Universidad de las Ciencias Informáticas

Facultad 3

Título: Sistema informático de envío y recepción de información entre Ventanilla Única y el GINA

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Eddie Nelson Beltrán González

Tutor: Ing. Julio Cesar Bravo Rodríguez

Junio del 2012

DECLARACIÓN DE AUTORÍA

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

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

Eddie Nelson Beltrán González

_____
Firma del Autor

Ing. Julio Cesar Bravo Rodríguez

_____
Firma del Tutor

DATOS DE CONTACTO

Julio Cesar Bravo Rodríguez: Graduado de Ingeniero en Ciencias Informáticas, con dos años de experiencia laboral.

AGRADECIMIENTOS

A mis padres Ovidia González y Nelson Beltrán por traerme al mundo y por formar parte imprescindible de mi vida

A mis abuelos Nelson e Idolidia por su amor, compresión y dedicación en todo momento, gracias por haber sido guía y ejemplo durante toda mi vida.

A mis hermanas Yaima, Roxana y Rosmery por ser un preciado tesoro en mi vida. A Vilma, mi otra hermana que aunque no lo es de sangre para mí no tiene diferencia alguna

A mi familia en general por formar parte fundamental de mi vida.

A mi otra familia, mis amigos que siempre me han apoyado, Julio, Remberto, Rolando, David.

A mis amigos de la UCI que me ha acompañado durante estos 5 años, en especial para Leonardo, Reinier, Miguel, Yosleianis,Maylevis, Félix Rdguez, y todos los demás que se me quedan en estas líneas por ser tantos pero van en mi corazón en todo momento.

A Fidel, a la Revolución y al pueblo de Cuba, por darme la oportunidad de estudiar y formarme como profesional.

A todos los que de una forma y otra me han apoyado, muchas gracias.

DEDICATORIA

Dedico el presente trabajo de diploma a mis padres y a mi familia, en especial a mis abuelos Nelson e Idolidia por todo su amor, sus sacrificios, su cariño y su confianza. Fueron la fuente de inspiración para la realización de este trabajo.

A Fidel y a la Revolución.

A todos mis compañeros y amigos.

A la Universidad de las Ciencias Informáticas.

RESUMEN

Con el objetivo de dar respuesta al alto nivel de operaciones comerciales que se llevan a cabo de manera presencial ante el sistema de Gestión Integral de Aduanas (GINA) y reducir el tiempo de obtención de los avales para realizar estas operaciones comerciales, surge la Ventanilla Única para el Comercio Exterior (VUCE). Esta consiste en un sistema que posibilite tener un único punto de entrada la información digital y estandarizada para las operaciones comerciales controladas bajo regímenes aduaneros.

La VUCE en su etapa actual de desarrollo no cuenta con un medio para establecer el intercambio de información con sistemas externos, haciéndose necesaria una solución informática que sea la encargada de dar respuesta a dicha necesidad. En una primera etapa el intercambio de información está concebido solo con el sistema GINA, teniendo en cuenta la incorporación de otros sistemas en el futuro.

Para dar solución a la situación planteada anteriormente se propone como objetivo general desarrollar una solución informática para el envío y recepción de información entre la VUCE y el GINA, de manera que se establezca una prioridad a los documentos, para garantizar la interoperabilidad entre ambos sistemas.

Como resultado de la investigación, haciendo uso del Modelo de Desarrollo del Departamento de Soluciones para la Aduana, se obtuvo una solución informática que gestiona el envío y recepción de información entre la VUCE y el GINA. Esta solución soporta la incorporación de otros sistemas en el futuro, haciendo uso de las ventajas que provee la utilización de los servicios web como tecnología principal para dar soporte a la solución desarrollada, aplicando estándares que la hacen más competente. De esta forma la VUCE logrará cumplir su objetivo, elevando de esta forma la competitividad del comercio cubano ante el mundo.

PALABRAS CLAVES:

GINA, interoperabilidad, VUCE.

TABLA DE CONTENIDOS

INTRODUCCIÓN

Desde hace varios años el comercio en el mundo ha crecido dos veces más rápido que el producto interno bruto mundial. Con gran visión en este contexto, las naciones que presten mayor atención a este aspecto, logren mayor atracción en las inversiones extranjeras e impulsen el comercio exterior, obtendrán mayor crecimiento económico. Para obtener un mayor intercambio comercial con otras naciones no basta con la reducción o eliminación de aranceles, es necesario implementar sistemas de modernización aduanera que permitan aumentar la competitividad internacional y reducir costos al mismo tiempo. Los países que presentan las mejores prácticas en este campo han adoptado el uso del sistema de Ventanilla Única para el Comercio Exterior (VUCE) como vía indispensable para agilizar los procesos del comercio exterior. (1)

La VUCE, es un sistema que posibilita tener un único punto de entrada de información digital y estandarizada para las operaciones comerciales controladas bajo regímenes aduaneros, posibilitando que las personas jurídicas o no, que intervienen en la transacción comercial obtengan datos solicitados a los organismos o entidades emisoras de permisos, simplificando los trámites a realizar y por consiguiente disminuyendo el tiempo de despacho de mercancía así como la estancia de buques y aeronaves en el país, permitiendo la entrada de información una única vez para todos los trámites a realizar durante la transacción comercial y disminuir el intercambio de información en copia dura.

La Aduana General de la República (AGR), órgano encargado del control en la frontera y de fiscalización en la actividad vinculada al comercio exterior, cuenta con un sistema de Gestión Integral de Aduanas (GINA) que está concebido para garantizar el funcionamiento rápido y efectivo de los diferentes procesos que se desarrollan en cada una de las entidades aduaneras de Cuba, siendo este diseño aplicable a cualquier lugar del mundo debido a que su implementación fue basada en las definiciones del Convenio de Kyoto del año 2001, convenio en el que plasman las directivas a cumplir en aras de informatizar el despacho de mercancías en frontera.

Hoy en día el comercio exterior en Cuba involucra un alto nivel de operaciones comerciales y personas en las mismas. En la realización de los trámites necesarios para efectuar una operación comercial las personas relacionadas deben acudir a varias instituciones, organismos y entidades solicitando permisos y/o liberaciones que le avalen la operación frente a las autoridades comerciales del país, hoy estas solicitudes se realizan de forma personal y con el inconveniente de que se debe presentar físicamente en

la institución a la que se solicita el aval, provocando la pérdida de tiempo y por consiguiente el gasto de divisas al país, ya que provoca la demora de los movimientos mercantiles en puertos, aeropuertos y depósitos, en detrimento de la agilidad y respuesta de las autoridades aduanales en el país. Estos trámites en la aduana general de la república de Cuba se realizan en el GINA, por lo que trae como consecuencia que la información que se maneja en esta institución afronte posibles riesgos de seguridad.

Por esto surge la idea de una ventanilla única para el comercio exterior de Cuba que posibilite la independencia de los involucrados externos a la aduana con el sistema GINA y que agrupe en un único punto la gestión de los trámites entre las diferentes entidades que convergen en los intercambios comerciales del país. Este sistema al convertirse en el centro de la entrada de toda información para trámites del comercio exterior tendrá un intercambio directo con los sistemas informáticos pertenecientes a las diferentes entidades emisoras de avales para este tipo de operaciones en el país, especialmente el sistema GINA al ser el principal sistema al que se envía información y en el que se procesan y avalan las operaciones en una etapa final.

La rapidez con la que estos sistemas son capaces de dar respuesta a las solicitudes de los clientes depende de varios factores, dentro de los que se encuentra uno muy importante que es la eficiencia con que se logre realizar la comunicación entre estos y la VUCE, repercutiendo directamente en la satisfacción de los clientes, la realización en tiempo de los procesos y por consiguiente en los ingresos al país. La VUCE en su etapa actual de desarrollo no cuenta con un medio para establecer intercambio de información con sistemas externos haciéndose necesaria una solución informática que sea la encargada de dar respuesta a dicha necesidad. En una primera etapa el intercambio de información está concebido solo con el sistema GINA y teniendo en cuenta la incorporación de otros sistemas en un futuro.

La urgencia con la que es necesario procesar un documento es dependiente de varios factores, por lo que si no son enviados por una prioridad definida de acuerdo a estos factores se corre el riesgo de que por alguna caída de la conexión entre los sistemas por algún tiempo o una gran concurrencia de entrada de información al sistema, existan documentos que no lleguen a su destino con tiempo suficiente o fuera del plazo válido, incurriendo en una operación comercial que no podrá culminarse y repercutiendo en pérdidas de ingresos para el país.

Lo anteriormente analizado arroja como problema a resolver: ¿Cómo enviar y recibir la información entre la VUCE y el GINA de manera que se establezca una prioridad a los documentos, para garantizar la interoperabilidad entre ambos sistemas?

La investigación tiene como objeto de estudio el intercambio electrónico de datos EDI.

Para dar solución al problema anteriormente planteado se propone como objetivo general: desarrollar una solución informática para el envío y recepción de información entre la VUCE y el GINA, de manera que se establezca una prioridad a los documentos, para garantizar la interoperabilidad entre ambos sistemas.

El campo de acción se enmarca en el intercambio electrónico de datos en las VUCE.

Con el objetivo de dar solución al objetivo general se plantean las siguientes tareas de la investigación a cumplir:

1. Realizar un estudio de los sistemas de Ventanilla Única para identificar posibles fortalezas para la solución.
2. Realizar Diagramas de modelo de negocio.
3. Realizar Descripción de los procesos de negocio.
4. Realizar Modelo Conceptual.
5. Realizar Descripción de los requisitos.
6. Describir reglas de negocio.
7. Diseñar Modelos de datos.
8. Diseñar la solución descrita.
9. Implementar la solución modelada.
10. Verificar la solución mediante pruebas funcionales.
11. Validar la solución por el cliente.

Durante el desarrollo de este trabajo son utilizados algunos métodos científicos de la investigación que permiten guiar exitosamente el desarrollo de la solución.

Histórico-lógico: con el objetivo de realizar el estudio del estado del arte del tema a investigar basado en un estudio de algunos de los sistemas VUCE existentes. De esta manera se puede obtener conocimiento

acerca de la existencia y características de estos, recopilando los elementos similares a la solución que se quiere obtener.

Analítico-sintético: con el objetivo de analizar teorías y documentos, y a partir de ello realizar un estudio acerca de las características y las necesidades vigentes para la realización de la solución que se persigue, determinando de esta manera los elementos que mejor definirían una buena solución.

Modelación: con el objetivo de realizar abstracciones para dar una explicación de la realidad existente. Su condición principal es la relación entre el modelo y el objeto que se modela, que en este caso sería el sistema en cuestión.

El presente trabajo se encuentra estructurado en tres capítulos, los cuales se describen a continuación:

Capítulo 1: Abarca lo relacionado con la Fundamentación Teórica de la investigación. Se enuncian los principales conceptos que se abordan en la investigación. Se realiza además un estudio de las soluciones al problema planteado ya existente, así como de las herramientas, metodologías y lenguajes a usar en el desarrollo de la solución. Todos estos elementos dan cabida a la inicialización del próximo capítulo.

Capítulo 2: En este capítulo se abordan el Análisis y el Diseño de la solución, describiendo a su vez varios artefactos que se generan durante estas fases. Se describe además la definición de los requisitos del sistema, los diagramas de secuencia, de paquetes, de clases del negocio, entre otros elementos.

Capítulo 3: En este capítulo se abordan los elementos de la implementación y validación de la solución propuesta; teniendo como base los artefactos concebidos en las etapas de análisis y diseño para la implementación de la solución y posteriormente haciendo uso de las métricas definidas por CALISOFT en la Universidad de las Ciencias Informáticas para conocer si se logró cumplir con los objetivos formulados.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se presentan los principales elementos que conforman la fundamentación de la presente investigación, se realiza un estudio de los sistemas de VUCE existentes en el mundo con el objetivo de estudiar los métodos de intercambio de datos con sistemas externos, así como las diferentes técnicas de integración de datos, con el objetivo de hallar fortalezas que aporten elementos sólidos para conformar la solución.

1.2 La Aduana General de la República

En Cuba, la Aduana constituye un órgano de control en la frontera y de fiscalización en la actividad vinculada al comercio exterior.

Entre sus misiones está garantizar la seguridad y protección de la sociedad socialista y de la economía nacional, así como la recaudación fiscal y la emisión de las estadísticas del comercio exterior, a través del cumplimiento de las políticas estatales de competencia aduanera para el tráfico internacional de viajeros, mercancías y medios de transporte. (1) En los últimos años es una tendencia el uso de los sistemas de VUCE para aumentar la competitividad en el sector del comercio exterior, es por ello que la implantación de este sistema es un aporte fundamental para lograr un crecimiento de este.

1.3 ¿Qué es VUCE?

La Ventanilla Única para el Comercio Exterior (VUCE) se define como un mecanismo de facilitación que permite a las partes involucradas, en el comercio y el transporte, alojar información estandarizada y documentos en un solo punto de entrada para cumplir con todos los trámites de importación, exportación y tránsito. La información al ser electrónica, debe ser remitida una sola vez.

La VUCE se conceptualiza como "un sistema integrado que permite a las partes involucradas en el comercio exterior y transporte internacional gestionar, a través de medios electrónicos, los trámites requeridos por las entidades competentes de acuerdo con la normatividad vigente, o solicitados por dichas partes, para el tránsito, ingreso o salida del territorio nacional de mercancías". (2)

1.4 Sistemas de VUCE en el mundo

En el mundo ha surgido desde hace varios años una gran tendencia al uso de las VUCE como estrategia para integrar todas las entidades que intervienen en el proceso del comercio de las naciones, con el

objetivo de facilitar y agilizar todos los procesos del comercio exterior. Con el objetivo de obtener información y analizar las características de estos sistemas que sirvan de apoyo en la construcción de la solución del presente trabajo se refiere a continuación un estudio de los sistemas VUCE existentes centrando la atención en el intercambio de datos e integración entre sistemas.

Diversas naciones son las que han implantado la VUCE como pilar para agilizar los procesos del comercio exterior, existen diversas pautas que definen como llevar a cabo la construcción de un sistema VUCE, incluso la Comisión Económica para Europa de las Naciones Unidas (ONU/CEPE) estableció una serie de consejos de cómo y qué debe poseer un sistema de este tipo para llegar a ser competente en su funcionamiento, basándose en la experiencia de los países que más tiempo llevan en este sector. Para la creación de estos sistemas se proponen la creación de varios componentes que serían el compuesto que define todo el funcionamiento.

Las VUCE implantadas por diversos países no presentan información referente a la comunicación de estas con otros sistemas, dígase tecnología y pautas que emplean para establecer un intercambio de datos con otros sistemas externos con los cuales es necesario mantener una activa comunicación. Esto hace que en este sector se haga difícil obtener información de buenas pautas para la elaboración de un sistema VUCE propio. Todo esto trae como consecuencia que se lleve a cabo un estudio de las tecnologías existentes que permitan establecer una solución informática que brinde rapidez y seguridad en la comunicación entre sistemas, con el objetivo de establecer una selección de las mejores prácticas y tecnologías para la elaboración de la solución.

Un sector que presenta grandes facilidades en este sentido son las técnicas de integración de datos, las cuales brindan una serie de tecnologías y métodos que pueden proveer los elementos necesarios para la definición de la solución buscada.

1.5 Técnicas de Integración de Datos

El esquema tradicional cliente/servidor permite que diferentes módulos de aplicación se comuniquen directamente sin una capa intermedia. El problema se presenta en sistemas complejos con componentes de diversos proveedores donde resulta poco flexible e inoperante la comunicación.

Se puede definir la integración con la entrega de Información:

- Precisa

- Consistente
- Oportuna
- Coherente (3)

La integración puede ser enfocada de varias maneras diferentes dependiendo de la idea de "Integración" que se posea. Fundamentalmente existen cuatro:

1. Replicación de Datos.
2. Extracción, Transformación y Carga de Datos o ETL[1]
3. Integración de Información Empresarial o EII[2]
4. Integración de Aplicaciones Empresariales o EAI[3]

A continuación se refieren datos de las mismas, los cuales serán analizados para obtener fortalezas que aporten elementos precisos a la solución que se quiere obtener.

1.5.1 Replicación de Datos

La Replicación de Bases de Datos es una técnica de integración que consiste en la creación y mantenimiento de múltiples instancias (copias) de una misma base de datos. En la mayoría de las implementaciones de replicación esto consiste en un servidor primario que mantiene la instancia primaria de la Base de Datos y otros servidores adicionales que mantienen las copias denominadas esclavas de la misma. Esto brinda varios aspectos beneficiosos:

- La lectura de los datos está distribuida entre todos los servidores de Base de Datos, lo cual brinda una gran ventaja en cuanto al uso compartido de carga.
- La información almacenada es escrita primeramente en el servidor primario y luego es replicado en las copias esclavas.

La transferencia se realiza de Base de Datos a Base de Datos (este es el estilo más antiguo de integración), lo cual es una desventaja para la resolución del problema a resolver dado la heterogeneidad de las fuentes. (3)

[1] **ETL**: Del inglés Extract, Transform and Load.
[2] **EII:** Del inglés Enterprise Information Integration.
[3] **EAI:** Del inglés Enterprise Application Integration.

1.5.2 Extracción, Transportación y Carga de Datos (ETL)

La técnica Extracción Transportación y Carga de Datos (Extract, Transform and Load), como su nombre refleja extrae información de un sistema fuente, los transforma para satisfacer los requisitos del negocio y carga el resultado en un sistema destino. Tanto la fuente como el destino son generalmente Bases de Datos y archivos.

La transformación puede implicar la reestructuración y reconciliación del registro de datos, limpieza del contenido de datos (es decir, revisados por si existen discrepancias y eliminación de datos obviamente falsos) y/o agregación del contenido de datos. Esto sucede mediante una serie de procedimientos especiales que permiten obtener un formato unificado común y mejorado. Sólo después de la revisión y unificación de los datos estos son cargados. Esta técnica se encarga de la integración de datos, no de aplicaciones que es el objetivo trazado en la presente investigación, y obtiene los datos directamente de las Bases de Datos incurriendo en la desventaja planteada en el caso anterior. (3)

1.5.3 Integración de Información Empresarial (EII)

La Integración de Información Empresarial EII es otra de las técnicas de integración existentes. Es un mecanismo de transformación y acceso a datos transparente y optimizado para suministrar una única interfaz a lo largo de los datos de las organizaciones. Dicha interfaz permite acceder a los datos y se obtiene como resultado con este método un Sistema de Información Heterogéneo Distribuido, Virtualmente Integrado. Este tipo de solución consiste en crear un intermediario que contenga los directorios de la Base de Datos y que a su vez sirva de canal de consulta y representación de la información recuperada. La información es capturada en tiempo real lo que implica que las fuentes de datos tengan una estructura tecnológica sólida y bien establecida. (4)

EII protege a las aplicaciones de la complejidad de recuperar datos de múltiples localizaciones, donde los datos pueden diferir en semántica y formato, y emplear diferentes interfaces de datos.

Teniendo en cuenta estos aspectos, para la integración de Datos a Tiempo Real la técnica EII constituye una buena alternativa, sin embargo no es factible para la integración de aplicaciones.

1.5.4 Integración de Aplicaciones Empresariales (EAI)

EAI es el proceso de integrar múltiples aplicaciones desarrolladas independientemente, que utilizan tecnología incompatible y que son gestionadas de forma independiente, permitiendo que se comuniquen e

intercambien transacciones de negocio, mensajes, y datos entre sí. Uno de los principales objetivos de EAI es proporcionar acceso transparente a la amplia gama de aplicaciones que existen en una organización. Las características más importantes de esta tecnología es que se utiliza para la integración de (aplicaciones con aplicaciones) y proporciona un enfoque de integración orientado a proceso basado en mensajes XML. (3) Generalmente es utilizado para el procesamiento de transacciones de negocio operacional en tiempo real.

La dirección de la industria de la EAI es hacia el uso de un bus de servicios empresariales (ESB[4]) que soporta la interconexión de las aplicaciones heredadas y envasados, así como los servicios web que forman parte de una arquitectura orientada a servicios (SOA[5]).

Beneficios que ofrece:

- Integración no invasiva.
- El contenido de los datos (el mensaje) es irrelevante.
- Puede generalmente acceder a muchos puntos finales.
- El contenido del mensaje podría ser XML, Servicios Web, datos planos, etc.
- Los puntos finales pueden ser colas, protocolos, Servicios Web, Bases de Datos, adaptadores de aplicaciones, entre otros.

El sistema VUCE necesita establecer comunicación (intercambio de datos) con el sistema GINA, teniendo en cuenta que:

- En un futuro se integrarán a la par de este otros sistemas pertenecientes a las entidades emisoras de permisos del resto del país.
- No se tiene definido el lenguaje de programación, el sistema operativo ni la plataformade desarrollo de los mismos.

EAI cumple con los requisitos necesarios para ser la solución propuesta, proveyendo de esta forma un medio válido de integración entre los sistemas.

Con el objetivo de indagar en los elementos manejados por los servicios web como centro de atención para el intercambio de información, se relacionan en el apígrafe a continuación algunas de las características que poseen los mismos, así como tecnologías y estándares relacionados. Teniendo en

[4]**ESB:** Del inglés, Enterprise Service Bus.
[5]**SOA:** Del inglés, Service Oriented Architecture.

cuenta que el objetivo trazado con este estudio no es la implantación de un ESB propio sino la identificación de elementos que permitan lograr los objetivos trazados.

1.6 Servicios Web

Existen múltiples definiciones sobre lo que son los Servicios Web, lo que muestra su complejidad a la hora de dar una adecuada definición que englobe todo lo que son e implican. Una posible sería hablar de ellos como un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web. Entre los problemas que resuelven se encuentran:

- Independencia de la Plataforma Hardware.
- Independencia del Sistema Operativo.
- Independencia del Lenguaje de Programación.

Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones o componentes de aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar. (5)

Un Servicio Web es un contenedor que encapsula funciones específicas y permite que estas puedan ser utilizadas en otros servidores. Los mismos se actualizan de forma transparente para el programador y los usuarios, mediante ellos se pueden desarrollar funciones complejas.

En el intercambio de servicios existen dos elementos de colaboración indispensables:

Consumidor de servicio: Es una aplicación, un módulo de software u otro servicio que requiere un servicio. Es la entidad de software que realiza una petición al proveedor de servicios. Tradicionalmente llamado “cliente”. Inicia la búsqueda en el registro, enlaza con el servicio a través del transporte y ejecuta la función del servicio de acuerdo con las reglas establecidas. (3)

Proveedor de servicios: La entidad de software que implementa la especificación del servicio. El proveedor de servicios es una entidad a la que se puede acceder a través de la red y que acepta y ejecuta

peticiones de los consumidores. Publica las interfaces de los servicios en el registro de servicios para que los consumidores puedan descubrirlos y puedan acceder a ellos. (3)

Ventajas que presentan los Servicios Web:

- Están basados en XML, siendo este un lenguaje abierto.
- Son auto-descriptivos.
- Pueden buscar registros de otros Servicios Web.
- Son programables.

La interoperabilidad se consigue a través de la adopción de estándares abiertos. Las organizaciones OASIS[6]y W3C[7]son los comités responsables de la arquitectura y reglamentación de los servicios Web. Para mejorar la interoperabilidad entre distintas implementaciones de Servicios Web se ha creado el organismo WS-I[8], encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares. A los Servicios Web hay una serie de elementos asociados: (3)

- **UDDI:** Descubrimiento.
- **WSDL:** Descripción.
- **SOAP:** Formato de Mensaje.
- **XML:** Codificación.
- **HTTP, SMTP:** Transporte.

- **UDDI (Universal Description, Discovery and Integration):**

Es donde se registran los Servicios Web para que los futuros usuarios los encuentren fácilmente, hace posible que empresas puedan tanto publicar como encontrar Servicios Web. Una vez descritos los negocios y los tipos de servicios que proporcionan se pueden registrar y publicar en un Registro UDDI. Tales negocios publicados pueden ser buscados, consultados o "descubiertos" por otros negocios utilizando mensajes con SOAP[9].

Permite fácilmente buscar una empresa que ofrece los servicios que necesita, leer acerca del servicio ofrecido y ponerse en contacto con una persona para solicitar más información

[6]**OASIS:** Del inglés, Organization for the Advancement of Structured Information Standards.
[7]**W3C:** Del inglés, World Wide Web Consortium.
[8]**WS-I:** Del inglés, Web Services Interoperability Organization.
[9]**SOAP:** Del inglés, Simple Object Access Protocol.

- **Lenguaje de descripción de servicios web (WSDL):**

WSDL (Web Services Description Language) es un formato XML para describir servicios de red como un conjunto de variables que operan en los mensajes que contengan cualquiera de la información orientada a documentos u orientada a procedimiento. Las operaciones y los mensajes se describen de manera abstracta a continuación, unido a un protocolo de red concreto y formato de los mensajes para definir un punto final. Criterios de valoración concretos relacionados se combinan en puntos finales abstractos (servicios). WSDL es extensible para permitir la descripción de los puntos finales y sus mensajes sin importar que formatos de mensaje o protocolos de red son utilizados para comunicarse. (6)

Un documento WSDL define los servicios como colecciones de puntos finales de red, o en los puertos. En WSDL, la definición abstracta de puntos finales y los mensajes se separa de su despliegue de red de hormigón o los enlaces de datos de formato. Esto permite la reutilización de definiciones abstractas: mensajes, que son descripciones abstractas de los datos que se intercambian, y tipos de puertos que son colecciones abstractas de operaciones. El protocolo concreto y especificaciones de formato de datos para un tipo de puerto en particular, constituye un enlace reutilizable. Un puerto se define mediante la asociación de una dirección de red con un enlace reutilizable, y una colección de puertos define un servicio. Por lo tanto, un documento WSDL utiliza los siguientes elementos en la definición de los servicios de red:

- **Tipos:** un contenedor para definiciones de tipos de datos utilizando un sistema de tipos (como por ejemplo XSD).
- **Mensaje:** una definición abstracta, con tipo de que los datos sean comunicados.
- **Operación:** una descripción abstracta de una acción apoyada por el servicio.
- **Tipo de puerto:** un conjunto abstracto de operaciones compatibles con uno o más extremos.
- **Unión:** un protocolo concreto y especificación de formato de datos para un tipo de puerto en particular.
- **Puerto:** un único extremo definido como una combinación de un enlace y una dirección de red.
- **Servicio:** una colección de extremos relacionados.

Es importante observar que WSDL no introduce un lenguaje de definición de tipo nuevo. WSDL reconoce la necesidad de sistemas de tipos ricos para la descripción de formatos de mensaje, y es compatible con la especificación de esquemas XML (XSD) como su sistema de tipo canónico. Sin embargo, ya que es razonable esperar un sistema de gramática de tipo simple para describir todos los formatos de los

mensajes presentes y futuros, WSDL permite el uso de otros idiomas a través de la definición de tipo de ampliación.

Además, WSDL define un mecanismo de enlace común. Esto se utiliza para conectar un formato específico de protocolo, de datos o la estructura de un mensaje abstracto, operación, o de punto final. Permite la reutilización de definiciones abstractas.

En adición al servicio básico de la definición del marco de trabajo, esta especificación introduce extensiones específicas de unión para los siguientes protocolos y formatos de mensaje:

- SOAP 1.1
- HTTP GET / POST
- MIME

Nada impide el uso de otras extensiones de unión con WSDL. (6)

- **SOAP (Simple Object Access Protocol):**

Protocolo Simple de Acceso a Objetos, es un protocolo basado en XML que permite la interacción entre varios sistemas, posee la capacidad de transmisión de información compleja donde los datos pueden ser tranmitidos por diferentes protocolos como pueden ser HTTP, SMTP, entre otros. SOAP no define un medio de trasporte de los mensajes, en sí especifica el formato de los mismos. Proporciona un mecanismo estándar de empaquetar un mensaje. Un mensaje SOAP se compone de un sobre que contiene el cuerpo del mensaje y cualquier información de cabecera que se utiliza para describir le mensaje. Ver **figura 1.1** para mejor entendimiento.

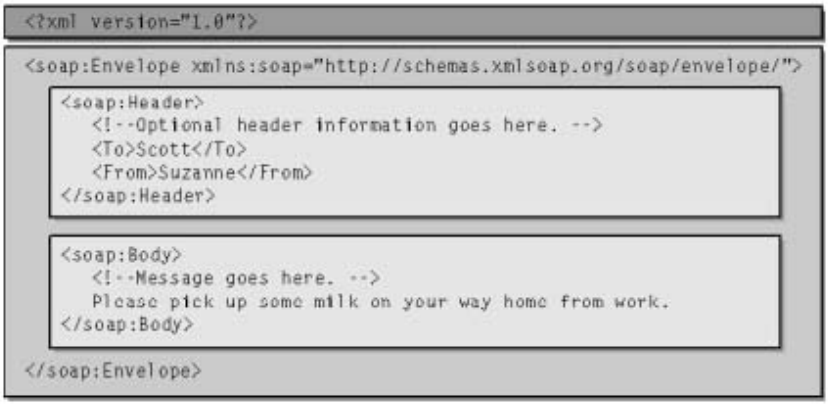


Fig. 1.1: Anatomía de un mensaje SOAP. Tomado de (7).

Algunas de las ventajas de SOAP son (7):

No está asociado con ningún lenguaje: SOAP no especifica una API, por lo que la implementación de la API se deja al lenguaje de programación, como en Java, y la plataforma como Microsoft .Net.

No se encuentra fuertemente asociado a ningún protocolo de transporte: La especificación de SOAP no describe como se deberían asociar los mensajes de SOAP con HTTP. Un mensaje de SOAP no es más que un documento XML, por lo que puede transportarse utilizando cualquier protocolo capaz de transmitir texto.

No está atado a ninguna infraestructura de objeto distribuido: La mayoría de los sistemas de objetos distribuidos se pueden extender, y ya lo están algunos de ellos para que admitan SOAP.

Aprovecha los estándares existentes en la industria: Los principales contribuyentes a la especificación SOAP evitaron, intencionadamente, reinventar las cosas. Optaron por extender los estándares existentes para que coincidieran con sus necesidades. Por ejemplo, SOAP aprovecha XML para la codificación de los mensajes, en lugar de utilizar su propio sistema de tipo que ya están definidas en la especificación esquema de XML. Y como ya se ha mencionado SOAP no define un medio de trasporte de los mensajes; los mensajes de SOAP se pueden asociar a los protocolos de transporte existentes como HTTP y SMTP.

Permite la interoperabilidad entre múltiples entornos: SOAP se desarrolló sobre los estándares existentes de la industria, por lo que las aplicaciones que se ejecuten en plataformas con dicho estándares pueden comunicarse mediante mensaje SOAP con aplicaciones que se ejecuten en otras plataformas. Por ejemplo, una aplicación de escritorio que se ejecute en una PC puede comunicarse con una aplicación del back-end[10]de alguna aplicación ejecutándose en un mainframe (marco principal) capaz de enviar y recibir XML sobre HTTP.

- **Extensible Markup Language (XML):**

Es un simple y muy sencillo formato de texto derivado de SGML (ISO 8879). Originalmente diseñado para satisfacer los desafíos de la gran escala de la publicación electrónica, XML 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. (8)

[10] **back-end:** Es la parte que procesa la entrada desde el front-end, que es la parte del software que interactúa con el o los usuarios.

XML es un metalenguaje extensible de etiquetas que permite definir la gramática de lenguajes específicos, se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Es un lenguaje con una importante función en el proceso de intercambio, estructuración y envío de datos en la Web.

Describe los datos de tal manera que es posible estructurarlos utilizando para ello etiquetas. Las características especiales son la independencia de datos, o de la separación de los contenidos de su presentación. Es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

Como principales ventajas del uso de XML se tienen:

Es extensible: Después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna.
El analizador es un componente estándar: No es necesario crear un analizador específico para cada versión de lenguaje XML. Esto posibilita el empleo de cualquiera de los analizadores disponibles. De esta manera se evitan errores y se acelera el desarrollo de aplicaciones.
Mejora la compatibilidad entre aplicaciones: Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarla. (9)

- **Hyper Text Transfer Protocol (HTTP):**

Protocolo de Transferencia de Hyper Texto, es el protocolo usado en las transacciones de la web. Fue desarrollado por la World Wide Web Consortium y la Internet Engineering Task Force. Este es un protocolo que sigue una orientación a transacciones, siguiendo un esquema petición-respuesta entre el cliente y el servidor.

Protocolo de Transferencia de Hyper Texto, es usado en cada transacción de la Web (WWW[11]), define la sintaxis y la semántica que utilizan los elementos software de la arquitectura web (clientes, servidores,

[11] **WWW:** Del inglés, World Wide Web.

proxies) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

La información transmitida mediante este protocolo se le llama recurso y se le identifica mediante un url. Los recursos pueden ser: archivos, el resultado de la ejecución de un programa, una consulta a una base de datos o la traducción automática de un documento, entre otros. (9)

HTTPS es la versión segura de este protocolo, utiliza un cifrado para asegurar el tráfico de Internet para protegerlo de husmear o manipulación por terceros en la red. HTTPS utiliza ya sea la capa de sockets seguros (SSL) o los Transport Layer Security (TLS) para proteger los datos. Es por esto que su uso es requerido cuando se necesita seguridad en los datos que son transmitidos por el mismo, siendo una buena opción a tener en cuenta para aumentar la seguridad del intercambio de datos, aspecto tratado en el próximo epígrafe.

1.7 Seguridad de la información

La seguridad de los datos en las aplicaciones que intercambian información por la web se hace un factor de vital importancia, el uso de protocolos que no brinden seguridad a los datos que por ellos se transmiten se hace cada día más riesgoso. Dada la necesidad de asegurar los datos que son enviados por estos protocolos es que surgen los protocolos seguros, los cuales haciendo uso de algoritmos de encriptado y certificados hacen que los datos viajen seguros, no permitiendo que terceros accedan obtengan su contenido.

La WS-I ha estandarizado el uso de los servicios web, llegando a definir entre sus estandares WS-Security. Este estándar describe las mejoras de la mensajería SOAP para proporcionar la calidad de la protección a través de la integridad del mensaje, la confidencialidad del mensaje y la autenticación de un solo mensaje. Estos mecanismos pueden ser utilizados para dar cabida a una amplia variedad de modelos y tecnologías de seguridad de cifrado. WS-Security también proporciona un mecanismo de propósito general para la asociación de los tokens de seguridad con los mensajes. Ningún tipo específico de token de seguridad es requerido por WS-Security. Está diseñado para ser extensible (por ejemplo, soporte a múltiples formatos de token de seguridad). Por ejemplo, un cliente podría proporcionar una prueba de identidad y prueba de que tienen una certificación de negocio en particular."

En el presente trabajo no se implementa la seguridad establecida por WS-Security, se hace uso de una variante propia descrita en el próximo epígrafe, que debido al nivel menos complejo de implementar permitirá un desarrollo más ágil de la solución que se desea obtener, no obstante queda como una meta a conseguir en futuras versiones la implantación de este estándar de seguridad que permitirá obtener resultados más competentes y cumplir con los estándares establecidos por la WS-I.

1.7.1 Estrategia de seguridad

En la imagen que se presenta a continuación (**Figura. 1.2**) se representa la razón por la cual el uso de protocolos seguros no satisface la necesidad de seguridad que se quiere obtener. En la parte superior, se muestra cómo una seguridad a nivel de protocolo, tal como https podría asegurar la aplicación. En el momento en que el mensaje se entrega, el texto en claro del mensaje es expuesto. En este momento un "intruso" puede echar un vistazo en el mensaje, esto pudiese ser aconsejable para algunas aplicaciones pero no para todas.

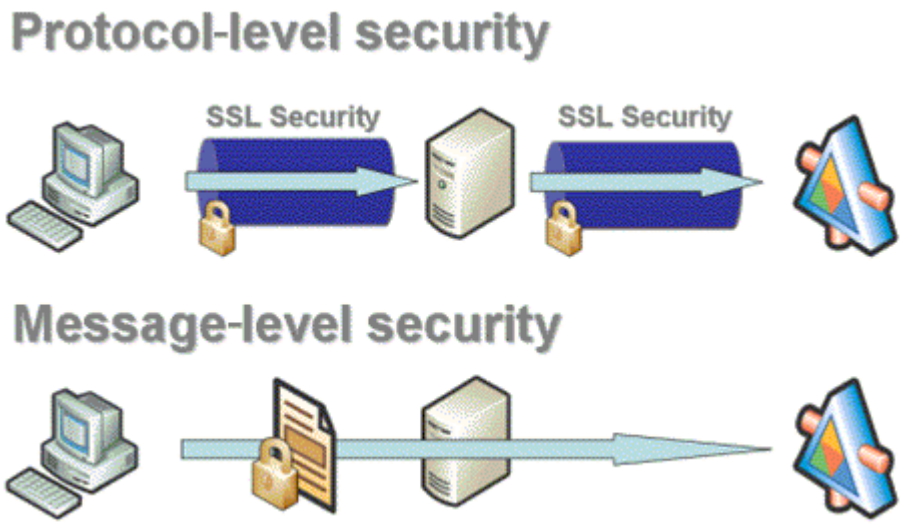


Fig. 1.2: Representación de seguridad a nivel de protocolo y a nivel de mensaje. Tomado de (10).

La seguridad ideal es el nivel de seguridad que aparece en la mitad inferior de la imagen (seguridad a nivel de mensaje), donde se fija el propio mensaje. Puede que el mensaje esté firmado o cifrado, y a veces ambas cosas. Ahora bien, esto no se puede hacer sólo con la seguridad del transporte como https. Hay que proteger el mensaje en sí. WS-Security está destinado sólo para esto. Se define la forma de asegurar el mensaje en sí mismo.

Debido a que se lleva a cabo una estrategia propia de seguridad a este nivel es que se hace selección del algoritmo que servirá para encriptar el contenido de los mensajes que se envíen a travez de los servicios web. Como estrategia se decide encriptar solo el cuerpo del mensaje que compone la información sencible. El destino tiene que estar al tanto de los datos necesarios para desencriptar dicha información y del cómo determinar que el mensaje fue enviado por un sistema válido.

El algoritmo para encriptar los mensajes será simétrico por poseer como característica el ser más rápido que los asimétricos (11). Para seleccionar el el mismo se definen aspectos claves que se tienen que tener en cuenta:

- Debe ser rápido.
- Debe estar avalado y probado.
- Debe poseer un alto nivel de seguridad.
- Debe poseer una llave de 128, 160 ó 256 bits.
- Debe ser de libre uso.

Luego de ser analizados los algoritmos 3DES (Triple DES), IDEA (International Data Encryption Algorithm) y AES (Advanced Encryption Standar) ó Rijndael, fue seleccionado este último por contar con las mejores características. A continuación se listan elementos por los cuales fue seleccionado.

- Fue elegido en el 2000 por el NIST (National Institute of Standards and Technology), para ser el estándar en los próximos 20 años, como reemplazo del 3DES.
- Soporta bloques de datos de 128 bits y claves de 80, 112, 128, 192, y 256 bits (aunque se aconseja usar de 128 bits en adelante).
- El tiempo ha permitido que AES sea adaptado poco a poco, desde los protocolos más usados como SSL, hasta las aplicaciones más especializadas, como VoIP.
- Para 80 bits de seguridad AES, es equivalente a 1024 bits de RSA, y 163 de ECDSA.
- Es rápido. (12)

1.8 Metodología, Lenguaje y Herramientas utilizadas para el desarrollo

En la actualidad, el uso intensivo de las soluciones informáticas en múltiples de los sectores de la sociedad y principalmente en los procesos productivos, ha provocado un notable crecimiento en la

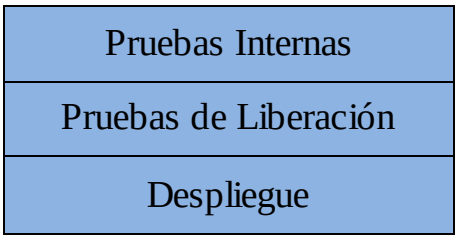
calidad y confiabilidad que estos deben poseer. Para lograr el desarrollo de un producto que satisfaga estos indicadores se debe tener en cuenta una serie de tendencias, tecnologías, lenguajes, metodologías y herramientas indispensables para garantizar la entrega de un software competitivo y que se ajuste a las necesidades de los clientes. A continuación partiendo de que esta selección se encuentra ya creada por parte del proyecto para el cual se elabora la solución que se describe, se relacionan las tecnologías, lenguajes, metodología y herramientas usadas para el desarrollo de la solución.

1.8.1 Modelo de desarrollo de software

En el Departamento de Soluciones para la Aduana debido a la experiencia acumulada en sus años de vigencia se ha elaborado un modelo de desarrollo propio. El mismo propone la generación de los artefactos necesarios para obtener un desarrollo simplificado y ágil, manteniendo los índices de calidad y eficiencia, permitiendo una mejor comunicación entre los analistas, diseñadores e implementadores de los proyectos. Algunos de estos artefactos son: Mapa de Procesos, Modelo Conceptual, Descripción y Modelado de los Procesos del Negocio, Especificación de los Requerimientos del Sistema, Diccionario de Datos, etc (13) . El ciclo de desarrollo de los proyectos en el departamento consta de 9 fases, las mismas son presentadas en la **Tabla 1.1**. El mismo además define los roles con sus resposabilidades, analista, desarrollador de interfaz de usuario, desarrollador de php, diseñador de sistemas, arquitecto de datos, diseñador de interfaz de usuario, especialista de calidad, arquitecto de sistemas, planificador o líder de gestión, y administrador de configuración.

Tabla. 1.1: Ciclo de vida de proyectos del Departamento de Soluciones para la Aduana. Tomado de (13).

Estudio Preliminar
Modelación del Negocio
Requisitos
Análisis y Diseño
Implementación
Pruebas Piloto



En la etapa de la Modelación del Negocio representada en el ciclo anterior, para el modelado de los procesos de negocio es usada la Notación para el Modelado de Procesos de Negocio (BPMN), aspecto del cual trata el epígrafe a continuación.

1.8.2 Notación para el Modelado de Procesos de Negocio

Busines Proccess Modeling Notation (BPMN)

Notación para el Modelado de Procesos de Negocio (BPMN), es una notación gráfica estandarizada para el modelado de procesos de negocio, en un formato de flujo de trabajo (workflow). BPMN fue inicialmente desarrollada por la organización Business Process Management Initiative (BPMI), y es actualmente mantenida por el OMG (Object Management Group), luego de la fusión de las dos organizaciones en el año 2005. Su versión actual es la 2 oficializada en enero de 2011. El principal objetivo de BPMN es proveer una notación estándar que sea fácilmente legible y entendible por parte de todos los involucrados e interesados del negocio (stakeholders). Entre estos interesados están los analistas de negocio (quienes definen y redefinen los procesos), los desarrolladores técnicos responsables de implementar los procesos y los gerentes y administradores del negocio (quienes monitorizan y gestionan los procesos). En síntesis BPMN tiene la finalidad de servir como lenguaje común para cerrar la brecha de comunicación que frecuentemente se presenta entre el diseño de los procesos de negocio y su implementación. (14)

1.8.3 Lenguaje de Modelado UML

El Lenguaje de Modelamiento Unificado (UML - Unified Modeling Language) es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables.

1.8.4 Herramienta de Modelado

Visual Paradigm para UML[12] 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 Visual Paradigm como herramienta de modelado posee licencia gratuita. Ayuda a una construcción más rápida de aplicaciones de calidad y permite el dibujo de todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación, así como una serie de tutoriales con demostraciones interactivas y proyectos. Ofrece un diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad. Cuenta con un modelo y código que permanece sincronizado en todo el ciclo de desarrollo, además de la disponibilidad de múltiples versiones, múltiples plataformas y capacidad de integrarse en los principales IDEs[13].

Permite la creación de diagramas mucho más rápido que cualquier herramienta en el mercado, además de la extensibilidad y diseño personalizado de apoyo. Tiene como característica principal ser mucho más fácil de utilizar en mercado en cuanto al uso que las herramientas CASE-UML le proporcionan, una buena interoperabilidad con otras aplicaciones y brinda muchas razones para ser el elegido dentro de un mercado que abarca muchas opciones. (15)

1.8.5 Entorno de Desarrollo Integrado (IDE)

NetBeans

Es un entorno de desarrollo integrado IDE (Integrated Development Environment) de código abierto para desarrolladores de software. Contiene todas las herramientas necesarias para crear aplicaciones profesionales de escritorio, empresariales, web y aplicaciones móviles con la plataforma Java, así como con C / C + +, PHP, JavaScript y Groovy. (16)

1.8.6 Lenguaje de Desarrollo

PHPHypertext Processor

Es un lenguaje interpretado de alto nivel usado para la programación web del lado del servidor y ejecutado en el mismo, este puede ser embebido en páginas HTML. El cliente solo visualiza el resultado de la ejecución del mismo en el servidor sin poseer manera de saber qué código ha generado tal resultado. Cuenta con innumerables funciones que hacen del mismo un lenguaje muy versátil y cuenta con soporte

[12] **UML:** Según sus siglas en inglés, Unified Modeling Language. Es un lenguaje de modelado para sistemas de software.
[13] **IDE:** Entorno de Desarrollo Integrado o en inglés Integrated Development Environment. Es un entorno de programación que ha sido empaquetado como un programa de aplicación.

para varios tipos de bases de datos. Para este lenguaje se han desarrollado varios marcos de trabajo de gran relevancia, a continuación se describe el marco de trabajo usado para la elaboración de la solución en el presente trabajo.

1.8.7 Marco de trabajo a usar.

Los marcos de trabajo son una buena herramienta para agilizar el proceso de desarrollo de software y más aún cuando se trata de sistemas complejos de gran envergadura, es por esto que se hace necesario el uso de los mismos para obtener una mayor eficiencia en el sistema final.

Symfony

Symfony es un Framework para PHP5 patentado bajo licencia MTI[14], es compatible con la mayoría de gestores de bases de datos, MySQL, PostgreSQL, Oracle y SQL Server de Microsoft, entre otros en dependencia del tipo de abstracción de la Base de Datos que se utilice.

Este Framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además proporciona una 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 está completamente diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Symfony está desarrollado completamente con PHP 5. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows.

Estos son algunos de los elementos por los que fué definido este como el Framework seleccionado para desarrollar en el Departamento de Soluciones para la Aduana y definido en el trabajo de tesis titulado “Línea Base Arquitectónica para el Polo Sistemas Tributarios y de Aduanas” realizado por José Antonio Cobo Rodríguez en junio de 2008. (17)

[14]Se autoriza, de forma gratuita, a cualquier persona que obtenga una copia de este software y archivos de documentación asociados (el "Software"), para trabajar con el Software sin restricción, incluyendo sin limitación los derechos para usar, copiar, modificar, fusionar , publicar, distribuir, sublicenciar, y/o vender copias del Software… (35)

Para el trabajo con este Framework existen inmensidad de plugins o complementos desarrollados que dotan al mismo de numerosas funcionalidades. Con el objetivo de hacer uso de las tecnologías ya desarrolladas para el mismo y explotar sus ventajas se estudió un plugin para el trabajo con los servicios web, este es el **ckWebService**[15].

Este se encuentra desarrollado para symfony 1.3 y es compatible con la versión 1.4, el mismo permite construir una interfaz de programación de aplicaciones o API (del inglés Application Programming Interface) de servicio web para las aplicaciones de Symfony. Viene con un potente y fácil de usar generador de descripción de lenguaje de servicio web o WSDL(Web Services Description Language), que crea archivos WSDL compatibles con WS-I[16] para un máximo de interoperatibilidad con PHP,. NET y los clientes de Java. (18)

El mismo tiene como requisitos para un correcto funcionamiento:

- PHP >= 5.2.4
- Extensión php_soap

1.9 Conclusiones del capítulo

En el presente capítulo se estudiaron los sistemas de VUCE existentes los cuales no arrojaron elementos relevantes para la definición de la solución, se estudiaron además las principales técnicas de integración de datos, en especial **EAI** de la cual se extrajeron los elementos claves para la construcción de la solución al problema planteado, definiéndose de esta manera el uso de los servicios web como elemento fundamental para el intercambio de información. Se definieron aspectos referentes al manejo de la seguridad de la información mediante el encriptado de los datos, seleccionándose como algoritmo de encriptado **AES** por las cualidades que presenta. Se determinaron a través de las bases planteadas en el modelo de desarrollo del centro CEIGE y la arquitectura base del proyecto VUCE las tecnologías y herramientas a utilizar para el desarrollo de la solución. Como principales elementos en la arquitectura base se estableció el uso de jQuery 1.6.4, html 4.0 y css3 para la elaboración de las vistas, el marco de trabajo symfony 1.4.16 y propel 1.6 para el desarrollo del negocio del lado del servidor y como sistema

[15] Plugin del framework symfony para el trabajo con los servicios web.
[16] Web Services Interoperability Organization (WS-I) es una organización industrial abierta creada para establecer mejores prácticas para la interoperabilidad de servicios Web, para grupos seleccionados de estándares de servicios Web, a través de plataformas, sistemas operativos y lenguajes de programación. (37)

gestor de base de datos Oracle 11g. De esta manera quedaron abordados los elementos necesarios para proseguir con los siguientes capítulos del presente documento.

CAPÍTULO 2: ANÁLISIS Y DISEÑO

2.1 Introducción

En este capítulo se presenta una propuesta del sistema a desarrollar, se exponen las principales características que presenta el negocio y se definen los principales requisitos funcionales. Se muestran las técnicas de captura y validación de requisitos, obteniéndose además los principales artefactos del diseño que guiarán el posterior proceso de implementación de la solución, haciendo énfasis en el modelo de diseño de la misma.

2.2 Modelado de los procesos de negocio

A continuación se hace una descripción de los procesos de envío de documentos a sistemas externos y recepción de documentos de sistemas externos, persiguiendo como finalidad obtener una visión más clara del negocio que dé la posibilidad de un desarrollo eficiente de la aplicación.

2.2.1 Descripción del proceso enviar documento

En el siguiente diagrama (ver **Figura. 2.1**) se presenta el proceso Enviar Documento a Sistema Externo a partir del cual se ejecuta el subproceso Guardar Documento, a partir de esto se ejecuta una tarea programada que inicializa el proceso de envío de los documentos que se encuentren almacenados listos para ser enviados permitiendo enviar los documentos luego de que estos hayan sido almacenados

Como se puede apreciar el flujo se inicia cuando un subsistema inicia el proceso enviando un documento para ser enviado a un sistema externo, a partir de aquí se ejecuta el subproceso guardar documento que se encarga de validar los datos del documento, comprobar que existe el sistema para el que se desea enviar dicho documento y almacenarlo. A partir de aquí se ejecuta cada cierto tiempo una tarea que ejecuta el envío de los documentos almacenados. Esta separación del proceso permite dar inmediata respuesta a las solicitudes de envío de documentos por parte de los usuarios y llevar a cabo por separado los pasos necesarios para completar el ciclo de envío.

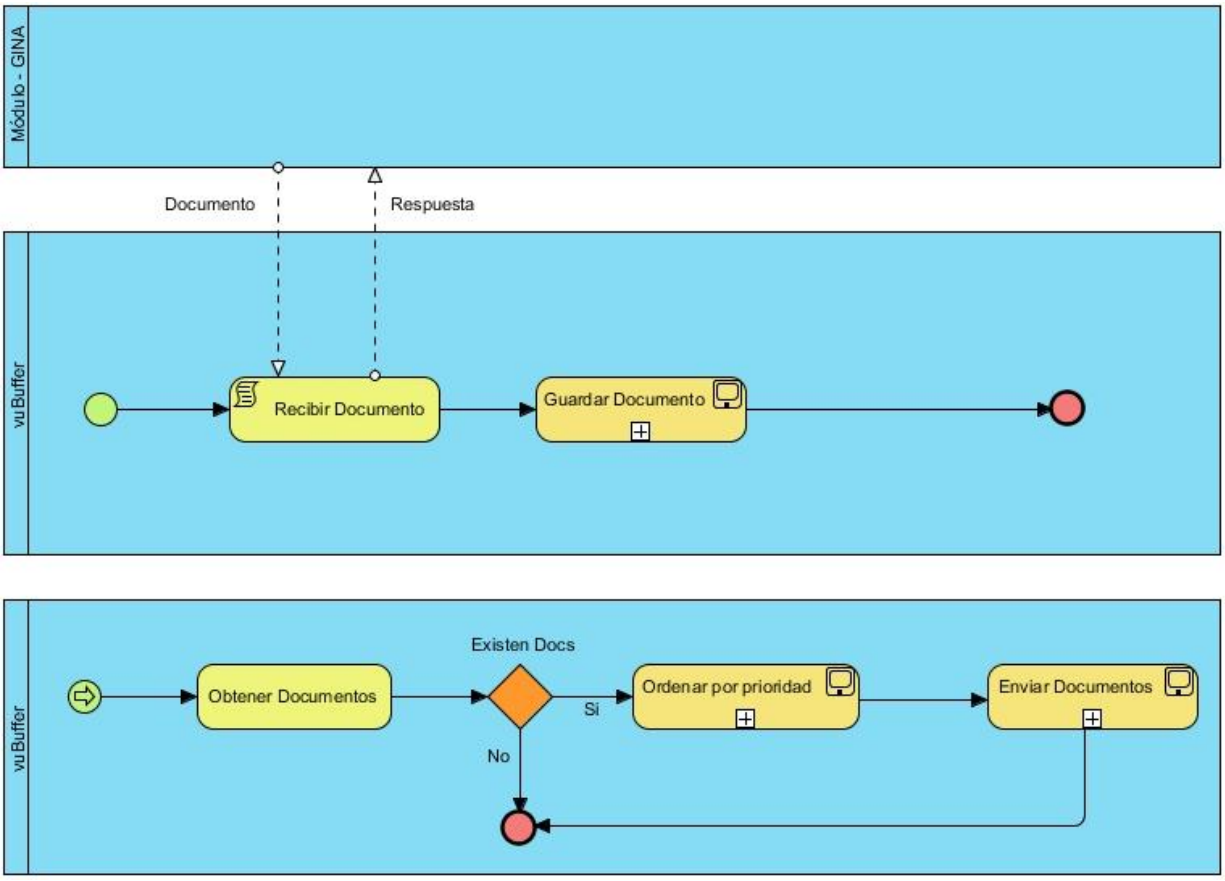


Fig. 2.1 Proceso Enviar Documento a Sistema Externo

2.3 Modelo conceptual

A continuación (ver **Figura. 2.2**) se presenta el modelo conceptual del negocio, presentándose los conceptos relevantes del negocio y sus relaciones.

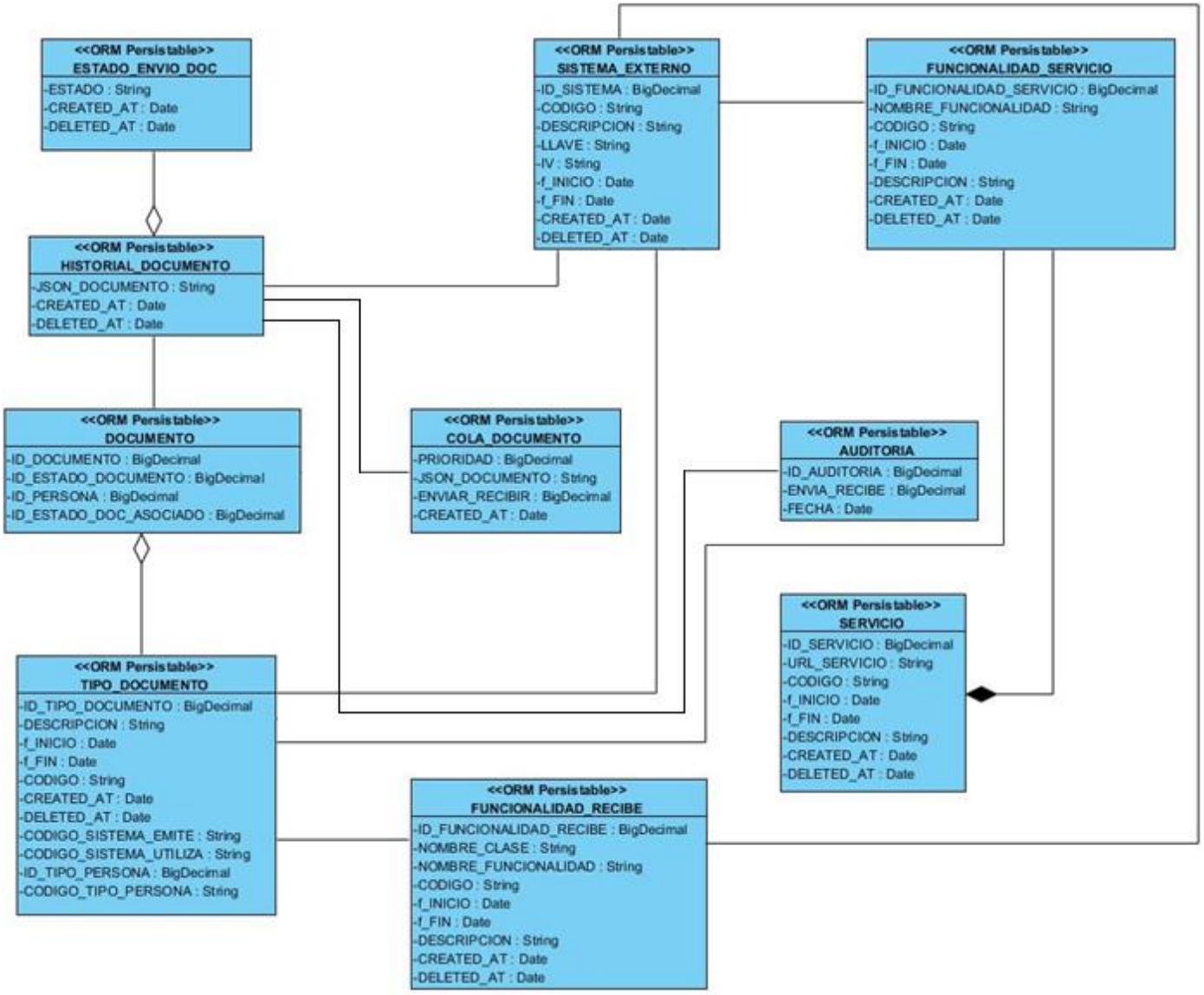


Fig. 2.2 Modelo Conceptual del Sistema de Envío y Recepción de Documentos

2.4 Especificación de los requisitos del sistema

Los requisitos para un sistema de software son las bases que determinan en gran parte las actividades de la ingeniería de software que prosiguen en el proceso de desarrollo, estos establecen lo que hará el sistema y definen las restricciones de su operación e implementación, por lo que se deben tener muy en cuenta las técnicas que mejores resultados arrojarían para la captura y validación de los requisitos.

Seguidamente se muestran las técnicas usadas para la captura de los requisitos así como un listado de los mismos.

2.4.1 Técnicas de captura de requisitos

Según el procedimiento para la ingeniería de requisitos establecida en el Departamento de Soluciones para la Aduana del CEIGE fueron seleccionadas como técnicas para la captura de requisitos la entrevista, la observación, la tormenta de ideas y los talleres. Dadas las particularidades de la solución que se desea desarrollar se seleccionaron como técnicas a utilizar solo la entrevista, la tormenta de ideas y los talleres, siendo estas las que arrojarían los mejores resultados.

Se hace una combinación de estas tres técnicas para obtener los mejores resultados posibles. Se entrevistaron los ingenieros creadores de la idea de la realización de la VUCE para Cuba por ser los que más conocimientos poseían del sistema en su totalidad, así como los principales desarrolladores del sistema GINA con el objetivo de establecer todos los requisitos que debía cumplir la solución informática. A la par se entrevistaron los principales analistas del sistema VUCE para identificar elementos de los documentos que permitieran crear prioridades para aumentar en eficiencia. Con todos los datos obtenidos se desarrollaron tormentas de ideas entre todo el equipo de desarrollo que permitieron refinar los datos obtenidos y como último medio se llevaron a cabo talleres donde cada miembro del equipo de desarrollo expuso su criterio respecto a los resultados obtenidos, llevándose a cabo un último paso de gran relevancia para la obtención de resultados certeros.

2.4.2 Requisitos funcionales

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requisitos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer. (19)

Se pueden determinar también como capacidades o condiciones que el sistema debe cumplir, hacen una descripción detallada de su función, las entradas y salidas, los mismos deben ser correctamente determinados y teniendo que ser de total entendimiento tanto para los implicados con el sistema como para los desarrolladores del mismo, facilitando de esta manera un entendimiento común.

A continuación se listan los requisitos definidos para el sistema modelado:

RF1 Enviar documento a sistema externo: el sistema debe ser capaz de transmitir un documento a un sistema externo con el que se tenga establecido un intercambio de información.

- Enviar documento en estado de (error al enviar).
- Cancelar envío de documento.
- Encriptar contenido de los documentos.

RF2 Recibir documento de sistema externo: el sistema debe ser capaz de proveer un medio por el cual hacer la recepción de los documentos que le sean enviados de un sistema externo con el que se tenga establecido un intercambio de información.

- Recibir documento en estado (error al recepcionar).
- Cancelar recepción de documento.
- Desencriptar contenido de los documentos.

RF3 Enviar incidencia a sistema externo: el sistema debe ser capaz de enviar una incidencia a un sistema externo en el menor tiempo posible.

RF4 Registrar auditoría del envío y recepción de documentos a sistemas externos: el sistema debe ser capaz de registrar datos de todos los eventos de envío y recepción de documentos con sistemas externos.

RF5 Registrar historial de documentos enviados y recibidos de sistemas externos: el sistema debe ser capaz de registrar los documentos que sean enviados y recibidos de sistemas externos así como los estados por los que pasan en todo el proceso de envío y recepción.

RF6 Establecer prioridad a los documentos para el envío a sistemas externos: el sistema debe ser capaz de establecer una prioridad en los documentos que permita un envío eficiente de los mismos.

RF7 Mostrar auditorias del envío y recepción de documentos a sistemas externos: el sistema debe ser capaz de mostrar los sucesos relacionados con el envío y recepción de información con sistemas externos en tiempo real.

RF8 Mostrar historial de los documentos enviados y recibidos de sistemas externos, así como los estados asociados: El sistema debe ser capaz de mostrar los documentos enviados y recibidos de

sistemas externos, permitiendo filtrar los mismos por diferentes parámetros (fecha de envío, estado, tipo documento, etc.), así como mostrar el historial de los estados asociados a los documentos.

2.4.3 Técnicas para la validación de requisitos

Siguiendo lo especificado en 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 requisitos 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 requisitos, construcción de prototipos, matrices de trazabilidad y generación de casos de pruebas. (20)

La validación de los requisitos se realizó mediante la unión de las siguientes técnicas:

- Revisiones del documento de requisitos y construcción de prototipos.
- Se realizaron diversas revisiones al documento de requisitos con la participación de los analistas del proyecto, jefe del proyecto y demás desarrolladores del mismo para logar una correcta interpretación de la información transmitida, los señalamientos planteados fueron recepcionados y aplicados posteriormente.
- Además se efectuó la construcción de prototipos, los cuales fueron presentados por cada requisito de software que lo permitía, aclarando con el jefe del proyecto si las necesidades fueron cubiertas por el sistema.

2.5 Aportes de la Solución y Beneficios Esperados

El sistema propuesto provee las funcionalidades necesarias para gestionar el envío y recepción de información entre la VUCE y el GINA proveyendo los elementos necesarios para el intercambio de información con otros sistemas cuando se requiera en un futuro de manera eficaz y eficiente, contribuyendo de esta forma a la realización rápida de los trámites comerciales realizador por la Aduana en las actividades del comercio exterior.

El sistema permitirá mediante el uso de servicios web el envío de información a sistemas externos permitiendo almacenar pedidos en una estructura de buffer que garantizará una total disponibilidad del

mismo para los módulos internos que lo usen, enviando la información a medida que tenga disponibilidad para hacerlo.

Almacenará los documentos que sean enviados, un historial de los estados por los que pasan los mismos y todos los eventos relacionados con el envío y recepción de información con el objetivo de poder citar dicha información en cualquier momento por parte de los administradores del sistema. Todo esto permitirá mantener una correcta trazabilidad de los sucesos ocurridos en el sistema.

El envío de estos documentos será basándose en una prioridad asignada a los mismos, en el próximo epígrafe se relacionan los detalles relacionados al cálculo de la prioridad de estos.

2.6 Cálculo de la prioridad de los documentos para su envío a sistemas externos

El envío de los documentos a sistemas externos de acuerdo a una prioridad establecida es un requisito de gran importancia, de esta forma se garantizará hasta cierto punto que no exista información que llegue atrasada o fuera del plazo establecido a su destino. Para establecer la prioridad se hace necesario puntualizar los aspectos por los cueles la misma será calculada de acuerdo a negocio que se maneje. Para los documentos que se manejarán en la VUCE en su primera versión se establecieron 3 aspectos por los cuales se regirá la asignación de la prioridad, estos son:

- Tiempo de validez: representa el tiempo que posee un documento, desde que es introducido a la VUCE, para que el trámite correspondiente al mismo sea culminado.
- Término de presentación: representa el tiempo con antelación a una fecha de vencimiento en que debe ser procesado un documento.
- Fecha de vencimiento: representa la fecha en que un documento deja de ser válido.

Los documentos a ser enviados desde la VUCE poseerán un tiempo de validez o un término de presentación en conjunto con una fecha de vencimiento, en caso de poseer ambos datos se regirá por el término de presentación con la fecha de vencimiento. La prioridad para este caso es representada como el tiempo de validez que le queda a un documento, variando la misma a medida que pasa el tiempo. Es por esto que mientras más pequeño sea este valor mayor será la prioridad que le corresponde, una vez llegue este valor al mínimo establecido, el documento será tratado como aquellos documentos que no poseen prioridad y serán colocados detrás de todos los que aún posean una prioridad por encima de la mínima.

De esta forma la prioridad quedaría definida por las siguientes fórmulas, estableciéndose la unidad de medida del tiempo en segundos.

Sean:

P: Prioridad.

T_{Resp}: Tiempo de respuesta.

F_{Venc}: Fecha de vencimiento.

F_{Act}: Fecha actual.

$T_{Present}$: Término de presentación.

Para un tiempo de respuesta definido sería:

$$P = T_{Resp}$$

Para un término de presentación y una fecha de vencimiento definida sería:

$$P = F_{Venc} - F_{Act} - T_{Present}$$

La prioridad de un documento al estar definida por los factores anteriormente mencionados, varía a medida que pasa el tiempo por lo que se hace necesario recalcularla una vez sean recuperados los documentos para ser enviados. Para recalcular la misma solo se hace necesario restarle a la prioridad previamente definida el tiempo que ha transcurrido desde su almacenamiento hasta el momento de su recuperación en segundos. De esta forma se trabajará con una prioridad real en cada momento. Mediante este mecanismo se garantizará el envío de los documentos de manera ordenada, minimizando el riesgo de que un documento llegue a su destino fuera de su término de validez y por consiguiente favoreciendo la realización efectiva de las operaciones comerciales.

2.7 Diseño del sistema

La importancia del diseño del software puede ser descrita como calidad. El diseño es la etapa en la que se fomentará la calidad en la ingeniería del software, proporciona las representaciones del software susceptibles de evaluar respecto de la calidad y es la única forma en que, de manera exacta, un requisito

del cliente se puede convertir en un sistema o producto de software terminado.Sirve como fundamento para todas las actividades subsecuentes de la ingeniería de software y del soporte de éste. Sin diseño se corre el riesgo de construir un sistema inestable, el cual fallará en la realización de pequeños cambios; que será difícil de probar; cuya calidad no podrá evaluarse sino hasta etapas tardías del proceso de desarrollo, cuando queda poco tiempo y ya se ha gastado mucho en él. (21)

2.7.1 Patrones utilizados

La utilización del marco de trabajo Symfony provee al sistema a desarrollar de varios de los patrones de diseño y arquitectónicos más utilizados en la actualidad, de esta forma se garantiza por una parte la utilización de buenas prácticas en el desarrollo, quedando solo por parte de los diseñadores y programadores el seguir estas prácticas en las estructuras que sean creadas dentro de la aplicación.

2.7.1.1 Patrones GRASP

Los patrones GRASP representan los principios básicos de la asignación de responsabilidades a objetos, expresados en forma de patrones. GRASP es el acrónimo para General Responsibility Assignment Software Patterns (Patrones Generales de Software para Asignar Responsabilidades).

Controlador: Es un evento generado por actores externos. Se asocian con operaciones del sistema, como respuestas a los eventos del sistema, tal como se relacionan los mensajes y los métodos. Normalmente un controlador delega en otros objetos el trabajo que se necesita hacer; coordina o controla la actividad. No realiza mucho trabajo por sí mismo. (22)

En Symfony todas las peticiones Web son manejadas por un solo controlador frontal, que es el punto de entrada único de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para enviar la acción al controlador responsable de la solución. Además de esto el controlador frontal se encarga de manejar la seguridad y la ejecución de los filtros en Symfony. Este patrón se evidencia en las clases sfFrontController, sfWebFrontController, sfContex, los “actions” y el index.php del ambiente.

Alta cohesión: La cohesión es una medida de la fuerza con la que se relacionan las clases y el grado de focalización de las responsabilidades de un elemento. Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable, una clase con baja cohesión hace muchas cosas no relacionadas o hace demasiado trabajo. (23)

Symfony agrupa las clases por funcionalidades que son fácilmente reutilizables, bien por su uso directo o por herencia. Un ejemplo de ello es la clase vuBufferActions, la misma es la responsable de procesar todas las acciones para el servicio web del cual consumen los sistemas externos, es capaz de colaborar con otras clases para la realización de diferentes operaciones, instanciar objetos y tener acceso a sus propiedades.

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. Una clase con bajo o débil acoplamiento no depende de muchas otras. (23)

Symfony asigna a cada clase una responsabilidad para mantener pocas dependencias entre las mismas. En el caso de vuBufferActions hereda solamente de sfActions para obtener un bajo acoplamiento de clases.

Experto: La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. (23)

Symfony utiliza en el caso presente usa Propel 1.6 para realizar su capa de abstracción en el modelo, encapsular toda la lógica de los datos y generar las clases con todas las funcionalidades comunes de las entidades, las clases de abstracción de datos poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen solo la información necesaria de la tabla a la cual representan.

Creador: Este patrón como su nombre lo indica es el que crea, el guía la asignación de responsabilidades relacionadas con la creación de objetos. (23)

La clase BufWorkTask es la encargada de crear las instancias de las tareas definidas en las clases BufEnvioSistemaTask y BufRecepcionSistemaTask, por lo que esta es definida como la clase “creador” de las otras dos.

2.7.1.2 Patrones GoF

Singleton: El controlador frontal de Symfony proporciona un punto de acceso global a la aplicación. En una acción, el método getContext() devuelve el mismo singleton. Se trata de un objeto muy útil que guarda

una referencia a todos los objetos del núcleo de Symfony relacionados con una petición dada, y ofrece un método de acceso para cada uno de ellos, ejemplo de ello son:

- **sfController**: El objeto controlador (->getController()).
- **sfUser**: El objeto de la sesión del usuario (->getUser()).
- **sfDatabaseConnection**: La conexión a la base de datos (->getDatabaseConnection()).

Decorator: Responde a la necesidad de añadir dinámicamente funcionalidad a un Objeto. Permitiendo no tener que crear sucesivas clases que hereden de la primera incorporando la nueva funcionalidad, sino otras que la implementan y se asocian a la primera. (24)

La plantilla no es un documento XHTML válido. Le faltan la definición del DOCTYPE y las etiquetas <html> y <body>. El motivo es que estos elementos se encuentran en otro lugar de la aplicación, un archivo llamado Layout.php que contiene el Layout de la página. Este archivo, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el Layout, o si se mira desde el otro punto de vista, el Layout decora la plantilla. En la **Figura 2.3** se aprecia una representación de este patrón.

Fig. 2.3: Implementación del patrón Decorador por Symfony. Tomado de (25).

2.7.1.3 Patrón MVC en Symfony

El patrón de arquitectura Modelo Vista Controlador **MVC** (del inglés Model View Controller), es el cual permite llevar a cabo una programación multicapa, separando a la interfaz del usuario, la lógica del control y los datos de la aplicación en tres componentes diferentes. Este patrón se ve principalmente de manera más usual en aplicaciones web, donde la vista es la página (html) que se carga en el navegador web y el código que provee de datos dinámicos a la página, el modelo, es el sistema de gestión de base de datos y el controlador representa la lógica del negocio, que está formada por estos tres niveles:

- **Modelo**: representa toda la información con la que opera la aplicación, o sea su lógica de negocio, responde a las peticiones de estado que vienen de la vista, gestiona el comportamiento y los datos del dominio y responde a instrucciones de cambio de estado provenientes del controlador.
- **Vista**: gestiona la presentación de la información de la aplicación, en otras palabras convierte el modelo en una página web que facilita al usuario la interacción con ella.
- **Controlador**: es el encargado de procesar las interacciones del usuario y ejecuta los cambios adecuados en el modelo o en la vista. La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista), lo que permite un mantenimiento más sencillo de las aplicaciones. El controlador es el encargado de aislar al modelo y a la vista de los detalles del protocolo usado para las peticiones (HTTP, consola de comandos, email, etc.). El modelo se encarga de la abstracción de la lógica referida a los datos, lo que permite que la vista y las acciones sean independientes del tipo de gestor de bases de datos que la aplicación utiliza por citar un ejemplo.

La implementación que realiza Symfony de la arquitectura MVC incluye varias clases:

- **sfRequest**: guarda los elementos que integran la petición (parámetros, cookies, cabeceras, entre otros elementos de las peticiones).
- **sfController**: es la clase del controlador y es la encargada de decodificar la petición y transferirla a la acción correspondiente.
- **sfResponse**: posee las cabeceras de la respuesta y los contenidos. El contenido de este objeto se convierte en la respuesta HTML que se remite al usuario.
- **sfContext::getInstance()**: representación de la aplicación del patrón singleton, guarda una referencia a todos los objetos que constituyen el núcleo de Symfony y puede ser accedido desde cualquier parte de la aplicación.

A continuación en la **Figura 2.4** se muestra la estructura de este patrón.

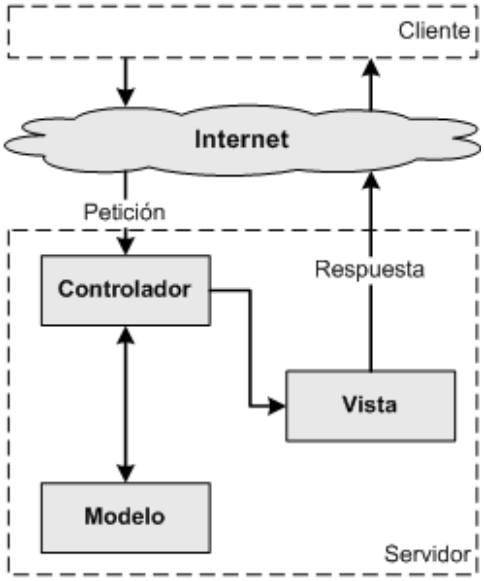


Fig. 2.4: Patrón MVC. Tomado de **(25)**.

2.7.2 Modelo del diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de usos centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen un impacto en el sistema a considerar. Además, el modelo de diseño sirve como abstracción de la implementación del sistema y es, de ese modo, utilizado como una entrada fundamental de las actividades de implementación. (26)

El modelo del diseño de la presente investigación se encuentra compuesto por el diagrama de clases del negocio, el diagrama de paquetes y por los diagramas de secuencia orientados a las actividades de los requisitos funcionales.

2.7.2.1 Diagrama de clases del negocio

A continuación (ver **Figura. 2.5**) se muestra el diagrama de diseño de las clases del negocio, el mismo es una aproximación al diseño definitivo del modelo de datos.

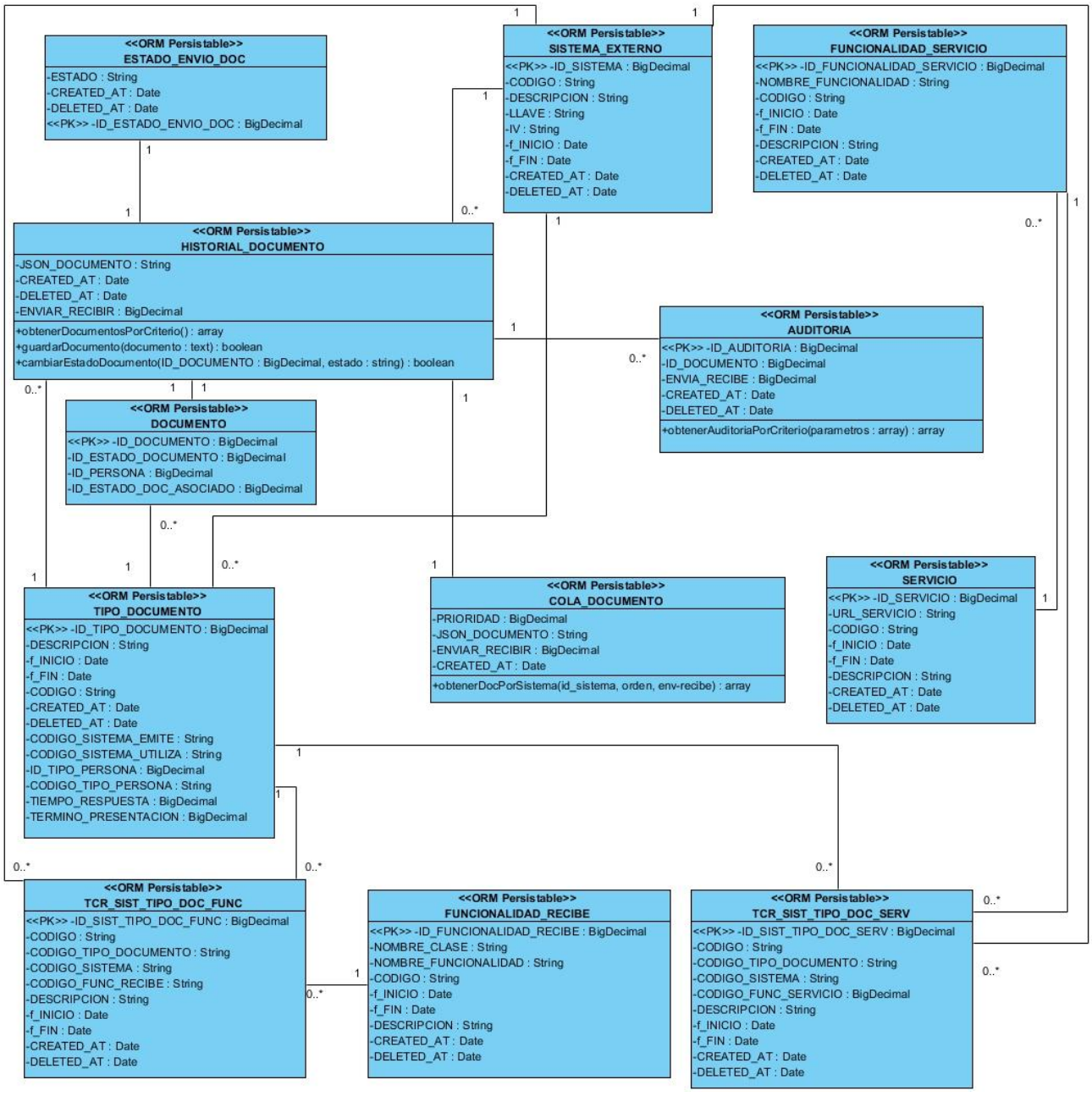


Fig. 2.5 Diagrama de clases persistentes.

2.7.2.2 Diagrama de paquetes.

Un diagrama de paquetes es un conjunto de agrupaciones lógicas de elementos del modelo, mostrando las dependencias entre dichas agrupaciones. Los paquetes están normalmente organizados para maximizar la coherencia interna dentro de cada paquete y minimizar el acoplamiento externo entre los

paquetes. Un paquete puede contener paquetes subordinados, así como otros tipos de elementos del modelo. Todo tipo de elementos del modelo UML pueden ser organizados en paquetes.

En la **Figura 2.6** se muestra el diagrama de paquetes correspondiente a la solución que se modela. El mismo está compuesto por los paquetes TC, Symfony lib, ckWebService, vuBuffer dentro del paquete VU, siendo estos los que presentan relación directa con la solución, denotándose en color verde el paquete vuBuffer por sobre los demás por ser el que contendrá toda la lógica programada concerniente a la solución modelada en la presente investigación. También aparecen dos paquetes GINA y Otros sistemas, los cuales representan los sistemas externos con los cuales se llevará a cabo el envío y recepción de información, estando como objetivo primario el sistema GINA.

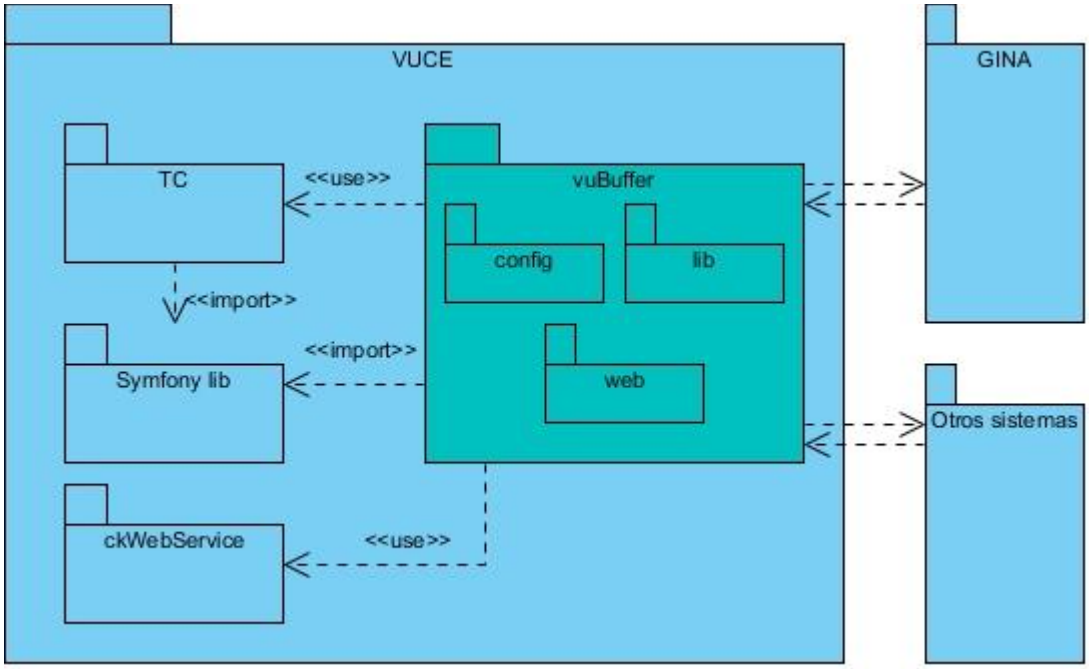


Fig. 2.6 Diagrama de paquetes.

2.7.2.3 Diagrama de interacción

Los diagramas de interacción muestran como los objetos se comunican unos con los otros para satisfacer los requisitos del sistema, explica gráficamente las interacciones existentes entre las instancias de las clases. Estos son importantes para modelar los aspectos dinámicos de un sistema, estos contienen objetos enlaces y mensajes. Existen dos tipos de diagramas de interacción basados en la misma

información pero cada uno enfatiza en un aspecto en particular, los diagramas de secuencia y los de colaboración.

2.7.2.3.1 Diagrama de Secuencia Orientado a Actividades del Negocio

Un diagrama de secuencia representa una interacción como un gráfico bidimensional. La dimensión vertical es el eje de tiempo, que avanza hacia abajo de la página. La dimensión horizontal muestra los roles de clasificador que representan objetos individuales en la colaboración. Cada rol de clasificador se representa mediante una columna vertical, línea de vida. (27)

El diagrama de secuencia orientado a actividades del negocio, fue definido con el objetivo principal de definir con una mayor certeza la relación que existe entre las funcionalidades que se implementarán y las clases donde deben ser implementadas. Este es formado a partir de las principales facilidades que brindan el Diagrama de Actividades y el Diagrama de Secuencia. Uno de los objetivos que se persigue con la realización del mismo es facilitar el trabajo a los programadores que no poseen un alto nivel de relación con el desarrollo del proyecto, facilitándose así el entendimiento del diseño de cada una de las actividades orientadas por los diseñadores.

Una de sus principales características es la declaración de funciones y variables, las especificaciones de los parámetros que se pasan a las funciones y las llamadas a funciones de otras clases. A continuación en la **Figura 2.7** se muestra un diagrama de este tipo perteneciente a la funcionalidad enviar incidencia.

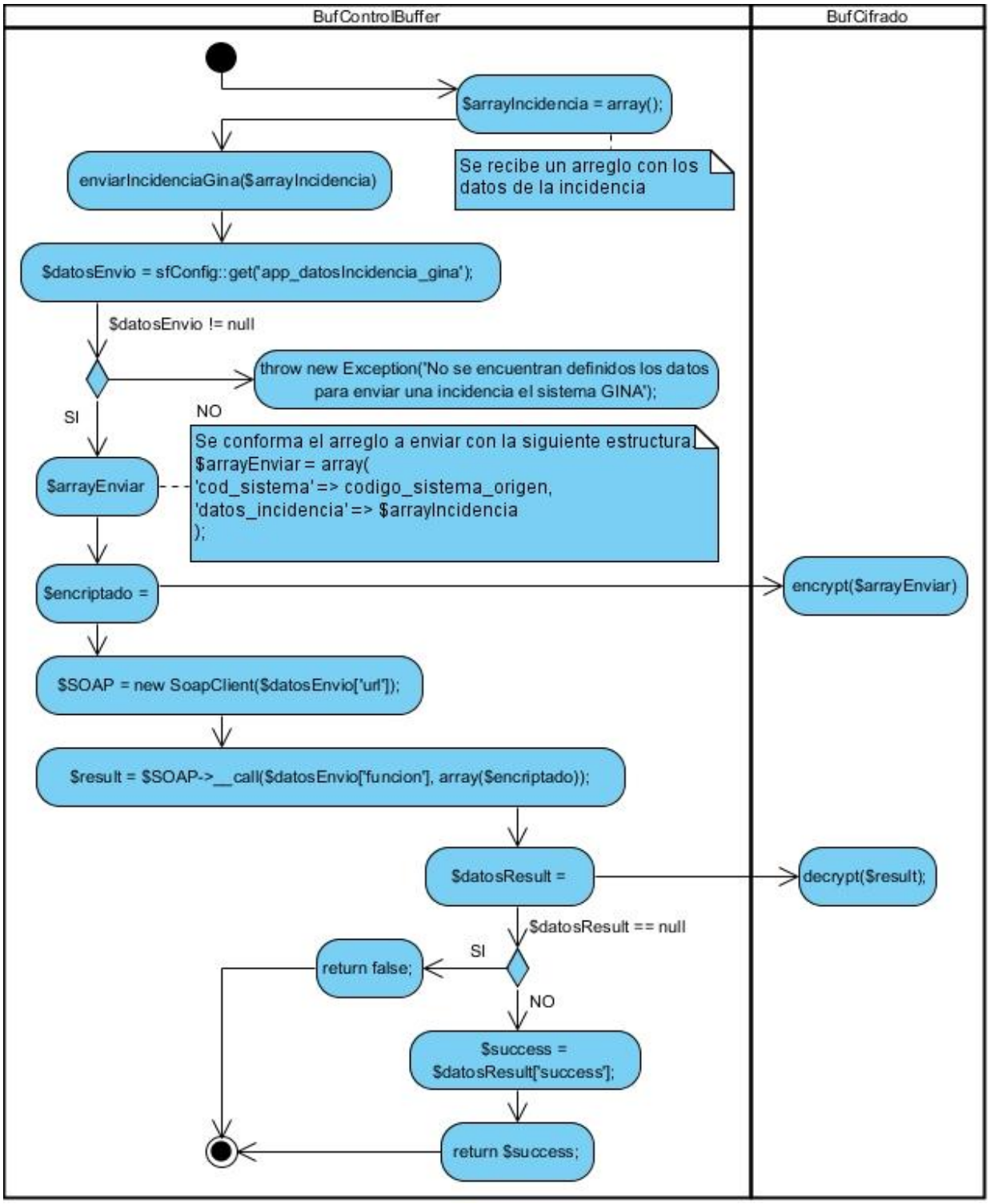


Fig. 2.7 Diagrama de Secuencia Orientado a Actividades del Negocio

2.7.3 Diseño de la Base de Datos

A continuación en la **Figura 2.8** se muestra el diagrama Entidad – Relación perteneciente al sistema. En el mismo se muestran las entidades que forman parte de la base de datos que almacenará toda la información correspondiente a la solución en desarrollo. En el mismo con el objetivo de obtener una mayor comprensión, se señalaron en color gris fuerte las tablas o entidades pertenecientes al subsistema TC (Tablas De Control), y en gris claro una entidad perteneciente al subsistema de notificaciones, dichas entidades pertenecen a estos otros subsistemas pero por la estrecha relación que poseen con el modelo de datos de la solución en desarrollo son añadidas a este diagrama.

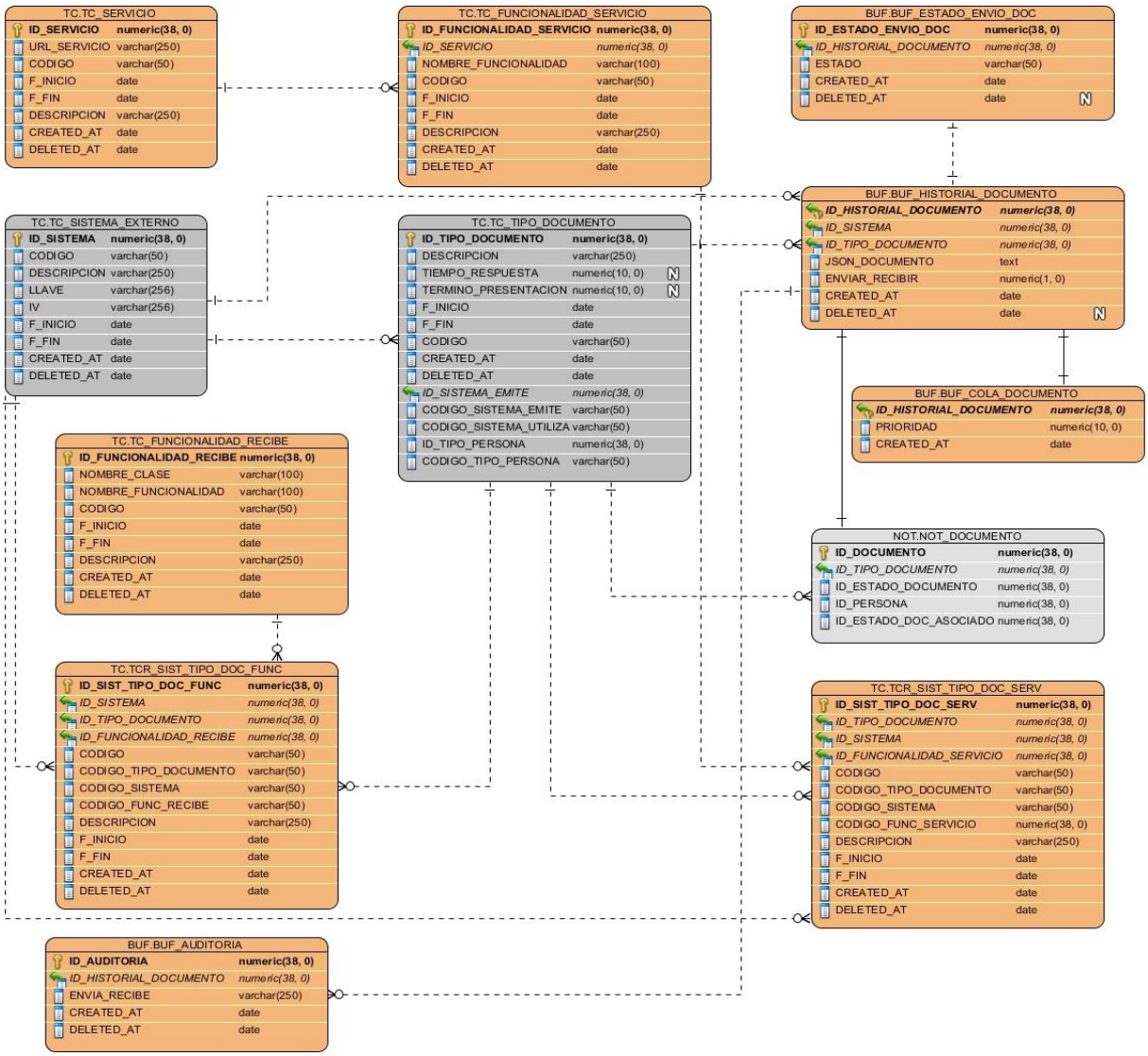


Fig. 2.8 Diagrama Entidad - Relación.

2.8 Conclusiones del capítulo

En el presente capítulo se llevó a cabo la etapa de análisis, donde se obtuvo como resultado la modelación y descripción de los procesos de negocio a fin de lograr un máximo entendimiento de los mismos, siendo esta la base para el correcto diseño, etapa en la cual fueron generados los artefactos que dan forma a la solución a desarrollar próximamente en el siguiente capítulo, donde se describen los elementos de la implementación de la solución modelada. Los resultados de la implementación dependen de la solidez con la que se hayan desarrollado las etapas anteriores, de aquí la importancia de estas.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS

3.1 Introducción

En el presente capítulo se abordará los temas referentes a la implementación de la solución anteriormente analizada y modelada, en la cual se hará evidente el uso de los patrones de diseño en la codificación. Esto será posible mediante la traducción del diseño, tomando guía los diagramas de interacción como base para llevar a cabo la codificación. En este capítulo serán abordadas las pruebas desarrolladas que garantizarán una correcta obtención de indicadores que permitan revelar la calidad de la solución desarrollada, mediante el análisis del comportamiento de la interoperabilidad.

3.2 Estándar de Codificación

Como estándar de codificación se utilizó el definido en el Departamento de Soluciones para la Aduana (titulado: Propuesta de un estándar de codificación. Dpto. Aduana CEIGE) al cual pertenece el proyecto para el cual se desarrolla la solución expuesta en el presente trabajo. En el mismo se definen los diferentes estándares a seguir tanto en la implementación de las pantallas, las clases del negocio como en las clases definidas para las pruebas. A continuación se exponen algunos de los principales elementos definidos en dicho documento.

Nombres de las clases:

1 Los nombres de las clases deben estar expresados en notación **UpperCamelCase**[17].
2 No se deben utilizar guiones bajos en su nombre "_".
3 Deben expresar con claridad cuál es el alcance y la responsabilidad de la clase.
4 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, no en dependencia de la clase de la que deriva.
5 En los nombres compuestos por más de tres palabras se debe revisarse el diseño, no sea que se le estén dando a la clase más responsabilidades de las que realmente tiene.

Nombre de las funciones:

Todas las funciones definidas por los desarrolladores deben seguir la nomenclatura **UpperCamelCase**, a no ser que para cierto ámbito se especifiquen características específicas.

[17] **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

Además deben cumplir con las siguientes disposiciones de forma general:

1. Los nombres de las funciones deben dejar reflejado claramente cuál es la acción que realiza el mismo.

Ejemplo: ManejarErrorBD, DescagarArchivoFTP, etc.

2. Se debe apoyar en la utilización de sufijos que ayuden a identificar el resultado final de la ejecución de un método:

Max: devuelve el máximo de una operación Contar: devuelve la cantidad de valores de una operación

3. Se debe apoyar en las prefijos para expresar la acción que realiza sobre un elemento determinado:

Set: para cambiar el valor de una variable. Get: para devolver un valor determinado. Is: devuelve un valor booleano de si un elemento cumple con una condición dada.

Políticas de {} llaves:

1. Colocar las llaves en líneas separadas del código al mismo nivel del inicio del bloque.

```
if ($condition)          while ($condition)
{                        {
  …                        …
}                        }
```

2. Política tradicional de Unix de colocar la primera llave en la misma línea del comienzo del bloque separada por un espacio y la llave de cierre del bloque en la última línea.

```
if ($condition) {        while ($condition) {

  …                        …

}                        }
```

3. Cualquiera que sea la estrategia adoptada debe ser mantenida en todo el código escrito por el programador y siempre colocar las llaves aunque sea para una sola línea de código dentro del bloque.

Política de () paréntesis:

1. No utilizar paréntesis seguido de palabras claves del lenguaje, se debe dejar un espacio por el medio.
2. Los paréntesis se sitúan seguidos de los nombres de las funciones.
3. No se debe utilizar paréntesis en las sentencias return de no ser necesario.

```
Ejemplo:
 if (condition)                while (condition)
 {                      {
 }                      }
strcmp($s, $s1);
 return 1;
```

Comentarios y documentación:

1. Para comentarear una línea se utilizará el o los caracteres que dicta el lenguaje de programación utilizado, al igual que para los comentarios de bloque, es decir, que requiera más de un línea.

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

```
Ejemplo de la documentación de una variable:
/**
*   Comentario u objetivo de la variable
*   @var string
*/
private $variable_text_msg;

Ejemplo de la documentación de una función:
/**
*   Comentario u objetivo de la función
*   @param $message
*   @return void
*/
public function __construct($message){
    //          TODO
}
```

3.3 Tratamiento de Errores

Para garantizar un correcto funcionamiento de un sistema se hace necesario identificar y controlar las posibles vías por las cuales pudiesen ocurrir errores, una de las principales vías son los formularios por los cuales los usuarios introducen datos que luego serán almacenados en la base de datos del sistema. Es por esto que se hace de vital importancia controlar todos los posibles errores que pudiesen ocurrir en un determinado entorno.

El sistema en desarrollo presenta varias características que rigen el comportamiento del tratamiento de errores, una la constituye el hecho de que no existen entradas que sean escritas por el usuario, aspecto favorable pues la información que se maneja ya ha pasado por varias validaciones por otros componentes del sistema como un todo que garantizan hasta cierto punto una entrada correcta de datos, lo cual se fortalece al usar para validar los datos que serán insertados en la base de datos los formularios de Symfony, los cuales presenta una estructura que permite la validación de los datos. Por otro lado tiene la característica de que sus principales procesos funcionan de manera automática donde no existe un usuario presenciando la ocurrencia del error y repitiendo la operación de esta no ser satisfactoria, es por esto que se hace de suma importancia el manejo estricto de todos los posibles casos de ocurrencias de errores y brindarles un correcto manejo.

En el sistema en desarrollo se hace un uso intensivo del tratamiento de excepciones otorgado por el lenguaje de programación usado (php), el cual brinda la posibilidad de capturar excepciones ocurridas por entrada de datos erróneos u ocurrencias de errores en tiempo de ejecución. Un correcto uso de esta estructura permite manejar de manera conveniente los errores que ocurran e impidiendo que el sistema deje de funcionar en algún momento, aspecto que contribuye de forma favorable a la disponibilidad del mismo.

Los errores ocurridos son registrados en los archivos log que presenta el marco de trabajo Symfony haciendo uso de una estructura otorgada por este la cual hace el manejo de estos registros una tarea sencilla, de esta forma queda registrados para un futuro análisis los eventos ocurridos.

3.4 Diagrama de Despliegue

El diagrama de despliegue (**Figura. 3.1)** es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo.

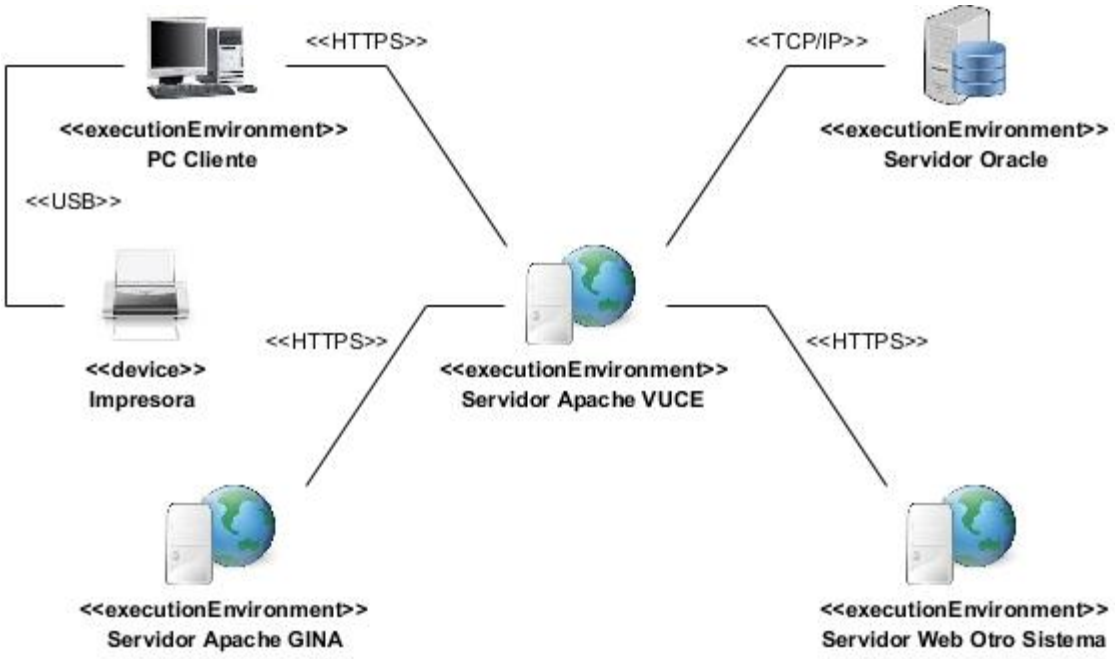


Fig. 3.1 Diagrama de despliegue.

Nodo Servidor Oracle:

Representa el servidor de bases de datos Oracle, en el cual se encuentran almacenados los datos del sistema VUCE.

Nodo PC Cliente:

Representa los ordenadores personales con los cuales los usuarios del sistema harán acceso al sistema VUCE.

Nodo Impresora:

Representa las impresoras de los usuarios que acceden al sistema pues en ocasiones se emiten documentos los cuales pueden ser impresos.

Nodo Servidor Apache VUCE:

Representa el servidor web donde se encontrará la aplicación del sistema VUCE en la cual se encuentra la solución abordada en el presente trabajo.

Nodo Servidor Apache GINA:

Representa el servidor web del sistema GINA con el cual el sistema VUCE llevará a cabo el intercambio de información mediante la solución descrita en el presente trabajo.

Nodo Servidor Web Otro Sistema:

Representa otro servidor web de un sistema con el cual la VUCE puede tener intercambio de información.

3.5 Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos y sus realizaciones en el entorno de un sistema de software, muestra como este es dividido en componente, así como la relación que existe entre ellos. Un componente se corresponde con una o más clases, interfaces o colaboraciones. A continuación se presenta en la **Figura. 3.2** el diagrama de componentes de la solución, en el mismo se representan los principales componentes por los que está formada la solución dentro del paquete sfVuBufferPlugin y otros componentes con los que se relacionan.

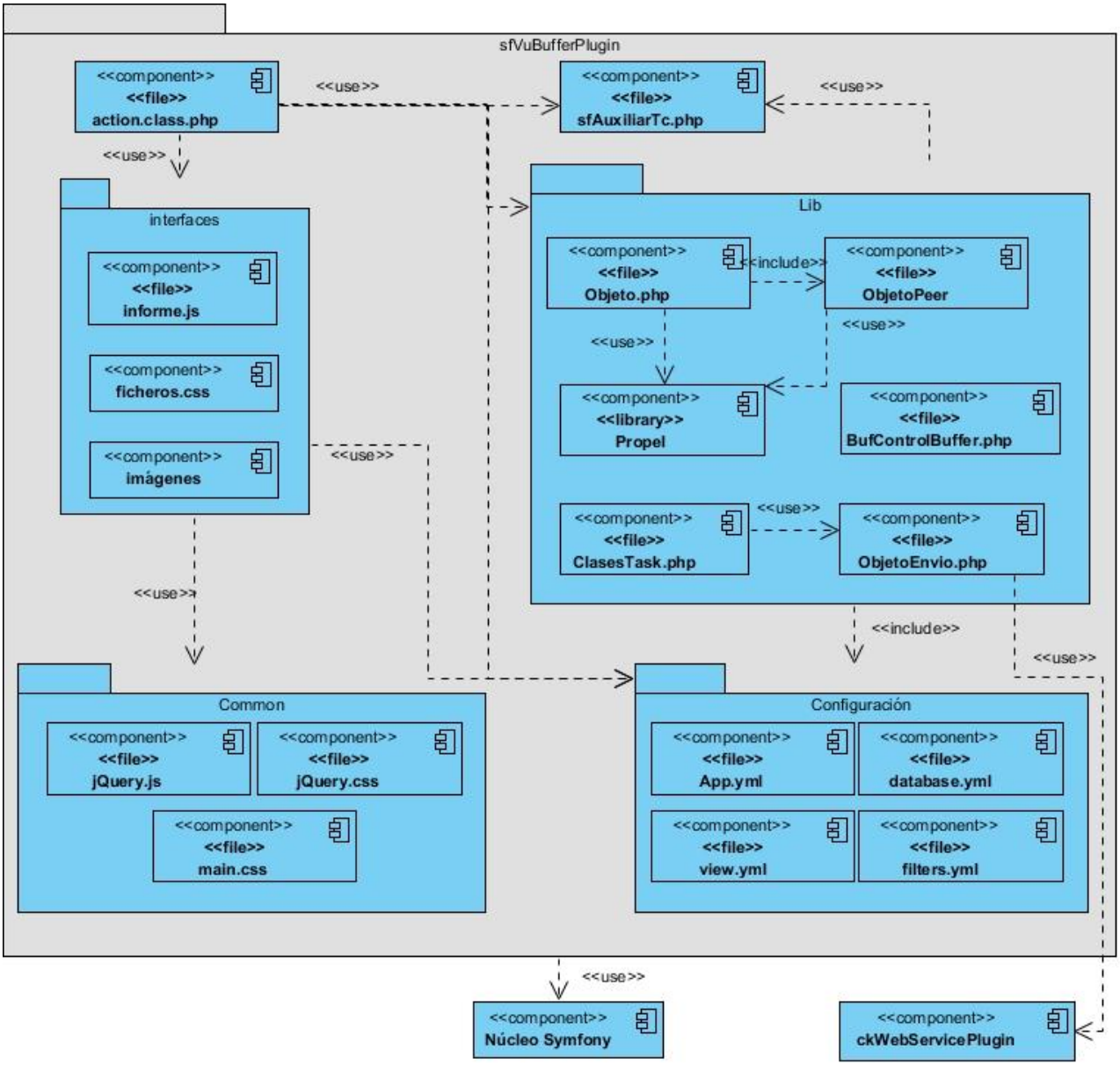


Figura. 3.2: Diagrama de componentes.

3.6 Características de la solución

La solución obtenida en el presente trabajo será integrada a la aplicación de la VUCE como un plugin que aportará a la misma la capacidad para realizar el intercambio de información con sistemas externos a esta. El mismo dentro de la aplicación para un correcto funcionamiento presenta varias dependencias las cuales son descritas a continuación:

- Debe existir instalado y habilitado el plugin para el manejo de servicios web ckWebServicePlugin.
- Debe existir integrado el componente de tablas de control (TC).
- Deben existir las entidades pertenecientes al modelo de datos de TC “TC_SISTEMA_EXTERNO” y “TC_TIPO_DOCUMENTO”, además de “NOT_DOCUMENTO” perteneciente al modelo de datos del módulo de notificaciones. Estas entidades no pertenecen al modelo del plugin pero son usadas por el mismo para satisfacer sus necesidades, por lo que de ser implantado en otra aplicación en el futuro se deberán tener en cuenta estas dependencias.
- Symfony deberá tener como ORM[18] a Propel 1.6 o superior debido a ciertos aspectos que son usados de este que no se encuentran en versiones anteriores, de hacerse necesario deberán ser re implementadas las funcionalidades que presenten dichas dependencias.

3.6.1 Configuración

Symfony establece para los plugins que sean instalados en un proyecto ciertos archivos que son reconocidos como la configuración del mismo, así como es también necesario configurar la aplicación para que active el plugin instalado. La solución que se describe en el presente trabajo presenta una estructura de carpetas mostrada en la **figura 3.3** en la cual en la carpeta config se localizan los ficheros de configuración.

[18] Del inglés Object Relational Model

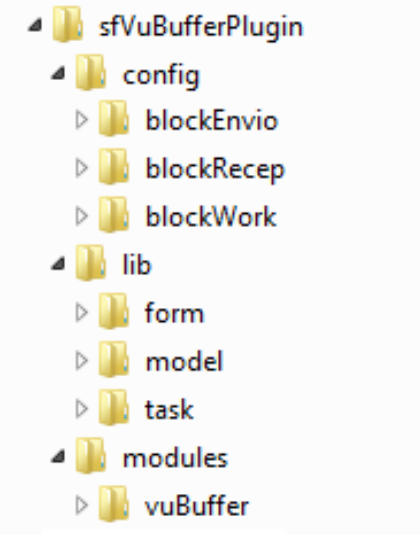


Figura. 3.3: Estructura de carpetas del plugin.

A continuación se describen los archivos que constituyen la configuración de la solución elaborada.

- **app.yml**: Contiene la configuración específica de la aplicación, variables globales que son utilizadas en el negocio de la aplicación, etc.
- **factories.yml:** Symfony incluye sus propias clases para el manejo de la vista, de las peticiones, de las respuestas, de la sesión, etc. No obstante, es posible definir otras clases propias para realizar estas tareas.
- **schema.yml:** Contiene la representación del modelo de datos relacional de la aplicación. Este archivo es opcional, ya que toda la información puede estar almacenada en el archivo a nivel de proyecto.
- **filters.yml:** En este archivo se definen los filtros que se van a procesar. En los plugins este archivo no se puede guardar en su propia estructura, se hace necesario incluir los filtros en el archivo del mismo nombre de la aplicación.

La solución cuenta con un módulo para el manejo de las funciones publicadas por el servicio web y el intercambio de datos con la vista, este módulo cuenta con un archivo (**module.yml**) que le permite establecer la configuración de las acciones y los parámetros específicos del módulo.

Los datos almacenados en estos archivos de configuración pertenecientes al plugin y guardados en su estructura son añadidos a los pertenecientes a las aplicaciones donde se use el mismo, de esta forma se pueden definir de manera separada y usarse en el momento de ejecución de la aplicación.

En la carpeta config perteneciente a la solución desarrollada se encuentran tres directorios encargados de almacenar los archivos de bloqueo para las tareas de envío y recepción de información a sistemas externos (blockEnvio, blockRecep, blockWork). Estos en conjunto a la clase BufBloqueo tienen la tarea de crear un sistema de banderas que permita la ejecución única en un espacio de tiempo de una tarea tanto de envío como recepción hacia o desde un sistema determinado, no permitiéndose la ejecución múltiple de estas para un mismo sistema externo. Del mismo modo la tarea encargada de ejecutar el intercambio de información posee un archivo de bloqueo para las ejecuciones secuenciales de las tareas, esto está dado debido a que se puede configurar el plugin para que ejecute la tareas tanto de manera secuencial como concurrente.

Un aspecto importante en el funcionamiento de la solución elaborada es la generación del archivo WSDL para el trabajo con los servicios web, elemento tratado en el epígrafe a continuación.

3.6.2 Generación del WSDL para el servicio web

El archivo WSDL que describe los servicios web que serán publicados por la solución informática de la presente investigación es generado por el plugin de Symfony ckWebServicePlugin descrito en el capítulo 1. Este cumple con los estándares establecidos por la WS-I, usando como estilo al generar los WSDL el rpc/literal. El uso de este estilo permite que los servicios web publicados puedan ser consumidos por aplicaciones que se encuentren desarrolladas en Java, .Net y el propio PHP que es el propio lenguaje de la solución en elaboración. En el **Anexo 1** se muestra el archivo generado por este plugin, el mismo dentro de la aplicación se localizará en la carpeta web donde se encuentran los archivos que son accedidos directamente desde la web, de esta forma se puede establecer el acceso al mismo para consumir los servicios publicados.

Para la publicación de una funcionalidad en un servicio web basta con tan solo en el Action de un módulo perteneciente a una aplicación o plugin crear encima de la declaración de una acción un comentario especificando los datos necesarios por el plugin ckWebService. Un ejemplo de esta estructura se presenta a continuación.

```
/**
     * Action del servicio web.
     * @WSMethod(name='ProcesarDocumento', webservice='vuBuffer')
     *
     * @param string $codigo código del sistema
     * @param string $documento documento en json
     *
     * @return string The result
*/
```

De esta forma publicar una funcionalidad y generar el WSDL se hace muy sencillo, esta es una de las facilidades que ofrece ckWebService para el trabajo con los servicios web. Para lograr mayor comodidad en cuanto a la generación de este archivo, la solución desarrollada en la presente investigación cuenta con una tarea para la generación del mismo con el objetivo de publicar las funcionalidades descritas en el módulo vuBuffer perteneciente a la solución. Basta con ejecutar el siguiente comando en la consola del sistema: php symfony buffer:generarWSDL aplicación http://ruta_hasta_web.

De esta forma el trabajo con la solución por parte del personal que le dará mantenimiento será más ameno al abstraerse del flujo interno de acciones que esto conllevarían de otro modo.

3.7 Validando la solución obtenida

El Centro de Calidad para Soluciones Informáticas (CALISOFT) en la Universidad de las Ciencias Informáticas (UCI) es el encargado de proveer las métricas para definir la calidad de los productos de software desarrollados en la misma. Este centro tiene definido para la evaluación de la interoperabilidad, variable establecida en el problema de la investigación, dos métricas que evalúan su cumplimiento según los factores de calidad de acuerdo a la ISO 9126. Una de ellas es para evaluarla de acuerdo a: la **intercambiabilidad de datos en base a su formato**, la cual no provee una buena representación pues en la solución definida se manejan todos los mensajes de manera indiferente mediante el método establecido, y la otra es la **intercambiabilidad de datos, en base al éxito del intento**, para la cual se tomarán los datos obtenidos en pruebas hechas a las funcionalidades que representan un acoplamiento directo al envío y recepción de información.

Seguidamente en la **Tabla 3.1** se presentan las principales características de esta métrica.

Tabla. 3.1 Métrica establecida por CALISOFT para la interoperabilidad.

Nombre	Definición	Metas	Procedimiento de análisis
Intercambiabilidad de datos, en base éxito del intento. (INTEROPERABILIDAD)	1-) X = 1 - A/B A – Número de casos en que falló el proceder a un intercambio de datos con otros software o sistemas. B – Número de casos en que se intentó proceder a un intercambio de datos. 2-) Y = A/T T – Período de tiempo de operación.	¿Cuán frecuentemente es “satisfactoria” la transferencia de datos entre el software objeto de la prueba y otro?	Ejecutar las pruebas. Cuente el número de casos en que las funciones de interfaces fueron usadas y fallaron. 0 <= X <= 1 A mayor cercanía al 1 resultará mejor 0 <= Y A mayor cercanía al 0 resultará mejor
		¿Cuán frecuentemente falló el intento de intercambio de datos entre el software objeto de la prueba y otro software?	**Escala:** 1) Absoluta 2) Valorativa • Tipo de medida: X = cont / cont A = contable B = contable • Y = cont / tiempo T – tiempo

			Fuente: Especificación de requisitos. Manual del usuario. Informe de pruebas.

En las pruebas funcionales se definieron como objetivos las funcionalidades, enviar documento a sistema externo, recibir documento de sistema externo y enviar incidencia. Por cada una de las mismas se presentan los datos de las pruebas realizadas y los resultados obtenidos según lo planteado en la métrica.

Tabla. 3.2 Tabla con los resultados de aplicar las pruebas funcionales.

No.	Funcionalidad	Cant. Casos	Cant. Casos exitosos	Cant. Casos fallidos	Tiempo medio (ms)	Tiempo total (ms)	Cant / min
1	Enviar documento a sistema externo.	40	39	1	220 ms	8800 ms	4,43
2	Recibir documento de sistema externo.	40	40	0	189 ms	7560 ms	5,29
3	Enviar incidencia.	40	40	0	450 ms	18000 ms	2,22

De acuerdo a los datos anteriormente representados se obtuvieron los siguientes resultados al aplicar las fórmulas planteadas en la definición de la métrica. Tener en cuenta que los tiempos se llevaron de milisegundos (ms) a segundos (s) para una mejor comprensión de los resultados.

Tabla. 3.1 Tabla con los resultados de aplicar la métrica.

No.	Funcionalidad	Cant. Casos	Cant. Casos exitosos	Cant. Casos fallidos	T (s)	X = 1 - A/B	Y = A/T
1	Enviar documento a sistema externo.	40	39	1	8,80	0,98	0,114
2	Recibir documento de sistema externo.	40	40	0	7,56	1,00	0,000
3	Enviar incidencia.	40	40	0	18,00	1,00	0,000

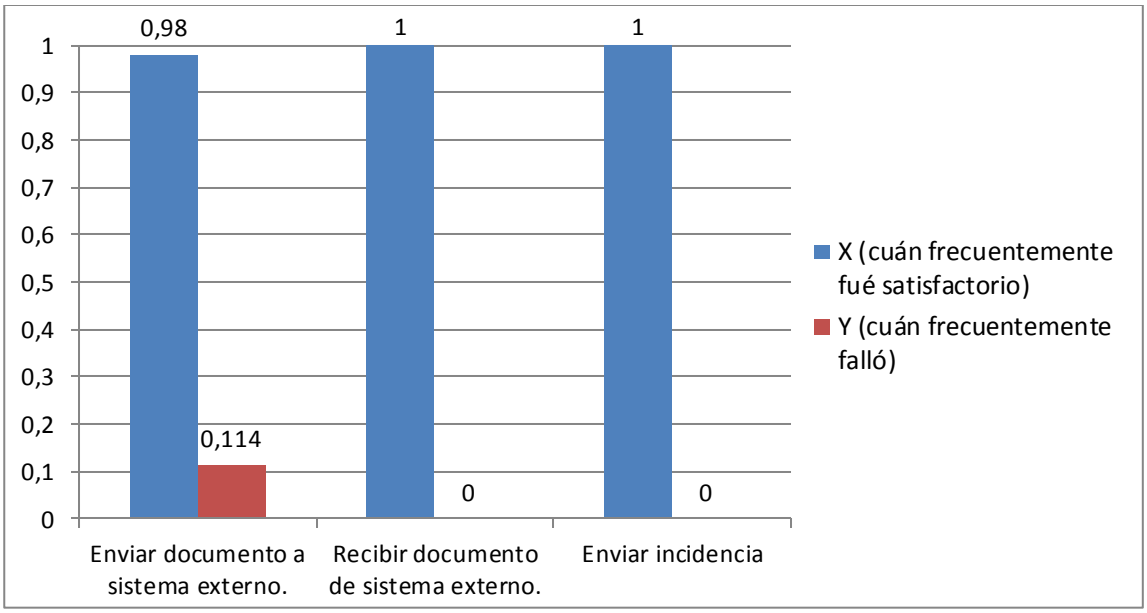


Fig. 3.4 Gráfica con los resultados de la métrica.

Como se puede apreciar en la **figura 3.4** los resultados son satisfactorios, obteniendo el máximo resultado posible en dos de las funcionalidades y un solo fallo en una única funcionalidad. Luego de la corrección de los errores que indujeron el fallo se logró obtener un resultado satisfactorio del 100%, demostrándose de esta manera que el sistema presenta excelentes indicadores de interoperabilidad.

3.8 Conclusiones del capítulo

En el presente capítulo se generaron los artefactos pertenecientes a la etapa de implementación como son el diagrama de despliegue y el diagrama de componentes. Definiéndose además los estándares de codificación usados para la implementación de la solución anteriormente modelada que permitieron crear un código de fácil entendimiento. De igual forma se analizaron los resultados obtenidos de las pruebas funcionales realizadas a 3 requisitos funcionales seleccionados con el objetivo de validar la solución obtenida. El análisis de estos datos arrojó resultados satisfactorios por lo que concluye que la solución desarrollada se encuentra en condiciones para ser implantada.

CONCLUSIONES

Al finalizar la presente investigación se logró cumplir con los objetivos planteados al inicio de la misma, obteniéndose como resultado una solución informática que permite el intercambio de información entre la VUCE y el GINA, permitiendo la comunicación con otros sistemas que sean incorporados en el futuro. Para el desarrollo de la misma se hizo necesaria la realización de un estudio de varios de los Sistemas de VUCE existentes en el mundo, no obteniéndose de estos resultados relevantes. Debido a esto se analizaron las principales técnicas de integración de datos, obteniéndose de la Integración de Aplicaciones Empresariales (EAI) las fortalezas que permitieron la elaboración de la solución.

El estudio brindó como resultados la necesidad del uso de los servicios web para lograr establecer una solución adaptable a sistemas de diferentes plataformas y elaborados en diferentes lenguajes. Culminado el estudio del estado del arte se definieron las herramientas, tecnologías y metodologías a utilizar para el desarrollo de la solución informática.

Posteriormente fueron identificadas las principales funcionalidades de la solución. Teniéndose como base el análisis realizado anteriormente, se analizaron y seleccionaron los patrones de diseño a utilizar en la solución planteada y se diseñaron los diagramas establecidos en el flujo de implementación que dieron paso a la programación de la solución.

El correcto funcionamiento de la solución desarrollada fue validado mediante el análisis de los resultados obtenidos de pruebas funcionales realizadas al sistema con las métricas definidas por CALISOF, obteniéndose resultados satisfactorios. La realización de estas pruebas permitió además encontrar y corregir funcionalidades defectuosas en la aplicación.

El desarrollo de la solución informática obtenida como resultado de la presente investigación proveerá al sistema VUCE de los elementos necesarios para garantizar la interoperabilidad con el sistema GINA, proveyendo soporte para la comunicación con otros sistemas externos. Garantizando de esta manera que se lleven a cabo exitosamente los procesos definidos entre ambos sistemas. Todo esto aportará a que se mejoren los procesos del comercio exterior en Cuba, permitiéndole alcanzar un mayor grado de competitividad a nivel mundial en este sector.

RECOMENDACIONES

Los objetivos trazados en la presente investigación fueron alcanzados satisfactoriamente, sin embargo se recomienda:

- Desarrollar nuevas funcionalidades para la generación de reportes si en el futuro son necesarios.
- Continuar el estudio de los estándares relacionados con los servicios web con el objetivo de proveer a la solución de las mejores prácticas en este sector.

ANEXOS

Anexo 1

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" name="vuBuffer"
targetNamespace="http://localhost/VU/web/" xmlns:tns="http://localhost/VU/web/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
 <wsdl:types xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace="http://localhost/VU/web/"/>
 </wsdl:types>
 <wsdl:portType name="vuBufferPortType">
  <wsdl:operation name="ProcesarDocumento" parameterOrder="codigo documento">
   <wsdl:input message="tns:ProcesarDocumentoRequest"/>
   <wsdl:output message="tns:ProcesarDocumentoResponse"/>
  </wsdl:operation>
 </wsdl:portType>
 <wsdl:binding xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" name="vuBufferBinding"
type="tns:vuBufferPortType">
  <soap:binding xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" name="ProcesarDocumento">
   <soap:operation xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
soapAction="http://localhost/VU/web/ProcesarDocumento" style="rpc"/>
   <wsdl:input xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
    <soap:body xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" parts="codigo documento"
use="literal" namespace="http://localhost/VU/web/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
   </wsdl:input>
```

```
    <wsdl:output xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
     <soap:body xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" parts="result" use="literal"
namespace="http://localhost/VU/web/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </wsdl:output>
   </wsdl:operation>
  </wsdl:binding>
  <wsdl:message name="ProcesarDocumentoRequest">
   <wsdl:part name="codigo" type="xsd:string"/>
   <wsdl:part name="documento" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="ProcesarDocumentoResponse">
   <wsdl:part name="result" type="xsd:string"/>
  </wsdl:message>
  <wsdl:service xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" name="vuBufferService">
   <wsdl:port xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" name="vuBufferPort"
binding="tns:vuBufferBinding">
    <soap:address xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
location="http://localhost/VU/web/vuBuffer.php"/>
   </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Anexo 1: Código del archivo WSDL del servicio web del plugin sfVuBufferPlugin.

BIBLIOGRAFÍA

1. Aduana General de la República de Cuba. [En línea] [Citado el: 12 de 12 de 2011.] http://www.aduana.co.cu.

2. VUCE - Ventanilla Única de Comercio Exterior Perú. [En línea] 25 de 01 de 2012. https://www.vuce.gob.pe/resena.html.

3. *Una experiencia en integración de aplicaciones empresariales.* **Azán Basallo, Yasser, Díaz Estrada, Anay y González Gómez, Salvador.** La Habana : s.n., 2009.

4. *Enterprise Information Integration: A Pragmatic Approach.* **Morgenthal, JP.** 2006.

5. [En línea] [Citado el: 13 de 02 de 2012.] http://www.w3c.es/divulgacion/guiasbreves/ServiciosWeb.

6. **Christensen, Erik , y otros, y otros.** W3C. [En línea] 15 de 03 de 2001. [Citado el: 14 de 02 de 2012.] http://www.w3.org/TR/wsdl.

7. **González C., Benjamín .** Desarrolloweb. [En línea] 03 de marzo de 2012. http://www.desarrolloweb.com/articulos/1557.php.

8. W3C. [En línea] 24 de enero de 2012. [Citado el: 03 de marzo de 2012.] http://www.w3.org/XML/.

9. **Díaz Estrada, Anay y González Morales, Andrés Bernardo.** *Propuesta de integración de los subsistemas del Sistema Informático de Gestión de Auditoría y Control (SIGAC).* La Habana : s.n., 2009.

10. PHP Webservices. [En línea] [Citado el: 13 de 03 de 2012.] http://phpwebservices.blogspot.com/2008/01/need-for-ws-security.html.

11. **López, Manuel J. Lucena.** *Criptografía y Se guridad, 4ª Edición. Versión 0.7.0.*

12. **Angel Angel, José de Jesús.** *AES - Advanced Encryption Standard. Versión 2005, principiantes.* 2005.

13. **Alonso Driggs, Idalberto Carlos.** *Modelo de Desarrollo de Software para el Departamento de Soluciones para Aduana del CEIGE.*

14. **Ing. Rodríguez Andrés, Rolando.** GestioPolis. [En línea] 16 de junio de 2008. [Citado el: 01 de marzo de 2012.] http://www.gestiopolis.com/administracion-estrategia/lenguajes-notaciones-y-herramientas-en-analisis-de-procesos.htm.

15. **Cedeño, K y Vega, Y.** *Propuesta de un procedimiento para el proceso de elicitación y control de los requisitos en el proyecto ICICV.* 2008.

16. NetBeans. [En línea] [Citado el: 03 de marzo de 2012.] http://netbeans.org/features/index.html.

17. **Cobo Rodríguez, José Antonio.** *Línea Base Arquitectónica para el Polo Sistemas Tributarios y de Aduanas.* La Habana : s.n., 2008.

18. [En línea] [Citado el: 13 de 02 de 2012.] https://github.com/christiankerl/ckWebServicePlugin.

19. **Sommerville, Ian.** *Ingeniería del Software.* Madrid (España) : Pearson Educación, 2005. Vol. 7. 84-7829-074-5.

20. **Martínez, Ing. Jenni Manso.** *Procedimiento para la Ingeniería de Requisitos en el Departamento de Desarrollo de Soluciones para la Aduana del CEIGE.* La Habana: Universidad de las Ciencias Informáticas : s.n., 2010.

21. **Pressman, Roger S.** *Ingeniería del Software. Un enfoque práctico. (Sexta Edición).* 2005. ISBN: 9701054733.

22. **Larman, Craig.** *UML y Patrones.* 1999. ISBN 970-1 7-0261-1.

23. —. *UML y Patrones, Introducción al Análisis y Diseño Orientado a Objetos.* Montevideo Uruguay : Prentice Hall, Pearson, 2001. 053681.

24. **Fowler, Martin.** *Patterns of Enterprise Application Architecture.* ISBN: 0-32112-742-0.

25. **Potencier, F.a.Z, Francois.** *The definitive Guide to Symfony.* s.l. : Apress, 2007.

26. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* s.l. : Addison Wesley, 1999. ISBN: 0-201-57169-2.

27. **Booch, Grady, Rumbaugh, James y Jacobson, Ivar.** *El Lenguaje Unificado de Modelado Manual de Referencia.* s.l. : Addison-Wesley.

28. Ventanilla Única México. [En línea] [Citado el: 12 de 12 de 2011.] https://www.ventanillaunica.gob.mx/vucem/SobreVU/SobrelaVU/index.htm.

29. VUCE - Ventanilla Única de Comercio Exterior Perú. [En línea] 2010. [Citado el: 12 de 12 de 2011.] https://www.vuce.gob.pe/index2.html.

30. *La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software.* **Arias Chaves, Michael.** ISSN 1409-4746, Universidad de Costa Rica : Revista InterSedes, 2005, Vol. VI.

31. *Ingeniería de Requisitos en Aplicaciones para la Web - Un estudio comparativo.* **José Escalona, María and Koch, Nora.** Universidad de Sevilla: Departamento de Lenguajes y Sistemas Informáticos : s.n., 2002.

32. IEEE Standard Glossary of Software Engineering Terminology. [En línea] 1990. [Citado el: 13 de 02 de 2012.] http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=2238. 610-12-1990.

33. **Garlan, David y Shaw, Mary.** An introduction to Software Architecture. s.l. : CMU Software Engineering Institute Technical Report, 1994. ESC-TR-94-21. CMU/SEI-94-TR-21.

34. **Ruiz Guerra, Boris Enrique.** *Tránsito y Transferencia para el Despacho Comercial en la Aduana General de la República de Cuba.* La Habana : s.n., 2010.

35. Symfony. [En línea] [Citado el: 03 de marzo de 2012.] http://symfony.com/license.

36. Librosweb. [En línea] [Citado el: 10 de 04 de 2012.] http://www.librosweb.es/symfony_1_2.

37. Web Services Interoperability Organization. [En línea] [Citado el: 12 de 03 de 2012.] http://www.ws-i.org/.

38. **de la Torre, César, y otros, y otros.** *Guia de arquitectura n-capas orientada al dominio con .net.* s.l. : Krasis press.

39. **Welicki, L.E.** *Meta-Especificación y Catalogación de Patrones de Software con Lenguajes de Dominio Específico y Modelos de Objetos Adaptativos: Una Vía para la Gestión del Conocimiento en la Ingeniería del Software, en Facultad de Informática.* Universidad Pontificia de Salamanca, Madrid : s.n., 2006.

40. **Welicki, León.** Patrones y Antipatrones: una Introducción - Parte II. [En línea] http://msdn.microsoft.com/es-es/library/bb972251.aspx.