# Universidad de las Ciencias Informáticas

**Facultad 3**


# Título: Diseño e Implementación del Módulo Tablas de Control

Trabajo de Diploma para optar por el título de

Ingeniero Informático

**Autor:** Gilberto Francesena Pérez

**Tutor:** Ing. Raymond Weeden Gamboa

Junio 2011

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al <nombre área> 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: Gilberto Francesena Pérez

_____

Tutor: Ing. Raymond Weeden Gamboa

_____

DATOS DE CONTACTO

Ing. Raymond Weeden Gamboa (rweeden@uci.cu)

Graduado de Ingeniería en Ciencias Informáticas desde el año 2009. Actualmente labora como diseñador del Módulo Tablas de Control en el proyecto Aduana.

**AGRADECIMIENTOS**

*A todos los que han tenido que ver con mi formación como profesional.*

## DEDICATORIA

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

**RESUMEN**

El presente trabajo está dirigido a lograr el diseño e implementación del módulo Tablas de Control (TC) surgido por la necesidad de relacionar las diferentes informaciones que interactúan en todos los módulos del sistema del proyecto de la Aduana. Consiste en la automatización de las operaciones que se realizan sobre las Tablas de Control en el Sistema para la Gestión Integral Aduanera (GINA) de la Aduana General de la República de Cuba.

El objetivo que se persigue con el trabajo es desarrollar el módulo TC capaz de gestionar los nomencladores y codificadores dinámicamente a través de una interfaz única logrando una mejor organización de la información. Con la implantación de esta nueva versión, este módulo queda integrado a un Sistema en ambiente WEB, que cumple con las políticas de programación actual. Se logra una estandarización de categorías y mejora las condiciones de trabajo de los usuarios del mismo. Se logran automatizar opciones que aún quedaban sin programar, con su implantación será mucho más asequible a los usuarios del Sistema la gestión de los nomencladores.

Esta solución proporciona una mejora considerable en las condiciones de trabajo de los especialistas de las distintas áreas de la Aduana que son usuarios del sistema; ya que logra optimizar la labor de los funcionarios y agilizar el trámite aduanero, se disminuyen al mínimo los errores que puedan existir por mala manipulación de los códigos o desconocimiento de los mismos.

PALABRAS CLAVE

Nomencladores, codificadores, interfaz única, Tablas de Control, automatización.

TABLA DE CONTENIDOS

**AGRADECIMIENTOS** ..... **I**

**DEDICATORIA** ..... **II**

**RESUMEN** ..... **III**

**INTRODUCCIÓN** ..... **6**

**CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA** ..... **10**

1.1 Introducción ..... 10
1.2 Estado del Arte ..... 10
1.3 Lenguajes, modelo y herramientas. ..... 17
1.4 Conclusiones del capítulo ..... 25

**CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA** ..... **26**

2.1 Introducción ..... 26
2.2 Propuesta de solución. ..... 26
2.3 Módulo Tablas de Control. ..... 26
2.4 Generalidades ..... 27
2.5 Requisitos Funcionales ..... 28
2.6 Diseño ..... 28
2.7 Escenario Dar Alta ..... 29
2.8 Escenario Modificar ..... 32
2.9 Escenario Cerrar Registro ..... 35
2.10 Escenario Búsqueda Avanzada. ..... 38
2.11 Diseño de la Base de Datos ..... 40
2.12 Patrón de diseño fundamental que se aplicó. ..... 43
2.12.1 Model-View-Controller ..... 43
2.12.2 Adapter ..... 45
2.12.3 Singleton ..... 45
2.12.4 Bajo acoplamiento ..... 46

2.13 Conclusiones del capítulo: ..... 46
**CAPÍTULO 3: IMPLEMENTACIÓN Y VALIDACIÓN DE LA SOLUCIÓN ..... 47**

3.1 Introducción. ..... 47
3.2 Diagrama de Componentes. ..... 47
3.3 Diagrama de Despliegue. ..... 48
3.4 Resultado esperado: Módulo Tabla de Control ..... 49
3.5 Estándares de codificación ..... 51
3.6 Clases de la aplicación. ..... 52
3.7 Tratamiento de excepciones ..... 54
3.8 Características esenciales ..... 55
3.9 Aporte práctico ..... 56
3.10 Validación de la solución implementada. ..... 56
3.11 Pruebas unitarias y funcionales realizadas. ..... 56
3.12 Pruebas pilotos realizadas en un entorno real de despliegue. ..... 57
**CONCLUSIONES ..... 58**

**RECOMENDACIONES ..... 59**

**BIBLIOGRAFÍA ..... 60**

**ANEXOS ..... 61**

Anexo 1 ..... 61
Anexo 2 ..... 74
Anexo 3 ..... 80
**GLOSARIO ..... 81**

# INTRODUCCIÓN

El creciente desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC) han marcado un cambio significativo en la sociedad con respecto a la información. El manejo de la información es cada vez más dependiente de las TIC, grandes volúmenes de información y la necesidad de utilizar y acceder a estas de manera instantánea son de vital importancia a nivel mundial. En Cuba el uso de las TIC está jugando un papel fundamental en las empresas logrando grandes mejoras en sus operaciones y beneficios para la economía.

La informatización de las empresas cubanas es una realidad hoy donde la Aduana General de la República de Cuba (AGR) está enfrascada en la automatización de sus procesos. La AGR creada el 5 de febrero de 1963 es el órgano encargado del control de tráfico nacional e internacional de medios de transporte, mercancías y viajeros ya sea vía marítima aérea o postal, garantizando la seguridad y protección de la sociedad y de la economía del país.

A diario entran y salen personas o mercancías por los puertos y aeropuertos del país generando grandes volúmenes de información, esta es almacenada en sistemas computacionales con bases de datos relacionales. Con el objetivo de mejorar los servicios y el control de la información se inicia la automatización de los procesos que tienen lugar en las distintas sucursales aduanales del país.

En la primera mitad de la década del 90 se implanta el sistema SIDUNEA (Sistema Aduanero Automatizado) creado por Conferencia de Naciones Unidas para el Comercio y Desarrollo (UNCTAD). SIDUNEA presentaba un conjunto de módulos que resolvían en gran medida a las necesidades que presentaban las aduanas del país en cuanto a la automatización de sus procesos, sustituyendo la mayoría de los procedimientos que se hacían de forma manual, pero no todos se pudieron implantar por incongruencias en su concepción con las características específicas del despacho de mercancías en Cuba y por el elevado costo de licencia y mantenimiento del sistema.

A raíz de estas dificultades en el Centro de Automatización para la Dirección y la Información de la AGR (CADI) se desarrolló el Sistema automatizado de despacho Mercantil (SADEM). Para la explotación del SADEM (Sistema automatizado de despacho Mercantil) se hizo necesario el uso de codificadores o nomencladores facilitando la flexibilidad del sistema, además de tener organizada la información y así asegurar la consistencia de los datos.

Según los estudios realizados hasta el momento en la etapa que se describe, se han detectado diferentes problemas en el SADEM que traen dificultades a los encargados del trabajo de este módulo y resta flexibilidad a las operaciones a realizar, los mismos son los siguientes:

- La herramienta Reflection es la utilizada por el sistema, lo cual trae dificultades a los encargados del trabajo y resta flexibilidad a las operaciones a realizar.
- Una problemática existente es el mayoritario y dependiente uso del teclado ante el ratón.
- Existencia de opciones que aún se encuentran sin automatizar.
- Uso de un sistema multiusuario el cual trabaja directamente en la Base de Datos lo cual implica un riesgo de seguridad, una recarga del trabajo del servidor y no permite la conexión de muchos usuarios.
- Dificultad a la hora de actualizar las versiones a todos los clientes.

Por estas dificultades se decide cambiar el sistema a plataforma Web, surgiendo así el Sistema Único de Aduanas (SUA).

En el curso 2003-2004 se creó el proyecto Sistema Único de Aduana (SUA) entre la Universidad de las Ciencias Informáticas (UCI) y el CADI con el objetivo de desarrollar nuevas soluciones que cumplan con las necesidades de los trabajadores de la Aduana. SUA estaba estructurado por varios módulos entre ellos el de Tablas de Control que tenía como objetivo:

- Actualizar la información manteniendo el histórico y la consistencia de los datos.
- Cada acción en el sistema sea auditable.
- Permita obtención de reportes para consultas.
- La información sea replicada a las unidades y publicada para terceros.

Dificultades que presentaba el módulo Tablas de Control del SUA:

- Programación estructurada.
- Base de Datos desordenada, no existía una BD de TC fijada,
- Las tablas de TC la mayoría sueltas sin relación
- Mal diseño de la interfaz Visual,
- Existía un formulario para cada una de las TC existentes

Todo esto provocaba falta de seguridad, dificultades a la hora de programar, redundancia de código, inconsistencia en los datos.

En la actualidad estos sistemas no responden a las necesidades que van surgiendo, a diario gran número de datos son generadas por estos sistemas creando dificultad al acceder a estas, tampoco comparte una base de dato única con los demás sistemas provocando redundancia de información. No han logrado mediante una interfaz única gestionar los nomencladores existentes lo que implica que cada vez que se crea un nomenclador sea necesario re-implementar parte del sistema. No permiten mantener un historial, lo que hace muy complicado realizar los procesos ordenados que dependen mucho de los nomencladores.

Por estas dificultades se está desarrollando el Sistema para la Gestión Integral Aduanera (GINA) compuesto por varios módulos encargados de automatizar todos los procesos aduanales. El módulo Tabla de Control (TC) es uno de ellos siendo indispensable para el funcionamiento de los diferentes módulos del sistema que necesitan acceder a estos datos de forma flexible mediante una interfaz única. Con la automatización de las tablas, se logra agilizar el trámite aduanero y optimizar la labor de los funcionarios. Así como se disminuye al mínimo los errores que puedan existir por mala manipulación de los códigos o desconocimiento de los mismos. Además se puede conocer los diferentes nomencladores con la información que estos poseen y garantizar el histórico de los datos. Creando una interfaz única para las Tablas de Control, no es necesario modificar o re-implementar el sistema ya que todo quedaría en mera configuración.

Ante la situación expuesta anteriormente surge el siguiente **problema científico**: ¿Cómo gestionar los nomencladores y codificadores del Sistema para la Gestión Integral Aduanera (GINA) dinámicamente y las relaciones existentes entre ellos, a partir de una interfaz única?

Debido a lo anteriormente planteado se tiene como **objeto de estudio**: Proceso de estandarización de codificadores y nomencladores en las Tablas de Control.

Como **campo de acción** abarca el proceso de gestión de las Tablas de Control de la AGR.

Para darle solución a la problemática planteada se establece como **objetivo general** implementar un sistema que permita gestionar la información asociada a las Tablas de Control, Nomencladores y la interacción con los demás módulos del GINA.

Para cumplir con los objetivos y lograr una solución adecuada del problema se plantean las siguientes **tareas de investigación:**

- Estudio de las herramientas para darle cumplimiento a los objetivos.

- Modelar el diseño de la aplicación.
- Realizar el modelo de Datos.
- Implementar las vistas del Diseño.
- Elaborar diagrama de despliegue.
- Realizar la implementación del modelo.
- Validar la solución propuesta.

**Estructuración del trabajo.**

El trabajo consta de tres capítulos que cubren la fundamentación teórica, características del sistema e implementación y validación de la solución, además de las conclusiones, recomendaciones, bibliografía y el glosario de términos. A continuación un resumen de cada capítulo:

**Capitulo1: Fundamentación teórica:** En este capítulo se explican las metodologías, los lenguajes usados, las herramientas utilizadas para el desarrollo de la aplicación y una breve panorámica sobre los Sistemas de Información.

**Capitulo2: Características del sistema:** En este capítulo se incluye un breve análisis del funcionamiento actual del negocio, características del módulo Tablas de Control y detalles relacionados con el diseño.

**Capitulo3: Implementación y validación de la solución:** En este capítulo se detalla el resultado final de la solución, características esenciales así como el aporte práctico de TC. Se muestran los elementos por los que se validó la solución implementada

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

## 1.1 Introducción

En el presente capítulo se detallan los elementos fundamentales asociados al dominio del problema. Se realiza una descripción de los sistemas que se utilizan en Cuba y en el resto del mundo para llevar a cabo la gestión de nomencladores.

Un nomenclador es un sistema de clasificación y de codificación aplicado para designar conceptos fundamentales tales como países, aranceles, tarifas, monedas, plazos, aduanas, modos de transporte, ministerios, formas de pago, entidades, etc., que facilitan la flexibilidad del sistema y la posibilidad de actualización rápida.(1)

También se puntualizan aspectos importantes de las herramientas, metodología y lenguaje que se utilizarán para el desarrollo del producto, entre otros aspectos importantes que guían el trabajo hacia la obtención de la solución.

## 1.2 Estado del Arte

### ERP

La Planeación de Recursos Empresariales (Enterprise Resource Planning, ERP) es un sistema compuesto por un conjunto de módulos funcionales estándar y que son susceptibles de ser adaptados a las necesidades de cada empresa.

Los Sistemas de Gestión Empresariales ERPs almacenan los datos correspondientes a cada una de las entidades del negocio de forma coherente, de forma que no haya duplicados e inconsistencias, manteniendo un formato homogéneo y facilitando que los datos sean fácilmente accesibles y procesables por las distintas partes o módulos del ERP. En todo el mundo se han desarrollado software ERP que integran todos los departamentos que gestionan la información referente en las entidades empresariales y unidades presupuestadas con subsistemas que implementan todas las funciones necesarias para el funcionamiento eficiente de la entidad. (3)

### OpenBravo

Software desarrollado por la Universidad de Navarra en España, sistema de gestión empresarial integrado (ERP) líder, en software libre y entorno web, dirigido a pequeñas y medianas empresas.

Características:

- Implementado en el lenguaje Java.
- Aplicación completamente web que ha sido desarrollada siguiendo el modelo MVC (Model, View, Control).

- Soporte para bases de datos PostgreSQL y Oracle.
- Se ejecuta sobre Apache y Tomcat.


*Figura 1. Arquitectura del software Openbravo ERP.*

Como se muestra en la figura1 la parte fundamental de OpenBravo es su “Gestión de maestros” que se encarga de gestionar toda la información que será accesible al resto de los módulos, permitiendo de esta manera un intercambio de información estandarizado entre todos ellos.

El módulo Gestión de Datos Maestros es la parte fundamental del sistema, dado que centraliza todos los datos que deben compartir los diferentes módulos de la aplicación. Aquí se mantienen los datos de productos, personas y empresas, que se van a usar en el resto de la aplicación.

El objetivo de este módulo es garantizar la coherencia y consistencia de todos esos datos que se van a usar en el resto de módulos, al mismo tiempo que se asegura la trazabilidad de los procesos de la empresa: mantiene una única codificación, evita duplicidades y comparte la información relevante entre todas las áreas de la empresa.

El principal inconveniente que podemos detectar de cara a la implantación de esta solución en la Aduana proviene de la ausencia de un desarrollo vertical de la aplicación para empresas del sector aduanero.

Se tiene acceso al código fuente, pero igualmente se debería realizar el desarrollo de las funcionalidades específicas que precisa la aduana del país. Una posibilidad más, es la de contratar con la propia empresa desarrolladora la programación de las funcionalidades.

Este desarrollo eleva unos costes en tiempo de desarrollo, en dinero para pagar a los analistas y programadores que desarrollen la verticalización y en tiempo global para realizar la implantación completa del sistema. (5)

**SAP**

Software desarrollado en la Ciudad de Mannheim, Alemania, por antiguos empleados de IBM. La corporación se ha desarrollado hasta convertirse en la quinta más grande compañía mundial de software. Cuenta con el módulo Controlling (CO), que permite el control de los gastos generales, costes de producto, cuenta de resultados y centros de beneficio (Maestre, 2008).

Características:

- Implementado en NET y WebSphere.
- SAP también ofrece una nueva plataforma tecnológica denominada SAP NetWeaver, esta plataforma tecnológica convierte a SAP en un programa Web-enabled, lo que significa que estaría totalmente preparado para trabajar con él mediante la web.
- Trabaja sobre el sistema operativo Windows.
- Soporte para bases de datos Oracle.

El componente de SAP NetWeaver permite consolidar, sincronizar, distribuir, publicar y gestionar centralmente cualquier tipo de datos maestros en un entorno de aplicaciones heterogéneo con una solución única.

La aplicación de gestión SAP NetWeaver es una plataforma conformada por varias herramientas enfocadas a optimizar y sincronizar los recursos informáticos con los requisitos particulares a sus estrategias y aplicaciones de gestión empresarial.

SAP NetWeaver ofrece:

- Unificar los datos - garantizar que los datos maestros sean precisos, sin duplicados y normalizados.
- Integrar procesos - hacer que diferentes aplicaciones y sistemas de socios de negocio funcionen en combinación.
- Administrar la información de negocios - aumentar la visibilidad y el acceso a datos empresariales estructurados y sin estructurar. (6)

Principales inconvenientes:

- No se ajusta a las necesidades de la Aduana del país
- Es un programa privativo, elevados coste de licencia.
- No ofrece la posibilidad de acceder al código fuente del producto para su personalización.

**SIDUNEA**

El Sistema computarizado de Aduanas SIDUNEA es una herramienta informática que se ha introducido en varios países, con el objeto de mejorar el comercio internacional, a través de medidas que incluyen reformas a la práctica administrativa ya existente.

Los módulos principales del sistema SIDUNEA incorporan tareas de: administración de sistemas, configuración nacional (regulaciones, tarifas, códigos, etc., de un país en particular), procesamiento de la declaración aduanera, evaluación de riesgos (selectividad), contabilidad, ingreso de información por parte de interesados, empleo de un documento administrativo y despacho electrónico de bienes.

El sistema SIDUNEA emplea códigos internacionales y estándares desarrollados por la Organización Internacional para la Estandarización (ISO), por la Organización Mundial de Aduanas (OMA) y por la Organización de Naciones Unidas (ONU). El sistema SIDUNEA puede ser configurado para adaptarse en los requerimientos operacionales de cualquier administración aduanera, pues su sistema de configuración permite a una aduana definir la información opcional, condicional y obligatoria que considere necesaria.

A principios de los años 1990 se introdujo en Cuba el sistema SIDUNEA, versión 2.51 y a partir de 1996 se actualizó a la versión 2.66. Sin embargo se comprobó que al no ser propio, suprimía las posibilidades de desarrollo al comercio.

El módulo MODSDI (tablas de control) del SIDUNEA utilizado por los funcionarios de aduana, permite consultar la información sobre las tablas de referencia del sistema como son Países, Códigos de Embalaje, Clasificación.

Arancelaria, Aduanas, Acuerdos, Empresas, etc. Incluyendo también una serie de reportes que permiten hacer un seguimiento a las declaraciones. (8)

Deficiencias del SIDUNEA:

- Elevados costos de capacitación, licencia, mantenimiento, soporte y actualización del sistema.
- No todos los Módulos se pudieron implantar por incongruencias en su concepción con las características específicas del despacho de mercancías en Cuba, por lo que se hizo necesario la elaboración de un conjunto de programas o aplicaciones complementarias al mismo, que dieran solución a estas especificidades, pero existiendo aún un gran número de registros y operaciones que se realizaban manualmente.
- Existencia en Cuba de la dualidad monetaria en las operaciones comerciales.

**SADEM**

A partir del estudio realizado al proceso de control de las importaciones y exportaciones comerciales en 1996, se decidió que era necesario y de vital importancia, desarrollar el SADEM (Sistema automatizado de despacho Mercantil), este sistema se realizaría basado en las características del comercio cubano, respondiendo a los intereses de la Aduana cubana.

El SADEM se implantó el primero de enero del 2001 en todas las unidades del Sistema de Órganos aduaneros (SOA), con un mínimo de módulos a los que se les fueron sumando el resto paulatinamente. Para la explotación del SADEM fue necesario el uso de codificadores o nomencladores los cuales llamamos Tablas de Control.

Las tablas de control facilitan la flexibilidad del sistema, además de tener organizada la información y así asegurar la consistencia de los datos. La estructura y contenido de los mismos estará en correspondencia con las necesidades de análisis y agrupamiento de la información que se requiera durante el procesamiento automatizado.

SADEM está formado por catorce módulos uno de los módulos es el de las Tablas de Control, que tenía como propósito relacionar las diferentes informaciones de carácter constante que serían usadas en el sistema, organizar estas informaciones a través de codificadores y nomencladores asignándole un código para facilitar su tratamiento automatizado.

El módulo de las tablas de control del SADEM es el encargado de:

- Crear, validar y actualizar toda la información referente a las tablas de control del sistema.
- Recibir información puntual por disco de las exenciones y bonificaciones que vienen del ministerio de Finanzas para actualizar en una misma tabla.
- Brindar listados del contenido de las tablas tanto por pantalla como por impresora.
- Facilidades para establecer un esquema de Auditoría y control sobre los resultados.
- Se controla la historia de las actualizaciones de cada artículo con el uso de la fecha de inicio y fecha de fin para conocer en qué momento estuvo vigente.
- Brinda la posibilidad de que los encargados de la actualización de las tablas en las unidades pueda ser cualquier inspector con conocimientos mínimos de computación al tener un ambiente amigable que facilita la actualización por medio de pantallas de captación de datos.
- En el momento de conformar las tablas para enviarlas a las unidades no necesariamente tiene que ser un especialista de la dirección de informática esta tarea se ha simplificado y actualmente el

sistema permite que cualquier especialista de Técnicas Aduaneras las actualice por la opción correspondiente.(1)

Deficiencias del SADEM:

Debido al gran volumen de información que manejan en entradas de datos, el uso del teclado es mayoritario ante el ratón, existencia de opciones que aún se encuentran sin automatizar.

El módulo Tablas de Control se encontraba implantado sobre el sistema operativo Unix Santa Cruz Operation, por lo que fue necesaria su reprogramación, por presentar dificultades en la instalación del mismo en los servidores actuales con los que cuenta la aduana. Además opera por medio de la herramienta Reflection, lo cual trae dificultades a los encargados del trabajo con este módulo, así como resta flexibilidad a las operaciones a realizar.

Debido a las dificultades existentes se decidió que era necesario desarrollar un sistema automatizado único, que contemplara todos los procesos de la Aduana para implantarse en todas las aduanas del país, este sistema debía responder a nuestros intereses y en el cual todo funcionario tuviera sus requerimientos satisfechos.

Debido a las deficiencias existentes y por concepción general del sistema se decide cambiar el sistema a plataforma Web, surgiendo así el Sistema Único de Aduanas (SUA).

**SUA**

El Sistema Único de Aduana (SUA) tiene como objetivo automatizar el procesamiento informativo referente a todas las operaciones que conforman los diferentes procesos, ya sea de Medios de Transporte Internacional, Importaciones y Exportaciones con y sin carácter comercial, Bultos Postales y Viajeros y las Tablas de Control en ambiente WEB, por las facilidades que brinda a los usuarios . SUA es un sistema en el cual todos los módulos validan y controlan las entradas de datos contra los nomencladores y clasificadores.

Los principales objetivo el módulo Tablas de Control de SUA son:

- Crear, validar y actualizar toda la información referente al control países, documentos, productos, regímenes, medios de transporte, atraques, muelles, personas, aduanas y muchas otras (ver listado de tablas)
- Brindar listados del contenido de las tablas tanto por pantalla como por impresora.
- Facilidades para establecer un esquema de Auditoria y control sobre los resultados.

- Se controla la historia de las actualizaciones de cada artículo con el uso de la fecha de inicio y fecha de fin para conocer en qué momento estuvo vigente.
- Brinda la posibilidad de que los encargados de la actualización de las tablas, con conocimientos mínimos de computación, tengan un ambiente amigable para realizar la actualización.
- En el momento de conformar las tablas para enviarlas a las unidades no necesariamente tiene que ser un especialista de la dirección de informática, esta tarea se ha simplificado y actualmente el sistema permite la replicación de las mismas o la instalación por la opción correspondiente.

Las tablas son actualizadas centralmente en la AGR por especialistas de las diferentes direcciones y además son publicadas en el sitio en Internet con el objetivo de que usuarios no aduaneros se informen acerca del estado de la información que se guarda en dichas tablas, que como ya se explicó tienen que ver con todo lo que controla la Aduana en todos sus procesos, así como los códigos que se utilizan.

Dificultades de SUA:

Programación del SUA estructurada, el código abierto al público, Base de Datos desordenada, no existía una BD de TC fijada, las tablas de TC la mayoría sueltas sin relación y la relación entre sus datos se manejaba mediante el código, mal diseño de la interfaz Visual, existían un formulario para cada una de las TC existentes, y el número de estas es bastante grande. Esto provocaba la repetición de código en los archivos del formulario de algunas TC, ya que hay muchas que se asemejan en funcionalidad. No estaban programadas las llamadas Tablas con relaciones, que son aquellas que mantienen relaciones de mucho a mucho con otras TC, debido al modelo de datos (BD) existentes en ese momento no existía un modelo código generalizado por el cual guiarse todo eso provoca falta de seguridad en el sistema.

Por estas deficiencias se está desarrollando el Sistema para la Gestión Integral Aduanera (GINA) compuesto por varios módulos encargados de automatizar los procesos aduanales.

**GINA**

GINA es desarrollado en base a solucionar los problemas presentados en el sistema SUA, se desarrolla también en la UCI en conjunto con el CADI, a diferencia del SUA, este emplea una metodología de desarrollo más robusta, programación orientada a objetos y patrones de diseño. El SUA se dejara de desarrollar por parte de la UCI y de usar por parte de la AGR, para dar paso a GINA que pasaría hacer una versión mejorada del SUA.

Esta solución es un sistema Web que se desarrolla sobre el sistema operativo GNU/Linux. Esta solución es multiplataforma por lo que su funcionamiento puede ser en cualquier plataforma tanto GNU/Linux como Windows, no depende de una en específico. A su vez son multiplataforma también el lenguaje utilizado

PHP, los marcos de trabajo o frameworks de desarrollo Symfony y Extjs y el gestor de base de datos Oracle.

El sistema en cuestión seria uno de los pasos más importantes para la actualización y modernización de la AGR, con una configuración y uso más amigable, tecnológicamente más moderno, más robusto y mejor desarrollado. Con un conjunto de subsistemas y módulos para todas las áreas de la aduana en desarrollo, por lo cual sería el primer sistema que cubriría todas las necesidades de la aduana.

El GINA compuesto por varios módulos entre los que se encuentra Tabla de Control (TC) siendo indispensable para el funcionamiento de los diferentes módulos del sistema que necesitan acceder a estos datos de forma flexible mediante una interfaz única. Los datos son introducidos con posibilidad de ser modificados, es válido aclarar que en el caso de las modificaciones y eliminaciones se realizan de forma lógica actualizando sólo la fecha de vencimiento del artículo (tomado como experiencia del SIDUNEA y que se aplicó en el SUA posteriormente), esto ocurre en todas las tablas ya que por requerimientos del sistema, la información se guarda durante cinco años y para garantizar la consistencia, se realizan las actualizaciones de esta manera.

Con la automatización de las tablas, se logra agilizar el trámite aduanero y optimizar la labor de los funcionarios. Así como se disminuye al mínimo los errores que puedan existir por mala manipulación de los códigos o desconocimiento de los mismos. Además se puede conocer los diferentes nomencladores con la información que estos poseen.TC debe garantizar el histórico de los datos tomando como experiencia los Sistemas ya estudiados permitiendo que los mismos sean auditables. Creando una interfaz única para las Tablas de Control, no es necesario modificar o re-implementar el sistema ya que todo quedaría en mera configuración.

### 1.3 Lenguajes, modelo y herramientas.

El módulo Tabla de Control forma parte del producto GINA el cual tiene ya definido la tecnología a usar para su desarrollo, por ello, el lenguaje, la metodología y las herramientas a usar se encuentran sujetas a las políticas especificadas ya de antemano para el producto. A continuación se realizará una fundamentación de las mismas.

**Modelo**

Los modelos de desarrollo de software surgen para guiar a las personas implicadas en el desarrollo de software, brindando un conjunto de procedimientos, técnicas y herramientas, de forma que sepan qué hacer en cada momento y cómo alcanzar un producto de alta calidad.

Para el desarrollo de esta solución se aplicó el modelo de desarrollo escogido por parte del centro CEIGE (Centro de Informatización para la Gestión de Entidades) tomando en consideración las adecuaciones realizadas en el departamento de desarrollo de Soluciones Aduaneras específicamente para la fase de diseño. Al aplicar dicho modelo se generarán un conjunto de artefactos para el diseño de la solución, los cuales serían: Diagrama de Clase, Diagrama de Modelo de Datos, Diagrama de Interacción y Actividades con un grupo de adecuaciones definidas en el departamento, Diagrama de Componente y Diagrama de Despliegue.

**Lenguajes**

**PHP**

PHP es el acrónimo de Hypertext Preprocessor, es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor pues funciona en un servidor remoto que procesa la página Web antes de que sea abierta por el navegador del usuario. PHP ha sido especialmente creado para el desarrollo de páginas Web dinámicas y puede ser incluido con facilidad dentro del código HTML. Ha alcanzado gran popularidad y existe una amplia comunidad de desarrolladores y programadores que continuamente implementan mejoras en su código. Se caracteriza por una sencilla integración con múltiples bases de datos y, aunque MySQL es la base de datos que mejor trabaja con PHP, puede conectarse también a PostgreSQL, Oracle entre otras bases de datos. Está dotado de un gran número de funciones predefinidas que simplifican enormemente tareas habituales como descargar documentos, generar imágenes GIF, enviar correos electrónicos, trabajar con cookies y sesiones, establecer conexiones a otros servicios de red y generar documentos PDF. La sintaxis de PHP se basa en otros lenguajes de programación, principalmente en C y Perl, o un lenguaje de tipo C como C++ o Java, se distingue por su facilidad de aprendizaje y uso. Entre los competidores principales de PHP se puede citar a Perl, Microsoft Active Server Pages (ASP), Java Server Pages (JSP) y Allaire ColdFusion. PHP también soporta el uso de servicios que usen protocolos como IMAP, SNMP, NNTP, POP3, HTTP y derivados. Y además puede interactuar con otros protocolos. (Thomson y Welling, 2003) PHP es un lenguaje multiplataforma, permite leer y manipular datos desde diversas fuentes, incluyendo datos que pueden ser ingresados por los usuarios desde formularios HTML, es libre por lo que es una alternativa de fácil acceso para todos, es Orientado a Objetos y además posee una arquitectura extensible. Dentro de las opciones que brinda PHP se encuentra: la simplicidad, la estabilidad y la compatibilidad. (13)

**JavaScript**

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Es un lenguaje que no requiere compilación, lo que se conoce como lenguaje de programación interpretado. Su sintaxis es muy parecida a la del lenguaje Java y el lenguaje C. Actualmente, todos los navegadores interpretan el código JavaScript integrado dentro de las páginas Web. Por tradición, el lenguaje JavaScript se utilizaba para realizar tareas y operaciones en el marco de la aplicación únicamente del lado del cliente, sin acceso a funciones del servidor. Actualmente, existen aplicaciones en JavaScript para el servidor. Aunque JavaScript de cliente y de servidor comparten el mismo conjunto base de funciones y características; en algunos casos se utilizan de distinta forma. De manera general, JavaScript permite crear aplicaciones específicamente orientadas a su funcionamiento en la red Internet. Usando JavaScript se pueden crear páginas HTML dinámicas que procesen la entrada del usuario y que sean capaces de gestionar datos persistentes usando objetos especiales, archivos y bases de datos relacionales. Además, se pueden construir aplicaciones que varían desde la gestión de la información corporativa interna y su publicación en Intranets hasta la gestión masiva de transacciones de comercio electrónico. (13)

**UML para el modelado**.

Unified Modeling Language que se traduce al español como Lenguaje de Modelación Unificado, es un lenguaje gráfico para especificar, construir, visualizar y documentar las partes o artefactos (información que se utiliza o produce mediante un proceso de software). Comprende el desarrollo de software que se basen en el enfoque OO (Orientado a Objetos), utilizándose también en el diseño Web, usa procesos de otras metodologías, aprovechando de esta manera la experiencia de sus creadores, además eliminó los componentes que resultaban de poca utilidad práctica y añadió nuevos elementos. UML se ha convertido en el estándar tan ansiado para representar y modelar la información con la que se trabaja en las fases de análisis y, especialmente, de diseño.

Una de las principales ventajas que brinda UML es que ayuda al usuario a entender la realidad de la tecnología y le posibilidad reflexionar antes de invertir y gastar grandes cantidades en proyectos que no estén seguros en su desarrollo, reduciendo el coste y el tiempo empleado en la construcción de las piezas que constituyen el modelo.

Algunas de las características de UML como lenguaje de modelado estándar son:

- Reemplaza a decenas de notaciones empleadas con otros lenguajes.
- Modela estructuras complejas.
- Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- Comportamiento del sistema: diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas.

**CSS**

El principio de las hojas de estilo consiste en la utilización de un solo documento para almacenar las características de presentación de las páginas asociadas a grupos de elementos. Esto implica nombrar un conjunto de definiciones y características de presentación de las páginas, y activar esos nombres para aplicarlos a una parte del texto. Las hojas de estilo se desarrollaron para compensar los defectos de HTML con respecto a la presentación y al diseño de las páginas. HTML tiene varias etiquetas para modificar la presentación y definir los estilos del texto, pero cada elemento tiene su propio estilo, independientemente de los elementos que lo rodean. Al utilizar hojas de estilo, cuando se necesite cambiar la apariencia de un sitio que tiene cientos de páginas Web todo lo que hay que hacer es editar las definiciones de la hoja de estilo en un solo lugar para cambiar la apariencia del sitio completo. Se denominan "hojas de estilo en cascada" porque se pueden definir múltiples hojas y los estilos pueden aplicarse a todas las páginas (con un sistema predefinido para resolver conflictos).

Las hojas de estilo pueden utilizarse para:

- Lograr una apariencia uniforme de todo el sitio al activar una sola definición de estilo en cada página.
- Cambiar un aspecto en todo el sitio Web con tan sólo editar unas pocas líneas.
- Facilitar la lectura de los códigos HTML ya que los estilos se definen por separado.
- Permitir que las páginas se carguen más rápido ya que hay menos cantidad de código HTML en cada página.
- Posicionar los elementos de la página de una manera más uniforme.(12)

**Marco de Trabajo**

**Symfony**

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, un framework facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa 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. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web.

Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows.

Características y Ventajas:

Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares). • Independiente del sistema gestor de bases de datos. Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos. Basado en la premisa de “convenir en vez de configurar”, en la que el desarrollador solo debe configurar aquello que no es convencional.

Sigue la mayoría de mejores prácticas y patrones de diseño para la web. Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo. Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo. Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros. (13)

**Framework Ext**

La programación con la librería Ext, que extiende la librería YUI e integra AJAX, Prototype y Scriptaculous. Con el tiempo se convirtió en un framework independiente y a principio de 2007 se creó una compañía para comercializar y dar soporte del framework Ext. De esta forma Ext tiene dos tipos de licencias, LGPL y comercial. Básicamente, se puede usar Ext para nuestros desarrollos, pero para obtener soporte se debe tener una licencia comercial.

Ext es neutral al lenguaje que se use en el servidor. Siempre que el resultado se envíe a la página en el formato adecuado, Ext no se preocupará de lo que pase en el servidor. Hay docenas de Widgets a escoger en Ext, incluyendo composiciones automáticas de páginas, pestañas, menús, barras de herramientas, diálogos, vistas en árbol. Proporciona un selector de nodos DOM extremadamente poderoso llamado DomQuery (puede usarse como una librería independiente, pero en el contexto de Ext se usará para seleccionar elementos para poder interactuar con ellos a través de la interfaz Element.Elemnts contiene mucho de los métodos y propiedades de DOM que se necesitará proporcionando una interfaz conveniente, unificada y multinavegador).

Para trabajar con las librerías de Ext es necesario que exista un adaptador, donde esta clase sobre la que luego se define las funciones de las librería (el elemento Ext como tal). A partir de la versión 1.1, Ext proporciona su propio adaptador, ya no es necesario incluir el de YUI, o JQuery o Scriptaculous. (13)

**Herramientas de modelado**

**Visual Paradigm**

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

Esta herramienta soporta hasta la fecha UML 2.1 completo y BPMN, permite realizar ingeniería tanto directa como inversa y es ahí donde más quiero incidir ya que a partir de un modelo relacional en Sql Server, MySql, etc., es capaz de desplegar todas las clases asociadas a las tablas (siguiendo el patrón de diseño Una Clase-Una Tabla) Además, la herramienta es colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto; genera la documentación del proyecto automáticamente en varios formatos como Web o .Pdf, y permite control de versiones y es multiplataforma. (13)

**ER\Studio**

Es una herramienta de modelado de datos fácil de usar y multinivel, para el diseño y construcción de bases de datos a nivel físico y lógico. Direcciona las necesidades diarias de los administradores de bases de datos, desarrolladores y arquitectos de datos que construyen y mantienen aplicaciones de bases de datos grandes y complejas. ER/Studio está equipado para crear y manejar diseños de bases de datos funcionales y confiables. Ofrece fuertes capacidades de diseño lógico, sincronización bidireccional de los diseños físicos y lógicos, construcción automática de bases de datos, documentación y fácil creación de reportes. Funcionalidades:

- Capacidad fuerte en el diseño lógico.
- Sincronización bidireccional de los diseños lógico y físico.
- Construcción automática de Base de Datos.
- Reingeniería inversa de Base de Datos.
- Documentación basada en HTML.
- Un Repositorio para el modelado.

La empresa lanzó al mercado la versión final de ER/Studio, su herramienta insignia para el modelado de datos. ER/Studio versión 7.5 ofrece soporte XML para la generación de esquemas, que permite a los modeladores de datos y desarrolladores de aplicaciones colaborar en iniciativas de arquitectura orientada a servicios (SOA).

ER/Studio 7.5 permite la conversión mejorada para incrementar la precisión durante la migración de modelos de datos desde herramientas tales como CA ERwin Data Modeler, Sybase PowerDesigner y muchas otras. (13)

**Herramienta de desarrollo**

**Netbeans**

NetBeans IDE (Entorno de desarrollo integrado) es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso, contiene las herramientas para que los desarrolladores de software puedan crear aplicaciones desktop, enterprise, web, y aplicaciones móviles, con el lenguaje Java, así como también C/C++, PHP, JavaScript, Groovy, y Ruby.

NetBeans permite crear aplicaciones Web con PHP 5, un potente debugger integrado y además viene con soporte para Symfony. Tiene también soporte para AJAX. NetBeans IDE funciona en sistemas operativos compatibles con la máquina virtual Java: Window, MacOs, Linux. (14)

**Sistemas Gestores de Base de Datos (SGDB)**

**Oracle 11g**

Oracle Database 11g Enterprise Edition ofrece confiabilidad, escalabilidad y desempeño de primer nivel para configuraciones en cluster y en un solo servidor. Ofrece las más completas características para soportar el procesamiento de transacciones más exigente, inteligencia de negocios, y aplicaciones para la administración de contenido. Protección ante las fallas del servidor, fallas del sitio, errores humanos, y reducción del tiempo de baja programado. Protección de datos con seguridad única en el nivel de filas, auditorías detalladas, y encriptación transparente de datos. Incluye data warehousing de alto desempeño, procesamiento analítico online, y características de extracción de datos. Constituye un sistema gestor de bases de datos con características objeto-relacionales. Sus principales características son las siguientes:

- Entorno cliente/servidor.
- Gestión de grandes bases de datos.
- Usuarios concurrentes.
- Alto rendimiento en transacciones.
- Sistemas de alta disponibilidad.
- Disponibilidad controlada de los datos de las aplicaciones.
- Adaptación a estándares de la industria, como SQL-92.
- Gestión de la seguridad.
- Autogestión de la integridad de los datos.
- Opción distribuida.
- Portabilidad.
- Compatibilidad.
- Replicación de entornos.

Provee un control de accesos discrecional, es decir, acceso restringido a la información basado en privilegios.

- Gestiona la seguridad de la base de datos usando:
- Usuarios y esquemas de la base de datos.

- Privilegios.
- Roles.
- Ajustes de rendimiento y cuotas.
- Límites sobre los recursos.
- Auditoría.

Cada usuario posee un dominio de seguridad, que determina:

- Acciones (privilegios y roles) disponibles para el usuario.
- Cuotas sobre tablespaces.
- Límites en los recursos del sistema.

Posee varias estructuras y mecanismos de software para proveer:

- Recuperación de la base de datos ante distintos tipos de fallos.
- Operaciones de recuperación flexibles.
- Disponibilidad de los datos durante las operaciones de backup y recovery.

Utiliza varias estructuras para proveer la recuperación completa de la instancia:

- Redo Log.
- Segmentos de rollback.
- Fichero de control.
- Copias necesarias de la base de datos. (15)

### 1.4 Conclusiones del capítulo

En el presente capítulo se realizó un estudio de los diferentes sistemas que existen en Cuba y el mundo que dan solución a la gestión de nomencladores y codificadores. Se analizó las ventajas y desventajas del uso de las mismas como solución al problema. Se realizó una fundamentación de las herramientas y tecnologías a utilizar en el diseño e implementación de la aplicación.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

## 2.1 Introducción

La solución propuesta está sustentada sobre la base de todo lo que se expone en este capítulo, en la cual se definen las actividades más importantes que posibilitaron la modelación del negocio. Se detallan los requisitos funcionales más importantes que se utilizaron, el diseño de la aplicación, así como las características específicas del sistema.

## 2.2 Propuesta de solución.

Se propone el desarrollo de un sistema informático que logre gestionar las tablas de control con sus relaciones a través de una interfaz única y de forma dinámica generando las pantallas de cada nomenclador automáticamente. El mismo será un módulo del sistema de Configuración GINA que se implementara orientado a objeto ajustándose a la arquitectura definida en dicho sistema.

## 2.3 Módulo Tablas de Control.

El presente módulo surge debido a la necesidad de relacionar las diferentes informaciones de carácter constante que serán usadas en el sistema, además de organizar estas informaciones y gestionar los nomencladores existentes mediante una interfaz única. La estructura y contenido de los nomencladores y clasificadores están en correspondencia con las necesidades de análisis, agrupamiento o desagrupamiento de la información que se requiera durante el procesamiento automatizado, ya que en los mismos se realizan una serie de controles y validaciones que dependen de la organización que se cree para garantizar y facilitar la flexibilidad del sistema, además de tener organizada la información y así asegurar la consistencia de los datos. Estos datos almacenados y tan relacionados a los procesos deben ser introducidos con posibilidad de ser modificados, es válido aclarar que en el caso de las modificaciones y eliminaciones se realizan de forma lógica actualizando sólo la fecha de vencimiento del artículo, esto ocurre en todas las tablas ya que por requerimientos del sistema la información se guarda durante cinco años por lo que para garantizar la consistencia del sistema se realizan las actualizaciones de esta manera. Se debe garantizar además que estas informaciones sean actualizadas por las personas autorizadas. Se puede concluir que para garantizar el objetivo el módulo debe ser capaz de:

- Gestionar los nomencladores y codificadores dinámicamente mediante una interfaz única.
- Actualizar la información manteniendo el histórico y la consistencia de los datos.
- Cada acción en el sistema sea auditable.
- Permita obtención de reportes para consultas.
- La información sea replicada a las unidades y publicada para terceros.

### 2.4 Generalidades

Para el desarrollo de TC se tuvieron en cuenta los siguientes aspectos:

- Los campos comunes para todas las tablas son Código, Descripción, Fecha de Inicio, Fecha de Fin, Created_At (fecha de creación de un registro) y Deleted_At (fecha de eliminación de un registro).

Ejemplo: TC_PROVINCIA.

| ID_PROVINCIA | CODIGO | DESCRIPCION | F_INICIO | F_FIN | CREATED_AT | DELETED_AT |
|---|---|---|---|---|---|---|
| **1** | 01 | Pinar del Rio | 21/10/2008 | 31/12/9999 | 21/10/2008 | 31/12/9999 |
| **2** | 02 | Matanzas | 21/10/2008 | 31/12/9999 | 21/10/2008 | 31/12/9999 |

- Los artículos vigentes serán aquellos en que la fecha de inicio sea menor o igual que la actual y la fecha de fin mayor o igual que la actual o vacía.
- Los artículos en las tablas tienen incluidas f_inicio y f_fin de validez ya que estos no pueden ser eliminados, se actualiza la fecha de fin y este continua en la tabla para efectos de auditoria posteriores.
- Las fechas de inicio y fin de validez son validadas al ser tecleadas para que sean lógicas y no se permite introducir un artículo con vigencia anterior a la fecha en que se está haciendo la actualización. Ni insertar artículos cuya fecha de inicio sea menor que la actual.
- Los campos CREATED_AT y DELETED_AT serán llenados automáticamente con la fecha de creación del registro (fecha actual del sistema) y la fecha de eliminación del registro (infinito 31/12/9999).
- Si las tablas se relacionan con otros datos de otras tablas en una relación, en el momento de la actualización la relación se actualiza también, o sea se debe garantizar la consistencia de los datos.

- En el caso que se modifiquen datos de artículos ya existentes si los campos implican consecuencias a los efectos de auditorías posteriores lo que se hace es repetir el artículo completo con los nuevos cambios y cambiarle la vigencia al anterior. Esto sucede siempre que se modifique la fecha de inicio y en los casos que los datos modificados sean de una importancia tal que sea necesaria su duplicación.

**2.5 Requisitos Funcionales**

Los requerimientos o requisitos funcionales son capacidades, condiciones, que el sistema debe poseer. Son un conjunto de características requeridas por el sistema, que expresan su capacidad de acción; una funcionalidad, que se expresa generalmente en una declaración en forma verbal.

Para el diseño e implementación del módulo Tablas de Control se tomaron una serie de requisitos.

1. Registrar artículos (Dar Alta).
2. Modificar artículos.
3. Cerrar artículos.
4. Buscar artículos.

Artículos: registros o tuplas en cualquier tabla.

Para más información de los casos de uso ver “Especificación de casos de uso v2.1.doc”.

**2.6 Diseño**

En el diseño se modela el sistema de manera que se cumplan los requisitos funcionales y no funcionales. Dentro de los propósitos del diseño están: adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución etc.; además de descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo. Ver anexo 1.

A continuación se muestran los Diagramas de Diseño y una breve descripción de cada uno de ellos. Los Diagramas de Diseño está compuesto por la clase servidor o controlador frontal *sfTcActions*, que re direcciona hacia el *layout.php* y mediante el fichero de configuración *view.yml* se crea la página cliente, el cual va a contener los formularios que son creados mediante las interfaces de extjs y estos a su vez le hacen las peticiones submit o Ajax request a la clase controladora.

### 2.7 Escenario Dar Alta

Consiste en insertar un nuevo artículo en alguna de las tablas. A la hora de dar alta a un requisito hay que tener en cuenta que no existan artículos con el mismo Código. Si no existe el código, agrego el nuevo a la tabla. Se debe tener en cuenta si existe un artículo con el mismo código, este no debe estar vigente ni futuro. Si existe uno que no esté vigente, es decir, que existió en algún momento, y lo que se desea es ponerlo en vigencia se muestra el articulo y lo que se va cambiar que son los campos de las fechas aparecerán en blanco y se consideraría una inserción. Si la tabla se relaciona con otras tablas hay que incluir el nuevo artículo en las tablas que se relacionan. Si existe un artículo vigente con el mismo código no se permite la inserción.

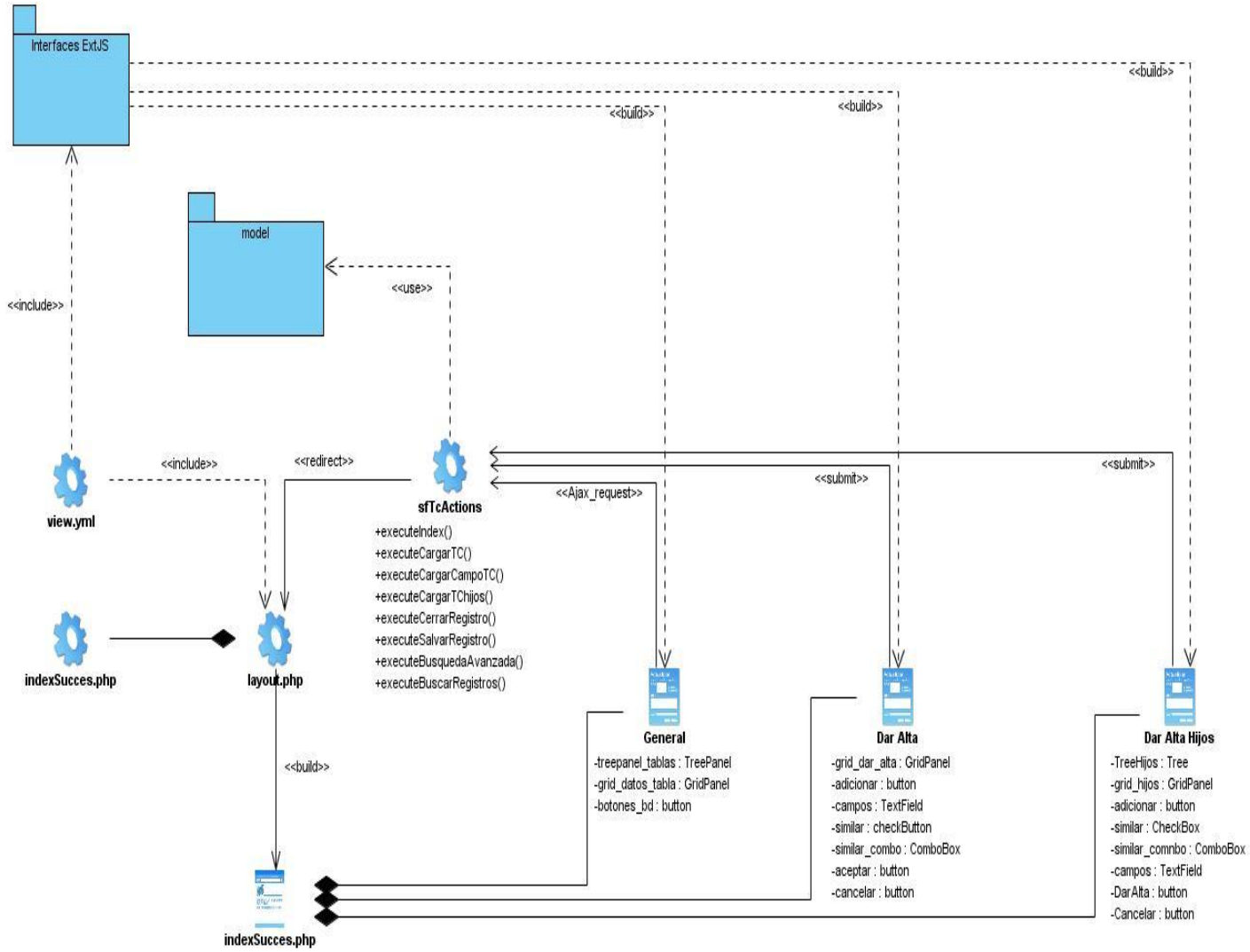



Diagrama de Clases del Diseño (Dar Alta).

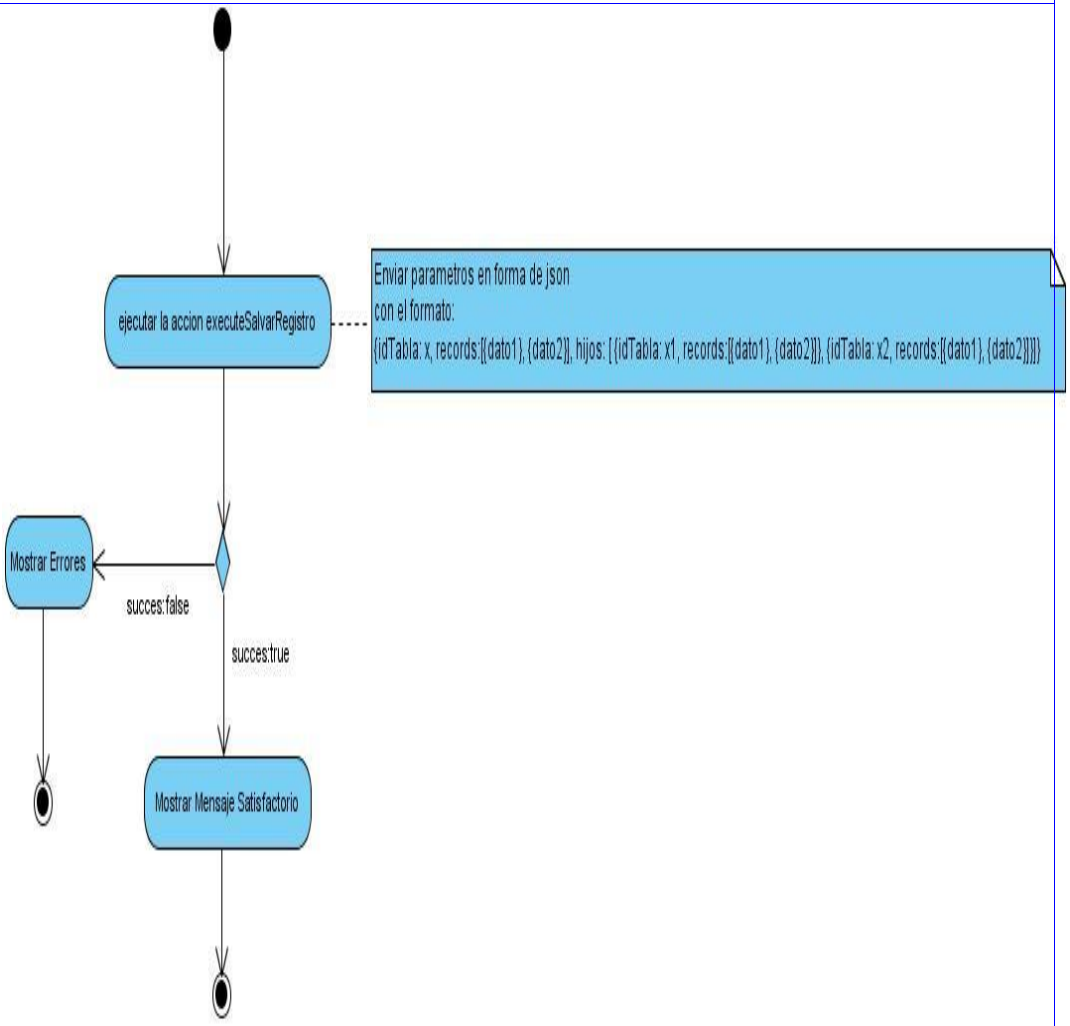
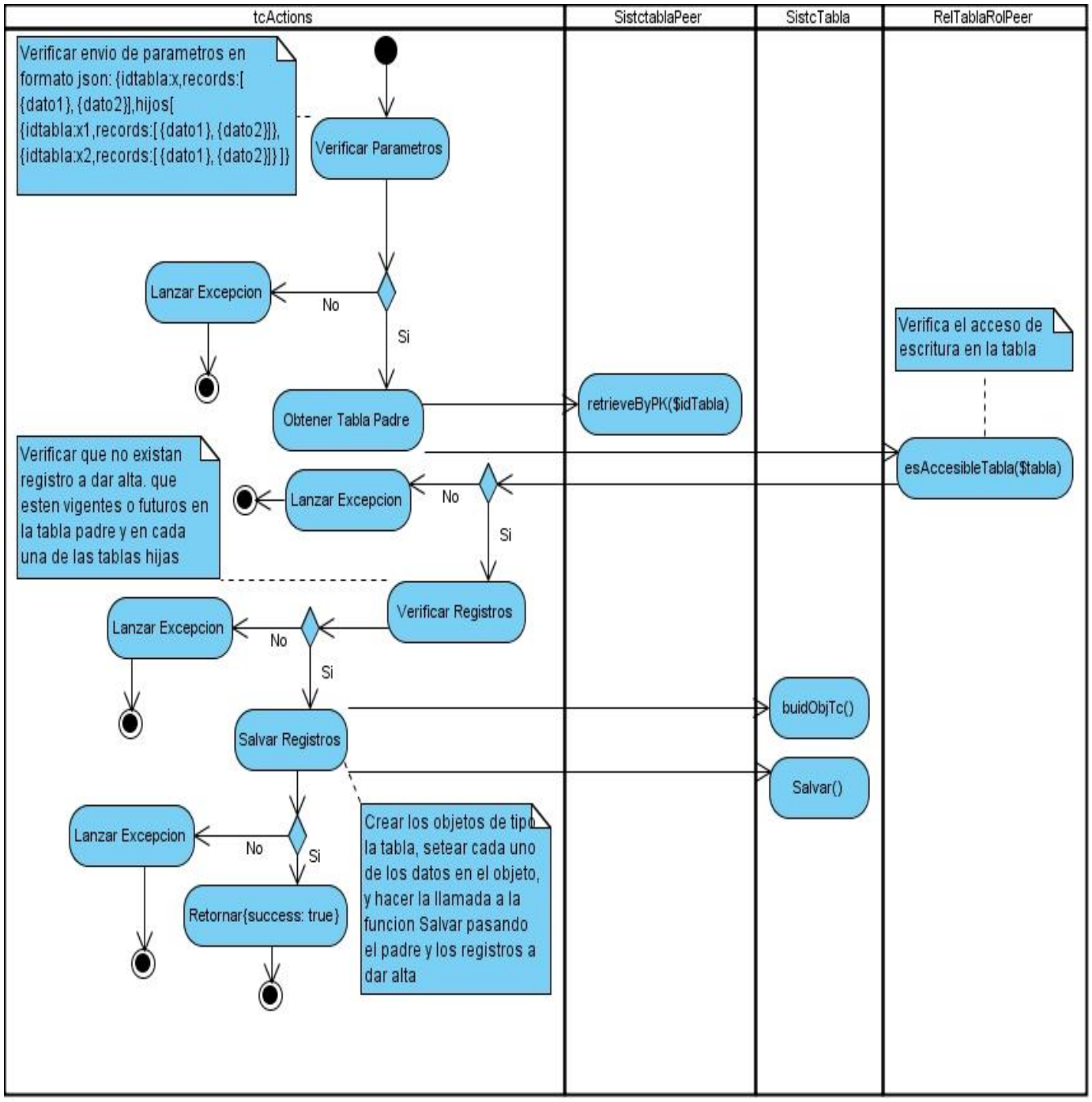
| | |
|---|---|
| Componente | Dar Alta |
| Método | onClick (Aceptar) |
| Propósito | Esta funcionalidad tiene como objetivo registrar un artículo nuevo o poner uno en vigencia o futuro en caso de que estuviera cerrado, junto con sus datos en las demás tablas hijas. |
| Descripción |  |

**Diagrama de interacción y actividades orientado a la acción *executeDarAltaRegistro.***



## 2.8 Escenario Modificar

En caso de que el artículo exista el sistema debe permitir la modificación de uno o varios registros con selección múltiple para modificar varios campos de diferentes registros a excepción del código. El sistema debe permitir realizar una modificación a un artículo que se encuentre vigente y futuro, en caso de que se encuentre un artículo cerrado no se permite la modificación

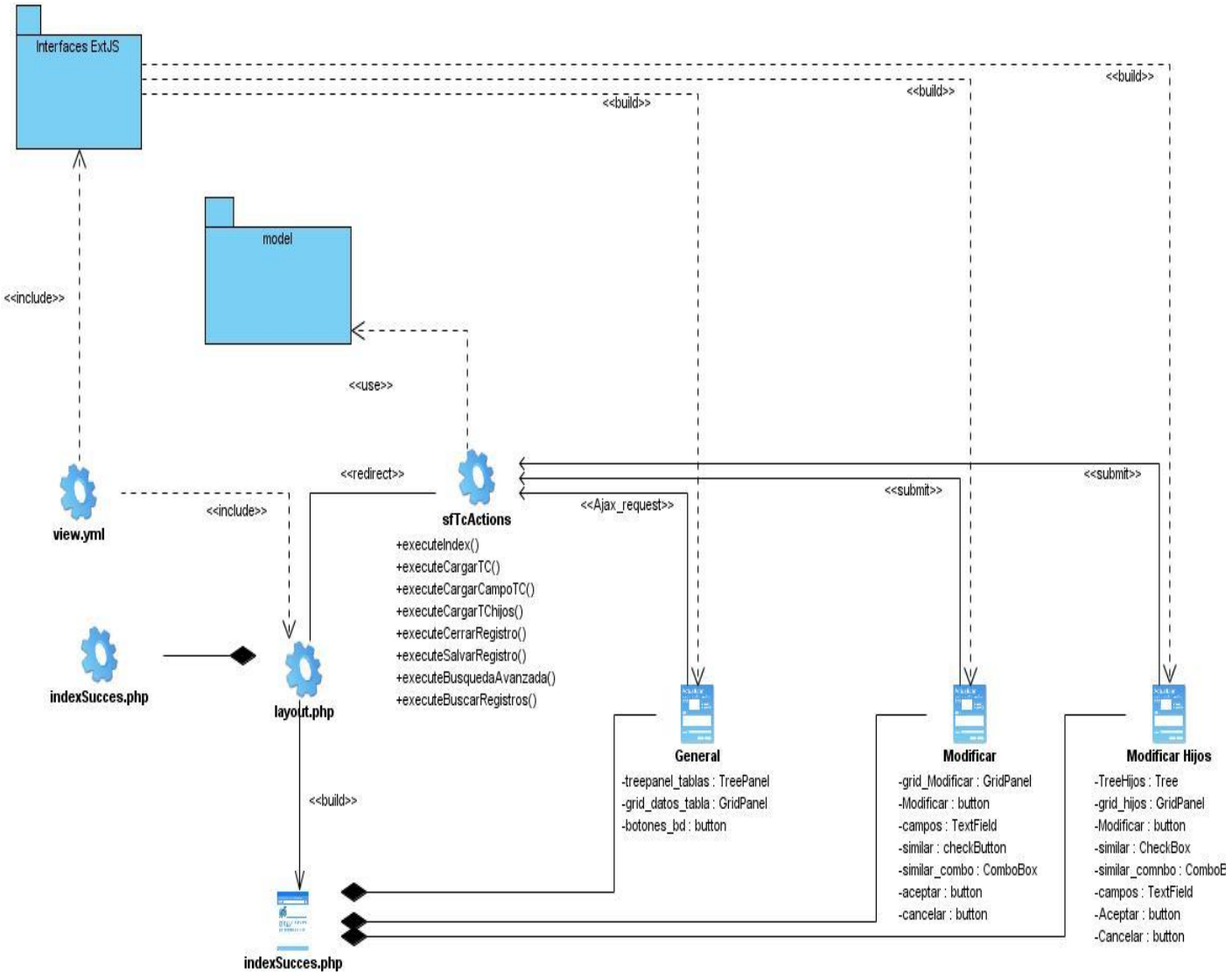



Diagrama de Clases del Diseño (Modificar Registro).

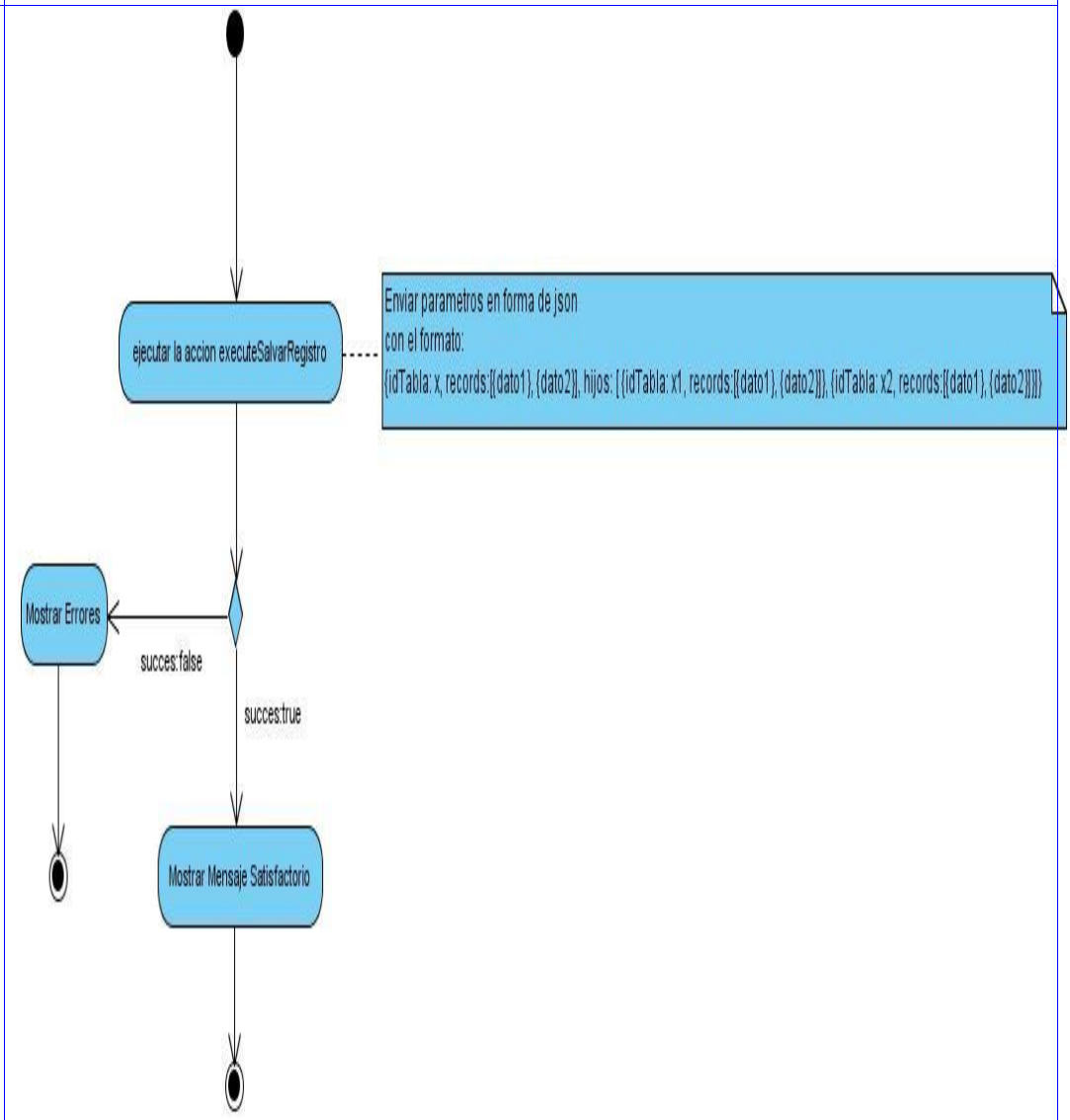
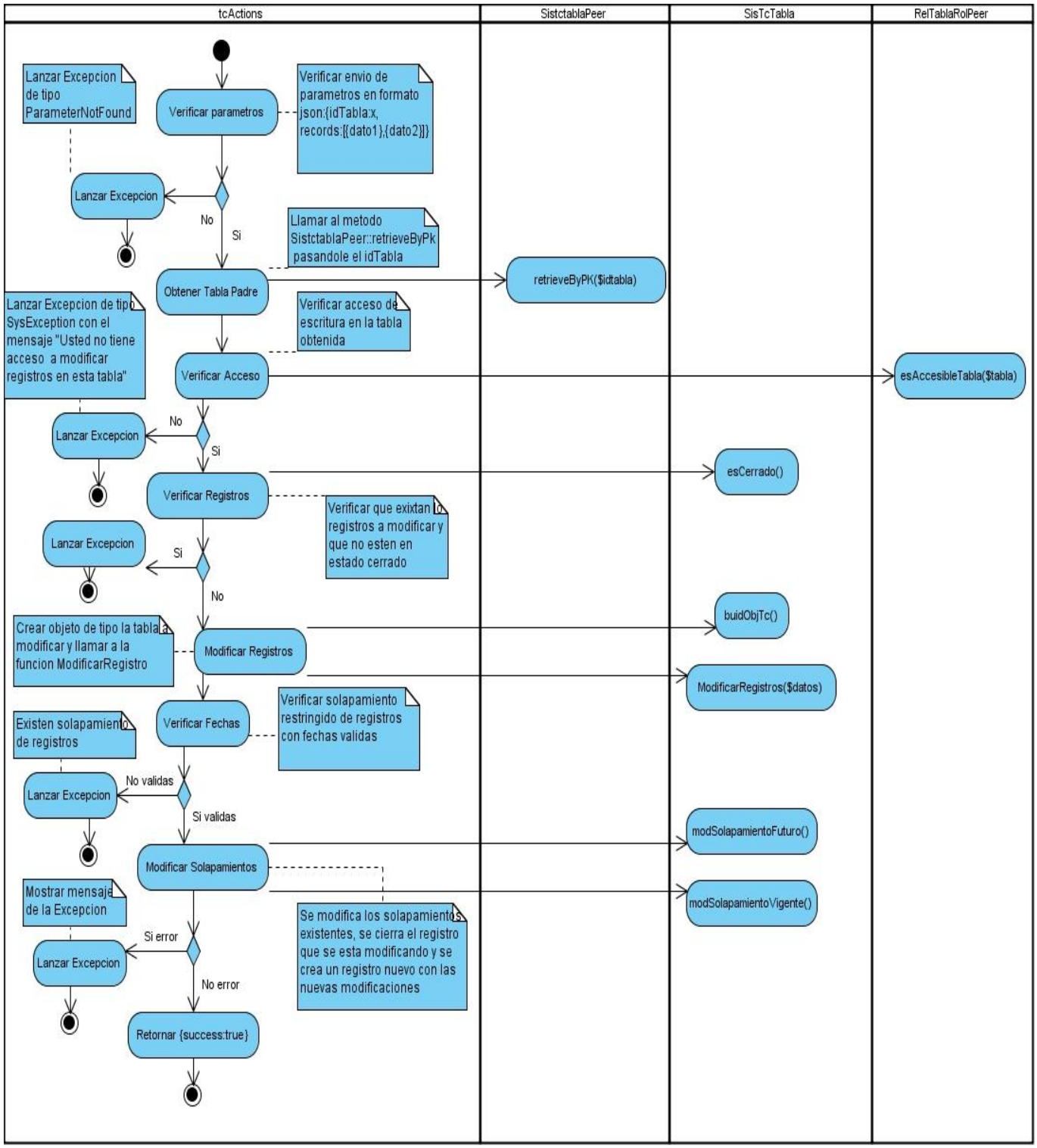
| | |
|---|---|
| Componente | Modificar |
| Método | onClick (Aceptar) |
| Propósito | Esta funcionalidad tiene como objetivo modificar un artículo en vigencia o futuro junto con sus datos en las demás tablas hijas. |
| Descripción |  |

**Diagrama de interacción y actividades orientado a la acción *executeModificarRegistro.***



### 2.9 Escenario Cerrar Registro

El sistema debe de permitir dejar sin vigencia un artículo dentro de las tablas de control para lo cual hay que permitir que modifique la fecha de fin de validez del articulo y debe existir la llave que solicita. El sistema debe permitir modificar la fecha de fin sobre la misma existente y debe estar en un rango permisible mayor o igual que la de inicio, mayor que la actual o vacía. De todas las tablas de control que impliquen relaciones con otras tablas también se podrán eliminar artículos de la misma manera, siempre eliminando también los artículos de las tablas relacionadas.

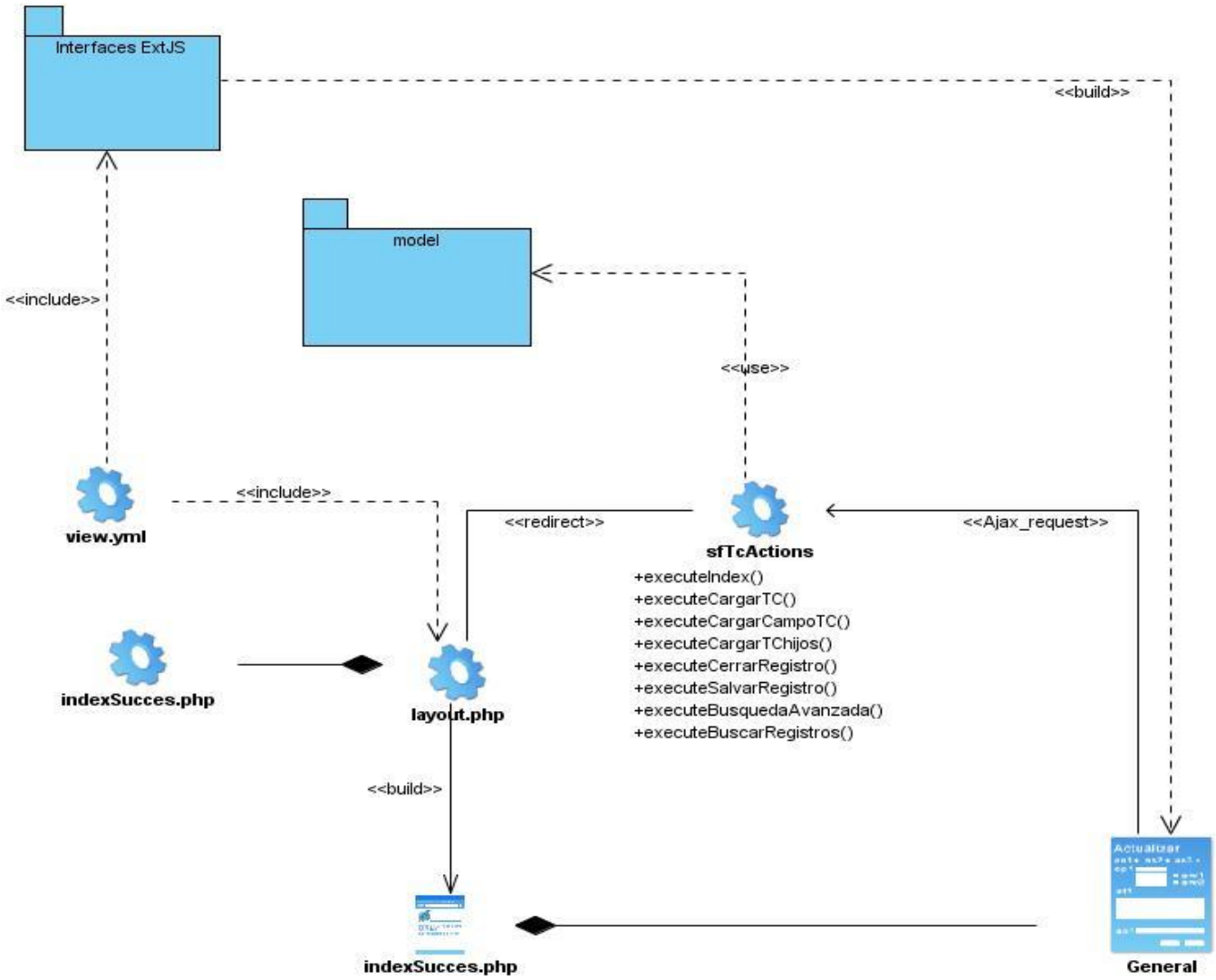



Diagrama de Clases del Diseño (Cerrar Registro)

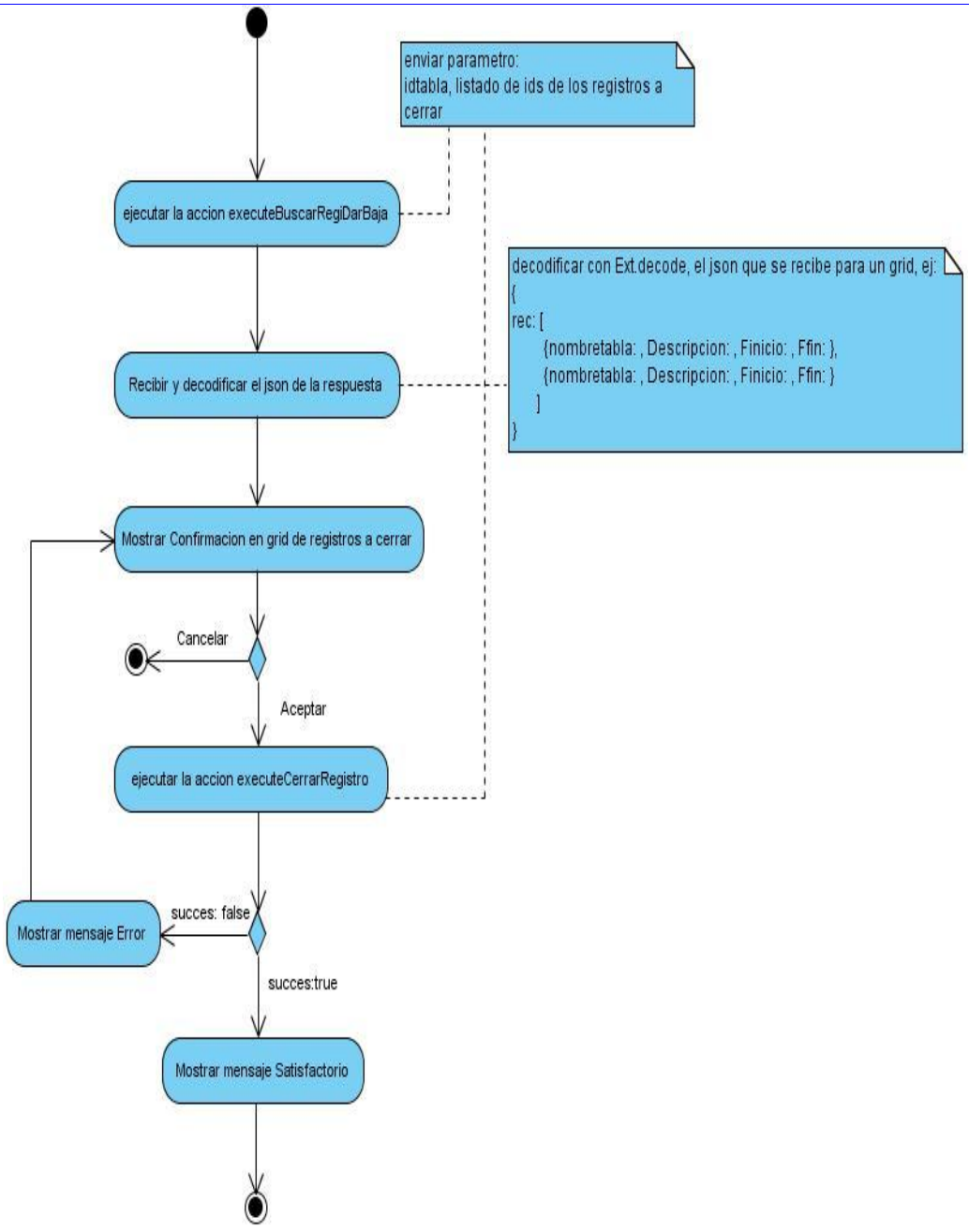
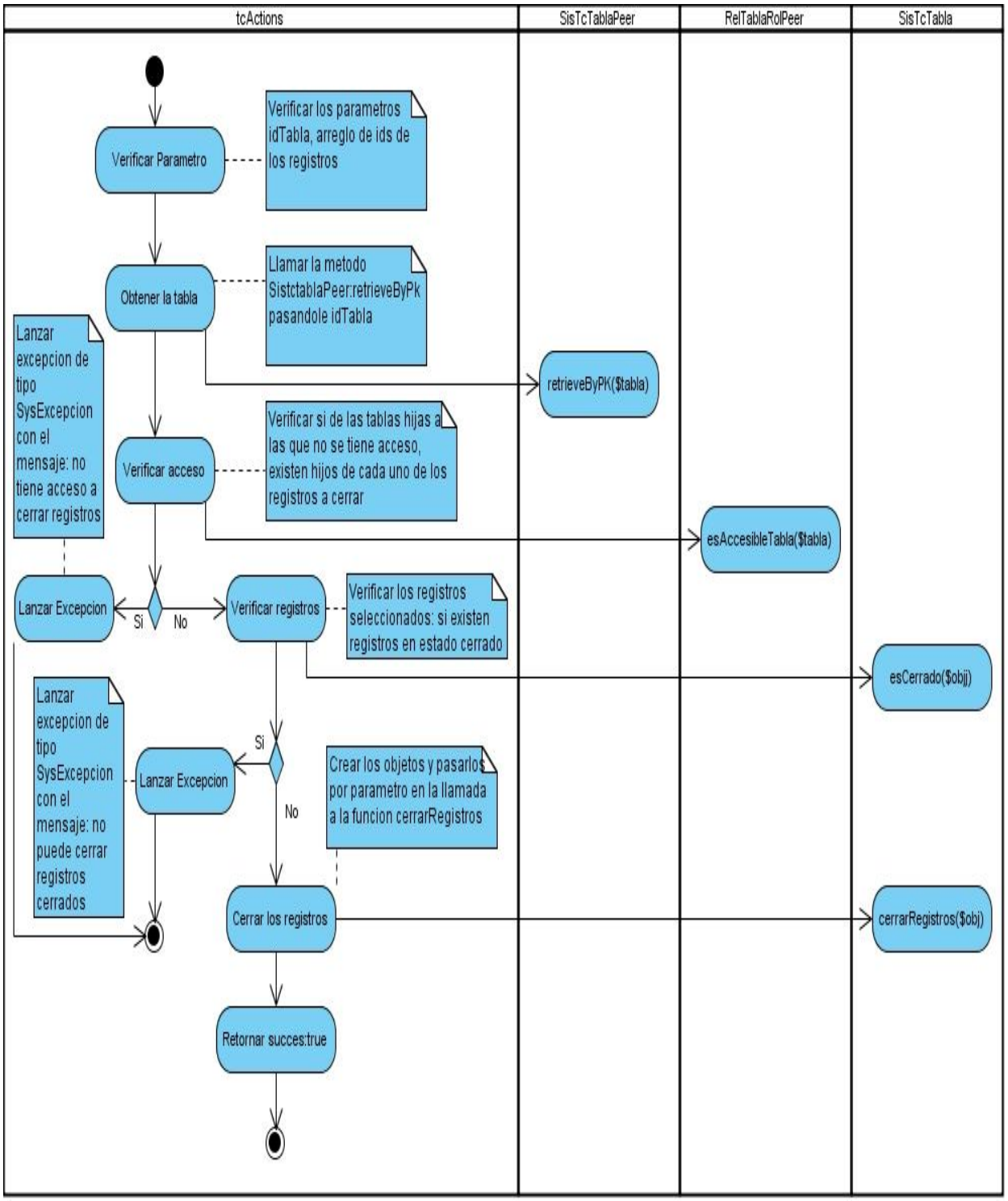
| | |
|---|---|
| Componente | Dar Baja |
| Método | onClick |
| Propósito | Esta funcionalidad tiene como objetivo cerrar los artículos seleccionados de una tabla, así como los registros hijos de estos en otras tablas. |
| Descripción |  |

**Diagrama de interacción y actividades orientado a la acción *executeCerrarRegistro.***



## 2.10 Escenario Búsqueda Avanzada.

Buscar uno o varios registros en la Base de Datos del sistema, permitiendo una búsqueda general o una avanzada.

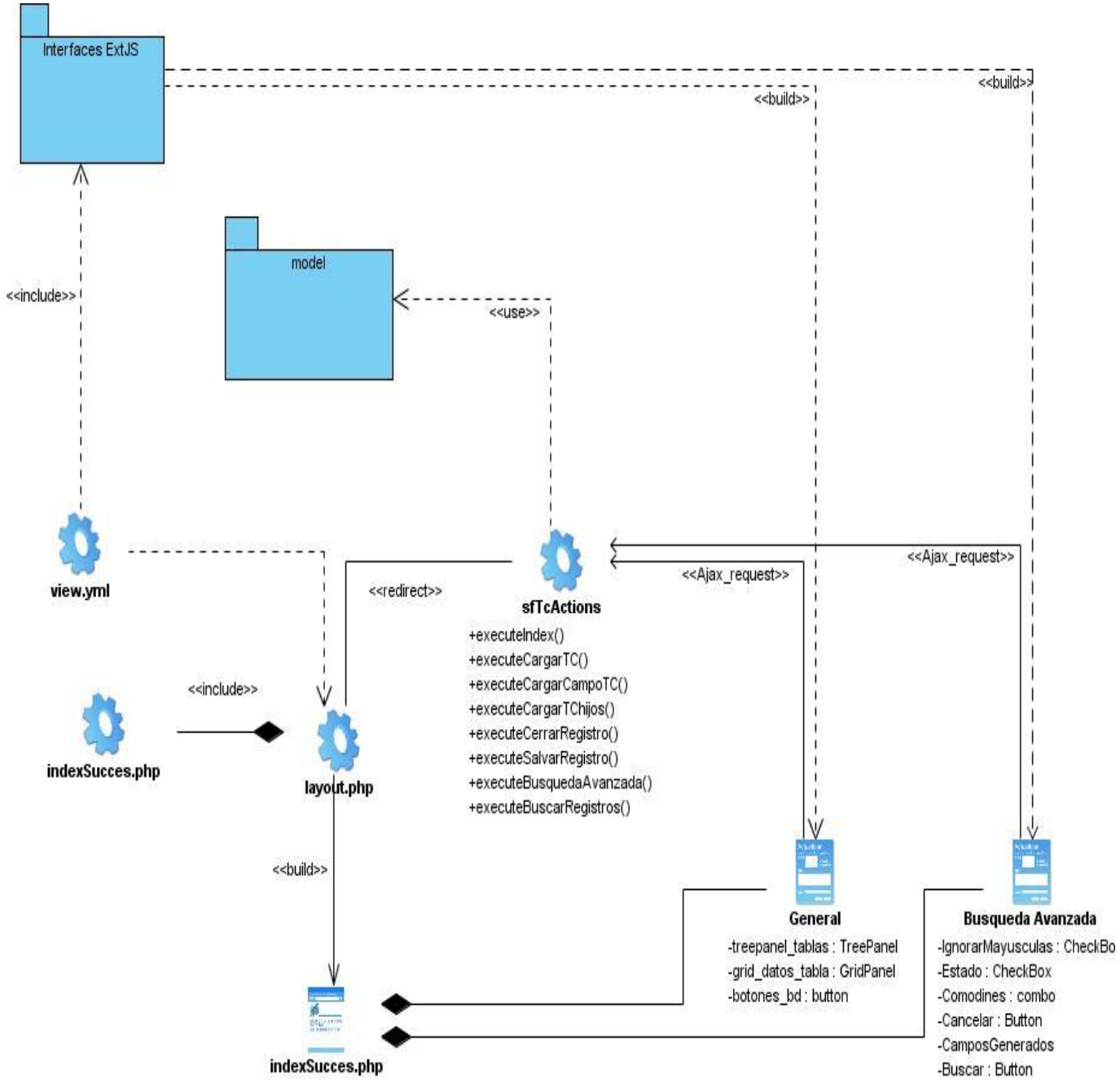
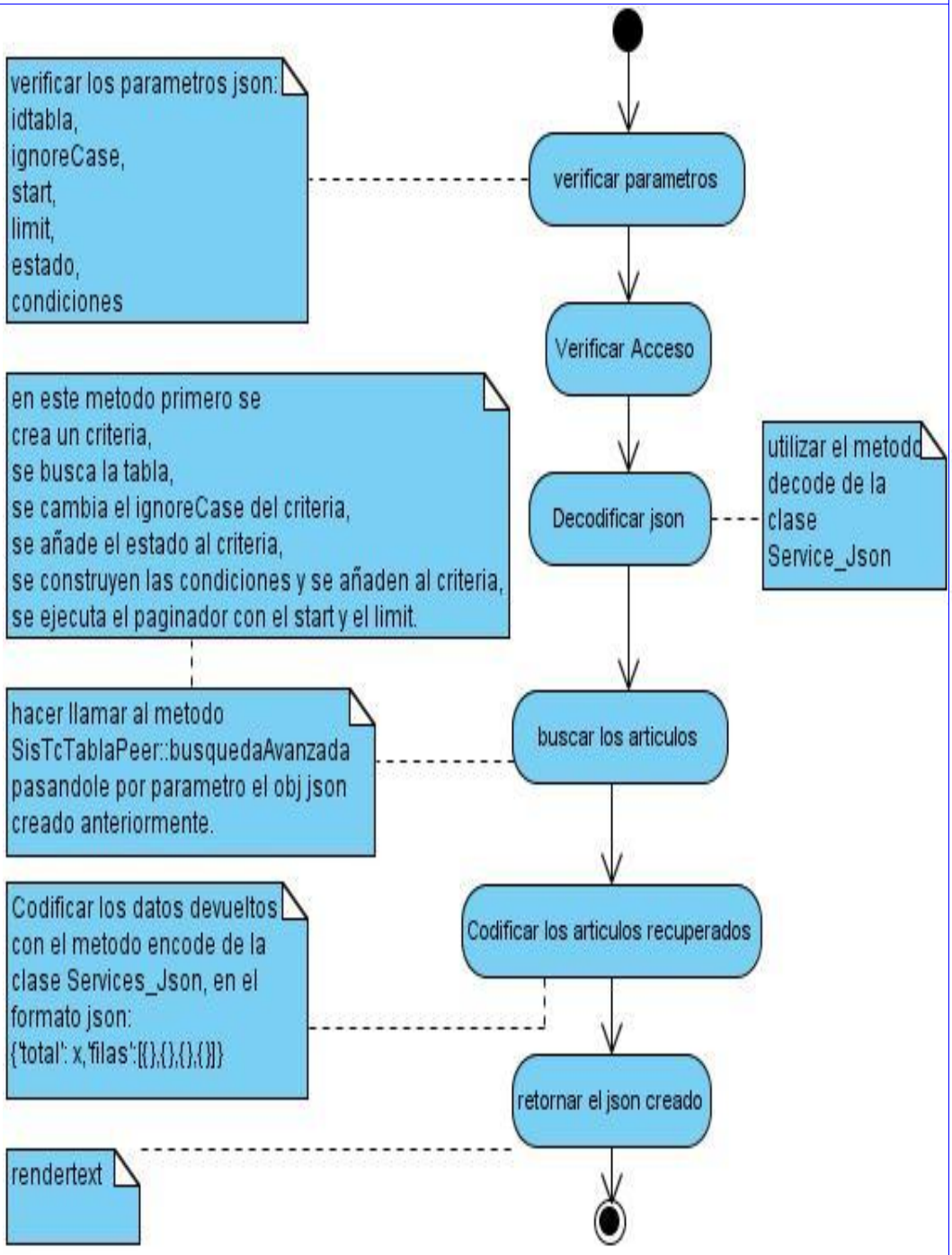



Diagrama de Clases del Diseño (Búsqueda Avanzada)

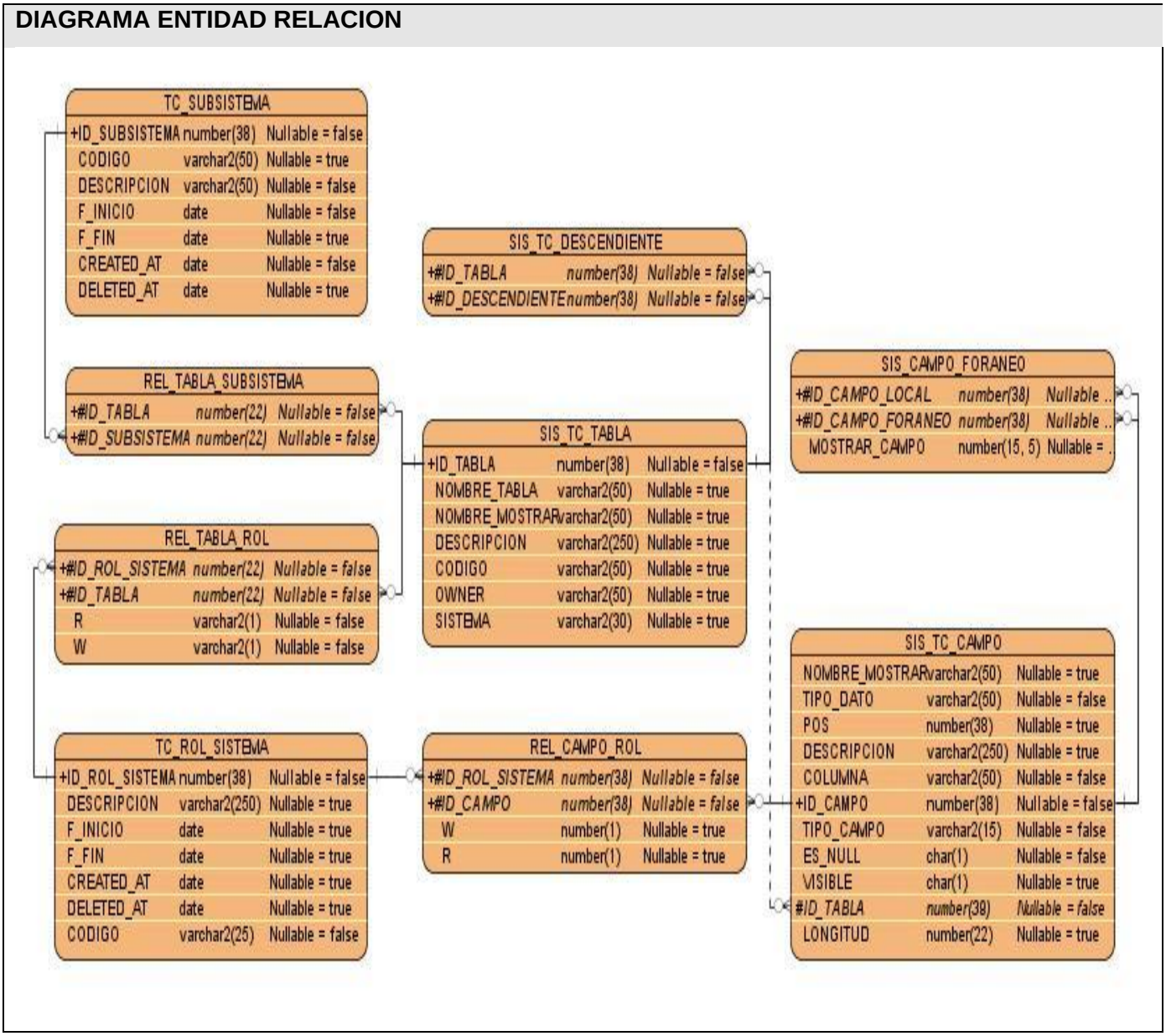
| | |
|---|---|
| Componente | Búsqueda Avanzada |
| Método | executeBusquedaAvanzada |
| Propósito | Esta funcionalidad tiene como objetivo realizar una búsqueda avanzada. |
| Descripción |  |

## 2.11 Diseño de la Base de Datos

- Diagrama Entidad Relación de la BD.

Un diagrama o modelo entidad-relación (a veces denominado por su siglas, E-R "Entity relationship", o, "DER" Diagrama de Entidad Relación) es una herramienta para el modelado de datos de un sistema de información. Estos modelos expresan entidades relevantes para un sistema de información, sus inter-relaciones y propiedades.

**DIAGRAMA ENTIDAD RELACION**



- Descripción de las tablas.

| **Nombre: SIS_TC_TABLA** | | |
|---|---|---|
| **Descripción:** tiene como objetivo de contener la información, manejar las acciones y el comportamiento de las Tablas de Control. | | |
| **Atributo** | **Tipo** | **Descripción** |
| ID_TABLA | number | Identificador de la tabla |
| NOMBRE_TABLA | varchar | Nombre de la tabla |
| NOMBRE_MOSTRAR | varchar | Nombre a mostrar |
| DESCRIPCION | varchar | Descripción de la tabla |
| CODIGO | varchar | Código de la tabla |
| OWNER | varchar | Conexión que utiliza |
| SISTEMA | varchar | Aplicación de acceso |

| **Nombre: SIS_TC_CAMPO** | | |
|---|---|---|
| **Descripción:** tiene como objetivo contener toda la información de los campos de las Tablas de Control. | | |
| **Atributo** | **Tipo** | **Descripción** |
| NOMBRE_MOSTRAR | varchar | Nombre del campo a mostrar |
| TIPO_DATO | varchar | Tipo de dato del campo |
| POS | number | Posición |
| DESCRIPCION | varchar | Descripción del campo |
| COLUMNA | varchar | Columna del campo |
| ID_CAMPO | number | Identificador del campo |
| TIPO_CAMPO | varchar | Tipo de campo |
| ES_NULL | char | Si está vacío o no |
| VISIBLE | char | Si es visible o no |
| ID_TABLA | number | Identificador de la tabla |
| LONGITUD | number | Longitud del campo |

| **Nombre: SIS_TC_DESCENDIENTE** | | |
|---|---|---|
| **Descripción:** tiene como objetivo mantener las relaciones que existen en cada una de las Tablas de Control. | | |
| **Atributo** | **Tipo** | **Descripción** |
| ID_TABLA | number | Identificador de la tabla |
| ID_DESCENDIENTE | number | Identificador de la tabla descendiente |

| **Nombre: SIS_CAMPO_FORANEO** | | |
|---|---|---|
| **Descripción:** tiene como objetivo mantener las relaciones entre los campos foráneos de las Tablas de Control. | | |
| **Atributo** | **Tipo** | **Descripción** |
| ID_CAMPO_LOCAL | number | Identificador del campo local |
| ID_CAMPO_FORANEO | number | Identificador del campo foráneo |
| MOSTRAR_CAMPO | number | Identificador del campo a mostrar |

| **Nombre: REL_CAMPO_ROL** | | |
|---|---|---|
| **Descripción:** tiene como objetivo definir el acceso a los campos de las Tablas de Control. | | |
| **Atributo** | **Tipo** | **Descripción** |
| ID_ROL_SISTEMA | number | Identificador del rol del sistema |
| ID_CAMPO | number | Identificador del campo |
| W | number | Indica si es de escritura |
| R | number | Indica si es de lectura |

| **Nombre: REL_TABLA_ROL** | | |
|---|---|---|
| **Descripción:** tiene como objetivo definir el acceso a las Tablas de Control. | | |
| **Atributo** | **Tipo** | **Descripción** |
| ID_ROL_SISTEMA | number | Identificador del rol del sistema |
| ID_TABLA | number | Identificador de la Tabla |
| R | varchar | Indica si es de lectura |
| W | varchar | Indica si es de escritura |

| **Nombre: TC_ROL_SISTEMA** | | |
|---|---|---|
| **Descripción:** tiene como objetivo describir una estructura para todos los tipos de organismos que liberan documentos de identificación existentes en el GINA. | | |
| **Atributo** | **Tipo** | **Descripción** |
| ID_ROL_SISTEMA | number | Identificador del rol del sistema |
| DESCRIPCION | varchar | Descripción del rol del sistema |
| F_INICIO | date | Fecha de apertura |
| F_FIN | date | Fecha de cierre |
| CREATED_AT | date | Define la fecha de creación de un registro |
| DELETED_AT | date | Define la fecha de eliminación de un registro |
| CODIGO | varchar | Código del rol del sistema |

| **Nombre: REL_TABLA_SUBSISTEMA** | | |
|---|---|---|
| **Descripción:** tiene como objetivo definir las relaciones existentes entre los subsistemas y las tablas. | | |
| **Atributo** | **Tipo** | **Descripción** |
| ID_TABLA | number | Identificador de la Tabla |
| ID_SUBSISTEMA | number | Identificador del Subsistema |

### 2.12 Patrón de diseño fundamental que se aplicó.

Los patrones son principios generales de soluciones que aplican ciertos estilos que ayudan a la creación de software. Es una descripción de un problema y la solución al mismo. Muchos patrones ayudan a asignar responsabilidades a los objetos.

Debido a la utilización del Framework de PHP Symfony para la desarrollar la implementación de la aplicación, nuestra arquitectura estará dada por la arquitectura del mismo, es decir, dado que el framework está basado en un patrón clásico de diseño web conocido como arquitectura MVC, nuestra aplicación tendrá la misma estructura.

#### 2.12.1 Model-View-Controller

El MVC es un patrón arquitectural aportado por SmallTalk y hoy en día muy difundido en uso en aplicaciones de entorno web. La evolución de lo que se conoce como modelo 2 de aplicaciones web (separación de responsabilidades de presentación, negocio y navegación) avanza un poco más en el

reparto de tareas en la aplicación web. Pese a que hay distintos puntos de vista acerca de la forma de aplicar e implementar este patrón, en esencia las ideas principales sobre su estructura y funcionalidad son las mismas. El MVC tiene tres piezas claves que se reparten la responsabilidad de la aplicación:

El modelo (Model)

La lógica de negocio de las aplicaciones web depende casi siempre en su modelo de datos. El componente que se encarga por defecto de gestionar el modelo en Symfony es una capa de tipo ORM (object/relational mapping) realizada mediante el proyecto Propel. En las aplicaciones Symfony, el acceso y la modificación de los datos almacenados en la base de datos se realiza mediante objetos; de esta forma nunca se accede de forma explícita a la base de datos. Este comportamiento permite un alto nivel de abstracción y permite una fácil portabilidad.

La principal ventaja que aporta el ORM es la reutilización, permitiendo llamar a los métodos de un objeto de datos desde varias partes de la aplicación e incluso desde diferentes aplicaciones, también encapsula la lógica de los datos.

La utilización de objetos en vez de registros y de clases en vez de tablas, tiene otra ventaja: permite añadir métodos accesorios en los objetos que no tienen relación directa con una tabla.

La vista (View)

La vista se encarga de producir las páginas que se muestran como resultado de las acciones. La vista en Symfony está compuesta por diversas partes, estando cada una de ellas especialmente preparada para que pueda ser fácilmente modificable por la persona que normalmente trabaja con cada aspecto del diseño de las aplicaciones.

▪ Los diseñadores web normalmente trabajan con las plantillas y con el layout. Estas partes están formadas por código HTML que contiene pequeños trozos de código PHP, que normalmente son llamadas a los diversos helpers disponibles.

▪ Para mejorar la reutilización de código, los programadores suelen extraer trozos de las plantillas y los transforman en componentes y elementos parciales. De esta forma, el layout se modifica para definir zonas en las que se insertan componentes externos. Los diseñadores web también pueden trabajar fácilmente con estos trozos de plantillas.

▪ Los programadores normalmente centran su trabajo relativo a la vista en los archivos de configuración YAML (que permiten establecer opciones para las propiedades de la respuesta y para otros elementos de la interfaz) y en el objeto respuesta. Cuando se trabaja con variables en las plantillas, deben considerarse

los posibles riesgos de seguridad de XSS (cross-site scripting) por lo que es necesario conocer las técnicas de escape de los caracteres introducidos por los usuarios.

Independientemente del tipo de trabajo, existen herramientas y utilidades para simplificar y acelerar el trabajo (normalmente tedioso) de presentar los resultados de las acciones. En este capítulo se detallan todas estas herramientas.

El controlador (Controller)

Responsable del flujo de control, la navegabilidad y el estado de la entrada del usuario. Habitualmente implementado por medio de un servlet (en proyectos java, lógicamente) es el corazón del funcionamiento del patrón. Es responsable de: 1. Interceptar y recoger las peticiones http del cliente. Así, el cliente no invocará directamente ninguna página jsp o html, sino que será redireccionado adecuadamente por el controlador. 2. Traducir la petición en una operación de negocio específica. 3. Invocar la operación o bien delegar en un manejador. 4. Determinar la siguiente vista a mostrarle al cliente 5. Retornar el control al cliente. 6. El hecho de que todas las peticiones http pasen por el controlador facilita el mantenimiento de la aplicación, sobre todo en lo referente al control de la navegabilidad, sustitución de páginas, etc.

### 2.12.2 Adapter

**Problema**: ¿Cómo tener una única interfaz de acceso a múltiples bases de datos?

**Propósito**: Convertir la interfaz de una clase para que se adapte a lo que el cliente que la usa necesita, permitiendo así que trabajen juntas clases cuyas interfaces son incompatibles.

**Solución:** Symfony da la posibilidad de cambiar a otro sistema de base de datos completamente diferente a mitad de desarrollo, sólo basta con configurar un archivo YAML. La capa de abstracción utilizada encapsula toda la lógica de los datos. El resto de la aplicación no tiene que preocuparse por las consultas SQL y el código SQL que se encarga del acceso a la base de datos.

### 2.12.3 Singleton

**Problema**: Obtener un punto de acceso global a una clase.

**Propósito**: Asegurar que sólo exista una instancia de una clase especifica en un sistema a desarrollar.

**Solución**: El controlador frontal proporciona un punto de acceso global a la aplicación. En una acción, el método *getContext()* devuelve el mismo *singleton*. Se trata de un objeto muy útil que guarda una referencia a todos los objetos del núcleo de Symfony relacionados con una petición dada, y ofrece un método de acceso para cada uno de ellos. Algunos ejemplos utilizados en la aplicación.

*Propel::getConnection('tc');*

*SysComponentsLocator::getInstance();*

### 2.12.4 Bajo acoplamiento

**Problema**: ¿Cómo dar soporte a una dependencia escasa y a un aumento de la reutilización?

**Propósito**: Aumentar la reutilización y eliminar las dependencias entre las clases para propiciar un fácil mantenimiento y entendimiento.

**Solución**: Symfony asigna a cada clase una responsabilidad para mantener pocas dependencias entre las mismas. La aplicación desarrollada mantiene el sistema de clases generadas por symfony y sus dependencias.

## 2.13 Conclusiones del capítulo:

En el presente capítulo se definió la estructura del Sistema con la presentación de los diagramas del Diseño, Secuencia, Entidad Relación y breve descripción de los mismos. Se pretende en el mismo, dejar listo el camino para el paso a los puntos en los que se debe enfocar en el siguiente capítulo. EL sistema propuesto marcará un paso de avance en el proyecto en general ya que se podrán gestionar todos los nomencladores y codificadores de la aduana a través de una interfaz única manteniendo el historial de los datos.

## CAPÍTULO 3: IMPLEMENTACIÓN Y VALIDACIÓN DE LA SOLUCIÓN

### 3.1 Introducción.

En el presente capítulo serán abordados todos los temas referentes a la implementación y validación de la solución. En el mismo se muestra el diagrama de componentes, donde se establece las relaciones entre los componentes y la estructura de los subsistemas que integran el sistema desarrollado, así como el diagrama de despliegue. También se abordara las pruebas pilotos realizadas a la aplicación bajo entornos reales.

### 3.2 Diagrama de Componentes.

Los diagramas de componentes representan cómo un sistema de software es dividido en componentes, mostrando las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables o paquetes. Se utilizan para modelar la vista estática y dinámica de un sistema, muestran la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, habitualmente se realizan por partes, donde cada diagrama describe un apartado del sistema.

Los diagramas pueden ser usados para modelar y documentar cualquier arquitectura de sistema, los elementos de modelado dentro de un diagrama de componentes son componentes y paquetes. Los diagramas de componentes muestran los componentes software que constituyen una parte reusable, sus interfaces, y sus interrelaciones. Un paquete en un diagrama de componentes representa una división física del sistema.

Seguidamente se muestra en la figura el diagrama de componentes del Módulo Tabla de Control (TC), que se resume en TC brinda servicios a los demás módulos del sistema (DNC, API, LCF) utilizando las librerías de Symfony.

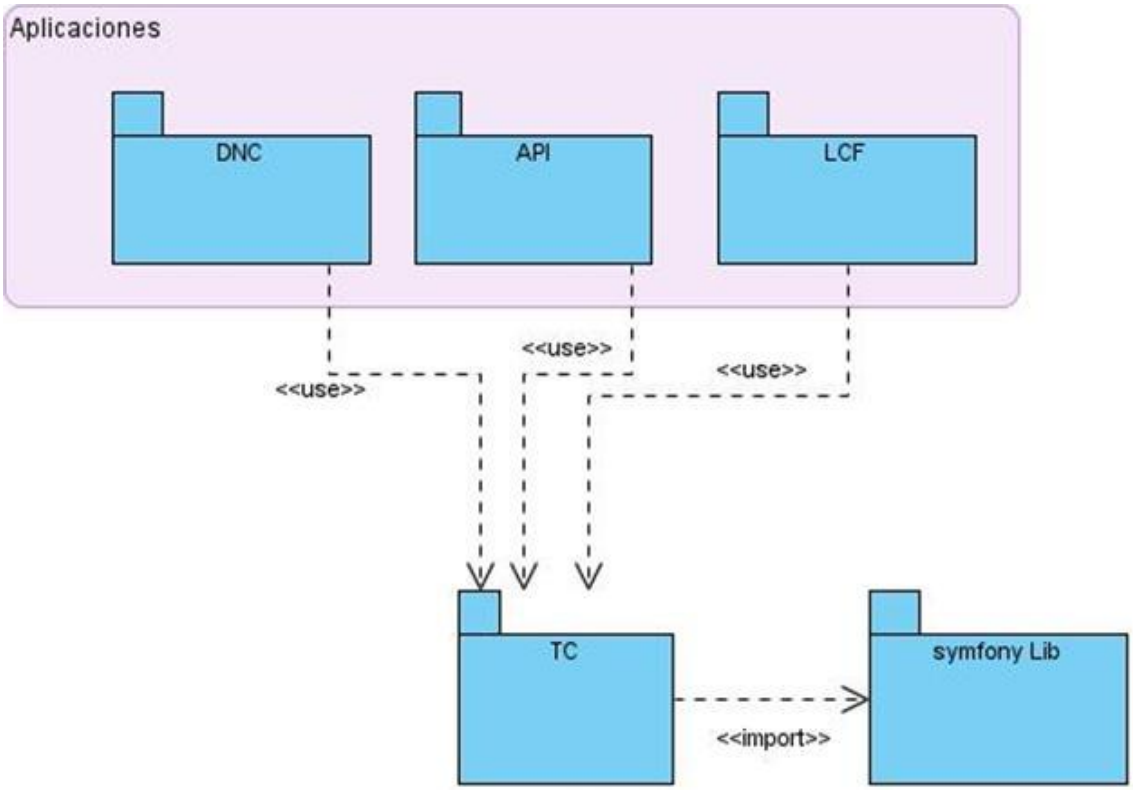



*Figura 2. Diagrama de Componentes*

**3.3 Diagrama de Despliegue.**

El Diagrama de Despliegue se utiliza para modelar la arquitectura en tiempo de ejecución de un sistema. Muestra la configuración de los elementos de hardware (nodos) utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Se representa mediante un grafo de nodos unidos por conexiones de comunicación, los nodos son elementos físicos de ejecución que representa un recurso computacional, que generalmente tiene por lo menos memoria y a veces también capacidad de proceso. Los nodos sirven para modelar la topología del hardware sobre el que se ejecuta el sistema. Un nodo representa, generalmente, un procesador o un dispositivo sobre el que se pueden desplegar los componentes y debe tener un nombre asignado que lo distinga del resto de nodos. En el siguiente diagrama se muestra un nodo que representa al Cliente Web donde el usuario para acceder a la aplicación, una PC Servidor Web que es donde estará situada la aplicación, una PC Servidor de Base de Datos en donde se encontrará la base de datos del sistema.

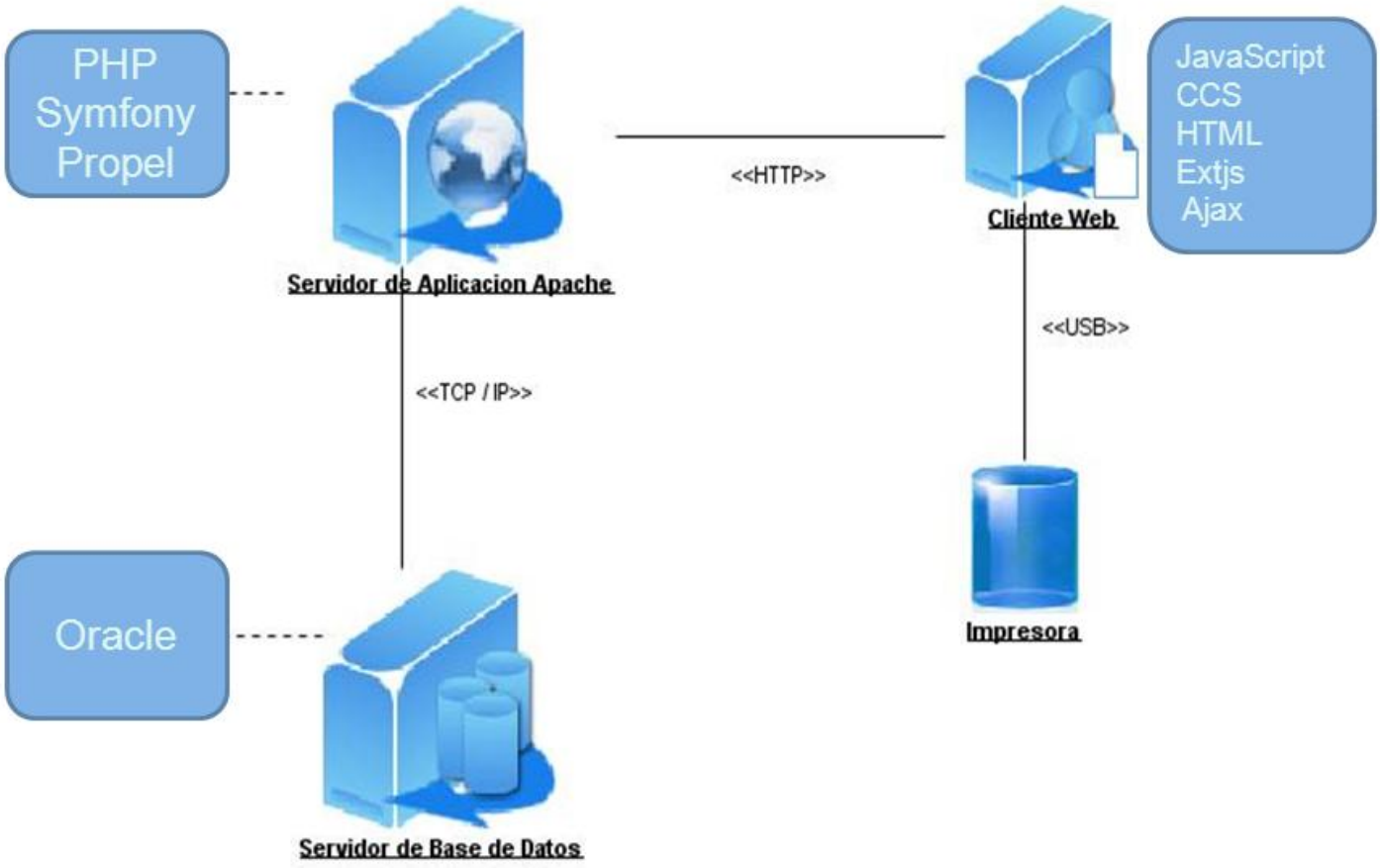



*Figura 3. Diagrama de Despliegue*

### 3.4 Resultado esperado: Módulo Tabla de Control

Como resultado del proceso de implementación se obtiene una versión funcional del Módulo Tabla de Control como se muestra en la Figura 4. La aplicación como característica principal tiene que cumple con los requerimientos críticos, en otras palabras, las funcionalidades básicas que satisfacen los requerimientos del cliente.

Una de las principales características que cumple la solución es que mediante una única interfaz gestiona los codificadores existes en el GINA de forma dinámica, permitiendo una mejor organización de los datos, coherencia y consistencias de la información, reutilización del código, evitando redundancia en la programación de la aplicación, logrando una mayor seguridad en el sistema.

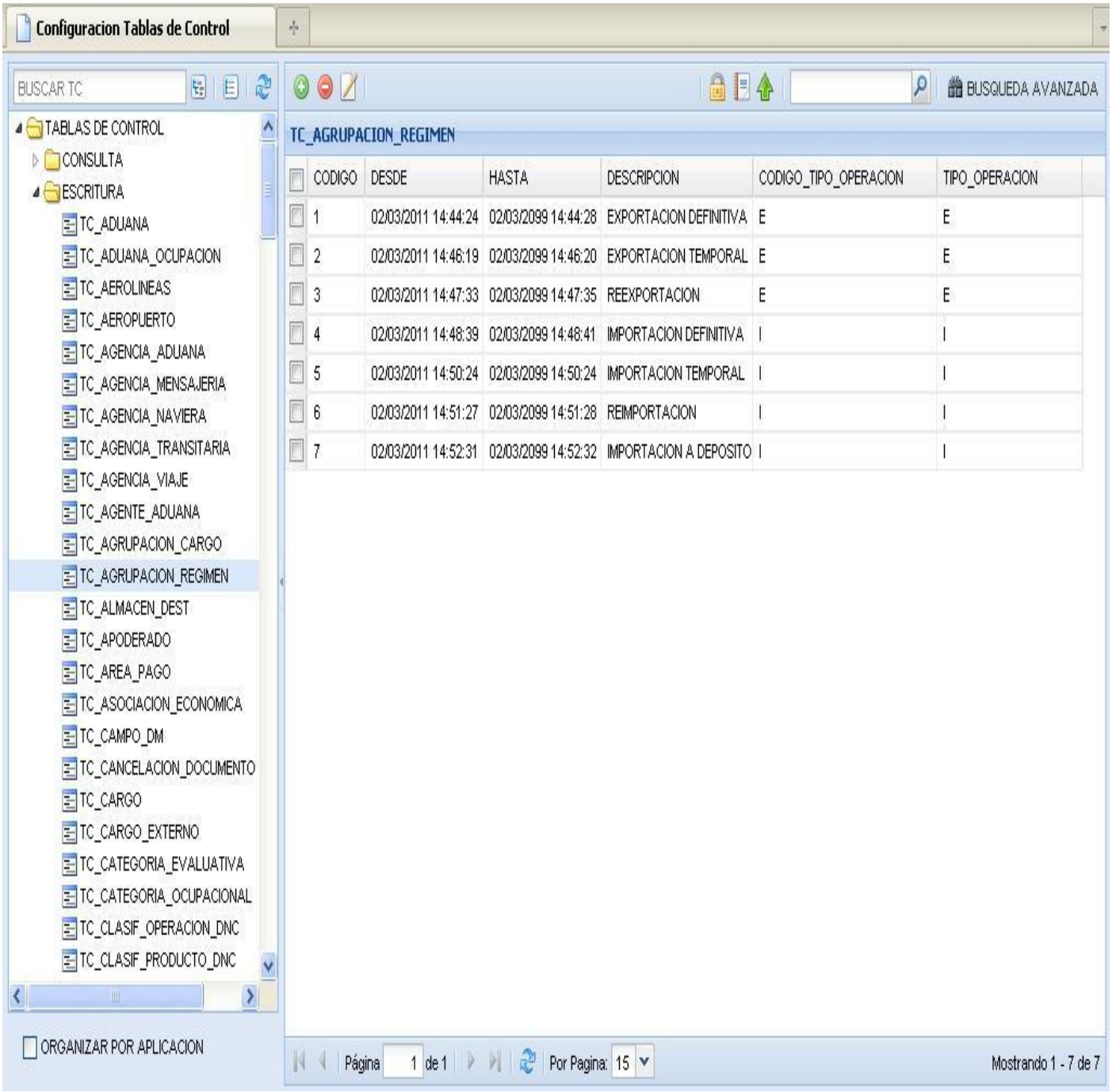



*Figura 4. Pantalla principal.*

### 3.5 Estándares de codificación

Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento. Además, si se aplica de forma continuada un estándar de codificación bien definido, se utilizan técnicas de programación apropiadas, y, posteriormente, se efectúan revisiones del código de rutinas, caben muchas posibilidades de que un proyecto de software se convierta en un sistema de software fácil de comprender y de mantener.

Para la implementación del módulo TC fue necesario regirse por los estándares de codificación utilizados en el proyecto GINA. Algunos de los más importantes se muestran a continuación. (19)

Acciones:

- Todos los nombres de acciones deben estar en la nomenclatura "*CamelCase"* comenzando por la palabra *execute*.

Nombres de las clases:

- Los nombres de las clases deben estar expresados en notación *"UpperCamelCase"*.
- Deben expresar con claridad cuál es el alcance y la responsabilidad de la clase.

Nombres de los archivos de las clases:

- Los nombres de los archivos de las clases deben estar compuestos por el nombre de la clase seguido de un punto y la palabra "*class*" y la extensión del archivo "*.php*".

Nombres de las funciones:

- Los nombres de las funciones deben dejar reflejado claramente cuál es la acción que realiza el mismo.
- Se debe apoyar en la utilización de sufijos que ayuden a identificar el resultado final de la ejecución de un método.
- Se debe apoyar en los prefijos para expresar la acción que realiza sobre un elemento determinado.

### 3.6 Clases de la aplicación.

Existen algunas clases que son esenciales para entender el funcionamiento del sistema. A continuación se presentarán algunas de ellas y fragmentos de código utilizados en las mismas.

Clase tcActions:

Esta clase funciona como controladora del sistema, respondiendo a los eventos generados en la vista. Cuenta con acciones como:

- *executeCargarTC*: Encargada de cargar las tablas de control a las que el módulo tiene acceso.
- *executeCargarCampoTc* Encargada de cargar los campos de la TC seleccionada.
- *executeCargarTCHijos*: Encargada de retornar las tablas hijas de la tabla seleccionada para dar alta o modificar.
- *executeVigentesFuturos*: Encargada de retornar los datos de una tabla por el estado de los artículos de dicha tabla.
- *executeExisteRegistro*: Encargada de verificar si cuando se va a dar de alta se introduce un código vigente o futuro.

Para más detalle ver Anexo 1 la sección Capa de Acceso a Datos.

Clase TcAduanaForm:

Es una de las clases generadas por symfony usando el ORM Propel que permite validar la estructura de los datos al insertar o modificar una tabla en este caso TC_ADUANA.

```
$this->setValidators(array(
'tipo' => new sfValidatorString(array('max_length' => 1, 'required' => false)),
'ocupacion' => new sfValidatorNumber(array('required' => false)),
'idAduana'=>new sfValidatorPropelChoice(array('model'=>'TcAduana','column'=>'ID_ADUANA', 'required' => false)),
'idPadre'=>new sfValidatorPropelChoice(array('model'=>'TcAduana', 'column' => 'ID_ADUANA', 'required' => false)),
'idEstructura'=>new sfValidatorPropelChoice(array('model'=>'RhEstructura','column'=>'ID_ESTRUCTURA','required'=>true)
'nivel' => new sfValidatorNumber(array('required' => false)),
'codigo' => new sfValidatorString(array('max_length' => 30, 'required' => false)),
'descripcion' => new sfValidatorString(array('max_length' => 50, 'required' => false)),
'fInicio'=> new sfValidatorDateTc(array('date_format' => sfValidatorDateTc::EXP_REG_DATE_FORMAT, 'with_time' => true,
'fFin' => new sfValidatorDateTc(array('date_format' => sfValidatorDateTc::EXP_REG_DATE_FORMAT, 'with_time' => true,
'deletedAt'=>new sfValidatorDateTc(array('date_format'=> sfValidatorDateTc::EXP_REG_DATE_FORMAT, 'with_time' => true,
'codigoPadre' => new sfValidatorString(array('max_length' => 30, 'required' => false)),
));
unset($this['createdAt']);
$this->widgetSchema->setNameFormat('[%s]');
$this->disableCSRFProtection();
```

*Figura 5. Ejemplo de uso de la clase TcAduanaForm.*

Clase main.js:

Es la clase que permite levantar la pantalla principal ver figura 4. A partir de las acciones de los usuarios se crean los demás eventos y componentes necesarios para la realización de los flujos de los requisitos del software.

Clase ConfigTc.js:

En esta clase se encuentra definidos las principales funcionalidades de TC tales como inserción, modificación y eliminación de un registro. También está presente la creación dinámica de la interfaz.

El siguiente fragmento de código javascript figura 6, muestra parte de la creación dinámica de la interfaz, que mediante la petición Ajax request llama a la función cargarCamposTC pasándole el id de la tabla, función que devuelve en formato json los campos asociados a la tabla especificada, luego se crean las columnas en el visual en dependencia de los campos de dicha tabla; de esta forma se logra así la gestión de las tablas mediante la interfaz única y dinámica.

```
changeView: function(node){
    this.node=node;
    Ext.Ajax.request({
        url:'../'+node.attributes.aplicacion+'/tc/cargarCampoTC',
        params:{ id: node.id },
        success: function(data){
            var resp=Ext.decode(data.responseText);
            if(resp.success==false){
                Ext.Msg.show({
                    title: 'CARGAR REGISTROS',
                    msg: resp.message,
                    buttons: Ext.Msg.OK,
                    icon: Ext.MessageBox.ERROR,
                    width: 350
                });}
            else { this.columnas = resp.columns;
                   this.updateGrid(node);}}});
},
```

*Figura 6.- Fragmento de código.*

**3.7 Tratamiento de excepciones**

Cada una de las funciones estará protegida por un bloque try{…}cath() {..};. Al ocurrir alguna falla en el flujo normal de los eventos se crea y se dispara una excepción con un mensaje que explica la naturaleza del error. Dicha excepción es capturada en la clase controladora y se conforma el json con la forma:

- *{success: false, error: "Mensaje de la excepción" }*

Dicho json se envía hacia la capa de la vista y se muestra una ventana de error con el mensaje que contiene la llave "*error*". En caso de que las instrucciones encapsuladas en el bloque donde se lanzó la excepción impliquen cambios en la base de datos se revierten dichos cambios.

```
public function executeCargarTChijos() {

    try {
        $idtabla = $this->getRequestParameter('node');
        $tabla = SisTcTablaPeer::retrieveByPK($idtabla);

        if (RelTablaRolPeer::esAccesibleTabla($tabla, 1, 1) == true) {

            $hijos = $tabla->getHijos();
            $tables = array();
            foreach ($hijos as $hijo) {
                if (RelTablaRolPeer::esAccesibleTabla($hijo, 1, 1) == true) {
                    $tables[] = $hijo;
                }
            }
            $json_tc = SystemToolkit::encodeArrayObjToJson($tables, "filas");
        }

        $json_tc = str_replace("{'filas': ", '', $json_tc);
        $json_tc = substr($json_tc, 0, count($json_tc) - 2);
        return $this->renderText($json_tc);
    } catch (PropelException $exc) {
        return $this->renderText('{"success": false, "message": "' .$exc->getCause()->getMessage(). '"}');
    } catch (Exception $exc) {
        return $this->renderText('{"success": false, "message": "' .$exc->getCause()->getMessage(). '"}');
    }
}
```

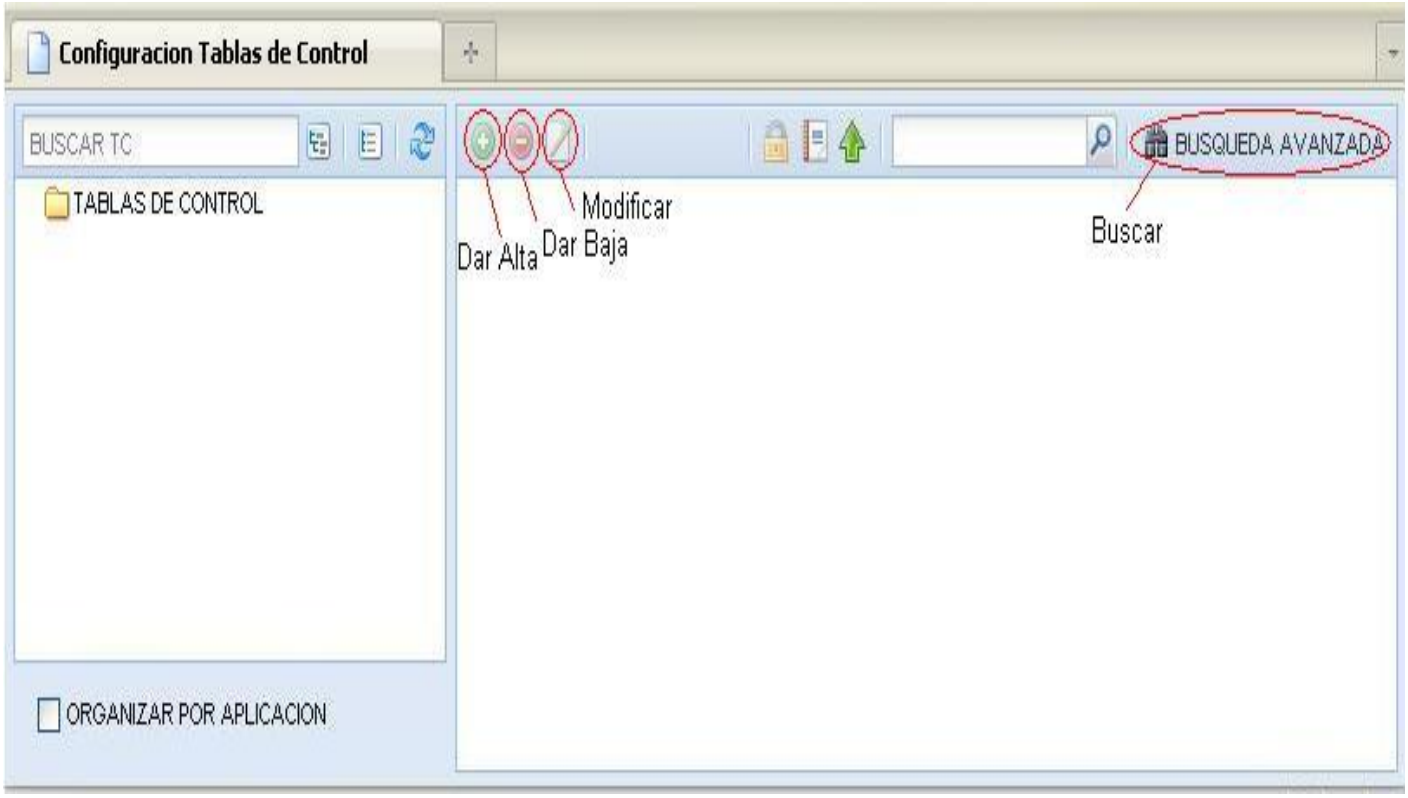
*Figure 7. Ejemplo de manejo de excepciones*

**3.8 Características esenciales**

Como se mencionó anteriormente, al ser una versión inicial contempla sólo algunos elementos. A manera general se integra completamente con el sistema general GINA y abarca hasta el momento los siguientes procesos y algunos más:

- Dar de alta a un registro.
- Modificar un registro con sus relaciones.
- Dar baja a un registro con sus relaciones.
- Buscar registro.

A continuación se muestran la pantalla principal para visualizar de mejor forma lo descrito anteriormente:




**3.9 Aporte práctico**

El Sistema, en su fase inicial, presenta las funcionalidades básicas requeridas por el cliente, mejorando en gran medida el proceso de la manera en que se desarrollaba en los Sistemas ya estudiados anteriormente. La gestión de nomencladores juega un papel importante y es un marcado avance en la automatización general de la gestión de los procesos aduanales. El módulo en cuestión permite disminuir los tiempos de respuesta, optimiza el proceso de gestión de las tablas garantizando el histórico de los datos e incrementa su seguridad.

**3.10 Validación de la solución implementada.**

Confirmación por inspección y provisión de evidencia objetiva de que los requerimientos particulares para un uso específico son alcanzados. En diseño y desarrollo, la validación está relacionada con el proceso de reexaminación de un producto para determinar la conformidad con las necesidades del usuario.

**3.11 Pruebas unitarias y funcionales realizadas.**

Las pruebas unitarias realizadas a TC aseguran que un único componente produzca una salida correcta para una determinada entrada, validando de esta forma que cada función trabaje correctamente para cada

caso particular. Cada test automatizado se encarga de un único caso a la vez, lo que significa que un único método puede necesitar varias pruebas unitarias. Ver Anexo 2 Casos de Pruebas.

Por su parte, las pruebas funcionales no sólo evalúan la transformación de una entrada en una salida, sino que validan procesos completos, motivo por el cual requieren de un escenario o entorno de despliegue específico. La realización de pruebas piloto enmarcadas en un entorno real de trabajo, constituyen un escenario de evaluación experimental muy eficaz para la validación de TC. En el siguiente epígrafe se ofrece un resumen detallado acerca de este tipo de validación realizada en la AGR.

**3.12 Pruebas pilotos realizadas en un entorno real de despliegue.**

El módulo Tablas de Control se encuentra actualmente desplegado en las estaciones de trabajo de la AGR, bajo un escenario similar al mostrado en la Figura 3, por lo que TC está siendo actualmente utilizado por las diferentes aplicaciones que requieren de sus servicios. Este tipo de pruebas realizadas en entornos reales de ejecución, han permitido explotar al máximo las potencialidades de TC desarrollado a partir de la propuesta planteada, de forma tal que las inconformidades presentadas han sido resueltas para el peor de los casos esperados, ver Anexo 3. Es importante destacar además, que la solución implementada cumple de forma óptima con las especificaciones de rendimiento requeridas por la dinámica actual de trabajo que exigen las aplicaciones del GINA.

CONCLUSIONES

Como resultado de la investigación realizada en el presente trabajo de diploma se arribaron a las siguientes conclusiones:

- Se realizó un análisis de los sistemas descubriendo deficiencias en los mismos.
- Se implementaron los requisitos funcionales planteados por los analistas.
- Se desplegó el módulo TC en la Aduana cumpliendo con las necesidades del cliente.

**RECOMENDACIONES**

Hechas las conclusiones del trabajo, se recomienda:

- Integrar Sistema de TC a la Administración del GINA.
- Crear pantalla de Configuración de TC que permita la configuración de cada uno de los campos de la tabla, así como la asignación de los permisos de los diferentes roles.
- Permitir desde el sistema tablas de control, la creación de Tablas en la Base de Datos.

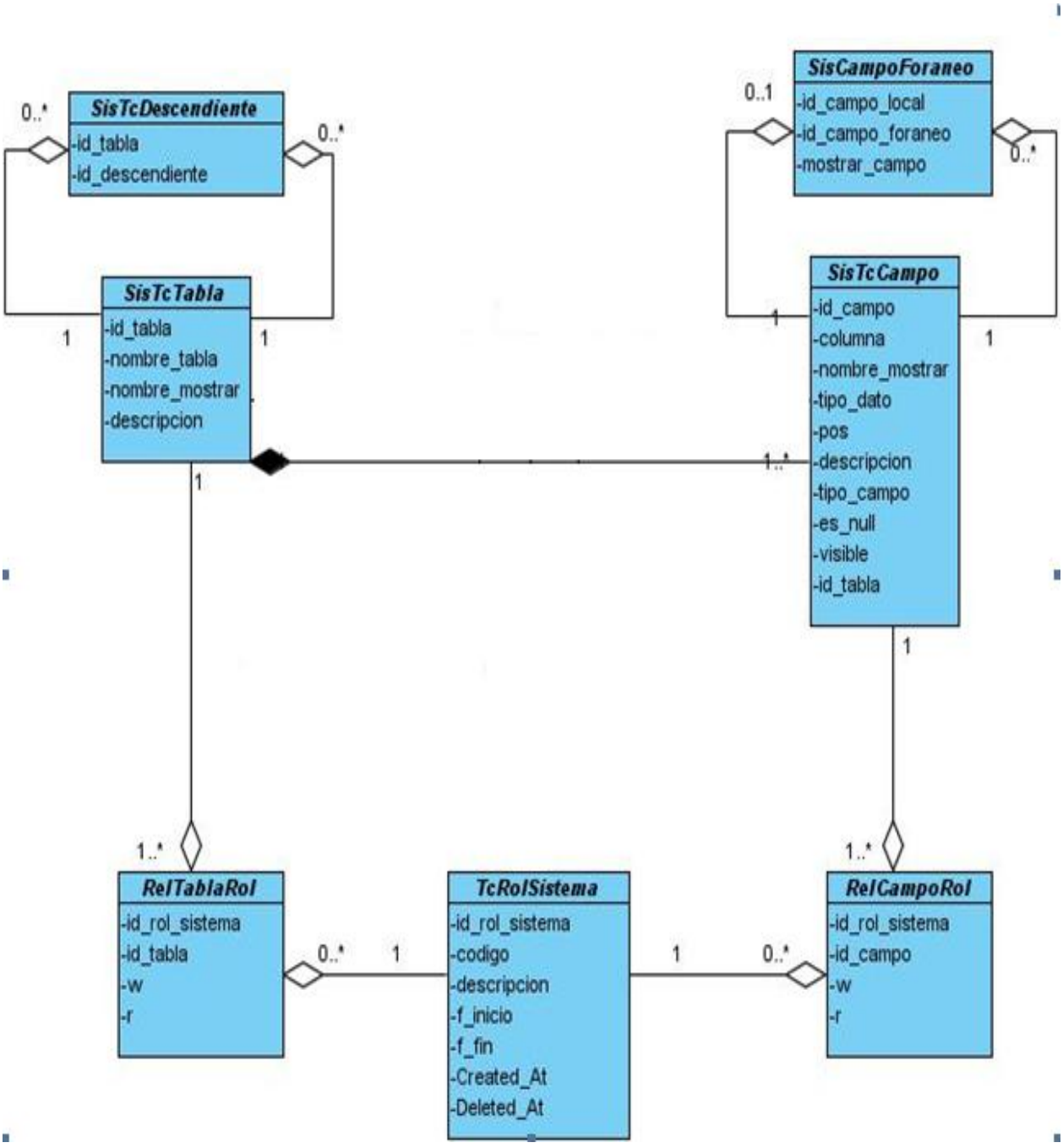
BIBLIOGRAFÍA


1. Tamayo, Islandy Antonio Guevara. *Análisis y Diseño del Modulo Tablas de Control del SUA.*
2. Pérez, Pedro Ramón Pupo. *ADUANA GENERAL DE LA REPÚBLICA, RESOLUCIÓN No. 807-2009.*
3. Muñoz, Javier Llácer, Burgos, Carlos Martínez and Gil, Sergio Catalá. *Informe de evaluación de ERP*
4. AGR. Glosario de términos Aduaneros. [Cited; Available from: http://www.aduana.co.cu/glosa1.htm.
5. http://www.openbravo.com. [Online].
6. http://www.sap.com/platform/netweaver/index.epx. *http://www.sap.com.* [Online] SAP.
7. SAP. *CRM con SAP Business.*
8. http://www.sidunea.aduana.gob.bo/capacitacion/ManualesUsuario/man_mod_sdi.pdf
9. http://www.unctad.org/sp/docs/TN21_Sidunea.pdf
10. http://www.asycuda.org/pdf%20docs/sffunct.pdf
11. *S/11658737581Aduana_Nacional_de_Bolivia.pdf.* [Online]
12. Kioskea .Net. [Online] [Cited: Enero 23, 2008.] http://es.kioskea.net/contents/css/cssintro.php3.
13. Cobo Rodríguez, José Antonio. *Línea Base Arquitectónica para el Polo Sistemas Tributarios y de Aduanas.*
14. NetBeans. *http://netbeans.org/index_es.html.*
15. Hernando, Roberto Velasco. [Online] [Cited: Frebero 10, 2010.] http://www.rhernando.net/modules/tutorials/doc/bd/oracle.html.
16. http://www.sugarforge.org/. *http://www.sugarforge.org/.* [Online].
17. www.asycuda.org. *www.asycuda.org.* [Online]
18. Aduana Bolivia. http://www.aduana.gov.bo. *http://www.aduana.gov.bo.* [Online]
19. Dpto Aduana. CEIGE. CIG-ADU-N-GS_Estándares de codificación.

**ANEXOS**

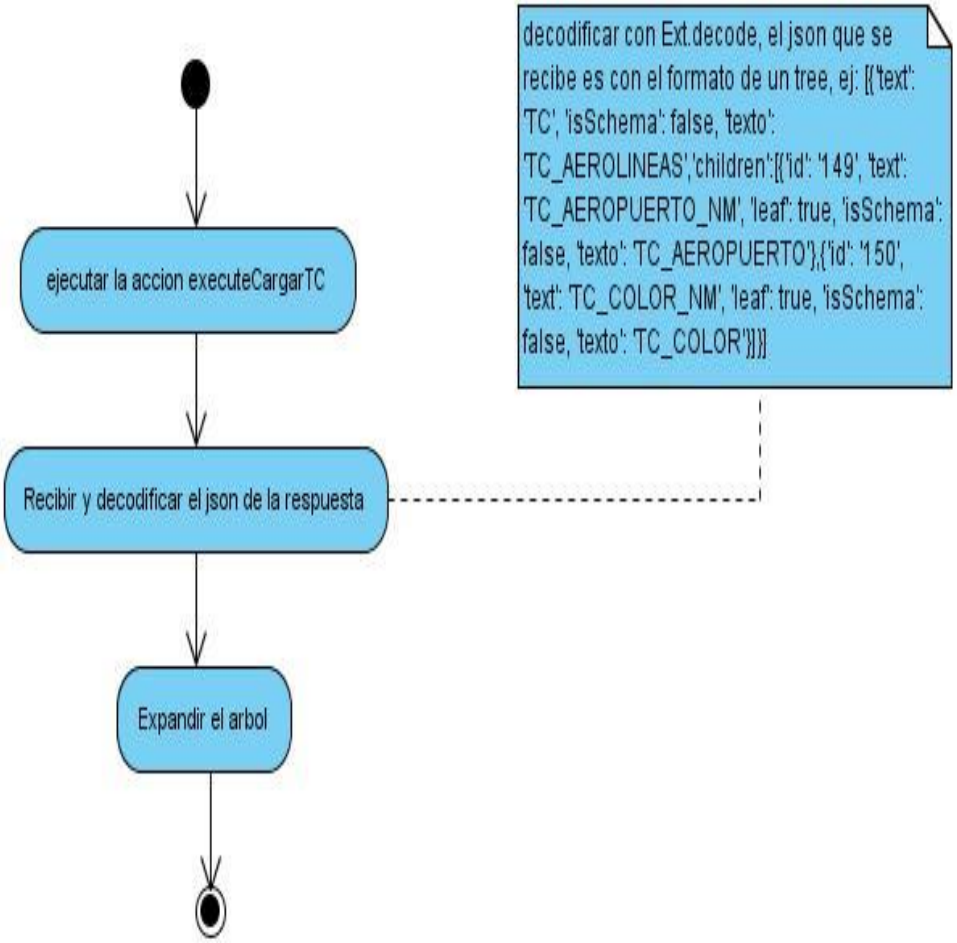
**Anexo 1**

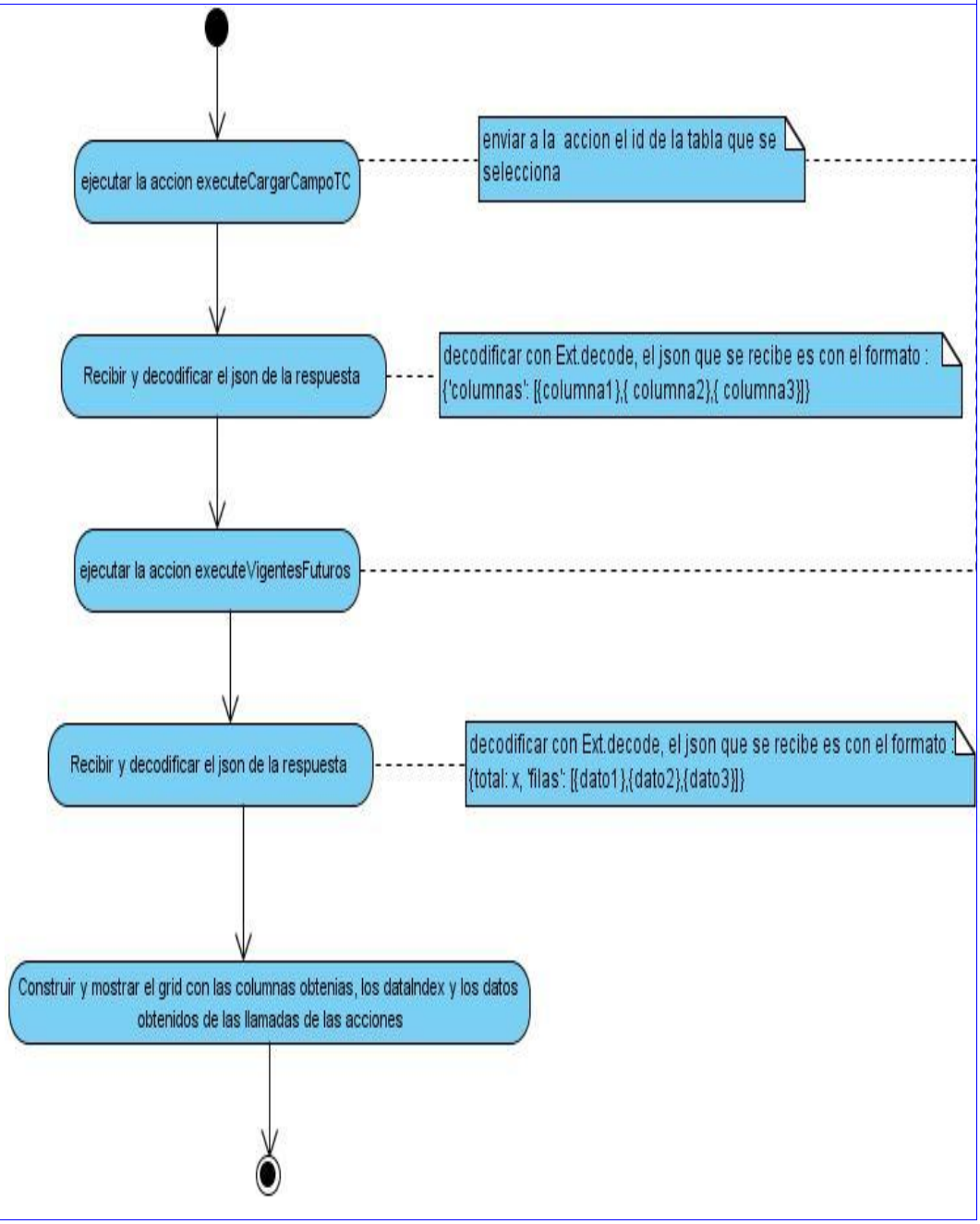
**Diagrama de Clases del Negocio**

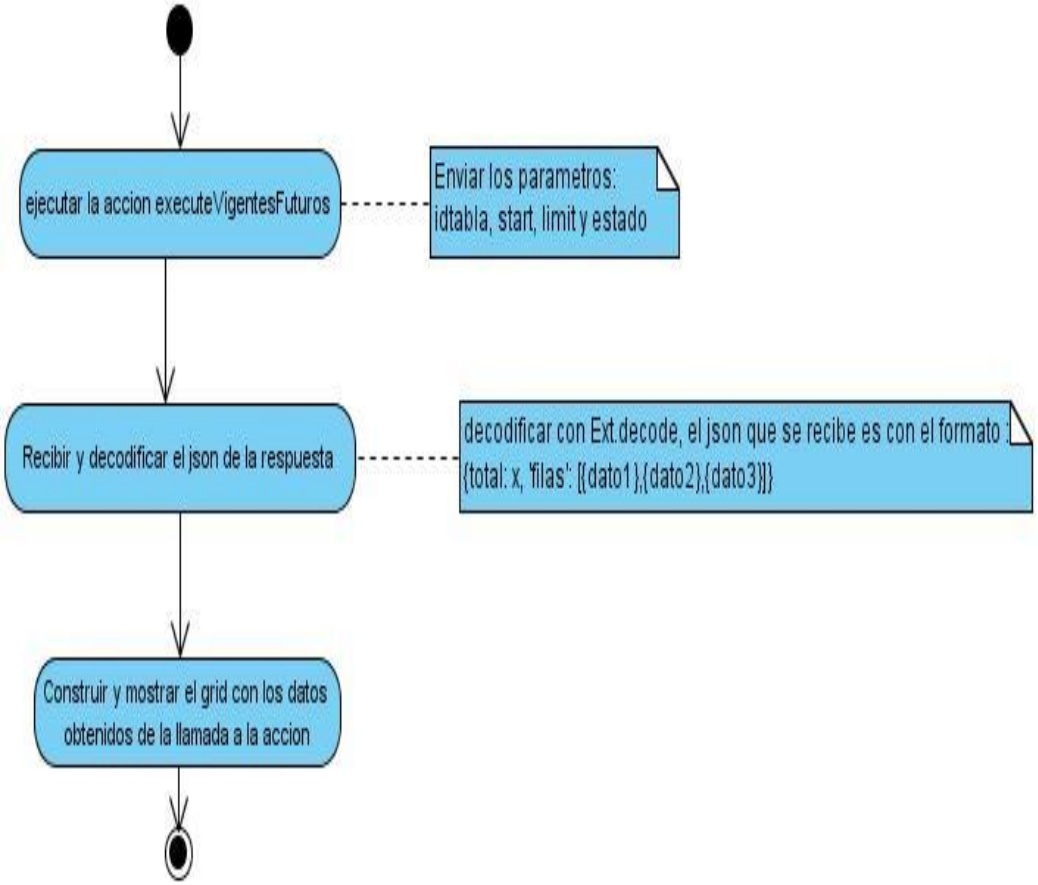


**Componentes Visuales**

***Componente Principal***

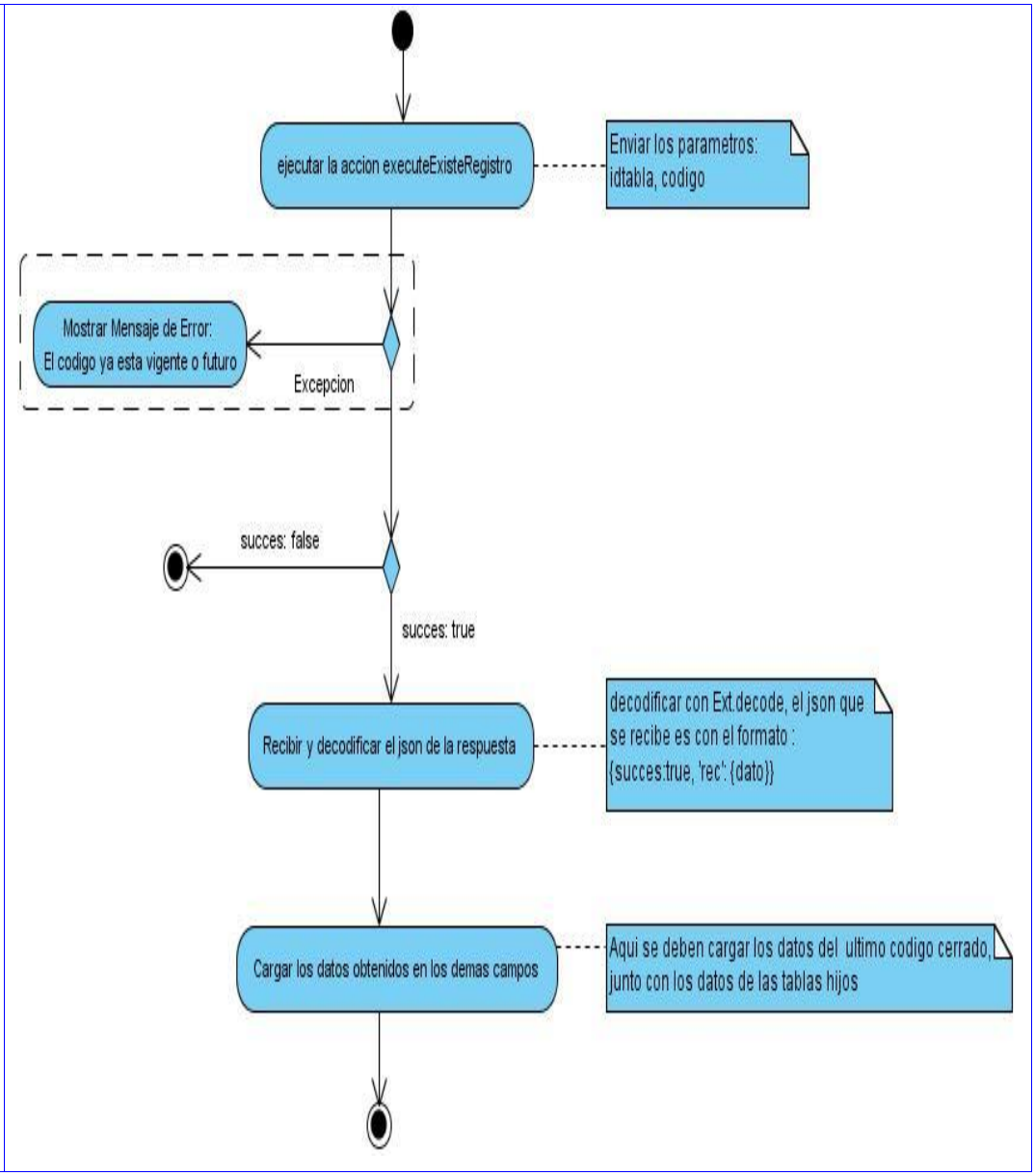
| | |
|---|---|
| **Componente** | *treePanel* |
| **Método** | *onClick (root)* |
| **Propósito** | Esta funcionalidad tiene como objetivo cargar las TC correspondientes al módulo. |
| **Descripción** |  |

| | |
|---|---|
| **Componente** | *treePanel* |
| **Método** | *onClick (node)* |
| **Propósito** | Esta funcionalidad tiene como objetivo cargar los campos de las TC y los datos vigentes y futuros. |
| **Descripción** |  |

| | |
|---|---|
| **Componente** | *Paginado* |
| **Método** | *onClick* |
| **Propósito** | Esta funcionalidad tiene como objetivo cargar los datos de la tabla especificada, dado un start y un limit. |
| **Descripción** |  |

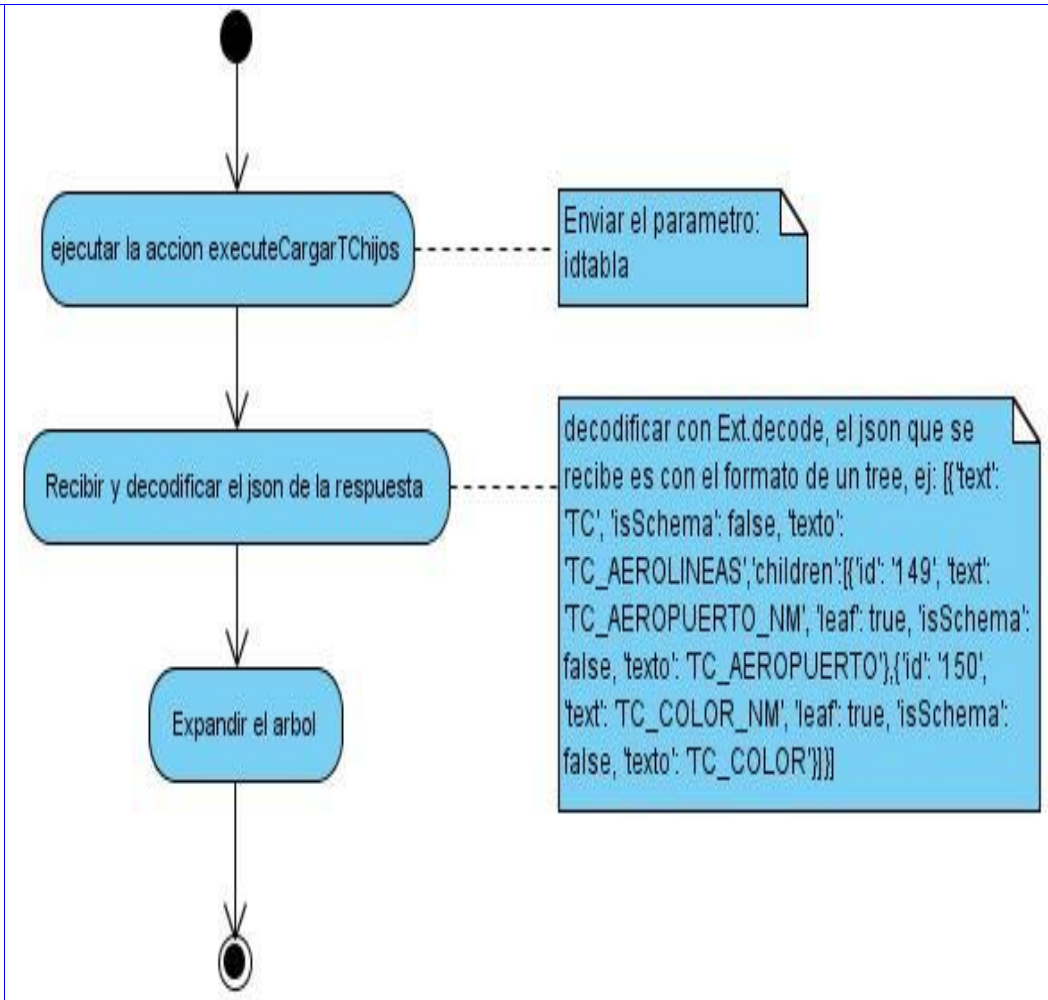
**Componente Dar Alta**

| | |
|---|---|
| **Componente** | *Dar Alta* |
| **Método** | *onBlur (texto del Código)* |
| **Propósito** | Esta funcionalidad tiene como objetivo verificar si el código introducido está cerrado, vigente o futuro y cargar los datos en caso de que exista ese código cerrado con los datos de las tablas hijos. |

| | |
|---|---|
| **Descripción** |  |

**Componente Gestionar Relaciones**

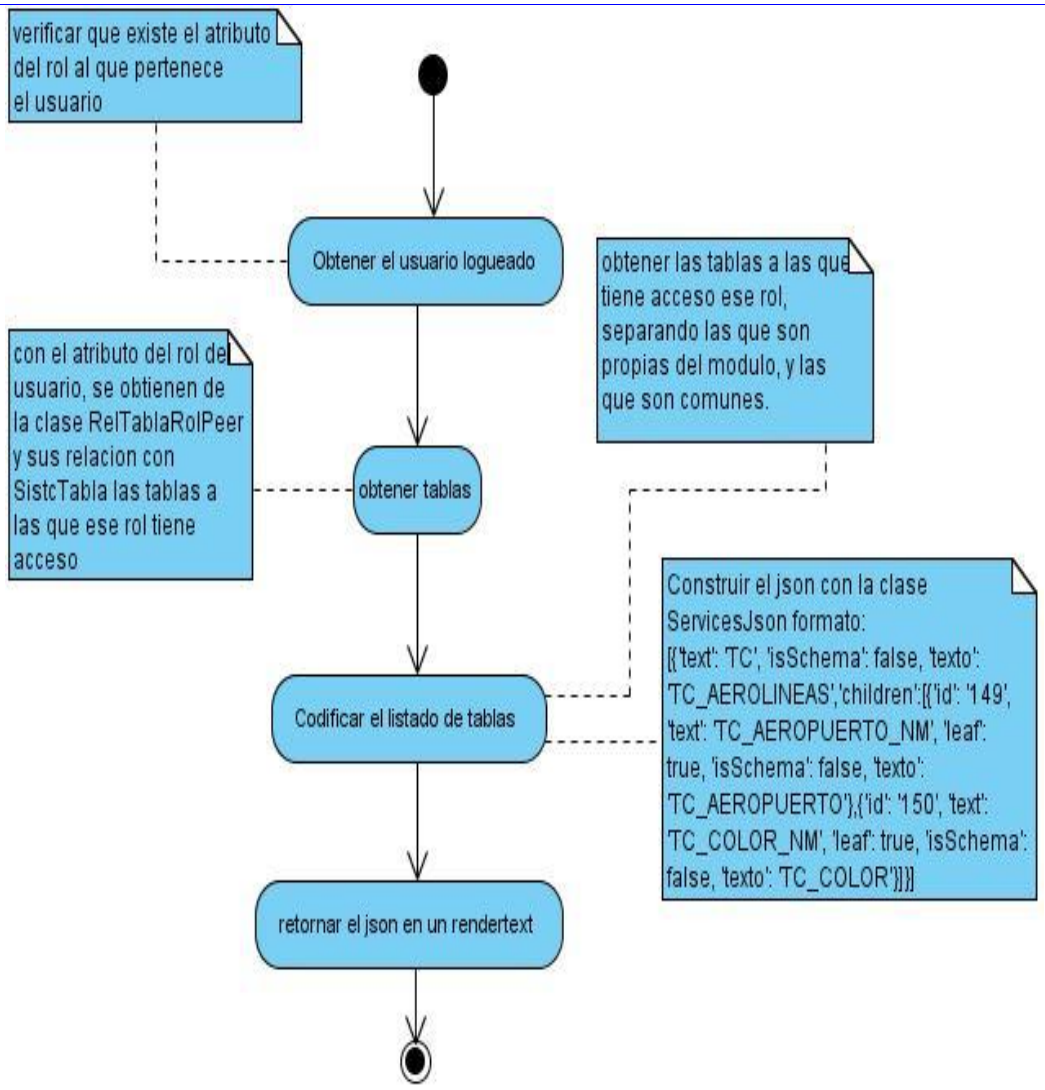
| | |
|---|---|
| **Componente** | *Gestionar Relaciones-treePanel* |
| **Método** | *onClick (root)* |
| **Propósito** | Esta funcionalidad tiene como objetivo cargar las TC hijas de la tabla donde se está dando de alta. |

| | |
|---|---|
| **Descripción** |  |

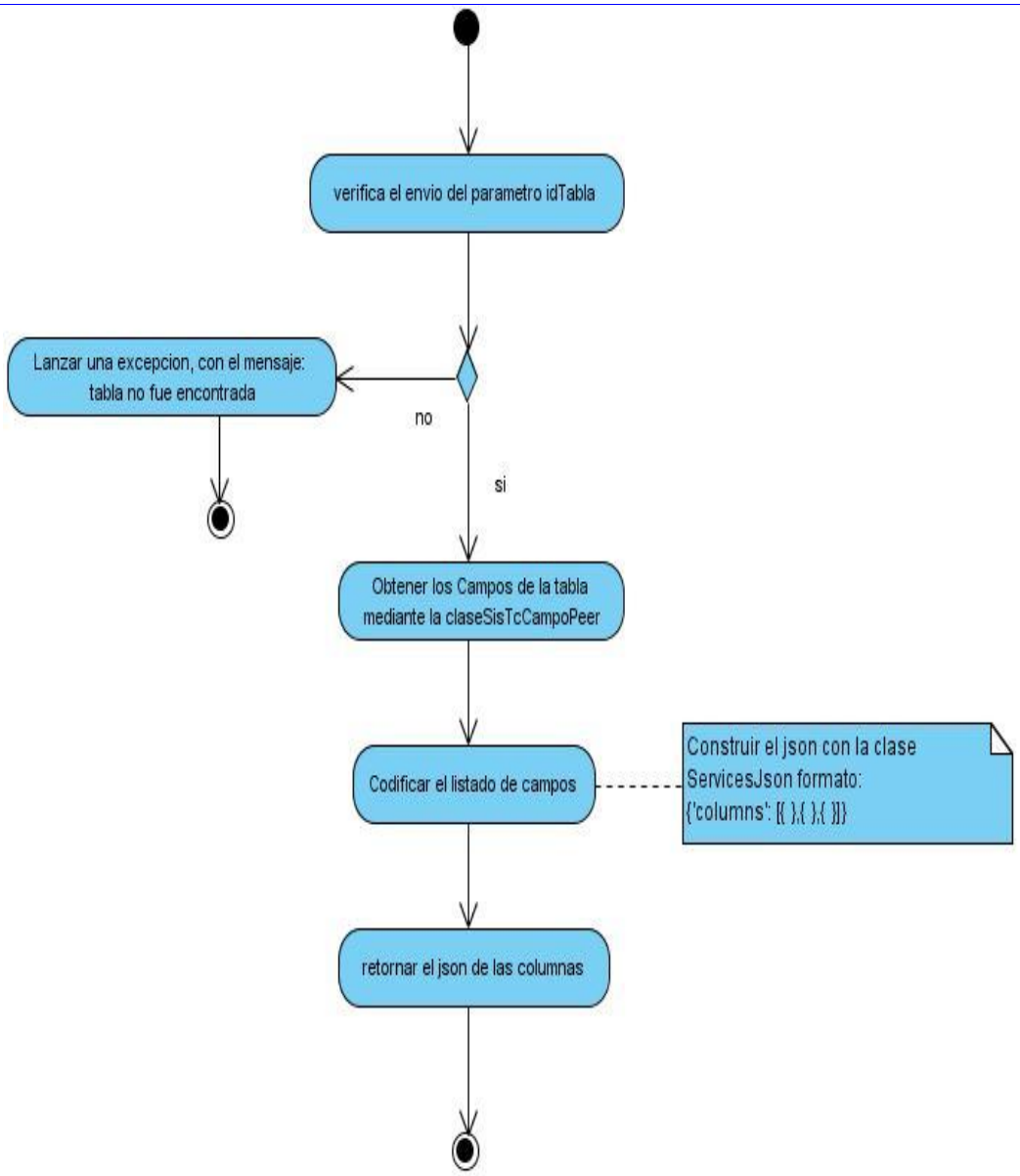
**Capa Acceso a Datos**

**Observaciones:** Cada una de estas acciones estará protegida por un bloque try{…}cath() {..}; para en caso de errores hacer rollback a la conexión creada y mostrar el error lanzado.

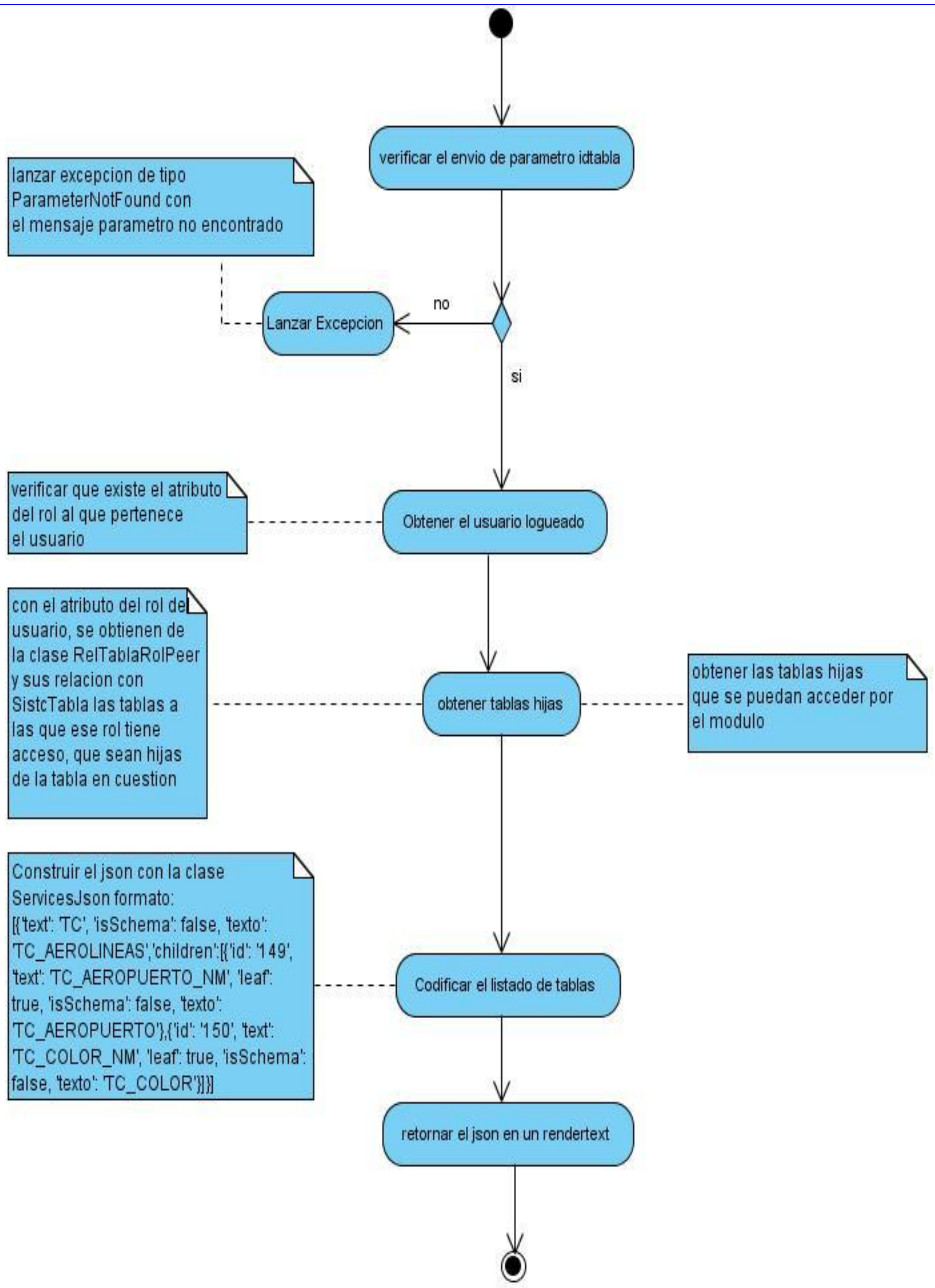
**Acción e*xecuteCargarTC***

| | |
|---|---|
| **Clase** | *tcActions* |
| **Método** | *executeCargarTC* |
| **Propósito** | Esta funcionalidad tiene como objetivo cargar las tablas de control a las que el módulo tiene acceso. |
| **Descripción** |  |

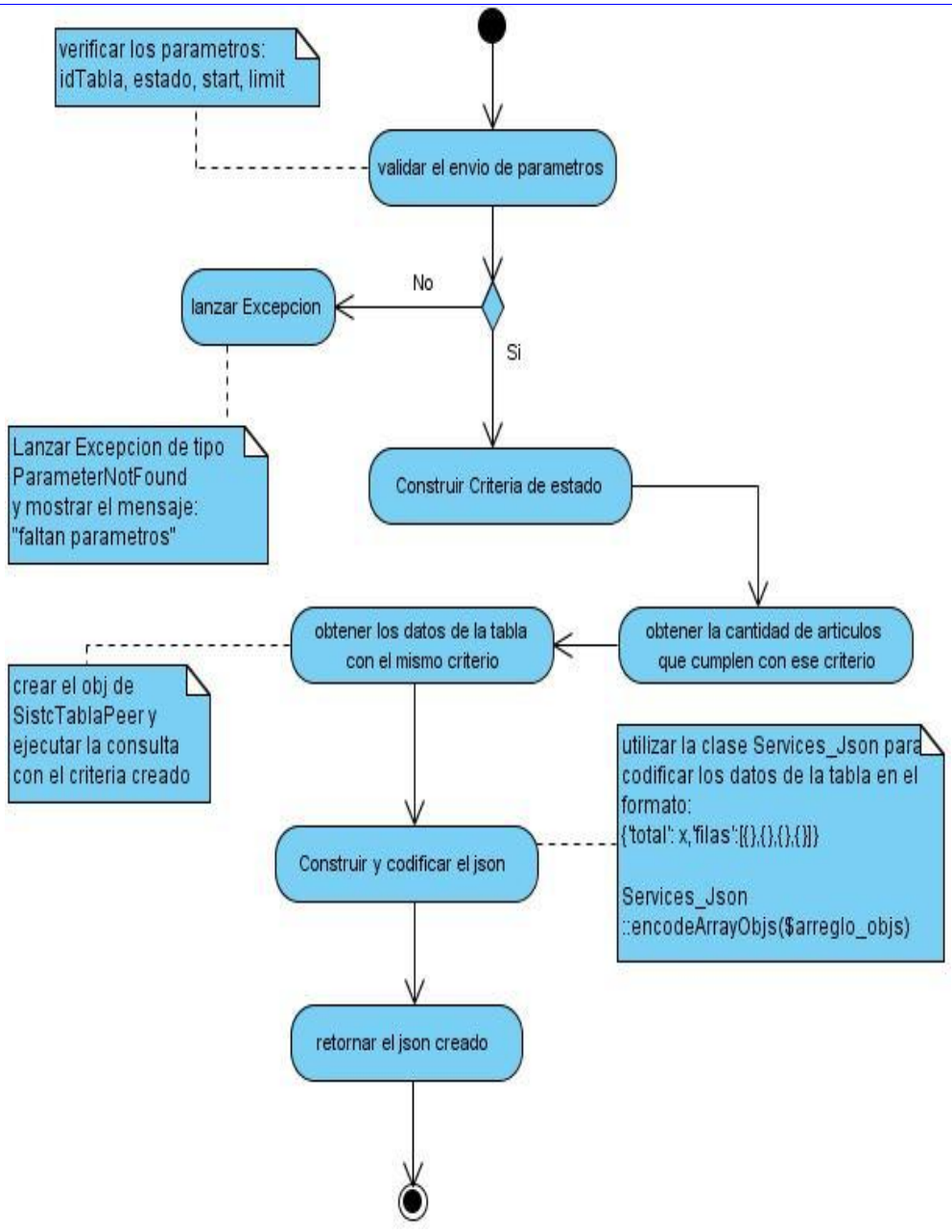
**Acción executeCargarCampoTc**

| | |
|---|---|
| **Clase** | *tcActions* |
| **Método** | *executeCargarCampoTc* |
| **Propósito** | Esta funcionalidad tiene como objetivo cargar los campos de la TC seleccionada. |
| **Descripción** |  |

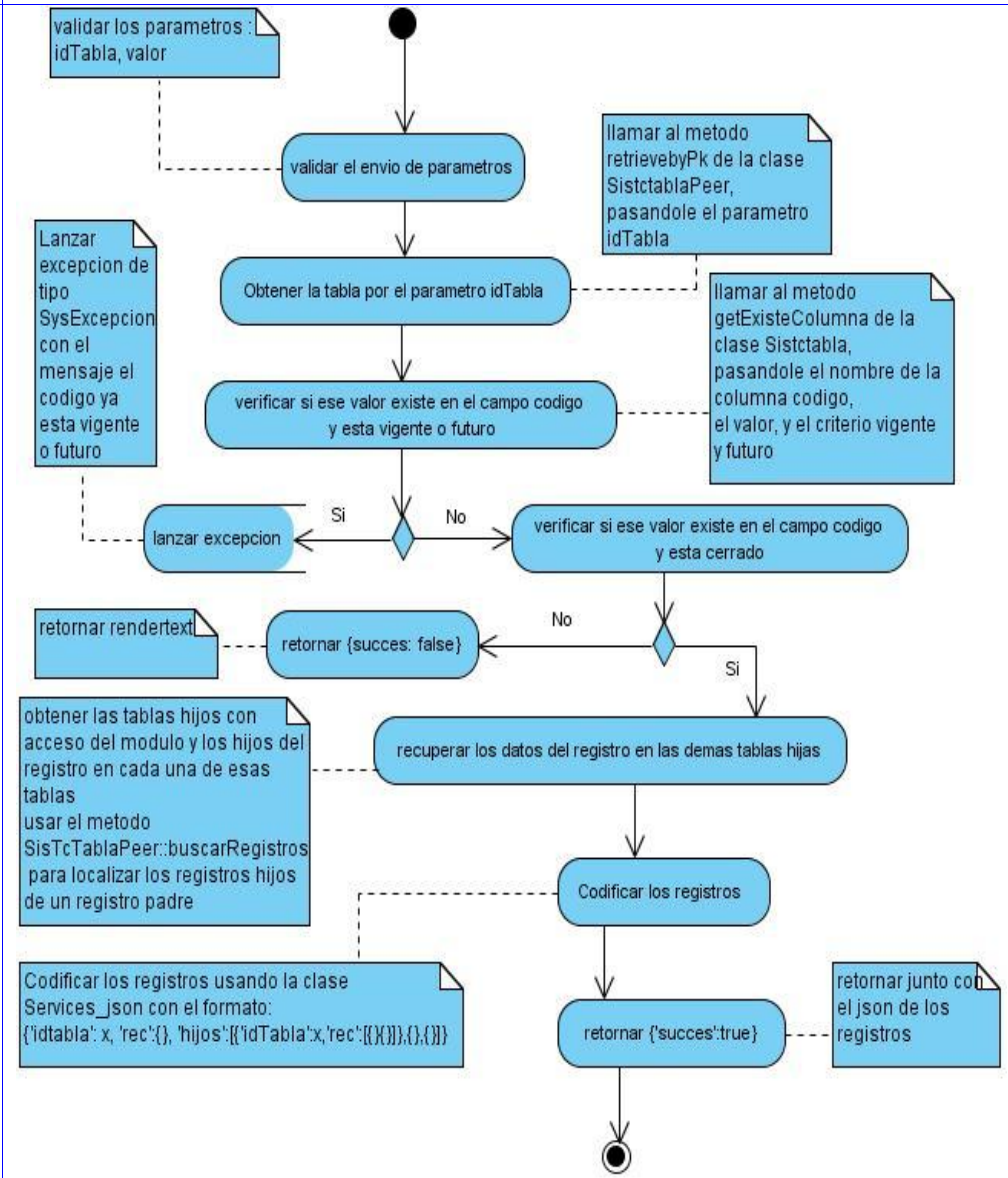
**Acción executeCargarTCHijos**

| | |
|---|---|
| **Clase** | *tcActions* |
| **Método** | *executeCargarTCHijos* |
| **Propósito** | Esta funcionalidad tiene como objetivo retornar las tablas hijas de la tabla seleccionada para dar alta o modificar. |
| **Descripción** |  |

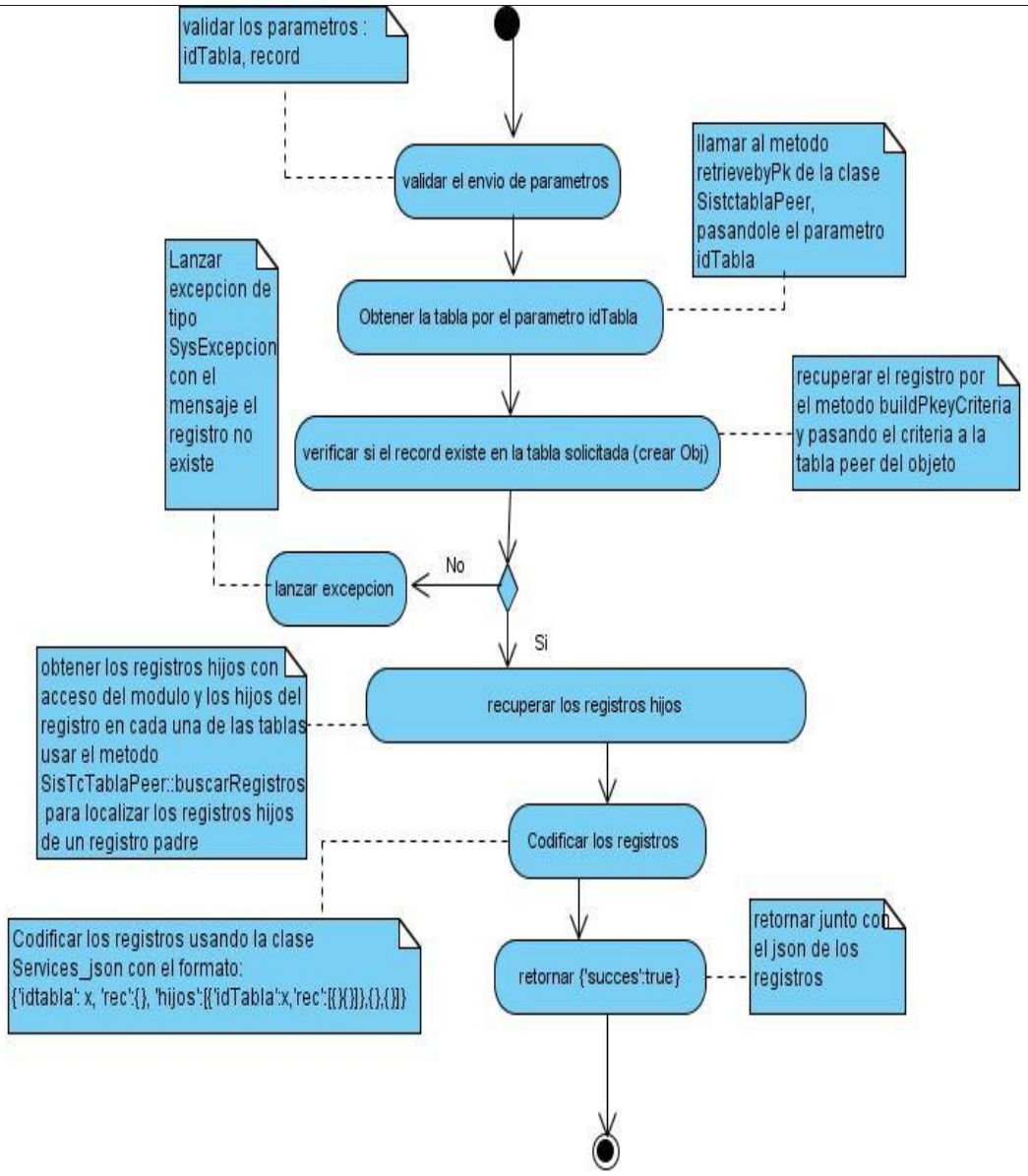
**Acción executeVigentesFuturos**

| | |
|---|---|
| **Clase** | *tcActions* |
| **Método** | *executeVigentesFuturos* |
| **Propósito** | Esta funcionalidad tiene como objetivo retornar los datos de una tabla por el estado de los artículos de dicha tabla. |
| **Descripción** |  |

**Acción executeExisteRegistro**

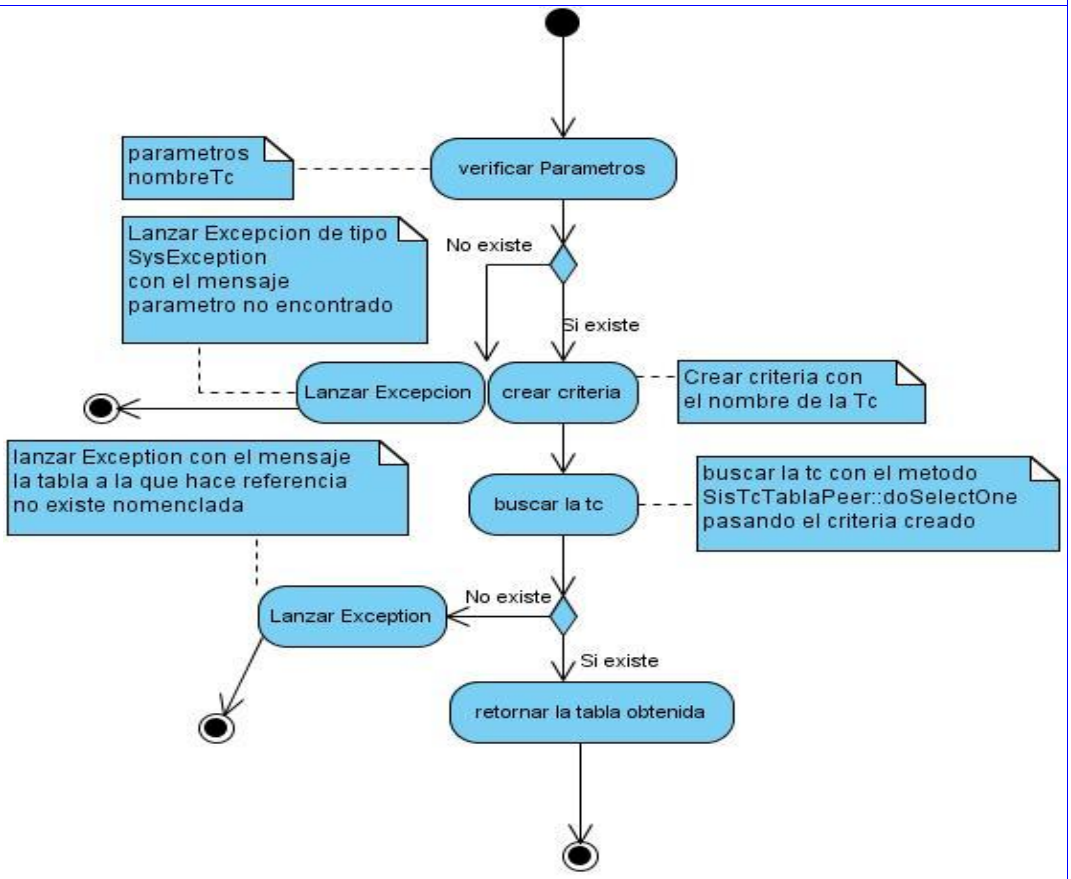
| | |
|---|---|
| **Clase** | *tcActions* |
| **Método** | *executeExisteRegistro* |
| **Propósito** | Esta funcionalidad tiene como objetivo verificar si cuando se va a dar de alta se introduce un código vigente o futuro, en cuyo caso se lanza una excepción, y si está cerrado se retorna el último artículo de ese código que estuvo vigente junto con sus relaciones. |
| **Descripción** |  |

**Acción executeSimilitud**

| | |
|---|---|
| **Clase** | *tcActions* |
| **Método** | *executeSimilitud* |
| **Propósito** | Esta funcionalidad tiene como objetivo buscar un registro junto con sus relaciones en las demás tablas. |
| **Descripción** |  |

**Servicio tc.obtenerTcNombre**

**Observaciones:** Cada uno de estos servicios se llaman mediante la clase SysComponentLocator.

| | |
|---|---|
| **Clase** | *SisTcTablaPeer* |
| **Método** | *obtenerTcNombre* |
| **Propósito** | Obtener un registro que contiene toda la información referente a una tabla especifica. |
| **Descripción** |  |

**Servicio obtenerDadoCriteria**

| | |
|---|---|
| **Clase** | *SisTcTablaPeer* |
| **Método** | *ObtenerDadoCriteria* |
| **Propósito** | Ejecutar un criterio en la TC especificada. |
| **Descripción** | |

**Anexo 2**

**Caso de prueba Dar alta a un registro**

**Descripción General**

Se crea un nuevo registro con los datos requeridos.

**Condiciones de Ejecución:**

El usuario encargado de la inserción debe tener los permisos necesarios.

**Secciones a probar en el Caso de Uso:**

| **Nombre de la sección** | **Escenarios de la sección** | **Descripción de la funcionalidad** | **Flujo Central** |
|---|---|---|---|
| SC 1: Dar alta a un registro | EC1.1: Inserción correcta | Se introduce un nuevo registro en el sistema luego de introducir los datos requeridos. | El usuario encargado introduce el código, descripción, fecha inicio, fecha fin del registro y presiona el botón aceptar. |
| | EC1.2: Datos incompletos | No son insertados todos los datos necesarios para la inserción y se muestra un mensaje de error informándolo. | El usuario encargado deja de introducir uno de los datos requeridos. |
| | EC1.3: Usuario repetido | El registro que se está insertando ya existe en la base de datos, es mostrado un mensaje de error. | El usuario encargado introduce el código, descripción, fecha inicio, fecha fin del registro y presiona el botón aceptar, pero este código del registro ya existe. |

**SC 1: Dar alta a un registro.**

| Id del escenario | Escenario | Código | Fecha Inicio | Fecha Fin | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|---|---|
| EC 1 | Inserción correcta | V | V | V | Se muestran los campos incorrectos. | |
| EC 2 | Datos incompletos | V(Vacío) | V | V | El sistema emite un mensaje para que llene los campos obligatorios. | |
| | | V | V(Vacío) | V | | |
| | | V | V(Vacío) | V(Vacío) | | |
| | | V(Vacío) | V(Vacío) | V(Vacío) | | |
| EC 3 | Registro repetido | V(=) | V | V | El sistema muestra un mensaje informativo. | |

**Caso de prueba Buscar registro**

**Descripción General**

Se busca el registro por los criterios de búsqueda especificados.

**Secciones a probar en el Caso de Uso:**

| Nombre de la sección | Escenarios de la sección | Descripción de la funcionalidad | Flujo Central |
|---|---|---|---|
| SC 1: Buscar registro | EC 1.1: Búsqueda satisfactoria | Se introducen los parámetros de búsqueda del registro y se muestran los datos del mismo. | El usuario introduce los parámetros del registro a buscar presiona el botón buscar. |
| | EC 1.2: No existe el registro | El usuario encargado de realizar la búsqueda introduce los parámetros de búsqueda del registro, pero este no existe en la Base de Datos. | El usuario encargado deja de introducir uno de los datos requeridos. |

**SC 1: Buscar registro.**

| Id del escenario | Escenario | Código | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|
| EC 1 | Búsqueda satisfactoria. | V () | Se muestra el registro solicitado. | |
| EC 2 | No existe el registro. | V(XXXX) | El sistema emite un mensaje diciendo que el registro no existe. | |

**Caso de prueba Cerrar registro**

**Descripción General**

Se selecciona el registro que se quiere eliminar. Este se elimina completamente si esta futuro.

**Condiciones de Ejecución:**

Al menos debe existir un registro en la BD.

**Secciones a probar en el Caso de Uso:**

| Nombre de la sección | Escenarios de la sección | Descripción de la funcionalidad | Flujo Central |
|---|---|---|---|
| SC 1: Cerrar registro | EC 1.1: Cerrar registro | Se selecciona el registro que se necesita eliminar y se elimina. | Se selecciona el registro que se requiere eliminar, se presiona el botón Cerrar Registro y es deshabilitado el registro de la BD. |

**Caso de prueba Modificar registro**

**Descripción General**

Se selecciona el registro que se desea modificar y se le realizan los cambios.

**Condiciones de Ejecución:**

Al menos debe existir un registro en la BD.

**Secciones a probar en el Caso de Uso:**

| Nombre de la sección | Escenarios de la sección | Descripción de la funcionalidad | Flujo Central |
|---|---|---|---|
| SC 1: Modificar registro | EC 1.1: Modificación correcta | Se selecciona el usuario que se quiere modificar y se introducen los nuevos datos. | Se selecciona la opción Modificar Registro, escoge el registro que se quiere modificar y se introducen los nuevos datos, se valida la completitud de los mismos se presiona el botón Aceptar y es modificado el registro. |
| | EC 1.2: Datos incompletos | No son insertados todos los datos necesarios para la modificación y se muestra un mensaje de error informándolo. | Se selecciona la opción Modificar Registro, escoge el registro que se quiere modificar y no son introducidos todos los datos del mismo para modificarlo. |
| | EC 1.3: Registro repetido | Existe un registro que posee los mismos datos que los que se introdujeron en el usuario seleccionado para la modificación. | Se selecciona la opción Modificar Registro, escoge el usuario que se quiere modificar, se introducen los nuevos datos, se valida la completitud de los datos y la no existencia de un registro igual en la Base de Datos, se comprueba que si existe ya un registro igual. |

**SC 1: Modificar registro.**

| Id del escenario | Escenario | Código | Fecha Inicio | Fecha Fin | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|---|---|
| EC 1 | Inserción correcta | V | V | V | Se muestran los campos incorrectos. | |
| EC 2 | Datos incompletos | V(Vacío) | V | V | El sistema emite un mensaje para que llene los campos obligatorios. | |
| | | V | V(Vacío) | V | | |
| | | V | V(Vacío) | V(Vacío) | | |
| | | V(Vacío) | V(Vacío) | V(Vacío) | | |
| EC 3 | Registro repetido | V(=) | V | V | El sistema muestra un mensaje informativo. | |

**Anexo 3**

Certificado de aceptación del módulo Tablas de Control.

## CERTIFICADO DE ACEPTACIÓN

Yo, Alain E. Rodríguez Arias, como J'Dpto Soluciones Aduanas de la (del) CEIGE certifico que la solución: Tablas de Control está siendo utilizada en la Aduana General de la República.

Resumen valorativo:

El desarrollo de esta aplicación constituye un paso de avance en el proceso de informatización de la Aduana General de la República. Esta solución proporciona una mejora considerable en las condiciones de trabajo de los especialistas de las distintas áreas de la Aduana; ya que logra optimizar la labor de los funcionarios y agilizar el trámite aduanero, se disminuyen al mínimo los errores que puedan existir por mala manipulación de los códigos o desconocimiento de los mismos, permite manejar de manera centralizada y sencilla la información gestionada mediante Tablas de Control y es compatible con las nuevas tecnologías desplegadas en la Aduana. La aplicación lleva más de 9 meses de explotación en la entidad y los trabajadores aduaneros que laboran con la misma consideran que se encuentra en óptimas condiciones y cumple con las características necesarias para su utilización.

Y para que así conste, se firma la presente a los 14 días del mes de Junio de 2011.

Firma / Cuño

**GLOSARIO**

**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.

**Interfaz de usuario:** Es la parte de una aplicación que se encarga de interactuar con el usuario.

**Módulo:** Es un componente autocontrolado de un sistema, el cual posee una interfaz bien definida hacia otros componentes; algo es modular si es construido de manera tal que se facilite su ensamblaje, acomodamiento flexible y reparación de sus componentes.

**Componente:** Agrupación de elementos que hacen parte de un subsistema. -Subsistema: Parte del sistema global con una interfaz razonablemente bien definida.

**SAP AG:** (Systeme, Anwendungen und Produkte) (Sistemas, Aplicaciones y Productos).

**JSON:** Acrónimo de 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.

**Histórico:** Conjunto de registros en una base de datos que describen las modificaciones realizadas durante el transcurso del tiempo a un concepto o entidad.

**AJAX:** Acrónimo de *Asynchronous JavaScript And XML* (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (*Rich Internet Applications*). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

**ORM:** (Object Relational Mapping) Mapeo de Objetos Relacional es una técnica de programación para convertir tipos de datos incompatibles entre sistemas de bases de datos y lenguajes orientados a objetos.

**Componente:** Agrupación de elementos que hacen parte de un subsistema.

**Subsistema:** Parte del sistema global con una interfaz razonablemente bien definida.

**Datos Maestros:** Los datos maestros son datos que describen uno o más atributos de entidades centrales como clientes, proveedores, productos e inventario.