

Universidad de las Ciencias Informáticas

Trabajo de Diploma para optar por el Título de Ingeniero en
Ciencias Informáticas.

Título: “Servicios de Valor Agregado de Noticias, Promociones,
Estado del Tiempo y Directorio Telefónico de Páginas Amarillas”

Autor(es): Maite Albear Maceo

Alejandro Cantero Díaz

Tutor(es): Ing. Yenlys Guerra Dávila

Cotutor: Ing. Darien Jesús Álvarez de la Cruz



“Año 52 de la Revolución”

Ciudad de la Habana. Junio 2010.

Declaración de autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos al <nombre área> de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

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

Maite Albear Maceo
Firma del Autor

Alejandro Cantero Díaz
Firma del Autor

Ing: Yenlys Guerra Dávila
Firma del Tutor

Ing: Darien Jesús Álvarez de la Cruz
Firma del Cotutor

Agradecimientos

Hoy es un día muy especial, no solo para mí sino para aquellas personas que me han apoyado desde el fondo de sus corazones, que aunque hoy no cuento con la presencia de todas siempre estarán acompañándome en los momentos buenos, malos y especiales como lo es el día de hoy, esas personas son mi mamá Juana Emilia Maceo Gaínza la cual representa para mí la mayor inspiración que puede tener una hija hacia su madre, un modelo a seguir, a mi abuelita querida que gracias a ella hoy soy como soy, a mis hermanas Milandrís y Marisleidys Albear Maceo las cuales me han apoyado incondicionalmente durante todo el trayecto de mi carrera y a mi tía Ana Luisa Maceo Gaínza por todo su apoyo y cariño.

Maite

Le agradezco a mi mamá Cecilia Díaz Díaz y a mi papá Jorge Cantero Escalona por haber hecho de mí el hombre que soy hoy, por haberme apoyado en todas mis pretensiones y haberme transmitido todo su amor y cariño. Les agradezco también a mis tíos Pedro, Mercedes, Melva, Kiki, Alejandro, Deivís, Luisa y a mis abuelos Bebo y María por su cariño y apoyo incondicional. A mi papá Boló y a mi mamá Fela (mis segundos padres) por todo el cariño y apoyo que me han brindado. A mis amigos Dania y Jorge por haberme brindado tanta ayuda y tanto apoyo durante los cinco años de mi carrera aquí en Ciudad de la Habana. A todos mis más sinceros agradecimientos por haber hecho posible esto.

Alejandro

Le agradecemos a nuestra Tutora Yenlys Guerra Dávila, a nuestro Cotutor Darien Jesús Álvarez de la Cruz, a todos los profesores que nos han ayudado y a todos los amigos que nos acompañaron en estos 5 años de estudio, nombrarlos a todos haría una lista interminable y cometeríamos el error de dejar fuera algunos, no porque fueran menos importantes, sino porque la memoria nos traicionaría a la hora de escribir estas líneas.

Dedicatoria

A mi mamá que es mi inspiración más grande.

A mi abuelita querida.

A mis hermanas.

Maite

A mis familiares fundamentalmente mis padres.

A mis amigos y compañeros.

En especial a mi mamá Fela.

Alejandro

Resumen

El presente trabajo tiene como objetivo desarrollar un Sub-Sistema que proporcione a CUBACEL una infraestructura que le permita brindar los Servicios de Valor Agregado (SVA) de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas a sus clientes de telefonía celular. El servicio de Promociones consiste en ofrecer un incentivo para la compra o adquisición de un producto o servicio a corto plazo, mientras que el de Noticias proporciona el acontecer nacional e internacional. Por otra parte el servicio de Estado del Tiempo brinda el parte meteorológico más reciente proporcionado por el Instituto de Meteorología y el Directorio Telefónico de Páginas Amarillas se especializa en la búsqueda de informaciones telefónicas. A estos servicios se puede acceder desde un Portal Web comercial o mediante un Portal WAP desde el móvil. Esto será posible mediante la implementación de Servicios Web, utilizando herramientas como: el Entorno de Desarrollo Integrado (IDE): Eclipse, Plataforma de Desarrollo: Java EE, lenguaje de programación: JAVA, Framework: Spring, Gestor de Base de Datos: PostgreSQL y como Metodología de Software: el Proceso Unificado de Desarrollo (RUP). El cumplimiento de este objetivo representa para CUBACEL una alternativa viable, cuyos resultados logran mejorar la calidad y variedad de los servicios de telefonía móvil en Cuba y dar pasos sólidos con vista a colocar al país a la altura de empresas internacionales experimentadas en este sector.

Palabras Claves: Framework, IDE, JAVA, Servicios de Valor Agregado, RUP, CUBACEL.

Índice

INTRODUCCIÓN	1
CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA	4
1.1. INTRODUCCIÓN	4
1.2. SERVICIOS DE VALOR AGREGADO (SVA)	4
1.3. SERVICIOS DE VALOR AGREGADO DE LA PLATAFORMA COMCEL	4
1.4. ESTADO DEL ARTE	5
1.4.1 <i>Content Delivery Platform</i>	5
1.4.2 <i>Telcel</i>	6
1.4.3 <i>Digicel Caribbean</i>	6
1.5. TECNOLOGÍAS Y HERRAMIENTAS A UTILIZAR.	6
1.5.1 <i>Entorno de Desarrollo Integrado (IDE)</i>	6
1.5.1.1. <i>Eclipse</i>	7
1.5.2 <i>Plataforma de desarrollo: Java EE</i>	7
1.5.3 <i>Lenguaje de programación: Java</i>	8
1.5.4 <i>Frameworks</i>	8
1.5.4.1 <i>Spring</i>	9
1.5.5 <i>Spring Web Services (Spring-WS)</i>	10
1.5.6 <i>Web Services</i>	11
1.5.7 <i>Spring Security</i>	13
1.5.8 <i>Servidor Web: Apache Tomcat</i>	13
1.5.9 <i>Gestor de Base de Datos: PostgreSQL</i>	14
1.5.10 <i>Sistema de Control de Versiones: SVN</i>	14
1.5.11 <i>Interfaz de Programación de Aplicaciones (API)</i>	15
1.5.11.1 <i>API SAX</i>	15
1.5.12 <i>Metodología de Desarrollo de Software</i>	16
1.5.12.1 <i>RUP</i>	16
1.5.13 <i>Lenguaje de modelado: UML</i>	17
1.5.14 <i>Herramienta CASE: Visual Paradigm</i>	17

Índice

1.5.15	<i>Hibernate</i>	17
1.5.16	<i>Especificaciones: Java Persistence API (JPA)</i>	17
1.6.	CONCLUSIONES	18
CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA.....		19
2.1	INTRODUCCIÓN	19
2.2	PROPUESTA DEL SISTEMA	19
2.3	MODELO DE DOMINIO	19
2.4	REQUERIMIENTOS DEL SISTEMA.....	21
2.4.1	<i>Requisitos Funcionales</i>	21
2.4.2	<i>Requisitos No Funcionales</i>	24
2.5	MODELO DE CASOS DE USO DEL SISTEMA.....	26
2.5.1	<i>Definición de actores del sistema a automatizar</i>	27
2.5.2	<i>Breve descripción de los Casos de Uso</i>	27
2.6	CONCLUSIONES	31
CAPÍTULO 3. ANÁLISIS Y DISEÑO DEL SISTEMA		32
3.1	INTRODUCCIÓN	32
3.2	ANÁLISIS	32
3.2.1	<i>Definición de Modelo de Análisis</i>	32
3.2.2	<i>Clases del Análisis</i>	32
3.3	DISEÑO.....	33
3.3.1	<i>Definición de Modelo de Diseño</i>	33
3.3.2	<i>Clases del Diseño</i>	33
3.4	DIAGRAMAS DE INTERACCIÓN	43
3.5	MODELO DE DATOS.....	45
3.5.1	<i>Modelo Lógico de Datos (Diagrama de Clases Persistentes)</i>	45
3.5.2	<i>Modelo Físico de Datos (Diagrama Entidad-Relación)</i>	46
3.6	DESCRIPCIÓN DE LA ARQUITECTURA	47
3.6.1	<i>Patrones Arquitectónicos</i>	47
3.6.2	<i>Patrones de Diseño (Patrones GRASP)</i>	47

Índice

3.7	CONCLUSIONES	48
CAPÍTULO 4. IMPLEMENTACIÓN		49
4.1	INTRODUCCIÓN	49
4.2	MODELO DE IMPLEMENTACIÓN	49
4.2.1	<i>Diagrama de Despliegue</i>	49
4.2.2	<i>Diagrama de Componentes</i>	50
4.3	CONCLUSIONES	53
CAPÍTULO 5. ESTUDIO DE FACTIBILIDAD.....		54
5.1	INTRODUCCIÓN	54
5.2	MÉTODO DE ESTIMACIÓN “PUNTOS POR CASOS DE USO”	54
5.2.1	<i>Cálculo de Puntos de Casos de Uso sin ajustar</i>	54
5.2.2	<i>Cálculo de Puntos de Casos de Uso ajustados</i>	56
5.3	CÁLCULO DEL ESFUERZO.....	59
5.4	DISTRIBUCIÓN DEL ESFUERZO ENTRE LAS DIFERENTES ACTIVIDADES DE UN PROYECTO.....	60
5.5	ANÁLISIS Y ESTIMACIÓN DE COSTO	61
5.6	BENEFICIOS TANGIBLES E INTANGIBLES	62
5.7	CONCLUSIONES	62
CONCLUSIONES GENERALES		63
RECOMENDACIONES.....		64
REFERENCIA BIBLIOGRÁFICA.....		65
BIBLIOGRAFÍA.....		67
GLOSARIO DE TÉRMINOS.....		69

Introducción

En los últimos años las tecnologías inalámbricas han tenido un gran desarrollo, destacándose dentro de estas la telefonía celular, la cual ha revolucionado los servicios brindados al cliente, pues, el teléfono celular se ha convertido en una herramienta necesaria para todas las personas, debido a los múltiples servicios que ofrece. A pesar de que esta tecnología fue concebida estrictamente para la voz, hoy es capaz de brindar otros servicios como datos, audio, video, imágenes, entre otros.

Nuestro país no se encuentra excluido en cuanto a la utilización y desarrollo de la telefonía celular, pues actualmente existen alrededor de un millón de clientes celulares. La empresa encargada de expandir y desarrollar las comunicaciones móviles en Cuba es ETECSA, dentro de la cual se encuentra CUBACEL como unidad de negocios móviles. La misma ofrece diferentes servicios a sus clientes, como son prepago, transmisión de datos, pospago, mensajes, servicios adicionales, etc.

La creciente demanda de los clientes de telefonía móvil en Cuba requiere un aumento en la variedad de los servicios brindados, haciéndose necesarios Servicios de Valor Agregado como Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas. Por esta razón se le dio la misión a la Universidad de las Ciencias Informáticas como Institución productora de Software, el desarrollo de un Sub-Sistema que le proporcione a CUBACEL una infraestructura capaz de brindar tales servicios.

Como resultado se obtuvo un Sub-Sistema que no contaba con los requerimientos necesarios, ya que presentaba varias imperfecciones en su funcionamiento, por lo cual se hizo necesaria la realización de una nueva implementación del mismo.

Para dar solución a la **situación problemática** descrita anteriormente, se planteó el siguiente **problema científico**: ¿Cómo proporcionar a CUBACEL una infraestructura que le permita brindar los Servicios de Valor Agregado de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas a sus clientes de telefonía celular?

Para solucionar lo antes expuesto, se propuso como **objetivo general**, desarrollar un Sub-Sistema que proporcione a CUBACEL una infraestructura que le permita brindar los Servicios de Valor Agregado de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas a sus clientes de telefonía celular.

El **objeto de estudio** se enmarca en los procesos de proporcionar los Servicios de Valor Agregado para los clientes de telefonía celular, siendo el **campo de acción** los procesos de proporcionar los Servicios de Valor Agregado de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas para los clientes de telefonía celular de CUBACEL.

Para dar cumplimiento al objetivo general planteado, se trazaron las siguientes **tareas de investigación**:

- Definición de la arquitectura a utilizar.
- Investigación de las herramientas necesarias para el desarrollo del Sub-Sistema.
- Definición de los datos persistentes relacionados con cada uno de los servicios web.
- Profundización en los componentes que integran un servicio web.
- Realización del diseño de los servicios web.
- Definición de los métodos que serán exportados como servicios web.
- Implementación de la capa de acceso a datos y la lógica de negocio de los servicios web.

Se plantea como **idea a defender** la siguiente afirmación: Con el desarrollo del Sub-Sistema de Servicios de Valor Agregado de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas, se logrará proporcionar a CUBACEL una infraestructura capaz de ofrecer estos servicios a sus clientes de telefonía celular.

En el transcurso de la investigación se utilizaron como guía los Métodos de la Investigación Científica específicamente los Métodos Teóricos:

Analítico-Sintético: Sirvió para realizar el procesamiento de toda la información pudiendo sintetizar y diferenciar cada una de ellas y de esta forma enfocarlas hacia la investigación.

Histórico-Lógico: Posibilitó conocer los antecedentes y tendencias actuales de las herramientas que posibilitan brindar servicios a clientes de telefonía móvil.

Estructura del documento

Para la elaboración del presente documento se realizó un amplio proceso investigativo, el cual se basó inicialmente en una revisión bibliográfica sobre todo lo referente a la temática a abordar. Teniendo en cuenta lo anterior se estructuró el trabajo de la siguiente manera: introducción, cinco capítulos (los cuales deben seguir la siguiente estructura: introducción, desarrollo y conclusiones), conclusiones generales, recomendaciones, referencia bibliográfica, bibliografía y glosario de términos. A continuación una breve descripción sobre lo contenido en cada capítulo.

Capítulo #1 “Fundamentación Teórica”: Se muestra la fundamentación general que soporta teóricamente la solución del problema. Se hace un análisis de las herramientas y lenguajes de programación para determinar los más apropiados para el desarrollo del Sub-Sistema de los Servicios de Valor Agregado.

Capítulo #2 “Características del Sistema”: Se realiza una descripción detallada del problema existente, así como las particularidades del sistema a desarrollar. En este capítulo además, está contenida la descripción de los principales conceptos mediante un Modelo de Dominio, la captura de requisitos funcionales y no funcionales, así como el diagrama de casos de usos del sistema y las descripciones de los mismos.

Capítulo #3 “Análisis y Diseño”: Describe cómo será realizado el Sub-Sistema de Servicios de Valor Agregado de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas, a partir de las funcionalidades previstas y las restricciones impuestas (requerimientos).

Capítulo #4 “Implementación”: Define la organización de las clases, los objetos en componentes y la estructura de capas de la aplicación.

Capítulo #5 “Estudio de la factibilidad”: Se realiza una estimación de la duración del proyecto de software, así como una evaluación del esfuerzo, el análisis de costos y los beneficios tangibles e intangibles asociados a la realización del Sub-Sistema, proporcionando una visión general de la factibilidad a la hora de realizarlo.

Posteriormente se muestran las conclusiones generales y recomendaciones, así como, la referencia bibliográfica, la bibliografía y glosario de términos que facilitan la comprensión del trabajo.

Capítulo I: *Fundamentación teórica*

1.1. Introducción

Este capítulo refleja un estudio crítico y valorativo sobre el proceso de prestación de Servicios de Valor Agregado para Dispositivos Móviles. Incluye el estado actual de este proceso a nivel mundial y nacional, así como las tendencias tecnológicas y metodológicas más utilizadas por los desarrolladores de soluciones para este fin. Además se definen las herramientas y tecnologías que serán utilizadas en el desarrollo del Sub-Sistema.

1.2. Servicios de Valor Agregado (SVA)

En la actualidad las comunicaciones a través de dispositivos móviles, están aptas para aplicaciones multimedia y transmisiones de datos, lo que ha traído consigo el surgimiento de nuevas tecnologías de comunicación que permiten llevar voz, gráficos, videos y otros tipos de datos directamente al cliente celular.

Los dispositivos móviles se definen como dispositivos electrónicos inalámbricos, que permiten conectarse y navegar a través de internet, tomar fotografías, videos, reproducir música e incluso cuenta con una agenda que viene integrada a la mayoría de estos aparatos, los cuales han tenido una gran difusión por su portabilidad (2006). Las comunicaciones a través de estos dispositivos no requieren de un punto fijo para lograr la comunicación y cada vez, se les incorporan funciones más sofisticadas.

La mayoría de los operadores de telefonía móvil a nivel global ofrecen nuevos servicios a través de otras entidades, dentro de los que podemos encontrar: meteorológicos, noticias, juegos, acceso a bancos, localización, información de tráfico, información turística, restaurantes, etc. Los mismos son conocidos como Servicios de Valor Agregado (SVA).

Los SVA son una fuente adicional de ingresos para las empresas de telefonía móvil. Para los clientes, representan una oportunidad de acceder a servicios y contenidos que van más allá de los usos originales de un celular.

1.3. Servicios de Valor Agregado de la Plataforma COMCEL

La Plataforma COMCEL además de los SVA relacionados con la descarga de contenidos, posibilita brindar otros como: Noticias, Promociones, Directorio Telefónico de Páginas Amarillas y Estado del Tiempo.

El servicio de Promociones ofrece un incentivo para la compra o adquisición de un producto o servicio a corto plazo, mientras que el de Noticias proporciona el acontecer Nacional e Internacional. Por otra parte el servicio de Estado del Tiempo brinda el parte meteorológico más reciente proporcionado por el Instituto de Meteorología y el Directorio Telefónico de Páginas Amarillas se especializa en la búsqueda de informaciones telefónicas. A todos estos servicios se puede acceder desde un Portal WEB de la plataforma o directamente desde el móvil a través de un Portal WAP.

A continuación se realizará una breve descripción acerca de algunas plataformas que prestan SVA para dispositivos móviles.

1.4. Estado del Arte

Existen muchas empresas que se han dedicado a la prestación de SVA para los dispositivos móviles y que han desarrollado productos de mucha calidad. Líderes en esta esfera, dentro de estos productos podemos mencionar “Content Delivery Platform”, de la empresa Ginger Alliance, Telcel perteneciente a América Móvil y Digicel de la Compañía Digicel Caribbean.

1.4.1 Content Delivery Platform

Delivery Platform es una solución completa para una fiable presentación, entrega y facturación de los contenidos digitales, adecuada al dispositivo móvil, además proporciona varios SVA por categorías como son:

- **Entretenimiento:** video clips, juegos móviles, radio móvil.
- **Sociales:** envío por correo, contactos, charlas, info-entretenimiento, búsqueda en Internet, publicidad, Messenger.
- **Empresa:** acceso a la Intranet y a aplicaciones de negocios, Internet móvil, e-mail, calendario Mobile.
- **Comercio:** banca móvil, venta de entradas, viajes y vacaciones de reservas.
- **Alertas y noticias:** información relacionada con películas, tickers de noticias y alertas.

1.4.2 Telcel

Telcel es el principal operador de México, tanto por número de clientes como por facturación. Al igual que la mayoría de los operadores celulares de Latinoamérica y el Caribe, Telcel divide su oferta de valor agregado en dos vertientes: entretenimiento y aplicaciones corporativas. En el primer caso, los servicios se agrupan bajo la marca “Ideas”. Allí el operador concentra las aplicaciones de mensajería (SMS, MMS), descargas (ring-tones, logos) y juegos. Las aplicaciones de Ideas se pueden descargar desde el portal de Telcel y también directamente desde el terminal mediante el uso de WAP (Gnius, et al., 2006).

1.4.3 Digicel Caribbean

Digicel se ha convertido en el operador de telecomunicación móvil de más rápido crecimiento en el Caribe. Actualmente cuenta con más de cuatro millones de usuarios y con operaciones en 22 países. Desde el lanzamiento de Digicel en el año 2001, se han presentado una variedad de opciones de servicio que antes no estaban disponibles para los usuarios de telefonía móvil.

Algunos de los servicios con los que cuenta Digicel son: minutos acumulados (minutos Rollover), servicios de datos GPRS (servicio de radiotransmisión de paquetes generales), envío de mensajes cortos a correo electrónico y mensajes de multimedia (2001).

En Cuba a pesar de que en los últimos años se ha incrementado la densidad de los clientes de telefonía celular, el uso de la tecnología móvil ha tenido poco desarrollo en comparación con el resto del mundo, por esta razón son escasos los servicios brindados a los clientes celulares. CUBACEL que es la unidad de negocios móviles de ETECSA brinda algunos SVA utilizando SMS, pero no cuenta con una plataforma de tecnología WAP.

1.5. Tecnologías y herramientas a utilizar.

1.5.1 Entorno de Desarrollo Integrado (IDE)

Un IDE 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 interfaz gráfica GUI. Provee un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, C#, Delphi, Visual Basic, etc. (Wiroos, 2008).

1.5.1.1. Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma (Angelfire, 2009). Tiene herramientas para desarrollar aplicaciones de consola, Web y Web Services. Da soporte a todo tipo de proyectos abarcando el ciclo de vida completo en el desarrollo de aplicaciones. Dicho IDE ha alcanzado un alto grado de madurez, así como más robustez y rendimiento.

Es soportado por los principales sistemas operativos. Continuamente se están desarrollando nuevos pluging y revisando los anteriores. Eclipse permite la instalación de pluging destinados a mejorar las funcionalidades del propio IDE y a extenderse en más tecnologías.

Entre sus principales características se encuentran:

- Editor de texto.
- Compilación en tiempo real.
- Control de versiones con CVS.

1.5.2 Plataforma de desarrollo: Java EE

Java EE es una plataforma de programación para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java con arquitectura de n niveles, basándose ampliamente en componentes de software modulares ejecutados sobre un servidor de aplicaciones. Incluye varias especificaciones de API, Servlets, JSP y varias tecnologías de servicios web (Alva, 2009). Esta plataforma facilita el desarrollo de aplicaciones multicapa como presentación, negocio y acceso a datos.

Beneficios de Java EE:

Permite al desarrollador crear una aplicación de empresa que es portable entre plataformas y escalable. El servidor de aplicaciones puede manejar las transacciones, seguridad, escalabilidad, concurrencia y gestión de los componentes que son desplegados, significando que los desarrolladores pueden concentrarse más en la lógica de negocio de los componentes, en lugar de las tareas de mantenimiento de bajo nivel (Alva, 2009).

Ventajas de Java EE:

- Soporte para múltiples plataformas y sistemas operativos
- Avalado por múltiples empresas (SUN, IBM, ORACLE, etc.)
- Competitividad
- Soluciones libres

1.5.3 Lenguaje de programación: Java

Java es un lenguaje de desarrollo de propósito general y como tal es válido para realizar todo tipo de aplicaciones profesionales. Incluye una combinación de características que lo hacen único y está siendo adoptado por multitud de fabricantes como herramienta básica para el desarrollo de aplicaciones comerciales de gran repercusión.

Una de las características más importantes es que los programas “ejecutables”, creados por el compilador de Java, son independientes de la arquitectura. Se ejecutan indistintamente en una gran variedad de equipos con diferentes microprocesadores y sistemas operativos.

- Es público.
- Permite escribir Applets (pequeños programas que se insertan en una página HTML y se ejecutan en el ordenador local).
- Se pueden escribir aplicaciones para intrarredes, aplicaciones cliente/servidor, aplicaciones distribuidas en redes locales y en Internet.
- Es fácil de aprender y está bien estructurado.
- Es un lenguaje sin punteros y sin necesidad de tener que controlar la asignación de memoria a estructuras de datos u objetos (Xerox, 2001).

1.5.4 Frameworks

Es un esquema para desarrollar un sistema integrando componentes personalizables e intercambiables. Constituyen estructuras de soporte definidas mediante la cual otros proyectos de software pueden ser organizados y desarrollados. Son diseñados con el objetivo de facilitar el desarrollo del producto,

permitiéndole a los diseñadores e implementadores enfocarse en aspectos de mayor nivel, concentrando más esfuerzo y tiempo en identificar los requerimientos de software.

Objetivos principales que persigue un Framework:

- Acelerar el proceso de desarrollo.
- Reutilizar el código ya existente.
- Promover buenas prácticas de desarrollo como el uso de patrones.
- Ofrece mucha libertad a los desarrolladores, gracias a su diseño en módulos.
- Facilita el desarrollo de funcionalidades específicas en diferentes paquetes entre los que pueden mencionarse DAO, JDBC, ORM entre otros.

Los Frameworks suelen incluir:

- Soporte de programas.
- Bibliotecas.
- Software para desarrollar y unir diferentes componentes de un proyecto de desarrollo de programas.

1.5.4.1 Spring

Spring es un framework de código abierto de desarrollo de aplicaciones para la plataforma Java. Una de las principales características de Spring Framework es la inyección de dependencias.

La inyección de dependencias radica 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 para todos los usuarios (al menos en el caso de Spring) evitando tener que andar extendiendo clases o tumbar el servidor de la BD (Vicente, 2006).

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.

Spring proporciona:

- Una potente gestión de configuración basada en JavaBeans, aplicando los principios de Inversión de Control (IoC), esto hace que la configuración de aplicaciones sea rápida y sencilla.
- Una capa de abstracción JDBC que ofrece una significativa jerarquía de excepciones (evitando la necesidad de obtener de SQLException los códigos que cada gestor de base de datos asigna a los errores), simplifica el manejo de errores y reduce considerablemente la cantidad de código innecesario.
- Integración con Hibernate, es decir especial soporte a Hibernate añadiendo convenientes características de IoC, todo ello cumpliendo con las transacciones genéricas de Spring y la jerarquía de excepciones DAO.

1.5.5 Spring Web Services (Spring-WS)

Spring Web Services, es un producto de la comunidad Spring enfocado a la creación de aplicaciones basadas en el uso de Servicios Web. Tiene como objetivo facilitar el desarrollo de servicios SOAP “contract-first”, permitiendo la creación de Servicios Web flexibles usando una de las tantas maneras de manipular contenido XML. Este producto está basado en el propio Spring, lo que significa que se pueden utilizar conceptos de Spring como la inyección de dependencias.

Hay dos maneras de hacer Servicios Web: contract-first y contract-last. En los Servicios Web contract-last no hay necesidad de manipular complejos archivos WSDL y XSD. Mediante esta vía se escribe primero el código en Java y mediante el framework se generan estos archivos. Este camino trae consigo varias dificultades debido a que un pequeño cambio en API, genera cambios también en el contrato del servicio, así como en cada uno de los clientes que consumen el servicio.

La solución a este problema del contract-last es crear primeramente el contrato y después decidir cómo debería ser implementado. Es ahí donde interviene contract-first. Mediante este, el contrato se escribe

cuidadosamente enfatizando en los servicios esperados y no en la manera de implementarlos, la cual puede estar sujeta a cambios (Walls, et al., 2008).

Principales características de Spring-WS:

- Facilita aplicar las mejores prácticas para la creación de servicios web.
- Facilidad para distribuir los pedidos XML a través de diferentes tipos de mapeos.
- Soporte para varias librerías de manejo de XML (DOM, SAX, entre otros).
- Integración con Spring Framework.

1.5.6 Web Services

Durante los últimos años, las empresas han utilizado las tecnologías para mejorar sus procesos de negocio, incorporando sistemas de gestión, atención al cliente y otras aplicaciones que automatizan estos procesos. Pero no es suficiente, actualmente es necesario contar con la capacidad necesaria para permitir que los sistemas conversen entre sí, estén integrados y operen de forma amigable como si fueran uno solo. Una solución a esto es utilizando Web Services, los cuales describen una forma estandarizada para integrar aplicaciones Web, permitiendo intercambiar datos entre sí con el objetivo de ofrecer servicios.

¿Qué es un Web Services?

Un Web Services es una interfaz capaz de recibir una petición, activar unos procesos y devolver los resultados. Todo esto, en Internet y a través de protocolos de red (HTTP, FTP, SMTP). La comunicación entre los diferentes entornos del Web Services se realiza mediante XML.

Para establecer un diálogo coherente entre el WSC (Web Services Client), que envía la petición y recibe la respuesta y el WSS (Web Services Server), el que ejecuta el proceso y envía la respuesta, se utiliza SOAP (Simple Object Access Protocol), que es una codificación basada en XML.

Para poder crear un WSC, necesitamos saber cómo está constituido el WSS, es decir, qué métodos tiene implementados, qué tipos de datos espera recibir y cuál será su respuesta. El WSDL (Web Services Description Language) es un documento XML creado para el WSS donde se describe toda esta información.

Un Web Services, en vez de obtener peticiones desde un navegador y devolver páginas web como respuesta, recibe peticiones, mediante un mensaje formateado con SOAP, desde otras aplicaciones realiza la labor que le han pedido y devuelve un mensaje de respuesta también con formato SOAP.

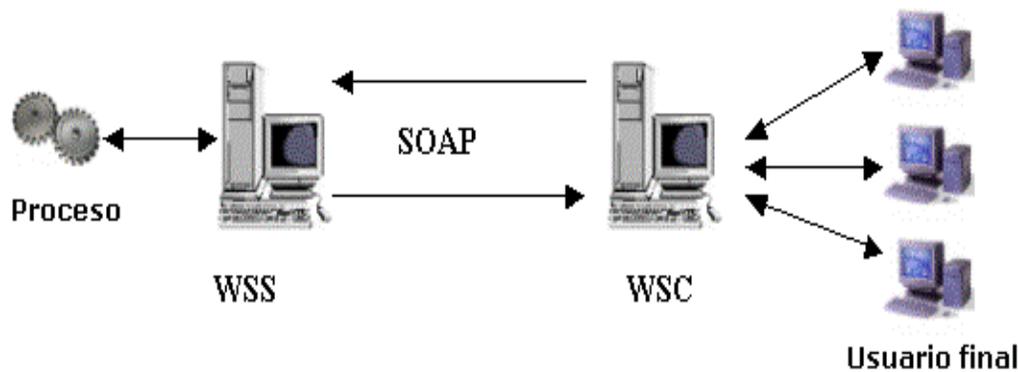


Figura 1: Funcionamiento de los Web Services.

¿Para qué sirve un Web Services?

Un Web Services sirve para tener acceso a información y procesos remotos a través de aplicaciones. La finalidad de un Web Services, es la de ofrecer, vender o alquilar un proceso y que este pueda ser invocado por otras aplicaciones sin considerar el lenguaje de programación utilizado.

Tecnologías Web Services.

Los Web Services están contruidos con varias tecnologías que trabajan conjuntamente con los estándares que están emergiendo para asegurar la seguridad y operatibilidad. A continuación se describen brevemente las tecnologías utilizados por los Web Services.

- XML

Es una especificación desarrollada por W3C. Permite a los desarrolladores crear sus propios tags, permitiéndoles habilitar definiciones, transmisiones, validaciones, e interpretación de los datos entre aplicaciones y entre organizaciones.

- SOAP

Es un protocolo de mensajería construido en XML, que se usa para codificar información de los requerimientos de los Web Services. Los mensajes SOAP son independientes de los sistemas operativos y pueden ser transportados por los protocolos que funcionan en Internet: SMTP, HTTP, etc. (Saffirio, 2006)

- **WSDL**

Es un lenguaje que está basado en XML y que permite la descripción de los servicios web desplegados, es decir no es más que un documento XML que describe ciertas características propias de un servicio web, así como su localización, parámetros y métodos que soportan (Piñol., 2001).

1.5.7 Spring Security

Es un framework de seguridad para las aplicaciones basadas en Spring. Proporciona una solución completa a la seguridad al implementar servicios de autenticación y autorización a recursos, tanto a nivel de peticiones Web como a nivel de invocación de métodos. Sobre la base de Spring, Spring Security aprovecha al máximo las técnicas de Inyección de Dependencia. Posee seguridad basada en roles, en listas de control de acceso, etc. Es capaz de gestionar seguridad en varios niveles: URLs que se solicitan al servidor, acceso a métodos, clases Java y acceso a instancias concretas de las clases (Walls, et al., 2008). En los Servicios Web del Sub-Sistema se utilizó la seguridad a nivel de métodos.

1.5.8 Servidor Web: Apache Tomcat

Es un servidor web que funciona como un contenedor de servlet. Implementa las especificaciones de los servlet y las Java Server Page (JSP). Apache Tomcat y las tecnologías afines, proporcionan a los programadores de Java un completo conjunto de herramientas para crear de forma rápida, sofisticadas aplicaciones web. Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache Software Licence.

Puede ser ejecutado en cualquier sistema operativo que cuente con una la Máquina Virtual de Java (JVM), debido a que fue escrito utilizando esta plataforma para su adecuado funcionamiento. Es fiable, resulta realmente sencillo de instalar con pocos requerimientos de capacidad en disco, es compatible con las

APIs más recientes de Java y permite desplegar aplicaciones relativamente en poco tiempo. Puede funcionar como servidor web en entornos con alto nivel de tráfico y alta disponibilidad.

1.5.9 Gestor de Base de Datos: PostgreSQL.

PostgreSQL es un Gestor de Base de Datos relacional libre, funciona en todos los sistemas operativos importantes, incluyendo Linux, UNIX y Windows, incluye la mayoría de los tipos de datos como son integer, numeric, boolean, char, varchar, date e interval.

Es ampliamente considerado como una de las alternativas de sistemas de Bases de Datos de código abierto. El código fuente está disponible para todos sin costo, permitiendo personalizarlo o extenderlo. Ofrece varios métodos de bloqueos para evitar el acceso concurrente a los datos en las tablas.

Este sistema gestor posee una serie de características positivas respecto a otros, por ejemplo:

- Posee una gran escalabilidad.
- Es capaz de ajustarse al número de procesadores y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta.
- Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.
- Aproxima los datos a un modelo objeto-relacional y es capaz de manejar complejas rutinas y reglas.
- Ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características, estabilidad y rendimiento (Espinoza, 2005).

1.5.10 Sistema de Control de Versiones: SVN

Un sistema de control de versiones es un sistema de gestión de archivos y directorios, cuya principal característica es que mantiene la historia de los cambios y modificaciones que se han realizado sobre ellos a lo largo del tiempo. De esta forma, el sistema es capaz de “recordar” las versiones antiguas de los

datos, lo que permite examinar el histórico de cambios o recuperar versiones anteriores de un fichero, incluso aunque haya sido borrado.

Los sistemas de control de versiones son ampliamente utilizados en los proyectos de desarrollo de software, para mantener las versiones del código fuente. No obstante, su aplicación no está limitada a esta actividad, sino que permiten gestionar documentos, imágenes y ficheros de todo tipo (García, 2008).

1.5.11 Interfaz de Programación de Aplicaciones (API)

Una Interfaz de Programación de Aplicaciones es un conjunto de funciones o métodos usados para acceder a cierta funcionalidad. La interfaz se encarga de mantener el diálogo con los datos para poder tener acceso a ellos y manipularlos (Lamarca, 2009).

1.5.11.1 API SAX

SAX es una API de XML con un parser de acceso secuencial propio, diseñada para proporcionar un mecanismo de lectura de datos en documentos XML basado en eventos. Surge como alternativa al modelo DOM.

A diferencia de DOM, que genera un árbol jerárquico en memoria, SAX procesa la información en XML conforme esta sea presentada (evento por evento), manipulando cada elemento a un determinado tiempo, sin incurrir en un uso excesivo de memoria.

La utilización de SAX como API de desarrollo en XML brinda las siguientes ventajas:

- SAX es ideal para manipular archivos de gran tamaño, ya que no requiere generar un árbol en memoria como es el caso de DOM.
- Es más rápido y sencillo que utilizar DOM.

La sencillez antes mencionada tiene su precio, puesto que SAX funciona por eventos y no es posible manipular información una vez procesada. En DOM no existe esta limitación, ya que se genera el árbol jerárquico en memoria y es posible recorrer la estructura de nuevo en cualquier momento (Jiménez, et al., 2008).

1.5.12 Metodología de Desarrollo de Software.

Una Metodología de Desarrollo de Software es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un software.

Constituye una forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo). Pretende implementar las mejores prácticas en Ingeniería de Software, desarrollo iterativo, administración de requisitos, control de cambios, modelado visual del software, verificación de la calidad del software, etc.

1.5.12.1 RUP

RUP es una metodología orientada al proceso de desarrollo de sistemas de Software, que junto con el lenguaje de especificación UML constituyen las herramientas más utilizadas para el análisis, diseño, implementación y documentación de Sistemas Orientados a Objetos (Ramírez, 2009). Está pensado para proyectos y equipos grandes o muy largos, genera gran cantidad de documentación, es fácilmente adaptable al ambiente de desarrollo de software deseado, por lo que puede considerarse como una metodología tradicional.

Beneficios que aporta RUP:

- Permite desarrollar aplicaciones sacando el máximo provecho de las nuevas tecnologías, mejorando la calidad, el rendimiento, la reutilización, la seguridad y el mantenimiento del software mediante una gestión sistemática de los riesgos.
- Proporciona guías explícitas para áreas tales como modelado de negocios, arquitectura Web, etc.
- Unifica todo el equipo de desarrollo de software y mejora la comunicación al brindar a cada miembro del mismo una base de conocimientos, un lenguaje de modelado y un punto de vista de cómo desarrollar software.

RUP tiene como objetivo guiar a los desarrolladores en la implementación y distribución eficiente de sistemas que se ajusten a las necesidades de los clientes.

1.5.13 Lenguaje de modelado: UML

UML es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, mantener y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar (Rumbaugh, et al., 1998).

1.5.14 Herramienta CASE: Visual Paradigm

Visual Paradigm es una poderosa herramienta CASE para el modelado. Es la herramienta por excelencia para ser utilizada en un ambiente de software libre. Permite crear tipos diferentes de diagramas en un ambiente totalmente visual. Es muy sencillo de usar, fácil de instalar y actualizar. Presenta un entorno de creación de diagramas para UML. Posibilita la representación gráfica de los diagramas permitiendo ver el sistema desde diferentes perspectivas, como el de componentes, despliegue, secuencia, casos de uso, clase, actividad, estado, entre otros.

1.5.15 Hibernate

Hibernate es un framework que realiza el mapping entre el mundo orientado a objetos de las aplicaciones y el mundo entidad-relación de las bases de datos en entornos Java. El término utilizado es ORM (object/relational mapping) y consiste en la técnica para realizar la transición de una representación de los datos de un modelo relacional a un modelo orientado a objetos y viceversa.

Hibernate no solo realiza esta transformación, sino que proporciona capacidades para la obtención y almacenamiento de datos de la base de datos reduciendo el tiempo de desarrollo. Es un generador de sentencias SQL, relaciones, colecciones y un gran número de tipos de datos. De una manera muy rápida y optimizada, puede generarse la base de datos en cualquiera de los entornos soportados (Crespo, 2004).

1.5.16 Especificaciones: Java Persistence API (JPA)

JPA proporciona un estándar para gestionar datos relacionales en aplicaciones Java EE, de forma que además se simplifique el desarrollo de la persistencia de datos. En su definición, ha combinado ideas y conceptos de los principales frameworks de persistencia, como Hibernate, Toplink, JDO, entre otros. Todos estos cuentan actualmente con una implementación JPA.

El mapeo objeto-relacional (es decir, la relación entre entidades Java y tablas de la base de datos) se realiza mediante anotaciones en las propias clases de entidad. No se requieren ficheros descriptores XML. También pueden definirse transacciones como anotaciones JPA (2008).

1.6. Conclusiones

En este capítulo se realizó un estudio de algunos sistemas existentes que brindan SVA para dispositivos móviles, abordándose conceptos relacionados con los mismos y determinándose que la tendencia a nivel global de los SVA fundamentalmente es hacia la descarga de contenidos. Por último se describen brevemente las herramientas y tecnologías que serán utilizadas en la creación del Sub-Sistema propuesto.

Capítulo II: Características del sistema

2.1 Introducción

En este capítulo se hace una descripción de la problemática existente, abordando además las características del Sub-Sistema que será desarrollado. Debido a que no están bien definidos los procesos de negocio, se reflejan los principales conceptos tratados y las relaciones entre ellos mediante un Modelo de Dominio o Modelo Conceptual. También se plantean las características que el Sub-Sistema debe cumplir y tener, es decir, los requisitos funcionales y no funcionales, además de los casos de uso del sistema y los actores del mismo.

2.2 Propuesta del sistema

Se propone el desarrollo de un Sub-Sistema que posibilite brindar los Servicios de Valor Agregado de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas, los cuales representan los módulos de la aplicación.

Cada uno de estos módulos presenta una Arquitectura de Tres Capas: la capa de servicios, la capa de acceso a datos y la capa de negocio. La capa de servicios proporciona interoperabilidad entre el Sub-Sistema y los portales WAP, Web y de Administración a través de Servicios Web utilizando Spring-WS. Esta capa está dividida en dos grupos de servicios, los que son consumidos por los portales Web y WAP que no presentan seguridad y los servicios que son consumidos por el Portal de Administración, que si tienen seguridad. La seguridad se aplicó a nivel de método, mediante el acceso a recursos utilizando Spring Security. Para tener acceso a estos recursos se utilizó la política de seguridad de la autenticación (usuario-contraseña). La lógica de los servicios se implementa en la capa de negocio. La actualización automática cada un tiempo determinado de los servicios del Estado del Tiempo y Noticias se realiza a partir de un canal RSS obtenido de Internet y su automatización la garantiza un Cron (estructura temporalizadora), el cual queda iniciado desde el arranque del sistema. Para la implementación de la capa de acceso a datos se utilizó el Framework de persistencia Hibernate con una implementación de JPA. La base de datos está montada en un servidor de PostgreSQL.

2.3 Modelo de Dominio

Como los procesos de negocio no están bien definidos y algunos todavía no tienen claras las actividades que desarrollan ni los involucrados en las mismas, se imposibilita realizar un modelo de Negocio, haciéndose necesario la realización de un Modelo de Dominio.

Un Modelo de Dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las "cosas" que existen o los eventos que suceden en el entorno donde trabaja el sistema (Rumbaugh, et al., 2000), mientras que un Modelo de Negocio describe los procesos que se desarrollan, quiénes participan en ellos y cómo interactúan entre sí.

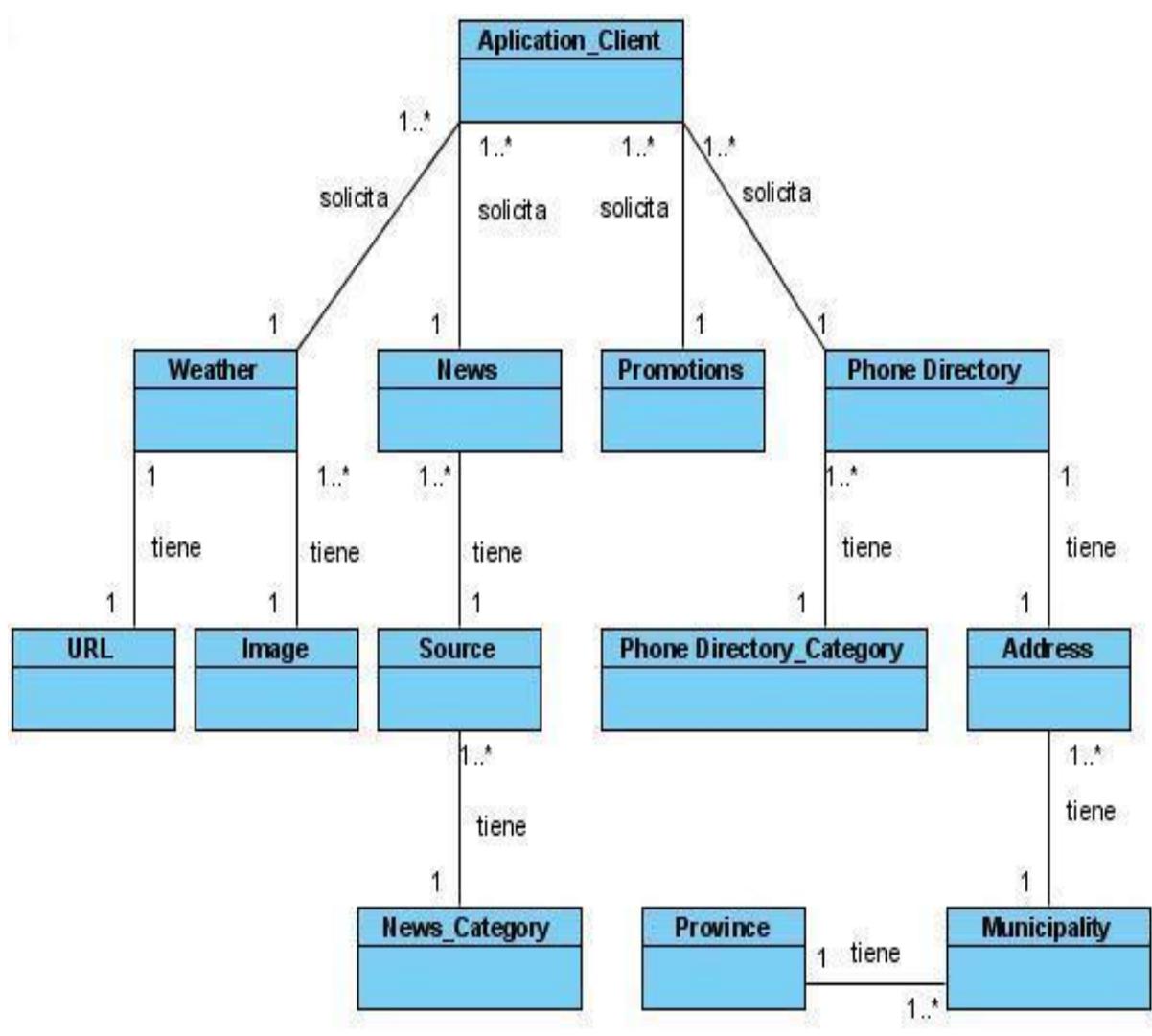


Figura 2: Modelo de Dominio.

Descripción de los objetos del dominio.

Application_Client: Solicita los servicios que ofrece el Sub-Sistema.

News: Representa los datos referentes a las Noticias.

Source: Representa los datos de una Fuente de Noticias de un Canal RSS.

News_Category: Representa la Categoría de una Noticia.

Promotions: Representa los datos referentes a las Promociones.

Weather: Representa los datos referentes al Estado del Tiempo.

URL: Representa la URL Fuente que proporciona el Estado del Tiempo de un Canal RSS.

Image: Representa la Imagen asociada a un Estado del Tiempo determinado.

Phone Directory: Representa los datos referentes al Directorio Telefónico de Páginas Amarillas.

Phone Directory_Category: Representa la Categoría de una Entrada Telefónica.

Address: Representa la Dirección de una Entrada Telefónica determinada.

Municipality: Representa el Municipio de la Dirección de una Entrada Telefónica determinada.

Province: Representa la Provincia de la Dirección de una Entrada Telefónica determinada.

2.4 Requerimientos del Sistema

2.4.1 Requisitos Funcionales

RF1 Adicionar Promoción

Permitirá adicionar al sistema una nueva Promoción.

RF2 Modificar Promoción

Permitirá modificar los datos de una Promoción en el sistema.

RF3 Eliminar Promoción

Permitirá eliminar una Promoción del sistema.

RF4 Adicionar Categoría de Noticias

Permitirá adicionar al sistema una nueva Categoría de Noticias.

RF5 Modificar Categoría de Noticias

Permitirá modificar los datos de una Categoría de Noticias existente en el sistema.

RF6 Eliminar Categoría de Noticias

Permitirá eliminar una Categoría de Noticias del sistema.

RF7 Buscar Categorías de Noticias

Permitirá buscar todas las Categorías de Noticias almacenadas en el sistema.

RF8 Adicionar Fuente de Noticias

Permitirá adicionar una nueva Fuente de Noticias al sistema.

RF9 Eliminar Fuente de Noticias

Permitirá eliminar una Fuente de Noticias del sistema.

RF10 Modificar Fuente de Noticias

Permitirá modificar los datos de una Fuente de Noticias existente en el sistema.

RF11 Modificar URL Fuente del Estado del Tiempo

Permitirá modificar los datos de una URL Fuente del Estado del Tiempo en el sistema.

RF12 Mostrar URL Fuente del Estado del Tiempo

Permitirá mostrar una URL Fuente del Estado del Tiempo existente en el sistema.

RF13 Buscar Noticias por Categoría

Permitirá buscar todas las Noticias dada una Categoría de Noticias determinada.

RF14 Mostrar Estado del Tiempo

Permitirá mostrar el Estado del Tiempo almacenado en el sistema.

RF15 Adicionar Entrada Telefónica

Permitirá adicionar una nueva Entrada Telefónica en el sistema.

RF16 Eliminar Entrada Telefónica

Permitirá eliminar una Entrada Telefónica del sistema.

RF17 Modificar Entrada Telefónica

Permitirá modificar los datos de una Entrada Telefónica en el sistema.

RF18 Buscar Categoría de Entrada Telefónica

Permitirá buscar todas las Categorías de Entradas Telefónicas almacenadas en el sistema.

RF19 Buscar Provincias

Permitirá buscar todas las Provincias almacenadas en el sistema.

RF20 Buscar Provincias por Categoría de Entrada Telefónica

Permitirá buscar todas las Provincias dada una Categoría de Entrada Telefónica determinada.

RF21 Buscar Municipios

Permitirá buscar todos los Municipios almacenados en el sistema.

RF22 Buscar Municipios por Provincia

Permitirá buscar los Municipios de una Provincia determinada.

RF23 Buscar Municipios por Categoría de Entrada Telefónica y Provincia

Permitirá buscar los Municipios dada una Provincia y una Categoría de Entrada Telefónica determinada.

RF24 Buscar Entradas Telefónicas por nombre

Permitirá buscar todas las Entradas Telefónicas dado un fragmento de nombre y además dará la posibilidad de buscar también por Provincia y Municipio.

RF25 Buscar Entradas Telefónicas por Categoría de Entrada Telefónica

Permitirá buscar las Entradas Telefónicas dada una Categoría de Entrada Telefónica determinada y dará la posibilidad además, de buscar también por Provincia y Municipio.

RF26 Actualizar Noticias

Permitirá actualizar todas las Noticias en el sistema.

RF27 Actualizar Estado del Tiempo

Permitirá actualizar el Estado del Tiempo en el sistema.

RF28 Buscar Promoción

Permitirá buscar todas las Promociones almacenadas en el sistema.

RF29 Buscar Noticias

Permitirá buscar todas las Noticias almacenadas en el sistema.

2.4.2 Requisitos No Funcionales

Soporte

RNF1 Elaborar un manual para los usuarios en el cual se encontrarán los aspectos relacionados con el sistema. Este manual también tratará las definiciones principales del sistema, el trabajo con los archivos que se generan, los LOGs y los posibles errores que pudieran ocurrir, especificando las posibles soluciones del mismo.

RNF2 Brindar soporte a todos los usuarios que necesiten capacitación para operar el sistema, en dependencia de las demandas del comprador, pueden ser mantenimiento, entrenamiento de personal, integración con otro software u operar directamente el sistema.

Portabilidad

RNF3 El Sub-Sistema debe ser multiplataforma.

RNF4 El Sub-Sistema debe ser desarrollado de forma modular.

Restricciones en el diseño y la implementación

RNF5 Utilizar para la implementación el lenguaje de programación: JAVA.

RNF6 Utilizar como herramienta CASE: Visual Paradigm con soporte para UML.

Usabilidad

RNF7 El Sub-Sistema debe ser escalable para poder agregar nuevos servicios sin afectar los que ya se encuentran en funcionamiento.

Seguridad

RNF8 Registrar una cantidad finita de eventos anormales del sistema en LOGS, para poder desarrollar una auditoría.

RNF9 Garantizar el tratamiento adecuado de las excepciones.

RNF10 Proporcionar a los servicios web de administración seguridad a nivel de método utilizando como política de seguridad la autenticación, dándole acceso solo a los Administradores.

Software

RNF11 Se requiere la presencia del Servidor Web Apache Tomcat.

RNF12 Utilizar la Máquina Virtual de Java para el adecuado funcionamiento del sistema.

Apariencia o Interfaz interna

RNF13 La implementación de las funcionalidades deben seguir los lineamientos de código establecidos por la Dirección del Proyecto, para garantizar la comprensión del código por desarrolladores en futuras versiones.

RNF14 Publicar los posibles servicios ofrecidos por el Sub-Sistema en Web Services para ser accedidos por el resto de las aplicaciones y adaptarse a cualquier sistema que implementen los operadores de telefonía móvil.

2.5 Modelo de Casos de Uso del Sistema

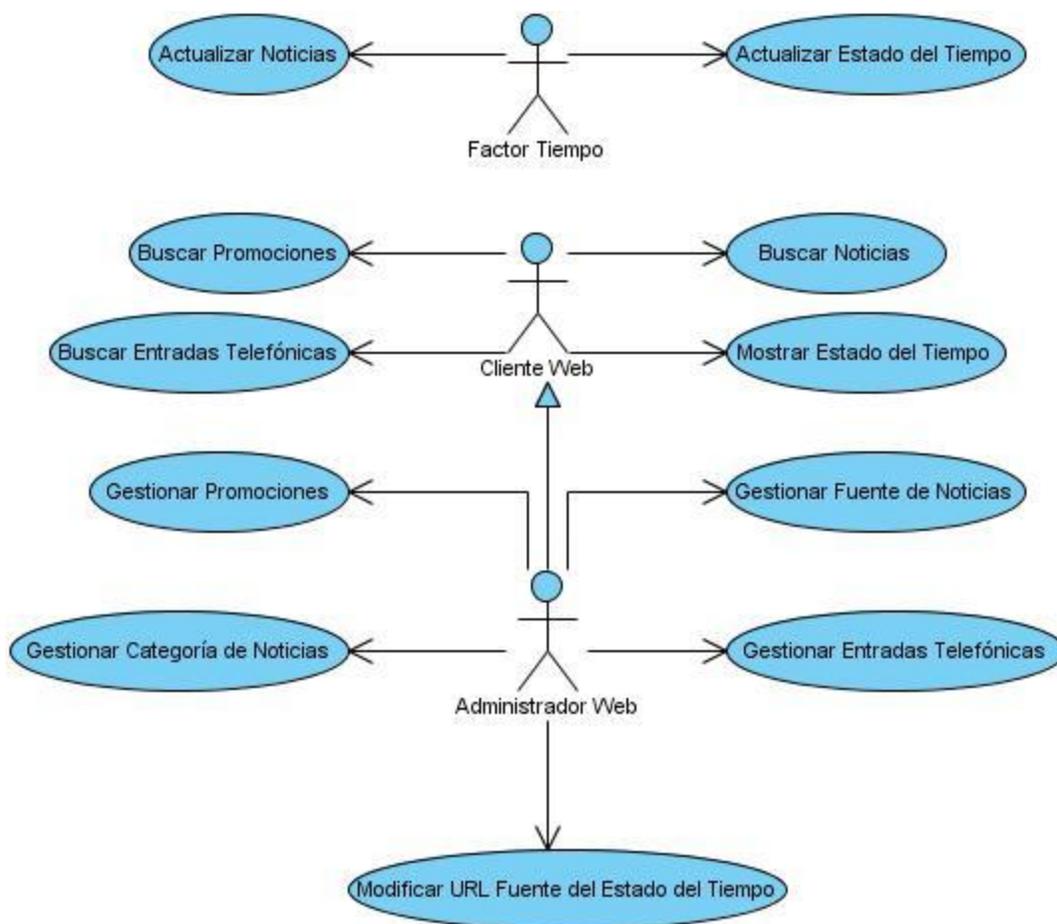


Figura 3: Modelo de Casos de Uso del Sistema.

2.5.1 Definición de actores del sistema a automatizar.

Actor	Descripción
Cliente Web	Solicita los servicios de Promociones, Noticias, Estado del Tiempo y Directorio Telefónico de Páginas Amarillas al sistema.
Administrador Web	Gestiona los datos referentes a los servicios de Promociones, Noticias, Estado del Tiempo y Directorio Telefónico de Páginas Amarillas en el sistema.
Factor Tiempo	Realiza la actualización automática de los servicios de Noticias y Estado del Tiempo.

2.5.2 Breve descripción de los Casos de Uso

Caso de Uso:	Gestionar Entradas Telefónicas
Actores:	Administrador Web
Resumen:	Permite adicionar, modificar y eliminar una Entrada Telefónica.
Precondiciones:	-
Referencias:	RF15,RF16,RF17
Prioridad:	media

Caso de Uso:	Gestionar Promociones
Actores:	Administrador Web
Resumen:	Permite adicionar, modificar y eliminar una Promoción.
Precondiciones:	-
Referencias:	RF1,RF2,RF3
Prioridad:	media

Caso de Uso:	Gestionar Fuente de Noticias
Actores:	Administrador Web
Resumen:	Permite adicionar, modificar y eliminar una Fuente de Noticias.
Precondiciones:	-
Referencias:	RF8,RF9,RF10
Prioridad:	media

Caso de Uso:	Gestionar Categoría de Noticias
Actores:	Administrador Web
Resumen:	Permite adicionar, modificar y eliminar una Categoría de Noticias.
Precondiciones:	-
Referencias:	RF4,RF5,RF6
Prioridad:	media

Caso de Uso:	Modificar URL Fuente del Estado del Tiempo
Actores:	Administrador Web
Resumen:	Permite mostrar y modificar una URL.
Precondiciones:	-
Referencias:	RF12,RF13
Prioridad:	media

Caso de Uso:	Buscar Promociones
Actores:	Cliente Web
Resumen:	Permite buscar las Promociones.
Precondiciones:	-
Referencias:	RF28
Prioridad:	media

Caso de Uso:	Buscar Entradas Telefónicas
Actores:	Cliente Web
Resumen:	Permite buscar las Entradas Telefónicas por un criterio de búsqueda.
Precondiciones:	-
Referencias:	RF18,RF19,RF20,RF21,RF22, RF23,RF24, RF25
Prioridad:	media

Caso de Uso:	Mostrar Estado del Tiempo
Actores:	Cliente Web
Resumen:	Permite mostrar el Estado del Tiempo existente en el sistema.
Precondiciones:	-
Referencias:	RF14
Prioridad:	media

Caso de Uso:	Buscar Noticias
Actores:	Cliente Web
Resumen:	Permite mostrar las Noticias por un criterio de búsqueda.
Precondiciones:	-
Referencias:	RF7,RF13,RF29
Prioridad:	media

Caso de Uso:	Actualizar Estado del Tiempo
Actores:	Factor Tiempo
Resumen:	Permite la actualización automática cada cierto tiempo.
Precondiciones:	-
Referencias:	RF27
Prioridad:	media

Caso de Uso:	Actualizar Noticias
Actores:	Factor Tiempo
Resumen:	Permite la actualización automática cada cierto tiempo.
Precondiciones:	-
Referencias:	RF27
Prioridad:	Media

2.6 Conclusiones

En este capítulo se definió el Modelo de Dominio, en el cual se representaron los conceptos fundamentales dentro del contexto del sistema. Además se realizó el Diagrama de Casos de Uso modelando las funcionalidades y actores del futuro sistema, llegando a un acuerdo con el cliente sobre las condiciones y funcionalidades que debe cumplir el mismo.

Capítulo III: Análisis y Diseño del sistema

3.1 Introducción

Este capítulo tiene como objetivo realizar un análisis de los requisitos funcionales definidos en el capítulo anterior y cómo estructurarlos para obtener una mejor comprensión de los mismos. Además se profundizará en los aspectos relacionados con los requisitos no funcionales, lenguajes de programación, componentes reutilizables, tecnologías, entre otros. También se examinará brevemente la arquitectura del Sub-Sistema.

3.2 Análisis

3.2.1 Definición de Modelo de Análisis.

El modelo de análisis contiene las clases de análisis. Puede ser un producto de trabajo temporal, como cuando evoluciona hacia un modelo de diseño o puede permanecer durante algunos o todos los proyectos y quizás más, sirviendo como visión general conceptual del sistema.

3.2.2 Clases del Análisis

Una clase del análisis representa una abstracción de una o varias clases y/o subsistema del diseño. Se centra en el tratamiento de los requisitos funcionales y pospone los no funcionales, denominándolos requisitos especiales, hasta llegar a las actividades de diseño e implementación subsiguientes (Rumbaugh, et al., 2000).



Figura 4: CU Gestionar Promociones.



Figura 5: CU Buscar Promociones.

3.3 Diseño

3.3.1 Definición de Modelo de Diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además, el modelo de diseño sirve de abstracción de la implementación del sistema y es, de ese modo, utilizado como entrada fundamental de las actividades de implementación (Rumbaugh, et al., 2000).

3.3.2 Clases del Diseño

Una clase del diseño representa una abstracción de una o varias clases en la implementación del sistema, exactamente, a qué corresponde depende del lenguaje de implementación. Es una construcción similar en la implementación del sistema, es decir:

- El lenguaje utilizado para especificar una clase del diseño es lo mismo que el lenguaje de programación.
- Las relaciones de aquellas clases del diseño implicadas por otras clases, a menudo tienen un significado directo cuando la clase es implementada.
- Los métodos de una clase del diseño tienen correspondencia directa con el correspondiente método en la implementación de las clases.
- Una clase del diseño puede posponer el manejo de algunos requisitos para las siguientes actividades de implementación, indicándolos como requisitos de implementación de la clase.

- Una clase del diseño puede proporcionar interfaces si tiene sentido hacerlo en el lenguaje de programación (Rumbaugh, et al., 2000).

El diseño del Sub-Sistema está formado por cuatro módulos, promotions, news, phonedirectory y weather permitiendo así su mejor comprensión. Cada módulo posee varias clases de diseño e interfaces, que se encuentran fuertemente asociados según sus funciones. Por esta razón se realizó un Diagrama de Clases del Diseño por Módulos.

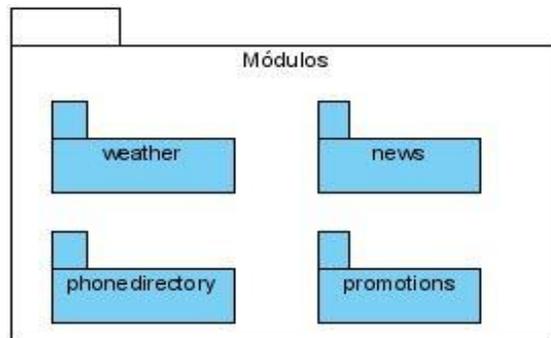


Figura 6: Módulos del Sub-Sistema.

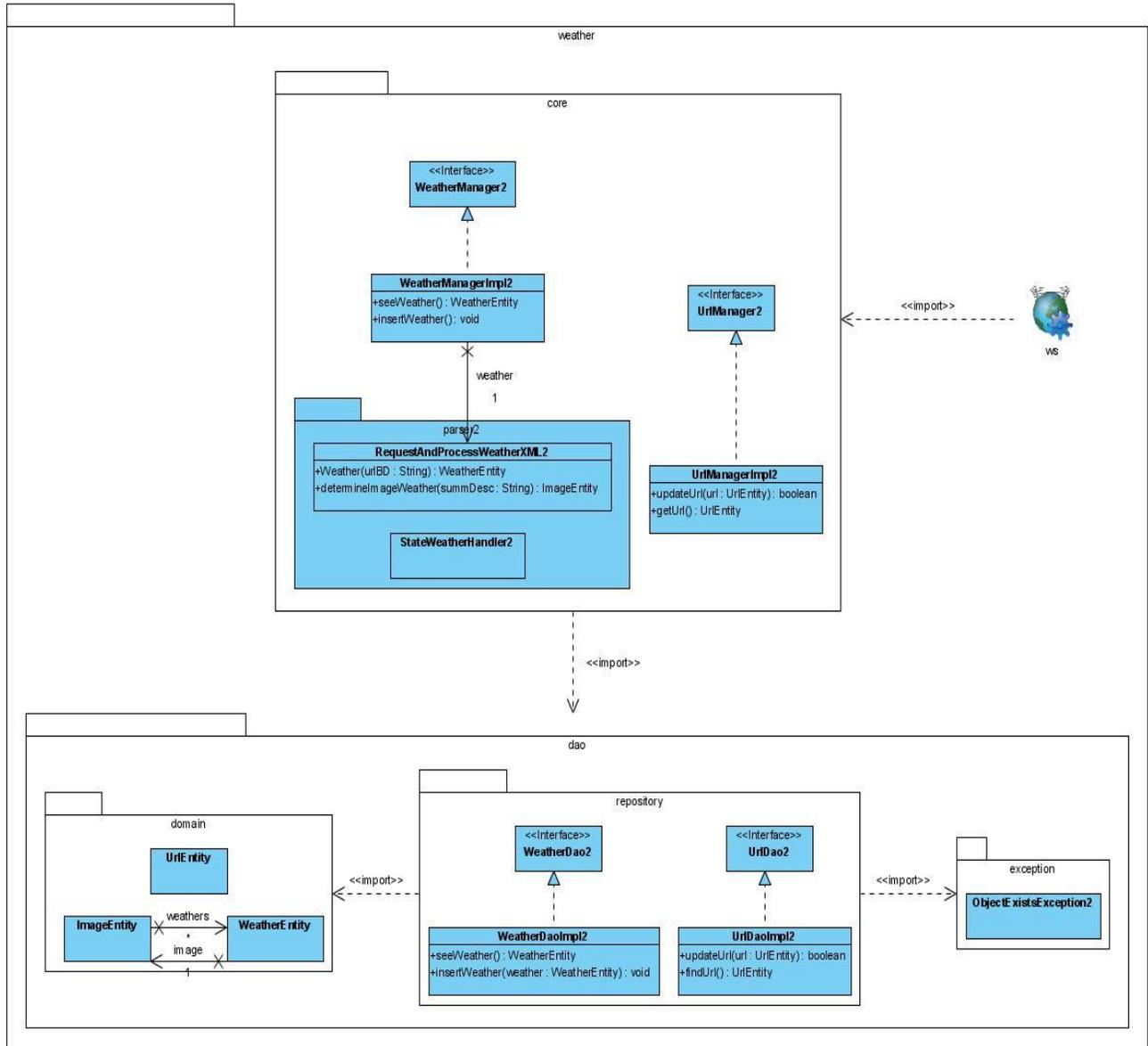


Figura 7: Diagrama de Clases del Diseño – Módulo weather.

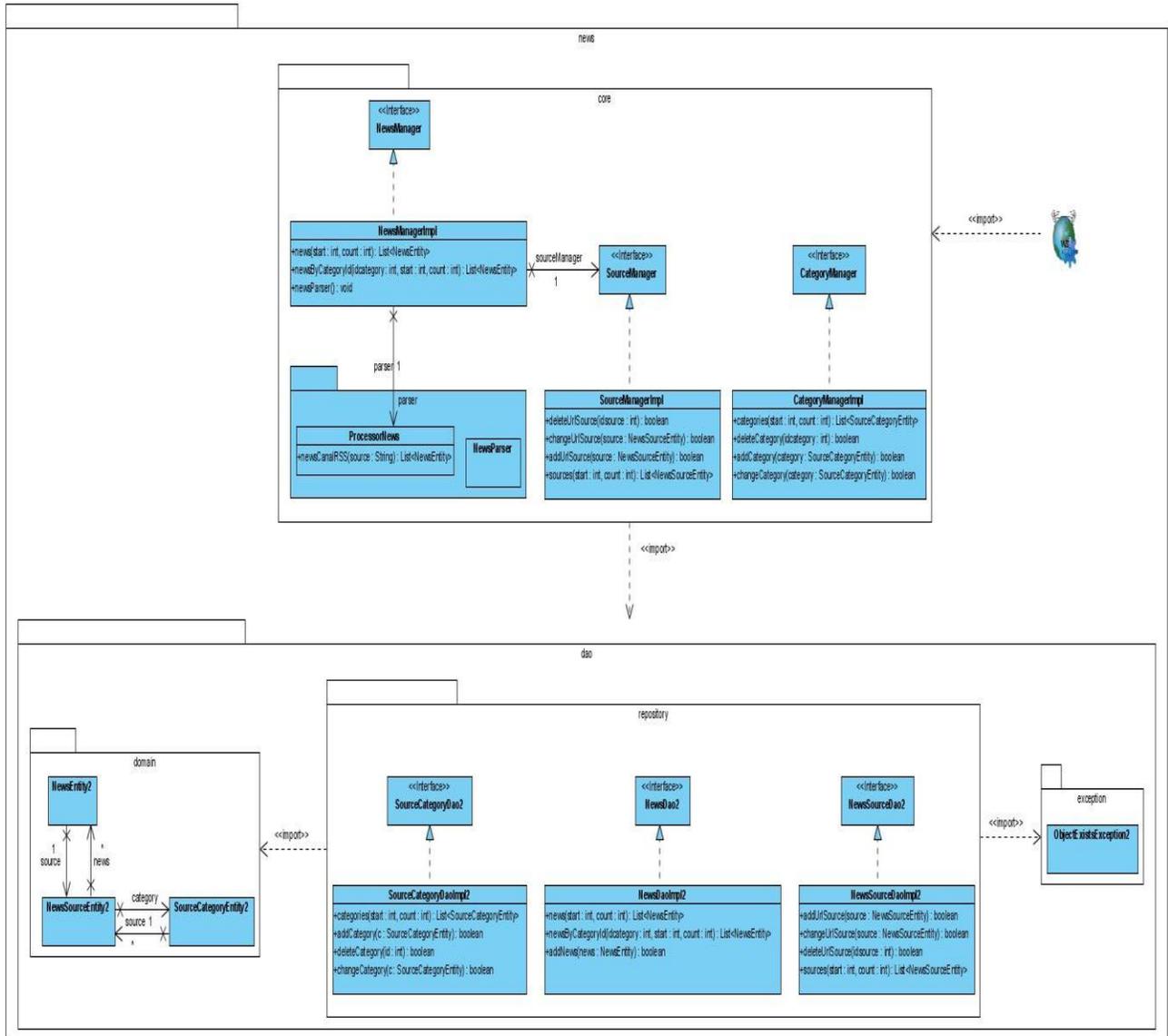


Figura 8: Diagrama de Clases del Diseño – Módulo news.

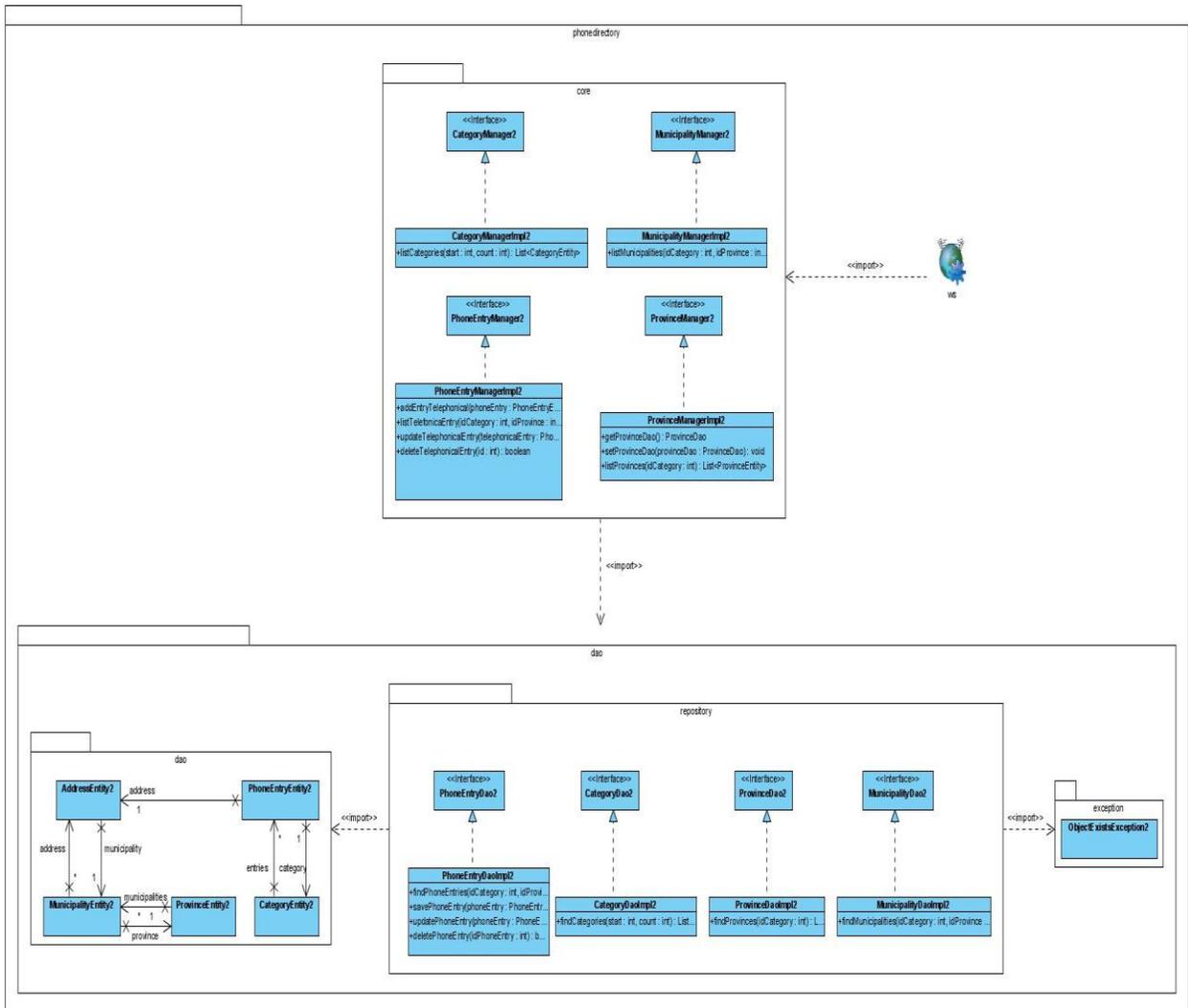


Figura 9: Diagrama de Clases del Diseño – Módulo phonedirectory.

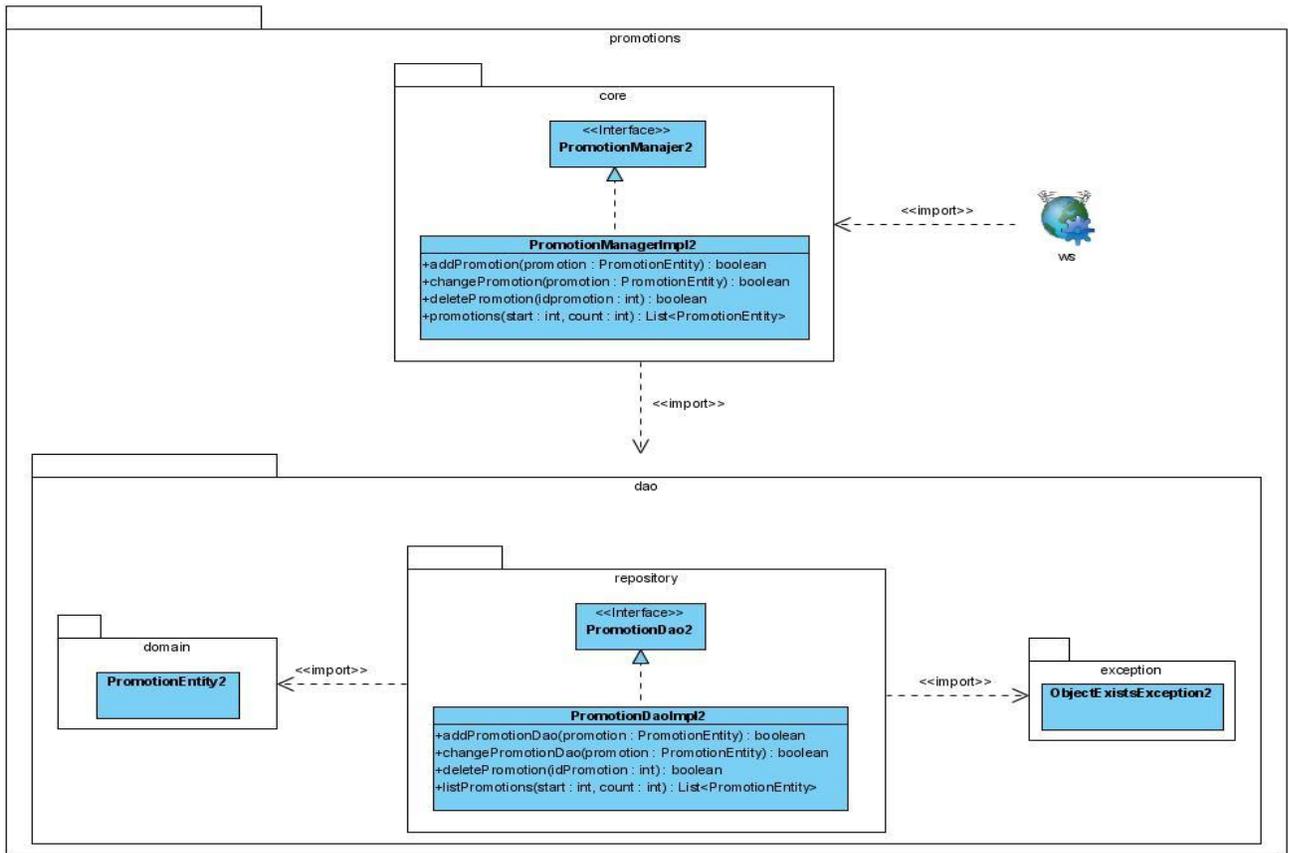


Figura 10: Diagrama de Clases del Diseño – Módulo promotions.



A continuación se mostrarán por cada uno de los Módulos del Sub-Sistema, un Diagrama de Clases del Diseño perteneciente al paquete ws.

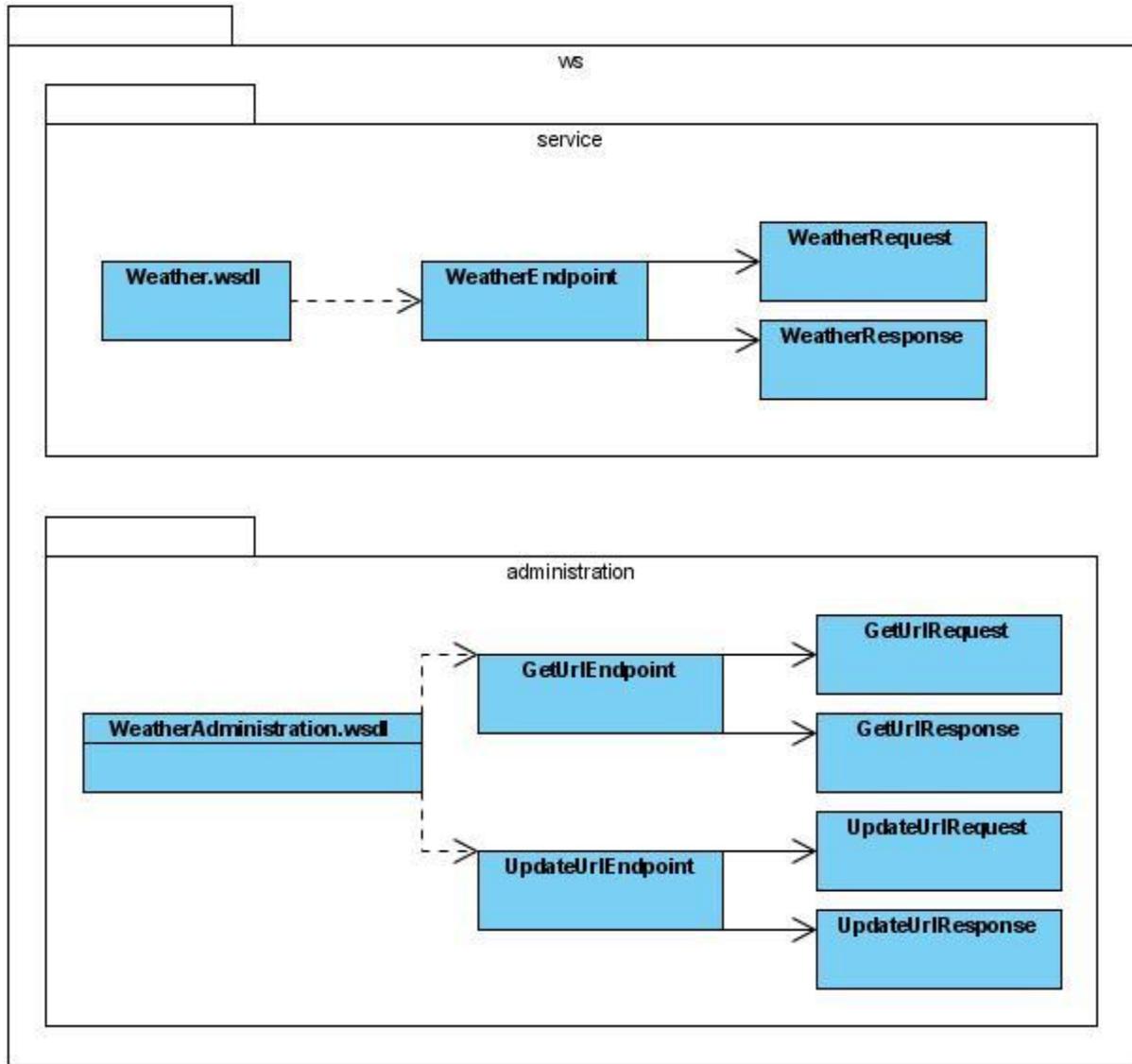


Figura 11: Diagrama de Clases del Diseño – Módulo weather (Paquete ws).

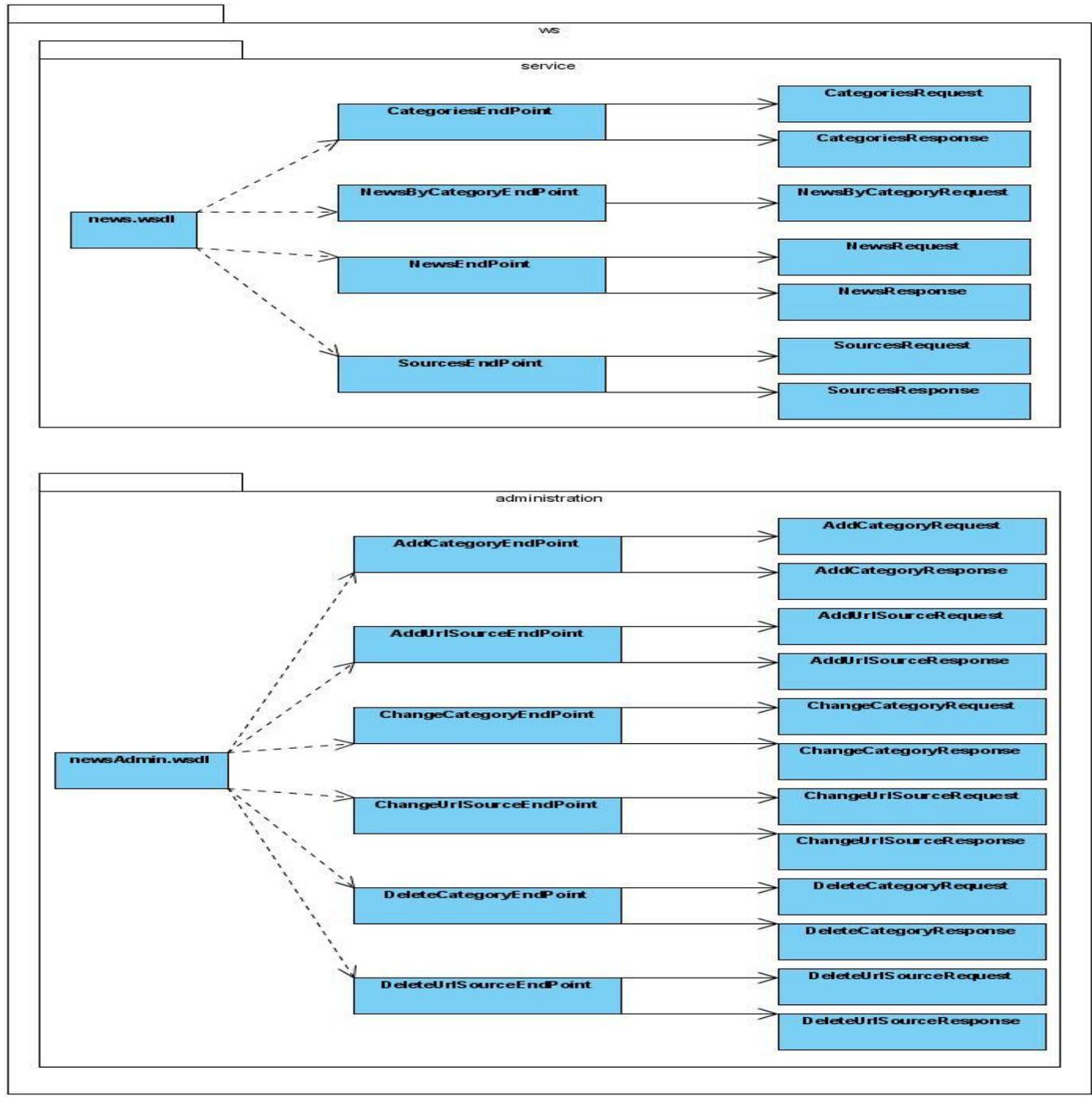


Figura 12: Diagrama de Clases del Diseño – Módulo news (Paquete ws).

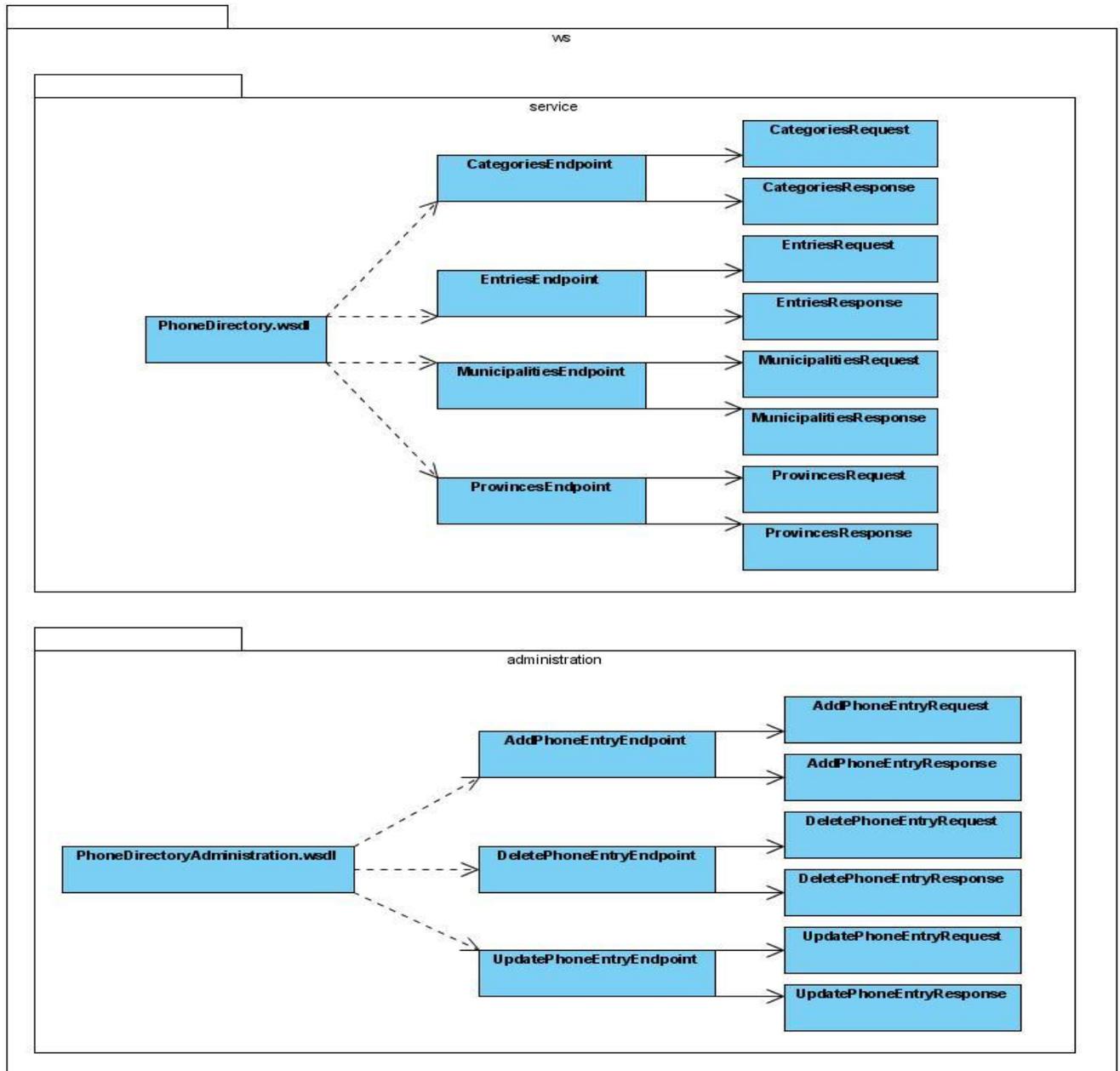


Figura 13: Diagrama de Clases del Diseño – Módulo phonedirectory (Paquete ws).

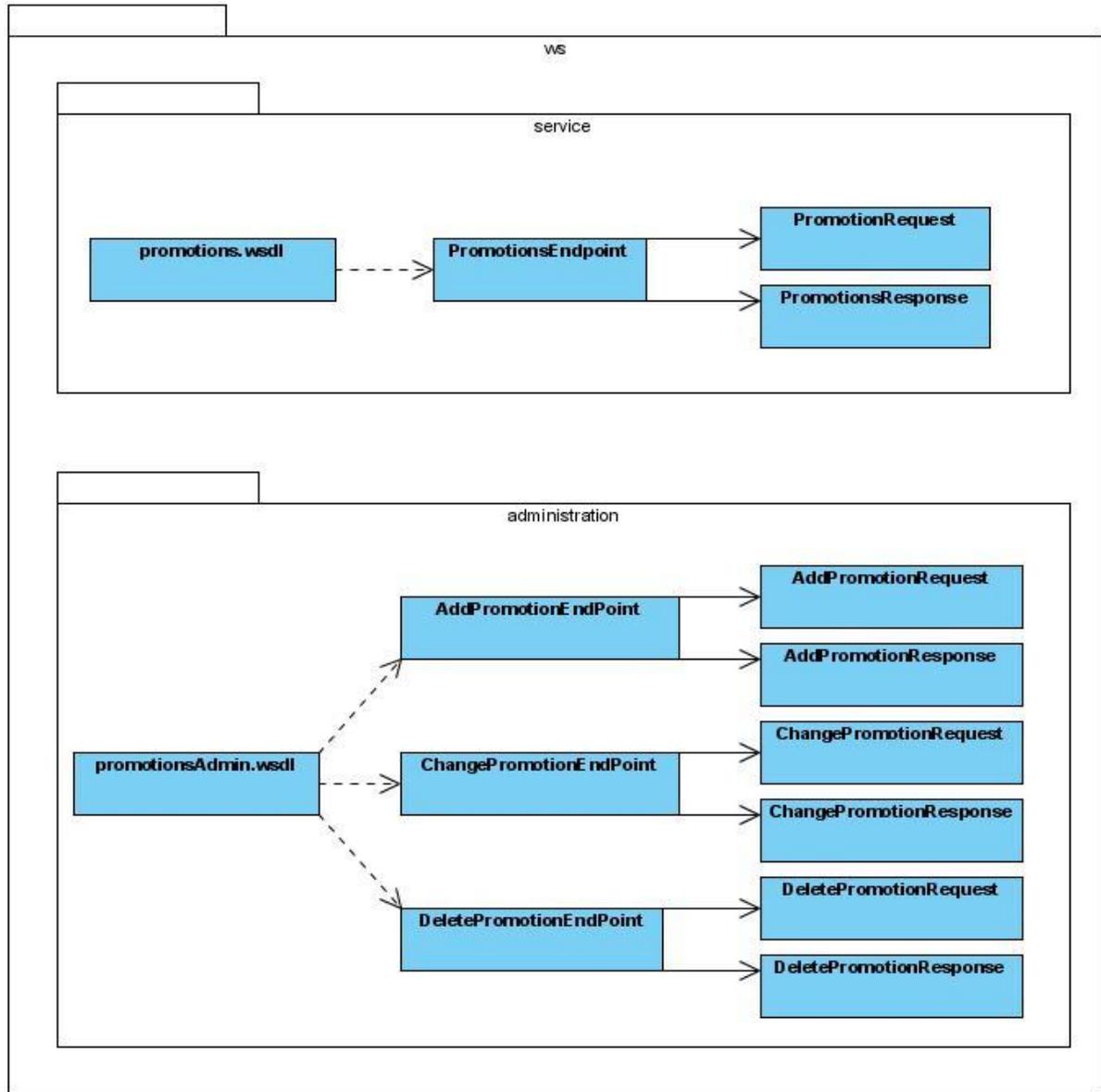


Figura 14: Diagrama de Clases del Diseño – Módulo promotions (Paquete ws).

3.4 Diagramas de Interacción

Los Diagramas de Interacción se utilizan para modelar los aspectos dinámicos del sistema. Consta de un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Existen dos tipos de Diagramas de Interacción: Secuencia y Colaboración. El primero destaca la ordenación temporal de los mensajes y el segundo la organización de los objetos que participan en una interacción.

Teniendo en cuenta la descripción textual de cada uno de los casos de uso en el capítulo anterior, se realizó un diagrama de colaboración por cada escenario del caso de uso Gestionar Promociones.

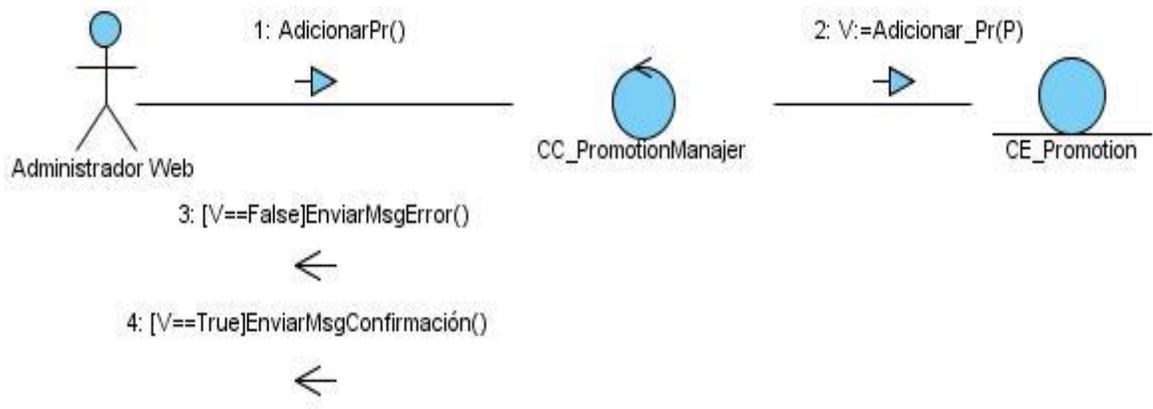


Figura 15: CU Gestionar Promociones. Escenario Adicionar Promoción.

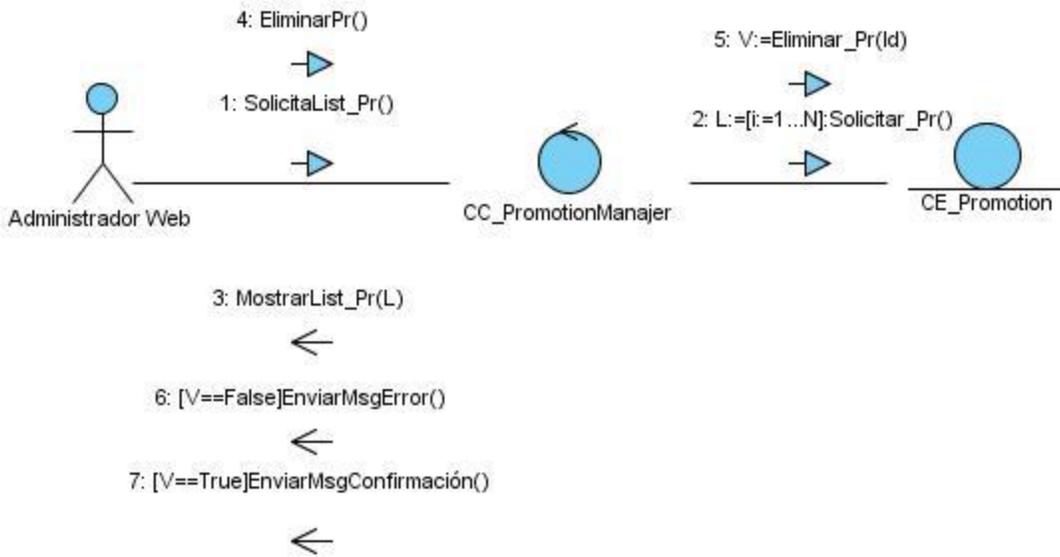


Figura 16: CU Gestionar Promociones. Escenario Eliminar Promoción.

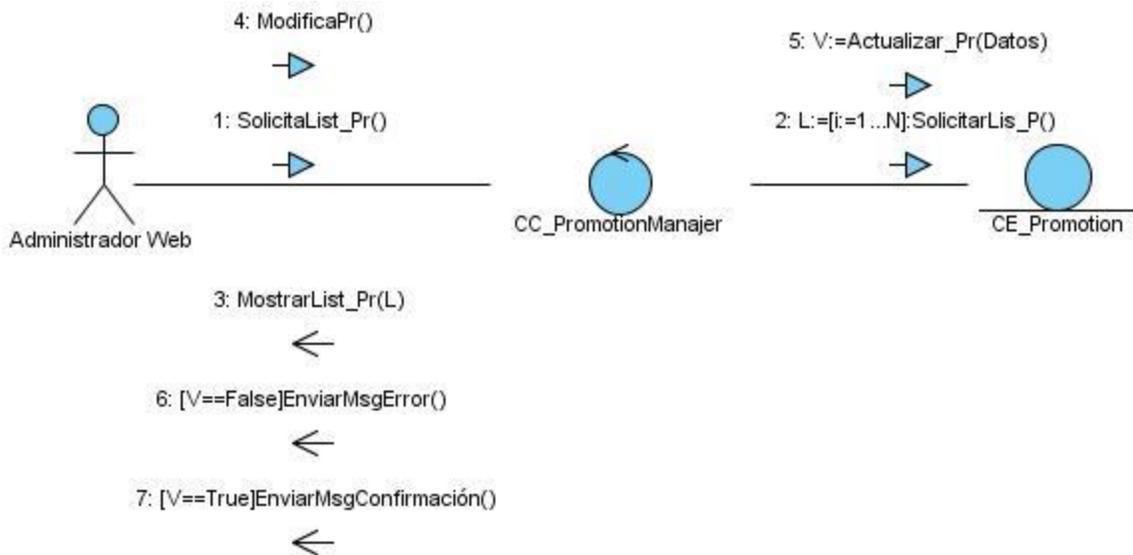


Figura 17: CU Gestionar Promociones. Escenario Actualizar Promoción.

3.5 Modelo de Datos

El Modelo de Datos describe las representaciones lógicas y físicas de datos persistentes utilizados por la aplicación. Permite definir una estructura que posibilita almacenar datos y recuperar información. La estructura tiene que estar en correspondencia con las necesidades de las aplicaciones que la usarán.

Responde a interrogantes importantes para el procesamiento de datos sobre las entidades que procesará el sistema, los atributos y las relaciones entre las mismas. Para responder estas interrogantes se realizan un conjunto de actividades que tienen como resultado primario un Modelo Lógico de Datos, a través de las Clases Persistentes y un Modelo Físico de Datos, mediante el Diagrama Entidad-Relación.

3.5.1 Modelo Lógico de Datos (Diagrama de Clases Persistentes).

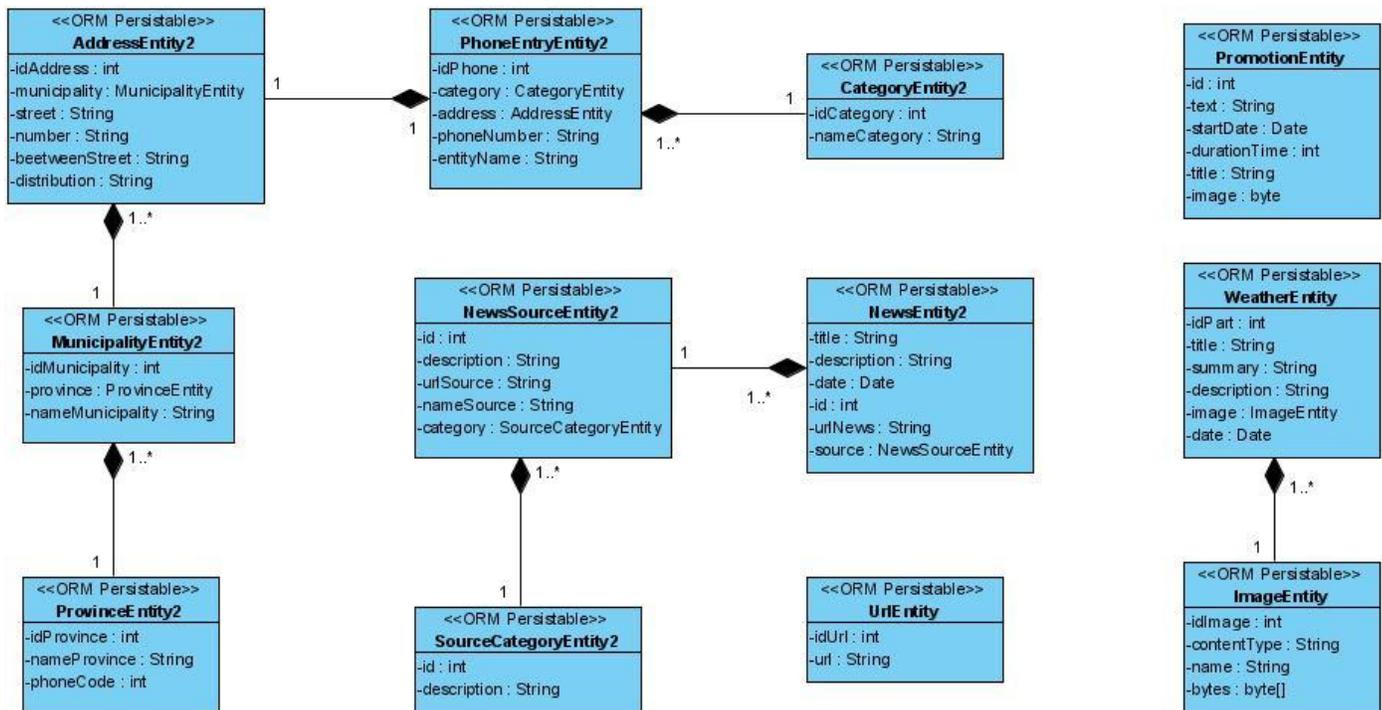


Figura 18: Diagrama de Clases Persistentes.

3.5.2 Modelo Físico de Datos (Diagrama Entidad-Relación).

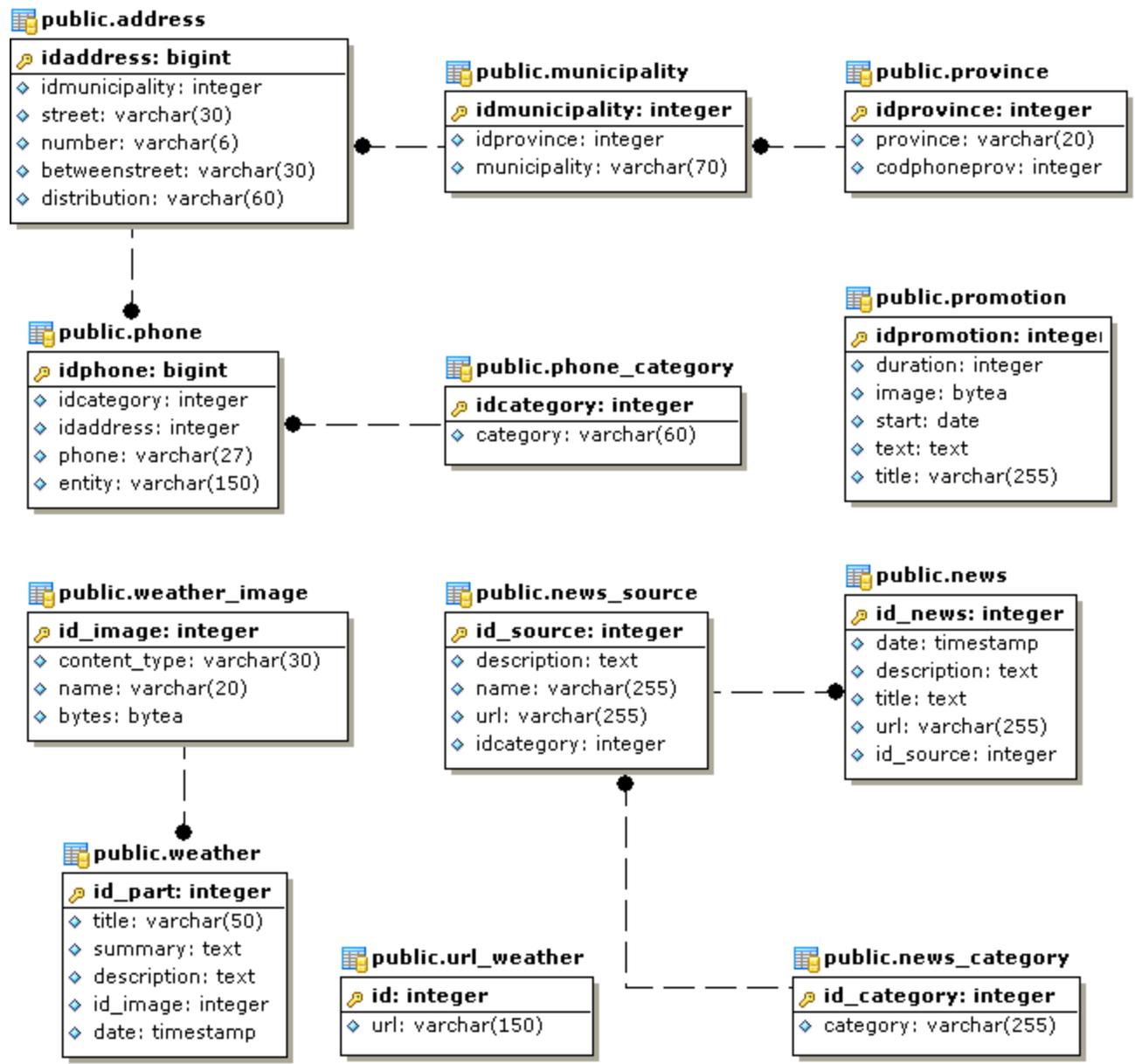


Figura 19: Diagrama Entidad-Relación.

3.6 Descripción de la Arquitectura

3.6.1 Patrones Arquitectónicos

La Arquitectura en Capas define cómo organizar el modelo de diseño a través de capas, que pueden estar físicamente distribuidas, lo cual quiere decir que los componentes de una capa solo pueden hacer referencia a componentes en capas inmediatamente inferiores. Este patrón es importante porque simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Además, nos ayuda a identificar qué puede reutilizarse. Cuando se trabaja en capas, se pueden usar diferentes niveles o cantidad de capas (2 o más capas), en este caso se utilizó para el desarrollo del Sub-Sistema una **Arquitectura de tres capas**, donde la aplicación se divide en tres capas lógicas distintas cada una de ellas con un grupo de interfaces perfectamente definidas. La primera capa se denomina capa de servicios, la capa de negocio y la tercera capa de acceso a datos.

3.6.2 Patrones de Diseño (Patrones GRASP)

Otro de los Patrones utilizados fueron los Patrones de Diseño. Los mismos representan un amplio repertorio de principios generales basados en la experiencia que guían la creación de un software, uno de los más conocidos es GRASP, el cual es utilizado en el desarrollo del Sub-Sistema.

GRASP es un acrónimo que significa Patrones Generales de Software para Asignar Responsabilidades. El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos (Visconti, et al.).

Patrón Experto: Consiste en asignar una responsabilidad a la clase que cuenta con toda la información para cumplirla. Permite conservar el encapsulamiento y soporta un bajo acoplamiento logrando sistemas más robustos y de fácil mantenimiento. Brinda además un soporte a la alta cohesión definiendo clases más sencillas y más cohesivas.

Patrón Creador: Consiste en asignar la responsabilidad de la creación de la instancia de una clase a aquella que agrega o contiene objetos de la misma, registra instancias de ella, utiliza específicamente sus objetos o tiene los datos de inicialización que le serán transmitidos cuando sea creada su instancia. Brinda

soporte a un bajo acoplamiento suponiendo menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización.

Patrón Alta Cohesión: La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas. Es un principio que debe estar presente durante el diseño, es un patrón evaluativo que se aplica al juzgar decisiones de diseño.

Patrón Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otra. Clases con fuerte acoplamiento son más difíciles de entender cuando están aisladas, son más difíciles de reutilizar porque se requiere la presencia de otras clases de las que dependen. El Bajo Acoplamiento y la Alta Cohesión constituyen un principio, un patrón evaluativo.

Controlador: Es un objeto de interfaz no destinada al usuario, que se encarga de manejar un evento del sistema. Define además el método de su operación.

3.7 Conclusiones

En este capítulo se obtuvo el modelo de análisis y diseño de la solución propuesta, realizándose los diagramas de clases y de colaboración para cada caso de uso del sistema identificado en el Flujo de Trabajo Requerimientos. Se realizó un bosquejo sobre los estándares establecidos para la construcción del código, de forma tal, que este trabajo pueda tener un desarrollo con la calidad requerida y entendimiento para futuros desarrolladores.

Capítulo IV: Implementación

4.1 Introducción

Este capítulo describirá algunos de los elementos claves identificados en el Modelo de Diseño, el cual es utilizado como entrada fundamental de las actividades de la implementación. Abordará además cómo se relacionan y organizan los componentes de acuerdo con los mecanismos de estructuración en el entorno de implementación así como la distribución física de los nodos que componen al Sub-Sistema.

4.2 Modelo de implementación

4.2.1 Diagrama de Despliegue

Describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre nodos de cómputo. Se utiliza como entrada fundamental en las actividades de implementación debido a que la distribución del sistema tiene una influencia principal en el diseño (Rumbaugh, et al., 2000).

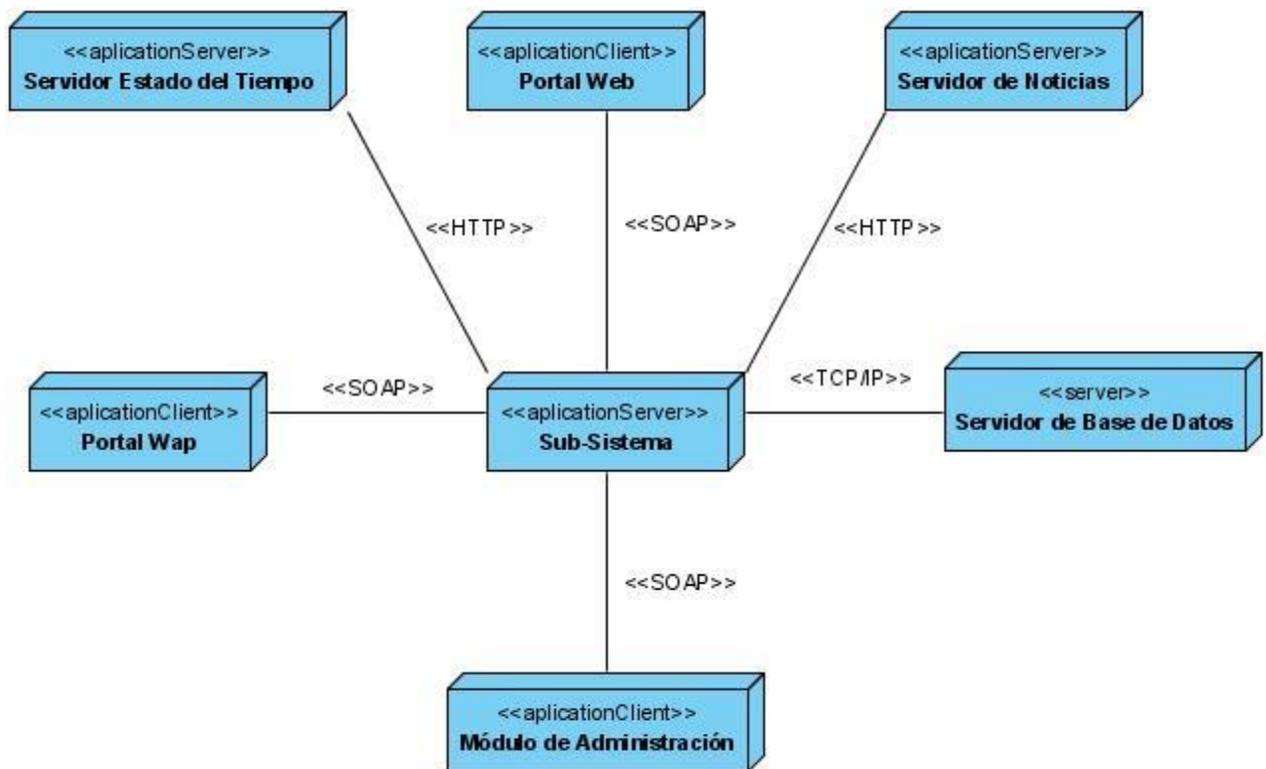


Figura 20: Diagrama de Despliegue.

4.2.2 Diagrama de Componentes.

Los diagramas de componentes muestran la estructura de componentes, incluyendo los clasificadores que especifican componentes y los artefactos que los implementan. También se pueden utilizar para mostrar la estructura de alto nivel del modelo de implementación en términos de subsistemas de implementación y las relaciones entre elementos de implementación. Se representa como un grafo de componentes unidos por medio de relaciones de dependencia (compilación, ejecución), pudiendo mostrarse las interfaces que estos soporten.

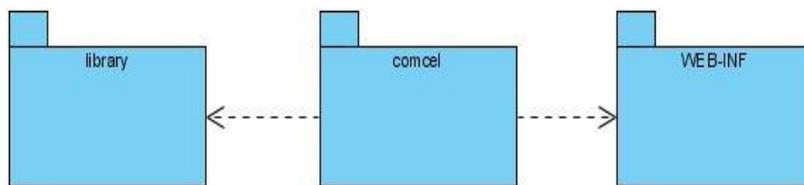


Figura 21: Diagrama de Componentes.

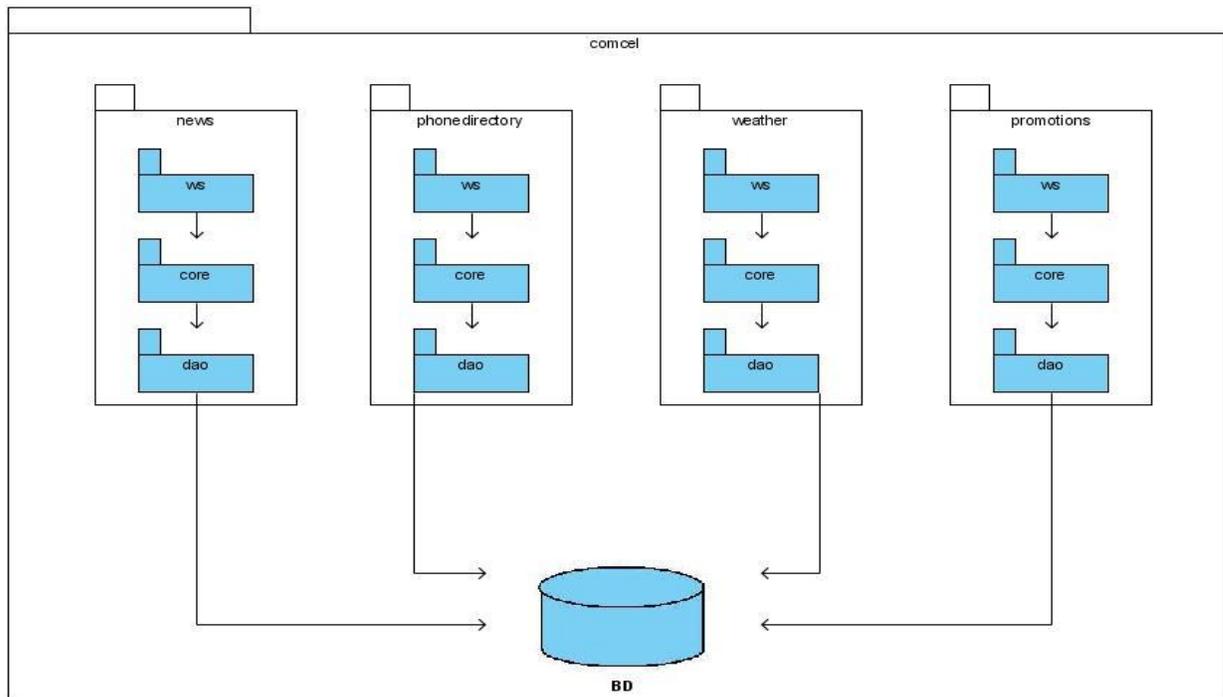


Figura 22: Diagrama de Componentes del paquete comcel.

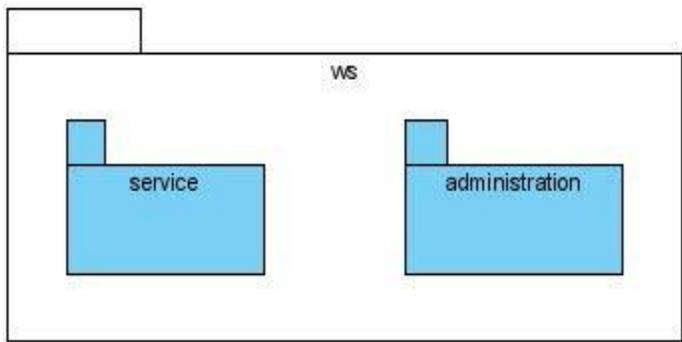


Figura 23: Diagrama de Componentes del paquete ws.

A continuación se representarán por cada uno de los paquetes modelados anteriormente los componentes más importantes presentes en el paquete promotions.

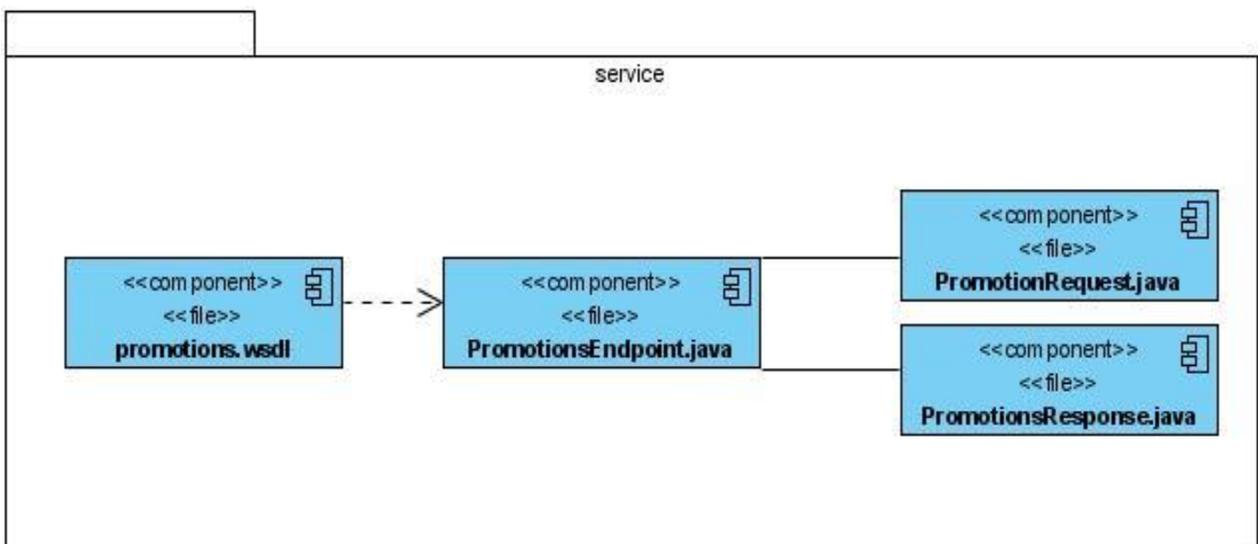


Figura 24: Diagrama de Componentes del paquete service.

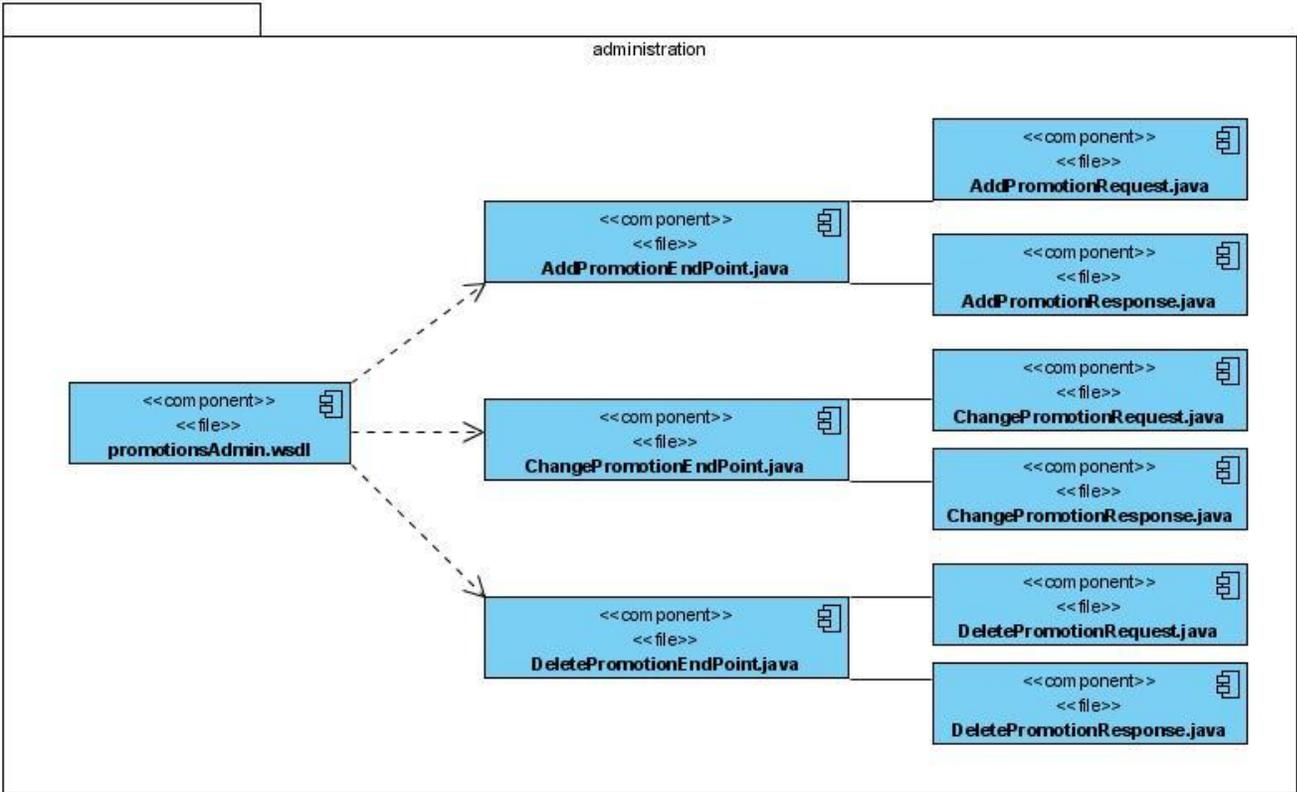


Figura 25: Diagrama de Componentes del paquete administration.

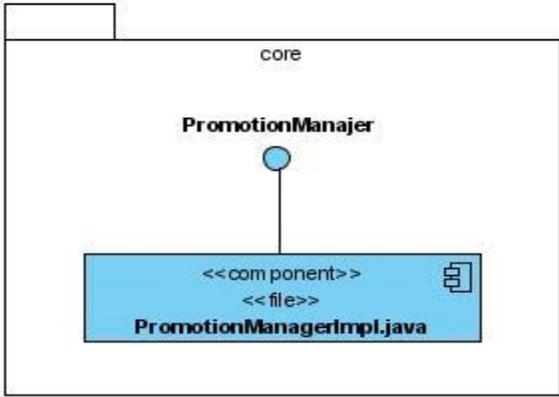


Figura 26: Diagrama de Componentes del paquete core.

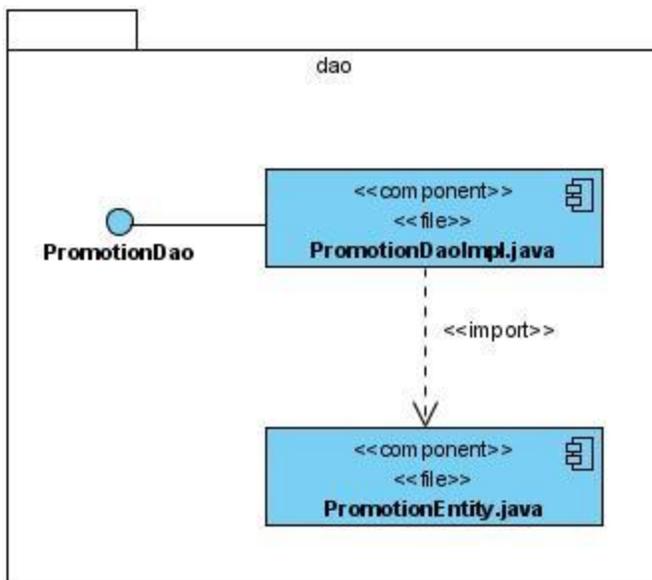


Figura 27: Diagrama de Componentes del paquete dao.

4.3 Conclusiones

En este capítulo se representaron los elementos necesarios para la comprensión de cómo el software fue implementado, elaborándose los artefactos más importantes correspondientes al Flujo de Trabajo de Implementación, como el Diagrama de Despliegue y el Diagrama de Componentes.

Capítulo V: Estudio de factibilidad

5.1 Introducción

En este capítulo se muestra la factibilidad de la realización de este proyecto de software, usando el método Puntos de Casos de Uso, el cálculo del esfuerzo y el tiempo de desarrollo, así como los beneficios y costos, mostrando de esta forma si es viable o no llevar a cabo la realización del Sub-Sistema.

5.2 Método de Estimación “Puntos por Casos de Uso”

Este método permite realizar una estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores.

5.2.1 Cálculo de Puntos de Casos de Uso sin ajustar

Identificar los Puntos de Casos de Uso sin ajustar:

$$UUCP = UAW + UUCW \quad (1)$$

Donde:

UUCP: Puntos de Casos de Uso sin ajustar.

UAW: Factor de Peso de los Actores sin ajustar.

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

Para calcular el UAW se analiza la cantidad de actores presentes en el sistema y la complejidad de cada uno de ellos. Esta complejidad puede definirse de la siguiente manera:

Tipo de Actor	Descripción	Factor de Peso
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API, Application Programming Interface).	1
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.	2

Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3
----------	--	---

Tabla 1: Determina el Factor de Peso de los Actores sin ajustar (UAW).

En el Sub-Sistema se determinaron 2 actores medios y un actor simple.

$$UAW = \sum (a * p) = 1 * 1 + 2 * 2 = 5 (2)$$

Donde:

a : Cantidad de actores presentes en el sistema.

p: Factor de peso asociado al tipo de actor.

El UUCW se calcula analizando la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde la cantidad de transacciones se determinan a partir de la descripción textual de los Casos de Uso.

Tipo de Caso de Uso	Descripción	Factor de Peso
Simple	El Caso de Uso contiene de 1 a 3 Transacciones.	5
Medio	El Caso de Uso contiene de 4 a 7 Transacciones.	10
Complejo	El Caso de Uso contiene más de 8 Transacciones.	15

Tabla 2: Determina el Factor de Peso de los Casos de Uso sin ajustar (UUCW).

En el Sub-Sistema se determinaron 11 tipos de casos de uso simples quedando UUCW:

$$UUCW = \sum (CU * p) = 11 * 5 = 55 (3)$$

Donde:

CU: Cantidad de casos de uso de un tipo presentes en el Sub-Sistema.

p: Factor de Peso asociado a un tipo de caso de uso.

Sustituyendo los resultados obtenidos en las ecuaciones (2) y (3) en la ecuación (1):

$$UUCP = 5 + 55 = 60 \text{ (4)}$$

5.2.2 Cálculo de Puntos de Casos de Uso ajustados

Identificar los Puntos de casos de uso ajustados:

$$UCP = UUCP \times TCF \times EF \text{ (5)}$$

Donde:

UCP: Puntos de Casos de Uso ajustados.

UUCP: Puntos de Casos de Uso sin ajustar.

TCF: Factor de Complejidad Técnica.

EF: Factor de Ambiente.

El Factor de Complejidad Técnica (TCF) es un coeficiente que se calcula cuantificando un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de estos factores se cuantifica con un valor de 0 a 5.

Significado de los valores:

0: No presente o sin influencia.

1: Influencia incidental o presencia incidental.

2: Influencia moderada o presencia moderada.

3: Influencia media o presencia media.

4: Influencia significativa o presencia significativa.

5: Fuerte influencia o fuerte presencia.

Factor	Descripción	Peso	Valor asignado
T1	Sistema distribuido	2	1
T2	Objetivos de performance o tiempo de respuesta	1	1
T3	Eficiencia del usuario final	1	1
T4	Procesamiento interno complejo	1	0
T5	El código debe ser reutilizable	1	2
T6	Facilidad de instalación	0.5	2
T7	Facilidad de uso	0.5	3
T8	Portabilidad	2	3
T9	Facilidad de cambio	1	3
T10	Concurrencia	1	2
T11	Incluye objetivos especiales de seguridad	1	1
T12	Provee acceso directo a terceras partes	1	3
T13	Se requieren facilidades especiales de entrenamiento a los usuarios	1	2

Tabla 3: Determina el Factor de Complejidad Técnica (TCF).

Calculando el Factor de Complejidad Técnica:

$$TCF = 0.6 + 0.01 \times \sum (\text{Peso}_i \times \text{Valor asignado}_i)$$

Donde:

Peso_i : Representa el peso del factor i .

Valor asignado_i : El valor que se le asignó según el aporte del mismo al sistema.

$$TCF = 0.6 + 0.01 * 25,5 = 0,855 (6)$$

El Factor de Ambiente (EF) se calcula similar al Factor de Complejidad Técnica. Los factores que se cuantifican están relacionados con las habilidades y el entrenamiento del grupo involucrado en el desarrollo del sistema.

Factor	Descripción	Peso	Valor Asignado
E1	Familiaridad con el modelo de proyecto utilizado	1.5	5
E2	Experiencia en la aplicación	0.5	3
E3	Experiencia en orientación a objetos	1	5
E4	Capacidad del analista líder	0.5	5
E5	Motivación	1	5
E6	Estabilidad de los requerimientos	2	3
E7	Personal part-time	-1	0
E8	Dificultad del lenguaje de programación	-1	3

Tabla 4: Determina el Factor de Ambiente (EF).

- Para los factores E1 al E4, un valor asignado de 0 significa sin experiencia, 3 experiencia media y 5 amplia experiencia (experto).
- Para el factor E5, 0 significa sin motivación para el proyecto, 3 motivación media y 5 alta motivación.
- Para el factor E6, 0 significa requerimientos extremadamente inestables, 3 estabilidad media y 5 requerimientos estables sin posibilidad de cambios.
- Para el factor E7, 0 significa que no hay personal part-time, 3 significa mitad y mitad, y 5 significa que todo el personal es part-time (nadie es full-time).
- Para el factor E8, 0 significa que el lenguaje de programación es fácil de usar, 3 medio y 5 que el lenguaje es extremadamente difícil.

Calculando el Factor ambiente:

$$EF = 1.4 - 0.03 \times \sum (\text{Peso}_i \times \text{Valor asignado}_i)$$

Donde:

Peso_i : Representa el Peso del factor i .

Valor asignado_i : El valor que se le asignó según el aporte del mismo al sistema.

$$EF = 1.4 - 0.03 * 24,5 = 0,665 \text{ (7)}$$

Sustituyendo los resultados obtenidos en las ecuaciones (4), (6), (7) en la ecuación (5).

$$UCP = 60 * 0,855 * 0,665 \approx 34,11 \text{ (8)}$$

5.3 Cálculo del Esfuerzo

Para realizar el cálculo del esfuerzo, el primer paso es definir el factor de conversión horas-hombre/Punto de Casos de Uso que se debe utilizar para calcular el esfuerzo. Para ello se contabilizan cuántos factores de los que afectan al Factor de Ambiente están por debajo del valor medio (3), para los factores E1 a E6, luego se contabilizan cuántos factores de los que afectan al Factor de Ambiente están por encima del valor medio (3), para los factores E7 y E8.

- Si el total es 2 o menos, se utiliza el factor de conversión 20 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 20 horas-hombre.
- Si el total es 3 o 4, se utiliza el factor de conversión 28 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 28 horas-hombre.
- Si el total es mayor o igual que 5, se recomienda efectuar cambios en el proyecto, ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

El Sub-Sistema utilizará el factor de conversión 20 horas-hombre/Puntos de Caso de Uso.

El esfuerzo en horas-hombre (E) viene dado por la multiplicación de los Puntos de Casos de Uso ajustados (UCP) por el factor de conversión (CF).

$$E = UCP \times CF \quad (9)$$

Donde:

E: Esfuerzo estimado en horas-hombre.

UCP: Puntos de Casos de Uso ajustados.

CF: Factor de conversión.

Sustituyendo el resultado obtenido en la ecuación (8) y el valor del factor de conversión en la ecuación (9) se obtiene como resultado el esfuerzo estimado en horas-hombre.

$$E = 34,11 \times 20 = 682$$

Este método proporciona una estimación del esfuerzo en horas-hombre contemplando solo el desarrollo de la funcionalidad especificada en los casos de uso.

5.4 Distribución del Esfuerzo entre las diferentes actividades de un proyecto

Para una estimación más completa de la duración total del proyecto, hay que agregar a la estimación del esfuerzo obtenida por los Puntos de Casos de Uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo del software.

Para ello se puede tener en cuenta el siguiente criterio, que estadísticamente se considera aceptable. El criterio plantea la distribución del esfuerzo entre las diferentes actividades de un proyecto, según la siguiente aproximación:

Actividad	% esfuerzo	Horas hombre
Análisis	10%	170,5
Diseño	20%	341
Implementación	40%	682

Prueba	15%	255,75
Sobrecarga	15%	255,75
Total	100%	1705

Tabla 5: Esfuerzo entre las diferentes actividades de un proyecto.

Tomando como entrada la estimación de tiempo calculada a partir de los Puntos de Casos de Uso, se pueden calcular las demás estimaciones para obtener la duración total del proyecto. Evaluando la utilización de aproximadamente 8 horas de trabajo diarias y de 24 días laborables en el mes, se estima que para la elaboración del Sub-Sistema es necesario **8,8 meses**.

5.5 Análisis y estimación de costo

El costo del Proyecto se estimará teniendo en cuenta la siguiente ecuación:

$$\text{Costo} = \text{CHM} * \text{ET} / \text{CH} \text{ (1)}$$

Donde:

CHM: Costo Hombre – Mes

ET: Esfuerzo Total (mes-hombre).

CH: Cantidad de hombres.

Para el Sub-Sistema:

CH: 2, ET: 8,8

$$\text{CHM} = \text{CH} * \text{Salario Promedio}$$

$$\text{CHM} = 2 * 100 = 200$$

Sustituyendo valores en la ecuación (1)

$$\text{Costo} = \text{CHM} * \text{ET} / \text{CH} = 200 * 8,8 / 2$$

$$\text{Costo} = \mathbf{880 \$}$$

5.6 Beneficios tangibles e intangibles

El Sub-Sistema tiene como objetivo desarrollar actividades afines para dar soporte a los Servicios de Valor Agregado de la Plataforma COMCEL. Esta plataforma es un producto que tiene dentro de sus premisas ofrecer a CUBACEL una infraestructura que permita brindar los SVA de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas a sus clientes celulares, implicando una ganancia financiera para dicha empresa. Debido a que el producto aún no se pone a disposición de los clientes, no es conveniente nombrar beneficios tangibles del mismo. Estos SVA contribuirán a aumentar la variedad de servicios brindados por CUBACEL a sus clientes.

5.7 Conclusiones

En este capítulo se describió el estudio de la factibilidad correspondiente al sistema propuesto, así como la estimación, el cálculo del esfuerzo, el costo para el desarrollo del software y los beneficios que representará el Sub-Sistema al ser implementado.

Conclusiones generales

Este Trabajo de Diploma culmina dando cumplimiento al Objetivo General planteado, el cual consiste en el desarrollo de un Sub-Sistema que proporcione a CUBACEL una infraestructura que le permita brindar los Servicios de Valor Agregado de Noticias, Estado del Tiempo, Promociones y Directorio Telefónico de Páginas Amarillas a sus clientes de telefonía celular.

Para la confección del Sub-Sistema, se realizó un profundo análisis de las herramientas y tecnologías a utilizar. Se realizó además un estudio crítico y valorativo sobre el proceso de prestación de Servicios de Valor Agregado para dispositivos móviles. Para un buen entendimiento del Sub-Sistema, se definió el diseño del mismo, el cual le da soporte a los requerimientos del sistema. Se determinó la utilización de una Arquitectura de Tres Capas, lo que simplificó la comprensión y organización del desarrollo del Sub-Sistema. Además se utilizaron los Patrones de Diseño los cuales brindaron soluciones a problemas comunes. Una vez definido el esqueleto o base de la aplicación, se realizó la distribución física del Sub-Sistema, es decir, el Diagrama de Despliegue, donde se representaron los nodos que tenían capacidad de procesamiento, así como los protocolos que permiten la comunicación entre ellos.

Al finalizar este Proceso de Desarrollo de Software, se ha obtenido como resultado una aplicación capaz de proporcionar a CUBACEL una infraestructura que le permita brindar los SVA de Promociones, Estado del Tiempo, Noticias y Directorio Telefónico de Páginas Amarillas a sus clientes de telefonía móvil, representando esto un paso en el desarrollo de la telefonía celular en Cuba.

Recomendaciones

Partiendo de las conclusiones generales y de la base de conocimiento aportada durante el trabajo de investigación se recomienda:

- Agregar nuevas funcionalidades por cada uno de los Servicios de Valor Agregado.
- Implementar nuevos Servicios de Valor Agregado.
- Analizar después de un período de tiempo determinado el impacto que ha tenido el software en los clientes de telefonía celular.

Referencia bibliográfica

- Alva, Víctor Balta. 2009.** Nuevo Libro de Lenguaje de Programación JAVA. [En línea] 29 de Abr de 2009. [Citado el: 20 de Ene de 2010.] <http://www.perutech.com.pe/node.php?new=1325>.
- Angelfire. 2009.** Eclipse Galileo. [En línea] 24 de Jun de 2009. [Citado el: 05 de Dic de 2009.] <http://geektheplanet.net/3324/eclipse-galileo-lanzado-oficialmente.xhtml>.
- Crespo Martín, César. 2004.** *Tutorial de Hibernate*. Madrid : s.n., 2004.
- Espinoza, Humberto. 2005.** *PostgreSQL. Una Alternativa de DBMS Open Source*. 2005.
- Galán Jiménez, Jaime y Melón Pérez, Miguel Ángel. 2008.** *Framework para el desarrollo del encaminamiento interdominio con QoS basado en PCE sobre MPLS*. GÍTACA : s.n., 2008.
- García, Luis. 2008.** Sistema de control de versiones: SUBVERSION. [En línea] 17 de Ene de 2008. [Citado el: 15 de Nov de 2009.] <http://observatorio.cnice.mec.es/modules.php?p=modload&name=News&file=artide&sid=548>.
- Gnius, Juan y Junquera, Rafael A. 2006.** *Servicios de Valor Agregado en las Redes Móviles Latinoamericanas: Casos de Operadores*. 2006.
- Lamarca, María Jesús. 2009.** Hipertexto, el nuevo concepto de documento en la cultura de la imagen. [En línea] 12 de Oct de 2009. [Citado el: 25 de Nov de 2009.] <http://www.hipertexto.info>.
- Marina Ramírez, Miguel Ángel. 2009.** *Análisis y Diseño siguiendo RUP para los Terminales*. 2009.
- Piñol, Carles Mateu. 2001.** *Desarrollo de aplicaciones web*. 2001.
- Rumbaugh, James, Booch, Grady y Jacobson, Ivar. 1998.** *El Lenguaje Unificado de Modelado Manual de Referencia*. Cupertino, California : s.n., 1998.
- Rumbaugh, James, Jacobson, Ivar y Booch, Grady. 2000.** *El Proceso Unificado de Desarrollo de Software*. Madrid : s.n., 2000.
- Saffirio, Mario. 2006.** ¿Qué son los Web Services? . [En línea] 05 de Feb de 2006. [Citado el: 01 de Dic de 2009.] <http://msaffirio.wordpress.com/2006/02/05/%C2%BFque-son-los-web-services/>.

Vicente, Raúl. 2006. Inyección de dependencias con Spring. [En línea] 2006. [Citado el: 15 de Feb de 2010.] <http://www.programania.net/disenio-de-software/inyeccion-de-dependencias-con-spring/>.

Visconti, Marcello y Astudillo, Hernán. *Fundamentos de Ingeniería de Software*.

Walls, Craig y Breidenbach, Ryan. 2008. *Spring in Action. Segunda edición*. Greenwick : Manning Publications Co., 2008.

Wiroos. 2008. Qué IDE usas? [En línea] 14 de Oct de 2008. [Citado el: 05 de Dic de 2009.] <http://www.taringa.net/posts/info/1649223/Que-IDE-usas---JAVA.html>.

Xerox. 2001. *El lenguaje de Programación Java™*. 2001.

2006. Definición de Celular y concepto. [En línea] 2006. [Citado el: 11 de Nov de 2009.] <http://www.definicionabc.com/tecnologia/celular.php>.

2001. Digicel. [En línea] 2001. [Citado el: 05 de Feb de 2010.] <http://www.digicelgroup.com/es/about>.

2008. *JPA (Java Persistence API)*. 2008.

Bibliografía

Agüero, Martín. 2007. Introducción a Spring Framework. Buenos Aires, Argentina : s.n., 2007.

Alva, Víctor Balta. 2009. Nuevo Libro de Lenguaje de Programación JAVA. [En línea] 29 de Abr de 2009. [Citado el: 20 de Ene de 2010.] <http://www.perutech.com.pe/node.php?new=1325>.

Angelfire. 2009. Eclipse Galileo. [En línea] 24 de Jun de 2009. [Citado el: 05 de Dic de 2009.] <http://geektheplanet.net/3324/eclipse-galileo-lanzado-oficialmente.xhtml>.

Céspedes, Juan. 2005-2006. *Automatización de tareas. Diseño y Administración de Sistemas y Redes.* 2005-2006.

Crespo Martín, César. 2004. *Tutorial de Hibernate.* Madrid : s.n., 2004.

Denzer, Patricio. 2002. *PostgreSQL 2002.*

Espinoza, Humberto. 2005. *PostgreSQL. Una Alternativa de DBMS Open Source.* 2005.

Galán Jiménez, Jaime y Melón Pérez, Miguel Ángel. 2008. *Framework para el desarrollo del encaminamiento interdominio con QoS basado en PCE sobre MPLS.* GÍTACA : s.n., 2008.

García, Luis. 2008. Sistema de control de versiones: SUBVERSION. [En línea] 17 de Ene de 2008. [Citado el: 15 de Nov de 2009.] <http://observatorio.cnice.mec.es/modules.php?p=modload&name=News&file=artide&sid=548>.

Gnius, Juan y Junquera, Rafael A. 2006. *Servicios de Valor Agregado en las Redes Móviles Latinoamericanas: Casos de Operadores.* 2006.

Gutiérrez, Juan. 2004. *Eclipse y Java.* Valencia : s.n., 2004.

Hernández León, Rolando Alfredo y Coello González, Sayda. 2002. *EL PARADIGMA CUANTITATIVO DE LA INVESTIGACIÓN CIENTÍFICA.* Cuba, Ciudad de la Habana : ISBN: 959-16-0343-6, 2002.

Hidalgo Martínez, Héctor Hugo. 2006-2009. *Trabajando con LOG4j.* 2006-2009.

Lamarca, María Jesús. 2009. Hipertexto, el nuevo concepto de documento en la cultura de la imagen. [En línea] 12 de Oct de 2009. [Citado el: 25 de Nov de 2009.] <http://www.hipertexto.info>.

Mak, Gary. 2008. *Spring Recipes. A Problem-Solution Approach.* New York : s.n., 2008.

Marina Ramírez, Miguel Ángel. 2009. *Análisis y Diseño siguiendo RUP para los Terminales.* 2009.

Piñol, Carles Mateu. 2001. *Desarrollo de aplicaciones web.* 2001.

Rumbaugh, James, Booch, Grady y Jacobson, Ivar. 1998. *El Lenguaje Unificado de Modelado Manual de Referencia.* Cupertino, California : s.n., 1998.

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

Saffirio, Mario. 2006. ¿Qué son los Web Services? . [En línea] 05 de Feb de 2006. [Citado el: 01 de Dic de 2009.] <http://msaffirio.wordpress.com/2006/02/05/%C2%BFque-son-los-web-services/>.

Vicente, Raúl. 2006. Inyección de dependencias con Spring. [En línea] 2006. [Citado el: 15 de Feb de 2010.] <http://www.programania.net/disenio-de-software/inyeccion-de-dependencias-con-spring/>.

Walls, Craig y Breidenbach, Ryan. 2008. *Spring in Action. Segunda edición.* Greenwick : Manning Publications Co., 2008.

Wiroos. 2008. Qué IDE usas? [En línea] 14 de Oct de 2008. [Citado el: 05 de Dic de 2009.] <http://www.taringa.net/posts/info/1649223/Que-IDE-usas---JAVA.html>.

Xerox. 2001. *El lenguaje de Programación Java™.* 2001.

2006. Definición de Celular y concepto. [En línea] 2006. [Citado el: 11 de Nov de 2009.] <http://www.definicionabc.com/tecnologia/celular.php>.

2001. Digicel. [En línea] 2001. [Citado el: 05 de Feb de 2010.] <http://www.digicelgroup.com/es/about>.

2008. *JPA (Java Persistence API).* 2008.

2009. *Tema 5. Plataforma Java EE.* 2009.

Glosario de términos

API: (Application Programming Interface, en español Interfaz de Programación de Aplicaciones). Es el conjunto de funciones y procedimientos (o métodos, si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

CVS: (Concurrent Version System, en español Sistema Concurrente de Versiones). El CVS es un modo muy elegante de administrar versiones de archivos que permite trabajar fácilmente a más de un desarrollador en el mismo proyecto.

DOM: (Document Object Model, en español Modelo de Objetos de Documento). Crea un árbol jerárquico en memoria que contiene todo documento XML y con él en memoria podemos realizar cualquier tipo de recorrido, movimiento o acción con los elementos que deseemos.

DAO:(Data Access Object, en español Objeto de Acceso a Datos). Es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo.

EJB: (Enterprise JavaBean): Es una de las APIs que forman parte del estándar de construcción de aplicaciones empresariales J2EE de Sun Microsystems. Su especificación detalla cómo los servidores de aplicaciones proveen objetos desde el lado del servidor que son, precisamente, los EJB.

FTP: (File Transfer Protocol, en español Protocolo para Intercambiar Archivos en Internet). Utiliza los protocolos de Internet TCP/IP para permitir la transferencia de datos, de la misma manera que el HTTP en la transferencia de páginas web desde un servidor al navegador de un usuario y el SMTP para transferir correo electrónico a través de Internet.

GUI: (Graphical User Interface, en español Interfaz Gráfica de Usuario). Es un programa informático que actúa de interfaz de usuario, utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz.

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

IDE: (Integrated Development Environment, en español 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 interfaz gráfica (GUI).

IoC: (Inversion of Control, en español Inversión de 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.

JavaBeans: Es un componente hecho en software que se puede reutilizar y que puede ser manipulado visualmente por una herramienta de programación en lenguaje Java.

JDBC: (Java Database Connectivity, en español Conexión a Bases de Datos en Java). Es un API para trabajar con bases de datos desde Java, independientemente de la base de datos a la que accedemos.

JDK: (Java Developers Kit). Conjunto de herramientas para desarrollar aplicaciones en Java.

JSP: (Java Server Pages, en español Páginas de Servidor Java). Es una tecnología orientada a crear páginas web en Java.

JVM: (Java Virtual Machine, en español Máquina Virtual de Java): Es una máquina virtual de proceso nativo, es decir, ejecutable en una plataforma específica, capaz de interpretar y ejecutar instrucciones expresadas en un código binario especial (el Java bytecode), el cual es generado por el compilador del lenguaje Java.

LOG: Es un registro oficial de eventos durante un período de tiempo en particular.

MMS: (Multimedia Messaging System, en español Sistema de Mensajería Multimedia). Es un estándar de mensajería que le permite a los teléfonos móviles enviar y recibir contenidos multimedia, incorporando sonido, video, fotos o cualquier otro contenido disponible en el futuro.

ORM: (Object-Relational mapping, en español Mapeo Objeto-Relacional). Es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional.

RSS: (Really Simple Syndication, en español Sindicación Realmente Simple): Es un documento de texto compuesto por etiquetas acotadas entre los símbolos mayor y menor similares a las utilizadas en el XHTML. Está diseñado para la distribución de noticias o información tipo noticias, es decir contienen un resumen de lo publicado en sitios web y weblogs.

RUP: (Rational Unified Process, en español Proceso Unificado de Desarrollo): Es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

SAX: (Simple API for XML, en español API Simple para XML). Se usa para realizar un recorrido secuencial de los elementos del documento XML, es decir, vamos tratando la información a la vez que la vamos leyendo.

Servlets: Aplicación sin interfaz gráfica que se ejecuta en un servidor de Internet, procesando información HTML previamente recogida por un navegador.

SGML: (Standard Generalized Markup Language, en español Lenguaje de Marcado Generalizado): Consiste en un sistema para la organización y etiquetado de documentos. El lenguaje SGML sirve para especificar las reglas de etiquetado de documentos y no impone en sí ningún conjunto de etiquetas en especial.

SMS: (Short Message Service, en español Servicio de Mensajería Corta). Servicio disponible en la red GSM que permite el intercambio de mensajes escrito de hasta 160 caracteres entre terminales GSM.

SMTP: (Simple Mail Transfer Protocol, en español Protocolo Simple de Transferencia de Correo): Es un protocolo de la capa de aplicación utilizado para el intercambio de mensajes de correo electrónico entre computadoras u otros dispositivos.

SOAP: (Simple Object Access Protocol, en español Protocolo de Acceso Objeto Simple): Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

SVN: El Sistema de Control de Versiones es un sistema centralizado para compartir información. En su núcleo está un repositorio, que es un almacén central de datos.

Tags: Es un comando que va insertado en un documento, cuyo fin es especificar cómo el documento o parte del documento debe ser formateado o interpretado.

TortoiseSVN: Es un cliente SVN, implementado como una extensión al shell de Windows.

UML: (Unified Modeling Language, en español Lenguaje Unificado de Modelado): Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es usado para visualizar, especificar, construir y documentar un sistema.

W3C: (World Wide Web Consortium). Fue fundada en 1994 por Tim Berners-Lee, el arquitecto de la World Wide Web. Los fines de esta organización son los desarrollos de estándares abiertos (que los pueda usar todo el mundo) de modo que la Web evolucione coherentemente.

WAP: (Wireless Application Protocol, en español Protocolo de Aplicación Inalámbrica): Es un estándar abierto internacional para aplicaciones que utilizan las comunicaciones inalámbricas, por ejemplo, acceso a servicios de Internet desde un teléfono móvil.

WSDL: (Web Services Description Language, en español Lenguaje de Descripción de Servicios Web): Es un documento XML que se utiliza para describir las funcionalidades de los Servicios Web.

XSD: (XML Schema Definition, en español Definición de Esquema de XML): Es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML.