Universidad de las Ciencias Informáticas
Facultad 2

Título: PolarysGuide: Sistema para la extracción de información desde Servicios Web y su visualización en móviles.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Ivelís Santos Ochoa

Joan Martínez Herrera

Tutor: Ing. Damián Ilizastegui Arriba

Consultante: Lic. Mijail del Toro Céspedes

Lic. Tania Gual

Junio del 2008

Ciudad de la Habana

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos la empresa ProcyonSoluciones a hacer uso del mismo en su beneficio.

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

Ivelís Santos Ochoa

_____

Joan Martínez Herrera

_____

Ing. Damián Ilizastegui Arriba

_____

AGRADECIMIENTOS

Queremos agradecer a todas las personas que de una forma u otra han ayudado al desarrollo de este trabajo, especialmente a Rigo que siempre encontró tiempo para ayudarnos con las dudas, a Erick (que es el que más quiere a Ive, después de mí), a Pedro (EJB) que también nos ayudó cuando tuvimos dudas, a Liane, David y Yeilin (los salvajes al Word 2007) por su cooperación en la elaboración del documento, a Mijail y a Abel que siempre encontraron tiempo para asesorarnos, a York por su preocupación, a nuestro tutor Damián que a pesar de sus problemas familiares encontró tiempo para nosotros, a todos nuestros compañeros de grupo que sin ellos no pudiéramos estar aquí, a todos de veras muchas gracias.

Ivelís y Joan

Primero que todo agradecer a mis padres que han constituido mi apoyo y mi fuerza en esta carrera, todo se los debo a ustedes, al Franky que aunque no esté cerca me sigue ayudando y aclarando dudas, a mi compañera de tesis Ivelís por siempre ser tan positiva y apoyarme cuando las cosas se complicaron además de darme más de 6 razones para admirarla cada día más, ha sido un orgullo compartir este trabajo contigo, a Yiset y Cealys por siempre estar dispuestas a darme una mano en cuanto problema surge en mi camino, tanto profesional como sentimentalmente, a la Gisse por darnos ánimos cada 5 minutos, a Mima por escuchar mis locuras de estudio aunque no las entienda, a Maury, Rufina, Tatica y Lázaro por estar siempre preocupados y darme ánimos para seguir adelante, a Dennis por ser tan buen amigo en todo este tiempo y compartir mis problemas, al Pedri(el Primo Primo) por sacarme una sonrisa cuando solo pensaba en las horas de sueño que me faltaban, al Rapho que aún con mucha pincha tuvo tiempo para hacernos el diseño de la presentación de la aplicación J2ME.

Joan

En primer lugar agradecer a mi mamá, mi papá, mi tía, mis abuelos, Eduardo y mi tío por apoyarme siempre y estar para mí en todo momento. A mi novio Onielkis, por apoyarme siempre, sobre todo en esta etapa tan difícil y ser mi conciencia y mi felicidad. A mis 101 Dálmatas (Yaro, Nairo, Liu, Ana, Pidio, Ele, Rosque, Yu y Nayo), mi fuerza, apoyo y alegría en estos 5 años de universidad. A Malena y Lida las otras chicas 1520, que siguen presentes. A mis amigos del pre Heidi, Amalia y Luis Alberto por no perder el contacto y seguir presentes a pesar de la distancia y el tiempo. A mis compañeros de Procyon, por hacerme reír y “sufrir” tanto, en especial a Joan por soportarme mis llegadas tardes y ayudarme en los momentos de “crisis de tesis”, muchas gracias por más de 6 razones.

Ivelís

DEDICATORIA

Este trabajo lo dedico en primer lugar a mi familia por confiar en mí y ser mi apoyo en cada momento, a mi novio Onielkis por darme tantas fuerzas y tanto amor y muy especialmente a mi abuela Abita que aunque ya no está sé que le hubiese encantado verme graduada y estaría muy orgullosa de mí.

Ivelís

Este trabajo se lo dedico a mis padres que han vivido junto a mí cada minuto de alegría o tristeza durante estos 5 años, y en especial a Pipo que siempre soñó con este momento y que aunque no este aquí para verlo físicamente se lo dedico con todo mi corazón.

Joan

RESUMEN

El auge de la telefonía celular ha permitido que en la actualidad los teléfonos móviles no se utilicen únicamente para comunicarse mediante llamadas telefónicas, o mensajes de texto sino que brinden otra serie de funcionalidades: facilidades de personalización como son las descargas de contenidos, ya sean juegos, fondos de pantalla o tonos, reproducción de música y el acceso a información publicada mediante Servicios Web. Estos últimos brindan una serie de servicios a los cuales los usuarios pueden acceder a través de la red y permiten que diferentes tipos de aplicaciones aunque se encuentren desarrolladas con diferentes tecnologías puedan comunicarse e integrarse.

En la empresa Procyon Soluciones se desarrollan disímiles aplicaciones para celulares y se desea incursionar en el campo de los Servicios Web.

Por ello se decidió el desarrollo de un sistema que facilite la extracción de información de los diferentes Servicios Web y la visualización de esta en los móviles. La aplicación permite a los usuarios móviles acceder a un listado de Servicios Web que se encuentran registrados en el sistema y a los datos que sobre ellos se tienen.

Para dar solución a estas necesidades se desarrolló un Servicio Web central, encargado de extraer la información desde distintos Servicios Web y responder a las peticiones de los usuarios, y una aplicación J2ME para los clientes móviles encargada de realizar las peticiones al Servicio Web central.

TABLA DE CONTENIDOS

INTRODUCCIÓN

En los últimos años la tecnología ha experimentado un desarrollo inesperado y este hecho se evidencia mayormente en el desarrollo de la telefonía celular. Es por ello que este tipo de telefonía ha superado las expectativas de sus creadores y ha ido incorporando toda una serie de nuevos servicios y ventajas, apropiándose de un lugar fundamental en cada uno de los aspectos de la vida diaria.

La tecnología celular surgió como un medio para facilitar la comunicación, ya sea por medio de la voz o la transmisión de datos, entre personas que se encontraban a grandes distancias o que no podían ser conectadas mediante una línea telefónica común. En la actualidad esta es simplemente su función básica, pues con el paso del tiempo se han desarrollado nuevos usos y significados.

En la actualidad los celulares ya no se limitan a la función de comunicar a dos personas entre sí, sino que han evolucionado hasta incluir modalidades como el acceso a la Internet en casi todos sus aspectos y transmisión de datos, mp3, teleconferencia, Servicios Web, transmisión de archivos fotográficos y videos, etc., abriendo nuevas y grandes oportunidades en esta área en el ámbito de los negocios.

La empresa Procyon Soluciones, que se encuentra enmarcada dentro de la Infraestructura Productiva de la Universidad de las Ciencias Informáticas (UCI), trabaja en diferentes líneas relacionadas con la telefonía celular. La empresa quiere hacer uso de las nuevas tecnologías aprovechando las facilidades y bondades de los Servicios Web permitiendo que sean accedidos desde terminales móviles. Los Servicios Web son una tecnología que no se encuentra estandarizada puesto que cada desarrollador los elabora según su conveniencia, por lo que existe una gran diversidad entre ellos y se dificulta la interacción con los clientes celulares. Estos Servicios Web al estar desarrollados de una manera independiente y estar pensados para un uso específico, no se adaptan con facilidad a la limitadas prestaciones de la mayoría de los dispositivos móviles que pueden acceder a las facilidades que estos brindan. Además, en la actualidad la empresa no cuenta con ninguna interfaz cliente para móviles que permita la interacción con Servicios Web ni que facilite a los usuarios de móviles la obtención de información de Servicios Web disponibles en la red.

Se dio inicio a una investigación con el objetivo de dar solución al problema científico planteado: ¿Cómo visualizar en un teléfono móvil información extraída desde Servicios Web publicados?

El objeto de estudio está enmarcado en los Servicios Web y las aplicaciones informáticas que permitan visualizar información obtenida de un Servicio Web.
El campo de acción se centra en el desarrollo de un Servicio Web central y una aplicación cliente J2ME (Java 2 Micro Edition) para acceder a Servicios Web.

Se han trazado los siguientes objetivos para obtener un software de calidad:

Objetivo General.

- Analizar, diseñar e implementar un Servicio Web central que estandarice los diferentes tipos de Servicios Web que existen y extraiga información desde distintos centros de información y una aplicación J2ME para la visualización del contenido obtenido desde el Servicio Web central en un teléfono celular.

Objetivos Específicos.

- Crear un Servicio Web central que extraiga información desde distintos centros de información.
- Crear una aplicación para dispositivos móviles que se conecte con el Servicio Web central y despliegue en pantalla la información obtenida.

Preguntas de Investigación

- ¿Cómo funcionan los Servicios Web?
- ¿Cómo vincular una aplicación J2ME a un Servicio Web y extraer la información que este brinda?
- ¿Qué facilidades ofrecería esta aplicación a los usuarios de dispositivos móviles?

Tareas de Investigación

- Estudiar la metodología de elaboración de Servicios Web capaces de extraer información de otros.
- Estudiar productos en el mercado internacional y su impacto en el mismo con el objetivo de determinar la posible efectividad que pueda tener el producto final.
- Estudiar aplicaciones móviles que sean capaces de conectarse a Servicios Web y extraer la información brindada por estos últimos.
- Crear una arquitectura que permita dar cumplimiento a los objetivos específicos trazados en nuestra investigación.
- Analizar e implementar patrones de diseño que nos ayuden a crear un producto robusto y flexible.

Para lograr una mejor organización del contenido se decidió que estuviera estructurado en capítulos:

Capítulo 1: Fundamentación Teórica: Se expone el estudio del estado del arte realizado en la universidad, en el país y a nivel internacional. También se exponen conceptos estrechamente relacionados con la aplicación y las metodologías, lenguajes, técnicas y herramientas involucradas en el desarrollo del software.

Capítulo 2: Descripción del Sistema: Se describen los requerimientos y el modelamiento del sistema.

Capítulo 3: Análisis y Diseño del Sistema: Se describen los temas relacionados con el análisis y diseño del sistema, además de los artefactos que fueron generados en esta etapa de desarrollo del software.

Capítulo 4: Implementación y Prueba: Se describe los temas relacionados con la implementación del sistema y la planificación y resultados de las pruebas efectuadas, así como los artefactos que se generaron durante estas etapas.

Capitulo 5: Estudio de Factibilidad: Se describe todo el proceso de estimación de costos, esfuerzo y el tiempo necesario para el desarrollo del proyecto así como la viabilidad del producto final.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se abordan algunos aspectos importantes para lograr una total comprensión del trabajo, así como las herramientas y tecnologías que se utilizaron para dar solución a los objetivos planteados.

1.2 Telefonía Celular

Los sistemas de telefonía móvil celular son aquellos que permiten la comunicación entre usuarios que se desplazan libremente en lugares geográficos diferentes, estos sistemas constituyen grandes redes de comunicaciones que actualmente permiten cursar diferentes servicios, entre ellos:

- Telefonía móvil
- Envío de mensajes cortos
- Datos a baja velocidad

Los sistemas de telefonía móvil celular se basan en un principio donde la zona de cobertura deseada se divide en zonas más pequeñas llamadas células, a las que se asigna un cierto número de radio canales, persiguiendo los siguientes objetivos:

· Gran capacidad de abonados.
· Calidad telefónica similar al servicio telefónico convencional.
· Utilización eficaz del espectro de radiofrecuencias.
· Conmutación automática de radio canales.
· Capacidad de expansión.
· Gran movilidad.
· Poder constituir una red de comunicaciones completa en sí mismos.

1.3 Generaciones de Telefonía Celular

La tecnología móvil ha pasado por varias etapas en su evolución hasta la más reciente versión mejorada de dicha tecnología. Esta son las generaciones de la telefonía celular.

Primera generación (1G): La 1G de la telefonía móvil hizo su aparición en 1979 y se caracterizó por ser analógica y estrictamente para voz. La calidad de los enlaces era muy baja, tenían baja velocidad. En cuanto a la transferencia entre celdas, era muy imprecisa ya que contaban con una baja capacidad y la seguridad no existía. La tecnología predominante de esta generación es AMPS (*Advanced Mobile Phone System*).

Segunda generación (2G): La 2G no arribó hasta 1990 y a diferencia de la primera se caracterizó por ser digital. Los sistemas de 2G utilizan protocolos de codificación más sofisticados que aun se emplean en los sistemas de telefonía celular actuales. Las tecnologías predominantes son: GSM (*Global System Mobile Communications*); CDMA (*Code Division Multiple Access*). Los protocolos empleados en los sistemas 2G soportan velocidades de información más altas por voz, pero limitados en comunicación de datos. Se pueden ofrecer servicios auxiliares, como datos, fax y SMS (*Short Message Service*). La mayoría de los protocolos de 2G ofrecen diferentes niveles de encriptación.

Generación 2.5 G: Esta generación no existe oficialmente, constituye un paso intermedio entre la 2G y 3G. Muchos de los proveedores de servicios de telecomunicaciones se moverán a las redes 2.5G antes de entrar masivamente a la 3. La tecnología 2.5G es más rápida, y más económica para actualizar a 3G. La generación 2.5G ofrece características extendidas, ya que cuenta con más capacidades adicionales que los sistemas 2G, como: GPRS (*General Packet Radio System*), además de que integra WAP (*Wireless Application Protocol*), MMS (*Multimedia Messaging Service*) y juegos para móviles.

Tercera generación 3G: La 3G se caracteriza por contener a la convergencia de voz y datos con acceso inalámbrico a Internet; en otras palabras, es apta para aplicaciones multimedia y altas transmisiones de datos. Los protocolos empleados en los sistemas 3G soportan altas velocidades de información y están enfocados para aplicaciones más allá de la voz como audio (mp3), video en movimiento, videoconferencia y acceso rápido a Internet, sólo por nombrar algunos. (Monografias.com, 2003)

1.4 Cuba y las telecomunicaciones.

Antes del triunfo de la Revolución los servicios de telecomunicaciones existentes en el país se encontraban en manos de monopolios norteamericanos y respondían a intereses puramente capitalistas y de enriquecimiento de los dueños de las compañías.

La *Cuban Telephone Company* era la compañía que se encargaba de los servicios telefónicos en el país y a pesar de que en 1958 no tenía permiso legal para realizar sus operaciones en el país ya que su concesión se había vencido continuaba operando y ni siquiera era fiscalizada por sus servicios.

Con el triunfo de la Revolución tienen lugar toda una serie de transformaciones y se deciden intervenir y nacionalizar las empresas y establecimientos económicos de producción y servicios. De esta forma comienza a patentizarse el carácter socialista de la naciente Revolución a la cual le esperaba una enorme labor en el plano de las comunicaciones ya que el desarrollo en el país en esta área bastante poco y centralizado en las ciudades.

Antes del 1 de enero de 1959 las centrales de teléfonos principalmente estaban localizadas en áreas urbanas con alta densidad de población. Los sistemas de conmutación telefónica eran de tecnología bastante atrasada y el sistema de comunicaciones se encontraba en muy malas condiciones y era necesario desarrollar un fuerte proceso inversionista para desarrollar esta rama.

En la década del 70 los planes de colaboración con los países socialistas posibilitan que se puedan adquirir nuevas técnicas y equipos de comunicaciones. A partir de los años 80 el desarrollo de las telecomunicaciones se enfoca en las comunicaciones rurales y en la infraestructura nacional de larga distancia. A principios de los años 90 se comienza a utilizar la fibra óptica en la capital y entran en servicio 5 estaciones terrenas portátiles de comunicaciones por satélite.

En el año 2001 se inicia la instalación de la Red Nacional de Fibra Óptica que constituye la mayor obra de telecomunicaciones acometida en Cuba.

1.4.1 Comienzos de la Telefonía Celular en Cuba.

El 11 de diciembre de 1991, el gobierno cubano autoriza la creación de la empresa mixta CUBACEL (Teléfonos Celulares de Cuba S.A.) con capital cubano y mexicano y por primera vez se comienza a prestar este servicio en nuestro país. Luego de 14 meses en los que se instaló la infraestructura técnica básica el 24 de febrero de 1993 da inicio la operación comercial de esta empresa en el subsistema occidental (La Habana y Varadero).

En los siguientes años se continúa ampliando la cobertura hacia el resto del país. En marzo de 1995 se inaugura el subsistema oriental que brindaba cobertura a Santiago de Cuba, y ya en mayo de 1996 se extiende este servicio a Moa y se brinda cobertura a una parte de la autopista Habana-Varadero.

En el mes de junio de 1997 comienza a funcionar el subsistema central dándole servicio a la ciudad de Cienfuegos y en 1998 se ve favorecida también la ciudad de Santa Clara. El 31 de mayo del 99 se inaugura en la ciudad de Holguín y ya en el 2000 tiene lugar el mayor crecimiento en este sector y comienzan a recibir cobertura las ciudades de Pinar del Río, Cayo Coco, Ciego de Ávila, Camagüey, Sancti Spíritus, Las Tunas, Bayamo, Guantánamo y Nueva Gerona. Durante esos años continúan ampliándose las capacidades y la cobertura en Ciudad de la Habana, Matanzas y Varadero.

También se realizan importantes inversiones en el área informática de esta empresa y en la actualización de la tecnología utilizada para brindar este servicio.

1.5 Servicios Web

1.5.1 ¿Qué son los Servicios Web?

Los Servicios Web tienen múltiples definiciones, lo que demuestra su complejidad a la hora de poder brindar una adecuada definición que englobe todo lo que son e implican. Una posible definición sería hablar de ellos como un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer una serie de servicios. Los proveedores ofrecen sus servicios como

procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web.

Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesario utilizar una arquitectura de referencia estándar. (World Wide Web Consortium, 2005)

Los Servicios Web presentan toda una serie de ventajas que los convierten en herramientas altamente populares. Sus principales ventajas son los siguientes:

- Independencia del lenguaje y de la plataforma

Separación de especificación de la implementación

- Interoperabilidad
- Acoplamiento débil: Sistemas basados en mensajes

Interacciones síncronas y asíncronas

- A través de Internet

Sin control centralizado

Utilización de Protocolos establecidos

Consideraciones de seguridad

- Modularidad y Reusabilidad de servicios
- Escalabilidad: Aplicaciones uno-a-uno frente a uno-a-muchos

No son aplicaciones con una interfaz gráfica con la que las personas puedan interactuar, sino que son software accesible en internet por otras aplicaciones. De esta forma podemos desarrollar aplicaciones que hagan uso de otras aplicaciones que estén disponibles en internet interactuando con ellas.

Un típico ejemplo podría ser un Servicio Web al que se le pudiese preguntar por una empresa y que nos retornase en tiempo real el valor al que están cotizando las acciones de dicha compañía. De esta forma cualquier aplicación (ya sea web o de escritorio) que quiera mostrar esta información sólo tendría que solicitarla a través de internet al Servicio Web cuando la necesitase.

Básicamente los Servicios Web permiten que diferentes aplicaciones, realizadas con diferentes tecnologías, y ejecutándose en toda una variedad de entornos, puedan comunicarse e integrarse.

1.5.2 Estándares para Servicios Web: SOAP, WSLD, UDDI

Los estándares son definiciones o formatos que se aprueban o reconocen desde organizaciones de estandarización. Generalmente estos organismos están formados por el conjunto de empresas más representativas de un sector o de un campo de la producción. Ellos permiten que las industrias desarrollen componentes con las garantías suficientes de: interacción, funcionalidad y calidad. Ayudan a desarrollar los bloques básicos sobre los que seguir construyendo el edificio tecnológico.

Son extremadamente importantes en la informática, ya que permiten que se combinen productos de diferentes fabricantes para el desarrollo de sistemas, tanto software como hardware.

Los Servicios Web se construyen sobre estándares y a su vez pretenden ser un estándar con los que construir sistemas a partir de piezas dispares, desarrolladas por distintos fabricantes, funcionando en distintos sistemas, y construidas con distintas tecnologías. (Toledano)

Los principales estándares para el desarrollo de Servicios Web son los siguientes:

- SOAP (*Simple Object Access Protocol*)
- WSDL (*Simple Object Access Protocol*)
- UDDI (*Universal Description, Discovery and Integration*)

SOAP (Simple Object Access Protocol)

SOAP es un protocolo de mensajería XML extensible que forma la base de los Servicios Web. SOAP proporciona un mecanismo simple y consistente que permite a una aplicación enviar mensajes XML a otra aplicación. Un mensaje SOAP es una transmisión de una vía desde un

emisor SOAP a un receptor SOAP, y cualquier aplicación puede participar en este intercambio como emisor o receptor.

Los mensajes SOAP se pueden combinar para soportar muchos comportamientos de comunicación, incluyendo, solicitud/respuesta, respuesta solicitada, mensajería asíncrona de una vía, o incluso notificación. SOAP es un protocolo de alto nivel que sólo define la estructura del mensaje y unas pocas reglas para su procesamiento. Es completamente independiente del protocolo de transporte subyacente, por eso los mensajes SOAP se pueden intercambiar sobre HTTP o protocolos de transporte de e-mail.

Dentro del paradigma orientado a objetos, usar un Servicio Web es igual que usar cualquier otra clase. Y esto significa instanciarlo, y llamar a sus métodos, pasándoles los parámetros que sean necesarios, y obteniendo a su vez el resultado que nos retornen.

SOAP es un protocolo que define precisamente cómo debemos codificar las llamadas a los métodos de un Servicio Web, y cómo debe el Servicio Web codificar el resultado para que nosotros lo podamos interpretar.

Estos mensajes son los que transportarán los protocolos de transporte, por lo general, HTTP.

WSDL (Simple Object Access Protocol)

WSDL es un fichero XML usado para describir la interfaz de un Servicio Web como un conjunto de puntos finales de comunicación (métodos) capaces de intercambiar mensajes (es decir recibir llamadas con sus parámetros correspondientes y generar respuesta con el resultado que le corresponda).

Proporciona toda la información necesaria para acceder y utilizar un Servicio Web. Un documento WSDL describe qué hace el Servicio Web, cómo se comunica, y dónde reside.

WSDL se considera parte integral de UDDI, que es un registro de Servicio Web XML. Es el lenguaje usado por UDDI para describir a los Servicio Web.

Una petición de un Servicio Web constaría de los siguientes pasos:

1. En el cliente se elabora una petición de una operación que puede o no tener parámetros.
2. La petición se transforma a formato XML utilizando WSDL
3. La petición transformada se envía vía HTTP utilizando SOAP
4. El servidor de Servicio Web recibe la petición
5. El servidor determina que operación debe realizarse y transforma los parámetros, en caso de tenerlos, de formato XML a su representación correspondiente en el lenguaje utilizado para implementar el servidor
6. El servidor invoca la operación con los parámetros enviados, elabora una respuesta y se la envía al cliente de la misma forma que se ha explicado

UDDI (Universal Description, Discovery and Integration)

Es un directorio de Servicios Web distribuido y basado en Web que permite que se listen, busquen y descubran este tipo de software. Podríamos compararlo con las típicas páginas amarillas.

El registro en el directorio se hace en XML. UDDI es una iniciativa industrial abierta entroncada en el contexto de los Servicios Web. El registro de un negocio en UDDI tiene tres partes:

- Páginas blancas - dirección, contacto y otros identificadores conocidos.
- Páginas amarillas - categorización industrial basada en taxonomías.
- Páginas verdes - información técnica sobre los servicios que aportan las propias empresas.

UDDI es uno de los estándares básicos de los Servicios Web cuyo objetivo es ser accedido por los mensajes SOAP y dar paso a documentos WSDL, en los que se describen los requisitos del protocolo y los formatos del mensaje solicitado para interactuar con los Servicios Web del catálogo de registros.

1.6 Arquitectura Cliente Servidor

La arquitectura Cliente\Servidor es un modelo para desarrollar sistemas de información en el que las transacciones se dividen en elementos independientes que tienen la facilidad de cooperar entre ellos para intercambiar información, recursos o servicios.

En esta arquitectura el cliente inicia un proceso de dialogo que comienza con la solicitud de información u otros recursos y el servidor responde a la solicitud efectuada por el cliente. Con este modelo los clientes tienen la posibilidad de solicitar toda la información que deseen en el momento dado de una o varias fuentes y el servidor o los servidores pueden intercambiar información.

Esta arquitectura constituye la integración distribuida de de un sistema de red, con los recursos, medios y aplicaciones que se encuentren definidos en los servidores, estos administran, atienden y ejecutan las diferentes solicitudes generadas por los clientes. Todos los componentes se encuentran interrelacionados compartiendo la información y los procesos donde el proceso de comunicación es de cierta forma transparente.

Las principales características del modelo Cliente\Servidor son las siguientes:

- El servidor presenta a todos sus clientes una interfaz única y bien definida.
- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente. (Departamento de Control de Calidad y Auditoría Informática, 2001)

Los componentes de esta arquitectura son el cliente, el servidor y la infraestructura de comunicaciones (*middleware*). A continuación explicamos más específicamente las características y funciones de cada uno de los componentes de esta arquitectura.

Cliente

El cliente es la entidad a través de la cual el usuario realiza su solicitud de un servicio, su petición o demanda la utilización de ciertos recursos. Principalmente es el encargado de presentar los datos o la información al usuario desde un ambiente gráfico.

Se comunican a través de procesos que establecen la conexión con el servidor, envían el pedido, reciben las respuestas, manejan las fallas que puedan surgir y realizan las actividades de sincronización y seguridad. Pueden establecer conexión con uno o más servidores.

Sus principales funciones son:

- Manejo de la interfaz del usuario.
- Captura y validación de los datos de entrada.
- Generación de consultas e informes sobre las bases de datos. (Departamento de Control de Calidad y Auditoría Informática, 2001)

Servidor

El servidor es la entidad encargada de proveer el servicio y devolver los resultados, ejecutando el procesamiento de datos, aplicaciones y el manejo de recursos. Recibe las solicitudes enviadas por los clientes y ejecuta los procesos, manejando los interbloqueos, la recuperación ante fallas y otros aspectos. Además deben manejar servicios como administración de la red, mensajes, control y administración de la entrada al sistema, auditoría y recuperación, y contabilidad.

Sus principales funciones son:

- Gestión de periféricos compartidos.
- Control de accesos concurrentes a bases de datos compartidas.
- Enlaces de comunicaciones con otras redes de área local o extensa.
- Siempre que un cliente requiere un servicio lo solicita al servidor correspondiente y éste, le responde proporcionándolo. Normalmente, pero no necesariamente, el cliente y el servidor están ubicados en distintos procesadores. Los clientes se suelen situar en computadoras personales y/o estaciones de trabajo y los servidores en procesadores departamentales o de grupo. (Departamento de Control de Calidad y Auditoría Informática, 2001)

Infraestructura de comunicaciones (Middleware)

El *middleware* es el encargado de que los clientes y los servidores puedan comunicarse ya que el proporciona la infraestructura lógica que proporciona los medios básicos de direccionamiento y transporte. Es un módulo intermedio que no pertenece a los dominios del servidor, ni a la interfaz de usuario ni a la lógica de aplicación en los dominios del cliente. Tampoco es parte de la red física, es simplemente una interfaz lógica estándar de los servicios de red.

Sus funciones son:

- Independizar las dos entidades: El cliente y el servidor no necesitan saber comunicarse entre ellos, sino cómo comunicarse con el módulo de middleware.
- Traducir la información de una aplicación y pasarla a la otra: acepta consultas y datos recuperándolos de la aplicación cliente, los transmite y envía la respuesta de regreso. También genera los códigos de error.
- Controlar las comunicaciones: da a la red las características adecuadas de desempeño, confiabilidad, transparencia y administración. (Departamento de Control de Calidad y Auditoría Informática, 2001)

Ventajas de la arquitectura cliente servidor:

- Aumento de la productividad: Los usuarios pueden utilizar herramientas que les son familiares y la interfaz grafica de usuario reduce el tiempo de aprendizaje de las aplicaciones.
- Menores costos de operación: Permite utilizar maquinas considerablemente más baratas que las requeridas por una solución centralizada. Proporciona un mejor acceso a los datos.
- Mejoras en el rendimiento de la red: Elimina la necesidad de mover gran cantidad de información a las computadoras personales. Los sistemas son escalables para poder ajustarse a las necesidades de las aplicaciones. (Departamento de Control de Calidad y Auditoría Informática, 2001)

Debido a que esta arquitectura constituye algo abstracto esta definida en base a dos modelos fundamentalmente, de dos o tres niveles de acuerdo al número de componentes, aunque ya en la actualidad existen modelos de arquitectura multicapas.

El modelo a dos niveles o capas es el mas simple y cuenta con dos componentes solamente: cliente y servidor, y se acostumbra a instalar la base de datos en el servidor por las ventajas de almacenamiento y velocidad que permite esta práctica. (Monografias.com, 2007)

Normalmente esta arquitectura se utiliza en las siguientes situaciones:

- Cuando se requiera poco procesamiento de datos en la organización.
- Cuando se tiene una base de datos centralizada en un solo servidor.
- Cuando la base de datos es relativamente estática.
- Cuando se requiere un mantenimiento mínimo.

Pero este modelo es difícil de mantener ya que como las reglas del negocio de la lógica aplicación están programadas en cada cliente cualquier cambio que se realice tiene que ser redistribuido en cada uno de los clientes; comprometiendo también la confidencialidad ya que el cliente tiene a su disposición todas las reglas del negocio. Otra dificultad es que los cliente están configurados para acceder a una base de datos específica por lo que si es necesario mover los datos a otra base de datos el proceso se vuelve realmente engorroso.

Por su parte el modelo a tres niveles o capas facilita el mantenimiento de la aplicación y permite centralizar la gestión de las reglas del negocio, por lo que nos decidimos a utilizarlo ya que se adapta a nuestras necesidades de trabajo y permite desarrollar la aplicación respondiendo a los objetivos planteados.

1.6.1 Modelo a tres niveles o capas

Este modelo divide las funciones de una aplicación en tres componentes:

Presentación**:** Este componente se encarga de la interacción hombre máquina a través del monitor, teclado, ratón, o bien mediante algún otro medio como reconocedor de voz.

Servidores**:** Compuesto por varios servidores o componentes de software localizados en una o más plataformas que se encargará de conectar los sistemas existentes.

Información**:** En este componente se incluye la información en sí, los sistemas y aplicaciones existentes. (Departamento de Control de Calidad y Auditoría Informática, 2001)

Ventajas:

- Flexibilidad: La información se adapta a las características del cliente
- Eficiencia: No es necesario tener almacenada toda la información
- Los componentes están centralizados lo que facilita su fácil desarrollo, mantenimiento y uso.
- Se minimiza la limitación de las conexiones a las bases de datos ya que esta solo se ve desde la lógica de aplicación y no desde los clientes.
- Se pueden conceder o denegar los permisos componente a componente y de esta forma se simplifica la administración.

Posibilidades

- Computación en cliente
- Computación en servidor

1.7 Metodología de Desarrollo

El objetivo de los procesos de desarrollo de software es elevar la calidad del software en todas las fases que atraviesa y lograra una mayor transparencia y control sobre el proceso. Es responsabilidad del proceso de desarrollo que estas medidas para elevar la calidad del software sean reproducibles en cada desarrollo.

En los últimos tiempos la variedad y cantidad de estos procesos de desarrollo ha aumentado considerablemente. Existen dos corrientes fundamentales en la actualidad los llamados métodos pesados y los métodos ligeros. La diferencia fundamental entre ambos métodos es que los métodos pesados intentan mejorar la calidad a través de orden en el desarrollo del software y de la

documentación. Los métodos ligeros por su parte apuestan por la comunicación directa e inmediata entre las persona que intervienen en el proceso de desarrollo.

Existen tres metodologías de desarrollo fundamentales: RUP, XP y MSF.

La metodología XP (Extreme Programing) consiste en una programación rápida que cuenta con el usuario final como parte del equipo, es utilizada principalmente en proyectos a corto plazo y con un equipo reducido por lo que no responde a nuestros intereses de desarrollo y no permite la obtención de todos los artefactos y documentación requerida por el proyecto.

Por su parte la metodología MSF (Microsoft Solution Framework) es flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, los que se encargan de controlar la planificación, el desarrollo y la gestión de proyectos tecnológicos, pero se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas, que son de una importancia fundamental para el desarrollo de este proyecto.

La metodología RUP (Rational Unified Process) es uno de los procesos de desarrollo de software catalogados como pesados y es el que se utilizará para documentar y diseñar el sistema por las ventajas que este ofrece las cuales son enunciadas en el siguiente epígrafe. (Metodologías De Desarrollo De Software, 2004)

1.7.1 RUP (Rational Unified Process)

RUP constituye uno de los proceso de desarrollo de software mas generales de los que existen en la actualidad ya que tiene la facilidad de que esta pensado para que pueda adaptarse a cualquier proyecto y no tan solo de software.

La metodología RUP se divide en 4 fases que abarcan perfectamente todo el ciclo de vida de nuestro proyecto:

- Inicio: El objetivo en esta etapa es la puesta en marcha del proyecto.
- Elaboración: En esta etapa el objetivo es la definición, el análisis y el diseño.
- Construcción: En esta etapa el objetivo es la implementación.
- Transición: El objetivo es alcanzar el fin del proyecto y la puesta en producción.

Además permite que en cada fase se puedan ejecutar una o varias iteraciones (de tamaño variable según el proyecto que se desarrolle) y dentro de cada una de estas fases se siga un modelo de cascada para los flujos de trabajo que requieren las nuevas actividades.

RUP define nueve actividades a realizar durante el proceso de desarrollo:

1. Modelado de negocio
2. Análisis de requisitos
3. Análisis y diseño
4. Implementación
5. Prueba
6. Distribución
7. Gestión de configuración y cambios
8. Gestión del proyecto
9. Gestión del entorno

Los elementos del RUP son:

- Actividades: Son los procesos que se llegan a determinar en cada iteración.
- Roles: Vienen hacer las personas o entes involucrados en cada proceso.
- Artefactos: Un artefacto puede ser un documento, un modelo, o un elemento de modelo.

RUP se basa en casos de uso para describir lo que se espera del software y esta muy orientado a la arquitectura del sistema documentándose lo mejor posible. Permite obtener una descripción mucho mas completa de lo que se espera del software además de que genera gran cantidad de artefactos que permiten documentar trabajo con mayor profundidad y rigor. (Monpeceres, 2002)

1.8 Lenguaje de Programación y Plataforma de Desarrollo

La tecnología Java consta de un lenguaje de programación y una plataforma de desarrollo.

Java es un lenguaje de programación que fue ideado para que se pareciera bastante a C++ pero con algunas mejoras. Es orientado a objetos y cuenta con una gran robustez gracias a que los punteros no existen y la gestión de memoria no es responsabilidad del programador ya que el

recogedor de basura (*garbage collector*) elimina los objetos a los que no se esta haciendo referencia.

También se puede decir que es un lenguaje de alto rendimiento ya que soporta la concurrencia a través de hilos (*threads*). Es altamente portable ya que los programas desarrollados con este lenguaje pueden ejecutarse en cualquier tipo de hardware o sistema operativo. Contiene un gran número de librerías que constituyen extras y de esta forma mantiene las facilidades básicas del lenguaje en un mínimo y le agrega gran cantidad de funcionalidades necesarias para nuestro software, también evita la posibilidad de manipular la memoria lo cual le confiere una mayor seguridad al no permitir el acceso a los recursos del sistema. (Centro de Tecnología Informática Universidad de Navarra)

Como complemento a todas las ventajas que brinda este lenguaje se puede decir que nos encontramos familiarizados con él a través de nuestro trabajo previo dentro de la empresa Procyon Soluciones.

La Plataforma Java es la plataforma de computación capaz de ejecutar aplicaciones desarrolladas usando el lenguaje de programación Java y un conjunto de herramientas de desarrollo, en este caso gran cantidad de APIs estandarizados. La plataforma Java es en si una máquina virtual.

La maquina virtual de Java es la encargada de procesar las instrucciones del programa y se entiende con el hardware y el sistema operativo de manera tal que la aplicación solo tiene que comunicar las instrucciones a al maquina virtual y esta se encarga del resto. De esta manera se alcanza una independencia del hardware y las aplicaciones se pueden ejecutar en sistemas de diferentes arquitecturas.

Las APIs de Java son especificaciones que permitan que un componente Java pueda ofrecer una funcionalidad determinada. Dan la capacidad de usar de una manera estandarizada en las aplicaciones una gran cantidad de software ya hecho para una enorme cantidad de cosas diferentes, que permite extender las capacidades de las aplicaciones minimizando el esfuerzo de los programadores. (Garrido, 2003)

La plataforma incluye:

- Plataforma Java, Edición Estándar (Java Platform, Standard Edition), o J2SE
- Plataforma Java, Edición Empresarial (Java Platform, Enterprise Edition), o J2EE
- Plataforma Java, Edición Micro (Java Platform, Micro Edition), o J2ME

1.8.1 J2EE (Java 2 Enterprise Edition)

J2EE es una plataforma de programación para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura de niveles distribuida, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. La plataforma J2EE está definida por una especificación. Es también considerada informalmente como un estándar debido a que los suministradores deben cumplir ciertos requisitos de conformidad para declarar que sus productos son conformes a J2EE.

J2EE incluye varias especificaciones de API, tales como *e-mail*, Servicios Web, XML, etc. y define cómo coordinarlos. J2EE también configura algunas especificaciones únicas para J2EE para componentes. Estas incluyen *Enterprise Java Beans(EJB)*, *servlets*, *portlets* (siguiendo la especificación de *Portlets Java*), *Java Server Pages(JSP)* y varias tecnologías de servicios web. Esto permite al desarrollador crear aplicaciones portable entre plataformas y escalable, a la vez que integrable con tecnologías anteriores. Otros beneficios añadidos son, por ejemplo, que el servidor de aplicaciones puede manejar transacciones, la seguridad, escalabilidad, concurrencia y gestión de los componentes desplegados, significando que los desarrolladores pueden concentrarse más en la lógica de negocio de los componentes en lugar de en tareas de mantenimiento de bajo nivel.

Esta versión está orientada al entorno empresarial y orientada especialmente al desarrollo de servicios web, servicios de nombres, persistencia de objetos, XML, autenticación, APIs para la gestión de transacciones, etc. El cometido de esta especificación es ampliar la plataforma J2SE para dar soporte a los requisitos de las aplicaciones de empresa.

1.8.2 J2ME (Java 2 Micro Edition)

La plataforma J2ME es una familia de especificaciones que definen varias versiones minimizadas de la plataforma Java 2; estas versiones minimizadas pueden ser usadas para programar en

dispositivos electrónicos; en teléfonos celulares, PDAs, y tarjetas inteligentes, etc. Estos dispositivos presentan en común que no disponen de abundante memoria ni mucha potencia en el procesamiento, ni tampoco necesitan de todo el soporte que brinda el J2SE.

Esta versión de Java está enfocada a la aplicación en dispositivos electrónicos con capacidades computacionales y gráficas muy reducidas, tales como teléfonos móviles, PDAs o electrodomésticos inteligentes. Esta edición tiene unos componentes básicos que la diferencian de las otras versiones, como el uso de una máquina virtual denominada KVM (Kilo Virtual Machine, debido a que requiere sólo unos pocos Kilobytes de memoria para funcionar) en vez del uso de la JVM (Java Virtual Machine) clásica, inclusión de un pequeño y rápido recolector de basura y otras diferencias que surgirán más adelante. (J2MEgrasia!, 2004)

Los principales componentes de esta plataforma son Connected Device Configurations (CDC), Connected Limited Device Configurations (CLDC) y Mobile Information Device Profiles (MIDP), así como otras muchas herramientas y tecnologías que llevan las soluciones Java a los mercados de consumo y dispositivos integrados. (Java)

MIDP (Mobile Information Device Profiles)

MIDP se combina con la configuración CLDC para proporcionar un entorno de ejecución para dispositivos móviles (teléfonos, PDAs, etc.). Los usuarios pueden seleccionar las aplicaciones MIDP situadas en un servidor web. El dispositivo comprueba si puede ejecutarla y la descarga, la verifica y compila a byte code para poder ejecutarla. Las aplicaciones instaladas se pueden ejecutar, actualizar y borrar de forma sencilla.

Características:

Las aplicaciones MIDP permiten tener aplicaciones intuitivas y gráficas. La GUI se ha optimizado para las pequeñas pantallas, mecanismos de introducción de datos y otras características de los dispositivos móviles.

Existen dos versiones de MIDP:

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

Los requisitos hardware mínimos que exige MIDP 1.0 son los siguientes:

- Pantalla: 96x54 con una profundidad de color por pixel de 1 bit
- Introducción de datos: Se ha de permitir alguno de los siguientes mecanismos:

✓ teclado de una mano
✓ teclado de dos manos
✓ pantalla táctil

- Memoria (exclusivamente para aplicaciones MIDP)

✓ 128 KB de memoria no volátil para las aplicaciones MIDP
✓ 8 KB de memoria no volátil para los datos persistentes de la aplicación
✓ 32 KB de memoria volátil para el entorno de ejecución Java

- Trabajo en red: Ancho de banda limitado, bidireccional, sin cables, normalmente intermitente (Gálvez Rojas, et al.)

1.9 Persistencia de Datos

Los sistemas de bases de datos están diseñados para gestionar grandes volúmenes de información y constituyen la interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. De forma sencilla, un sistema de gestión de bases de datos se puede definir como una colección de datos interrelacionados y un conjunto de programas para acceder a esos datos.

Objetivos del sistema de gestión de base de datos que podemos identificar son:

- Independencia de datos
- Accesibilidad limitada
- Datos al día y sin redundancias
- Consistencia
- Interfaz única
- Entrada directa a los datos
- Recuperación por diferentes accesos
- Función completa de interrogantes
- Estandarización

- Seguridad (Raga, 2007)

1.9.1 Por qué MySQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario. Es muy utilizado en aplicaciones web, en diferentes plataformas, y por herramientas de seguimiento de errores. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. Es una base de datos muy rápida en la lectura, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

Las características principales de MySQL son:

- **Es un gestor de base de datos.** Una base de datos es un conjunto de datos y un gestor de base de datos es una aplicación capaz de manejar este conjunto de datos de manera eficiente y cómoda.
- **Es una base de datos relacional.** Una base de datos relacional es un conjunto de datos que están almacenados en tablas entre las cuales se establecen unas relaciones para manejar los datos de una forma eficiente y segura. Para usar y gestionar una base de datos relacional se usa el lenguaje estándar de programación SQL.
- **Es Open Source.** El código fuente de MySQL se puede descargar y está accesible a cualquiera, por otra parte, usa la licencia GPL para aplicaciones no comerciales.
- **Es una base de datos muy rápida, segura y fácil de usar.** Gracias a la colaboración de muchos usuarios, la base de datos se ha ido mejorando optimizándose en velocidad. Por eso es una de las bases de datos más usadas en Internet. (Web Estilo, 2004)

1.10 Servidor Web

Los servidores web son programas que permiten atender y responder a peticiones HTTP. Sirve contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este proceso tiene lugar a través del protocolo HTTP (*Hypertext Transfer Protocol*) o el protocolo HTTPS (*Hypertext Transfer Protocol Secure*) que es la versión cifrada y autenticada.

El funcionamiento básico de un servidor web es el siguiente:

1. Espera peticiones en el puerto TCP indicado (el estándar por defecto para HTTP es el 80).
2. Recibe una petición.
3. Busca el recurso.
4. Envía el recurso utilizando la misma conexión por la que recibió petición.
5. Recibe una petición nuevamente.

A partir de este diseño básico se han continuado desarrollando otros servidores lo que con variaciones referentes al tipo de peticiones que cada uno puede atender. (Cibernetia)

1.10.1 Servidor Apache

Apache es un servidor web desarrollado por la *Apache Software Foundation*, es de gran solidez y esta considerado el servidor web por excelencia ya que corre en gran cantidad de sistemas operativos lo que lo vuelve prácticamente universal. Es gratis y de código abierto lo que permite que se pueda ver el software sin ninguna restricción. Permite la personalización de las respuestas ante los errores que puedan que puedan ocurrir en el servidor y está estructurado en módulos que se pueden configurar mediante directivas que se encuentran contenidas en ellos.

1.10.2 Servidor de Aplicaciones Tomcat de Apache

Tomcat es un servidor web con soporte de *servlets* y *JSPs*. Puede funcionar como un servidor web por si mismo, está integrado en la implementación de referencia *Java 2 Enterprise Edition* (J2EE) y funciona en cualquier sistema operativo que tenga instalada la maquina virtual de Java.

1.10.3 Apache Axis

Es una implementación de código abierto de SOAP que brinda un entorno de ejecución para Servicios Web que se encuentren implementados en Java. Proporciona varias facilidades para el trabajo con Servicios Web como: herramientas para crear WSDL desde clases Java y clientes Java desde un WSDL. Además permite desplegar, probar y monitorizar los Servicios Web y facilita la integración con servidores de aplicaciones y contenedores de servlets.

Su principal ventaja es la facilidad con la que permite la creación y depuración de Servicios Web, además de su solidez y que se encuentra respaldado por Apache. Por estas razones fue seleccionado para desplegar el Servicio Web principal.

1.11 Herramientas de desarrollo empleadas

Las herramientas de desarrollo que empleadas en el desarrollo del software contribuyen a obtener un producto realmente completo y robusto y que responda a los objetivos planteados.

1.11.1 Herramientas de modelado UML

Desde los inicios del desarrollo de la informática siempre se han intentado utilizar distintas formas de representar los diseños de una forma fundamentalmente personal o con algún que otro modelo gráfico. Como no existía una estandarización en la manera de representar los modelos los diferentes diseñadores no podían compartir sus diseños entre ellos. Es por ello que surgió la necesidad de tener un lenguaje que no solo permitiera estandarizar estos procesos sino que también sirviera de apoyo en los procesos de análisis de los problemas.

De esta forma es que surge UML (Unified Modeling Lenguage) o Lenguaje Unificado de Modelado, el cual se ha convertido en el principal estándar para representar y modelar la información con que se trabaja en las fases de análisis y diseño. Posee una notación grafica que permite representar en mayor o menor medida todas las fases de un proyecto informático: desde el análisis con los casos de uso, el diseño con los diagramas de clases, objetos, etc., hasta la implementación y configuración con los diagramas de despliegue.

UML es ante todo un lenguaje. Un lenguaje que proporciona un vocabulario y una reglas para permitir una comunicación. En este caso, este lenguaje se centra en la representación gráfica de un sistema.

Las funciones de UML son:

- Visualizar: Permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.

- Especificar: Permite especificar cuáles son las características de un sistema antes de su construcción.
- Construir: A partir de los modelos especificados se pueden construir los sistemas diseñados.
- Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

Para representar los sistemas UML ofrece gran variedad de diagramas para visualizar el sistema desde diferentes perspectivas.

- Diagrama de casos de uso.
- Diagrama de clases.
- Diagrama de objetos.
- Diagrama de secuencia.
- Diagrama de colaboración.
- Diagrama de estados.
- Diagrama de actividades.
- Diagrama de componentes.
- Diagrama de despliegue. (Orallo)

Existen numerosas herramientas de modelado pero para este análisis se enfocarán dos principalmente, Rational Rose y Visual Paradigm.

Visual Paradigm es una herramienta CASE de modelado que da soporte al modelado con UML. Soporta el ciclo de vida completo de desarrollo del software. Permite capacidades de ingeniería directa e inversa. Esta herramienta lleva muy poco tiempo de desarrollo y a pesar de ser código abierto no es libre por lo cual habría que pagar por su utilización.

Por todas las facilidades que ofrece y porque es la herramienta CASE que comercializan los desarrolladores de UML y que soporta de forma completa la especificación del UML utilizaremos Rational Rose. (Vizcaíno, et al.)

Rational Rose Enterprise Edition 2003

Rational Rose es la herramienta líder para el modelado de sistemas complejos y de tiempo real. Permite que varias personas trabajen a la vez en el proceso de desarrollo del software, para esto permite que cada desarrollador trabaje en un espacio privado que contenga el modelo completo y tenga un control total sobre los cambios en su espacio de trabajo. Facilita la visualización, comprensión, y refinamiento de los requerimientos y la arquitectura antes de la puesta en marcha de la implementación del sistema.

Por otra parte facilita la generación de código para diferentes lenguajes de programación a partir del diseño en UML. Posee también un mecanismo que brinda la posibilidad de realizar ingeniería inversa o sea a partir del código se puede obtener información sobre su diseño. Otra de sus principales características es que mantiene la consistencia de los modelos del sistema de software, chequea la sintaxis UML y genera documentación automáticamente.

1.11.2 Entorno Integrado de Desarrollo

Un entorno de desarrollo integrado o en inglés Integrated Development Environment (IDE) es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios.

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.

Los IDEs proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación. Es posible que un mismo IDE pueda funcionar con varios lenguajes de programación. Este es el caso de Eclipse, que mediante *pluggins* se le puede añadir soporte de lenguajes adicionales.

Eclipse

El Eclipse es un IDE de código abierto que emplea módulos para proporcionar diferentes funcionalidades, a diferencia de otros entornos monolíticos, donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software.

En cuanto a las aplicaciones clientes, Eclipse provee al programador con *frameworks* muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc. Atendiendo a estas ventajas y a la familiarización que existe con este entorno de desarrollo ha sido escogido para el desarrollo del software.

1.11.3 Herramienta para el control de versiones

Una versión, revisión o edición de un producto, es el estado en el se encuentra en un momento dado en su desarrollo o modificación. Se llama control de versiones a la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, así como las posibles especializaciones realizadas (por ejemplo, para algún cliente específico).

El control de versiones se realiza principalmente en la industria informática para controlar las distintas versiones del código fuente. Sin embargo, los mismos conceptos son aplicables a otros ámbitos como documentos, imágenes, sitios web, etcétera.

Un sistema de control de versiones debe proporcionar:

- Mecanismo de almacenaje de los elementos que deba gestionar (ej. archivos de texto, imágenes, documentación, etc.)
- Posibilidad de realizar cambios sobre los elementos almacenados (ej. modificaciones parciales, añadir, borrar, renombrar o mover elementos)
- Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos (normalmente pudiendo volver o extraer un estado anterior del producto)

Aunque no es estrictamente necesario, suele ser muy útil la generación de informes con los cambios introducidos entre dos versiones, informes de estado, marcado con nombre identificativo de la versión de un conjunto de ficheros, etcétera. La principal clasificación que se puede establecer está basada en el almacenamiento del código:

- Centralizados: existe un repositorio centralizado de todo el código, del cual es responsable un único usuario (o conjunto de ellos). Se facilitan las tareas administrativas a cambio de reducir la potencia y flexibilidad, pues todas las decisiones fuertes (como crear una nueva rama) necesitan la aprobación del responsable.
- Distribuidos: se aumenta la capacidad de decisión distribuida. Esto da más flexibilidad pero puede dificultar bastante la sincronización. (LugFi, 2008)

Subversion

Es un software de sistema de control de versiones centralizado diseñado específicamente para reemplazar al popular CVS, el cual posee varias deficiencias. Es software libre y se lo conoce también como SVN por ser ese el nombre de la herramienta de línea de comandos. Una característica importante de Subversion es que, a diferencia de CVS, los archivos versionados no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo.

Sus principales ventajas frente a otros sistemas de control de versiones son:

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.
- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente ya que tiene costo de complejidad constante.
- Se envían sólo las diferencias en ambas direcciones.
- Puede ser servido mediante Apache.
- Maneja eficientemente archivos binarios

- Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.
- Cuando se usa integrado a Apache permite utilizar todas las opciones que este servidor provee a la hora de autentificar archivos.

Para el desarrollo del software se utilizó específicamente TortoiseSVN, ya que es extremadamente rápido, cómodo de utilizar, provee integración con el explorador de Windows y con el Eclipse que es el IDE utilizado además de presentar una interfaz muy amigable para el usuario. (OsmosisLatina, 2005)

1.12 Descripción de soluciones existentes a nivel nacional e internacional

Durante la búsqueda de otras soluciones existentes con características similares a nuestra solución no se pudo encontrar un sistema que brinde servicios y beneficios similares al que se desarrolla. Están desarrollados numerosos clientes para Servicios Web específicos pero no existe una guía de Servicios Web reconocida que facilite el acceso de los usuarios desde los clientes móviles a estos ni que estandarice el proceso.

1.13 PolarysGuide: Sistema para la extracción de información desde Servicios Web y su visualización en móviles.

Para el desarrollo del sistema se utilizaron los Servicios Web como una herramienta que facilita la comunicación entre diferentes aplicaciones y que permite una independencia del lenguaje y la plataforma. Se utilizó la metodología RUP (Rational Unified Process) y como herramienta de modelado el Rational Rose Enterprise Edition 2003. Para el desarrollo del sistema se trabajó sobre la plataforma Java, específicamente J2EE y J2ME, con Java como lenguaje de programación y el Eclipse como entorno integrado de desarrollo.

1.14 Conclusiones

En este capítulo se explicaron y fundamentaron los principales conceptos y herramientas utilizados durante la investigación y que permiten sustentar la solución propuesta. Se profundizó en los Servicios Web y sus principales ventajas y facilidades para el desarrollo del sistema.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En este capitulo se expone la necesidad de la empresa Procyon Soluciones de contar con un Servicio Web central capaz de conectarse con Servicios Web y extraer información desde ellos así como estandarizar la misma y con una aplicación J2ME capaz de conectarse a este Servicio Web central y permitir visualizar la información en los móviles de los clientes. Además se definirán los requisitos funcionales y no funcionales del sistema; así como la descripción de los actores y casos de uso del sistema.

2.2 Situación Problémica

La empresa Procyon Soluciones, que se encuentra enmarcada dentro de la Infraestructura Productiva (IP) de la Universidad de las Ciencias Informáticas (UCI), trabaja en diferentes líneas relacionadas con la telefonía celular. La empresa quiere hacer uso de las nuevas tecnologías aprovechando las facilidades y bondades de los Servicios Web permitiendo que sean accedidos desde terminales móviles. Los Servicios Web son una tecnología que no se encuentra estandarizada puesto que cada desarrollador los elabora según su conveniencia, por lo que existe una gran diversidad entre ellos y se dificulta la interacción con los clientes celulares. Estos Servicios Web al estar desarrollados de una manera independiente y están pensados para un uso específico, no se adaptan con facilidad a la limitadas prestaciones de la mayoría de los dispositivos móviles que pueden acceder a las facilidades que estos brindan.. Además en la actualidad la empresa no cuenta con ninguna interfaz cliente para móviles que permita la interacción con Servicios Web ni que facilite a los usuarios de móviles la obtención de información de Servicios Web disponibles en la red.

2.3 Objeto de Automatización

La entidad Procyon debe desarrollar un sistema que sea capaz de facilitar la extracción de la información que se encuentra publicada en distintos Servicios Web y la visualización de esta en un teléfono celular.

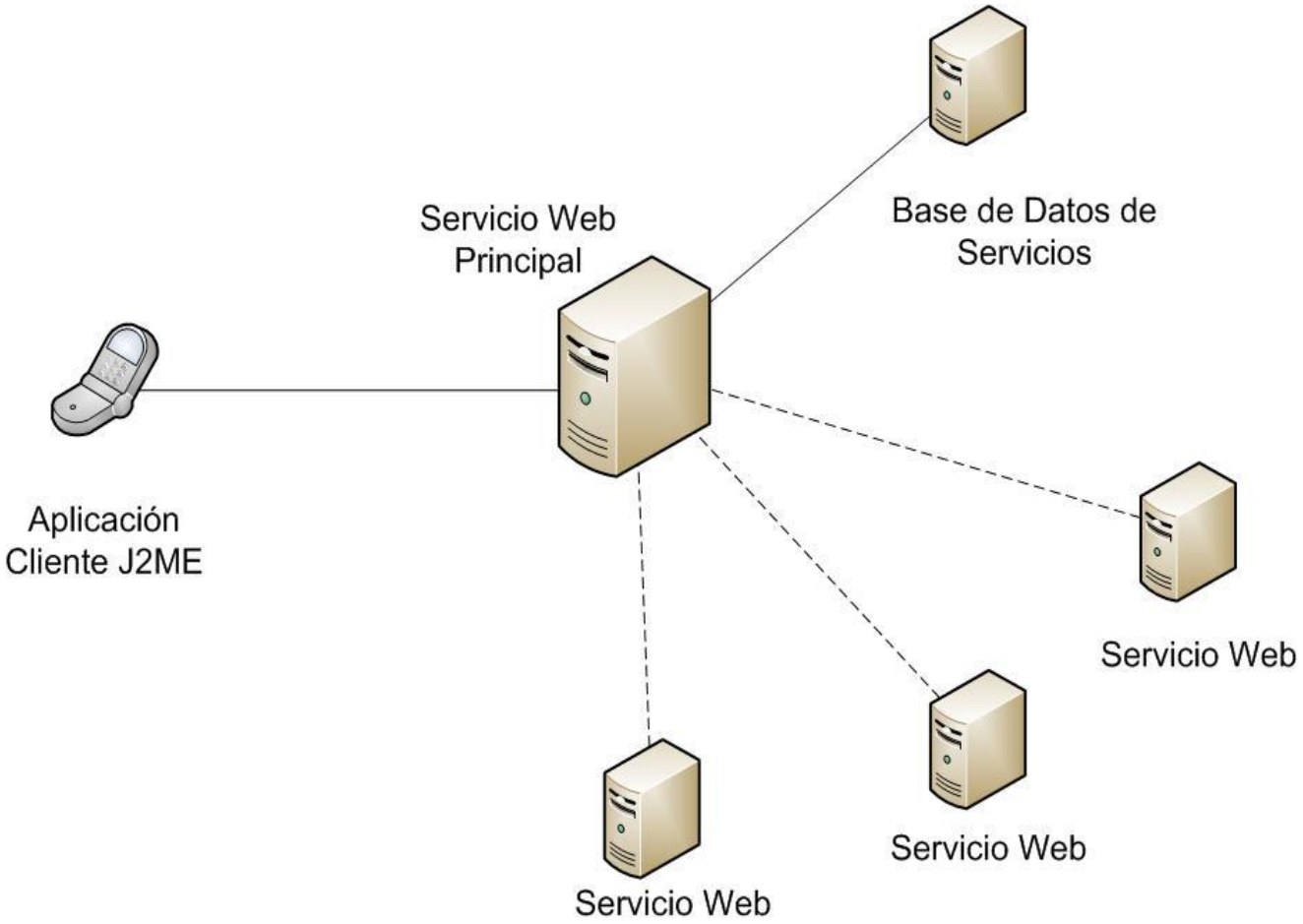
Los usuarios móviles tendrán un listado de Servicios Web disponibles a los cuales podrán tener acceso desde cualquier lugar en que se encuentren y en cualquier momento que deseen. El

sistema debe proponer un estándar que permita que los usuarios puedan acceder a cualquier tipo de Servicio Web siempre y cuando este se encuentre registrado en la base de datos del sistema.

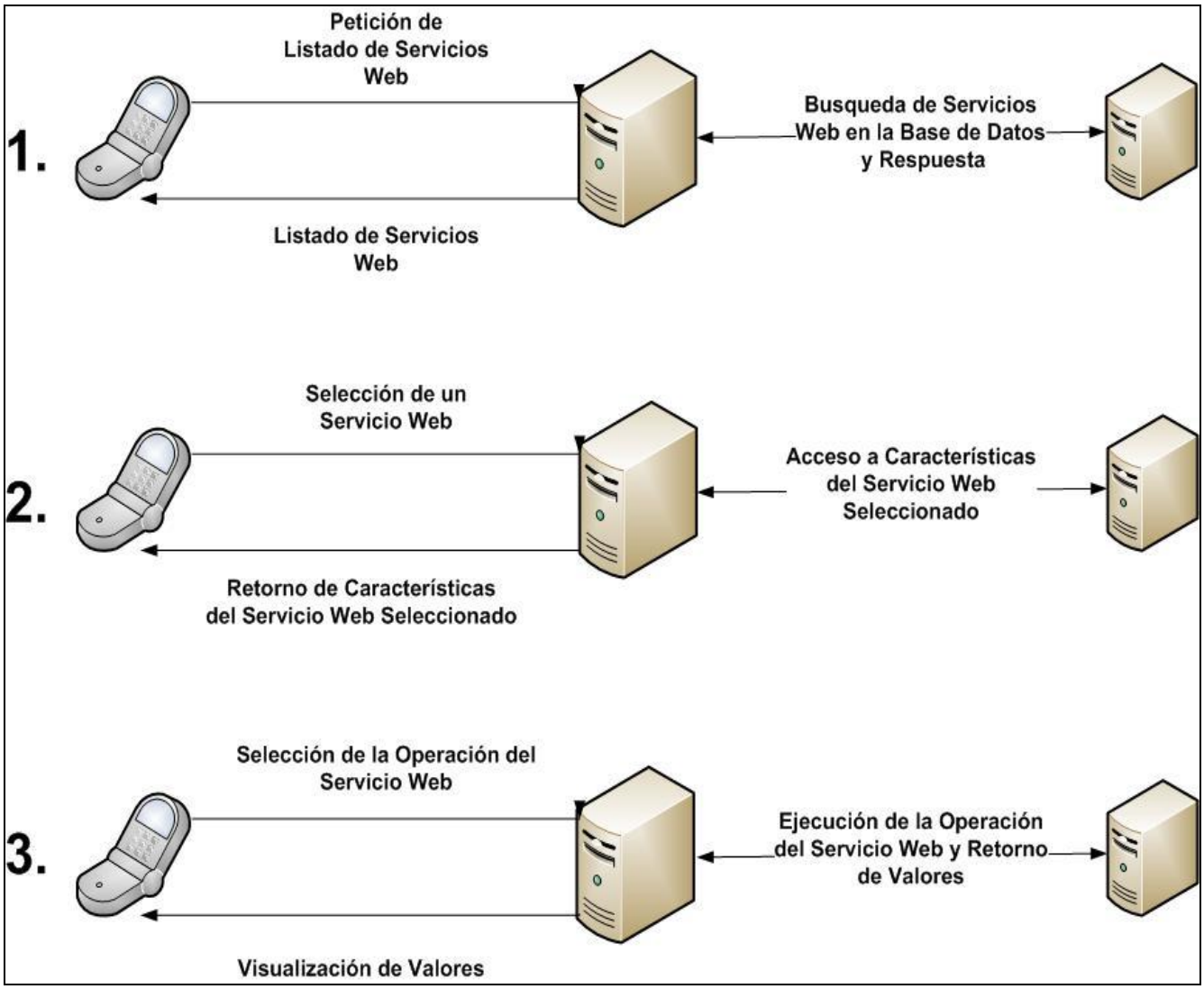
2.4 Propuesta del Sistema

Se propone el desarrollo de un Servicio Web central y una aplicación J2ME para que faciliten la interacción de los usuarios móviles con los Servicios Web.

La aplicación J2ME será capaz de conectarse al Servicio Web central y visualizar la información obtenida en la pantalla del celular.

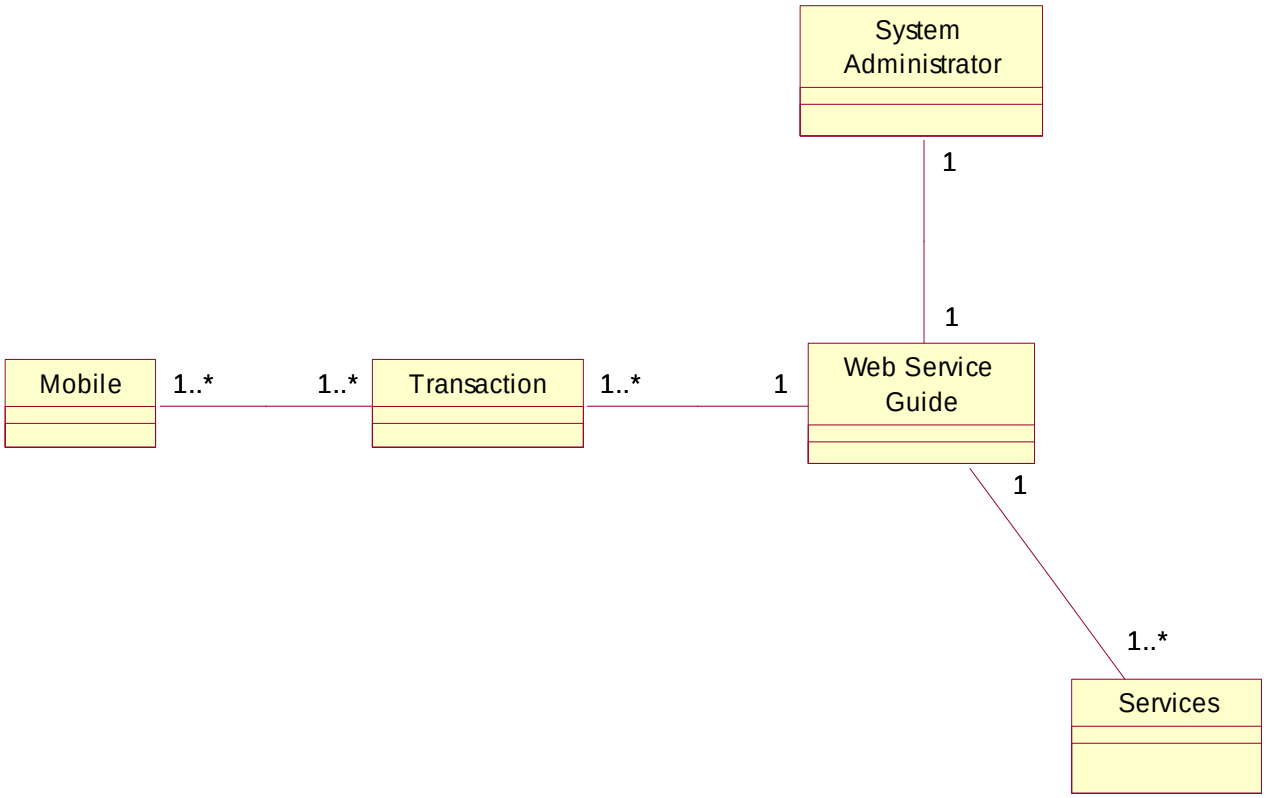


Figura#1. Propuesta del Sistema



Figura#2. Diagrama de Mensajes

2.5 Modelo de dominio



Figura#3. Modelo de Dominio

2.5.1 Descripción de los conceptos del dominio

Mobile: Define al teléfono celular cliente que va a efectuar las peticiones al sistema.

Transaction: Engloba el conjunto de transacciones, peticiones y respuestas que se realizan entre el Servicio Web y el cliente móvil.

Web Service Guide: Define el Servicio Web encargado de recibir y procesar todas las peticiones realizadas por los clientes móviles.

Services: Define al conjunto de Servicios Web que se encuentran registrados en la base de datos del sistema.

System Administrator: Define la aplicación web encargada de controlar y gestionar los Servicios Web.

2.6 Especificación de los requisitos del software

2.6.1 Requerimientos funcionales

RF1. Gestionar Servicios Web

1.1 Permitir a los administradores adicionar un Servicio Web.

1.2 Permitir a los administradores modificar un Servicio Web.

1.3 Permitir a los administradores eliminar un Servicio Web.

RF2. Visualizar Servicio Web

Permitir a los clientes visualizar datos de Servicios Web.

2.6.2 Requerimientos no funcionales

- Apariencia o interfaz externa.

  Diseño sencillo y agradable, con una alta funcionalidad.

- Usabilidad.

  El sistema podrá ser usado por los administradores.

- Rendimiento.

  El sistema debe ser eficiente, la velocidad de procesamiento debe ser la menor posible y el tiempo en que se de respuesta a las solicitudes del cliente debe ser mínimo.

- Portabilidad.

  El sistema será desarrollado en Java por lo que será independiente de la plataforma sobre la que se utilice y la aplicación para los clientes móviles será desarrollada utilizando la plataforma J2ME lo que permitirá que sea posible utilizarla en la mayoría de los dispositivos móviles.

- Hardware

La aplicación del sistema solo podrá ser utilizada en teléfonos que sean compatible con el estándar JAX-RPC y que sean capaces de ejecutar la maquina virtual de Java. El servidor debe tener como mínimo 512 MB de RAM.

2.7 Definición de los casos de uso

2.7.1 Definición de los actores

Tabla 1. Definición de los actores del sistema

Actores	Justificación
Administrador	Define al actor relacionado con la administración de los procesos del sistema.
Cliente_Aplicación	Define al actor que interactúa con el sistema desde una aplicación cliente haciendo uso de los servicios que el sistema brinda.

2.7.2 Listado de casos de uso

Tabla 2. Listado de casos de uso del sistema: **Gestionar Servicios Web**

CU1	Gestionar Servicios Web
Actor	Administrador
Descripción	El actor decide que acción va a realizar (adicionar, modificar, eliminar). En el caso de adicionar y modificar llena o actualiza los datos de un Servicio Web. Si la acción es eliminar selecciona el Servicio Web deseado y lo elimina.
Referencia	RF1

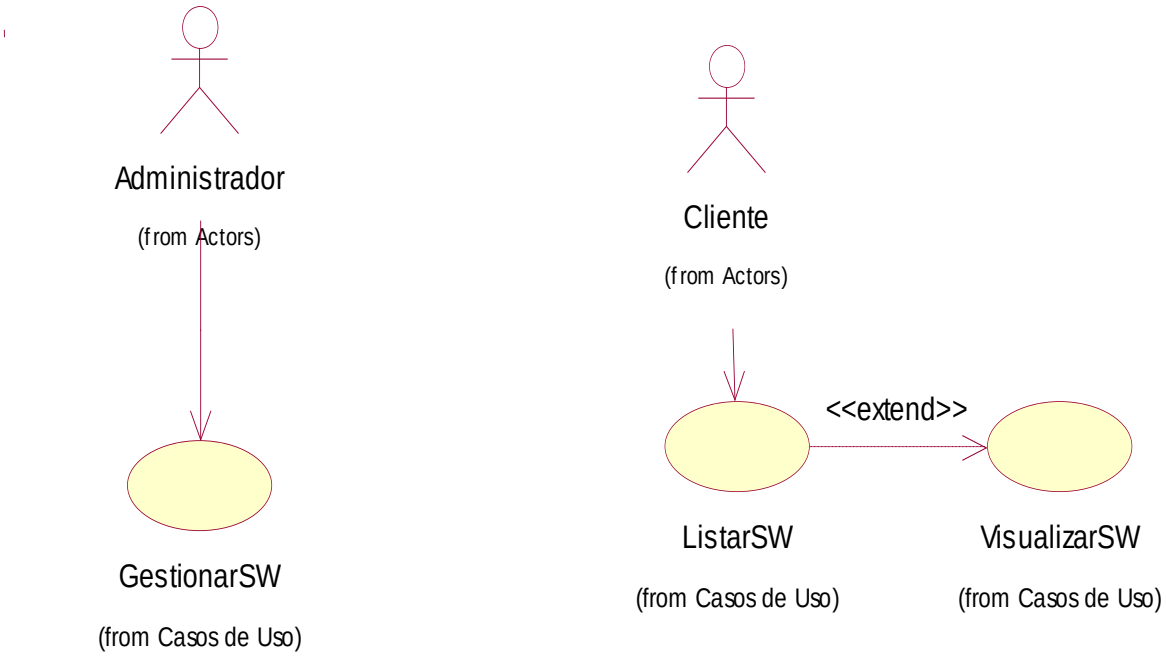
Tabla 3. Listado de casos de uso del sistema: **Listar Servicios Web**

CU2	Listar Servicios Web
Actor	Cliente_Aplicación
Descripción	El actor solicita visualizar todos los Servicios Web que se encuentran en el sistema.
Referencia	RF2

Tabla 4. Listado de casos de uso del sistema: **Visualizar Datos Servicio Web**

CU3	Visualizar Datos Servicio Web
Actor	Cliente_Aplicación
Descripción	El actor selecciona un Servicio Web de los que se encuentran en el sistema y visualiza todos los datos que posee el sistema sobre este.
Referencia	RF2

2.7.3 Diagrama de casos de uso



Figura#4. Diagrama de Casos de Uso del Sistema

2.8 Conclusiones

En este capitulo se expusieron las principales características del sistema y se definieron los actores que existen. Además se determinaron las funcionalidades y requisitos con los que debe cumplir el sistema y se determinó la propuesta para el mismo.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

3.1 Introducción

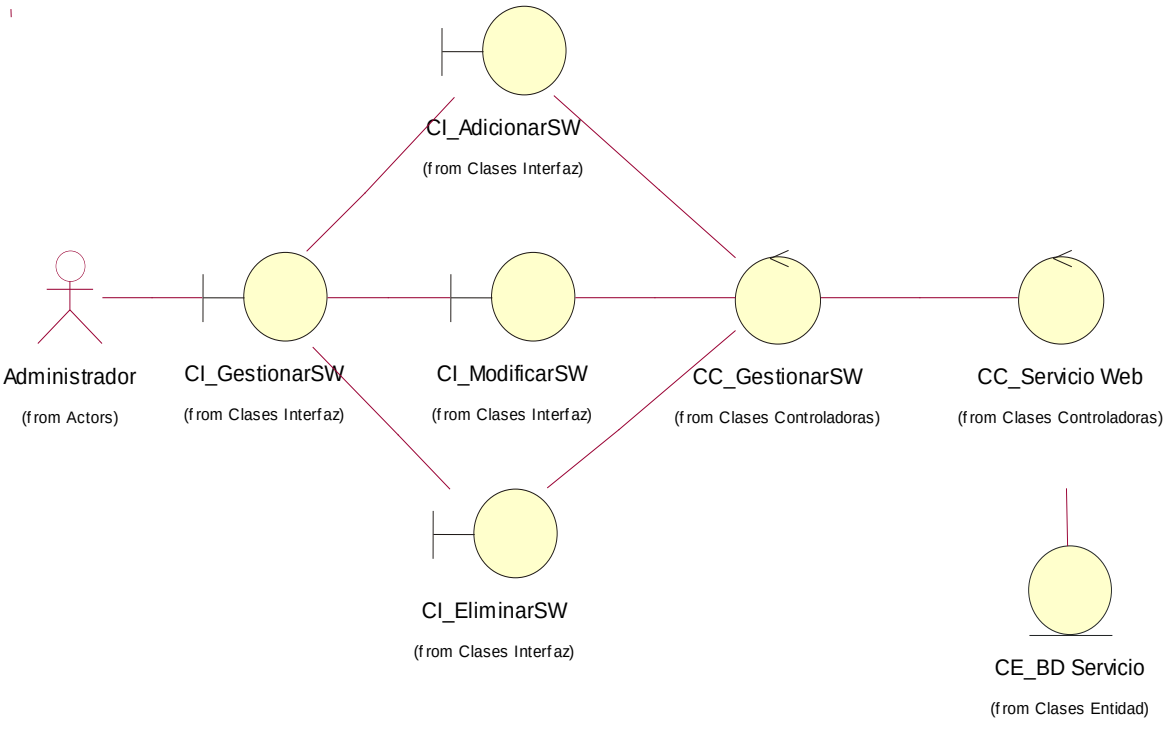
En este capitulo se lleva a cabo la descripción de los procesos que se desarrollan en el flujo de trabajo de análisis y diseño. Se expondrán los diagramas que permiten la descripción de análisis y el diseño así como los que representan la interacción entre objetos.

3.2 Análisis.

3.2.1 Definición del modelo de análisis. Modelo de clases de análisis.

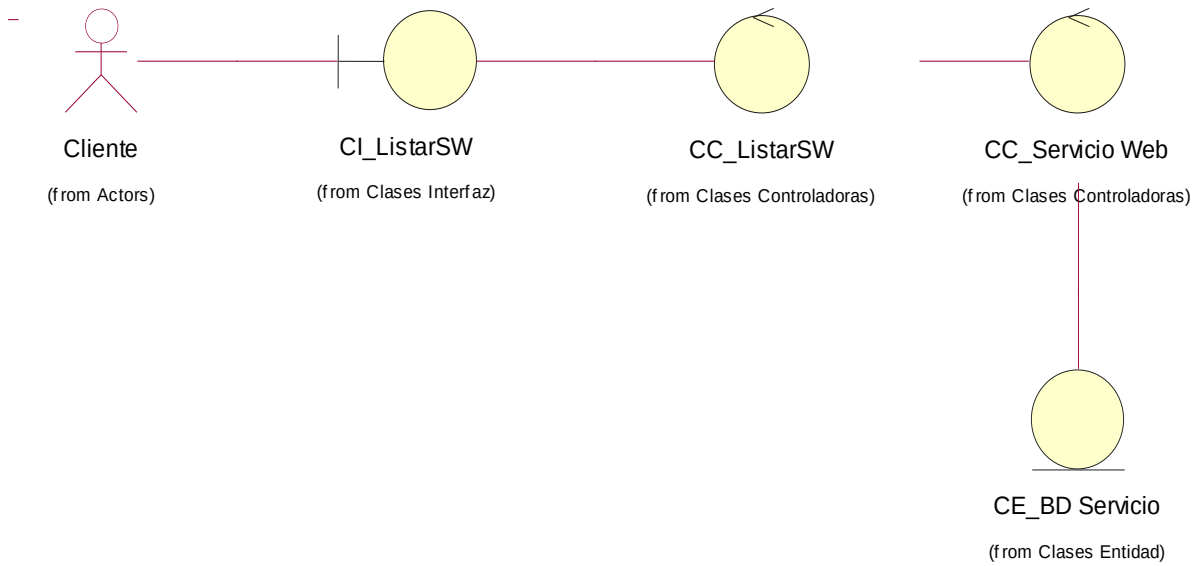
Los diagramas de clases del análisis no llegan al nivel de detalle necesario para el diseño pero si brindan una aproximación representando los conceptos en un dominio del problema.

Diagrama de Clases del Análisis: CU Gestionar Servicios Web



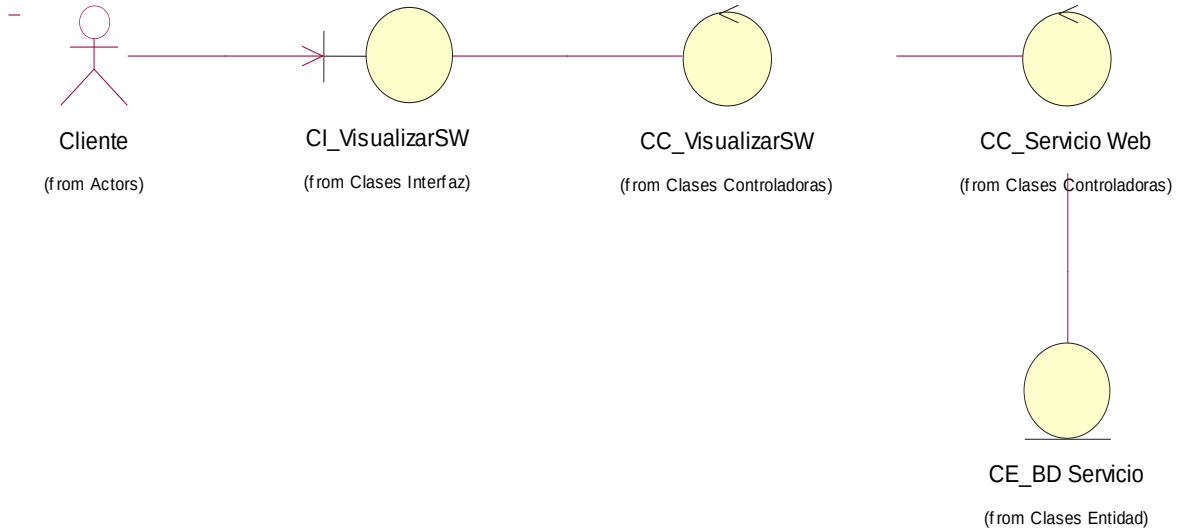
Figura#5. Diagrama de Clases de Análisis: Gestionar Servicios Web

Diagrama de Clases del Análisis: CU Listar Servicios Web



Figura#6. Diagrama de Clases del Análisis: Listar Servicios Web

Diagrama de Clases del Análisis: CU Visualizar Datos Servicio Web



Figura#7. Diagrama de Clases del Análisis: Visualizar Datos Servicio Web

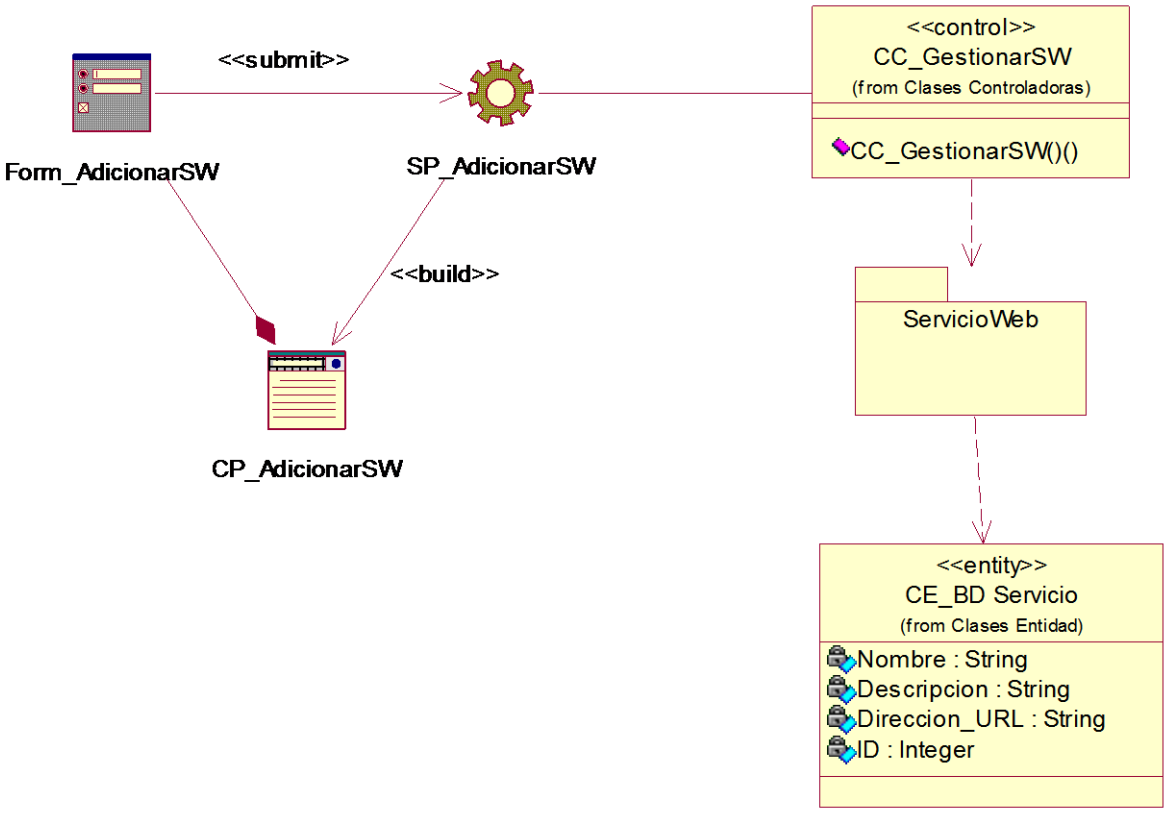
3.2.2 Diagramas de Colaboración del Análisis

Ver Anexo 2

3.3 Diseño.

3.3.1 Diagramas de Clases del Diseño

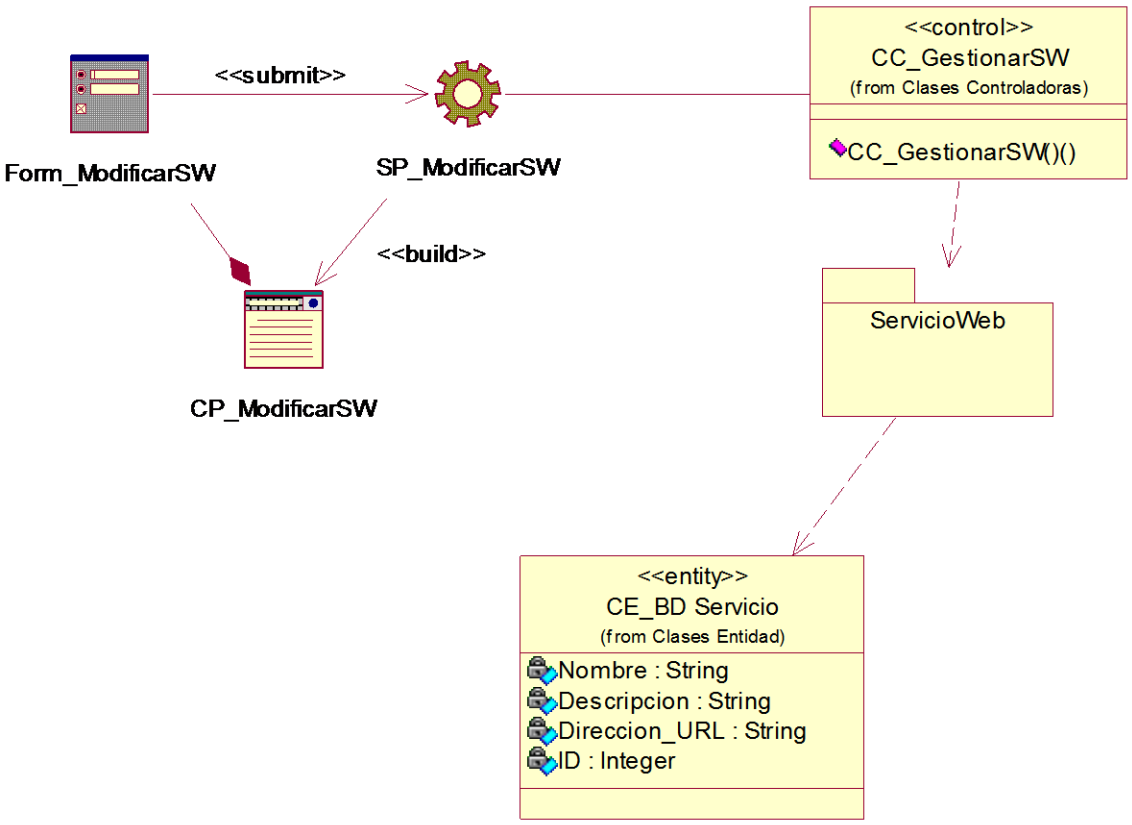
Diagrama de Clases del Diseño: CU Gestionar Servicios Web: Adicionar Servicio Web



Figura#13. Diagrama de Clases del Diseño: Gestionar Servicios Web

Escenario :Adicionar Servicio Web

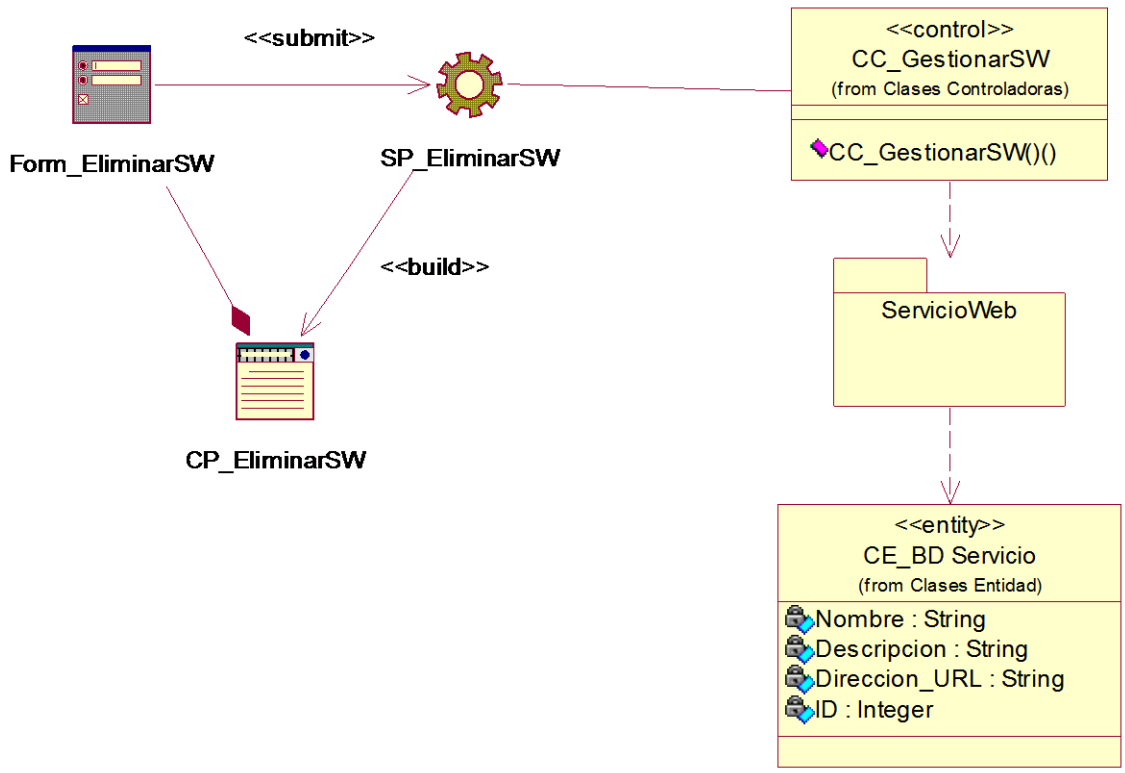
Diagrama de Clases del Diseño: CU Gestionar Servicios Web: Modificar Servicio Web



Figura#14. Diagrama de Clases del Diseño: Gestionar Servicios Web

Escenario :Modificar Servicio Web

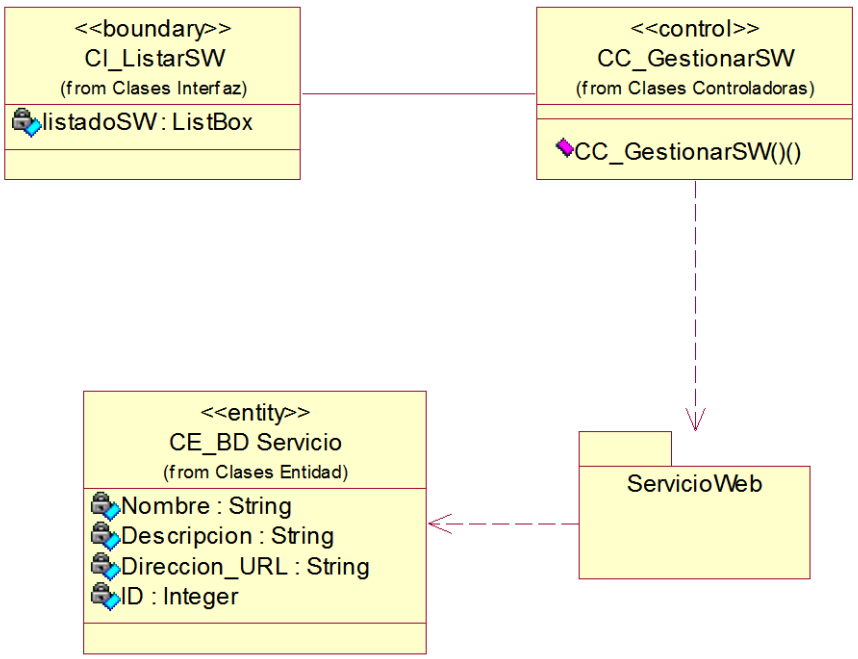
Diagrama de Clases del Diseño: CU Gestionar Servicios Web: Eliminar Servicio Web



Figura#15. Diagrama de Clases del Diseño: Gestionar Servicios Web

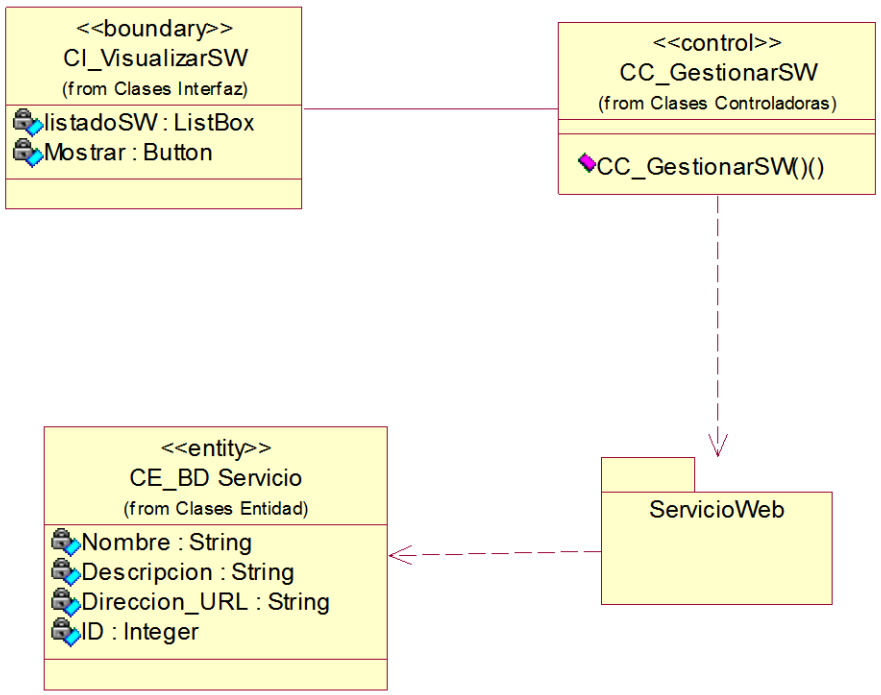
Escenario :Eliminar Servicio Web

Diagrama de Clases del Diseño: CU Listar Servicios Web



Figura#16. Diagrama de Clases del Diseño: Listar Servicios Web

Diagrama de Clases del Diseño: CU Visualizar Datos Servicio Web



Figura#17. Diagrama de Clases del Diseño: Visualizar Datos Servicio Web

3.3.2 Diagramas de secuencia del diseño

Anexo 3

3.3.3 Diseño de la BD

Tabla 8. Descripción de las tablas de la base de datos.

Nombre: Servicio Web		
Descripción: En esta tabla se almacenan todos los datos correspondientes a los servicios web que manejará el sistema.		
Atributo	Tipo	Descripción
ServicioWeb_ID	Int	Es el identificador de cada Servicio Web almacenado.

Nombre	Text	El nombre que recibe el Servicio Web.
Descripción	Text	Esta descripción contendrá datos de interés y que faciliten la comprensión de las funciones que brinda el Servicio Web.
Dirección_URL	Varchar	La dirección URL a través de la cual se puede acceder al Servicio Web.

3.3.4 Definiciones de diseño que se apliquen.

Un patrón es una descripción de un problema y su solución, la cual que recibe un nombre y que tiene la facilidad de que puede emplearse en otros contextos; indicando teóricamente la manera de utilizarlo. Muchos patrones ofrecen orientación sobre cómo asignar las responsabilidades a los objetos ante determinada categoría de problemas.

Patrones GRASP (General Responsibility Asignment Software Patterns)

Los patrones GRASP permiten describir los principios fundamentales de asignación de responsabilidades a objetos. Estos patrones son esenciales en el diseño eficaz de un software. Para el desarrollo del sistema se han utilizado algunos de ellos lo que permite fortalecer el diseño del software y obtener un producto de mayor calidad.

Experto

Este patrón GRASP de diseño define asignar la responsabilidad a la clase que cuente con la información necesaria para manejar esta responsabilidad. Es necesario que la asignación de responsabilidades a las clases se lleve a cabo de una forma adecuada, lo que permitirá que el sistema sea más fácil de entender así como que se puedan reutilizar componentes.

Ventajas:

- Se conserva el encapsulamiento ya que los objetos se valen de su propia información para hacer lo que se les pide.

- El comportamiento se distribuye entre las clases que cuentan con la información requerida permitiendo la creación de clases sencillas y más cohesivas. Se brinda soporte a una alta cohesión.

La utilización de este patrón en el sistema permitió que al estar asignadas las responsabilidades a las clases que realmente las pueden manejar el tiempo de respuesta a las peticiones realizadas desde el cliente sea mucho menor.

Creador

Este patrón GRASP plantea que se debe asignar a una clase X la responsabilidad de crear una instancia de una clase Y. Su propósito fundamental es encontrar un creador que debemos conectar con el objeto producido en cualquier evento.

Ventajas:

- Disminuye las dependencias con respecto al mantenimiento y mejora la reutilización, brindando un soporte de bajo acoplamiento.

Este patrón facilitó la realización de cambios y mejoras dentro del sistema ya que las dependencias eran menores y se evito la creación de objetos repetidos e innecesarios.

Controlador

Este patrón GRASP establece la asignación de la responsabilidad de manejar los eventos a una clase que será la encargada de controlarlos.

Ventajas:

- Este patrón permite la reutilización de componentes al delegar las responsabilidades en una clase lo que facilita que aunque existan cambios desde el punto de vista de la interfaz del sistema las funcionalidades se mantengan intactas y puedan ser reutilizadas.

Con la utilización de este patrón se logró que los cambios realizados en la interfaz del sistema no afectaran el resto de las funcionalidades que se encontraban implementadas y disminuyó el tiempo de desarrollo del producto.

Patrones GOF (Gang of Four)

Los patrones GOF se clasifican según el propósito para el que han sido definidos:

- Creacionales: solucionan problemas de creación de instancias. Nos ayudan a encapsular y abstraer dicha creación.
- Estructurales: solucionan problemas de composición (agregación) de clases y objetos.
- De Comportamiento: soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan.(Gamma y otros,1994)

Modelo Vista Controlador

Este patrón permite la separación en tres componentes diferentes de los datos de la aplicación, la interfaz de usuario y la lógica de control, lo cual facilita la organización del código.

Ventajas:

- Facilidad para realizar cambios en la aplicación ya que cuando se llevan a cabo cambios en la base de datos, en la programación o en la interfaz solo se ve afectado uno de los componentes y se pueden llevar a cabo sin conocer el funcionamiento del resto de los componentes.

Este patrón contribuyó a un mejor desarrollo de la aplicación disminuyendo el número de afectaciones en el resto de los componentes cuando uno sufría cambios y permitiendo una mejor organización en el desarrollo del sistema.

3.4 Tratamiento de errores

Durante la ejecución de la aplicación pueden surgir algunos errores que impidan el correcto desempeño del software. A estos errores se les brinda tratamiento a través de un análisis del error

ocurrido y desencadenando una serie de acciones para mantener el buen funcionamiento del sistema.

Las siguientes acciones pueden provocar el mal funcionamiento del sistema y para ello se ha elaborado una respuesta que permita que el cliente comprenda con facilidad que esta sucediendo y como se recupera el sistema del fallo.

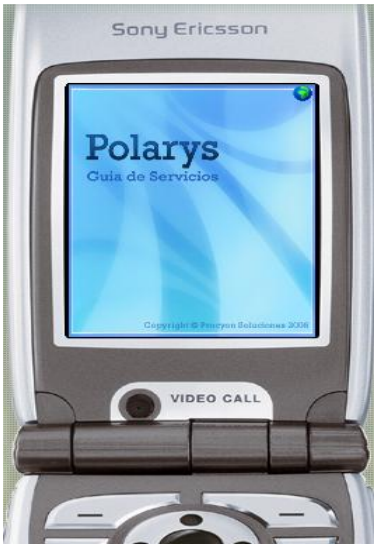
Tabla 9. Tabla de tratamiento de errores

ERROR	CAUSA	RESPUESTA DEL SISTEMA
ERROR_CONEXION_SISTEMA	Ocurre por problemas en la red del proveedor del servicio y ocurren independientemente de las acciones que realice el cliente	El usuario es informado de la falla de la conexión por el sistema y se le sugiere que intente nuevamente.
ERROR_CONEXION_SERVICIOS	Ocurre por problemas técnicos en el servidor del Servicio Web central o por problemas de disponibilidad de los Servicios Web a los cuales brinda acceso.	Se le informa al usuario que el Servicio Web no esta disponible en ese instante y que lo intente nuevamente en un tiempo determinado.

3.5 Interfaz.

El sistema esta compuesto por un Servicio Web central y por una aplicación J2ME que interactúa con este Servicio Web central. La interfaz de usuario de la aplicación fue desarrollada de forma sencilla y utilizando pocas imágenes lo que disminuye el tiempo de respuesta y el consumo de memoria, un factor fundamental en las aplicaciones para dispositivos móviles. Cuenta con diferentes menús de selección utilizando los componentes que brinda J2ME.

Figura#24. Pantalla Presentación



Figura#25. Listado Servicios Web

Figura#26. Datos Servicio Web

3.6 Concepción de la ayuda.

La ayuda del Servicio Web central estará orientada fundamentalmente a la documentación de las clases encargadas de realizar la gestión de los Servicios Web (Adicionar, Modificar, Eliminar) para facilitar el trabajo de los programadores en el desarrollo de una futura versión del sistema que implemente un módulo de administración web. También se documenta el resto de las clases para posibles mejoras en próximas versiones que se quieran desarrollar del sistema.

3.7 Conclusiones

En este capítulo se expusieron los principales artefactos del análisis y diseño del sistema propuesto como solución, en específico los diagramas de clases de análisis y diseño y los diagramas de interacción. Además se describieron los patrones de diseño utilizados y los prototipos de interfaz de usuario.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

4.1 Introducción

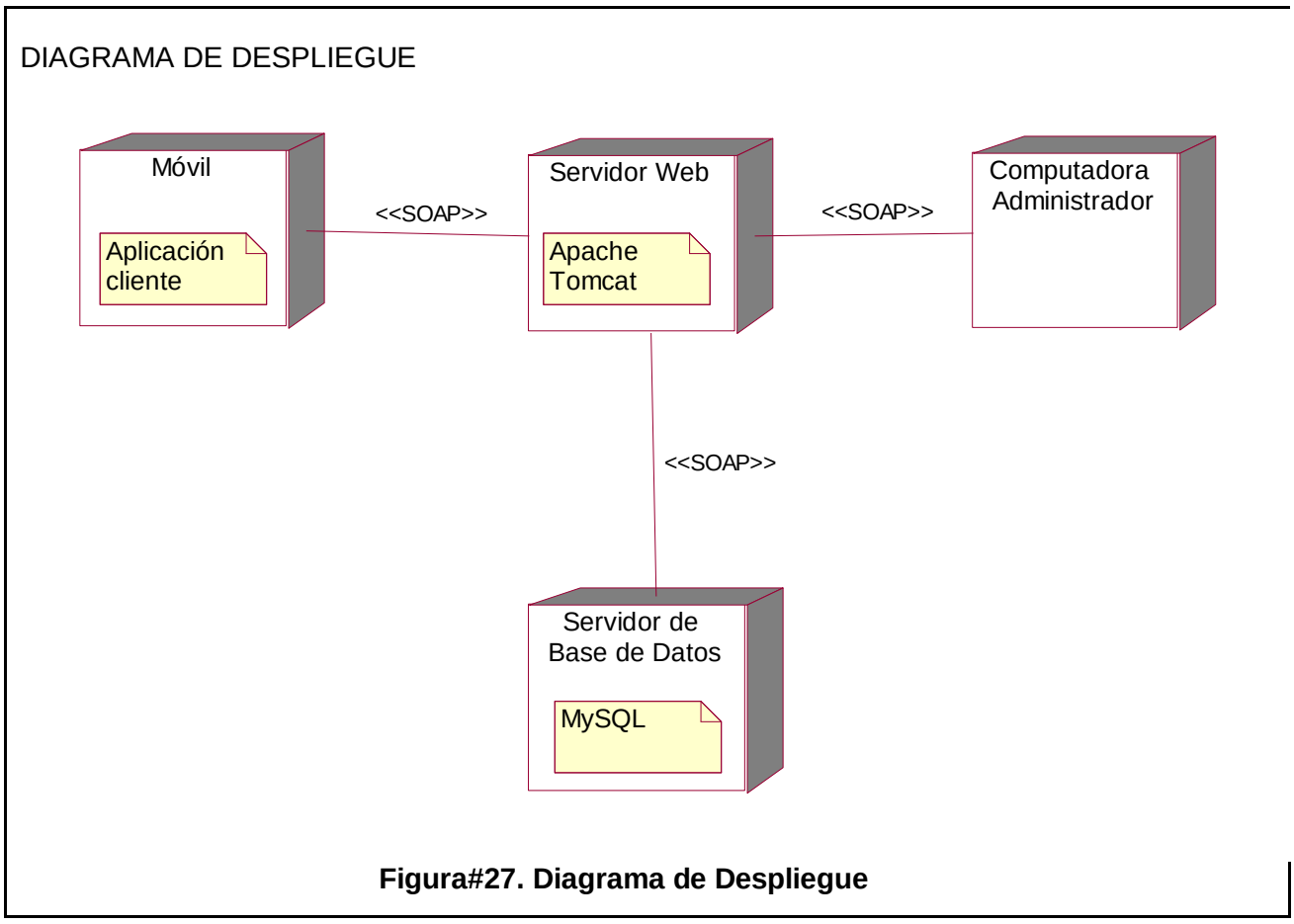
En este capitulo se lleva a cabo la descripción de las principales tareas del flujo de implementación y de prueba, además de los artefactos que se generan en estos flujos.

4.2 Modelo de Implementación

El modelo de despliegue se crea durante las últimas actividades del flujo de trabajo de diseño y provee la descripción de la distribución física del sistema. Se utiliza como entrada fundamental en las actividades de diseño e implementación. El diagrama de componentes por su parte estructura el modelo de implementación y muestran las relaciones entre los elementos de implementación.

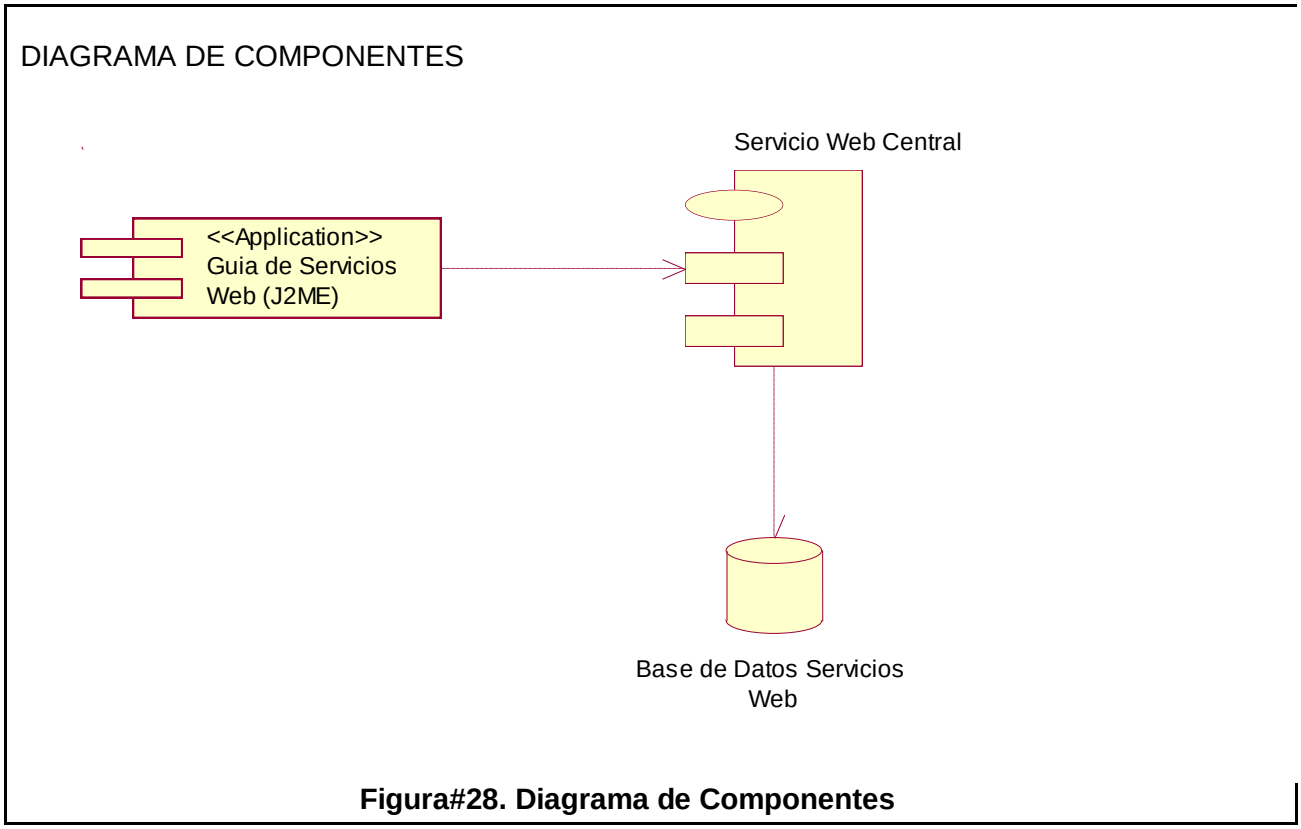
Ambos diagramas conforman el modelo de implementación, que describe como los elementos del diseño se implementan en términos de componentes, ficheros de código fuente, ejecutables, etc.

4.3 Diagrama de despliegue.



Figura#27. Diagrama de Despliegue

4.4 Diagrama de componentes.



Figura#28. Diagrama de Componentes

4.5 Modelo de Pruebas

En este modelo se lleva a cabo la descripción de las pruebas que se realizan a los componentes ejecutables y como deben ser probados aspectos específicos del sistema. Constituye una colección de casos de prueba, procedimiento de prueba y componentes de prueba.

Las pruebas permiten verificar los resultados de la implementación para eliminar los defectos importantes que puedan entorpecer o impedir el funcionamiento del sistema.

4.5.1 Métodos de Prueba

Existen dos tipos de prueba fundamentales: Pruebas de Caja Blanca y Pruebas de Caja Negra.

Las **pruebas de caja blanca** están basadas en el examen de detalles de los procedimientos. Comprueba los caminos lógicos del software ejercitando conjuntos de condiciones o bucles

específicamente. Examina el estado del programa para ver si el estado real coincido con lo que se esperaba del funcionamiento del software.

Las **pruebas de caja negra** se llevan a cabo sobre la interfaz del software. Su objetivo es demostrar que las funciones del software son operativas, que se aceptan las entradas y se producen los resultados correctos. Examina el sistema sin tener mucho en cuenta la estructura lógica interna del software.

4.5.2 Diseño de Casos de Prueba: Caja Negra

Caso de Uso: Listar Servicios Web

Caso de Prueba: CPR1 Listar Servicios Web

Descripción del flujo:

1. Se ejecuta la aplicación

Precondición:

Tabla 10. Caso de Prueba Listar Servicios Web

Clases válidas	**Clases inválidas**	**Resultado esperado**	**Resultado**
Se ejecuta la aplicación en el teléfono móvil		Se muestra el listado de Servicios Web.	Satisfactorio

Caso de Uso: Visualizar Datos Servicio Web

Caso de Prueba: CPR2 Visualizar Datos Servicio Web

Descripción del flujo:

2. Se selecciona el Servicio Web deseado para la ejecución de la acción

Precondición:

Ejecutar el CU Listar Servicios Web

Tabla 11. Caso de Prueba Visualizar Datos Servicio Web

Clases válidas	Clases inválidas	Resultado esperado	Resultado
Se selecciona el Servicio Web: **Servicio Web:** Calculadora		Se muestra la descripción del Servicio Web.	Satisfactorio

4.6 Conclusiones

En este capitulo se expusieron los principales artefactos de los flujos de trabajo de implementación y prueba, en especifico los diagramas de despliegue y de componentes y una descripción de los casos de prueba realizados al sistema.

CAPÍTULO 5: ESTUDIO DE FACTIBILIDAD

5.1 Introducción

En este capitulo se lleva a cabo la planificación del proyecto con el objetivo de obtener una aproximación de los recursos necesarios así como del esfuerzo, costo y tiempo necesario para culminar el proyecto. Para ello se utilizarán métodos de estimación de costos que también permitirán determinar la viabilidad del software.

5.2 Planificación basada en casos de uso

Paso 1. Cálculo de los Puntos de casos de uso desajustados

UUCP = UAW + UUCW

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.

Tabla 12. Factor de peso de los actores sin ajustar (UAW)

Tipo de actor	**Descripción**	**Factor de peso**	**Actores**	**Total**
Simple	Sistema con sistema a través de interfaz de programación.	1	0	0
Medio	Sistema con sistema mediante protocolo de interfaz basada en texto.	2	0	0
Complejo	Persona que interactúa con el sistema mediante interfaz gráfica.	3	2	6
Total				6

UAW = Σ (Factor * Actores)

UAW=6

Tabla 13. Factor de peso de casos de uso sin ajustar (UUCP)

Tipo de CU	Descripción	Peso	Cantidad de CU	Total
Simple	El caso de uso tiene de 1 a 3 transacciones.	5	1	5
Medio	El caso de uso tiene de 4 a 7 transacciones.	10	0	0
Complejo	El caso de uso tiene más de 8 transacciones.	15	2	30
Total				35

UUCW = Σ (Factor * CantCU)

UUCW = 35

UUCP = UAW + UUCW

UUCP = 41

Paso 2. Cálculo de los Puntos de casos de uso ajustados

UCP = UUCP x TCF x EF

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

Tabla 14. Factor de complejidad técnica (TCF)

Factor	Descripción	Peso	Valor asignado	Total
T1	Sistema distribuido	2	5	10
T2	Tiempo de respuesta	1	3	3
T3	Eficiencia del usuario final	1	1	1
T4	Funcionamiento Interno complejo	1	4	4
T5	El código debe ser reutilizable	1	3	3
T6	Facilidad de instalación	0.5	5	2,5
T7	Facilidad de uso	0.5	5	2,5
T8	Portabilidad	2	5	10
T9	Facilidad de cambio	1	4	4
T10	Concurrencia	1	3	3
T11	Incluye objetivos especiales de seguridad	1	0	0
T12	Provee acceso directo a terceras partes	1	3	3
T13	Se requieren facilidades especiales de entrenamiento de usuarios	1	0	0
Total				46

TCF = 0.6 + 0.01 * Σ (Peso * Valor)

TCF = 1.0.

Tabla 15. Factor de ambiente (EF)

Factor	**Descripción**	**Peso**	**Valor asignado**	**Total**
E1	Familiaridad con el modelo de proyecto utilizado	1.5	3	4,5
E2	Experiencia en la aplicación	0.5	3	1,5
E3	Experiencia en la orientación a objetivos.	1	4	4
E4	Capacidad del analista líder.	0.5	5	2,5
E5	Motivación.	1	5	5
E6	Estabilidad de requerimientos	2	2	4
E7	Personal Part–Time	-1	0	0
E8	Dificultad del lenguaje de programación	-1	4	-4
Total				17.5

EF = 1.4 - 0.03 * Σ (Peso * Valor)

EF = 0,875

UCP = UUCP * TCF * EF

UCP = 41 * 1.06 * 0.875

UCP = 38.0275

Paso 3. Estimación de esfuerzo a través de los puntos de casos de uso

E = UCP * CF

Donde:

- UCP: Puntos de Casos de Uso ajustados
- CF: Factor de Conversión
- E: Esfuerzo estimado en horas hombres

CF = 20 Horas-Hombres/Puntos de Casos de Uso

E = UCP * CF

E = 38.0275 * 20

E = 760.55

Paso 4. Calcular el esfuerzo de todo el proyecto

Tabla 16. Esfuerzo Total

Actividad	Porcentaje %	Horas-Hombres
Análisis	20	380,275
Diseño	20	380,275
Implementación	40	760,55
Pruebas	10	190,1375
Sobrecarga (otras actividades)	10	190,1375
Total	100	1901,375

Si ET = 1901.375 horas-hombre y se estima que cada mes tiene como promedio 192 horas laborables, eso daría un ET = 9.90299479 mes-hombre.

Esto quiere decir que 1 persona puede realizar el problema analizado en 10 meses aproximadamente.

Costo del Proyecto.

Se asume como salario promedio mensual $100.00

Donde:

CH: Cantidad de hombres.

Tiempo: Tiempo total del proyecto.

CH = 2 hombres

CHM = 2 * Salario Promedio

CHM = 200.00 $/mes

Costo = CHM * ET / CH

Costo = 200.00 *9.90299479 / 2

Costo = $ 1980,59896 ≈ $ 1980.6

Tiempo = ET / CH

Tiempo = 9.90299479 / 2

Tiempo = 3.30099826 **≈ 3.31 meses**

De los resultados obtenidos se interpreta que con 2 hombres trabajando en el proyecto el mismo se desarrolla en 3.31 meses y su costo total se estima que sea $1980.6.

5.3 Beneficios Tangibles e Intangibles

El sistema esta compuesto por un Servicio Web central y una aplicación desarrollada para comunicarse con este. Ambos tienen el propósito comercial de facilitar la visualización de información obtenida de un Servicio Web en un teléfono celular.

Los principales beneficios del sistema son:

- El usuario final tiene la posibilidad de recibir información de un Servicio Web seleccionado por él en la aplicación y de esta manera obtener noticias, imágenes, estadísticas, etc. desde cualquier lugar donde se encuentre.
- El sistema esta basado en Servicios Web y desarrollado sobre la plataforma Java lo que proporciona independencia del sistema operativo garantizando una rápida migración de software libre a software propietario y viceversa en dependencia de las necesidades de la empresa sin afectar al usuario final.
- El Servicio Web central es capaz de reconocer cualquier tipo de Servicio Web por lo que no es necesario realizar cambios en la lógica del mismo, es por ello que los usuarios pueden visualizar cualquier tipo de Servicio Web con la única condición de que se encuentre registrado en la base de datos del sistema.

Beneficios inmediatos del sistema:

- Estandarización de los Servicios Web lo que facilita el acceso a estas potentes herramientas.

5.4 Análisis de costos y beneficios

El análisis de los costos y beneficios contribuye en la toma de decisiones para determinar si un proyecto es viable o si provocará pérdidas.

Para el desarrollo del sistema se ha trabajado fundamentalmente con tecnología de software libre por lo que los gastos por licencias son mínimos. El sistema permite trabajar con cualquier tipo de Servicio Web lo que estandariza la interacción con estas tecnologías y esta desarrollado para funcionar sobre cualquier plataforma. Ambos puntos constituyen beneficios a tener en cuenta en factibilidad del software.

El costo del proyecto y los beneficios que este brinda a la empresa permiten llegar a la conclusión de que es viable y factible la implementación de este sistema.

5.5 Conclusiones

En este capitulo se llevo a cabo un estudio de la factibilidad del proyecto teniendo en cuenta el posible costo, el tiempo de duración y los posibles beneficios que podrá brindar.

CONCLUSIONES

Con el objetivo de darle cumplimiento a los objetivos generales y específicos de este trabajo se han cumplido cada una de las tareas que fueron trazadas al comienzo del mismo. Se realizó una reseña sobre las generaciones de los dispositivos móviles y sus prestaciones desde su comienzo hasta la actualidad, además se explicó de manera exhaustiva qué es un Servicio Web y las facilidades por las cuales fueron seleccionados para implementar la solución.

Se utilizó la metodología RUP para documentar el proceso de desarrollo del software y una arquitectura Cliente – Servidor, modelo de 3 capas para dar cumplimientos a los objetivos planteados. Se utilizaron diferentes patrones de diseño, tanto GRASP como GOF que nos permitieron desarrollar un producto que cumple con los estándares de calidad requeridos.

Se implementó una aplicación J2ME que se encarga de mostrar la información extraída desde diversos Servicios Web en los teléfonos móviles del usuario y un Servicio Web central que se encarga de estandarizar la comunicación entre otros Servicios Web y la aplicación anteriormente mencionada.

Como fruto de toda la investigación llevada a cabo y la implementación de la solución realizada se puede concluir que se cumplieron todos los objetivos trazados al inicio de nuestro trabajo, además se realizaron recomendaciones de manera que se puedan desarrollar mejoras al sistema en un futuro.

RECOMENDACIONES

- Completar el módulo de administración el cual ha sido diseñado mediante un Servicio Web para ajustarse a las necesidades del cliente, pues utilizando esta variante el sistema puede ser administrado desde una aplicación de escritorio, un sitio web o inclusive desde un móvil en dependencia de las características y necesidades propias del cliente.
- Reforzar el sistema de seguridad utilizado por el módulo de administración del sistema para evitar que esta actividad sea comprometida.

BIBLIOGRAFÍA

1. **Armstrong, Eric, et al. 2005.** *The J2EE 1.4 Tutorial.* 2005.
2. **Centro de Tecnología Informática Universidad de Navarra.** Java. *Centro de Tecnología Informática Universidad de Navarra.* [Online] http://www.unav.es/cti/manuales/Java/indice.html#1.1..
3. **Ciberaula. 2006 .** ¿Qué es Java? *Ciberaula.* [Online] 2006 . http://java.ciberaula.com/articulo/que_es_java/.
4. **—. 2006 .** Introducción al J2ME. *Ciberaula.* [Online] 2006 . http://java.ciberaula.com/articulo/introduccion_j2me/.
5. **Cibernetia.** Conceptos básicos del servidor web. *Cibernetia.* [Online] http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php.
6. **Cid, Jaime. 2007.** *Plataforma JAVA: Especificaciones Evolución.* s.l. : Sun Microsystems Ibérica, 2007.
7. *Cliente / Servidor.* **Reyes-Gavilán, Ignacio González de los.**
8. **Cuenca, Carlos Luis.** Arquitectura del servidor Apache. *desarrolloweb.com.* [Online] http://www.desarrolloweb.com/articulos/1112.php.
9. **Denoncourt, Don. 2003.** Elección del servidor de aplicaciones web . *Help400.* [Online] 2003. http://www.help400.es/asp/scripts/nwart.asp?Num=131&Pag=10&Tip=T.
10. **Departamento de Control de Calidad y Auditoría Informática. 2001.** *Sistemas en Arquitectura Cliente/Servidor.* 2001.
11. **Gálvez Rojas, Sergio and Ortega Díaz, Lucas.** *Java a tope: J2ME (Java 2 Micro Edition).* Electrónica. Málaga : Universidad de Málaga. p. 200. 84-688-4704-6.
12. **Gamma, Erich, et al. 1994.** *Design Patterns: Elements of Reusable Object-Oriented Software.* s.l. : Addison-Wesley, 1994.
13. **Garrido, Daniel Fernández. 2003.** Desmitificando Java (I): La plataforma. *El rincón del Programador.* [Online] 2003. http://www.elrincondelprogramador.com/default.asp?pag=articulos%2Fleer.asp&id=56.
14. **Gayo, Jose Emilio Labra.** *Tecnologías Web.*
15. *Introducción a Rational Rose.* **González Blanco, Rubén and Pérez Tobalina, Sergio.**
16. *Introducción al Modelado Visual, UML y Metodología.* **González Blanco, Rubén and Pérez Tobalina, Sergio.**
17. **J2MEgrasia! 2004.** La arquitectura J2ME. *J2MEgrasia!* [Online] 2004. http://grasia.fdi.ucm.es/j2me/_J2METech/.

18. **Jacobson, Ivar, Booch, Grady and Rumbaugh, James. 2000.** *El proceso unificado de desarrollo de software.* Madrid : Addison Wesley, 2000. p. 438.
19. **Jacobson, Ivar, Booch, Grady and Runbaugh, James. 2000.** *El proceso unificado de desarrollo de software.* s.l. : Addison-Wesley, 2000. 84-7829-036-2.
20. **Java en Castellano. 2007.** Introducción a los Servicios Web en. *Java en Castellano.* [Online] 2007. http://www.programacion.com/java/tutorial/servic_web.
21. **Java.** Mobile Java o Java ME. *Java.* [Online] http://www.java.com/es/download/faq/whatis_j2me.xml.
22. **Korth, Henry F. and Silberschatz, Abraham. 1993.** *Fundamentos de bases de datos.* Madrid : McGraw-Hill, 1993.
23. **LugFi. 2008.** Introducción a los sistemas de control de versiones. [Online] 2008. http://www.lug.fi.uba.ar/documentos/scms/index.php.
24. **masadelante.com. 2008.** ¿Qué es un servidor web (Web Servers)? - Definición de servidor web. *masadelante.com.* [Online] 2008. http://www.masadelante.com/faq-servidor-web.htm.
25. *Metodologías De Desarrollo De Software.* **Sanchez, María A. Mendoza. 2004.** 2004.
26. **MIGUEL, Adoración de and Piattini, Mario G. 1997.** *Fundamentos y modelos de bases de datos.* Madrid : RA-MA, 1997.
27. **Monografias.com. 2007.** Sistemas distribuidos. *Monografias.com.* [Online] 2007. http://www.monografias.com/trabajos16/sistemas-distribuidos/sistemas-distribuidos.shtml#CL.
28. **—. 2003.** Telefonía Celular. *Monografias.com.* [Online] 2003. http://www.monografias.com/trabajos30/telefonia-celular-universitarios/telefonia-celular-u.
29. **Monpeceres, Alberto. 2002.** *Procesos de desarrollo: RUP, XP y FDD.* 2002.
30. **Orallo, Enrique Hernández.** *El Lenguaje Unificado de Modelado (UML).*
31. **OsmosisLatina. 2005.** Subversion . [Online] 2005. http://www.osmosislatina.com/subversion/basico.htm.
32. **Pressman, Roger S.** *Ingeniería del Software: Un enfoque práctico.* s.l. : MacGrawHil. 601.
33. **ProgramaciónWeb.net. 2008.** MVC - Modelo Vista Controlador. *ProgramaciónWeb.net.* [Online] 2008. http://www.programacionweb.net/articulos/articulo/?num=505.
34. **Raga, Charlis. 2007.** Base de Datos. *Monografias.com.* [Online] 2007. http://www.monografias.com/trabajos7/bada/bada.shtml.
35. **Rational. 1998.** *Rational Unified Process Best Practices for Software Development Teams.* 1998.
36. *Rational Unified Process.* **Guerrero, Luis A.**

37. **Rodríguez, Noelia Barreira. 2002.** Introducción a Servicios Web. *El Rincón del Programador.* [Online] 2002. http://www.elrincondelprogramador.com/default.asp?pag=articulos/leer.asp&id=32.
38. **SECURED.** *El modelo cliente-servidor de tres capas para el trabajo en redes de información.*
39. **Serrano, Alberto García. 2007 .** Programación de juegos para móviles con J2ME. *Java en Castellano.* [Online] 2007 . http://www.programacion.com/java/tutorial/ags_j2me/2/.
40. **Soide.** Introducción a J2ME. [Online] http://cinderalla.iespana.es/invierno/introj2me.pdf.
41. **SUPERINTENDENCIA DE TELECOMUNICACIONES DEL ECUADOR.** DEFINICIÓN: TELEFONÍA MÓVIL CELULAR. [Online] http://www.supertel.gov.ec/telecomunicaciones/t_celular/informacion.htm.
42. *Tipos de Plataformas disponibles en Java.* **Queiruga, Claudia. 2005.** 2005.
43. **Toledano, Moisés Daniel Díaz.** Web Services.Introducción y Escenarios para su Uso. [Online] http://www.moisesdaniel.com/es/wri/wsepsu.htm.
44. **Vizcaíno, Aurora, García, Felix Óscar and Caballero, Ismael.** *Una Herramienta CASE para ADOO: Visual Paradigm.*
45. **Web Estilo. 2004 .** Introducción a MySQL. *Web Estilo.* [Online] 2004 . http://www.webestilo.com/mysql/intro.phtml.
46. **World Wide Web Consortium. 2005.** Guía Breve de Servicios Web. *World Wide Web Consortium.* [Online] 2005. http://www.w3c.es/Divulgacion/Guiasbreves/ServiciosWeb.
47. *XML, Servicios Web y Web Semántica.* **Departamento de Informática Universidad de Oviedo.**

ANEXOS

Anexo 1: Descripción detallada de los casos de uso

Tabla 5. Descripción detallada del CUS: Gestionar Servicios Web

Caso de uso	
CU1	Gestionar Servicios Web
Propósito	Gestionar los Servicios Web
Actores: Administrador	
Resumen: El caso de uso se inicia cuando el actor solicita adicionar, modificar o eliminar un Servicio Web, el sistema realiza la acción seleccionada y termina el caso de uso.	
Referencias	RF1
Curso Normal de los Eventos:	
Acción del actor	**Respuesta del sistema**
1. El actor selecciona gestionar Servicios Web.	1.1 El sistema muestra tres opciones: Adicionar, Modificar y Eliminar Servicio Web.
Escenario 1: Adicionar Servicio Web.	
1. El actor selecciona la opción Adicionar Servicio Web.	1.1 El sistema muestra el formulario a llenar.
2. El actor llena los datos del formulario.	2.1 El sistema verifica que los datos son correctos.
	2.2 Si los datos son correctos el sistema adiciona el Servicio Web a la BD y termina el caso de uso.
Curso Alterno:	
Acción 2.1	Si los datos no son correctos el sistema le muestra un mensaje de error y solicita que los datos sean llenados nuevamente.
Escenario 2: Modificar Servicio Web.	
1. El actor selecciona la opción Modificar Servicio Web.	1.1 El sistema muestra los Servicios Web que se pueden modificar.
2. El actor selecciona el Servicio Web a modificar.	2.1 El sistema muestra los datos del Servicio Web seleccionado.
3. El actor modifica los datos.	3.1 El sistema verifica que los datos son correctos
	3.2 Si los datos son correctos el sistema modifica el Servicio Web exitosamente, actualiza la BD y termina el caso de uso.
Curso Alterno:	
Acción 3.1	Si los datos no son correctos el sistema le muestra un mensaje de error y solicita que los datos sean llenados nuevamente.
Escenario 3: Eliminar Servicio Web.	
1. El actor selecciona la opción Eliminar Servicio Web.	1.1 El sistema muestra los Servicios Web.

2. El actor selecciona el Servicio Web a eliminar.	2.1 El sistema muestra los datos del Servicio Web seleccionado.
3. El actor selecciona eliminar Servicio Web.	3.1 El sistema muestra un mensaje de confirmación.
4. El actor acepta eliminar el Servicio Web.	4.1 El sistema elimina el Servicio Web, actualiza la BD y termina el caso de uso.
Curso Alterno:	
Acción 3.1	El actor selecciona la opción cancelar y termina el caso de uso sin ejecutar ninguna acción.

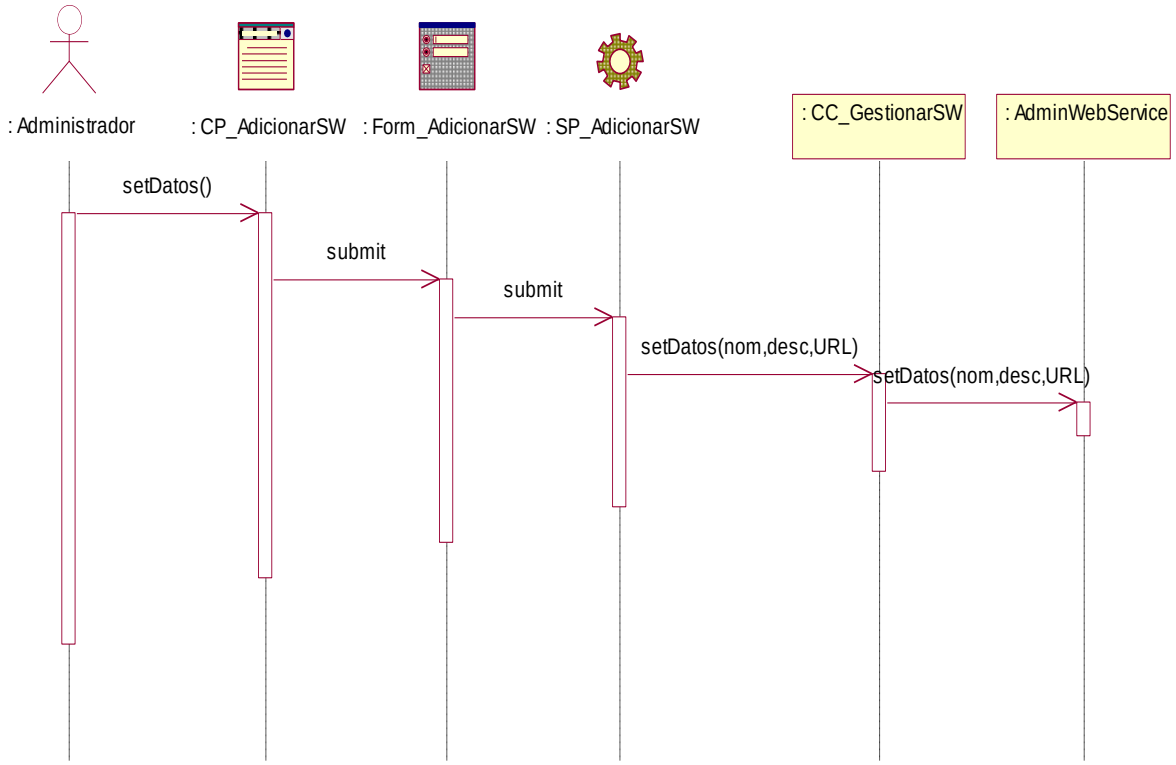
Tabla 6. Descripción detallada del CUS: Listar Servicios Web

Caso de uso	
CU2	Listar Servicios Web
Propósito	Mostrar todos los Servicios Web que se encuentran en la BD.
Actores: Cliente_Aplicación	
Resumen: El caso de uso se inicia cuando el actor solicita que el sistema le muestre todos los Servicios Web que se encuentran en la BD, el sistema recibe la petición y muestra todos los Servicios Web con lo que culmina el caso de uso.	
Referencias	RF2
Curso Normal de los Eventos:	
Acción del actor	**Respuesta del sistema**
1. El usuario solicita listar los Servicios Web.	1.1 El sistema recibe la petición y le muestra todos los Servicios Web que están en la BD.
Puntos de Extensión	
CU Visualizar Datos Servicio Web	

Tabla 7. Descripción detallada del CUS: Visualizar Datos Servicio Web

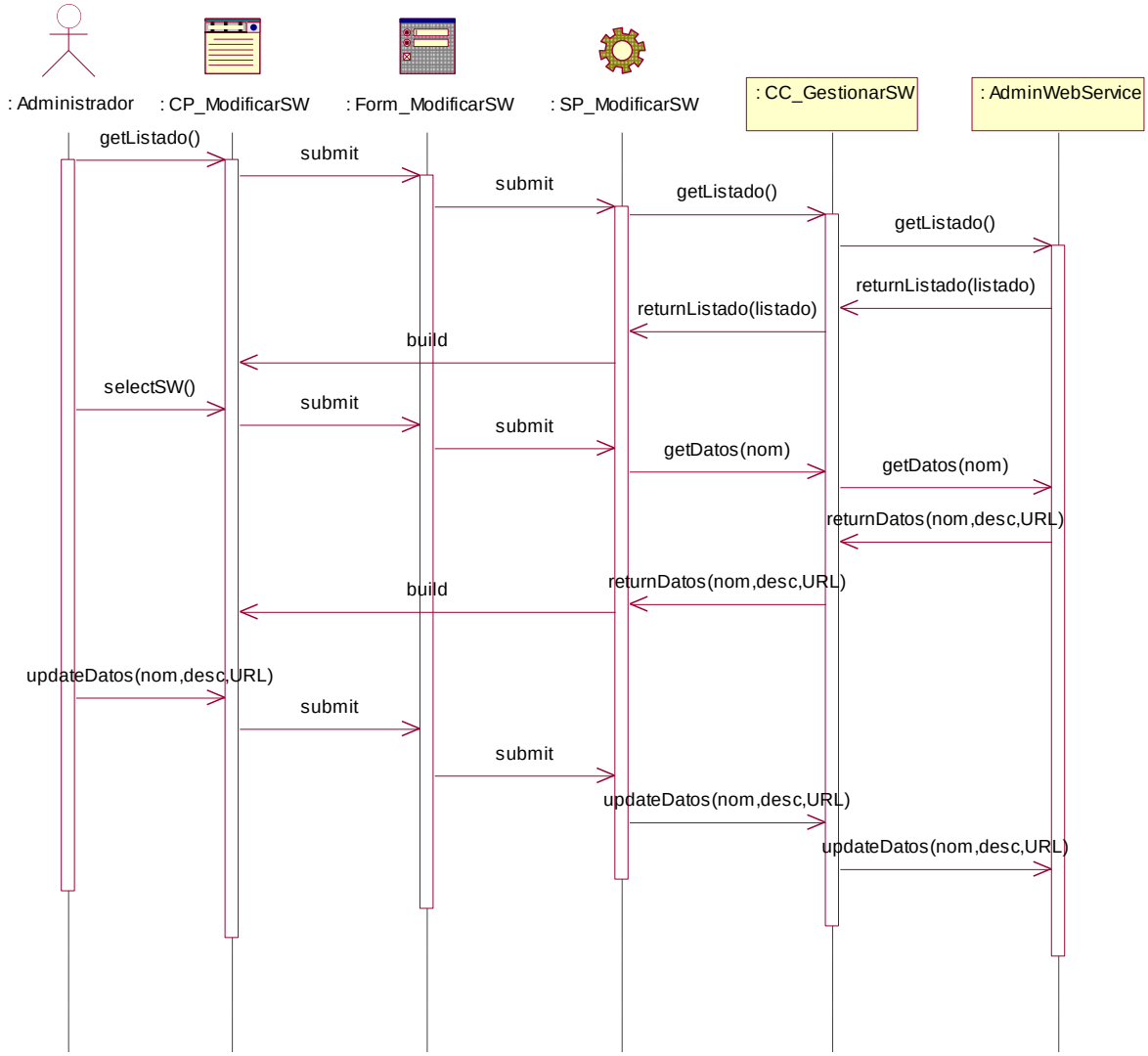
Caso de uso	
CU3	Visualizar Datos Servicio Web
Propósito	Visualizar los datos de un Servicio Web.
Actores: Cliente_Aplicación	
Resumen: El caso de uso se inicia cuando el actor solicita visualizar datos de un Servicio Web seleccionado, el sistema le muestra los datos con lo que termina el caso de uso.	
Referencias	RF2
Precondiciones	Que el cliente haya listado todos los Servicios Web de la BD.
Curso Normal de los Eventos:	
Acción del actor	**Respuesta del sistema**
1. El actor selecciona el Servicio Web.	1.1 El sistema recibe la petición y le muestra los datos del Servicio Web seleccionado.

Anexo 2: Diagramas de secuencia del diseño



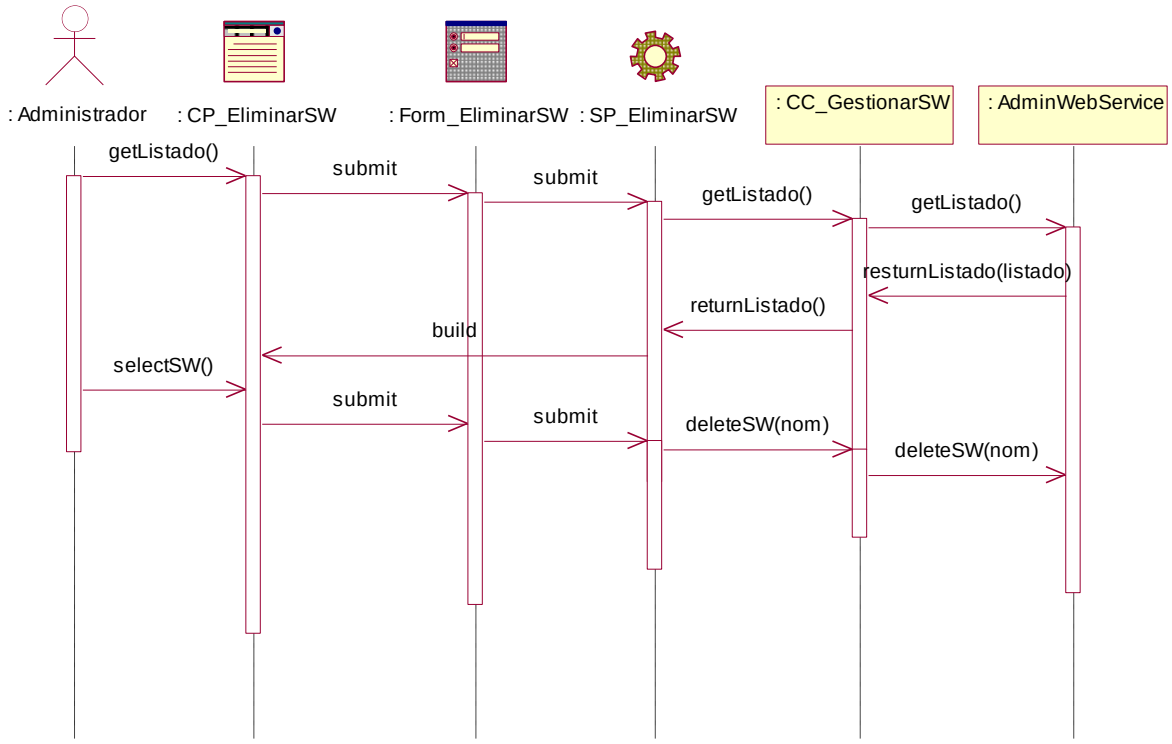
Figura#18. Diagrama de Secuencia del Análisis: Gestionar Servicios Web

Escenario: Adicionar Servicio Web



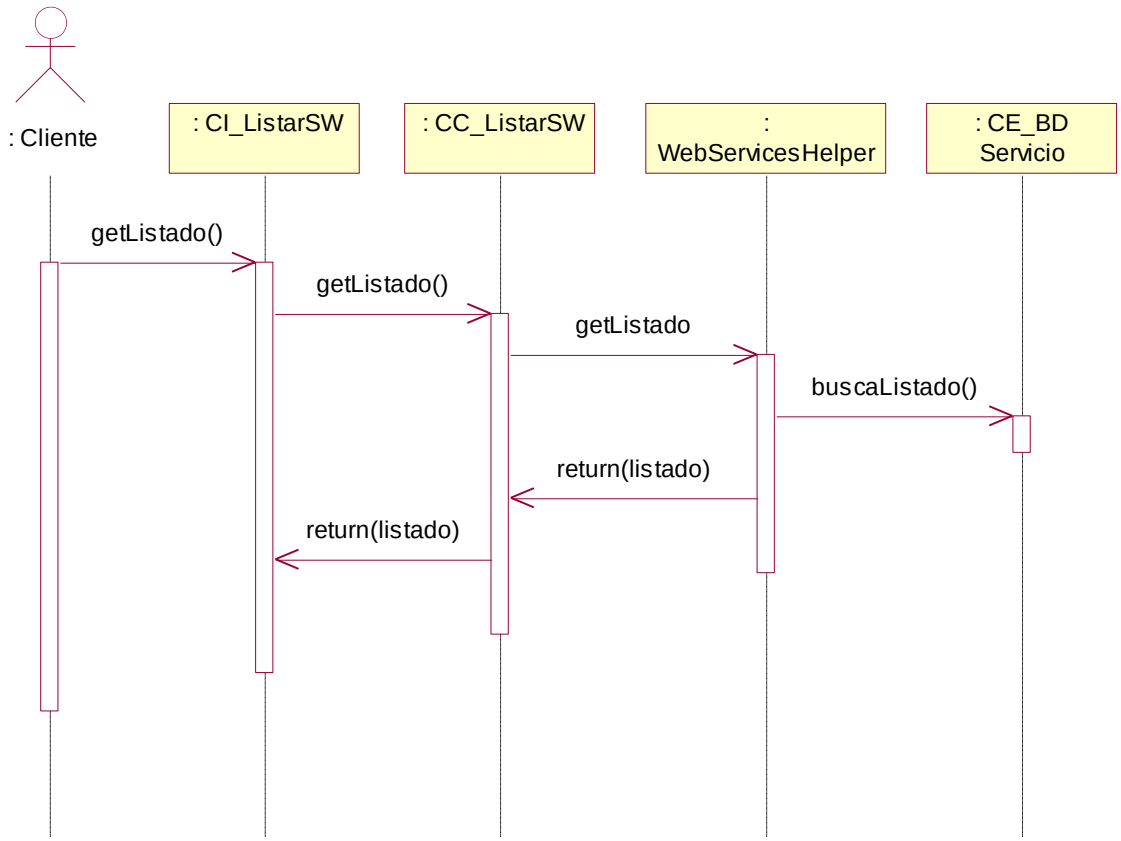
Figura#19. Diagrama de Secuencia del Análisis: Gestionar Servicios Web

Escenario: Modificar Servicio Web

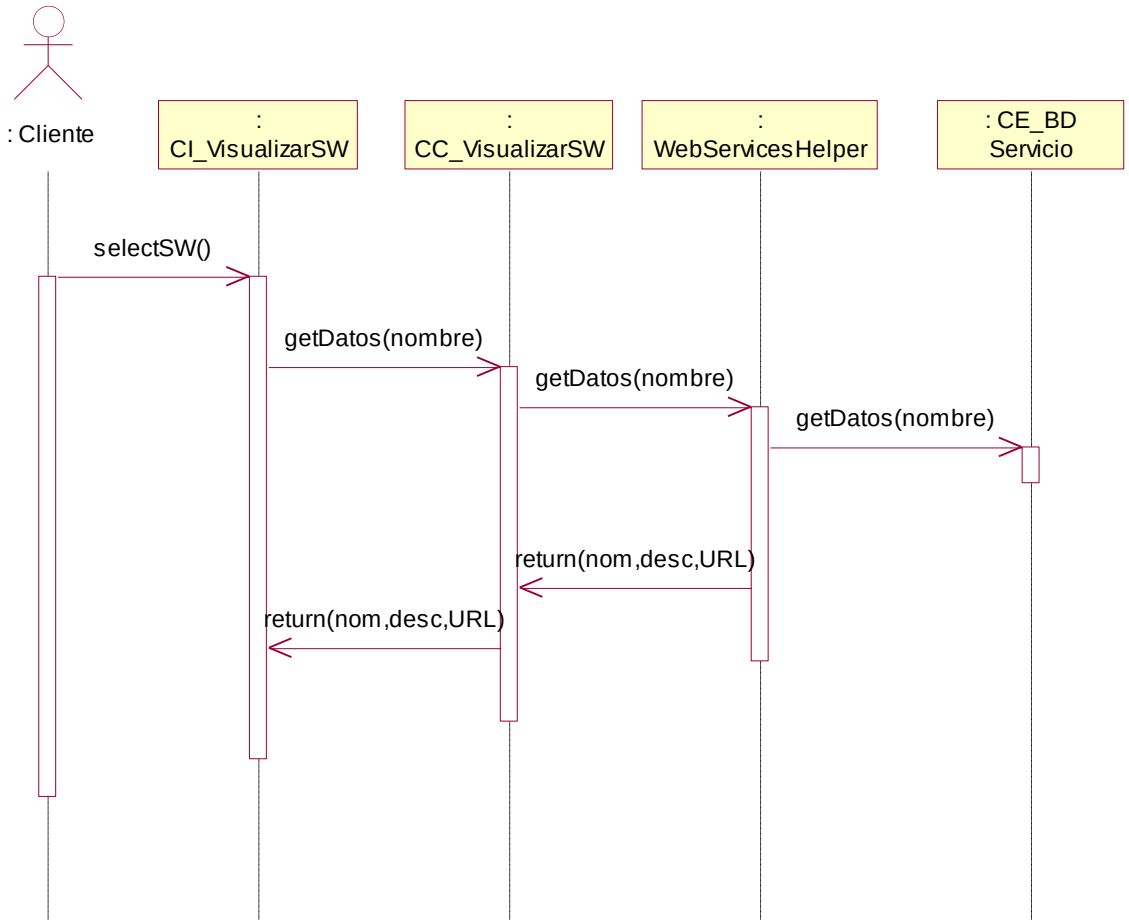


Figura#20. Diagrama de Secuencia del Análisis: Gestionar Servicios Web

Escenario: Eliminar Servicio Web



Figura#22. Diagrama de Secuencia del Análisis: Listar Servicios Web



Figura#23. Diagrama de Secuencia del Análisis: Visualizar Datos Servicio Web

GLOSARIO

AMPS (*Advanced Mobile Phone System*): Red analógica de telefonía móvil de 1ra generación desarrollada por los laboratorios Bell y utilizada principalmente en Estados Unidos, América Latina, Nueva Zelanda, Australia y empleada en Europa (MoviLine) antes del advenimiento de GSM.

API (*Application Programming Interface*): Conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

BYTE CODE: es un código intermedio más abstracto que el código máquina. Habitualmente es tratado como un fichero binario que contiene un programa ejecutable similar a un módulo objeto, que es un fichero binario producido por el compilador cuyo contenido es el código objeto o código máquina.

C++: Lenguaje de programación orientado a objetos, basado en el lenguaje C.

CASE (*Computer Aided Sofware Engineering*): Herramienta para automatizar o apoyar una o mas fases del ciclo de vida del desarrollo de sistemas y destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.

CDC (*Connected Device Configuration*): Conjunto de clases básicas orientadas a realizar implementaciones para la familia de dispositivos con cierta capacidad computacional y de memoria. Con procesadores de 32 bits, en torno a 2Mb o más de memoria y con conexión de red. Ejemplos de estos dispositivos serían sistemas de navegación en automóviles, PDAs de gama alta.

CDMA (*Code Division Multiple Access*): Sistema de telefonía digital inalámbrica basada en la tecnología de acceso múltiple por división por código o de espectro expandido. La conversación se marca con un código y se reparte entre varias frecuencias dentro de la banda. El receptor, a partir del código, reconstruye la señal.

CLDC (*Connected Limited Device Configuration*): Conjunto de clases básicas orientadas a realizar implementaciones para la familia de dispositivos dotados de conexión y con restricciones gráficas, de procesamiento y memoria. Dispositivos con procesadores de 16 o 32 bits, un mínimo de 160KB de memoria no volátil, y por lo menos 32KB de memoria volátil (un total de 192KB de memoria) y dotados de conexión. Ejemplos de estos dispositivos son los teléfonos móviles o las PDAs de gama baja.

CUBACEL: Es la unidad en nuestro país que presta servicio público de radio telefonía celular y de valor agregado, haciendo uso de tecnologías de avanzada (GSM y TDMA) con cobertura nacional.

CVS (*Concurrent Versions System*): Herramienta que permite que varios programadores trabajen de forma colaborativa en un mismo proyecto llevando un control de las versiones de los ficheros. De esta forma se permiten cambios concurrentes en un mismo fichero sin perder lo cambios realizados.

EJB (*Enterprise Java Beans*): Arquitectura que define la forma de construir componentes distribuidos del lado del servidor. Esta tecnología garantiza que los componentes programados sean escalables, eficaces y seguros a pesar de lo sencillo de su desarrollo.

E-MAIL: Servicio de correo electrónico en la red, a través del cual cualquier usuario podrá enviar y recibir mensajes a través de la red.

FRAMEWORK: Esquema, esqueleto o patrón para el desarrollo y/o la implementación de una aplicación. Conjunto de clases que cooperan y forman un diseño reutilizable para un tipo específico de software ofreciendo una guía arquitectónica partiendo el diseño en clases abstractas y definiendo sus responsabilidades y sus colaboraciones.

GARBAGE COLLECTOR: Reciclador de memoria dinámica que se encarga de eliminar de la memoria, los objetos que no están siendo utilizados por nuestra aplicación.

GPL (*General Public License*): Licencia diseñada para garantizar la libertad de compartir y modificar software libre y para asegurar que el software es libre para todos sus usuarios.

GPRS (*General Packet Radio Service*): Tecnología que permite la transmisión de datos a alta velocidad a través de redes digitales inalámbricas de primera generación, como GSM.

GRASP (*General Responsibility Assignment Software Patterns*): Describe los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable.

GSM (*Global System for Mobile Comunication*): Estándar internacional para comunicaciones digitales celulares, utilizado por más de la mitad de los teléfonos móviles del planeta. La familia de sistemas GSM incluye varias frecuencias, la original a 900 MHz, GSM 1800 y GSM 1900.

GUI (*Graphical User Interface*): Componente de una aplicación informática que visualiza el usuario y a través de la cual opera con ella. Está formada por ventanas, botones, menús e iconos, entre otros elementos.

HTTP (*Hypertext Transfer Protocol*): es el protocolo usado en cada transacción de la Web. Esta orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores.

HTTPS (*Hypertext Transfer Protocol Secure*): es un protocolo de red basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto, es decir, es la versión segura de HTTP.

IDE (*Integrated Development Environment*): Programa compuesto por un conjunto de herramientas para un programador. 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.

J2EE (*Java 2 Enterprise Edition*): Define un estándar para el desarrollo de aplicaciones empresariales multicapa. Simplifica las aplicaciones empresariales basándolas en componentes modulares y estandarizados, proveyendo un completo conjunto de servicios a estos componentes, y manejando muchos de las funciones de la aplicación de forma automática, sin necesidad de una programación compleja.

J2ME (*Java 2 Micro Edition*): Familia de especificaciones que definen varias versiones minimizadas de la plataforma Java 2; estas versiones minimizadas pueden ser usadas para programar en dispositivos electrónicos; desde teléfonos celulares, en PDAs, hasta en tarjetas inteligentes, etc. Estos dispositivos presentan en común que no disponen de abundante memoria ni mucha potencia en el procesamiento.

J2SE (*Java 2 Standard Edition*): es una colección de APIs del lenguaje de programación Java útiles para muchos programas de la Plataforma Java.

JAX-RPC: API Java para RPC basado en XML que hace posible escribir aplicaciones Java que usen XML para hacer llamadas a procedimientos remotos (RPC). Está basado en XML y está dirigido a servicios Web.

JSP (*Java Server Page*): Es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.

JVM (*Java Virtual Machine*): es un ejecutable en una plataforma específica, capaz de interpretar y ejecutar instrucciones expresadas en un código binario especial (Java bytecode), el cual es generado por el compilador del lenguaje Java.

KVM (*Kilo Virtual Machine*): Forma parte de la versión más reducida del entorno de ejecución y se incluye en el software de la Plataforma J2ME; se usa en dispositivos con memoria y potencia de procesador limitadas. Es un motor que ejecuta las aplicaciones y los applets(componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador web) que utilizan tecnología Java.

MIDP (*Mobile Information Device Profile*): Es el perfil para dispositivos de información móviles que combina con la configuración CLDC para proporcionar un entorno de ejecución para dispositivos móviles.

MMS (*Multimedia Mobile Service*): Es un estándar de mensajería que le permite a los teléfonos móviles enviar y recibir contenidos multimedia, incorporando sonido, video, fotos, etc.

OPEN SOURCE: Término con el que se conoce al software distribuido y desarrollado libremente.

PDA (*Personal Digital Assistant*): Organizador personal con memoria entre 2 y 16 MB que funciona como un miniordenador.

PHP: lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica.

PIXEL: Es la menor unidad homogénea en color que forma parte de una imagen digital, ya sea esta una fotografía, un fotograma de vídeo o un gráfico.

PLUGGIN: módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

PORTLETS: Componentes modulares de interfaz de usuario gestionadas y visualizadas en un portal web. Los portlets producen fragmentos de código de marcado que se agregan en una página de un portal.

RAM (*Random Access Memory*): Lugar donde la computadora guarda los datos que está utilizando en el momento presente. El almacenamiento es considerado temporal por que los datos y programas permanecen en ella mientras que la computadora este encendida o no sea reiniciada.

SDK (*Software Development Kit*): Conjunto de herramientas y programas de desarrollo que permite al programador crear sus aplicaciones.

SERVLET: Pequeño programa que corre en un servidor. Por lo general son aplicaciones Java que corren en un entorno de servidor web. Esto es análogo a una aplicación Java que corre en un navegador.

SMS (*Short Message Service*): Tecnología para el envío y recepción de mensajes de texto de hasta 160 caracteres entre teléfonos móviles GSM a través del centro de mensajes del operador.

SQL (*Structured Query Language*): es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

SVN (*Subversion*): Aplicación para el control de versiones que nos permite gestionar los cambios y versiones que realizamos en nuestros desarrollos de una forma sencilla.

TCP (*Transmission Control Protocol*): Uno de los protocolos fundamentales en Internet ya que garantiza que los datos serán entregados en su destino sin errores y en el mismo orden en que se transmitieron. También proporciona un mecanismo para distinguir distintas aplicaciones dentro de una misma máquina, a través del concepto de puerto.

THREADS: hilo de ejecución, en sistemas operativos, es una característica que permite a una aplicación realizar varias tareas concurrentemente. Los hilos de ejecución comparten una serie de recursos como el espacio de memoria, los archivos abiertos, situación de autenticación, etc. Esto permite simplificar el diseño de una aplicación que debe llevar a cabo distintas funciones simultáneamente.

UML (*Unified Modeling Language*): Lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad.

WAP (*Wireless Application Protocol*): Protocolo global y abierto para la transmisión de datos entre dispositivos móviles que permite ofrecer servicios y acceso a determinados contenidos de Internet a los terminales inalámbricos.

XML (*eXtensible Markup Language*): Nivel superior al lenguaje HTML, que permite definir códigos propios, con lo que en teoría no sólo da formato al texto (que es lo que hace HTML), sino que se define el tipo de datos que se introducen.