**Universidad de las Ciencias Informáticas**

**FACULTAD 3**

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

# Título: Componente para el intercambio de información electrónica en la Aduana General de la República de Cuba.

**Autor:** Roberto Nuñez Corrales

**Tutor:** MSc**.** Julio Cesar Diaz Vera

La Habana, junio del 2012

"Año 54 de la Revolución"

## DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Departamento de Soluciones para la Aduana de la Universidad de las Ciencias Informáticas hacer uso del mismo en su beneficio.

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

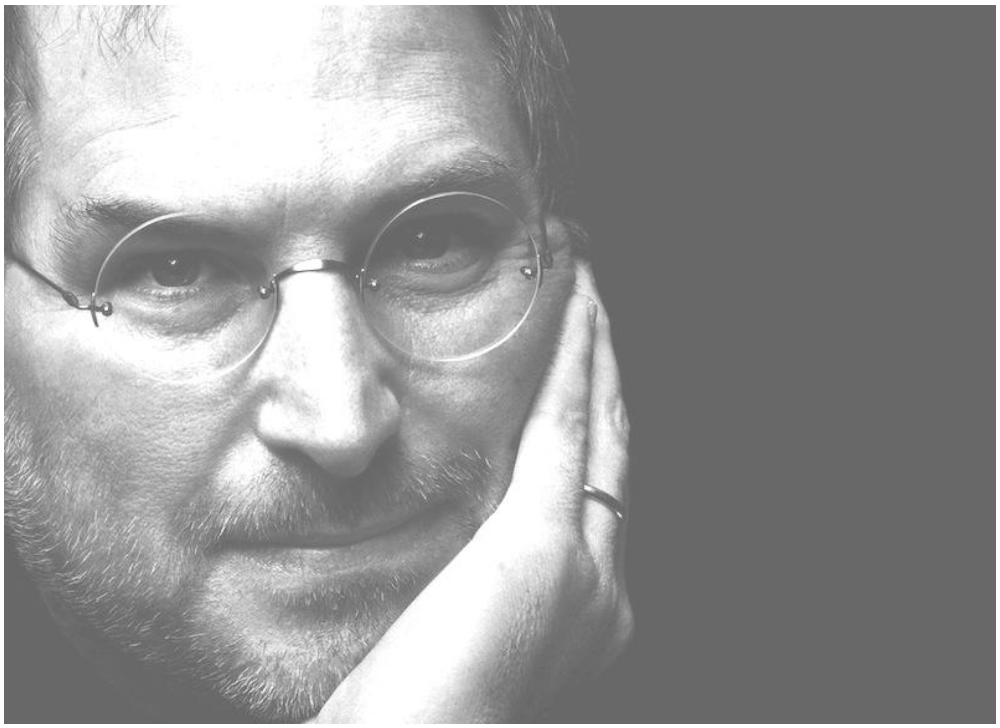
Roberto Nuñez Corrales

_____

MSc. Julio Cesar Diaz Vera

_____

**FRASE**



**"Tu tiempo es limitado, de modo que no lo malgastes viviendo la vida de alguien distinto. No quedes atrapado en el dogma, que es vivir como otros piensan que deberías vivir. No dejes que los ruidos de las opiniones de los demás acallen tu propia voz interior. Y, lo que es más importante, ten el coraje para hacer lo que te dicen tu corazón y tu intuición. "**

**"A veces cuando se innova, se cometen errores. Es mejor admitirlo rápidamente y continuar con otras innovaciones."**

**Steve Jobs**

## AGRADECIMIENTOS

**Agradecer a mi madre y mi padre por el apoyo incondicional durante toda mi vida y más en esta etapa.**

**A toda mi familia, que de una forma u otra siempre me han dado aliento en cada situación.**

**Mi novia por bridarme su compañía y apoyo estos años.**

**A mi tutor un agradecimiento infinito, pues es más que un tutor es un amigo sin el cual no hubiese sido de la misma forma mi desarrollo como ingeniero y la realización de este trabajo final.**

**Mis amigos, que han sido mi familia durante cinco arduos años en la universidad y por dedicarme gran parte de su tiempo.**

## DEDICATORIA

**Dedico este trabajo a todo aquel que siempre confió en que podía cumplir este sueño.**

**La dedicatoria más especial que puedo hacer es para mis padres y mi familia, sin ellos no creo que fuese posible realizarme como ingeniero y como persona.**

# RESUMEN

El intercambio de información es una de las tareas más comunes que se realizan en el mundo actual, Cuba no está ajena al uso de las nuevas tecnologías, por lo que está enfrascada en una batalla por informatizar las entidades estatales. La Aduana General de la República en convenio con la UCI desarrolla un sistema para informatizar los procesos aduanales, entre los que se encuentra el intercambio de información con entidades externas. Para la solución del problema identificado en la AGR referido a este proceso, se decide crear un componente que gestione el intercambio de información en la entidad, componente que será integrado en el sistema GINA desarrollado por la UCI. Para la creación del mismo se realizó un estudio de diferentes conceptos que sustentarían la investigación, la creación de la solución estuvo guiada por la metodología de desarrollo definida por el Departamento de Soluciones para la Aduana, se generaron los artefactos necesarios para cumplimentar la solución, se creó un componente arquitectónicamente integrable en el sistema y se validó la solución utilizando diferente técnicas que dieron como resultado un producto que cumplió todas las demandas del cliente.

**Palabras claves:** Intercambio de información, informatizar, componente, metodología, artefactos.

# CONTENIDO

# CONTENIDO DE FIGURAS

# INTRODUCCIÓN

La Aduana General de la República (AGR), como institución de Cuba, forma parte del sistema de control estatal y actúa en función de ello, llevando a cabo la conciliación de las medidas de control con las de facilitación del comercio tanto exterior como interior.

Dicha entidad como el órgano encargado de regir el control en materia aduanera, está designada a recaudar los derechos de aduana, dar respuesta dentro de su jurisdicción y competencia a los hechos que incidan en el tráfico internacional de mercancías, viajeros, postal y los medios que los transportan (1). Para realizar estas funciones la aduana necesita intercambiar información con personas naturales o jurídicas.

Las TIC (Tecnologías de la Información y las Comunicaciones) acuñadas en la década del 80 como concepto que engloba a un grupo de tecnologías que han permitido a la humanidad alcanzar grandes avances en las comunicaciones y la automatización de actividades entre otras áreas; tienen como objetivo central: gestionar información, trasladarla y compartirla entre personas o aplicaciones.

La Organización Mundial de Aduanas (OMA) es la única organización intergubernamental representativa de las aduanas. Dicha organización dejó plasmado en el convenio de Kioto (2) y las normativas de la OMA la necesidad de automatizar los procesos aduanales. Entre las políticas referidas en ambos documentos se encuentra la necesidad de contar con la información asociada a cargas y pasajeros con anticipación a su arribo, para poder utilizarla en la definición del tratamiento que se le asignará, es imposible cumplimentar esta necesidad sin la utilización de documentos digitales.

Entre las metas trazadas por la AGR se encuentra la de transformarse en una entidad de cero papel. Esto también exige de la informatización de procesos y la digitalización de los documentos vinculados en estos procesos. En aras de contar con información digital y cumplir con lo pactado, la AGR requiere de mecanismos que les permitan interactuar, de forma natural, con una gran cantidad de personas ya sean naturales o jurídicas y procesar una amplia variedad de formatos de documentos.

La interacción con entidades externas mediante la recepción de documentos o con el propio sistema es una de las funcionalidades indispensables para la informatización de los procesos en la AGR, por solo

mencionar algunas se destacan la necesidad de contar con la información adelantada de pasajeros (API) y las declaraciones de mercancías (DM) entre otras.

Para el procesamiento de la información en la AGR existen diversos tipos de ficheros, formatos y extensiones de acuerdo con el tipo de información que contenga el mismo, ejemplo de ellos es el caso de la exportación de las tablas de control[1] en formato **dbf**, las Declaraciones de Mercancías[2](DM) de tipo tránsito con formato **dmt**, las DM de tipo transferencia[3] **dmtx**, el fichero de Información adelantada de pasajeros [4](API) con extensiones .**api** o .**txt**.

Ninguno de los documentos antes mencionados está estandarizado y la forma de procesamiento por el sistema informático es totalmente diferente en cada caso, por lo que asimilar un nuevo fichero trae consigo una serie de actividades de desarrollo, que pueden posponer hasta en 5 meses la introducción de los datos referente al fichero para su posterior procesamiento. Existen documentos con diferencias estructurales, unos más complejos que otros, como son los casos las DM los cuales independientemente de la complejidad necesitaron tiempos extremos para su procesamiento. La AGR no recibe documentos provenientes de entidades externas solamente, sino que también crea documentos y los envía a las mismas, dichos documentos presentan los mismos problemas de estructura, extensión o están carentes de estandarización, por lo que las entidades deben lidiar con cada uno de estos documentos de forma particular.

La Universidad de Ciencias Informáticas (UCI) ha estado a la vanguardia en la aplicación de las TIC y en la preparación del personal calificado para hacer frente a este concepto. En el año 2003 la AGR establece relaciones con la UCI para desarrollar la informatización de los diversos procesos que se llevan a cabo en esa entidad. De esta cooperación nace el proyecto nombrado **SUA** (Sistema Único de Aduanas) que posteriormente cambia su nombre por el que se rige en estos momentos **GINA** (Gestión Integral de Aduanas), con el objetivo de desarrollar una aplicación que supla los puntos débiles del sistema de la AGR y posibilite informatizar otros procesos de manera eficiente.

---

[1] **Tablas de control (TC):** Tablas que contienen los metadatos o nomencladores comunes en la AGR.
[2]**Declaración de Mercancía (DM):** Documento procesado por la AGR, el cual contiene los datos de mercancía a procesar
[3] **Declaración de Mercancía de tipo tránsito (DMT):** Documento que contiene mercancías pertenecientes al tránsito.
[4]**Información adelantada de pasajeros (API):** Documento que se emite antes de realizarse un vuelo, conteniendo toda la información referida al mismo.

A partir de la situación anteriormente expuesta se define como **problema a resolver:** ¿Cómo mejorar el tiempo de implantación de los proyectos basándose en la recepción y/o validación de los documentos electrónicos en el sistema **GINA**?

Para la posterior solución de lo planteado se define como objeto de estudio: **La integración de datos.**

Se establece como campo de acción: **Proceso de recepción y envío de documentos electrónicos en la AGR.**

Como **objetivo general** se define: **Desarrollar un componente para la integración de datos, recepción y envío de documentos electrónicos en la AGR.**

Como objetivo central de la investigación se pretende realizar la modelación e implementación de una plataforma o componente que permita el intercambio de información entre las personas ya sean jurídicas o naturales y la AGR.

Para lograr lo antes mencionado se identifican como **objetivos específicos:**

1. Establecer el marco conceptual de referencia.
2. Definir un estándar de intercambio de información.
3. Determinar los requerimientos del sistema.
4. Definir el modelo de paquetes.
5. Implementar el componente.
6. Validar el componente.

Con el fin de llevar a cabo la investigación y obtener los resultados esperados se conciben las siguientes **tareas de investigación**:

1. Recopilación de la bibliografía referente al tema.
2. Selección de la bibliografía.
3. Análisis de la bibliografía.
4. Selección de un lenguaje de intercambio.

5. Propuesta del formato de intercambio.
6. Identificación de los requisitos de usuario (siguiendo el procedimiento definido en el departamento).
7. Identificación requisitos cumplimentados en otro software de intercambio.
8. Desarrollo del modelo de especificación de requisitos.
9. Determinación de los componentes teniendo en cuenta la arquitectura del sistema.
10. Descripción formal de los componentes de acuerdo a los procedimientos del departamento.
11. Definición del modelo de diseño de clases de los componentes.
12. Desarrollo del modelo de implementación.
13. Definición de las pruebas de validación.
14. Implantación de las pruebas.
15. Presentación de los resultados.

El presente trabajo está estructurado en tres capítulos, cada uno de ellos cuenta con una introducción seguida de los epígrafes necesarios.

El capítulo 1 **Fundamentación teórica** agrupa la definición del marco conceptual y el estado del arte que presenta las soluciones informáticas a nivel nacional y mundial que se utilizan con propósitos similares.

El capítulo 2 **Descripción de la solución propuesta** contiene la definición del modelo de desarrollo que guiará la solución, se describirán los principales elementos que formarán la misma: requisitos funcionales, modelos de procesos, descripción de requisitos.

El capítulo 3: **Implementación y validación de la solución**, en este capítulo se expondrán las principales técnicas de validación, una explicación de cada una de ellas además de la explicación del modelo de implementación, seguido de las pruebas unitarias y funcionales realizadas para cumplimentar las técnicas de evaluación dinámica.

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

## 1.1 Introducción

El intercambio de información es una de las actividades más remotas y necesarias que lleva a cabo la humanidad. En la sociedad esta actividad ha ido ganando cada vez más espacio, el ámbito empresarial es una de las áreas donde intercambiar información es vital para obtener resultados satisfactorios, la sociedad actual es denominada **sociedad de la información** debido a la diversidad de actividades y volúmenes de información que se manejan en el día a día en todas sus áreas.

Cada día los volúmenes de información que debe manejar una empresa crecen exponencialmente siendo imposible el procesamiento manual de los mismos, haciéndose necesario la introducción de nuevas vías y conceptos para manejar eficazmente la información y obtener los resultados deseados. Las TIC son una vía fundamental para llevar la información a los lugares más recónditos y desarrollar tareas antes no pensadas, teniendo presente la eficiencia que conlleva el uso de dichas tecnologías.

Hoy en día el intercambio de datos es un proceso donde se tiene de manera inevitable la intervención de las TIC, donde se maneja la **integración de datos** y la **interoperabilidad de datos** como conceptos fundamentales, se utiliza estándares para la comunicación y además se definen mecanismos y lenguajes con los que se pueda establecer la comunicación entre las partes.

En este capítulo se presenta el marco conceptual para este trabajo, así como las principales tendencias en el desarrollo de sistemas para el intercambio de información.

## 1.2 Interoperabilidad de datos

Según el Instituto de Ingenieros Electrónicos y Eléctricos (IEEE) en el glosario estándar de definiciones se define como **interoperabilidad**:

*"… la capacidad de dos o más sistemas o componentes de intercambiar información y manipular la información intercambiada…".* (3)

En la definición dada por el Laboratory of Interactive and Cooperative Technologies (ICT) se entiende por **interoperabilidad**:

*"… todo aquel proceso que tiene como meta el compartir recursos homogéneos y heterogéneos (distintos formatos, plataformas de hardware y software) utilizando un protocolo básico para interactuar con los datos…"* (4)

El Comité Técnico TC de la Organización Internacional de Estandarización (ISO) en la norma

ISO/DIS 19101 (ISO, 20/01) define:

*"… la capacidad de un sistema o componente de un sistema para proporcionar intercambio de*

*Información y el procesamiento cooperativo entre aplicaciones ".* (5)

De manera general todas estas definiciones hacen énfasis en la interacción o manipulación de datos externos que por ende deben ser mezclados con los propios y funcionar de manera homogénea.

La manera en que en la AGR se incorporan los datos que provienen de las fuentes externas, incluyéndolos en el sistema para su posterior utilización, no promueve la mezcla de los datos ajenos con los datos aduanales, de la misma forma son tratadas las personas o entidades externas. De esta forma el uso de mecanismos de interoperabilidad lejos de ser una ventaja se convertiría en una carga que no generaría beneficios para la solución y es por ello necesario estudiar otros mecanismos para el intercambio de información en aras de establecer el marco conceptual de la solución.

## 1.3 Integración de datos.

Para Mark R. Madsen (6) la integración de datos se centra en el manejo del flujo de datos y en proporcionar formas estandarizadas para acceder a la información. La integración de datos en pocas ocasiones estandariza las transacciones o los servicios que hacen las llamadas, por lo cual provee mejores procedimientos para manejar la información a través de los sistemas que interactúan.

Los especialistas de la empresa **Integra2** definen como integración de datos:

*"... el intercambio de información a través de medios telemáticos, entre los sistemas del cliente y los de proveedores. Pretende cubrir las necesidades que puedan surgir entre ambos, a través de diferentes sistemas de comunicaciones e informáticos…"* (7)

La integración de datos está definida por modelos, que es a su vez es donde se define como se llevará a cabo el tratamiento de los datos y las aplicaciones, mediante la especificación de la naturaleza y los mecanismos empleados en el proceso (8)

El profesor David Melendi Palacios (8) de la Universidad de Oviedo define como los principales modelos de integración:

**Modelo de integración de presentación (MIP):** A nivel de interfaz de usuario. Se basa en el acceso a las aplicaciones y datos a través de su lógica de presentación, desarrollándose una nueva interfaz de usuario. (9)

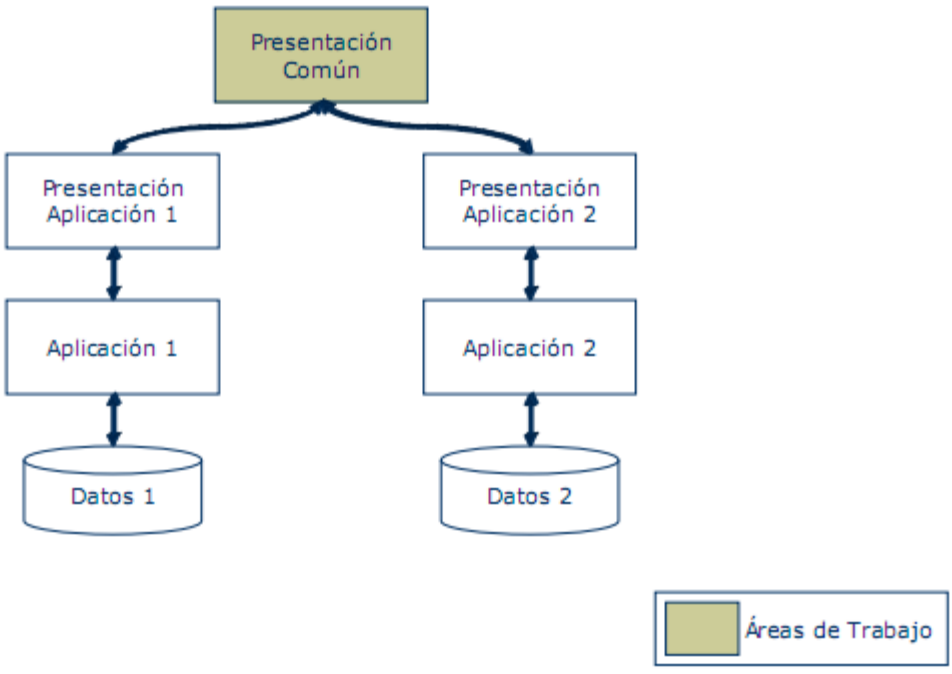


**Figura1: Integración de presentación. (9)**

Ejemplo de apliación:

Una empresa dispone de varias apliaciones web para la gestión de proyectos


**Figura2: Anterior a MIP (9)**

Desea disponer de una herramienta única para la gestión de sus proyectos.

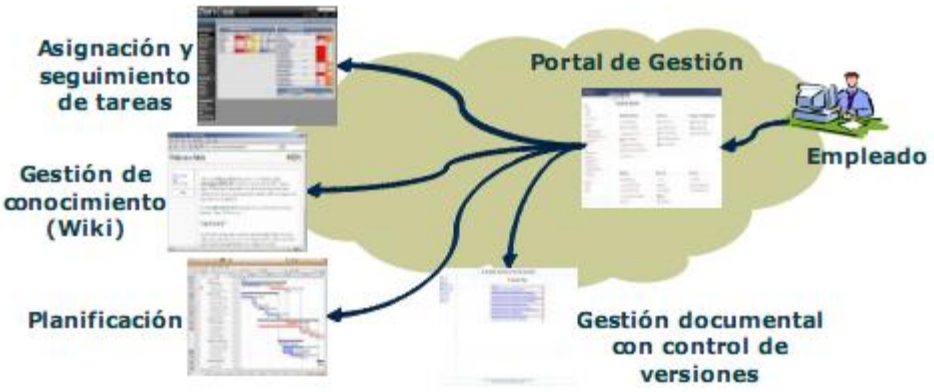



**Figura3: Posterior a MIP (9)**

Es viable utilizar este modelo de integración cuando se desea contar con una interfaz de comunicación para aplicaciones donde el acceso se realice mediante línea de comandos o la necesidad de integrar varias aplicaciones en una sola interfaz.

Este modelo puede presentar diversos inconvenientes, pues el proceso puede verse afectado en cuanto a rendimiento, dado que existirá una sola interfaz pero las tareas se realizarán por las aplicaciones encargadas de procesar la lógica de negocio (9)

La utilización de este modelo de integración no sería factible, dada que la principal característica que se describe o para la cual es viable su uso, en este caso la integración de sistemas mediante una sola interfaz, no es posible en el proceso de integración de datos en la AGR y las entidades externas entre las cuales se puede mencionar el MININT la cual tiene como objetivo primordial mantener la seguridad de sus datos, por lo que entre las partes involucradas en el proceso no existe la posibilidad de integración de interfaces dado que cada sistema es independiente y la AGR no tiene dominio sobre ellos.

**Modelo de integración de datos (MID):** Es un modelo basado en el acceso directo a los datos empelados por las aplicaciones. (10)

Ejemplo de aplicación común antes de aplicarle el MID:


**Figura4: Anterior a MID. (10)**

Como objetivo fundamental de este ejemplo es la necesidad de disponer de un único medio de comunicación entre usuario y sistema a la hora de la introducción de datos.

Ejemplo de aplicación después de aplicarle el MID:

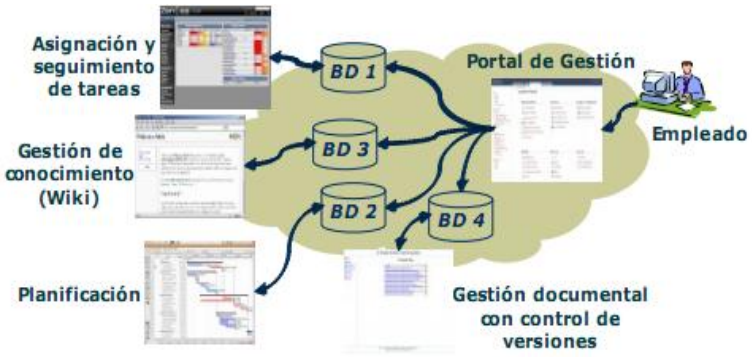



**Figura5: Posterior a MID (10)**

El uso del MID es recomendado en situaciones donde se desee combinar datos de varias fuentes para su posterior procesamiento, en sistemas que tengan acceso a diversas aplicaciones que necesiten hacer uso de la misma fuente de datos o la extracción de datos para la inserción posterior en otra fuente o transformación a un formato diferente. (10)

Este es un modelo que tiene como principal característica la necesidad de tener el control o poder determinar el grado de utilización de las fuentes de datos a integrar. La AGR para cumplimentar la tarea de recepción de documentos electrónicos no tiene acceso a las fuentes de datos desde donde se generan los documentos, por lo que la utilización de dicho modelo no sería adecuada para la solución.

**Modelo de integración funcional (MIF)**: Es un modelo donde la lógica de negocio es la parte fundamental de la aplicación, la característica fundamental consiste en la integración a nivel de negocio. (11)

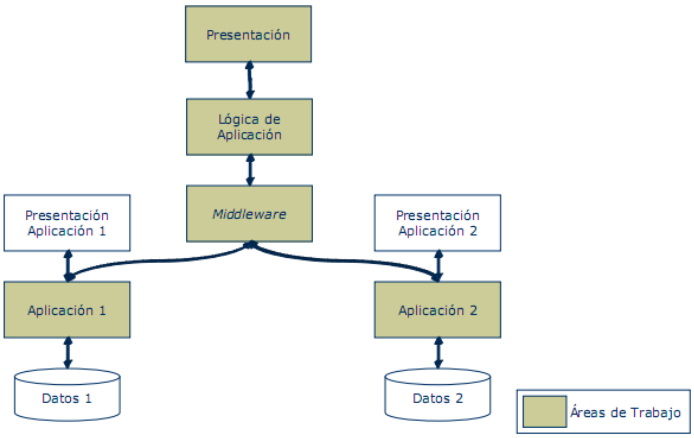



**Figura6: Esquema para MIF (11)**

**Middleware**: Se define como la capa de software que se encuentra entre el sistema operativo y las aplicaciones en cada sitio del sistema (12).

El middleware usado en esta representación está enfocado en el acceso a las aplicaciones.

El punto de integración está en el código de la aplicación original, se disponen de APIS de desarrollo, interfaces de comunicación ampliando el código anterior y servicios internos. Se utilizan distintas herramientas para realizar la integración funcional, tales como **RCP** para la construcción de procesos que puedan ser ejecutados a través de sockets o sistemas de objetos distribuidos para el acceso a objetos de forma remota, estos últimos categorizados como de tipo invocación de procesos de forma remota donde se agrupan los sistemas de mensajería como servicios web o middleware orientados a mensajes. (11)

El estudio de cómo realizar integración de datos y la aplicación de un modelo de integración funcional serán los conceptos rectores que permitirá definir una solución eficiente para el intercambio de información en la AGR.

## 1.4 Estándares

**Estándar:** (Del inglés **standard**).Que sirve como tipo, modelo, norma, patrón o referencia. (13)

Los estándares son un concepto necesario a nivel mundial en todas las ramas de la tecnología y la vida en general, dada la necesidad de comunicar diferentes medios se hace imprescindible el uso de una vía común de intercambio entendible por las partes involucradas. Los estándares son aplicables a cada

acción, aplicación o dispositivo desarrollado por el hombre. Definir un estándar brinda la posibilidad de comparación, llegar a consensos entre instituciones, medios tecnológicos y personas.

Diferentes organizaciones mundiales reconocidas hacen referencia a este concepto brindando definiciones aplicables, tal es el caso de la Organización Mundial de Normalización (ISO) la cual define estándar como:

"*Documento establecido por conceso y aprobado por estructuras reconocidas, para un uso común y repetido, son guías, líneas bases o características usadas en actividades o resultados, encaminado al resultado óptimo en un contexto dado*." (14)

El British Standard Institute (BSI) define por su parte como estándar lo referido a:

*"(…) una especificación publicada que establece un lenguaje común, y contiene una técnica u otro criterio, que está diseñado para ser usado constantemente, como una regla o una definición (…)"* (15)

La problemática presentada en la AGR sin la aplicación de un estándar para la información carecería de eficiencia, el núcleo del problema radica en diferentes tipos de estructuras, a los cuales se enfrentan los desarrolladores a la hora de integrar la información de un documento determinado.

## 1.5 La AGR y el proceso de envío, recepción e integración de datos.

Las aduanas mundiales son entidades que desarrollan múltiples y complejos procesos en su trabajo diario, entre los que se pueden mencionar la lucha contra las ilegalidades, monitorización de los puntos comerciales, la regulación del comercio exterior e interior así como proveer vías para el mejoramiento del mismo, la necesidad de contar con la información adelantada de pasajeros (API), que requiere de un gran intercambio de información tanto interna como con entidades externas. Las TIC han posibilitado la automatización de la información haciendo más fácil y rápido el procesamiento e intercambio de información a niveles mundiales para las aduanas.

La AGR como órgano de control del estado y reconocida ante la OMA, cumple y desarrolla muchos de los procesos referidos, por lo cual no está exenta de intercambiar información para la realización de los mismos, de la información generada o necesitada en el cumplimiento de las actividades aduanales. No toda es llevada al sistema informático y la que es procesada por el mismo presenta muchas irregularidades, lo que hace del intercambio de información es un proceso complejo y que requiere de grandes períodos de tiempo. Haciendo patente la inminente necesidad de crear mecanismos de integración de datos, recepción y envió que viabilicen esta comunicación de forma eficiente, acortando los tiempos de desarrollo e implantación de soluciones.

Dadas las definiciones anteriores y teniendo en cuenta la problemática referida por la AGR, la utilización del concepto de integración de datos definido por Mark R. Madsen (6), la aplicación de un modelo de integración funcional (**MFI**) y la definición de un estándar para los datos que se intercambian, serían los conceptos rectores a aplicar en la investigación para un posterior desarrollo de la solución.

## 1.6 Situación mundial de intercambio e integración de datos en sistemas aduanales.

El mundo está inmerso en un constante cambio de las tecnologías y la información, día a día surgen nuevos medios y vías para la comunicación y por ende el intercambiar información es una actividad cotidiana. El mundo para hacer frente a esta evolución ha desarrollado innumerables soluciones para desarrollar esta actividad de forma eficaz.

La búsqueda de las soluciones estará marcada por tres características fundamentales que deben poseer las mismas para su posterior estudio: la capacidad de realizar el proceso de envío y recepción de documentos digitales, deben ser soluciones libres y permitir la integración con el sistema GINA y los proceso aduanales cubanos.

Los principales sistemas que se encontraron que cumplían con al menos una de estas características se mencionan a continuación.

**SIDUNEA**

Es el sistema aduanero automatizado utilizado para controlar y administrar procesos aduanales desarrollados por la Conferencia de las Naciones Unidas sobre el Comercio y el Desarrollo (UNCTAD), en la actualidad más de 70 países lo usan de manera satisfactoria. Este sistema permite realizar un seguimiento automatizado de las operaciones aduaneras y controlar efectivamente la recaudación de los impuestos aduaneros, verifica automáticamente los registros, calcula los impuestos y contabiliza todo lo relativo a cada declaración con la mínima intervención del factor humano subjetivo. Al ser un sistema multidisciplinario, está especializado en cada área del trabajo aduanero para ser la herramienta de trabajo de todos los clientes de la aduana, sean usuarios internos o externos, privados o públicos. (16)

En la actualidad la UNCTAD ofrece tres versiones del sistema, las cuales son:

- SIDUNEA World

- SIDUNEA++

- SIDUNEA v2

Siendo SIDUNEA World el último producto lanzado a nivel mundial y basado en la web, con funcionalidades nuevas que permitirán actividades como transacciones - Declaraciones de la Aduana de Manifiestos de Carga y los documentos de tránsito a través de Internet. (16)

Las funcionalidades básicas del sistema están diseñadas para:

1. Facilitar y mejorar el cálculo, recaudación y contabilidad de los derechos de aduana y otros
2. Cargos relacionados con las operaciones aduaneras.
3. Acelerar el despacho de las mercancías y evitar el contrabando.
4. Proporcionar la gestión aduanera con información oportuna y precisa.
5. Permite el intercambio electrónico de datos entre comerciantes y aduana.
6. Directorios de mensajes XML, estándar que hace posible la cooperación internacional entre sistemas y la creación de la red Customs Global.

**VUCE- Ventanilla Única de Comercio Exterior (Perú)**

Es un sistema integrado que permite a las partes involucradas en el comercio exterior y transporte internacional gestionar a través de medios electrónicos y estándares por un único punto, los trámites requeridos por las entidades de control competentes para el tránsito, ingreso o salida del territorio nacional de mercancías. Es una plataforma desarrollada para cumplir con objetivos específicos que se desarrollan en Perú.

La VUCE, fue creada mediante Decreto Supremo Nº 165-2006-MEF y se establecieron los alcances a través del Decreto Legislativo Nº 1036 y su Reglamento contenido en el Decreto Supremo Nº 09-2008-MINCETUR. Le fue otorgado el rango de Ley a través de la Primera Disposición Complementaria del Decreto Legislativo Nº 1036. (17)

**Korea Custom Service (KCS)**

El Sistema de Servicios Aduanales de la República de Corea trabaja sobre la base de los sistemas EDI (Intercambio Electrónico de Datos), logrando la transferencia por medios electrónicos de información comercial o de negocios. Este sistema presenta un eficaz servicio de aduanas mediante el establecimiento de un sistema de liquidación preciso y rápido, mejora la competitividad nacional al reducir el tiempo y el costo requerido para el despacho de aduana, y la reducción de los costos logísticos, además perfecciona los servicios públicos, ofreciendo la información relativa a la Administración de Aduanas de manera precisa y rápida.

Esta solución presenta una serie de características que lo sitúan entre los primeros de su tipo en el mundo: desarrolla un sistema de información, investigación y vigilancia para el enfrentamiento a acciones delictivas que se cometan en las aduanas. Este sistema de información se compone de cinco subsistemas: Información General, Actividades de Investigación sobre Infracciones, Área de Investigación de la Empresa, Área de Vigilancia de Pasajeros, Vigilancia de los Buques y la Tripulación. Todas estas áreas en conjunto se encargan principalmente de la gestión de ex-convictos y sospechosos de violaciones en las aduanas, de la notificación a las autoridades en caso de violaciones, así como la gestión de la información sobre los pasajeros de alto riesgo y de viajeros frecuentes. (18)

## 1.7 ¿Por qué desarrollar un componente de intercambio de datos propio?

En la actualidad toda aplicación o software que se desarrolle, concibe entre sus funciones o procesos la necesidad de intercambiar información con otros agentes externos, por lo que existen una gran variedad de aplicaciones y arquitecturas que soportan e implementan estas funciones. El software aduanal conocido o el de más uso y popularidad es el SIDUNEA, el cual desarrolla el intercambio de información de documentos digitales como se señala en la funcionalidad 5 y 6 de la descripción anterior, este es un proceso interno que desarrolla el sistema, siendo imposible la separación del núcleo y la integración con el sistema GINA, esta solución es propietaria y de código cerrado incumpliendo con otra de las características que debe tener la solución buscada.

La VUCEP es una solución de integración de información e intercambio de documentos electrónicos, dicho proceso se encuentra formando parte de la solución, mezclado en los proceso que requieren de dicha actividad, por lo que no cumple con lo establecido por los estándares requeridos por la AGR, la integración con el sistema GINA se hace imposible siendo poco factible la utilización del mismo.

El sistema para el control aduanal de Corea, generan un gran tráfico de documentos y realizan complejas operaciones de procesamiento e intercambio con ellos, esta solución presenta patentes propietarias y no es posible la utilización de las funciones de intercambio de información separadas de la aplicación central. Ninguno de estos software o plataformas dan respuesta a las características necesarias para su utilización por parte de la AGR y la inclusión en el sistema GINA, por las cuales se decide crear un componente propio para el intercambio de información en la AGR. La necesidad de comprar alguno de estos software para su uso, es uno de los impedimentos principales a la hora de optar por ellos, dados los elevados costos, patentes y gastos que incluyen su adquisición, además de la complejidad para integrarlos con las especificaciones de la AGR, dado que la aduana presenta procesos diferentes a los usados mundialmente como consecuencias de las restricciones y condiciones de Cuba, así como la imposibilidad de acceso al código, por lo que el costo aumentaría en caso de necesitar modificaciones por parte de la AGR y se decidió incluir esta como una de las principales características a tener en cuenta a la hora de elegir una solución.

Existen plataformas para la integración de datos, que la solución pudiera usar pero esto implicaría la necesidad de lidiar con una interfaz[5] diferente al sistema de la AGR, lo cual pudiera conllevar a problemas de integración y estandarización de la información. El construir un sistema de intercambio nativo para el software que se desarrolla libraría a la AGR de todo costo, el mantenimiento sería por los especialistas, los estándares de documentación serían los deseados sin restricciones, la integración de datos entre el sistema y las fuentes externas no dependería de un sistema extra, todo se manejaría por la aplicación. A todo lo anteriormente expuesto, se suma la necesidad de convertirse en una nación independiente tecnológicamente, comprando software o integrando soluciones extranjeras no llegaría a tener éxito en el proceso de migración en que se encuentra envuelta

La AGR cuenta con un sistema informático llamado SUA, desarrollado por especialistas propios en el cual se identificaron los problemas referidos en el **capítulo 1**, en dicho sistema el proceso de recepción, envió e integración de datos es dependiente de cada aplicación o módulo encargado de procesar la información propia.

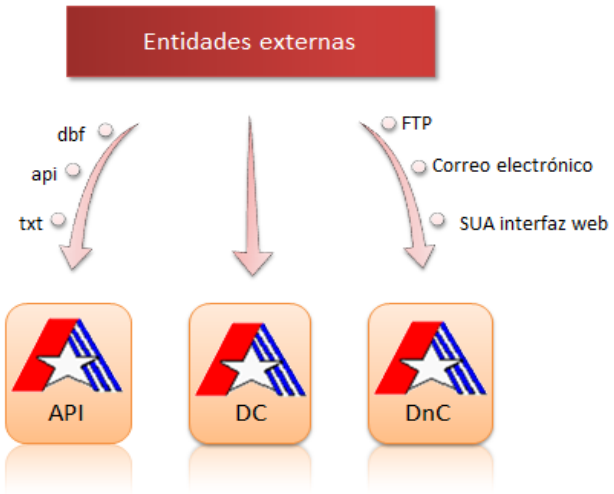



**Figura7: Funcionamiento del SUA.**

[5] **Interfaz**: Conexión que permite la comunicación entre dos o más dispositivos o sistemas al mismo nivel

En el sistema **GINA** que se desarrolla en la universidad, dada la poca experiencia al usar soluciones externas a la entidad y teniendo el objetivo de mejorar y agregar nuevas funcionalidades para hacer posible el proceso de recepción y envió de datos más fiable, fácil y eficaz se definió un estándar para los datos y el desarrollo de una solución, la cual se describe en el siguiente capítulo.

## 1.8 Conclusiones del capítulo.

Existen varios mecanismos a partir de los cuáles se pueden integrar datos entre varias aplicaciones, que pueden ser relevantes para la situación a la que se enfrenta la AGR. El análisis realizado sobre los diferentes esquemas permite descartar los modelos de interoperabilidad, como solución factible, debido a que en este se plantea como necesidad mezclar los datos de las aplicaciones y que estos funcionen de manera conjunta y una de las principales necesidades planteadas para este componente está asociada a la necesidad de mantener autónomos los datos de cada aplicación externa.

La integración de datos fue seleccionada como la propuesta teórica válida para la situación planteada y en particular el modelo de integración funcional ya que este permite mantener la autonomía de los datos de las aplicaciones al tiempo que los centraliza en la base de datos de la aduana garantizando un único punto de acceso a todos los datos. Con esto se garantizan los requisitos que fueron establecidos para el sistema.

# CAPÍTULO 2: DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA.

## 2.1 Introducción

En este capítulo se llevará a cabo la descripción de la solución propuesta abarcando las principales etapas para el desarrollo de la misma. En primer lugar se explicarán, de forma breve, las tecnologías empeladas para dar solución al problema planteado, que se corresponden con las referenciadas en el modelo de desarrollo definido para el Departamento de Soluciones para la Aduana (19). Se describen los procesos de negocio más significativos desde el punto de vista arquitectónico, ya que contienen las principales actividades que se van a automatizar, se definen los requisitos identificados correspondientes a los proceso descritos, se presentará el diagrama de clases, se realizará una breve descripción de la base de datos y sus principales entidades.

## 2.2 Propuesta de solución.

La solución que se propone es la creación de un componente encargado de manejar todo lo referente al proceso de envío y recepción de documentos digitales en las actividades aduanales automatizadas en el sistema GINA, sería una especie de pasarela o único punto de acceso a la hora de intercambiar documentos digitales con entidades externas.

El componente será el encargado de recibir los documentos emitidos por las entidades externas en formato XML, procesarlos sintácticamente a través de las plantillas en formato XSD (20), posteriormente se le envía al subsistema encargado de llevar a cabo las validaciones de negocio, el cual de encontrar errores los devolvería al componente, que sería el encargado de suminstrarle la información necesaria al usuario de acuerdo con la vía de procesamiento que haya comenzado el proceso.

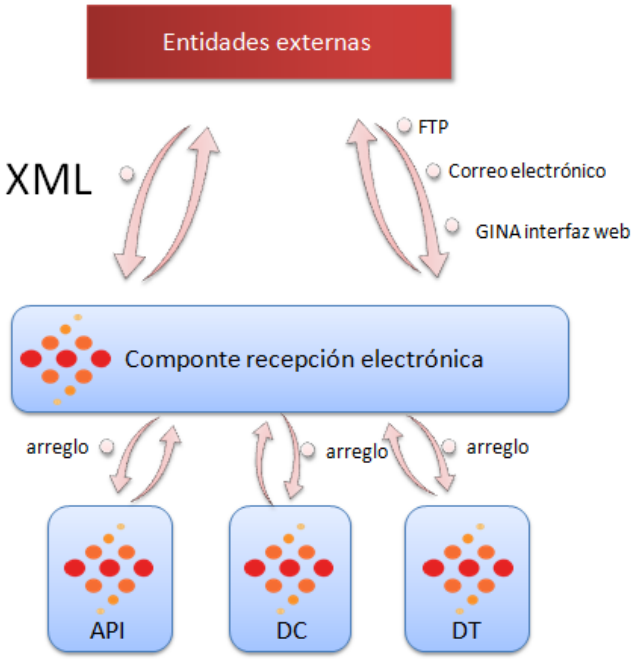



**Figura8: Propuesta de solución**.

## 2.3 Arquitectura del sistema.

La solución propuesta está definida arquitectónicamente por lo descrito en el documento Arquitectura de software correspondiente al Departamento de Soluciones para la Aduana, el cual fue desarrollado sobre las bases de la tesis de José Antonio Cobo (21), a partir de las consultas realizadas a los arquitectos y especialistas del departamento se determinó que bajo esta arquitectura se lograría un desarrollo completo y eficaz de la solución y su integración con el sistema GINA.

La arquitectura contempla entre sus definiciones la utilización del lenguaje PHP, haciendo uso del framework Symfony (22) para realizar la implementación del lado del servidor, el lenguaje JavaScript utilizando el framework ExtJS (23) para llevar a cabo la implementación de las interfaces de usuario, para las cuales fueron definidos los estándares en la tesis de grado de Mario Alberto Álvarez (24).

## 2.4 Metodología de desarrollo.

Como metodología de desarrollo que sustentará la solución es la definida por el Departamento de Soluciones para la Aduana, la cual quedó descrita en el documento **CEIGE-ADUANA-Modelo de Desarrollo de Software** (25)**.** Dicho procedimiento describe una serie de actividades realizadas por los especialistas encargados, se generan los artefactos por cada etapa del ciclo de desarrollo.

Se define para la creación de proyectos o sub-proyectos como ciclo de vida:

1. Estudio Preliminar
2. Modelación del Negocio
3. Requisitos
4. Análisis y Diseño
5. Pruebas
6. Despliegue

Las tablas que contienen la descripción de cada una de las etapas del ciclo de vida de un proyecto que se muestran en lo adelante son un extracto textual sin modificaciones del documento **Modelo de Desarrollo de Software** (25)**.**

Como principales artefactos y actividades generados se definen para cada una de las etapas:

**Etapa Estudio Preliminar:**

| **Actividades** | **Descripción** | **Artefactos** |
|---|---|---|
| Planear la Planeación | Definir el calendario para las principales actividades del proceso de planeación del proyecto. Debe incluirse: la planeación del levantamiento de requisitos, la definición del plan de proyecto, el establecimiento de compromisos. | Calendario (inicial, con las actividades de planeación) |

| | | |
|---|---|---|
| Realizar reunión de conciliación con proveedores | Se realiza la reunión de Conciliación, en la reunión se negocian los intereses de ambas partes, se llega a un acuerdo y se elabora el cronograma de trabajo. | Minuta de Reunión |
| Identificar los elementos de configuración y línea base | Identificar los elementos de configuración del proyecto basado en criterios predeterminados. Asignar identificador único. Especificar características de los elementos de configuración. | Elementos de Configuración del Proyecto |

**Etapa Modelado de Negocio**

| **Actividades** | **Descripción** | **Artefactos** |
|---|---|---|
| Re planificar en caso necesario | Se re planifica si es necesario | |
| Planear levantamiento de negocio y de requisito | Se actualiza el WBS y Calendario especificando las actividades relacionadas con el negocio y los requisitos | WBS y Calendario (actualizado con las actividades de negocio y requisitos ) |
| Modelar negocio | Recibir Documentación del negocio. Identificación de Procesos de Negocio Registrar los procesos identificados en el Mapa de procesos. Revisar los mapas de procesos identificados previamente e identificar subprocesos de los procesos. Realizar un estado del arte para ver lo que se ha hecho hasta el momento y lo que falta por hacer. Desarrollar talleres, encuentros y entrevistas a los proveedores de requisitos para obtener la información necesaria para completar la plantilla | - Descripción de requisitos del cliente<br><br>-Modelo Conceptual<br><br>-Modelo de Procesos de Negocio con BPM<br><br>-Descripción de procesos de negocio |

| | | |
|---|---|---|
| | de procesos de negocio y aclarar las dudas existentes. Completar la plantilla de procesos de negocio. Obtener Modelo de dominio si el proyecto lo requiere. Definir pautas de diseño gráfico y Arquitectura de Información del sistema si al proyecto le da valor agregado. | |

**Etapa Requisitos**

| Actividades | Descripción | Artefactos |
|---|---|---|
| Obtener y especificar requisitos | Describir los requisitos funcionales y no funcionales. Evaluar los requisitos según su complejidad Elaborar prototipo de interfaz de usuario. Definir arquitectura de información si al proyecto le da valor agregado. | - Especificación de requisitos de software<br>- Descripción de requisitos.<br>- Documento de salidas del sistema (actualizado) |

**Etapa Análisis y Diseño**

| Actividades | Descripción | Artefactos |
|---|---|---|
| Diseño de la base de datos | Se crea el Script de la BD a partir del modelo de datos y el diccionario de | -Script de BD<br>- Diseño de la BD |

| | | |
|---|---|---|
| | datos. Se diseña la BD. Se muestra a los funcionales el resultado y los mismos deciden si está correcto el modelo o si necesita algunos cambios. Se realiza un reporte en HTML de la base de datos. | -Reporte de HTML |
| Diseño de aplicación | Desde descripción de CUS, prototipo de IU, reporte HTML y descripción de la arquitectura se realiza el diagrama de clases y el modelado. Con la descripción de CUS, prototipo de IU, reporte HTML y descripción de la arquitectura se realiza el diagrama de flujo para cada uno de los métodos. | -Diagrama de clases<br>-Diagrama de flujo(por cada método) |

**Etapa Implementación**

| **Actividades** | **Descripción** | **Artefactos** |
|---|---|---|
| Definir estándar de codificación | Se definen los estándares de codificación con los cuales se implementará el sistema. | -Estándar de codificación |
| Implementación y Aplicación de pruebas unitarias | Se implementa toda la lógica de negocio diseñada. Se le aplican pruebas a la implementación para encontrar posibles errores en el código. | -Artefactos de implementación<br>-Resultados de las prueba unitarias. |

**Etapa Pruebas**

| Actividades | Descripción | Artefactos |
|---|---|---|
| Solucionar no conformidades | Se resuelven las no conformidades obtenidas | No conformidades(resueltas) |
| Solicitar servicios de Pruebas de liberación | Se solicita los servicios de pruebas internas al jefe de calidad del centro. | Jefe de proyecto |

## 2.5 Procesos de negocio de la solución.

Este epígrafe estará dedicado la definición y explicación de los artefactos generados en el proceso de descripción de negocio y el modelado de los procesos correspondientes para darle cumplimiento al mismo. Explicando únicamente en cada caso los artefactos más importantes para la solución. La documentación completa que complementa dicho proceso se encuentra en el repositorio oficial del Departamento de Soluciones para la Aduana (25).

Esta etapa de la solución está concebida dentro del flujo Requisitos definido en el modelo de desarrollo del departamento, el cual tiene como finalidad:

1. Capturar los requisitos y definir las prioridades.
2. Realizar la especificación de requisitos.
3. Desarrollar el modelo de análisis del sistema.

### 2.5.1 Descripción de los procesos de negocio.

El modelado de los proceso de negocio se hará utilizando la notación Business Process Modeling Notation según sus siglas en inglés (BPMN). La utilización de esta notación para la representación de los procesos identificados está definida por Jenni Manso Martínez en su tesis de grado. (26).

La solución cuenta con un total de 9 procesos modelados de los cuales se describen 4, por ser los más importantes en la solución .La descripción completa y el modelado se encuentran en el documento de Modelo de Procesos de Negocio con BPM (27) que se localiza en el repositorio del Proyecto de soluciones para la aduana.

### 2.5.1.1 Proceso: Intercambio electrónico de datos.

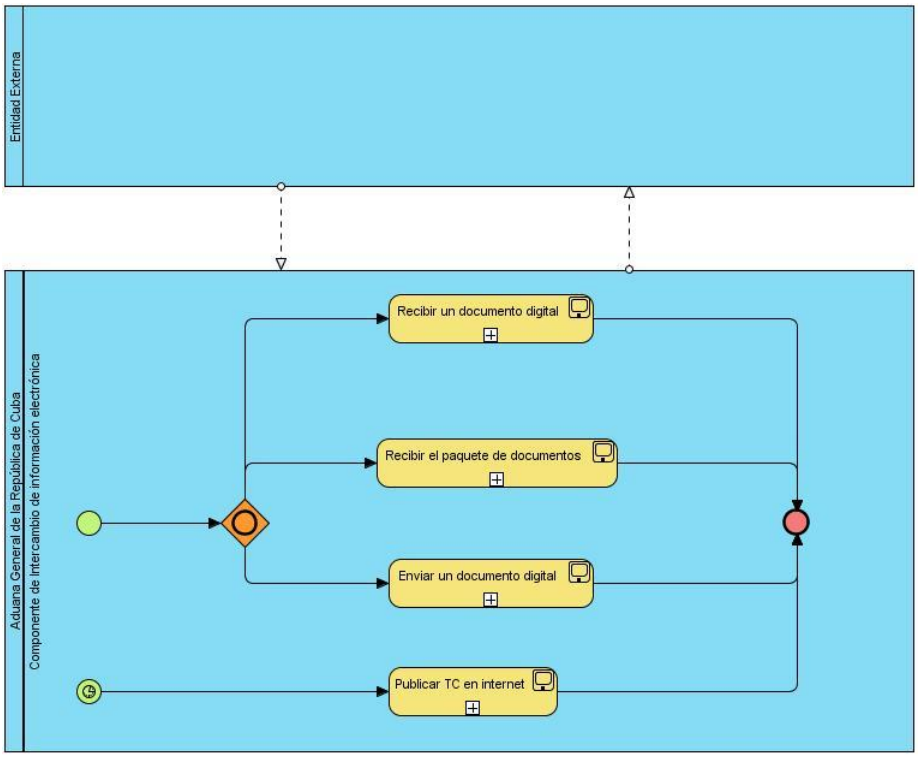



**Figura9: Proceso intercambio electrónico de datos**.

De forma general se muestra el modelado de los procesos que intervendrán en el intercambio de datos entre al AGR y las entidades externas. Este proceso permite controlar la entrada y la salida al sistema que se utiliza en la Aduana General de la República de Cuba de todos los documentos que se intercambian de manera electrónica.

### 2.5.1.2 Proceso: Recepción de documentos digitales.

Para la recepción y validación de los documentos emitidos por las entidades externas, previamente se debe confeccionar un documento de tipo XSD (20), mediante el cual se realizará la validación estructural del documento a procesar.

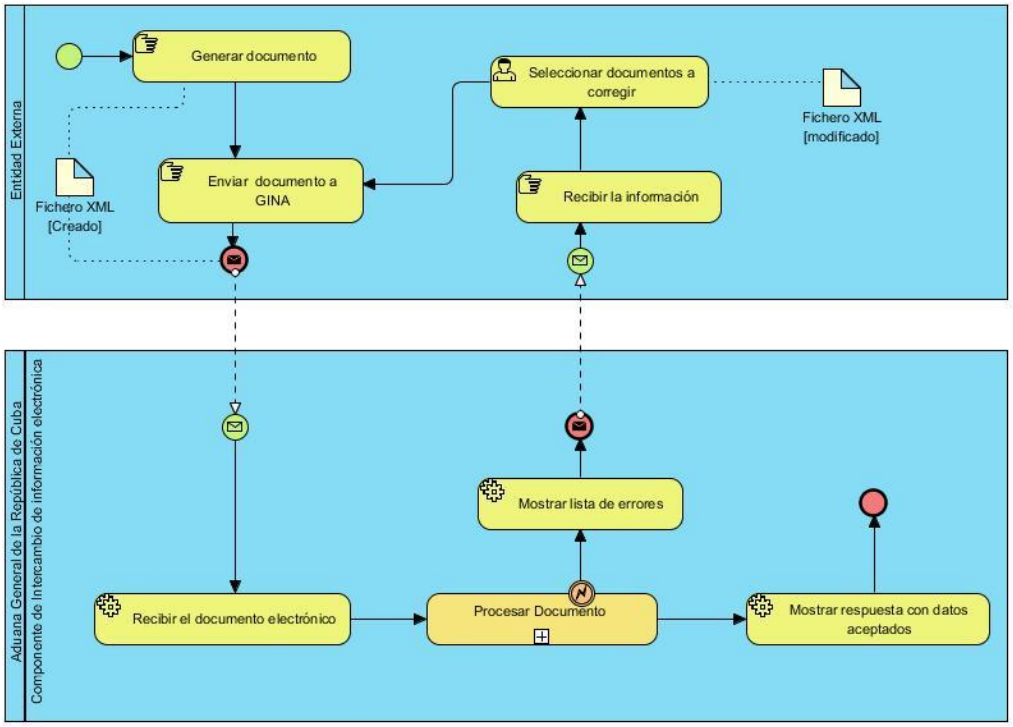



**Figura10: Proceso de recepción electrónica de documentos.**

Para la recepción de documentos se han definido 2 formas de intercambio de información con la AGR.

1. Mediante el módulo web incluido en el proyecto GINA.

    1.1. El componente comprueba con el XSD correspondiente al documento, en caso de existir problemas, le muestra los errores al usuario en forma de ventana emergente y registra en la auditoría relacionada con la recepción de documentos los errores encontrados y el estado en que quedó ese documento.

1.2. En caso de no encontrar errores con respecto al XSD, se registra la auditoría y se guarda físicamente el documento en un directorio específico en el servidor (en ambos lugares al nombre del fichero se le pone un consecutivo al final de tres dígitos).

1.3. Se revisa a qué subsistema le corresponde dicho documento, una vez conocido este, se invoca el método encargado de realizar las validaciones de negocio, pasándole como parámetro el arreglo que crea la clase que valida el XML, luego se guarda la información. El componente espera respuesta de dicho método, la cual puede ser de aceptación o una lista de errores de negocio los cuales serán mostrados al usuario.

1.4. Nota: el subsistema en caso de encontrar errores de negocio, no los envía hasta no terminar la ejecución del método, para que al final pueda enviar la lista de errores completos.

1.5. Todos los errores que presenten los documentos serán registrados en la auditoría de documentos.

2. .A través del sitio FTP de la aduana, donde escribe y lee el usuario externo.

2.1. El componente se conecta a la dirección del FTP cada cierto período de tiempo y descargarán los ficheros.

2.2. Se valida de forma similar, pero el usuario debe conectarse al FTP para comprobar si se aceptó o rechazó la información enviada, en caso de rechazarse ver los errores en el fichero de respuesta.

El proceso descrito anteriormente presenta un subproceso llamado **Procesar documento,** el cual será descrito a continuación.

### 2.5.1.3 Subproceso: Procesar documento.

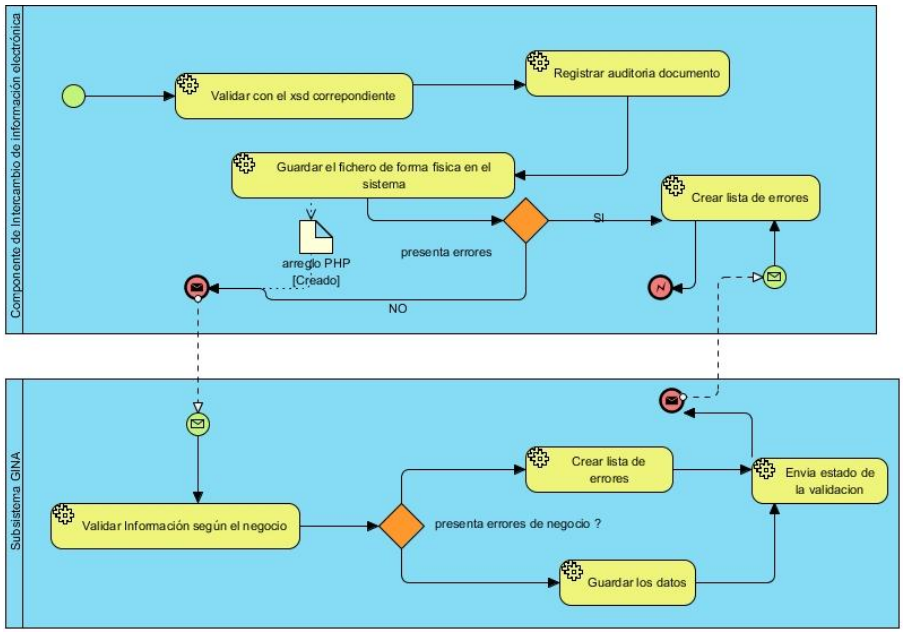



**Figura11: Subproceso: Procesar documento.**

El inicio del subproceso se realiza después que el componente cuenta con el documento XML por una de las vías determinadas, FTP o mediante la interfaz del sistema GINA.

1. Se valida el documento XML con la plantilla de estructura en formato XSD.

2. De no contener errores de estructura se realiza la auditoría para documentos sintácticamente correctos.

   2.1. Se guarda el fichero de forma física en el servidor.

   2.2. Se envía el documento en formato de arreglo al subsistema que corresponde el procesamiento de negocio.

   2.3. En caso que en la validación por parte del subsistema no se encuentren errores se guardan los datos.

2.4. Si el subsistema encuentra errores envía al componente de recepción electrónica la lista de errores encontrados.

3. De contener errores de estructura se realiza la auditoría para documentos con errores de sintaxis.

3.1. Se crea la lista de los errores encontrados en el procesamiento de sintaxis.

### 2.5.1.4 Proceso: Enviar un documento digital.

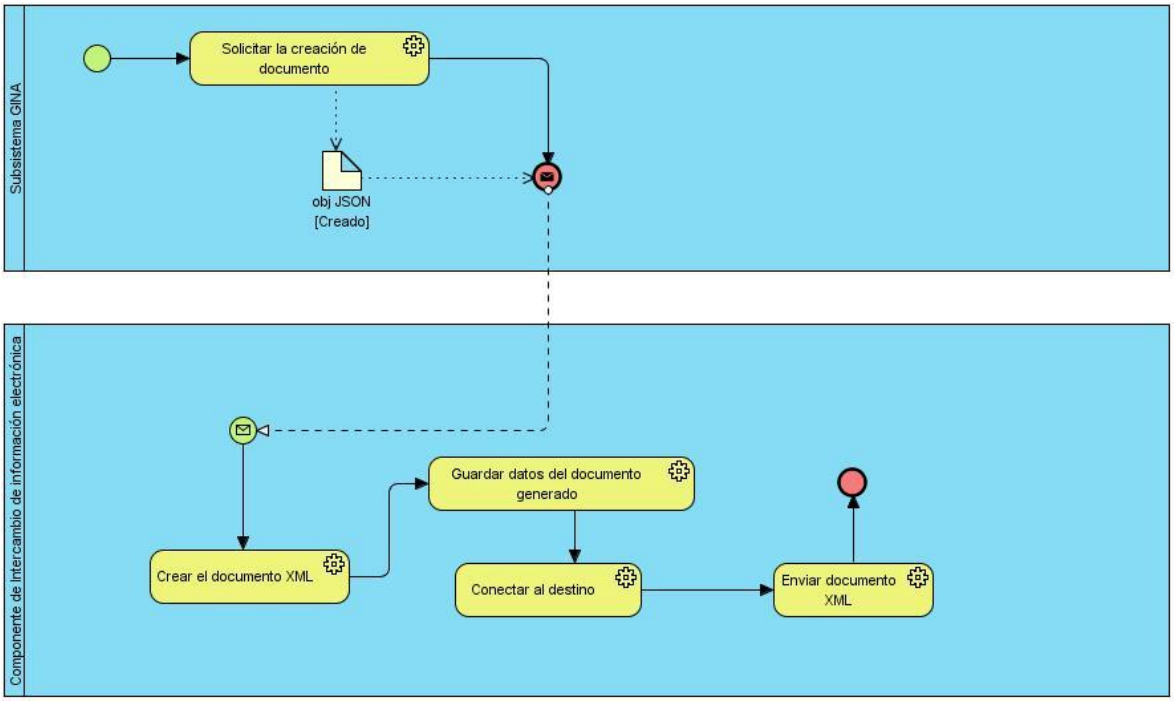



**Figura12: Proceso enviar un documento digital.**

Este proceso es creado dada la necesidad de las entidades externas de contar con información procesada por el sistema GINA, para brindar la información requerida se generan documentos o paquetes de documentos en formato XML y son enviados por la vía requerida por la entidad necesitada.

Para la iniciación de este proceso como precondición debe existir una solicitud de creación del documento y se debe enviar un documento XML para la finalización del mismo.

1. Realizar la solicitud de creación del documento por parte del subsistema, pasando la información que debe contener.
2. Confección del documento XML con los datos pasados al componente de intercambio por el subsistema que realiza la petición.
3. Guardar la información que se genera por el proceso y envío del documento XML.
4. Se conecta al servidor de destino para enviar el fichero generado o se copia en la dirección especificada por el subsistema.
5. Enviar el documento XML al destino el cual lo procesará para posteriormente utilizarlo.

## 2.6 Listado y descripción de requisitos funcionales.

La terminología definida por Ingeniería de Software de la IEEE define requerimiento o requisito como:

*"Una condición o necesidad de un usuario para resolver un problema o alcanzar un objetivo."*

*"Una condición o capacidad que debe estar presente en un sistema o componente de un sistema para satisfacer un contrato, estándar, especificación u otro documento formal."*

Los requisitos funcionales (RF) 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 requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer. (28)

### 2.6.1 Listado de requisitos.

En la solución propuesta, tras aplicada la captura de requisitos se identificaron un total de 7 requisitos funcionales de alto nivel, quedando definidos y descritos según las especificaciones del modelo de desarrollo en los artefactos necesarios para cumplimentar esta etapa. El principal artefacto que se necesita en esta etapa es el documento **Descripción de requisitos** (29)**,** quedando la descripción completa de todos los requisitos identificados según la plantilla definida por el departamento. A

continuación se mostrará una descripción exacta de los principales requisitos de la solución y se pueden consultar otros en los anexos.

**Principales requisitos funcionales:**

1. Recibir documento mediante internet.
2. Recibir documento mediante servidor FTP.
3. Procesar documento.
4. Generar documento.
5. Recibir paquete de documentos.
6. Procesar paquete de documentos.
7. Publicar paquete de documentos.

### 2.6.1.1 Especificación de requisito Recibir Fichero (sistema GINA).

Descripción textual del requisito en el documento **Especificación de requisitos funcionales 1.0**

| **Precondiciones** | El usuario se ha autenticado ante el sistema. | |
|---|---|---|
| **Flujo de eventos** | | |
| **Flujo básico Recibir fichero XML mediante internet** | | |
| **No** | **Actor** | **Sistema** |
| 1 | Selecciona la opción enviar fichero XML. | |
| 2 | | Muestra la pantalla para enviar el fichero al sistema de la Aduana. |
| 3 | Selecciona el fichero a enviar. Oprime el botón enviar. | |
| 4 | | Valida el fichero XML. *Ver Requisito Procesar fichero XML*. |

| | | |
|---|---|---|
| 5 | | Concluye el requisito. |
| **Pos-condiciones** | | |
| 1 | Se ha enviado el fichero correctamente.<br>Se reciben errores en el documento. | |
| **Validaciones** | | |
| 1 | Se validan los datos según lo establecido en el Modelo Conceptual. | |
| 2 | | |
| **Relaciones** | **Requisitos Incluidos** | Valida el fichero XML: Procesar fichero XML. |
| | **Extensiones** | No aplicable |
| **Requisitos especiales** | No aplicable | |
| **Asuntos pendientes** | No aplicable | |

### 2.6.1.2 Especificación de requisito Generar Fichero XML.

Descripción textual del requisito en el documento **Especificación de requisitos funcionales 1.0**
Este requisito es iniciado por un subsistema que ejecuta el servicio y envía los datos necesarios

| | | |
|---|---|---|
| **Precondiciones** | Un subsistema del GINA solicita la generación de un fichero a enviar con los datos deseados. | |
| **Flujo de eventos** | | |
| **Flujo básico Generar Fichero XML** | | |
| **No** | **Actor** | **Sistema** |
| | Ejecuta el método para la generación del fichero XML, pasándole los datos | |

| | | |
|---|---|---|
| | en el formato establecido para la generación del fichero. | |
| | | Genera el fichero solicitado con los datos que envía el subsistema que hace la solicitud.<br><br>En caso de no poder generar el fichero ver *Flujo Alterno 2.a "no se pudo generar el fichero".* |
| | | Envía el fichero a su respectivo destino por el modo de comunicación definido para dicho fichero.<br><br>En caso de error en el envío ver *Flujo Alterno 3.a "Error al enviar el fichero".* |
| | | Concluye el requisito. |
| Pos-condiciones | | |
| | Que el fichero sea generado correctamente. | |
| | Que el fichero sea enviado correctamente a su destino. | |
| **Flujos alternativos** | | |
| **Flujo alternativo 2.a No se pudo generar el fichero** | | |
| **No** | **Actor** | **Sistema** |
| | | Se le comunica al subsistema que no se pudo generar el fichero por algún error en alguno de los datos. |
| | | Concluye el requisito. |
| **Pos-condiciones** | | |
| | Informar que no se pudo generar el fichero XML. | |
| **Flujo alternativo 3.a Error al enviar el fichero** | | |

| No | Actor | Sistema |
|---|---|---|
| | | Se le comunica al subsistema de que no se pudo enviar el fichero por algún error en la conexión. |
| | | Concluye el requisito. |
| **Pos-condiciones** | | |
| | Informar al subsistema que no se pudo enviar el fichero XML. | |
| **Validaciones** | | |
| | Se validan los datos según lo establecido en el Modelo Conceptual [2]. | |
| **Relaciones** | Requisitos Incluidos | No aplicable |
| | Extensiones | No aplicable |
| **Requisitos especiales** | No aplicable | |

## 2.7 Modelo de diseño.

El modelo de diseño es un artefacto 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 utilizado como una entrada fundamental de las actividades de implementación. (30)

El modelo de diseño de la presente investigación está compuesto por los diagramas de clases de negocio, el diagrama de paquetes y el diagrama de persistencia de datos o diagrama de base de datos.

## 2.7.1 Diagrama de clases del negocio.

Los diagramas de clases muestran la estructura estática del modelo en particular, lo que se define como clases y tipos, su estructura interna y sus relaciones con otras clases. Los diagramas de clases no muestran información temporal, aunque puede contener ocurrencias de los eventos temporales.

En la **figura 5** se define el diagrama de clases del negocio del componente de recepción electrónica para el sistema GINA. En este esquema se clasifican las clases representadas en **nomencladores pertenecientes al esquema**, **nomencladores fuera del esquema**, **clases de relación y clases significativas de negocio.**

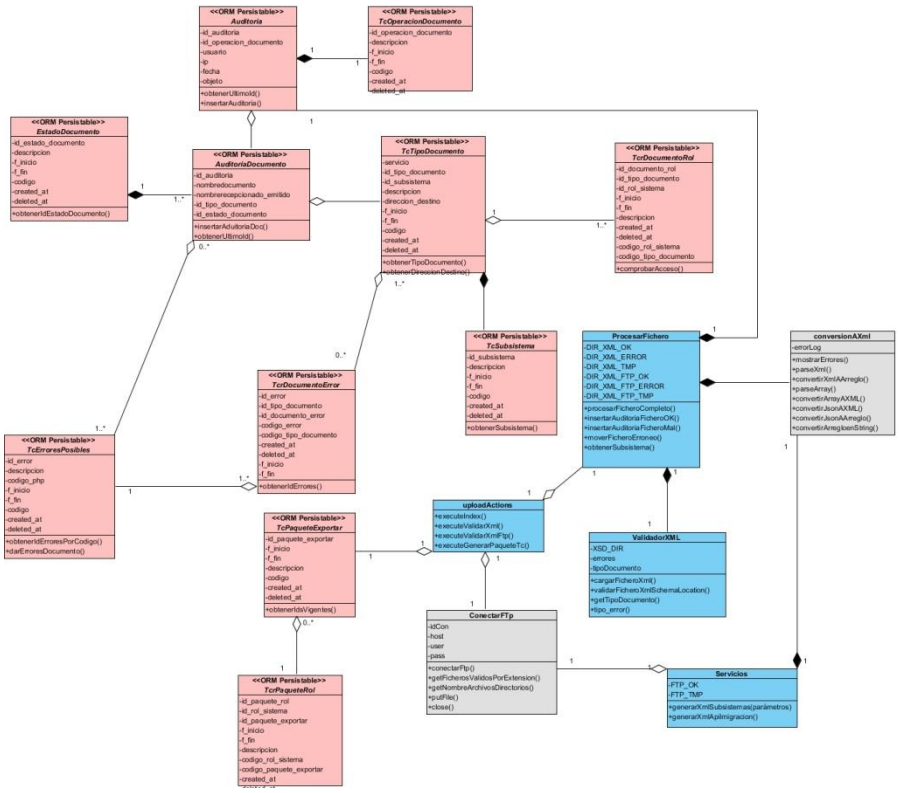



**Figura13: Diagrama clases.**

## 2.7.2 Diagrama físico de la base de datos.

Los diagrama entidad relación (DER) son una potente herramienta para la confección y entendimiento de la persistencia de datos en los sistemas gestores de base de datos teniéndose como definición:

Datos organizados en conjuntos interrelacionados de objetos (entidades) con atributos asociados. (31)

El diagrama correspondiente puede ser consultado en el anexo1.

**Tablas fundamentales para el negocio:**

1. **Auditoria**: Mediante esta tabla se podrán gestionar los datos auditables en las operaciones con los documentos.
2. **AuditoriaDocumento**: Mediante esta tabla se podrá gestionar la relación de auditoría con el documento específico.
3. **Servicios**: Mediante esta tabla se podrá gestionar los servicios para que los subsistemas puedan generar los documentos XML.

**Metadatos fundamentales pertenecientes al esquema:**

1. **TcTipoDocumento**: Esta tabla tiene como objetivo almacenar los tipos de documentos de cada uno de los subsistemas.
2. **TcErroresPosibles**: Esta tabla tiene como objetivo almacenar los errores posibles de un documento.
3. **TcEstadoDocumento**: Esta tabla tiene como objetivo almacenar los estados de los documentos. (Aceptado, Rechazado).
4. **TcOperacionDocumento:** Esta tabla tiene como objetivo almacenar las operaciones de los documentos.

**Metadatos fundamentales fuera del esquema (color Azul):**

1. **TcRolSubsistema**: Esta tabla tiene como objetivo almacenar el acceso de un subsistema de acuerdo al rol que tenga asignado.
2. **TcSubsistemas:** Esta tabla tiene como objetivo almacenar los subsistemas que procesarán los documentos y su correspondencia.

3. **SisTcTabla:** En esta tabla se encuentra la información referente a todas las tablas que componen el sistema.

**Clases de relación:**

1. **TcrDocumentoRol:** Esta tabla tiene como objetivo establecer la relación de los tipos de documentos para cada uno de los roles**.**
2. **RAuditoriaDocErrores:** Esta tabla tiene como objetivo mantener la auditoria de los errores de los documentos procesados.

## 2.7.3 Diagrama de paquetes.

Un diagrama de paquetes es una agrupación de elementos del modelo. Los cuales se pueden anidar unos dentro de otros. 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. (32).

En el anexo2 se representa el diagrama de paquetes correspondiente a la solución desarrollada, teniendo como bases los paquetes que deben formar la estructura de un componente en Symfony. Este diagrama está compuesto por 7 paquetes los cuales se describen a continuación.

El diagrama correspondiente puede ser consultado en el anexo 2.

**Paquete sfEaduana**: Este paquete engloba las funcionalidades del componente y los paquetes necesarios para el funcionamiento esencial del mismo, siendo el encargado de gestionar todo lo referente al intercambio de información.

**Paquete config:** Contiene los ficheros de configuración entre los que se pude mencionar el fichero para la definición de los servicios web[6] (*services.yml*).

**Paquete lib:** Contiene 2 paquetes **MODEL** y **ÚTILES,** el primero contiene las clases del modelo de datos y el segundo las clases donde se implementan los servicios y otras funcionalidades necesarias para el funcionamiento del mismo.

[6]**Servicio Web:** es una pieza de software que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

**Paquete web:** Encargado de agrupar los ficheros (imágenes, clases CSS y clases JavaScript) que decoran la vista de la aplicación.

**Paquete modules:** Contiene las principales funciones de acceso de componente agrupadas por módulos.

De esta forma queda estructurada la solución a desarrollar.

## 2.8 Conclusiones del capítulo.

A partir de la aplicación del modelo de desarrollo establecido para el departamento Soluciones para la Aduana se establecieron 7 requisitos funcionales de alto nivel que recogen las funcionalidades necesarias para garantizar que el componente satisfaga las demandas del sistema. La transformación de estos requisitos en clases de diseño y en modelos de datos ajustados al marco de trabajo **Symfony** haciendo uso de la arquitectura definida, se garantiza la entrada necesaria para la implementación del componente.

# CAPÍTULO 3: IMPLEMENTACIÓN Y VALIDACIÓN DE LA SOLUCIÓN.

## 3.1 Implementación de la solución.

Culminada la fase de diseño de la solución corresponde realizar la fase de implementación de la solución, la cual tiene como principal objetivo generar el código ejecutable y funcional, basado en el diseño correspondiente. Dicha fase tiene como principales actividades según el modelo de desarrollo definido:

1. Se re-planifica la fase en caso que sea necesario.
2. Se definen los estándares de codificación con los cuales se implementará el sistema.
3. Se implementa toda la lógica de negocio diseñada.

### 3.1.1 Estilos de codificación.

Los estilos de codificación son una guía que define la estructura sobre la cual se debe regir el código de la implementación del sistema. El crear un estándar de codificación permite que el código fuente esté estructurado de igual forma en toda la solución. El estilo debe ayudar al personal del proyecto para identificar de forma sencilla cuál es el objetivo y las funcionalidades que brinda cada una de las clases, funciones y demás componentes de software dada su nomenclatura, es necesario que esto se pueda identificar a simple vista. Además debe servir de guía para posteriores implementaciones o modificaciones del sistema. (25)

En el Departamento de Soluciones para la Aduana del centro CEIGE se definió un estándar para la codificación.

### 3.1.2 Reglas de codificación.

Las principales reglas a seguir utilizando el citado estándar se mencionan a continuación.

1. Los encabezados en las funciones y acciones deberán contener el requisito a que da solución, los parámetros que necesita, así como la respuesta esperada:

```
/**
* RF: Insertar Auditoria
* @return: id
**/
```

2. Utilizando la notación de Symfony para la creación de los plugins, la cual define que el nombre de los mismos siempre debe estar atados al sufijo Plugin. El nombre de un plugin debe indicarse con la menor cantidad de palabras, especificar cuál es el objetivo del plugin e iniciarse con el prefijo **sf**.

```
sfEaduanaPlugin /* Correcto


eAduanaPlugin /* Incorrecto
```

3. Los nombres de las clases deben estar expresados en notación **UpperCamelCase**, no se deben utilizar guiones bajos en su nombre "_", deben expresar con claridad cuál es el alcance y la responsabilidad de la clase, no deben estar atados a las clases de las que se derivan.

### 3.1.3 Tratamiento de errores.

La detección y control de errores en un sistema es una de las tareas fundamentales que lleva una implementación, se debe tener sumo cuidado a la hora de realizar un tratamiento de forma correcta, pues una mala implementación puede traer consigo un mal funcionamiento del sistema más adelante y malas experiencias a los usuarios. Existen diversos mecanismos de validación de errores que nos evitarían estos futuros problemas.

Del lado del cliente se valida la información que es suministrada por el mismo en busca de incoherencias o posibles errores tales como el tipo de fichero que se sube al sistema, el tamaño del fichero entre otras.

Por parte del servidor para la detección de errores se realizó esta tarea mediante el uso del sub-framework para formularios que contiene el marco de trabajo **Symfony** haciendo uso del lanzamiento de excepciones, además del tratamiento de los posibles errores en los procesos a realizar por el componente en las diferentes funcionalidades.

### 3.1.4 Implementación basada en diagramas de interacción.

En la fase de diseño entre los artefactos generados se encuentran los diagramas de interacción como clasificación superior, específicamente desarrollando los diagramas de secuencia orientados a actividades, este es el principal tipo de diagrama que se definió para ser generados en el Departamento de Soluciones para la Aduana referido en el Modelo de Desarrollo confeccionado en el mismo. Este diagrama tiene como principal objetivo realizar un diseño más orientado al desarrollo, de forma tal que la fase de implementación sea más viable, dotando a los programadores de mayor claridad en la implementación a desarrollar. El diagrama mencionado es una combinación de los diagramas de actividades y colaboración propuestos por RUP, permitiendo agregar comentarios para facilitar su entendimiento. Se mostrará un fragmento de código correspondiente a uno de los diagramas generados en la solución.

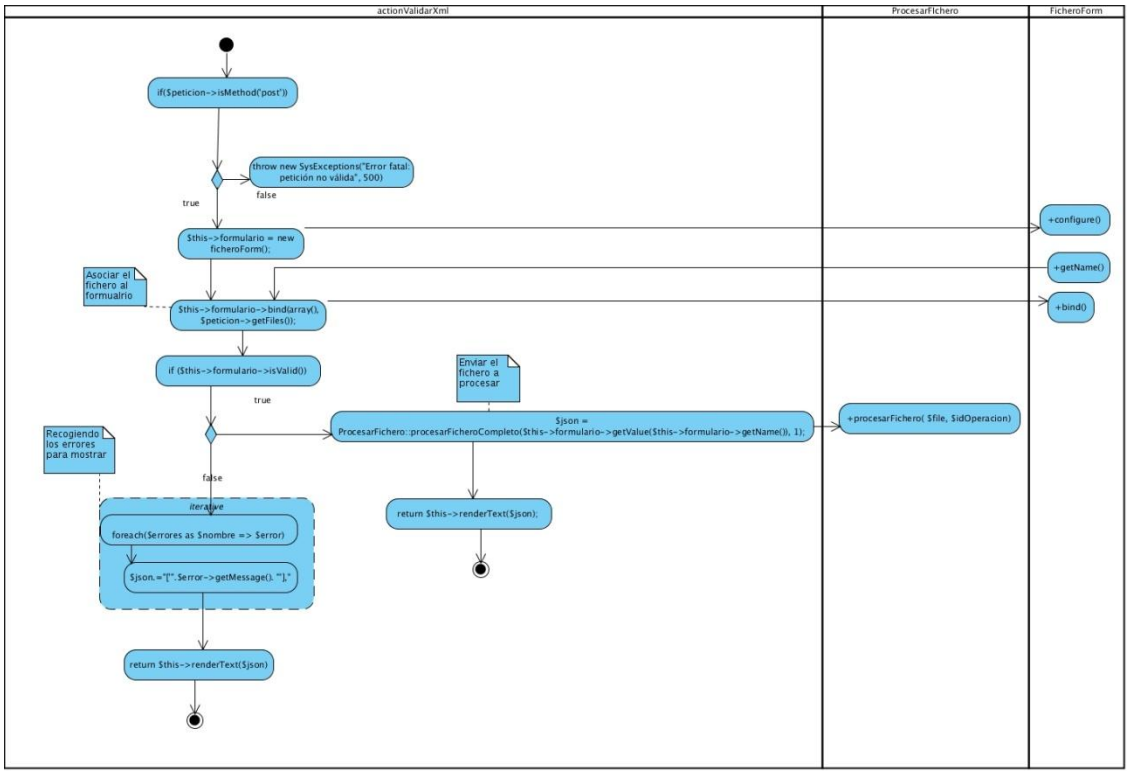



**Figura14: Diagrama de Secuencia orientado a las actividades.**

Código correspondiente al diagrama presentado anteriormente**:**

```
// GINA/plugins/sfeAduanaPlugin/modules/upload/actions/action.class.php

public function executeValidarXml($peticion) {
  if ($peticion->isMethod('post')) {
   $this->formulario = new ficheroForm();
   $this->form->bind(array(), $peticion->getFiles());

 if ($this->form->isValid()) {
try {

$json = ProcesarFichero::procesarFicheroCompleto($this->form->getValue($this->form->getName()),
1);
     return $this->renderText($json);
    } catch (Exception $exc) {
return $this->renderText("{'success': false,errores:[['" . str_replace("'", "\'",
str_replace("\"", "\\\" ", str_replace("\n", "<br>", strtoupper($exc->getMessage())))) .
"']]}");
    }
   } else {
    $json = "{'success': false,errores:[";
foreach ($this->form->getErrorSchema() as $nombre => $error)
{
     $json.="['" . $nombre . ": " . $error->getMessage() .'],";
    }
    $json.=']}';
    return $this->renderText($json);
   }
  } else {
   throw new SysExceptions("ERROR FATAL: PETICIÓN NO VÁLIDA", 500);
}
 }
```

Como se puede observar existe una gran semejanza entre el diagrama generado por los diseñadores y la implementación, ya que son muy cercanos a la tarea realizada por los desarrolladores, haciendo mucho más fácil el entendimiento de los mismo y eliminando en gran parte los errores de implementación.

En esta etapa también se realizarán las pruebas necesarias para validar la solución desarrollada, correspondiente a la **fase de pruebas** según el modelo de desarrollo del Departamento de Soluciones para la Aduana (25). Entre las principales actividades a realizar en esta fase se describen:

1. Pruebas internas del sistema antes de incorporarlo al equipo central de calidad para que realice la pruebas de liberación, tratando de garantizar las detecciones de la menor cantidad de no conformidades.
2. Se resuelven las no conformidades obtenidas.
3. Se re-planifica la fase en caso que sea necesario. (19)

La automatización de pruebas es uno de los mayores avances en la programación desde la invención de la orientación a objetos. Concretamente en el desarrollo de las aplicaciones web, las pruebas aseguran la calidad de la aplicación incluso cuando el desarrollo de nuevas versiones es muy activo. (22)

## 3.2 Técnicas de validación del sistema.

Una solución se determina como factible debido al grado de coincidencia entre lo esperado y los resultados reales, por lo cual se hace necesario el determinar la validez del cumplimiento de los requisitos, se pueden utilizar dos técnicas de pruebas de software las cuales se describen a continuación:

**Técnicas de Evaluación Estáticas:** buscan faltas sobre el sistema en reposo. Estudian los distintos modelos que componen el sistema de software buscando posibles faltas en los mismos. Así, estas técnicas se pueden aplicar, tanto a requisitos como a modelos de análisis, diseño y código. Teniendo como nombre genérico de **Revisiones**.

**Técnicas de Evaluación Dinámicas:** generan entradas al sistema con el objetivo de detectar fallos cuando el sistema ejecuta dichas entradas. Los fallos se observan cuando se detectan incongruencias entre la salida esperada y la salida real. La aplicación de técnicas dinámicas es también conocida como pruebas de software o *testing* y se aplican generalmente sobre código. (33)

La siguiente figura muestra la aplicación de dichas técnicas en el proceso de desarrollo del software.

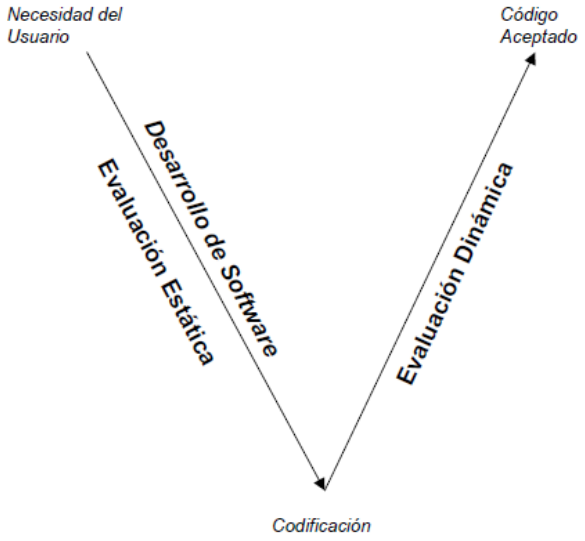



**Figura15 Técnicas de evolución. (33)**

## 3.1.1 Técnicas de evaluación estáticas.

La evaluación de un software debe ser un proceso que se va realizando en la misma medida que se construye el mismo, de esta forma la detección de errores de forma temprana afecta de manera positiva varios aspectos del ciclo de desarrollo, se mencionan entre estos aspectos la variable tiempo de entrega, el esfuerzo que se debe realizar en la fase de desarrollo. Se puede decir que las actividades de revisión acompañan las actividades del modelo de desarrollo de software que guía el proyecto. Las revisiones pretenden detectar manualmente defectos en cualquier producto del desarrollo como objetivo o idea principal.

Para lograr esta estrategia que se propone, las técnicas de evaluación estática brindan una serie de pasos a seguir preferentemente en las etapas de requisitos y análisis y diseño.

La experiencia demuestra que entre el 30% y el 70% de los defectos de diseño y código son detectados por las técnicas estáticas. Esto supone un gran ahorro, pues la corrección es más fácil y menos costosa durante la evaluación estática que durante la dinámica. (33)

La evaluación estática para su cumplimiento de manera exitosa está guiada por una serie de objetivos o defectos a encontrar, que evaluarán los artefactos generados durante el proceso de desarrollo.

Los defectos que se buscan al evaluar estáticamente los productos software son:

**Para los requisitos**:

1. Corrección. Los requisitos especifican correctamente lo que el sistema debe hacer. Es decir, un requisito incorrecto es un requisito que no cumple bien su función. Puesto que la función de un requisito es indicar qué debe hacer el sistema, un requisito incorrecto será aquel que indica incorrectamente lo que debe hacer el sistema. En otras palabras, un requisito incorrecto no se corresponde con lo acordado o adecuado; contiene un error.
   **Resultado:** Entre las necesidades identificadas por el cliente en este caso la AGR, se tiene que el sistema debe recibir documentos desde entidades externas, para lo cual se describe el **RF Recibir Fichero XML**, requisito que claramente define una actividad requerida por la solución, quedando así aplicada de forma correcta esta etapa.

2. Compleción. Especificación completa del problema. Está especificado todo lo que tiene que hacer el sistema y no incluye nada que el sistema no deba hacer. En dos palabras: no falta nada; no sobra nada.
   **Resultado:** Como resultado de la aplicación de esta etapa se determinó que en los 7 requisitos funcionales descritos se encontraban cubiertas todas las necesidades del cliente, las cuales a grandes rasgos se determinan como la necesidad de generar documentos y paquetes de documentos electrónicos, procesar dichos documentos y paquetes, para esto se deben recepcionar por diferentes vías los mismos, quedando descritos los requisitos mencionados en el epígrafe 2.6.1.

3. Consistencia. No hay requisitos contradictorios.
   **RF Generar documento:** En este requisito se describen las tareas a realizar para cumplimentar la generación de documentos electrónicos.
   **RF Procesar documento:** Requisito encargado de procesar los documentos.
   **Resultado:** Cada requisito especifica solo las tareas que debe realizar y cada uno realiza tareas diferentes.

4. Ambigüedad. Los requisitos no pueden estar sujetos a interpretación. Si fuese así, un mismo requisito puede ser interpretado de modo distinto por dos personas diferentes y por tanto, crear dos sistemas distintos. Si esto es así, los requisitos pierden su valor pues dejan de cumplir su función (indicar qué debe hacer el sistema). Las ambigüedades provocan interpretación por parte de la persona que use o lea los requisitos. Por tanto, una especificación debe carecer de ambigüedades.

   **Resultado:** Los requisitos identificados y descritos son claramente entendibles, pues se nombran con la actividad central que describen, por lo que la ambigüedad se elimina.

   **Recibir documento XML:** Describe claramente la actividad para la que fue descrito en este caso, muestra las actividades a realizar para recibir los documentos a procesar.

5. Claridad. Se entiende claramente lo que está especificado.

   **Resultados:** Las consultas realizadas al documento que contiene la descripción de los requisitos, se realizaron sin ningún problema de entendimiento por lo que contaba de buena calidad y claridad.

Control de cambios del documento de Requisitos funcionales:

| Versión | Lugar* | Tipo** | Fecha | Autor | Descripción |
|---|---|---|---|---|---|
| 1.0 | Todo el Documento | A | 15/05/2011 | Est. Roberto Nuñez Corrales | Creación del documento. |
| 1.0 | Todo el documento | M | 17/05/2011 | Est. Roberto Nuñez Corrales | Corrección de datos en las diferentes secciones |
| 1.0 | Requerimientos | M | 15/05/2011 | Est. Roberto Nuñez Corrales | Adicción de un nuevo requerimiento de la publicación de TC en internet. |
| 1.0 | Todo el documento | M | 06/06/2011 | Est. Roberto Nuñez Corrales | Corrección de datos |
| 1.0 | Todo el documento | M | 12/06/2011 | Est. Roberto Nuñez Corrales | Revisión de ambigüedad |
| 1.0 | Todo el documento | M | 15/06/2011 | Est. Roberto Nuñez Corrales | Modificación de la plantilla |

**Para el diseño:**

1. Corrección. El diseño no debe contener errores. Los errores de corrección se pueden referir a dos aspectos. Defectos de “escritura”, es decir, defectos en el uso de la notación de diseño empleada (el diseño contiene detalles prohibidos por la notación). Defectos con respecto a los requisitos: el diseño no realiza lo que el requisito establece. Hablando apropiadamente, los primeros son los puros defectos de corrección, mientras que los segundos son defectos de validez.

   **Resultados:** Los resultados arrojados a esta etapa se pueden revisar en el diagrama de clases de negocio, el uso de los estilos de codificación se realizó correctamente. Teniendo como ejemplo las clases TcTipoDocumento, Auditoria, TcSubsistemas.

2. Compleción. El diseño debe estar completo. Ya sea que diseña todo el sistema marcado por los requisitos; ya sea no diseñando ninguna parte no indicada en los requisitos. De nuevo, nada falta, nada sobra.

   **Resultados:** La implementación realizada cumplió con todos los requisitos descritos y por tanto como la misma está basada en el diseño realizado, se concluye que el diseño presenta una compleción correcta.
3. Consistencia. Al igual que en los requisitos, el diseño debe ser consistente entre todas sus partes. No puede indicarse algo en una parte del diseño, y lo contrario en otra.
4. Factibilidad. El diseño debe ser realizable. Debe poderse implementar.

   **Resultado:** Este diseño se implementó obteniendo resultados satisfactorios, pues todo lo especificado y diseñado se pudo implementar y desplegar con resultados satisfactorios.
5. Trazabilidad. Se debe poder navegar desde un requisito hasta el fragmento de diseño donde éste se encuentra representado.

   **Resultado:** Este aspecto puede ser comprobado en varios casos, citando el ejemplo del requisito funcional descrito para el procesamiento de documentos, para visualizar el segmento de código correspondiente basta con remitirse a la clase **ProcesarFichero** a la funcionalidad específica **procesarFicheroCompleto**.

Control de cambios del documento de Modelo de diseño:

| Versión | Lugar* | Tipo** | Fecha | Autor | Descripción |
|---|---|---|---|---|---|
| 1.0 | Todo el Documento | A | 24/04/2012 | Est. Roberto Nuñez Corrales | Creación del documento. |
| 1.0 | Todo el documento | M | 27/04/2011 | Est. Roberto Nuñez Corrales | Corrección de diagramas. |
| 1.0 | Todo el documento | M | 06/052012 | Est. Roberto Nuñez Corrales | Corrección de datos. |
| 1.0 | Todo el documento | M | 15/05/2012 | Est. Roberto Nuñez Corrales | Modificación de la plantilla. |

**Código Fuente:**

1. Corrección. El código no debe contener errores. Los errores de corrección se pueden referir a dos aspectos. Defectos de “escritura”, es decir, lo que habitualmente se conoce por “programa que no funciona”. Por ejemplo, bucles infinitos, variable definida de un tipo pero utilizada de otro, contador que se sale de las dimensiones de un array, etc. Defectos con respecto al diseño: la implementación no realiza lo que el diseño establece. (33)
   **Resultado:** La implementación está libre de los errores que se definen en esta etapa, no existen ciclos infinitos o desbordamiento de memoria, dado que a cada una de las funcionalidades se le realizaron pruebas básicas de funcionamiento.

Después de aplicadas las mencionadas fases, para realizar las pruebas siguiendo la estrategia estática, la solución desarrollada tiene un nivel de calidad óptima, pues se corrigieron todas las fallas o errores descritos para cada una en la etapa de desarrollo, dejando lista la solución para la aplicación de las técnicas dinámicas, en este caso pruebas unitarias y funcionales.

## 3.3 Pruebas unitarias.

Realizar las pruebas unitarias es la primera fase de las pruebas dinámicas y se realizan sobre cada módulo del software de manera independiente. El objetivo es comprobar que el módulo, entendido como una unidad funcional de un programa independiente, está correctamente codificado. En estas pruebas cada módulo será probado por separado y lo hará, generalmente, la persona que lo creó (33)

La principal técnica de validación que se utilizó para validar la solución fueron las pruebas unitarias, desarrollando las mismas haciendo uso del framework Symfony, utilizado en la implementación de la solución. Este tipo de pruebas son altamente recomendadas llevarlas a cabo antes de realizar las pruebas funcionales, debido a que las pruebas unitarias permiten encontrar los errores más evidentes y fáciles de corregir, en la etapa de pruebas funcionales el sistema debería estar bastante estable y con muy pocos errores críticos. (34)

Las pruebas unitarias aseguran que un único componente de la aplicación produce una salida correcta para una determinada entrada. Este tipo de pruebas validan la forma en la que las funciones y métodos trabajan en cada caso particular, se encargan de un único caso cada vez, lo que significa que un único método puede necesitar varias pruebas unitarias si su funcionamiento varía en función del contexto. (22)

### 3.3.1 Aplicación de pruebas unitarias al sistema.

El desarrollo y aplicación de las pruebas que determinarán el estado de la solución, estará orientado por el procedimiento explicado en la tesis de Elizabeth Quintas Sánchez (35). Dicho procedimiento define un número de etapas que se deben seguir para dar un cumplimiento satisfactorio del mismo.

**Etapas del Procedimiento**

1. Planificación de las Pruebas – PP
2. Diseño de las Pruebas – DP
3. Ejecución de las Pruebas – EP

Cumpliendo con la etapa de diseño de las pruebas se determinó realizar un total de 10 pruebas unitarias a diferentes funcionalidades por ser las más significativas para el funcionamiento de la solución:

**Pruebas unitarias**

1. Realizar auditoría.
2. Obtener tipo de documento.
3. Obtener subsistema.
4. Conectar a FTP.
5. Obtener id estado documento.
6. Procesar documento.
7. Procesar paquete documento.
8. Generar documento.
9. Obtener último id auditoría.
10. Generar XML.

A continuación se mostrará un extracto de 8 de las principales pruebas aplicadas y el resultado arrojado.

**Prueba 1:** Realizar auditoría

**Recibe**: Parámetros para registrar la auditoría.

**Resultado esperado del método:** Id de la auditoría insertada.

**Método Utilizado**: is().

**Resultado esperado de la prueba**: Si el Id devuelto es igual al Id devuelto por la secuencia antes de realizar la inserción se considera como satisfactoria.

```
$num = $db->getId($conexion, 'AUDITORIA_SEQ');
$test = new lime_test(1, new lime_output_color());
$auditoria = new AuditoriaPeer();
$result = $auditoria->insertar("127.0.0.2");
$test->is($result, $num,"la auditoria se ejecutó sin problemas");
```


**Figura 16: PU Realizar auditoría**

**Prueba 2:** Obtener id estado documento

**Descripción:** Esta prueba es la encargada de comprobar los estados de un documento.

**Recibe**: Tipos de estados posibles para un documento. RECHAZADO, ACEPTADO, ENVIADO.

**Resultado esperado del método:** Id del estado de documento enviado.

**Método:** isa_ok

**Resultado esperado de la prueba**: Si retorna una cadena se considera como satisfactoria.

```
$test = new lime_test(3, new lime_output_color());
$tiposDocumentos = array(
 'RECHAZADO',
 'ENVIADO',
 'ACEPTADO'
);
foreach ($tiposDocumentos as $value) {
 $var= EstadoDocumentoPeer::obtenerIdEstadoDocumento($value);
 $test->isa_ok($var, string, 'Tipo de documento '.$value.' encontrado');
}
```

```
ceige-ad-pc29:/home/robe/public_html/GINA_MIO # php symfony test:unit obtenerIdEstadoDocumento
1..3
ok 1 - Tipo de documento RECHAZADO encontrado
ok 2 - Tipo de documento ENVIADO encontrado
ok 3 - Tipo de documento ACEPTADO encontrado
 Looks like everything went fine.
```

**Figura 17: PU Obtener id estado documento**

**Prueba 3:** Obtener subsistema.

**Descripción:** Obtención de los subsistemas que serán encargados de procesar los documentos.

**Recibe**: Identificadores de los subsistemas a obtener.

**Resultado esperado del método**: Cadena con los subsistemas pedidos.

**Método :**isa_ok

**Resultado esperado de la prueba**: Si la cadena devuelta coincide con lo esperado se considera satisfactoria.

```
$test = new lime_test(3, new lime_output_color());
//12- EADUANA
//15- DEPOSITO TEMPORAL
```

```
//4- RECURSOS HUMANOS
$subsistemas = array(
'12',
'15',
 '4'
);
foreach ($subsistemas as $value) {
$var= ProcesarFichero::obtenerSubsistema($value);
$test->isa_ok($var->getDescripcion(), string, 'El subsistema '.$var-
>getDescripcion().' fue encontrado');
}
```

```
ceige-ad-pc29:/home/robe/public_html/GINA_MIO # php symfony test:unit obtenerSubsistema
1..3
ok 1 - El subsistema  EADUANA fue encontrado
ok 2 - El subsistema  DEPOSITO TEMPORAL fue encontrado
ok 3 - El subsistema  RECURSOS HUMANOS fue encontrado
 Looks like everything went fine.
```

**Figura 18: PU Obtener subsistema**

**Prueba 4**: Obtener tipo documento**.**

**Descripción:** Esta prueba tiene como objetivo el comprobar la obtención de los tipos de documentos nomenclados en la base de datos.

**Recibe**: Arreglo de los tipos de documentos.

**Método:** is

**Resultado esperado del método**: Cadena con el nombre del tipo documento encontrado.

**Resultado esperado de la prueba**: Si la cadena devuelta coincide con lo el tipo de documento pasado se considera satisfactoria.

```
$test = new lime_test(1, new lime_output_color());
$tiposDocumentos = array(
 'DMTransferencia',
 'ACI',
 'DMTransito',
 'RemisionEntrada',
 'ListaEmpaque',
 'CargaInicial',
```

```
 'CancelacionDeposito',
 'InformeRecepcion',
  'AcomodarMercancias',
 'Certificado Origen',
 'ACI',
 'Liberaciones',
 'ResolucionesMFP',
 'Factura',
 'RemisionSalida',
 'ActualizacionSACLAP',
 'EnmiendaSACLAP',
 'Colaboradores',

);
foreach ($tiposDocumentos as $value) {
 $var=TcTipoDocumentoPeer::obtenerTipoDocumento($value);
 $test->is($var->getDescripcion(),$value, 'Tipo de documento '.$value.'
encontrado');
```

```
ceige-ad-pc29:/home/robe/public_html/GINA_MIO # php symfony test:unit obtenerTipoDocumento
1..19
ok 1 - Tipo de documento DMTransferencia encontrado
ok 2 - Tipo de documento ACI encontrado
ok 3 - Tipo de documento DMTransito encontrado
ok 4 - Tipo de documento RemisionEntrada encontrado
ok 5 - Tipo de documento ListaEmpaque encontrado
ok 6 - Tipo de documento CargaInicial encontrado
ok 7 - Tipo de documento CancelacionDeposito encontrado
ok 8 - Tipo de documento InformeRecepcion encontrado
ok 9 - Tipo de documento AcomodarMercancias encontrado
ok 10 - Tipo de documento Certificado Origen encontrado
ok 11 - Tipo de documento ACI encontrado
ok 12 - Tipo de documento Liberaciones encontrado
ok 13 - Tipo de documento ResolucionesMFP encontrado
ok 14 - Tipo de documento Factura encontrado
ok 15 - Tipo de documento RemisionSalida encontrado
ok 16 - Tipo de documento ActualizacionSACLAP encontrado
ok 17 - Tipo de documento EnmiendaSACLAP encontrado
ok 18 - Tipo de documento Colaboradores encontrado
```

**Figura 19: PU Obtener tipo documento**

**Prueba 5:** Conectar a FTP**.**

**Descripción:** Esta prueba es la encargada de comprobar la conexión con los medios por el protocolo FTP.

**Recibe**: Datos para la conexión. Host, usuario y contraseña.

**Método:** isa_ok

**Resultado esperado del método**: Verdadero o falso.

**Resultado esperado de la prueba**: Si devuelve verdadero se considera satisfactoria la prueba, si retorna falso la prueba falla.

```
$test = new lime_test(1, new lime_output_color());
$servidorFtp='10.0.0.22';
$usuario="anonymous";
$contrasena="anonymous";
$ftp= new ConexionFtp();
$result = $ftp->conectarFtp($servidorFtp, $usuario, $contrasena);
$test->isa_ok($result, 'resource',"la conexión se realizo con exito");
```


**Figura 20: PU Conectar FTP**

**Prueba 6:** Generar documento para subsistema.

**Descripción:** Esta prueba es la encargada de comprobar la generación de un documento XML desde un subsistema.

**Recibe**: Datos requeridos para la generación del documento. Datos del XML, host, usuario, contraseña, cabecera.

**Método:** is

**Resultado esperado del método**: Verdadero o falso.

**Resultado esperado de la prueba**: Si devuelve verdadero se considera satisfactoria la prueba, si retorna falso la prueba falla.

```
$test = new lime_test(1, new lime_output_color());

$arr = array(
   "datosXML" =>array('calve' => 'dato1', 'calve2' => 'dato2ss'),
   'tipoDocumento' => 'ResolucionesMFP',
'host' => 'localhost',
   'usuario' => "anonymous",
```

```
   'contrasena' => "anonymous",
'nombreFichero' => 'ficheroGenerado',
   'cabecera' => 'cabecera',
);
$result = ProcesarFichero::generarXmlSubsistemas($arr);
$test->is($result, true, 'Doc generado');
```

```
ceige-ad-pc29:/home/robe/public_html/GINA # php symfony test:unit generarDocSubsistema
1..1
ok 1 - Documento generado para el subsitema
Looks like everything went fine.
```

**Figura 21: PU Generar documento para subsistema**

**Prueba 7:** Generar XML desde arreglo.

**Descripción:** Esta prueba es la encargada de comprobar la generación de un documento XML desde un arreglo.

**Recibe**: Datos requeridos para la generación del XML.

**Método:** is

**Resultado esperado del método**: Verdadero o falso.

**Resultado esperado de la prueba**: Si devuelve verdadero se considera satisfactoria la prueba, si retorna falso la prueba falla.

```
$test = new lime_test(1, new lime_output_color());
$arr = array(
   "datosXML" =>array('calve' => 'dato1', 'calve2' => 'dato2ss'),
   'tipoDocumento' => 'ResolucionesMFP',
'host' => 'localhost',
   'usuario' => "anonymous",
   'contrasena' => "anonymous",
'nombreFichero' => 'ficheroGenerado',
   'cabecera' => 'cabecera',
  );
$doc = new conversionAXml();

$source=$doc->convertirArrayAXML($arr, 'prueba');
$docum= new DOMDocument();
$result=$docum->loadXML($source);
```

```
$test->is($result, true,"el documento se genero correctamente");
```

```
ceige-ad-pc29:/home/robe/public_html/GINA # php symfony test:unit generarDocXML
1..1
ok 1 - Documento generado
Looks like everything went fine.
```

**Figura 22: PU Generar XML desde arreglo**

**Prueba 8:** Obtener último id auditoría**.**

**Descripción:** Obtener el identificador de la última auditoría realizada en la base de datos.

**Recibe**: Último id obtenido de la secuencia en la base de datos, mediante una consulta SQL.

**Método:** is

**Resultado esperado del método**: Último Id obtenido con el método.

**Resultado esperado de la prueba**: Si el id devuelto por el método a probar, es igual al valor devuelto por la secuencia más uno se considera satisfactoria.

```
$configuration = ProjectConfiguration::getApplicationConfiguration('tc',
'test', true);
$f = new sfDatabaseManager($configuration);
$conexion = Propel::getConnection('eaduana');
$conexion->beginTransaction();
$db = Propel::getDB('eaduana');
$num = $db->getId($conexion, 'AUDITORIA_SEQ');
$test = new lime_test(1, new lime_output_color());
$result = AuditoriaPeer::obtenerUltimoId();
$test->is($result, $num-1,"ultimo id obtenido".$num);

$conexion->rollBack();
```

```
ceige-ad-pc29:/home/robe/public_html/GINA_MIO # php symfony test:unit obtenerIdauditoria
1..1
ok 1 -  ultimo id obtenido 4952
Looks like everything went fine.
```

**Figura 23: PU Obtener último id auditoría**

## 3.4 Aplicación de pruebas funcionales.

La prueba funcional se enfoca en la óptica del usuario, las funciones son probadas proporcionando las entradas y examinando las salidas. La estructura interna del programa no es objetivo de este tipo de pruebas.

A continuación se muestra el caso de prueba diseñado para la interfaz principal de la solución, que tiene como objetivo recibir los documentos o paquetes de documentos a procesar sintácticamente por el componente y mostrar los errores correspondientes a negocio o sintaxis. El caso de prueba se lleva a cabo mediante la interfaz del módulo tablas de control del sistema GINA, el cual procesa haciendo uso del componente diferentes documentos, como resoluciones emitidas por el ministerio de finanzas y precios, documentos de colaboradores, entre otros. El diseño de este caso de prueba se realizó como parte del citado módulo y fue probado en la fase de pruebas planificadas al mismo, no detectando ninguna no conformidad con respecto al funcionamiento del componente.

| Escenario | Descripción | Respuesta del sistema | Flujo central |
|---|---|---|---|
| EC 1.1 Registrar Documentos. | Procesa y valida datos sintácticamente correspondientes al documento a procesar | Mensaje satisfactorio al cumplir el procedimiento | 1. Selecciona el módulo TC (Datos tablas de Control).<br>2.Oprime el botón Gestión Datos Tablas de Control<br>3. Oprime le botón subir documento (está identificado con una flecha hacia arriba) .<br>4. Oprime le botón Adicionar.<br>5. Busca en disco duro el archivo a subir, lo sube.<br>6. Oprime subir en la ventana emergente. |
| EC 1.2 Registrar documento error | Procesa y valida datos, registra fecha, descripción y cierra mercancías | El sistema muestra que el archivo subido tiene errores. | 1. Selecciona el módulo TC (Datos tablas de Control).<br>2. Oprime el botón Gestión Datos Tablas de Control.<br>3. Oprime le botón subir documento (está identificado con una flecha hacia arriba).<br>4. Oprime le botón Adicionar. |

| | | | |
|---|---|---|---|
| | | | 5. Busca en disco duro el archivo a subir,(en este caso un archivo con error) lo sube.<br>6. Oprime subir en la ventana emergente. |

**Descripción de las variables.**

| No | Nombre de campo | Clasificación | Valor Nulo | Descripción |
|---|---|---|---|---|
| 1 | Campo contenedor de documentos a procesar | Fichero | No. | Contiene el documento que se debe procesar |

## 3.5 Resultados de la solución.

El estudio de la problemática presentada por la AGR arrojó la afectación de los tiempos de implantación de los proyectos a desarrollar, debido a la carencia de estandarización en los documentos a procesar. A continuación se muestra un estudio realizado a subsistemas desarrollados con documentos sin estandarizar y subsistemas desarrollados con documentos a los cuales se le aplicó la utilización del lenguaje XML para su confección y las validaciones correspondientes a sintaxis realizadas por el componente de recepción electrónica mediante los XSD.

El componente de recepción electrónica está desplegado en la AGR hace más de 8 meses posibilitando para la entidad el procesamiento de 64 nuevos documentos agrupados en 26 tipos de documentos.

ACI, Acomodar mercancía, Actualización SACLAP, Cancelación Depósito, Carga Inicial, Colaboradores, Cruce Frontera, DM Transferencia, DM Tránsito, Enmienda SACLAP, Factura, Informe Recepción, Liberaciones, Lista Empaque, Remisión Entrada, Remisión Salida, Resoluciones MFP, Contacto, Datos Depósito, Datos MFP, MTI, Negocios, SACLAP, Persona Depósito.

| Tipo de documento | Complejidad | Cant validaciones sintaxis en tiempo | Cant validaciones negocio en tiempo | Tiempo desarrollo total |
|---|---|---|---|---|
| **API** | Media | 45/1mes | 95/3 meses | 5 meses |
| **Resoluciones MFP** (8 documentos) | Media | 180/8 días | 260/21 días | 29 días |

Se puede determinar que la solución cumple el objetivo primordial por la cual se decidió su creación, los tiempos de implantación de los proyectos se ve reducido debido a la reducción de los tiempos de procesamiento de los documentos propios. Los subsistemas se liberan de las validaciones de sintaxis, dejando esta tarea al componente de recepción electrónica el cual realiza la misma mediante la combinación del lenguaje XML, las plantillas XSD y el lenguaje PHP. Teniendo los subsistemas a su cargo la realización de las validaciones de negocio sobre arreglos de datos estructurados, haciendo uso de varias funciones y clases incorporadas en el lenguaje a utilizar y de probados resultados.

## 3.6 Conclusiones del capítulo.

La implementación de la solución estuvo guiada por varios estándares de codificación y un tratamiento de errores, con el objetivo de evitar problemas en tiempo de ejecución y un entendimiento más claro del código generado. La aplicación de diferentes técnicas de validación dio la posibilidad de contar con una solución estable y de un alto nivel de calidad, entre las técnicas dinámicas utilizadas se encuentran la aplicación de pruebas unitarias las cuales dejaron funcionalmente el componente listo para el despliegue. Los tiempos de implantación de los proyectos se vieron mejorados dadas la implantación de los estándares y la eliminación a los subsistemas de la responsabilidad de validaciones de sintaxis. Por lo que se determina que la solución cumple con los objetivos propuestos.

# CONCLUSIONES

El estudio de la problemática encontrada en la AGR en conjunto con diversos conceptos y su utilización, se determinó la necesidad de la creación de un componente para integrarlo con el sistema GINA que se encargara de gestionar el proceso de envío y/o validación de documentos electrónicos. El desarrollo del componente estuvo guiado por la metodología de desarrollo y la arquitectura definida por el Departamento de Soluciones para la Aduana.

El componente obtenido después de las validaciones correspondientes, permitió a la AGR el contar con un mecanismo de interacción con entidades externas mediante la recepción de documentos electrónicos y el poder realizar diferentes procesos aduanales en el entorno del mismo, dichas validaciones permitieron tener una solución fiable y que cumplía a cabalidad todo lo requerido por el cliente.

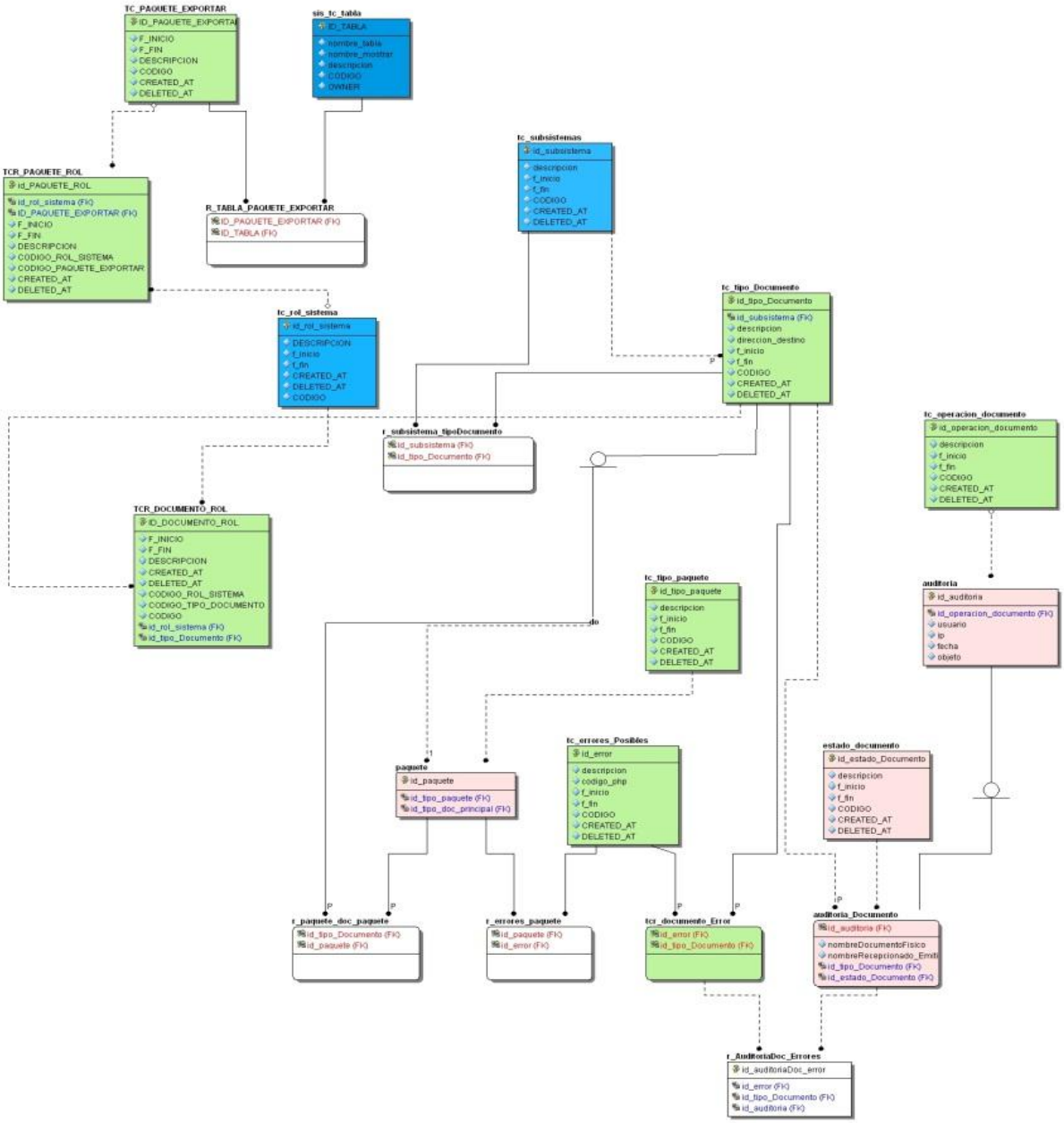
# RECOMENDACIONES

Culminar la etapa de captura de errores referentes a la validación de documentos para su posterior utilización en el sistema.

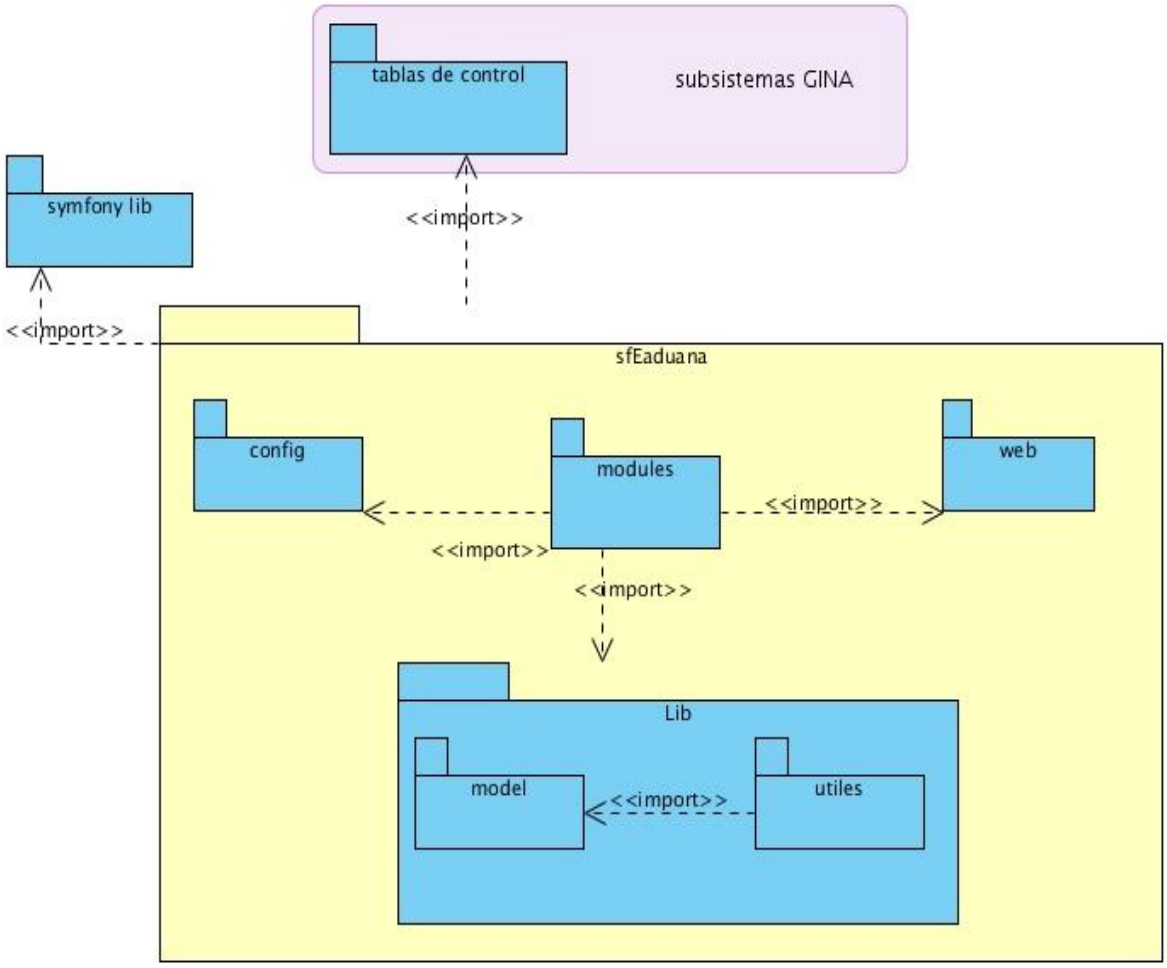
Desarrollar un sistema de firmas digitales para la autenticidad de los documentos envueltos en el proceso de recepción.

# ANEXOS

## Anexo1 Diagrama físico de base de datos.



## Anexo 2: Diagrama de paquetes.



## Anexo 3: Especificación de requisito Procesar fichero XML.

Descripción textual del requisito en el documento **Especificación de requisitos funcionales 1.0**
Este requisito es iniciado por algún otro requisito como Recibir fichero mediante Servidor FTP como ejemplo.

| | | |
|---|---|---|
| **Precondiciones** | El fichero se ha recibido de forma correcta por cualquiera de las vías posibles. | |
| **Flujo de eventos** | | |
| **Flujo básico Procesar Fichero XML** | | |
| **No** | **Actor** | **Sistema** |
| | | Lee el fichero y comprueba que sea un fichero XML.<br><br>En caso contrario ver *Flujo Alterno 1.a "Mostrar mensaje".* |
| | | Verifica si existe un fichero en estado aceptado con el mismo nombre que el que se está procesando.<br><br>En caso de existir ver *Flujo Alterno 2.a "procesar nuevamente el fichero".* |
| | | Compara el fichero XML con el XSD correspondiente, donde realiza la validación del fichero recibido. En caso de encontrar errores en el chequeo de sintaxis ver *Flujo alterno 3.a "errores en la comparación con el XSD".* |
| | | Registra los datos auditables de los documentos. Como son usuario, ip, fecha y hora en que se subió a la |

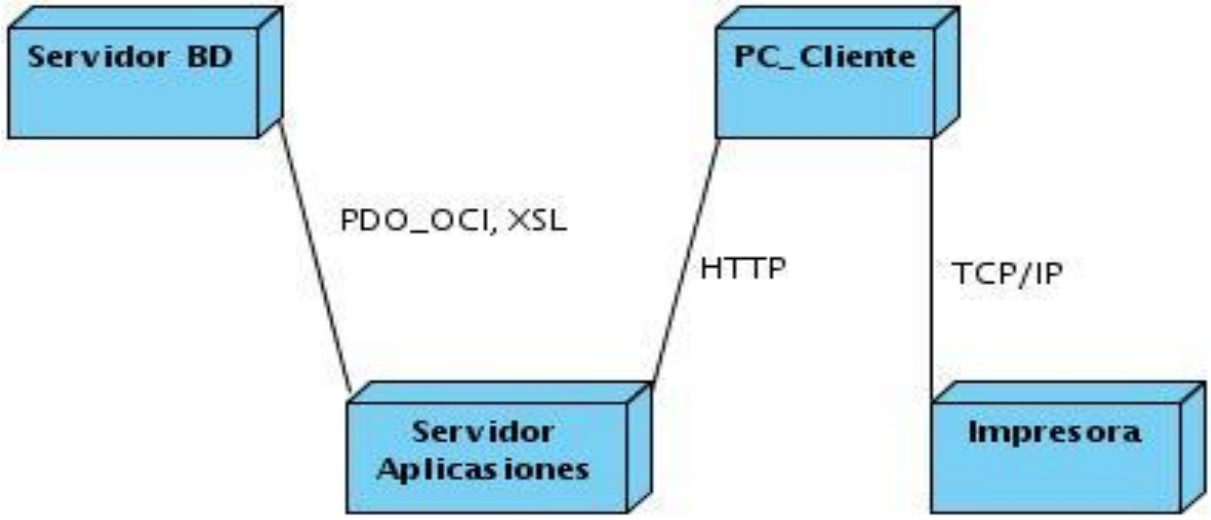
| | | |
|---|---|---|
| | | aplicación de la aduana, estado del documento, tipo de documento, errores que presentó. |
| | | Selecciona al subsistema que le corresponde dicho fichero, y ejecuta el método encargado de validar con el negocio dicho documento. |
| | | Recibe la respuesta de la validación del subsistema del fichero.<br><br>En caso de presentar errores ver *Flujo Alterno 6.a "errores en la comparación con el XSD".* |
| | | Se muestra un mensaje "El documento se ha guardado satisfactoriamente". |
| | Acepta el mensaje. | |
| | | Guarda el fichero de forma física en el servidor en el directorio correspondiente a los ficheros aceptados. |
| | | Se le añade el estado en que quedó el documento después de ser procesado (Aceptado o No Aceptado). |
| | | Concluye el requisito. |
| Pos-condiciones | | |
| | Procesado el fichero correctamente. | |
| | Se muestra la respuesta del procesamiento. | |
| **Flujos alternativos** | | |
| **Flujo alternativo 1a. Mostrar mensaje** | | |
| **No** | **Actor** | **Sistema** |

| | | |
|---|---|---|
| 1 | | Muestra un mensaje "*El fichero no es un documento XML*". |
| 2 | | Se guarda el error. |
| 3 | | Concluye el requisito. |
| Pos-condiciones | | |
| | Se muestra el error al usuario. | |
| | Se registra el error. | |
| **Flujo alternativo 2a. Procesar nuevamente el fichero** | | |
| **No** | **Actor** | **Sistema** |
| 1 | | Muestra el mensaje "Ya existe un fichero aceptado con ese nombre, desea sustituirlo" |
| 2 | Escoge si desea sustituir el fichero. | |
| 3 | | En caso de seleccionar la opción SI, continua en el punto 3 del flujo principal. |
| 4 | | En caso de seleccionar la opción NO, Concluye el requisito. |
| Pos-condiciones | | |
| | Tomada la decisión de sustituir o no, el fichero existente. | |
| **Flujo alternativo 3a. Errores en la comparación con el XSD.** | | |
| **No** | **Actor** | **Sistema** |
| | | Guarda el fichero de forma física en el servidor en el directorio correspondiente a los ficheros erróneos. |
| | | Guarda los errores registrados en la |

| | | |
|---|---|---|
| | | base de datos del sistema. |
| | | Muestra al usuario en una ventana emergente la lista de errores encontrados en la comparación con el XSD o los errores emitidos por el método del subsistema encargado de validar dicho fichero. |
| | Acepta el Mensaje | |
| | | Continúa en el punto 10 del flujo básico. |
| **Pos-condiciones** | | |
| | Almacenados los ficheros. | |
| | Mostrado al usuario la lista de errores encontradas. | |
| | | |
| **Validaciones** | | |
| | Se validan los datos según lo establecido en el Modelo Conceptual. | |
| **Relaciones** | **Requisitos Incluidos** | **No aplicable** |
| | Extensiones | No aplicable |
| **Requisitos especiales** | No aplicable | |
| **Asuntos** | No aplicable | |

## Anexo 4 Modelo de despliegue

Este es el modelo de despliegue definido para el proyecto GINA, del cual la solución creada forma parte con un plugin o paquete.



## Anexo 5 Certificado de aceptación

Aduana General de la República de Cuba

### Certificado de Aceptación

Yo Nancy Rodríguez Calderín, como Especialista Principal del Centro de Automatización para la Dirección y la Información, certifico que la solución *Componente para el Intercambio de Información Electrónica Peruana*, cumple con todas las funcionalidades solicitadas por nuestra parte y está desplegado en nuestra institución en fase de prueba con resultados satisfactorios.

Se firma como constancia el día 7 del mes 6 del año 2012.

Nancy Rodríguez Calderín
Esp. Principal. CADI

# GLOSARIO DE TÉRMINOS

**Aduana:** Es una oficina pública de constitución fiscal establecida generalmente en costas y fronteras.

**AGR:** Aduana General de la República de Cuba.

**CSS:** *Cascade Style Sheet*u Hojas de Estilos en Cascada es un lenguaje creado para controlar la presentación de los documentos electrónicos definidos con HTML y XHTML.

**Ext JS:** Es una librería JavaScript ligera y de alto rendimiento, compatible con la mayoría de navegadores que nos permite crear páginas e interfaces web dinámicas.

**JavaScript:** Es un lenguaje de programación interpretado utilizado principalmente en la realización páginas web. Presenta una sintaxis semejante a la del lenguaje Java y el lenguaje C.

**PHP:** *PHP Hypertext Pre-processor e*s un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas.

**Symfony:** *Framework* de trabajo para la realización de aplicaciones web escrito en PHP 5.

**UCI:** Universidad de las Ciencias Informáticas.

**XML:** *eXtensible Markup Language*, es un metalenguaje extensible de etiquetas desarrollado por el *World Wide Web Consortium*(W3C)

**BPMN**: Business Process Modeling Notation

# REFERENCIAS BIBLIOGRÁFICAS


1. **Estado, Consejo de.** Decreto Ley No 162. La Habana : s.n., 1996.

2. **Kyoto, Convenio de.** Aplicación de la tecnología de la informacion. Capitulo 7.

3. **IEEE, Engineers Institute of Electrical and Electronics.** IEEE Standard Computer Dictionary New York, NY : s.n., 1990.

4. **ICT, Laboratory of Interactive and Cooperative Technologies.** Laboratory of Interactive and Cooperative Technologies. *ICT.* [Online] UDLAP, 2011. [Cited: 11 16, 2011.]

5. **ISO, Comité Técnico TC de la Organización Internacional de Estandarización.** ISO/DIS 19101. 2006. (ISO, 2001).

6. **Madsen, Mark R.** The Role of Open Source in. 2009.

7. **Integra2.** Integra2. [Online] 2002. [Cited: 12 5, 2011.].

8. **Melendi, Palacios David.** Area ingienería telemática Universidad de Oviedo. 2011.

9. **Rodríguez, Abel Rionda.** Modelo de integración de presentación. 2011.

10. **Melendi, Palacios David.** Modelo de integración de datos. 2011.

11. **Rodríguez, Abel Rionda.** Modelo de integración funcional. Oviedo : s.n., 2011.

12. **Sacha, Krakowiak.** Middleware. What is middlware. [Online] 2009. [Cited: 12 06, 2011.]

13. **Española, Real academia.** Diccionario. http://buscon.rae.es/draeI/. [Online] 2011.

14. **ISO, Organización mundial denormalización.** Estandar. 2006.

15. **(BSI), British Standard Institute.** Standar. 2008.

16. **Arellano, Ana Alejandra Febres.** RRPP/ UNCTAD-SIDUNEA. *Conociendo al SIDUNEA* -

17. **VUCE.** Ventanilla Única de Comercio Exterior. [Online] 2010. https://www.vuce.gob.pe/.

18. **SKC, Services Korean Customs.** Korea Customs Service. Customs Automation. [Online] 2011.

19. **Modelo Aduana, Dpto Soluciones.** *Modelo de desarrollo.* 2012.

20. **w3c.** w3c. [Online] 2010. [Cited: 02 25, 2012.]

21. **Rodríguez, José Antonio Cobo.** Línea Base Arquitectónica para el Polo Sistemas Tributarios y de Aduanas. 2008.

22. **Potencier, Fabien.** Symfony la guía definitiva. 2008.

23. **ExtJS, Comunidad de Desarrollo de Ext.** Ext JS en Español. [Online] 2011. [Cited: 02 25, 2012.]

24. **Alvares, Mario Alberto.** Estándares de soluciones a los problemas de presentación en el sistema CERES. 2009.

25. **Aduana Repositorio, Dpto Soluciones para la.** Repositorio DSA. [Online] 2012. http://10.52.17.252:5700/Repo/AGR.

26. **Martínez, Jenni Manso.** Procedimiento para la Ingeniería de Requisitos en el Departamento de Desarrollo de Soluciones para la Aduana del CEIGE. La Habana : s.n., 2010.

27. **Aduana, Proyecto de soluciones para la.** Modelo de Procesos de Negocio con BPM. [Online] 2011.
28. **Sommerville, Ian.** Ingeniería del Software. Madrid (España) : s.n., 2008.

29. **Calisoft.** Calisoft. *Calisoft.* [Online] UCI, 2011. http://calisoft.uci.cu

30. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. s.l. : Addison Wesley, 1999.

31. **ICT. Interactive and Cooperative Technologys**. [Online] 2008. [Cited: 02 22, 2012.]

32. **OMG, Unified Modeling Language.** *OMG Unified Modeling Language.* 1999.

33. **Juristo, Natalia, Moreno, Ana M. and Vegas, Sira.** *Técnicas de Evaluación de Software.* 2006.

34. **Oré, Ing. Alexander.** [Online] 2009. [Cited: 05 06, 2012.] http://www.calidadysoftware.com/

35. **Sánchez, Elizabeth Quintas.** Procedimiento para la realización de pruebas de unidad dentro del proyecto Sistema. 2010.

36.**Santillán, Ing Jesús RobertoLópez.** [Online] [Cited: ] http://www.buenastareas.com/ensayos/Pruebas-De-Caja-Negra/1477804.html .

37. **Rodríguez, Abel Rionda.** Tipos de Integración. 2011.

# BIBLIOGRAFÍA CONSULTADA

**Potencier, Fabien. 2009.** *El tutorial Jobeet.*, *guía definitiva de Symfony*2009.

**Meléndrez, Edelsys Hernández**. 2006. Cómo escribir una tesis. La Habana: s.n., 2006.

**Collin, Peter.** Publishing Dictionary of Computing, 4th Edition.

**Martin Fowler**,Patterns of Enterprise Application Architecture