Universidad de las Ciencias Informáticas

Facultad 3

Componente de Servicios Web de la Ventanilla Única para el Comercio Exterior Cubano

Trabajo de Diploma para optar

por el título de Ingeniero en Ciencias Informáticas

Autor:

Astroberto Cárdenas Orfila

Tutor:

Ing. Yaniris Blanco Zamora

Co-Tutores:

Ing. Leonardo Antúnez Naranjo

Ing. Yorlen Guirado Más

Ciudad de La Habana, Junio del 2013
Año 54 de la Revolución

Declaración de Autoría

Declaro ser el autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Astroberto Cárdenas Orfila

_____

Firma del Autor

Ing. Yaniris Blanco Zamora

_____

Firma del Tutor

Agradecimientos

A mi mamá y mi papá por siempre estar a mi lado y apoyarme en todo momento.

A mi tutora Yaniris por ser tan preocupada y comprensible…

A mis dos co-tutores Leonardo y Yorlen por enseñarme tanto…

A mi tío Gino y mi prima Miriam por ayudarme y apoyarme en todo.

A todos los que de alguna forma hicieron posible este trabajo y mi convivencia en la universidad.

Dedicatoria

A mis padres Migdalia Orfila López y Astroberto Cárdenas Cárdenas quienes siempre con amor me han apoyado en todo, me han enseñado mucho en la vida y han colaborado en la formación de quien soy.

Resumen

El presente trabajo de diploma, tiene como objetivo implementar un componente que permita tanto la gestión de servicios web, como la comunicación de la VUCEC con el sistema GINA, basado en las especificaciones establecidas por el estándar UDDI referente a la publicación y consumo de servicios web y para la comunicación el protocolo SOAP.

El desarrollo del sistema estuvo guiado por las especificaciones que propone el modelo de desarrollo del CEIGE, obteniendo artefactos de las diferentes disciplinas como: el estudio preliminar, modelado del negocio, requisitos, análisis, diseño, implementación y pruebas. Para la implementación del mismo se emplearon varias herramientas libres y de código abierto, cuya selección fue definida por la dirección del proyecto. Como lenguajes de programación fueron utilizados PHP y Javascript. Como sistema gestor de bases de datos Oracle 11g.

El sistema posibilitará que las entidades puedan acceder, consumir un servicio web determinado, además de encontrar información técnica referente al mismo. Se establece igualmente un proceso de registro y autenticación al sistema, garantizándose así, la seguridad de acceso y el control de las peticiones realizadas.

Palabras clave: UDDI, integración, gestión, entidad, servicios web, publicación.

Tabla de contenido

Introducción

En la actualidad, la informática ha alcanzado un lugar privilegiado en la sociedad; aportando beneficios como la implementación de sistemas para la mayoría de los sectores de la economía. Uno de los avances más importantes hasta el momento es Internet, con aproximadamente 2.095.006.005 de usuarios en todo el mundo.(1) Esta cifra aumenta constantemente debido al volumen de servicios que brinda y a las facilidades que le proporciona a los usuarios, entre las que se pueden señalar los servicios web, los cuales permiten la interoperabilidad entre aplicaciones independientemente de hardware, software, sistema operativo o plataforma de desarrollo para poder llevar a cabo los procesos de negocio. Con el surgimiento y desarrollo de los servicios web estuvo presente la necesidad de conocer su disponibilidad y cómo acceder a ellos. Debido a esto, el consorcio Organización para el Avance de los Estándares de Información Estructurada (OASIS)[1], uno de los encargados de establecer protocolos como Protocolo de Acceso de Objeto Simple (SOAP)[2] y adoptar estándares, aprueba el Protocolo de Descripción Exploración e Integración Universal (UDDI)[3] posibilitando poder interactuar con los servicios web.

Los servicios web se han convertido en un importante medio para recibir y transmitir información por lo que se hace necesario garantizar la integridad de los datos que se transmiten por los mismos. Es por esto que es recomendable utilizar estándares de seguridad como Seguridad en Servicios Web (WS-Security). Este contiene especificaciones sobre cómo debe garantizarse la integridad y seguridad en mensajería de servicios web.

Cuba siempre con la primicia de tener un pueblo culto y preparado, ha emprendido la tarea de informatizar la sociedad y utilizar los avances de la informática en beneficio de la población y la economía. La Universidad de las Ciencias Informáticas (UCI) como programa de la Revolución de importancia fundamental para lograr este objetivo, tiene la tarea de formar profesionales que agilicen y sean el soporte fundamental para la informatización de las empresas y organismos de la Administración Central del Estado. Un ejemplo de este soporte lo constituye la Aduana General de la República de Cuba (AGR), que en conjunto con la UCI, están desarrollando el Sistema de Gestión Integral de Aduana (GINA). Este sistema informático permite a la AGR una mayor rapidez en las

[1] OASIS: por sus siglas en inglés Organization for the Advancement of Structured Information Standards.
[2] SOAP: por sus siglas en inglés Simple Object Access Protocol.
[3] UDDI: por sus siglas en inglés Universal Description, Discovery and Integration

operaciones de importación y exportación a través de la informatización de sus trámites y procesos que tienen un impacto positivo en el actual ritmo de crecimiento del comercio exterior que se desarrolla en Cuba. En aras de agilizar estos procesos se requiere de una mayor calidad y rapidez en los flujos relacionados con los controles que ejercen las entidades públicas del Estado, de forma tal que reduzcan la diversidad y dispersión de trámites.

Aunque el GINA representa una mejora en la gestión de los procesos internos de la AGR, todavía la persona o entidad interesada en realizar operaciones de exportación o importación con el país tiene que presentarse físicamente en cada ministerio para obtener los documentos que avalen y autoricen estas operaciones, lo que trae como consecuencia molestias al cliente y en muchos de los casos, una demora innecesaria en la llegada de dichos documentos a la aduana.

Por todo esto, fue creada la Ventanilla Única del Comercio Exterior de Cuba (VUCEC), que es un sistema encargado de la gestión de las operaciones comerciales que tienen lugar en el país. Esta permite que todas las actividades se realicen a través de un único punto de acceso que constituye la cara comercial de Cuba ante el mundo. Este sistema necesita intercambiar grandes volúmenes de información con otras aplicaciones pertenecientes a los organismos de la Administración Central del Estado y otras entidades como la AGR. Por el alto grado de importancia que tiene dicho intercambio y por la sensibilidad de la información que es gestionada, se hace necesario además, proveer seguridad en el intercambio de datos.

En la versión inicial de la VUCEC y entre los módulos que la conforman, se encuentra el módulo Buffer, que se encarga del intercambio de información entre los sistemas, de proveer de seguridad al mismo y de la priorización del envío de los documentos. A raíz de los cambios arquitectónicos y de concepción que tuvieron lugar en el desarrollo de la presente versión de la VUCEC, el Buffer, al estar desarrollado en una versión inferior del marco de trabajo dejó de ser compatible y su función pasa a ser solo la priorización y selección de documentos. Otra consecuencia del cambio en la concepción de la forma de trabajo provocó que no fuera funcional la lógica necesaria para garantizar la comunicación con los sistemas externos así como la seguridad de la misma.

Luego de haberse realizado un profundo análisis de las dificultades existentes, surge el siguiente **problema a resolver:** ¿Cómo garantizar el flujo de información entre el sistema GINA y la Ventanilla Única de Comercio Exterior de Cuba?

Se propone como **idea a defender** que: si se desarrolla el componente de servicios web de la VUCEC, entonces se garantizará el flujo de información entre el sistema GINA y la VUCEC.

Definiéndose como **objeto de estudio** el intercambio de información entre sistemas informáticos. Y el **campo de acción** comprende el intercambio de información entre el sistema GINA y la VUCEC, proponiéndose como **objetivo general** el desarrollo de un componente que permita el flujo de información entre el sistema GINA y la VUCEC. Definiéndose los siguientes **objetivos específicos**:

- Establecer el marco teórico de la investigación para identificar los estándares de comunicación.
- Realizar el análisis y diseño del componente de gestión de los servicios web.
- Realizar la implementación del componente de gestión de los servicios web.
- Validar la solución mediante pruebas que garanticen la calidad de la misma.

Métodos Teóricos

Análisis y Síntesis: Por medio de este método se realizó una investigación de los aspectos relacionados con la implementación de servicios web, permitiendo analizar la documentación existente. Además facilitó la extracción de los elementos más importantes relacionados con el campo de acción.

Histórico-Lógico: Este método permitió constatar teóricamente cómo ha evolucionado la utilización de los estándares que utilizan los servicios web desde su surgimiento hasta estos momentos.

Modelación: Se utilizó este método para ayudar a la comprensión de los procesos que se van a realizar en la solución propuesta.

Métodos Empíricos

Tormenta de Ideas: La tormenta o lluvia de ideas es una técnica de pensamiento creativo, utilizada para estimular la producción de un elevado número de ideas, por parte de un grupo, acerca de un problema y de sus soluciones o en general, sobre un tema que requiere de ideas originales.(2) Este método fue utilizado para obtener información acerca de los principales servicios que se deben brindar.

Resumen de los capítulos

El presente trabajo consta de tres capítulos, los cuales están distribuidos de la siguiente manera:

En el primer capítulo se abordan los aspectos fundamentales que permitirán la descripción teórica del componente a implementar. Se describe el estado actual de las tecnologías que pueden ser adecuadas para el desarrollo del componente. Se establece una comparación entre los estándares de comunicación que utilizan los servicios web y, finalmente, se define cuál de estos es el que va a ser utilizado para el desarrollo del componente.

En el segundo capítulo se describen los principales procesos involucrados en el objeto de estudio y los conceptos relacionados con el dominio del problema. Se realiza una descripción general de los requisitos funcionales y no funcionales del sistema a desarrollar. Así como, las descripciones del diseño elaborado para alcanzar las metas trazadas, además de la estrategia a seguir durante el proceso de implementación.

En el tercer capítulo se muestra el modelo de implementación, que pone en práctica el diseño de la solución, realizado en el capítulo anterior y se especifica el conjunto de validaciones y pruebas que evalúan la calidad del sistema.

Capítulo 1: Fundamentación Teórica

En el presente capítulo se abordan los aspectos fundamentales que permitirán la descripción teórica del componente a implementar. Se describe el estado actual de las tecnologías que pueden ser adecuadas para el desarrollo del componente. Se establece una comparación entre los estándares de comunicación que utilizan los servicios web y, finalmente, se define cuál de estos es el que va a ser utilizado para el desarrollo del componente.

1.1 Conceptos de servicio web

Existen múltiples definiciones sobre los servicios web, las que muestran su complejidad a la hora de elegir una adecuada, que englobe todo lo que son e implican. Una posible definición sería un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la web. Estas aplicaciones o tecnologías intercambian datos entre sí, con el objetivo de ofrecer servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios los solicitan llamando a estos procedimientos a través de la web.(3) Otros los definen como un sistema de software diseñado para soportar la interoperabilidad máquina-máquina a través de una red. Otros sistemas interactúan con el servicio web utilizando mensajes SOAP, los cuales se encuentran establecidos previamente.(4)

De los conceptos antes mencionados se decidió utilizar para la presente investigación el primero porque es el más acorde con el componente que se desea implementar.

1.2 Estándares utilizados por los servicios web

1.2.1 Pila de protocolos para servicios web[4]

Entre los estándares que son utilizados por los servicios web se encuentra la pila de protocolos para servicios web. Esta es una colección de protocolos y estándares para redes de computadores que son utilizados para definir, localizar, implementar y hacer que un servicio web interactúe con otro. La pila de protocolos para servicios está comprendida principalmente por cuatro áreas:

[4]Por sus siglas en inglés Web Services Protocol Stack

Servicio de Transporte: Responsable del transporte de mensajes entre las aplicaciones de red y los protocolos en los cuales se incluyen HTTP, SMTP, FTP, así como también el más reciente Protocolo de Intercambio de Bloques Extensible (BEEP)[5].

Mensajería XML: Responsable por la codificación de mensajes en un formato común en Lenguaje de Marcado Extensible (XML)[6], logrando que sean entendidos en cualquier extremo de una conexión de red. Actualmente, esta área incluye protocolos tales como XML-RPC, SOAP y REST.

Descripción del Servicio: Usado para describir la interfaz pública de un servicio web específico. El formato de interfaz web WSDL es típicamente usado para este propósito.

Descubrimiento de servicios: Centraliza servicios en un registro común tal que los servicios web de la red puedan publicar su localización y descripción. Hace que sea fácil descubrir qué servicios están disponibles en la red. Actualmente, la Interfaz de Programación de Aplicaciones (API)[7] UDDI se utiliza normalmente para el descubrimiento de servicios.

1.2.2 Lenguaje de Marcado Extensible (XML)

Es un formato simple de texto muy flexible derivado del Estándar de Lenguaje de Marcado Generalizado (SGML)[8] (ISO 8879). Originalmente diseñado para afrontar los retos de la gran edición electrónica, también está desempeñando un papel cada vez más importante en el intercambio de una amplia variedad de datos en la web y en otros lugares.(5)

1.2.3 Lenguaje de descripción de servicios web (WSDL)

Es una especificación estándar para describir servicios basados en XML de red. Proporciona a los proveedores de servicios un modo sencillo de describir el formato básico de las peticiones a sus sistemas, independientemente de la implementación del motor de ejecución subyacente.(6)

WSDL define un formato XML para describir servicios de red como un conjunto de puntos finales que operan en mensajes que contienen información orientada a documentos u orientada a procedimientos. Primero se describen las operaciones y mensajes de forma abstracta y luego se enlazan a un

[5] BEEP: por sus siglas en inglés Blocks Extensible Exchange Protocol
[6] XML: por su siglas en inglés Extensible Markup Language
[7] API: por sus siglas en inglés Application Programming Interface
[8] SGML: por su siglas en inglés Standard Generalized Markup Language

protocolo de red y formato de mensaje concreto para definir un punto final. Los puntos finales concretos relacionados, se combinan en puntos finales abstractos (servicios). WSDL es ampliable para permitir la descripción de puntos finales y sus mensajes, independientemente de los formatos de mensaje o los protocolos de red que se utilicen para comunicarse. Esto significa que se definen las interfaces de forma abstracta con el esquema XML y luego se enlazan a representaciones concretas que son adecuadas para el protocolo.

WSDL permite a los proveedores de servicios especificar las siguientes características de un servicio web:

- El nombre del servicio web y la información de direccionamiento.
- El protocolo y el estilo de codificación que se van a utilizar al acceder a las operaciones públicas del servicio web.
- La información de tipos, como las operaciones, los parámetros y los tipos de datos que componen la interfaz del servicio web.

Los documentos WSDL permiten a los desarrolladores exponer sus aplicaciones como servicios accesibles de red en Internet. Mediante UDDI, otras aplicaciones pueden encontrar documentos WSDL y enlazarlos para ejecutar transacciones o realizar otros procesos de empresa.(6)

1.2.4 Protocolo de Acceso de Objeto Simple (SOAP)

Es un protocolo ligero para el intercambio de información en un entorno descentralizado y distribuido. Un mensaje SOAP es una transmisión de información desde un emisor a un receptor. Estos mensajes pueden ser combinados para efectuar pautas de solicitud o respuesta.(7)

SOAP es independiente del transporte pero es comúnmente utilizado junto al Protocolo de Transferencia de Hipertexto (HTTP), para poder ser empleado con la infraestructura de internet existente. Este protocolo permite la unión y uso de servicios web descubiertos mediante la definición de una ruta de mensajes para el encaminamiento de estos. Otra de sus fortalezas reside en que se utiliza para consultar UDDI para la obtención de servicios web.

1.2.4.1 Partes fundamentales de un mensaje SOAP

SOAP es un protocolo basado en XML que define tres partes fundamentales en cada mensaje:

Sobre: En el sobre se especifica un marco para describir lo que está en un mensaje y cómo procesarlo. Un mensaje SOAP es un sobre que contiene cero o más cabeceras en un solo cuerpo. El sobre es el elemento superior del documento XML, proporcionando un contenedor de información de control, la dirección de un mensaje, y el mensaje en sí. Las cabeceras de transporte contienen cualquier información de control, tal como los atributos de calidad de servicio. El cuerpo contiene la identificación de los mensajes y sus parámetros. Tanto los encabezados como el cuerpo son elementos secundarios del sobre.(7)

Codificación de las reglas: El conjunto de reglas de codificación, expresa los casos de aplicación de tipos de datos definidos. Reglas de codificación para definir un mecanismo de serialización que se puede utilizar para intercambiar los casos de aplicación y tipos de datos definidos. SOAP define un lenguaje de programación independiente del esquema de tipo de datos basado en la Definición de Esquemas XML (XSD)[9], además de reglas de codificación para todos los tipos de datos definidos de acuerdo con este modelo.(7)

Estilos de comunicación: Las comunicaciones pueden seguir una Llamada a Procedimiento Remoto (RPC) o un mensaje orientado a formato (documento). Estos se discuten a continuación:

1.2.4.2 Estilos de comunicación utilizados por SOAP

Entre los estilos de comunicación soportados por SOAP se encuentran las llamadas a procedimiento remoto (RPC), las cuales son la invocación de una operación que devuelve un resultado. RPC se utiliza típicamente con codificación SOAP y no es compatible con la Interoperabilidad de Servicios Web (WS-I)(7).

Estilo del documento: El estilo del documento también se conoce como el estilo orientado a documentos o mensaje-orientado. Este estilo proporciona una capa de abstracción más baja, y requiere más trabajo de programación.(7)

En entornos informáticos distribuidos, mediante la codificación de estilos se puede precisar cómo los valores de datos que se definen en la aplicación pueden ser traducidos a un formato de protocolo particular. El proceso de traducción se conoce como serialización y deserialización.(7)

[9] XSD: por sus siglas en inglés XML Schema Definition

La codificación SOAP: permite serializar/deserializar valores de tipos de datos del modelo de datos SOAP. Este estilo de codificación se define en el estándar SOAP 1.1, y no es compatible con WS-I.(7)

1.2.4.3 WSDL define el estilo de codificación XML Literal:

XML Literal: Literal se refiere al hecho de que el documento debe leerse tal como está, es decir, sin codificar. El documento se serializa con XML de Intercambio de Metadatos (XMI)[10], lo que significa que el mensaje XML se ajusta al esquema en el WSDL. Cuando se utiliza la codificación literal, cada parte del mensaje hace referencia una definición de esquema concreto. Codificación literal es compatible con WS-I.(8)

1.2.5 XML-Llamadas a Procedimientos Remotos (XML-RPC)

Es un protocolo muy simple porque solo define tipos de datos y comandos útiles, además de una descripción completa de corta extensión. La simplicidad del XML-RPC contrasta con la mayoría de protocolos RPC que tiene una documentación extensa y requiere considerable soporte de software para su uso.

- XML-RPC es un protocolo de llamada a procedimiento remoto que funciona a través de Internet.
- Un mensaje XML-RPC es una petición HTTP-POST.
- El cuerpo de la solicitud se encuentra en XML.
- Un procedimiento se ejecuta en el servidor y el valor que devuelve también está formateado en XML.
- Los parámetros del procedimiento pueden ser escalas, números, cadenas o fechas y también pueden ser de registro complejo y estructuras de lista.(9)

1.2.6 Protocolo de Descripción Exploración e Integración Universal (UDDI)

La especificación UDDI define un modo de publicar y encontrar información sobre servicios web. La misma cuenta con dos funciones, las cuales son:

[10] XMI: por sus siglas en inglés XML Metadata Interchange

- Es un protocolo basado en SOAP, que define cómo se comunican los clientes con los registros UDDI.
- Es un conjunto de registros duplicados globales en particular.

UDDI incluye un esquema XML para mensajes SOAP que define un conjunto de documentos para describir información de empresas y servicios, un conjunto común de API para consultar y publicar información en los directorios y una API para duplicar entradas de directorio entre nodos UDDI iguales.

UDDI gestiona el descubrimiento de servicios web, que confía en un registro distribuido de empresas y sus descripciones de servicio implementado en un formato XML común. Antes de poder publicar la entidad de empresa y el servicio web a un registro público, primero se debe registrar la entidad de empresa con un registro UDDI.

Registro UDDI

Los registros UDDI tienen dos formatos: público y privado. Ambos tipos se ajustan a las mismas especificaciones. Un registro privado permite publicar y probar las aplicaciones de negocio electrónico internas en un entorno seguro y privado. Un registro público es una colección de directorios iguales que contienen información sobre empresas y servicios. Localiza servicios que se registran en uno de sus nodos iguales y facilita el descubrimiento de servicios web publicados. Duplica los datos en todos los registros de forma regular, lo que asegura la coherencia en los formatos de descripción de servicios y facilita el seguimiento de los cambios a medida que se producen.(10)

1.2.7 Seguridad en Servicios Web (WS-Security)[11]

En la actualidad existen recomendaciones para abordar la seguridad de los servicios web a través de marcos de trabajo definidos por OASIS, la cual es una organización para la estandarización de servicios web con la cooperación de las empresas tecnológicas más grandes del mundo. OASIS define WS-Security para tratar las siguientes consideraciones:

- Las relaciones de confianza entre las organizaciones podrían necesitar la definición de contratos legales y responsabilidades.
- Los requerimientos a un servicio web deberían ser autenticados y autorizados apropiadamente.

[11] WS-Security: por sus siglas en inglés Web Service Security

- Los mensajes hacia y desde un servicio web deberían ser protegidos de accesos y modificación no autorizados.

WS-Security no es un reemplazo de alguna tecnología de seguridad, sino que provee un modelo unificado que utiliza las tecnologías existentes y que permite que las aplicaciones intercambien mensajes SOAP de forma segura. También provee mecanismos extensibles y flexibles, pero esta extensibilidad puede afectar la interoperabilidad de las distintas plataformas tecnológicas.

De las consideraciones antes mencionadas se derivan tres grandes problemas en la protección de intercambios de mensajes SOAP, como lo son identificar y autenticar al cliente, garantizar la integridad del mensaje y mantener el mensaje seguro contra la interceptación. La seguridad en servicios web brinda respuestas para todos los problemas antes mencionados, aunque no de una forma directa ya que no habla sobre cómo proteger el mensaje, sino cómo hacer saber al destinatario que se ha protegido el mensaje. A continuación se explica como WS-Security resuelve estos problemas:

1. El primer problema es identificar y autenticar al cliente. Debido a que hay muchas maneras diferentes de crear tokens de seguridad, WS-Security no especifica ningún medio particular, sino que define cómo se deben transferir los token de seguridad dentro de los mensajes SOAP. En otras palabras, hace saber al destinatario cómo extraer tokens de seguridad del mensaje para su procesamiento.
2. El segundo problema es garantizar la integridad del mensaje. WS-Security utiliza firmas digitales para ello, empleando la especificación de firma XML en lugar de inventar algo nuevo. La firma XML es una recomendación del Consorcio de la Web (W3C)[12] que incluye un mecanismo para firmar documentos XML de manera digital.
3. El tercer problema es mantener el mensaje seguro contra la interceptación mientras está en tránsito. Una vez más, WS-Security utiliza otro estándar W3C, esta vez hace uso del cifrado XML, que brinda un mecanismo para cifrar documentos XML.

1.3 Estilos arquitectónicos de los servicios web

Entre los estilos arquitectónicos más utilizados por los servicios web se encuentran:

[12] W3C: por sus siglas en inglés World Wide Web Consortium

- Llamadas a Procedimientos Remotos (RPC)[13]: Los servicios web basados en RPC presentan una interfaz de llamada a procedimientos y funciones distribuidas, lo cual es familiar a muchos desarrolladores. Típicamente, la unidad básica de este tipo de servicios es el trabajo con el Lenguaje de Descripción de Servicios Web (WSDL)[14]. Las primeras herramientas para servicios web estaban centradas en esta visión, algunos lo llaman la primera generación de servicios web, razón por la cual este estilo está muy extendido. Sin embargo, ha sido algunas veces criticado por no ser débilmente acoplado, ya que suele ser implementado por medio del mapeo de servicios directamente a funciones específicas del lenguaje o llamadas a métodos. Muchos especialistas creen que este estilo debe desaparecer.
- Arquitectura Orientada a Servicios (SOA)[15]: Los servicios web pueden ser implementados siguiendo los conceptos de la arquitectura SOA, donde el elemento fundamental de la comunicación es el mensaje. Esto es típicamente referenciado como servicios orientados a mensajes. Los servicios web basados en SOA son soportados por la mayor parte de desarrolladores de software y analistas. Al contrario que los servicios web basados en RPC, este estilo es débilmente acoplado, lo cual es preferible ya que se centra en el "contrato" proporcionado por el documento WSDL, más que en los detalles de implementación subyacentes.
- Transferencia de Estado Representacional (REST)[16]. Este estilo de arquitectura de software se ha popularizado en los últimos años proponiendo una nueva opción de uso de los servicios web. Los servicios web basados en REST intentan emular al protocolo HTTP o protocolos similares mediante la restricción de establecer la interfaz a un conjunto conocido de operaciones.

1.3.1 Características de REST y SOAP

Según se ha podido ver a lo largo del documento, el principal beneficio de SOAP recae en ser débilmente acoplado, permitiendo ser testado y depurado antes de poner en marcha la aplicación. En cambio, las ventajas de la aproximación basada en REST, recaen en la potencial escalabilidad de este tipo de sistemas, así como el acceso con escaso consumo de recursos a sus operaciones debido al

[13] RPC: por sus siglas en inglés Remote Procedure Calls.
[14] WSDL: por sus siglas en inglés Web Services Description Language.
[15] SOA: por sus siglas en inglés Service-oriented Architecture
[16] REST: por sus siglas en inglés Representational State Transfer

limitado número de operaciones y el esquema de direccionamiento unificado. A continuación se muestran algunas de las características de REST y SOAP, así como las ventajas y desventajas declaradas.

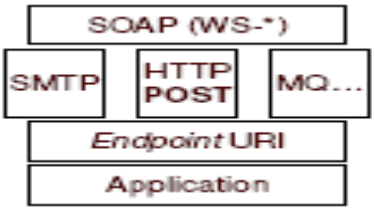
	REST	**SOAP**
Características	Las operaciones se definen en los mensajes.	Las operaciones son definidas como puertos WSDL.
	Una dirección para cada instancia del proceso.	Una dirección para todas las operaciones.
	Cada objeto soporta las operaciones estándares definidas.	Múltiples instancias del proceso comparten la misma operación.
	Componentes débilmente acoplados.	Componentes débilmente acoplados.
Ventajas declaradas	Bajo consumo de recursos.	Fácil de utilizar.
	Las instancias del proceso son creadas explícitamente.	La depuración es posible.
	El cliente no necesita información de enrutamiento a partir del Identificador Uniforme de Recurso (URI)[17] inicial.	Las operaciones complejas pueden ser escondidas detrás de una fachada.
	Los clientes pueden tener una interfaz escuchadora (*listener*) genérica para las notificaciones.	Envolver APIs existentes es sencillo.
	Generalmente fácil de construir y adoptar.	Incrementa la privacidad.
Posibles Desventajas	Gran número de objetos.	Los clientes necesitan saber las operaciones y su semántica antes del uso.
	Manejar el espacio de nombres (URIs) puede ser engorroso.	Los clientes necesitan puertos dedicados para diferentes tipos de notificaciones.
	La descripción sintáctica/semántica muy informal (orientada al usuario).	Las instancias del proceso son creadas implícitamente.
	Pocas herramientas de desarrollo.	

Tabla 1: Principales características de SOAP y REST

[17] URI: por sus siglas en inglés Uniform Resource Identifier

1.3.2 Diferencias entre REST y SOAP

A continuación se van a esbozar las diferencias entre REST y SOAP desde varios puntos de vista:

	REST	SOAP
Tecnología	Interacción dirigida por el usuario por medio de formularios.	Flujo de eventos orquestados.
	Pocas operaciones con muchos recursos	Muchas operaciones con pocos recursos.
	Mecanismo consistente de nombrado de recursos (URI).	Falta de un mecanismo de nombrado.
	Se centra en la escalabilidad y rendimiento a gran escala para sistemas distribuidos hipermedia.	Se centra en el diseño de aplicaciones distribuidas.
Protocolos		
	XML autodescriptivo.	Tipado fuerte, XML Schema.
	HTTP.	Independiente del transporte.
	HTTP es un protocolo de aplicación.	HTTP es un protocolo de transporte.
	Síncrono.	Síncrono y Asíncrono.
Descripción del servicio	Confía en documentos orientados al usuario que define las direcciones de petición y las respuestas.	WSDL.
	Interactuar con el servicio supone horas de testeado y depuración de URIs.	Se pueden construir automáticamente stubs (clientes) por medio del WSDL.
	No es necesario el tipado fuerte, si ambos lados están de acuerdo con el contenido.	Tipado fuerte.
	Lenguaje de Descripción de Aplicaciones Web (WADL).	WSDL 2.0.
Gestión del estado	El servidor no tiene estado.	El servidor puede mantener el estado de la conversación.
	Los recursos contienen datos y enlaces representando transiciones a estados válidos.	Los mensajes solo contienen datos.
	Los clientes mantienen el estado siguiendo los enlaces.	Los clientes mantienen el estado suponiendo el estado del servicio.
	Técnicas para añadir sesiones: Cookies	Técnicas para añadir sesiones: Cabecera de sesión (no estándar)
Seguridad	HTTPS.	WS-Security.
	Implementado desde hace muchos años.	Las implementaciones han comenzado a aparecer en los últimos años.

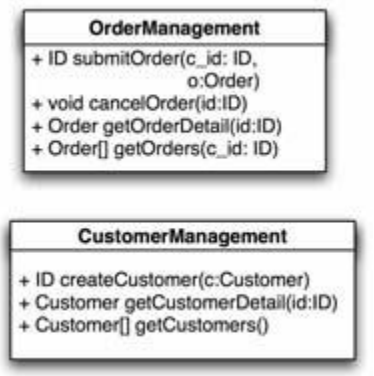
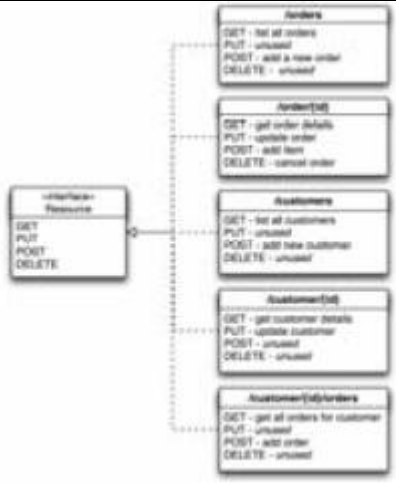
	Comunicación punto a punto segura.	Comunicación origen a destino segura.
Metodología de diseño		
	Identificar recursos a ser expuestos como servicios.	Listar las operaciones del servicio en el documento WSDL.
	Definir URLs para direccionarlos.	Definir un modelo de datos para el contenido de los mensajes.
	Distinguir los recursos de solo lectura (GET) de los modificables (POST, PUT, DELETE).	Elegir un protocolo de transporte apropiado y definir las correspondientes políticas de Calidad de Servicios (QoS), de seguridad y transaccional.
	Implementar e implantar el servidor web.	Implementar e implantar el contenedor del servicio web.

Tabla 2: Comparación entre SOAP y REST

1.4 Herramientas y metodologías de desarrollo.

1.4.1 Metodología para el desarrollo

El desarrollo de software no es una tarea fácil, prueba de ello es que existen numerosas propuestas metodológicas tanto ágiles como tradicionales que inciden en distintas dimensiones del proceso de desarrollo. Por una parte están las propuestas más tradicionales que se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir y las herramientas y notaciones que se usarán. Estas propuestas han demostrado ser efectivas y necesarias en un gran número de proyectos, pero también han presentado problemas en otros muchos.(11) Por otra parte se encuentran las metodologías ágiles basadas en heurísticas provenientes de prácticas de producción de código, que están especialmente preparadas para cambios durante el proyecto donde el cliente es parte del equipo de desarrollo y es empleada por grupos pequeños de menos de 10 personas.

Las instituciones que llevan varios años en la producción de software han obtenido una gran experiencia utilizando todas estas metodologías de desarrollo. Llevándolos a crear modelos propios que definen las fases del ciclo de vida de los proyectos que se van a desarrollar. El CEIGE cuenta con un modelo de este tipo que incluye la especificación de las actividades de cada una de las fases del ciclo de vida de los proyectos del centro teniendo en cuenta los procesos de Integración de Modelos de Madurez de Capacidades (CMMI)[18] nivel 2 para la UCI. Se detallan por tanto los artefactos a generar en cada momento, independientemente de las herramientas o métodos que se utilicen para ello, Modelo de Desarrollo de Software Versión 1.1(12)

1.4.2 Lenguaje de modelado

1.4.2.1 Lenguaje Unificado de Modelado (UML)[19]

Este lenguaje prescribe un conjunto de notaciones y diagramas estándares para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan; posibilitando así visualizar, especificar y documentar los artefactos o toda información que se obtiene o modifica durante un proceso de desarrollo de software, además de poder utilizarse para modelar distintos tipos de sistemas de software, hardware y organizaciones del mundo real.(13)

Principales características:

- Posibilita mejores tiempos totales de desarrollo.
- Permite modelar sistemas, no sólo de software.
- Tecnología orientada a objetos.
- Establecer conceptos y artefactos ejecutables.
- Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- Mejor soporte a la planeación y al control de proyectos.
- Alta reutilización y minimización de costos.

[18] CMMI: por sus siglas en inglés Capability Maturity Model Integration
[19] UML: por su siglas en inglés Unified Modeling Language

1.4.3 Herramienta Case: Visual Paradigm para UML

Es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, haciéndolas mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.(14)

1.4.4 Lenguajes de programación

1.4.4.1 PHP (PHP Hipertext Preprocesor)

PHP es el acrónimo recursivo que significa *PHP Hypertext Pre-processor*, es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación.(15)

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores. (15)

Una vez que se conoce el concepto de lenguaje de programación de scripts del lado del servidor se da paso a enunciar algunas de las fortalezas que brinda. PHP al escribirse dentro del código HTML se hace realmente fácil de utilizar, al igual que ocurre con ASP[20] de Microsoft, pero con algunas ventajas como su gratuidad, independencia de plataforma, rapidez y seguridad. Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor web. Esto hace que cualquier sistema pueda ser compatible con el lenguaje y significa una ventaja importante, ya que permite portar el sitio desarrollado en PHP de un sistema a otro fácilmente. PHP, en el caso de estar

[20] ASP: por sus siglas en inglés Active Server Pages.

montado sobre un servidor Linux o Unix, es más rápido que ASP, dado que se ejecuta en un único espacio de memoria y esto evita las comunicaciones entre componentes COM que se realizan entre todas las tecnologías implicadas en una página ASP.

Por último, es necesario señalar que respecto a la seguridad, es importante el hecho de que en muchas ocasiones PHP se encuentra instalado sobre servidores Unix o Linux, que brindan un elevado nivel de seguridad intrínseco, presentan una gran estabilidad y el acceso al código fuente permite personalizar el funcionamiento y auditar la seguridad y privacidad de los datos tratados. PHP permite configurar el servidor de modo que se apruebe o rechacen diferentes usos, lo que puede hacer al lenguaje tener un nivel de seguridad elevado dependiendo este de las necesidades de cada cual. Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos hasta tratamiento de conexiones de red.

Principales capacidades de PHP:

- Compatibilidad con las bases de datos más comunes, como MySQL, Oracle, Informix, y ODBC
- Incluye funciones para el envío de correo electrónico, subida de archivos, crear dinámicamente en el servidor imágenes en formato GIF, incluso animadas y una larga lista de utilidades adicionales.
- Para desarrollar en este lenguaje no se requiere tener grandes capacidades de hardware. Luego en el caso de los servidores, una aplicación desarrollada en PHP no requiere tanta memoria. Además tiene una de las comunidades más grandes en Internet, por lo que es fácil encontrar ayuda, documentación, artículos, noticias y demás recursos.(15)

1.4.4.2 Javascript

Es un lenguaje de programación utilizado para crear pequeños programas encargados de realizar acciones dentro del ámbito de una página web. Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento, siendo uno de los más utilizados con este fin y que brinda muchas posibilidades, porque permite la programación de pequeños scripts, pero también de programas más grandes orientados a objetos con funciones, estructuras de datos, entre otras.

Javascript pone a disposición del programador todos los elementos que forman la página web, para que este pueda acceder a ellos y modificarlos dinámicamente, posibilitándole el control total sobre la misma.(16)

1.4.5 Marco de trabajo: Symfony 2.1

Es un framework que ayuda a simplificar el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.(14) Entre las características más destacadas que ofrece a los desarrolladores de productos de software se encuentran las siguientes:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo. (14)

1.4.6 Gestor de Base de Datos: Oracle 11g

Es un Sistema de Gestión de Base de Datos Objeto-Relacional (ORDBMS)[21], desarrollado por la corporación Oracle. Se considera como uno de los sistemas de bases de datos más completos, destacando:

- Compilación automática de PL/SQL y Java en la base de datos.
- Triggers más rápidos que en versiones anteriores, incluidas las llamadas más eficientes de triggers por fila.
- Operaciones SQL sencillas más rápidas.
- Replicación más rápida en *Oracle Data Guard* y *Oracle Streams*.
- Conexiones directas más rápidas confiables con los dispositivos de almacenamiento del Sistema de Archivos de la Red (NFS, por sus siglas en inglés).
- Actualizaciones más rápidas
- *Backup*/restauración más rápida de archivos grandes.
- Compresión de *backup* más rápida.

1.4.7 Entorno de Desarrollo Integrado (IDE)[22]: NetBeans 7.3

Es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extenderlo. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso.(17)

Entre las utilidades que brinda este IDE se encuentra la de crear, modificar y eliminar ficheros WSDL a través del WSDL Editor, y esquemas XML mediante el *XML Schema Designer*. Los ficheros WSDL y XSD se crean a través de un *wizzard* en el que, además de propiedades comunes como el nombre y el lugar de ubicación del fichero, se pueden especificar, dentro del primero los tipos de datos para un esquema XML. Esto permite la reducción de la necesidad de volver a especificar estos elementos para un WSDL que puedan estar ya en la aplicación y pueden disponer de varios tipos de enlaces para protocolos específicos. El *WSDL Editor* tiene tres vistas sincronizadas, *Source view*, que muestra el código XML, *WSDL view* que cuenta con varias opciones de visualización y *Partner view* en la que los

[21]ORDBMS: por sus siglas en inglés de Object-Relational Data Base Management System
[22] IDE: por sus siglas en inglés Integrated Development Environment

mensajes y la interacción con diferentes *partners* pueden ser modelados gráficamente. El *XML Schema Designer* tiene, además de *Source view* y *Schema view*, la vista de diseño, en la que se pueden agregar visualmente elementos, tipos complejos y otros tipos de datos o características que admitan los XSD.(17)

1.4.8 Servidor de Aplicaciones: Apache 2.2.2

Apache es el servidor web por excelencia, su facilidad de configuración, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. Se ejecuta en gran cantidad de sistemas operativos, lo que lo hace prácticamente universal. Es una tecnología gratuita, *open source* y altamente configurable de diseño modular por lo que resulta muy sencillo ampliar las sus capacidades. Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor y es posible configurarlo para que ejecute un determinado script cuando esto suceda.(18)

1.4.9 Controlador de versiones: Subversion (SVN)

Es un sistema de control de versiones libre y de código fuente abierto. Gestiona un árbol de ficheros en un repositorio central. El repositorio es como un servidor de ficheros ordinario, excepto porque recuerda todos los cambios hechos a sus ficheros y directorios. Esto le permite recuperar versiones antiguas de sus datos, o examinar el historial de cambios de los mismos. Subversion es un sistema general que puede ser usado para administrar cualquier conjunto de ficheros. Pudiendo ser estos ficheros código fuente, archivos multimedia, o cualquier otro tipo de documento.(19)

Ventajas

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.
- Las modificaciones son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente. Tiene costo de complejidad constante (O(1)).
- Se envían sólo las diferencias en ambas direcciones.
- Puede ser servido mediante Apache, sobre WebDav/DeltaV.
- Maneja eficientemente archivos binarios.
- Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.

- Cuando se usa integrado a Apache permite utilizar todas las opciones que este servidor provee a la hora de autentificar archivos (SQL, LDAP, PAM, etc.).

1.5 Conclusiones Parciales

Una vez realizado el estudio del marco teórico de la presente investigación se arribó a las siguientes conclusiones:

- Se realizó un estudio que arrojó las principales tendencias, estilos y estándares para la implementación de servicios web y proveer seguridad a los mismos.
- Se propone el empleo de una arquitectura orientada a servicio.
- Se utilizarán estándares como XML, WSDL, UDDI, WS-Security y dentro de la arquitectura orientada a servicios se escogió para su uso SOAP.
- Para el desarrollo del sistema se empleará el modelo de desarrollo del centro CEIGE en su versión 1.1 y la herramienta CASE Visual Paradigm para la descripción y modelado del sistema.
- Como sistema gestor de base de datos se utilizará Oracle en su versión 11g.
- La implementación del sistema se hará empleando IDE NetBeans y como marco de trabajo Symfony en su versión 2.1. Para la capa de presentación se utilizará Bootstrap.
- Para el control de versiones se define Subversion v1.6 y como servidor de aplicaciones el Apache2.

Capítulo 2: Características del Sistema

En el presente capítulo se describen los principales procesos involucrados en el objeto de estudio y los conceptos relacionados con el dominio del problema. Se realiza una descripción general de los requisitos funcionales y no funcionales del sistema a desarrollar. Así como, las descripciones del diseño elaborado para alcanzar las metas trazadas, además de la estrategia a seguir durante el proceso de implementación.

2.1 Propuesta de solución

En el capítulo anterior se realizó un estudio de las tecnologías existentes para lograr implementar servicios web, por lo que se propone el uso de WSDL para su descripción, para la comunicación el protocolo SOAP bajo transporte HTTP y para su registro UDDI. Siendo estos los tres estándares más comunes que se utilizan en una SOA. La solución que se propone consiste en crear un Bundle que se encargue de contener la implementación de los servicios que serán publicados para ser consumidos por los clientes externos. Una vez implementados estos servicios se procede a parsearlos y obtener la información necesaria para generar la UDDI de la aplicación.

Debido a que se espera que la cantidad de servicios web disponibles aumente de manera considerable en poco tiempo, se hace necesario un registro que permita su administración. Este registro es un lugar central donde se pueden ubicar los servicios web de manera categorizada. Por lo que se propone utilizar la iniciativa UDDI.

Este componente también será el encargado de gestionar la seguridad de los servicios web permitiendo autentificar a los sistemas externos que necesiten conectarse. Luego de estar autentificado el sistema le dará acceso solo a los servicios que esté autorizado a consumir. Para la autenticación de los sistemas externos se propone utilizar un token de seguridad encriptado que contendrá el nombre del usuario y su respectiva contraseña con el algoritmo asimétrico RSA[23]. Al encriptar estos datos con la clave pública, se impide que esta información sea legible al viajar por la red y permite que solo pueda ser desencriptada por la llave privada de la VUCEC. Una vez que este token llegue al componente de servicios web se desencriptaría y se procedería a verificar si es válido

[23] RSA: por sus siglas en inglés Rivest, Shamir y Adleman

el usuario y su contraseña, luego se confirmaría si este tiene acceso al servicio que está solicitando. En el caso de cumplir con todo esto se le enviaría una respuesta indicándole que se realizó la operación correctamente. En caso de ser negativa se le notificaría que las credenciales presentadas no son válidas.

La estructura que va a presentar este Bundle sería la siguiente:

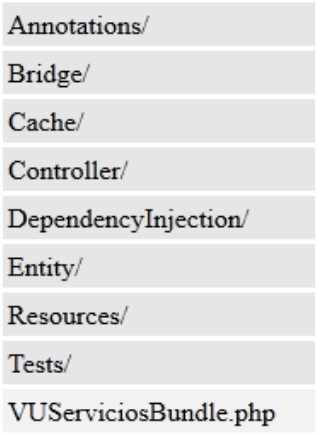


Figura 1: Estructura de la solución.

El directorio Bridge contendrá los paquetes de servicios web que van a ser consumidos por la VUCEC pertenecientes a cada uno de los sistemas externos con los que se van a interactuar. El directorio Controller contendrá los paquetes de servicios web que publicará cada subsistema de la VUCEC para que sean consumidos por los sistemas externos autorizados. El directorio Entity contendrá las entidades que necesite el negocio.

2.2 Modelo dominio

Un modelo del dominio se utiliza con frecuencia como fuente de inspiración para el diseño de los objetos de software, y será una entrada necesaria para varios de los artefactos que se generarán en el diseño de la solución. El modelo del dominio muestra a los modeladores las clases conceptuales significativas en un dominio del problema. Es el artefacto más importante que se crea durante el análisis orientado a objetos.(20)

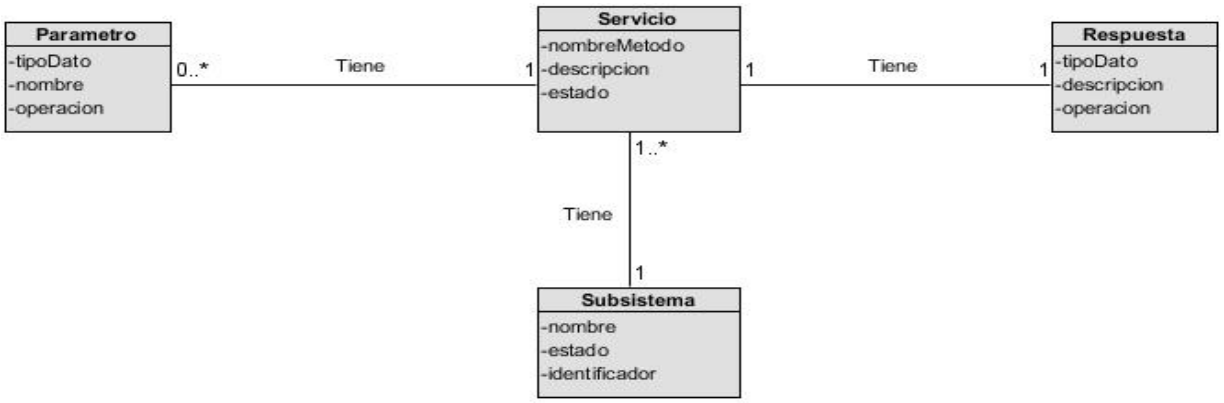


Figura 2: Modelo conceptual.

A continuación se describe una de las clases pertenecientes a este modelo:

Servicios

Descripción	Almacena la información de los servicios existentes					
Atributos						
Nombre	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clases válidas	Clases no válidas
nombreMetodo	Nombre por el cual se identifica el servicio web	String	No	Si	Caracteres alfabéticos A...Z,a...z.	Caracteres Especiales,. / ?][';!~@#$%^&*()_+-. Caracteres numéricos 0..9
descripción	Descripción del servicio	String	No	No	Caracteres alfabéticos A...Z,a...z. Caracteres Especiales,. / ?][';!~@#$%^&*()_+-. Caracteres numéricos 0..9	Ninguna

estado	Estado en el que se encuentra el servicio (publicado o no)	Boolean	No	No	True, False	Caracteres Especiales,. / ?][';!~@#$%^&*()_+-. Caracteres numéricos 0..9 Caracteres alfabéticos A...Z,a...z.

Tabla 3: Descripción de la clase conceptual Servicio.

2.3 Requisitos de Software

2.3.1 Técnicas para la captura de requerimientos

Siguiendo el procedimiento para la Ingeniería de Requisitos en el Departamento de Desarrollo de Soluciones para la Aduana del CEIGE, el cual plantea sobre las técnicas para la captura de requerimientos lo siguiente y cito: “Teniendo en cuenta las características del cliente, la AGR, se decidió utilizar las siguientes técnicas que permitirán a los analistas, la recopilación y obtención de la información necesaria para la elicitación de requisitos, las mismas son: entrevista, observación, tormenta de ideas y talleres.”(21) Se decide utilizar para la captura de requisitos la **tormenta de ideas** donde en conjunto con el jefe de proyecto, el arquitecto y la analista principal, se acumularon ideas para tener una perspectiva general de las necesidades del sistema.

2.3.2 Requisitos funcionales

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas y situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer. Los requerimientos funcionales de un sistema describen lo que debe hacer. Estos requerimientos dependen del tipo de software que se desarrolle, de los posibles usuarios del software y del enfoque general tomado por la organización al redactar requerimientos. Cuando se expresan como requerimientos del usuario, habitualmente se describen de una forma bastante abstracta. Sin embargo, los requerimientos funcionales del sistema describen con detalle la función de este.(22)

Se identificaron los requisitos a cumplir por el sistema, listados a continuación:

Nº	Agrupación	Funcionalidad	Descripción	Complejidad
RF1	Gestionar Servicios Web	Generar Servicios Web	Genera los servicios pertenecientes al subsistema seleccionado.	Media
RF2	Gestionar Servicios Web	Modificar Servicio Web	Modifica los datos del servicio seleccionado.	Media
RF3	Gestionar Servicios Web	Eliminar Servicios Web	Genera los servicios pertenecientes al subsistema seleccionado.	Media
RF4	Generar UDDI	Generar UDDI	Genera la UDDI a partir de los servicios generados.	Media
RF5	Implementar Servicios Web	ObtenerAduanasDespacho	Devuelve las aduanas de despacho registradas en el GINA	Baja
RF6	Implementar Servicios Web	ObtenerDeclaranteDadoCodigo	Devuelve los datos del declarante dado su código en el GINA.	Baja
RF7	Implementar Servicios Web	ObtenerSolicitanteDadoCi	Devuelve los datos del solicitante dado su número de carné de identidad.	Baja
RF8	Implementar Servicios Web	ObtenerEntidadDadoSolicitante	Realiza una búsqueda que devuelva una entidad dado el código de la entidad y el CI del solicitante	Baja
RF9	Implementar Servicios Web	ObtenerEntidadDadoDeclarante	Realiza una búsqueda que devuelva una entidad dado el código de la entidad y el código del declarante	Baja
RF10	Implementar Servicios Web	ObtenerLugaresEmbarque	Devuelve los lugares de embarque registrados en el GINA	Baja
RF11	Implementar Servicios Web	ObtenerTiposFacilidad	Devuelve los diferentes tipos de facilidad registrados en el GINA	Baja

RF12	Implementar Servicios Web	ObtenerMotivosAsociadosSolFacilidad	Devuelve los motivos asociados a la solicitud de facilidad.	Baja
RF13	Implementar Servicios Web	ObtenerOperacionesTipoFacilidad	Devuelve las operaciones que puede realizar la entidad dada atendiendo al tipo de facilidad (tipo de DM) también dado.	Baja
RF14	Implementar Servicios Web	ObtenerMotivosLiberacionCotejo	Devuelve los motivos asociados a la solicitud de liberación de cotejo.	Baja
RF15	Implementar Servicios Web	ObtenerOperacionesLiberacionCotejo	Devuelve las operaciones que puede realizar la entidad en solicitudes de liberación de cotejo	Baja
RF16	Implementar Servicios Web	ObtenerOperacionesRegimenTemporal	Devuelve las operaciones que puede realizar determinada entidad en las solicitudes de régimen temporal.	Baja
RF17	Implementar Servicios Web	ObtenerProvincias	Devuelve las provincias del país.	Baja
RF18	Implementar Servicios Web	ObtenerRegimenes	Devuelve los regímenes que sean temporales, que sean de los que se solicitan, que se correspondan con la operación, que estén autorizados para la aduana de despacho y que estén autorizados para la entidad.	Baja
RF19	Implementar Servicios Web	ObtenerMunicipiosDadaProvincia	Devuelve los municipios pertenecientes a una provincia determinada	Baja
RF20	Implementar Servicios Web	ObtenerTipoTelefono	Devuelve los tipos de teléfonos.	Baja
RF21	Implementar Servicios Web	ObtenerMotivosSolCodigoEventual	Devuelve los motivos asociados a la solicitud de código eventual.	Baja

RF22	Implementar Servicios Web	ObtenerOperacionesSolFacilidadDMGlobal	Devuelve las operaciones que puede realizar la entidad en solicitudes de Facilidad DM Global.	Baja
RF23	Implementar Servicios Web	ObtenerPaises	Devuelve los países.	Baja
RF24	Implementar Servicios Web	ObtenerEscaques	Devuelve los escaques que pueden ser solicitados como códigos eventuales.	Baja
RF25	Implementar Servicios Web	ObtenerOperacionesSolProrrogaPagoFacilidad	Devuelve las operaciones asociadas a la solicitud de prórroga pago facilidad.	Baja
RF26	Implementar Servicios Web	ObtenerDMAsociadaSolProrrogaPagoFacilidad	Devuelve los siguientes datos asociados a la DM (código del declarante de la DM, nombre y apellidos del declarante, código y nombre de la entidad y tipo de facilidad (tipo de DM)).	Baja
RF27	Implementar Servicios Web	ObtenerMotivosAsociadosSolProrrogaPagoFacilidad	Devuelve los motivos asociados a la solicitud de prórroga pago facilidad.	Baja
RF28	Implementar Servicios Web	ObtenerOperacionesSolProrrogaRegimenTemporal	Devuelve las operaciones asociadas a la solicitud de prórroga régimen temporal.	Baja
RF29	Implementar Servicios Web	ObtenerDMSolProrrogaRegimenTemporal	Se devuelve los siguientes datos asociados a la DM (código del declarante, nombre del declarante y apellidos del declarante de la DM, código y nombre de la entidad) y de los segmentos de artículos que contengan regímenes temporales que se soliciten el número de orden, la partida, el régimen y la descripción de la mercancía.	Baja

RF30	Implementar Servicios Web	ObtenerOperacionesSolProrrogaEspecialRegimenTemporal	Devuelve las operaciones asociadas a la solicitud de prórroga especial de régimen temporal.	Baja
RF31	Implementar Servicios Web	ObtenerDMSolProrrogaEspecialRegimenTemporal	Se devuelve los siguientes datos asociados a la DM (código del declarante, nombre del declarante y apellidos del declarante de la DM, código y nombre de la entidad) y de los segmentos de artículos que contengan regímenes temporales que se soliciten el número de orden, la partida, el régimen y la descripción de la mercancía.	Baja
RF32	Implementar Servicios Web	ObtenerOperacionesSolAnulacionDm	Devuelve las operaciones asociadas a la solicitud de anulación de la DM.	Baja
RF33	Implementar Servicios Web	ObtenerDMAsociadaSolAnulacionDm	Devuelve los siguientes datos asociados a la DM (código del declarante de la DM, nombre del declarante de la DM, apellidos del declarante de la DM, código y nombre de la entidad de la DM)	Baja
RF34	Implementar Servicios Web	ObtenerMotivosAnulacionDM	Devuelve los motivos asociados a la solicitud de anulación de la DM	Baja
RF35	Implementar Servicios Web	ObtenerOperacionesSolDesanulacionDm	Devuelve las operaciones asociadas a la solicitud de anulación de la DM	Baja
RF36	Implementar Servicios Web	ObtenerDMAsociadaSolDesanulacionDm	Devuelve los siguientes datos asociados a la DM (código del declarante de la DM, nombre del declarante de la DM, apellidos del declarante de la DM, código y nombre de la entidad de la DM)	Baja
RF37	Implementar Servicios Web	obtenerMotivosDesanulacionDM	Devuelve los motivos asociados a la solicitud de anulación de la DM	Baja

RF38	Implementar Servicios Web	ObtenerOperacionesSolDevolucionMercanciasProveedor	Devuelve las operaciones asociadas a la solicitud de devolución de mercancías al proveedor	Baja
RF39	Implementar Servicios Web	ObtenerDMSolDevolucionMercanciasProveedor	Devuelve los siguientes datos asociados a la DM: código del declarante, nombre del declarante, apellidos del declarante, código de la entidad, código NIT de la entidad, nombre de la entidad, número de manifiesto, bl o ga, número de BL/GA, código de buque o aeronave.	Baja
RF40	Implementar Servicios Web	ObtenerRegimenDevolucionMercProv	De cada uno de los regímenes de devolución devuelve el código y la descripción	Baja
RF41	Implementar Servicios Web	ObtenerOperacionesSolDevolucionDerechoAduana	Devuelve las operaciones asociadas a la solicitud de devolución de derechos de aduana.	Baja
RF42	Implementar Servicios Web	ObtenerDMSolDevolucionDerechosAduana	Devuelve los siguientes datos asociados a la DM: número de manifiesto, numero de BL/GA, numero de Buque/Aeronave, nombre del declarante, código de la entidad importador, nombre de la entidad Importador y de cada una de las partidas asociadas a la DM devuelve: número de orden, código, descripción, estado	Baja
RF43	Implementar Servicios Web	ObtenerDatosCompradorDadoCodigo	Obtiene el nombre y el código NIT de un comprador dado su código	Baja
RF44	Implementar Servicios Web	ObtenerMotivosSolDevolucionDerechosAduana	Obtiene el código y la descripción de los motivos asociados a solicitud de devolución derechos de aduana	Baja

RF45	Implementar Servicios Web	ObtenerMonedasPagoTramiteDM	Obtiene el código y la descripción de las monedas para el pago del trámite de la DM	Baja
RF46	Implementar Servicios Web	ObtenerOperacionesSolPlanificacionRecFisico	Devuelve las operaciones asociadas a la solicitud de planificación del reconocimiento físico.	Baja
RF47	Implementar Servicios Web	ObtenerLugaresPlanificacionReconocimientoFisico	Devuelve los lugares para la planificación de reconocimiento físico registrados	Baja
RF48	Implementar Servicios Web	obtenerDMSolPlanificacionRecFisico	Servicio que obtiene los siguientes campos de la DM: número de manifiesto, número de BL/GA, lugar de embarque, nombre del declarante de la DM.	Baja
RF49	Implementar Servicios Web	ObtenerOperacionesSolRealizacionReconocimientoFisico	Devuelve las operaciones asociadas a la solicitud.	Baja
RF50	Implementar Servicios Web	ObtenerDMSolRealizacionReconocimientoFisico	Servicio que obtiene los siguientes campos de la DM: código Declarante de la DM, nombre del Declarante de la DM, apellidos del declarante de la DM, numero de manifiesto, numero de buque o aeronave, bl/ga, numero de BL/GA, código del importador/exportador, nombre del importador/exportador, código NIT del importador/exportador, listado de contenedores(si tiene).	Baja
RF51	Implementar Servicios Web	ObtenerMotivoSolRealizacionReconocimientoFisico	Obtiene el código y la descripción de los motivos de la solicitud.	Baja
RF52	Implementar Servicios	ObtenerLugaresReal	Obtiene el código y la descripción de los	Baja

	Web	izacionReconocimientoFisico	lugares de realización de reconocimiento físico.	
RF53	Implementar Servicios Web	ObtenerDMSolReintegro	Servicio que obtiene los siguientes campos de la DM: numero manifiesto, número BL/GA, Fecha liquidación, cantidad de subpartidas, valor de las operaciones. Descripción Po Arancel, Descripción de la Mercancía, Cantidad, descripción UM, Índice de Consumo, Merma, Importe solicitado, código y nombre del importador	Baja
RF54	Implementar Servicios Web	ObtenerEntidadSolCodigoEventualDadoSolicitante	Realiza una búsqueda que devuelva una entidad dado el código de la entidad y el CI del solicitante	Baja
RF55	Implementar Servicios Web	ObtenerEntidadSolCodigoEventualDadoDeclarante	Realiza una búsqueda que devuelva una entidad dado el código de la entidad y el código del declarante	Baja
RF56	Implementar Servicios Web	Cambiar Estado Documento	Cambia el estado del documento pasado por parámetro	Media
RF57	Implementar Servicios Web	Asignar Numero Sistema	Asigna un numero al sistema	Media
RF58	Implementar Servicios Web	Adicionar Notificación	Adiciona una notificación con los parámetros pasados por parámetro	Media
RF59	Implementar Servicios Web	Crear Usuario Persona Externa	Crea un usuario a partir de los datos introducidos por parámetro.	Media
RF60	Implementar Servicios Web	Procesar Solicitud Facilidad	Procesa la solicitud de Facilidad a partir de los parámetros que se pasan de la misma.	Media
RF61	Implementar Servicios	Procesar Solicitud	Procesa la solicitud de Liberación Cotejo	Media

	Web	Liberación Cotejo	a partir de los parámetros que se pasan de la misma	
RF62	Implementar Servicios Web	Procesar Solicitud Régimen Temporal	Procesa la solicitud de Régimen Temporal a partir de los parámetros que se pasan de la misma	Media
RF63	Implementar Servicios Web	Procesar Solicitud Código Eventual	Procesa la solicitud de Código Eventual a partir de los parámetros que se pasan de la misma	Media
RF64	Implementar Servicios Web	Procesar Solicitud Facilidad Dm Global	Procesa la solicitud de Facilidad Dm Global a partir de los parámetros que se pasan de la misma	Media
RF65	Implementar Estándar de Seguridad de Servicios Web	Implementar Estándar de Seguridad de Servicios Web	Implementa el estándar de seguridad empleado para la autentificación, autorización de los usuarios, así como de la encriptación de la información que es enviada y recibida.	Alta

Tabla 4: Requisitos Funcionales.

2.3.2.1 Descripción del Requisito Gestionar Servicios Web

Descripción textual del requisito Generar Servicios Web

Precondiciones		• El actor ha sido autenticado en el sistema.
Flujo de eventos		
Flujo básico Generar Servicios Web		
1	El sistema muestra la interfaz con los datos: Nombre subsistemas, Estado, Identificador	
2	Se selecciona el subsistema al que se le desean generar los servicios web y selecciona la opción Editar.	
3	El sistema muestra una interfaz que contiene los siguientes datos: Nombre, Acción a realizar, Identificador.	
4	El usuario selecciona la acción generar servicios web.	
5	Se ejecuta el paso 2 tantas veces como dese el usuario.	
6	Se selecciona la opción Aceptar.	
7	El sistema genera los servicios web correspondientes al subsistema seleccionado.	
8	El sistema muestra un mensaje notificando que el subsistema ha sido actualizado.	
9	Concluye el requisito.	
Pos-condiciones		
1	Se actualiza el subsistema y se generan los servicios web.	
Flujos alternativos		
1	N/A	
Pos-condiciones		
1	N/A	
Validaciones		
1	N/A	
Conceptos	Subsistema	Visibles en la interfaz: • Nombre subsistema, acción realizada, subsistema Utilizados internamente: • N/A

Requisitos especiales	N/A
Asuntos pendientes	N/A

Tabla 5: Descripción del requisito funcional Generar Servicio Web.

Prototipo elemental de interfaz gráfica de usuario

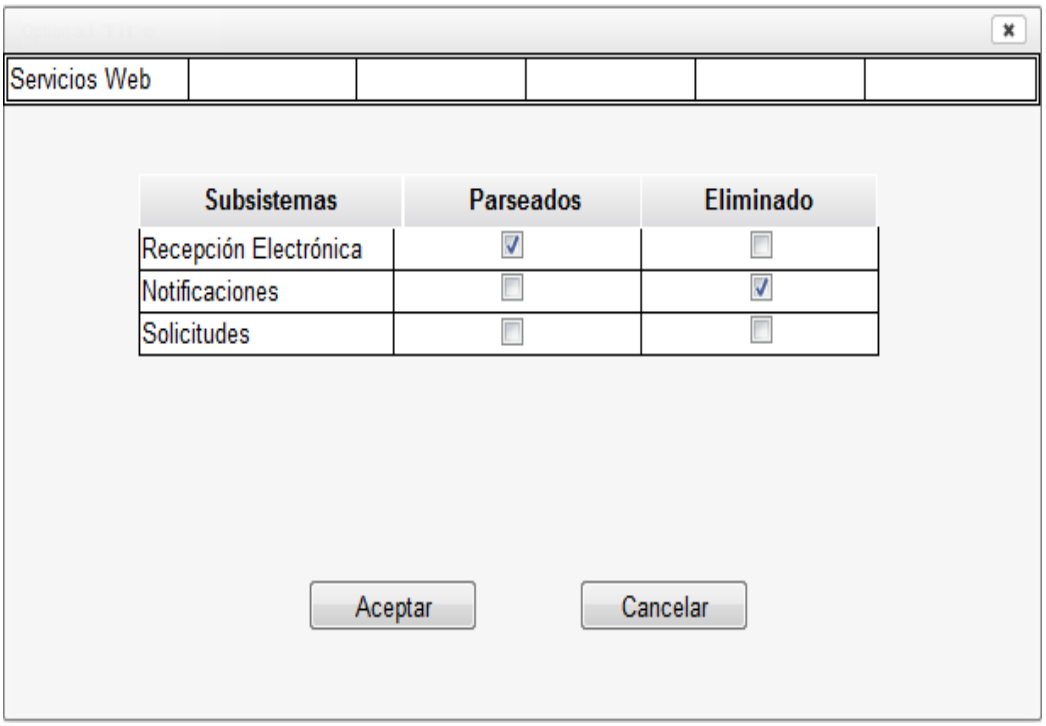


Figura 3: Prototipo de Interfaz de Usuario.

Descripción textual del requisito Modificar Servicios Web

Precondiciones	• El actor ha sido autenticado en el sistema.
Flujo de eventos	
Flujo básico Modificar Servicios Web	
1	El sistema carga la interfaz con todos los datos de los servicios web. Dichos datos son: Método, Descripción, Estado, Parámetros, Salida, Subsistema.
2	El usuario selecciona el servicio que desea modificar y selecciona la opción

	Editar	
3	El sistema carga una interfaz que contiene los siguientes datos: Subsistema al que pertenece dicho servicio, Estado, Nombre del Método, Descripción, Tabla con sus Parámetros (Nombre, Tipo de dato, Operación), Tabla con la Salida (Tipo de dato, Operación), Descripción de salida.	
4	El usuario modifica los datos deseados.	
5	Se ejecuta el paso 2 tantas veces como dese el usuario del Flujo Básico de Eventos Modificar Servicios Web.	
6	Seleccionar la opción Aceptar.	
7	El sistema actualiza el dato seleccionado.	
8	Concluye el requisito.	
Pos-condiciones		
1	Se modifican los datos del servicio seleccionado	
Flujos alternativos		
1	N/A	
Pos-condiciones		
1	N/A	
Validaciones		
1	N/A	
Conceptos	**Servicios**	Visibles en la interfaz: • Método, descripción, estado, parámetros, salida, subsistema. Utilizados internamente: • N/A
Requisitos especiales	N/A	
Asuntos pendientes	NA	

Tabla 6: Descripción del requisito funcional Modificar Servicio Web.

Prototipo elemental de interfaz gráfica de usuario

Figura 4: Prototipo de Interfaz de Usuario.

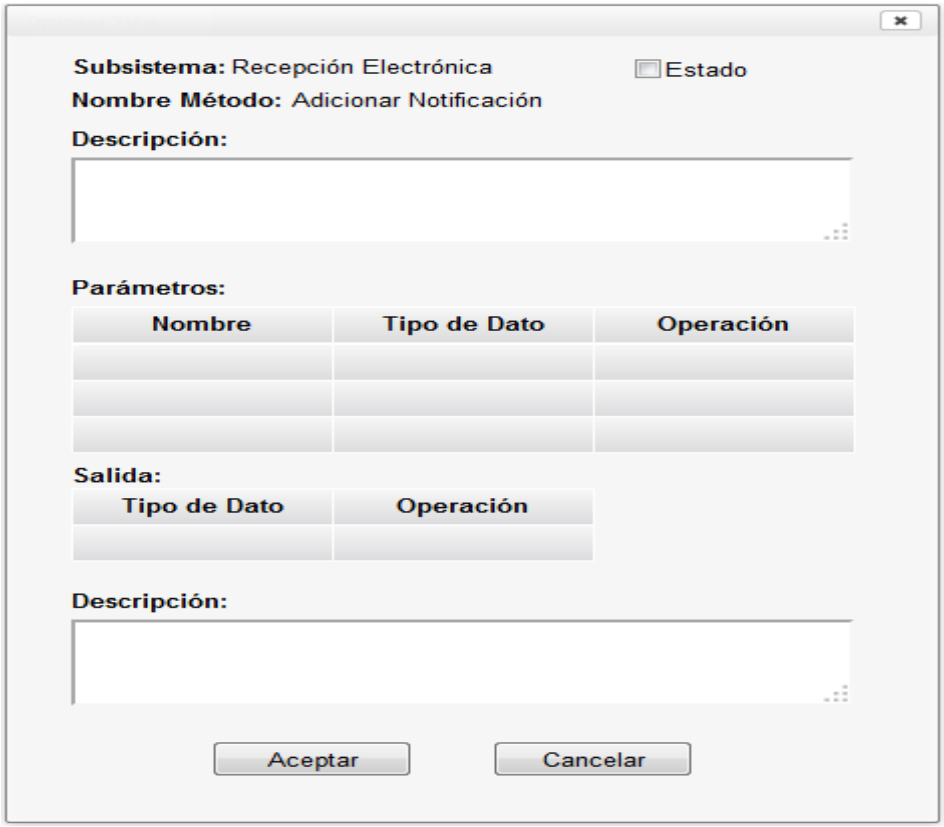


Figura 5: Prototipo de Interfaz de Usuario.

Descripción textual del requisito Eliminar Servicios Web

Precondiciones		• El actor ha sido autenticado en el sistema.
Flujo de eventos		
Flujo básico Generar Servicios Web		
1	El sistema muestra la interfaz con los datos: Nombre subsistemas, Estado, Identificador	
2	Se selecciona el subsistema al que se le desean eliminar los servicios web y selecciona la opción Editar.	
3	El sistema muestra una interfaz que contiene los siguientes datos: Nombre, Acción a realizar, Identificador.	
4	El usuario selecciona la acción eliminar servicios web.	
5	Se ejecuta el paso 2 tantas veces como dese el usuario.	
6	Se selecciona la opción Aceptar.	
7	El sistema elimina los servicios web correspondientes al subsistema seleccionado.	
8	El sistema muestra un mensaje notificando que el subsistema ha sido actualizado.	
9	Concluye el requisito.	
Pos-condiciones		
1	Se eliminan los servicios web asociados al subsistema seleccionado.	
Flujos alternativos		
1	N/A	
Pos-condiciones		
1	N/A	
Validaciones		
1	N/A	
Conceptos	Subsistema	Visibles en la interfaz: • Nombre subsistema, acción realizada, subsistema Utilizados internamente: • N/A
Requisitos	N/A	

especiales	
Asuntos pendientes	N/A

Tabla 7: Descripción del requisito funcional Eliminar Servicio Web.

2.3.3 Requisitos no funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requerimientos no funcionales a menudo se aplican al sistema en su totalidad. Normalmente, apenas se aplican a características o servicios individuales del sistema. Los requerimientos no funcionales, como su nombre sugiere, son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema. A continuación se describen los requisitos no funcionales que fueron detectados.(22)

2.3.3.1 Rendimiento

Los tiempos de respuesta y velocidad de procesamiento de la información serán rápidos, no mayores de 3 segundos para las actualizaciones y 10 para las recuperaciones.

2.3.3.2 Seguridad

Confiabilidad: El sistema debe prevenir posibles fallos y/o errores y recuperarse ante ellos. La información manejada por el sistema está protegida de acceso no autorizado.

Integridad: La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción y estados inconsistentes.

Disponibilidad: Los usuarios autorizados tendrán garantizado el acceso a la información en todo momento independientemente de los mecanismos utilizados para manipular los datos.

2.3.3.3 Software

Para el cliente:

- Navegador Mozilla Firefox 7.0 o superior o Chrome 1.0 o superior.
- Sistema operativo Windows 98 o superior o Linux.

Para el servidor:

- Sistema operativo Linux en cualquiera de sus distribuciones.
- Un servidor Apache 2.0 o superior con módulo PHP 5.3.6 disponible o superior. Este debe estar configurado con la extensión “oci” incluida.
- Un servidor de base de datos Oracle en su versión 11g.

2.3.3.4 Hardware

Para el servidor:

- Requerimientos mínimos: Procesador Core 2 Duo a 2.0GHz de velocidad de procesamiento y 4Gb de memoria RAM.
- Al menos 80Gb de espacio libre en disco duro.
- Tarjeta de red.

Para el cliente:

- Requerimientos mínimos: Procesador Pentium IV a 2.0GHz con 512Mb de memoria RAM.
- Tarjeta de red.

2.3.4 Técnicas para la validación de requerimientos

Siguiendo el procedimiento para la Ingeniería de Requisitos en el Departamento de Desarrollo de Soluciones para la Aduana del CEIGE, el cual plantea sobre las técnicas para la validación de los requerimientos lo siguiente:

En el procedimiento propuesto se recomienda el uso de varias técnicas para la correcta validación de los requisitos en el Departamento de Soluciones para la Aduana las cuales son: revisiones del

documento de requerimientos, construcción de prototipos, matrices de trazabilidad y generación de casos de pruebas.(23)

La validación de los requerimientos se realizó con la combinación las siguientes técnicas: revisiones del documento de requerimientos y construcción de prototipos. Se realizaron diversas **revisiones al documento de requerimientos** con la participación de la analista principal del proyecto, el arquitecto y el jefe del proyecto, para logar una correcta interpretación de la información transmitida, los señalamientos planteados fueron recogidos y aplicados posteriormente. Además se efectuó la **construcción de prototipos**, los cuales fueron presentados por cada requerimiento de software.

2.4 Modelo de diseño del componente ServiciosBundle

El Modelo de diseño es una abstracción del modelo de implementación y su código fuente, el cual fundamentalmente se emplea para representar y documentar su diseño. Es utilizado como entrada esencial en las actividades relacionadas a la implementación. El modelo de diseño puede contener: diagramas, clases, paquetes, subsistemas, cápsulas, protocolos, interfaces, relaciones, colaboraciones y atributos.(24) Dentro de este epígrafe se verán los elementos que se tuvieron en cuenta durante el diseño de la solución, dando paso así a la exposición de los resultados de este flujo de trabajo, que refleja cómo será implementado el sistema en términos de clases del diseño.

2.4.1 Diagrama de clases

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación, contiene información como clases, asociaciones, atributos, métodos y dependencias. A continuación se describe el diagrama de clases del diseño basado en estereotipos web que se realizó durante el proceso de desarrollo perteneciente al RF 2. Ver figura 6

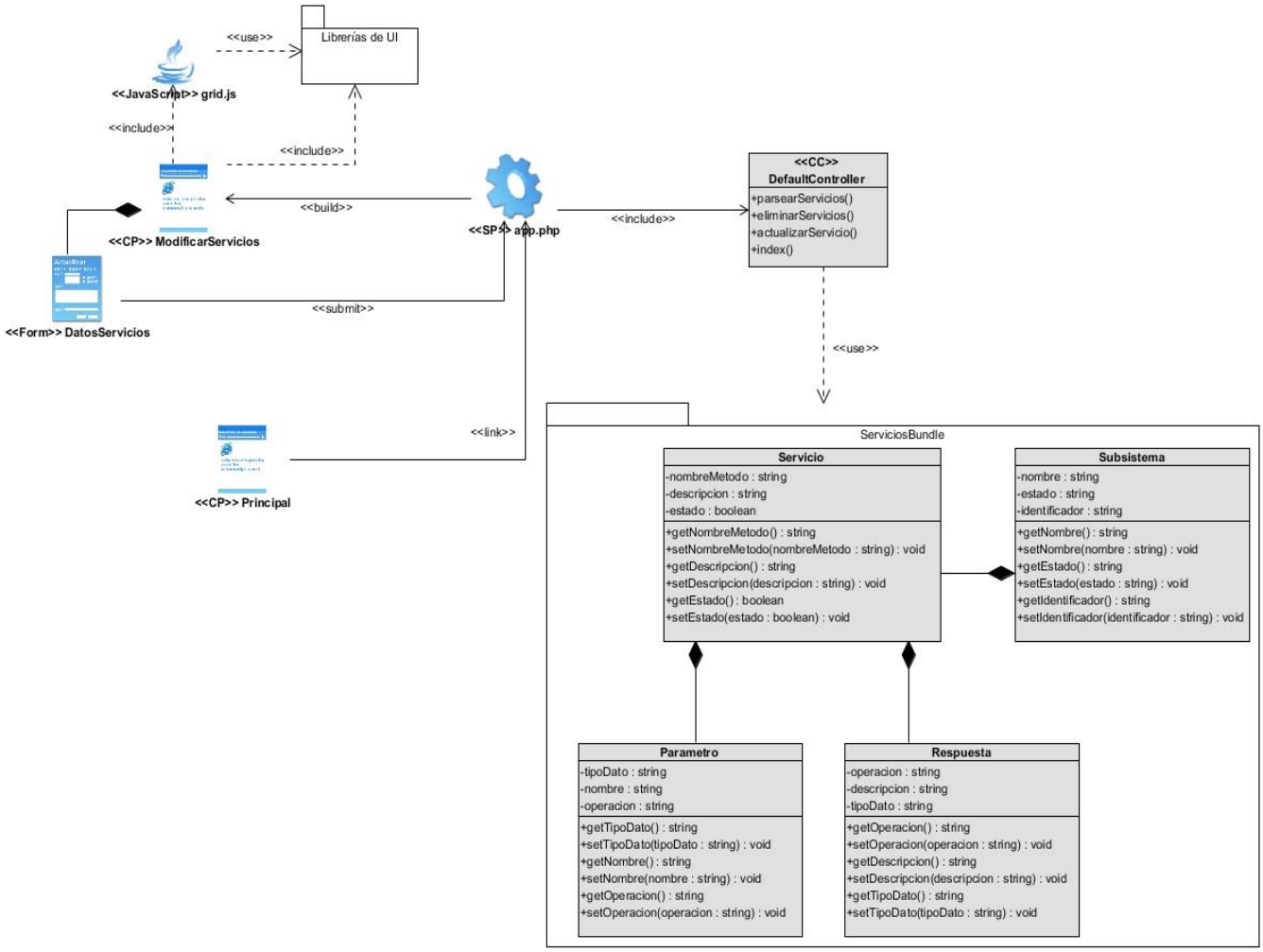


Figura 6: Diagrama de clases del diseño.

El diagrama anterior contiene dos páginas clientes, la principal y la clase modificar servicio. La primera de esta es la encargada de hacer un link a la página servidor indicándole que esta debe construir una instancia de la página cliente modificar servicio la cual se encargara de mostrar todos los datos de los servicios a modificar utilizando un grid. Esta página cliente también contiene un formulario en el cual se carga y se modifican los datos de un servicio específico y al ser enviado este formulario a la página servidora esta hace la llamada al método actualizarServicio() ubicado en la clase controladora, el cual es el encargado de actualizar la información de dicho servicio web usando las clases entidades Subsistema, Método, Parámetros y Respuesta.

2.4.2 Patrones Utilizados

2.4.2.1 Patrones de Asignación de Responsabilidades (GRASP)

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. A continuación se describen los utilizados en el desarrollo del componente.

Bajo acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Acoplamiento bajo significa que una clase no depende de muchas clases. Acoplamiento alto significa que una clase recurre a muchas otras clases.(25) Esto presenta los siguientes problemas:

- Los cambios de las clases afines ocasionan cambios locales.
- Difíciles de entender cuando están aisladas.
- Difíciles de reutilizar puesto que dependen de otras clases.

Entre los beneficios que trae utilizar un bajo acoplamiento se encuentran:

- No se afectan por cambios de otros componentes.
- Fáciles de entender por separado.
- Fáciles de reutilizar.

Este patrón se evidencia en la clase AGRBridge.

Alta cohesión: La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Una baja cohesión hace muchas cosas no afines o realiza trabajo excesivo.(25) Esto presenta los siguientes problemas:

- Son difíciles de comprender.
- Difíciles de reutilizar.
- Difíciles de conservar.
- Las afectan constantemente los cambios.

Entre los beneficios que trae utilizar una alta cohesión se encuentran:

- Mejoran la claridad y facilidad con que se entiende el diseño.
- Se simplifica el mantenimiento y las mejoras de funcionalidad.
- A menudo se genera un bajo acoplamiento.
- Soporta mayor capacidad de reutilización.

Este patrón se evidencia en la clase AGRBridge.

Experto: El uso de este patrón permite que se conserve el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. Esto soporta un bajo acoplamiento, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento. El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clases sencillas y más cohesivas que son más fáciles de comprender y de mantener. Así se brinda soporte a una alta cohesión.(25)

Este patrón se evidencia en la clase AGRBridge ya que la misma es la que permite la conexión con todos los servicios que brinda el sistema GINA.

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

Entre los beneficios que brinda la utilización del patrón controlador se encuentra:

- Mayor potencial de los componentes reutilizables. Garantiza que la empresa o los procesos de dominio sean manejados por la capa de los objetos del dominio y no por la de la interfaz.
- Reflexionar sobre el estado del caso de uso. A veces es necesario asegurarse de que las operaciones del sistema sigan una secuencia legal o poder razonar sobre el estado actual de la actividad y las operaciones en el caso de uso subyacente.(25)

Este patrón se evidencia en las clases NotificacionesApiController, AGRApiController y AdministracionApiController.

2.4.2.2 Patrones GoF (Gang Of Four)

Puente: El patrón *bridge* es muy útil en casi cualquier desarrollo de software, porque permite la manipulación de la implementación en tiempo de ejecución. Es utilizado para desacoplar una abstracción de su implementación, de manera que ambas puedan ser modificadas independientemente sin necesidad de alterar por ello la otra.(26)

Este patrón se evidencia en la clase AGRBridge, que es la encargada de contener todos los servicios que van a ser consumidos por los subsistemas de la VUCEC.

Inyección de dependencias: La inyección de dependencias es un patrón de diseño, consistente en que los objetos que necesita una clase para funcionar no son creados dentro de la propia clase, sino que son suministrados desde afuera.(27) Este patrón se evidencia en la clase AGRBridge, la cual al heredar de la clase Bridge esta le inyecta el contenedor.

2.4.3 Diagrama de secuencia

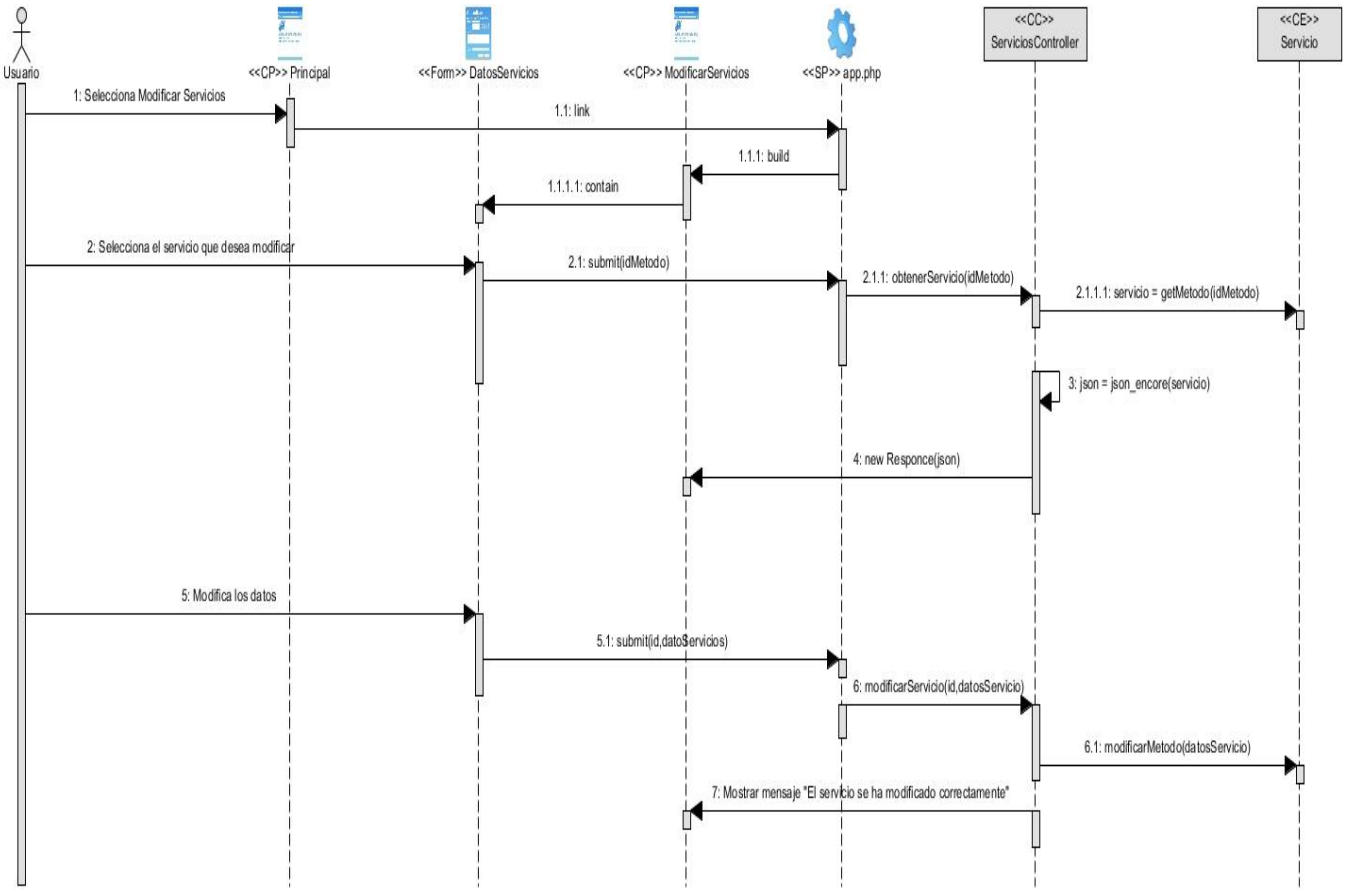


Figura 7: Diagrama de secuencia.

2.4.4 Diagrama de paquetes

Un paquete es un mecanismo utilizado para agrupar elementos de UML. Permite organizar los elementos modelados con UML, facilitando de esta forma el manejo de los modelos de un sistema complejo. Permiten dividir un modelo para agrupar y encapsular sus elementos en unidades lógicas individuales. En general, pueden tener una interfaz (métodos de clases e interfaces exportadas) y una realización de estas interfaces (clases internas que implementan dichas interfaces).

Se pueden utilizar para plantear la arquitectura del sistema a nivel macro. Los paquetes pueden estar anidados unos dentro de otros y unos paquetes pueden depender de otros paquetes.(28) A continuación se muestra el diagrama de paquetes del sistema:

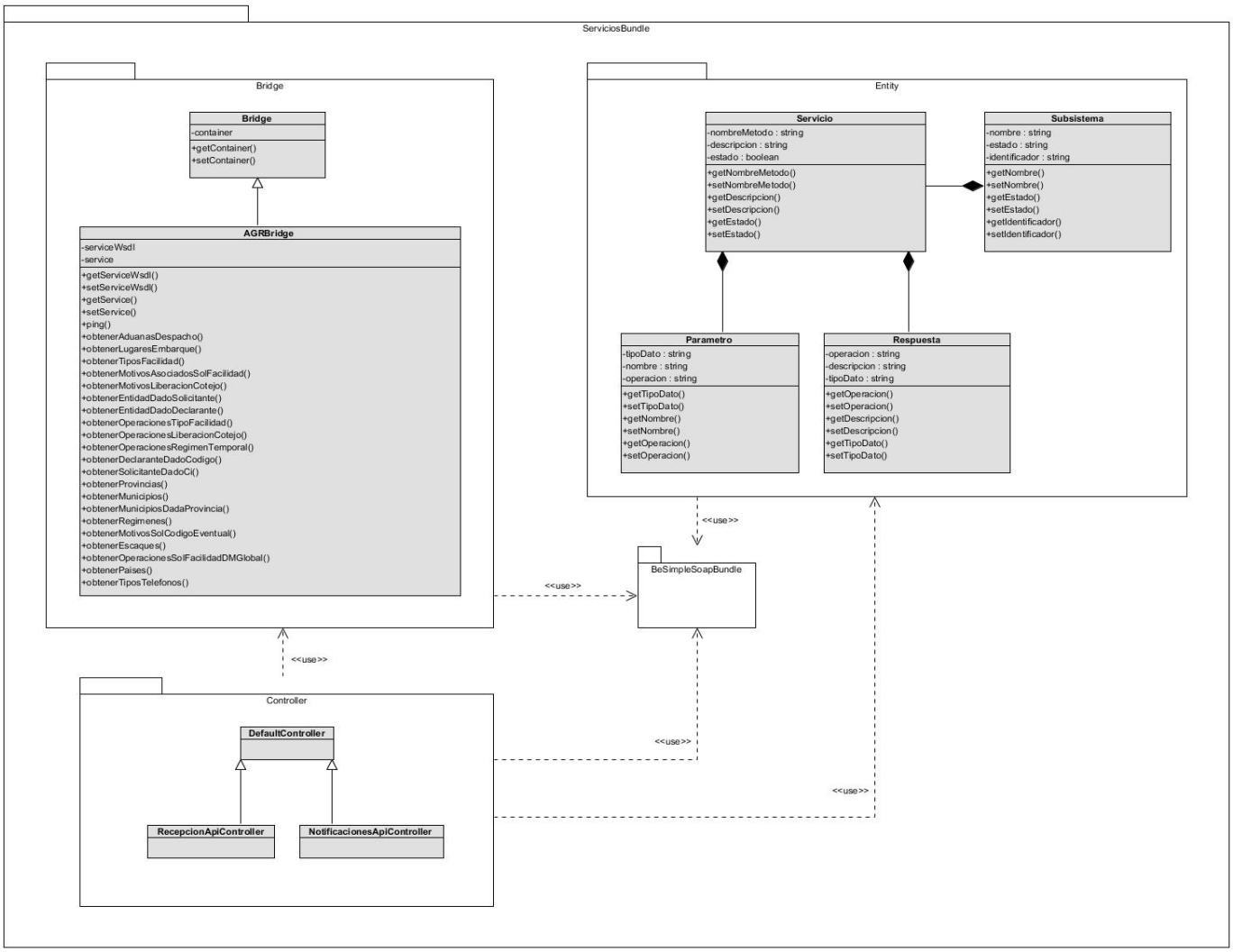


Figura 8: Diagrama de paquetes.

El componente está compuesto por los paquetes *Bridge, Controller, Entity* y *BeSismpleSoapBundle* de los cuales el paquete *Bridge* es el encargado de manejar los servicios que consume la VUCEC de los diferentes clientes externos. En paquete *Controller* es el encargado de contener la implementación de los servicios que brinda cada uno de los subsistemas de la VUCEC. El paquete *Entity* contiene las entidades necesarias que permiten registrar la información de los servicios que se publicarán. El paquete BeSimpleSoapBundle es el encargado de la publicación de los servicios web mediante Soap y WSDL.

2.5 Modelo de datos

Un modelo de datos es una colección de conceptos bien definidos matemáticamente que ayudan a expresar las propiedades estáticas y dinámicas de una aplicación con un uso de datos intensivo.(29) El modelo de datos de la solución cuenta con cuatro clases persistentes. Para su confección se tuvieron en cuenta los procesos de normalización y la utilización de nomencladores para facilitar su diseño. A continuación se muestra el diagrama entidad-relación que lo describe.

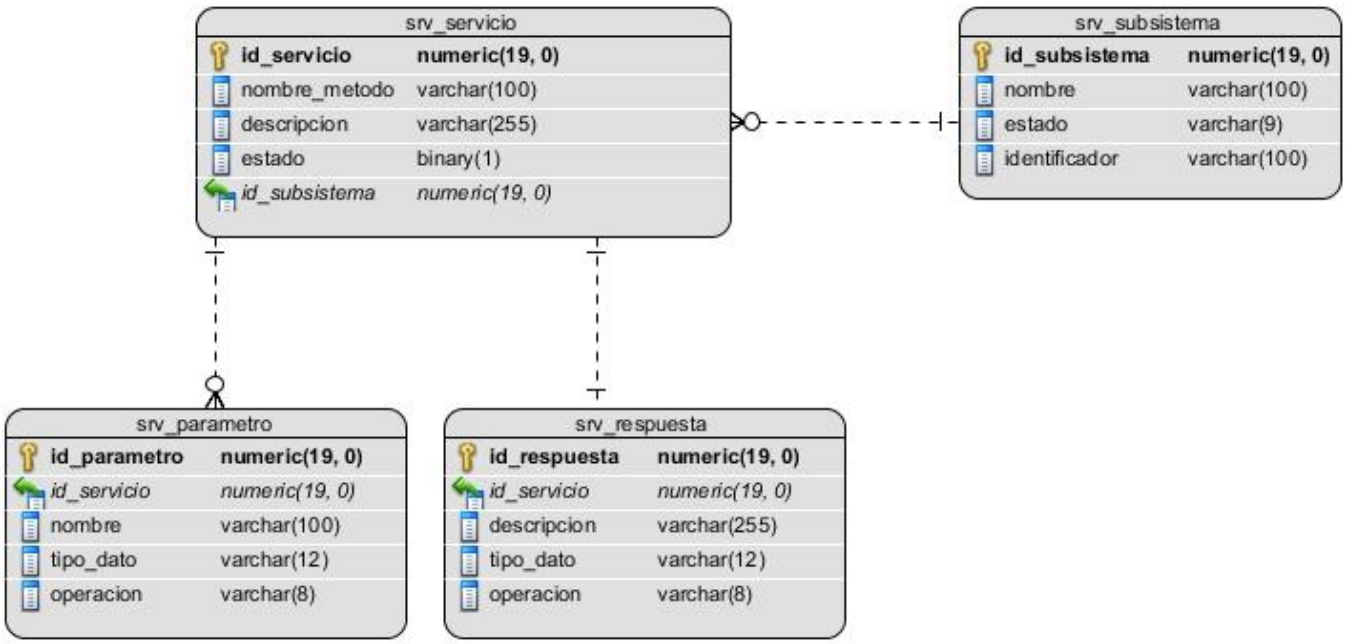


Figura 9: Diagrama del modelo de datos.

2.6 Métricas para la evaluación del diseño

Las métricas de software son una medida cuantitativa que permite a los desarrolladores tener una visión profunda de la eficacia del proceso del software y de los proyectos que dirigen utilizando el proceso como un marco de trabajo. Se reúnen los datos básicos de calidad y productividad. Estos datos son entonces analizados, comparados con promedios anteriores, y evaluados para determinar las mejoras en la calidad y productividad. Las métricas son también utilizadas para señalar áreas con problemas de manera que se puedan desarrollar los remedios y mejorar el proceso del software.(30)

2.6.1 Métrica de Tamaño Operacional de Clase (TOC)

La métrica TOC se basa esencialmente en la cantidad de funcionalidades contenidas en las clases, a partir de las cuales se determina la afectación que ejerce en el diseño. A continuación se describen los tipos de afectaciones que se pueden observar al evaluar el diseño según la métrica.

Atributos	Afectación
Responsabilidad	El aumento del TOC implica el aumento de la responsabilidad asignada a la clase.
Complejidad de implementación	El aumento del TOC implica el aumento de la complejidad de implementación de la clase.
Reutilización	El aumento del TOC implica la disminución del grado de reutilización de la clase.

Tabla 8: Afectaciones en el diseño según la métrica TOC.

Para la evaluación de cada atributo se establece un rango de valores que determinan la complejidad en la aplicación del TOC. A continuación se muestra el rango de valores y la complejidad por cada uno de los atributos.

	Criterio	Categoría	
Responsabilidad	Baja	<=Promedio	<= 11,9
	Media	>Promedio y <= 2*Promedio	>11,9<=23,8
	Alta	>2*Promedio	>23,8
Complejidad	Baja	<=Promedio	<=11,9

Implementación	Media	>Promedio y <= 2*Promedio	>11,9<=23,8
	Alta	>2*Promedio	>23,8
Reutilización	Baja	>2*Promedio	>23,8
	Media	>Promedio y <= 2*Promedio	>11,9<=23,8
	Alta	<=Promedio	<= 11,9

Tabla 9: Descripción del requisito funcional Generar Servicio Web.

A continuación se muestran los resultados de la evaluación de la métrica:

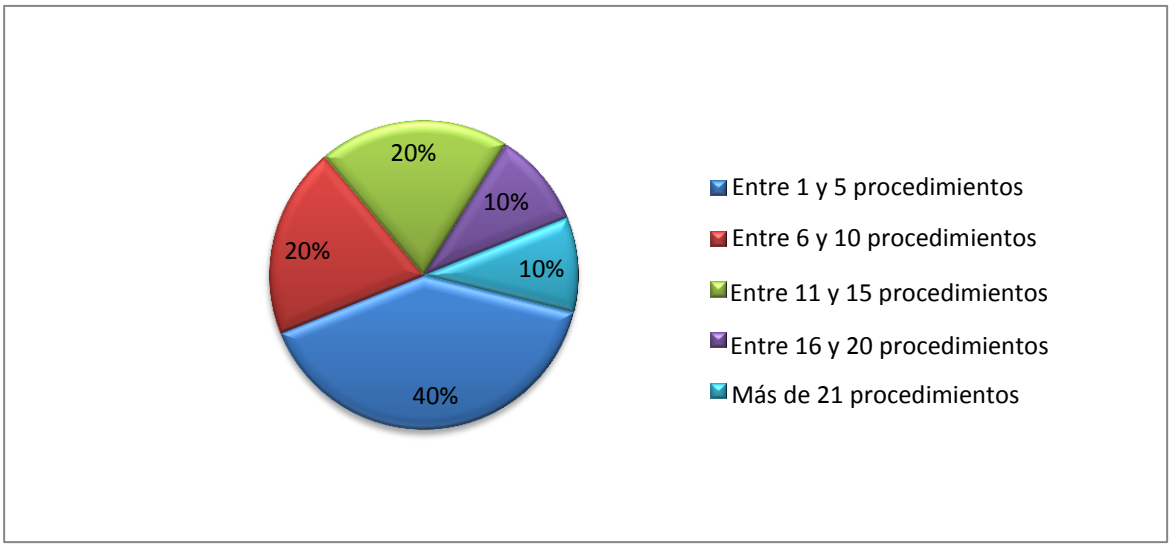


Figura 10: Representación del tamaño de las clases del subsistema ServiciosBundle.

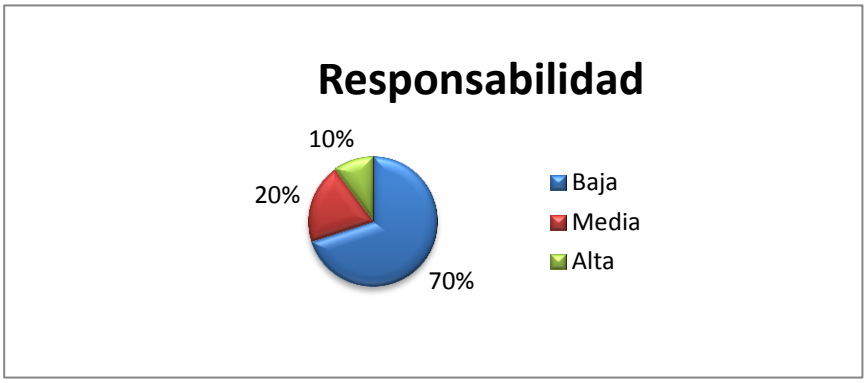


Figura 11: Representación de las clases según la responsabilidad.

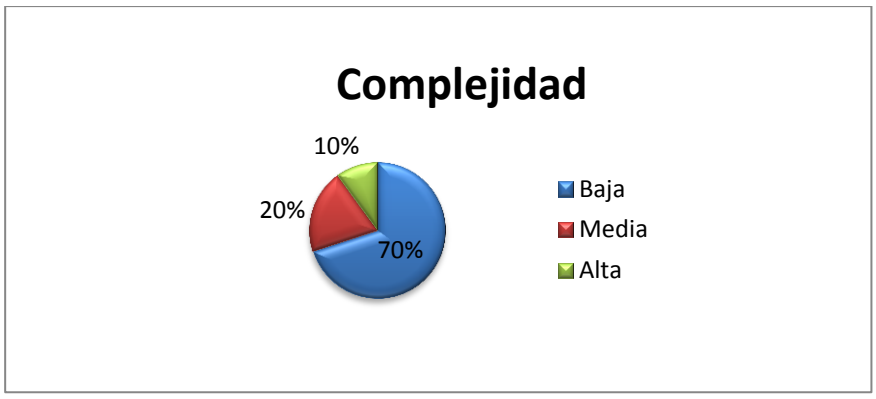


Figura 12: Representación de las clases según la complejidad de implementación.

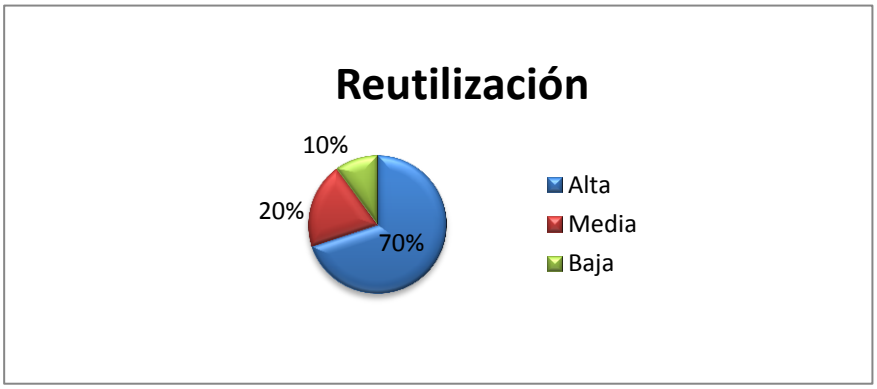


Figura 13: Representación de las clases según la reutilización.

Una vez obtenidos los resultados de la evaluación del instrumento de medición de la métrica TOC, se puede concluir que el diseño propuesto tiene una calidad aceptable, teniendo en cuenta que el 70% de las clases empleadas en el sistema posee una elevada reutilización (elemento clave en el proceso de desarrollo de software) y cómo se reducen la responsabilidad y la complejidad de implementación.

2.7 Conclusiones parciales

- En el presente capítulo se detallaron las características fundamentales de la propuesta de solución para comprender mejor las acciones a desarrollar por el componente de servicios.
- Con el análisis y diseño de la solución se obtuvieron 65 requisitos funcionales para el componente modelado y se generaron los artefactos definidos por el Departamento de Soluciones para la Aduana, obteniéndose entre ellos el diagrama Entidad-Relación, con un total de 4 entidades y normalizado en Tercera Forma Normal.
- Estos artefactos, unido al buen uso de los patrones de diseño permitieron sentar las bases para la implementación del software con un alto grado de calidad.

Capítulo 3: Implementación y Prueba

En el presente capítulo se muestra el modelo de implementación que pone en práctica el diseño de la solución realizado en el capítulo anterior y se especifica el conjunto de validaciones y pruebas que evalúan la calidad del sistema.

3.1 Diagrama de componentes

El diagrama de componentes presenta elementos tangibles como son los archivos. Se utiliza, para describir la estructura física del código de la aplicación en términos de sus componentes (código fuente, binario o ejecutable) y sus dependencias. Estos son agrupados en paquetes según sus funcionalidades. A continuación se muestra el diagrama de componentes de la solución:

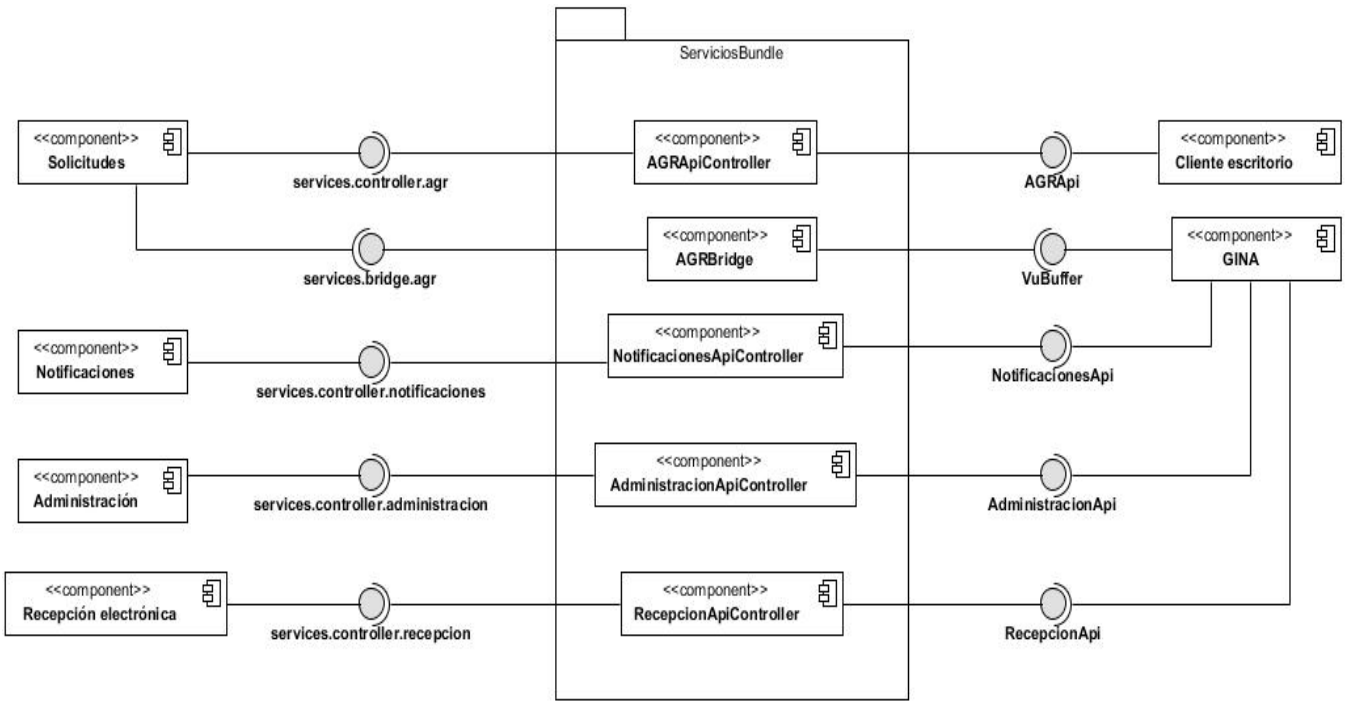


Figura 14: Diagrama de componentes.

3.2 Estándares de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, este debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si todo el código

fuente fuese escrito por un único programador. Para definir el estándar de codificación se siguieron las reglas del proyecto VUCEC, del departamento Aduana, del CEIGE que se plantean a continuación:(31)

- Los nombres de las clases deben estar expresados en notación UpperCamelCase[24].
- No se deben utilizar guiones bajos en su nombre "_".
- El nombre de las clases debe expresar con claridad el alcance y la responsabilidad de la misma.
- Los nombres de las clases no deben estar atados a las clases de las que se deriva, cada clase debe tener un significado por ella misma.
- Los formularios tendrán el sufijo Form para identificarlos.

Ejemplo:

serviciosForm, parametrosForm, etc

- Todas las funciones definidas por los desarrolladores deben seguir la nomenclatura UpperCamelCase, a no ser que para cierto ámbito se especifiquen características específicas.
- Los nombres de las funciones deben dejar reflejado claramente cuál es la acción que realiza el mismo.

Ejemplo:

AdicionarSubsistema, ObtenerParametro, etc.

- Cualquier operación relacionada con una tabla de la base de datos debe situarse en el modelo de la clase correspondiente a dicha tabla.
- Los nombres de las variables deben expresar claramente el contenido de la misma.
- En caso de que no se le asigne un valor inicial a las variables se deben inicializar con un valor que indique el tipo de dato al que debe pertenecer.
  - Los tipos de datos cadena son definidos con comillas dobles (").
  - Los tipos de datos de caracteres se definen con comillas simples (').
  - En caso de que se espere almacenar tipos de datos diversos no se inicializa.

Ejemplo:

```
$numeroBajas = 0;

$nombreAcotado = "";
```

[24]**UpperCamelCase**: Consiste en escribir frases o palabras compuestas eliminando los espacios intermedios y poniendo en mayúscula la primera letra de cada palabra incluyendo la primera letra de la frase.

```
$arreglo_temp = array();

$objeto = new Clase;

$mixta;
```

- Para la documentación de las variables y las funciones se utilizará un comentario de bloque donde se especifique el objetivo de las mismas, el o los tipos de parámetros y el tipo de retorno si es necesario para la función.

Ejemplo de la documentación de una función:

```
/**
*	Comentario u objetivo de la función
*	@param $message
*	@return void
*/
public function ParsearServicios ($clase,$idSubsistema){
	//		TODO
$result = array() ; // comentario u objetivo de la función
}
```

3.3 Tratamiento de errores

En el desarrollo de software el tratamiento de errores es importante para garantizar el correcto funcionamiento del sistema y que no se afecte la calidad del mismo. Es fundamental identificar y controlar los problemas que puedan presentarse a la hora de interactuar con el software. En el desarrollo de la solución se pueden apreciar diferentes mecanismos para el tratamiento de errores, los cuales son mencionados a continuación:

Se utiliza Javascript para depurar los errores de parte del cliente validando los formularios y evitando consultas a la base de datos sin sentido o que tengan malas intenciones como son las inyecciones sql*, en cualquiera de estas situaciones el Javascript funciona como una barrera y no deja pasar estos valores lanzando alertas o excepciones.

Las validaciones del negocio se encargan de controlar el flujo de los datos recibidos en el controlador para evitar consecuencias alarmantes. Se llevan a cabo en las diferentes clases con el uso de

funciones propias del lenguaje. En caso de que exista algún error, es notificado al usuario para que pueda corregirlos.

Otro de los aspectos importantes utilizado en el tratamiento de errores del sistema desarrollado es la utilización de los formularios generados por Symfony para insertar, eliminar o modificar valores de la base de datos, utilizando cada uno de los validadores implementados de manera que se garantiza una mayor coherencia de los mismos.

3.4 Diagrama de despliegue

El modelo de despliegue describe la distribución física del sistema, muestra cómo están distribuidos los componentes de software entre los distintos nodos de cómputo. Permite comprender la correspondencia entre la arquitectura software y la arquitectura hardware. A continuación se muestra el diagrama de despliegue perteneciente a la solución:

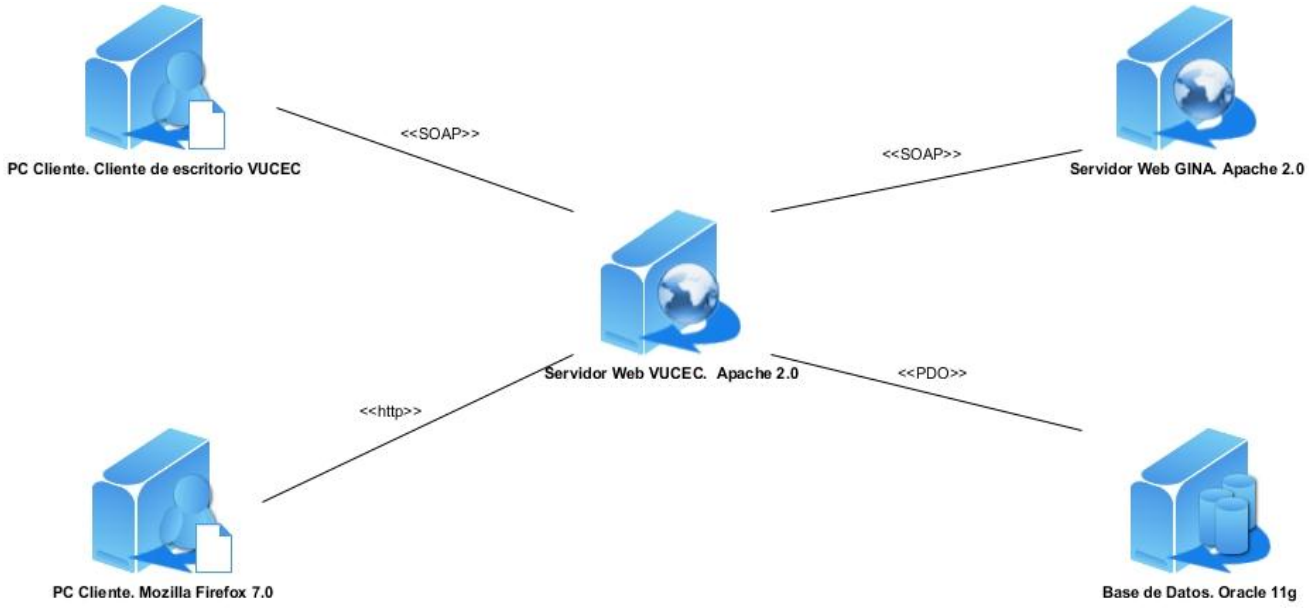


Figura 15: Diagrama de despliegue.

3.5 Pruebas del Software

La prueba del software es un elemento crítico para la garantía de la calidad del software, representa una revisión final de las especificaciones, del diseño y de la codificación y estos elementos juntos componen la aplicación computacional. A nivel general, la etapa de pruebas del software consiste en que el ingeniero crea una serie de casos de prueba que intentan demoler el software construido.

3.5.1 Prueba del camino Básico

La prueba del camino básico es una técnica de prueba de caja blanca. El método del camino básico permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y utilizar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. Los casos de prueba obtenidos del conjunto básico garantizan que durante la prueba se ejecute por lo menos una vez cada sentencia del programa. Con el uso del cálculo de la complejidad ciclomática, se obtiene la cantidad de caminos que se deben buscar. La misma está basada en la teoría de grafos y brinda una métrica del software extremadamente útil.

Antes de realizar el método del camino básico se realiza una representación del flujo de control, denominada grafo de Flujo o grafo del programa. Este permite trazar mejor los caminos del programa. Para ello se representa el flujo de control lógico mediante la notación del grafo de flujo mostrada a continuación:

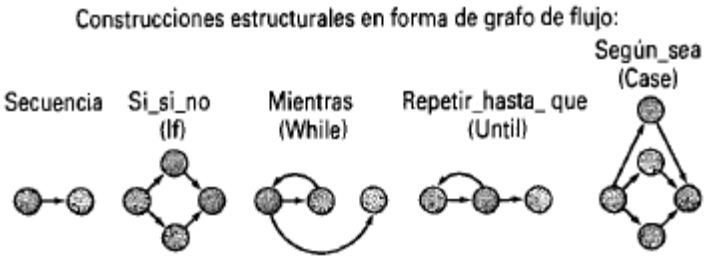


Figura 16: Notación del grafo de flujo.

A continuación se muestra el resultado de la aplicación de la prueba del camino básico a la funcionalidad AdicionarNotificacion(), perteneciente a la clase NotificacionesApiController, primero se realiza la elaboración del correspondiente grafo de flujo y a partir de este se calcula la complejidad ciclomática.

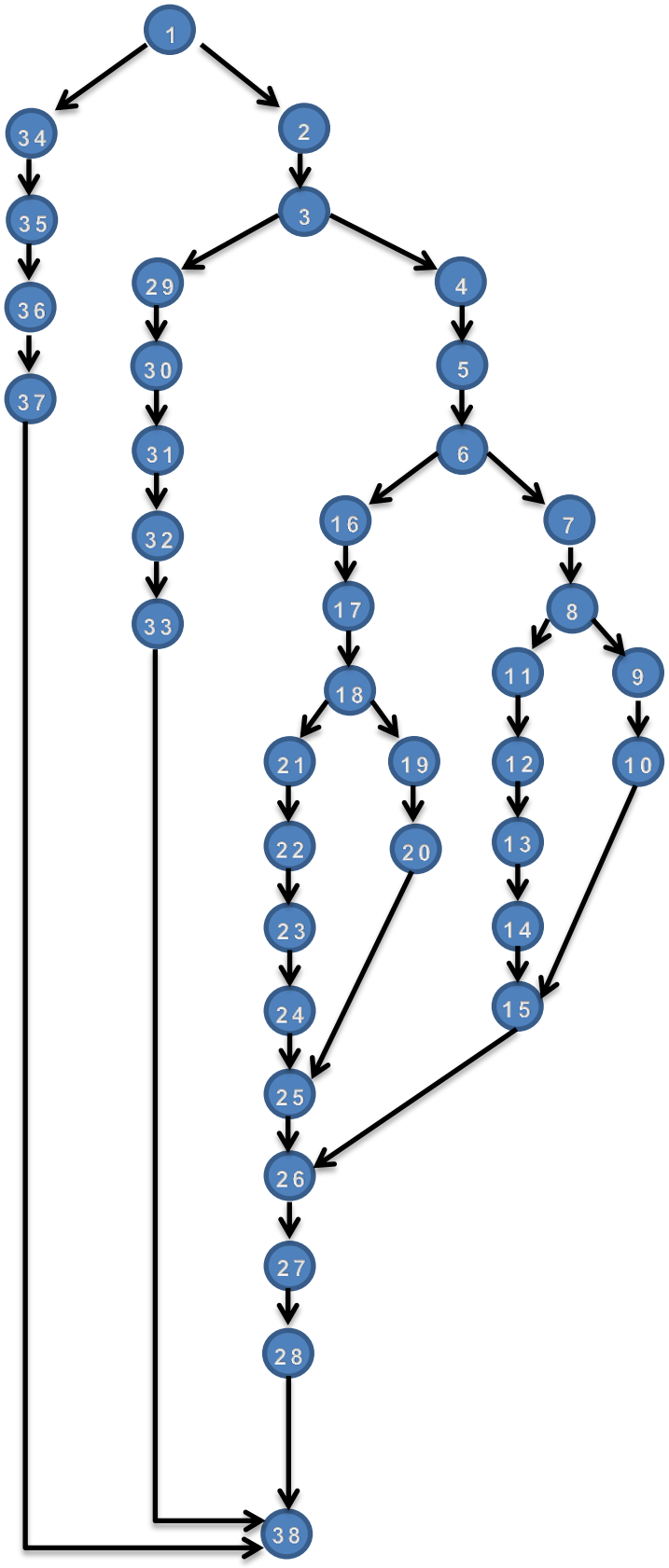


Figura 17: Grafo de flujo. Adicionar Notificación.

Una vez construido el grafo de flujo asociado al procedimiento anterior se determina la complejidad ciclomática, el cálculo es necesario efectuarlo mediante tres vías o fórmulas de manera tal que quede justificado el resultado, siendo el mismo en cada caso:

1. V (G) = (A – N) + 2
Siendo “A” la cantidad total de aristas y “N” la cantidad total de nodos.
V (G) = (42 -38) + 2
V (G) = 6

2. V (G) = P + 1
Siendo “P” la cantidad total de nodos predicados (son los nodos de los cuales parten dos o más aristas).
V (G) = 5 + 1
V (G) = 6

3. V (G) = R
Siendo “R” la cantidad total de regiones, se incluye el área exterior del grafo, contando como una región más.
V (G) = 6

A partir del resultado del cálculo de la complejidad ciclomática se obtiene que 6 es el límite superior de número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez. Además, se construyen los casos de prueba, garantizando que con los parámetros introducidos se recorra cada uno de los nodos del camino. A continuación se presentan los caminos creados:

- Camino 1: 1-2-3-4-5-6-7-8-9-10-15-26-27-28-38.
- Camino 2: 1-2-3-4-5-6-7-8-11-12-13-14-15-26-27-28-38.
- Camino 3: 1-2-3-4-5-6-16-17-18-19-20-25-26-27-28-38.
- Camino 4: 1-2-3-4-5-6-16-17-18-21-22-23-24-25-26-27-28-38.
- Camino 5: 1-2-3-29-30-31-32-33-38.
- Camino 6: 1-34-35-36-37-38.

3.5.2 Pruebas Unitarias

La prueba de unidad o pruebas unitarias son la primera fase de las pruebas dinámicas y se realizan sobre cada módulo del software de manera independiente. El objetivo es comprobar que el módulo, entendido como una unidad funcional de un programa independiente, está correctamente codificado.(32)

Las pruebas unitarias aseguran que un único componente de la aplicación produce una salida correcta para una determinada entrada. Este tipo de pruebas se encargan de un único caso cada vez, lo que significa que un único método puede necesitar varias pruebas unitarias si su funcionamiento varía en función del contexto.

Para la realización de este tipo de prueba se utilizó la biblioteca PHPUnit, la cual se encuentra integrada en el marco de trabajo Symfony 2. Para cada prueba unitaria que se realice, se hace necesario crear una clase PHP en el subdirectorio Tests/ de los paquetes con su nombre terminado en Test.php.

A continuación se muestra un fragmento de las pruebas aplicadas a las funcionalidades pertenecientes a la clase AGRBridge y el resultado arrojado:

```
// src/VU/ServiciosBundle/Controller/NotificacionesApiController.php
namespace VU\ServiciosBundle\Controller;

class AGRBridge
{
    public function adicionarNotificacionAction()
    {
        return $respuesta;
    }
}


// src/VU/ServiciosBundle/Tests/Controller/NotificacionesApiController.php
namespace VU\ServiciosBundle\Tests\Controller;

use VU/ServiciosBundle/Controller/NotificacionesApiController;

class NotificacionesApiControllerTest extends \PHPUnit_Framework_TestCase
{
    public function testAdicionarNotificacionAction()
    {
        $notController = new NotificacionesApiController();
        $result = $notController -> adicionarNotificacionAction();
```

```
// ¡Reporta el un error identificado en el mensaje si la variable result no es del
// tipo esperada!
        $this-> void assertType("string", $result, "Los tipos no coinciden");
    }
}
```

A continuación se muestra una tabla que arroja los resultados obtenidos en la prueba unitaria a la funcionalidad **AdicionarNotificacionAction()**, correspondiente al RF58:

Prueba: Unitaria		
Nombre Prueba: ServiciosBundle_NotificacionesApiController		
Estado: Satisfactoria	**Tipo**: Regresión	**Última ejecución**: 14/05/2013
Ejecutado por: Astroberto Cárdenas Orfila	**Verificado por**: Astroberto Cárdenas Orfila	
Descripción: Se comprueba que la funcionalidad funcione correctamente, realizando las validaciones correspondientes, almacenando el servicio, enviando la solicitud de y retornando un arreglo con la incidencia y la notificación.		
Entrada: Ninguna		
Criterio de aceptación: Retorna los datos que son solicitados. En caso de existir errores, devuelve una incidencia con los errores encontrados.		
Resultado: Retorna los datos que son solicitados.		

Tabla 10: Prueba unitaria perteneciente a la funcionalidad AdicionarNotificacionAction().

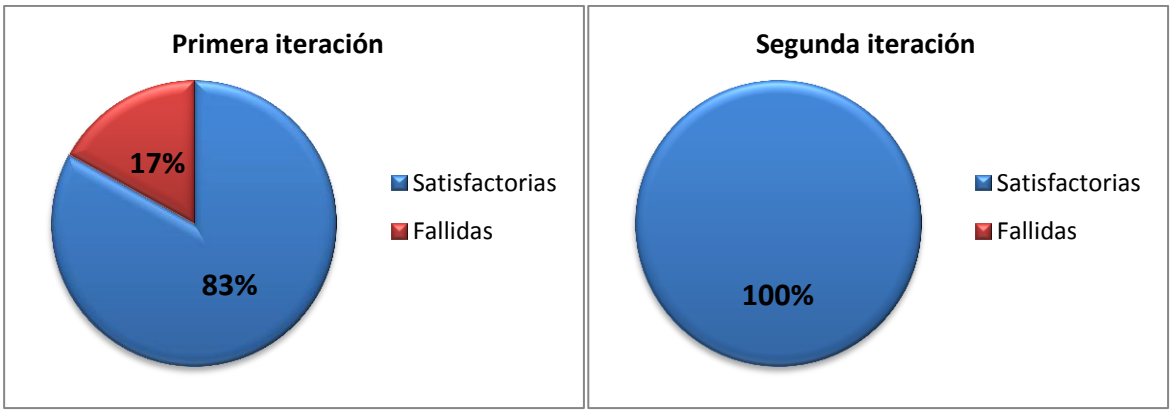


Figura 18: Resultado de las pruebas unitarias.

Como se pudo apreciar en la imagen anterior, se aplicaron dos iteraciones de las pruebas unitarias a las funcionalidades que posee el componente, dando como resultado en la primera iteración un 17% de estado fallido. Los errores detectados fueron corregidos y en la segunda iteración se alcanzó un 100% de las funcionalidades con resultados satisfactorios.

3.5.3 Pruebas de Carga y Estrés

Entre las pruebas que existen para evaluar el desempeño de una aplicación se encuentra la prueba de carga que es utilizada para probar una aplicación bajo cargas normales, para determinar: los tiempos de respuesta de varios procesos críticos de negocio y transaccionales y qué se necesita respecto a hardware, software o configuración. Garantizando que un sistema o aplicación es capaz de funcionar correctamente en el ambiente productivo, bajo la carga que se espera en dicho ambiente.(33)

Otras de las pruebas que se utilizan para evaluar el desempeño es la prueba de estrés que evalúa un sistema con una específica limitación de carga, para que la misma haga fallar el Sistema y se mide cuanto soporta el mismo, porque el objetivo es saber si una situación de estrés puede dar como resultado una falla catastrófica del sistema o una lentitud del mismo.(33)

Para la realización de estas pruebas se utilizó la herramienta JMeter la cuál arrojó los siguientes resultados:

Peticiones	Cantidad de Peticiones	Media	Mediana	Tiempo Mínimo	Tiempo Máximo
/VUC/web/app.php/servicios/	400	3991ms	4405ms	187ms	7234ms
/VUC/web/app.php/servicios/gestionar-subsistema	100	7182ms	8031ms	609ms	9999ms
/VUC/web/app.php/servicios/obtener-subsistemas	300	6132ms	6015ms	4015ms	9812ms
/VUC/web/app.php/servicios/editar-subsistema	200	4398ms	4343ms	3889ms	7327ms
/VUC/web/app.php/servicios/modificar-servicio	100	6922ms	6952ms	5483ms	7795ms

/VUC/web/app.php/servicios/obtener-servicios	200	5754ms	5655ms	5014ms	6795ms
/VUC/web/app.php/servicios/obtener-respuesta	100	5079ms	4983ms	4593ms	6139ms
/VUC/web/app.php/servicios/obtener-parametros	100	5112ms	5062ms	4124ms	5920ms
/VUC/web/app.php/servicios/editar-servicio	100	4399ms	4311ms	3827ms	5326ms
/VUC/web/app.php/servicios/obtener-subsistemas-combo	100	4728ms	4670ms	4077ms	5655ms
/VUC/web/app.php/servicios/mostrar-uddi	100	4867ms	4843ms	4093ms	5779ms
Total	**1800**	**5164ms**	**5046ms**	**187ms**	**9999ms**

Tabla 11: Datos de las pruebas de carga y estrés.

Figura 19: Cantidad de peticiones.

Con la realización de estos dos tipos de prueba se logró validar el RNF de Rendimiento, ya que al obtener un tiempo de respuesta medio de 5.1 segundos en un total de 1800 peticiones en un servidor de 4GB de RAM. Teniendo en cuenta que la aplicación va a ser desplegada utilizando un servidor que posee unas mejores características de hardware se espera que el rendimiento aumente.

3.6 Conclusiones Parciales

En este capítulo, la implementación de la solución estuvo guiada por varios estándares de codificación permitiendo un entendimiento más claro del código generado. Se realizó un tratamiento de errores para evitar problemas en tiempo de ejecución. Con la aplicación de pruebas se comprobó que la solución cumple con los objetivos propuestos.

Conclusiones

El estudio del estado del arte de los estándares que utilizan los servicios web en el mundo, acentuando el análisis en el envío de información entre sistemas, arrojó que debido a las especificaciones del negocio de la VUCEC se debía utilizar SOAP para el intercambio de información ya que este protocolo era el que más se ajustaba a las necesidades del sistema.

A partir de la aplicación del modelo de desarrollo establecido para el CEIGE se definieron los requisitos que recogen las funcionalidades necesarias para garantizar que el componente satisfaga las necesidades del cliente.

Con la implementación se logró obtener un componente que permite la comunicación con el sistema GINA, logrando que los datos que son transmitidos y recepcionados viajen de manera segura. Mediante la aplicación de las pruebas unitarias, de carga y estrés se validó la solución.

La integración del componente de servicios web con el sistema GINA brindará una mayor agilidad y transparencia en los procesos comerciales, una respuesta rápida y efectiva a los importadores y exportadores, así como una mayor facilidad en la gestión de los trámites necesarios para el tráfico de mercancías en territorio nacional.

Al finalizar la investigación se logró cumplir el objetivo trazado de manera satisfactoria, obteniendo como resultado el componente de servicios web de la VUCEC.

Recomendaciones

Se recomienda que:

- Que se cree un plan de contingencia, el que garantice la disponibilidad de los servicios web.
- Que se integren las aplicaciones pertenecientes a la administración central del estado con la VUCEC.

Glosario de Términos

ACI: Información Adelantada de Carga.

AGR: Aduana General de la República.

Artefactos: Un artefacto puede ser un modelo, un elemento de modelo o un documento, en fin todo lo que puede ser generado en el proceso.

CASE: (Computer-Aided Software Engineering**)** Conjunto de herramientas y métodos asociados que proporcionan asistencia automatizada en el proceso de desarrollo del software a lo largo de su ciclo de vida.

CEIGE: Centro de Informatización para la Gestión de Entidades.

Declaración de Mercancías: Manifestación en la forma prescrita por la Aduana, por la que los interesados indican el régimen aduanero que se ha de aplicar a las mercancías y proporcionan los datos que la Aduana exige para la aplicación de este régimen.

Declarante: Toda persona natural o jurídica que hace una declaración en aduana o en nombre de la cual esta Declaración es hecha.

Despacho Comercial: Cumplimiento de las formalidades aduaneras necesarias para exportar, importar o para colocar las mercancías bajo otro régimen aduanero.

DM: Declaración de Mercancías.

EDI: Intercambio Electrónico de Datos.

Gestión: Actividades coordinadas para dirigir y controlar una organización.

GINA: Sistema Integral de Gestión Aduanera.

Información asociada a la Aduana: Toda la información que debe introducir un OCE para realizar una operación comercial ya sea de importación, exportación o un tránsito de mercancías.

Nomenclador: Es un sistema de clasificación y de codificación aplicado para designar conceptos fundamentales tales como países, aranceles, tarifas, monedas, plazos, aduanas, modos de transporte, ministerios, formas de pago, entidades, etc., que facilitan la flexibilidad del sistema y la posibilidad de actualización rápida.

OMI: Organización Marítima Internacional.

Operador de Comercio Exterior: Persona natural o jurídica que, en virtud de la autorización que le ha sido otorgada, administra y opera puerto, instalación marítima, aeropuerto o almacén de depósito bajo control aduanero. Es operador también la persona natural o jurídica autorizada a establecerse en una zona franca para realizar en ella alguna o algunas de las actividades que en ella se desarrollan.

Organización: Conjunto de personas e instalaciones con una disposición determinada de responsabilidades, autoridades y relaciones.

ORM: (Object-Relational Mapping**),** en español (mapeo objeto-relacional), es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional, utilizando un motor de persistencia.

Proceso: Un proceso se define como un conjunto de tareas, actividades o acciones interrelacionadas entre sí que, a partir de una o varias entradas de información, materiales o de salidas de otros procesos, dan lugar a una o varias salidas también de materiales (productos) o información con un valor añadido.

Régimen: Tratamiento aplicable a las mercancías sometidas al control de la Aduana, de acuerdo con la Normativa Aduanera, según la naturaleza y objetivos de la operación.

Sistema: Conjunto de elementos interrelacionados que trabajan juntos para obtener un resultado deseado.

SNA: Servicio Nacional de Aduanas.

TIC: Tecnologías de la Información y las Comunicaciones.

Tiempo de Levante: Tiempo que demora la Aduana en autorizar a los interesados a disponer de una mercancía que ha sido objeto de un despacho.

UCI: Universidad de las Ciencias Informáticas.

Usuario/cliente: Organización o persona que recibe un producto o servicio como resultado de la gestión de información.

VUCE: Ventanilla Única del Comercio Exterior.

XML: (eXtensible Markup Language), es un lenguaje de marcas que deriva del lenguaje SGML y permite definir la gramática de lenguajes específicos para estructurar documentos grandes.

Bibliografía

1. Radiografía del uso de internet en el mundo 2012 «. In: [online]. [Accessed 6 December 2012]. Available from: http://www.fix-si.com/blog/2012/10/radiografia-del-uso-de-internet-en-el-mundo-2012/.

2. Tormenta de Ideas | Lluvia de Ideas. In: [online]. [Accessed 22 April 2013]. Available from: http://www.aiteco.com/tormenta-de-ideas/.

3. Guía Breve de Servicios Web. In: [online]. [Accessed 6 December 2012]. Available from: http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb.

4. Servicios Web con PHP (NuSOAP). In: [online]. [Accessed 11 April 2013]. Available from: http://www.nociondigital.com/webmasters/php-tutorial-servicios-web-con-php-nusoap-detalle-168.html.

5. Extensible Markup Language (XML). In: [online]. [Accessed 6 December 2012]. Available from: http://www.w3.org/XML/.

6. WSDL (Web Services Description Language). In: [online]. [Accessed 6 December 2012]. Available from: http://pic.dhe.ibm.com/infocenter/rsahelp/v8/index.jsp?topic=/org.eclipse.jst.ws.doc.user/concepts/cwsdl.html.

7. SOAP (formerly known as Simple Object Access Protocol). In: [online]. [Accessed 3 April 2013]. Available from: http://pic.dhe.ibm.com/infocenter/rsahelp/v8/index.jsp?topic=/com.ibm.webservice.doc/topics/core/csoap.html.

9. XML-RPC Specification. In: [online]. [Accessed 11 April 2013]. Available from: http://xmlrpc.scripting.com/spec.html.

10. UDDI (Universal Description, Discovery, and Integration). In: [online]. [Accessed 6 December 2012]. Available from: http://pic.dhe.ibm.com/infocenter/rsahelp/v8/index.jsp?topic=/org.eclipse.jst.ws.consumption.ui.doc.user/concepts/cuddi.html.

11. *TodoAgil.Pdf (objeto application/pdf)* [online]. S.l.: s.n. [Accessed 28 January 2013]. Available from: http://www.willydev.net/descargas/prev/TodoAgil.Pdf.

12. *CEIGE-Modelo de Desarrollo de Software v1.1.pdf (objeto application/pdf)* [online]. S.l.: s.n. [Accessed 28 January 2013]. Available from: file:///C:/Users/Astro/Desktop/CEIGE-Modelo%20de%20Desarrollo%20de%20Software%20v1.1.pdf.

13. *doc-modelado-sistemas-uml.pdf (objeto application/pdf)* [online]. S.l.: s.n. [Accessed 28 January 2013]. Available from: http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/doc-modelado-sistemas-uml.pdf.

14. UML, BPMN and Enterprise Architecture Tool for Software Development. In: [online]. [Accessed 28 January 2013]. Available from: http://www.visual-paradigm.com/.

15. Qué es PHP. In: [online]. [Accessed 11 April 2013]. Available from: http://www.desarrolloweb.com/articulos/392.php.

16. MIGUEL ANGEL ALVAREZ. Qué es Javascript. In: [online]. [Accessed 22 January 2013]. Available from: http://www.desarrolloweb.com/articulos/25.php.

17. NetBeans IDE - NetBeans Rich-Client Platform Development (RCP). In: [online]. [Accessed 3 April 2013]. Available from: https://netbeans.org/features/platform/index.html.

18. Servidor Apache HTTP. In: [online]. [Accessed 28 January 2013]. Available from: http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/ch-httpd.html.

19. ¿Qué es Subversion? In: [online]. [Accessed 28 January 2013]. Available from: http://svnbook.red-bean.com/nightly/es/svn-ch-1-sect-1.html.

20. Microsoft Word - Modelo del Dominio.docx - ModeloDominio.pdf. In: [online]. [Accessed 23 April 2012]. Available from: http://is.ls.fi.upm.es/docencia/is2/documentacion/ModeloDominio.pdf.

21. ING. JENNI MANSO MARTÍNEZ. *Procedimiento para la Ingeniería de Requisitos en el Departamento de Desarrollo de Soluciones para la Aduana del CEIGE*. 2010. S.l.: s.n.

22. Requerimientos Funcionales y No Funcionales (RF/RNF). In: [online]. [Accessed 11 April 2013]. Available from: http://ingenieriadesoftware.bligoo.com.mx/requerimientos-funcionales-y-no-funcionales-rf-rnf.

23. ING. JENNI MANSO MARTÍNEZ. *Procedimiento para la Ingeniería de Requisitos en el Departamento de Desarrollo de Soluciones para la Aduana del CEIGE*. S.l.: s.n.

24. MeRinde - Modelo de Diseño *. In: [online]. [Accessed 23 April 2013]. Available from: http://merinde.net/index.php?option=com_content&task=view&id=92.

25. 08-Patrones1.ppt - 08-Patrones.pdf. In: [online]. [Accessed 11 April 2013]. Available from: http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf.

26. Patrones de Diseño: Bridge. In: [online]. [Accessed 11 April 2013]. Available from: http://patronesdediseno-poo.blogspot.com/2009/05/bridge.html.

27. Inyección de dependencias: Guice. In: [online]. [Accessed 11 April 2013]. Available from: http://www.metafisicainformatica.com/2011/05/02/guice/.

28. UML_clase_05_UML_paquetes.pdf. In: [online]. [Accessed 23 April 2013]. Available from: http://www.codecompiling.net/files/slides/UML_clase_05_UML_paquetes.pdf.

29. DRA. ANAISA HERNÁNDEZ GONZÁLEZ. *Un método para el diseño de la base de datos a partir del modelo orientado a objetos.* S.l.: s.n., 2004.

30. ROGER S. PRESSMAN. *Ingenieria del Software - Un Enfoque Práctico.* S.l.: McGraw-Hill Companies, 2002. 8448132149.

31. DPTO ADUANA CEIGE. *Propuesta de un Estándar de Codificación.* 2013. S.l.: s.n.

32. SIRA. JURISTO, NATALIA, MORENO, ANA M. Y VEGAS. *Técnicas de Evaluación de Software.* S.l.: s.n., 2005.

33. Pruebas de desempeño. In: [online]. [Accessed 15 May 2013]. Available from: http://qvision.us/articulos-de-interes/126-pruebas-de-desempeno.html?format=html&lang=es.