Universidad de las Ciencias Informáticas

Facultad 15

Trabajo de Diploma para optar por el

Título de Ingeniero en Ciencias Informáticas

Tema: Modelado de un sistema informático para los Depósitos Temporales de Frontera de la Aduana General de la República.

Autores:

Roxana Fonseca Arias

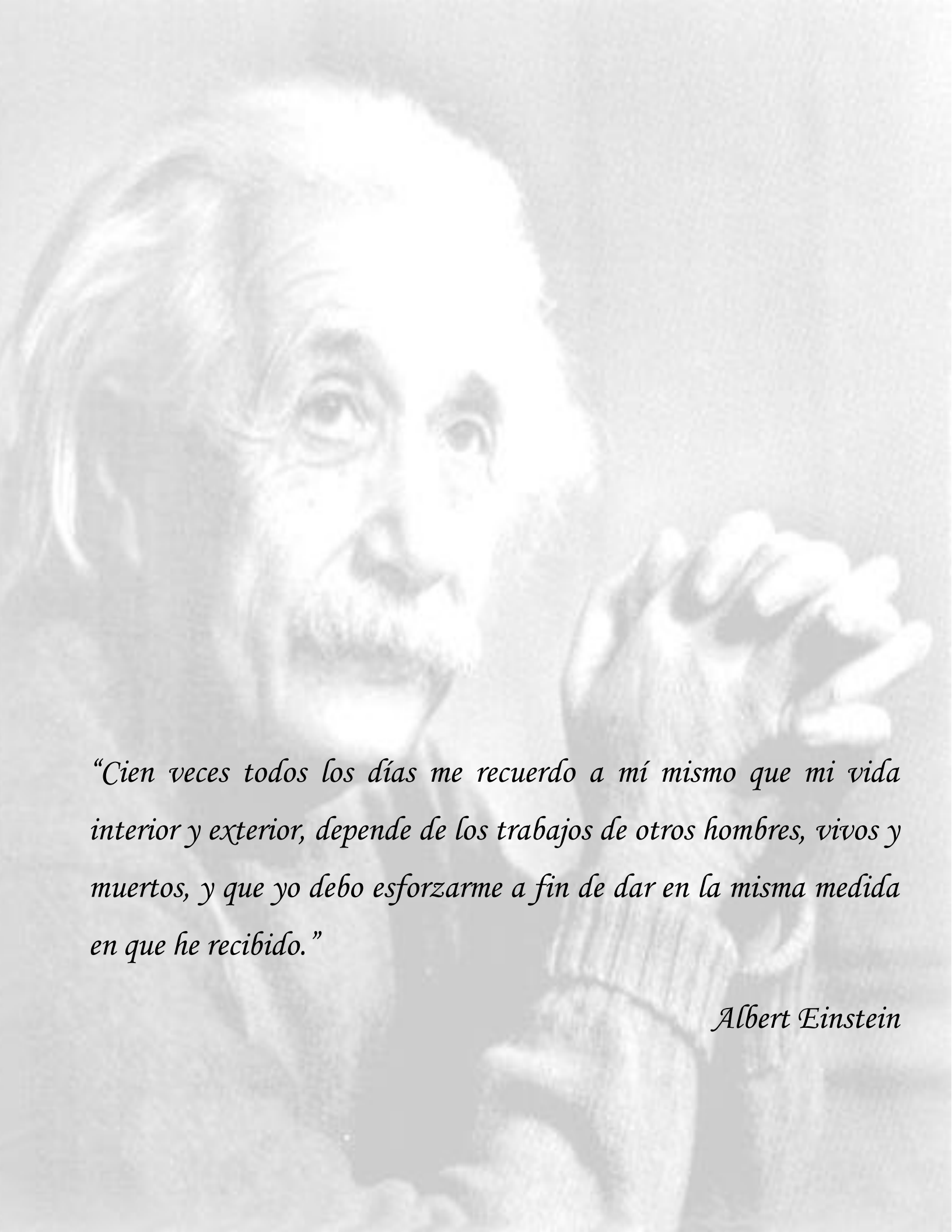
Ariel Calzadilla del Sol

Tutor:

Ing. Lianet Pineda De la Nuez

Ciudad de la Habana, Junio 2010

"Año 52 de la Revolución"



"Cien veces todos los días me recuerdo a mí mismo que mi vida interior y exterior, depende de los trabajos de otros hombres, vivos y muertos, y que yo debo esforzarme a fin de dar en la misma medida en que he recibido."

Albert Einstein

Declaración de Autoría

Declaramos ser los únicos autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____ _____

Roxana Fonseca Arias

Firma del Autor

Ariel Calzadilla Del Sol

Firma del Autor

_____

Ing. Lianet Pineda De la Nuez

Firma del Tutor

Agradecimientos

A todas las personas que están y los que ya no, pero que influyeron en mi proceso educativo:

… A mi tutora Lianet por estar ahí siempre que la necesité.

…A mi hermana Katia por su apoyo y cariño incondicional.

…A mi papito José por guiarme siempre por el camino correcto.

…A mi abuela Milagros por haber sido mi madre y abuela a la vez.

…A mi novio Ariel porque desde que le conocí ha hecho de todo por verme reír y porque me ha tendido su mano para no caerme ante las dificultades.

… A todo el que me ayudó y confió en mí, de corazón, muchas gracias.

Roxana

…A todos los educadores que han contribuido a mi formación como persona y profesional.

…A mis padres por ser mi ejemplo e inspiración.

…A mi novia que me animó cuando más necesitaba.

…A todo aquel que me ayudó a resolver cualquier problema o aclarar una duda.

…A la Revolución por permitirme esta posibilidad; a todos muchas gracias.

Ariel

Dedicatoria

Roxana:

A la memoria de Magalis, mi madre querida.

A mi familia por ser el centro de mi existencia, porque sin ellos no hubiera podido cumplir este sueño.

Ariel:

A la memoria de mi abuelo Jesús.

A mis padres.

Resumen

La Aduana General de la República de Cuba se ha trazado como meta la automatización de la mayoría de los procesos de sus disímiles espacios de trabajo. La pretensión más importante del área de Depósitos Temporales es llevar un control real del inventario de las mercancías dentro del depósito y en el destino final, para que las inspecciones sean efectivas y pueda evitarse el fraude y la malversación de bienes. El presente trabajo de diploma tiene como objetivo diseñar el subsistema Depósitos Temporales de Frontera para el sistema Gestión Integral de Aduanas que facilite la gestión de las operaciones que se ejecutan en los depósitos temporales de frontera.

Con este fin se aplicó la metodología de desarrollo RUP, la cual se vale de UML para elaborar todos los artefactos. Para comprender el negocio se realizó el modelado de procesos con la notación BPMN utilizando la herramienta Visual Paradigm. Partiendo de las técnicas de captura y validación de requisitos se definieron los requerimientos funcionales del sistema, se realizaron los diagramas de clases del diseño así como la validación del mismo, además de otros elementos que son abordados en el transcurso y desarrollo del trabajo.

Palabras claves: depósitos temporales, frontera, mercancías, aduana.

Tabla de contenidos

Introducción

El Comercio Internacional necesariamente implica la circulación entre fronteras tanto de mercancías, como de personas y bultos postales. La capacidad de mover artículos a través de fronteras internacionales de forma rápida, segura y a un costo razonable y predecible, puede otorgar a un país ventajas notables con respecto a las competencias e impulsar el desarrollo socioeconómico del mismo. Junto a estos innegables beneficios el tráfico internacional conlleva riesgos de diversos tipos que pueden comprometer el desarrollo de la economía nacional, la salud y la seguridad del país. En este contexto las aduanas desempeñan un rol crítico, no solo para lograr las metas gubernamentales, sino también en la efectividad de los controles que aseguren las recaudaciones, el cumplimiento de la legislación nacional y en garantizar la protección y seguridad de la sociedad.

Las mercancías que circulan hacia nuestro país y desde él hacia otros, permanecen temporalmente en almacenes autorizados por la Aduana General de la República de Cuba (AGR), hasta continuar su curso al destino final. Por ende la AGR necesita llevar un control de las operaciones de importación y exportación que se realizan en los Depósitos Temporales, debido al volumen de mercancía que en estos se almacena.

Un Depósito Temporal no es más que un lugar o instalación autorizado por la AGR, para el almacenamiento de mercancías de toda clase, cualquiera que sea su naturaleza, cantidad, país de origen, destino o procedencia; que esperan un destino final. (1)

Los Depósitos Temporales, según su ubicación, se clasifican en:

1. De Frontera: Están ubicados en áreas de los puertos y aeropuertos.
    - Portuarios: Situados dentro del área del recinto portuario.
    - Aeroportuarios: Situados dentro del área del recinto aeroportuario.
2. Interiores: Están ubicados en áreas fuera de los puertos y aeropuertos.

En el Centro de Automatización para la Dirección y la Información de la AGR (CADI) se ha implementado un módulo dentro del Sistema Único de Aduanas (SUA), denominado Almacén, que automatiza solamente

la extracción y recepción de mercancías bajo los regímenes aduaneros[1] de tránsitos[2] y transferencias[3] entre depósitos temporales, por lo cual no implementan todos los requerimientos necesarios para la gestión y control de la totalidad de los procesos que se efectúan, tales como el control en el destino de las mercancías cuando van hacia el almacén del importador. Este sistema al ser desarrollado en PHP, utilizando el paradigma de la programación estructurada, complejiza su mantenimiento y reutilización; además no se acopla con la nueva arquitectura definida para el sistema Gestión Integral de Aduanas, por lo que es obsoleto e insuficiente.

Entre las desventajas de esta solución está el paradigma de programación utilizado, es inapropiado y poco robusto. El principal inconveniente de este método de programación, es que se obtiene un único bloque de programa, que cuando se hace demasiado grande puede resultar problemático su manejo.

Debido a la incapacidad del módulo Almacén para gestionar la totalidad de los procesos que se llevan a cabo en los depósitos temporales de frontera, así como para integrarse al modelo arquitectónico establecido en el Departamento de Soluciones para la Aduana, el trabajo de diploma se propone resolver el siguiente **problema:** inexistencia de un sistema integral que gestione y controle la totalidad de los procesos de los Depósitos Temporales de Frontera de la Aduana General de la República de Cuba.

El **objeto de estudio** está enmarcado en: los procedimientos, regulaciones, normas y leyes que rigen los Depósitos Temporales de la Aduana General de la República de Cuba; y el **campo de acción** en: procesos de los Depósitos Temporales de Frontera de la Aduana General de la República de Cuba.

---

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

[2] Régimen aduanero bajo el cual son transportadas las mercancías de una aduana a otra en territorio nacional bajo control aduanero.

[3] Régimen aduanero bajo el cual son transportadas las mercancías de un depósito a otro de la misma aduana en territorio nacional bajo control aduanero.

Con el propósito de dar solución al problema expuesto se persigue como **objetivo general:** modelar un sistema informático que facilite la gestión y control de los procesos de los Depósitos Temporales de Frontera de la Aduana General de la República de Cuba.

Del objetivo general se derivan los siguientes **objetivos específicos:**

- Elaborar el marco teórico relativo a las soluciones de software que gestionen depósitos de las aduanas y el diseño de sistemas informáticos.
- Modelar los procesos de negocio de los Depósitos Temporales de Frontera de la AGR.
- Realizar la captura y validación de los requerimientos funcionales del sistema.
- Realizar el diseño del sistema.

Con el fin de dar cumplimiento a los objetivos del presente trabajo, se establecieron las siguientes **tareas de investigación**:

- Análisis del estudio del estado del arte de las soluciones de software nacionales e internacionales que gestionen y controlen las operaciones de los Depósitos de las Aduanas para establecer una posición al respecto.
- Análisis de la metodología de desarrollo de software y las tecnologías utilizadas en el proyecto para contribuir a la confección del marco teórico.
- Análisis de las normas, procedimientos y regulaciones que rigen los Depósitos Temporales de la AGR para determinar las reglas del negocio.
- Modelado de los procesos de negocio de los Depósitos Temporales de Frontera de la AGR, para obtener un mayor entendimiento e identificar requerimientos funcionales.
- Desarrollo de entrevistas a los especialistas de la AGR para efectuar la captura de los requerimientos del sistema.
- Elaboración de los artefactos del diseño para favorecer a la concepción del diseño del sistema.

Los **resultados esperados** con el desarrollo de la investigación constituyen los siguientes:

- Documentación del Estado del Arte.
- Modelo de Procesos del Negocio.
- Especificación detallada de los requerimientos funcionales del sistema.
- Modelado del diseño del sistema para su posterior implementación.

El trabajo está constituido por cuatro capítulos en los que se encuentra el contenido distribuido. El primer capítulo muestra la fundamentación teórica de la investigación: el estado del arte de algunas de las soluciones informáticas existentes para la gestión de inventario y el control de las mercancías en los depósitos temporales; las tecnologías, lenguajes y herramientas utilizadas, técnicas en la ingeniería de requisitos, y los patrones de software más utilizados. El capítulo dos se refiere a las características del sistema, donde se expone y se describe una panorámica global de los procesos que se ejecutan en los Depósitos Temporales Frontera de la Aduana General de la República de Cuba, se analiza la ejecución de los mismos, se presentan las técnicas utilizadas para la captura y validación de los requerimientos, y se definen los requerimientos funcionales, concluyendo con la presentación de los beneficios esperados con esta solución. El tercer capítulo contiene los patrones de diseño utilizados en el desarrollo de la solución, los diagramas de clases del diseño con estereotipos Web, los diagramas de secuencia y la construcción del modelo físico de datos. El capítulo cuatro se dedica a la explicación de algunas de las métricas más referenciadas y relevantes para la validación del diseño de software orientado a objetos y la exposición de los resultados de su aplicación al modelo presentado.

Capítulo 1: Fundamentación Teórica

Introducción

En el capítulo se realiza un estudio del estado del arte de las soluciones informáticas nacionales e internacionales para la gestión y control de las operaciones de los depósitos temporales de las aduanas. Se detallan aspectos importantes de los estilos arquitectónicos, patrones de diseño, patrones de arquitectura, herramientas y tecnologías utilizadas en el proyecto. También se realiza un estudio de algunas de las técnicas de captura y validación de requisitos existentes.

1.1 Generalidades de los Depósitos Temporales de Frontera

En la actualidad el intercambio comercial entre naciones ha alcanzado altos niveles de movimiento de mercancías atendiendo a su frecuencia, volumen y variedad. Debido a que las importaciones y exportaciones de mercancías influyen en el desarrollo económico y social del país, es de vital importancia llevar un estricto control de todos los detalles relativos a las mismas mientras estén bajo control aduanero.

La inspección, vigilancia y registro de las mercancías almacenadas dentro del depósito, que exporta e importa el país, incluso de algunas que nunca llegan a formar parte de nuestro patrimonio, con su adecuado control, así como velar por que ciertas legislaciones sean cumplidas son responsabilidades de los especialistas, inspectores y jefes de aduana que laboran en los Depósitos Temporales de la Aduana General de la República de Cuba.

Entre las principales operaciones que intervienen en los procesos de importación y exportación se encuentran: Carga, Descarga, Clasificación, Extracción y Almacenamiento de mercancías.

Estas operaciones consisten respectivamente en:

Carga: operación que consiste en colocar en un medio de transporte, las cargas; desde el muelle, la pista, almacén de depósito o desde otro medio de transporte, que serán transportadas por éste, para ser llevadas a su destino final.

Descarga: operación que consiste en extraer de un medio de transporte las cargas transportadas por éste, para ser depositadas sobre el muelle, la pista, almacén de depósito o sobre otro medio de transporte.

Clasificación: conjunto de operaciones dirigidas a identificar y separar las cargas, a partir de las marcas que contengan sus embalajes contra las marcas consignadas en el Conocimiento de Embarque.

Almacenamiento: operación que consiste en colocar las mercancías en el local del Depósito Temporal, atendiendo a las normas establecidas según su naturaleza, características, marcas y números de embalaje.

Extracción: acción y efecto de retirar las mercancías del recinto portuario o aeroportuario para su traslado hacia el destino final.

En las operaciones de los Depósitos Temporales intervienen:

1. **Operadores portuarios y aeroportuarios:** persona natural o jurídica que en virtud de la autorización que le ha sido otorgada, administra y opera un almacén de depósito bajo control aduanero. Se compromete mediante la firma de un contrato a prestar servicios de carga, descarga, recepción, clasificación y entrega de las mercancías.

2. **Inspector de Aduana:** controla las actividades de importación y exportación que puede incluir: examen integral de todas las operaciones realizadas a fin de asegurarse del cumplimiento de todas las formalidades aduaneras.

3. **Declarante (Agente de Aduanas o Apoderado):** toda persona natural o jurídica que hace una declaración en Aduana o en nombre de la cual esta declaración es hecha. **Agente de Aduanas o Apoderado** que se ocupa de realizar los trámites aduaneros para poder obtener las mercancías y llevarlas hasta su destino final en la importación o ser exportadas.

1.2 Soluciones informáticas para la gestión y control de las operaciones en los depósitos bajo potestad del Servicio de Aduanas

Hoy en día, donde las tecnologías han avanzado tanto, poseer un sistema automatizado que sirva de apoyo al control de las mercancías dentro de los depósitos, es algo muy común y lo mejor para evitar pérdidas innecesarias y desvío de valores.

A continuación se realiza un análisis de algunas de las soluciones extranjeras y cubanas existentes actualmente, utilizadas para la gestión de las operaciones de los depósitos que se encuentran bajo control aduanero.

1.2.1 Soluciones Extranjeras

S4 tr@nsERP

Los distintos módulos de depósito disponibles están basados en los requerimientos de cada una de las figuras aduaneras que gestionan, de modo que, además de los controles clásicos de un almacén, se incorporan las transmisiones EDI[4] y los diferentes listados de control requeridos por la Aduana. Pueden funcionar como una aplicación independiente o integrada con el resto de módulos de S4 tr@nsERP.(2)

Módulos:

- Almacén de Depósito Temporal (ADT).
- Depósito Aduanero.
- Depósito Distinto del Aduanero.
- Depósito Fiscal.

[4] Intercambio electrónico de datos: intercambio entre sistemas de información, por medios electrónicos, de datos estructurados de acuerdo con normas de mensajes acordadas.

- Depósito para Restitución.

Características Técnicas:

- Multiempresa, Multidelegación.
- Parametrizable:
    - En acceso y búsqueda de la información.
    - En trazabilidad.
    - En funcionalidad y modelización de datos.
    - En reportes y análisis.
- Control del acceso a la información por módulos de seguridad.
- Comunicación por mail, fax e internet.
- Mensajería electrónica integrada.
- Integración:
    - Con su gestión de almacén.
    - Con su web, personalizado para clientes, usuarios y colaboradores.
    - Con otros módulos de S-4.
- Construido con herramientas Microsoft, SQL Server como almacén de datos. Intercambio con XML y herramientas de Office System de Microsoft para importación y exportación de información.

Características específicas del módulo Almacén de Depósito Temporal

- Cambios de ubicación.
- Recepción de tránsitos.

- Confección de tránsitos NCTS[5] (*del inglés, "New Computerized Transit System"*).
- Mensajes EDI.
- Registro de entradas y salidas.
- Control de vencimiento de declaraciones sumarias[6]. (3)
- Listado de existencias.
- Contabilidad de existencias.
- Trazabilidad.
- LAME (Local Autorizado Mercancía de Exportación).
- Certificados de recepción.
- Listas de carga.
- Incidencias e informes para Calidad.
- Integración con módulo de Aduanas.
- Integración con facturación emitida y soportada (costes).
- Control de cierre de expedientes.
- Análisis de rendimientos previstos y reales.

[5] Sistema basado en el intercambio de mensajes electrónicos.

[6] Relación de la carga del medio de transporte que ha de presentarse a la aduana por el responsable de la conducción de las mercancías o por medio de su representante, en un plazo máximo de 24 horas después de la llegada del medio de transporte a territorio aduanero.

- Exportación de datos a Office System.

EV Sys

Sistema de Gestión de Recintos Aduaneros y Control de Depósitos Aduaneros Industriales

El sistema EV Sys, desarrollado por Softway, fue creado al tomar como base el Acto Declaratorio Ejecutivo (ADE) Coana/Cotec 02/2003 que especifica el funcionamiento del sistema informatizado exigido, así como las modificaciones realizadas por el ADE Coana/Cotec 01/2005, y permite controlar totalmente el Depósito Aduanero Industrial. (4)

Esta solución está preparada para atender también toda la operación del régimen DAC (Depósito Aduanero Certificado), especificado en las Instrucciones Normativas (IN) 266/02, 322/03, 362/03 y ADE Coana 02/2003, que considera la mercancía como exportada a partir del momento en que se la admita en este régimen, incluso si permanece en territorio nacional, y brinda un sin fin de ventajas fiscales y logísticas a la operación.

La solución está compuesta por los módulos:

- **Gestión del Recinto:** módulo de control operativo del recinto con controles de Normativa, Recepción / Expedición, Almacén y Control Aduanero;
- **Informes Oficiales - ADE Coana/Cotec 02/2003:** módulo capaz de conciliar, procesar y poner a disposición los 59 informes especificados en el adjunto único del Acto de Declaración, con todas sus variaciones;
- **Industrialización en Recintos Aduaneros (IN 241/02):** atiende los controles exigidos del recinto aduanero por la referida Instrucción Normativa;
- **Controles DAC IN 266/02:** atiende los controles exigidos del recinto aduanero por la referida Instrucción Normativa;
- **Módulo del Beneficiario de la Industrialización - MB:** atiende los controles exigidos por el ADE Coana/Cotec 02/2003 referentes a los beneficiarios industriales instalados dentro del recinto aduanero industrial.

La solución está totalmente adecuada a las legislaciones que rigen el sector, tales como: Acto de Declaración Ejecutivo Coana/Cotec No. 2 del 26/09/2003; Instrucción Normativa No. 241 del 06/11/2002; Instrucción Normativa No. 266 del 23/12/2002; Acto de Declaración Ejecutivo Coana No. 2 del 13/01/2003. Se puede comercializar cada módulo por separado y todos atienden el artículo 4.o del ADE Coana/Cotec 02/2003 que establece que todas las correcciones y modificaciones realizadas por el sistema de control deben tratarse como un nuevo registro, derivado del original.

Existe la posibilidad de hospedar el sistema en las instalaciones de Softway y minimizar la inversión en hardware/software, sin exponer su red a accesos externos. El DCS (Data Center Softway) cuenta con una infraestructura completa para poner a disposición su aplicación las 24 horas del día los 7 días de la semana. Sitio Web y equipamientos redundantes, administradores de base de datos, seguridad y redes exclusivas son algunos de los factores que aseguran un alto nivel de calidad y tranquilidad a sus clientes, tales como: Eadi Mesquita, Dell Computadores, Cosan, Flextronics, Visteon, Caterpilar, Kodak entre otros.

Alquiler de Software: Softway ofrece otra opción ventajosa para la utilización de los sistemas: el alquiler. Por medio de una mensualidad, los clientes de Softway pueden utilizar los sistemas en su totalidad, sin necesidad de realizar una inversión inicial en la compra de sus licencias.

CLIP™

Esta herramienta ha sido desarrollada por la empresa Cotecna, líder internacional en servicios de inspección comercial, seguridad y certificación, que lleva 30 años siendo una autoridad mundial en esta industria. (5)

De conformidad con los requerimientos de control aduanero fue creada esta solución que no sólo proporciona una estimación instantánea y en tiempo real de los niveles de inventario, sino que asegura una aplicación continua de las medidas de seguridad y un control aduanero a lo largo de toda la cadena logística del depósito aduanero.

CLIP™ utiliza un sistema de administración de transporte, logística y almacenaje; una solución efectiva que reúne en una sola plataforma informática el almacenaje y la logística, asegurando una coexistencia óptima entre el transporte, el manejo de espacios, la elaboración de informes y las operaciones de control aduanero.

Depósito Aduanero

Aplicación desarrollada por USINAD, S.L. especialmente diseñada y desarrollada para la explotación de Depósitos Aduaneros en cualquiera de sus facetas: (6)

- Depósito Aduanero (público o privado).
- Depósito Distinto del Aduanero.
- Almacén de Avituallamiento.
- Almacén de Depósitos Temporal (ADT).
- Local Autorizado para Mercancía de Exportación (L.A.M.E.).
- Almacenaje Mercancías Nacionales.

Aplicación "Abierta": aunque, en principio, la aplicación se contempla como estándar, existe la posibilidad de efectuar las modificaciones o ampliaciones en la medida en que los usuarios requieran para la perfecta integración de la aplicación con la general de la empresa mediante la importación o exportación de ficheros con la información necesaria.

Gestión Telemática de Todos los Documentos E.D.I. (Vía Internet): el envío y recepción de documentos EDI es totalmente transparente para el usuario.

Además de permitir el control de las mercancías del almacén, llevando la contabilidad de las existencias (entradas y salidas), posibilita gestionar la recepción de tránsitos generados desde aduanas nacionales y comunitarias, con todos los mensajes aceptados en formato EdiFact (TNN, AVI, OBS, TNA, TAO, AVO). También permite realizar emisiones de tránsito mediante el mensaje EdiFact correspondiente (DUA[7] de expedición de Tránsito), así como la impresión del documento de acompañamiento y su ejemplar de

[7] Documento aduanero creado por la Comisión Europea para armonizar los trámites e información aduanera en la Unión Europea (operaciones de importación, exportación y tránsito).

reenvío (NCTS), o del DUA (Documento Único Administrativo) si fuese necesario. Igualmente contempla la gestión de solicitudes de Cambios de Ubicación (mensaje CUB) y del Certificado de Recepción para exportaciones. Se incluye la opción para los ADT que además tengan autorizado un L.A.M.E. para mercancía destinada a la exportación sujeta a control de la Aduana.

SIDUNEA

El Sistema Aduanero Automatizado (SIDUNEA) es la herramienta informática para el control y administración de la gestión aduanera, desarrollada por la Conferencia de las Naciones Unidas sobre el Comercio y el Desarrollo (UNCTAD), y que actualmente es usada con éxito en más de 80 países. SIDUNEA permite realizar un seguimiento automatizado de las operaciones aduaneras y controlar efectivamente la recaudación de los impuestos aduaneros, porque este sistema 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. De este modo se convierte en un único lenguaje, seguro y comprensible para todos los actores del proceso. SIDUNEA se puede configurar de acuerdo a las características nacionales de cada régimen aduanero, al arancel nacional y a la legislación de cada país, además de implementar los estándares internacionales para procesar los datos de comercio exterior ya acordados por la Organización Mundial de Aduanas (OMA) y por la Organización Internacional para la Estandarización (ISO).

Entre las ventajas que se pueden obtener con la aplicación del Sistema Aduanero Automatizado se encuentran:

- Optimizar los tiempos y recursos del proceso aduanero
- Aplicar la ley con toda justicia
- Cobrar correctamente los impuestos y tasas
- Detectar los errores en los valores de la declaración

- Monitorear el pago de los impuestos
- Minimizar el contrabando
- Crear incentivos para el declarante
- Poner en práctica un esquema de garantía con la modalidad de pago anticipado, para facilitar el comercio y asegurar el cobro de los derechos aduaneros

¿Cómo está constituido el sistema SIDUNEA?

Al ser cliente-servidor, SIDUNEA es un programa que en términos prácticos permitirá trabajar con conexión a la red o sin ella. Como servidor, tiene una configuración de red Ethernet para que un gran número de usuarios ingresen y procesen información sin perder rendimiento. Así mismo, genera datos estadísticos sobre comercio exterior y permite el intercambio electrónico de datos entre comerciantes y aduana. En el marco de este programa, la UNCTAD es responsable de su distribución a los países miembros de las Naciones Unidas que lo soliciten, así como de todas las modificaciones, actualizaciones y nuevas versiones del núcleo del sistema. Es un software que continuamente se mejora y se actualiza según las experiencias en el despacho de mercadería y estadísticas de la aduana internacional.

Características tecnológicas

Gracias a la tecnología cien por ciento JAVA, permite el uso de tarjetas inteligentes con procesadores y tecnología JAVA para controlar los accesos al sistema y los pagos electrónicos. Así mismo, su aplicabilidad vía Internet, permite acceder al sistema a través de dispositivos inalámbricos. También es resistente a las caídas de las telecomunicaciones, es compatible con la mayoría de los Sistemas Administradores de Bases de Datos Relacionales (JDBC) e independiente de las plataformas (equipos) y de los sistemas de base de datos. Además cuenta con:

- Modernos conceptos de seguridad (PKI) o Conceptos DOM (*del inglés, Document Object Model*), XML.
- Directorios de mensajes XML estándar que hacen posible la cooperación internacional entre sistemas y la creación de la red Customs Global o Protocolo REWI extendido, TCP/IP

- Interfaz de usuario amigable
- Extensión e implementación dinámica
- Adaptabilidad (según el número de operaciones)
- Aspectos de seguridad ya integrados
- Funciones especiales como Multilenguaje, Gestión, Propiedad de documentos, Auditoría e Historización.

Beneficios de este sistema

SIDUNEA permitirá trabajar con mayor comodidad para elaborar recaudos cuando más convenga, dentro de los tiempos permitidos; así como la posibilidad de revisar los datos ingresados tantas veces como sea necesario, sin pérdida de tiempo ni recursos. Permite transparencia en los procesos, rapidez en el control de las operaciones en tiempo real, reducción de trámites y tiempo de almacenamiento; sustitución del papel por documentos electrónicos, así como el pago electrónico de los impuestos.

La modularidad de SIDUNEA, como la de la mayoría de los otros programas de computación para la administración aduanera que existen en el mercado, significa que se le pueden agregar programas (módulos) nuevos o avanzados en el momento que le convenga al país respectivo. Tales módulos suplementarios pueden cubrir funciones aduaneras, tales como la administración del riesgo, las operaciones de tránsito o los nuevos estándares de seguridad, según sean las prioridades nacionales.(7)

MODSHD (Módulo de Depósito Temporal): permite a las almacenadoras, la generación e impresión del nuevo formato del Pase de Salida en el Sistema Aduanero Automatizado SIDUNEA, en el que se integra el código de barra que requiere el circuito de inspección de Rayos X, así como la descongestión de impresión de los mismos, lo que repercutirá positivamente en la rapidez de las operaciones; así como realizar los procedimientos de Localización o Relocalización de las mercancías, sobrantes y faltantes, y la emisión del Acta de Recepción. (8)

1.2.2 Soluciones Cubanas

SUA

En el año 2004 especialistas del CADI iniciaron el desarrollo de una solución de software cubana para la gestión de los procesos aduanales denominada Sistema Único de Aduanas (SUA), a lo cual se unió posteriormente el esfuerzo de estudiantes y profesores de la Universidad de las Ciencias Informáticas.

Actualmente dentro del Sistema Único de Aduanas existe un subsistema, denominado Almacén, que es utilizado para gestionar solo una parte de los procesos que se realizan en los depósitos temporales, transferencias y tránsitos de mercancías de importación y exportación. Es una aplicación de entorno Web, que utiliza gestor de Base de Datos Oracle, lenguaje de programación PHP (versión 5.0.4) y paradigma de programación estructurada.

De las soluciones informáticas extranjeras para la gestión y control de las operaciones que se realizan en los diferentes depósitos bajo control aduanero, analizadas anteriormente, no se encontró ninguna factible para utilizar en la AGR, debido a que implementan legislaciones y normas específicas de sus países de origen (a excepción de SIDUNEA que es adaptable); además de ser herramientas privativas, lo que supone un alto costo en inversiones por concepto de licencias y soporte.

En el caso de la solución cubana las razones de no adecuación, están sustentadas además de la tecnología sobre la cual está desarrollada y paradigma de programación utilizados, en su incapacidad para gestionar una gran cantidad de procedimientos y operaciones comunes que ocurren en los depósitos temporales de frontera.

1.3 Ingeniería de Requerimientos

¿Qué es un requisito?

El Glosario Estándar de Terminología de Ingeniería de Software de la IEEE define un requerimiento como:

- “Condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo”. (9)

- "Una condición o capacidad que debe estar presente en un sistema o componente de sistema para satisfacer un contrato, estándar, especificación u otro documento formal". (9)

El objetivo de la ingeniería de requerimientos es hacer que los mismos alcancen un estado óptimo antes de seguir adelante con el sistema. La etapa de requerimientos es de gran importancia para el desarrollo del software porque suministra la base para todo el trabajo que sigue, por lo que constituye la parte que afecta más al sistema final si se realiza de manera incorrecta, debido a que es muy difícil de rectificar, implicaría grandes costos mientras más avanzado esté el proyecto.

1.3.1 Actividades fundamentales de la ingeniería de requisitos

El desarrollo de software presenta inconvenientes para llegar al objetivo final, un producto terminado que posea las características deseadas por el cliente y que realmente resuelva sus necesidades, cuestiones como que los proyectos en muchos casos suelen costar más de lo previsto o se entregan tarde. Estos contratiempos se ubican principalmente en los momentos donde es requerido un pensamiento más abstracto: requerimientos y arquitectura de software.

Para realizar este proceso, no existe una única técnica estandarizada y estructurada que ofrezca un marco de desarrollo que garantice la calidad del resultado. Se debe tener en cuenta que la selección de las técnicas y el éxito de los resultados que se obtengan, depende en gran medida tanto del equipo de análisis y desarrollo, como de los propios clientes o usuarios que en ella participen.

El proceso se inicia con la realización de la captura de requisitos, el grupo de técnicos (desarrolladores, analistas, diseñadores) toma la información suministrada por los usuarios y clientes, esta información puede originarse de diversas fuentes: documentos, aplicaciones existentes, entrevistas, o mediante la observación, etc. A partir de esta información, el equipo de desarrollo elabora el documento de especificación de requisitos. Finalmente con la validación se realiza la valoración de los mismos, comprobando si existen inconsistencias, errores o si faltan requisitos por definir. El proceso de definición-validación es iterativo y en algunos proyectos complejos puede resultar necesario ejecutarlo varias veces.

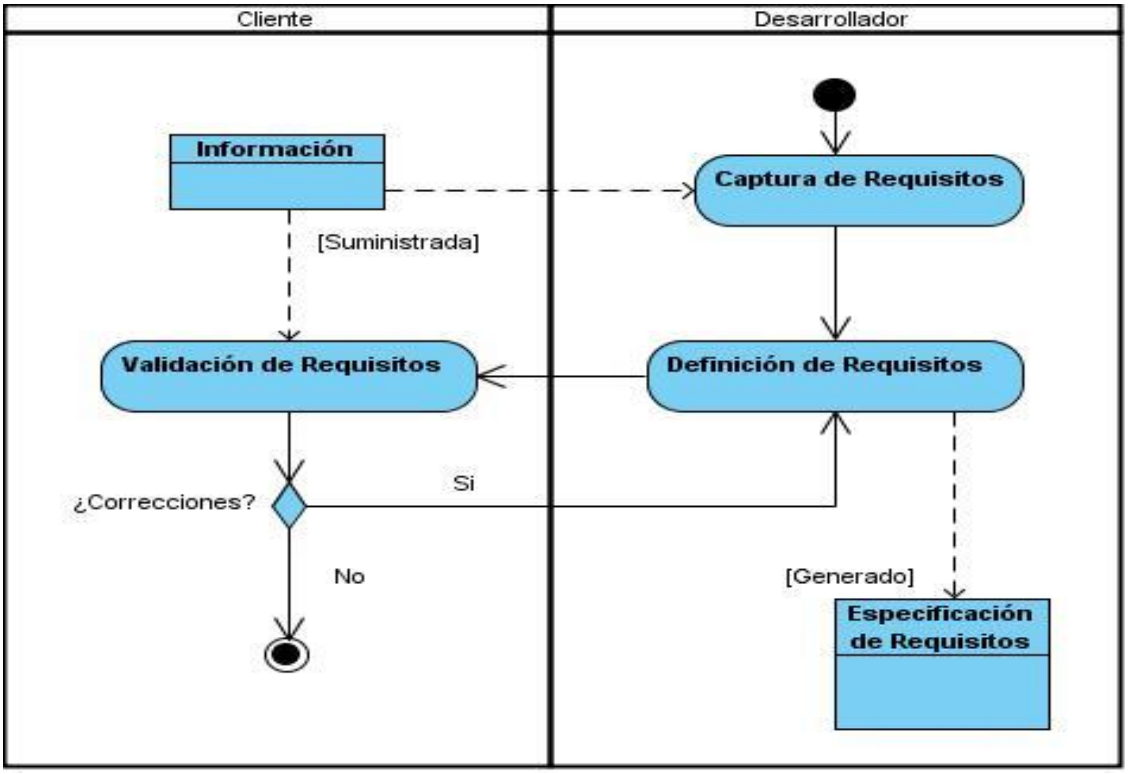


Fig. 1. 1 Proceso de Ingeniería de Requisitos

1.3.2 Técnicas para la captura y validación de los requisitos

La captura de requisitos es el proceso mediante el cual el equipo de analistas se enfoca en determinar las necesidades del cliente que debe satisfacer el sistema. Esta información puede extraerse de varias fuentes y por ser considerablemente difícil la comunicación entre clientes y desarrolladores, principalmente si para estos últimos el entorno de trabajo es desconocido, puede resultar complejo. Para contrarrestar esta complejidad han surgido algunas técnicas que pueden hacer que estas actividades se realicen con eficiencia y en menos tiempo. (10)

Una vez definidos los requerimientos del sistema es preciso que el equipo de analistas confirme con los clientes si estos representan las necesidades reales del sistema deseado, con el propósito además, de detectar los errores de forma temprana, para no conducir a resultados inesperados evitando provocar gastos excesivos y pérdida de tiempo.

A continuación se presentan algunas de estas técnicas para la captura y validación de requisitos del sistema.

Captura de Requisitos:

- **Entrevistas:** es muy aceptada y permite acercarse al problema de una manera natural, los desarrolladores pueden interpretar ampliamente las necesidades del usuario. Es una técnica complicada pues depende de la habilidad del entrevistador, ya que es determinante seleccionar bien a los entrevistados para obtener la mayor cantidad de información importante en un período de tiempo limitado.

- **Tormentas de ideas:** es una técnica grupal y consiste en la simple acumulación de ideas sin detenerse en el análisis del valor de las mismas, es muy fácil de realizar a diferencia de otras técnicas. Es más usada en encuentros iniciales pues solo evidencia una perspectiva general de las necesidades del sistema, no los detalles concretos.

- **Comparación de la terminología:** es utilizada para complementar otras técnicas y el objetivo es llegar a un consenso de qué términos serán usados durante todo el proyecto. Para ello es necesario identificar el uso de términos diferentes para los mismos conceptos (correspondencia), misma terminología para diferentes conceptos (conflictos) o cuando no hay concordancia exacta ni en el vocabulario ni en los conceptos (contraste).

- **Observación:** es utilizada para capturar visualmente y registrar por escrito el desenvolvimiento de las actividades habituales, es tenida en cuenta cuando el experto no sabe explicar una tarea, los analistas no los entienden o es omitido algo que para él es obvio.

- **Introspección:** reside en mirar la situación desde la perspectiva del cliente y una vez en su lugar, suponer (entendiendo el negocio plenamente, por supuesto) necesidades que pudiera cubrir el subsistema, elementos que debe satisfacer o funcionalidades que le se pudiesen agregar.

Validación de requisitos:

- **Matrices de trazabilidad:** esta técnica consiste en marcar los objetivos del sistema y chequearlos contra los requisitos del mismo. Es necesario ir viendo qué objetivos cubre cada requisito, de esta forma se podrán detectar inconsistencias u objetivos no cubiertos.
- **Prototipos:** esta técnica tiene el problema de que el usuario debe entender que lo que está viendo es un prototipo y no el sistema final, puesto que no es totalmente funcional y debe hacerse una idea de la estructura de la interfaz del sistema, esto es a partir de la definición de los requisitos.
- **Revisiones:** esta técnica consiste en la lectura y corrección de la documentación o modelado de la definición de requisitos para refinar los artefactos generados.

1.4 Diseño de Software

En el desarrollo de software es preciso que se tengan presente ciertos estándares para la construcción de la solución; estilos, patrones y tendencias utilizados que son muy útiles en la obtención de un diseño de software que sea fiel y robusta representación del sistema final.

1.4.1 Arquitectura de Software

En el desarrollo de aplicaciones informáticas la arquitectura de software es un término muy frecuente y de gran referencia ya que sirve para guiar la implementación y el desarrollo del sistema desde etapas tempranas del diseño. Existen muchas definiciones acerca del tema, y aunque todas son consideraciones especializadas de distintos autores, coinciden en que a grandes rasgos se refiere a la estructura del sistema, constituida por componentes de software y relaciones entre estos.

Algunas definiciones formales:

Según Bass, “La arquitectura del software de un programa o sistema de computación es la estructura o estructuras del sistema que comprende los elementos del software, las propiedades externamente visibles de esos elementos, y las relaciones entre ellos”. (11)

“La Arquitectura de Software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y

las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la mayor parte de las abstracciones". (12)

En una definición tal vez demasiado amplia, David Garlan establece que la Arquitectura de Software constituye un puente entre el requerimiento y el código, ocupando el lugar que en los gráficos antiguos se reservaba para el diseño. (13)

De todas las definiciones una de las más aplicadas es la ofrecida en el documento IEEE STD 1471-2000 (Software Engineering Standards Committee of the IEEE Computer Society, 2000): "La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución."

1.4.2 Estilos Arquitectónicos

En el texto fundacional de la Arquitectura de Software, Dewayne E. Perry y Alexander L. Wolf establecen el razonamiento sobre estilos de arquitectura como uno de los aspectos fundamentales de la disciplina. Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software, mientras que las formas complejas se articulan mediante composición de los estilos fundamentales.

Brevemente descritos, los estilos conjugan elementos, conectores, configuraciones y restricciones. Al estipular los conectores como elemento de juicio de primera clase, el concepto de estilo, incidentalmente, se sitúa en un orden de discurso y de método que el modelado orientado a objetos en general y UML en particular no cubren satisfactoriamente. La descripción de un estilo se puede formular en lenguaje natural o en diagramas, pero lo mejor es hacerlo en un lenguaje de descripción arquitectónica o en lenguajes formales de especificación.

A diferencia de los patrones de diseño, que son centenares, los estilos se ordenan en seis o siete clases fundamentales y unos veinte ejemplares, como máximo. Es digno de señalarse el empeño por incluir todas las formas existentes de aplicaciones en un conjunto de dimensiones tan modestas. Las arquitecturas complejas o compuestas resultan del agregado o la composición de estilos más básicos. (14)

Estilos de Flujo de Datos

- Tubería y filtros

Estilos Centrados en Datos

- Arquitecturas de Pizarra o Repositorio

Estilos de Llamada y Retorno

- Modelo-Vista-Controlador (MVC)
- Arquitecturas en Capas
- Arquitecturas Orientadas a Objetos
- Arquitecturas Basadas en Componentes

Estilos Derivados

- C2
- GenVoca
- REST

Estilos de Código Móvil

- Arquitectura de Máquinas Virtuales

Estilos heterogéneos

- Sistemas de control de procesos
- Arquitecturas Basadas en Atributos

Estilos Peer-to-Peer

- Arquitecturas Basadas en Eventos

- Arquitecturas Orientadas a Servicios (SOA)
- Arquitecturas Basadas en Recursos

1.4.3 Patrones

Durante el desarrollo de software suelen surgir problemas comunes, los cuales son resueltos con el uso de alternativas de soluciones que pueden ser muy efectivas, pues en la práctica y en el transcurrir de los años se ha demostrado que son eficientes frente a dificultades presentadas, estas variantes son como esquemas de situaciones que se presentan con frecuencia, y traen asociada la manera en que se solventan, estos elementos son llamados patrones.

El término comenzó a tener fuerza cuando Christopher Alexander (15) en 1977 definió lo que sería un patrón de manera muy genérica: “Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, para describir después el núcleo de la solución a ese problema, de tal manera que esa solución pueda ser usada más de un millón de veces sin hacerlo ni siquiera dos veces de la misma forma.”

Una definición de patrones según Craig Larman (16):

“Un patrón es un par problema/solución con nombre que se puede aplicar en nuevos contextos, con consejos sobre cómo aplicarlo en nuevas situaciones, o sea, un patrón es una descripción de un problema bien conocido que suele incluir: descripción, escenario de uso, solución concreta, consecuencias de utilizar el patrón, ejemplos de implementación y lista de patrones relacionados.”

Son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Cada patrón es adecuado para ser adaptado a un cierto tipo de problema.

Características de los patrones (17):

- Solucionan un problema: los patrones capturan soluciones, no sólo principios o estrategias abstractas.
- Son un concepto probado: capturan soluciones demostradas, no teorías o especulaciones.

- La solución no es obvia: los mejores patrones generan una solución a un problema de forma indirecta.
- Describen participantes y relaciones entre ellos: describen módulos, estructuras del sistema y mecanismos complejos.
- El patrón tiene un componente humano significativo: todo software proporciona a los seres humanos confort y calidad de vida (estética y utilidad).

1.4.3.1 Patrones de Arquitectura

Un patrón de arquitectura de software describe un problema particular y recurrente del diseño, que surge en un contexto específico, y presenta un esquema genérico y probado de su solución.

Patrón Modelo-Vista-Controlador (MVC)

Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres capas o componentes distintos. El patrón MVC se utiliza frecuentemente en aplicaciones web, donde la vista es la página HTML que se visualiza en el navegador y el código que proporciona de datos dinámicos a la página, el modelo es el Sistema de Gestión de Base de Datos y la lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista, enviarlos al modelo y manejar también las respuestas del modelo y transmitirlas hacia la vista. (18)

- El modelo representa la información con la que trabaja la aplicación y se encarga de acceder a los datos.
- La vista transforma la información obtenida por el modelo en las páginas web a las que acceden los usuarios.
- El controlador es el encargado de coordinar todos los demás elementos y transformar las peticiones del usuario en operaciones sobre el modelo y la vista.

1.4.3.2 Patrones de Diseño

Los patrones de diseño son descripciones de clases cuyas instancias colaboran entre sí y brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares.

Grady Booch plantea: "Una arquitectura orientada a objetos bien estructurada está llena de patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles". (19)

Patrones GRASP

GRASP es un acrónimo que significa General Responsibility Assignment Software Patterns (*en español, patrones generales de software para asignar responsabilidades*). El nombre se eligió para indicar la importancia de captar *(grasping)* estos principios, si se quiere diseñar eficazmente el software orientado a objetos.

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. (16)

Patrones GoF

GOF es el acrónimo de Gang of Four (*en español, banda de los cuatro*), se clasifican de acuerdo a su propósito en:

Patrones de creación: abstraen la creación de instancias.

Patrones estructurales: se ocupan de la relación entre clases, la combinación de objetos y la formación de estructuras de mayor complejidad.

Patrones de comportamiento: caracterizan la manera en la cual clases u objetos interactúan, cooperan y distribuyen responsabilidades.

1.5 Metodología, lenguajes y herramientas de modelado utilizadas para el desarrollo.

En la Universidad de las Ciencias Informáticas el trabajo de los proyectos productivos se realiza mediante centros de desarrollo que están distribuidos por las distintas facultades que la constituyen. Cada centro tiene sus políticas, normas y estándares definidos para el desempeño de los distintos proyectos que lo componen. En el Centro de Informatización de la Gestión de Entidades (CEIGE), específicamente en el Departamento de Soluciones para la Aduana se encuentra desarrollándose el producto Gestión Integral de Aduanas (GINA) que constituye la nueva solución integrada para las operaciones aduanales.

A continuación se muestran algunos de los rasgos característicos de la metodología, lenguajes, herramientas y tecnologías utilizadas para el desarrollo la solución.

1.5.1 Metodología de desarrollo de software

RUP

RUP (*del inglés, Rational Unified Process*) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Esta metodología representa una guía que establece **quién** hace **qué**, **cuándo** y **cómo** lo hace.

El RUP está basado en 5 principios claves que son:

- Adaptar el proceso.
- Balancear prioridades.
- Demostrar valor iterativamente.
- Elevar el nivel de abstracción.
- Enfocarse en la calidad.

Esta se divide en cuatro fases y nueve flujos de trabajo (ver Fig. 1.1) para la mejor organización del software, seis de ingeniería: Modelado del negocio, Requerimientos, Análisis y Diseño, Implementación, Prueba, Despliegue; y tres de soporte los cuales son: Gestión de configuración y cambios, Gestión de

proyectos y por último Entorno. Cada fase se desarrolla mediante iteraciones que generan artefactos logrando ser una de las metodologías más importantes para establecer un acuerdo entre desarrolladores y clientes, está diseñada para proyectos de gran envergadura y genera gran cantidad de documentación permitiendo un mejor entendimiento del proyecto.

Fases del ciclo de vida de RUP:

Inicio: en esta fase se hace un plan de fases, se identifican los principales casos de uso, los riesgos y se define el alcance del proyecto.

Elaboración: en esta segunda fase de desarrollo, se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos, las iteraciones se orientan al desarrollo de la línea base de la arquitectura, abarcan más los flujos de trabajo requerimientos, modelo del negocio (refinamiento), análisis y diseño y una parte de implementación orientado a la arquitectura.

Construcción: etapa donde se concentra la elaboración de un producto totalmente operativo, eficiente y el manual de usuario.

Transición: en esta fase se hace entrega del producto al cliente, se instala y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados.

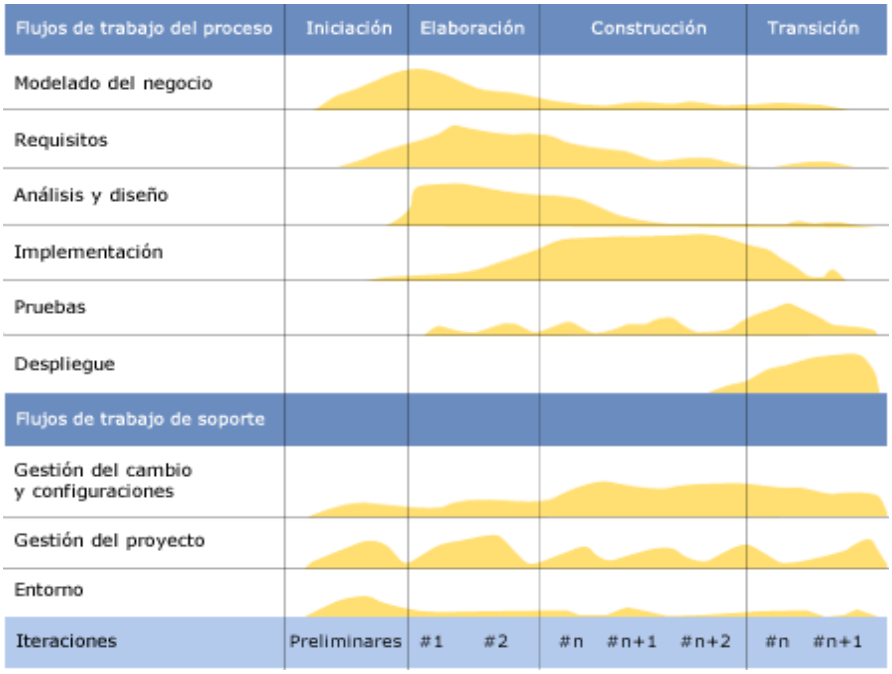


Fig. 1. 2 Fases y flujos de trabajo de RUP

Sus características principales son:

- **Dirigido por casos de uso:** los casos de uso guían el proceso de desarrollo pues los modelos que se obtienen representan la realización de los mismos.
- **Centrado en la arquitectura:** la arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo.
- **Iterativo e incremental:** una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. En el caso de una iteración de la fase de elaboración, se centra la atención en el análisis y diseño, a la vez que se refinan los requerimientos y se obtiene un producto con un determinado nivel, que irá creciendo incrementalmente en cada iteración. (20)

1.5.2 Notación para el Modelado de Procesos de Negocio

BPMN (versión 1.x)

El modelado del negocio se maneja mediante la notación estándar BPMN (*del inglés, Business Process Modeling Notation*). El objetivo principal de los esfuerzos de BPMN era dar una notación rápidamente comprensible por todo el personal de negocios, desde el analista de negocio que hace el borrador inicial de los procesos, pasando por los desarrolladores técnicos responsables de implementar la tecnología que llevarán a cabo dichos procesos, llegando finalmente a las personas de negocio que gestionarán y monitorizarán esos procesos.

Utilizando esta notación, interpretar un Diagrama de Procesos de Negocio (BPD) suele resultar fácil gracias a que se han establecido categorías específicas para organizar los elementos gráficos de la notación, haciéndolos distinguibles los unos de los otros.

Las cuatro categorías básicas de elementos son:

- Objetos de flujo: son los principales objetos que expresan la semántica del modelo de procesos.
- Objetos conectores: estos elementos proveen a los objetos de flujo la posibilidad de relacionarse en un diagrama, y expresan cómo ellos interactúan entre sí para crear la estructura básica del esqueleto de un proceso de negocio.
- Artefactos: son utilizados para proporcionar información adicional acerca del proceso.
- Carriles de Piscina (Swimlanes): son utilizados para agrupar los elementos primarios de modelado, como un mecanismo para organizar actividades en categorías visuales separadas con el objetivo de representar capacidades funcionales diferentes o responsabilidades.

1.5.3 Leguaje de Modelado

UML

El Lenguaje Unificado de Modelado (UML) está compuesto por una notación muy específica y por las reglas semánticas relacionadas para la construcción de sistemas de software. El UML en sí mismo no

prescribe ni aconseja cómo usar esta notación en el proceso de desarrollo o como parte de una metodología de diseño orientada a objetos.

Provee beneficios significativos para los ingenieros de software y las organizaciones al ayudarles a construir modelos rigurosos, trazables y fáciles de mantener, que soporten el ciclo de vida de desarrollo de software completo.

1.5.4 Herramienta CASE para el modelado

Visual Paradigm (versión 3.4)

Esta herramienta ofrece un entorno para crear diagramas UML utilizando un lenguaje estándar para todo el equipo de desarrollo y facilitando la comunicación entre estos, además de que permite la ingeniería directa e inversa y es multiplataforma, esta herramienta soporta aplicaciones web y permite generar imágenes y reportes de muy buena calidad.

Posee una intuitiva interfaz de usuario y de fácil uso, que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, prueba y despliegue. Está compuesta por varios productos como en sus diferentes ediciones: Enterprise, Professional, Standard, Modeler, Personal y en su versión libre Community.

1.5.5 Lenguaje de programación

PHP

PHP es un acrónimo recursivo que significa "**P**HP **H**ypertext **P**re-processor" (*inicialmente PHP Tools, o Personal Home Page Tools*), es un lenguaje de programación interpretado, usado normalmente para la creación de páginas web dinámicas, además permite la conexión a diferentes tipos de servidores de base de datos, es multiplataforma, libre, por lo que se presenta como una alternativa de fácil acceso para todos y permite las técnicas de Programación Orientada a Objetos. Actualmente también se puede utilizar para la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

La versión utilizada de PHP es la 5.2.5 que incluye varias ventajas:

- Mejoras de rendimiento.
- Mejor soporte para MySQL con extensión completamente reescrita.
- Mejor soporte a XML (XPath, DOM, entre otros).
- Soporte nativo para SQLite.
- Soporte integrado para SOAP.
- Iteradores de datos.
- Manejo de excepciones.

1.5.6 Marco de trabajo (Framework)

Symfony (version 1.2.x)

Symfony es un framework desarrollado en lenguaje PHP, elegante, estable, productivo y muy bien documentado, muy difundido en la actualidad para la construcción de aplicaciones web, utilizando las mejores prácticas y los patrones de diseño más importantes. Symfony incorpora muchas de las ideas del RAD ("desarrollo rápido de aplicaciones") para conseguir que la programación de las aplicaciones sea lo más productiva, correcta y divertida posible. Es multiplataforma y se integra con varios sistemas gestores de bases de datos, lo que le brinda una versatilidad muy considerable, al poder migrar la base de datos sin tener que realizar ningún otro cambio en la aplicación, más que editar un archivo de configuración. El mismo hace uso del patrón de diseño MVC (modelo-vista-controlador) para un mejor control de las partes de la aplicación y las mantiene por separado.

Entre sus principales características se encuentran:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- Independiente del sistema gestor de bases de datos.

- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de *"convenir en vez de configurar"*, en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales, y adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

1.5.7 Interfaz de usuario

ExtJS (versión 3.1)

Es un framework diseñado para la creación de páginas web dinámicas del lado del cliente basado en lenguaje JavaScript, permite una gran reutilización de componentes, personalización de los mismos, haciendo uso del manejo de tecnologías como AJAX (*del inglés, Asynchronous JavaScript And XML*) para el intercambio asincrónico de los datos entre el cliente y el servidor, además de lo ventajoso que puede representar la similitud de su aspecto con el de una aplicación de escritorio.

1.5.8 Gestor de Base de Datos

Oracle (11g)

Es un potente sistema gestor de base de datos (SGBD), multiplataforma, que se caracteriza por haber sido concebido con el fin de manejar grandes cantidades de información, además de admitir conexiones concurrentes de multitud de usuarios (entornos multi-usuario) hacia los mismos datos.

Las principales funcionalidades aportadas por todo el SGBD Oracle son:

- Soporte y tratamiento de una gran cantidad de datos (Gbytes).
- Soporte de una gran cantidad de usuarios accediendo concurrentemente a los datos.
- Seguridad de acceso a los datos, restringiendo dicho acceso a las necesidades de cada usuario.
- Conectividad entre las aplicaciones de los clientes en sus puestos de trabajo y el servidor de base de datos Oracle (estructura cliente/servidor).
- Conectividad entre bases de datos remotas (estructura de bases de datos distribuidas).
- Portabilidad.
- Compatibilidad.

Conclusiones

En el presente capítulo se demuestra por qué es importante para la Aduana contar con un software que facilite el control de las mercancías en los depósitos temporales, se realiza un estudio de los sistemas extranjeros que son utilizados para tales fines, los cuáles no son viables porque no cumplen con las legislaciones y normativas aduaneras cubanas vigentes (excepto SIDUNEA, que es adaptable) para el funcionamiento y control de los depósitos temporales de frontera, además de ser herramientas privativas. Se definieron además, metodología, lenguajes y tecnologías a utilizar en el desarrollo de la solución con el objetivo de que esta cuente con todas las ventajas posibles que garanticen una gestión más completa de las operaciones que se realizan en los depósitos temporales de frontera.

Capitulo 2: Características del Sistema

Introducción

En el capítulo se presenta una propuesta del sistema a desarrollar, se describen las características principales del negocio, se muestra el flujo principal de procesos y el modelado de los mismos, se plasma un análisis crítico sobre cómo se ejecutan los procesos en la actualidad, se exponen los requerimientos funcionales definidos para el sistema, y se expresan además las técnicas de captura y validación de requerimientos utilizadas.

2.1 Modelado de los procesos del Negocio

A continuación se describen los principales procesos que se llevan a cabo en los Depósitos Temporales de Frontera Portuarios y Aeroportuarios de la Aduana General de la República de Cuba, con la finalidad de lograr un mayor entendimiento del negocio que dé al traste con el desarrollo de un software que implemente estos procesos con la mayor calidad posible. El resto de los diagramas y sus descripciones se encuentran en el expediente del proyecto.

El siguiente diagrama presenta los procesos globales a partir de los cuales se derivan cada uno de los subprocesos que se efectúan en los Depósitos Temporales de Frontera, asociados a la importación y exportación de mercancías de carácter comercial y que constituyan de interés para la AGR. Dichos subprocesos son: Importar mercancías en el DTFP (Depósito Temporal de Frontera Portuario), Exportar mercancías en el DTFP, Importar mercancías en el DTFA (Depósito Temporal de Frontera Aeroportuario), Exportar mercancías en el DTFA.

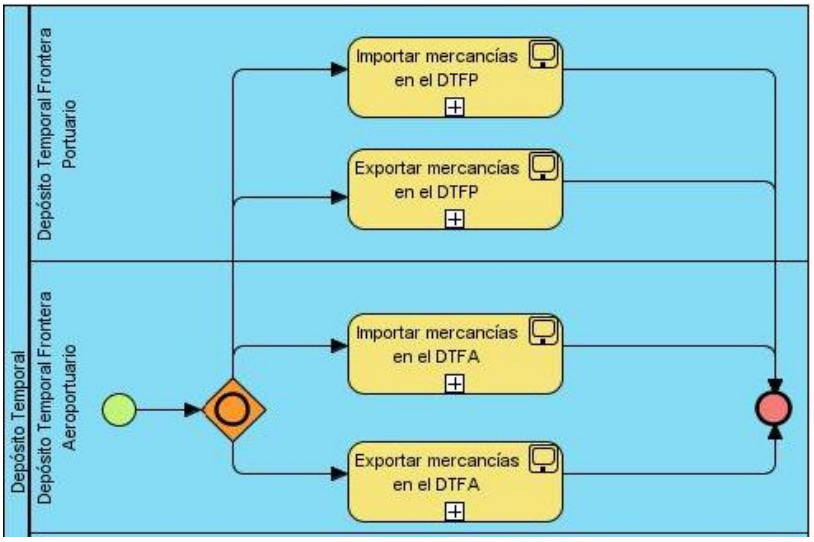


Fig. 2. 1 Proceso Gestionar operaciones en los Depósitos Temporales de Frontera

2.1.1 Descripción del subproceso Importar mercancías en el DTFP

El subproceso se inicia (ver Fig. 2.2) a partir de la descarga de mercancías de un buque, que posteriormente son gestionadas durante su estancia en el depósito. En el caso de las mercancías que fueron importadas con el propósito de ser exportadas se procede a su cambio de régimen de importación a exportación, aquellas que serán trasladadas bajo los regímenes de cabotaje[8] o trasbordo[9] se cargan en el buque correspondiente y las que se extraen del depósito cuyo destino sea otro depósito o el propio almacén del importador deben ser recibidas una vez que arriben a su destino. Si se extraen contenedores a los cuales se les definió revisión radiológica, el transportista que los traslada está en la obligación de llevarlas al control respectivo, si el reconocimiento arroja resultados positivos y el inspector en función determina que deben regresar al depósito estas son recibidas en el mismo.

---

[8] Es el régimen aduanero aplicable a las mercancías que se cargan en un buque, en un puerto del territorio nacional y se transportan a otro puerto del territorio nacional donde se descargan. Las operaciones de cabotaje se realizan por buques cubanos o sometidos a jurisdicción cubana, habilitados para este tráfico, de forma temporal o permanente.

[9] Régimen aduanero mediante el cual se transfieren bajo control aduanero, mercancías de una unidad de transporte a otra, o a la misma en distinto viaje, con el fin de que continúen hasta su lugar de destino.

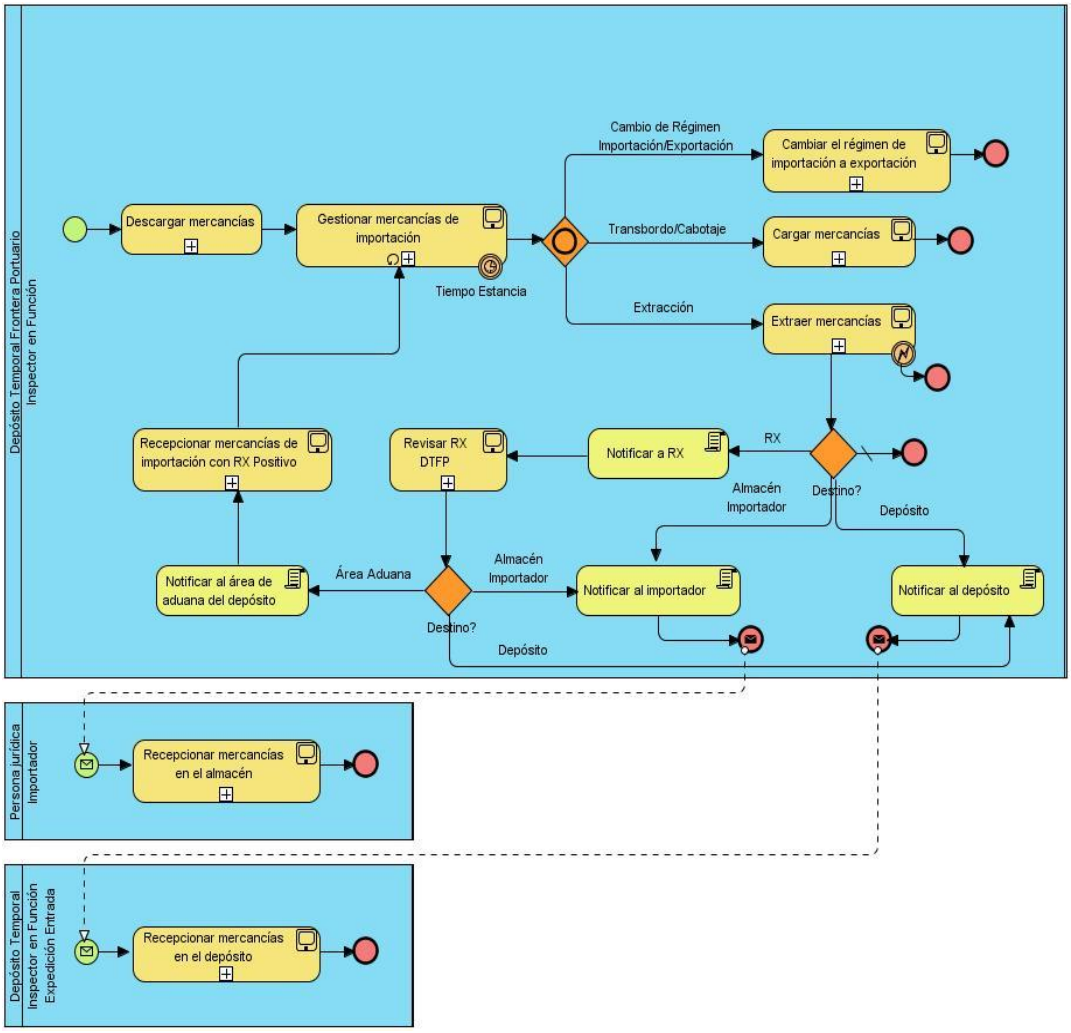


Fig. 2. 2 Subproceso Importar mercancías en el DTFP

2.1.2 Descripción del subproceso Exportar Mercancías en el DTFP

El subproceso comienza (ver Fig. 2.3) con la extracción de mercancías de un depósito o del almacén del exportador, las cuales son revisadas radiológicamente en los casos que aplique, y recibidas en el depósito temporal de frontera correspondiente, si el reconocimiento radiológico arroja resultados positivos y el inspector en función determina que deben regresar a su origen, estas son recibidas en el mismo. También se pueden recibir mercancías bajo el régimen de cabotaje. Una vez que las mercancías se encuentran en el depósito son gestionadas durante su estancia, y finalmente son cargadas en el buque para la respectiva exportación o son extraídas por necesidades del exportador.

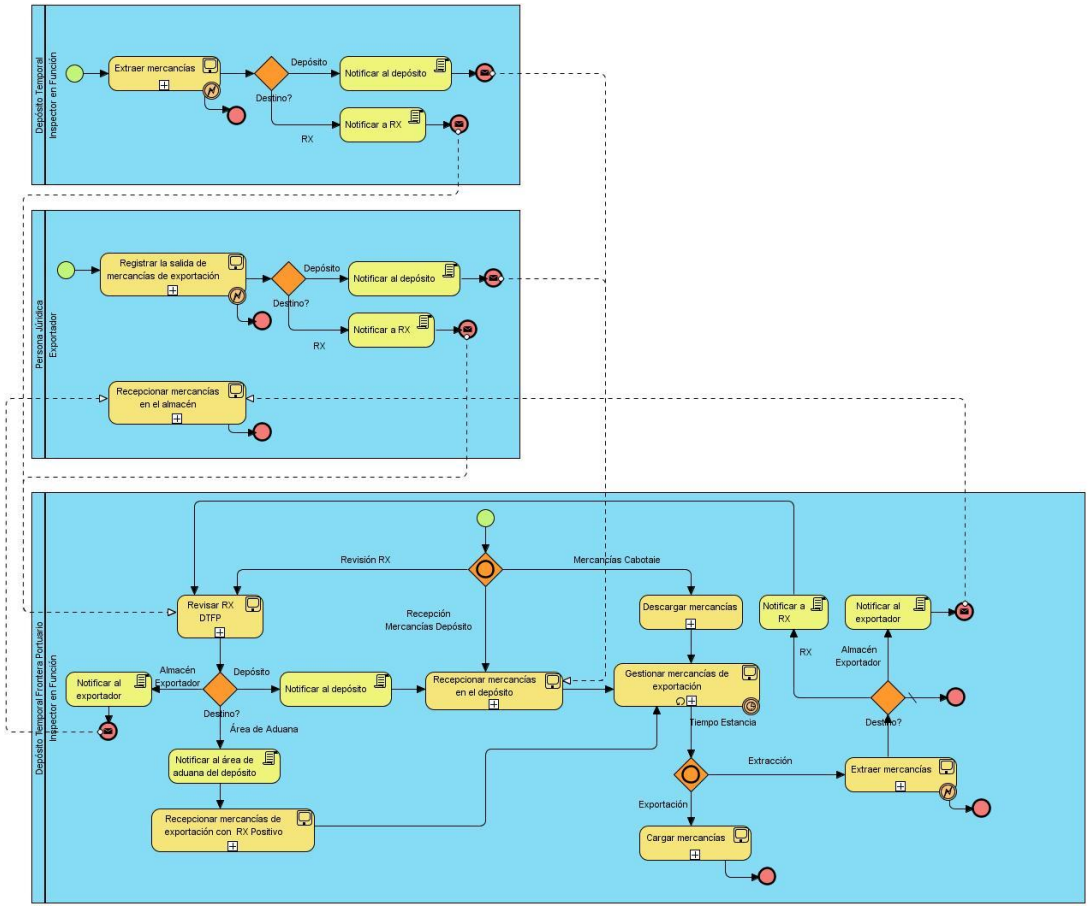


Fig. 2. 3 Subproceso Exportar Mercancías en el DTFP

2.2 Análisis Crítico de la Ejecución de los Procesos

En las operaciones que se ejecutan en los depósitos temporales se mueven grandes cantidades de mercancías, de las cuales para conocer sus cifras reales en el inventario, realizar resúmenes estadísticos y reportes, los inspectores de la Aduana se auxilian de los sistemas informatizados que poseen los operadores de los depósitos.

Los usuarios finales del módulo Almacén del SUA han identificado como principal deficiencia del mismo su incapacidad de gestionar la totalidad de las operaciones, la mayoría de las cuales se ejecutan de manera manual como por ejemplo: los cotejos contra los manifiestos correspondientes durante las descargas y cargas de mercancías en los puertos o aeropuertos, el control de las averías y su reparación, las prórrogas otorgadas y las declaraciones de abandono legal.

Si se automatizaran todas estas operaciones se lograría una mayor eficiencia en el proceso global y un mejor acceso a la información para la toma oportuna de decisiones.

2.3 Especificación de los Requerimientos de Software

Detallar los requerimientos del sistema deja sentadas las bases para etapas posteriores del desarrollo del mismo, por lo cual deben tenerse muy en cuenta las técnicas que son más factibles utilizar para la captura y la validación de estos requisitos; seguidamente se exponen las técnicas empleadas además de un listado de los mismos.

2.3.1 Técnicas para la Captura de Requerimientos

Siguiendo las estrategias propias de desarrollo del Centro de Informatización de la Gestión de Entidades y específicamente del Departamento de Soluciones para la Aduana se combinan las siguientes técnicas para la captura de los requisitos: entrevistas, introspección, observación y tormenta de ideas.

Se **entrevistaron** inspectores de la aduana del puerto y del aeropuerto, funcionarios del Departamento de Técnicas Aduaneras de la AGR, a los Jefes del Depósito del Aeropuerto y del Puerto, en visitas realizadas a las instalaciones, ya que las entrevistas permiten un intercambio más abierto con los especialistas, mediante preguntas que esclarecen con precisión el funcionamiento de todo el trabajo en los Depósitos Temporales de Frontera. Se utilizó **la introspección** para desde el punto de vista del interesado apreciar, si por ejemplo, hacía falta una determinada funcionalidad o se podían simplificar ciertas operaciones, pensando principalmente en los inspectores de aduana que serían los usuarios más beneficiados con la herramienta. La **observación** se llevó a cabo para percibir como se desarrollan las operaciones en los distintos depósitos del Puerto y Aeropuerto, pudiendo captar directamente las particularidades de estos procesos en las visitas realizadas; y la **tormenta de ideas** donde en conjunto con la especialista de la Aduana que atiende el módulo de Depósitos Temporales, inspectores, los jefes del depósito del puerto y el personal de técnica aduanera, se acumularon ideas para tener una perspectiva general de las necesidades del sistema.

2.3.2 Requerimientos Funcionales

Los requisitos funcionales representan las capacidades o condiciones que debe satisfacer el sistema, estos deben estar bien determinados y ser convenientemente comprendidos tanto por los implicados para con el sistema como por los desarrolladores del mismo para que exista un entendimiento común.

A continuación se exponen los requerimientos definidos para el sistema modelado:

RF1 Gestionar lo real descargado: se registra, modifica y anula lo real descargado de un buque o aeronave en el depósito temporal de frontera correspondiente.

RF2 Gestionar la recepción de mercancías: se registra, modifica y anula la recepción de mercancías provenientes de otro depósito, del almacén del exportador o la revisión radiológica.

RF3 Gestionar la recepción de contenedores de cabotaje: se registra, modifica y anula la recepción de contenedores de cabotaje de exportación que provienen de la revisión radiológica correspondiente.

RF4 Gestionar lo real cargado: se registra, modifica y anula lo real cargado en un buque o aeronave en el depósito temporal de frontera correspondiente.

RF5 Gestionar la extracción de mercancías: se registra, modifica y anula la extracción de mercancías efectuada en el depósito.

RF6 Gestionar la extracción de contenedores de cabotaje: se registra, modifica y anula la extracción de contenedores de cabotaje de exportación para que sean revisados radiológicamente.

RF7 Gestionar las averías en la carga granel de importación: se registran, modifican y anulan las averías producidas en la carga granel de importación, así como los resultados de la reparación de las mismas.

RF8 Gestionar las averías en los bultos de importación: se registran, modifican y anulan las averías producidas en los bultos de importación, así como los resultados de la reparación de las mismas.

RF9 Gestionar las averías en la carga granel de exportación: se registran, modifican y anulan las averías producidas en la carga granel de exportación, así como los resultados de su reparación.

RF10 Gestionar las averías en los bultos de exportación: se registran, modifican y anulan las averías producidas en los bultos de exportación, así como los resultados de la reparación de las mismas.

RF11 Gestionar agrupe de contenedores: se registran, modifican y anulan los resultados del agrupe de mercancías en contenedores.

RF12 Gestionar el desagrupe de contenedores: se registran, modifican y anulan los resultados del desagrupe o vaciado de mercancías en contenedores.

RF13 Gestionar el desagrupe de bultos: se registran, modifican y anulan los resultados del desagrupe de mercancías en los bultos.

RF14 Gestionar el cambio de mercancías de un contenedor hacia otro: se registra, modifica y anula el cambio de las mercancías de un contenedor hacia otro, debido a roturas producidas en el primero.

RF15 Gestionar la prórroga en mercancías de importación: se registra, modifica y anula la prórroga debidamente autorizada en mercancías de importación.

RF16 Gestionar la prórroga en mercancías de exportación: se registra, modifica y anula la prórroga debidamente autorizada en mercancías de exportación.

RF17 Declarar el abandono legal: se registran las mercancías declaradas en abandono legal.

RF18 Registrar Declaración de Abandono Legal: se registra el documento Declaración de Abandono Legal correspondiente a un abandono legal efectuado.

RF19 Anular el abandono legal: se anula una declaración de abandono legal registrada previamente.

RF20 Revocar el abandono legal: se registra la revocación del abandono, una vez presentada la Carta de Revocación de Abandono aprobada.

RF21 Gestionar la revisión radiológica de mercancías en el depósito: se registra, modifica y anula la revisión radiológica de mercancías que se encuentran dentro del depósito.

RF22 Gestionar la revisión radiológica de mercancías posterior a una extracción: se registra, modifica y anula la revisión radiológica de mercancías posterior a su extracción del depósito o envío del almacén del exportador.

RF23 Gestionar confirmación de llegada de mercancías al destino final: se registra, modifica o anula la confirmación de llegada de las mercancías a su destino final.

RF24 Gestionar los resultados de la recepción de mercancías en el destino final: se registran, modifican o anulan los resultados de la recepción de mercancías en el destino final.

RF25 Gestionar el envío de mercancías desde el almacén del exportador: se registra, modifica o anula el envío de mercancías desde el almacén del exportador hacia el depósito correspondiente.

RF26 Procesar información electrónica recibida: el sistema debe ser capaz de procesar la información electrónica recibida y emitir una respuesta al subsistema Recepción Electrónica.

RF27 Mostrar reportes: el sistema debe mostrar reportes.

2.3.3 Técnicas para la Validación de los Requerimientos

Validar los requisitos consiste en examinar exhaustivamente las especificaciones de requerimientos para corregir inconsistencias, eliminar ambigüedades, garantizar que no haya omisiones y determinar que estén cubiertas todas las necesidades de los usuarios/clientes y otros implicados, a fin de que el resultado del trabajo se ajuste a los estándares definidos para el proceso, el proyecto y el producto.

La validación de los requisitos se efectuó utilizando la técnica de **Prototipos de Interfaz de Usuario**, presentando los prototipos por cada requerimiento de software, aclarando con la Especialista Principal de Sistemas Automatizados del CADI, Nancy Rodríguez Calderín, si las necesidades del área de trabajo Depósitos Temporales de Frontera fueron cubiertas por el sistema. Para lograr una correcta interpretación de la información transmitida se realizaron **Revisiones**; donde la documentación fue puesta en tela de juicio ante otros analistas de mayor experiencia en el Departamento de Soluciones para la Aduana y fueron corregidos ciertos elementos. También se le hizo seguimiento a los requisitos por medio de la **Matriz de trazabilidad** Procesos del Negocio contra Requisitos Funcionales, la cual permitió marcar los procesos del negocio y chequearlos contra los requisitos que se obtuvieron de los mismos.

2.4 Aportes de la Solución y Beneficios Esperados

La solución propuesta posee las prestaciones necesarias para gestionar los procesos de los Depósitos Temporales de Frontera de forma segura, eficaz y eficiente, simplificando notablemente la labor de los inspectores de aduana que en ellos intervienen.

El sistema permitirá gestionar el inventario del depósito a partir de la totalidad de las entradas y extracciones de mercancías. Con la finalidad de garantizar el control en el destino final de las mercancías, en cada movimiento de estas que se efectúe el sistema le notificará automáticamente al destino, el cual estará en la obligación de confirmar la recepción o recibirlas según corresponda. También permitirá gestionar los procesos que se desarrollen debido al almacenamiento temporal de las mercancías en el depósito: el desagrupe[10] aplicado a contenedores y bultos que se encuentren con carga agrupada, el registro de las averías producidas a la mercancía y su reparación, el registro de la prórroga de la mercancía a la cuál le fue asignada y el registro de la mercancía declarada en abandono legal, así como la documentación asociada.

Proporcionará reportes que garanticen la toma de decisiones oportuna y rapidez en el procesamiento de informes, que sirvan de contrapartida a la información ofrecida por el operador del depósito y eviten al máximo la infracción de leyes.

Conclusiones

En el capítulo se describieron los procesos del negocio, a modo de solventar el problema con la solución propuesta y se realizó un análisis crítico de la ejecución de dichos procesos. Se definieron los requerimientos funcionales del sistema para satisfacer las expectativas del cliente y finalmente se expusieron los aportes y beneficios esperados con el desarrollo de la solución.

[10] Separación física a que deben ser sometidas las mercancías que se encuentran amparadas por un mismo conocimiento de embarque u otro documento que lo sustituya y que vienen consignadas a diferentes personas.

Capítulo 3: Diseño del Sistema

Introducción

En el capítulo se representa el diseño del subsistema Depósitos Temporales de Frontera para su inclusión en el sistema Gestión Integral de Aduanas como complementación a su concepción. Se describen los patrones de diseño utilizados en el desarrollo de la solución; se muestran los diagramas de clases del diseño con estereotipos Web y de secuencia, correspondientes al requerimiento Gestionar Extracción de Mercancías, por considerarse fundamental, ya que constituye una de las principales funciones que debe garantizar el sistema; se presentan además el diseño de la base de datos y el diagrama de despliegue.

3.1 Diseño del Sistema

El diseño de software es: "*…esa etapa de un sistema que describe cómo se implementará el sistema, en un nivel lógico sobre código real. En el diseño, las decisiones estratégicas y tácticas se toman para resolver los requisitos funcionales y de calidad requeridos de un sistema. Los resultados de esta etapa son representados por los modelos a nivel de diseño, especialmente la vista estática, vista de la máquina de estados, y vista de interacción..."*(21)

En sentido general el objetivo del diseño es modelar el sistema para concebir una arquitectura estable y sólida, que garantice el cubrimiento de todos los requerimientos, y represente una entrada adecuada, una fuente para las actividades de la implementación.

3.2 Patrones utilizados

Debido a que el marco de trabajo Symfony implementa varios de los patrones de diseño y arquitectónicos más utilizados actualmente, unificando buenas prácticas de trabajo y posibilitando que los desarrolladores de software no se tengan que preocupar por su implementación, brinda al desarrollo del sistema relevantes ventajas.

3.2.1 Patrones GRASP

Controlador: las peticiones Web son operadas por un mismo controlador frontal (dt_dev.php), que es el único punto de entrada de toda la aplicación en un determinado entorno. Al recibir una petición, el

controlador frontal utiliza el sistema de enrutamiento para relacionar el nombre de una acción y el nombre de un módulo, con la URL introducida por el usuario en el navegador.

Experto: Propel constituye la biblioteca de abstracción al modelo de datos que incorpora Symfony en su núcleo y se encarga de importar el modelo, encapsular toda la lógica de datos y generar las clases con las funcionalidades que son comunes en todas las entidades. De esta manera cada una de estas clases es experta en manipular su información.

Creador: la clase depositotemporalAction.class posee las acciones definidas para el módulo Depósitos Temporales de Frontera y es donde se ejecutan las funciones que conciben al sistema. Es ella la responsable de crear los objetos de las clases que representan las entidades, asegurando de este modo que es la "creadora" de estas entidades.

Alta Cohesión: Symfony admite la asignación de responsabilidades garantizando alta cohesión, por ejemplo la clase depositotemporalAction.class se encarga de definir acciones sobre las plantillas y asistir a otras para realizar diferentes operaciones e instanciar objetos, o sea, consiste en cubrir diferentes funcionalidades que se relacionan estrechamente unas con otras, permitiendo que el software presente una gran flexibilidad ante cambios significativos.

Bajo Acoplamiento: teniendo en cuenta que la localización de las relaciones entre las clases del modelo no debía incrementar el nivel de acoplamiento tanto que afectara negativamente, y reduciendo el impacto ante cambios, se obtuvo un diseño con clases independientes, aprovechando la reutilización donde las responsabilidades estuviesen mejor distribuidas.

3.2.2 Patrones GOF

Singleton (Instancia única): utilizado para garantizar la existencia solamente de una instancia para determinada clase y la creación de un mecanismo de acceso global a dicha instancia. Es el caso del controlador frontal, donde se aloja la función sfContext::getInstance() que resuelve que siempre se acceda a la misma instancia.

Abstract Factory (Fábrica abstracta): se utiliza este patrón para trabajar con objetos de familias diferentes evitando que se mezclen entre sí, pudiendo lograr que sea transparente el tipo de familia

concreta que se esté utilizando. Una vez el framework necesite crear un objeto nuevo, determina en la definición de la factoría cuál es el nombre de la clase que debe usarse para este fin.

Decorator (Decorador): amplía dinámicamente funcionalidades a una clase. El archivo layout.php, que constituye una plantilla global, porta el código HTML común a todas las páginas de un módulo definido en Symfony, para no tener que repetirlo en cada una por separado. El contenido de la plantilla se integra en el layout, o de igual manera se puede decir que el layout decora la plantilla.

3.2.3 Patrón MVC según Symfony

Symfony está basado en un patrón clásico del diseño web conocido como arquitectura Modelo-Vista-Controlador (MVC), que está formado por 3 niveles:

- El Modelo
- La Vista
- El Controlador

El elemento más representativo de la arquitectura MVC es la separación del código del programa en tres capas, dependiendo de su naturaleza. La lógica relacionada con los datos se incluye en el modelo, el código de la presentación en la vista y la lógica de la aplicación en el controlador. La programación se puede simplificar si se utilizan otros patrones de diseño. De esta forma, las capas del modelo, la vista y el controlador se pueden subdividir en más capas.

La capa del modelo se puede dividir en la capa de acceso a los datos y en la capa de abstracción de la base de datos. De esta forma, las funciones que acceden a los datos no utilizan sentencias ni consultas que dependen de una base de datos, sino que utilizan otras funciones para realizar las consultas. Así, si se cambia de sistema gestor de bases de datos, solamente es necesario actualizar la capa de abstracción de la base de datos.

La capa de la vista también separa su código. Las páginas web suelen contener elementos que se muestran en toda la aplicación: cabeceras, el layout genérico, el pie de página y la navegación global. Como lo más normal es que sólo cambie el interior de la página, la vista se divide en un layout y en una plantilla.

La capa del controlador incluye como parte importante de su trabajo, elementos comunes a todos los controladores de la aplicación. Entre estas tareas comunes se encuentran el manejo de las peticiones del usuario, el manejo de la seguridad, cargar la configuración de la aplicación y otras tareas similares. Por este motivo, el controlador normalmente se divide en un controlador frontal, que es único para cada aplicación, y las acciones, que incluyen el código específico del controlador de cada página.

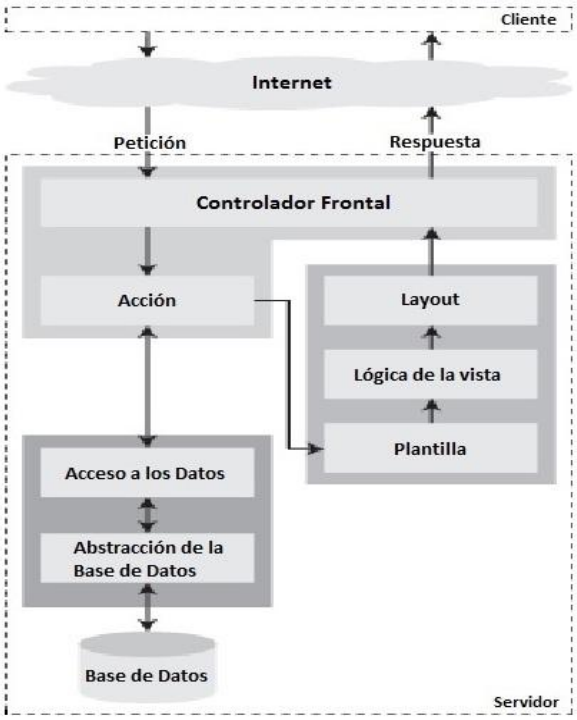


Fig. 3. 1 El flujo de trabajo de Symfony

3.3 Clases del Diseño

"Una clase del diseño es una abstracción sin costuras de una clase o construcción similar en la implementación del sistema" (22).

Las extensiones UML para el diseño Web, exponen la solución para el modelado de diagramas de clases del diseño sobre tecnologías Web, de forma que quedan presentados los siguientes estereotipos.

Página Servidora (Server Page ): simboliza una página Web dinámica que posee el código construido por el servidor, normalmente porta scripts que se ejecutan en el servidor y que interactúan con los recursos de este lado: las bases de datos, los componentes de la lógica del negocio y sistemas externos.

Página Cliente (Client Page ): se utiliza para visualizar una instancia de página cliente. Figura una página Web con código HTML, que interpreta y muestra el navegador del cliente.

Formulario (Form ): representa un formulario, es el elemento que se encarga de enviar datos desde la página cliente hacia la página servidora.

3.3.1 Diagramas de Clases con Estereotipos Web

El manejo del marco de trabajo Symfony en contraste con las bibliotecas ExtJS facilita el desempeño del diseño con una distribución similar en varios requerimientos. El controlador frontal envía las peticiones hacia la página servidora depositotemporalAction.class que redirecciona hacia el layout, este carga el cuerpo de la plantilla (generalmente es una página en blanco y no será representada) y se construye una página cliente contenedora de la interfaz de usuario correspondiente (estas interfaces se encuentran definidas en el archivo de configuración view.yml y aparecen representadas en el paquete Interfaces JS). Es muy habitual además la presencia de una relación de agregación entre la página cliente y un formulario, responsable de llevar los componentes para la entrada de datos que serán enviados al servidor y procesados por las funciones de la clase depositotemporalAction.class y por medio de las clases del paquete “Acceso a Datos” y el Núcleo del Sistema*, que garantizan el registro y obtención de la información (ver figura 3.2). Para el resto de los diagramas de clases del diseño con estereotipos Web ver expediente de proyecto.

***Núcleo del sistema**: alude a la parte del GINA encargada del intercambio de información entre subsistemas, los cuales son expertos en el manejo de información específica. Los depósitos temporales manipulan datos relacionados directamente con el sistema que gestiona los datos referentes a los medios de transporte internacional como por ejemplo (manifiestos de exportación, importación, tránsito y transferencia), el de despacho comercial que tramita lo concerniente a declaraciones de mercancías, el de recepción electrónica que se encarga de manejar los ficheros electrónicos y el de enfrentamiento que se ocupa de definir tratamientos especiales a determinada mercancía.

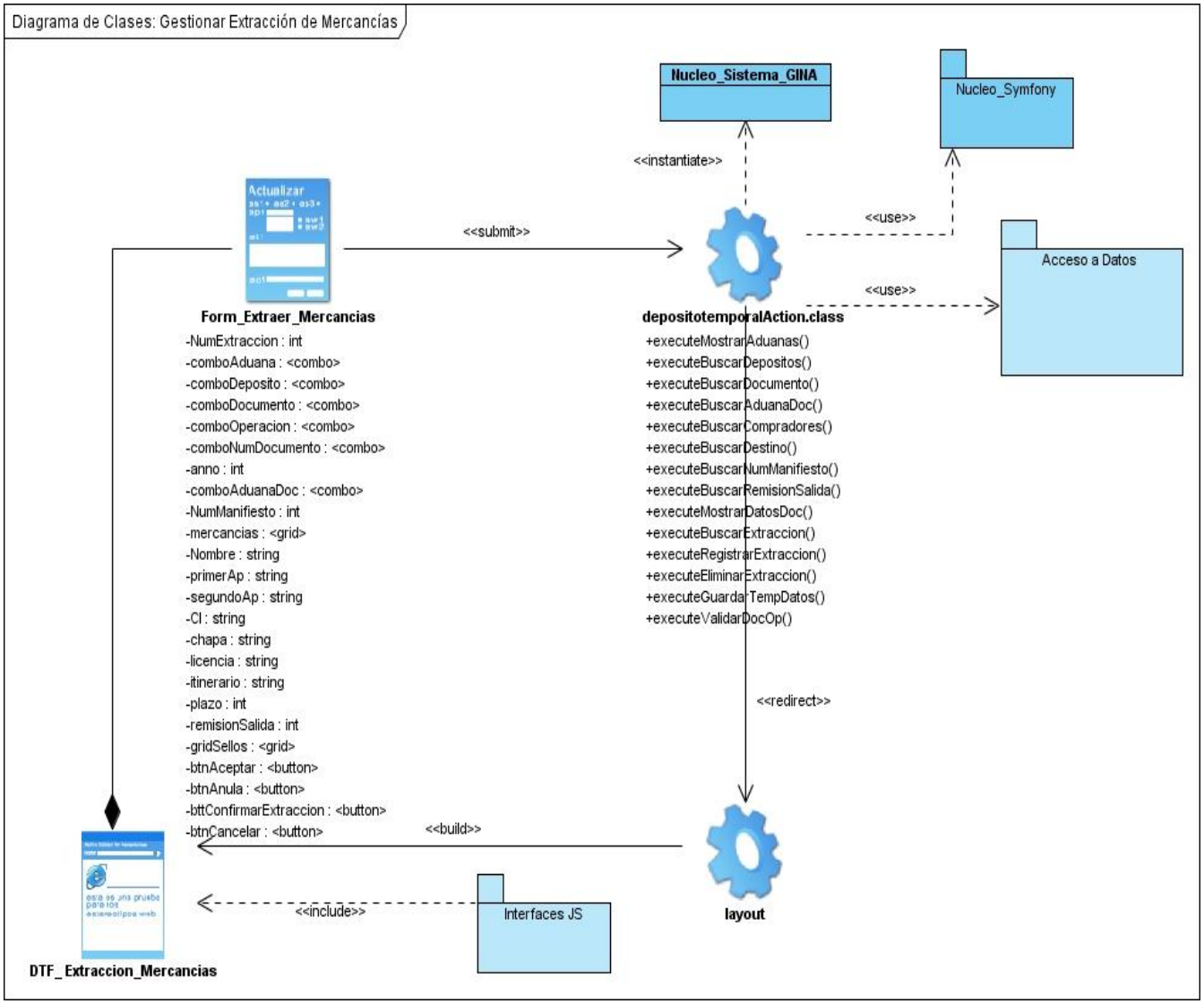


Fig. 3. 2 Diagrama de clases Gestionar extracción de mercancías

3.3.2 Paquetes de Acceso a Datos e Interfaces JS

El paquete de “Acceso a Datos” está compuesto por un paquete “Objetos” que incluye un paquete “Objetos Peer”, además ambos hacen uso del paquete “Propel” (ver Figura 3.3). Estos paquetes son los encargados de gestionar el intercambio de información entre la aplicación y la base de datos.

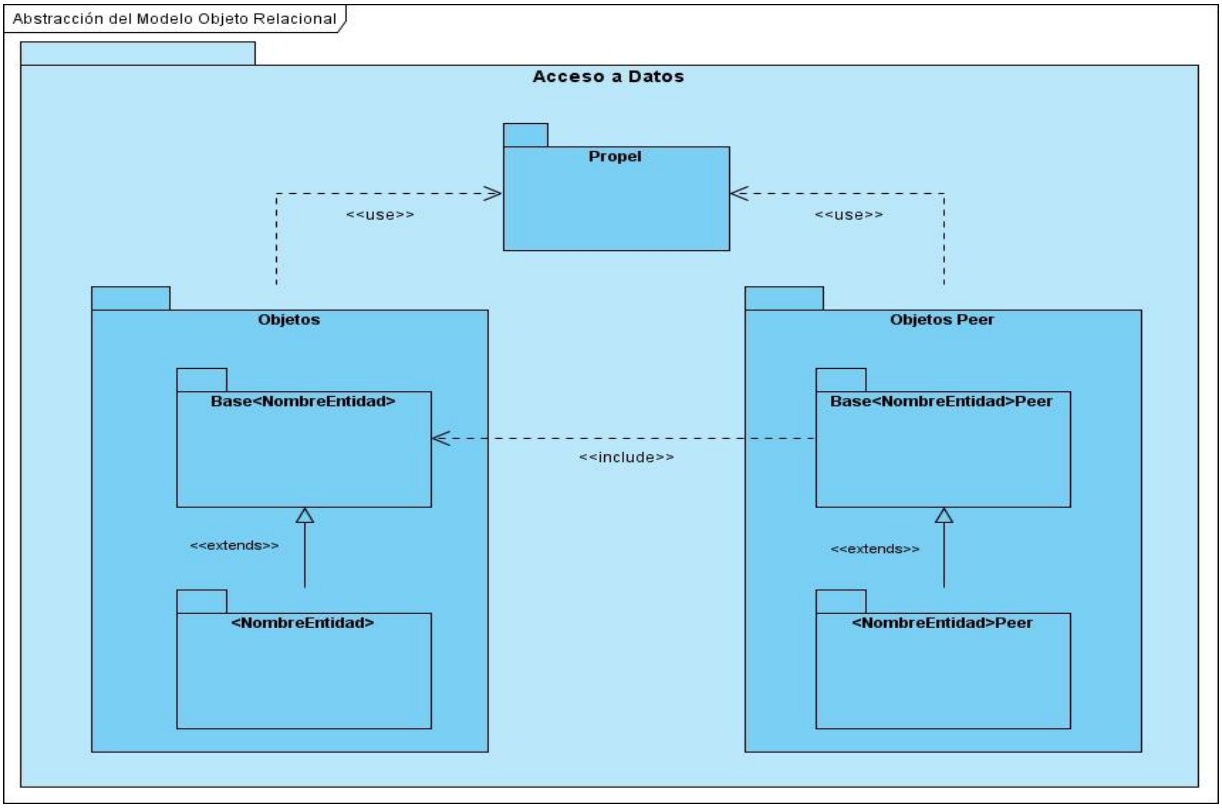


Fig. 3. 3 Paquete de Acceso a Datos

En el caso del paquete Objetos está definido por las clases de la lógica de la aplicación que contienen los atributos y métodos necesarios, que mediante el uso de Propel garantizan la inserción y actualización de los datos persistentes. Por cada entidad del modelo físico de datos, existen dos clases dentro del paquete Objetos, una llamada <NombreEntidad> que hereda de otra denominada Base<NombreEntidad> (Por ejemplo: Extraccion hereda de BaseExtraccion). Las modificaciones se efectúan sobre la clase hija, pues la Base<NombreEntidad> posee las funciones para acceder a la base de datos mediante Propel, e incluyendo del paquete Objetos Peer su clase correspondiente Base<NombreEntidad>Peer (para el ejemplo: BaseExtraccionPeer).

El paquete Objetos Peer está compuesto por dos clases por cada una de las entidades de la base de datos, similar al paquete Objetos, estas implementan los métodos y atributos necesarios para ejecutar las consultas a la base de datos (siguiendo el ejemplo sería: ExtraccionPeer que hereda de BaseExtraccionPeer).

El paquete “Interfaces JS” porta los códigos JavaScript que son interpretados por el navegador en la representación de la página cliente (Ver Fig. 3.4). Los entes Ext-base.js y Ext-all.js constituyen el núcleo de las bibliotecas ExtJS manejadas, y el resto representan las interfaces del módulo Depósitos Temporales.

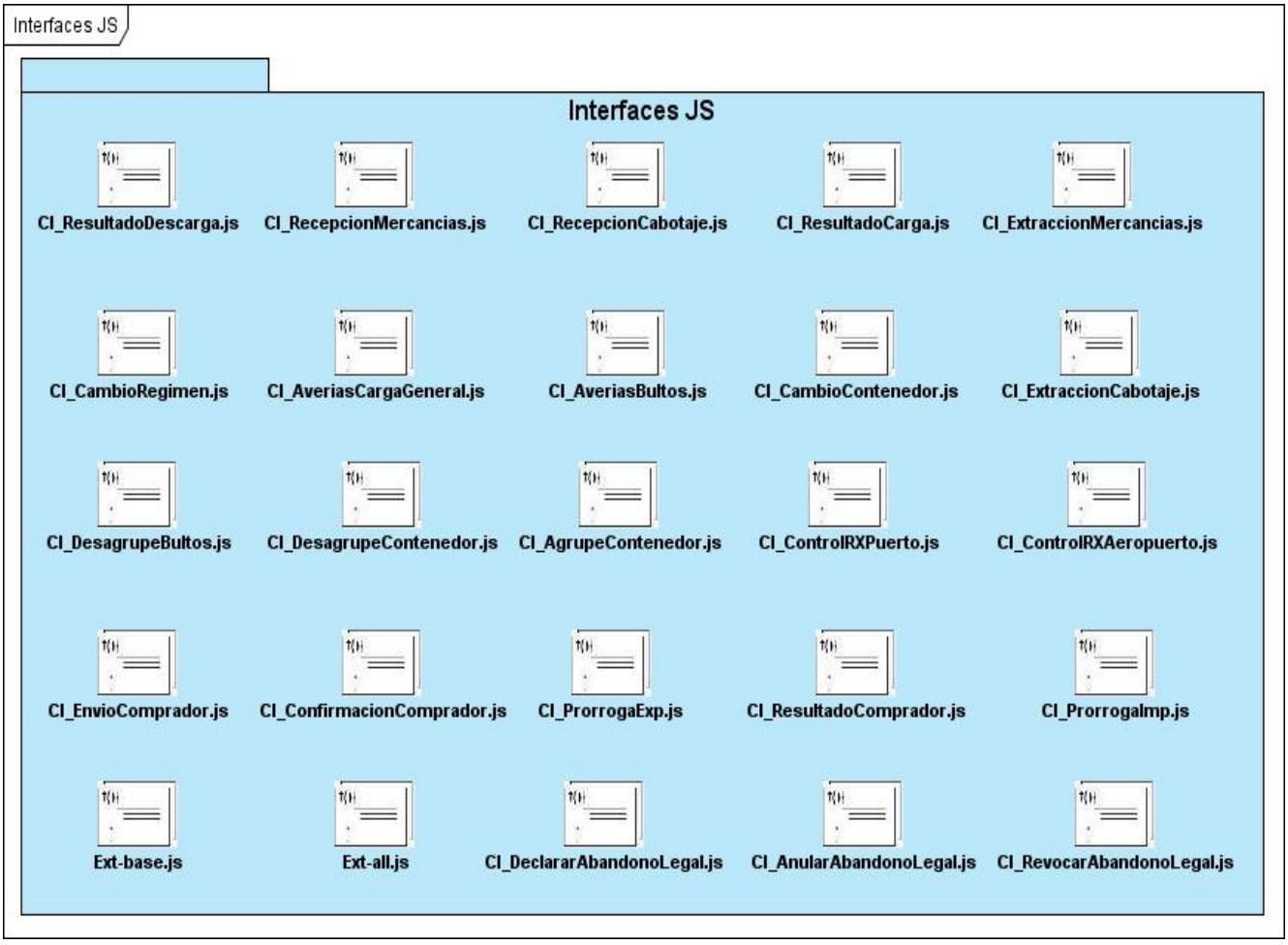


Fig. 3. 4 Paquete Interfaces JS

3.4 Diagramas de Secuencia del Diseño

A continuación se muestra el Diagrama de secuencia representativo del requerimiento Gestionar Extracción de Mercancías. Para el resto de los diagramas de secuencia ver expediente de proyecto.

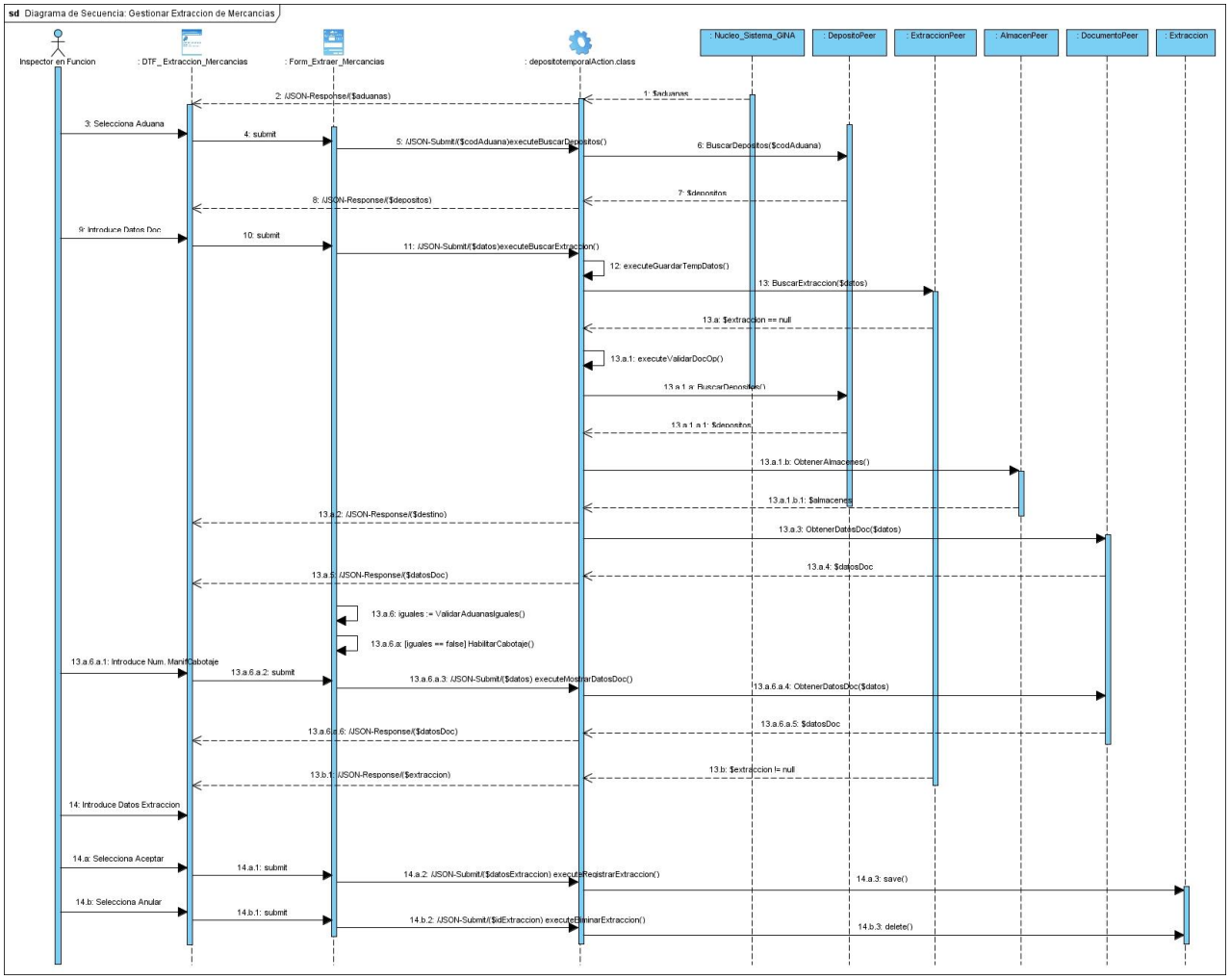


Fig. 3. 5 Diagrama de secuencia Gestionar extracción de mercancías

3.5 Diseño de la Base de Datos

La siguiente imagen representa la distribución lógica de las entidades de la base de datos del sistema que se modela.

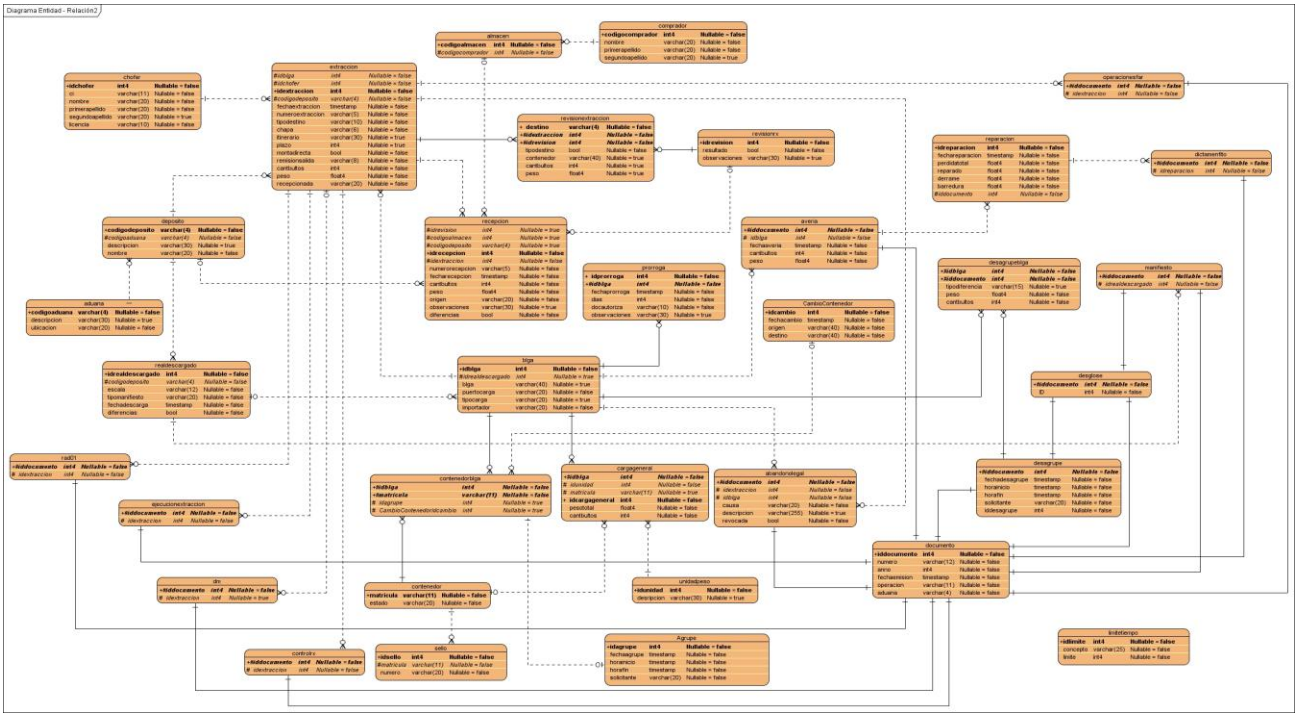


Fig. 3. 2 Modelo de base de datos para Depósitos Temporales

3.6 Diagrama de Despliegue

El diagrama de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo (22).

Fig. 3. 3 Diagrama de Despliegue

Conclusiones

En este capítulo se expusieron los patrones de diseño aplicados al modelo realizado y se muestra una panorámica de cómo el marco de trabajo Symfony aplica el patón MVC, se obtuvieron los siguientes artefactos del diseño de la aplicación: diagramas de clases del diseño con estereotipos Web, diagramas de secuencia, el diseño de la Base de Datos y el diagrama de despliegue.

Capítulo 4 Validación del Diseño

Introducción

En el capítulo se caracterizan algunas de las métricas existentes para sistemas orientados a objetos (OO), y se aplican al modelo de diseño propuesto con la finalidad de comprobar la complejidad del mismo y su acabado para ser llevado a fases posteriores del desarrollo del software.

Se conoce que las medidas y las métricas son componentes claves de cualquier disciplina de la ingeniería; la ingeniería de software orientada a objetos no es una excepción. Lamentablemente, la utilización de métricas para sistemas orientados a objetos ha progresado con mucha más lentitud que la de los demás métodos OO (23). En la actualidad con el auge de los sistemas OO, resulta fundamental que los ingenieros del software dispongan de mecanismos cuantitativos para evaluar la calidad de los diseños y la efectividad de los programas.

Los objetivos principales de las métricas orientadas a objetos son los mismos de las métricas surgidas para el software estructurado: comprender mejor la calidad del producto, estimar la efectividad del proceso y mejorar la calidad del trabajo realizado en el nivel del proyecto.

4.1 Características del software Orientado a Objetos

Las métricas para sistemas OO deben coincidir con las características del software OO que lo distinguen del convencional. Berard (23) define cinco características que dan lugar a unas métricas especializadas:

- Localización
- Encapsulamiento
- Ocultamiento de información
- Herencia
- Técnicas de abstracción de objetos

4.2 Métricas para el modelo de diseño Orientado a Objetos

Gran parte del diseño orientado a objetos es subjetivo; un diseñador experimentado domina como caracterizar un sistema OO para que se implementen de forma efectiva los requisitos del cliente. Pero a medida que los modelos de diseño OO aumentan en tamaño y complejidad, puede resultar beneficiosa una visión más objetiva de las características del diseño, tanto para el diseñador experimentado como para el menos. Para esto las métricas OO son de gran apoyo pues brindan un componente cuantitativo que le confiere ecuanimidad a la perspectiva de dicho modelo.

4.2.1 Métricas orientadas a Clases

La clase es la unidad principal de todo sistema OO. Por ende, las medidas y métricas para una clase, la jerarquía de clases, y las colaboraciones de clases resultarán muy valiosas para un ingeniero de software que necesite juzgar la calidad de un diseño.

4.2.1.1 El conjunto de métricas Chidamber-Kemerer

Uno de los conjuntos de métricas de software OO a los que se hace mayor referencia es el propuesto por Shyam R.Chidamber y Chris F.Kemerer en 1994. Algunas de estas métricas están enmarcadas dentro de lo que se conoce como métricas a nivel de herencia, otras a nivel de acoplamiento y otras a nivel de clases. (24)

Las métricas consideradas a nivel de herencia son:

Profundidad del árbol de herencia (DIT: Depth of Inheritance Tree): se define como la profundidad del árbol de una clase (en los casos de herencia múltiple es la longitud máxima desde el nodo hasta la raíz del árbol). Se basa en que cuanto más profunda está la clase en la jerarquía, mayor número de operaciones puede heredar.

Número de hijos (NOC: Number of children): número de subclases inmediatas subordinadas a una clase, es decir, la cantidad de subclases que pertenecen a una clase. Es un indicador del nivel de reutilización, la posibilidad de haber creado abstracciones erróneas, y del nivel de pruebas requerido. Aunque mayor número de hijos representa mayor reutilización de código, puede presentarse el

inconveniente de que modificar una clase base con muchos hijos implique perturbaciones en todos los hijos que tienen dependencia.

Las métricas a nivel de acoplamiento son:

Acoplamiento entre objetos (CBO: Coupling between object): es el número de clases a las cuales una clase está ligada. Se da dependencia entre dos clases cuando una de ellas usa métodos o variables de la otra. Las relaciones por herencia no se tienen en cuenta. Las clases con excesivo acoplamiento dificultan la comprensión y hacen más difícil el mantenimiento, por lo que será necesario un mayor esfuerzo y unas pruebas rigurosas. Al reducir el acoplamiento se reduce la complejidad, se mejora la modularidad y se promueve la encapsulación.

Las métricas que se consideran a nivel de clases identifican características dentro de las clases destacando diferentes aspectos de sus abstracciones y ayudando a descubrir clases que podrían necesitar ser rediseñadas.

Respuesta para una clase (RFC: Response For a Class): el conjunto de respuesta de una clase es "*una serie de métodos que pueden potencialmente ser ejecutados, en respuesta a un mensaje recibido por un objeto, en la clase*" (25). En otras palabras, RFC cuenta las ocurrencias de llamadas a otras clases desde una clase particular.

Métodos ponderados por clase (WMC: Weighted Methods per Class): se establece como una medida de la complejidad de una clase. Clases con un gran número de métodos requieren más tiempo y esfuerzo para desarrollarlas y mantenerlas, ya que influirán en las subclases heredando todos sus métodos. Además, estas clases tienden a ser específicas de la aplicación, con lo que se limita su posibilidad de reutilización.

Carencia de cohesión en los métodos (LOCM: Lack of Cohesion in Methods): establece en qué medida los métodos hacen referencia a atributos. LOCM mide la cohesión de una clase mediante el número de atributos comunes usados por diferentes métodos, indicando la calidad de la abstracción hecha en la clase. Un valor alto de LOCM implica falta de cohesión, es decir, escasa similitud de los métodos. Esto quizás signifique que la clase está compuesta de elementos no relacionados, incrementándose la complejidad y la probabilidad de errores durante el desarrollo.

4.2.1.2 Métricas propuestas por Lorenz y Kidd

Lorenz y Kidd (26) definen métricas basadas en clases en cuatro amplias categorías: tamaño, herencia, valores internos y valores externos. Las métricas orientadas al tamaño para las clases OO se centran en el recuento de atributos y operaciones para cada clase individual, y los valores promedio para el sistema OO como un todo. Las métricas basadas en la herencia se enfocan en la forma en que las operaciones se reutilizan en la jerarquía de clases. Las métricas para valores internos de clase examinan la cohesión y los aspectos orientados al código; las métricas orientadas a valores externos, examinan el acoplamiento y la reutilización.

Métricas de tamaño:

Número de Métodos Públicos (NPM: Number of Public Methods): es el número total de métodos públicos de instancias (los que están disponibles como servicios para otras clases). Se considera que mide la cantidad de responsabilidad que tiene una clase.

Número de Métodos (NM: Number of Methods): se define como la suma de todos los métodos (públicos, protegidos y privados) de una clase. Según los autores es una medida de la cantidad de colaboración utilizada.

Número de Variables (NV: Number of Variables): se determina por el número total de variables (privadas y protegidas) a nivel de instancia que tiene una clase.

Número de Métodos de Clase (NCM: Number of Class Methods): es el número total de métodos a nivel de clase.

Número de Variables de Clase (NCV: Number of Class Variables): es el total de variables a nivel de clase.

Métricas de herencia:

Número de Métodos Redefinidos (NMO: Number of Methods Overridden): es el número total de métodos redefinidos en una subclase, se utiliza para medir la calidad del uso de la herencia.

Número de Métodos Heredados (NMI: Number of Methods Inherited): se define como el número de métodos que hereda una clase, también mide la calidad del uso de la herencia.

Número de Métodos Añadidos (NNM: Number of New Methods): número total de métodos que se definen en una subclase, igual que las anteriores mide la calidad de uso de la herencia.

Índice de Especialización (SIX: Specialization Index): para una clase se define como el número de métodos sobrescritos multiplicado por el nivel de anidamiento en la jerarquía y dividido por el número total de métodos.

Métricas de características internas de una clase:

Promedio de Parámetros por Método (APM: Average Parameters per Method): cociente entre el número total de parámetros por método y el número total de métodos. Lorenz y Kidd refieren que APM no debe exceder de 0.7.

4.2.1.3 La colección de métricas MDOO

Harrison, Counseil y Nithi (27) propusieron un conjunto de métricas para el diseño orientado a objetos (MDOO), que proporcionan indicadores cuantitativos para el diseño de características OO, establecidas para medir algunos de los principales mecanismos de los modelos orientados a objetos como encapsulamiento (MHF y AHF), polimorfismo (PF), herencia (MIF y AIF) y paso de mensajes (CF); para poder evaluar la productividad del desarrollo y la calidad del producto. Están encuadradas dentro las métricas a nivel de sistema y se definieron para aplicarlas a nivel de diagramas de clases.

Factor de ocultamiento de los métodos (MHF: Method Hiding Factor): cociente entre la suma de las invisibilidades de todos los métodos definidos en todas las clases y el número total de métodos definidos en el sistema. La invisibilidad de un método es el porcentaje total de clases desde las cuales el método es invisible. No se consideran los métodos heredados.

Factor de ocultamiento de los atributos (AHF: Attribute Hiding Factor): se define como el cociente entre el número de invisibilidades de todos los atributos definidos en todas las clases y el número total de atributos definidos en el sistema. Idealmente el valor de esta métrica debería ser siempre del 100%, evitando el uso de atributos públicos para no violar principios de encapsulación.

Factor de herencia de métodos (MIF: Method Inheritance Factor): es el cociente entre el número de métodos heredados en todas las clases del sistema y el número total de métodos (heredados y locales) en todas las clases.

Factor de herencia de los atributos (AIF: Attribute Inheritance Factor): está definido como el cociente entre el número de atributos heredados en todas las clases del sistema y el número total de atributos existentes (heredados y definidos localmente) en todas las clases.

Si bien es conveniente aprovechar la herencia para reproducir el comportamiento común de algunas clases, demasiada reutilización de código hace que el sistema sea más difícil de entender y mantener.

Factor de polimorfismo (PF: Polymorphism Factor): Harrison, Counsell y Nithi (27) definen PF como "*el número de métodos que redefinen métodos heredados, dividida por el máximo número de posibles situaciones polimórficas distintas...; de esta manera, el PF es una medida indirecta de la cantidad relativa de ligadura dinámica en un sistema*".

Factor de acoplamiento (CF: Coupling Factor): se define como la proporción entre el máximo número posible de acoplamientos en el sistema y el número real de acoplamientos no imputables a la herencia. En otras palabras, indica la comunicación entre clases.

El acoplamiento se ve como una medida del incremento de la complejidad, reduciendo la encapsulación y la posible reutilización; limita, por tanto, la facilidad de comprensión y de mantenimiento del sistema.

4.2.2 Métricas Orientadas a Operaciones

Tamaño medio de operación (TOavg): el número de mensajes enviados por la operación proporciona una alternativa para el tamaño de la operación. A medida que asciende el número de mensajes enviados por una única operación, es posible que las responsabilidades no hayan sido bien asignadas dentro de la clase.

Complejidad de operación (CO): la complejidad de una operación se consigue calcular empleando cualquiera de las métricas de complejidad propuestas para el software convencional. Sabiendo que convendría limitar las operaciones a una responsabilidad específica, en donde el diseñador debería esforzarse por mantener el valor de CO tan bajo como sea posible.

Número Medio de Parámetros por operación (NPavg): cuanto más grande sea el número de parámetros de la operación, será más compleja la colaboración entre objetos. En general, NPavg debería de mantenerse tan bajo como sea posible.

4.3 Aplicación de las métricas

Tras el análisis de algunas de las métricas más importantes para valorar el diseño de software OO, se decidió concentrar la evaluación sobre dos métodos de cada una de las métricas orientadas a clases, a continuación se exponen los resultados de las medidas seleccionadas.

En el caso de las métricas propuestas por Chidamber y Kemerer se aplicó **Profundidad del árbol de herencia** y **Número de hijos.**

Para la aplicación de la métrica **Profundidad del árbol de herencia** es primordial definir el umbral a utilizar para la medida y poder establecer un criterio a partir de los resultados. El umbral que se ha determinado (ver Tabla 4.1) usar ha sido establecido por algunos especialistas y se considera modesto para medir la complejidad de la herencia del diseño.

Jerarquía de clases	
DIT	**Umbral**
Sencillo	<= 5
Complejo	> 5

Tabla 4. 1 Umbral para métrica DIT

Luego de emplear la métrica se obtuvo un valor de 3 como la profundidad máxima que manifiesta el árbol representativo de la jerarquía de clases del diseño, pues se presenta herencia en 92 de las 93 clases, donde se reutilizan la mayoría de los métodos, siendo de esta manera una ventaja el hecho de que no exista una jerarquía de clases muy profunda, facilitando el mantenimiento y minimizando la complejidad. Además se encuentra este valor dentro del umbral que representa un diseño sencillo, eliminando dificultades para predecir el comportamiento de una clase.

El **Número de hijos** se mide obteniendo por cada clase que tenga descendientes, el número total de estos. Si una clase presenta muchos descendientes puede visualizarse mayor reutilización pero esto

puede ser además un indicador de la disolución de la clase predecesora y de que probablemente existan hijos que no sean realmente miembros de esa clase.

Para lograr un diseño donde cada clase sea realmente miembro de su predecesora y se reduzca la cantidad de pruebas necesarias para ejercitar cada uno de los miembros de la jerarquía de clases debe obtenerse un valor bajo de NOC, el diseño propuesto presenta 46 clases con solo 1 subclase respectivamente, lo que representa un índice verdaderamente ajustado a las condiciones de un diseño menos complejo y fácil de mantener.

Los bajos valores de DIT y NOC indican que las oportunidades de reutilización (mediante la herencia) quizás se vean comprometidas a favor de la comprensión de toda la arquitectura de la aplicación. En tales casos, los bajos valores de la métrica revelan que se han seguido las adecuadas particularidades de diseño (28).

Del conjunto de métricas planteadas por Lorenz y Kidd se aplicaron para medir el tamaño de clases (TC) **Número de Métodos** y **Número de Variables**, para lo cual el umbral utilizado se muestra a continuación (Tabla 4.2).

Número de Operaciones o Atributos	
TC	**Umbral**
Pequeño	<= 20
Medio	>20 y <=30
Grande	>30

Tabla 4. 2 Umbral para TC

Para un total de 93 clases que pertenecen al diseño se obtuvieron tras aplicar la métrica valores promedio de 20.4 y 7.4, operaciones y atributos por clase respectivamente, y habiendo representado el conteo de las clases en correspondencia con el umbral especificado, fue detectada la presencia de 46 de ellas de tamaño pequeño, 11 medianas y 36 grandes (ver Fig. 4.1). En el caso de las clases grandes, un gran por ciento de estas es utilizado por el framework Symfony para la gestión de la abstracción de datos y muy ocasionalmente necesitan mantenimiento; razón por la cual los valores de TC anómalos no necesariamente aluden a un diseño con dificultades.

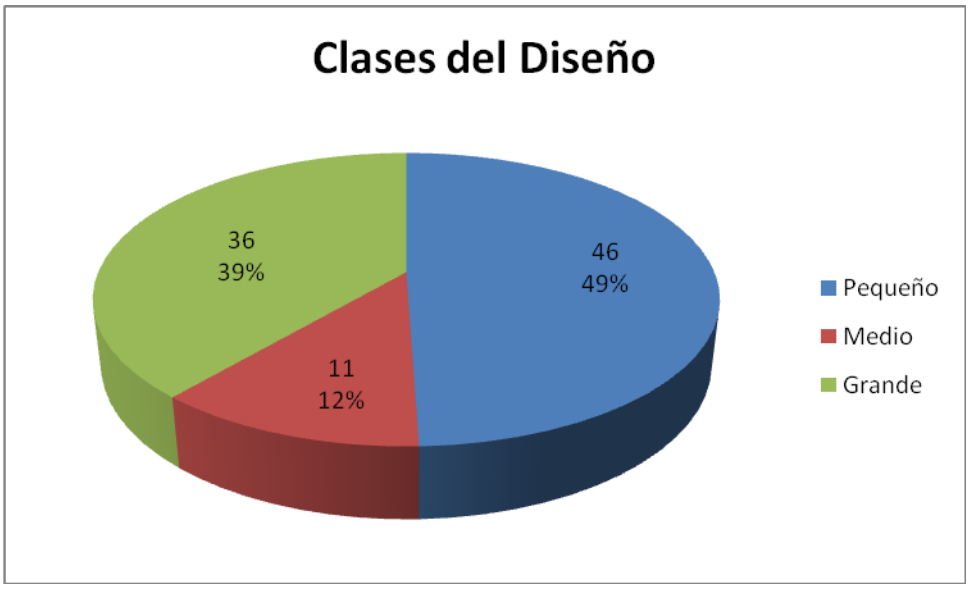


Fig. 4. 1 Porcentaje según Tamaño de Clase

Entre las métricas MDOO han sido seleccionadas para aplicar al diseño propuesto **Factor de herencia de métodos** y **Factor de herencia de atributos**.

El resultado de aplicar estas métricas arrojó valores de 0.96 y 0.17 respectivamente, lo que significa que la herencia está basada mayoritariamente en los métodos y en menor grado en los atributos. Esto le confiere al diseño capacidades de reutilización, minimiza dilemas de acoplamiento y cohesión, y facilita su comprensión.

Conclusiones

En este capítulo se aplicaron métricas para la valoración del modelo de diseño elaborado, alcanzando de esta manera el objetivo trazado. Con el apoyo de estas medidas se garantizó que el diseño propuesto no manifestara una alta complejidad logrando su integración sin inconvenientes al GINA con el resto de los subsistemas y evitando que existan complicaciones innecesarias en las fases de implementación y pruebas. Se consiguió utilizar correctamente la herencia, el acoplamiento y la cohesión entre clases, reduciendo el impacto ante cambios y facilitando su reutilización y mantenimiento.

Conclusiones

Al finalizar este trabajo se lograron cumplir los objetivos planteados de manera satisfactoria. El estudio del estado del arte de las soluciones informáticas existentes para la gestión de las mercancías en los depósitos temporales bajo control aduanero arrojó como resultado que ninguna de las analizadas gozaba de aceptación para ser acogida por la Aduana General de la República de Cuba teniendo en cuenta las especificaciones definidas para el sistema Gestión Integral de Aduanas.

Se establecieron los requerimientos funcionales del subsistema a partir de confrontaciones con los clientes y sobre la base de las características del negocio, y aplicando las técnicas pertinentes se chequeó la validez de la propuesta. Aprovechando las herramientas utilizadas se obtuvieron los artefactos correspondientes al flujo de trabajo de RUP “Análisis y Diseño”, establecidos en el Departamento de Soluciones para la Aduana, confeccionando la documentación que determina el expediente de proyecto, lo cual se aprobó mediante métricas asociadas al software orientado a objetos que avalara un modelo de diseño apto totalmente y con la calidad pretendida para continuar con el proceso de desarrollo de software.

Recomendaciones

Los objetivos propuestos en este trabajo fueron alcanzados satisfactoriamente; sin embargo a lo largo del desarrollo de los mismos surgieron nuevas ideas que serían recomendables tener en cuenta:

1. Realizar la implementación del subsistema diseñado a fin de brindarle al personal de los depósitos temporales la herramienta que facilite su trabajo.
2. Dar seguimiento al estudio de los procesos en los depósitos temporales y al desarrollo del subsistema con el propósito de detectar nuevas necesidades e incorporarle funcionalidades no contempladas.
3. Extender la búsqueda de nuevas tecnologías informáticas para perfeccionar el subsistema Depósitos Temporales del sistema de Gestión Integral de Aduanas en futuras versiones.

Referencias Bibliográficas

1. AGR. *Resolución 33/96, Glosario de Términos Aduaneros.*

2. S-4. *S-4* Disponible en: http://www.s-4.es/ms_Depositos.aspx.

3. *Taric S.A. Glosario de Comercio Exterior*. Disponible en: http://www.taric.es/Services/glosario/glosario.asp.

4. SOFTWAY. *SOFTWAY - Soluções e Software para Comércio Exterior | Produtos: EV Sys* Disponible en: http://www.softcomex.com.br/site_espanhol/produtos_ev.shtml.

5. *Cotecna*. Disponible en: http://www.cotecna.com/COM/ES/home.aspx.

6. USINAD, S. L. *Aplicación de Software para DEPOSITO ADUANERO, DEPOSITO DISTINTO ADUANERO, A.D.T—L.A.M.E.* Disponible en: http://www.usinad.es/pdf/DEPOADUWEB.pdf.

7. CENTRO NACIONAL DE TECNOLOGÍAS DE INFORMACIÓN, C. N. T. I. *Sistema Aduanero Automatizado SIDUNEA* Gobierno Bolivariano de Venezuela, Disponible en: http://www.gobiernoenlinea.ve/cartelera/Sidunea.html.

8. UNCTAD. *Fondo Fiduciario de UNCTAD para las Negociaciones de Facilitación del Comercio Nota Técnica № 21. Sistema Aduanero Automatizado SIDUNEA.* 2007, Disponible en: http://r0.unctad.org/ttl/technical-notes-sp/TN21_Sidunea.pdf.

9. IEEE STD 610.12-1990. *IEEE Standard Glossary of Software Engineering Terminology.* Disponible en: http://standards.ieee.org/reading/ieee/std_public/description/se/610.12-1990_desc.html.

10. ESCALONA, M. J., KOCH N. *Ingeniería de Requisitos en Aplicaciones para la Web - Un estudio comparativo.* 2002.

11. BASS L., CLEMENTS P., KAZMAN R. *Software Architecture in Practice.* Second ed. 2003. ISBN 0-321-15495-9.

12. CLEMENTS P. A Survey of Architecture Description Languages. En *Proceedings of the International Workshop on Software Specification and Design.* Alemania: 1996.

13. GARLAN D. Software Architecture: A Roadmap. En Anthony, F. (editor). *The future of software engineering.* ACM Press, 2000.

14. REYNOSO C. B. *Introducción a la Arquitectura de Software Versión 1.0.* Buenos Aires: 2004.

15. CHRISTOPHER A., ISHIKAWA S., SILVERSTEIN M. *A Pattern Language: Towns, Buildings, Construction (Center for Environmental Structure Series).* New York: Oxford University Press, 1977. ISBN 0195019199.

16. LARMAN C. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México: PRENTICE HALL, 1999, 536 p. ISBN 970-17-0261-1.

17. COPLIEN J. *Software Patterns.* 1996. ISBN 978-1884842504.

18. REYNOSO C., KOCH N. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* UNIVERSIDAD DE BUENOS AIRES: 2004, Disponible en: http://www.willydev.net/descargas/prev/Estiloypatron.pdf.

19. BOOCH G. *Object-Oriented Analysis and Design with Applications.* Addison-Wesley, 2007, ISBN 0-201-89551-X.

20. CORPORATION, R. S. *Ayuda extendida de RUP (Rational Unified Process) Version 2003.06.00.65.* 2003.

21. GRADY BOOCH, RUMBAUGH J., JACOBSON I. *El Lenguaje Unificado de Modelado. Manual de referencia.* Madrid: Pearson Educación, 2000. ISBN 84-7829-037-0.

22. ---. *El Proceso Unificado de Desarrollo de Software.* Addison Wesley: 2000. ISBN 84-7829-036-2.

23. LARANJEIRA A. L. *Software Size Estimation of Object-Oriented System.* 5 ed. IEEE Transactions on Software Engineering, 1990. vol. 16.

24. PRESSMAN R. S. *Ingeniería del software, un enfoque práctico.* 5 ed. Madrid: McGraw-Hill, 2002. ISBN 84-481-3214-9.

25. CHIDAMBER S. R., KEMERER C.F. *A Metrics Suite for Object-Oriented Design.* 6 ed. IEEE Trans. Software Engineering, 1994. vol. 20, 476-493 p.

26. LORENZ M., KIDD J. *Object-Oriented Software Metric.* Prentice-Hal, 1994.

27. HARRISON R., COUNSEIL S.J., NITHI R.V. *An Evaluation of the MOOD Set of Object-Oriented Software Metrics.* 6 ed. IEEE Trans. Software Engineering, 1998. vol. 24, 491-496 p.

28. CHIDAMBER S.R., DARCY D. P., KEMERER C. F. *Managerial Use of Metrics for Object-Oriented Software: An Exploratory Analysis.* IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 1998. vol. 24, 629-639 p.

Bibliografía Consultada

BUSCHMANN, F. Y. O. Pattern-Oriented Software Architecture. A System of Patterns. John Wiley, vol. 1, ISBN 978-0-471-95869-7.

CONALLEN, J. *Building Web Applications with UML.* Addison Wesley, 2002. 496 p. ISBN 0-201-73038-3

CONALLEN, J. *Modeling Web Application Design with UML.* 2006, Disponible en: http://www.itmweb.com/essay546.htm.

ERIKSSON H.-E., M. PENKER. *Business Modeling with UML: Business Patterns at Work.* John Wiley & Sons, 2000. 416 p. ISBN 0471295515.

GAMMA, E. Y. O. *Design Patterns-Elements of Reusable Object-Oriented Software.* Addison-Wesley, 1995. ISBN 0201633612.

JAMALI, S. M. *Object Oriented Metrics (A Survey Approach).* Department of Computer Engineering Sharif University of Technology, 2006.

JAMES RUMBAUGH, M. B., WILLIAM PREMERLANI, FREDERICK EDDY, WILLIAM LORENSEN. *Modelos y Diseño Orientados a Objetos.* Prentice-Hall, 1996. 675 p. ISBN 0-13-240698-5.

MARIO G. PIATTINI VELTHUIS, F. O. G. R. Calidad en el desarrollo y mantenimiento del software. RA-MA EDITORIAL, 2003. 344 p. ISBN 8478975446

MINISTERIO DE ECONOMÍA Y FINANZAS, U. *PROGRAMA DE MODERNIZACION DE LA ADUANA URUGUAYA.* Disponible en: http://www.mef.gub.uy/documentos/aduana_modernizacion.pdf.

MOHAMED EL-WAKIL, A. E.-B., MOKHTAR BOSHRA. Object-Oriented Design Quality Models A Survey and Comparison. Cairo, Egypt: Faculty of Computers and Information, Cairo University, 2004.

POTENCIER F., F. ZANINOTTO. *Symfony la guía definitiva.* 2007, Disponible en: http://www.librosweb.es.

SCHMULLER J. *Aprendiendo UML en 24 horas.* Mexico: PEARSON EDUCACION, 2000. 448 p. ISBN 968-444-463-X.

Glosario de Términos

Aduana: es la institución de un país encargada de aplicar la legislación aduanera y de recaudar los derechos e impuestos que se aplican a la importación, a la exportación, al movimiento o al almacenaje de mercancías.

Aeronave: todo medio de transporte aéreo, incluidos los de carga, viajeros y recreo.

AJAX: se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes en aplicaciones interactivas para la web. Para ello se utiliza XHTML y CSS (*del inglés, Cascading Style Sheets*) para formatear la información; DOM para interactuar y visualizar dinámicamente la información; se apoya en XML (*del inglés, Extensible Markup Language*) y XSTL para manipular la información mostrada, para la recuperación de datos asincrónica usa el objeto XMLHttpRequest, y JavaScript para actualizar los datos sin necesidad de refrescar la página y para manipular todas estas tecnologías.

Artefacto: pieza de información tangible que es creada, modificada y usada por los trabajadores al realizar actividades.

Buque: toda embarcación que navegue, fondee, se mueva, atraque y desatraque en territorio aduanero, con independencia de su tipo, calado, tamaño o cualquier otra característica.

Cabotaje: es el régimen aduanero aplicable a las mercancías que se cargan en un buque, en un puerto del territorio nacional y se transportan a otro puerto del territorio nacional donde se descargan. Las operaciones de cabotaje se realizan por buques cubanos o sometidos a jurisdicción cubana, habilitados para este tráfico, de forma temporal o permanente.

Carga: es todo bien, mercancía o artículo de cualquier clase transportado en un vehículo, nave, aeronave o ferrocarril, con exclusión del equipaje de los tripulantes, suministros y repuestos para el vehículo.

Carga agrupada: mercancías que amparadas en un sólo conocimiento de embarque o guía aérea, están destinadas a varios consignatarios.

CASE: Computer Aided Software Engineering, que en su traducción al español significa Ingeniería de Software Asistida por Computación, es la aplicación de tecnología informática a las actividades, las

técnicas y las metodologías propias de desarrollo de software, su objetivo es acelerar el proceso de automatizar o apoyar una o más fases del ciclo de vida del desarrollo de sistemas.

CSS: es un lenguaje de hojas de estilos en cascada, creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas.

Clase: descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica.

Consignatario de Mercancías: persona natural o jurídica a quien está destinada un cargamento de mercancías o parte de este.

Declaración de Mercancías: es aquella declaración que se realiza del modo prescrito por la aduana, por la cual las personas interesadas indican qué régimen aduanero pretenden aplicar a las mercancías y suministran los detalles informativos que la Aduana requiere para la aplicación del régimen elegido.

Depósito Temporal de Mercancías: lugar o instalación autorizado por la Aduana para el almacenamiento de mercancías que no hayan sido declaradas, extraídas, reclamadas, formalizadas o que esperen un destino final después de haberse aceptado o declarado su abandono, o cualesquiera otras que la Ley disponga. Puede ser de frontera (portuario o aeroportuario) o interior.

Diagrama: representación gráfica de un conjunto de elementos. Visualizan un sistema desde diferentes perspectivas.

DOM: el DOM es la estructura de objetos que genera el navegador cuando se carga un documento y se puede alterar mediante Javascript para cambiar dinámicamente los contenidos y aspecto de la página.

Especificación de requisitos: captura los requerimientos de software para el sistema completo o una porción del mismo.

Exportación: es la salida de cualquier mercadería de un territorio aduanero.

Framework: un framework, es una estructura de soporte definida mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas,

bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Guía aérea: documento equivalente al Conocimiento de Embarque, utilizado en el transporte aéreo de mercancías, mediante el cual la empresa de aeronavegación reconoce el hecho del embarque de mercancías y expresa las condiciones del transporte convenido.

Importación: las mercancías que provenientes del extranjero, entran en el territorio nacional, aunque estén libres del pago de los derechos de aduanas o gocen de suspensión, exención o franquicia.

JDBC: (*del inglés, Java Database Connectivity*) es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice.

Manifiesto de Carga: lista de mercancías que constituyen la carga de un medio de transporte o de una unidad de transporte. El Manifiesto de Carga expresa los datos comerciales de las mercancías, tales como el número de los documentos de transporte, el nombre del expedidor y del destinatario, las marcas y números, la cuantía y naturaleza de los embalajes y la cantidad y denominación de las mercancías.

Mercancías: todos los bienes corporales muebles de comercio o no, con la sola excepción de los Efectos Personales de los viajeros.

Metodología: es un proceso de software detallado que define con precisión los artefactos, roles y actividades involucradas.

Origen (Régimen de): son normas específicas con capacidad para determinar en qué país fueron producidas o elaboradas cumpliendo determinados requisitos, condición indispensable para beneficiarse de las preferencias otorgadas en el Acuerdo de que se trate.

Rayos X: banda de radiación electromagnética de ondas de longitud intermedia entre la radiación ultravioleta y los rayos gamma.

Régimen Aduanero: dícese del tratamiento aplicable a las mercaderías sometidas al control de la aduana, de acuerdo con las leyes y reglamentos aduaneros, según la naturaleza y objetivos de la operación.

RUP: Proceso Unificado del Rational, es un proceso de desarrollo de software, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

Scripts: conjunto de comandos u órdenes en un fichero que ordenados producen una salida concreta. El uso habitual de los scripts es realizar diversas tareas como combinar componentes, interactuar con el sistema operativo o con el usuario.

Software: se refiere a los programas y datos almacenados en un ordenador.

Transferencia: régimen aduanero bajo el cual son transportadas las mercancías de un depósito a otro de la misma aduana en territorio nacional bajo control aduanero.

Tránsito: régimen aduanero bajo el cual son transportadas las mercancías de una aduana a otra en territorio nacional bajo control aduanero.

UML: es el Lenguaje de Modelado Unificado para detallar, construir, visualizar y documentar las partes o artefactos de un software.

URL: el Localizador Uniforme de Recurso (*del inglés, Uniform Resource Locator*), es el sistema unificado de identificación de recursos en la red el cual es utilizado para especificar un objeto en Internet.

XHTML: (*en español, Lenguaje de Marcado de Hipertexto Extensible*), es una versión más estricta y limpia de HTML, que nace precisamente con el objetivo de reemplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML. XHTML extiende HTML 4.0 combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos.

XML: es la abreviatura de lenguaje de marcas extensible, está diseñado para estructurar, transportar y almacenar información. XML no es un reemplazo de HTML sino un complemento a este.