# Universidad de las Ciencias Informáticas

**Facultad 3**


# Título: Implementación del Módulo Selectividad para la Aduana General de la República de Cuba.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

**Autor:** Luis Alberto Pérez Blanco

**Tutores:** Ing. Yisel Rodríguez Pérez

Ing. Andy Fernández Garabote

La Habana.

Junio de 2013.

## PENSAMIENTO

*“Crecer sin que la altura me haga perder de vista lo importante. Y lo importante... Es la vida.”*

*Jorge Bucay*

# DECLARACIÓN DE AUTORÍA

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

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

**Autor**

_____

Luis Alberto Pérez Blanco

**Tutores**

_____

Ing. Yisel Rodríguez Pérez

_____

Ing. Andy Fernández Garabote

## DATOS DE CONTACTO


### Autor:

Luis Alberto Pérez Blanco

Universidad de las Ciencias Informáticas, La Habana, Cuba

Email: laperez@estudiantes.uci.cu

### Tutores:

Ing. Yisel Rodríguez Pérez

Universidad de las Ciencias Informáticas, La Habana, Cuba

Email: yrperez@.uci.cu

Ing. Andy Fernández Garabote

Universidad de las Ciencias Informáticas, La Habana, Cuba

Email: agarabote@.uci.cu

# AGRADECIMIENTOS

*A todos los que han influido en mi formación como profesional.*

*Luis Alberto Pérez Blanco*

# DEDICATORIA

*A mi familia, mis amigos y a la Revolución que me ha dado la posibilidad de realizar este sueño.*

*Luis Alberto Pérez Blanco*

# RESUMEN

La Aduana General de la República de Cuba tiene entre sus principales objetivos prevenir, detectar y enfrentar el fraude comercial, el contrabando y otras infracciones aduaneras. Para lograr este objetivo la Aduana necesita realizar controles cada vez más efectivos de los objetos definidos en cada proceso aduanero. Por tal motivo la dirección del Centro de Automatización para la Dirección y la Información toma como acuerdo realizar, en conjunto con la Universidad de las Ciencias Informáticas, el módulo de Selectividad, para que facilite la selección de objetos a controlar, utilizando las nuevas tecnologías definidas para el proyecto e integrándose éste, con los demás módulos del Sistema de Gestión Integral de Aduanas.

Para este propósito se efectúa un minucioso estudio del análisis y diseño previamente realizados. Se realiza un correcto seguimiento del modelo de desarrollo propuesto, generando los artefactos necesarios. Se presentan además, las pruebas realizadas al módulo, especificándose el resultado arrojado por las mismas, las cuales validan el total cumplimiento de los requisitos establecidos.

La puesta en marcha de este módulo brindará una alerta a los subsistemas que automatizan los procesos aduaneros ante un posible blanco o riesgo. También permitirá analizar los resultados alcanzados para determinar las necesidades de introducir nuevos criterios o modificar los existentes; gestionándose los tipos de control propuestos por los analistas, los cuales están conformados por criterios, condiciones y valores. El módulo facilitará el trabajo de los funcionales ya que ahorran tiempo a la hora de realizar los controles pertinentes, permitiendo tomar decisiones más seguras y efectivas.

PALABRAS CLAVE
Aduana, GINA, Selectividad.

# Tabla de Contenido

ÍNDICE

# INTRODUCCIÓN

En los últimos años las Tecnologías de la Información y las Comunicaciones (TIC) han tomado un auge considerable. Su dinámico desarrollo y gran proliferación se han ido apoderando de cada uno de los sectores de nuestras vidas propiciando que las profundas transformaciones en este campo se incrementen con mayor rapidez. Esto trae consigo que hoy en día las empresas se vean necesitadas de adquirir un sistema informático que contribuya al desarrollo y evolución de cada uno de sus negocios y que les permita estar a tono con el mercado internacional.

En la actualidad, los sistemas informáticos constituyen una herramienta de integración que facilita el desarrollo de las estrategias empresariales; han alcanzado tal nivel de dinamismo que les permite estar disponible a tiempo y adaptarse a los cambios que se precisen en la empresa. Además de cumplir con algo primordial en la empresa como es: informatizar las tareas de los procesos previstos, estos ayudan a tomar decisiones, crear conocimiento a la vez que incorporan la seguridad que garantiza la confidencialidad, integridad y disponibilidad de la información, así como los requerimientos legales que en los distintos países se exigen a la misma en ámbitos como: datos personales, transacciones financieras, etc.

La necesidad de integración y respuesta rápida en un contexto cada vez sometido a mayores cambios, ha dado origen a nuevas formas de desarrollar sistemas informáticos capaces de cumplir con estas necesidades, que son los llamados sistemas a la medida. Estos son especializados para un área y permiten mejorar el funcionamiento de la entidad.

En Cuba son muchas las organizaciones dedicadas a la producción de software dentro de las que se encuentra la Universidad de las Ciencias Informáticas (UCI) que como centro docente-productivo lleva un Sistema de Ciencia e Innovación Tecnológica (SCIT) que integra todos los factores, recursos y acciones de la institución en función de los objetivos propuestos. Es un modelo de universidad que vincula la formación, la producción y la investigación, que teniendo en cuenta los intereses del país y con el potencial tecnológico y humano disponible, está dirigido a la producción de software y servicios para informatizar la sociedad cubana. Además de automatizar entidades y empresas extranjeras, permitiendo el aporte de importantes ingresos al país.

# INTRODUCCIÓN

La UCI cumpliendo con la nueva política de informatizar la sociedad cubana está desarrollando desde el año 2010 un sistema informático aplicable a la Aduana General de la República de Cuba (AGR), el cual lleva como nombre Sistema de Gestión Integral de Aduanas (GINA), con el objetivo de informatizar todos sus procesos y funciones para un mejor trabajo.

La AGR, organización creada el 5 de febrero de 1963, constituye un órgano de control en frontera y en la actividad interna vinculada al comercio exterior; que garantiza la seguridad y protección de la sociedad socialista y de la economía nacional, así como la recaudación fiscal y las estadísticas del comercio exterior, a través del cumplimiento de las políticas estatales de competencia aduanera para el tráfico internacional de viajeros, mercancías y medios de transporte (1).

Para la Aduana, resulta una tarea difícil inspeccionar el gran volumen de mercancía existente en el área comercial, porque se habla de millones de mercancías en todo un año fiscal tanto en importación como en exportación; además de que no se cuenta con el personal necesario para inspeccionarla. Esto puede provocar atrasos en los procesos de despacho y posibles fraudes, poniendo en riesgo la seguridad nacional. Según plantea la Organización Mundial de Aduanas (OMA, por sus siglas en español) sólo se detectan en fronteras el 10% de los fraudes aduaneros (2). Un 90% de fraudes no detectados es un número realmente grande y se debe tener presente que hay que cumplir con los principios de agilidad y transparencia. Esto trae consigo que la Aduana necesite detectar o adivinar qué mercancía será inspeccionada en busca de posibles fraudes aduaneros. Cuando se analiza toda esta información surge una pregunta: ¿cómo determinar qué mercancía controlar y cuál no?

Para realizar un control en la parte comercial la AGR se apoya en un sistema informático de gestión implantado en 2007, su nombre es Sistema Único de Aduanas (SUA), actual sistema informático que utiliza la AGR para las operaciones de despacho. El mismo, presenta un módulo de Selectividad encargado de informar y determinar cuáles mercancías son las que se van a controlar o las que se van a sacar del flujo normal para inspeccionarlas detenidamente según amparen las resoluciones. La Aduana necesita realizar controles cada vez más efectivos sobre los objetos de control que entran y salen del país, dígase personas, mercancías, medios de transporte internacional, entre otros; para esto establece tipos de control con criterios, condiciones y valores preestablecidos por los analistas que permiten especificar las acciones a ejecutar en cada situación.

# INTRODUCCIÓN

El módulo de Selectividad existente solo podía ser aplicable a la Declaración de Mercancías, que resultaba lo necesario controlar en su momento. De esta manera siempre que fuera necesario aplicar un control a un nuevo objeto de control determinado, requería generar una selectividad particularizada que traía consigo demasiadas tareas para su implementación. La aplicación comenzó a ser inapropiada debido que no era capaz de abarcar los requerimientos mínimos estipulados bajo las nuevas condiciones.

A raíz de lo anteriormente descrito, surge como **problema a resolver**: ¿cómo lograr un sistema informático que permita gestionar la selectividad para la Aduana General de la República de Cuba partiendo de los artefactos obtenidos durante el análisis y siguiendo las pautas de arquitectura establecidas?

Con la expectativa de darle solución a este problema, se define como **objeto de estudio**: los procesos de selectividad, definiéndose como **campo de acción**: los procesos de selectividad en la Aduana General de la República de Cuba.

Como consecuencia de lo planteado anteriormente se traza el siguiente **objetivo general**: implementar un sistema informático para la gestión de los procesos de selectividad en la Aduana General de la República de Cuba.

En aras de lograr el cumplimiento del objetivo general, se establecen los siguientes **objetivos específicos**:

- ✓ Desarrollar un estudio del estado del arte para la elaboración del marco teórico de la investigación.
- ✓ Implementar la solución de manera que cumpla con las necesidades del cliente.
- ✓ Validar el módulo implementado mediante la ejecución de técnicas y métricas aplicables al desarrollo de la solución.

Durante la investigación se definió como **idea a defender** que: si se implementa un sistema informático para la gestión de los procesos de selectividad en la Aduana General de la República de Cuba, quedarán materializados los requisitos explícitos en el análisis de estos procesos; partiendo de los artefactos obtenidos durante el análisis y siguiendo las pautas de arquitectura establecidas.

# INTRODUCCIÓN

Para sustentar el desarrollo de la investigación y dar respuesta al problema planteado se utilizaron los siguientes **métodos científicos**:

## Métodos Teóricos

**Análisis histórico-lógico**: Este método permite estudiar de forma teórica, cómo ha estado evolucionando el módulo de Selectividad existente, así como el desarrollo de los distintos procesos que tienen lugar en este.

**Analítico-sintético**: Su utilización facilita el estudio y entendimiento de los procesos que se analizan. Determinando los aspectos esenciales y el arribo a conclusiones prácticas y teóricas que contribuyen adecuadamente a emitir la solución al problema planteado.

## Métodos Empíricos

**Observación**: Al utilizar este método se puede, en los distintos momentos de la investigación, obtener información de la gestión de los procesos relacionados con la selección de objetos a controlar en cada proceso de la AGR y así poder evaluar su manifestación.

**Entrevista**: Este método permite obtener información del módulo a implementar, (módulo de Selectividad), a través de conversaciones planificadas con el personal encargado de la AGR.

**Experimento**: Basándose en el módulo de Selectividad ya existente, permite que se adapten las características y funcionalidades del mismo al nuevo módulo de forma tal que cumpla con las especificaciones requeridas.

### El presente trabajo de diploma está conformado por tres capítulos:

### Capítulo 1: Fundamentación teórica y evaluación de las tecnologías

En este capítulo se abordan temas referentes al estado del arte de la investigación sobre el tema tratado, las tendencias y evolución de este en la actualidad mundial, nacional y específicamente en la UCI. Además se describe la utilización de las herramientas, técnicas y tecnologías para el desarrollo del módulo a implementar y su uso como guía para darle solución al problema presentado.

### Capítulo 2: Implementación de la solución

En este capítulo se realiza un análisis crítico y valorativo de la problemática existente, se abordan los aspectos relacionados con la construcción de la solución propuesta. Además se describe cómo el módulo implementado se integra al GINA sobre el framework Symfony describiéndose también el estándar de codificación seguido para la implementación de dicho módulo. Se analiza el funcionamiento de los algoritmos y las funcionalidades más importantes durante esta etapa, quedando de esta manera implementado el módulo de Selectividad.

### Capítulo 3: Validación y prueba

Se trata el tema de las pruebas a realizar, así como los tipos que se utilizarán para la validación del módulo. Se analizan los resultados obtenidos en dichas pruebas y se representan gráficamente para su mejor comprensión.

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se realizará el estudio del estado del arte, teniendo como principales basamentos el estudio realizado en las fases de análisis y diseño del módulo de Selectividad. Se describirá el proceso de selección de objetos a controlar por la AGR por parte de los especialistas del departamento de Lucha Contra el Fraude (LCF) y se abordarán una serie de conceptos asociados a este proceso para una mejor comprensión del problema. Se caracterizarán además las tecnologías, herramientas y lenguajes de programación que se utilizarán para el desarrollo de la solución.

## *1.1 Antecedentes*

Actualmente la función de las aduanas en el contexto mundial es facilitar el comercio, hacerlo más práctico, expeditivo y funcional, siendo creadas para el control del intercambio comercial entre países. Es por ello que las naciones desarrollan sus actividades comerciales a través de las aduanas, en el sentido de regular la entrada y salida de mercancías de su territorio. Las instituciones aduaneras desde sus inicios se dedican a proteger el comercio interior de cada país mediante impuestos, permitiendo evitar una caída de los indicadores económicos por el cobro de aranceles (3).

Para contribuir al continuo desarrollo y prestar un mejor servicio al mercado se han utilizado a escala mundial sistemas informáticos de gestión aduanal, los cuales cuentan con un módulo de Selectividad que determina e informa qué mercancía será controlada y cuál no.

En el año 2011 se realizó el análisis de dicho módulo donde se estudiaron algunos de los sistemas que incluían este proceso en los niveles mundial y nacional, con el objetivo de encontrar en estos características similares a las que debe poseer la solución a desarrollar, estudiar las funcionalidades que estos brindan y, fundamentalmente, analizar cómo se ejecutan en ellos los procesos de selección.

Para ello se tuvieron en cuenta sistemas líderes en la ejecución de los controles aduaneros en el mundo, como el Sistema Informático María (SIM) y el Sistema Aduanero Automatizado (SIDUNEA); donde el análisis de riesgo o control de selectividad es uno de los procesos de gran importancia. Luego de analizar estos sistemas el que más sobresale es el SIDUNEA ya que es un programa de administración aduanera modular, estándar y protegido por derechos de propiedad intelectual, a su vez es compatible con los

principales sistemas operativos y de administración de bases de datos. La modularidad del sistema le permite agregar módulos nuevos o avanzados en el momento que convenga. SIDUNEA es el software más difundido ya que es el más implantado en varios países por sus grandes beneficios. Una de sus grandes ventajas es la actualización de los datos de referencia sin necesidad de programación así como sus características integradas de seguridad, tales como la autenticación del usuario y la encriptación asimétrica. Ambos sistemas, SIDUNEA y María, presentan puntos en común como los canales, donde María usa tres y SIDUNEA cuatro. Apoyándose en las valoraciones analizadas SIDUNEA se convierte en una mejor opción ante los demás sistemas debido a las facilidades que posee (4).

En el ámbito nacional se analizaron las soluciones con las que cuenta la AGR para la ejecución de todos los procesos, controles, informes y gestión de la información y se decidió mantener el software existente en el SUA con las funcionalidades que posee, añadiendo otras que permitan un funcionamiento óptimo (4). Para una mejor comprensión del módulo a desarrollar, se describe a continuación cómo se realizaría el proceso de selección de objetos a controlar en el GINA.

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 al perfeccionamiento de los procesos que intervienen en la selección de objetos. La dinámica actual necesita que el tratamiento entre el personal aduanero y los objetos de control, se realice de forma ágil para que las mercancías lleguen en el menor tiempo posible a su destino final, 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 objeto analizado.

En la aduana existen personas con amplia capacidad de análisis y mucha experiencia acumulada en el tiempo (son los llamados analistas de selectividad), estas personas definen los objetos de control con sus características y establecen los criterios de búsqueda con sus condiciones y valores. Esta información la utiliza el sistema y aplica una especie de filtro a los objetos que se especifiquen y emite un canal según sea el caso. Los canales varían de color en dependencia del grado de procesamiento que requiera y una vez otorgado, indican la medida a aplicar (ver Tabla 2.2.1).

Como GINA pretende ser una solución integral de aduanas debe contar con un módulo de Selectividad que mejore el que se encuentra en el SUA además de incorporar más valores agregados en cuanto a

funcionalidades. Por tanto se define que el GINA debe tener su propio módulo el cual se integrará y retroalimentará con los demás subsistemas. La selectividad proyectada para el GINA está pensada de forma genérica, la cual pueda gestionar lo mismo una Declaración de Mercancías (DM), que una Persona o un Medio de Transporte Internacional (MTI).

Para lograrlo el sistema presenta un flujo para crear los objetos de control según las características físicas del mismo. Y cada objeto de control puede ser controlado en un momento dado, conocido como tipo de control de selectividad. Entonces un mismo objeto físico puede tener distintas respuestas según el tipo de control.

A continuación se describe un flujo de las actividades que comienza por una petición de un tipo de control realizada por los analistas de las distintas áreas de la AGR. De este tipo de control propuesto se analiza su necesidad, siendo rechazado o aprobado en el sistema por el jefe de selectividad. Luego si fue aceptado se debe configurar por las personas que más conocimiento técnico-teórico tengan, tanto del sistema como de los procesos del negocio, teniendo en cuenta la información que puede ser relevante o no para el objeto de control en cuestión. El tipo de control cuenta con criterios, condiciones y valores. Los criterios son un conjunto de reglas que otorgan un canal a un objeto determinado, están compuestos por condiciones que buscan coincidir en alguno de los valores indicados por los analistas que los configuran.

En la siguiente imagen se muestra cómo se gestionan los criterios pertenecientes a un tipo de control determinado:

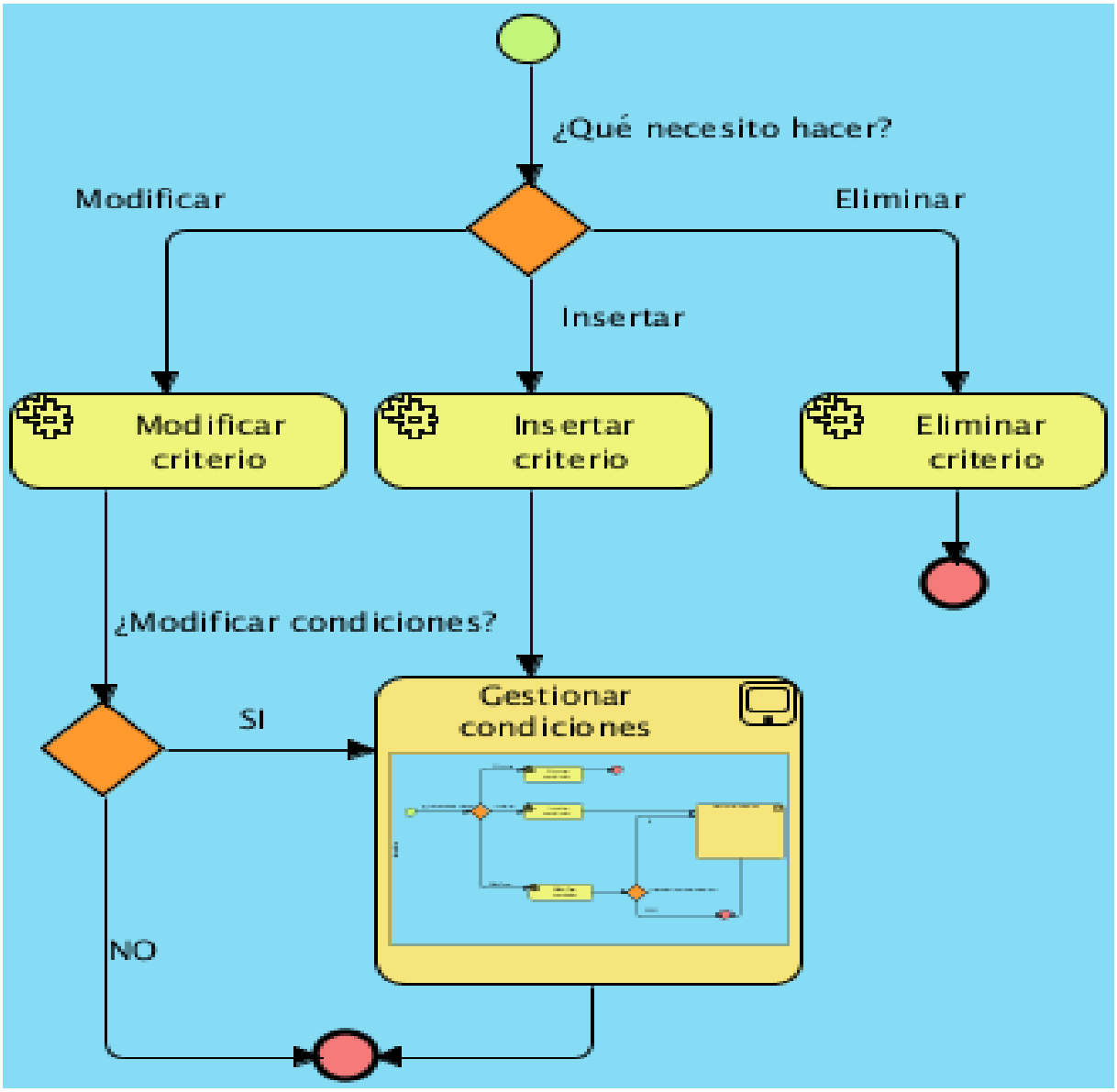


**Figura # 1.1.2 Modelo de Negocio gestionar criterio**

Estos criterios establecidos son supervisados a través del pre-monitoreo, funcionalidad que permite comprobar la fiabilidad de los mismos. El módulo de Selectividad propuesto presenta además reportes estadísticos los cuales se apoyan en las características que más se manejan en los objetos de control mencionados para brindar los resultados.

# CAPÍTULO 1

La propuesta de solución realizada en el diseño tiene como garantía que será totalmente configurable a cualquier objeto de control y su servicio puede ser consultado por cualquier subsistema del GINA. Logrará además un flujo de trabajo ordenado y funciones que hacen posible verificar la importancia o relevancia de un criterio determinado.

## *1.2 Modelo de desarrollo*

Para el desarrollo de cualquier producto de software se realizan varias tareas hasta llegar al producto final. Es por ello que se han realizado varias metodologías de trabajo para organizar las actividades a realizar en cada disciplina. El modelo de desarrollo es una guía que ayuda a definir el orden en el que se harán las cosas en los proyectos, provee requisitos de entrada y de salida para cada una de las actividades definidas por fase.

Para el departamento de Soluciones para la Aduana al cual pertenece el proyecto GINA del Centro de Informatización de la Gestión de Entidades (CEIGE), se realizó un análisis profundo con los participantes de todos los departamentos para estandarizar las actividades y artefactos generados en cada disciplina dando lugar a la actualización del modelo de desarrollo para el centro. Dicho modelo está orientado a componentes, define claramente las responsabilidades de cada uno de los roles involucrados en el desarrollo de la solución, el flujo de actividades y los artefactos que deben ser generados por cada uno de los roles involucrados en el proceso de desarrollo de software.

El mismo fue aprobado por el equipo de profesionales del departamento donde será utilizado para realizar la implementación y generar los artefactos definidos en la fase de implementación. En este se define el ciclo de vida de los proyectos del departamento, incluyendo las fases de desarrollo, los roles que intervienen durante el desarrollo de cada producto y las responsabilidades que tiene cada rol. Además, se definen todas las actividades que se deben realizar en cada una de las fases del desarrollo; así como los artefactos que se deben generar en cada una (5).

El objetivo de este trabajo está centrado en la implementación del módulo Selectividad, los artefactos que se deben generar en el transcurso del presente trabajo se obtienen en dependencia de la actividad que se realiza. A continuación se enuncian las actividades que se deben realizar en la fase de implementación y los artefactos que se deben generar en estas.

| | Artefactos |
|---|---|
| Definir estándar de codificación. | Estándar de codificación. (En este caso se aplicará el estándar definido para el desarrollo del GINA) (7). |
| Implementación y aplicación de pruebas internas. | Artefactos de implementación. Dígase ficheros de código y algoritmos de implementación.<br>Resultados de las pruebas internas (no conformidades y diseños de casos de prueba). |

**Tabla # 1.2.1 Actividades y artefactos de las disciplinas implementación y pruebas**

## *1.3 Tecnologías, Lenguajes y Herramientas para el Desarrollo Web*

La selección de las tecnologías y herramientas a utilizar para el desarrollo de la solución, es un proceso de gran importancia en el que se deben tener en cuenta las características y los beneficios que brindan a la hora de ser utilizadas. Las tecnologías, lenguajes y herramientas que se usarán para el desarrollo de la solución fueron definidos con anterioridad debido a que el módulo Selectividad está comprendido dentro del sistema GINA (8).

Se estableció como lenguaje de programación el PHP 5.0.4 o superior, luego de realizar una caracterización y comparación de los principales lenguajes de programación web que más se utilizan internacionalmente. Para esta selección se tuvo en cuenta la petición de los analistas de la Aduana cubana que planteaban que se usara este lenguaje, puesto que han adquirido con el paso del tiempo experiencia en el trabajo con este, además en la AGR cuentan con los recursos y el conocimiento necesario para el manejo y soporte de sistemas creados en este lenguaje (8).

# CAPÍTULO 1

### *Lenguaje de Programación: PHP*

PHP es un lenguaje interpretado de propósito general, ampliamente usado y que está diseñado especialmente para el desarrollo web. Entre sus principales características se destaca que es libre, multiplataforma y cuenta con muchas funcionalidades de manera nativa. Tiene comportamiento modular, brinda la posibilidad de adicionar nuevas funcionalidades a través de nuevos módulos. Permite la conexión a varios sistemas de bases de datos brindando múltiples funcionalidades, destacando en este aspecto la conectividad con PostgreSQL y MySQL, que son gestores muy utilizados en el desarrollo de aplicaciones web (9). Cuenta con amplia documentación, tanto en su página oficial como en distintos foros y publicaciones. Permite el empleo de Programación Orientada a Objeto, no requiere que se le especifique el tipo de datos de las variables y maneja excepciones a partir de la versión 5.0.

Entre las desventajas que posee este lenguaje cabe destacar que en ocasiones por sus características promueve la creación de código desordenado y complejo de mantener.

### *Framework Arquitectónico: Symfony*

Un aspecto fundamental a definir dentro de la arquitectura era decidir que marco de trabajo o framework de desarrollo, como también se le conoce, se iba a utilizar para el desarrollo del sistema. Para esto era necesario dominar el concepto, estudiar a fondo las ventajas que ofrece el desarrollo de un sistema utilizando uno en específico.

Existen diferentes frameworks para diferentes propósitos, algunos orientados al desarrollo de aplicaciones web, dentro de este grupo se encuentra Symfony que fue el que se definió como framework arquitectónico a utilizar para llevar a cabo la implementación del módulo de Selectividad y lograr así un sistema más completo, potente y flexible. Para la selección de este se tuvieron en cuenta algunos aspectos importantes basándose en la comparación de diferentes framework existentes en la actualidad, entre los que se analizaron se encuentran: Kumbia, CakePHP y Zend framework (8).

Symfony permite el desarrollo del sistema basándose en el patrón Modelo Vista Controlador (MVC), con Propel para manejar el modelo de la arquitectura y Creole para manejar la abstracción de la base de datos. Creole se utilizó mientras se usaban las versiones de Symfony 1.0.x; pero luego de la migración a Symfony 1.2.8 se comienza a trabajar con PDO y sus extensiones pdo_oci para Oracle y pdo_pgsql para

Postgres SQL. El framework escogido está diseñado para optimizar el desarrollo de las aplicaciones web, separando la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación (10).

Symfony es compatible con la mayoría de los gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft; permitiendo una mayor portabilidad del sistema haciéndolo independiente de cualquier gestor de base de datos que se utilice. Se puede ejecutar tanto en plataformas Unix y Linux, como en plataformas Windows (8). A continuación se muestran algunas de sus características:
Symfony está diseñado para que se ajuste a los siguientes requisitos:

- ✓ Fácil de instalar y configurar en la mayoría de las plataformas.
- ✓ Es independiente del sistema gestor de bases de datos.
- ✓ Es sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- ✓ Sigue la mayoría de las mejores prácticas y patrones de diseño para la web.
- ✓ Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.

### *Sistema Gestor de Base de Datos: Oracle*

Como sistema gestor de base de datos se definió usar Oracle en su versión 11g o superior, tiene entre sus características más notables que es considerado uno de los más completos y potentes, destacado por su soporte de transacciones, estabilidad, escalabilidad y ser multiplataforma. Es seguro, pues cuenta con un proceso de sistema de respaldo y recuperación de información. Soporta Data Warehouse por lo que facilita el acceso a la información y brinda mayor versatilidad (6).

**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 escala 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.

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 (6).

### *Framework para la Interfaz de Usuario: ExtJS*

Para manejar la concepción de los paradigmas de la Web 2.0 y la seguridad de las aplicaciones se definió el framework para JavaScript ExtJS permitiendo la creación de efectos visuales y validaciones del lado del cliente.

ExtJS brinda la posibilidad de utilizar un gran número de componentes que facilitan el trabajo de los desarrolladores y que mejoran notablemente la calidad de las aplicaciones desde el punto de vista del usuario final. Una característica importante de este es que una vez que el navegador haya interpretado el código JavaScript el usuario se sentirá trabajando en el sistema como si fuera una aplicación de escritorio. Esto está dado porque la transacción de información se realiza a partir de comunicaciones asincrónicas (8).

# CAPÍTULO 1

ExtJS cuenta con un conjunto de librerías JavaScript que permiten desarrollar aplicaciones web interactivas integrando tecnologías del desarrollo web como AJAX (Asynchronous JavaScript And XML), JSON (JavaScript Object Notation) y DHTML (Dynamic HTML) (11). Algunas de las tecnologías antes mencionadas que se integran a ExtJS serán descritas a continuación:

### *Lenguaje de Programación: JavaScript*

Se trata de un lenguaje de programación del lado del cliente porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos es el lenguaje de programación del lado del cliente más utilizado. Permite validar los datos enviados en los formularios, detectar navegadores y mejorar el diseño, su fácil aprendizaje lo hace un lenguaje muy demandado (12).

Es un lenguaje basado en acciones que posee menos restricciones. Gran parte de la programación en este lenguaje está centrada en describir objetos, escribir funciones que respondan a movimientos del mouse, aperturas, utilización de teclas, cargas de páginas entre otros (13).

### *Notación de Objetos de JavaScript*

Conocido como JSON por sus siglas en inglés JavaScript Object Notation. Es un formato ligero de intercambio de datos, basado en un subconjunto del lenguaje JavaScript. Es un formato de texto que es completamente independiente del lenguaje de programación. Estas propiedades hacen que JSON sea ideal para el intercambio de datos (14).

JSON está constituido por dos estructuras:

- ✓ Una colección de pares de nombre/valor: esto es conocido como un objeto, registro, estructura, diccionario, lista de claves o un arreglo asociativo.
- ✓ Una lista ordenada de valores: esto se implementa como arreglos, vectores, listas o secuencias.

### *Sistema de Control de Versiones*

Para el control de versiones en el desarrollo de la solución se utilizará la herramienta Subversion en su versión 1.5.1. La cual contribuye a la calidad de un buen producto y corre en plataformas libres. Con este sistema se puede registrar el historial de todos los archivos fuentes y documentos. Mediante sentencias de comandos se puede proporcionar el registro de todas las operaciones realizadas sobre un documento en específico y logra hacer cumplir las políticas de seguridad establecidas. Permite que las unidades de trabajo funcionen como un solo equipo de desarrollo aunque no necesariamente trabajen como tal. El historial de todas las versiones se almacena en un único servidor de versiones (15).

### *Plataforma de Desarrollo*

El GINA es un sistema web que se desarrolla sobre el sistema operativo GNU/Linux, gracias a las tecnologías que se usan para su desarrollo es multiplataforma, permitiendo funcionar tanto en distribuciones GNU/Linux como en Windows (8). Al estar contenido el módulo Selectividad dentro del GINA, cumple también con esta característica.

### *Entorno Integrado de Desarrollo (IDE)*

NetBeans es el entorno a utilizar para el desarrollo de la solución, es un entorno modular y basado en estándares, escrito en el lenguaje de programación Java. Consta de un IDE de código abierto y gran variedad de funciones.

## *1.4 Buenas Prácticas para el Desarrollo Web*

Luego de seleccionadas todas las herramientas, tecnologías y lenguajes de programación que se utilizarán para el desarrollo del módulo Selectividad es necesario adentrarse en las características y pautas del desarrollo web con las que debe cumplir el módulo.

Se puede mencionar una serie de problemas que por lo general se presenta durante el desarrollo de aplicaciones web. Los cuales pueden hacer que fracase o no el desarrollo, que luego de realizado el producto este no cuente con la calidad requerida por el cliente. A continuación se enumeran una serie de problemas identificados y las buenas prácticas que se pueden aplicar:

1. **Problemas para ver el diseño en varios navegadores:** en ocasiones se presentan problemas para la visualización del diseño, estos se pueden ver alterados de un navegador a otro, incluso los mismos navegadores en diferentes sistemas operativos. Uno de los motivos que causa esto son las tres hojas de estilos que se utilizan (el del navegador, el del usuario y el del diseñador).

**Buena práctica:** se debe comprobar el diseño en diferentes navegadores (16).

2. **Uso de los sistemas para el control de versiones:** puede ocurrir que por diversas razones, por desconocimiento o falta de experiencia, práctica o cualquier otra razón, no se controlen, almacenen y gestionen correctamente las versiones del sistema, ocurre que cada cual tiene en su poder una versión distinta del sistema que se desarrolla y esto provoca conflictos a la hora de realizar la integración.

**Buena práctica:** se recomienda utilizar un sistema para el control de las versiones, sea Subversion, Git, Cervicia o cualquier otro.

**3. Organización y planificación:** para trabajos de gran envergadura, proyectos realmente importantes no se pueden llevar a cabo cada actividad sin tener un orden o viceversa, tratar de planearlo todo exhaustivamente limitará la espontaneidad y la inspiración.

**Buena práctica:** para un correcto desarrollo se deben tener las ideas claras, conocer el diseño a groso modo, la estructura en la que se organizan los archivos (16), etc.

## *1.5 Estándares de Codificación*

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, este debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un proyecto de software es necesario establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Cuando el proyecto de software incorpore código fuente previo o bien cuando realice el mantenimiento de un sistema de software creado anteriormente, el estándar de codificación debería establecer cómo operar con la base de código existente (17).

El mejor método para asegurarse de que un equipo de programadores mantenga un código de calidad es establecer un estándar de codificación sobre el que se efectuarán luego revisiones del código de rutinas. La incorporación de técnicas de codificación generalmente no afectan la funcionalidad del sistema, estas contribuyen a una mejor compresión del código fuente. En general una técnica de codificación no pretende formar un conjunto inflexible de estándares de codificación. Más bien intenta servir de guía en el desarrollo de un estándar de codificación para un proyecto específico de software (17). El estándar de codificación a utilizar para la implementación de módulo de Selectividad fue definido por la dirección del departamento de Soluciones para la Aduana desde sus inicios y se encuentra registrado en el documento “Propuesta de un estándar de codificación” (7).

## *1.6 Pruebas de Calidad de Software*

En la cadena de valor del desarrollo de un software, el proceso de prueba es clave a la hora de detectar errores o fallas. Las pruebas de software son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada. Las aplicaciones de software han crecido en complejidad y tamaño, por consiguiente también en costos, por ello es crucial verificar y evaluar la calidad del software para minimizar el costo de su reparación. El proceso de prueba es un proceso técnico especializado de investigación que requiere de profesionales altamente capacitados en lenguajes de desarrollo, métodos de pruebas y herramientas especializadas (18).

### *Tipos de prueba*

A continuación se definen los tipos de prueba que se aplicarán para garantizar la calidad y el buen funcionamiento del módulo Selectividad.

**Pruebas de Caja Blanca:** la prueba de caja blanca se basa en un examen cercano al detalle procedimental. Se prueban las rutas lógicas del software y la colaboración entre componentes, al proporcionar casos de pruebas que ejerciten conjuntos específicos de condiciones, bucles o ambos. Se realizan sobre las funciones internas de un módulo en concreto (19).

Entre las técnicas de caja blanca que se pueden realizar se encuentran:

- ✓ Prueba de Condición: Método de diseño de casos de prueba que ejercita las condiciones lógicas contenidas en el módulo de un programa.

- ✓ Prueba de Flujo de Datos: Se seleccionan caminos de prueba de un programa de acuerdo con la ubicación de las definiciones y los usos de las variables del programa.

- ✓ Prueba de Bucles: Técnica de prueba de caja blanca que se centra exclusivamente en la validez de las construcciones de bucles.

- ✓ Prueba del Camino Básico: Esta permite obtener una medida de la complejidad lógica del código de cada método, programa o módulo dado. La idea es derivar casos de prueba a partir de un conjunto dado de caminos independientes que existen en la codificación por los cuales puede circular el flujo de control. Es además una de las más eficientes en cuanto a cobertura de código, pues logra que se ejecuten todos los bucles en sus límites operacionales (20).

Estas pruebas se pueden aplicar en varios niveles, su cometido es comprobar los flujos de ejecución dentro de cada unidad, las unidades pueden ser (funcionalidades, clases, módulos, etc.).

**Pruebas de Caja Negra:** este tipo de prueba se realiza sobre las interfaces del software con el objetivo de asegurar la completitud de los requisitos funcionales, parte fundamental en el desarrollo de un software.

Permite encontrar:

- ✓ Funciones incorrectas o ausentes.

- ✓ Errores de interfaz.

- ✓ Errores de validación.

- ✓ Errores de ortografía.

Entre las técnicas de caja negra que se pueden aplicar se encuentran:

- ✓ Partición Equivalente: Esta técnica divide el dominio de entrada de un programa en clases de datos, a partir de las cuales deriva los casos de prueba. Cada una de estas clases de equivalencia representa a un conjunto de estados válidos o inválidos para las condiciones de entrada (21).

- ✓ Tabla de decisión: Su utilización ofrece salidas, a los casos de la prueba, basadas en la decisión. Se compone de una tabla que contiene 4 cuadrantes, las condiciones, las entradas de estado, las declaraciones de acción y las entradas de acción. Precisamente, todas las condiciones se ponen a prueba para comprobar si el resultado deseado se alcanza después de haber probado diferentes entradas (21).

## *1.7 Conclusiones Parciales*

En el presente capítulo se identificaron y abordaron aspectos importantes que se tendrán en cuenta para el desarrollo de la solución. Partiendo de un análisis de los antecedentes del tema se tomaron experiencias del comportamiento y las funcionalidades que ofrecen los sistemas que incluyen un módulo de Selectividad.

Se describió además el proceso de selección de objetos a controlar permitiendo una mejor comprensión del mismo y sirviendo como base para el desarrollo del módulo a implementar. Se definieron las herramientas, técnicas y tecnologías a utilizar para solucionar el problema planteado. Además se analizó el modelo de desarrollo a seguir, sirviendo este como guía para las actividades a realizar en la implementación teniendo en cuenta sus características y adaptabilidad.

Se concluye además, aplicar pruebas a nivel de sistemas para evaluar la calidad del software, específicamente utilizando la técnica del camino básico dentro de las pruebas de caja blanca y la técnica de partición equivalente en las pruebas de caja negra.

# CAPÍTULO 2: IMPLEMENTACIÓN DE LA SOLUCIÓN

En este capítulo se muestra una panorámica de como la solución propuesta dará respuesta al problema planteado del capítulo anterior, se describen elementos fundamentales a tener en cuenta para la implementación de dicha solución. Se analiza exhaustivamente el análisis y diseño realizados anteriormente, teniéndose en cuenta los cambios necesarios de acuerdo a la implementación que se desea realizar. También se describen las funcionalidades y algoritmos más importantes para una mejor comprensión del módulo a implementar.

## *2.1 Valoración del Análisis y Diseño*

Luego de obtener los resultados del análisis y diseño descritos en los trabajos de diploma titulados “Análisis del módulo Selectividad para el Sistema Gestión Integral de Aduanas” y “Diseño del módulo Selectividad para el Sistema de Gestión Integral de Aduanas” respectivamente, se puede proceder a la implementación de la solución. Para ello se estudia el diseño elaborado examinando críticamente cada una de las funcionalidades necesarias, clases persistentes relacionadas con las mismas y artefactos obtenidos. Este estudio garantiza la efectividad en el registro de información y comprueba que lo expuesto en las descripciones de las funcionalidades sea claro y posible de implementar. Se discuten las inconsistencias encontradas con el objetivo de realizar los cambios necesarios para obtener un modelo de datos acorde a la solución pensada. Finalmente se determina si el diseño es el indicado y se puede continuar con las demás actividades que dan paso a la implementación.

Durante la disciplina de análisis del módulo de Selectividad fueron definidos de manera clara los requisitos funcionales que debe cumplir el software para que satisfaga las expectativas del cliente y brinde mayores prestaciones que los antiguos sistemas informáticos. Quedó reflejado el modelo de procesos del negocio así como la descripción de cada uno de los subprocesos y actividades, permitiendo esto adentrarse y comprender los procesos efectuados en el área LCF, específicamente los relacionados con Selectividad. Entre los artefactos obtenidos se encuentra el modelo conceptual, las reglas del negocio, las especificaciones y descripciones de cada uno de los requisitos funcionales. Se identificaron un total de 20 requisitos, 14 de estos de complejidad alta. Se actualizaron etiquetas referentes a elementos del diseño de los prototipos de interfaz de usuario, de acuerdo a especificaciones del cliente para su mejor

interpretación (ver Anexo 1). Además se agregaron 12 reportes establecidos por el personal de la AGR para lograr el correcto y completo funcionamiento del módulo.

Partiendo del estudio de los artefactos del análisis se logró desarrollar el diseño del módulo. La utilización de los artefactos que se generan en la disciplina de diseño resulta una actividad esencial para el desarrollo de todo sistema informático. Durante esta disciplina se desarrollaron los diagramas de clases del diseño con estereotipos web, diagramas de secuencia orientados a actividades, el diseño de la base de datos, diagrama de despliegue, entre otros; sirviendo estos de entrada para el flujo de implementación. Estos resultados permiten un mayor entendimiento del módulo y menos demora para llegar a la solución final. Los diagramas utilizados están definidos y representados dentro del modelo del diseño del módulo Selectividad (22), uno de los artefactos generado en la disciplina de diseño.

## *2.2 Funcionalidades y Algoritmos más Importantes*

Para la realización del módulo se desarrollaron interfaces amigables acorde a lo requerido por el personal de la Aduana, que serán los encargados de trabajar con las mismas. A continuación se presenta la interfaz principal del módulo (ver Figura 2.2.1). En la misma se muestran los vínculos hacia todas las funcionalidades con las que cuenta y cumple el módulo desarrollado.


**Figura # 2.2.1 Interfaz principal de Selectividad**

A continuación se realiza la descripción de las funcionalidades y algoritmos principales para el correcto funcionamiento del módulo que se desarrolla.

### *Funcionalidad Gestionar Objeto de Control*

Esta funcionalidad permite definir y modificar los parámetros o atributos del objeto de control que será inspeccionado. Los parámetros serán definidos por niveles, en el nivel 1 (ver Figura 2.2.2) se especifica el nombre, nombre a mostrar y la descripción por la cual va a ser identificado el objeto de control en proceso. En el nivel 2 (ver Figura 2.2.3) se especifica además el tipo de dato, el nombre de la tabla donde se encontrará el atributo correspondiente, el nombre del campo en dicha tabla y se especificará también la tabla con la cual tiene relación. En la columna "Principal" se especificará con "SI" al atributo principal o llave del objeto de control en cuestión para conocimiento del sistema.

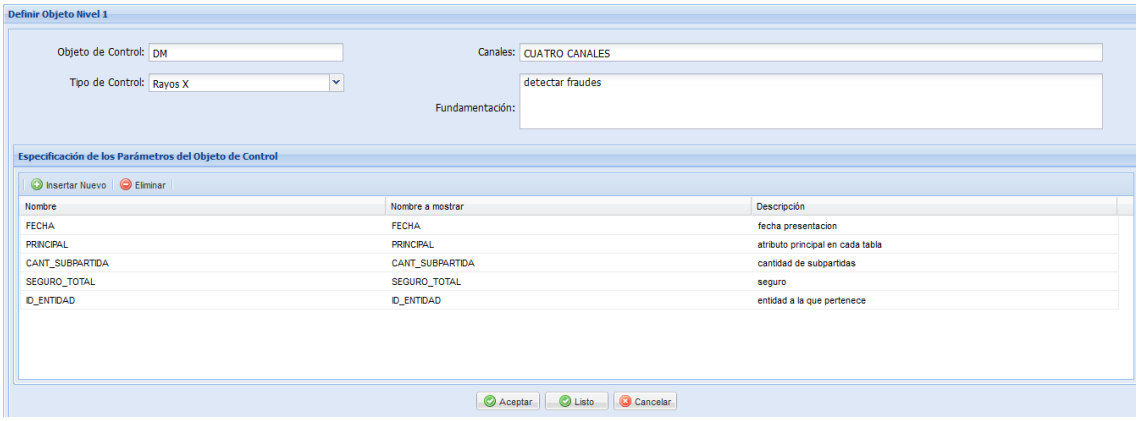


**Figura # 2.2.2 Interfaz definir objeto nivel 1**

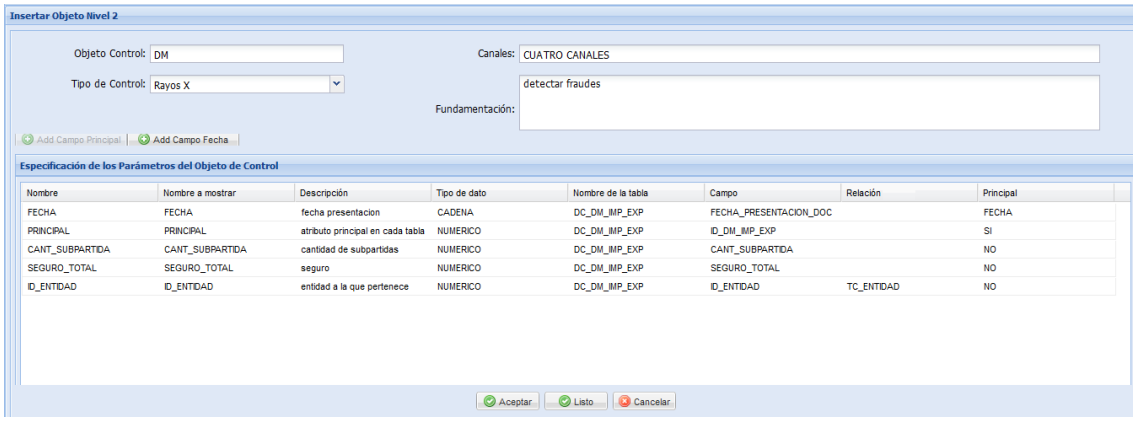


**Figura # 2.2.3 Interfaz definir objeto nivel 2**

### *Funcionalidad Gestionar Criterios*

La gestión de los criterios de control constituye una actividad esencial, a partir de esta se especificarán los parámetros por los cuales la AGR determinará la actitud a tomar ante un objeto de control determinado. En esta funcionalidad se gestionan los criterios necesarios para cada tipo de control, definiéndose todos los parámetros de cada criterio a insertar, fecha en que será puesto vigente, objetivos, alcance, porciento por canal (ver Figura 2.2.4) y se establecerán las condiciones con valores definidos por los analistas que permitirán las acciones a ejecutar en cada situación (ver Figura 2.2.5). Cuando un criterio esté insertado pasará a estar establecido, pudiendo este posteriormente ser modificado o eliminado (ver Figura 2.2.6).

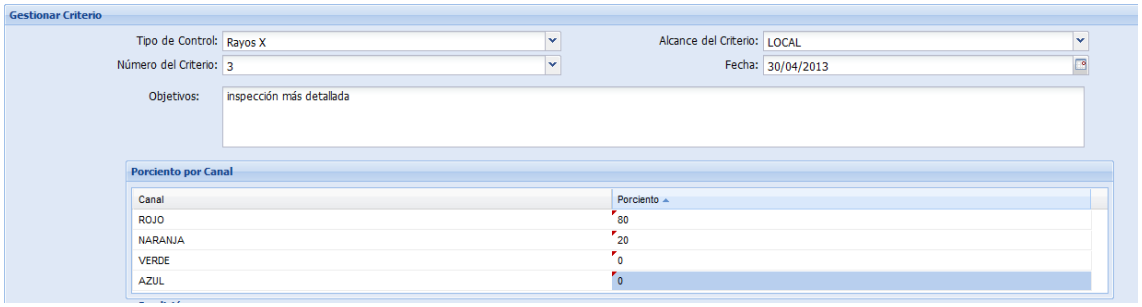


**Figura # 2.2.4 Interfaz gestionar criterio**

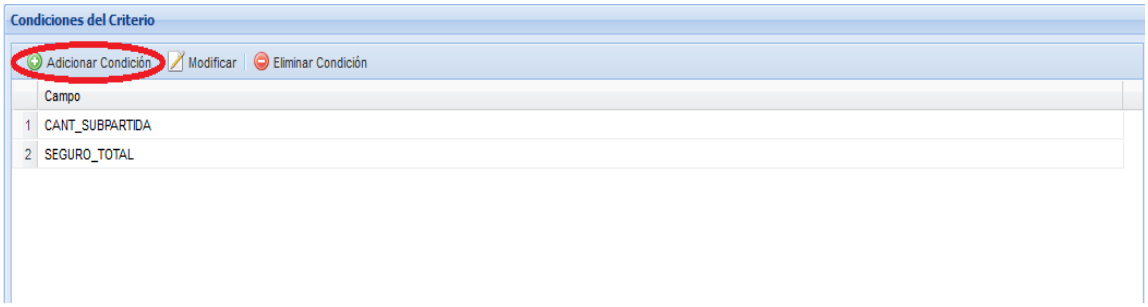


**Figura # 2.2.5 Interfaz insertar condiciones**

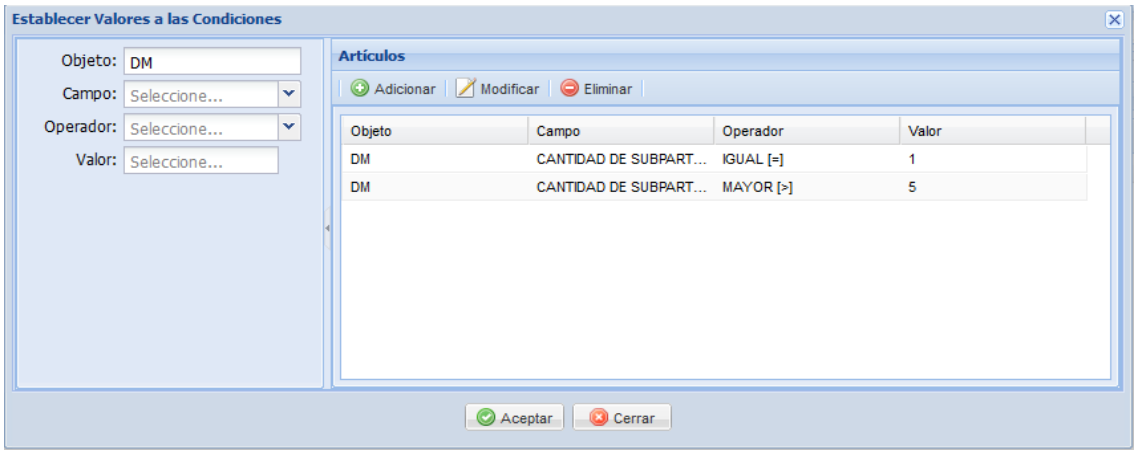


**Figura # 2.2.5 Interfaz establecer valores a las condiciones**

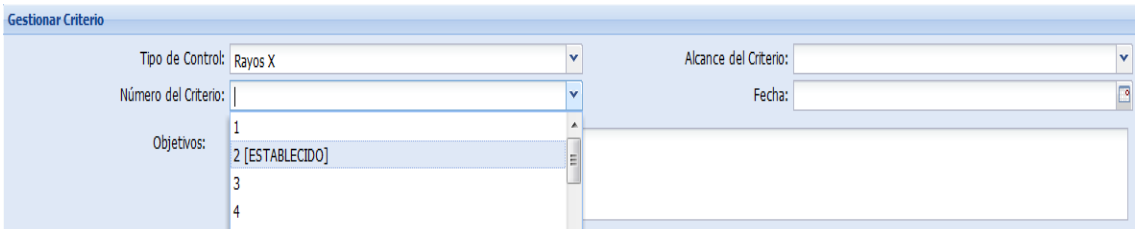


**Figura # 2.2.6 Interfaz de cuando un criterio está establecido**

### *Funcionalidad Realizar Pre-monitoreo*

Una de las funcionalidades de mayor importancia que presenta el módulo de Selectividad es el pre-monitoreo que permite verificar el estado de los criterios actuales sobre los datos antiguos de forma que el usuario pueda valorar cuán apropiado son los criterios que está proponiendo. Se especifica el tipo de control determinado, un rango de fechas entre el cual estarán los objetos que se van a controlar y un conjunto de condiciones (opcional) para filtrar la búsqueda (ver Figura 2.2.7).

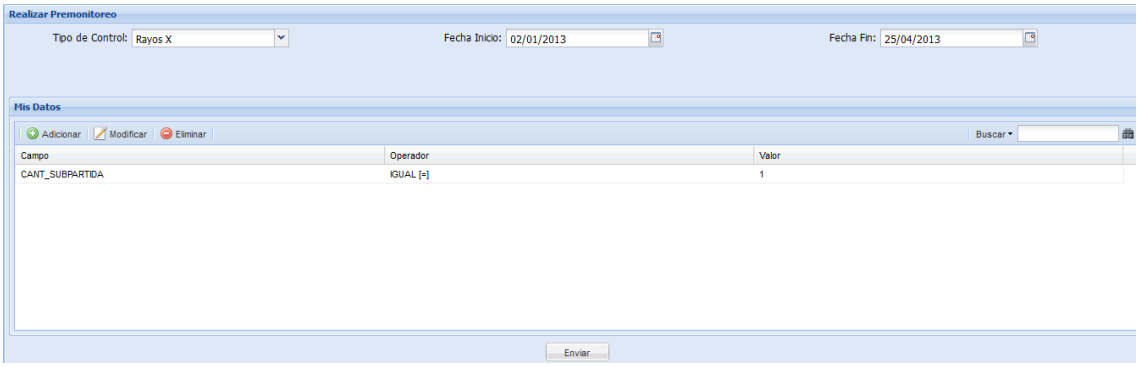


**Figura # 2.2.7 Interfaz realizar premonitoreo**

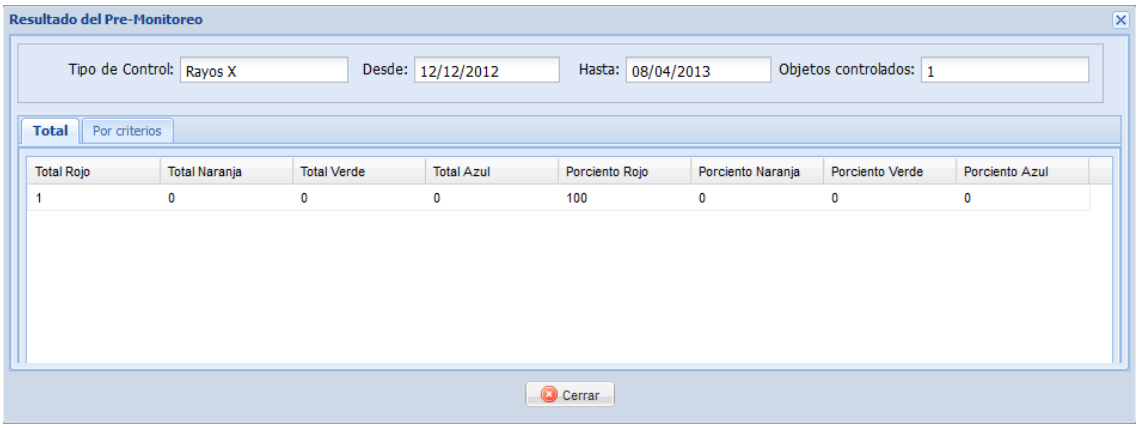


**Figura # 2.2.8 Interfaz resultado del pre-monitoreo por total**

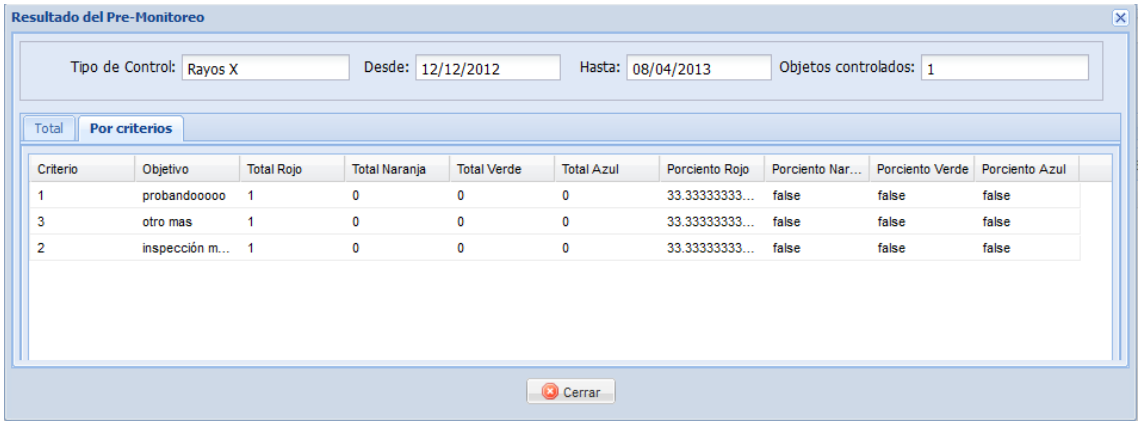


**Figura # 2.2.9 Interfaz resultados del pre-monitoreo por criterio**

Luego de ser enviados estos datos aparecerá una pantalla con los resultados del reporte. Esta funcionalidad mostrará los canales otorgados y el porciento correspondiente, ya sea total o por criterio (ver Figura 2.2.8 y 2.2.9), así como la cantidad de objetos controlados; guardando al final los resultados obtenidos. Para esto, dicha funcionalidad utiliza un algoritmo para otorgar canales, llamado selector, que se describirá a continuación.

### *Algoritmo Utilizado para Otorgar Canales*

Otorgar canal es en gran medida una de las estrategias con las que cuenta GINA en el módulo de Selectividad, pues depende de muchos valores expuestos por los analistas. El mismo fue actualizado por un grupo conformado por analistas de la aduana, especialistas informáticos y un matemático. El flujo comienza con la llegada de un objeto de control para ser inspeccionado, el lugar donde será analizado se llama tipo de control. El algoritmo distribuye entonces de forma equitativa un canal por cada criterio de cada tipo de control siguiendo el orden de prioridad, ROJO, NARANJA, VERDE y AZUL. Para esto el mismo se basa en una desigualdad algebraica de forma iterativa para cada criterio y plantea lo siguiente:

$$(TC) < Total(CT) * \%$$

Donde TC es la cantidad de criterios que cumplen con un determinado canal, Total (CT) es un incremento del número por cada objeto que cumple con el criterio y % es el porciento de asignación al mismo. El número de canales que puede otorgar un criterio puede variar dependiendo del área sobre la que este actúe, por ejemplo, un criterio de control radiológico y de pesaje, solo otorga dos canales Verde o Rojo, es decir se controlan o no. Esto no sucede de la misma manera con otras áreas como en las Declaraciones de Mercancías que se les pueden otorgar los 4 canales. Las probabilidades de distintos criterios están determinadas por una combinatoria inmensa dada por la cantidad de elementos tenidos en cuenta en la conformación de cada criterio y el conjunto de sus posibles valores. De forma que es muy difícil que se determine a simple vista por qué un canal ha sido determinado para cierto objeto. De esta manera se logra que un criterio que se defina con 100% de canal rojo, arroje con idéntico resultado a todos los objetos que coincidan con él, así como si se define con 27%, entonces se logrará que 27 de cada 100 objetos de control estén determinados por este resultado.

Es importante señalar que puede suceder que ninguno de los objetos de control que se especifican tengan asociados un canal de control a partir de los criterios en cada uno de los momentos de controles. Teniendo en cuenta esto se creó la afectación por azar la cual siempre es importante en estos casos. De

forma que el usuario siempre puede definir si desea o no realizar el control a partir de azares para los casos en que los criterios no arrojen ningún resultado, permitiendo que el objeto tratado no se deje de controlar.

## *2.3 Integración con el GINA*

La arquitectura del software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán los principios que orientan su diseño y evolución (23).

Su importancia radica en que constituye la columna vertebral para construir un sistema de software. Es en gran medida responsable de permitir o no ciertos atributos de calidad del sistema, como la confiabilidad y el rendimiento, etc. Se puede comprender como un modelo abstracto reutilizable que puede transferirse de un sistema a otro o un sistema adecuarse a una arquitectura ya definida. También representa un medio de comunicación y discusión entre participantes del proyecto (23).

La arquitectura que se presenta para el GINA está definida sobre el framework Symfony del lenguaje de programación PHP en su versión 1.2.8. Su utilización proporciona ventajas significativas para los desarrolladores de software. El marco de trabajo mencionado es capaz de fusionar buenas prácticas de trabajo por sí mismo, de forma que los desarrolladores no tengan que preocuparse por implementar varios de los patrones de diseño y arquitectónicos más utilizados, ya que el mismo framework los utiliza.

Symfony proporciona una detallada estructura y organización en el módulo que se desarrolla (ver Figura 2.3.1). Además de almacenar los archivos del proyecto en una estructura estandarizada de tipo árbol.

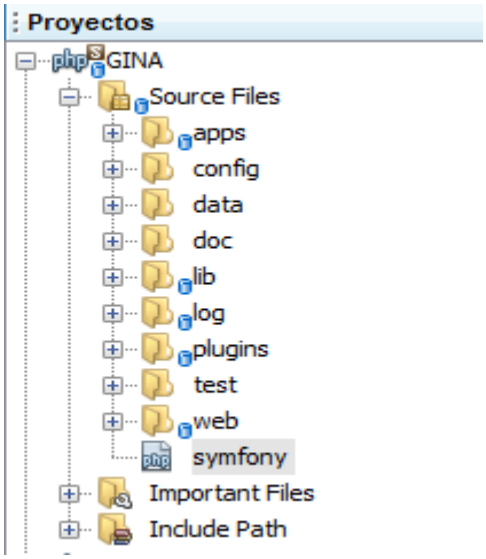


**Figura # 2.3.1 Estructura y organización de los directorios de Symfony**

**apps:** contiene un directorio por cada aplicación del proyecto (selectividad).

**cache:** aquí se almacena la versión cacheada de la configuración. El mecanismo de cache utiliza los archivos de este directorio para acelerar la respuesta a las peticiones web.

**config:** guarda los archivos de configuración general del proyecto.

**data:** contiene todos los archivos relacionados con los datos, como por ejemplo el esquema de la base de datos, el archivo que contiene las instrucciones SQL para crear las tablas.

**lib:** almacena las clases y librerías externas, normalmente código común a todas las aplicaciones del proyecto. El subdirectorio model/ guarda el modelo de objetos del proyecto.

**log:** contiene los archivos log generados por Symfony, también los generados por el servidor web, bases de datos y otros componentes del proyecto. Symfony crea un archivo log por cada aplicación del proyecto.

**pluggins:** guarda los pluggins instalados en la aplicación, pueden ser utilizados para la integración con otros subsistemas.

**test:** guarda las pruebas unitarias y funcionales escritas en PHP, y compatibles con el framework de prueba de Symfony.

**web:** esta es la raíz del servidor web. Los únicos archivos accesibles desde Internet son los que se encuentran en este directorio.

El módulo Selectividad consta de cuatro directorios (ver Figura 2.3.2), estos son: *config, lib, modules y templates.* En el interior del directorio ‘‘*config*’’ se almacena la mayor parte de la configuración de la aplicación; dentro de *“lib”* se encuentran las clases y librerías utilizadas exclusivamente por la aplicación; en *“modules”* se almacenan los módulos que definen las características de la aplicación y *“templates”* contiene las plantillas globales de la aplicación, es decir, las que utilizan todos los módulos. Actualmente se encuentra el fichero *indexSuccess.php* el cual representa la plantilla, que aunque es una página en blanco es importante definirla para que funcione correctamente la aplicación.

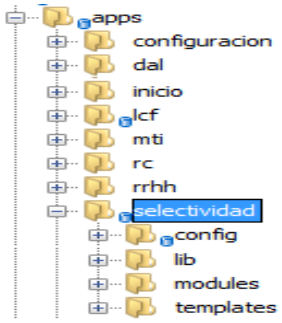


**Figura # 2.3.2 Estructura y organización de Selectividad sobre Symfony**

De forma general el módulo de Selectividad se encuentra ubicado estructuralmente dentro de GINA y este a su vez sobre el framework Symfony. El módulo desarrollado cuenta entonces con un servicio interno llamado *determinarCanal* que puede ser consumido por otros subsistemas del GINA a petición de los mismos. Dicho servicio se encuentra implementado en un fichero llamado *selector.php* (ver Figura 2.3.3) y cuenta con los siguientes parámetros: el tipo de control, el o los objetos de control que se analizarán, la Aduana de la que se trata y el subsistema que solicita el servicio. Dado estos parámetros *determinarCanal* permitirá entonces encontrar el canal de control a otorgar a los objetos recibidos y de esta manera el subsistema solicitante sabrá el proceder ante cada objeto de control analizado (ver Figura 2.3.4).

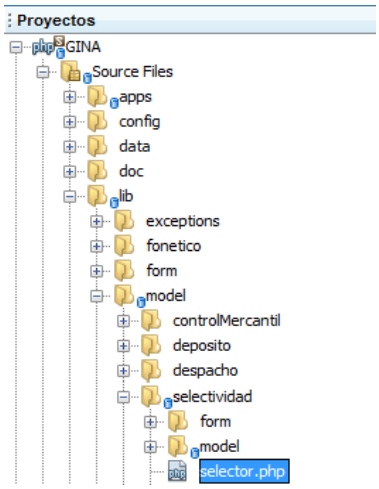


**Figura # 2.3.3 Ubicación del selector en Symfony**

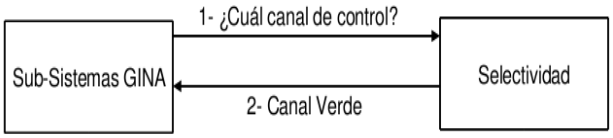


**Figura # 2.3.4 Utilización de Selectividad por otros subsistemas**

Cada uno de los subsistemas que requieren los servicios de Selectividad conoce qué flujo de trabajo es el que se activa para cada posible respuesta. De forma que encaminan sus procesos condicionados por el resultado que arroje el módulo en cada uno de los casos. En la siguiente imagen se muestra como un subsistema del GINA, en este caso Despacho Aduanero, hace una petición al selector y este le devuelve un canal para el cual el despacho debe seguir uno de sus flujos según lo establezcan sus resoluciones y procesos.

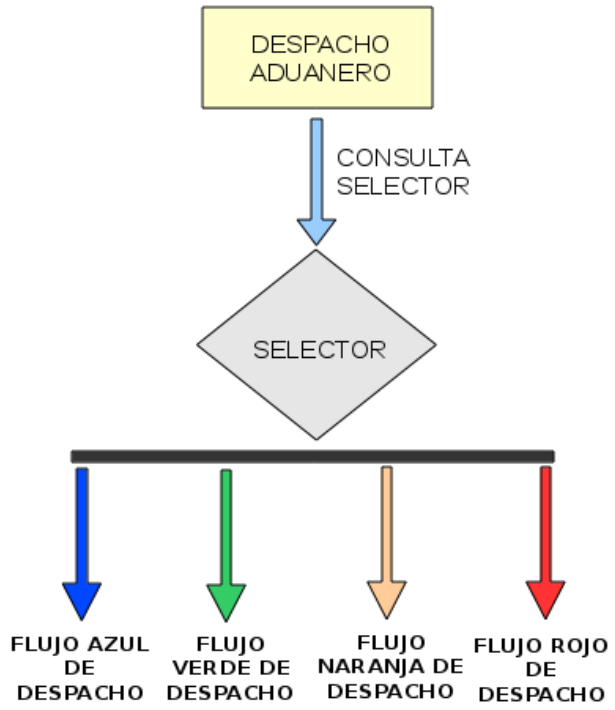


**Figura # 2.2.11 Petición al selector por Despacho Aduanero**

### *Patrón Modelo Vista Controlador en Symfony*

Symfony utiliza como patrón arquitectónico el MVC (Modelo Vista Controlador) separando su contenido en tres capas fundamentales: El modelo: representa la información con que trabaja la aplicación, la lógica de negocio (abstracción a la base de datos, acceso a los datos). La vista: se encarga de mostrar al usuario la información (vista, plantilla, layout), normalmente como lenguaje HTML. El controlador: procesa las interacciones del usuario y lleva a cabo los cambios en el modelo o en la vista (controlador frontal, acción) (24).

Con la estructura arquitectónica presentada, se consigue un mantenimiento más sencillo de las aplicaciones. Un ejemplo sería: si una misma aplicación debe ejecutarse tanto en un navegador estándar como un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original.

Symfony toma lo mejor de la arquitectura MVC y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo. En primer lugar, el controlador frontal y el layout son comunes para todas las

acciones de la aplicación. Se pueden tener varios controladores y varios layouts, pero solamente es obligatorio tener uno de cada uno. El controlador frontal es un componente que sólo tiene código relativo al MVC, por lo que no es necesario crear uno, ya que Symfony lo genera de forma automática (24).

Las clases de la capa del modelo también se generan automáticamente, en función del modelo de datos de la aplicación. La librería Propel se encarga de esta generación automática, ya que crea el esqueleto o estructura básica de las clases y genera automáticamente el código necesario. La abstracción de la base de datos es completamente transparente para el programador, ya que se realiza de forma nativa mediante PDO (PHP Data Objects). Así, si se cambia el sistema gestor de bases de datos en cualquier momento, no se debe reescribir ni una línea de código, ya que tan sólo es necesario modificar la cadena de conexión en el archivo de configuración *database.yml*.

La siguiente imagen ilustra el funcionamiento del patrón MVC según Symfony.

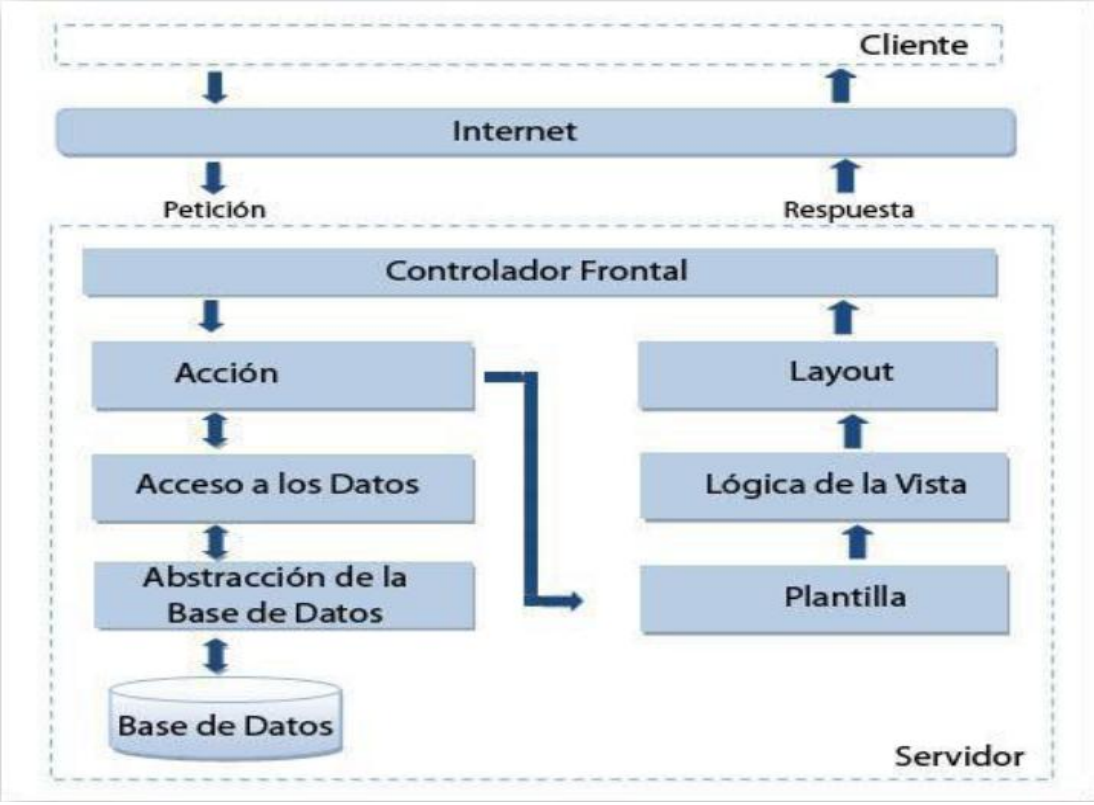


**Figura # 2.3.4 Implementación del patrón MVC según el framework Symfony (24)**

# CAPÍTULO 2

A continuación se presentan varios de los patrones de diseño utilizados durante la implementación de la solución y definidos en el diseño de la misma.

## *Patrones GRASP Implementados*

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 (25). A continuación se evidencian varios de estos patrones implementados por Symfony durante el desarrollo del módulo.

### *Creador*

La clase selectividadActions contiene las acciones definidas para el módulo Selectividad y es en ella misma donde se ejecutan las funciones que dan cumplimiento a los requerimientos del módulo. En esta clase las acciones se encargan de crear los objetos de las clases que representan las entidades (*SelCriterio*, *SelCanal*, *SelCondicion*, etc), evidenciando de este modo que la clase selectividadActions es el “creador” de las mismas.

### *Experto*

Se evidencia este patrón puesto que Propel es la librería externa que utiliza Symfony para realizar su capa de abstracción al modelo de datos, encapsulando toda la lógica de los datos y generando las clases con todas las funcionalidades comunes de las entidades (*BaseSelCriterio*, *BaseSelCanal*, *BaseSelCondicion*). Por tanto cada clase creada por Propel a partir de una entidad es experta en manejar su información.

### *Controlador*

Todas las peticiones web son manejadas por un solo controlador frontal (*selectividad_dev.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 y el nombre de un módulo con la URL entrada por el usuario.

## *Patrones GOF Utilizados*

Los patrones GOF acrónimo de Gang of Four (banda de los cuatro), se dividen en tres grupos fundamentales, de creación, estructura y comportamiento. Los primeros se encargan de mostrar una guía

de cómo crear objetos cuando sus creaciones requieren tomar decisiones. Estas decisiones serán normalmente resueltas dinámicamente, decidiendo qué clases instanciar o sobre qué objetos se delegarán responsabilidades. Los de estructura se encargan de describir la forma en que diferentes tipos de objetos pueden ser organizados para trabajar unos con otros. Y los de comportamiento se utilizan para manejar, organizar y combinar comportamientos. Seguidamente se describen los utilizados por Symfony durante la implementación.

***Singleton (Instancia única)***

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

***Decorator (Decorador)***

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 otro punto de vista, el layout decora la plantilla.

## *2.4 Estándar de Codificación*

La adopción de un estándar de codificación solo es viable si se sigue desde el principio hasta el final del proyecto de software, no es práctico adoptar un estándar de codificación una vez iniciado el trabajo. En el caso del departamento de Soluciones para la Aduana, el estándar de codificación fue definido por la dirección del proyecto en sus inicios y se encuentra registrado en el documento “Propuesta de un estándar de codificación” (7). Fue creado con el objetivo de mejorar el trabajo de los desarrolladores del departamento. También debe permitir identificar de forma sencilla cuál es el objetivo y las funcionalidades que brinda cada una de las clases, funciones y demás componentes de software dado su nomenclatura, es necesario que esto se pueda identificar a simple vista. Además debe servir de guía para los implementadores que tienen que continuar el desarrollo de las aplicaciones (7).

Entre las reglas que deben cumplirse para la implementación de cualquier solución de este departamento se especificó, que todas las funciones definidas por los desarrolladores deben seguir la nomenclatura UpperCamelCase, a no ser que para cierto ámbito se especifiquen características específicas. Los nombres de las funciones deben dejar reflejado claramente cuál es la acción que realiza. Los nombres de las clases deben estar expresados en notación UpperCamelCase, no deben contener guiones bajos en su nombre "_".

Symfony trae su propia nomenclatura para las clases de las acciones y sus funciones, pero no especifica claramente cuál es el nombre que se le debe poner a cada una de las funcionalidades del modelo que serán accedidas por el usuario (acciones). Dentro de las especificaciones del framework se encuentra que cada una de estas acciones debe comenzar con la palabra *execute* y estar en la nomenclatura "CamelCase". Los nombres de las acciones deben especificar con la menor cantidad de palabras cuál es el objetivo de la misma y de ser posible estar en infinitivo.

En cuanto a las pruebas que se le realizan a la solución también se definen reglas. Las pruebas deben tener su propia nomenclatura, tanto para los nombres de los archivos como para los estilos de codificación. Dentro del estándar de codificación se incluyen algunas buenas prácticas que deben seguir los desarrolladores para garantizar la calidad de las aplicaciones realizadas por el proyecto (7).

## *2.5 Aportes de la Solución y Beneficios Esperados*

La solución propuesta proporcionará grandes beneficios a los especialistas del área de Lucha Contra el Fraude ya que contiene las contribuciones que estos requieren para mejorar notablemente su trabajo. Se obtendrá un producto informatizado que será posible aplicarse ante cualquier cambio organizacional. Contará con un sistema de apoyo para alertar posibles riesgos en la entrada y salida del país. Dicha solución gestionará todos los objetos existentes y permitirá el control de otros objetos que sean requeridos por la aduana, facilitando la supervisión de los criterios de control establecidos en los procesos aduaneros. Se reducirán los gastos de la organización producto a la asignación innecesaria de personal para realizar los controles de las mercancías.

Por otra parte los analistas del área de Lucha Contra el Fraude podrán retroalimentarse de los resultados aplicados en las áreas correspondientes, ya que la solución desarrollada almacenará dichos resultados

facilitando una mejor gestión de los objetos de control. Permitirá realizar pre-monitoreo sobre objetos proporcionando de esta manera un control sobre los criterios establecidos y permitiendo comprobar la fiabilidad de los mismos. Esto, junto al aumento de las funcionalidades del sistema, contribuye a la existencia de una mayor certeza en la toma de decisiones.

### *Conclusiones Parciales*

Siguiendo el modelo de desarrollo de software definido por el centro se realizó la implementación del módulo Selectividad. Para lograr el completo desarrollo de la solución se siguieron buenas prácticas de programación web definidas en el capítulo anterior. La aplicación de las mismas y el cumplimiento de las reglas planteadas en el estándar de codificación del departamento permitieron generar el código fuente de la solución de manera uniforme, facilitando la comprensión de este por los desarrolladores en el futuro.

Fueron implementados 20 requisitos funcionales y junto a estos, importantes funcionalidades que permiten la integración de Selectividad con los demás módulos pertenecientes al GINA. Esto facilita el flujo de la información entre cada uno de ellos, permitiendo que se retroalimente el módulo desarrollado. El sistema facilitará el trabajo de los funcionales de la Aduana General de la República de Cuba y a su vez permitirá el ahorro de tiempo a la hora de realizar los controles pertinentes en cada uno de los procesos aduaneros. De forma general la solución propuesta y desarrollada facilitará la toma de decisiones más seguras y efectivas.

# CAPÍTULO 3: VALIDACIÓN Y PRUEBA

Una vez culminada la implementación del módulo es de vital importancia comprobar el estado funcional del mismo antes que se ponga a disposición del cliente. En el presente capítulo se exponen las principales pruebas realizadas sobre el módulo de Selectividad y se lleva a cabo el análisis de los resultados obtenidos en las mismas. Se diseñan los casos de prueba para comprobar la solución desarrollada, con vista a mitigar los errores para así obtener un módulo completamente funcional acorde a las necesidades del cliente.

## *3.1 Tipos de Prueba*

El proceso de prueba se centra en los procesos lógicos internos del software y en los externos funcionales, asegurando que todas las sentencias sean comprobadas. De esta manera permite realizar las pruebas para la detección de errores y asegurar que la entrada definida produce resultados reales, de acuerdo con los resultados requeridos (19). Ambos enfoques serán aplicados sobre el módulo desarrollado, a continuación se ejemplifican los tipos de prueba aplicados a la solución desarrollada.

## *3.2 Pruebas de Caja Blanca*

Se procede a la comprobación de los procesos lógicos internos del software, para esto se utilizó las pruebas de caja blanca, tipo de prueba definido en el Capítulo 1. Dentro de este se aplicó la técnica del camino básico, también definida en el Capítulo 1. Se tomó como ejemplo la funcionalidad **buscar propuestas de tipo de control** (ver Figura 3.2.1). Mediante esta funcionalidad, dado un rango de fechas determinado y el estado seleccionado, se obtienen las propuestas de los tipos de controles creados; permitiendo aceptar, rechazar o dejar pendiente las propuestas obtenidas. A continuación se muestra el código fuente perteneciente a dicha funcionalidad. El mismo se encuentra enumerado, permitiendo de este modo una mejor aplicación de la técnica definida.

```
public function executeBuscarPropuestaDeTipoControl($request) {
    $estado = $request->getParameter('estado'); // (1)
    $fechaInicio = UtilFecha::fechaAConsultar($request->getParameter('fechaInicio')); // (1)
    $fechaFin = UtilFecha::fechaAConsultar($request->getParameter('fechaFin')); // (1)
    $gruposPropuestas = SelGrupoPeer::darGruposPropuesta($fechaInicio, $fechaFin, $estado); // (1)
    $resp = new stdClass(); // (1)
    $resp->articulos = array(); // (1)
    if (empty($estado)) // (2)
        $estado = 'P'; // (3)
    else
        $esatdo = substr($estado, 0, 1); // (4)
    $gruposPropuestas = SelGrupoPeer::darGruposPropuesta($fechaInicio, $fechaFin, $estado); // (5)
    foreach ($gruposPropuestas as $grupo) { // (6)
        $dato = new stdClass(); // (7)
        $dato->id = $grupo->getIdGrupo(); // (7)
        $dato->propuestaPor = $this->getNombrePersonaPorID($grupo->getIdUsuario()); // (7)
        $dato->fecha = $grupo->getFecha('d/m/Y'); // (7)
        $dato->nombre = $grupo->getNombreGrupo(); // (7)
        $dato->estado = $grupo->getEstado(); // (7)
        $resp->articulos[] = $dato; // (7)
    } // (8)
    return $this->renderText(json_encode($resp)); // (9)
}
```

**Figura # 3.2.1 Algoritmo de la funcionalidad buscar propuestas de tipo de control**

### *Técnica del Camino Básico*

Para aplicar esta técnica, es necesario dibujar el grafo de flujo asociado al código fuente de la funcionalidad tomada como ejemplo (ver Figura 3.2.2). Luego se aplica la métrica del cálculo de la complejidad ciclomática del grafo, determinándose un conjunto básico de caminos independientes. Y finalmente se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico. En la siguiente figura se muestra el grafo de flujo asociado al código anterior.

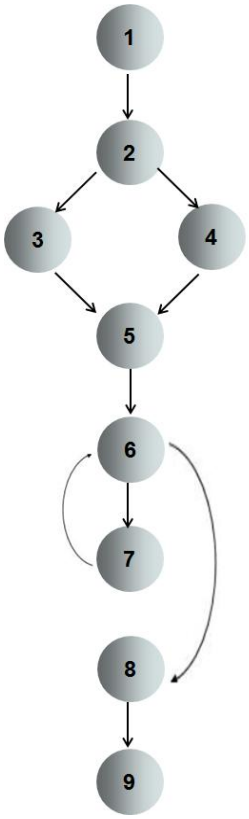


**Figura # 3.2.2 Grafo de flujo asociado a la funcionalidad buscar propuestas de tipo de control**

### *Cálculo de la complejidad ciclomática a partir de un segmento de código*

La complejidad ciclomática es una métrica de software extremadamente útil pues proporciona una medición cuantitativa de la complejidad lógica de un programa. El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y brinda un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez. Un camino independiente es cualquier camino del programa que introduce por lo menos un nuevo conjunto de sentencias de procesamiento o una nueva condición. El camino independiente se debe mover al menos por una arista que no haya sido recorrida anteriormente (20). A continuación se muestran diferentes maneras de obtener dicha complejidad:

- ✓ La complejidad ciclomática coincide con el número de regiones del grafo de flujo.

  CC = 3

- ✓ La complejidad ciclomática, de un grafo de flujo, se define como CC = Aristas - Nodos + 2

  CC = (A – N) + 2

  CC = (9 – 8) + 2

  CC = 3

- ✓ La complejidad ciclomática de un grafo de flujo también se define como CC = Nodos de predicado + 1

  CC = P + 1

  CC = 2 + 1

  CC = 3

El cálculo efectuado mediante las tres maneras presentadas arroja el mismo valor, por lo que se puede plantear que la complejidad ciclomática del código es 3, lo que significa que existen 3 posibles caminos por donde el flujo puede circular. Este valor representa el límite mínimo del número total de escenarios de prueba para el procedimiento tratado.

Seguidamente es necesario representar los caminos básicos por los que puede recorrer el flujo. En la siguiente tabla se muestran dichos caminos.

| Número | Camino |
|---|---|
| **1** | **1-2-3-5-6-8-9** |
| **2** | **1-2-4-5-6-7-6-8-9** |
| **3** | **1-2-3-5-6-7-6-8-9** |

**Tabla # 3.2.1 Caminos básicos del flujo**

Después de haber elaborado el grafo de flujo y extraído los caminos básicos, se preparan los escenarios de prueba que forzarán la ejecución de cada uno de esos caminos. Se escogen los datos de forma que las condiciones, de los nodos predicados o de condición, estén adecuadamente establecidas con el fin de comprobar cada camino. Los escenarios de prueba deben cumplir con las siguientes exigencias:

- ✓ Descripción: se hace la entrada de datos necesarios, validando que ningún parámetro obligatorio pase nulo al procedimiento o no se entre algún dato erróneo.

✓ Condición de ejecución: se especifica cada parámetro para que cumpla una condición deseada para ver el funcionamiento del procedimiento.

✓ Entrada: se muestran los parámetros que entran al procedimiento.

✓ Resultados esperados: se expone el resultado que se espera que devuelva el procedimiento.

**Escenario para el camino básico 1**

| **Descripción** | **Condición de ejecución** | **Entrada** | **Resultados Esperados** |
|---|---|---|---|
| Los datos de entrada cumplirán con lo siguiente:<br><br>El parámetro estado será vacío.<br>La fecha de inicio será 01/02/2013.<br>La fecha fin será 30/06/2013. | El **estado** será vacío.<br>La **fechaInicio** será 01/02/2013.<br>La **fechaFin** será 30/06/2013. | $estado= null<br>$fechaInicio=01/02/2013<br>$fechaFin=30/06/2013 | Se espera que se devuelva los tipos de control en estado Pendiente. |

**Escenario para el camino básico 2**

| **Descripción** | **Condición de ejecución** | **Entrada** | **Resultados Esperados** |
|---|---|---|---|
| Los datos de entrada cumplirán con lo siguiente:<br><br>El parámetro estado será Aceptado.<br>La fecha de inicio será 20/03/2013.<br>La fecha fin será 30/06/2013. | El **estado** será Aceptado.<br>La **fechaInicio** será 20/03/2013.<br>La **fechaFin** será 30/06/2013. | $estado= Aceptado<br>$fechaInicio=20/03/2013<br>$fechaFin=30/06/2013 | Se espera que se devuelva los tipos de control en estado Aceptado. |

**Escenario para el camino básico 3**

| Descripción | Condición de ejecución | Entrada | Resultados Esperados |
|---|---|---|---|
| Los datos de entrada cumplirán con lo siguiente:<br><br>El parámetro estado será Rechazado.<br>La fecha de inicio será 20/03/2013.<br>La fecha fin será 15/05/2013. | El **estado** será Rechazado.<br>La **fechaInicio** será 20/03/2013.<br>La **fechaFin** será 15/05/2013. | $estado= Rechazado<br>$fechaInicio=20/03/2013<br>$fechaFin=15/05/2013 | Se espera que se devuelva los tipos de control en estado Rechazado. |

### *Resultados*

Luego de confeccionar y ejecutar los diferentes escenarios por los que puede transitar la funcionalidad, se estará seguro de que todas las sentencias del programa se han ejecutado por lo menos una vez. Comprobándose que el flujo de trabajo de la funcionalidad es correcto ya que cumple con las condiciones de ejecución y resultados esperados. Siguiendo el procedimiento descrito, se aplicó este tipo de prueba a las demás funcionalidades que agrupaban los 14 requisitos de complejidad alta, estas fueron: **gestionar criterios**, **realizar pre-monitoreo** y **gestionar objeto de control**. De esta manera quedaron comprobadas las funcionalidades más críticas de la implementación.

## *3.3 Pruebas de Caja Negra*

Luego de concluido el desarrollo de la solución se realizan las pruebas de caja negra. Su objetivo es validar si el comportamiento observado del software cumple o no con sus especificaciones. Las mismas fueron ejecutadas por analistas del departamento de Soluciones para la Aduana aplicando la técnica de partición equivalente definida en el Capítulo 1. Para ello se realizaron los diseños de casos de prueba correspondientes a las funcionalidades requeridas.

A continuación se muestran dos de los diseños realizados para las funcionalidades **proponer tipo de control** y **buscar propuestas de tipo de control**. Ambos diseños se encuentran distribuidos en 2

escenarios de pruebas, de forma que las funcionalidades sean probadas con un mayor número de juegos de datos.

### *Diseño de Caso de Prueba: Proponer Tipo de Control*

**Descripción de las variables**

| No | Nombre campo | Clasificación | Descripción |
|---|---|---|---|
| 1 | Tipo Control | Campo texto. | Nuevo tipo de control a proponer. |
| 2 | Canales | Lista desplegable. | Cantidad de canales del tipo de control a proponer. |
| 3 | Objeto Control | Lista desplegable. | Objeto de control a controlar. |
| 4 | Nuevo Objeto | Campo texto. | Nuevo objeto de control propuesto. |
| 5 | Fundamentación | Campo texto. | Fundamentación de la proposición del tipo de control. |

**Condiciones de ejecución:** Para la ejecución de este requisito el usuario debe estar autenticado en el sistema.

| Escenario | Descripción | Flujo central |
|---|---|---|
| EC1.1 Proponer Tipo de Control. | El analista de la AGR después de analizar los posibles riesgos propone al administrador del CADI la creación de un nuevo tipo de control. | 1. Clic en la opción del menú principal en **Gestionar tipos de control** en el submenú **Proponer Tipo de Control.**<br>2. El sistema muestra la ventana correspondiente con los campos siguientes: Tipo de Control, Canales, Objeto Control, Nuevo Objeto Control y Fundamentación.<br>3. El usuario introduce los valores en los campos.<br>4. Clic en el botón Aceptar.<br>5. Termina el escenario. |

| | | |
|---|---|---|
| EC1.2 Datos incorrectos. | Se realiza este escenario en el caso que se inserte algún dato incorrecto. | 1. Clic en la opción del menú Proponer Tipo de Control.<br>2. El sistema muestra la ventana correspondiente con los campos siguientes: Tipo de Control, Canales, Objeto Control, Nuevo Objeto Control y Fundamentación.<br>3. El usuario introduce los valores en los campos.<br>4. Clic en el botón Aceptar.<br>5. El sistema muestra un mensaje de error: "Debe corregir los campos inválidos".<br>6. Termina el escenario. |

**Datos de prueba Escenario 1.1 "Proponer Tipo Control"**

| Tipo de Control | Canales | Objeto Control | Nuevo Objeto | Fundamentación | Respuesta del sistema |
|---|---|---|---|---|---|
| V | V | N/A | V | V | Se inserta el tipo de control correctamente. Se muestra el siguiente mensaje Se ha registrado correctamente la petición del tipo de control de selectividad. |
| Rayos X | 2 canales | vacío | DM | mejorar el control | |
| V | V | V | N/A | V | |
| Rayos X | 4 canales | DM | vacío | mejorar el control | |

**Figura # 3.3.1 Escenario 1.1 "Proponer Tipo Control"**

**Datos de prueba Escenario 1.2 "Datos Incorrectos"**

| Tipo de Control | Canales | Objeto Control | Nuevo Objeto | Fundamentación | Respuesta del sistema |
|---|---|---|---|---|---|
| V | I | V | N/A | I | Muestra un mensaje de error: Debe corregir los campos inválidos. |
| Rayos X | vacío | DM | N/A | vacío | |

**Figura # 3.3.2 Escenario 1.2 "Proponer Tipo Control"**

### *Diseño de Caso de Prueba: Buscar Propuestas de Tipo de Control*

**Descripción de las variables**

| No | Nombre campo | Clasificación | Descripción |
|---|---|---|---|
| 1 | Fecha Inicio | Campo fecha. | Fecha a partir de la cual se desea obtener las propuestas. |
| 2 | Fecha Fin | Campo fecha. | Fecha hasta la cual se desea obtener las propuestas. |
| 3 | Estado | Lista desplegable. | Estado de las propuestas. |

**Condiciones de ejecución:** Para la ejecución de este requisito el usuario debe estar autenticado en el sistema.

| Escenario | Descripción | Flujo central |
|---|---|---|
| EC 1.1 Buscar Propuestas. | Busca y muestra las propuestas de creación de un tipo de control almacenadas en el sistema. | 1. Clic en la opción del menú principal **Gestionar Tipos de Control** en el submenú **Definir Tipo de Control**.<br>2-El sistema muestra la ventana correspondiente con los campos siguientes:Fecha Inicio, Fecha Fin y Estado.<br>3.El usuario introduce los valores en los campos.<br>4-Clic en el botón Aceptar.<br>5-Termina el escenario. |
| EC 1.2 Analizar Propuestas. | El usuario analiza la propuesta donde puede Aceptar, Rechazar o Cerrar la misma. | 1. Clic en la opción del menú principal **Gestionar Tipos de Control** en el submenú **Definir Tipo de Control**.<br>2. El sistema muestra la ventana correspondiente con los campos siguientes: Fecha Inicio, Fecha Fin y Estado.<br>3. El usuario introduce los valores en los campos.<br>4. El sistema muestra las propuestas.<br>5. Clic encima de la propuesta que desea analizar.<br>6. El sistema muestra la propuesta realizada anteriormente.<br>7. El usuario analiza e inserta en el campo Correo Electrónico del usuario preciso y presiona el botón Aceptar.<br>8-Termina el escenario. |

**Figura # 3.3.3 Descripción y flujo central de la funcionalidad "Buscar Propuestas de Tipo de Control"**

**Datos de prueba Escenario 1.1 "Buscar Propuestas"**

| Fecha Inicio | Fecha Fin | Estado | Respuesta del sistema |
|---|---|---|---|
| V | V | V | Se realiza la búsqueda correctamente. Se muestra en pantalla el resultado correspondiente. |
| 10/05/2012 | 13/04/2013 | Aceptada | |
| NA | NA | NA | Se realiza la búsqueda correctamente. Se muestra en pantalla las propuestas con el estado de Pendientes. |
| vacío | vacío | vacío | |

**Figura # 3.3.4 Escenario 1.1 "Buscar Propuestas de Tipo de Control"**

**Datos de prueba Escenario 1.2 "Analizar Propuestas"**

| Correo electrónico | Respuesta del sistema |
|---|---|
| NA | Muestra el siguiente mensaje El correo electrónico especificado no es válido |
| vacío | |
| NA | Envía al correo electrónico la propuesta especificada con anterioridad. |
| laperez@estudiantes.uci.cu | |

**Figura # 3.3.5 Escenario 1.2 "Buscar Propuestas de Tipo de Control"**

### *Resultados*

Las pruebas funcionales fueron aplicadas al resto de los requisitos de la misma forma que se describió en el epígrafe anterior. Las no conformidades detectadas durante la primera iteración fueron corregidas completamente, lo que posibilitó que en la segunda iteración los escenarios arrojaran resultados satisfactorios en todos los casos. Los resultados por cada una de las funcionalidades se describen a continuación:

| Funcionalidad | Cantidad No conformidad | Tipo de error |
|---|---|---|
| Proponer Tipo de Control | 2 | Validación |
| Buscar Propuestas de Tipo de Control | 4 | Interfaz, Funcionalidad, Ortografía |
| Definir Objeto de Control Nivel 1 | 4 | Validación, Interfaz, Ortografía, Otros |
| Definir Objeto de Control Nivel 2 | 4 | Interfaz, Ortografía |
| Gestionar Criterio | 3 | Validación, Funcionalidad |
| Realizar Premonitoreo | 1 | Interfaz |
| Porciento de Afectación | 1 | Validación |

De forma general fueron detectadas 19 no conformidades y 2 recomendaciones, representando esto sobre el total de los requisitos funcionales un 21.9% en la primera iteración. Estos datos quedan representados teniendo en cuenta la clasificación de errores más frecuentes para una aplicación según la plantilla de calidad del centro CEIGE. A continuación se muestra la gráfica que representa los tipos de errores encontrados en la primera iteración de las prueba.


**Figura # 3.3 Resultados de las pruebas por errores**

En la siguiente gráfica queda representado de forma general las no conformidades encontradas durante las pruebas en esta primera iteración con un 78.1% de aceptación del módulo realizado.

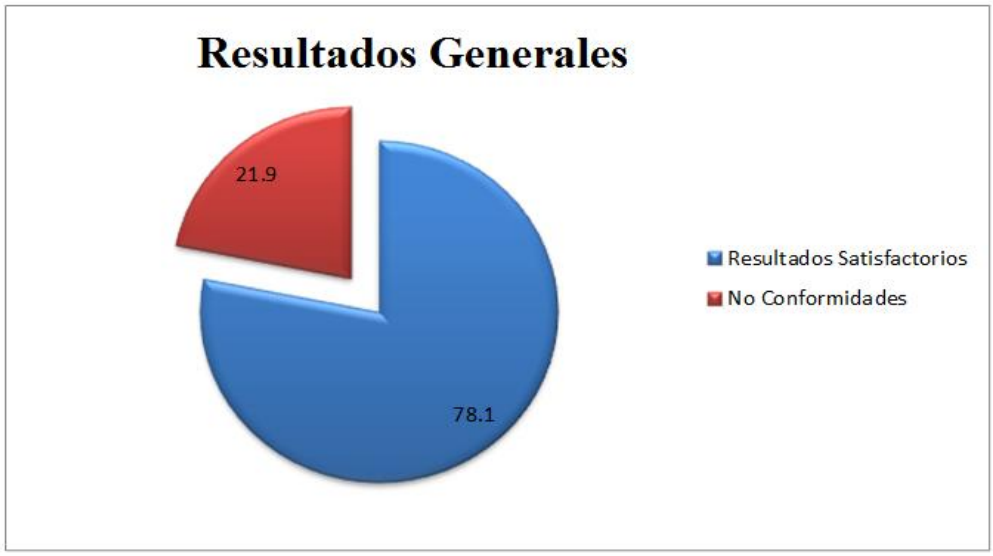


**Figura # 3.4 Resultados de las pruebas**

## *3.4 Conclusiones Parciales*

Durante el presente capítulo se validó el módulo desarrollado. Para lograrlo, se utilizaron diferentes pruebas de calidad de software que permitieron comprobar el cumplimiento de todos los requisitos funcionales establecidos en la fase inicial del proceso de desarrollo del producto.

Dentro de los tipos de prueba aplicados, se encuentra las pruebas de caja blanca, en específico las de camino básico. Para la ejecución del mismo se diseñaron casos de prueba específicos para el algoritmo comprobado, permitiendo así validar el código fuente desarrollado durante la implementación del módulo y facilitando además la corrección de los errores identificados.

Se aplicaron también pruebas de caja negra, aplicando la técnica de partición equivalente, lo que permitió detectar una gran cantidad de no conformidades dando paso a la corrección de las mismas. Las pruebas aplicadas permitieron corroborar el correcto y completo funcionamiento del módulo desarrollado, logrando de esta manera un producto totalmente funcional.

# CONCLUSIONES

Se estudiaron detalladamente los artefactos generados en el análisis y diseño, como las especificaciones y descripciones de los requisitos, el modelo de base de datos, los diagramas de diseño, entre otros. Aspectos que posibilitaron una mayor comprensión de las tareas a desarrollar.

Siguiendo el modelo de desarrollo de software definido por el centro, algunas buenas prácticas de la programación web detalladas en el presente trabajo, así como las herramientas, técnica y tecnologías identificadas, se realizó la implementación de la solución.

Se obtuvo un código fuente de alta calidad y buena legibilidad debido al estricto cumplimiento del estándar de codificación definido. El módulo desarrollado facilitará el trabajo de los funcionales de la Aduana General de la República y a su vez permitirá el ahorro de tiempo a la hora de realizar los controles pertinentes en cada uno de los procesos aduaneros.

A través del pre-monitoreo se podrán supervisar los criterios establecidos, de tal forma que el usuario pueda valorar si los criterios que está proponiendo son apropiados. Por otra parte los analistas del área de Lucha Contra el Fraude podrán retroalimentarse de los resultados obtenidos en las áreas correspondientes facilitando una mejor gestión de los objetos de control.

Se obtuvo un producto que cuenta con un sistema de apoyo para alertar posibles riesgos en la entrada y salida del país. Para comprobar la calidad y el correcto funcionamiento del módulo implementado se le aplicaron pruebas de caja blanca y de caja negra, permitiendo el cumplimiento de todos los requisitos funcionales establecidos en la fase inicial del proceso de desarrollo del producto.

De esta manera la solución propuesta y desarrollada facilitará la toma de decisiones más seguras y efectivas. De forma general y partiendo de los artefactos obtenidos durante el análisis y siguiendo las pautas de arquitectura establecidas, se logró un sistema informático que permite gestionar la selectividad para la Aduana General de la República de Cuba.

# RECOMENDACIONES

A pesar de haberse cumplido satisfactoriamente los objetivos propuestos y que los resultados obtenidos sean los esperados, a continuación se brindan algunas recomendaciones a tenerse en cuenta:

- ✓ Se tiene valorado para futuras versiones del módulo de Selectividad, que la gestión de criterios se haga de forma automática, de forma que el sistema evalúe y proponga los nuevos criterios a aplicar.
- ✓ Consultar este documento como material de estudio, guía y apoyo para un mejor entendimiento del módulo, así como para garantizar mejoras en futuras versiones del mismo.
- ✓ Continuar con el proceso de prueba, proponiendo el módulo desarrollado para que sea analizado detalladamente por el Centro Nacional de Calidad de Software, con el objetivo de lograr identificar y erradicar los posibles errores que aún puedan existir.

# REFERENCIAS BIBLIOGRÁFICAS

1. **Aduana.** Aduana General de la República. [En línea] http://www.aduana.co.cu.

2. **OMA, Organización Mundial de Aduanas.** 1997. Convenio de Kyoto - Directivas de control aduanero. Kyoto: http://www.wcoomd.org/Kyoto/20Sp/cap6, 1997.

3. **Arce, Jorge I Rodríguez.** La Importancia de las Aduanas en el Comercio Exterior.

4. **Nápoles, Lixanis Matos.** Análisis del módulo Selectividad para el Sistema Gestión Integral de Aduanas. La Habana: s.n., 2011.

5. **Soluciones para la Aduana.** Modelo de desarrollo de sowtware. La Habana: s.n., 2012.

6. **Álvarez, Sara.** Desarrollo Web. [En línea] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.

7. **(CEIGE), Dpto. Aduana.** Propuesta de un estándar de codificación. Ciudad de La Habana: s.n.

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

9. **definición.org. definición.org.** [En línea] [Citado el: 12 de Diciembre de 2009.] http://www.definicion.org/lenguaje-de-programacion.

10. **Gutierrez, Javier J.** ¿Qué es un framawork web?

11. **Ext JS "otro" Framework para JavaScript** [En línea] 2010. [Citado el: 25 de febrero de 2010.] Disponible en: http://www.masadelante.com/faqs/servidor.

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

13. **D, Schwave.** Engineering Web Applications for Reuse. s.l. : IEEE Multimedia. Vol. 8, 1.

14. **Json.org.** Introducción a JSON. [En línea] http://json.org/json-es.html.

15. **C, Leopoldo.** [En línea] 19 de Septiembre de 2008. http://techtastico.com/post/7-sistemas-control-versiones.

16. **Información, Depto. Arquitectura Técnica y Dirección de Sistemas de.** BUENAS PRÁCTICAS DE DISEÑO Y PROGRAMACIÓN DE WEB DYNPRO ABAP. 2009.

17. **1471-2000, IEEE Std.** [En línea] http://standards.ieee.org/findstds/standard/1471-2000.html.

# REFERENCIAS


18. **Anna C. Grimán, María Pérez, Luis E Mendoza y Laboratorio de Investigación de Sistemas de Información.** Estrategia de Pruebas para Software OO que garantiza Requerimientos No Funcionales. Caracas: s.n.

19. **Pressman, Roger S.** Técnicas de prueba. Ingeniería de Software: un enfoque práctico. Nueva York, E.U.A: s.n.

20. **Barrios, J.R.** Grupo ARQUISOFT. [En línea] http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node28.html.

21. **Javier Tuya, Isabel Ramos Román y José Javier Dolado Cosín.** Técnicas Cuantitativas para la Gestión en la Ingeniería del Software. 2007.

22. **Márquez, Dayán Trujillo.** Modelo de Diseño de Selectividad. La Habana: s.n., 2011.

23. **Márquez, Dayán Trujillo.** Diseño del módulo Selectividad para el Sistema de Gestión Integral de Aduanas. La Habana: s.n., 2011.

24. **Zaninotto, Fabien Potencier & François.** Symfony 1.2. La guía definitiva. s.l. : Autoedición, 2008.

25. **LARMAN, C. UML y Patrones Introducción al análisis y diseño orientado a objetos.** Primera edición. México, Prentice Hall, 1999. 536 p.

# GLOSARIO

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

**AJAX:** Asynchronous JavaScript And XML o JavaScript asíncrono y XML, es una técnica de desarrollo web para crear aplicaciones interactivas. Esta se ejecuta en el lado de cliente y mantiene comunicación asincrónica con el servidor en segundo plano.

**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.

**CADI:** Centro de Automatización para la Dirección y la Información.

**CamelCase:** Consiste en escribir palabras compuestas eliminando los espacios intermedios y poniendo en minúscula la primera letra y en mayúscula las demás primeras letras de cada palabra contigua.

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

**DM:** Declaración de Mercancías.

**DHTML:** El HTML Dinámico o DHTML (del inglés Dynamic HTML) designa el conjunto de técnicas que permiten crear sitios web interactivos utilizando una combinación de lenguaje HTML estático, un lenguaje interpretado en el lado del cliente (como JavaScript), el lenguaje de hojas de estilo en cascada (CSS) y la jerarquía de objetos de un Document Object Model (DOM).

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

**Framework:** 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.

**HTML:** HyperText Markup Language. Es el lenguaje de marcado predominante para la elaboración de páginas web.

**IDE:** Entorno de Desarrollo Integrado. Consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI).

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

# GLOSARIO

**JSON:** JavaScript Object Notation es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.

**LCF:** Lucha Contra el Fraude.

**Linux:** GNU/Linux es uno de los términos empleados para referirse a la combinación del núcleo o kernel libre similar a Unix denominado Linux, que es usado con herramientas de sistema GNU.

**MTI:** Medio de Transporte Internacional. Vehículo utilizado para el transporte internacional de mercancías y personas. En Cuba por su situación geográfica se refiera a los medios utilizados en la vía aérea y marítima solamente.

**MVC:** Modelo Vista Controlador, es un patrón de arquitectura de software que recibe el nombre que separa los datos de una aplicación, la interfaz de usuario, y la lógica de negocio.

**Oracle:** es un sistema de gestión de base de datos objeto-relacional desarrollado por Oracle Corporation.

**PDO:** PHP Data Objects es una extensión que provee una capa de abstracción de acceso a datos para PHP 5, con lo cual se consigue hacer uso de las mismas funciones para hacer consultas y obtener datos de distintos manejadores de bases de datos.

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

**PhpDocumentor:** consiste en un estándar formal para comentar código PHP.

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

**SQL:** Lenguaje de acceso a datos, utilizado para acceder a los registros de una base de datos.

**UNIX:** Es un sistema operativo portable, multitarea y multiusuario; desarrollado, en principio, en 1969 por un grupo de empleados de los laboratorios Bell de AT&T.

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

**Windows:** Es el nombre de la familia de sistemas operativos desarrollados por Microsoft desde 1981.

# ANEXOS

**Anexo 1. Minuta de reunión obtenida en el encuentro entre el equipo de desarrollo de la UCI y el personal de la AGR.**

Universidad de las Ciencias Informáticas

## Minuta de reunión

| | | | |
|---|---|---|---|
| **Autor** | Yisel Rodríguez Pérez | **Fecha** | 09/10/2012 |
| **Lugar** | CADI | **Hora Inicio** | 09:20 am |
| **Proyecto/ Grupo** | Enfrentamiento - Selectividad | **Hora Terminación** | 10:40 am |
| **Asunto** | Análisis del módulo de Selectividad | | |
| **Asistentes** | Rafael Céspedes racespedes@uci.cu<br>Yisel Rodríguez Pérez yrperez@uci.cu<br>Luis Alberto Pérez Blanco laperez@estudiantes.uci.cu<br>Mercedes Suarez Invitada del CADI<br>Víctor García Analista de Manifiesto de Carga.<br>Natasha Padilla Analista del puerto.<br>Rafael D. Águila Jefe de selectividad.<br>Nancy Rodríguez Especialista Informática. | | |
| | | | |

**Minuta de reunión**

## Orden del día

.1 Analizar las funcionalidades del módulo Selectividad.
.2 Analizar la integración del módulo de Selectividad con Despacho Comercial.
.3 Aprobar los acuerdos tomados en la reunión.

## Actividades

| No | Descripción |
|---|---|
| 1 | Analizar las funcionalidades del módulo Selectividad guiándose por los prototipos de interfaz de usuario. |
| 2 | Analizar la integración del módulo Selectividad y Despacho Comercial. |
| 3 | Leer los acuerdos tomados en la reunión para esclarecer cualquier duda existente. |

**Minuta de reunión**

## Acuerdos Tomados

| No | Acuerdo | Responsable | Fecha de cumplimiento |
|---|---|---|---|
| 1 | Mandar los reportes que se desean implementar en la versión de selectividad. | Dáguila | No definido |
| 2 | Realizar una nueva funcionalidad que genere automáticamente un nuevo criterio a partir de un resultado (premonitoreo). | Victor / Luis | 10/11/12 |
| 3 | En el algoritmo del selector cuando se hable del azar significa que la DM por ejemplo no cumplió con ningún criterio y sale marcada por Azar. Si se da canal verde cumplió con algún criterio. | Yisel | No aplica |
| 4 | Se concluye que cuando varios criterios coinciden se deben guardar todos y dar un solo resultado. | Ysel | No aplica |
| 5 | Se concluye que el alcance es nacional cuando la aduana del usuario es AGR. | Yisel | No aplica |
| 6 | Verificar si en la Tabla de la Base de datos se tiene un campo Fundamentación si no se debe insertar. | Luis / Cespedes | 11/102012 |
| 7 | Realizar encuentro de Enfrentamiento y Despacho para determinar que hace falta para que los criterios los pueda generar selectividad rápido (analizar modelo de BD de despacho y si fuese necesario modificar este modelo) | Cespedes/Yordanis/Luis | 12/10/12 |
| 8 | Notificar el cumplimiento de los acuerdos tomados en la reunión. | Yisel | 30/10/12 |
| 9 | Cambiar la etiqueta Nombre del grupo por Tipo de control. | Luis | 11/10/12 |

**Minuta de reunión**

**Tareas**

| No | Descripción | Responsable | Fecha de cumplimiento |
|---|---|---|---|
| 1 | Arreglar las deficiencias existentes del sistema selectividad. | Luis /Cespedes | 30/10/2012 |
| 2 | Implementar los Reportes acordados en la reunión. | Luis/ Dáguila | 18/11/12 |
| 3 | Informar y gestionar los puntos descritos en los acuerdos con el objetivo de hacerlos viables. | Yisel /Cespedes | 30/10/12 |

**Minuta de reunión**

**Anexo 2. Descripción de los tipos de canales.**

| **Color** | **Significado** |
|---|---|
| Rojo | Implica la revisión documental y el reconocimiento físico o físico-químico de los objetos de importación o exportación. |
| Naranja | Implica solamente la revisión de los documentos presentados para el despacho de importación o exportación. |
| Verde | Este canal no requiere ningún control sobre los documentos ni los objetos en cuestión aunque los documentos deben ser presentados ante la Aduana antes de la liquidación, quedando a disposición del inspector hacerle inspección o no. |
| Azul | Solamente se asignará este canal por excepción a las entidades seleccionadas por ser extremadamente confiables en su actuar ante la Aduana y no requiere de presentación de la Declaración de Mercancías ni sus documentos complementarios para su liquidación. |

**Tipos de canales**

**Anexo 3. Especificación de requisitos de software.**

### 1.1 Requisitos funcionales

| No | Funcionalidad | Descripción | Complejidad | Prioridad |
|---|---|---|---|---|
| 1 | Proponer tipo de control | El analista de la AGR después de analizar los posibles riesgos propone al administrador del CADI la creación de un nuevo tipo de control. | Alta | Requerido |
| 2 | Buscar propuesta de tipo de control | Se muestra la propuesta de creación de un tipo de control con lo que se necesita: Canales correspondientes, alcance que comprenderá (nacional o local), eventos de ejecución (automático o por el usuario o ambos) y la relación que tendrá con otros módulos. | Media | Requerido |
| 3 | Actualizar estado de propuesta. | El administrador del CADI debe modificar el estado de las propuestas realizadas por los analistas cuando crea el tipo de control. | Media | Requerido |
| 4 | Insertar tipo de control. | Se procede a la creación del tipo de control con los parámetros determinados como son: Definir tablas y sus relaciones, canales, eventos de ejecución, alcance, relación con otros módulos y el azar. | Alta | Requerido |
| 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. | Alta | Requerido |
| 6 | Eliminar criterios | Permite que se elimine un criterio de selectividad determinado. | Media | Requerido |
| 7 | Insertar criterio. | Permite insertar un nuevo criterio de selectividad. | Alta | Requerido |
| 8 | Modificar criterio | Permite modificar un criterio atendiendo a sus condiciones y datos generales. | Alta | Requerido |
| 9 | Eliminar condiciones. | Permite que se elimine una condición determinada teniendo en cuenta los valores correspondientes de los mismos. | Media | Requerido |
| 10 | Insertar condiciones | Permite insertar una condición determinada teniendo en cuenta sus valores. | Alta | Requerido |

| | | | | |
|---|---|---|---|---|
| 11 | Modificar condiciones | Permite modificar una condición determinada teniendo en cuenta sus valores. | Alta | Requerido |
| 12 | Eliminar valores. | Permite eliminar el valor o los valores que determina el analista. | Media | Requerido |
| 13 | Insertar valores. | Permite insertar el valor o los valores por el analista. | Alta | Requerido |
| 14 | Modificar valores. | Permite realizar cambios en los valores por los analistas. | Alta | Requerido |
| 15 | Registrar resultados | Luego de aplicar la selectividad, se registran los resultados obtenidos para poder analizar la efectividad de los criterios. | Alta | Requerido |
| 16 | Mostrar reportes de la información obtenida. | Brindar las informaciones recopiladas en la Selectividad que permitan realizar un mejor análisis de blancos. | Alta | Requerido |
| 17 | Realizar Premonitoreo. | Se realiza para verificar el comportamiento que tendrán los criterios vigentes ante un tipo de control determinado. | Alta | Requerido |

| | | | | |
|---|---|---|---|---|
| 18 | Porciento de afectación. | El sistema debe permitir modificar el porciento azar que se había especificado en la creación del tipo de control para determinar el nivel de afectación por cada aduana. | Baja | Requerido |
| 20 | Modificar tipo de control | El sistema debe permitir modificar los campos del tipo de control cuando fue creado, teniendo en cuenta el alcance, relación con otros tipos de control, tablas y canales. | Alta | Requerido |

**Requisitos funcionales**