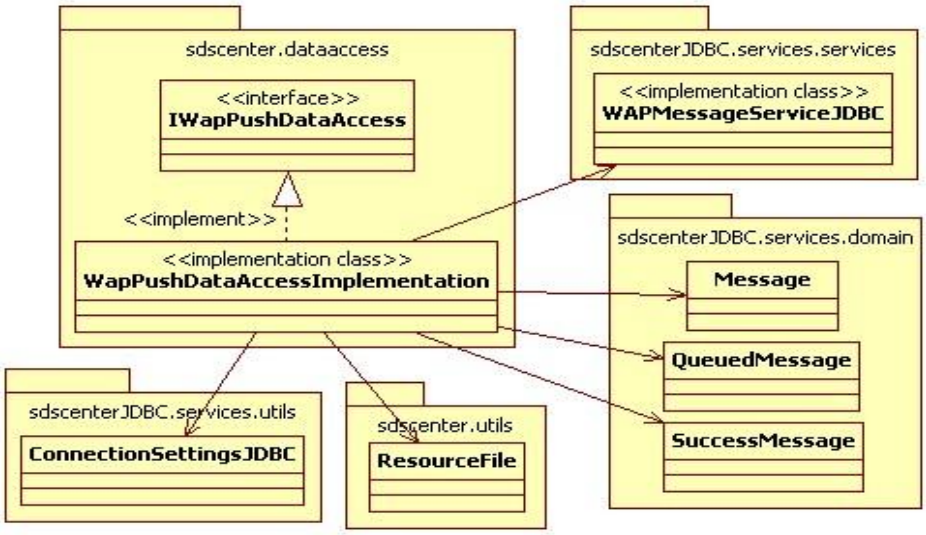

Figura #13. Diagrama de clases del paquete "dataaccess" del Módulo SDSCenter.

3.2.3 Diagramas de secuencia

Los diagramas de secuencia muestran la forma en que los objetos se comunican entre sí al transcurrir el tiempo a través de relaciones y mensajes enviados entre ellos. Son uno de los tipos de diagramas de interacción, los cuales son un término genérico que se aplica a varios diagramas que hacen hincapié en las interacciones entre objetos.

Los diagramas de secuencia para los casos de uso más significativos se muestran en:

Ver **Anexo #4** *Diagramas de secuencia.*

3.3 Diseño de la base de datos

El diseño de la base de datos es una de las tareas más importantes en la construcción de un sistema que utilice base de datos. El modelo de datos permite describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan estos elementos entre sí.

3.3.1 Modelo Entidad-Relación

El modelo entidad-relación permite esquematizar la estructura de datos de forma que se pueda entender mejor lo que se está diseñando. El modelo de datos del sistema SDSCenter describe tres tablas usadas para la gestión de los mensajes.

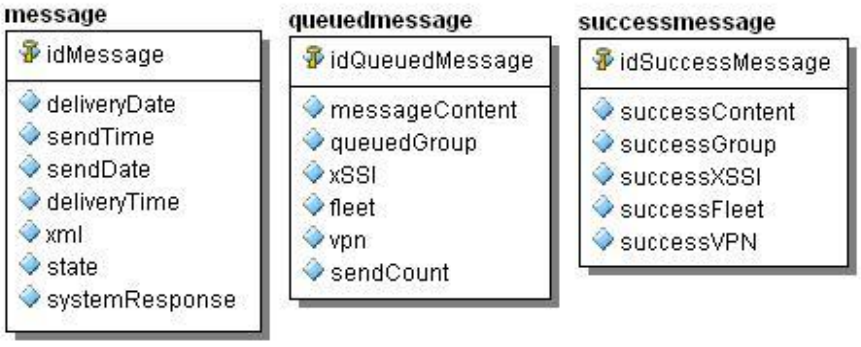

Figura #14. Modelo de Datos

3.4 Arquitectura del Sistema.

La arquitectura de software consiste en el diseño de componentes de una aplicación, generalmente utilizando patrones de arquitectura. Un diseño arquitectónico describe en general el cómo se construirá una aplicación de software. [39]

El sistema a desarrollar estará basado principalmente en el patrón de arquitectura **Cliente-Servidor**, el cual provee una arquitectura escalable, donde cada ordenador o proceso en la red es o bien cliente o bien servidor. Esta arquitectura se encuentra dentro de la clasificación de estilo de Llamada y Retorno. El cliente y el servidor generalmente están localizados en diferentes sistemas, sin embargo pueden encontrarse en el mismo sistema. El cliente es la entidad que hace la petición por un servicio. El servidor es la entidad que provee el servicio correspondiente a la petición. El servicio debe procurar el resultado, el cual es retornado al cliente. [39]

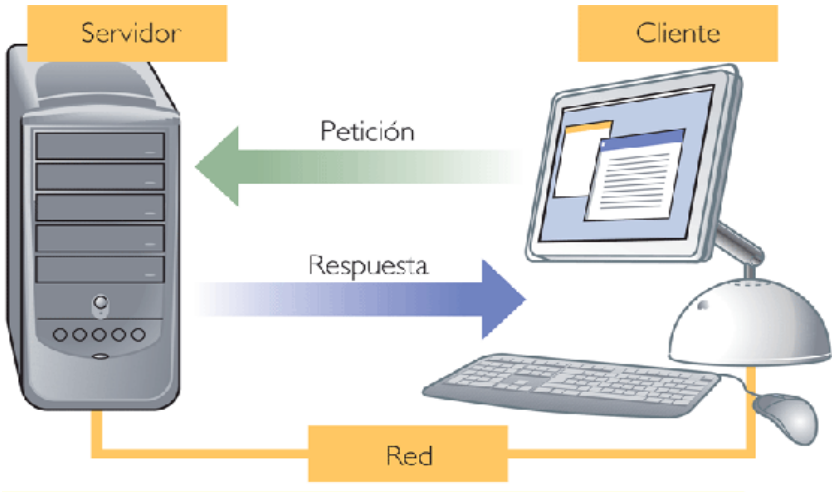

Figura #15. Arquitectura Cliente-Servidor.

Los clientes pueden ser clasificados como clientes ligeros o pesados. Los clientes pesados típicamente contienen, además de la lógica de presentación, gran parte de la lógica de negocio de la aplicación. Los clientes ligeros manejan usualmente sólo la lógica de presentación, permitiendo que futuros cambios de negocio en la aplicación no afecten al cliente. [39]

El uso de este patrón permitió dividir el sistema en general en diferentes partes que se comportan como cliente o servidor, o ambos a la vez. Una de estas es el cliente de socket ubicado en los servicios de RMI y Web Services de SERWAP, que se comporta como un cliente, ya que realiza el envío de mensajes hacia el SDSCenter donde se realizarán las operaciones correspondientes con los mismos. Y otra parte

que se comporta como un servidor, que es la que se encuentra siempre escuchando a que se realice un envío desde el cliente, al recibir el flujo el servidor realiza diferentes acciones con el mensaje tales como enviarlo hacia NEBULA, que en este caso, se comporta como un servidor ya que envía los mensajes hacia los terminales que son los clientes finales, y luego devuelve una respuesta al SDSCenter que se comporta como un cliente ante NEBULA.

3.5 Patrones de diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software, por lo que utilizarlos proporciona una serie de ventajas debido a que:

- Proponen una forma de reutilizar la experiencia de los desarrolladores, para ello clasifica y describe formas de solucionar problemas que ocurren frecuentemente en el desarrollo.
- Están basados en la recopilación del conocimiento de los expertos en el desarrollo de software.
- Brindan soluciones eficientes y satisfactoriamente probadas a determinados problemas. [26]

3.5.1 Patrones GRASP [40]

Los patrones GRASP (del inglés: General Responsibility Assignment Software Patterns), permiten describir los principios fundamentales de asignación de responsabilidades a objetos. Estos patrones son esenciales en el diseño eficaz de un software. Para el desarrollo del sistema se han utilizado algunos de ellos lo que permite fortalecer el diseño del software para obtener un producto de mayor calidad.

- **Experto:** Para el desarrollo de un sistema se pueden definir desde unas pocas hasta cientos de clases de software, y una aplicación tal vez requiera el cumplimiento de ciertas responsabilidades. Este patrón se usó para asignar responsabilidades a clases que cuentan con la información necesaria para cumplirlas. El uso de este patrón proporciona que el sistema sea más fácil de mantener y ampliar, y presenta la característica de poder reutilizar los componentes en futuras aplicaciones ya que indica que la creación de un objeto debe recaer sobre la clase que conoce todo lo referente para su realización.
- **Creador:** La creación de objetos es una de las actividades más frecuentes en un sistema orientado a objetos. Para el desarrollo de este sistema resulta muy conveniente el uso de este patrón ya que, para crear una instancia de una clase lo tiene que hacer la clase que tiene la información para ello aplicándolo generalmente en relaciones asociativas o de composición.

- **Bajo Acoplamiento:** Un bajo acoplamiento significa que una clase no depende de muchas clases. El uso de este patrón permitió que los componentes sean fáciles de reutilizar debido a la poca dependencia entre clases y que estas no se vean afectadas por cambios en otros componentes.
- **Controlador:** Es su función asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. Resultó muy útil el uso de este patrón al implementar un sistema separado por secciones o capas; separando las funciones, pudiendo invocar funcionalidades contenidas en capas más profundas como el acceso a datos.
- **Alta Cohesión:** Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable. En el sistema se agruparon clases por funcionalidades que son fácilmente reutilizables, ya que cada una realiza la labor que le corresponde.

3.5.2 Patrón Inversión de control: Inyección de dependencia

El patrón inyección de dependencia, es una técnica que como su nombre lo indica busca facilitar la resolución de dependencias entre objetos. Consiste en resolver las dependencias de cada clase (atributos) generando los objetos cuando se arranca la aplicación y luego inyectarlos en los demás objetos que los necesiten a través de métodos set o bien a través del constructor, pero estos objetos se instancian una vez, se guardan en una factoría y se comparten por todos los usuarios. [41]

Por lo que el uso de este patrón soportado por el framework Spring fue de gran utilidad ya que los objetos son inyectados en tiempo de ejecución a través de métodos set a las clases que posean dichas dependencias en la aplicación.

3.6 Conclusiones

En este capítulo se mostraron modelos del diseño, incluyéndose diagramas de clases para lograr una mejor representación de estas en el sistema. Se mostraron diagramas de secuencia, los cuales facilitaron la comprensión y descripción de la comunicación entre las clases a través de mensajes. Estos modelos sirvieron de ayuda para lograr una comprensión detallada de los requerimientos del software. Además se mostró el modelo de datos y se realizó un análisis de los patrones de arquitectura y diseño, contribuyendo a obtener la calidad requerida, y de esta forma dar paso a la implementación del sistema.

CAPÍTULO 4: IMPLEMENTACION Y PRUEBAS

4.1 Introducción

El presente capítulo tiene como objetivo abordar la fase de trabajo: Implementar en base a las funcionalidades, donde se describe cómo se implementan en términos de componentes los elementos del modelo de diseño. Se modelan los diagramas de componentes y de despliegue, dando una visión de cómo quedará desarrollada y distribuida la aplicación. Además se especifica el tipo de prueba a utilizar para encontrar y documentar los defectos que puedan afectar la calidad del software y validar que funcione como fue diseñado, cumpliendo con los requisitos planteados.

4.2 Modelo de implementación

El modelo de implementación especifica cómo los elementos del modelo de diseño, fundamentalmente las clases, se implementan en términos de componentes como ficheros de código fuente, ejecutables, librerías, entre otros. También describe cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje de programación utilizado, y cómo dependen los componentes unos de otros, así como su organización. [42]

4.2.1 Diagrama de componentes

Los diagramas de componentes muestran los componentes del software, sus interfaces y sus interrelaciones. Se representan como un grafo de componentes unidos por medio de relaciones de dependencia (compilación, ejecución). Se utilizan para modelar la vista estática de un sistema y muestra la organización y las dependencias lógicas entre un conjunto de componentes de software, sean éstos componentes de código fuente, librerías, binarios o ejecutables. [42]

Los diagramas de componentes correspondientes a la implementación del sistema SDSCenter se pueden observar en:

Ver **Anexo #5.** *Diagramas de componentes.*

4.2.2 Diagrama de despliegue

Un diagrama de despliegue muestra las relaciones físicas entre los componentes hardware y software en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes de software (procesos y objetos que se ejecutan en ellos). Se representa mediante un grafo

de nodos unidos por conexiones de comunicación. Un nodo es un objeto físico en tiempo de ejecución que representa un recurso computacional. [43]

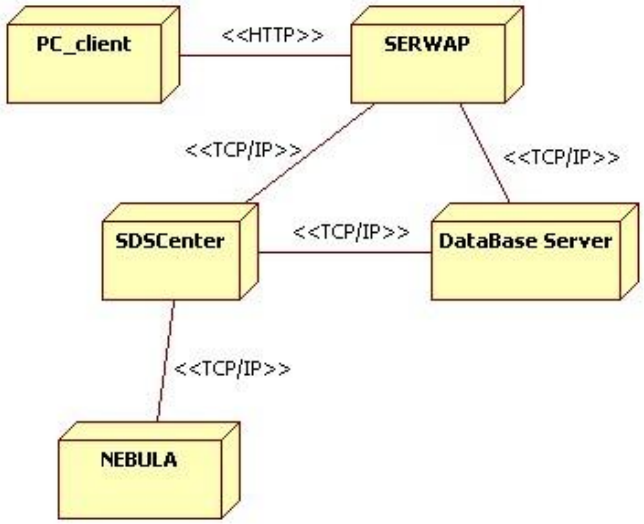

Figura #16. Diagrama de despliegue.

4.3 Modelo de pruebas

Las pruebas son un proceso de ejecución de un programa con la intención de descubrir errores, su principal objetivo es evaluar la calidad del producto a través de la validación del cumplimiento de los requerimientos, además de buscar y documentar los errores. [44]

4.3.1 Descripción de los casos de prueba

A continuación se describen un conjunto de pruebas cuya finalidad es comprobar el correcto funcionamiento del sistema y que cumpla con las funcionalidades requeridas, tales como: comprobar que se envíen los mensajes WAP PUSH a la infraestructura NEBULA mediante el API N2AClient, que una vez que es enviado un mensaje, quede registrada la información referente al mismo en la base de datos del sistema. Comprobar que se muestre mediante una interfaz web dicha información. Además verificar que se puedan visualizar los campos de los mensajes decodificados, y probar la comunicación entre los diferentes componentes que intervienen en el funcionamiento del sistema.

4.3.2 Pruebas por funcionalidades

Se aplicaron dos tipos de pruebas:

- **Pruebas de unidades de caja negra:** Las cuales se llevaron a cabo sobre la interfaz visual del módulo Web del SDSCenter. Se centraron en lo que se esperaba del módulo, es decir, intentar encontrar casos en que el módulo no se atiene a su especificación. Para ello se suministraron datos de entrada y a partir de estos se estudió la salida, basándose en lo que realiza la aplicación externamente y comparando los resultados con las especificación de los requisitos. [45][46]
- **Pruebas de integración de tipo funcional:** Estas pruebas se aplicaron para detectar posibles fallas en el funcionamiento del sistema cuando el mismo depende de otros servicios, como por ejemplo la comunicación con el Portal Web y con la infraestructura NEBULA. Así como para comprobar el funcionamiento de los distintos componentes del sistema en general y comprobar la correcta integración entre los mismos, realizándose desde los inicios del desarrollo incrementalmente hasta la versión final del sistema. [45]

Columna “Gravedad” de los casos de prueba.

CRI	Crítico	Consecuencias críticas, pueden provocar la interrupción de servicio del producto en el cliente final.
GRA	Grave	Consecuencias graves, pueden provocar mal funcionamiento de servicio del producto en el cliente final, sin interrupción de servicio.
MOD	Moderado	Consecuencias significativas, detectables por el usuario final sin afectar al funcionamiento normal.
LEV	Leve	Consecuencias menores, no hay repercusión sensible en el funcionamiento normal del producto.

Tabla #14. Describe los valores que toma la columna Gravedad en el caso de prueba.

Casos de pruebas

4.3.2.1 Listar mensajes WAP PUSH
TIPO DE PRUEBA
Prueba de unidad de caja negra.
OBJETIVO
Verificar que se muestren correctamente los datos de los mensajes.
CONDICIONES PREVIAS
✓ El usuario debe estar autenticado para acceder al Portal Web de SERWAP.

✓ Los mensajes a mostrar deben ser enviados y registrados previamente.			
PROCEDIMIENTO			
Acción		**Resultado esperado**	**Gravedad**
1.	Seleccionar la opción “Listar Mensajes Enviados”.	Se muestran los campos “Fecha de Inicio”, “Fecha de Fin”, el botón “Buscar” y una lista de los mensajes enviados que se encuentran registrados en la base de datos.	CRI
2.	Pulsar el botón “Buscar” sin introducir el criterio de búsqueda de las fechas.	Se muestra el mensaje de error: “Debe llenar al menos un campo para realizar la búsqueda.”	MOD
3.	- Introducir en el campo “Fecha de Inicio” caracteres extraños (por ejemplo: #@*/), o letras. - Pulsar el botón “Buscar”.	Se muestra el mensaje: “Debe llenar los campos correctamente”.	MOD
4.	- Introducir en el campo “Fecha de Fin” caracteres extraños (por ejemplo: $&*/), o letras. - Pulsar el botón “Buscar”.	Se muestra el mensaje: “Debe llenar los campos correctamente”.	MOD
5.	- Introducir un criterio de búsqueda por el que no se encuentre ningún resultado. - Pulsar el botón “Buscar”.	Se muestra el mensaje: “No se encontraron resultados acorde con el criterio de búsqueda.”	MOD
6.	- Introducir correctamente el criterio de búsqueda. - Pulsar el botón “Buscar”.	Se muestra una lista de los mensajes enviados en el rango de fechas introducido con todos los datos del mensaje.	CRI
Comentarios			
- Los campos de las fechas tanto de inicio como de fin se deben llenar utilizando el componente de fecha que aparece para seleccionar el día, el mes y el año. - Si ocurre algún error se registra en el fichero “SerwapWEB_Error.log” el error ocurrido. - En el fichero “SerwapWEB_Info.log” se registran los eventos ocurridos en este tipo de acción.			

Tabla #15. Caso de prueba #1

4.3.2.2 Eliminar mensajes WAP PUSH			
TIPO DE PRUEBA			
Prueba de unidad de caja negra.			
OBJETIVO			
Verificar que se eliminen correctamente los datos de los mensajes.			
CONDICIONES PREVIAS			
✓ El usuario debe estar autenticado para acceder al Portal Web de SERWAP. ✓ Los mensajes a eliminar deben ser enviados y registrados previamente.			
PROCEDIMIENTO			
Acción		**Resultado esperado**	**Gravedad**
1.	Seleccionar la opción “Eliminar Mensajes Enviados”.	- Se muestran los campos “Fecha de Inicio”, “Fecha de Fin” y el botón “Buscar” - Además se muestra el botón “Eliminar” (que se encuentra desactivado) y el botón “Cancelar”.	CRI
2.	Pulsar el botón “Buscar” sin introducir el criterio de búsqueda de las fechas.	Se muestra el mensaje de error: “Debe llenar al menos un campo para realizar la búsqueda.”	MOD
3.	Pulsar el botón “Cancelar” solamente.	Sale de la vista correspondiente a la opción “Eliminar Mensajes Enviados” y regresa a la vista en la que se encontraba la aplicación anteriormente.	MOD
4.	- Introducir en el campo “Fecha de Inicio” caracteres extraños (por ejemplo: *>%/), o letras. - Pulsar el botón “Buscar”.	Se muestra el mensaje: “Debe llenar los campos correctamente”.	MOD
5.	- Introducir en el campo “Fecha de Fin” caracteres extraños (por ejemplo: <|+%/), o letras. - Pulsar el botón “Buscar”.	Se muestra el mensaje: “Debe llenar los campos correctamente”.	MOD
6.	- Introducir un criterio de búsqueda por el que no se encuentre ningún resultado.	- Se muestra el mensaje: “No se encontraron resultados acorde con el criterio de búsqueda.”	MOD

	- Pulsar el botón “Buscar”.		
7.	- Introducir correctamente el criterio de búsqueda. - Pulsar el botón “Buscar”.	- Se muestra una lista de los mensajes enviados en el rango de fechas introducido con todos los datos del mensaje. - Delante de cada mensaje mostrado aparece un checkbox que permite marcar el mensaje que se desea eliminar, y además un checkbox en el menú superior de la lista, que permite marcar todos los mensajes de una vez. - Se activa el botón “Eliminar”.	CRI
8.	- Marcar el mensaje que se desea eliminar. - Pulsar el botón “Eliminar”.	Aparece un mensaje de confirmación preguntando: “¿Está seguro que desea eliminar?” y los botones “Aceptar” y “Cancelar”.	MOD
9.	Pulsar el botón “Cancelar” del mensaje.	Desaparece el mensaje de confirmación.	MOD
10.	Pulsar el botón “Aceptar” del mensaje.	- Se elimina de la lista el mensaje seleccionado. - Se muestra un mensaje de confirmación: “La operación se realizó correctamente”. - Regresa a la vista principal correspondiente a la opción “Eliminar Mensajes Enviados”.	CRI
Comentarios			
- Los campos de las fechas tanto de inicio como de fin se deben llenar utilizando el componente de fecha que aparece para seleccionar el día, el mes y el año. - Si ocurre algún error se registra en el fichero “SerwapWEB_Error.log” el error ocurrido. - En el fichero “SerwapWEB_Info.log” se registran los eventos ocurridos en este tipo de acción.			

Tabla #16. Caso de prueba #2

4.3.2.3 Decodificar mensajes WAP PUSH
TIPO DE PRUEBA
Prueba de caja negra a nivel de Interfaz Visual.
OBJETIVO

<table>
<tr><td colspan="4">Comprobar que se decodifiquen los mensajes correctamente.</td></tr>
<tr><td colspan="4">CONDICIONES PREVIAS</td></tr>
<tr><td colspan="4">✓ El usuario debe estar autenticado previamente en el Portal Web de SERWAP.
✓ Los mensajes a decodificar deben ser enviados previamente.</td></tr>
<tr><td colspan="4">PROCEDIMIENTO I</td></tr>
<tr><td colspan="2">Acción</td><td>Resultado esperado</td><td>Gravedad</td></tr>
<tr><td>1.</td><td>Seleccionar la opción “Listar Mensajes Enviados”.</td><td>Se muestran los campos “Fecha de Inicio”, “Fecha de Fin”, el botón “Buscar” y una lista de los mensajes enviados que se encuentran registrados en la base de datos.</td><td>CRI</td></tr>
<tr><td>2.</td><td>- Buscar en la lista mostrada la columna “Campos del mensaje”.
- Presionar el vínculo “Ver” del mensaje que se desee ver su información.</td><td>- Se muestra una tabla con los campos del mensaje seleccionado:
• Si es un mensaje Service Indication:
Se muestra el tipo de mensaje, el identificador, el texto del mensaje, la dirección URL, la fecha de envío y de entrega, la hora de envío y de entrega y el tipo de señal.
• Si es un mensaje Service Loading:
Se muestra el tipo de mensaje, la dirección URL y el tipo de señal.
• Si es un mensaje Cache Operation:
Se muestra el tipo de mensaje, el campo “Invalidar Objeto” y el campo “Invalidar Servicio”.
- Además de la tabla se muestra un botón “Atrás” para volver a la lista de todos los mensajes enviados.</td><td>CRI</td></tr>
<tr><td colspan="4">PROCEDIMIENTO II</td></tr>
<tr><td>1.</td><td>Seleccionar la opción “Listar Mensajes Enviados”.</td><td>Se muestran los campos “Fecha de Inicio”, “Fecha de Fin”, el botón “Buscar” y una lista de los mensajes enviados que se encuentran registrados en la base de datos.</td><td>CRI</td></tr>
<tr><td>2.</td><td>- Introducir correctamente el criterio de búsqueda.
- Pulsar el botón “Buscar”.</td><td>- Se muestra una lista de los mensajes enviados en el rango de fechas introducido con todos los datos del mensaje.</td><td>CRI</td></tr>
</table>

3.	- Buscar en la lista mostrada la columna “Campos del mensaje”. - Presionar el vínculo “Ver” del mensaje que se desee ver su información.	- Se muestra una tabla con los campos del mensaje seleccionado: • Si es un mensaje Service Indication: Se muestra el tipo de mensaje, el identificador, el texto del mensaje, la dirección URL, la fecha de envío y de entrega, la hora de envío y de entrega y el tipo de señal. • Si es un mensaje Service Loading: Se muestra el tipo de mensaje, la dirección URL y el tipo de señal. • Si es un mensaje Cache Operation: Se muestra el tipo de mensaje, el campo “Invalidar Objeto” y el campo “Invalidar Servicio”. - Además de la tabla se muestra un botón “Atrás” para volver a la lista de todos los mensajes enviados.	CRI
PROCEDIMIENTO III			
1.	Seleccionar la opción “Eliminar Mensajes Enviados”.	- Se muestran los campos “Fecha de Inicio”, “Fecha de Fin” y el botón “Buscar” - Además se muestra el botón “Eliminar” (que se encuentra desactivado) y el botón “Cancelar”.	CRI
2.	- Introducir correctamente el criterio de búsqueda. - Pulsar el botón “Buscar”.	- Se muestra una lista de los mensajes enviados en el rango de fechas introducido con todos los datos del mensaje.	CRI
3.	- Buscar en la lista mostrada la columna “Campos del mensaje”. - Presionar el vínculo “Ver” del mensaje que se desee ver su información.	- Se muestra una tabla con los campos del mensaje seleccionado: • Si es un mensaje Service Indication: Se muestra el tipo de mensaje, el identificador, el texto del mensaje, la dirección URL, la fecha de envío y de entrega, la hora de envío y de entrega y el tipo de señal. • Si es un mensaje Service Loading: Se muestra el tipo de mensaje, la dirección URL y el tipo de señal.	CRI

		• Si es un mensaje Cache Operation: Se muestra el tipo de mensaje, el campo "Invalidar Objeto" y el campo "Invalidar Servicio". - Además de la tabla se muestra un botón "Atrás" para volver a la página principal de la opción "Eliminar Mensajes Enviados".	
Comentarios			
- Si ocurre algún error se registra en el fichero "SerwapWEB_Error.log" el error ocurrido. - En el fichero "SerwapWEB_Info.log" se registran los eventos ocurridos en este tipo de acción.			

Tabla #17. Caso de prueba #3

4.3.2.4 Enviar mensajes WAP PUSH			
TIPO DE PRUEBA			
Prueba de Integración de tipo funcional: entre el Portal Web de SERWAP y el SDSCenter.			
OBJETIVO			
Verificar que se envíen correctamente los mensajes WAP PUSH desde el Portal Web hacia el SDSCenter.			
CONDICIONES PREVIAS			
✓ Se debe enviar correctamente el mensaje WAP PUSH desde el Portal Web de SERWAP.			
PROCEDIMIENTO			
Acción		**Resultado esperado**	**Gravedad**
1.	Enviar un mensaje WAP PUSH desde el Portal Web.	-Se recibe el mensaje enviado a través del cliente de socket ubicado en los servicios de Web Services o de RMI.	CRI
2.		-Se registra en un fichero log en el SDSCenter el contenido del mensaje recibido por el servidor de socket.	LEV
3.		-El mensaje es almacenado en una tabla temporal en la base de datos.	CRI
4.		-El servidor de socket cierra la conexión y vuelve a estar disponible para recibir nuevos mensajes.	CRI

5.		-El SDSCenter busca sistemáticamente en la base de datos si hay mensajes nuevos para enviar.	CRI
6.		-Se extraen de la base de datos los mensajes para enviar y se realiza el envío de los mismos a través del API N2AClient.	CRI
Comentarios			
- Si ocurre algún error se registra en el fichero "Error.log". - En el fichero "Info.log" se registran los eventos ocurridos durante el proceso.			

Tabla #18. Caso de prueba #4

4.3.2.5 Enviar mensajes WAP PUSH			
TIPO DE PRUEBA			
Prueba de Integración de tipo funcional: entre el SDSCenter y NEBULA.			
OBJETIVO			
Verificar que se envíen correctamente los mensajes WAP PUSH desde el SDSCenter hacia NEBULA.			
CONDICIONES PREVIAS			
✓ Se debe enviar correctamente el mensaje WAP PUSH desde el Portal Web de SERWAP.			
PROCEDIMIENTO			
Acción		**Resultado esperado**	**Gravedad**
1.	Enviar un mensaje WAP PUSH desde el SDSCenter hacia NEBULA.	-El SDSCenter le entrega al N2AClient el mensaje a enviar.	CRI
2.		-El N2AClient se conecta a NEBULA y envía el mensaje WAP PUSH.	CRI
3.		-Se obtiene la respuesta del N2AClient proveniente NEBULA que indica lo sucedido al enviar el mensaje al terminal.	CRI
4.		-El sistema registra en un fichero log la respuesta del N2AClient.	LEV
5.		-El sistema registra los datos del mensaje enviado en una base de datos.	GRA
6.		-Se registra la acción en un fichero log.	MOD

Comentarios
- Si ocurre algún error se registra en el fichero “Error.log”. - En el fichero “Info.log” se registran los eventos ocurridos durante el proceso.

Tabla #19. Caso de prueba #5

Resultados de las pruebas:

Se obtuvieron resultados satisfactorios durante la aplicación de las pruebas. El sistema SDSCenter en general realiza correctamente las funciones correspondientes, cumpliendo satisfactoriamente con los requerimientos funcionales propuestos.

4.4 Conclusiones

En este capítulo se representaron los elementos necesarios para la comprensión de cómo el módulo SDSCenter fue implementado, tanto las organizaciones y las dependencias lógicas entre componentes a través del diagrama de componentes, como la distribución del sistema en nodos mediante el diagrama de despliegue. Además de algunos ejemplos de las pruebas realizadas para verificar que las funcionalidades trabajaran correctamente. Por lo que queda implementado y probado el módulo SDSCenter para dar solución al problema planteado al inicio del trabajo.

CONCLUSIONES

En el presente trabajo de diploma se presentó la investigación y posterior proceso de desarrollo del sistema SDSCenter, el cual funciona como intermediario para el envío de mensajes entre SERWAP y la infraestructura NEBULA mediante el API N2AClient.

Permitiendo mantener una cola de mensajes para poder reenviarlos si no pueden ser entregados inmediatamente por problemas con el terminal o la infraestructura. A la vez que brinda la posibilidad de consultar mediante una interfaz web la información referente a los mensajes enviados, para lo cual se permite la decodificación de los mismos convirtiendo su contenido hexadecimal a texto plano.

Para la realización de los procesos descritos anteriormente el sistema es capaz de almacenar los mensajes antes y después de enviados.

Además se logró establecer comunicación con los servicios de Web Services o RMI del sistema SERWAP a través de los cuales se envían los mensajes WAP PUSH del Portal Web, para lo que se utilizó una conexión socket la cual permite obtener los mensajes en el SDSCenter para realizar su proceso de envío hacia la infraestructura.

De esta forma se le da cumplimiento al objetivo general de la investigación presentada. Para la construcción y desarrollo del sistema se utilizaron las herramientas y tecnologías seleccionadas como son: lenguaje de programación Java, el framework Spring, para el acceso a datos el gestor de base de datos MySQL 5.0, la herramienta de modelado utilizada fue el StarUML y además se desarrolló el sistema en base a la metodología de desarrollo FDD que permitió desarrollar un software de alta calidad y ayudó a controlar el desarrollo del software a lo largo de todo el proceso.

Por lo anteriormente planteado se concluye que los objetivos propuestos para el presente proyecto han sido cumplidos satisfactoriamente.

RECOMENDACIONES

Como resultado del presente trabajo se desarrolló el sistema SDSCenter que funciona adjunto a un sistema informático. Dicho módulo aunque resuelve la problemática planteada inicialmente, puede ser mejorado y refinado con el objetivo de lograr un incremento en su calidad y alcance.

Para lograr una mayor escalabilidad en futuras versiones del producto, se recomienda extender el mismo, adicionándole funcionalidades para permitir la gestión del envío de todos los mensajes SDS que se envíen en la red, ya que está implementado solo para procesar mensajes de tipo WAP PUSH que se envían desde el Portal Web de SERWAP.

Además se recomienda que el producto se pueda independizar totalmente del proyecto SERWAP, incluyéndole una interfaz de comunicación genérica capaz de interactuar con otros sistemas, que así tendrían la posibilidad de utilizar las funcionalidades del SDSCenter para su beneficio.

BIBLIOGRAFÍA

1. Departamento de Tecnologías de la Información y las Comunicaciones (UPCT). Departamento TIC - Universidad Politécnica de Cartagena. “Un Poco de Historia de las Telecomunicaciones”. [En línea] [Citado el: 6 de Noviembre del 2010]
   Disponible en: http://www.tic.upct.es/index.php?option=com_content&task=view&id=36&Itemid=64.

2. Unión Radiotelegráfica Internacional. FACULTAD TELEMATICA. [En línea] [Citado el: 10 de Noviembre del 2010]
   Disponible en:http://docente.ucol.mx/al023423/public_html/Web/Index/Tareas/Antecedentes.htm.

3. Huidobro Moya, José Manuel. Redes y Servicios de Telecomunicaciones. Madrid, España. Paraninfo S.A. 2001.

4. SMS Center [En línea] [Citado el: 10 de Noviembre del 2010]
   Disponible en: http://www.smscenter.com.

5. Tecnotree. Tecnomen_Leaflet_TETRA_medres_screen_RGB.pdf [En línea] [Citado el: 13 de Noviembre del 2010]
   Disponible en: http://www.tecnotree.com/Solutions.

6. ATS [En línea] [Citado el: 13 de Noviembre del 2010]
   Disponible en: http://www.ats-connection.com.

7. Nokia [En línea] [Citado el: 13 de Noviembre del 2010]
   Disponible en: http://www.nokia.es/productos.

8. Motorola [En línea] [Citado el: 14 de Noviembre del 2010]
   Disponible en: http://www.motorola.com/Consumers/ES-ES/Home.

9. ITespresso. Redes TETRA, más allá de la seguridad y servicios de emergencia [En línea] [Citado el: 20 de Noviembre del 2010]
   Disponible en: http://www.itespresso.es/redes-tetra-mas-alla-de-la-seguridad-y-servicios-de-emergecia-37213. html.

10. Empresa TELTRONIC. Una realidad en comunicaciones profesionales. 2008.

11. IDG COMMUNICATIONS. TETRA apuesta por una comunicación crítica y segura. [En línea] [Citado el: 23 de Noviembre del 2010]
    Disponible en: http://www.idg.es/computerworld/TETRA-apuesta-por-una-comunicacion-critica-y-segur/ seccion-mob/artículo-184625.

12. Teltronic.Teltronic.pdf [En línea] [Citado el: 23 de Noviembre del 2010]
    Disponible en: http://www.teltronic.es.

13. Teltronic [En línea][Citado el: 24 de Noviembre del 2010]
Disponible en: http://www.teltronic.es.

14. Joan Martínez Herrera, David Rodríguez Rodríguez, Yeilin Pérez Martínez. SERWAP para redes TETRA. API para la codificación de mensajes WAP.pdf.

15. Alarcos. Ingeniería de Software 1. Tema04.pdf [En línea] [Citado el: 27 de Noviembre del 2010]
Disponible en: http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema04.pdf.

16. Jorge H. Canós, Patricio Letelier, María del Carmen Panadés. Metodologías ágiles de desarrollo de software. Universidad Politécnica de Valencia. 2004.

17. Coad P., Lefebvre E., De Lucas. Metodologías de desarrollo (FDD). Java Modeling In Color With UML: Enterprise Components and Process. Prentice Hall, 1999.

18. StarUML. The Open Source UML/MDA Platform. [En línea] [Citado el: 27 de Noviembre del 2010.]
Disponible en: http://staruml.sourceforge.net.

19. SourceForge. Summary StarUml [En línea] [Citado el: 28 de Noviembre del 2010]
Disponible en: http://staruml.sourceforge.net/about.php.

20. SourceForge. Features StarUml [En línea] [Citado el: 29 de Noviembre del 2010]
Disponible en: http://staruml.sourceforge.net/about-2.php.

21. Spring Framework. [En línea] [Citado el: 29 de Noviembre del 2010.]
Disponible en: http://static.springframework.org/docs/Spring-MVC-step-by-step.

22. Seam City. [En línea] [Citado el: 1 de Diciembre del 2010.]
Disponible en: http://seamcity.madeinxpain.com/archives/comparativa-spring.

23. Universidad de Salamanca. Guía de iniciación al lenguaje Java.pdf [En línea] [Citado el: 4 de Diciembre del 2010]
Disponible en: http://zarza.usal.es/~fgarcia/doc/tuto2/java20.pdf.

24. Ciberaula [En línea] [Citado el: 6 de Diciembre del 2010]
Disponible en: http://java.ciberaula.com/articulo/que_es_java.

25. Ajaxman. Pensando en java: Qué es J2SE, J2EE, J2ME y Java card.[En línea] [Citado el: 7 de Diciembre de 2010]
Disponible en: http://www.ajaxman.net/657/javapensando-en-java-que-es-j2se-j2ee-j2me-y-java-card.

26. Craig Larman. UML y patrones. Prentice Hall. 2002.

27. Manzana mecánica. [En línea] [Citado el: 11 de Diciembre del 2010]
Disponible en: http://www.manzanamecanica.org/2008/07/eclipse_ganymede.html.

28. Agapea. [En línea] [Citado el: 15 de Diciembre del 2010]
Disponible en: http://www.agapea.com/libros/Tomcat-6-0-La-guia-definitiva-isbn-8441524319-i.htm.

29. MySQL. Why MySQL? [En línea] [Citado el: 10 de Enero del 2011]
Disponible en: http://www.mysql.com/why-mysql.

30. Tigris. Subversion [En línea] [Citado el: 10 de Enero del 2011]
Disponible en: http://subversion.tigris.org.

31. Facultad Experimental de Ciencias y Tecnología de Carabobo. UML. Lenguaje de Modelado.pdf [En línea] [Citado el: 11 de Enero del 2011]
Disponible en:http://alfa.facyt.uc.edu.ve/computacion/pensum/cs0347/download/exposiciones2006-2007/ presentacion%20UML.pdf.

32. UIT (Unión Internacional de Telecomunicaciones). Informe para el desarrollo mundial de las telecomunicaciones. Indicadores de acceso para la sociedad de información. Ginebra, 2003.

33. IEEE (del inglés: Institute of Electrical and Electronics Engineers). IEEE Standard Glossary of Software Engineering Terminology .[En línea] [Citado el: 11 de Febrero del 2011]
Disponible en: http://standards.ieee.org/findstds/standard.

34. PRESSMAN, Roger S. “Ingeniería de Software. Un enfoque Práctico” 6ta edición.2007.

35. SOMMERVILLE, I. “Ingeniería del Software” 7ma Edición. 2005.

36. Empresa Teltronic. Documento Especificación de Diseño Funcional para plataforma SERWAP. Código SERWAP FDS01. Versión 02. Revisión 03. Abril de 2009. Página 15-16.

37. “SERWAP. Mecanismo de aprovisionamiento continuo mediante el protocolo WAP”. Presentado en la XV Convención Científica de Ingeniería y Arquitectura. ISBN: 978-959-261-317-1.

38. James Rumbaugh, Ivar Jacobson y Grady Booch. El lenguaje unificado de modelado. Manual de referencia. 2007

39. Pimentel González, Luis Alberto, Pérez Rivero, Iósev y Haro Pérez, Madelín. ArBaWeb*:* ARQUITECTURA BASE SOBRE LA WEB*.* 2007.

40. Gamma, J.M.V.R.H.R.J.E. Design Patterns*: Elements of Reusable Object-Oriented Software.* 1995.

41. Springhispano.org. Contenedores de inversión de control y el patrón inyección de dependencia. [En línea] [Citado el: 16 de Marzo del 2011] Disponible en: http://www.springhispano.org/?q=node/46

42. Object Management Group. Unified Modeling Language. [En línea] [Citado el: 20 de Marzo del 2011] Disponible en: http://www.uml.org/#Links-Methodologies.

43. Vila, Ana Fernández. [En línea] [Citado el: 3 de Abril del 2011.]
Disponible en: http://tvdi.det.uvigo.es/~avilas/UML/node50.html.

44. *Flujo de trabajo de pruebas.* Aguilar, Violena Hernández.

45. Software Engineering Research Group [En línea] [Citado el: 17 de Mayo del 2011]
Disponible en: http://in2test.lsi.uniovi.es/gt26/.

46. Grupo de Trabajo AEN/CTN71/SC7/GT26 Pruebas de Software. ISO/IEC 29119: The New International Software Testing Standard. [En línea] [Citado el: 17 de Mayo del 2011]
Disponible en: http://in2test.lsi.uniovi.es/gt26/.

ANEXOS

Diagramas de clases de los paquetes de acceso a datos del módulo Web.

Anexo #1

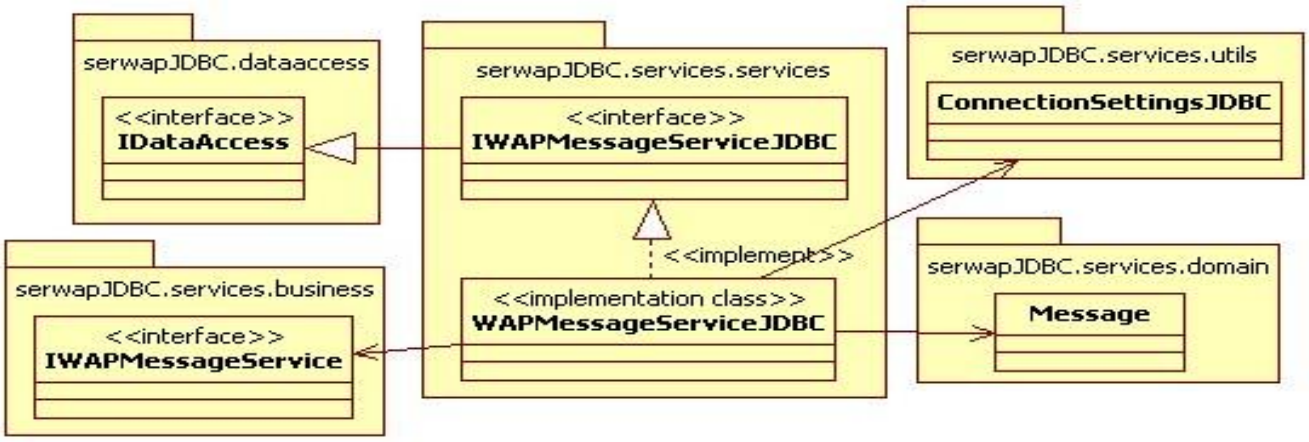

Figura #17. Diagrama de clases del paquete "services" del jar JDBC.

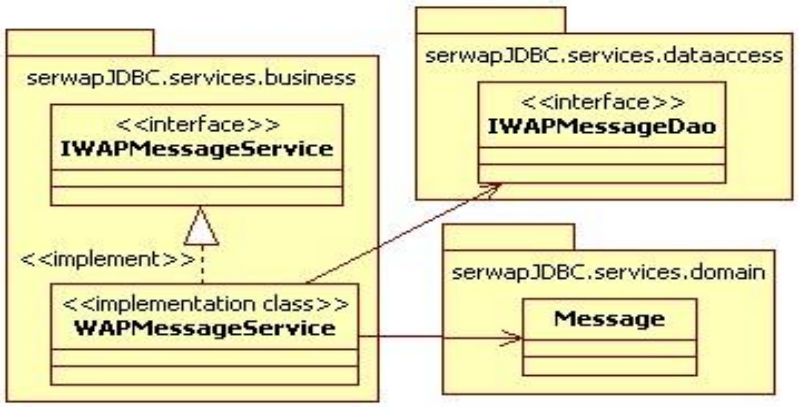

Figura #18. Diagrama de clases del paquete "business" del jar JDBC.

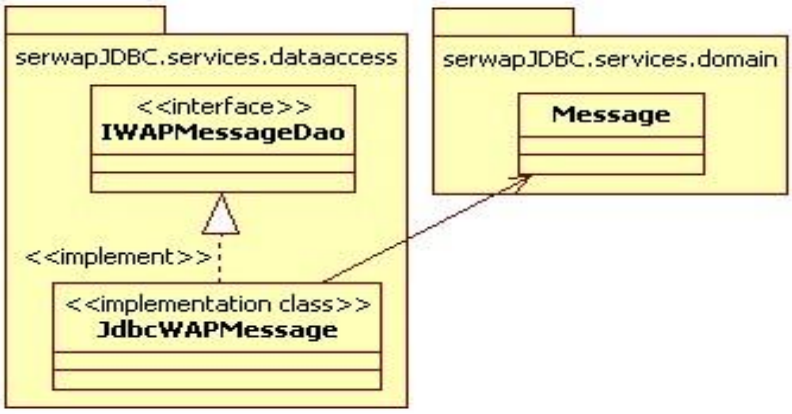

Figura #19. Diagrama de clases del paquete "dataaccess" del jar JDBC.

Diagramas de clases de los paquetes N2AClient y acceso a datos del módulo SDSCenter.

Anexo #2

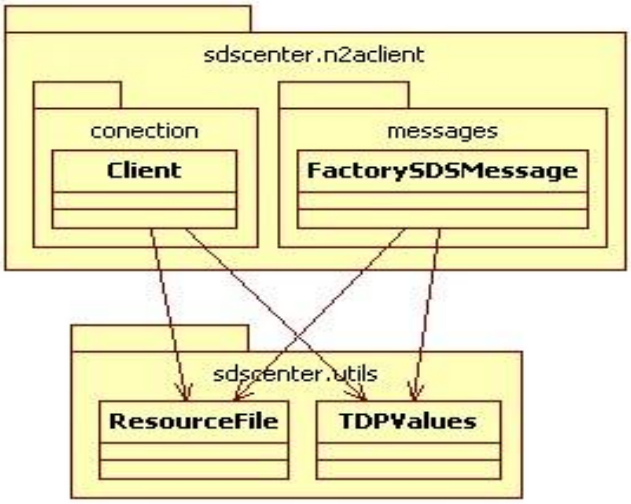

Figura #20 Diagrama de clases del paquete "N2AClient".

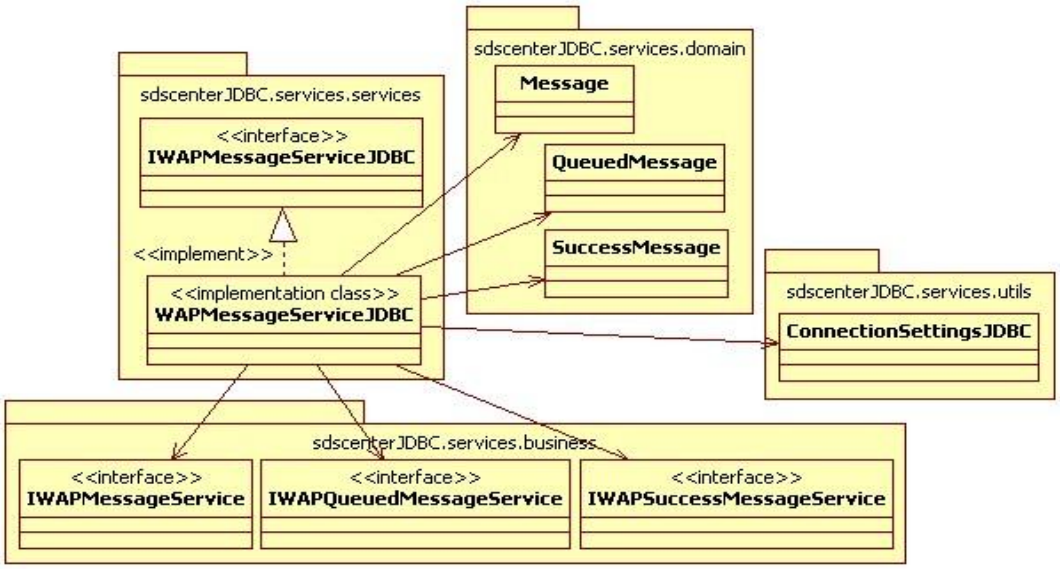

Figura #21. Diagrama de clases del paquete "services" del jar JDBC.

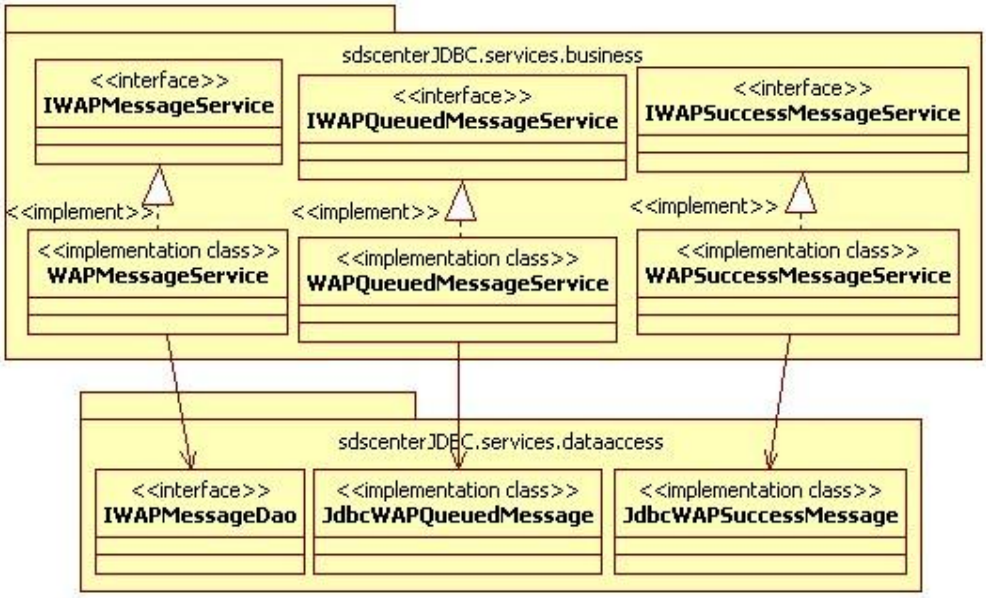

Figura #22. Diagrama de clases del paquete "business" del jar JDBC.

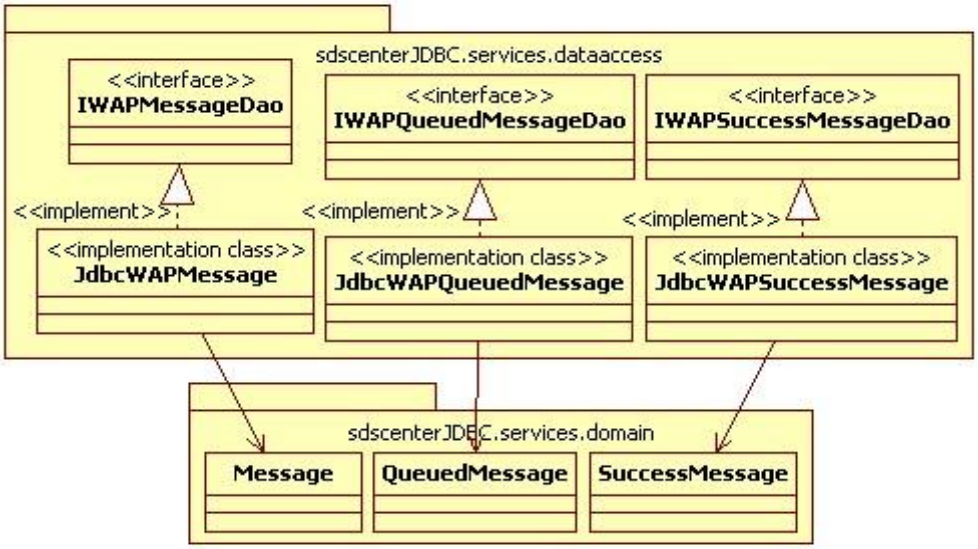

Figura #23. Diagrama de clases del paquete "dataaccess" del jar JDBC.

Diagramas de clases del diseño por funcionalidades del módulo Web.

Anexo #3

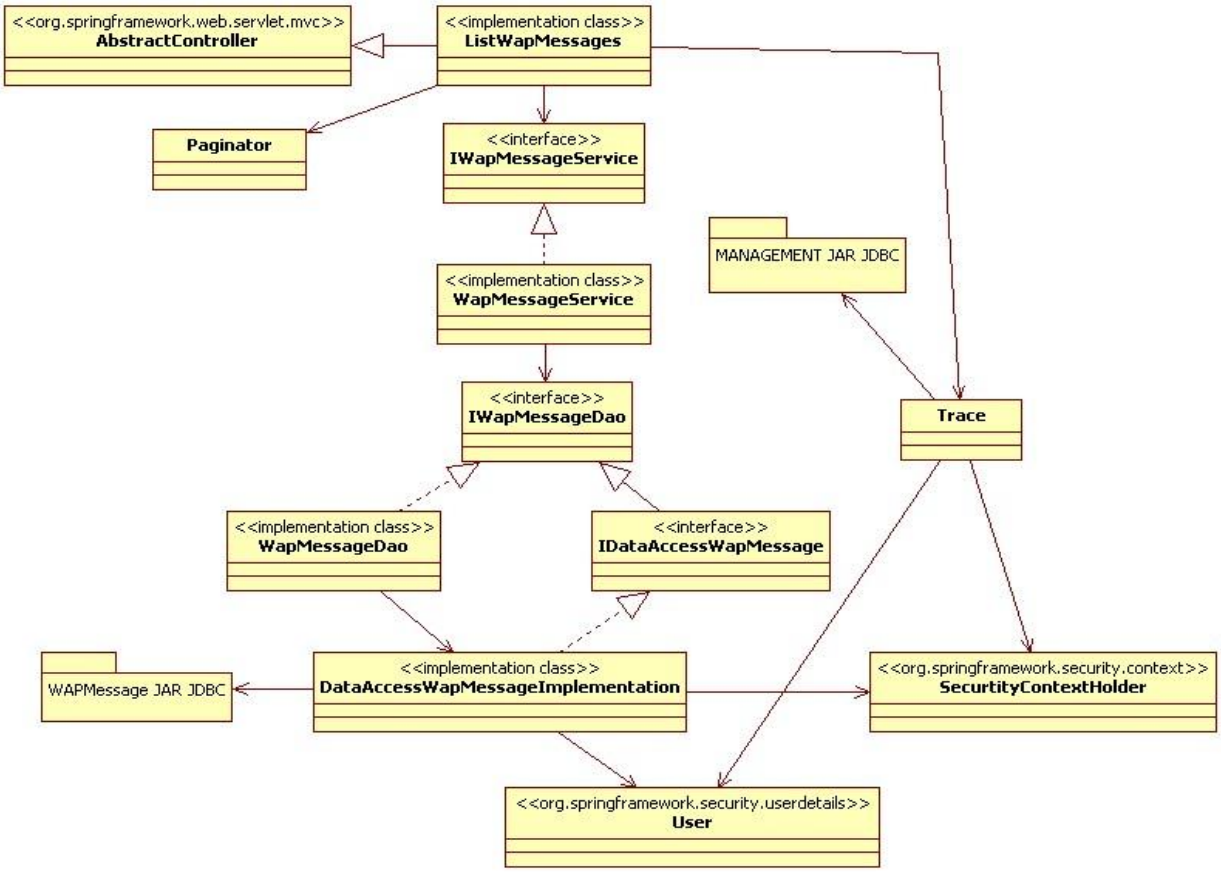

Figura #24. Diagrama de clases del paquete WAPMessage. Acción: "Listar mensajes". Módulo Web.

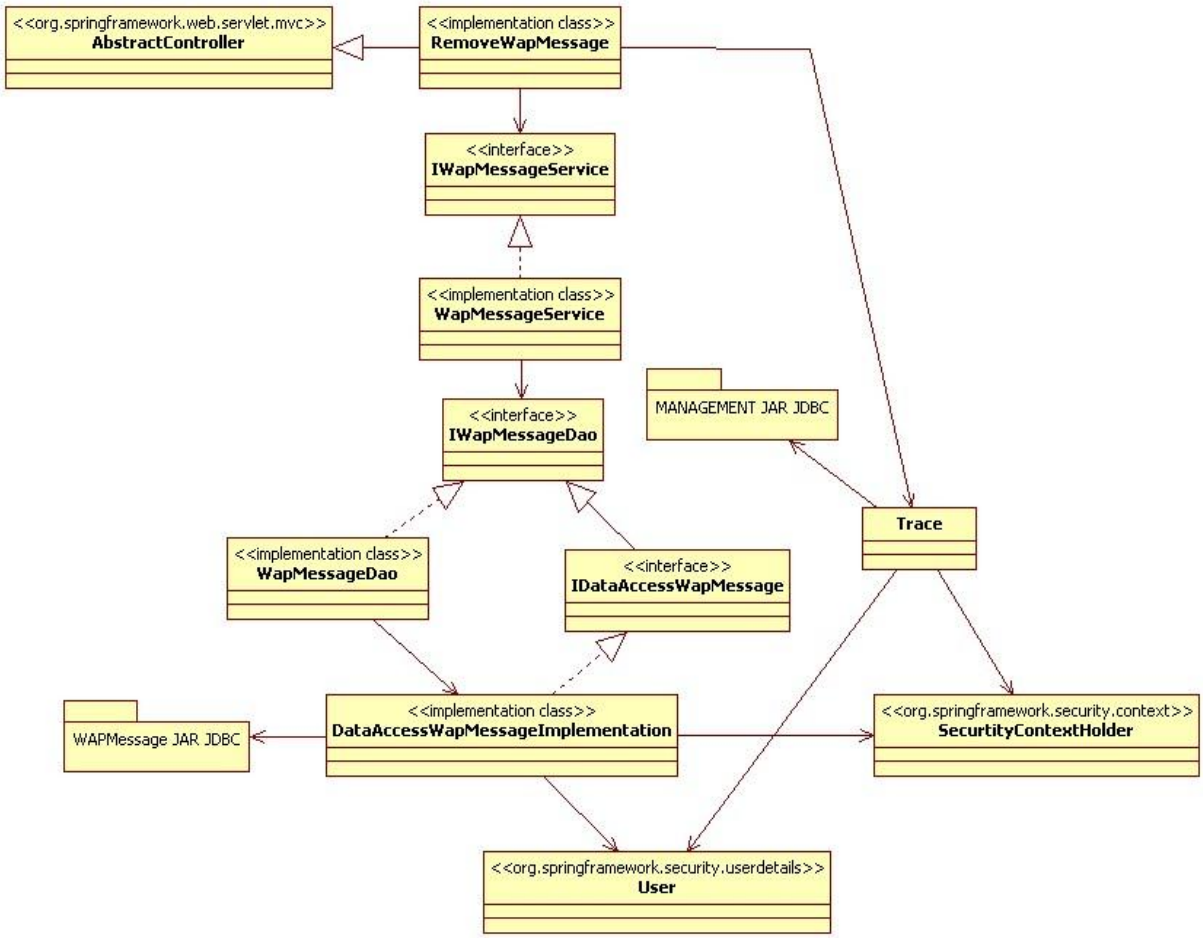

Figura #25. Diagrama de clases del paquete WAPMessage. Acción: "Eliminar mensajes". Módulo Web.

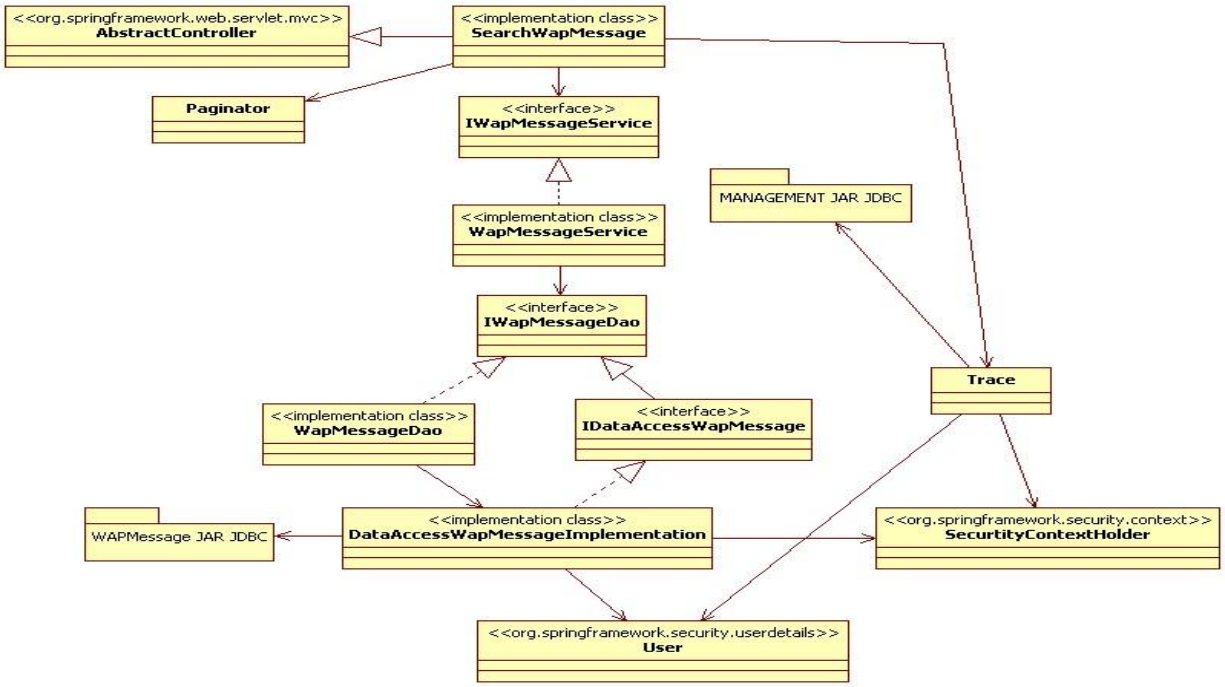

Figura #26. Diagrama de clases del paquete WAPMessage. Acción: "Buscar mensajes". Módulo Web.

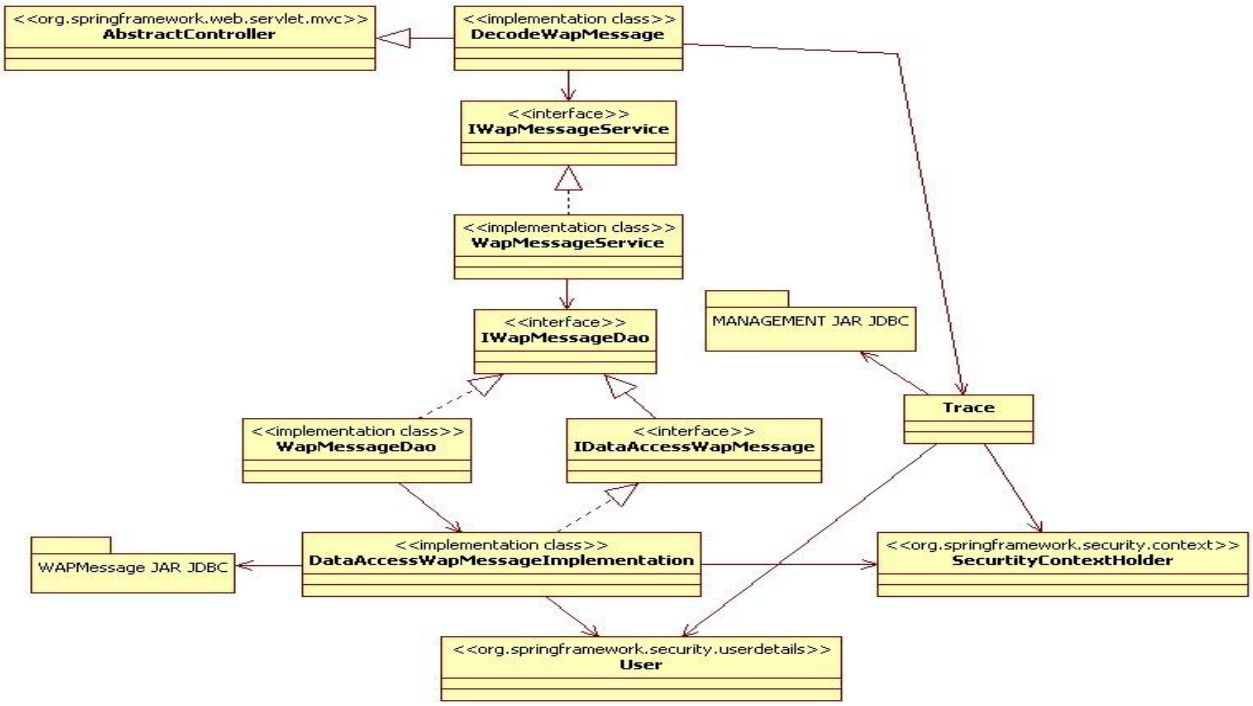

Figura #27. Diagrama de clases del paquete WAPMessage. Acción: "Decodificar mensajes". Módulo Web.

Diagramas de secuencia.

Anexo #4

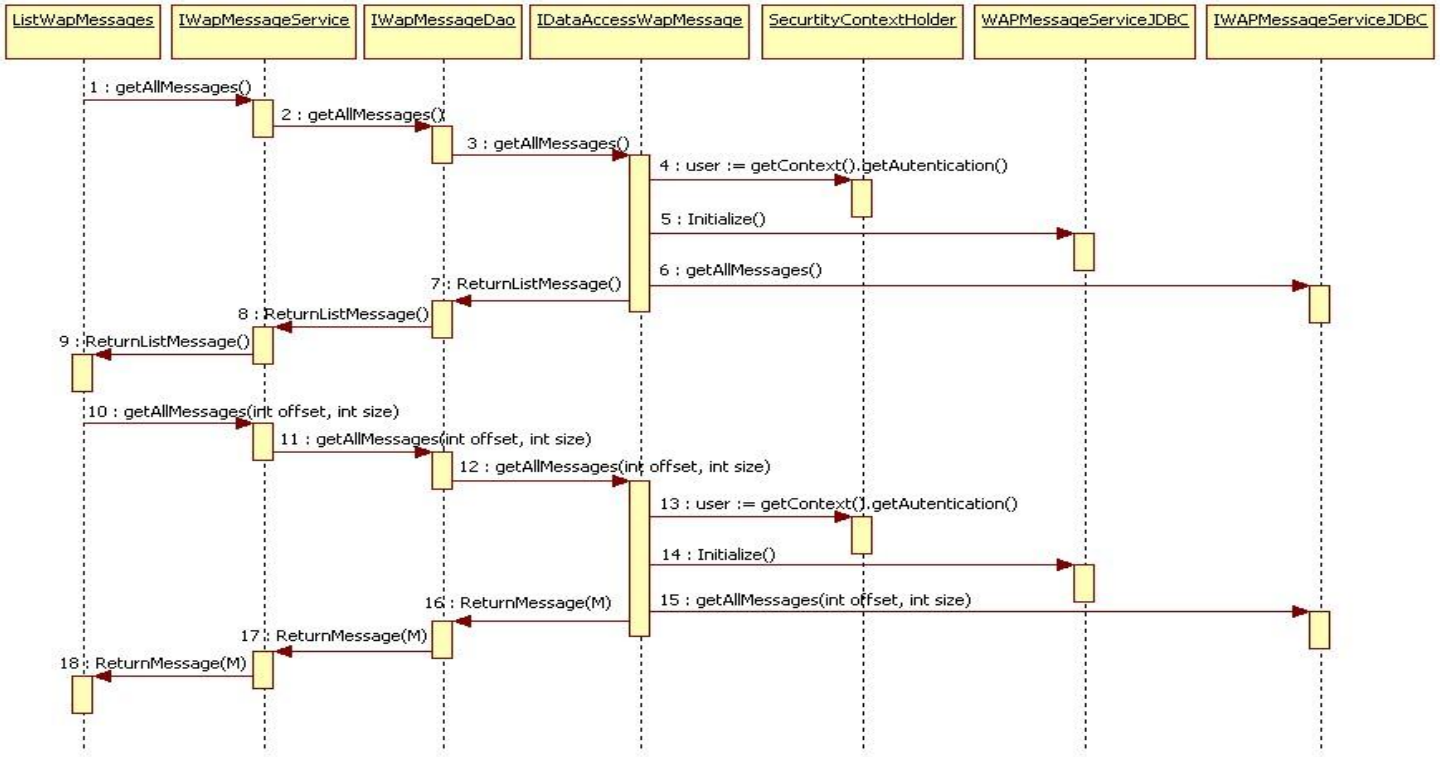

Figura #28. Diagrama de secuencia para la funcionalidad Listar Mensaje.

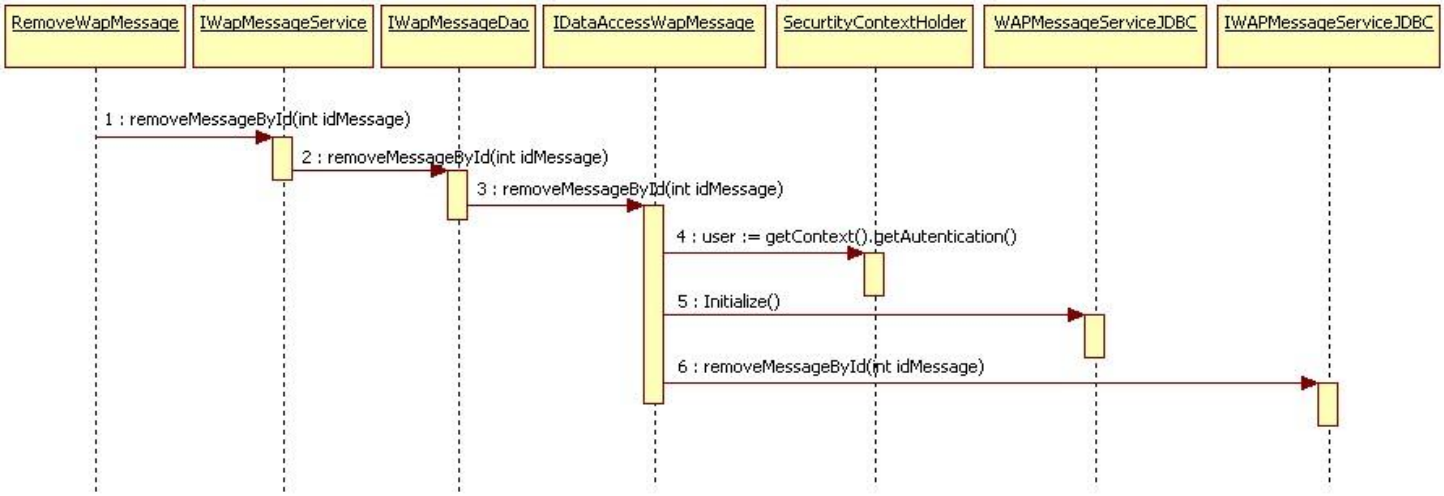

Figura #29. Diagrama de secuencia para la funcionalidad Eliminar Mensaje.

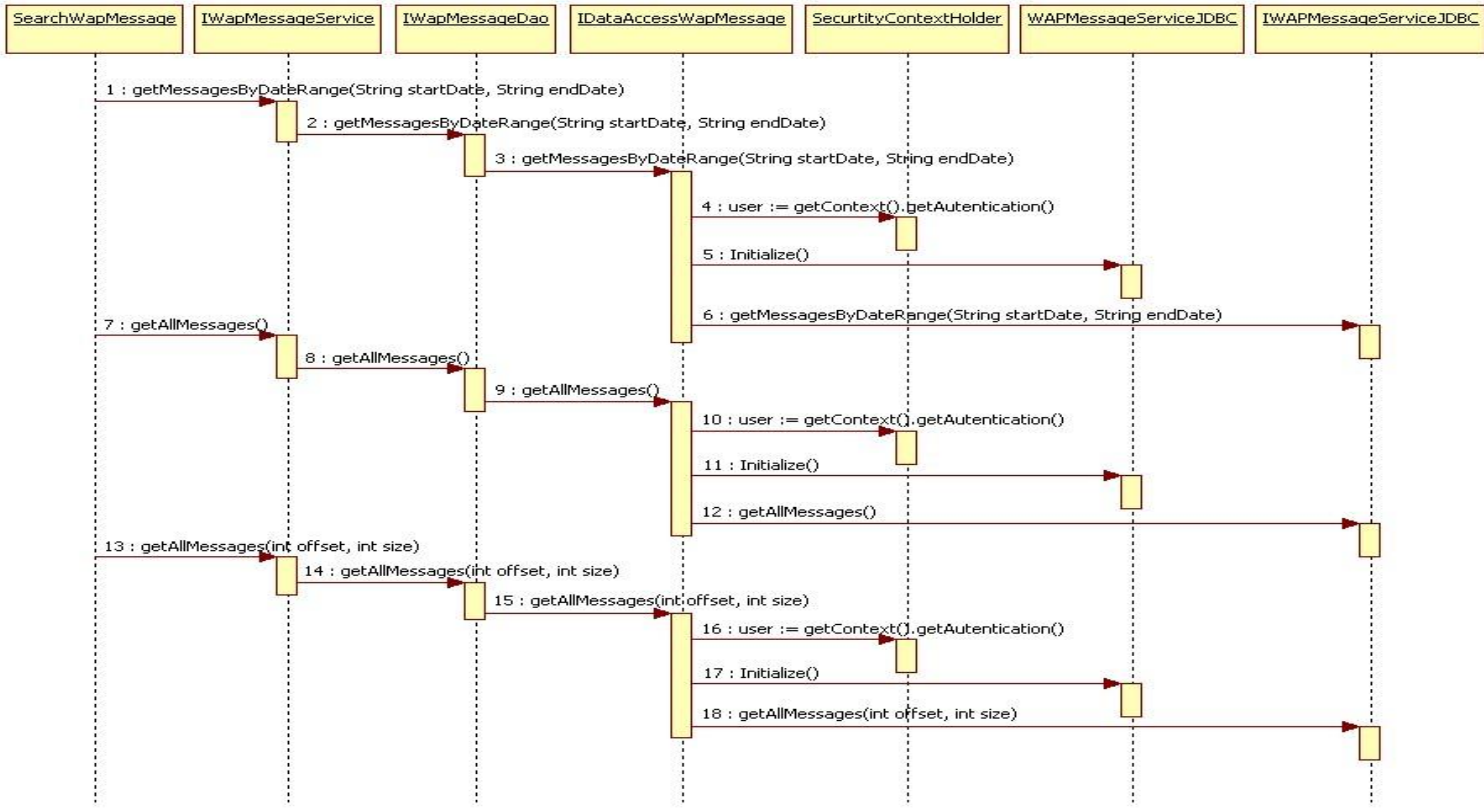

Figura #30 Diagrama de secuencia para la funcionalidad Buscar Mensaje.

Diagramas de componentes del sistema.

Anexo #5

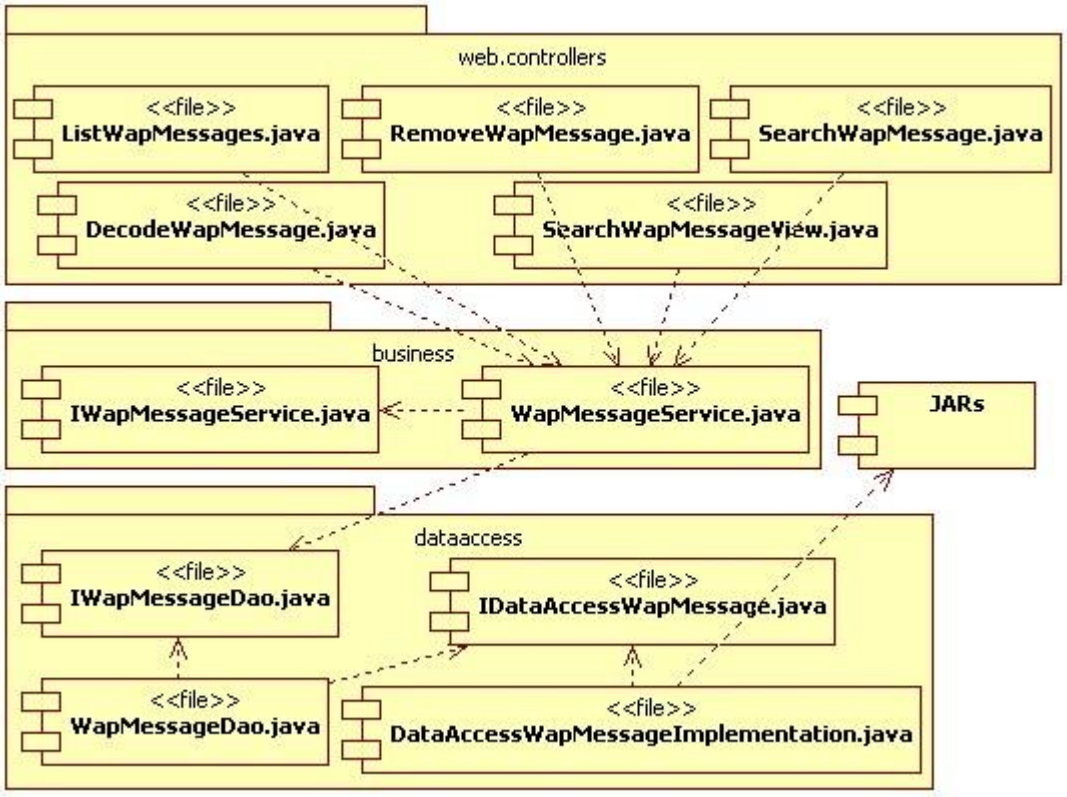

Figura #31. Diagrama de componentes del módulo Web.

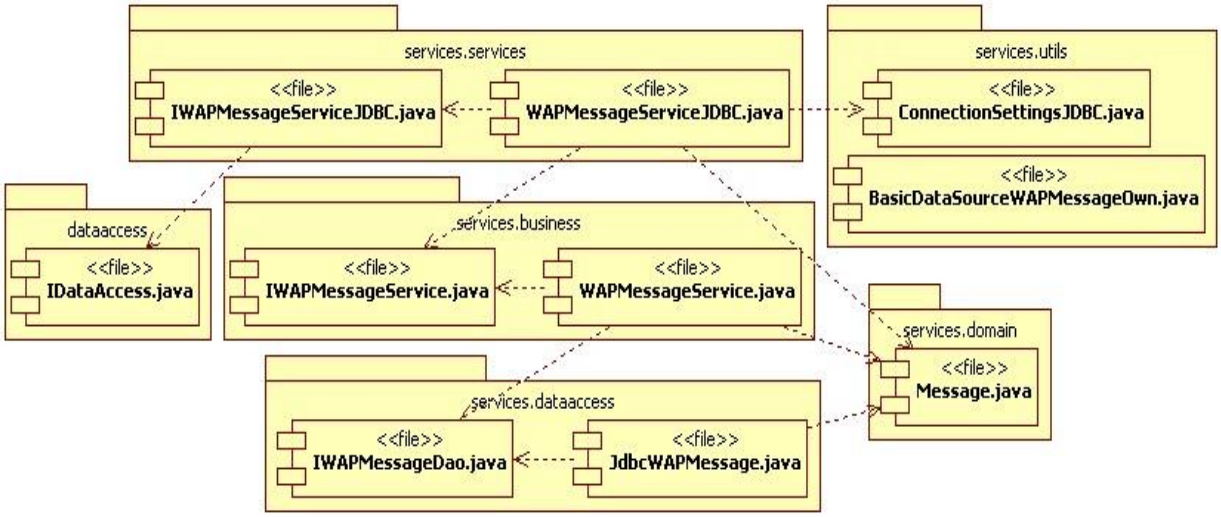

Figura #32. Diagrama de componentes del jar JDBC para el módulo Web.

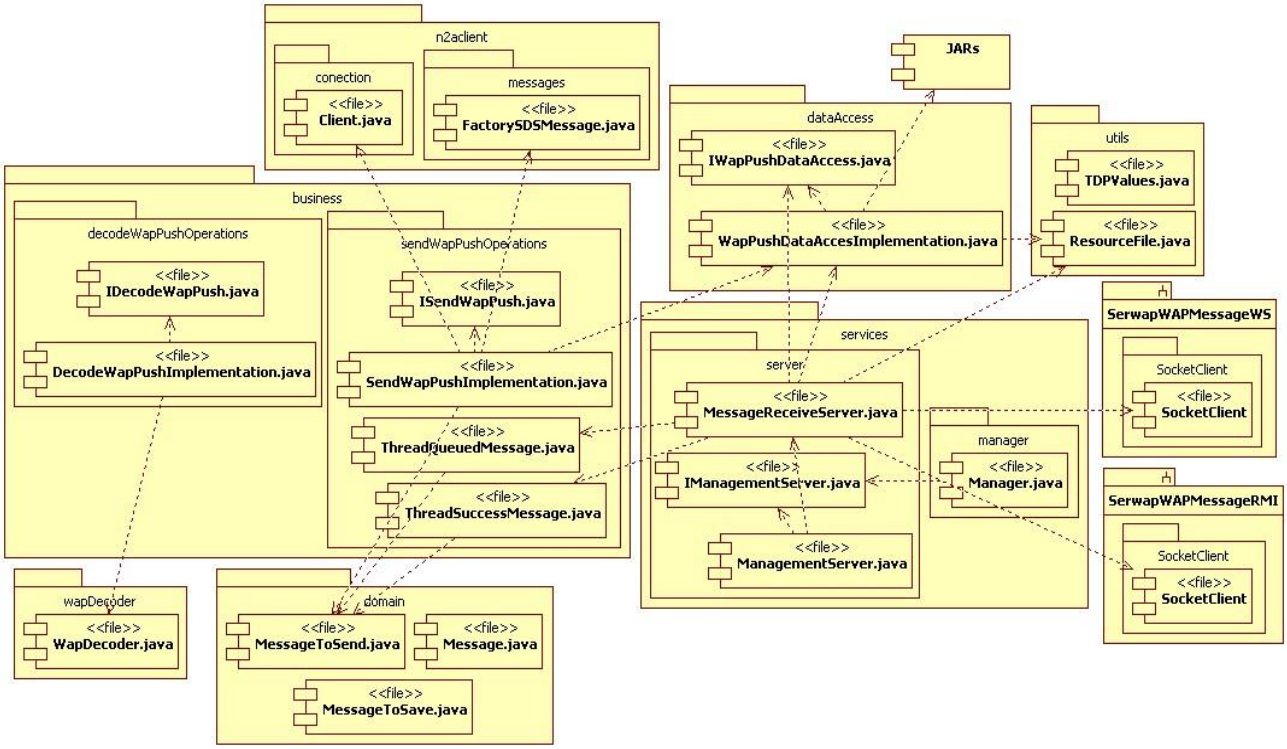

Figura #33. Diagrama de componentes del módulo SDSCenter.

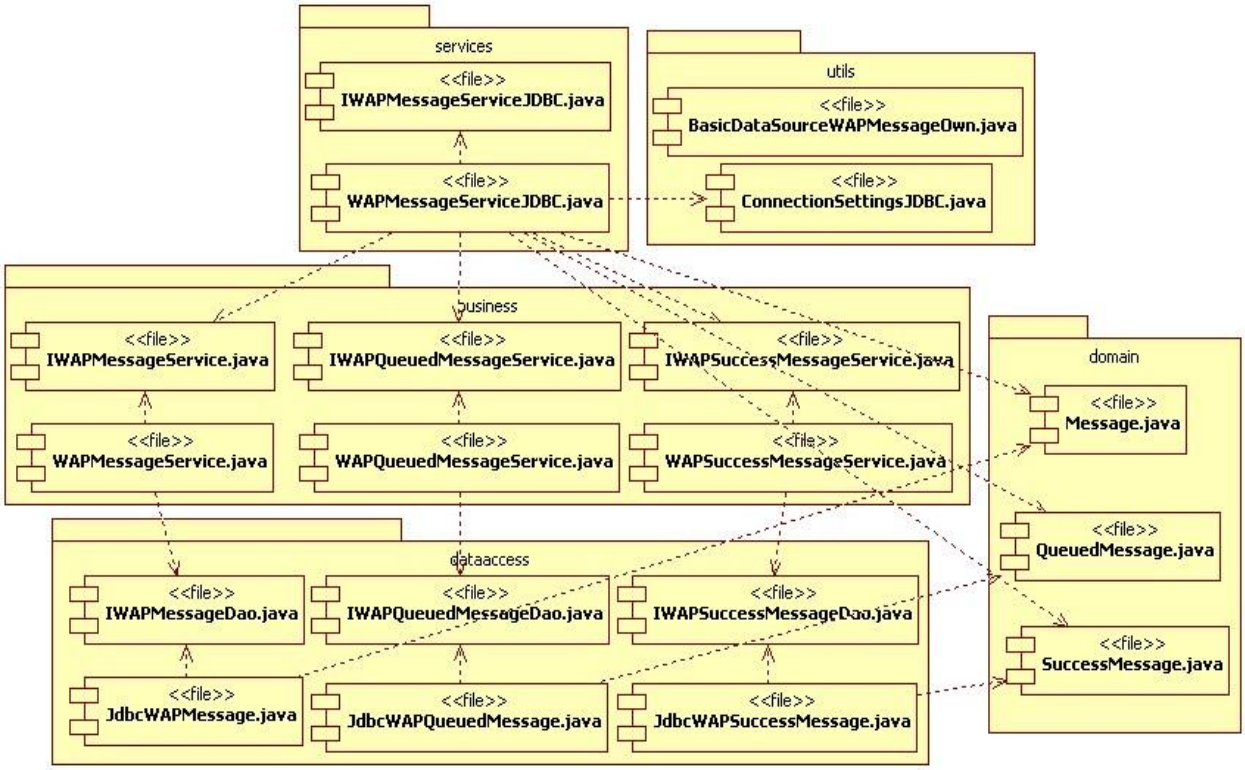

Figura #34. Diagrama de componentes del jar JDBC para el módulo SDSCenter.

GLOSARIO DE TÉRMINOS

ALBET: Albet Ingeniería y Sistemas, S.A. es una empresa cubana, cuyo origen y desarrollo se vincula estrechamente a la Universidad de Ciencias Informáticas (UCI).

API: Conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

ETSI: Del inglés: European Telecomunication Standars Institute. Instituto Europeo de Normas de Telecomunicaciones. Organización que se dedica a la estandarización de las comunicaciones de Europa.

Espectro: Diagrama que representa la distribución de una magnitud en función de otra ligada a ella. Generalmente la primera es una intensidad y la segunda una energía.
Resultado de la separación de las componentes de distinta frecuencia de un fenómeno ondulatorio.
Serie de frecuencias que resultan de la dispersión de un fenómeno formado por ondas.

GPRS: Es una extensión del Sistema Global para Comunicaciones Móviles para la transmisión de datos no conmutada (o por paquetes).

GSM: Del francés: Groupe Special Mobile. Sistema Global para las Comunicaciones Móviles. Sistema estándar definido para la comunicación mediante teléfonos móviles que incorporan tecnología celular.

HTTP: Del inglés: HyperText Transfer Protocol, en español: Protocolo de Transferencia de Hipertexto. Es el protocolo usado en cada transacción de la World Wide Web.

IDE: Del inglés: Integrated Development Environment. Entorno de Desarrollo Integrado. Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de gráficas GUI. Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de otras aplicaciones existentes.

Internet: Conjunto descentralizado de redes de comunicación interconectadas, que utilizan la familia de protocolos TCP/IP y posee alcance mundial constituyendo una red mundial de comunicación.

Inversión de Control (IoC): Del inglés: Inversion of Control. Es un método de programación en el que el flujo de ejecución de un programa se invierte respecto a los métodos de programación tradicionales, en los que la interacción se expresa de forma imperativa haciendo llamadas a procedimientos (procedure calls) o funciones.

JDBC: Del inglés: Java Database Connectivity. Es un API para trabajar con bases de datos desde Java, independientemente de la base de datos a la que se accede.

RF o Radiofrecuencia: Término que se aplica a la porción menos energética del espectro electromagnético, situada entre unos 3 Hz y unos 300 GHz.

RMI: Del inglés: Remote Method Invocation. Es un mecanismo ofrecido por Java para invocar un método de manera remota. Forma parte del entorno estándar de ejecución de Java y provee de un mecanismo simple para la comunicación de servidores en aplicaciones distribuidas basadas exclusivamente en Java.

RUP: Del inglés: Rational Unified Process. Proceso Unificado de Racional. Proceso de desarrollo de software que en su modelación define como principales elementos a los trabajadores, las actividades, los artefactos y flujo de actividades. Guía el proceso de desarrollo de software.

Servicios Push: Servicio mediante el cual se envía información a los usuarios de dispositivos móviles.

Servlets: Es un programa que se ejecuta en un servidor. El uso más común de los servlets es generar páginas web de forma dinámica a partir de los parámetros de la petición que envíe el navegador web.

TELTRONIC: No es solamente el fabricante de uno de los sistemas más innovadores disponible en la actualidad, sino que también proporciona una amplia gama de servicios completos para proyectos como replanteos, instalación, puesta en marcha, mantenimiento, servicios postventa y opciones de financiación.

Tecnología Push: Describe un estilo de comunicaciones donde la petición de transacción se origina en el servidor.

Terminal WAP: Terminal que usa el estándar abierto internacional para comunicaciones móviles WAP.

Triggers: (en español Disparador): En una Base de datos, es un procedimiento que se ejecuta cuando se cumple una condición establecida al realizar una operación de inserción (INSERT), actualización (UPDATE) o borrado (DELETE).

UMTS: Es una de las tecnologías usadas por los móviles de tercera generación (3G), sucesora de GSM.

WAP: Del ingles: Wireless Application Protocol. Es un estándar seguro que permite que los usuarios accedan a información de forma instantánea a través de dispositivos inalámbricos.

WBXML: Del ingles: Wireless Binary Extensible Markup Language. Es un estándar para permitir que los documentos XML sean transmitidos de una manera compacta a través de la redes de telefonía móvil.

XML: Del inglés: Extensible Markup Language. Lenguaje de Marcas Extensibles. Constituye un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C) que permite definir la gramática de lenguajes específicos.