Universidad de las Ciencias Informáticas
Facultad 3

Título: Diseño del módulo de Selectividad del sistema Gestión Integral de Aduanas.

Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autor: Dayán Trujillo Márquez

Tutor: Ing. Rafael Andrés Céspedes Basteiro

La Habana Junio, 2011

Año 53 de la Revolución.

DECLARACIÓN DE AUTORÍA

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

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

_____ _____

Dayán Trujillo Márquez Ing. Rafael Andrés Céspedes Basteiro

La más hermosa de las victorias, es la que nos cuesta mayor esfuerzo y nos agota hasta el cansancio.

Gualberto Alcántara Olalde

AGRADECIMIENTOS

A todos aquellos que alguna vez encontraron en este servidor una persona en quien confiar o compartir buenos momentos o una mano para atravesar grandes dificultades.

A la gran familia que pude crear en Venezuela porque de ustedes me llevo grandes memorias y cambios de mi vida.

Mayelín, Nadeiny, Miroslava, Michel, Alfredo, Gilber, Idelisa, Cristina, Yanela, Ida Iris, Rebeca, Belén, Danaisy.

Te debo la vida entera por entregarme tan incondicional cariño, por tenderme la mano cuando no te la pedí. Contigo aprendí a tener un hermano de convivencia y afecto. Pase lo que pase, has sido y serás mi sangre.

Yordanky

Por los buenos momentos que compartimos, por las emociones vividas, por estar presente para enseñarme el momento en que crecí como persona y brindarme sin temor alguno tu amistad.

Elvis

A mi siempre querido grupo desde que comenzamos un camino que no conocíamos. Una vez les dije que eran mi familia en esta escuela y lo mantengo hoy día. Todas las etapas por las que he pasado han estado presente para guiarme y eso es grandeza. Tanto los presentes futuros ingenieros como los que buscaron otro camino. En especial a mi equipo de guerra Yasel y Yordani siempre geniales y mi consejero personal Eyeris.

A mis hermanos de contienda, a prueba de balas y a los que no tengo que decirle que los quiero para que sepan que me tienen hasta el último de sus días. Son lo más duradero que he sabido apreciar. La amistad se trata de complicidad sin preguntar por qué, de poner el dedo en la llaga cuando te caes, de reír cuando te sobrevienen desgracias y de perder los momentos de alegría personal para superar tus mayores miedos o desgracias. Ustedes son eso y mucho más. Porque son gran parte de mi historia, la que se ve desde mis ojos y la que no.

Lisett y Eriel

A los que desde que tengo uso de razón me han profesado su amor para hacerme crecer como ser humano, los vecinos de toda la vida por la mano tendida.

AGRADECIMIENTOS

La vida esta llena de cómplices en cada etapa, para esta que es la última como estudiante y en la que se cometen las mayores locuras se la dedico a mi piquete, donde cada uno juega un papel siempre distinto.
Pablo, Ivian Laobel, Ivian Naivis, Raquel

Por tan especial cariño que siempre brindas. Por estar a mi lado en las buenas y malas, y sobre todo por estar donde no pude y emprender mis tareas por encima de tu bienestar. Me quitaré siempre el sobrero ante ti mi hermano.

Yoandy Cervela

Por tenerte cerca cuando creí que la amistad era una palabra por definición y no por acciones. Porque me brindaste momentos que los tendré por siempre, de esos que hacen cambiar tu corazón si eso hace posible verte sonreír mi compinche.

Yoandy Bonilla

Por demostrarme que a todos nos llega esa amistad que refleja como un espejo cual eres, eres como verme a mi mismo en una versión mejorada por eso siempre te llevaré presente.

Yaksel

Mi eterna princesa, por estar desde los primeros días de ingreso hasta la fecha en la que me hago ingeniero. Por la confianza que supe ganarme a golpe de regaños, por esos regaños que siempre me impulsan un poco más, por entenderme en cada una de mis facetas y estados de ánimo y nunca haber perdido su esencia que la hace notar aún cuando crea que nunca ha sido diferente. Por darme la oportunidad de definirla como mi paradigma de lo que espero de una mujer.

Daylin

A mi familia siempre presente en lo bueno y en lo malo, por hacerme sentir orgulloso de tenerlos cerca y de ser bastante numerosos. A cada uno de mis tíos, de mis primos lejanos y de siempre por un cariño que quien no sepa que significa, no ha vivido la vida ni sabrá hacerlo. Por enseñarme que siempre tendremos momentos de duras diferencias pero que el saber que la pérdida deja cicatrices permanentes nos hace reencontrarnos. A todos los que están, y a los que nos encontraremos algún día.
A todos gracias porque forman parte de mi pasado, moldean mi presente, para que tenga un futuro.

Dedicatoria

A mi razón de ser en esta vida, a quienes han sido el pilar de mis acciones y por los cuales puedo decir con total orgullo que son el mejor ejemplo que he recibido en mi paso por el mundo.

A mi mentora y por quien he llegado lo más lejos que mis posibilidades me brindan.

A mi consejero, mi paradigma ante las acciones que como hombre debo acometer.

Pocos tienen la oportunidad de decir con satisfacción, yo tengo MAMÁ y PAPÁ... y yo soy uno de ellos.

Necesito su gracia, Para recordarme a mí, Para hallar mi esencia
Esas dos palabras que se dicen demasiado, pero no lo suficiente

Rosidalia y Eduardo todo su esfuerzo se resume en esas palabras, Los amo.

RESUMEN

Los controles aplicados a las personas y mercancías que cruzan los diferentes puntos fronterizos del país, son cada vez más exhaustivos, en correspondencia con la modernización de la tecnología y las vías de cometer la gran variedad de infracciones existentes. El proceso de Selectividad como módulo de la Lucha contra el Fraude de la Aduana General de la República de Cuba, distribuye los recursos materiales y humanos hacia los puntos más vulnerables y con la mayor probabilidad de acierto en materia de actividades no permitidas, que son monitorizadas a través de controles especializados según su gravedad.

Existe implementada y desplegada una aplicación que cubre los propósitos que requiere este proceso en la AGR, pero es necesario reimplementar la solución para incorporarle mejoras que permitan la obtención de un producto adaptable ante cambios estructurales. El presente trabajo tiene como objetivo obtener el diseño del módulo Selectividad para el Sistema Gestión Integral de Aduanas para facilitar el trabajo de control realizado sobre las personas y productos por parte del personal correspondiente. Para realizar el diseño de la solución se utilizó el análisis que se obtuvo de los procesos involucrados, que muestra la arquitectura que se debe satisfacer. A través de la metodología RUP con algunas adecuaciones se realizó el diagramado del diseño y todos los artefactos, utilizando el UML mediante la herramienta Visual Paradigm.

Con el diseño obtenido será posible erradicar los problemas existentes en la solución anterior, disponiéndose de un software para aplicar la Selectividad, compatible ante cambios futuros.

PALABRAS CLAVES

Selectividad, Diseño

ÍNDICE

TABLA DE CONTENIDOS

INTRODUCCIÓN

Desde los inicios de las relaciones que han ido en aumento entre las diferentes naciones existentes en el mundo la necesidad de controlarlos fue una premisa necesaria para evitar que cada uno de los procesos se realizara en determinadas maneras que no fueran del todo correctas ni equitativas. Los controles son parte importante de cada intercambio en el cual intervengan personas, objetos comerciales o patrimoniales que representen un peligro o un bien a preservar.

La pasada década de los años noventa propiciaría un giro total al tipo de relaciones comerciales, con la nueva tendencia de la economía neoliberal y las nuevas posibilidades de intercambio aún mayor. El año 2001 sería el inicio de una nueva etapa totalmente marcada por el control total aduanero con la caída del Wall Street Center y la cruzada contra el terrorismo. Ambos puntos resaltaron la creciente importancia de las aduanas en cada frontera mundial, siendo este en ese momento una red enorme de comunicación.

Con el paso del tiempo, principalmente en los últimos años el desarrollo alcanzado por las diferentes naciones llevó a un alza considerable de intercambio tanto de personal como mercantil que obligó radicalmente a establecer medidas superiores de control para preservar los intereses nacionales correspondientes. La tecnología también ha jugado un papel primordial en este sentido cambiando a muchos de los paradigmas anteriores establecidos para aplicar este tipo de controles necesarios, de otra manera quedarían obsoletas e incapaces de contribuir eficientemente con su principal razón de existencia.

La Aduana General de la República (AGR) se ha visto inmersa en estos nuevos tiempos y se ha planteado el reto de alcanzar los requerimientos mínimos que deben ser gestionados en busca de efectuar los controles adecuados y que en todo momento sean precisos. Con este objetivo, al cual centraron toda su atención, se tomaron la tarea de planificarse y establecer un cronograma de trabajo ajustado con la intención de aceptar y aplicar estos cambios en sus diferentes procesos y subsistemas para mantener los niveles de certeza en sus funcionalidades lo más alto posible y con ello garantizar la calidad de sus diferentes funciones. En este sentido uno de sus principales procesos internos es la selectividad, aplicada a cada entrada realizada por las vías establecidas y conocidas a lo largo de toda la isla, que gestiona todo lo relacionado con el control a las posibles infracciones cometidas o a cometer según determinadas condiciones y criterios que han sido estandarizados.

INTRODUCCIÓN

La Aduana General de la República de Cuba es el organismo encargado de velar por los intereses nacionales y evitar a través de sus diferentes mecanismos la violación de la Seguridad Nacional tanto en sus exportaciones como sus importaciones. Para este propósito es que resulta creada el área de Lucha Contra el Fraude (LCF) como la primera línea de defensa eficaz, como se declara en el Decreto Ley No. 162 de Aduanas, en su artículo 16, inciso c) que establece entre las atribuciones y funciones especiales de la Aduana, la de prevenir, detectar y enfrentar el fraude comercial, el contrabando y otras infracciones aduaneras en el desarrollo del tráfico comercial de mercancías, viajeros y envíos por vía aérea, marítima y postal.

Se hace necesario encontrar la vía factible que relacione tanto la tecnología como las relaciones comerciales para la Aduana de Cuba, es por eso que en el año 2008 se genera un pequeño sistema que aplicaba la selectividad requerida por el subsistema de Lucha Contra el Fraude a los distintos puntos fronterizos existentes en ese entonces.

Las necesidades fueron en crecimiento paulatino y lo que en un comienzo resultaba eficaz comenzó a parecer obsoleto e impreciso por la magnitud de la información que se controla y la cantidad de inspecciones necesarias. La selectividad solo podía ser aplicable a los objetos, declaración de mercancías y rayos x, que resultaban los necesarios controlar en su momento. La aplicación comenzó a ser inapropiada debido que no era capaz de abarcar los requerimientos mínimos estipulados bajo las nuevas condiciones. De esta manera siempre que se aplicara un control a un nuevo objeto requería generar una selectividad particularizada y propia generando demasiadas tareas para su implementación. Los especialistas correspondientes a la línea LCF del Centro de Automatización para la Dirección y la Información del AGR (CADI) requerían nuevas funcionalidades y un trabajo mucho más profundo y perfeccionado que la operabilidad de la aplicación usada no brindaba, unido al tipo de herramientas utilizadas que ya no resultaban viables para el tipo de sistema que se quería implantar.

La gestión Integral de Aduanas (GINA) surge entonces como la solución recomendada para los nuevos lineamientos que se plantean lograr, integrando todos los módulos reestructurados. Con esta nueva solución dentro de la Lucha Contra el Fraude se necesita modificar la manera en que hasta el momento se

ha dado utilidad a los procesos de selectividad que desde el año 2008 se informatizaron para integrar los procesos vinculados a esta actividad.

La selectividad bajo las nuevas determinaciones requiere de que este tipo de control se haga posible en las dos variantes existentes, manual y automática, así tanto una persona como el propio sistema estaría habilitado para brindar un respuesta certera basada en criterios, condiciones y valores anteriormente evaluados por las personas capacitadas. Tendría que incorporar nuevas funcionalidades que la propia práctica determinó como obligatoria para que los procesos relacionados alcanzaran altos niveles de efectividad. Todas estas propiedades conducirían a lograr en resumen, delimitar una selectividad genérica capaz de poder ser aplicable a cualquier objeto que los especialistas de LCF determinen.

Con estos requerimientos delimitados se procede a generar un análisis sobre cómo se trabaja la selectividad y entender la nueva lógica del negocio y el sistema para poder encaminar lo que será posteriormente el resultado visual de lo pedido. Análisis que ya ha sido aprobado y evaluado por las autoridades correspondientes. Este análisis realizado demuestra que existe la necesidad de reestructurar la arquitectura utilizada hasta la actualidad para que pueda cumplir con los requisitos levantados con lo que es preciso redefinir todas sus interfaces y cambiar las relaciones de sus clases para erradicar la problemática. Significa encontrar el diseño adecuado que cumpla con las nuevas regulaciones de desarrollo y que satisfaga a la arquitectura que a través del análisis se pudo definir.

Sería obligatorio entonces pasar a la siguiente fase planteada por RUP y ofrecer un nuevo diseño ajustado a lo revisado en las fases de negocio, requerimiento y análisis que cumpla con los resultados de aplicar las fases anteriores para avanzar con el cronograma predeterminado, lo que lleva a plantear el siguiente **Problema a resolver:** ¿Qué diseño de software satisface los requisitos y permite una correcta implementación en el módulo Selectividad del Sistema Gestión Integral de Aduanas?

Centrándose en el **Objeto de estudio:** Los procesos relacionados con la selección de objetos a controlar en la Aduana General de la República de Cuba. Enmarcados en el **Campo de acción:** El diseño de los procesos relacionados con la selección de objetos a controlar en la Aduana General de la República de Cuba. Alcanzando el **Objetivo general:** Obtener el diseño que responda a las necesidades de

información y permite una correcta implementación en los procesos de Selectividad de la Aduana General de la República de Cuba. Con lo cual se hace necesario desarrollar las siguientes tareas.

Tareas a cumplir por el estudiante:

1. Analizar el estado del arte de los sistemas selectores en el Mundo y dentro de Cuba.
2. Analizar las herramientas y técnicas que apoyan el desarrollo de una solución para el problema planteado.
3. Realizar un análisis de los procesos de negocios descritos en el análisis de selectividad.
4. Asimilar la arquitectura definida para la solución Gestión Integral de Aduanas en el desarrollo de sus aplicaciones.
5. Fundamentar el uso de metodologías, tecnologías y herramientas de desarrollo de software a utilizar.
6. Definir un diseño con las funcionalidades que deben estar presentes en el sistema.
7. Obtener el modelo de diseño de la solución Selectividad para el sistema Gestión Integral de Aduanas.
8. Validar técnica y funcionalmente el diseño obtenido.

Se desea a través de esta investigación:

Si se realiza un diseño adecuado para dar cumplimiento a los requisitos capturados de software, se garantiza obtener un diseño funcional y la correcta implementación del módulo Selectividad de la Aduana General de la República de Cuba.

Métodos Científicos:

Método Histórico – Lógico

Métodos históricos: Analiza todo el trayecto por el cual atraviesa el objeto a estudiar. De esta manera es posible conocer sus etapas y reacción ante los períodos de la historia.

Métodos lógicos: Estudia internamente la lógica de su desarrollo desde el punto de vista histórico para obtener el conocimiento primario en el cual se haya su teoría. Básicamente se manifiestan en forma teórica así es posible relacionar la estructura que ha adquirido el objeto a través de la historia. Lo lógico no repite lo histórico en todos sus detalles, se enfoca solo en lo importante del fenómeno.

La utilización de ambos métodos resulta en una relación necesaria para encontrar resultados que no se basen en lo especulativo y muestre bases de razonamiento previo en la investigación. Para esto el método lógico debe enfocarse en lo que se revele en el método histórico.

Método Analítico – Sintético

Se trata de analizar toda la documentación bibliográfica como el contexto donde se desarrolla, para poder descomponer en partes pequeñas a las cuales poder realizarle una selección adecuada de lo que se necesita y elaborar las ideas correspondientes que tributen a todo el objeto de estudio. De esta manera es práctico dividir mentalmente todo el contenido, analizarlo y posteriormente de manera analítica integrarlo

Método Modelación

La modelación es el método mediante el cual se crean abstracciones con el objetivo de explicar la realidad. Es un modelo que recrea al objeto de estudio por lo que genera una correspondencia objetiva entre ambos elementos. Resultando una vía de investigar la realidad donde se encuentra el problema.

Dado que cada problema resulta único en su concepción, las maneras de modelarlo cambian para una mejor comprensión.

Modelo analógico

Este tipo de modelado solo se enfoca en la estructura de las relaciones y determinadas propiedades fundamentales de toda la realidad. Se establece una analogía entre el sistema real y el modelo, y se estudia el primero, utilizando como medio auxiliar el segundo.

La estructura del trabajo de Diploma está desglosada en los siguientes capítulos:

Capítulo 1 Fundamentación Teórica: Aborda la investigación previa que debe realizarse para conocer exactamente donde se encuentra ubicada la solución que se desea implementar. Incorpora una serie de propiedades generadas por las herramientas o aplicaciones encontradas en la universidad y utilizadas en el mundo para desarrollar soluciones informáticas o computacionales. Es la manera más racional de explicar con qué se dispone para la evolución del producto y demostrar por qué ha sido necesario conducir en un determinado sentido el trabajo.

Capítulo 2 Diseño de la solución propuesta: Muestra el resultado de aplicar los diferentes patrones de diseños analizados para ser utilizados en busca del diseño indicado en conjunto con los diferentes diagramas necesarios a utilizar para documentar a través del modelado como quedará la solución propuesta.

Capítulo 3 Validación de la solución propuesta: Desarrollo de diferentes medidas que comprueban que se ha alcanzado un diseño proporcional y ha sido aprobado para su aplicación final. Medidas resultantes de aplicar métricas e indicadores evaluados con anterioridad para conocer si el resultado ha terminado como positivo.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se elabora un acercamiento a la utilización de la selectividad de blancos[1] como proceso aduanero a través de las distintas aplicaciones desarrolladas para acometer estos propósitos mediante la descripción de estos sistemas informáticos y las peculiaridades en el tratamiento de los elementos de control. Se detallarán aspectos importantes de la metodología, lenguaje y programas que se utilizarán para el desarrollo del producto y se analizarán los patrones de diseño más factibles de utilizar.

El área de Lucha Contra el Fraude de la Aduana General de la República de Cuba hoy día cuenta con gran información dinámica, donde los especialistas deben tomar decisiones de forma rápida y efectiva basándose en la última información disponible. Esta información debe ser tratada diferenciadamente para lograr aplicar efectivamente los controles necesarios y mínimos a lo que resulte verdaderamente un riesgo potencial con altas probabilidades de ocurrencia. Los sistemas selectivos brindan las condiciones necesarias para satisfacer esta petición pero debe ofrecer una visión relativamente diferente de otras aplicaciones con iguales responsabilidades.

Internacionalmente se ha establecido una serie de normas que condicionan cómo se debe procesar ante una determinada situación de riesgo. Pero cada riesgo resulta diferente y por lo tanto no posee ni la misma intensidad ni la misma atención a la hora de aplicar los distintos tipos de controles según el canal asignado. Canales que varían de color según el grado de procesamiento que requiera y tener a priori conocimiento anticipado, si es importante o carece de ella el chequeo de un elemento a requisar. Verde para la liberación de las mercaderías sin examen; naranja para el control de la documentación antes de la liberación de las mercaderías; rojo para el examen físico de las mercaderías antes de la liberación; azul, para indicar que las mercaderías serán liberadas pero quedarán sujetas a un control de auditoría post-despacho por parte de la aduana. Esta manera organizada de trabajo aumenta la efectividad en el chequeo de los riesgos y viabiliza los trámites involucrados para acometer esta tarea.

[1] **Blancos:** Se refiere a un término aduanero, que se utiliza para denominar a las personas que son posibles infractores de acuerdo a un análisis previo.

El comercio moderno no puede permitirse interrupciones en las cadenas productivas y los servicios aduaneros no cuentan con los recursos como para controlar cada envío. Con el crecimiento del volumen de las transacciones del comercio internacional, la inspección física y en profundidad rápidamente está pasando a ser historia. (1)

1.1. Definiciones establecidas sobre Selectividad.

1.1.1. Selectividad Aduanera.

Es un proceso dirigido a establecer determinados controles sobre las diferentes mercancías que se reciben y las personas internas o externas que cruzan por los puntos aduaneros establecidos en un determinado país. Mediante el uso de técnicas de inteligencia se trata de obtener un resultado automático que muestre si es necesario o no, requisar a la selección de blancos. Estas técnicas son adquiridas a través de todos los procesos relacionados dentro de la aduana con la manipulación del contenido que se recibe o se envía desde y hacia un determinado lugar, garantizando que no se puedan cometer infracciones que atentan contra la seguridad.

La Gestión de Riesgos en el contexto aduanero es la aplicación sistemática de prácticas y procedimientos que proporciona a la Administración informaciones valiosas, empleadas para segregar el tráfico de carga que presenten características de riesgos determinadas, partiendo de renglones o áreas sensibles, expuestas a acciones ilícitas. (2)

1.1.2. Análisis de Riesgo.

Es una terminología utilizada mayoritariamente para inferir la gestión que antecede a la aplicación de la selectividad. Se basa en criterios de selección múltiples empleados para determinar donde asignar correctamente los recursos humanos y materiales disponibles en la búsqueda de elementos que no garantizan en un alto grado la oportunidad de entrada al sistema nacional del país que lo aplique. La recolección de datos es un paso importante para con las herramientas y aplicando la metodología adecuada obtener resultados factibles y es posible centralizar en un método efectivo todos los elementos que conforman el trabajo aduanero en la frontera.

1.1.3. Selectividad en Cuba.

La selectividad en Cuba ha tenido un tratamiento diferente por las propias características independientes del resto de las aduanas que posee esta actividad. En principio sigue siendo un proceso para realizar controles diferenciados en las fronteras que corresponden, pero basados en principios que difieren de los aplicados en otras aduanas a nivel mundial. Debido a la estructuración por departamentos internos es preciso establecer criterios que se basan en determinadas condiciones argumentadas según la necesidad y los controles que deben aplicarse a cada objeto de control dígase personas, mercancías, productos o medios de transporte internacional entre otros. Esto posibilita que como resultado se logre obtener un producto genérico informatizado, que es posible aplicar ante cualquier cambio organizacional y donde no es necesario particularizar los controles aplicados.

1.2. Aplicaciones informáticas que aplican Selectividad o Análisis de riesgo en las Aduanas.

El desarrollo de las aduanas con el crecimiento comercial y el aumento de los controles realizados en la búsqueda de elementos no autorizados, se ha encaminado en la perfección de los procesos que intervienen en la selección de blancos. La dinámica actual necesita que el tratamiento entre el personal aduanero y los elementos objetos de control, se realicen de forma rápida, poco invasivo pero sin perder la calidad y con mayor efectividad en los tipos de controles a aplicar. Al mismo tiempo estos controles deben ser más precisos y estar delimitados de acuerdo a la peligrosidad o la potencialidad de riesgo que pueda tener determinado elemento analizado. Esta necesidad propició la informatización de los procesos aduaneros en la creación de una aplicación para tratar la selectividad adecuadamente o la incorporación de un módulo a la solución ya establecida para atender correctamente esta funcionalidad.

1.2.1. Aplicaciones Extranjeras.

1.2.1.1. Sistema Informático María (S.I.M)

Es una aplicación informática de la aduana de Argentina que está compuesta por aplicaciones independientes internas que tributan al sistema María, pero que tratan a la información según sus características propias. Tiene incorporado un sistema de selectividad que es capaz de obtener un diseño de una selectividad inteligente y desprovista de discrecionalidad.

1.2.1.2. Sistema Piloto de Evaluación de riesgos

Es un sistema implementado por la administración de Boca Chica para encontrar una solución que sea aplicable a las aduanas. Es un sistema que clasifica su información en áreas según el tipo de seguridad que esté tratando. De esta manera puede clasificar por prioridades de alerta. Se basa principalmente en el cumplimiento de las exigencias de quien utiliza la solución lo que imposibilita obtener un diseño de selectividad inteligente.

1.2.1.3. Sistema Aduanero Automatizado (SIDUNEA)

Aplicado como sistema informático en las aduanas de los países que forman parte de las Naciones Unidas que lo soliciten. Útil para la gestión aduanera total. Posibilita a través de sus procesos desarrollar la ejecución de la selectividad a la selección de blancos correspondientes. Se puede aplicar en cualquier tipo de régimen por su facilidad de adaptación a las características nacionales y sus respectivos procesos aduaneros.

Se rige por las reglas definidas para el procesamiento de los datos establecidos por la Organización Mundial de Aduanas (OMA) y la Organización Internacional para la Estandarización (OIE). Utiliza la tecnología JAVA en su totalidad para el control de acceso al sistema. Se considera una aplicación WEB a la cual se puede acceder desde cualquier dispositivo inalámbrico, por lo que se reconoce su arquitectura de Cliente - Servidor. Con esta arquitectura en la parte servidor tiene una configuración Ethernet para evitar la demora en los servicios ante una demanda excesiva por parte de los usuarios.

Es independiente de las plataformas y de los Sistemas de Bases de Datos utilizados. Patentado por un fuerte sistema de Seguridad de Clave privada a través de la infraestructura de llave Privada (PKI). Se logra una interface de usuario amigable.
La última edición, una verdadera versión de aduana electrónica, es ASYCUDAWorld, introducida en 2004, que es compatible con los principales sistemas operativos y de administración de bases de datos (Oracle, DB2, Sybase MS/Windows, Linux, HP-UX).

El uso del lenguaje de marcas extensible (XML) permite el intercambio de cualquier documento entre las administraciones aduaneras y los comerciantes, a nivel nacional, y de las administraciones aduaneras entre sí, vía Internet y a nivel internacional. Una característica interesante de esta versión para los países en desarrollo que cuentan con telecomunicaciones no confiables, o que directamente carecen de ellas, es el hecho de que no se requiere una conexión permanente a un servidor nacional. Mediante el uso de instalaciones como VSAT, el acceso independiente por Internet de alta velocidad entre la aduana central y todos los controles fronterizos puede estar asegurado. 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.

El ***MODSEL*** o Módulo de Selectividad del sistema SIDUNEA permite a la Aduana controlar la selección y el flujo de las declaraciones a través del sistema de procesamiento de la declaración de aduana. Contiene los controles para bloquear la liquidación de las declaraciones seleccionadas y tiene muchas funciones para hacer consultas e imprimir reportes. Para poder aplicar efectivamente la selectividad se requiere de las habilidades de un equipo especializado. Los usuarios del sistema en la oficina de la Aduana tienen un acceso a bajo nivel que les permite ver las declaraciones seleccionadas y el respectivo criterio de selección y la posibilidad de redirigir las declaraciones después de que se han llevado a cabo algunas verificaciones por la Aduana. (3)

Los controles del módulo de selectividad *MODSEL* permiten seleccionar las declaraciones con dos métodos. El primero, Selección Aleatoria descansa en un número al azar generado por el sistema. Con

este método las declaraciones se seleccionan para ser revisadas puramente sobre la base al azar. La selección aleatoria es extremadamente útil particularmente para propósitos de control de calidad.

La Aduana escoge usar la selección como una estrategia particular de administración que ayuda a interceptar transacciones de importación o exportación de alto riesgo sin retrasar el movimiento de los otros bienes. El uso efectivo de la selectividad requiere de personal con habilidades especiales y con buenos métodos para reunir, registrar y analizar la inteligencia de la carga. La calidad y efectividad de la selección de Aduana descansa en el acceso y uso de la información.

El proceso de selectividad requiere primero que la Aduana decida qué bienes o transacciones se interceptarán. El especialista en selectividad traduce la decisión en un formato que el sistema comprende. Esto se describe como criterio de selección o perfil de selección.

Se trata de un módulo destinado a tratar el análisis de riesgo y la selectividad en la Aduana. Pensado para incorporarse al sistema existente que se encuentra operando.

1.2.1.4. *Sistema* Integrado Financiero y Administrativo (Sistema SOFIA)

Es un sistema informático de despacho aduanero que interactúa en forma directa con sus usuarios. El Sistema de Información se trata de la construcción de un conjunto operacional que toma como base y punto de partida el Sistema SOFI (Sistema de Computación para el Flete Internacional) francés, que va incorporando nuevas funcionalidades para satisfacer las necesidades de gestión de la Aduana Paraguaya, conforme a la evolución impuesta por el comercio globalizado.

Objetivos.

- Mejoramiento de la eficacia administrativa y capacidad de gestión de la Dirección Nacional de Aduanas.
- Información para la Lucha contra el Fraude.

Principios.

- Procesamiento descentralizado.
- Descentralización en el ingreso de la información.
- Descentralización del procedimiento Aduanero.
- Centralización de los datos considerados de interés estadístico en un servidor central.
- Usuarios (despachantes, depositarios, transportistas, aduaneros) conectados directamente al sistema.

Funcionamiento.

- SOFIA permite la conexión de los Despachantes de Aduana o de los Agentes de Transporte para la formulación de sus Despachos de Importación/Exportación o Manifiestos desde sus propias oficinas.
- Existe también la posibilidad de realizar las operaciones desde los Centros Públicos habilitados a tal efecto.

Utilidad de las aplicaciones extranjeras.

Todas estas aplicaciones tienen en común la incorporación de la Selectividad como parte de sus procesos pero cada una se encuentra en distintas facetas de su utilización. Para algunas es mucho más importante la documentación que se efectúa y otras manejan a este proceso como una aplicación pero pobremente ejecutada o incapaz de ser aplicables a las especificidades de la aduana cubana, completamente distintas del resto del mundo.

1.2.2. Aplicaciones cubanas

1.2.2.1. Sistema Único de Aduana (SUA)

Es la solución informática que ha sido desarrollada para prestar servicios en las aduanas cubanas. Desarrollado como una aplicación Web, incorpora una serie de herramientas en su mayoría libres para la construcción de la solución. Capaz de fraccionarse en módulos destinados a diferentes despachos aduaneros, los cuales presentan sus propias características.

Dentro de SUA se encuentra el módulo LCF, el cual tiene incorporado la selectividad como parte de sus procesos. Su mayor inconveniente es la ineficiencia operática y de tiempo en caso de una restructuración interna de la aduana de Cuba. Actualmente se utiliza para la arquitectura establecida pero ante los cambios estructurales para obtener un rendimiento adecuado carece de precisión como aplicación informática. Maneja los principios y estándares actuales en las tendencias de las aplicaciones a soluciones Web.

1.3. Buenas prácticas para el Diseño

La mayoría de los software fallan. Alrededor del 80% de los proyectos son insatisfactorios ya sea porque están sobre cargados, retrasados, contienen funciones perdidas o la combinación de ellos. Además el 30% de las aplicaciones informáticas son tan pobremente ejecutadas que deben ser canceladas antes de su terminación. Las aplicaciones actuales desarrolladas con herramientas potentes no son la excepción. (4)

Incluso con una buena arquitectura todavía es posible tener un mal diseño. Muchas aplicaciones son o bien sobre diseñadas o poco diseñadas. Los dos principios básicos son *Keep it Simple*[2] y *information hiding*[3]. Para muchos proyectos es importante desarrollar un análisis y diseño orientado a objeto usando UML como se plantea en el libro UML *User Guide*. Reusar es una de las grandes promesas del orientado a objetos, pero es a menudo irrealizable debido los esfuerzos adicionales requeridos para crear capital reusable. El reuso de código es una de las formas de utilización dentro de las existentes para proveer mejor ganancia productiva.

Para mejorar enormemente el éxito de cualquier proyecto de desarrollo de software, es útil seguir una guía con algunas características para un buen diseño.

Contratar a personas calificadas y con experiencia: el entorno actual es más complejo que nunca. Las herramientas ayudan, pero al final, las personas sin experiencia producen resultados mediocres en el

[2] **Keep it simple**: Una expresión denominada KISS keep it simple, stupid para referirse a la premisa de que la simplicidad debe ser la clave para el diseño y se debe evitar la complejidad.

[3] **Information hiding:** Es un principio de segregación de las decisiones del diseño en un programa computacional. Esta medida protege otras partes del programa de excesiva modificación.

mejor ejemplo y, en la mayoría de los casos, fracasan, porque no entienden la administración de un buen proyecto y las mejores maneras de aplicar las nuevas tecnologías. Un excelente gestor de proyectos y líderes como el arquitecto o el técnico se encargará de dirigir conjuntamente el proyecto. Ellos marcan la pauta y tienen un impacto enorme en el éxito final.

Utilizar el proceso de desarrollo adecuado: la naturaleza de los proyectos de software modernos exige un proceso de desarrollo basado en espiral. Un proceso en espiral tiene múltiples fases que, sucesivamente, disminuyen el riesgo del proyecto. Al final de cada fase es una decisión de ir o no ir. En las primeras fases, la creación de prototipos se utiliza para explorar las nuevas tecnologías para el equipo o una interfaz de usuario.

Proporcionar las herramientas adecuadas: un proyecto de software necesita las herramientas adecuadas que proporcionan ayuda a la productividad para el equipo. Las herramientas incluyen el hardware adecuado, así como ayudas para el diseño, programación y prueba de productividad. La justificación de los costes de estas herramientas es relativamente fácil. Formación en las nuevas herramientas o técnicas es también esencial para asegurar que se utilizan para su provecho.

Entender que la única constante es el cambio: estos cambios ocurren por muchas razones, como alguien que al no hacer las preguntas correctas en el momento adecuado, el problema a resolver ha cambiado, los usuarios cambiaron de opinión o la percepción, el entorno empresarial ha cambiado o que el mercado ha cambiado. "Invasión de características" es la fuente más común de los excesos de costes y de calendario. En las primeras etapas de un proyecto, hay una gran cantidad de cambios en los requisitos. En algún momento, los requisitos deben ser resueltos y cerrados, en esencia.

1.4. Patrones para el desarrollo de la aplicación

1.4.1. Patrones de arquitectura y diseño

Los patrones arquitectónicos expresan una organización estructural fundamental o esquema para los sistemas de software. Proporcionan un conjunto de subsistemas predefinidos, especifica sus responsabilidades, e incluye las normas y directrices para la organización de las relaciones entre ellos. Es

por ello, que el logro del mejoramiento y la calidad del diseño de un sistema, depende en gran medida de la utilización de patrones que proporcionen esquemas para refinar subsistemas o componentes del mismo.

Un patrón de diseño expresa esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas automatizados, es una descripción de clases y objetos comunicándose entre sí para resolver un problema de diseño general en un contexto particular. Identifica clases, instancias, roles, colaboraciones y la distribución de responsabilidades. Es una solución estándar para un problema común de programación, una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios, un proyecto o estructura de implementación que logra una finalidad determinada o de manera general una vía más práctica de describir ciertos aspectos de la organización de un programa.

Los patrones en los últimos años se han convertido en solución por excelencia, constituyéndose como una importante técnica para construir software orientado a objetos. El valor principal de estos se encuentra en el aprovechamiento de la experiencia de los expertos que los han identificado y documentado.

El conocimiento de los patrones de software facilita la identificación de determinados problemas a resolver y a buscar una solución rápida y elegante; son una herramienta la cual, basándose en experiencias anteriores, resuelve ciertos problemas que son frecuentes; permitiendo que los sistemas sean mucho más adaptables al cambio y logrando la reusabilidad de componentes que presentan características similares. Surgen en base de la Arquitectura en donde se presentan una serie de problemas de forma recurrente y la forma de solución es similar.

Varios autores han listado las cualidades que son deseables en un patrón, dependiendo del contexto y su capacidad de adaptabilidad al problema que debe resolver, como es el caso de las listadas en el libro "*Christopher Alexander: an Introduction for Object-Oriented Designers*" de Doug Lea:

• **Encapsulamiento y abstracción**: cada patrón encapsula un problema bien definido y su solución en un dominio particular. Los patrones deberían proporcionar límites claros que ayuden a cristalizar el entorno del problema y el entorno de la solución.

• **Extensión y variabilidad**: cada patrón debería ser abierto por extensión de tal forma que pueden aplicarse junto con otros patrones para solucionar problemas de mayor complejidad. Un patrón debería ser también capaz de realizar una variedad infinita de implementaciones.

• **Generatividad y composición:** cada patrón, una vez aplicado, genera un contexto resultante, el cual concuerda con el contexto inicial de uno o más de uno de los patrones del lenguaje. Está secuencia de patrones podría luego ser aplicada progresivamente para conseguir la generación de una solución completa. Los patrones se encuentran jerárquicamente relacionados, la mayoría admiten tanto una composición ascendente como una descendente dentro de la jerarquía. Cada uno de ellos debe ser capaz de trasmitir la idea contextual pero no debe estar determinado solamente para ser usado por diseñadores, debe ser entendible para quienes no son diseñadores.

• **Equilibrio:** cada patrón debe realizar algún tipo de balance entre sus efectos y restricciones, para brindar una razón que permita rastrear cada paso del diseño.

Entre las diversas categorías en los que se clasifican los patrones de software basándose en sus niveles de abstracción según Riehle y Züllighoven se establece el de diseño:

Diseño: patrones cuya forma es descrita usando diseño de software construidos como objetos, clases o módulos. Tales patrones facilitan soluciones a problemas de diseño en general en un contexto en particular.

Patrones de Diseño.

Un patrón de este tipo identifica, abstrae y nombra los aspectos elementales de una estructura de diseño, donde los componentes, son las clases y objetos, y sus mecanismos de interacción son mensajes. (5)

Cada patrón prescribe una estructura de clases, sus roles y colaboraciones, y una adecuada asignación de métodos para resolver un problema de diseño en una manera flexible y adaptable. La aplicación de los patrones en un diseño consiste en identificar el patrón que resuelve el problema de diseño encontrado y aplicar la solución abstracta prescrita por el patrón a dicho problema. Los patrones de diseño ayudan a elegir diseños alternativos que hacen un sistema reutilizable y evitan alternativas que comprometan la

reutilización. El libro que ha popularizado los patrones de diseño es el conocido "*Design Patterns: Elements of Reusable Object Oriented Software*".

Abstract Factory

Proporciona una interfaz para crear familias de objetos relacionados sin especificar sus clases concretas.

Adapter

Convierte la interfaz de una clase en otra distinta, que es la que esperan los clientes.

Builder

Separa la construcción de un objeto complejo de su representación.

Decorator

Añade dinámicamente nuevas responsabilidades a un objeto. Alternativa de la herencia.

Facade

Proporciona una interfaz unificada para un conjunto de interfaz de un subsistema.

Factory Method

Define una interfaz para crear un objeto, pero deja que sean las subclases las que decidan que clase instanciar.

Iterator

Proporciona un modo de acceder secuencialmente a los elementos de un objeto agregado sin exponer su representación interna.

Observer

Defina una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambia de estado, se notifican y se actualizan automáticamente todos los objetos que dependen de él.

Singleton

Garantiza que una clase solo tenga una instancia y proporciona un punto de acceso global a la misma.

State

Permite que un objeto modifique su comportamiento cada vez que cambia su estado interno.

Patrones GoF.

Existen otras clasificaciones que se encuentran en el medio de las antes descritas, como son los patrones GoF (acrónimo de Gang of Four en español banda de los cuatro). Estos patrones tienen sus tres propias clasificaciones:

Creación: estos patrones se centran en la creación de objetos, como son las clases u otros objetos.
Estructurales: patrones que utilizan herencia para componer una clase o la manera específica de ensamblar un objeto.
Comportamiento: usan la herencia para describir algoritmos y flujos de control o para describir como un grupo de objetos pueden cooperar entre sí para lograr mejorar la realización de una tarea que un objeto solo no podría realizar.

De acuerdo a su ***ámbito*** se clasifican en:

- **Clases:** relaciones estáticas entre clases.
- **Objetos:** relaciones dinámicas entre objetos.

GRASP.

Los patrones GRASP acrónimo de General Responsibility Assignment Software Patterns (patrones generales de software para asignar responsabilidades), describen los principios fundamentales de la asignación de responsabilidades a objetos, expresadas en forma de patrones. (6)

Definición de métrica.

Cada diseño siempre debe ser respaldado que certifique que se ha empleado correctamente o que la aplicación de un determinado patrón fue correcta por eso es necesario aplicarle métricas que dan una

medida de lo que se ha estado haciendo con estos patrones y permiten guiar mejor los esfuerzos de quienes lo utilizan.

Métrica: Es cualquier medida o conjunto de medidas destinadas a conocer o estimar el tamaño u otra característica de un software o un sistema de información, generalmente para realizar comparativas o para la planificación de proyectos de desarrollo. (7) Las métricas se crean en categorías que reflejan características de diseño importantes. Es un instrumento que cuantifica un criterio.

Los objetivos principales de las métricas orientadas a objetos son los mismos que los existentes para las métricas surgidas para el software estructurado:

- Comprender mejor la calidad del producto
- Estimar la efectividad del proceso
- Mejorar la calidad del trabajo realizado en el nivel del proyecto.

1.5 Definición de artefactos usados en la fase de Diseño.

La utilización de los artefactos que se generan en la fase de diseño, resulta una actividad esencial para el desarrollo de cualquier sistema informático donde sea importante que se documente el tratamiento que se realiza a los procesos que se desean describir para obtener la arquitectura de la solución y proceder a la implementación. Cada proyecto debe gestionar la manera en que debe proceder para la ejecución y generación de aquellos artefactos, que decidan, son importantes en la concepción de su planificación.
Estableciendo una comparación entre algunas soluciones del centro para la Informatización de la Gestión de Entidades y el Centro de Gobierno Electrónico se puede evidenciar la similitud existente entre los artefactos que se utilizan en el proyecto aduana y el resto, como garantía de que no existen grandes discrepancias, ni irregularidades en la manera de tratar el proceso de desarrollo de una determinada metodología. Conociendo qué se debe obtener en esta fase, permite saber si se ha garantizado su elaboración para el modulo de Selectividad del GINA.

Centro de Gobierno electrónico

Digitalización alfabética de Proyecto Identidad

- Modelo de Datos.
- Documento arquitectura.
- Modelo del Diseño.

VIRTEVALL

- Modelo de Datos.
- Diagrama de Clases del Diseño.

Centro para la informatización de la Gestión de Entidades

Línea desarrollo de Productos

- Modelo de Diseño.
- Clases del Diseño.
- Diagrama de Secuencia.
- Diagrama de despliegue.
- Diagrama de Paquetes.

Soluciones Bancarias

- Modelo de datos.
- Diagrama de paquetes.
- Diagrama de Secuencia.
- Diagrama de Clases.

Línea Planificación – ERP

- Diagrama de Clases.
- Mapa de Procesos.
- Diagrama de Proceso de Negocio.

Tomando en consideración que las metodologías en su mayoría son adaptables, se puede decir que cada proyecto tiene la capacidad de establecer qué le hace falta para obtener lo que le es necesario. Pero

existen artefactos en común que modelados en diferentes perspectivas establecen la línea base de lo que puede resultar fundamental.

Comparando proyectos se puede observar que en algunos casos se subdivide estos propios diagramas para hacer más explícita la documentación. El proyecto aduana genera en esta fase 6 elementos base:

- Modelo de Diseño.
- Diagrama de paquetes.
- Diagrama de Secuencia orientado a actividades.
- Diagrama de Clases del Diseño.
- Diagrama de Despliegue.
- Modelo Físico de la Base de datos

Es notable que es imprescindible la presencia de los diagramas de clases del diseño, de secuencia, el modelo del diseño y el modelo de datos como los artefactos claves para la arquitectura. Aduana genera estos artefactos ya sea igual a las soluciones escogidas o con algunas modificaciones o especificaciones que manejan la misma información pero con características diferentes.

1.6. Arquitectura utilizada

1.6.1. Arquitectura en capas:

Patrón Modelo-Vista-Controlador (MVC).

La arquitectura en capas es un estilo cuyo objetivo primordial es la separación de la lógica de negocios de la lógica de diseño, para ello se definen capas o niveles. La capa de presentación reúne todos los aspectos del software que tiene que ver con las interfaces y la interacción con los diferentes tipos de usuarios. Estos aspectos típicamente incluyen el manejo y aspecto de las ventanas, el formato de los reportes, menús, gráficos y elementos multimedia en general. La capa del Dominio de la Aplicación reúne todos los aspectos del software que automatizan o apoyan los procesos de negocio que llevan a cabo los usuarios. Estos aspectos típicamente incluyen las tareas que forman parte de los procesos, las reglas y restricciones que aplican. Esta capa también recibe el nombre de capa de la Lógica de la Aplicación. La

capa del Repositorio reúne todos los aspectos del software que tienen que ver con el manejo de los datos persistentes, por lo que también se le denomina capa de Bases de Datos.

El patrón Modelo – Vista – Controlador permite separar los datos de una aplicación, la interfaz de usuario y la lógica de negocio en tres componentes distintos. Esto proporciona múltiples vistas sobre un mismo modelo de datos. Este patrón se usa frecuentemente en aplicaciones web donde se utilicen diferentes interfaces de usuario y el código que provee los datos a la página es dinámico. Los tres elementos esenciales de este patrón son los siguientes:

- Vista.
- Modelo.
- Controlador.
-

1.7. Tecnologías utilizadas por capas.

1.7.1.1. Tecnologías empleadas en la Capa Vista.

1.7.1.2. Ext. JS

Es un Framework de JavaScript que permite realizar aplicaciones Web enriquecidas basándose en tecnología AJAX[4], JSON, DHTML[5] y DOM[6]. Ext 3.0 está patentado bajo licencia LGPL[7] lo que posibilita su uso para aplicaciones empresariales privativas de código cerrado. Ext. JS brinda la posibilidad de utilizar un gran número de componentes visuales que mejoran considerablemente la calidad de las aplicaciones. Brinda la posibilidad de validaciones de formularios de todo tipo, basándose en expresiones regulares y tipos de datos. Trae implícitos componentes como vista en árboles, arrastrado y soltado, cambio de tamaño de imágenes, rejillas, paginado, agrupado de objetos, *tabs*, asistentes, entre otros muchos. (5)

La utilización de un Framework de JavaScript como Ext. JS facilita la separación de las capas de la vista con la del controlador desde el punto de vista productivo ya que el código utilizado en la primera es

[4] **AJAX**: acrónimo de ***A****synchronous* ***J****avaScript* ***A****nd* ***X****ML,* es una técnica de desarrollo web para crear aplicaciones interactivas.
[5] **DHTML**: (Dynamic HTML) Designa el conjunto de técnicas que permiten crear sitios web interactivos
[6] **DOM:** document Object Model para la representación de Documentos en las aplicaciones Web.
[7] **LGPL**: (Lesser General Public License) Licencia Pública General Reducida.

solamente JavaScript y no es necesario utilizar ningún tipo de código PHP, así los desarrolladores pueden centrarse más en el aprendizaje de un solo lenguaje. Además al soportar serialización de objetos mediante tecnología JSON permite que los datos enviados desde el controlador como respuesta a la vista contengan solo las propiedades de dichos objetos, pero no el comportamiento, minimizando los posibles errores de programación y los accidentes de que los objetos sean modificados erróneamente desde la vista. (5)

1.7.1.3. Lenguaje de Marcado de Hipertexto Extensible (XHTM)

Es una versión más estricta y limpia de HTML, que nace precisamente con el objetivo de remplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML. XHTML extiende HTML 4.0, por lo que tiene, básicamente, las mismas funcionalidades, combina la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos. XHTML, al estar orientado al uso de un etiquetado correcto, exige una serie de requisitos básicos a cumplir en lo que a código se refiere. Su objetivo es avanzar en el proyecto del *World Wide Web Consortium* de lograr una web semántica, donde la información, y la forma de presentarla estén claramente separadas. (5) Entre estos requisitos básicos se puede mencionar, una estructuración coherente dentro del documento donde se incluirían elementos correctamente anidados, etiquetas en minúsculas, elementos cerrados correctamente y atributos de valores entrecomillados.

1.7.1.4. JavaScript

Este lenguaje es el más utilizado del lado del cliente, porque es el navegador el que soporta la carga de procesamiento, además de ser compatible con la mayoría de los navegadores modernos. Es un lenguaje con muchas posibilidades, que permite la programación de pequeños scripts, aunque también de programas más grandes, orientados a objetos, con estructura de datos y funciones. El programador puede acceder a los elementos que forman parte de la página Web y modificarlos dinámicamente. (8)
Sobre las páginas Web se permiten realizar efectos especiales, posibilitando la creación de contenidos dinámicos y elementos de la página que tengan movimiento, cambios de colores y otras funcionalidades.

Se permite ejecutar instrucciones como respuesta a las acciones del usuario, creando páginas interactivas con programas como agendas, tablas de cálculo o calculadoras. (8)

1.7.2. Tecnología empleada en la Capa Modelo.

1.7.2.1. Propel

Propel es un Framework en su versión 1.3 programado en PHP5 de persistencia de Datos y consulta, lo que significa que brinda un mecanismo para almacenar objetos PHP en una base de datos y un sistema para búsqueda y restauración de objetos PHP desde una base de datos, todo esto a partir de un mecanismo de abstracción de los datos y utilizando ORM[8]. Propel permite realizar consultas complejas y manipulación de datos sin escribir una sola cláusula SQL. Hace más fácil la escritura de aplicaciones, el despliegue y mucho más, migrar la aplicación a otro gestor de Base de Datos si alguna vez la situación lo amerita. La última versión estable utilizada en el proyecto es la 1.3 y corre sobre PHP 5.3 Propel puede ser descrito como un mapeado objeto-relacional, una capa DAO[9], o una capa objeto persistente. Su implementación está basada principalmente en los patrones *Row Data Gateway* y *Table Data Gareway*. Propel está dividido en dos componentes principales: (5)

1. Un motor generador para construir sus clases y archivos SQL (*generador-propel*).

2. Un ambiente de ejecución que proporciona herramientas para construir consultas SQL, ejecutando consultas compiladas, y herramientas para el manejo de conexiones para múltiples bases de datos simultáneamente.

1.7.3. Tecnología empleada en la Capa Controladora.

1.7.3.1. PHP

El PHP (del inglés *Hypertext Preprocessor*) es un lenguaje de script incrustado dentro del HTML. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características especificas de sí mismo.

[8] **ORM**: (Object relational maping). Mapeo de Objeto relacional. Es una capa de abstracción entre el desarrollador y la Base de Datos.
[9] **DAO**: patrón Data Access Object o Objeto de Acceso a dato.

La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. Una de sus características más potentes es su soporte para gran cantidad de bases de datos, pueden mencionarse InterBase, MySQL, Oracle, Informix, PosgreSQL, entre otras. (9)

PHP no necesita que el navegador lo soporte, es independiente de este, pero sin embargo para que sus páginas funcionen, el servidor donde están alojadas debe soportar este lenguaje, que tiene numerables ventajas sobre otros lenguajes de programación que se ejecutan de igual manera.

1.8. Herramientas para el desarrollo de la aplicación.

1.8.1. Symfony

Symfony es un Framework para PHP5 patentado bajo licencia MTI[10], es compatible con la mayoría de gestores de bases de datos, MySQL, PostgreSQL, Oracle y SQL Server de Microsoft, entre otros en dependencia del tipo de abstracción de la Base de Datos que se utilice. Este Framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además proporciona una estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. (5)

[10] **MTI**: Licencia de Software Libre originaria del Instituto de Tecnología de Massachusetts.

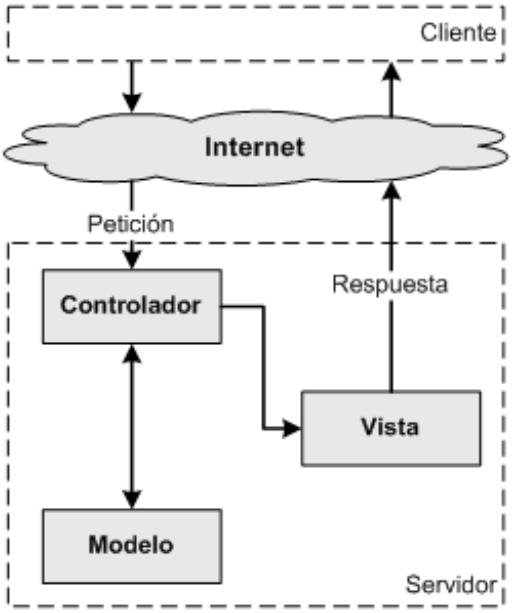

Figura 1.8.1.1 Arquitectura en capas

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

Construido específicamente para Sitios web Empresariales, de necesidades complejas y para mejorar los procesos y las prácticas de desarrollo. Los tres pilares de su desarrollo son:

1. Reunir las empresas mundiales y el mundo del código abierto.
2. Desarrollo rápido.
3. No reinventar la rueda.

Figura 1.8.1.2 Estructura de las herramientas utilizadas para desarrollar la solución por niveles

1.9. Herramientas Case

1.9.1. Visual Paradigm para UML

En su versión 6.4, genera documentos, está disponible para varios sistemas operativos como son: Windows, Linux y Unix. Es una herramienta amigable para el usuario, puede ser usada en varios idiomas y cada componente utilizado en el diagrama que se esté creando. Sugiere nuevos posibles componentes a utilizar, por lo que ya no es necesario localizarlos en la barra y así se crea fácilmente cualquier tipo de diagrama. Posee un amplio número de estereotipos que proporciona la creación de diagramas de fácil entendimiento, además de que estos diagramas los organiza automáticamente.

Es una herramienta UML profesional que da soporte al análisis y diseño orientados a objetos, construcción, pruebas y despliegue del ciclo de desarrollo de un software. (10) Permite la rápida

construcción de aplicaciones de calidad y a un menor coste. Brinda la posibilidad de crear todos los tipos de diagramas de clases, código inverso.

SlideShare (SlideShare, 2009) define que las características generales de Visual Paradigm son:

- Soporte de UML versión 2.1
- Interoperabilidad con modelos UML2 (meta modelos UML 2.x para plataforma Eclipse) a través de XMI (nueva característica).
- Generación de código: modelo a código, diagrama a código.
- Generación de bases de datos: transformación de diagramas de Entidad- Relación en modelos de base de datos.
- Ingeniería inversa de bases de datos: desde sistemas gestores de bases de datos (SGBD) existentes a diagramas de entidad-relación.
- Generador de informes para generación de documentación.

Esta herramienta se encuentra disponible en múltiples plataformas y en múltiples versiones. Sus características lo hace una opción viable ante herramientas como el Rational Rose, que es una herramienta muy recomendada y además profesional, pero obliga al usuario a desarrollar en máquinas con el sistema operativo Windows.

1.10. Lenguaje Modelado de Sistema

1.10.1. Lenguaje Unificado de Modelado (UML por sus siglas en inglés)

El lenguaje unificado de modelado (UML, por su nombre en inglés), es el lenguaje de modelado de sistemas de software más conocido y utilizado actualmente. Es un lenguaje gráfico que permite visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como: procesos de negocios,

funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. (11) . Se utiliza la versión 2.0 para el modelado. Dentro de sus características se pueden mencionar algunas tales como:

- Tecnología orientada a objetos.
- Viabilidad en la corrección de errores.
- Desarrollo iterativo e incremental.
- Participación del cliente en todas las etapas del proyecto.
- Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones).
- Cubre las cuestiones relacionadas con el tamaño propio de los sistemas complejos y críticos.
- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas.
- Existe un equilibrio entre expresividad y simplicidad, pues no es difícil de aprender ni de utilizar.

Este puede ser utilizado en sistemas desarrollados en varios lenguajes de implementación y plataformas, incluyendo lenguajes de programación y bases de datos. Las estructuras más importantes que soporta tienen su fundamento en las tecnologías orientadas a objetos, tales como: objetos, clase, componentes y nodos. Está especialmente pensado para apoyar un estilo de desarrollo iterativo e incremental.

1.11. Sistema Gestor de Base de Datos para la solución.

1.11.1. Oracle

Oracle es un Sistema de Gestión de Base de Datos (SGBD) relacional, desarrollado por Oracle Corporation. Es considerado una potente herramienta cliente/servidor para la gestión de bases de datos y uno de los SGBD más completos.

Ventajas:

- Potente y flexible.
- Se utiliza prácticamente en todas las industrias alrededor del mundo.
- Las últimas versiones han sido certificadas para poder trabajar bajo GNU/Linux.
- Está disponible en una serie de plataformas y sistemas operativos a nivel mundial y es catalogado como el segundo motor de base de datos para el desarrollo de aplicaciones.

Desventajas:

- Costoso en cuanto a su licenciamiento.
- Para poder diseñar aplicaciones útiles basadas en Oracle es necesario entender cómo manipula los datos almacenados en el sistema. Se utiliza Oracle como gestor de base de datos por ser una herramienta potente, flexible, uno de los SGBD más completos y fundamentalmente por ser el establecido por líderes y arquitectos del proyecto.

Es un entorno de gestión totalmente profesional, con el cual se pueden gestionar y administrar varias bases de datos a la vez. Para ello, tiene un estricto sistema de control de seguridad, con cuentas de usuarios y contraseñas, además de administración de privilegios para cada tipo de usuario.

Características Distintivas.

- Documenta y mantiene un registro periódico del mantenimiento, actualizaciones de hardware y software.
- Es una herramienta muy cómoda de utilizar.
- Apoya la definición de estándares de diseño y nomenclatura de objetos.
- Ayuda al análisis de datos, contribuye a la eficiencia y rendimiento de los datos que se encuentran almacenados.
- Apoya el diseño y optimización de modelos de datos.

Esta herramienta está orientada al acceso remoto y redes (Internet). En la actualidad Oracle se puede implementar en diferentes plataformas. Es soportado en computadoras personales, microcomputadoras y computadoras con procesamiento paralelo masivo.

Características de Oracle 11g.

La versión 11g de Oracle release 1, salió al mercado en febrero del 2008 con 482 nuevas características. Para así traer al mercado un producto de mayor calidad aún, que las anteriores versiones de Oracle. Este centra sus cambios en los siguientes grupos:

Nuevas características en el SQL.
Nuevas características en el soporte del lenguaje.
Nuevas características en el PL/SQL.
Nuevas características en el DBA.
Nuevas características y mejoras en el Oracle Real Application Clusters.
Nuevas características en el Rendimiento.
Nuevas características de la Seguridad.
Nuevas características en el Enterprise Manager.

1.12. Tecnologías y metodologías horizontales.

1.12.1. Proceso Unificado de Desarrollo (RUP)

RUP por su nombre en inglés, *Rational Unified Process*, es un proceso de desarrollo de software que junto a UML constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. (12) Es en realidad un refinamiento realizado por Rational Software del más genérico proceso unificado. Sus principales características son:

- Define una manera de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo)
- Pretende implementar prácticas en Ingeniería de Software de la mejor forma posible.
- Presenta un desarrollo iterativo.
- Hace posible la administración de requisitos.
- Hace uso de arquitectura basada en componentes.
- Posibilita el control de cambios.
- Modelado visual del software.
- Verificación de la calidad del software.

RUP está caracterizado por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Además incluye artefactos que son los productos tangibles obtenidos mediante su aplicación y trabajadores que no son más que las personas que juegan un rol determinado para la obtención de estos artefactos.

RUP divide el proceso de desarrollo en ciclos, se obtiene así un producto al culminar cada ciclo, estos se dividen en 4 fases que finalizan con un hito donde se debe tomar una decisión importante.

- Fase de Inicio: tiene como objetivo determinar la visión del proyecto.
- Fase de Elaboración: en esta etapa el objetivo es determinar la arquitectura óptima.
- Fase de Construcción: el objetivo es obtener la capacidad operacional inicial.
- Fase de Transición: obtener el *release* del producto.

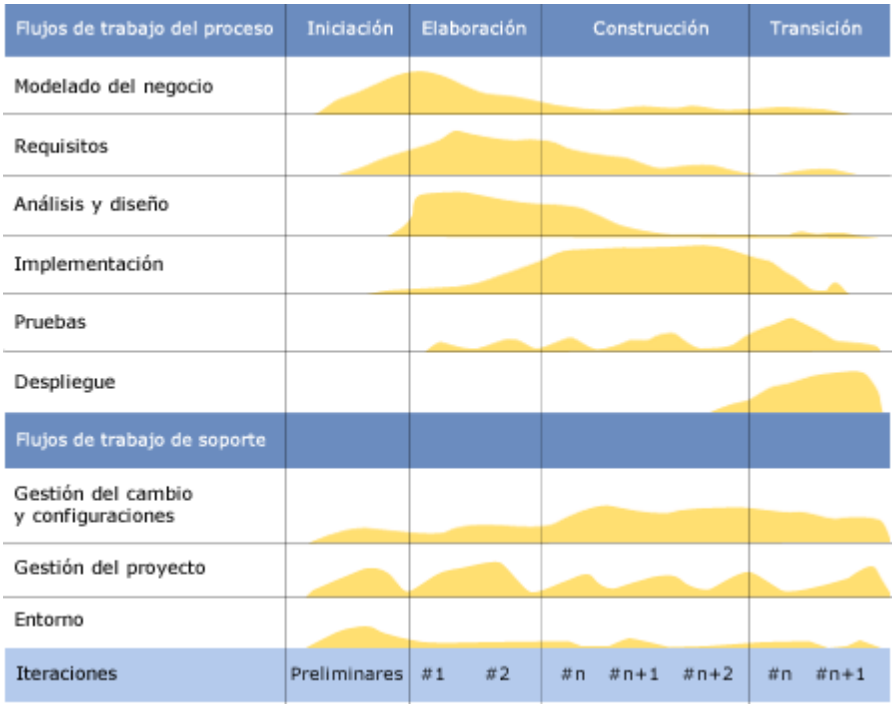

Figura 1.12.1 Flujos y Fases de RUP

Actualmente el proyecto Aduana se encuentra trabajando con nuevas adecuaciones que transforman la manera en que RUP propone sus artefactos en busca de mejor eficiencia y mayor rapidez en la entrega

de las soluciones. Por esta razón a pesar de tomar como base esta metodología no se centra el desarrollo de la aplicación en casos de uso sino que es guiado por procesos como se explica en el Trabajo Diploma Procedimiento para la Ingeniería de Requisitos en el Departamento de Desarrollo de Soluciones para la Aduana del CEIGE[11] (28).

1.13. Notación para el Modelado de Procesos de Negocio BPMN.

El modelado del negocio se realiza mediante *Business Process Modeling Notation* (BPMN o Notación para el Modelado de Procesos de Negocio), que es una notación gráfica que describe la lógica de los pasos de un proceso de Negocio, siendo esta notación diseñada para coordinar la secuencia de los procesos, y la comunicación que fluyen entre los participantes de las diferentes actividades. El modelado con BPMN es muy importante debido a:

- Es un estándar internacional de modelado de procesos aceptado por la comunidad.
- Es independiente de cualquier metodología de modelado de procesos.
- Crea un puente estandarizado para disminuir la brecha entre los procesos de negocio y la implementación de estos.
- Permite modelar los procesos de una manera unificada y estandarizada permitiendo un entendimiento a todas las personas de una organización.

Los elementos Gráficos en BPMN, están clasificados en cuatro categorías:

- **Objetos de Flujo:** son los principales elementos gráficos que definen el comportamiento de los procesos. Dentro de los cuales se encuentran:
    - *Eventos:* sucede en el transcurso de un proceso de negocio, afectando el flujo del proceso, y teniendo una causa y un resultado.
    - *Actividades:* representan el trabajo ejecutado dentro de un proceso de negocio. Las actividades pueden ser tareas o subprocesos.

[11] **CEIGE:** Centro de Informatización para la Gestión de Entidades.

- *Compuertas:* elementos del modelado utilizados para controlar la divergencia y la convergencia del flujo. Existen cinco tipos de compuertas: (Exclusiva, Basada en eventos, Paralela, Inclusiva y Compleja).

- **Objetos de Conexión:** son elementos usados para conectar dos objetos del flujo dentro de un proceso. Existen tres tipos:
  - Línea de secuencia
  - Asociaciones
  - Línea de mensaje
- **Canales:** elementos empleados para organizar las actividades de flujo en variadas categorías visuales, que pueden representar roles, responsabilidades o las áreas funcionales.
- **Artefactos:** Se utilizan para proveer información adicional sobre un proceso. Existen tres tipos:
  - Objetos de datos
  - Grupos
  - Anotaciones

1.14. Conclusiones parciales.

En este capítulo se demuestra la importancia que posee la incorporación de un módulo informatizado de selectividad a la aplicación de la aduana de Cuba para todo el país en la búsqueda del perfeccionamiento de aplicación de controles efectivos a los elementos con probabilidad de riesgo de realizar procederes ilícitos. Además se estudian los sistemas extranjeros que trabajan con un objetivo similar para de las funcionalidades de estos software tomar ideas, las cuales se implantarán en la propuesta, de forma que el producto cuente con opciones similares a las de sus competidores en el mundo, bajo las herramientas que mejoran el trabajo y con las cuales se obtienen resultados acordes a las necesidades evaluadas para cumplir con lo solicitado y superar en efectividad a la solución SUA, en funcionamiento actual.

CAPÍTULO 2: DISEÑO DE LA SOLUCIÓN PROPUESTA

2.1 Introducción

En el presente capítulo se describe el diseño del módulo selectividad para el Sistema Gestión Integral de Aduanas. En el mismo se abordan los patrones de diseño utilizados para la solución de la aplicación con sus variaciones, las clases del diseño con estereotipo Web. Se presentan los diagramas de secuencias con nuevas restructuraciones orientadas por el proyecto correspondientes a los procesos que cubren las principales tareas o funciones que el sistema ha de realizar, conjuntamente se presenta el diseño de la base de datos del sistema y como se adapta el módulo selectividad a la arquitectura definida para GINA.

Se puede decir que el diseño son arquitecturas probadas para construir software orientado a objetos que sea flexible y se pueda mantener. Estas arquitecturas son la base de cualquier aplicación que desee llegar al final con resultados deseables. Por eso establecer un vocabulario de diseño común entre los desarrolladores es fundamental en la búsqueda de acortar el tiempo y generar mejores productos.

Los propósitos que se desean obtener con un diseño son:

Adquirir una comprensión en profundidad de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia, tecnología de interfaz de usuario, tecnología de gestión de transacciones, etc. [I. Jacobsonm, G. Booch y J. Rumbaugh].

Crear una entrada apropiada y un punto de partida para actividades de implementación subsiguientes capturando los requisitos o subsistemas individuales, interfaces y clases. [I. Jacobsonm, G. Booch y J. Rumbaugh].

2.2 Patrones de diseños utilizados.

El desarrollo del sistema utilizando el framework Symfony proporciona ventajas significativas para los desarrolladores de software de manera que estos no necesitan preocuparse en mayor profundidad por cuestiones de diseño. El marco de trabajo mencionado es capaz de fusionar buenas prácticas de diseño por sí mismo a través de las capas que utiliza en la comunicación de la información, de forma que los desarrolladores no tengan que preocuparse por implementar varios de los patrones de diseño y arquitectónicos más utilizados en la actualidad, ya que el mismo framework los utiliza como buena práctica de trabajo. (13)

2.2.1. Patrones GRASP.

2.2.1.1. Creador

Este patrón como su nombre lo indica es el que crea, el guía la asignación de responsabilidades relacionadas con la creación de objetos. (14)

La clase selectividadActions contiene las acciones definidas para el módulo selectividad y es en ella misma donde se ejecutan las funciones que dan vida al sistema en su totalidad. En esta clase servidora principal, las acciones se encargan de crear los objetos de las clases que representan las entidades y manejan sus propias propiedades, evidenciando de este modo que la clase selectividadActions es el "creador" de las instancias de las entidades y el único que tiene conocimiento de cómo debe ser gestionada la información para lograr darle cumplimiento a las solicitudes introducidas.

2.2.1.2 Experto

La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). (14)

Se evidencia este patrón puesto que Propel es la librería externa que utiliza Symfony para realizar su capa de abstracción al modelo de datos por sus facilidades con la arquitectura de tres capas, encapsulando toda la lógica de los datos y generando las clases con todas las funcionalidades comunes de las entidades. Por tanto cada clase creada por Propel a partir de una entidad es experta en manejar su información por ser la raíz común entre todas sus propiedades.

2.2.1.3. Alta Cohesión

La cohesión es una medida de la fuerza con la que se relacionan las clases y el grado de focalización de las responsabilidades de un elemento. (14)

Symfony permite la asignación de responsabilidades con una alta cohesión según su definición, por citar un ejemplo la clase selectividadActions tiene la responsabilidad para definir las acciones sobre las plantillas y colabora con otras para realizar diferentes operaciones e instanciar objetos, es decir, está formada por diferentes funcionalidades que se encuentran estrechamente relacionadas, proporcionando que el software sea flexible frente a grandes cambios y garantizando que cada función cumpla un determinado papel que solo puede ser asumido por ella.

2.2.4. Controlador

Es un evento generado por actores externos. Se asocian con operaciones del sistema, operaciones del sistema como respuestas a los eventos del sistema, tal como se relacionan los mensajes y los métodos. (14)

Todas las peticiones Web son manejadas por un solo controlador frontal (lcf.php), que es el punto de entrada único de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción ejecutada y el nombre de un módulo con la URL entrada por el usuario en la búsqueda de darle cumplimiento a las funcionalidades capturadas.

2.2.1.5. Bajo Acoplamiento

El acoplamiento es una medida de fuerza con que un elemento esta a, tiene conocimiento de, confía en, otros elementos. Este patrón es un principio que asigna la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. (14)

En el diseño de las clases se aplicó este patrón logrando un modelo con las relaciones necesarias que impulsan la asignación de responsabilidades de manera que su localización no incremente el acoplamiento hasta un nivel que lleve a resultados negativos. De esta forma se tiene un diseño de clases más independiente que reduce el impacto al cambio, mejora la gestión de la búsqueda de información o su almacenamiento.

2.2.2 Patrones GOF utilizados.

2.2.2.1 Singleton (Instancia única)

Problema: se admite exactamente una instancia de una clase. Los objetos necesitan un único punto de acceso global.

Solución: definir un método estático de la clase que devuelva el singleton

Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Es el caso del controlador frontal, donde hay una llamada a la función sfContext::getInstance() que garantiza que siempre se acceda a la misma instancia.

2.2.2.2 Abstract Factory (Fábrica abstracta)

Se utilizó este patrón para trabajar con objetos de distintas familias de manera que no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando. Cuando el framework necesita crear un nuevo objeto, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea por su capacidad para saber que subclase se necesita para instanciar de ella en tiempo de ejecución.

2.2.2.3 Decorator (Decorador)

Problema: ¿Cómo añadirle una nueva característica a una clase, pudiendo así obtener gran funcionalidad combinando piezas sencillas?

Propósito: añadir responsabilidades adicionales a un objeto dinámicamente, proporcionando una alternativa flexible a la especialización mediante herencia, cuando se trata de añadir funcionalidades.

Añade funcionalidad a una clase, dinámicamente. El archivo layout.php, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que

repetirlo en cada página. El contenido de la plantilla se integra en el layout, o si se mira desde el otro punto de vista, el layout decora la plantilla.

2.3 Patrones arquitectónicos utilizados

Se utiliza un único patrón arquitectónico capacitado para asimilar una arquitectura que debe tener una separación entre lo visual al usuario y los datos introducidos por ellos y los datos persistentes del negocio. Y que posibilite aplicar los patrones de diseño que faciliten el trabajo y mejoren el resultado.

2.3.1 Patrón MVC.

El principio más importante 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.

Modelo: es la representación de la información que maneja la aplicación. El modelo en sí son los datos puros que puestos en contexto del sistema proveen de información al usuario y a la aplicación misma. Es el núcleo de la aplicación. Este mantiene el estado y los datos que la aplicación representa. Cuando se produzcan cambios significativos en el modelo a partir de especificaciones de cambio formuladas normalmente por el controlador, se actualiza todas las vistas asociadas al modelo.

Vista: es la representación del modelo en forma gráfica, disponible para la interacción con el usuario. En el caso de una aplicación Web, la “Vista” es una página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones. Este presenta el modelo en un formato adecuado para interactuar, usualmente un elemento de interfaz de usuario

Controlador: es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando el modelo en caso de ser necesario. Recibe o interpreta la información que proporciona el usuario y se inicia una respuesta. Hace llamadas de los objetos del modelo, provoca cambios en este y en caso de ser necesario en la vista.

Ventajas:

 Vistas múltiples: La aplicación puede mostrar el estado del modelo en una variedad de formas, y la creación y diseño de una manera escalable y modular. Esto es posible gracias al aislamiento o independencia que presenta el modelo con respecto a la vista.

 Adaptable a cambios: Los requerimientos de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas de negocios. Los usuarios pueden preferir distintas opciones de representación, o requerir soporte para nuevos dispositivos como teléfonos celulares. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación generalmente no afecta al modelo.

Desventajas:

 Costo de actualizaciones frecuentes: Si el modelo experimenta cambios frecuentes, por ejemplo, podría desbordar las vistas con una gran cantidad de requerimientos de actualización.

2.4 Extensiones para el diseño Web.

Las extensiones UML para el diseño Web, exponen la solución para el modelamiento de diagramas de clases del diseño sobre tecnologías Web. De forma que quedan presentados los siguientes estereotipos que intentan separar visualmente los componentes para diferenciar los extremos de esta arquitectura Web cliente-servidor.

Estereotipos	Imagen	Descripción
Server page		Representa una página Web dinámica que contiene el código ensamblado por el servidor cada vez que se solicita. Típicamente, una página servidora contiene scripts que se ejecutan en el servidor y que actúan recíprocamente con los recursos del lado del servidor estos son: las bases de datos, los componentes de la lógica del negocio y sistemas externos.
Client page		Representa una instancia de una página cliente. Es una página Web con formato HTML. Las páginas cliente son interpretadas y mostradas por los navegadores del cliente y además pueden contener scripts que se interpretan en el navegador.

Form		Simboliza un formulario, es el elemento encargado de realizar envíos a las páginas servidoras.

Tabla 2.4.1 Tipos de extensiones Web.

Relaciones que se establecen entre las clases que conforman la extensión UML para Web.

Hasta Desde	Server Page	Client Page	Form
server page	<<Redirect>>	<<Build>>, <<Redirect>>	--
client page	<<Link>>, <<Redirect>>	<<Link>> , <<Redirect>>	Contiene
form	<<Submit>>	Agregado por.	--

Tabla 2.4.1.2 Tipo de relaciones.

El autor Jim Conallen en su libro *"Buildin Web Applications with UML Second Edition"*, presenta las siguientes descripciones para los estereotipos utilizados en las relaciones entre las clases.

Estereotipos	Descripción
<<Link>>	Representa una relación entre una página del cliente y un recurso del lado del servidor, o una página Web.
<<Build>>	Representa una relación direccional entre una página servidora y cliente. Esta relación identifica la salida HTML de la ejecución de una página servidora.
<<Submit>>	Relación directa entre un formulario <<HTML form>> y una página servidora. Similar a la relación <<Link>>, pero solo referencia a recursos del lado servidor. Sin embargo, cuando los recursos son pedidos desde el servidor, todos los campos del formulario son enviados al servidor junto con la petición, donde son procesados.
<<Redirect>>	Relación direccional entre páginas clientes, páginas servidoras y unas con otras. Esta asociación indica un comando a la página cliente para realizar petición de otro recurso.
<<Include>>	Asociación direccional desde una <<server page>> a otra <<server page>> o <<client page>>. Esta asociación indica que las páginas incluidas son procesadas mientras que la página se ensambla.

Tabla 2.4.1.3 Tipo de estereotipos Web

2.5 Clases del diseño con estereotipos Web.

Una clase de diseño y sus objetos participan en varias realizaciones de procesos de negocio en este caso. También puede suceder que algunas operaciones, atributos y asociaciones sobre una clase específica sean solo relevantes para una sola realización de proceso de negocio. Para manejar todo esto se utilizan los diagramas de clases conectados a una realización de caso de uso (en este caso proceso), mostrando así sus clases participantes, subsistemas y sus relaciones**.** [I. Jacobsonm, G. Booch y J. Rumbaugh].

Requisitos Funcionales.

Cada uno de los procesos descritos en diagramas de clases para el diseño responden a un requisito funcional del negocio que es quien evidencia la necesidad de desarrollar estos diagramas de clases del diseño. Conociendo los requisitos se determina si se da cumplimiento a las necesidades.

Requisito Funcional 1 Proponer creación de grupo.

El analista de la AGR después de analizar los posibles riesgos propone al administrador del CADI la creación de un nuevo grupo.

Requisito Funcional 2 Buscar propuesta de creación de grupo.

Se muestra la propuesta de creación de un grupo con lo que se necesita: canales correspondientes, alcance que comprenderá (nacional ó local), eventos de ejecución (automático ó por el usuario o ambos) y la relación que tendrá con otros módulos.

Requisito Funcional 3 Actualizar estado de propuesta.

El administrador del CADI debe modificar el estado de las propuestas realizadas por los analistas cuando crea el grupo.

Requisito Funcional 4 Insertar grupo.

Se procede a la creación del grupo con los parámetros determinados como son: definir tablas y sus relaciones, canales, eventos de ejecución, alcance, relación con otros módulos y el azar.

Requisito Funcional 5 Aplicar Selectividad.

Una vez definidos los criterios por los analistas, estos se ejecutarán al objeto de control en el momento que defina el proceso aduanero por el que pasa.

Requisito funcional 6 Eliminar criterios.

Permite que se elimine un criterio de selectividad determinado.

Requisito funcional 7 Insertar criterio.

Permite insertar un nuevo criterio de selectividad.

Requisito Funcional 8 Modificar criterio.

Permite modificar un criterio atendiendo a sus condiciones y datos generales.

Requisito Funcional 9 Eliminar condiciones.

Permite que se elimine una condición determinada teniendo en cuenta los valores correspondientes de los mismos.

Requisito Funcional 10 Insertar condiciones.

Permite insertar una condición determinada teniendo en cuenta sus valores.

Requisito Funcional 11 Modificar condiciones.

Permite modificar una condición determinada teniendo en cuenta sus valores.

Requisito Funcional 12 Eliminar valores.

Permite eliminar el valor o los valores que determina el analista.

Requisito Funcional 13 Insertar valores.

Permite insertar el valor o los valores por el analista.

Requisito funcional 14 Modificar valores.

Permite realizar cambios en los valores por los analistas.

Requisito funcional 15 Registrar resultados.

Luego de aplicar la selectividad, se registran los resultados obtenidos para poder analizar la efectividad de los criterios.

Requisito funcional 16 Realizar Premonitoreo.

Se realiza para verificar el comportamiento que tendrán los criterios vigentes ante un grupo determinado.

Requisito Funcional 17 Modificar azar.

El sistema debe permitir modificar el porciento azar que se había especificado en la creación del grupo para determinar el nivel de afectación por cada aduana.

Requisito funcional 18 Aplicar selectividad de forma manual.

El sistema debe permitir aplicar la selectividad por una persona, siendo esta el analista, al objeto o a los objetos de control definidos por los procesos aduaneros.

Requisito Funcional 19 Modificar grupo.

El sistema debe permitir modificar los campos del grupo cuando fue creado, teniendo en cuenta el alcance, relación con otros grupos, tablas y canales.

2.5.1 Descripción de los Procesos a través de los Diagramas de Clases del Diseño

La utilización del framework Symfony y el uso de las librerías Ext. JS permiten desarrollar un diseño con una estructura similar para varios procesos y lograr una relación entre cada uno de los diseños y una armonía en la solución visual que debe presentarse. La página servidora selectividadActions.class es responsable de las peticiones que realiza el controlador frontal y sus respectivas manipulaciones para brindar una respuesta correcta y precisa; luego estas se redireccionan al Layout. El Layout carga en el cuerpo la plantilla (como se trata de un menú de procesos esta página carece de contenido visual descriptivo y no requiere de diagramización), luego se construye la página cliente que importa todas las

interfaces de usuario del módulo según sea la necesidad, estas interfaces están definidas en el fichero de configuración view.yml y se encuentran representadas dentro del paquete Interfaces JS.

Proceso: Gestionar Selectividad por Jefe de LCF AGR

Se tiene una relación de agregación entre la página cliente principal y el formulario Form_GestionarSelectividad, que contiene todos los componentes necesarios para la solución y cumplimiento de las necesidades expuestas en los procesos definidos. Luego los datos entrados a través de diferentes comandos son enviados al servidor y recibidos por las funciones de la clase selectividadActions.class, las cuales mediante las clases del paquete de Acceso a Datos y el Núcleo del sistema[12] son capaces de registrar y obtener información.

Para el entendimiento del negocio se generalizó el procedimiento adecuado de envío y recibo de datos entre los elementos que componen a cada proceso descrito en este diagrama particularizando en cada requisito la interacción entre los diferentes patrones utilizados en esta arquitectura de tres capas.

[12] **Núcleo del sistema:** representa la parte del GINA dedicada al intercambio de información entre subsistemas, los cuales son expertos en el manejo de información específica a las cuales la selectividad debe recurrir para dar las soluciones necesarias.

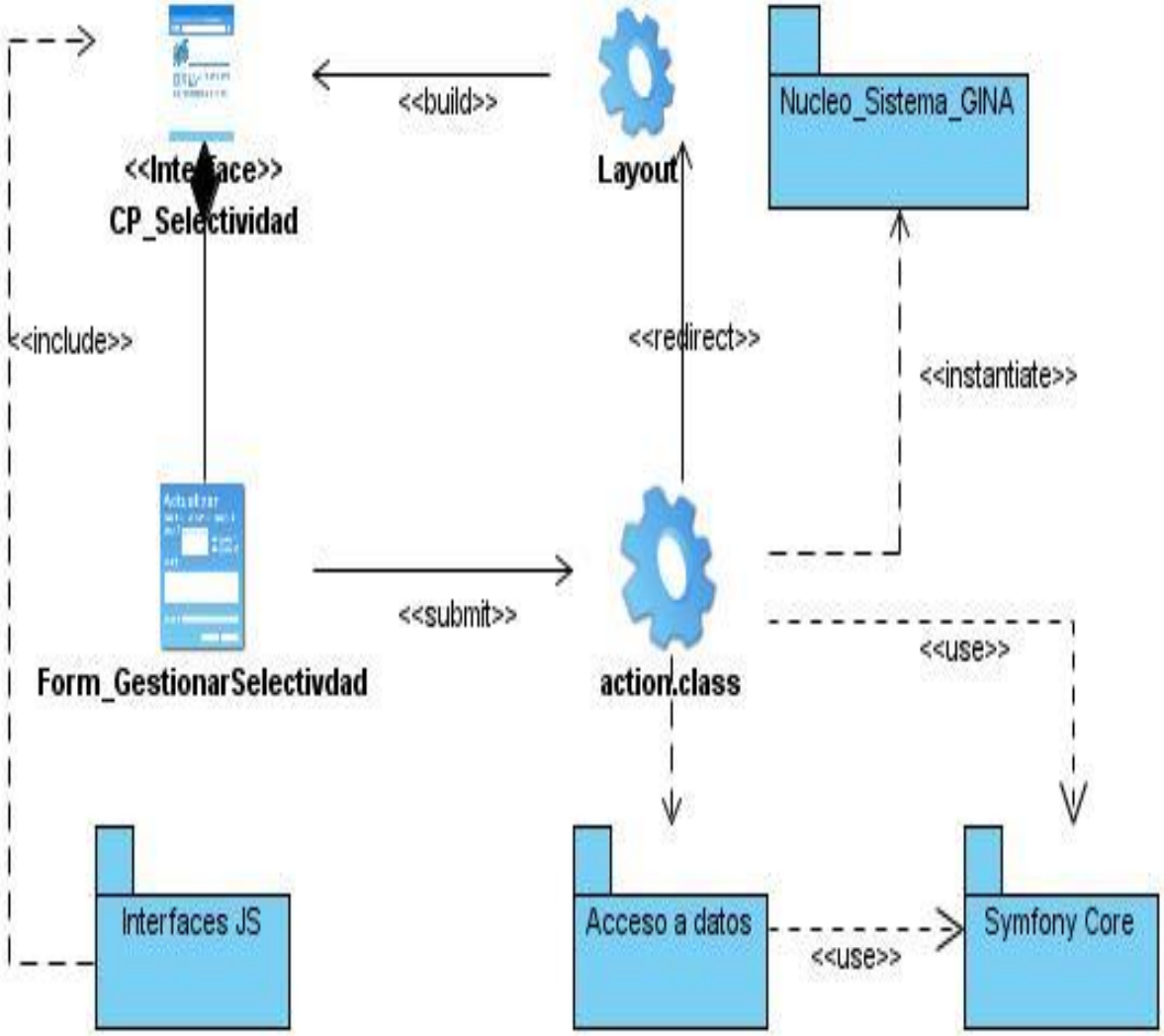

Figura 2.5.1 Diagrama de Clases con estereotipos para Gestionar Selectividad

El resto de los diagramas utilizados por requisitos están definidos y representados dentro del Modelo del diseño del módulo selectividad, uno de los artefactos que se generan en esta fase.

2.6 Diagrama de Paquetes

El paquete “Acceso a Datos” contiene tres paquetes en su interior Objetos, Objetos Peer y Propel. Ver Fig. 2.6.1. Dentro de estos paquetes están las clases necesarias para efectuar la conexión y el intercambio de información de la aplicación con la base de datos.

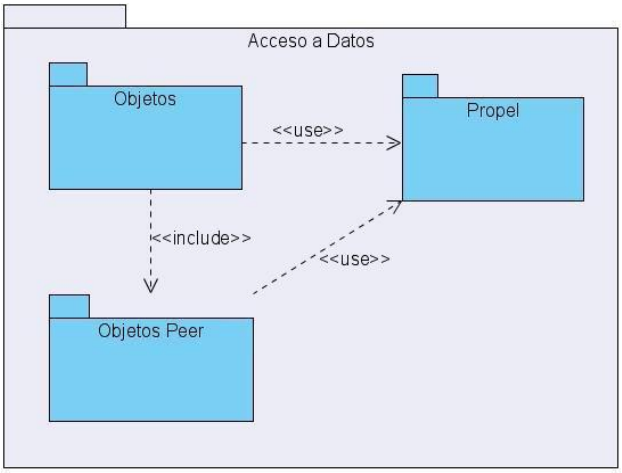

Figura 2.6.1 Paquete Acceso a Datos

El paquete Propel representa la capa de abstracción de objetos/relacional utilizada por Symfony, implementa una de las mejores capas para la abstracción a bases de datos disponible en PHP5, y se encuentra completamente integrado al framework además de proveer todas las relaciones existentes entre cada una de las tablas que son los componentes del modelo.

Por su parte el paquete Objetos tiene las clases de la lógica de la aplicación que contienen los atributos y métodos necesarios que mediante el uso de Propel permiten la inserción y actualización de información persistente. Por cada tabla de la base de datos, existen 2 clases que pertenecen al paquete Objetos, por ejemplo: la clase sel_grupo hereda de Basesel_grupo, de esta forma se logran los objetivos (inserción y actualización) sólo modificando la clase hija, ya que la clase base tiene las funcionalidades para el acceso a la base de datos con el uso del paquete Propel.

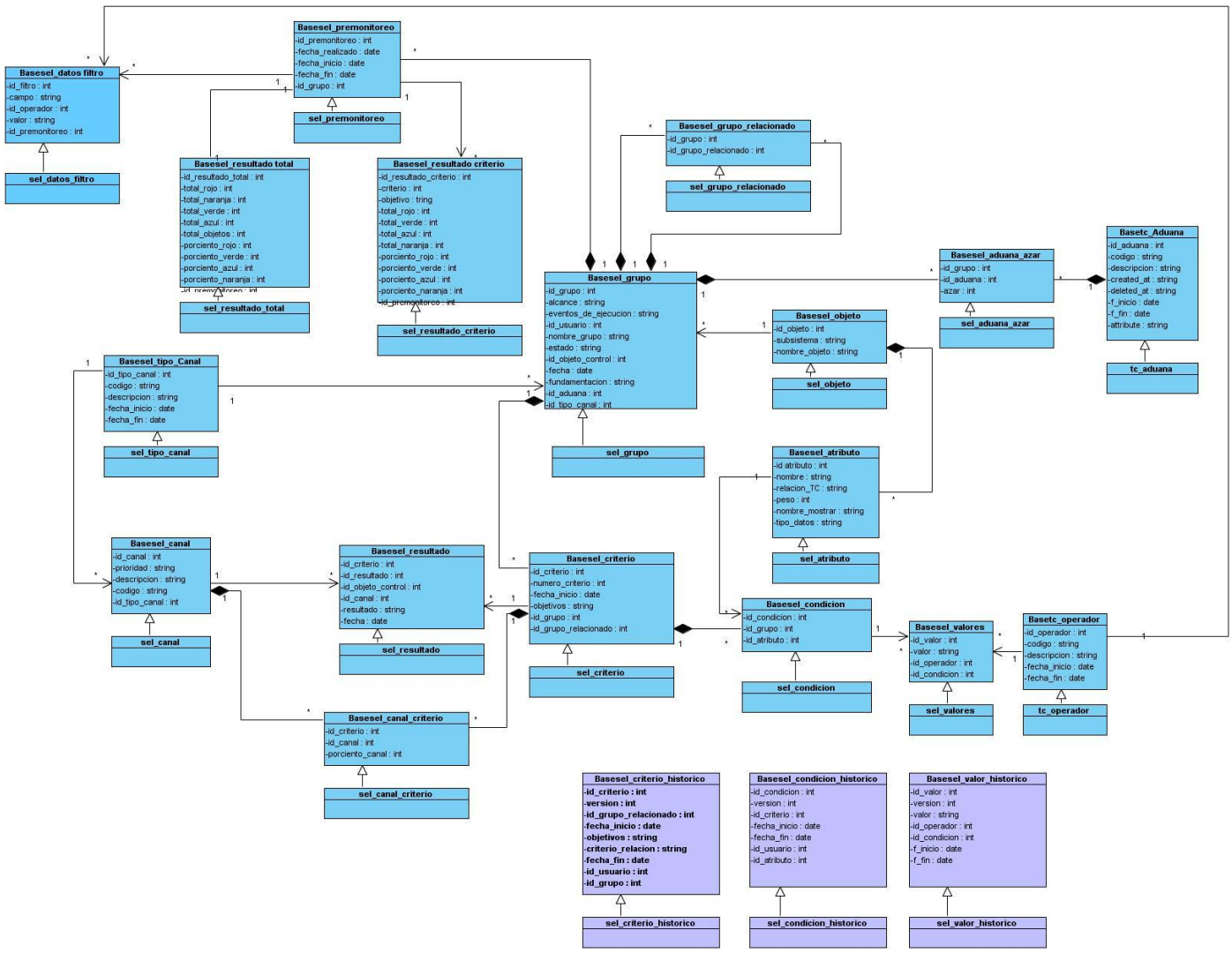

Figura 2.6.2 Paquete Objeto

Las clases del paquete Objetos Peer contienen los métodos y atributos para efectuar consultas a la base de datos, aquí también por cada una de las tablas en la base de datos se presentan dos clases siguiendo el mismo ejemplo se tendría que sel_criterioPeer hereda de Basesel_criterioPeer, así solo se modificará la clase hija y se podrán utilizar todos los métodos y atributos de la clase padre para lograr las consultas. Ver Fig. 2.6.3.

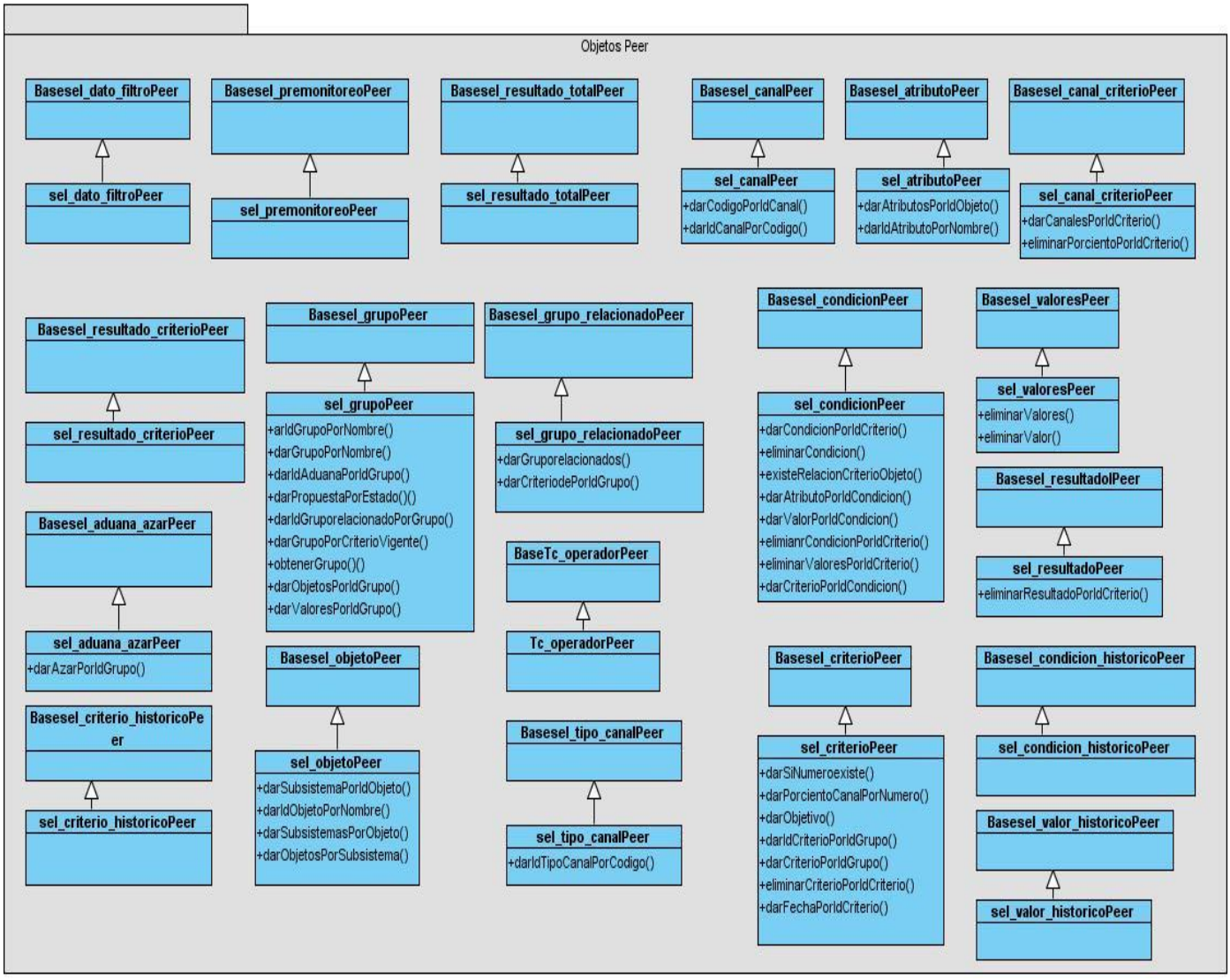

Figura 2.6.3 Paquete de Objetos Peer

El paquete “Interfaces JS” contiene los códigos JavaScript que son ensamblados por el navegador una vez que la página cliente se está interpretando. Ver Fig. 2.6.4.

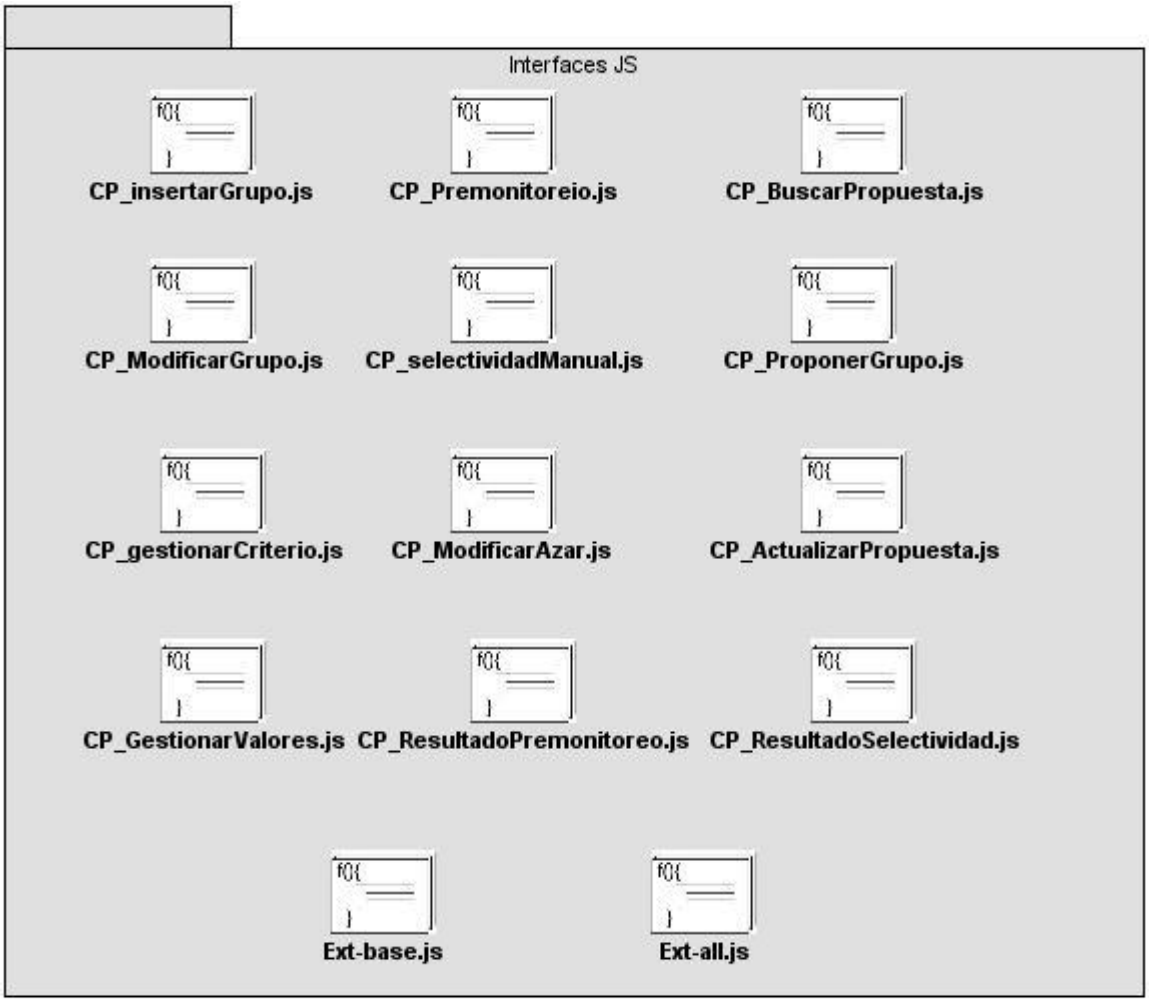

Figura 2.6.4 Paquete Interfaces

Los script Ext-base.js y Ext-all.js representan el núcleo de las librerías Ext.JS utilizadas, y los otros constituyen las interfaces del módulo selectividad.

Todos estos paquetes son utilizados e interactúan entre ellos para brindar las soluciones que cada requisito expuesto necesita para que haya sido cumplido. Se puede resumir en un diagrama de paquetes que manifieste estas relaciones de uso.

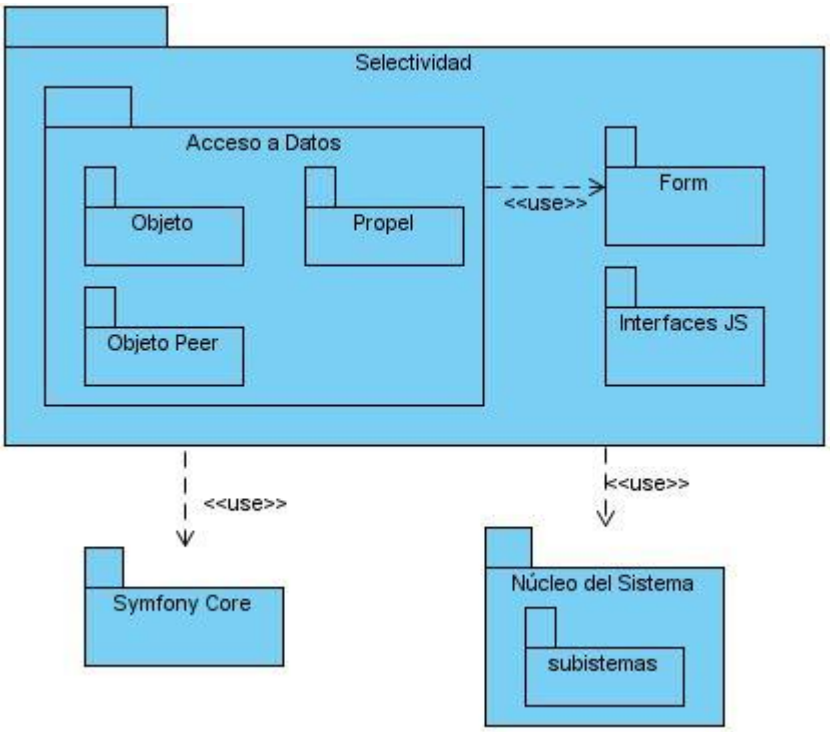

Figura 2.6.5 Diagrama de Paquetes.

2.7 Diagramas de Secuencia orientados a Actividades.

Estos diagramas son empleados dentro de las nuevas regulaciones en el proyecto aduana que se alejan un poco del trabajo tradicional que se hace con la metodología RUP. En este caso siguen existiendo calles que representan las clases, las cuales deben comunicarse para la transmisión de los datos pero cada actividad que necesite de cambio de estado es graficada con todos los pasos que necesita para culminar la tarea. Este diagrama trata a grandes rasgos de evitar el engorroso trabajo que realizan los diseñadores a la hora de diagramar una solución y que resulte rápida y eficaz para el programador que espera su turno. Resulta casi incompresible y tedioso lograr entender las bifurcaciones que en ocasiones son

imprescindibles en un diseño de aplicaciones Web a través del diagrama de secuencia, mal que se trata de erradicar con este nuevo concepto que se adentra dentro de las adecuaciones a RUP realizadas en el proyecto Aduana explicadas en el trabajo diploma titulado Procedimiento para la Ingeniería de Requisitos en el Departamento de Desarrollo de Soluciones para la Aduana del CEIGE.

Tipos de Diagramas

Este tipo de diagrama presenta dos soluciones gráficas para las dos vertientes que posee, Diagrama Visual y Diagrama del negocio. Ambos diagramas tratan de lograr el mismo objetivo pero con la diferencia que del negocio se obtiene más provecho de los datos almacenados y requiere de mucha más profundidad en el alcance de los métodos mientras el visual trata a las funcionalidades necesarias para incorporarle a la vista del negocio elementos que permitan la selección de los datos para la ejecución de los diagramas del sistema y no requieren de la petición a través de datos del controlador frontal para su funcionamiento en su mayoría.

2.7.1 Diagrama de Secuencia orientado a actividades Visual.

Este tipo de diagrama es más corto en su ejecución y debe ser más rápido por la necesidad de responder a los diagramas del negocio. Trabaja fundamentalmente en la capa visual del Symfony.

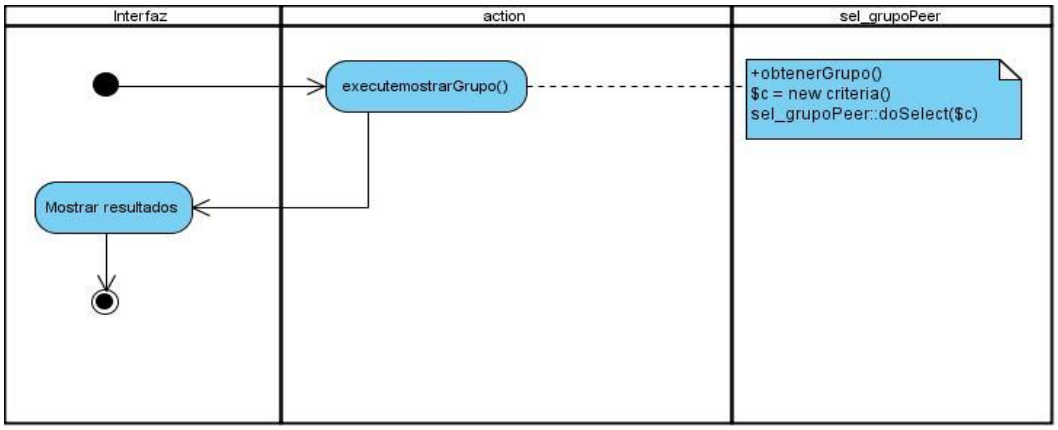

Figura 2.7.1.1 Diagrama de Secuencia Orientado a Actividades Visual Mostrar Grupo

2.7.2 Diagrama de Secuencia orientado a Actividades Negocio.

Este tipo de diagrama es la respuesta del diseño a los requisitos funcionales capturados en fases anteriores y evidencia paso a paso la interacción entre las tres capas de la arquitectura y los pasos que debe seguir el programador para concluir satisfactoriamente la aplicación final. Tiene un nivel superior de interacción con los datos del sistema.

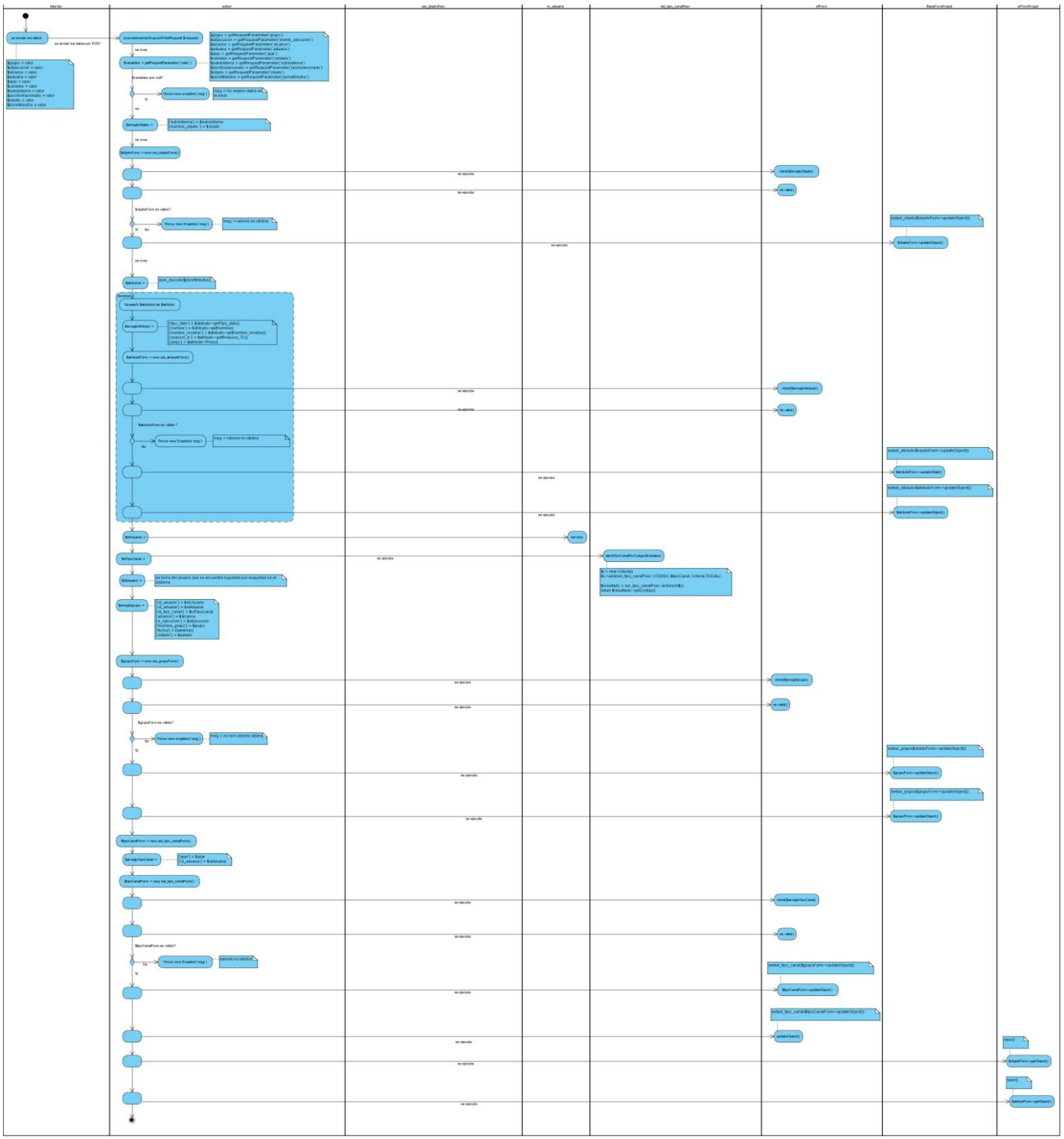

Figura 2.7.2.1 Diagrama de Secuencia orientado a Actividades Negocio Insertar grupo

Todos los diagramas responden a la descripción realizada en el requisito funcional y dan cumplimiento a las peticiones. El resto de los diagramas se encuentran anexados al documento y detallados en el Modelo del Diseño.

2.8 Diseño de la Base de Datos.

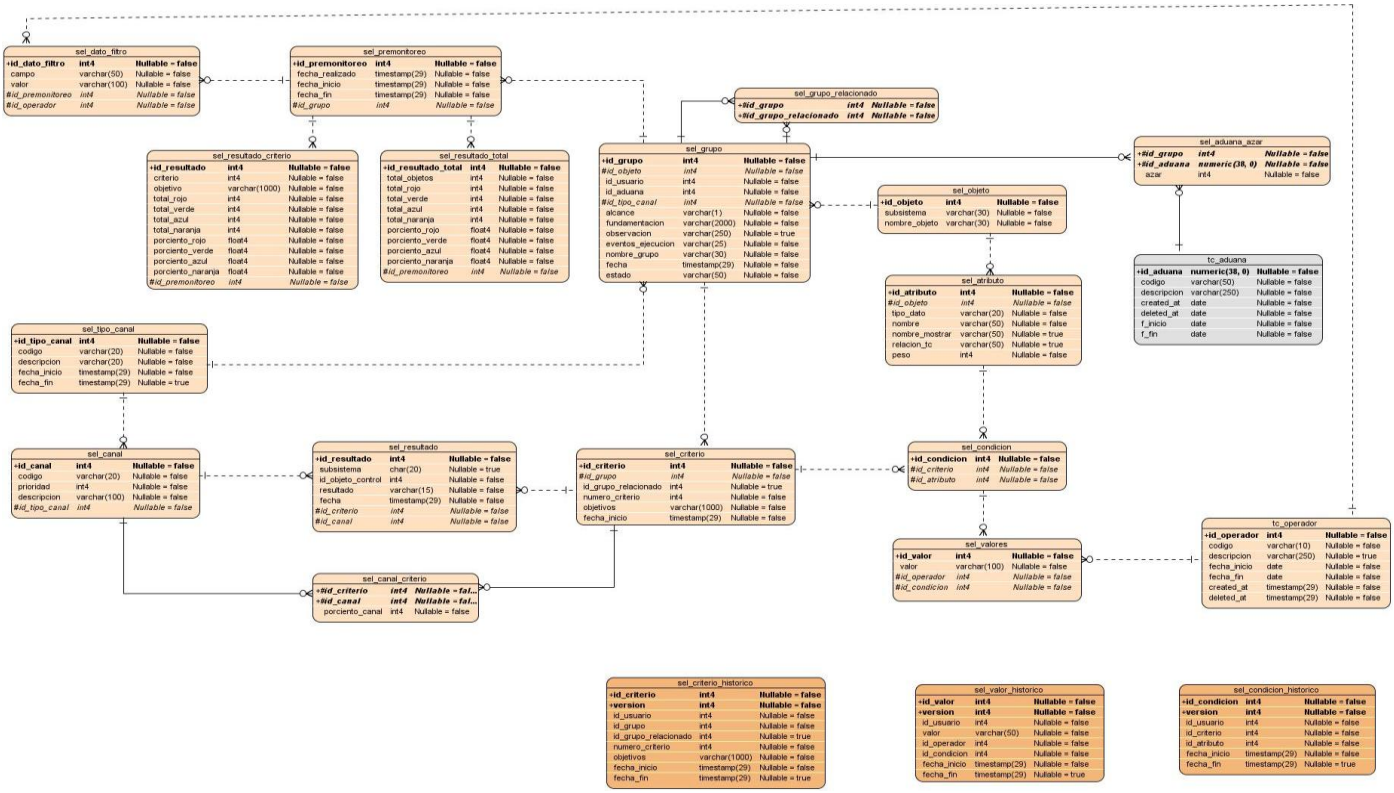

Figura 2.8.1 Diseño Base de Datos

2.9 Diagrama de Despliegue.

Este diagrama se inicia en la fase de diseño para terminarlo con la implementación de la aplicación. Por lo tanto se muestra un diagrama que aún debe ser modificado para completarlo correctamente. En este diagrama se visualiza como se realiza la interacción entre los diferentes equipos que necesitan de una

conexión para realizar las diferentes peticiones y devolver los resultados de estas peticiones, donde se encuentra el núcleo de toda la solución.

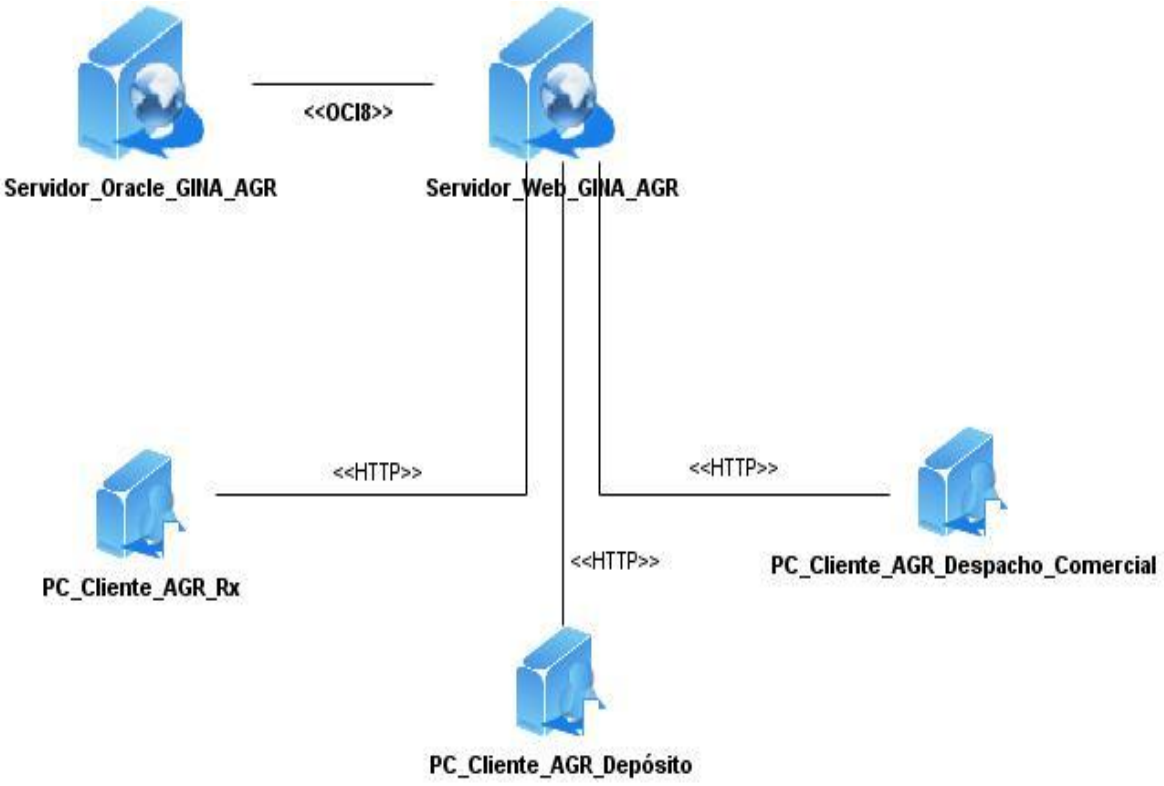

Figura 2.9 Diagrama de despliegue de Selectividad.

En este diagrama se emplean varias PC_Clientes para demostrar que en el servidor GINA donde se ubica el módulo de Selectividad, todos los subsistemas incorporados en esta solución en algún momento pueden realizar la petición de efectuar a este proceso.

2.10 Conclusiones parciales.

En el transcurso de este capítulo se obtuvieron los artefactos del diseño de la aplicación en sus diferentes variantes. Se realizaron las clases del diseño con estereotipo Web, diagrama de secuencia orientado a actividades, el diseño de la Base de Datos y el diagrama de Despliegue. Estos servirán de entrada para el flujo de implementación, facilitando un mayor entendimiento de la aplicación a los desarrolladores y menos demora para llegar a la solución final.

CAPÍTULO 3: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

3.1 Introducción.

En este capítulo se aborda todo lo referente a las métricas que deben utilizarse para corroborar si el diseño que se ha propuesto cumple con la serie de requisitos de calidad que debe tener para considerarse adecuado y que no representa ningún tipo de limitante para realizar la implementación basada en esta propuesta.

En la ingeniería de software se ha hecho recurrente la necesidad de comprobar si los artefactos que se generan en la búsqueda de una solución son factibles. Por lo tanto es importante aplicar técnicas que permitan obtener resultados cuantificados que evidencien si se ha tomado el camino correcto. Precisamente porque ese resultado numérico es lo que posibilita obtener respuestas efectivas. Con este objetivo se aplican las métricas que existen para el software, que permiten centralizar las funcionalidades operativas de un programa. Estas métricas se especializan en las distintas fases por las que atraviesa una aplicación informática, pero basadas en principios generales que referencian la calidad con la que se ha estructurado los diferentes artefactos. La calidad es el factor fundamental para determinar objetivamente si se ha obtenido un producto que cumpla con las características que debe tener toda solución completamente funcional.

Existen seis características para determinar la viabilidad del producto informático:

Funcionalidad: la capacidad del software de proveer las funciones que cumplen con las necesidades implícitas y explícitas cuando el mismo es utilizado bajo ciertas condiciones.

• **Fiabilidad:** la capacidad del software de mantener un nivel específico de rendimiento bajo determinadas condiciones de uso.

• **Usabilidad:** la capacidad del producto software de ser entendido, aprendido, usado y atractivo al usuario, cuando se usa bajo ciertas condiciones.

• **Eficiencia:** la capacidad del software de ofrecer el rendimiento apropiado con respecto a la cantidad de recursos utilizados, bajo condiciones prefijadas.

• **Mantenibilidad:** la capacidad del producto de ser modificado. Dichas modificaciones pueden incluir correcciones, mejoras o adaptaciones a cambios en el entorno y en los requisitos y especificaciones funcionales.

• **Portabilidad:** la capacidad del software de ser trasladado de un entorno (informático) a otro.

Debido a que estas características responden a la calidad de la aplicación sin importar que tipo de paradigma se utiliza y generalmente solo la mantenibilidad da respuesta en la fase de diseño, no antes o después, es necesario recurrir a una serie de métricas propuestas por diferentes ingenieros que aseguran encontrar una respuesta objetiva y certera sobre el diseño Orientado a Objetos o Diseño OO[13].

Basándose en que la calidad es un componente entre la calidad externa e interna es importante darle un correcto tratamiento a ambas. La calidad interna es medida a través de diferentes aspectos que se pueden cuantificar para demostrar con evidencias que los resultados esperados son verídicos y a su vez esta tributa a la calidad externa que se enfoca en la parte humana por lo que logrando validar la calidad interna de un software se estaría determinando si el diseño propuesto es correcto.

Gran parte del diseño orientado a objetos es subjetivo un diseñador experimentado “sabe” como puede caracterizar un sistema OO para que se implemente de forma efectiva los requisitos del cliente. Pero a medida que los modelos de diseño OO van creciendo de tamaño y complejidad, puede resultar beneficiosa una visión más objetiva de las características del diseño, tanto para el diseñador experimentado como para el menos experimentado. (15)

Es necesario recopilar información que pueda ser observable y para esto se debe recurrir a propiedades que por su carácter restrictivo permiten encontrar métricas adecuadas que aplicar a un diseño cuando se trata de una clase:

[13] **OO:** Terminología utilizada para referir Orientada a Objetos.

- Acoplamiento
- Cohesión
- Herencia

3.1.1. ¿Qué hace tan importante utilizar métricas que contengan a estas propiedades en un diseño?

Se sabe que la clase es la unidad principal de todo sistema OO. Por consiguiente, las medidas y métricas para una clase individual, la jerarquía de clases, y las colaboraciones de clases resultarán sumamente valiosas para un ingeniero de software que tenga que estimar la calidad de un diseño. Se ha visto que la clase encapsula a las operaciones (procesamiento) y a los atributos (datos). La clase suele ser el "predecesor" de las subclases (que a veces se denominan "descendientes") que heredan sus atributos de operaciones. La clase suele colaborar con otras clases. (15)

3.1.2. ¿Qué hace a estas características importantes en una métrica?

Precisamente estas propiedades permiten diferenciar al diseño, de una métrica convencional para software.

El software orientado a objetos es esencialmente distinto del software que se desarrolla utilizando métodos convencionales. Por esta razón, las métricas para sistemas OO deben de concordarse a las características que distinguen el software 00 del software convencional. (15)

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

Cada una de estas características logra tratar de diferentes ángulos los elementos que permiten generar métricas especializadas que retornan valores contables y comparables. Es notable que todas estas características responden a la unidad central de toda aplicación, la clase, junto a todos sus componentes.

3.1.3. ¿Por qué es necesario utilizar métricas especializadas en clases?

Es el elemento principal que permiten contabilizar la información que maneja, principal parámetro para demostrar la calidad de un diseño, el cual se basa fundamentalmente en las clases y sus respectivas relaciones entre ellas en un esquema.

3.1.4 Umbrales

Los umbrales en cada una de estas métricas ha sido difícil de establecer porque influyen muchos factores para determinar una media que valide el trabajo tanto de la persona que interviene directamente con la solución como cada una de las posibles fallas o aristas que se generen durante el desarrollo del diseño. Las aproximaciones dependen en gran medida de quien se encuentre diseñando, por lo tanto no es posible encontrar umbrales estándares que se utilicen en cualquier tipo de esquema que requiera de diseño.

En este diseño se trató de adecuar los umbrales utilizados, a las características propias que presenta el mismo y mantener una concordancia y niveles de referencias que cubran varias posibilidades de respuesta. En algunas métricas, referencias utilizadas en otros documentos y que pueden ser aplicados.

3.2 El conjunto de métricas CK

Se trata de un conjunto de métricas desarrolladas por los ingenieros Chidamber y Kemener[14] ampliamente conocidas y utilizadas para la validación de un diseño. Elaboradas precisamente para el trabajo con las clases y sus operaciones o atributos.

[14] **Chidamber y Kemener:** ingenieros estadounidenses que desde el año 1991 son la referencia mundial para aplicar métricas de diseño Orientado a Objeto.

3.2.1 Carencia de Cohesión en los Métodos (CCM) o Lack of Cohesion in Methods –LCOM

Esta métrica relaciona todo método situado dentro de una clase con el acceso a uno o más atributos, en donde CCM es el número de métodos que acceden a uno o más de los mismos atributos. Es importante su utilización porque brinda una medida de la cohesión de una clase entre sus procedimientos y datos, una de las propiedades fundamentales para la validación de un diseño Orientado a Objeto.

Teniendo en cuenta que la obtención de los umbrales es bastante difícil de lograr debido a la falta de una norma o patrón que permita su aplicación se aplica la siguiente regulación considerando la relación existente entre la cantidad de métodos por clases y la probabilidad de la utilización de sus atributos:

Umbrales	Clasificación
0 - 21	Baja
22 - 41	Media
42 - 62	Alta

Aplicando estos umbrales a las clases diseñadas en el esquema de selectividad se obtiene como resultado:

Clase	Cant. Métodos	CMM
sel_criterioPeer	7	5
sel_grupoPeer	9	5
sel_condicionPeer	7	5
sel_canal_criterioPeer	2	2
sel_valoresPeer	2	2
Total		**19**

Tabla 3.2.1.1 Relación métodos - CMM

Se puede determinar que teniendo como resultado la clasificación Baja por estar debajo de la norma para este criterio, aplicando umbrales, es notable que existe muy poca complejidad en el diseño propuesto. Por

lo tanto no es necesario rediseñar las clases obtenidas y se ha logrado mantener un elevado nivel de cohesión.

3.2.2 Coupling Between Objects –CBO o Acoplamiento entre Objetos.

Esta métrica se especializa en conocer el número de clases a las cuales una clase determinada está asociada. Se da dependencia entre dos clases cuando una usa métodos o variables de la otra. Su utilización permite hacer referencia a otra de las propiedades que garantizan obtener un diseño adecuado, el acoplamiento.

El umbral utilizado para este tipo de métrica plantea que cada clase debe tener como único criterio de referencia 5 asociaciones, lo que garantiza que solo se logra un bajo o alto acoplamiento (16), teniendo 20 clases, se puede determinar que como media se utilizará el valor de umbral 100 para establecer el límite entre un bajo y un alto acoplamiento.

Clases	CBO
sel_criterioPeer	1
sel_grupoPeer	5
sel_grupo_relacionadoPeer	1
sel_condicionPeer	3
Total	**10**

Tabla 3.2.1.2 Relación métodos - CBO

Como resultado se obtiene que para este esquema existe un Bajo acoplamiento que garantiza que se realice mucho menos compleja la reutilización de las funcionalidades o de las clases. Este diseño propuesto permite aplicar la mantenibilidad como una de las funcionalidades de la calidad con mayor precisión, además de garantizar una mayor independencia de las clases que posibilita aplicar un correcto mantenimiento.

El presente diseño no presenta ningún tipo de herencia, pero Chidamber y Kemener proponen dos métricas para el trabajo con la herencia que no pueden ser aplicables, pero denotan precisamente que esta carencia, beneficia la calidad de la propuesta.

3.2.3 Árbol de Profundidad de Herencia (APH) o *Depth of Inheritance Tree* –DIT

Esta métrica expresamente se utiliza para esta propiedad. Se define como la longitud máxima desde el nodo hasta la raíz del árbol. Al no poder aplicarse directamente denota que las clases no poseen un nivel elevado de complejidad en su estructura lo que posibilita la mantenibilidad de la aplicación. De tener un diseño un alto valor de jerarquía es mayor la posibilidad de que deba heredar muchos métodos, lo que dificulta el reuso.

3.2.4 Número de Descendientes (NDD) o *Number of Children* –NOC

Se trata de las subclases que son inmediatamente subordinadas a una clase, que se denominan descendientes. Ante la ausencia de descendientes en esta propuesta no hay alguna probabilidad de que se pierda la esencia de la clase padre. A medida que crece el número de descendientes la abstracción representada por la clase predecesora puede verse diluida, lo que aumenta la posibilidad de que estas últimas clases no sean realmente miembros de las clases predecesoras, inconveniente que no se presenta en este esquema de diseño.

3.3 Métricas propuestas por Lorenz y Kidd

Existe otro grupo de métricas muy conocidas igualmente que fueron propuestas por estos ingenieros también basadas en las clases como la base para la obtención de métricas especializadas. La gran diferencia consiste en las definiciones hacia las cuales fueron dirigidas en su concepción.

- Tamaño
- Herencia
- Valores Internos
- Valores externos

Para la validación del diseño se toma en consideración las orientadas al tamaño.

¿Por qué se debe excluir el resto de las definiciones que también proponen métricas para el diseño?

De las categorías propuestas por Lorenz y Kidd [15] solo esta definición es aplicable al diseño presentado, porque permite trabajar con los valores introducidos en las clases y adentrarse en el código de los métodos de cada una. Mientras herencia se centra en una propiedad que no está presente y no puede ser medida, y los valores internos y externos reflejan otros intereses referidos al código y la reutilización que ya han sido abordados. Además de lograr ponderar los resultados a nivel de sistema lo que no es posible con el resto de las definiciones.

3.3.1 Métrica Tamaño de Clase

Esta métrica puede determinar el tamaño de una clase para conocer el grado de responsabilidad que existe entre cada una de ellas en un diseño específico. Pueden ser medidas a través de dos factores.

- El número total de operaciones (tanto operaciones heredadas como privadas de la instancia) que están encapsuladas dentro de la clase.
- El número de atributos (tanto atributos heredados como atributos privados de la instancia) que están encapsulados en la clase.

Utilizando los umbrales determinados por el CEIGE en el proyecto ERP[16] para un diseño se genera como resultado:

[15] **Lorenz y Kidd:** Ingenieros que en el año 1994 proponen otro grupo de métricas divididas en categorías y enfocadas en el código.
[16] **ERP:** siglas en inglés de Enterprise Resource Planning (Planificación de Recursos Empresariales), proyecto del Centro de Informatización para la Gestión de Entidades.

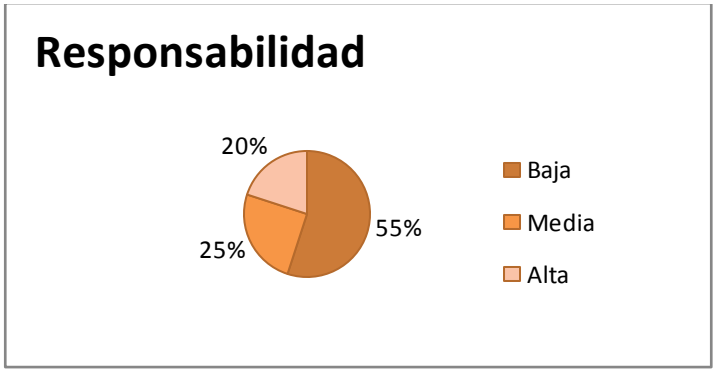

Figura 3.3.1.1 Resultado de la Responsabilidad en clases

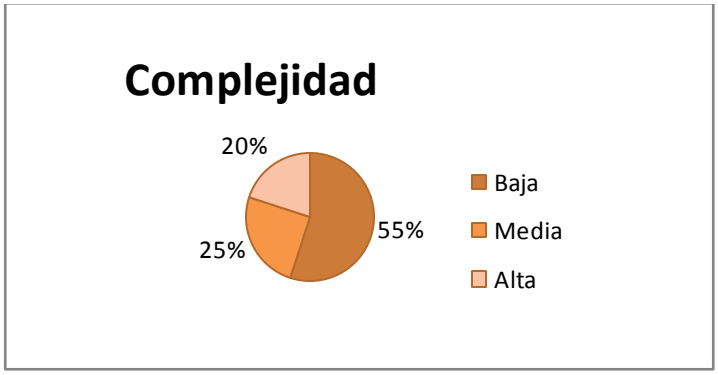

Figura 3.3.1.2 Resultado de la Complejidad en clases

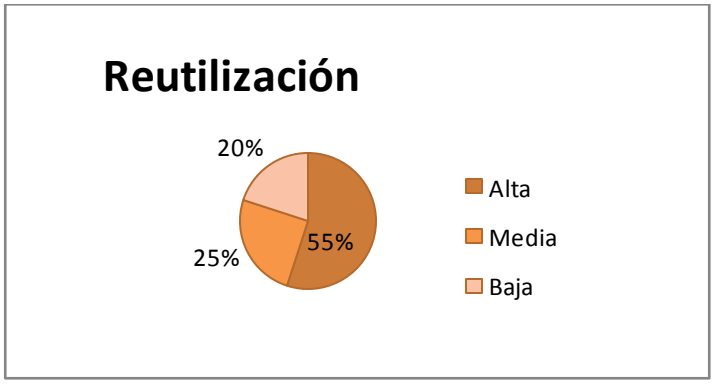

Figura 3.3.1.3 Resultado de la Reutibilidad en clases

Este resultado corrobora los obtenidos por métricas del grupo de Chidamber y Kemener. Este diseño presenta un 55% de baja responsabilidad de las clases evitando darle demasiada responsabilidad a la mayoría de las clases para poder aumentar la reutilización de las mismas beneficiadas por 55% de reutibilidad. Esta proporcionalidad logra evitar un alto porciento de complejidad, uno de los factores que impide el correcto funcionamiento de un diseño y de todas las medidas que se ejecutan para hacerlo mucho más manejable.

Esta es una métrica que es aplicable a todas las clases que tiene el esquema de selectividad, con lo que se puede decir que se utilizaron las 21 clases y todas sus respectivas operaciones dado que es necesario ponderar el resultado a nivel de sistema.

3.3.2 Métrica Relaciones entre Clases

Se trata de encontrar todas las relaciones que existan entre la clase A con una o varias clases B. Estas relaciones se definen por los atributos o métodos que utilice la clase A de la clase B al menos una vez para darle respuesta a alguna petición.

Para aplicarla fue necesario definir de todas las clases persistentes del diseño las que se apoyan en las operaciones o datos que no se encuentran dentro de sus responsabilidades. A su vez estas clases fueron analizadas para saber si continúan estableciendo relaciones con esta propia clase o con otras y en caso de que no existan tal relación se considera que tienen 0 relaciones pero se incluyen dentro de las clases analizadas.

Utilizando los umbrales determinados por el CEIGE en el proyecto ERP para un diseño se genera como resultado:

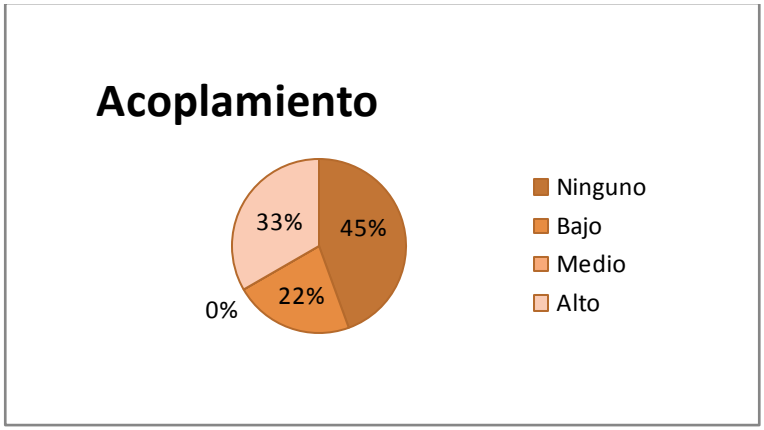

Figura 3.3.2.1 Resultado del acoplamiento en clases

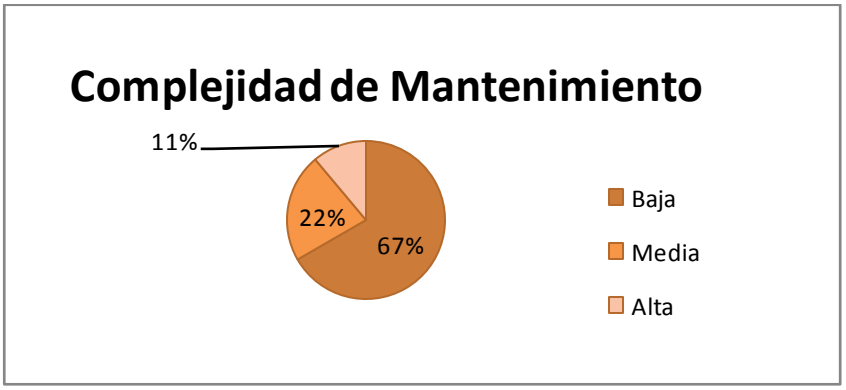

Figura 3.3.2.2 Resultado del mantenimiento en clases

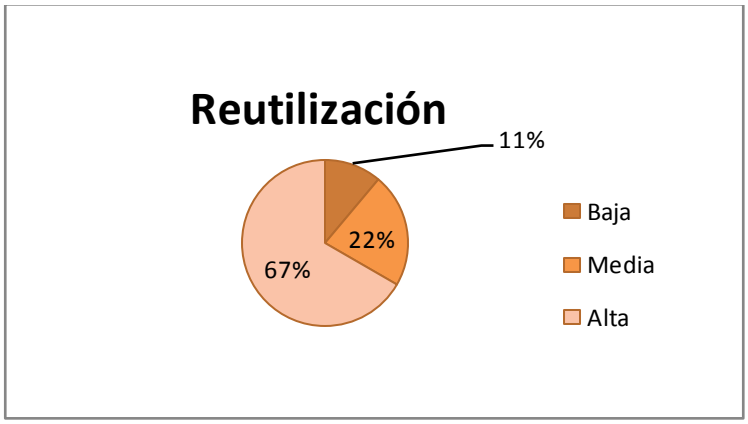

Figura 3.3.2.3 Resultado de la reutilización en clases

Este resultado corrobora que la propuesta del diseño no presenta problemas para poder realizar un adecuado funcionamiento. Teniendo un acoplamiento alto entre clases que no sobrepasa el 33% y un 67% porciento de baja complejidad de mantenimiento, es aplicable y posible poder darle reutilidad a las operaciones, con un 67% alto, un aspecto clave para facilitar el trabajo y mejorar la obtención de respuestas.

Es válido reconocer que resulta difícil determinar exactamente cuando se cruzan los límites entre un resultado y otro para determinar que se ha llegado a una respuesta satisfactoria o ineficiente. La aplicación de umbrales varía mucho dependiendo de qué se debe considerar para establecer parámetros a utilizar en una métrica. Los factores son cambiantes y por tal razón no existe una guía que defina cómo proceder a la hora de aplicar una métrica específica. Este es uno de los mayores temas de discusión mundialmente, por la dificultad que cobra estandarizar valores cuando se trata de medir propiedades tan diversas y ambivalentes.

3.4 Conclusiones parciales.

Se puede decir que se ha logrado obtener a través de la aplicación de métricas, que se basan en propiedades de las clases para cuantificar los resultados, un diseño que cumple con los requisitos necesarios para poder aplicar la mantenibilidad, como funcionabilidad de la calidad. Se ha logrado desde distintos puntos corroborar si el esquema que se presenta es eficiente y capaz de asimilar los procesos que permiten mantener a un diseño funcional. Muestra claras evidencias de que es posible continuar con su desarrollo hacia otras fases.

CONCLUSIONES

Una vez concluido el presente trabajo se puede confirmar que a través de los artefactos recibidos de fases anteriores, utilizando la metodología RUP como base, se ha logrado la modelación de la solución mediante los artefactos necesarios generados, por el cumplimiento de los siguientes aspectos descritos:

- ✓ Se pudo lograr el enriquecimiento del conocimiento propiciado por la fundamentación teórica de los temas sobre los procesos selectivos en los diferentes sistemas nacionales e internacionales y como lograr desarrollar la solución propuesta bajo las condiciones exigidas.
- ✓ Fue posible asimilar la arquitectura para el GINA definida en tres capas, la separación de la lógica del negocio de los componentes visuales y lograr una compatibilidad funcional dada las características del patrón arquitectónico escogido MVC.
- ✓ Se logró definir un diseño que cumpliera con todos los requisitos funcionales obtenidos para este proceso de Selectividad describiendo toda la documentación necesaria con la calidad requerida y nivel de precisión imprescindible para darle continuidad al proceso de desarrollo.

Con estos aspectos ya ejecutados se puede establecer que se ha dado cumplimiento al objetivo trazado y a las tareas definidas exitosamente.

RECOMENDACIONES

Es importante desarrollar estas ideas que quedan en espera de respuestas como recomendaciones:

- ✓ Llevar a cabo la implementación del diseño propuesto para dar solución a los problemas que hoy enfrenta el actual sistema para la aplicación de la Selectividad en la Aduana General de la República de Cuba.

- ✓ Que el proyecto Aduana continúe el uso de la metodología RUP con las adecuaciones que ha introducido siempre que se encuentren fundamentadas para el análisis y diseño de sus Sistemas de Información Computacional.

- ✓ Mantener un estrecho vínculo entra la Aduana General de la República de Cuba y el Centro de Informatización para la Gestión de Entidades para conocer los posibles cambios a surgir en el tratamiento de este proceso para futuras mejoras en la solución.

- ✓ Delimitar la información necesaria para mostrar en los diferentes reportes que posteriormente serán incorporados para la mejora del trabajo aduanero y sean incluidos en la aplicación.

- ✓ Continuar con la investigación de nuevas tecnologías informáticas para garantizar mejoras en futuras versiones del módulo Selectividad del Sistema Gestión Integral de Aduanas (GINA).

REFERENCIAS

1. **sitorus.** *TN21_Asycuda f_no_logo.doc.* 2007.

2. **Sánchez, Luis M.** *Aduana Siglo XXI.* 2007.

3. **Gvera.** *Aduana Nacional Bolivia Manual de Usuario SIDUNEA.* 2002.

4. The Standish Group. [En línea] www.standishgroup.com.

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

6. **LARMAN, C.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos. Primera Edición.* Prentice Hall : s.n., 1999.

7. es.wikipedia.org. [En línea] Mayo de 2011. http://es.wikipedia.org/wiki/M%C3%A9trica_del_software.

8. icesegurity.org. [En línea] Julio de 2007. http://icesegurity.org/conceptos/dhtml.htm.

9. *uji.es.* [En línea] Febrero de 2002. www3.uji.es/~alegre/informatica/.../php/php_manual_es.html.

10. plusformacion.org. [En línea] http://www.plusformacion.com/Recursos/r/Herramientas-CASE-para-proceso-desarrollo-Software#herramiena.

11. es.wikipedia.org. [En línea] http://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado.

12. es.wikipedia.org. [En línea] http://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational.

13. **Céspedes, Rafael Andrés.** *Diseño del módulo Control de Personas del Sistema Único de Aduanas.* La habana : s.n., 2009.

14. **Diaz, Luis.** *patrones.* 2006.

15. *Cap_6_Métricas_Sistemas_OO.* 2001.

16. **Negro, Ing. Pablo Ariel.** *Umbrales para métricas orientadas a objetos.* Buenos Aires : s.n., 2008.

BIBLIOGRAFÍA

1. **sitorus.** *TN21_Asycuda f_no_logo.doc.* 2007.

2. **Sánchez, Luis M.** *Aduana Siglo XXI.* 2007.

3. **Gvera.** *Aduana Nacional Bolivia Manual de Usuario SIDUNEA.* 2002.

4. The Standish Group. [En línea] www.standishgroup.com.

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

6. **LARMAN, C.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos. Primera Edición.* Prentice Hall : s.n., 1999.

7. es.wikipedia.org. [En línea] Mayo de 2011. http://es.wikipedia.org/wiki/M%C3%A9trica_del_software.

8. icesegurity.org. [En línea] Julio de 2007. http://icesegurity.org/conceptos/dhtml.htm.

9. *uji.es.* [En línea] Febrero de 2002. www3.uji.es/~alegre/informatica/.../php/php_manual_es.html.

10. plusformacion.org. [En línea] http://www.plusformacion.com/Recursos/r/Herramientas-CASE-para-proceso-desarrollo-Software#herramiena.

11. es.wikipedia.org. [En línea] http://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado.

12. es.wikipedia.org. [En línea] http://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational.

13. **Céspedes, Rafael Andrés.** *Diseño del módulo Control de Personas del Sistema Único de Aduanas.* La habana : s.n., 2009.

14. **Diaz, Luis.** *patrones.* 2006.

15. *Cap_6_Métricas_Sistemas_OO.* 2001.

16. **Negro, Ing. Pablo Ariel.** *Umbrales para métricas orientadas a objetos.* Buenos Aires : s.n., 2008.

17. IBM. [En línea] http://www.ibm.com/developerworks/websphere/library/techarticles/0306_perks/perks2.html..

18. ww.tienaduanas.blogspot.com. [En línea] http://tienaduanas.blogstop.com/2008/02/selectividad-y-anlisis-de-riesgo-en-las.html.

19. www.asycuda.org. [En línea] http://www.asycuda.org/pdf%20docs/sffunct.pdf..

20. www.gestiopolis.com. [En línea] http://www.gestiopolis.com/canales2/economia/sidunea.htm.

21. **Team Propel, Propel ORM .** www.propelorm.org. [En línea] 2007. http://www.propelorm.org..

22. wwww.todoexpertos.com. [En línea] http://www.todoexpertos.com/categorias/tecnologia-e-internet/comercio-electronico/respustas/252736/caracteristicas-oracle..

23. **H., Riehle D. y Züllighoven.** *Understanding and Using Patterns in Software Development.Theory and Practice of Object Systems.* 1996.

24. **Lea, Doug.** *Alexander Christpher: an Introduction for Object-Oriented Designers.* 1994.

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

26. **Nicodemos, Samanta B.** *Sistema María, necsaria evolución constante.* 2008.

27. **Olivares, M.C Juan Carlos.** *Patrones de diseño.* 2007.

28. **Olmedilla, Juan José.** *Revisión sistemática de Métricas de Diseño Orientado a objetos.* Madrid : s.n., 2005.

29. **Oracle., Documentación Oficial de.** *Replication Administrator?s Guide 11g Release 1 (11.1).* 2007.

30. **Quiroga, J.** *Introducción a los patrones de software.* Perú : s.n.

31. **R.J, ohnson R. E. y Helm.** *Design Patterns: Elements of Reusable Object-Oriented Software.* 1995.

32. **Rodríguez, Daniel.** *Medición en la orientación a objetos.* 2007.

33. **Userudla.** *anexo 2. Sistema informático Maríay sus ventajas operativas.* 2009.

34. **Veitía, Yojan Tang.** *Análisis y diseño del módulo Agenda Médica del Trabajo Diploma para Grado.* La habana : s.n., 2010.

35. **Yorio, Ing. Dario.** *Identificación y Clasificación de Patrones en el Diseño de Aplicaciones Móviles.Trabajo Diploma para alcanzar el Magister en ingeniería de Software. U.N.L.P.*

36. es.wikipedia.org. [En línea] http://es.wikipedia.org/wiki/Derecho_aduanero.

37. **Alexander, Christopher.** *An Introduction for Object-Oriented Designers.* 1994.

38. es.wikipedia.org. [En línea] http://es.wikipedia.org/wiki/Business_Process_Modeling_Notation.

39. *capítulo 8.*

40. **Rolando Hernández León, Sayda Coello.** *EL PARADIGMA CUANTITATIVO DE LA INVESTIGACIÓN CIENTIFICA.* La Habana : s.n., 2002.

41. **Martinto, MSc. Pedro Carlos Pérez.** *Diseño teórico de la investigación científica.* La Habana : s.n., 2006.

42. **Mike Perks.** www.computerworld.com. [En línea] IBM, Septiembre de 2003. http://www.computerworld.com/s/article/85198/Best_Practices_for_Software_Development_Projects?taxonomyId=11&pageNumber=2.

43. *Symfony 1.2, la guia definitiva.* 2088.

GLOSARIO

Aduana: oficina pública, establecida generalmente en las costas y fronteras, para registrar, en el tráfico internacional, los géneros y mercaderías que se importan o exportan, y cobrar los derechos que adeudan.

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

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, su objetivo es acelerar el proceso de automatizar o apoyar una o más fases del ciclo de vida del desarrollo de sistemas.

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

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

DAO: se trata de un patrón utilizado en Propel por su capacidad de abstraer y encapsular todos los accesos a la fuente de datos. Oculta completamente los detalles de implementación de la fuente de datos a sus clientes.

ERP: se trata del Sistema Cedrux para la planificación de recursos empresariales en una determinada empresa. Aplicación multiplataforma desarrollada por la Universidad de las Ciencias Informáticas en el Centro de Informatización para la Gestión Empresarial.

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.

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

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.

Sistema Gestor de Bases de Datos (SGBD): es el software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez.

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

ANEXOS