THËWALA: SISTEMA DE INFORMACIÓN DE PRODUCTOS Y SERVICIOS PARA LAS EMPRESAS PERTENECIENTES A LA CORPORACIÓN PARQUE TECNOLÓGICO DE POPAYÁN

Diego Andrés Asenjo González

Alejandro Ríos Peña

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Telemática

Popayán, Colombia

Mayo de 2.005

THËWALA: SISTEMA DE INFORMACIÓN DE PRODUCTOS Y SERVICIOS PARA LAS EMPRESAS PERTENECIENTES A LA CORPORACIÓN PARQUE TECNOLÓGICO DE POPAYÁN

Diego Andrés Asenjo González

Alejandro Ríos Peña

Documento Final de Trabajo de Grado para optar al título de:

Ingeniero en Electrónica y Telecomunicaciones.

Director

Mario Fernando Solarte Sarasty

Magíster en Ingeniería Telemática

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Telemática

Popayán, Colombia

Mayo de 2.005

Dedicatoria

A mi padre, por la locura; a mi madre, por la conciencia;

a los dos por la pasión.

A mi hermana, por su paciencia;

a mis tíos y abuelo, por su ejemplo;

a mis amigos, por todo;

a la naturaleza y las artes, por su compañía;

y a ella, por la inspiración.

Alejandro

Especialmente, a las tres mujeres más importantes de mi vida:

mi mamá, mi abuela e Isabel.

A todos los amigos, los que confiaron y confían en mí.

Cada ayuda y cada consejo fueron un paso.

Diego Andrés

AGRADECIMIENTOS

En primer lugar, al director de este proyecto, Mario Fernando Solarte Sarasty, por su diligente y amable ayuda, y por su oportuna y profesional asesoría durante todas las etapas del mismo.

A los emprendedores que conforman la empresa comercializadora de la Corporación Incubadora de Empresas de Software de Popayán, Plazatech, por la colaboración prestada durante la Fase de Inicio del proyecto, la cual resultó valiosa para capturar los requerimientos del sistema y caracterizar los procesos de negocio relacionados.

Y finalmente, a todos los desarrolladores y usuarios de Software Libre del mundo, por su incalculable aporte a la construcción de una sociedad con herramientas tecnológicas más justas y asequibles.

CONTENIDO

ILUSTRACIONES

TABLAS

ANEXOS

Anexo I, Herramientas de Software Libre Utilizadas
Anexo II, Arquitectura Completa del Sistema
Anexo III, Artículo de Divulgación
Anexo IV, Modelo de Pruebas del Sistema
Anexo V, Licencia de Documentación Libre GNU
Anexo VI, Distribución del Sistema

0. INTRODUCCIÓN

El presente es el documento final del Trabajo de Grado titulado "**Thëwala: *Sistema de Información de Productos y Servicios para las Empresas de la Corporación Incubadora de Empresas de Software de Popayán***". El propósito fundamental del Trabajo de Grado es "Construir un sistema de información de productos y servicios para las empresas pertenecientes a la Corporación Parque Tecnológico de Popayán, que les permita integrar sus capacidades tecnológicas y comerciales, mejorando su competitividad individual y colectiva; usando para dicha construcción dos tecnologías emergentes: los patrones de diseño y los servicios web".

0.1. Reseña del Proyecto Thëwala

El proyecto Thëwala fue creado con el propósito fundamental de construir un sistema de información de productos y servicios para Parquesfot Popayán, que permita integrar las capacidades tecnológicas y comerciales de todas las empresas que lo conforman, mejorando su competitividad individual y colectiva.

La construcción del sistema de información se planteó desde un principio, como la oportunidad de explorar las posibilidades tecnológicas que le pudieran permitir a un sistema como Thëwala una futura integración con otros sistemas de información relacionados o similares, que se puedan encontrar en el sector ISTIR (Industria del Software y Tecnologías Relacionadas) del país. Es por esto que se utilizaron dos elementos principales como referencia para la concepción técnica del proyecto: los servicios web como tecnología base y los patrones de diseño como práctica de desarrollo, los cuales se explican con detalle en el siguiente capitulo, el Marco Teórico.

Otro objetivo muy importante para los integrantes del equipo de desarrollo del proyecto, integrantes y líderes del Grupo GNU/Linux de la Universidad del Cauca, era demostrar que en esta institución, se podía realizar un proyecto de grado que fuera ***desarrollado y distribuido en su totalidad bajo la filosofía del software libre***. El modelo de desarrollo conocido como Software Libre y las herramientas de este tipo que hicieron posible el desarrollo del proyecto se describen en el Marco Teórico y en el Anexo II, Herramientas de Software Libre Utilizadas.

El proyecto fue planeado y ejecutado tratando de aprovechar el Proceso Unificado como herramienta metodológica para organizar el desarrollo y utilizando Patrones de Diseño de Servicios Web para garantizar la calidad del sistema construido. Fue así como el proyecto se dividió en cuatro fases: Inicio, Preparación, Construcción y Transición. Durante cada una de éstas fases se realizaron diversas actividades, de las cuales se obtuvieron uno a uno los artefactos o productos que se muestran en este documento final y en sus anexos.

Es importante resaltar que la colaboración prestada durante la Fase de Inicio del proyecto por las personas que conforman la empresa comercializadora de Parquesoft Popayán, Plazatech, resultó valiosa para capturar los requerimientos del sistema y caracterizar los procesos de negocio relacionados con los productos y servicios de Parquesoft Popayán.

0.2. Organización de este Documento

El documento muestra un ejemplo de utilización del Proceso Unificado, soportado en Patrones de Diseño para construir un Sistema de Información basado en Servicios Web. Los capítulos 3 al 7 contienen una aproximación a la forma en la que cronológicamente se fueron construyendo los diversos artefactos y modelos. Es importante hacer notar que un punto crítico y, por lo tanto, muy importante en el proceso de desarrollo, fue el paso del análisis al diseño y se puede resaltar la valiosa ayuda de los Patrones de Diseño como una herramienta poderosa para afrontar esta etapa.

A continuación se mencionan cada uno de los capítulos y anexos de éste

documento, con una pequeña descripción de su contenido:

- **Capítulo 0, Introducción:** incluye una reseña del proyecto Thëwala y una descripción de la forma como está organizado este documento.
- **Capitulo 1, Marco Teórico**: muestra el contexto tecnológico bajo el cual se desarrolló el sistema. En primer lugar se presenta una aproximación al Estado del Arte del desarrollo de sistemas de información con Servicios Web y patrones de diseño. En segundo lugar, se menciona la forma como se integran los Patrones de Diseño dentro del Proceso Unificado. Por último, se hace una pequeña exposición de las tecnologías y los modelos de desarrollo de software que se han utilizado en la implementación e implantación del sistema.
- **Capitulo 2, Modelo de negocio:** presenta la descripción de los procesos de negocio de la empresa comercializadora de Parquesoft Popayán, Plazatech, que van a ser soportados por Thëwala. La identificación y posterior descripción de éstos procesos fueron realizadas en base a los objetivos específicos de la empresa Plazatech, ya que ésta es la responsable de la comercialización de los productos y servicios de Parquesoft Popayán.
- **Capitulo 3, Modelo de Casos de Uso:** presenta un resumen de la forma en que se llevó a cabo el proceso de construcción del Modelo de Casos de Uso del Sistema, principal elemento de la captura de requerimientos y base esencial de los Modelos de Análisis y Diseño. Este proceso tuvo tres hitos fundamentales: la identificación de los casos de uso, la descripción extendida de los casos de uso reales y la construcción de los prototipos de interfaz.
- **Capitulo 4, Modelo de Análisis:** resume el proceso de Análisis del sistema, que se dividió principalmente en la construcción de tres artefactos: el diagrama de interacción de los casos de uso de análisis, el diagrama de paquetes de análisis y los diagramas de clases por cada paquete.
- **Capitulo 5, Modelo de Implantación:** muestra la arquitectura del sistema como una colección de nodos físicos conectados entre sí, junto con la distribución de los componentes software en estos nodos. El objetivo de este modelo, es que sirva como una ayuda para estimar las características óptimas del hardware que soportaría el sistema, así como los canales de comunicación

necesarios para su correcta interacción.

- **Capitulo 6, Modelo de Diseño:** muestra el proceso llevado a cabo para diseñar el sistema. La construcción del modelo de diseño se basó en cuatro actividades principales: la identificación de subsistemas, la implementación de un prototipo de aprendizaje, la identificación de los servicios web ofrecidos por el sistema y la construcción de los diagramas de paquetes y de clases del sistema.
- **Capitulo 7, Integración de los Portafolios al Sistema:** resume las directrices e instrucciones para que las empresas de la incubadora integren sus portafolios de productos y servicios al sistema.
- **Capitulo 8, Conclusiones y Trabajos Futuros:** conjunto de observaciones y conclusiones del trabajo llevado a cabo, y las recomendaciones realizadas para los trabajos futuros alrededor de los temas tratados.
- **Capitulo 9, Glosario:** compendio de términos, siglas y expresiones usados en el documento, con sus respectivos significados y descripciones.
- **Capitulo 10, Bibliografía:** banco de referencias a materiales de varios autores, que han servido de soporte a la realización del proyecto.
- **Anexo I, Herramientas de Software Libre Utilizadas:** descripción de cada una de las aplicaciones y programas libres que hicieron posible el desarrollo del proyecto.
- **Anexo II, Arquitectura Completa del Sistema:** recopilación de todos los artefactos del Proceso Unificado que fueron llevados a cabo durante el desarrollo del proyecto.
- **Anexo III, Artículo de Divulgación:** artículo con la información general del proyecto para ser divulgado por diferentes medios de publicación.
- **Anexo IV, Modelo de Pruebas del Sistema:** documentación del plan de pruebas del sistema y los resultados obtenidos.
- **Anexo V, Licencia de Documentación Libre GNU:** copia de la licencia del documento (en inglés).

- **Anexo VI, Distribución del Sistema:** distribución electrónica de la aplicación, tanto en CD como en el sitio web del proyecto, incluyendo la implementación de los prototipos y toda la información relevante del sistema.

1. MARCO TEÓRICO

Este capítulo muestra el contexto tecnológico bajo el cual se desarrolló Thëwala, el Sistema de Información de Productos y Servicios para las empresas de la Corporación Incubadora de Empresas de Software de Popayán. En primer lugar se presenta una aproximación al Estado del Arte respecto a la integración, o uso conjunto de la tecnología emergente y la práctica de desarrollo que han sido utilizadas para desarrollar este sistema: los Servicios Web y los Patrones de Diseño. En segundo lugar, se presenta un sección dedicada exclusivamente a mostrar cómo se integran los Patrones dentro del Diseño del sistema dentro de los procesos metodológicos que han sido seguidos. Por último, se hace un recuento y una pequeña exposición de las tecnologías y los modelos de desarrollos de software que se han utilizado en la implementación e implantación del sistema. Cabe resaltar que todas las tecnologías utilizadas son Software Libre, este factor tan importante, es mencionado también en éste Marco Teórico.

1.1. Estado del Arte del Desarrollo de Sistemas de Información utilizando Servicios Web y Patrones de Diseño.

1.1.1. Sistemas de Información basados en Servicios Web

Un sistema de información es un conjunto de recursos de capital humano y computacional que permite la recolección, almacenamiento, recuperación, difusión y empleo de la información de una empresa u organización con el propósito de hacer eficiente su operación.

Los sistemas de información han evolucionado en gran medida desde su aparición, llegando a convertirse en un factor clave dentro de las actividades de las empresas, y en uno de los nichos comerciales más importantes en el mercado de

la tecnología informática.

Los sistemas de información de productos y servicios son las aplicaciones que sirven como soporte del comercio electrónico. Durante varios años se ha explotado la difusión del Internet como un medio de promoción y venta de productos entre empresas (Business-to-Business, B2B) y entre las empresas y los consumidores directos (Business-to-Costumer, B2C), con una gran rentabilidad. Es por esta razón que se dirigen en el mundo innumerables esfuerzos por encontrar nuevas maneras de aprovechar las tecnologías de información y comunicación para facilitar la comercialización y distribución de productos y servicios. A través de los años, se han empleado múltiples y variadas tecnologías para tratar de alcanzar éstos objetivos. Una de las tecnologías que más se utilizan actualmente para el desarrollo de sistemas de información son las Aplicaciones Web.

Las Aplicaciones Web tienen su origen en la WWW (World Wide Web) o Telaraña Mundial de Información, que es la parte visiblemente más importante de Internet. En sus orígenes, la WWW fue pensada como un medio para desplegar información codificada en lenguaje HTML (Hyper Text Markup Language) que reposaba de manera estática en diversos servidores y podía ser accedida a través de consultas hechas por un navegador valiéndose del protocolo HTTP (Hyper Text Transfer Protocol: Protocolo de Transferencia de Hipertexto).

Actualmente se emplea el modelo de comunicación cliente-servidor (navegador – servidor web) para desarrollar Aplicaciones Web, pero la respuesta a la petición del navegador no es necesariamente una página estática, sino que puede ser el resultado de la ejecución en el servidor de alguna lógica de programación. Es decir, las aplicaciones Web no sólo se encargan de desplegar información, sino que también, deben contener una lógica asociada que permita apoyar algún proceso propio del negocio para el cual fue diseñada [Mar03]. Para el desarrollo de aplicaciones Web se han utilizado múltiples tecnologías [Mar03], entre las que se encuentran:

- **CGI (Common Gateway Interface)**. Fue la primera técnica utilizada para que el contenido de las páginas web se generara de manera dinámica. CGI es un mecanismo de comunicación entre el servidor WEB y una aplicación externa, que puede estar desarrollada en casi cualquier lenguaje y que utiliza sentencias

de impresión para generar las páginas HTML que se envían como respuesta al navegador.

- **Servlets**. Los Servlets pueden ser considerados como una evolución de los CGIs desarrollada por SUN Microsystems como parte de la tecnología JAVA. De forma general, consiste en la ejecución de aplicaciones Java en el motor de Servlets (Servlet engine) el cuál hace parte del servidor Web. Su principal ventaja con respecto a los CGIs es que por cada petición de usuario se crea un hilo, lo cuál disminuye el consumo de recursos del sistema. Esta tecnología se complementa con JSP y hace parte de la arquitectura propuesta por SUN en su plataforma J2EE (Java 2 Enterprise Edition).

- **Lenguajes de Programación Embebidos**. Con la aparición de esta tecnología se cambió el proceso a través del cuál se desarrollan las aplicaciones, ya que facilita el trabajo del diseñador gráfico de la aplicación web, quien no está obligado a tener conocimientos en lenguajes de programación. Este enfoque consiste en insertar pequeños fragmentos de lógica de programación en la estructura HTML de la página, al contrario de lo que se hacía en los CGIs. En este sentido se conocen diferentes alternativas, entre las que se encuentran tecnologías como PHP, ASP y JSP.

- **Servicios Web**. La arquitectura de servicios Web plantea algo mas que una técnica para el desarrollo de aplicaciones web, representa un modelo de computación distribuida para Internet basado en XML (eXtensible Markup Language). Bajo este concepto ya no sólo se trata la comunicación usuario - aplicación, sino que de manera adicional se maneja la interacción aplicación - aplicación.

Los Servicios Web se convierten en uno de los más recientes esfuerzos por encontrar la manera de realizar computación distribuida que aproveche el interés masivo de la industria por habilitar Internet para la realización de negocios y la aparición de estándares que se acercan a una integración de plataformas tecnológicas variadas.

Alan Kotok, en su artículo «Tell me about Web services, and make it quick» («Háble me de Servicios Web y Hágalo Rápido») [Kot02], da una definición de los Servicios Web que se resume en las siguientes líneas:

«Los Servicios Web son funciones de negocio auto-contenidas que operan sobre Internet y se ajustan a especificaciones estrictas para trabajar conjuntamente con otros Servicios Web ó con componentes similares.

«Los Servicios Web son importantes para los negocios porque permiten que sistemas de diferentes compañías interactúen entre sí más fácilmente que antes. En la actualidad, las compañías tienen negocios que requieren mayor cooperación entre proveedores y clientes, e involucran la toma de riesgos compartidos y la realización de alianzas de mercadeo a corto plazo. Los Servicios Web brindan a las compañías la posibilidad de comunicar rápidamente sus sistemas con los de otras compañías y, de esta forma, realizar más negocios electrónicamente, con más socios potenciales, en mayores y más variadas maneras que antes, y a un costo razonable.

«Debido a que los Servicios Web están basados en estándares, todas las partes involucradas trabajan sobre el mismo diseño básico. A partir de ahí, las compañías añaden valor y ventajas competitivas al diseño básico para cumplir con las necesidades específicas de sus clientes.

«Los estándares sobre los que se basan los Servicios Web también hacen posible que muchos desarrolladores de sistemas ingresen al mercado, lo cuál aumenta la competencia y reduce los costos. La competencia entre vendedores también causa una mayor innovación en los productos y servicios ofrecidos a los clientes.»

Los principales estándares tecnológicos con los que se implementan los Servicios Web son los siguientes:

- **XML eXtensible Markup Language.** Este es un lenguaje de marcado al estilo de HTML que pretende dar las pautas generales para la estructuración universal de información. De XML se pueden generar dialectos para adaptarlos a un problema particular de intercambio de información.
- **SOAP (Simple Object Access Protocol).** Es un dialecto de XML que permite a

las aplicaciones enviar mensajes a objetos remotos, así como recibir las respuestas de los mismos.

- **WSDL (Web Service Description Language).** Es al igual que SOAP, un dialecto de XML que contiene información acerca de la interfaz, semántica y administración de una llamada a un Servicio Web.
- **UDDI (Universal Description, Discovery, and Integration).** Es un mecanismo para hacer disponibles los Servicios Web a través de un directorio. Este estándar permite a las empresas registrarse en un directorio de Internet que les ayuda a publicar sus Servicios Web, de tal forma que las compañías puedan encontrarse unas a otras y realizar transacciones en la Web.

Un ejemplo de la aplicación de los Servicios Web en los sistemas de información, puede verse en una cadena de distribución de café: cuando un distribuidor minorista (posiblemente dueño de un gran depósito en una ciudad), quiere contactar a un distribuidor mayorista para conseguir una variedad de café deseada al mejor precio posible, ingresa al sistema de información que probablemente esté en Internet, y a través de su navegador envía una petición de compra de cierto número de libras de café. El medio por el cual se comunican el navegador y el sistema de información del distribuidor mayorista esta basado en peticiones típicas de HTTP.

Para dar respuesta a una petición del distribuidor minorista, por ejemplo, a una petición de compra, el sistema de información del distribuidor mayorista, puede contar con una base de datos del inventario que posee en sus instalaciones de almacenamiento, y además de esto, basado en la potencia y escalabilidad de los Servicios Web, tener la posibilidad de conectarse y realizar consultas a los sistemas de información de sus principales proveedores comunicándose con éstos a través de mensajes SOAP o buscar sistemas de información de otros proveedores además de los conocidos.

Por ejemplo, en el caso de que las existencias propias del distribuidor o de sus proveedores no sean suficientes, el sistema de información está en la capacidad de contactar un servicio de directorio para buscar Servicios Web de otros proveedores que puedan realizar las transacciones necesarias para satisfacer el

pedido del cliente. Después de haber ubicado un servicio, el sistema lo enlaza y se comunica con el, para llevar a buen término la petición hecha por el distribuidor minorista. La Ilustración 1 muestra un diagrama explicativo del ejemplo:

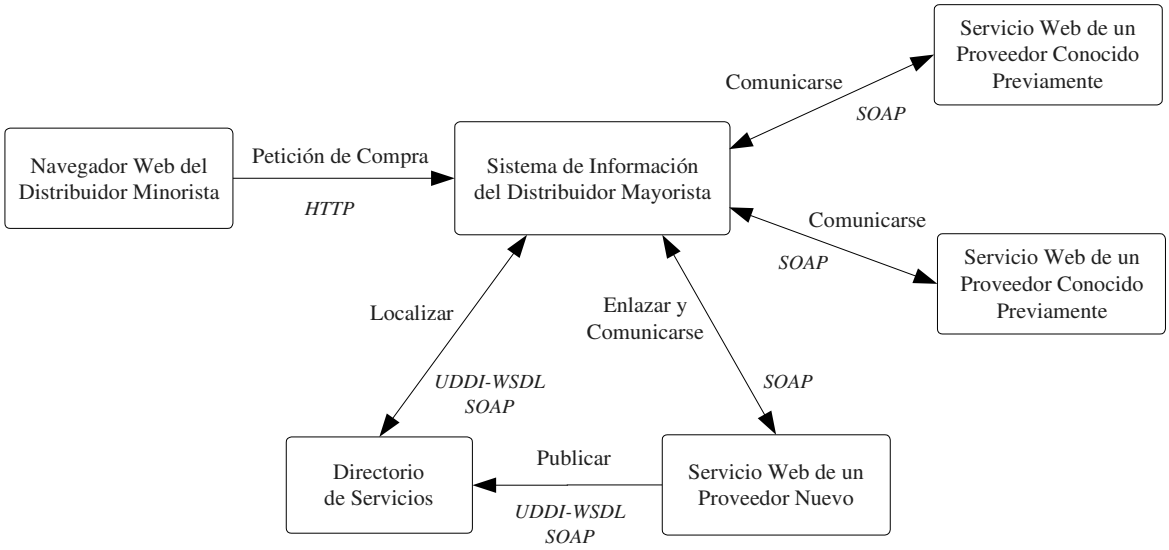

Ilustración 1 - Ejemplo de la aplicación de los Servicios Web.

1.1.2. La Arquitectura de Servicios Web como implementación de varios patrones

Los patrones de diseño son herramientas que proveen facilidades para hacer software reutilizable y de buena calidad. Cada patrón describe un problema que ocurre repetidamente en el entorno y el núcleo de la solución dada a ese problema, de tal forma que ésta pueda ser usada un millón de veces, sin repetir trabajo [Ale77]. Esto sucede porque los diseñadores expertos en orientación a objetos (y también otros diseñadores de software) van formando un amplio repertorio de principios generales y de expresiones que los guían en su proceso de creación de software. A estos principios y expresiones se les puede dar el nombre de patrones si se codifican en un formato estructurado que describe el problema y su solución, y si se les asigna un nombre [Lar98]. Otra definición, bastante simple, dice que un patrón es una pareja de problema/solución que tiene un nombre, es aplicable a muchos contextos, y contiene sugerencias sobre la manera de usarlo en situaciones nuevas [Lar98].

Los patrones de diseño se relacionan con los Servicios Web de tres maneras diferentes: primero, el paradigma de los Servicios Web está soportado en tres patrones que explicaremos brevemente más adelante; segundo, el desarrollo de implementaciones basadas en Servicios Web puede hacerse más fácil aplicando patrones de diseño existentes; y finalmente, algunos patrones cuya aplicación no era muy común en el paradigma de la orientación a objetos, pueden llegar a ser más útiles, educativos e importantes si se aplican al paradigma de los Servicios Web.

Los tres patrones en los que se basa la Arquitectura de los Servicios Web son:

- Arquitectura orientada al servicio
- Adaptador de arquitectura
- Directorio de servicios

Estos tres patrones se explican a continuación.

Arquitectura orientada al servicio

Los servicios web son una implementación de la arquitectura conocida como Arquitectura orientada al servicio. Existen múltiples implementaciones de ésta arquitectura, entre las cuales, la de los servicios web es la que más ha impactado la industria hasta la fecha. Esto se debe a que los servicios web han logrado una mejor estandarización de contratos e interfaces para los servicios que implementaciones como ebXML y CORBA [Mon03]. Las implementaciones de este tipo de arquitecturas hacen énfasis en dos características: transparencia en la implementación y transparencia en la localización. La transparencia en la localización es la habilidad para encontrar y usar un componente que se encuentra en cualquier parte de una red sin tener que modificar el programa que lo necesita dependiendo de la ubicación de éste componente. La transparencia en la implementación es la habilidad para usar un componente sin tener en cuenta el lenguaje o los mecanismos usados por éste para llevar a cabo las funciones especificadas en su interfaz pública.

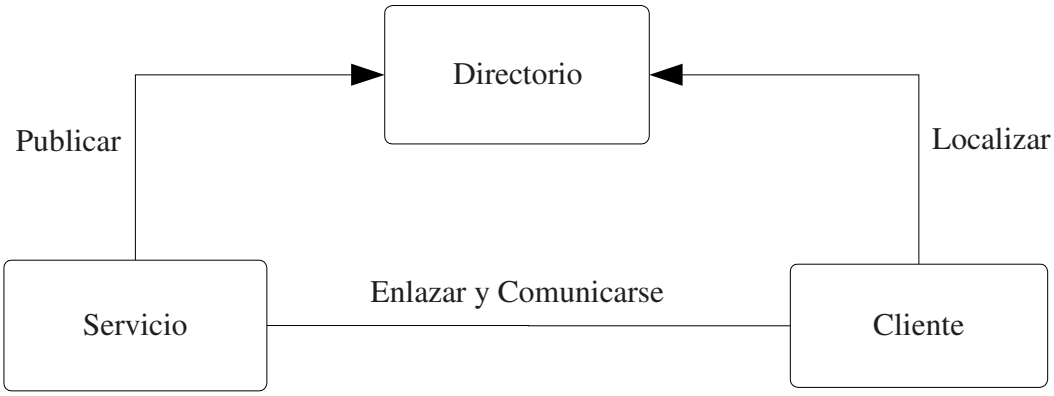

Ilustración 2 - Arquitectura orientada al servicio.

La descripción de alto nivel de una arquitectura orientada al servicio comprende, como se muestra en la Ilustración 2, tres componentes: el servicio, el directorio y el cliente. Se presentan tres tipos de colaboración entre éstos componentes: localización de servicios, publicación de servicios y comunicación entre un cliente y un servicio.

A continuación se describen cada uno de los componentes del patrón de Arquitectura Orientada al Servicio:

- Servicio: En una arquitectura orientada al servicio participan cualquier número de servicios. Cada servicio encapsula una función particular que queda disponible a otros servicios y clientes.
- Directorio: El directorio mantiene información sobre procesos de negocio y servicios que cada negocio publica. También contiene información acerca de los mecanismos de comunicación con cada servicio en términos de su ubicación, interfaz, y detalles acerca de los protocolos a utilizar.
- Cliente: El cliente usa el directorio para localizar los servicios que luego usa para satisfacer sus necesidades de negocio. Un cliente puede ser otro servicio o puede ser un cliente Web de cualquier complejidad.

Un descripción detallada de los tres tipos de interacción presentados en la arquitectura orientada al servicio es:

- Publicación: Un componente publica un servicio haciéndolo disponible a otros a

través de una interfaz dispuesta en el directorio. La publicación de un servicio implica incluir información acerca de las interfaces del servicio y otros detalles adicionales que dependen de cada implementación particular del directorio. Una vez publicado, otros clientes y servicios pueden localizar el servicio a través del directorio.

- Localización: Los clientes potenciales de los servicios los localizan a través del directorio. Los clientes pueden localizar una instancia específica de un servicio o realizar una búsqueda de varios servicios que cumplan con algún criterio en particular, dependiendo de si la implementación del directorio incluye estas características. El directorio responde al cliente con información sobre la forma de enlazar el servicio, sus interfaces y cualquier otra información que el cliente haya solicitado.

- Comunicación: El cliente hace peticiones del servicio a través del protocolo de red especificado en la información obtenida en el directorio de servicios. El servicio atiende la petición y devuelve la información al cliente.

Adaptador de arquitectura

Sin atreverse a dar una definición precisa, se podría decir que una arquitectura es la estructura organizacional de un sistema. Esta estructura puede ser descompuesta recursivamente en partes que interactúan entre si por medio de interfaces, relaciones que conectan las partes, y restricciones para ensamblar las partes. [BCK97]

En el ámbito del software, los sistemas se han basado en diversas arquitecturas que generalmente no tienen buenas capacidades de interoperabilidad, dado que en los inicios estas arquitecturas fueron consideradas sistemas aislados y eran diseñados y construidos de manera ajustada a los requerimientos de información y de software. La tendencia actual que está guiando su diseño e implementación es la consideración de requerimientos del modelo de negocio de las organizaciones para las cuales son construidas, mas aún, el modelo de negocio se rige por la arquitectura.

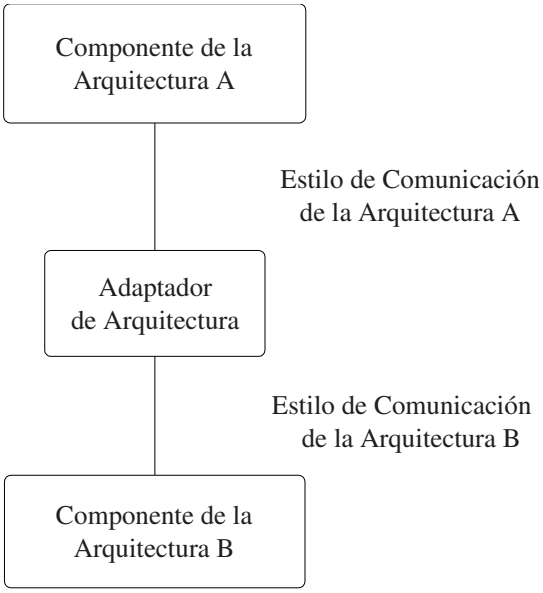

Ilustración 3 – Adaptador de arquitectura.

Como se muestra en la Ilustración 3, La representación más conveniente para un adaptador de arquitectura, es un único componente que cumple con una serie de requerimientos que permiten la comunicación entre dos componentes basados en arquitecturas diferentes, dejando que los detalles de mediación sean tenidos en cuenta en un nivel más bajo.

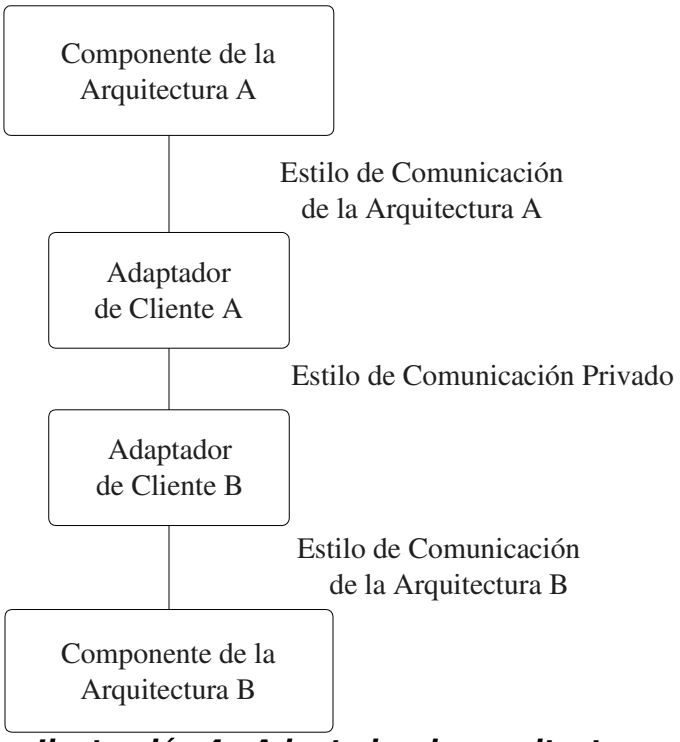

Ilustración 4 - Adaptador de arquitectura detallado.

El adaptador puede ser dividido en dos partes, una que se comunica con el componente de la arquitectura A y otra que se comunica con el componente de la arquitectura B, esta situación puede observarse en la anterior ilustración. Las dos partes implementan sus propios estilos de comunicación dependientes de la arquitectura y emplean un estilo común para comunicarse entre ellas. Esta división, permite que el proceso de comunicación con otras arquitecturas sea más fácil, ya que sólo va a ser necesario crear adaptadores los estilos de comunicación de las nuevas arquitecturas y el estilo de comunicación común.

Esta es una de las características más llamativas de los Servicios Web: poder contar con mecanismos reales de mediación entre componentes basados en diferentes arquitecturas.

Directorio de servicios

Un directorio es uno de los principales componentes de la arquitectura de los

servicios web. El patrón de diseño Directorio de Servicios explica la forma en que éste componente se integra a la arquitectura.

Un directorio de servicios ayuda a las aplicaciones dándoles una única ubicación en la cual encontrar los servicios que requieren basados en criterios de búsqueda estandarizados.

Los directorios de servicios son estructuras planas, asociadas con estructuras que representan la información que almacenan acerca de un servicio. Dependiendo de la cantidad de datos que un directorio de servicios almacena, su estructura interna puede llegar a ser bastante compleja, pero hay que tener en cuenta que de todas formas, por ser un servicio, debe presentar una interfaz única.

En la siguiente ilustración se muestran todos los objetos que participan en un directorio de servicios:

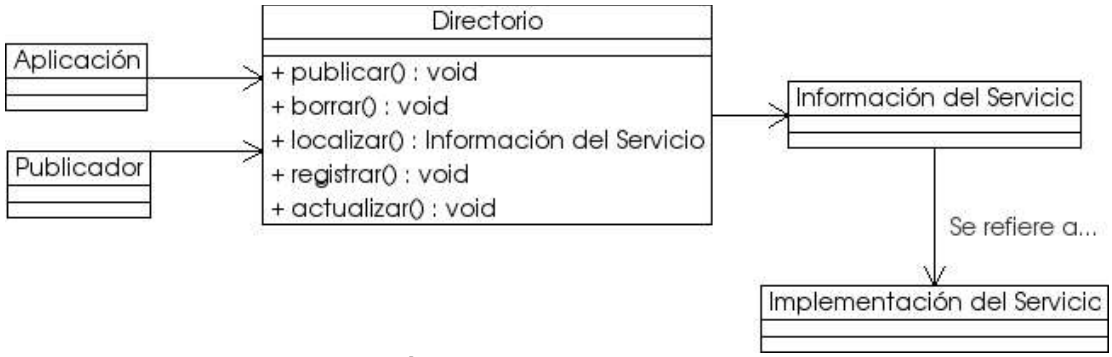

Ilustración 5 - Directorio de servicios.

- Directorio: El directorio de servicios es el principal punto de contacto para publicar, localizar, suscribirse para recibir notificaciones y mantener actualizados los registros del directorio. Para un componente externo al directorio, éste debe mostrarse persistente y disponible todo el tiempo. Los publicadores determinan los contenidos del directorio y las aplicaciones consultan la información de los servicios a través de la misma interfaz.
- Información del Servicio: La información de los servicios es el activo más importante del directorio. La información más común presente en un directorio

es la interfaz del servicio, los estándares implementados por éste, el negocio al que pertenece y alguna información adicional respecto al servicio como tal.

- Aplicación: Una aplicación usa las operaciones de localización y registro del directorio. Las aplicaciones hacen peticiones al directorio para buscar implementaciones de servicios acordes a sus procesos de negocio. Las aplicaciones usan los métodos de notificación para ser advertidos de eventos que impliquen el registro en el directorio, el cambio en la implementación o el cambio de estado de un servicio específico.
- Publicador: Es un componente intermedio que utiliza las operaciones del directorio para publicar información acerca de un servicio. Muchas veces es el servicio como tal quién actúa como publicador de sí mismo, pero considerarlos como componentes separados resulta conveniente, ya que permite un bajo acoplamiento de la implementación del servicio con la implementación del directorio de servicios.
- Implementación del Servicio: El servicio como tal nunca interactúa directamente con el directorio de servicios, a menos de que sea él mismo quién realice la función de publicación. En el directorio residen componentes de información para cada servicio que representan la información acerca de su ubicación e interfaz, así como de otra información dependiente del contexto específico de cada servicio. Generalmente, los servicios no conocen su información contextual o su ubicación.

Las operaciones o colaboraciones entre los participantes del directorio de servicios son publicar, borrar, actualizar, localizar, registrar y notificar.

1.1.3. Patrones comunes en la construcción de Sistemas de Información

Los patrones de diseño no empezaron como patrones ligados al software, nacieron como patrones para construir elementos arquitectónicos, tales como casas, edificios, etc. La primera persona que trató de generar cierta teoría y documentación sobre esto fue el arquitecto Christopher Alexander [Ale77].

En 1995, Erich Gamma, Richar Helm, Ralph Johnson y John Vlissides publicaron el libro "**Design Patterns: Elements of Reusable Object-Oriented Software**"

(“Patrones de Diseño: Elementos de Software Orientado a Objetos Reutilizable”) [GHJV95], en donde recopilaron una serie de patrones aplicados habitualmente por expertos diseñadores de software orientado a objetos. Éste libro tuvo mucho éxito y sus autores fueron desde entonces conocidos como la “pandilla de los cuatro”.

Fue precisamente en ése libro que se establecieron los elementos que un patrón de diseño debe contener [GHJV95]:

- El **nombre**.
- El **problema**.
- La **solución**.
- Las **consecuencias**.

Los principales patrones documentados en este libro son:

Patrones de Comportamiento:

- Iterador.
- Observador.
- Estado.

Patrones de Creación:

- Constructor.
- Fábrica Abstracta.
- Singleton.

Patrones Estructurales:

- Adaptador.
- Fachada.
- Composición.
- Proxy.

Poco después de que esto sucediera, muchos otros diseñadores experimentados

de Software Orientado a Objetos empezaron a plasmar sus diseños en innumerables patrones, de tal forma que hoy es prácticamente imposible recopilarlos todos en un documento.

Una de las referencias importantes para la aplicación de patrones de diseño en desarrollos orientados a objetos la constituyen los patrones GRASP (Patrones Generales de Software para Asignar Responsabilidades), que Craig Larman publicó en 1.998, en su libro “UML y Patrones: Introducción al Análisis y Diseño Orientado a Objetos” [Lar98].

Los cinco principales patrones GRASP mostrados en este libro se han utilizado intensivamente en el diseño de clases aplicando el paradigma de orientación a objetos:

- Alta Cohesión.
- Bajo Acoplamiento.
- Creador.
- Experto.
- Controlador.

A principios del año 2001, Sun Microsystems publicó un conjunto de patrones para ser usados en el contexto del diseño de aplicaciones Java 2 Enterprise Edition, J2EE. La plataforma J2EE está basada en una especificación abierta y se utiliza ampliamente en el desarrollo de sistemas de información empresariales en todo el mundo. Muchas de las soluciones descritas en los patrones de J2EE se basan en los patrones documentados en el libro de la “pandilla de los cuatro”.

Los principales patrones mostrados por Sun son:

Capa de Presentación:

- Controlador Frontal.
- Vista Compuesta

Capa de Negocio:

- Objeto de Valor.
- Delegado de Negocios.
- Fachada.
- Capa de Integración:
- Objetos de Acceso a Datos.

1.1.4. Adaptación de patrones a los sistemas basados en Servicios Web

Actualmente, los patrones de diseño están siendo utilizados con éxito en el desarrollo de sistemas basados en servicios web. Los patrones utilizados para tal fin son generalmente adaptaciones de patrones clásicos y de los patrones desarrollados para diseñar aplicaciones sobre la arquitectura J2EE.

Según Chris Peltz [Pel03], «El desarrollo de aplicaciones basadas en servicios web ha resultado complejo en el ámbito de la Ingeniería de Software moderna. La experiencia ha mostrado que no es suficiente aproximarse a esta tecnología soportándose únicamente en la documentación y las tecnologías de bajo nivel como SOAP y WSDL. Los desarrolladores deben también estudiar las características y los patrones de diseño relacionados con el dominio del problema».

En el año 2003, Paul B. Monday en su libro “Web Services Patterns: Java Edition” [Mon03] se convirtió en uno de los primeros arquitectos de software en mostrar una visión general de cómo se pueden aprovechar los patrones de diseño para estudiar e implementar Servicios Web. Además de los patrones que sirven como base a la Arquitectura de los Servicios Web, mencionados en un apartado anterior, los patrones de diseño expuestos en el libro que fueron estudiados con mayor detenimiento en el marco de este trabajo de grado son:

Adaptación:

- Objeto de Negocio.
- Colección de Objetos de Negocio.

- Proceso de Negocio (Composición).
- Proceso de Negocio Asíncrono.

Refinación de la Estructura:

- Capas Físicas

A continuación se presenta una corta descripción de cada uno de éstos patrones para Servicios Web.

Patrón "Objeto de Negocio"

Problema

¿Cómo exportar Objetos del Negocio autocontenidos desde el Paradigma Orientado a Objetos hacia el paradigma de los Servicios Web?

La razón de ser de un objeto de negocio es representar un concepto de negocio altamente cohesivo, como un pedido, un contacto de negocios, o una empresa. Un objeto de negocio es de naturaleza singular, pero frecuentemente se colecciona en grupos, como lo es una colección de pedidos. El objeto de negocio es una conveniente metáfora de un concepto físico del dominio de negocio[Mon03].

Solución

El patrón propone dos componentes principales para la estructura de un objeto de negocio: ObjetoDeNegocio y TipoDeDatoComplejo. La estructura se muestra en la siguiente ilustración:

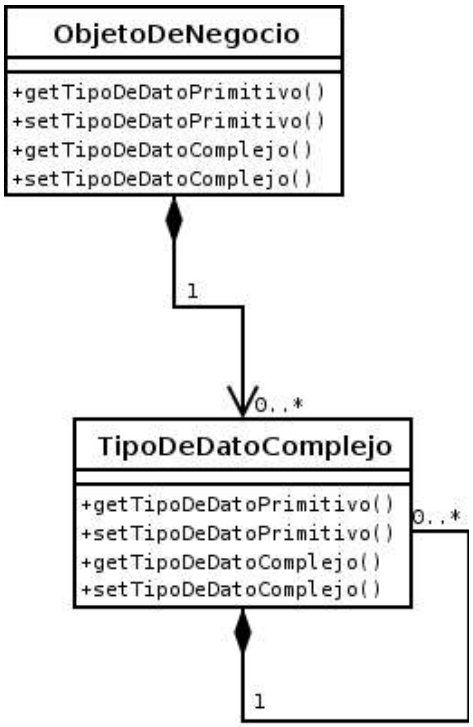

Ilustración 6 - Objeto de negocio.

ObjetoDeNegocio

Esta es la representación de un concepto de negocio de alto nivel. El objeto de negocio abarca sólo propiedades o, en objetos más complejos, operaciones sobre múltiples propiedades. Las propiedades pueden representar datos primitivos, como cadena, entero o valores booleanos. El componente TipoDeDatoComplejo también puede tratarse como una propiedad del objeto de negocio [Mon03].

TipoDeDatoComplejo

Este es un subcomponente de un concepto de negocio de alto nivel que es más complejo que un tipo primitivo de datos. Los tipos complejos de datos pueden contener a su vez a otros tipos complejos de datos [Mon03].

Consecuencias

Cuando se trata de exportar objetos del negocio autocontenidos desde el paradigma orientado a objetos hacia el paradigma de los servicios web, se obtiene un efecto colateral infortunado, desde el punto de vista de la pérdida de riqueza en el diseño orientado a objetos. Esto se debe a que se debe aplanar el modelo de objetos para que sea compatible con otras arquitecturas y lenguajes [Mon03].

Patrón "Colección de Objetos de Negocio"

Problema

¿Cómo construir colecciones de objetos de negocio, para utilizarlas en servicios web?

Las colecciones son muy útiles en todo tipo de aplicaciones, incluyendo los servicios web. Es mucho más común encontrar objetos de negocio reunidos en colecciones que por separado, por ejemplo, cuando se solicitan listados de pedidos, clientes, productos, etc.

Solución

La información almacenada en una colección consiste en objetos de negocio, tal y como se definen en el patrón anterior [Mon03]. El patrón Colección de Objetos de Negocio simplemente agrega una clase adicional de colección a la estructura del objeto de negocio vista en el patrón anterior. La estructura se muestra en la siguiente ilustración:

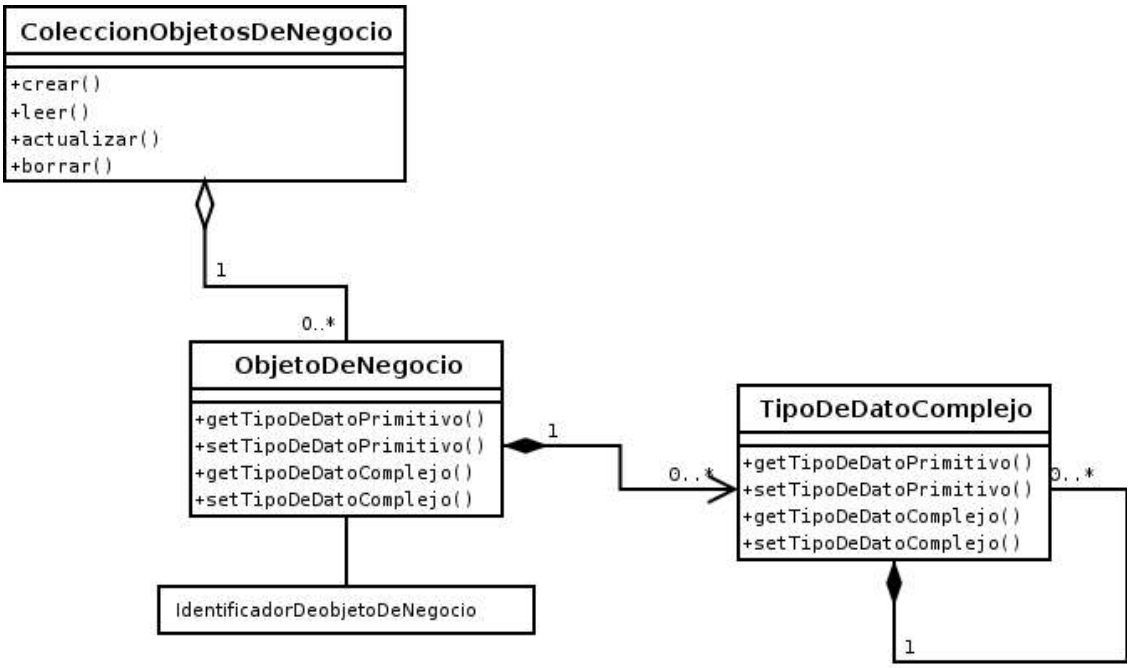

Ilustración 7 - Colección de objetos del negocio.

En particular, se debe notar el uso de un IdentificadorDeObjetoDeNegocio. Los identificadores son útiles para establecer la identidad exacta de un objeto y garantizar su unicidad dentro de la colección.

ColeccionObjetosDeNegocio

La implementación de la colección debe abarcar métodos de ciclo-de-vida y consulta para los objetos que están en la colección. Típicamente, las colecciones ofrecen varias alternativas de métodos de ciclo-de-vida, especialmente los métodos “leer” y “crear”, dependiendo del dominio que sirve la colección [Mon03].

IdentificadorDeObjetoDeNegocio

Es un identificador único para cada objeto de negocio. Este tipo de identificador es útil en aplicaciones empresariales como un mecanismo para forzar la unicidad. Los identificadores de objetos de negocio también ayudan en la transición de objetos complejos a modelos de persistencia relacional [Mon03].

Consecuencias

Usar colecciones de objetos es una práctica extendida en aplicaciones empresariales, y crear una interfaz común hará que las colecciones sean más predecibles y, en consecuencia, utilizables [Mon03].

Patrón "Proceso de Negocio (Composición)"

Problema

¿Cómo debe ser la estructura de una implementación de Procesos de Negocios expuestos como Servicios Web?

Un proceso de negocio es todo aquello que permita la realización de una o más actividades pertenecientes a una práctica de negocio. La diferencia entre un Objeto de Negocio y un Proceso de Negocio, es que el primero es una representación física de entidades de negocio, mientras el segundo se encarga de modelar el negocio de manera global.

Solución

El objetivo de este patrón es ilustrar y proveer una guía para combinar actividades de negocio y formar un único servicio web consumible a través de una interfaz bien definida. El patrón propone los siguientes componentes principales: ActividadDeNegocio, ImplDeActividadDeNegocio, ProcesoDeNegocio, SecuenciaDeActividades, ImplDeProcesoDeNegocio y Datos. La estructura se muestra en la siguiente ilustración:

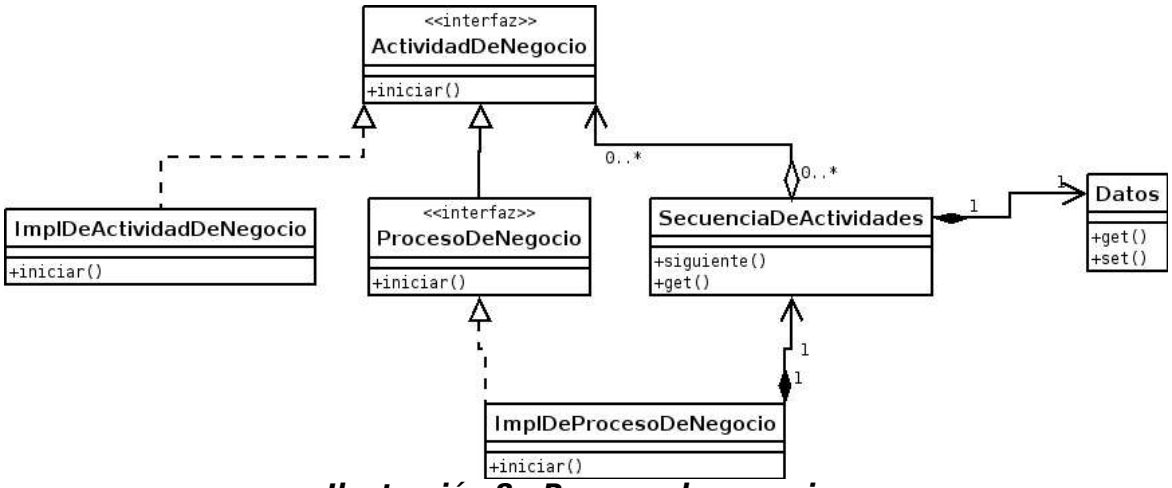

Ilustración 8 - Proceso de negocio.

ProcesoDeNegocio

Esta es la interfaz al proceso de negocio. En la implementación real, pueden existir múltiples interfaces para el mismo proceso de negocio, pero construidas para distintos consumidores. [Mon03].

ImplDeProcesoDeNegocio

Esta es la implementación del servicio correspondiente a la interfaz del proceso de negocio [Mon03].

SecuenciaDeActividades

Esta es la lista de actividades de negocio que deben ocurrir antes de que el proceso de negocio finalice. Esta secuencia no debe ser síncrona por naturaleza, y no se requiere que siga un único camino. Varios pasos en el proceso de negocio pueden ejecutarse asíncronamente, con puntos de sincronización inmersos dentro del proceso [Mon03].

Datos

Cada actividad de negocio puede tener efectos colaterales en forma de nuevos datos que otras actividades de negocio requieren. Al igual que la secuencia, diferentes datos pueden ser encaminados a una actividad de negocios en

particular, dependiendo de los resultados de algunos datos o de la ejecución de otra actividad de negocio [Mon03].

ActividadDeNegocio

Cada actividad de negocio es una unidad de trabajo que es requisito para completar un proceso de negocio. Una actividad de negocio puede ser un proceso de negocio, pero no es requisito que lo sea. El nivel de granularidad de las actividades de negocio es responsabilidad del diseñador de la solución.

Consecuencias

Éste patrón permite estructurar una implementación de un proceso de negocio expuesto como servicio web, ofreciendo la posibilidad de tener diferentes interfaces para cada tipo de consumidor del servicio y organizando las actividades del negocio en una secuencia bien definida. [Mon03].

Patrón "Proceso de Negocio Asíncrono"

Problema

¿Cómo manejar Procesos de Negocios asíncronos expuestos como Servicios Web?

Las llamadas a un proceso de negocio pueden tardar desde unos pocos milisegundos hasta varios días para completarse. Esta variabilidad tiene su origen en la granularidad del proceso de negocio, así como en las actividades de negocio dependientes. Por ejemplo, desde el momento en que un cliente realiza un pedido hasta que éste se encuentra disponible para ser entregado por una empresa de transporte, pueden pasar unas cuantas horas o varios días [Mon03].

Solución

Este patrón es una evolución del anterior, e ilustra la estructura básica de un proceso de negocio asíncrono. Adicionalmente a los componentes identificados en el patrón anterior, dos clases adicionales forman la estructura del patrón Proceso de Negocio Asíncrono: EstadoDeProceso y GestorDeProcesosDeNegocio. La estructura se muestra en la siguiente ilustración:

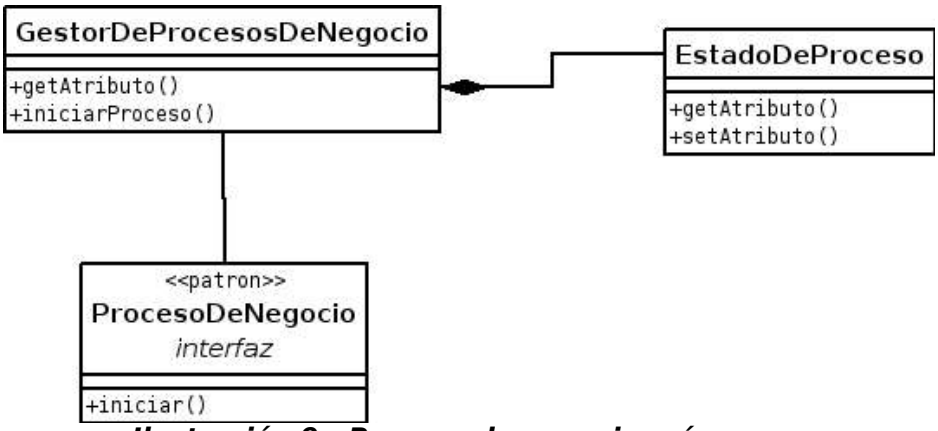

Ilustración 9 - Proceso de negocio asíncrono.

La representación de un proceso de negocio asíncrono asume más una estructura de administración de tareas que una estructura de un proceso de negocio único. Internamente, el servicio lanza tareas y mantiene su estado. La interfaz al GestorDeProcesosDeNegocio contiene métodos tanto para crear procesos de negocio, como para determinar el estado de un proceso de negocio en particular [Mon03].

GestorDeProcesosDeNegocio

Es responsable de exponer una forma de iniciar un proceso de negocio y llamar operaciones sobre un proceso de negocio individual. Un gestor retorna un identificador único al ser creada una nueva instancia de un proceso de negocio, o usa ese identificador para identificar una instancia particular de proceso de negocio. El GestorDeProcesosDeNegocio facilita la habilidad de los clientes para buscar atributos de un proceso de negocio en particular que sea de su interés, tal como el progreso de un pedido [Mon03].

EstadoDeProceso

Las instancias de los procesos de negocio son transitorios por naturaleza, ya que el proceso de negocio tiene puntos de inicio y terminación bien definidos. El EstadoDeProceso mantiene información sobre un proceso de negocio específico y también cualquier información que un cliente o servidor pueda necesitar una vez

concluya el proceso. Esta información también puede resultar valiosa en caso de que el Gestor de Procesos se caiga o sea detenido por mantenimiento del sistema [Mon03].

Consecuencias

Los procesos de negocio tienden a ser largos y ejecutarlos en forma síncrona puede frustrar a los usuarios. Este patrón permite a los procesos de negocio ser ejecutados en un hilo separado del hilo de ejecución del cliente que llama al servicio [Mon03].

Patrón "Capas físicas"

Problema

¿Cómo lograr la separación física de las capas de lógica de negocio y de lógica de presentación Web?

Cuando se utiliza una arquitectura de aplicación de n-capas, generalmente se separa la capa con la lógica del negocio, de la capa con la lógica de presentación. En servicios web, la capa de presentación corresponde a la capa Web. La separación puede ser física, al momento del despliegue, o lógica, durante el desarrollo. Hay múltiples razones para buscar una separación física de las capas Web y de lógica, entre las cuales se encuentran la posibilidad de utilizar mecanismos de manejo de eventos y la posibilidad de reutilización de la lógica del negocio para aplicaciones autocontenidas o independientes de la implementación Web [Mon03].

Solución

El patrón de Capas Físicas propone identificar las capas en las que debe ser divida la aplicación y luego utilizar el patrón "Conector" para enlazar las capas adyacentes. Existen dos aspectos a tener en cuenta respecto a éste patrón: su estructura de componentes y su despliegue físico [Mon03].

A continuación se muestra la estructura de componentes de la implementación del patrón:

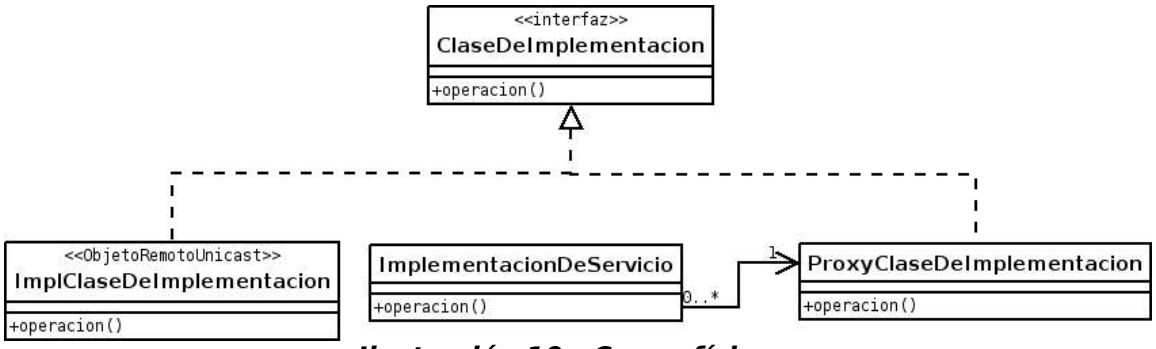

Ilustración 10 - Capas físicas.

A continuación se muestra el despliegue de la estructura del patrón “Conector” dentro del patrón “Capas Físicas”:

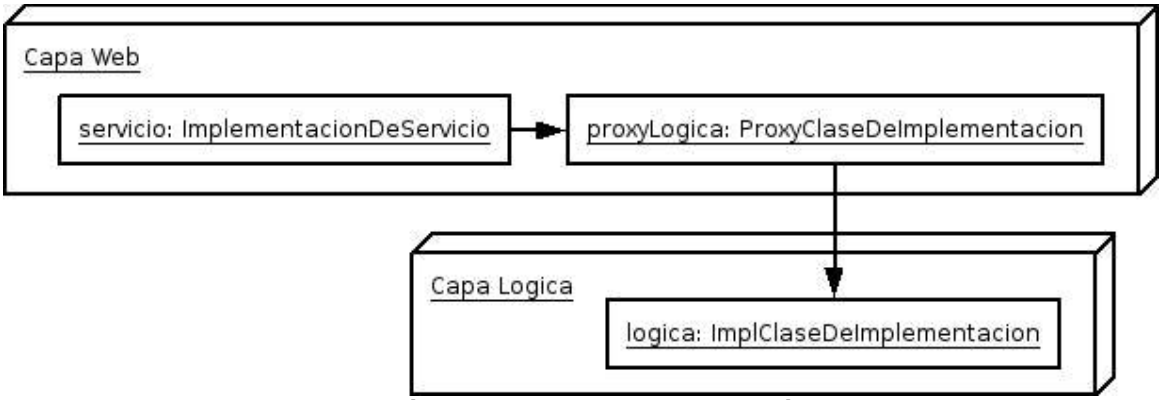

Ilustración 11 - Conector (Capas físicas).

ClaseDeImplementación

Esta es una interfaz que sirve para mejorar la transparencia de implementación. Las instancias de ImplClaseDeImplementación y ProxyClaseDeImplementación implementan esta interfaz [Mon03].

ImplClaseDeImplementación

Esta clase contiene las funcionalidades reales de cualquier implementación del servicio [Mon03].

ProxyClaseDeImplementación

Este componente facilita la comunicación entre la instancia de ImplementaciónDeServicio y la instancia de ImplClaseDeImplementación que reside en un proceso separado. La instancia de ImplementaciónDeServicio no debe darse por enterada de que se está comunicando con un servicio que no reside en la misma capa física [Mon03].

ImplementaciónDeServicio

Este componente se convierte en un simple cascarón de lo que ha sido en los patrones anteriores. Toda la lógica fuerte del negocio ha pasado ahora a la clase ImplClaseDeImplementación que reside en la capa de lógica. Esta situación implica que ImplementaciónDeServicio queda como un mecanismo para reenviar las peticiones del servicio web a la instancia de ImplClaseDeImplementación, a través de los mecanismos provistos por la instancia de ProxyClaseDeImplementación [Mon03].

Consecuencias

Este patrón es crítico para permitir la comunicación entre un servicio web y una aplicación autocontenida (stand-alone) o una capa de lógica separada. La comunicación entre las capas físicas se logra a través del patrón Conector.

1.2. El Proceso Unificado y Patrones de Diseño

El Proceso Unificado “es un proceso de desarrollo de software configurable que puede adaptarse a proyectos de tamaño y complejidad variada. Se basa en muchos años de experiencia de la compañía Rational Software en el uso de la tecnología orientada a objetos en el desarrollo de software de misión crítica en gran variedad de industrias” [MR98]. El Proceso Unificado es un proceso porque "define quién está haciendo qué, cuándo lo hace y cómo alcanzar cierto objetivo, en este caso el desarrollo de software" [Jac98].

Las características primordiales del Proceso Unificado son:

- Iterativo e incremental.

- Centrado en la arquitectura.
- Guiado por casos de uso.
- Con confrontación de riesgos.

Los conceptos clave del Proceso Unificado son [Boo98]:

- Fase e iteración: ¿Cuándo se hace?
- Flujo de trabajo de procesos (actividades y pasos): ¿Qué se está haciendo?
- Artefacto (modelo, reporte, documento): ¿Qué se produjo?
- Trabajador (por ejemplo, un arquitecto): ¿Quién lo hace?

1.2.1. El ciclo de vida del software en el Proceso Unificado

Las fases del ciclo de vida del software son: iniciación, elaboración, construcción y transición. El principal hito de la fase de iniciación o concepción es definir el alcance del proyecto e identificar gran parte de los casos de uso. Las principales actividades de la fase de elaboración son proyectar un plan, definir las características del sistema y cimentar su arquitectura. El principal hito de la fase de construcción es crear el producto y el de la fase de transición es transferir el producto a sus usuarios [Boo98].

Cada una de las fases del proceso contiene actividades pertenecientes a los siguientes flujos de trabajo:

- Modelado de la Organización.
- Captura de Requerimientos.
- Análisis y Diseño del sistema.
- Implementación y pruebas del sistema.
- Gestión de configuración y cambios.
- Configuración del entorno de desarrollo.

A continuación se muestra una gráfica que resume lo que plantea el proceso unificado en cuanto a las fases y flujos de trabajo de un proyecto de desarrollo de software:

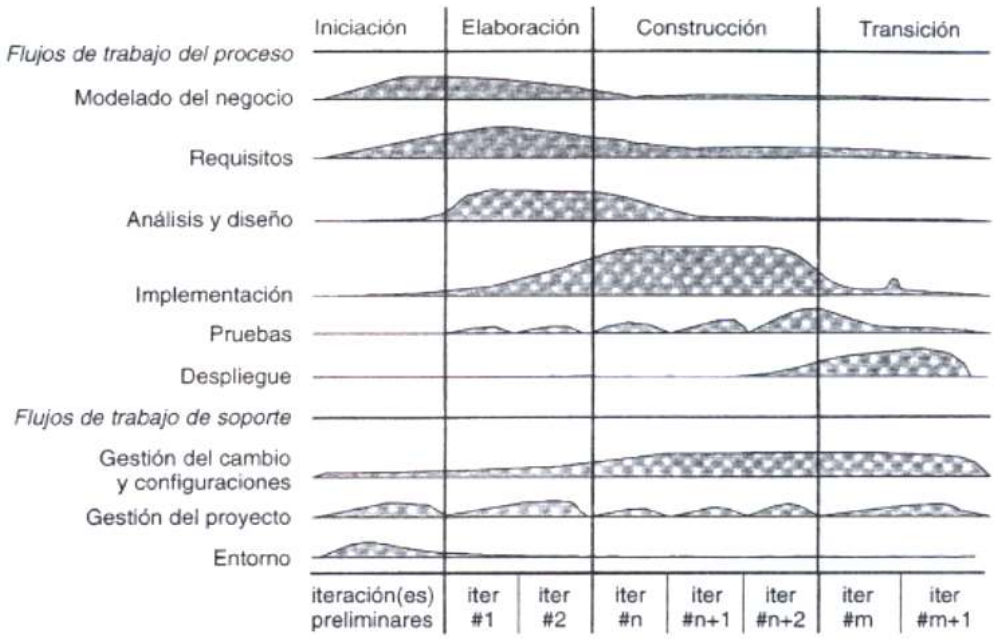

Ilustración 12 - Fases y flujos de trabajo en el Proceso Unificado.

Durante cada una de las fases, se da cumplimiento a cierto porcentaje del total de actividades dispuestas para cada uno de los anteriores flujos de trabajo. El cumplimento de estos porcentajes representa el logro de los hitos necesarios para pasar de una fase a la siguiente. Cada fase puede requerir varias iteraciones para alcanzar los objetivos propuestos para la misma.

Es importante tener en cuenta que una de las herramientas más recomendadas actualmente para pasar del análisis al diseño dentro de este proceso son los Patrones de Diseño, expuestos anteriormente en en este documento.

1.3. Tecnologías y modelos de desarrollo de Software

Software Libre

Uno de los paradigmas tecnológicos de la actualidad es el Software Libre. Grandes

empresas multinacionales que influyen considerablemente en los rumbos de la informática, la electrónica y las telecomunicaciones han puesto sus ojos en este modelo de desarrollo y han conocido y aprovechado sus ventajas. Es importante saber exactamente a qué se hace referencia cuando se utiliza este término, y por ello es importante repetir, que el Software Libre no es cuestión de precio, es cuestión de Libertad[Ase04-1]. Cualquier pieza de software que sea comercializada, incluye una licencia, que es un documento hecho por el autor del Software en el que se indica cómo es que éste (el autor) desea protegerlo. Los programas de Software Libre también tienen licencia, un programa es Software Libre si y sólo si en su licencia se incluyen las siguientes cuatro libertades:

- Libertad para usar el programa.
- Libertad para estudiar el programa.
- Libertad para copiar el programa cuantas veces se desee.
- Libertad para modificar y redistribuir el programa bajos los mismos términos de libertad.

Contextualizando este movimiento dentro de un entorno académico, se pueden mostrar algunas consecuencias favorables que se obtienen al utilizar Software Libre:

- El software libre le permite a un investigador, o un académico, tener a su disposición un gran LABORATORIO con tecnología de punta. No sólo en las ramas de investigación ligadas con la informática, sino también con la genética, la biología, y muchas otras ciencias.
- El Software Libre es una experiencia de aprendizaje continuo y sin barreras, que permite y lleva a que se aumente constantemente la base de conocimiento de las personas relacionadas con él.
- El software libre presenta retos tecnológicos bastante interesantes.
- El software libre es una gran oportunidad de negocio y como tal una fuente de ingresos de los profesionales de la informática. Las personas de la academia pueden usarlo para generar desarrollo económico socialmente consiente.

Software Libre y Servicios Web

Las principales herramientas de software libre disponibles actualmente para desarrollar aplicaciones basadas en servicios web se resumen a continuación:

PEAR::SOAP

- Lenguaje: PHP.
- URL: http://pear.php.net/package/SOAP
- Primera versión disponible en 2002-07-08.
- Estadísticas: 33,000 descargas durante el primer semestre del 2.005
- Licencia: Licencia de PHP.

NuSOAP

- Lenguaje: PHP.
- URL: http://dietrich.ganx4.com/nusoap
- Estadísticas: cerca de 9.000 descargas durante el primer semestre del 2.005.
- Primera versión disponible en 2002-04-29.
- Licencia: LGPL.

PHP-SOAP

- Lenguaje: PHP.
- URL: http://phpsoaptoolkit.sourceforge.net/phpsoap/
- Estadísticas: menos de 2.000 descargas durante el primer semestre del 2.005.
- Primera versión disponible en 2002-04-27.
- Licencia: GPL.

SOAPpy

- Lenguaje: Python.

- URL:http://pywebsvcs.sourceforge.net/
- Estadísticas: cerca de 4.000 descargas durante el primer semestre del 2.005.
- Primera versión disponible en 2003-12-18.
- Licencia: Licencia de Python (CNRI Python License).

ZSI, Zolera SOAP Infrastructure

- Lenguaje: Python.
- URL:http://pywebsvcs.sourceforge.net/
- Estadísticas: cerca de 2,100 descargas durante el primer semestre del 2.005.
- Primera versión disponible en 2002-03-04.
- Licencia: Licencia de Python (CNRI Python License).

Soap4r

- Lenguaje: Ruby.
- URL: http://dev.ctor.org/soap4r
- Primera versión disponible en 2000-06-17.
- Licencia: Licencia de Ruby.

Tcl SOAP

- Lenguaje: TCL.
- URL: http://tclsoap.sourceforge.net
- Estadísticas: cerca de 300 descargas durante el primer semestre del 2.005.
- Primera versión disponible en 2001-05-03.
- Licencia: Licencia del MIT.

SOAP::Lite

- Lenguaje: Perl.

- URL: http://soaplite.com/
- Estadísticas: cerca de 4,200 descargas durante el primer semestre del 2.005.
- Primera versión disponible en 2003-07-04.
- Licencia: Licencia Artística de Perl.

gSOAP

- Lenguaje: C/C++.
- URL: http://www.cs.fsu.edu/~engelen/soap.html
- Estadísticas: cerca de 16,000 descargas durante el primer semestre del 2.005.
- Primera versión disponible en 2002-12-13.
- Licencia: GPL y Licencia Pública de Mozilla(MPL).

De todas las anteriores opciones, se decidió escoger el lenguaje Perl, ya que desde el inicio del proyecto uno de los intereses personales de los desarrolladores era aprender a utilizar este popular lenguaje libre.

Perl

Perl, el Lenguaje Práctico para Extracción y generación de Reportes o “Practical Extraction and Report Language”, es un "lenguaje de scripting" que corre en la gran mayoría de los sistemas operativos conocidos: Linux, Windows, OS/2, Macintosh, UNIX, AIX, CYGWIN, entre otros. Para ser más exactos, Perl corre en más de 120 sistemas operativos[Ase04-3].

La principal plataforma de desarrollo de este lenguaje son los "sistemas UNIX-like", y obviamente en la actualidad, Linux. Pero debido al enorme potencial y versatilidad, y a la gran cantidad de aplicaciones escritas en este lenguaje, es inclusive, usado intensivamente en la plataforma Windows y Microsoft Corporation lo incluye dentro de los lenguajes de scripting usados como plataforma de desarrollo Web.

Perl es un lenguaje de alto nivel, hecho por Larry Wall, quién trató de heredar las mejores características de los lenguajes de su época, entre los que se encuentran

C, sed, awk, los "shell" de Unix shell, y muchos otros más. Perl fue construido para ser práctico, fácil de usar, eficiente y completo, sacrificando por ello algo de elegancia. Perl es fácil de usar, soporta programación estructurada y orientada a objetos, tiene un potente soporte incluido dentro del lenguaje para manejo de texto con expresiones regulares, y tiene la colección más grande del mundo de módulos producidos por desarrolladores externos al lenguaje.

La filosofía de Perl se basa en la frase: "Hay más de una forma de hacerlo". Puede ser utilizado para procesar texto y archivos, construir prototipos rápidos, utilidades de sistema, herramientas y tareas para gestión de sistemas, acceso a bases de datos, programación gráfica, programación en red y aplicaciones Web. Por ello, Perl es muy popular entre los desarrolladores Web y los administradores de sistemas, pero también es muy usado por algunos científicos.

No se puede hablar de Perl sin hablar de CPAN. CPAN mantiene miles de módulos de Perl desarrollados por muchas personas en el mundo. Como ya se había dicho antes, este lenguaje tiene la mayor cantidad de módulos "externos" o hechos por terceras personas. Prácticamente se puede decir que hay un módulo de Perl para cada tarea que se quiera realizar.

Perl es uno de los lenguajes libres por excelencia, y por sus características técnicas se convierte en una muy buena alternativa de tecnología base para desarrollar un sistema de información basado en servicios Web.

Apache

El servidor HTTP de Apache es actualmente el principal proyecto de la "Apache Software Foundation" o ASF (Fundación de Software Apache) http://www.apache.org/ , organización que soporta, hoy en día, varias decenas de proyectos LIBRES relacionados en general con servicios basados en Internet. Generalmente los proyectos de Apache son líderes en sus sector; claros ejemplos de esto son, aparte de su servidor HTTP, Jakarta Tomcat, ANT y Apache Axis entre otros[Ase04-2].

La ASF fue en fundada en 1999 debido al enorme crecimiento del servidor HTTP Apache (que de ahora en adelante será llamado Apache) para proveer infraestructura hardware, de comunicaciones y de negocios a los múltiples

proyectos soportados o liderados por esta. Aparte de esto nació también por la necesidad de mantener una figura legal bien constituida para proteger al Software y a los miembros de la comunidad que lo desarrollan.

Apache puede ser presentado como el mejor servidor HTTP o servidor Web del mundo. Nació en 1995 en la Universidad de Illinois, basado en el servidor httpd de la NCSA que también era Software Libre. Después de seguir un proceso completo de rediseño y codificación, llegó a su versión 1.0 en 1995. Desde mayo de 1996 mantiene el primer lugar en el mercado de los Servidores Web.

Apache es un servidor multihilo, multiplataforma (UNIX-like y Win32), y basado en módulos multi-proceso.

Entre las múltiples funcionalidades de Apache están:

- Grandes rangos de personalización.
- Hosts virtuales
- Negociación de contenido
- Autenticación, Autorización y Control de Acceso
- Archivos .htaccess para manejo de configuración por directorio
- Manejo de directorios por usuario
- Ajuste dinámico del número de procesos servidores
- Manejo de "server-side includes"

Aparte de sus grandes funcionalidades se ha demostrado que es también líder en desempeño y en seguridad en varios sistemas operativos.

En este momento aproximadamente el 70% de los sitios en Internet sirven sus páginas Web a través del servidor Apache, y como si no fuera suficiente, este porcentaje crece mensualmente. Para ver un análisis de los diferentes servidores Web utilizados en Internet se puede visitar el siguiente enlace: http://news.netcraft.com/archives/web_server_survey.html

mod_perl

mod_perl es un módulo de apache construido para integrar dos grandes proyectos líderes en el mundo de Internet: Apache y Perl. Esta tecnología aumenta las capacidades del servidor Apache cuando sirve aplicaciones basadas en Perl. Las principales funciones de este módulo son[Apa04]:

- Se aumenta casi cien veces la velocidad de ejecución de “scripts” de Perl y se reduce considerablemente la carga del servidor.
- Es posible crear más fácilmente módulos que se integren con el servidor Apache.
- Se puede tener acceso y control de todas las etapas de petición-respuesta para que éstas sean manejadas por Perl.
- Se puede utilizar Perl en los archivos de configuración del servidor Web.
- mod_perl soporta algunas plataformas de desarrollo bastante potentes.
- mod_perl está disponible tanto para la versión 1.3.* como para la versión 2.0.* de Apache.

Servicios Web con Perl

El lenguaje Perl cuenta principalmente con dos módulos que pueden ser usados para crear servicios web de manera fácil y rápida, desde cero o reutilizando aplicaciones de Perl ya existentes. Éstos módulos son SOAP::Lite y WSDL::Generator. A continuación se explicarán cada uno de ellos y se mostrará un ejemplo de su uso [Rio04].

SOAP::Lite

SOAP::Lite es un conjunto de módulos de Perl que proveen una interfaz simple y liviana para el protocolo SOAP, tanto en el lado cliente como en el del servidor. SOAP::Lite es actualmente el kit de desarrollo de web services para Perl más difundido y utilizado. Su página en SourceForge ha registrado más de 10,700 descargas en casi 2 años.

SOAP::Lite provee clases para implementar funcionalidades de un cliente SOAP,

varios servidores, soporte a datos y muchas otras tareas. La siguiente es una lista resumida de sus características:

Soporte de Protocolos

- Soporta las especificaciones SOAP 1.1 y SOAP 1.2.
- Incluye XMLRPC::Lite, una implementación del protocolo XML-RPC en el lado del cliente y del servidor. Entre los protocolos de trasporte disponibles, están HTTP, SMTP, POP3 y TCP.
- Soporta publicación y peticiones UDDI del lado del cliente, a través de un API.

Interoperabilidad

Se han realizado pruebas de interoperabilidad con diferentes implementaciones: Apache SOAP, Frontier, Microsoft SOAP, Microsoft .NET, DevelopMentor, XMethods, 4s4c, Phalanx, Kafka, SQLData, Lucin (en Java, Perl, C++, Python, VB, COM, XSLT).

Protocolos de Transporte

- Provee implementaciones de servidores TCP con multiservidor “no-bloqueante”
- Soporta transporte sobre Jabber, MQSeries y SMTP.
- Provee compresión transparente para HTTP.
- Soporta el protocolo HTTPS.
- Provee soporte para proxy.
- Provee implementaciones de servidor POP3.
- Soporta M-POST y redirección HTTP.

Soporte para WSDL

Soporta el esquema WSDL con “stub” y acceso en tiempo de ejecución. Soporta descripciones de servicio por directivas y cortas (tModel).

Otras

- Provee implementaciones de servidores CGI, daemon, mod_perl, Apache::Registry yFastCGI.

- Incluye los módulos de Apache mod_soap y mod_xmlrpc, los cuales permiten crear servidores SOAP o XML-RPC con algunas líneas en los archivos .htaccess o httpd.conf.
- Soporta el enlace dinámico y estático de clases y métodos.
- Provee un intérprete de comandos para sesiones SOAP interactivas.
- Incluye una gran cantidad de ejemplos en su documentación.

WSDL::Generator

WSDL::Generator es un módulo de Perl para crear archivos de descripción de servicios (WSDL) automáticamente a partir de módulos de perl expuestos como servicios web. Éste módulo, desarrollado por Pierre Denis <pdenis@fotango.com>, es tal vez el único que se ha creado hasta ahora con ésta funcionalidad en el mundo de Perl.

SOAP::Lite y WSDL::Generator en la Práctica

A continuación, se muestra un ejemplo de la utilización del lenguaje Perl para exponer un servicio web sencillo. En la documentación de los módulos puede encontrarse una descripción extendida de todas las funcionalidades que poseen y cómo utilizarlas.

El siguiente es el código fuente de un módulo de Perl que implementa una clase llamada “Cafetera”, con un único método llamado “prepararCafe”, que recibe como parámetro el número de tasas y devuelve como resultado un mensaje indicando el número de tasas preparadas:

Código del archivo Cafetera.pm

```
#!/usr/bin/perl -w

# Servicio web de ejemplo

package Cafetera;

use strict;

# Este es el constructor
```

```
sub new

{

        my $proto = shift;

        my $class = ref($proto) || $proto;

        my $tasas = 0;

        bless($tasas,$class);

}

sub prepararCafe{

        shift;

        my $tasas = shift;

        print STDERR "Llamado con el parámetro $tasas";

        return "Se prepararon ".$tasas." tasas";

}

1;
```

A continuación se muestra el uso del módulo SOAP::Lite para exponer el módulo Cafetera como un servicio web, usando el servidor tipo SOAP::Transport::HTTP::Daemon:

Código del archivo servidor.pl

```
#!/usr/bin/perl -w

# Servidor SOAP

use SOAP::Transport::HTTP;

use Cafetera;

#En el parámetro dispatch_to se especifica la ruta a los módulos disponibles
```

```
my $daemonio = SOAP::Transport::HTTP::Daemon

        -> new(LocalAddr => 'localhost', LocalPort => 8070)

        -> dispatch_to('/home/tesis/Thëwala/preparacion/P.5-IP/productos/servicio-
en-perl','Cafetera')

        ;

print "La url del daemonio es:".$daemonio->url()."\n";

$daemonio->handle();
```

Desde éste momento, cualquier aplicación puede empezar a consumir el servicio web, si sabe cómo invocarlo. A continuación se muestra un cliente del servicio hecho en Perl con SOAP::Lite:

Código del archivo cliente.pl

```
#!/usr/bin/perl -w

use strict;

use SOAP::Lite;

my $soap = SOAP::Lite

        ->uri('http://localhost:8070/Cafetera')

        ->proxy('http://localhost:8070/')

        ;

my $resultado= $soap->prepararCafe(5)->result;

print "Preparando cafe: ".$resultado."\n";
```

Para que aplicaciones hechas en otros lenguajes o que no saben cómo invocar el servicio puedan acceder al mismo, se debe crear el archivo de descripción WSDL. A continuación se muestra el guión o “script” de perl que utiliza WSDL::Generator para crear el archivo WSDL:

Código del archivo generar-wsdl.pl

```
#!/usr/bin/perl -w

use WSDL::Generator;

my $init = {

        'schema_namesp' => 'http://localhost:8070/Cafetera.xsd',

        'services'      => 'Cafetera',

        'service_name'  => 'Cafetera',

        'target_namesp' => 'http://localhost:8070/Cafetera.wsdl',

        'documentation' => 'Servicio Web de Prueba',

        'location'      => 'http://localhost:8070/Cafetera'

};

my $w = WSDL::Generator->new($init);

Cafetera->prepararCafe(5);

print $w->get(Cafetera);
```

La salida de este guión de Perl constituye el documento wsdl que debe ser publicado para que cualquiera pueda acceder al servicio Cafetera. La siguiente es la salida del guión, la cuál puede guardarse en el archivo Cafetera.wsdl:

Código del archivo Cafetera.wsdl

```
<?xml version="1.0"?>

<definitions

        name="Cafetera"

        xmlns:xsd="http://www.w3.org/2001/XMLSchema"

        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
```

```
        targetNamespace="http://localhost:8070/Cafetera.wsdl"
        xmlns:tns="http://localhost:8070/Cafetera.wsdl"
        xmlns="http://schemas.xmlsoap.org/wsdl/"
        xmlns:xsdl="http://localhost:8070/Cafetera.xsd">
        <types>
                <xsd:schema targetNamespace="http://localhost:8070/Cafetera.xsd">
                        <xsd:element name="prepararCafeRequest" type="xsd:string"
/>
                                        <xsd:element  name="prepararCafeResponse"
type="xsd:string" />
                </xsd:schema>
        </types>
        <message name="prepararCafeRequest">
                                       <part   name="prepararCafeRequestSoapMsg"
element="xsdl:prepararCafeRequest"/>
        </message>
        <message name="prepararCafeResponse">
                                     <part    name="prepararCafeResponseSoapMsg"
element="xsdl:prepararCafeResponse"/>
        </message>
        <portType name="CafeteraCafeteraPortType">
                <operation name="prepararCafe">
                        <input message="tns:prepararCafeRequest" />
                        <output message="tns:prepararCafeResponse" />
```

```
                </operation>
        </portType>
                                        <binding     name="CafeteraCafeteraBinding"
type="tns:CafeteraCafeteraPortType">
                                                <soap:binding      style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
                        <operation name="prepararCafe">
                                <soap:operation style="document" soapAction=""/>
                                <input>
                                        <soap:body use="literal"/>
                                </input>
                                <output>
                                        <soap:body use="literal"/>
                                </output>
                        </operation>
        </binding>
        <service name="Cafetera">
                <documentation>
                        Servicio Web de Prueba
                </documentation>
                                                <port   name="CafeteraCafeteraPort"
binding="tns:CafeteraCafeteraBinding">
                        <soap:address location="http://localhost:8070/Cafetera"/>
                </port>
```

```
        </service>

</definitions>
```

2. MODELO DE NEGOCIO

En este capítulo se presenta la descripción de los procesos de negocio de la comercializadora Plazatech que están relacionados con los productos y servicios de Parquesoft Popayán y van a ser soportados por Thëwala (Sistema de Información de Productos y Servicios para las empresas de Parquesoft Popayán).

2.1. Diagrama de Casos de Uso de Negocio

En la siguiente figura se muestran los actores del negocio (Público en General, Clientes, Emprendedores, Agentes Comerciales y Administradores) que inician cada uno de los casos de uso del negocio o procesos de negocio de la comercializadora Plazatech que están relacionados con los productos y servicios de Parquesoft Popayán. En la siguiente sección se explican con detalle cada uno de estos.

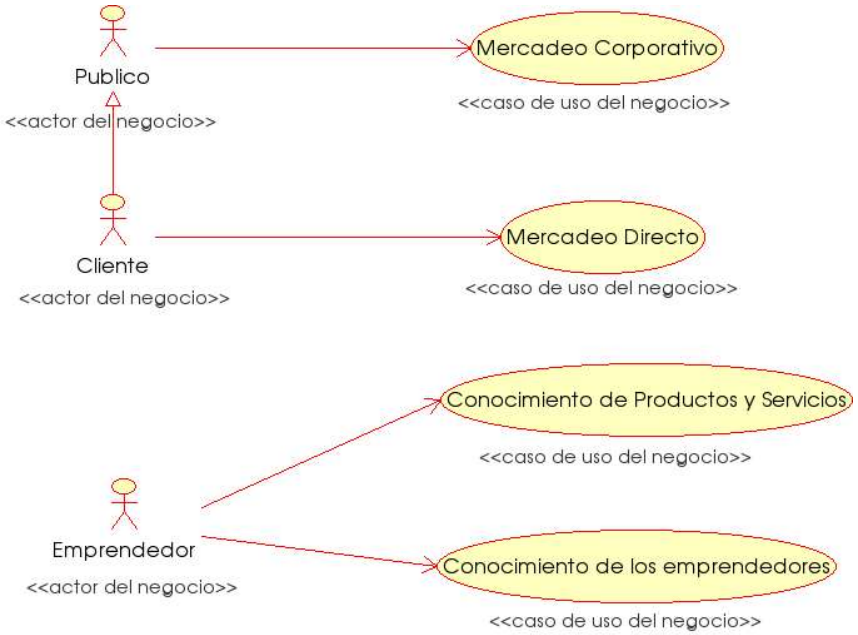

Ilustración 13 - Diagrama de Casos de Uso de Negocio

2.2. Procesos de Negocio

2.2.1. Conocimiento de los Emprendedores

Resumen:

Cuando los emprendedores llegan al Parque después de haber pasado por el comité de emprendimiento, Plazatech se reúne con ellos, les informa el modelo de comercialización del Parque y recopila su información personal, de estudios, experiencia, etc. Luego los emprendedores realizan una presentación de sus productos, orientada a los beneficios de los mismos, y algunos detalles técnicos. La siguiente tabla describe este proceso de negocio:

Proceso de Negocio	Conocimiento de los emprendedores.
Objetivo	Mantener información actualizada acerca de las capacidades de las personas que proveen los productos y servicios que ofrece el Parque.

Descripción	0. Un grupo de emprendedores pasa por el “Comité de Emprendimiento” y es aceptado dentro del Parque. ● Es identificada la información personal de los nuevos emprendedores. ● Son identificadas algunas capacidades y aptitudes de los nuevos emprendedores. ● Se identifican algunos roles iniciales ejercidos por los emprendedores dentro de su nueva organización. 1. La comercializadora realiza un análisis básico de la estructura, los roles y el posible desempeño de cada uno de los emprendedores dentro de la empresa. Después de esto, es necesario esperar un período de tiempo mientras la empresa se estabiliza y se centra en uno o en pocos nichos de mercado, o sea, mientras le empresa llega a su ciclo de estabilidad. 2. Con cada proyecto desarrollado por la empresa debe realizarse una actualización de la información referente a las capacidades y responsabilidades de los emprendedores dentro de ésta. 3. Aunque este proceso es continuo, se puede llegar a un punto de estabilidad donde se conozcan claramente los roles de los emprendedores y las capacidades de cada uno de ellos. Aparte de esto se puede tener una idea de las capacidades sinérgicas de la empresa, incluyendo las capacidades de los colaboradores.
Prioridad	Media Alta.
Riesgos	● Obtener información errada debido a expectativas equivocadas de los emprendedores respecto al futuro de su empresa. ● Manejo de información que puede cambiar abruptamente de acuerdo al empeño y la continuidad de los emprendedores dentro de sus empresas.
Posibilidades	● Mantener “Registros de desempeño” de cada uno de los emprendedores en las que se consigne información acerca del desempeño, capacidad y responsabilidad de los mismos. ● Mantener un “Historial de desempeño” de cada una de las empresas en el que se mantenga un registro de los tiempos de respuesta, magnitud, recursos invertidos y demás información sobre los proyectos que ha realizado dicha empresa.

Tiempo de Ejecución	Este es un proceso continuo y largo. Depende en gran parte del tiempo que tome la empresa para estabilizarse, tiempo que puede oscilar entre 6 meses y 2 años.

Tabla 1 - Proceso de Negocio "Conocimiento de los Emprendedores".

2.2.2. Conocimiento de los Productos y Servicios de los Emprendedores

Resumen:

La comercializadora realiza reuniones periódicas con los emprendedores con el fin de conocer el estado de desarrollo de los diferentes productos y servicios. Los emprendedores realizan presentaciones de sus productos y servicios, incluyendo demostraciones de su funcionamiento y folletos, y la información de contactos y clientes potenciales que se haya identificado. La coherencia entre las diferentes empresas de la Red de Parques es muy importante, por esta razón el contacto entre las comercializadoras de los diferentes Parques es indispensable para que la Red muestre una sola cara a cualquier cliente, para que las empresas conozcan las capacidades de todas las empresas de la red y para que no existan problemas de “canibalismo”. Se intercambian los portafolios de productos y servicios e información (contactos, oportunidades de negocio, documentación, conocimiento, entre otros) por diferentes medios, entre otros, la realización de visitas y citas de negocios. La siguiente tabla describe este proceso de negocio:

Proceso de Negocio	Conocimiento de los Productos y Servicios de los Emprendedores.
Objetivo	Conocer el estado de desarrollo de los diferentes productos y servicios que ofrecen las empresas del Parque y basados en esto, permitir que la Red de Parques pueda ofrecer un portafolio amplio de productos y servicios que satisfagan las necesidades de las organizaciones colombianas.
Descripción	1 La comercializadora realiza reuniones periódicas con los emprendedores, donde éstos presentan el estado de avance de sus productos y servicios, incluyendo demostraciones de su funcionamiento y folletos, y la información de contactos y de clientes potenciales que hayan sido identificados. 2 Las comercializadoras de cada uno de los parques intercambian los listados y la información de los productos y servicios existentes en cada parque.
Prioridad	Alta
Riesgos	● Debido a una comunicación inadecuada entre la comercializadora y los emprendedores, se puede dar origen a confusiones respecto a las capacidades, utilidades y características reales de los productos y servicios ofrecidos, generando un trastorno en los demás procesos de comercialización. ● Es posible que las empresas del Parque no puedan alcanzar los tiempos de respuesta que la comercializadora espera de ellos. ● La rápida dinámica de trabajo de las comercializadoras puede llevar a una falta de comunicación entre sí, generando división entre los Parques y cerrando posibles oportunidades de acción conjunta ó sinergia entre emprendedores. ● Es probable que debido a procesos de comunicación ineficientes o mal llevados las empresas del Parque pierdan oportunidades de negocio. ● Si la comercializadora no conoce exactamente los productos y servicios de las empresas del Parque, esto puede producir un desgaste en las personas que realizan los procesos de comercialización si éstas ofrecen productos y servicios que las empresas no están en capacidad de suministrar o en los cuales no tienen interés.

Posibilidades	Se podría tener un registro del estado de avance y capacidades reales de cada uno de los productos y servicios, el cual sea constantemente actualizado por los emprendedores y socializado a través de la comercializadora.
Tiempo de Ejecución	Este proceso se realiza de dos formas, la primera detenidamente cada cierto tiempo menor a cuatro meses; y la segunda, en forma rápida como un paso previo al proceso de negocio “Conocimiento de las necesidades específicas del Cliente, Inspiración tecnológica”.

Tabla 2 - Proceso de Negocio "Conocimiento de los Productos y Servicios de los Emprendedores".

2.2.3. Mercadeo Corporativo

Resumen:

Plazatech debe manejar la marca y, en general, todos los conceptos inscritos en el manual de imagen corporativa de ParqueSoft, ya que todo debe reflejar una unidad de conjunto o conglomerado empresarial; y a la par, debe promover la asistencia a ferias y eventos, la realización de fichas técnicas (donde se consigna la descripción de cada producto), plantillas corporativas de presentaciones, artículos de prensa, ruedas de negocio, etc. La siguiente tabla describe este proceso de negocio:

Proceso de Negocio	Mercadeo Corporativo
Objetivo	Realizar las actividades de mercadeo basándose en la imagen corporativa de Parquesoft, mientras que al mismo tiempo se da a conocer el modelo de la Red de Parques.
Descripción	1 Teniendo en cuenta el manifiesto de imagen corporativa de Parquesoft, la comercializadora realiza la divulgación de los portafolios de productos y servicios de las empresas a través de diversos medios, como el sitio web, folletos de los productos y servicios, reuniones sociales, eventos, etc.
Prioridad	Media Alta.
Riesgos	● Si no se maneja con cuidado la imagen corporativa de Parquesoft, la comercializadora puede incurrir en la violación del manifiesto. ● Si no se respeta el manifiesto, hacia el exterior de Parquesoft se puede proyectar una falta de uniformidad entre las empresas que lo conforman.
Posibilidades	● Dar a conocer el manifiesto del Parque a través de diferentes medios de comunicación como la radio, la televisión, etc. ● Tomar medidas correctivas por parte de la comercializadora con las empresas o con los emprendedores que no cumplan con el manifiesto de Parquesoft.
Tiempo de Ejecución	Se realiza en forma constante, con revisiones de los manifiestos de imagen corporativa determinadas por las directivas de la Red de Parques y la Red de comercializadoras.

Tabla 3 - Proceso de Negocio "Mercadeo Corporativo".

2.2.4. Mercadeo Directo

Resumen:

Después de que se ha identificado un cliente, se realiza un contacto formal con él por medio de una cita, para organizar presentaciones personalizadas de los productos y servicios de los emprendedores. La tabla de la siguiente página describe este proceso de negocio:

Proceso de Negocio	Mercadeo Directo.
Objetivo	Realizar contactos directos con clientes potenciales de los productos y servicios ofrecidos y, en caso de ser necesario, identificar posibles desarrollos adicionales que les provean soluciones a la medida.
Descripción	1. La comercializadora construye una lista de clientes potenciales por sector. 2. La comercializadora concierta una cita con cada uno de los clientes. 3. En la cita convenida, la comercializadora presenta el modelo del parque, los productos y servicios que pueden ser de interés para el cliente e identifica algunas otras necesidades que éste pueda tener. 4. La comercializadora organiza reuniones en el cliente y el emprendedor o la empresa del Parque. 5. La comercializadora muestra los productos o servicios de las empresas del Parque que puedan satisfacer las necesidades del cliente. ● En caso de que la organización cliente no encuentre productos o servicios que cumplan con sus requerimientos, busca entre los de las demás empresas de la red de Parques, y en caso de que en éstas empresas tampoco cuenten con portafolios que satisfagan estas necesidades, la comercializadora está en capacidad de organizar reuniones entre los clientes y las empresas del Parque indicadas para que éstas desarrollen el tipo de soluciones necesarias. ● En caso de que el producto o servicio necesitado por el cliente sea de una gran magnitud, la comercializadora, junto con la Dirección del Parque, están en capacidad de constituir equipos de trabajo conformados por diferentes empresas para dar solución a la necesidad del cliente. 6. Este proceso de mercadeo debe extenderse a nivel regional y nacional hasta consolidar un mercado para todos los productos y servicios importantes de las empresas.
Prioridad	Alta.

Riesgos	Cuando la comercializadora desarrolla su proceso de mercadeo corre el riesgo de afectar a las empresas de tres formas: ● Distraer su nicho de mercado, haciendo que no se concentren todos los esfuerzos en el desarrollo o consolidación del producto o productos (o servicio) principales sino en atender algunos negocios *relacionados* con el nicho de mercado que la empresa aborda. ● Teniendo en cuenta que muchas de las empresas son nuevas y no tienen mucha experiencia ni una capacidad de respuesta alta, es posible que la comercializadora llegue a sobrecargar a las empresas realizando más negocios de los que éstas pueden atender. ● Es posible que la comercializadora pueda llegar a comprometer a los emprendedores o a las empresas del Parque con desarrollos o soluciones que por diversos motivos no puedan ser implementados.

Tabla 4 - Proceso de Negocio "Mercadeo Directo".

3. MODELO DE CASOS DE USO

En este capítulo se presenta un resumen de la forma en que se llevó a cabo el proceso de construcción del Modelo de Casos de Uso del Sistema, principal elemento de la captura de requisitos y base esencial de los Modelos de Análisis y Diseño. Este proceso tuvo tres hitos fundamentales: la identificación de los casos de uso, la descripción extendida de los casos de uso reales y la construcción de los prototipos de interfaz que se pueden encontrar en el sitio web del proyecto (http://thewala.sourceforge.net).

En primer lugar, los casos de uso del sistema fueron identificados a partir del modelo de negocio, haciendo uso de la Ingeniería de Requisitos. Los artefactos generados antes de alcanzar este hito fueron el documento de Identificación de Actores y el documento de Identificación de Casos de Uso. En la descripción extendida de los casos de uso reales se adaptaron los casos de uso de alto nivel a tecnologías de implementación más específicas, como los sistemas de información basados en interfaz web. Basándose en la descripción de cada uno de los casos de uso reales se construyeron los prototipos de interfaz del sistema, que constituyeron un artefacto importante dentro de la captura de requisitos del sistema. *Todos los artefactos del Modelo de Casos se encuentran en el el Anexo II***.**

Es importante resaltar que construyendo los prototipos de interfaz en la fase de inicio se llevan a cabo pequeñas actividades del Flujo de Implementación y Pruebas.

3.1. Identificación de los Casos de Uso

3.1.1. Identificación y descripción de Actores

Antes de poder identificar los casos de uso, se tuvieron que identificar los Actores

del sistema. Los actores del sistema son agentes externos al mismo que lo utilizan para llevar a cabo un proceso. Los actores identificados fueron:

- **Cliente**: Este actor utiliza el sistema para poder satisfacer sus necesidades de productos y servicios basados en Software. Puede registrarse, plantear sus necesidades y obtener respuesta de un intermediario o agente comercial con soluciones construidas dentro de Parquesoft Popayán.
- **Emprendedor**: Este actor mantiene actualizada su información personal, la trayectoria de su proyecto de emprendimiento y la información de productos y servicios de su empresa. Aparte de esto, puede consultar información de las otras empresas pertenecientes a Parquesoft Popayán para poder adquirir productos y servicios de éstas que le puedan ayudar a desarrollar sus propios productos.
- **Público**: El público en general sólo utiliza el sistema para conocer el Modelo del Parque y los Portafolios de Productos y Servicios (PPS) de las empresas del Parque. De esta forma se realiza mercadeo corporativo dando a conocer la "marca Paquesoft".
- **Agente Comercial**: Puede ser un miembro del personal de la comercializadora, de la dirección del parque ó un emprendedor. Utiliza la información de productos y servicios que los emprendedores mantienen actualizada para ofrecer PPS que satisfagan por completo las necesidades del cliente.
- **Administrador**: El administrador gestiona el correcto funcionamiento del sistema, los usuarios y los diferentes privilegios y niveles de acceso de éstos.

3.1.2. Diagrama de Casos de Uso del Sistema

Los casos de uso del sistema narran la secuencia de eventos que los actores deben completar para llevar a cabo un proceso. Los casos de uso identificados se muestran en el siguiente diagrama:

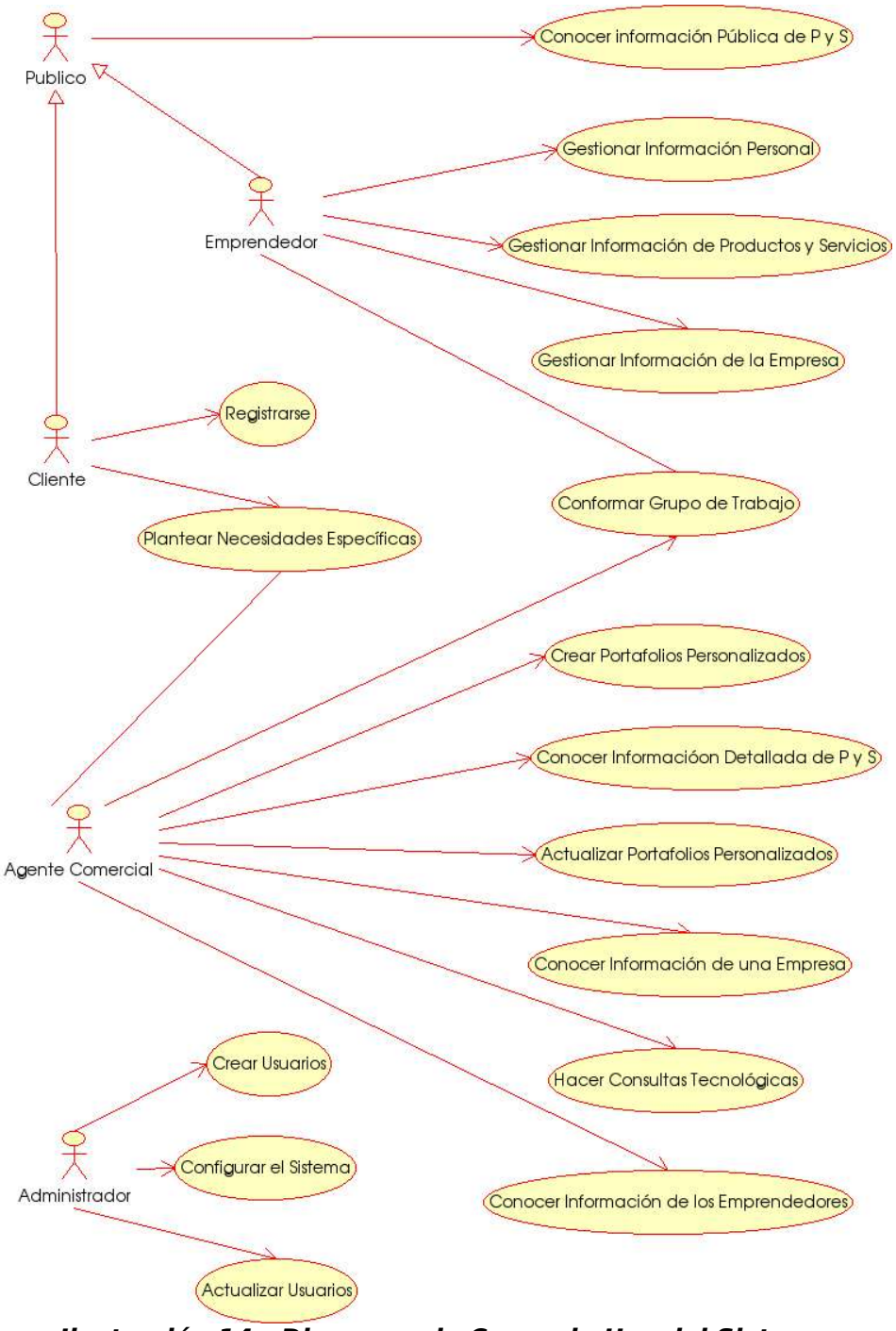

Ilustración 14 - Diagrama de Casos de Uso del Sistema

3.1.3. Descripción de alto nivel de los Casos de Uso del Sistema

A continuación se presentan las descripciones de alto nivel para todos los casos de

uso del sistema.

Nombre	Crear Usuarios
Actores	Administrador
Tipo	Primario
Resumen	En este caso de uso el administrador del sistema puede crear un nuevo usuario.
Descripción	1. El Administrador le indica al sistema que desea crear un nuevo usuario. 2. El sistema pide que se ingresen los parámetros necesarios para crear el nuevo usuario, como tipo de usuario, datos personales, nombre de usuario y clave, etc. 3. El Administrador ingresa los parámetros y le indica al sistema que complete la operación. 4. El sistema almacena la información y muestra un mensaje de éxito.

Tabla 5 Descripción de alto nivel del caso de uso "Crear Usuarios"

Nombre	Conocer Información de Emprendedores
Actores	Agente Comercial
Tipo	Primario
Resumen	En este caso de uso el agente comercial puede revisar y conocer la información de desempeño, trayectoria y disponibilidad de los emprendedores.
Descripción	1. El Agente Comercial le indica al sistema que desea buscar la información de desempeño de los emprendedores. 2. El Sistema pide que se ingresen los parámetros de búsqueda, como nombre del emprendedor, fecha de ingreso al parque, tecnologías de interés o dominio, disponibilidad de tiempo, perfil, etc. 3. El Agente Comercial ingresa los parámetros y le indica al sistema que proceda con la búsqueda. 4. El Sistema realiza la búsqueda y presenta los resultados. 5. El Agente Comercial selecciona uno de los emprendedores presentados. 6. El Sistema presenta la información detallada del emprendedor seleccionado.

Tabla 6Descripción de alto nivel del caso de uso "Conocer Información de los Emprendedores"

Nombre	Conocer Información de una Empresa
Actores	Agente Comercial
Tipo	Primario
Resumen	En este caso de uso el agente comercial puede revisar y conocer la información de desempeño, trayectoria y disponibilidad de una empresa.
Descripción	1. El Agente Comercial le indica al sistema que desea buscar la información de una empresa de Parquesoft. 2. El Sistema pide que se ingresen los parámetros de búsqueda, como nombre de la empresa, nombre de un emprendedor, fecha de ingreso al parque, tecnologías de interés o dominio, clientes, disponibilidad, etc. 3. El Agente Comercial ingresa los parámetros y le indica al sistema que proceda con la búsqueda. 4. El Sistema realiza la búsqueda y presenta los resultados. 5. El Agente Comercial selecciona una de las empresas presentadas. 6. El Sistema presenta la información detallada de la empresa seleccionada.

Tabla 7Descripción de alto nivel del caso de uso "Conocer Información de una Empresa"

Nombre	Crear Portafolios Personalizados de Productos y Servicios
Actores	Agente Comercial
Tipo	Primario
Resumen	En este caso de uso, el agente comercial puede construir un portafolio personalizado de productos y servicios integrando varios portafolios de las empresas del Parque.
Descripción	1. El Agente Comercial le indica al sistema que desea crear un Portafolio Personalizado. 2. El Sistema pide que se ingresen los parámetros de búsqueda, como nombre de la empresa, nombre del producto ó servicio, nombre del emprendedor, tecnología de la solución, sector de aplicación, palabra clave, etc. 3. El Agente Comercial ingresa los parámetros y le indica al sistema que proceda con la búsqueda. 4. El Sistema realiza la búsqueda y presenta los resultados. 5. El Agente Comercial selecciona uno de los productos ó servicios presentados. 6. El Sistema presenta la información detallada del producto ó servicio seleccionado. 7. El Agente Comercial le indica al sistema que desea agregar el producto o servicio seleccionado al Portafolio Personalizado actual. 8. El Sistema agrega el producto ó servicio al Portafolio Personalizado. 9. El Agente repite los pasos anteriores hasta que le indica al sistema que desea terminar de construir el Portafolio Personalizado. 10. El Sistema crea el Portafolio y muestra un mensaje de éxito.

Tabla 8Descripción de alto nivel del caso de uso "Crear Portafolios Personalizados de P. y S."

Nombre	Conformar Grupo de Trabajo
Actores	Agente Comercial (iniciador), Emprendedor
Tipo	Primario
Resumen	En este caso de uso el agente comercial puede conformar grupos de trabajo que puedan satisfacer las necesidades específicas planteadas por un cliente.
Descripción	1. El Agente Comercial le indica al sistema que desea conformar un grupo de trabajo. 2. El sistema le pide al Agente Comercial que ingrese los datos del equipo del equipo de trabajo (líder, número de personas, perfil de los integrantes, disponibilidad de tiempo, etc.) 3. El Agente Comercial ingresa los datos y le indica al sistema que busque los emprendedores que cumplan con los perfiles requeridos y cuenten con la disponibilidad de tiempo especificada. 4. El sistema realiza la búsqueda y presenta los resultados para cada uno de los roles del grupo de trabajo. 5. El Agente Comercial selecciona los posibles integrantes del grupo de trabajo. 6. El sistema notifica el evento a los emprendedores seleccionados para conformar el grupo de trabajo y les da la posibilidad de confirmar su participación. 7. Los emprendedores confirman su participación en el grupo de trabajo. 8. El sistema almacena la información del grupo de trabajo y envía una notificación al Agente Comercial.

Tabla 9Descripción de alto nivel del caso de uso "Conformar Grupo de Trabajo"

Nombre	Plantear necesidades
Actores	Cliente(iniciador), Agente Comercial
Tipo	Primario
Resumen	En este caso de uso el cliente puede plantear sus necesidades específicas relacionándolas con algunas de las categorías de productos y servicios ofrecidos, y dando información detallada de sus requerimientos adicionales.
Descripción	1. El Cliente le indica al sistema que desea plantear una necesidad específica. 2. El sistema le pide al Cliente que ingrese el sector de aplicación y las tecnologías con los cuales se relaciona el producto ó servicio. 3. El cliente selecciona de una lista el sector de aplicación y las tecnologías con los cuales está relacionada su necesidad. 4. El sistema le pide al cliente que ingrese la información del entorno de implantación del sistema y la descripción específica de la necesidad. 5. El cliente ingresa los datos. 6. El sistema envía una alerta al agente comercial para que revise las necesidades específicas, dándole a conocer un listado de los posibles productos y servicios, empresas y emprendedores que pueden ser tenidos en cuenta para cubrir la necesidad del cliente. 7. El agente comercial revisa las relaciones existentes entre las necesidades del cliente y los productos ofrecidos por las empresas, y luego integra portafolios de productos y servicios y conforma grupos de trabajo entre empresas con el fin de dar solución a estas necesidades. 8. El agente comercial le indica al sistema que la necesidad planteada ha sido analizada y ha sido conformado el equipo de trabajo para resolverla.

Tabla 10Descripción de alto nivel del caso de uso "Plantear Necesidades"

Nombre	Gestionar Información Personal
Actores	Emprendedor
Tipo	Primario
Resumen	El emprendedor ingresa y mantiene actualizada su información personal, incluyendo su trayectoria y roles dentro de la empresa a la que pertenece.

Descripción	1. El Emprendedor le indica la sistema que desea ingresar o actualizar datos relacionados con su información personal. 2. El sistema muestra la información personal actual del emprendedor y da la posibilidad de que ésta sea actualizada. 3. El Emprendedor actualiza o ingresa la información al sistema. 4. El sistema valida la información y le indica al Emprendedor que la actualización se ha realizado exitosamente.

Tabla 11Descripción de alto nivel del caso de uso "Gestionar Información Personal"

Nombre	Gestionar Información de Productos y Servicios
Actores	Emprendedor
Tipo	Primario
Resumen	El emprendedor ingresa y mantiene actualizada la información de sus productos y servicios.
Descripción	1. El Emprendedor indica al sistema que desea actualizar la información de sus P y S. 2. El Sistema muestra la lista de los P y S que ha definido el Emprendedor previamente, si los hay. 3. El Emprendedor puede seleccionar uno de los P y S para actualizarlo ó para borrarlo, ó puede indicarle al sistema que desea crear uno nuevo. 4. El Sistema despliega la interfaz correspondiente. 5. El Emprendedor ingresa ó modifica los parámetros de del Producto ó Servicio, como nombre, tecnologías de la solución, sector de aplicación, palabras clave, descripción, precios, costos de producción, estado de desarrollo, inventario, etc. 6. El sistema actualiza la información y muestra un mensaje de éxito en la operación.

Tabla 12Descripción de alto nivel del caso de uso "Gestionar Información de Productos y Servicios"

Nombre	Gestionar Información de la Empresa
Actores	Emprendedor
Tipo	Primario
Resumen	El emprendedor ingresa y mantiene actualizada la información de desempeño, trayectoria y disponibilidad de su empresa.
Descripción	1. El Emprendedor le indica la sistema que desea ingresar o actualizar datos relacionados con su la información de su empresa, incluyendo historial de contratación, hitos, problemas presentados y clientes importantes. 2. El sistema muestra la información actual de la empresa y da la posibilidad de que ésta sea actualizada. 3. El Emprendedor actualiza o ingresa la información al sistema. 4. El sistema valida la información y le indica al Emprendedor que la actualización se ha realizado exitosamente.

Tabla 13Descripción de alto nivel del caso de uso "Gestionar Información de la Empresa"

Nombre	Conocer Información Pública de Productos y Servicios
Actores	Público
Tipo	Primario
Resumen	En este caso de uso, el público en general puede revisar y conocer el portafolio de productos y servicios de una empresa particular.
Descripción	1. El Público le indica al sistema que desea buscar un producto ó servicio. 2. El Sistema pide que se ingresen los parámetros de búsqueda, como nombre de la empresa, nombre del producto ó servicio, tecnología de la solución, sector de aplicación, palabra clave, etc. 3. El Público ingresa los parámetros y le indica al sistema que proceda con la búsqueda. 4. El Sistema realiza la búsqueda y presenta los resultados. 5. El Público selecciona uno de los productos ó servicios presentados. 6. El Sistema presenta la información que ha sido definida como pública del producto ó servicio seleccionado.

Tabla 14Descripción de alto nivel del caso de uso "Conocer Información Pública de Productos y Servicios"

Nombre	Registrase en el Sistema
Actores	Cliente
Tipo	Primario
Resumen	El cliente debe registrarse en el sistema para poder establecer un contacto con los agentes comerciales que le permita adquirir los productos y servicios de su interés.
Descripción	1. El Cliente le indica al sistema que desea registrarse. 2. El Sistema le pide al Cliente que ingrese sus datos en de registro, incluyendo nombre, información de contacto, sector productivo al que pertenece, etc. Y le indica al sistema que proceda con el registro. 3. El sistema actualiza la información y muestra un mensaje de éxito, indicándole al Cliente sus datos de acceso al sistema y la forma como puede utilizar los servicios del mismo.

Tabla 15Descripción de alto nivel del caso de uso "Registrarse en el Sistema"

3.2. Descripción Extendida de los Casos de Uso Reales

De aquí en adelante se utiliza el caso de uso “Conocer Información Pública de Productos y Servicios” como ejemplo para cada uno de los artefactos y actividades mostrados en este resumen.

3.2.1. Descripción extendida del Caso de Uso Real “Conocer Información Pública de Productos y Servicios”

Nombre	Conocer Información Pública de Productos y Servicios
Actores	Público
Tipo	Primario
Resumen	En este caso de uso, el público en general puede revisar y conocer el portafolio de productos y servicios de una empresa particular.
Precondiciones	El usuario ha ingresado a la página principal del sistema.

Flujo de Eventos	Acciones de los actores	Acciones del sistema
	1. El Público le indica al sistema que desea buscar un producto ó servicio haciendo clic en el enlace “Productos y Servicios” de la página principal de la aplicación.	2. El Sistema muestra un formulario en el que pide que se ingresen los parámetros de búsqueda que son, nombre de la empresa, nombre del producto ó servicio, tecnología de la solución y palabras claves.
	3. El Público ingresa los parámetros, escoge el sector de la aplicación por medio de una caja de selección y le indica al sistema que proceda con la búsqueda haciendo clic en el botón “Buscar”.	4. El Sistema valida los datos, realiza la búsqueda y presenta una lista de los resultados, en la que se encuentra el nombre del producto y una pequeña descripción.
	5. El Público selecciona uno de los productos ó servicios presentados haciendo clic en el nombre.	6. El Sistema presenta la información que ha sido definida como pública del producto ó servicio seleccionado.
Flujos Alternos	Flujo Alterno 1	
	1. , 3. , 5. El Público hace clic en el botón “Cancelar búsqueda”	2. , 4. , 6. El sistema presenta la página principal de la aplicación.
Poscondiciones	El sistema muestra información pública de un producto o servicio.	

Tabla 16 - Descripción extendida del Caso de Uso Real “Conocer Información Pública de Productos y Servicios”.

3.3. Construcción de los Prototipos de Interfaz

Como se dijo anteriormente, los prototipos de interfaz se encuentran en el sitio web del proyecto (http://thewala.sourceforge.net). En el sitio se pueden ver prototipos funcionales y también está disponible su respectivo código fuente e información de “implantación” de la aplicación que los contiene.

4. MODELO DE ANÁLISIS

En este capítulo se resume el Modelo de Análisis del sistema, indicando la forma como se realizaron las actividades y se construyeron uno a uno los diferentes artefactos y las relaciones y dependencias entre ellos. *Todos los artefactos de este modelo se encuentran en el el Anexo II***.**

El proceso de análisis se dividió principalmente en la construcción de tres artefactos principales: el diagrama de interacción de los casos de uso de análisis, el diagrama de paquetes de análisis y los diagramas de clases por cada paquete. A continuación se presenta un pequeño resumen del proceso de construcción de cada uno de estos artefactos:

4.1. Diagramas de Interacción de los Casos de Uso de Análisis

Para cada uno de los casos de uso de alto nivel descritos, se trataron de identificar objetos de análisis que al intercambiar mensajes cumplieran con el objetivo del caso de uso. Es importante aclarar que dentro de este modelo solamente se definen clases de tres tipos: clases de frontera, que son las clases que tienen interacción con el usuario final; clases de control, que son las que llevan la secuencia de las acciones, y clases de entidad, que manejan la información de los objetos del modelo del negocio. Los objetos presentados en los diagramas de secuencia son candidatos a ser de uno de este tipo de clases, e incluso esta información se muestra claramente en el diagrama por medio de estereotipos. Este artefacto fue un insumo importante para la elaboración de los diagramas de clases de análisis y diseño. A continuación se muestra el diagrama para el caso de uso “Conocer Información Pública de Productos y Servicios”:

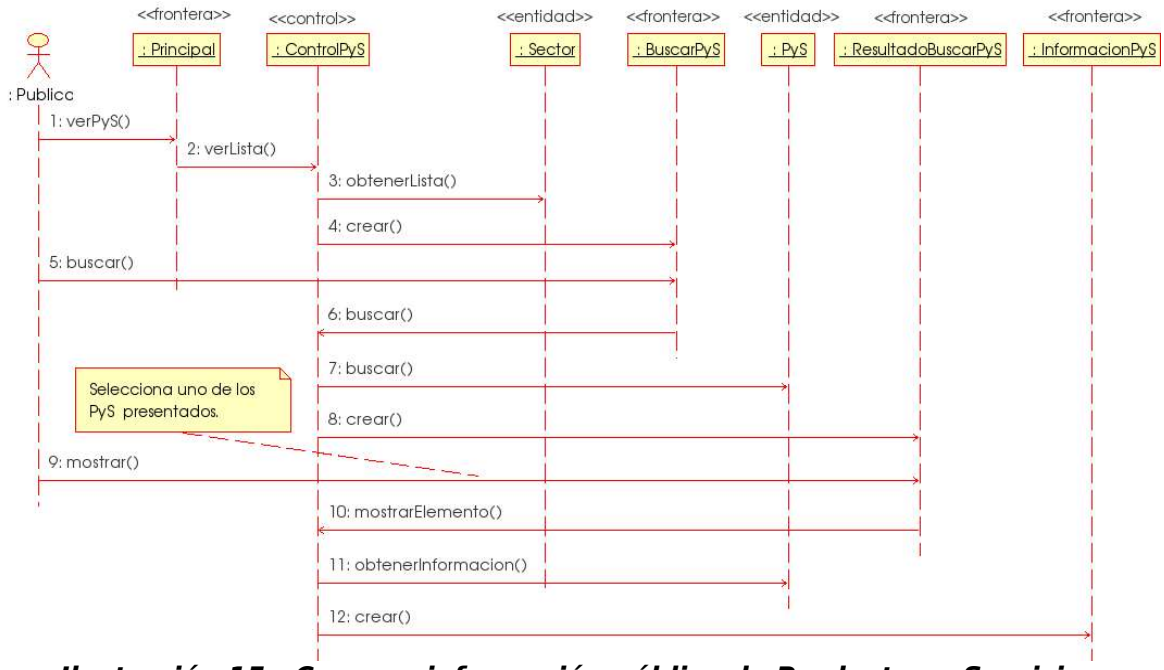

Ilustración 15 - Conocer información pública de Productos y Servicios (Diagrama de Interacción)

La Ilustración 15 muestra como se puede completar el caso de uso mencionado intercambiando mensajes entre objetos de tipo vista, control y frontera. En primer lugar (mensajes 1 al 4) se construye la interfaz de búsqueda, para lo cual es necesario obtener la lista de sectores de los productos y servicios. Cuando el usuario le indica al sistema que desea hacer la búsqueda de acuerdo a ciertos criterios escogidos, se obtiene la lista de productos y/o servicios que cumplen con dichos criterios y se construye una interfaz que la muestre (mensajes 5 al 8). Por último, cuando el usuario escoge uno de los productos de la lista, se obtiene la información pública y se muestra en una interfaz (mensajes 9 al 12).

4.2. Diagrama de Paquetes de Análisis

En este diagrama se muestran los paquetes agrupados funcionalmente, junto con sus dependencias Este diagrama fue construido partiendo de una agrupación funcional de los casos de uso y del análisis de las dependencias y las relaciones entre los objetos del análisis. La base principal para definir los paquetes fue la agrupación de los casos de uso, y la base para determinar las dependencias entre

ellos fueron las relaciones de los objetos de análisis en los diagramas de secuencia. A continuación se puede ver el diagrama de paquetes de análisis:

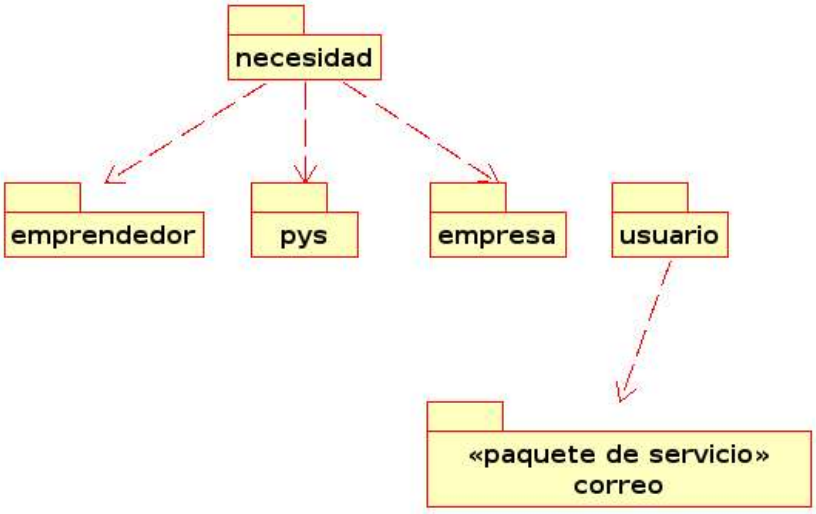

Ilustración 16 - Diagrama de paquetes de análisis

La agrupación funcional de los principales casos de uso dio como resultado la identificación de cuatro paquetes principales e independientes: emprendedor, pys, empresa y usuario. El paquete *emprendedor* maneja información personal y de desempeño de los integrantes de las empresas; el paquete *pys*, que se muestra con más detalle más adelante, agrupa las clases que manejan la información de los productos y servicios y sus agrupaciones por medio de portafolios; el paquete *empresa* agrupa las clases que maneja información de cada una de las empresas del parque, tal como su historial de desempeño, entre otros; el paquete *usuario* contiene las clases que manejan la información de los clientes, los agentes comerciales y el administrador del sistema. Aparte de estos cuatro paquetes principales, la Ilustración 16 muestra otros dos paquetes: el paquete *necesidad,* que agrupa las clases que manejan la información de las necesidades específicas planteadas por los clientes; y el paquete de servicio correo, que agrupa las clases que permiten que el sistema envíe correos electrónicos a sus usuarios.

4.3. Diagrama de Clases de Análisis

Una de las actividades del análisis fue la de construir diagramas de clases por

cada uno de los paquetes de análisis identificados. Esto se hizo por medio de la agrupación de los objetos de análisis dentro de "supertipos" o clases. A continuación se muestra el diagrama de clases de análisis del paquete pys (productos y servicios), uno de los más importantes del sistema, y en el Anexo II se pueden encontrar los diagramas de clases por cada uno de los paquetes de análisis restantes:

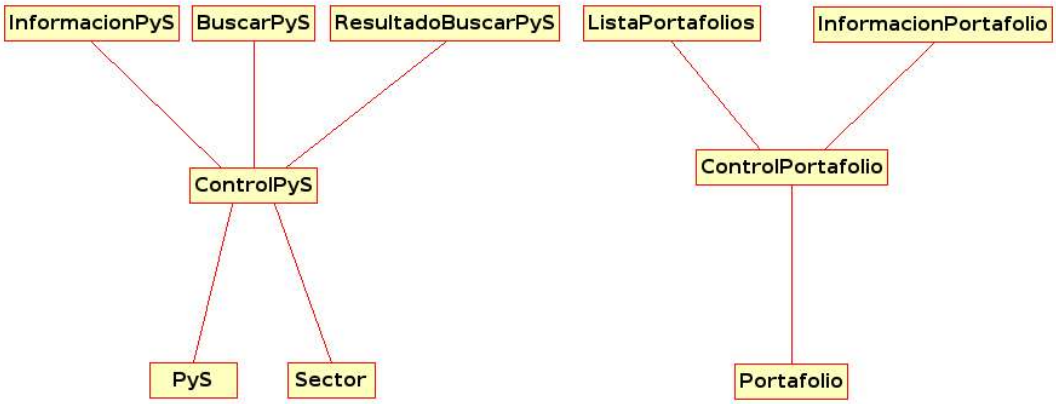

Ilustración 17 - Diagrama de clases de análisis

A continuación se presenta una pequeña descripción de cada una de las clases que aparecen en la Ilustración 17:

- InformaciónPyS: esta clase de tipo frontera se encarga de mostrar la información de un producto o servicio dado.
- BuscarPyS: esta clase de tipo frontera se encarga de mostrar la interfaz de búsqueda de productos y servicios, utilizando para ello una lista de sectores.
- ResultadoBuscarPyS: esta clase de tipo frontera se encarga de mostrar la lista de productos y/o servicios que cumplen con los criterios de búsqueda introducidos por el usuario.
- ControlPyS: esta clase se encarga de manejar la secuencia de mensajes que deben ser intercambiados para cumplir con los casos de uso relacionados con productos y servicios.
- PyS: esta clase representa un producto o servicio.
- Sector: esta clase representa un sector productivo.

- ListaPortafolios: esta clase de tipo frontera se encarga de mostrar la lista de portafolios de un agente comercial dado.
- InformaciónPortafolio: esta clase de tipo frontera se encarga de mostrar la información de un portafolio.
- ControlPortafolio: esta clase se encarga de manejar la secuencia de mensajes que deben ser intercambiados para cumplir con los casos de uso relacionados con los portafolios de productos y servicios.
- Portafolio: esta clase representa un portafolio.

5. MODELO DE DISEÑO

El Modelo de Diseño fue uno de los artefactos más importantes dentro del proceso de desarrollo del sistema. A continuación se trata de mostrar el proceso llevado a cabo para construir este modelo. La construcción se basó en cuatro actividades principales: la identificación de subsistemas, la implementación de un prototipo de aprendizaje, la identificación de los servicios web ofrecidos por el sistema y la construcción de los diagramas de paquetes y de clases de diseño del sistema. Es importante anotar que estas actividades se soportan en parte en la utilización de patrones de diseño, que han sido explicados de forma más profunda dentro del Marco Teórico del sistema. Aparte de esto, cabe destacar también que dentro el proceso de utilización de los patrones de diseño mencionados, tuvo lugar un proceso de adaptación al diseño de Thëwala, debido a que estos patrones son específicos del lenguaje Java y una de las decisiones previas de diseño del sistema fue la utilización de perl como lenguaje de programación.

5.1. Identificación de subsistemas

El diseño del sistema se basa en primera instancia, en la utilización del patrón de diseño de Capas Físicas, expuesto en el Marco Teórico del Trabajo, para construir una aproximación de la arquitectura por división en subsistemas de diseño. En un primer nivel de abstracción, el sistema se divide en dos subsistemas independientes mostrados en el siguiente diagrama:

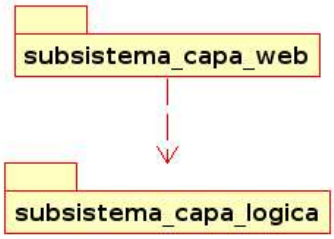

Ilustración 18 - Identificación de subsistemas

Una de las características de los sistemas basados en servicios web es su facilidad de interacción con otros sistemas. Esto se puede entender mejor analizando el diagrama, ya que los subsistemas mostrados se comunican sólo mediante mensajes SOAP. Sirviéndose de este protocolo y de la especificación de WSDL, la capa Web está en capacidad de hacer llamadas a los servicios web expuestos por la capa lógica, independientemente de las tecnologías de implementación de cada una de estas capas. Se podría pensar por lo tanto, en tener para una implementación particular del sistema, una capa Web hecha en php, con un subsistema lógico en java. No sobra decir que después de haber hecho una comparación de las posibilidades basadas en Software Libre para implementar el sistema, se decidió utilizar perl como lenguaje de programación. A continuación se se explica el funcionamiento de cada uno los subsistemas.

5.1.1. Subsistema Capa Web

El subsistema capa web es la capa superior de la aplicación. Teniendo en cuenta que una de las formas recomendadas para que una organización adopte Servicios Web como base de la comunicación de sus sistemas de información, es la utilización de esta tecnología en las implementaciones de los sistemas, es fundamental para el diseño del sistema que esta capa se comporte como un cliente que "consume" los Servicios Web del subsistema capa lógica de la misma forma en la cual los "consumirían" los sistemas de información de las empresas relacionadas con la cadena productiva.

5.1.2. Subsistema Capa Lógica

El subsistema capa lógica expone los servicios web del sistema, de tal forma que otros sistemas, o en este caso un subsistema (capa web), los pueda utilizar para llevar a cabo un proceso, llevando cierta secuencia lógica y cumpliendo las reglas de la lógica del negocio. Es importante aclarar que este subsistema debe ser diseñado sin tener en cuenta el subsistema cliente web mencionado anteriormente, sino teniendo en cuenta la interoperabilidad y versatilidad de los servicios web que ofrece.

5.2. Construcción del Prototipo de Aprendizaje

La construcción de este prototipo de aprendizaje fue muy importante para completar el diseño del sistema. Dentro de la construcción de este prototipo se tuvieron en cuenta diversos elementos técnicos entre los cuales cabe mencionar los siguientes:

- Implementación en el lenguaje perl de una parte de la capa lógica del caso de uso “Conocer Información Pública de Productos y Servicios”, donde se obtiene el listado de los sectores para que la capa cliente pueda desplegar el formulario de búsqueda de productos y servicios.
- Uso de los módulos de perl DBI y Class::DBI para realizar la interfaz con el almacenamiento en el motor de base de datos MySQL, aunque en realidad DBI provee una interfaz independiente del motor de base de datos. Se puede encontrar más acerca de DBI, en el Anexo I.
- Utilización de los patrones para servicios web Objeto de Negocio y Colección de Objetos de Negocio para realizar la implementación del prototipo. Se implementó el servicio web 'SectorCollection' o 'Colección de Sectores'.
- Utilización de los módulos de perl SOAP::Lite y WSDL::Generator para exponer, publicar y consumir el servicio web 'SectorCollection', que permite agregar sectores a la base de datos y obtener el listado de los sectores existentes.

Después de implementar este prototipo se aclararon las dudas de diseño

relacionadas con el lenguaje de programación perl y se pudo cuantificar de mejor forma el esfuerzo necesario para implementar un prototipo funcional del sistema. El código fuente del prototipo se puede encontrar en el sitio web del proyecto.

5.3. Identificación de los Servicios Web

La identificación de los servicios web del sistema se basó principalmente en los diagramas de secuencia del análisis y en las decisiones de diseño tomadas de la experiencia de la implantación del criterio utilizado para identificar los servicios web fue el de estudiar las llamadas a métodos desde las interfaces gráficas de usuario hacia las clases de control de la capa web, revisando la relación entre éstos métodos y los provistos por las colecciones de objetos de negocio de la capa lógica. El resultado de la aplicación del criterio, y las decisiones de diseño mencionadas, dio como resultado la identificación de los siguientes servicios web en la capa lógica del sistema:

- ColeccionSector.
- ColeccionPyS.
- PlantearNecesidades.
- ColeccionEmpresa.
- ColeccionPortafolio.
- ColeccionUsuario.
- ColeccionPerfil.

5.4. Identificación de paquetes y clases de diseño

Después de haber identificado los servicios web del sistema, se construyeron los diagramas de secuencia de diseño por cada caso de uso. La construcción de estos diagramas se basó principalmente en la correcta aplicación del patrón de Capas Físicas y en decisiones de diseño basadas en la experiencia adquirida al

implementar el prototipo. Tomando como base los diagramas de secuencia de diseño, por medio de la agrupación en supertipos y funcional de los objetos participantes en dichos diagramas se pudieron identificar los paquetes de diseño de cada subsistema y se pudieron construir los diagramas de clases por cada uno de los paquetes identificados. Tomando como ejemplo el caso de uso “Conocer Información Pública de Productos y Servicios” se muestran a continuación los artefactos mencionados anteriormente:

5.4.1. Diagramas de Secuencia de los casos de uso de diseño

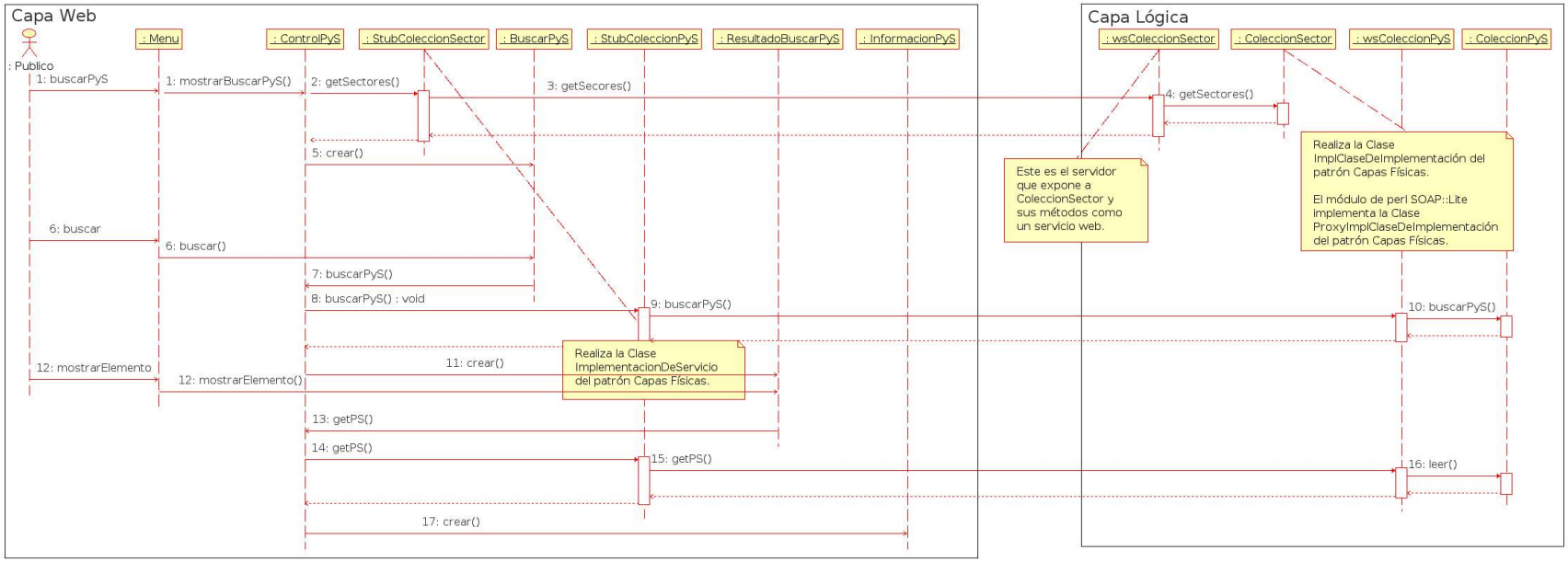

Ilustración 19 - Diagrama de Secuencia del Caso de Uso "Conocer Información Pública de Productos y Servicios".

La Ilustración 19 muestra el diagrama de secuencia de diseño para el caso de uso “Conocer Información Pública de productos y servicios”. Un elemento a tener en cuenta en este diagrama es la utilización de “stubs” (StubColeccionSector y StubColeccionPyS) y “proxies” (wsColeccionSector y wsColeccionPyS) para el intercambio de mensajes SOAP.

5.4.2. Identificación de paquetes de diseño

Para cada uno de los subsistemas se construyó un diagrama de paquetes de diseño teniendo en cuenta los requisitos funcionales y no funcionales.

Subsistema Capa Web

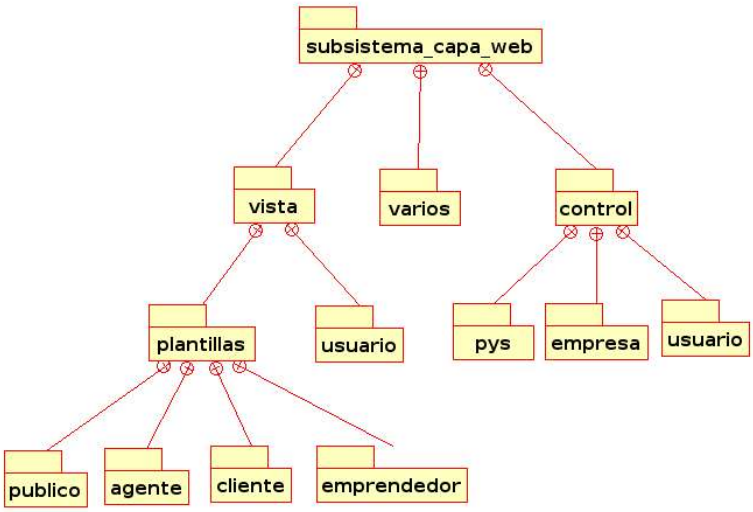

Ilustración 20 - Paquetes de diseño del Subsistema Capa Web.

El subsistema capa web está dividido en tres paquetes: el paquete *vista* que contiene todas las clases de interfaz y las plantillas y archivos necesarios para construir las páginas mostradas al usuario; el paquete *varios*, que contiene módulos de utilidad para el envío de correos electrónicos y la generación de reportes de la aplicación; y el paquete *control* que contiene todas las clases que manejan la secuencia lógica de los mensajes y los “stubs” necesarios para hacer los llamados a los métodos remotos expuestos como servicios web. El paquete *vista* se divide en dos partes: las plantillas y archivos que se encuentran en los paquetes contenidos por el paquete *plantillas* y las clases de interfaz (una por

cada tipo de usuario) que se encuentran en el paquete *usuario.*

Subsistema Capa Lógica

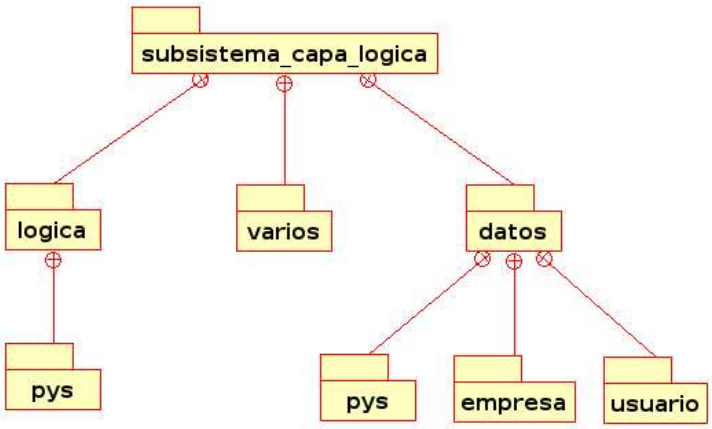

Ilustración 21 - Paquetes de dieño Subsistema Capa Lógica.

El subsistema capa lógica se divide en tres paquetes: el paquete datos, que contiene toda la jerarquía de clases que **expone colecciones de objetos de negocio** como servicios web, el paquete *logica*, que expone el único servicio web que no es una colección de objetos de negocio (PlantearNecesidades) y el paquete *varios*, que contiene módulos de utilidad para el envío de correos electrónicos y la generación de reportes de la aplicación. El paquete datos se divide a su vez en tres. Cada uno de los paquetes en los que se divide este paquete agrupa los servicios web de la capa lógica así:

pys: ColeccionSector, ColeccionPyS y ColeccionPortafolio.

empresa: ColeccionEmpresa.

usuario: ColeccionUsuario y ColeccionPerfil.

5.4.3. Identificación de clases de diseño

Para cada uno de los paquetes se han construido diagramas de clases de diseño, basados en las clases relevantes para la arquitectura. A partir de la experiencia obtenida tras realizar el prototipo de aprendizaje, se decidió utilizar los patrones

Objeto de Negocio y Colección de Objetos de Negocio para construir las clases relevantes del sistema, y se utilizaron los módulos de perl DBI, Class::DBI y SOAP::Lite como tecnologías sobre las cuales basar el diseño.

Diagramas de Clases para la Capa Lógica

A continuación se muestra el diagrama de clases para el paquete 'datos.pys' de la capa lógica:

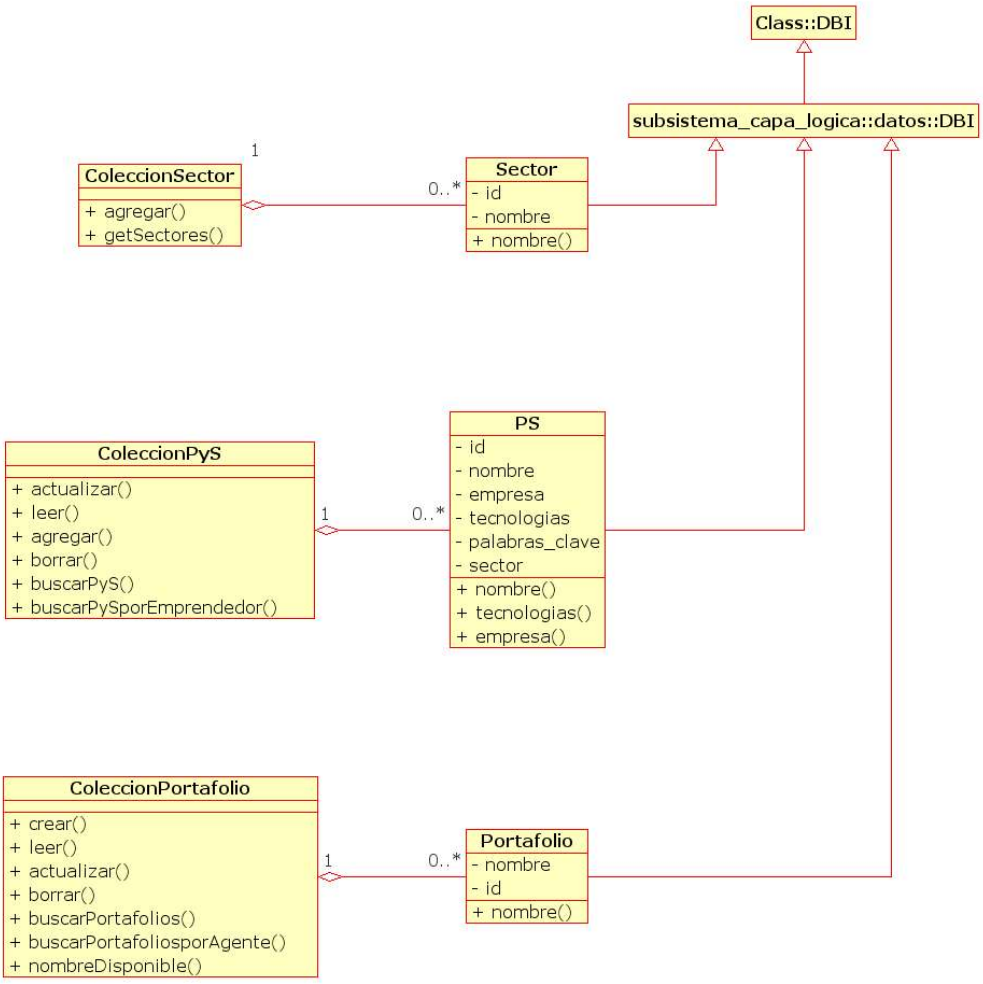

Ilustración 22 - Diagramas de Clases para la Capa Lógica.

A continuación se muestra una pequeña descripción de cada una de las clases mostradas en la Ilustración 22.

Class::DBI: Este módulo de perl, descrito anteriormente, provee la funcionalidad necesaria para manejar de forma sencilla y eficaz la conexión con la base de datos.

datos::DBI: esta clase hereda y utiliza la funcionalidad de Class::DBI para establecer efectivamente la conexión con la base de datos. Una característica my importante de esta clase es que sus clases hijas “heredan” o pueden usar la conexión a la base de datos que ésta haya establecido.

Sector: Esta clase representa un Sector y maneja automáticamente su persistencia en la base de datos.

Portafolio: Esta clase representa un portafolio de productos y servicios y maneja automáticamente su persistencia en la base de datos.

PyS: Esta clase representa un producto o servicio y maneja automáticamente su persistencia en la base de datos.

Todas las clases con nombre ColecciónObjeto agrupan objetos de tipo Objeto de acuerdo a como lo recomienda el patrón “Colección de Objetos de Negocio”. Esto es, ofreciendo los métodos agregar, leer, actualizar y borrar. Esta clase no tiene ningún mecanismo de persistencia ya que ésta se maneja a través de las clases que representan cada uno de los objetos.

ColecciónPyS: Aparte de los métodos ofrecidos por todas las colecciones, esta clase ofrece un método de búsqueda de productos y/o servicios de acuerdo a ciertos criterios y un método especializado de búsqueda por emprendedor.

ColecciónPortafolio: Aparte de los métodos ofrecidos por todas las colecciones, esta clase ofrece un método de búsqueda de portafolios de productos y servicios de acuerdo a ciertos criterios y un método especializado de búsqueda por agente comercial.

Diagramas de Clases para la Capa Web

A continuación se muestra el diagrama de clases para el paquete 'control.pys' de la capa web:

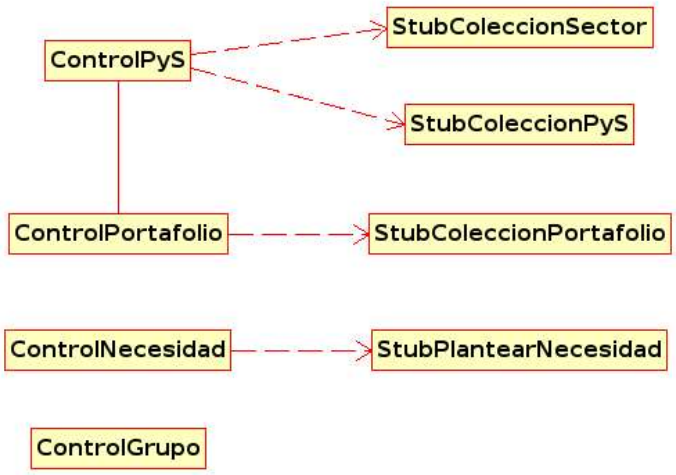

Ilustración 23 - Diagramas de Clases para la Capa Web.

A continuación se muestra el diagrama de clases para el paquete 'vista.pys' de la capa web:

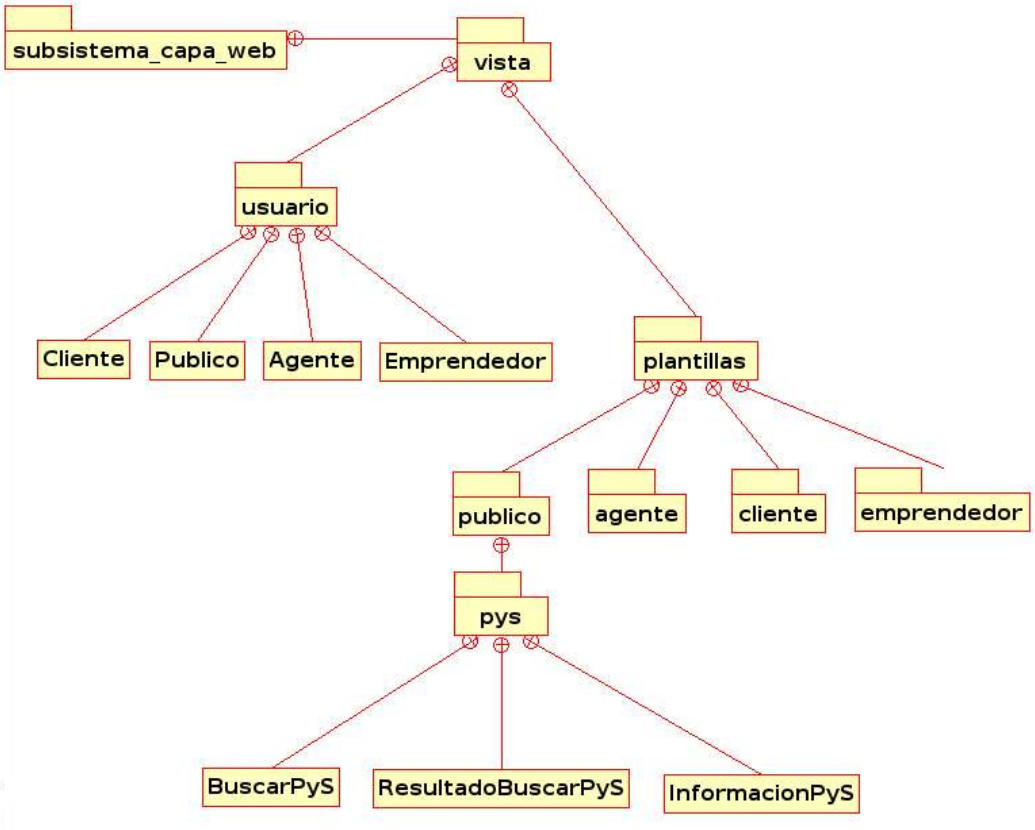

Ilustración 24 - Diagrama de clases para el paquete '*vista.pys*' de la capa web.

6. MODELO DE IMPLANTACIÓN

Este capítulo trata de mostrar la arquitectura del sistema como una colección de nodos físicos conectados entre sí, junto con la distribución de los componentes software en estos nodos. El objetivo de este modelo, es que sirva como una ayuda para estimar las características óptimas del hardware que soportaría el sistema, así como los canales de comunicación necesarios para su correcta interacción. La conexiones o relaciones entre los nodos muestran los diversos medios y protocolos de comunicación.

6.1. Diagrama de despliegue

El diagrama de despliegue muestra los nodos del sistema, junto sus conexiones y los protocolos de comunicación utilizados entre ellos. Éste se muestra a continuación:

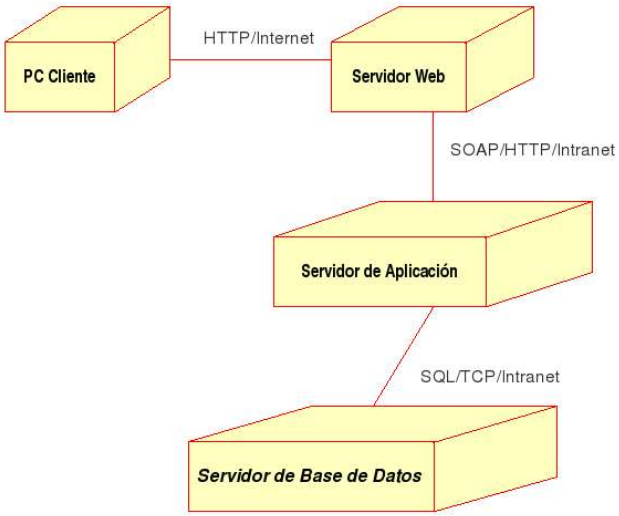

Ilustración 25 - Diagrama de despliegue

Cualquier actor del sistema puede acceder a través de un computador conectado a Internet. El computador del cliente se comunica únicamente a través del protocolo HTTP con un servidor Web que se encarga de presentar una aplicación cliente. El servidor Web procesa las diversas peticiones del cliente y se comunica con un servidor de aplicaciones en el cual se encuentra expuesta la lógica del negocio en forma de servicios Web. El servidor de aplicaciones maneja la representación de los objetos del modelo y su persistencia dentro de uno o varios gestores de bases de datos.

Este diagrama de implantación tiene completa aplicabilidad en el contexto de los espacios de nombres. Cada uno de los nodos representa un espacio de nombres diferente y en cada uno de estos nodos se ejecutan procesos completamente independientes.

6.2. Distribución de componentes

A continuación se exponen los componentes software distribuidos dentro de cada uno de los nodos del sistema. El diseño del sistema se basa en primera instancia,

en la utilización del patrón diseño de Capas Físicas, expuesto en el Marco Teórico del Trabajo, para construir una aproximación de la arquitectura por paquetes de diseño.

6.2.1. PC Cliente

El sistema es a gran escala una Aplicación Web basada en Servicios Web, por lo tanto, es posible que ningún componente del sistema resida en los computadores cliente. Simplemente es necesario contar con un navegador Web.

6.2.2. Servidor Web

El proceso o aplicación que se ejecuta dentro del Servidor Web tiene dos funciones esenciales: mostrar una interfaz de presentación del sistema y comportarse como un cliente o "consumidor" de los Servicios Web expuestos en el servidor de aplicación. Viendo el sistema de acuerdo al lenguaje manejado en el patrón de Capas Físicas, este nodo alojaría la Capa Web del sistema.

6.2.3. Servidor de Aplicación

El servidor de aplicación aloja los procesos que exponen la lógica del negocio como Servicios Web e incluye los objetos de negocio que gestionan de la mejor forma posible la interacción con el servidor de base de datos. Viendo el sistema de acuerdo al lenguaje manejado en el patrón de Capas Físicas, este nodo alojaría la Capa Lógica del sistema.

6.2.4. Servidor de Base de Datos

El servidor de base de datos aloja el gestor de bases de datos que ofrece el mecanismo de persistencia para la información de la aplicación, incluidos los objetos de negocio.

7. INTEGRACIÓN DE LOS PORTAFOLIOS AL SISTEMA

Este capítulo resume las directrices e instrucciones para que las empresas de la incubadora integren sus portafolios de productos y servicios al sistema. Este es uno de los objetivos principales del proyecto, ya que se podría mejorar ampliamente el proceso de negocio “Conocimiento de los Productos y Servicios de los Emprendedores”.

Según este proceso de negocio, identificado durante la fase de inicio del proyecto, la comercializadora de Parquesoft Popayán realiza reuniones periódicas con los emprendedores con el fin de conocer el estado de desarrollo de los diferentes productos y servicios de los mismos. Los emprendedores realizan presentaciones de sus productos y servicios, incluyendo demostraciones de su funcionamiento y folletos, y la información de contactos y clientes potenciales que ya se haya identificado.

Este proceso se realiza de dos formas: la primera, detenidamente cada cierto tiempo, menor a cuatro meses; y la segunda, en forma rápida como un paso previo al proceso de negocio “Conocimiento de las necesidades específicas del Cliente, Inspiración tecnológica”.

A partir de la implantación del sistema Thëwala, la forma como se realiza el proceso “Conocimiento de los Productos y Servicios de los Emprendedores” cambia sustancialmente, ya que se puede llevar un registro centralizado del estado de avance y otros datos relevantes de cada uno de los productos y servicios, el cual sea fácilmente actualizado por los emprendedores y socializado a través de la comercializadora y del mismo sistema.

El sistema Thëwala se ha diseñado de tal forma que tenga varios casos de uso que permitan cumplir con este requerimiento, de los cuales el más importante es el de “Gestionar Información de Productos y Servicios”. En este caso de uso, el

emprendedor ingresa y actualiza la información de los productos y servicios de su empresa de manera sencilla, una vez que ha entrado a la página principal y ha ingresado su información de acceso al sistema, provista por el administrador del mismo.

A continuación se resume el proceso descrito por este caso de uso, el cuál se muestra con detalle en el anexo II, Arquitectura Completa del Sistema, y en el anexo III, Manuales de Usuario:

1. El Emprendedor indica al sistema que desea actualizar la información de sus P y S.
2. El Sistema muestra la lista de los P y S que ha definido el Emprendedor previamente, si los hay.
3. El Emprendedor puede seleccionar uno de los P y S para actualizarlo ó para borrarlo, ó puede indicarle al sistema que desea crear uno nuevo.
4. El Sistema despliega la interfaz correspondiente.
5. El Emprendedor ingresa ó modifica los parámetros de del Producto ó Servicio, como nombre, tecnologías de la solución, sector de aplicación, palabras clave, descripción, precios, costos de producción, estado de desarrollo, inventario, etc.
6. El sistema actualiza la información y muestra un mensaje de éxito en la operación.

Gracias a que el sistema Thëwala ha sido diseñado con Servicios Web, es posible que pueda integrarse con otros sistemas de información. Esto puede resultar útil si se presentan casos en los que algunas empresas de Parquesoft Popayán cuenten previamente con su propio sistema de información de productos y servicios.

La integración de productos y servicios a partir de otros sistemas de información, haciendo uso de Servicios Web, requiere un proceso adicional de desarrollo de software, dependiente del lenguaje de programación y de la información que maneje el sistema de información a integrar.

Este proceso de desarrollo tendría como objetivo realizar una extensión del

sistema Thëwala, en su capa lógica, y del sistema de información de la empresa en cuestión. El cambio realizado en Thëwala haría que los productos y servicios de la empresa en cuestión sean buscados y manipulados a través de servicios web, desde su propio sistema. El cambio realizado en el sistema de la empresa se enfocaría en darle o aprovechar su soporte para Servicios Web, permitiendo que se pueda comunicar con Thëwala.

Dado que el sistema ha sido diseñado pensando en el rol clave de Plazatech como responsable de la comercialización de productos y servicios de todas las demás empresas de Parquesoft Popayán, es esta la indicada para asignar los recursos necesarios para una exitosa implantación del sistema, incluyendo el hardware que lo soportaría y las charlas, reuniones, capacitaciones, publicidad, etc.

8. CONCLUSIONES Y TRABAJOS FUTUROS

Este capítulo reúne un conjunto de observaciones y conclusiones del trabajo llevado a cabo, y las recomendaciones realizadas para los trabajos futuros alrededor de los temas tratados.

8.1. Conclusiones

- El Proceso Unificado puedo ser aprovechado como herramienta metodológica para organizar el desarrollo, al igual que los Patrones de Diseño de Servicios Web para garantizar la calidad del sistema construido.
- Los patrones de diseño se relacionan con los Servicios Web de tres maneras diferentes: primero, el paradigma de los Servicios Web está soportado en tres patrones de diseño; segundo, el desarrollo de implementaciones basadas en Servicios Web puede hacerse más fácil aplicando patrones de diseño existentes; y finalmente, varios de los patrones para Servicios Web son simplemente adaptaciones de patrones creados para el paradigma de la orientación a objetos, y en algunos casos pueden llegar a ser más útiles, educativos e importantes si se aplican al paradigma de los Servicios Web.
- Un punto crítico e importante en el proceso de desarrollo, es el paso del análisis al diseño y se puede resaltar la valiosa ayuda de los Patrones de Diseño como una herramienta poderosa para afrontar esta etapa, ayudando a definir y refinar la arquitectura del sistema y proporcionando pistas para la identificación de las clases de diseño.
- Se pueden aplicar con éxito tres patrones de diseño para Servicios Web utilizando el lenguaje de programación Perl: Objeto de Negocio, Colección de Objetos de Negocio y Capas Físicas.
- Perl es una buena alternativa de tecnología base para desarrollar un sistema de información basado en servicios Web. El módulo SOAP::Lite funciona en forma

excelente al consumir servicios web, y en forma aceptable al proveerlos, y el módulo WSDL::Generator funciona bien cuando se intercambian mensajes con tipos de datos sencillos, pero no respondió a las expectativas que se tenían para el manejo de tipos de datos más complejos.

- Los diagramas de interacción de los casos de uso de análisis son un importante insumo para la elaboración de los diagramas de clases del análisis y del diseño, y que los prototipos de aprendizaje son una herramienta fundamental para tomar las decisiones de diseño más relevantes de un sistema.
- Se puede afirmar que el criterio utilizado para identificar los servicios web, consistente en estudiar las llamadas a métodos desde las interfaces gráficas de usuario hacia las clases de control de la capa web, revisando la relación entre éstos métodos y los provistos por las colecciones de objetos de negocio de la capa lógica, dio como resultado un diseño aceptable del sistema.
- El software libre presenta retos tecnológicos bastante interesantes, siendo una experiencia de aprendizaje continuo y sin barreras, que permite y lleva a que se aumente constantemente la base de conocimiento de las personas relacionadas con él.

8.2. Trabajos Futuros

- En un futuro cercano se podría intentar una implantación real del sistema, en coordinación con las directivas de la Corporación Incubadora de Empresas de Software de Popayán y la empresa comercializadora, Plazatech.
- El proyecto SOAP::Lite aún está en una etapa de crecimiento, y sería bueno aportar a su desarrollo, al igual que contribuir a mejorar el módulo de perl WSDL::Generator.
- En el mediano plazo se pueden integrar aspectos de usabilidad y de mercadeo para mejorar la presentación de la información alojada en el sistema Thëwala.

- En el largo plazo se podrían plantear proyectos complementarios con el objetivo de desarrollar clientes móviles y telefónicos para el sistema, los cuales den una mayor cobertura y movilidad a los usuarios.
- En el largo plazo también se podría plantear a Fedesoft la integración de varios sistemas de información como el diseñado aquí, para construir una imagen comercial unificada del sector en Colombia.

9. GLOSARIO

- **Agente comercial:** Puede ser un miembro del personal de la comercializadora, de la dirección del parque ó un emprendedor. Utiliza la información de productos y servicios que los emprendedores mantienen actualizada para ofrecer Portafolios de P y S (PPS) que satisfagan por completo las necesidades del cliente.
- **Canibalismo:** Situación que se presenta cuando una empresa entra en conflicto con otra debido a que compiten en el mismo nicho de mercado.
- **Cliente:** Este actor utiliza el sistema para poder satisfacer sus necesidades de productos y servicios basados en Software. Puede registrarse, plantear sus necesidades y obtener respuesta de un intermediario o agente comercial con soluciones construidas dentro de Parquesoft Popayán.
- **Emprendedor:** Este actor mantiene actualizada su información personal, la trayectoria de su proyecto de emprendimiento y la información de productos y servicios de su empresa. Aparte de esto, puede consultar información de las otras empresas pertenecientes al Parque para poder adquirir productos y servicios de éstas que le puedan ayudar a desarrollar sus propios productos.
- **Imagen corporativa:** Información consignada en un manifiesto que busca unificar la imagen del Parque que se va a dar a conocer por diversos medios, como el sitio web, folletos de los productos y servicios, reuniones sociales, eventos, etc.

- **Mercadeo Corporativo:** Teniendo en cuenta el manifiesto de imagen corporativa de Parquesoft, la comercializadora realiza la divulgación de los portafolios de productos y servicios de las empresas a través de diversos medios, como el sitio web, folletos de los productos y servicios, reuniones sociales, eventos, etc.
- **Mercadeo Directo:** Proceso mediante el cuál la comercializadora presenta el modelo del parque, los productos y servicios que pueden ser de interés para el cliente e identifica las demás necesidades que éste pueda tener.
- **ParqueSoft Popayán / Parque:** La Corporación Incubadora de Empresas de Software de Popayán, Parquesoft Popayán, hace parte de la Red de Parques Tecnológicos de Software del Suroccidente Colombiano.
- **Plazatech:** Esta empresa, transversal dentro del modelo de trabajo del Parque, se presenta como la responsable de la comercialización de productos y servicios de todas las demás empresas que lo conforman.
- **Productos y Servicios (PyS):** productos software o servicios relacionados con la industría del software, desarrollados u ofrecidos por las empresas de Parquesoft Popayán.
- **Portafolio de P y S (PPS):** Es un conjunto de diversos medios (textos, fotos, videos, etc.) que agrupan la información comercial y técnica concerniente a varios Productos y Servicios.
- **Público:** El público en general sólo utiliza el sistema para conocer el Modelo del Parque y los PPS de las empresas del Parque. De esta forma se realiza mercadeo corporativo dando a conocer la “marca Parquesoft”.
- **Parquesoft / Red de Parques:** La Red de Parques Tecnológicos de Software del suroccidente colombiano es una entidad sin ánimo de lucro, creada con el fin de incubar, acompañar, asesorar e impulsar los emprendimientos relacionados con la producción de Software en un ambiente y con un modelo de emprendimiento creado por emprendedores, para desarrollar un nuevo y dinámico sector de la economía para la región Sur Occidental de Colombia.
- **Software Libre:** Modelo de desarrollo de software, donde los programas son

distribuidos con un tipo de licencia que incluye la libertad para usar el programa, libertad para estudiar el programa, libertad para copiar el programa cuantas veces se desee, y la libertad para modificar y redistribuir el programa bajos los mismos términos de libertad.

- **Solución a la medida:** Es un tipo de desarrollo que busca identificar y satisfacer completamente las necesidades específicas de un cliente, más que obtener un producto genérico.
- **Thëwala:** Sistema de Información de Productos y Servicios para las empresas de Parquesoft Popayán.

10. BIBLIOGRAFÍA

[Ale77] Alexander, Christopher. «A Pattern Language». Oxford University Press. 1977.

[Apa04] Apache Software Foundation. «What is mod_perl». http://perl.apache.org/. 2004.

[Ase04-1] Asenjo González, Diego Andrés. «¿Qué es y qué no es Software Libre?». http://gluc.unicauca.edu.co/. 2004.

[Ase04-2] Asenjo González, Diego Andrés. «Acerca de Apache». http://gluc.unicauca.edu.co/. 2004.

[Ase04-3] Asenjo González, Diego Andrés. «Acerca de Perl». http://gluc.unicauca.edu.co/. 2004 .

[BCK97] Bass, Len; Clements, Paul y Kazman, Rick. «Software Architecture In Practice». Addison-Wesley. 1997.

[Boo98] Booch, G.. «Software Architecture and the UML». http://www.rational.com/uml. 1998.

[GHJV95] Gamma, Erich; Helm, Richard; Johnson, Ralph y Vlissides, John . «Design Patterns. Elements of Reusable Object-Oriented Software.». Addison-Wesley. 1995.

[Jac98] Jacobson, I.. «Applying UML in The Unified Process». http://www.rational.com/uml. 1998.

[Kot02] Kotok, Alan. «Tell me about Web services, and make it quick». WebServices.org. 2002.

[Lar98] Larman, Craig. «UML Y Patrones». Prentice Hall. 1998.

[Mar03] Mariaca Minda,Cinxgler. «Desarrollo de aplicaciones y Servicios Web». GLUD - Grupo Linux Universidad Distrital. 2003.

[Mon03] Monday, Paul B.. «Web Services Patterns: Java Edition». Apress. 2003.

[MR98] Rational Software Corporation y Microsoft Corporation. «A White Paper on the Benefits of Integrating Microsoft Solutions Framework and The Rational Process». http://www.rational.com/uml/papers. 1998.

[Pel03] Peltz, Chris. «Applying Design Issues and Patterns in Web Services». devx.com. 2003.

[Rio04] Rios Peña, Alejandro. «Servicios Web con Perl». http://gluc.unicauca.edu.co/. 2004.