Universidad de las Ciencias Informáticas.
Facultad 3

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

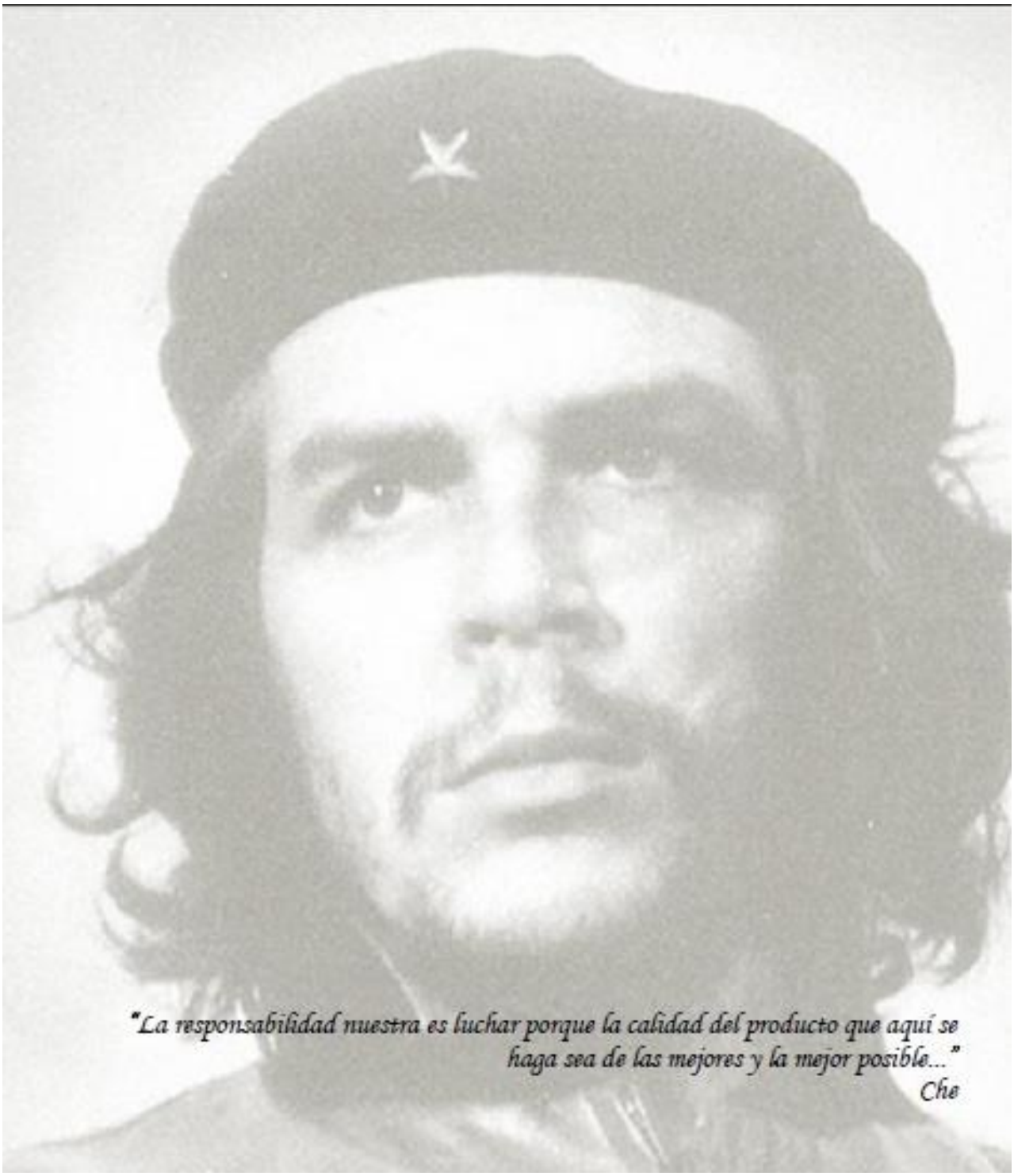
Título: Diseño, implementación y prueba del componente Ajuste al Costo por Procesos del subsistema Costos y Procesos de CEDRUX.

Autor: Carlos Daniel García Hernández

Tutores: Ing. Yanay Hernández Sosa
Ing. Juan Alberto Caballero Portelles

La Habana, Cuba

Julio 2013


"La responsabilidad nuestra es luchar porque la calidad del producto que aquí se haga sea de las mejores y la mejor posible..."
Che

Declaro que soy el único autor del presente trabajo “Diseño, implementación y prueba del componente Ajuste al Costo por Procesos del subsistema Costos y Procesos de CEDRUX” y autorizo a la Universidad de las Ciencias Informática a hacer uso del mismo en su beneficio.
Para que así conste firmamos la presente a los ____ días del mes de _____del año _____.

_____
Carlos Daniel García Hernández

_____ _____

Ing. Yanay Hernández Sosa Ing. Juan Alberto Caballero Portelles

Ing. Yanay Hernández Sosa

Ingeniera en Ciencias Informáticas, graduada en la Universidad de las Ciencias Informáticas en el 2009. Se desempeña como Jefa de Línea de Contabilidad dentro de CEIGE.

correo: yhsosa@uci.cu

Ing. Juan A Caballero Portelles

Ingeniero en Ciencias Informáticas, graduado en la Universidad de las Ciencias Informáticas en el 2011. Se desempeña como Arquitecto de sistema y desarrollador en CEDRUX.

correo: jacaballero@uci.cu

Antes que nada agradecer a la revolución y a nuestro comandante Fidel Castro Ruz.

A mi madre querida que siempre ha confiado en mí, que me ha enseñado a ser un hombre mejor en la vida, a siempre ayudar al necesitado, a valorar lo sentimental y no lo material, a saber escuchar cuando tenga o no la razón, a ser agradecido, y millones de valores de los cuales hoy me hacen una mejor persona. Por todo el amor que me has dado, te dedico esta tesis. Te quiero mamita

A mi padre que lo quiero con la vida, siempre dándome consejos, y compartiendo conmigo momentos inolvidables, como aquella final de Francia e Italia un momento de rivalidad y tensión entre nosotros, el por Italia y yo por Francia, o cuando vamos al latino a disfrutar del mejor equipo de pelota de Cuba, también le agradezco el haberme inculcado todo conocimiento de cine y música (mis hobits preferidos).

A mi tía Matilde que supo llevar las riendas de la familia en los momentos duros, que supo reponerse de tantas caídas y problemas por los que pasamos cuando mama estuvo de viaje, a siempre tener una solución para mis problemas, a ser la mejor cocinera de todas y prepararme mis platos preferidos.

A mi abuela que como ella siempre dice, saliste bien porque le recé a dios. Quiero dedicarte esta tesis a ti, por todo el amor y cariño que me has dado, por criarme y aguantar todas mis malas crianzas, por correr conmigo cuando estaba enfermo.

A mi primo Rey, que si no fuera por su ayuda no estuviera aquí, gracias por todo lo que hiciste.

A mi prima Arlene, que te quiero con la vida, que eres como una hermana para mí.

A mis abuelos Mireya y Juanito.

A mis tíos en especial a Olga.

Agradecimientos

A mis tutores Yanay y Juan por ayudarme a llegar a este momento, por hacerme esforzar, por llenarme de conocimiento, por confiar en mi hacer posible esto.

A Joisel que mostró ser una persona increíble, con toda su ayuda, concejos y apoyo que me diste, por buscar siempre un hueco para explicarme algo. Te admiro amigo personas como tu existen pocas en la vida.

A Giselle, que es una persona maravillosa y me ayudó mucho en la recta final, y le estoy agradecido.

A el tribunal por decir la frase que todos esperamos, se le otorga el título de ing. Por la ayuda brindada, por hacerme superar mis conocimientos.

A mis amigos de la UCI:

Quiero empezar agradeciendo a mis hermanos caídos en combate, que siempre los tengo presente en cada anécdota y momento que pasamos juntos. Alexei, Juan, Alain, Ranero, Jota, Raudel. De los cuales se extraña mucho su compañía, en especial a Jota que llegó hacer mi mano derecha, cualquier duda de programación, problema de informática, problema familiar él estuvo presente.

A Yandy por ser mi hermanito chiquito del cual siempre mortifico, David por siempre seguir mis pasos de maldad, a Juanca y Hernán por toda la ayuda y conocimiento de programación que me dieron, Alfredo por siempre estar presente en nuestras platicas de rock, futbol, chuchos. A la sangre, que lo único que me gusta de él es la novia que tiene.

A Rolando que ha sido un amigo estelar, siempre presente, dándome concejos, cada vez que me sentía mal por algo que me había pasado, estuvo presente. Te llevo aquí mi herma, sabes que puedes contar conmigo para lo que sea.

A Jiubel, Pompa y Yoandry los únicos sobrevivientes de primer año, con los que siempre me he sentido bien, a pesar de que estemos en grupos separados, o no andemos juntos sé que puedo contar con ellos.

Agradecimientos

A mis mujeres de toda la carrera Lili, Danay, Liliam, Yudi, Yuri, Aliska, Ayle las quiero, saben que mi corazón es de ustedes, en especial mi negrita y rubita.

A la gente del edif. 7 Mono, Evelio, Aroldo, Diogenes, Raidel, Pollo, Camilo, Mantys, Luis Carlos, Ramón, el Cangri, Los 2 Jorges, Los Yordan, Lisvany, El Narra, a Karel que no me ayudó ni un poquito.

A la gente del grupo Yasmany, Miguel, Yesika, Gustavo, Yenni, Leiza.

Y como dice nuestro padre Alexander, ya después que te gradúes o termines la universidad es a trabajar, otros a cuidar niños y otros que viven lejos, quien sabe cuándo nos veamos de nuevo, entonces lo que nos queda son los momentos que pasamos juntos, momentos como los días de champion cuando nos reuníamos todo el piquete en casa de Cesar y Evelio a ver el futbol y a discutir. O las broncas de Raidel y Diogenes contra David y Alfredo, o cuando Hernán se berreaba que no había quien le ganara una discusión, o se ponía a discutir con Camilo a las 3 de la mañana de netbeans, o cuando Raidel me abrió la Toshiba para arreglármela y lo que hizo fue rompérmela o las noches de futbol, cuando el piquete del edif. 7 se reunían hasta las 12 am para jugar futbol y quitarnos un poco el estrés y problemas, donde siempre alguien terminaba con un golpe (Mantys con un ojo inchado, Cangri con un pelotaso en el pecho a quemarropa). Por eso disfrutemos cada momento al máximo como si fuera el ultimo, que la vida es corta y la universidad ya se acabó.

Índice

Índice de figuras

Índice de tablas

Resumen

Desde tiempos remotos la contabilidad ha sido una herramienta para el control y seguimiento de la economía, la cual provee de información a las empresas para que estas puedan realizar el proceso de gestión, planeación y administración. El ajuste al costo, es un mecanismo necesario dentro de la contabilidad de costo, ya que permite realizar planes sobre el comportamiento económico futuro, basados en ganancias y pérdidas para poder establecer los costos reales de producción. El sistema empresarial cubano presenta la necesidad de informatizar todos sus procesos de gestión, por lo que en la Universidad de las Ciencias Informáticas (UCI) junto al Organismo de la Administración Central del Estado (OACE) se viene desarrollando una solución informática denominada CEDRUX, perteneciente al proyecto CEIGE (Centro de Informatización de Gestión de Entidades), con el propósito de suplir dicha necesidad. El presente trabajo consiste en desarrollar un componente para la aplicación del ajuste al costo, lo cual será de gran importancia, ya que facilitará la toma de decisiones en las entidades cubanas. Cuenta con un estudio del estado del arte enmarcado en los sistemas vinculados a la realización de dicho proceso. La especificación, construcción y documentación de la solución se realizó basándose en el modelo de desarrollo propuesto por la subdirección de producción del CEIGE. Se realiza además una descripción de las herramientas, tecnologías y lenguajes necesarios para el desarrollo de dicho componente. Se logra medir la calidad de la solución aplicando pruebas para validar la implementación de las clases.

Palabras claves: Ajuste al Costo, CEIGE, componente.

Introducción

La competencia entre las empresas es una característica del mundo industrializado, que ha traído un incremento progresivo de la calidad de los productos o servicios y de los procesos que se han de desarrollar por obtenerlos; para lograrlo se ha hecho necesario el uso de sistemas informáticos que permitan elaborar un producto de la forma más rápida y eficiente (1). Para lograr estos es necesario utilizar las Tecnologías de la Información y las Comunicaciones (TIC), las cuales forman parte de la cultura tecnológica contemporánea, permitiendo ampliar las capacidades físicas, intelectuales, científicas, culturales así como posibilitan un mayor desarrollo social, de gran impacto en todos los ámbitos de la vida cotidiana, haciéndose cada vez más difícil que se pueda actuar eficientemente prescindiendo de ellas.

La productividad y eficiencia en el cumplimiento del objeto social de las empresas está fuertemente apoyado en el registro de sus hechos contables. La Contabilidad de Costos permite conocer, entre otras cosas, información básica para la toma de decisiones económicas y financieras; uno de los procesos implicados en esta actividad lo constituye el ajuste de los costos reales de la producción al final de cada período. Las empresas realizan planes sobre su comportamiento económico futuro, basados en ganancias y pérdidas para poder establecer los costos reales de producción; a este mecanismo se le denomina Ajustes al Costo por Procesos.

El país ha hecho uso de sistemas informáticos con estos fines, como por ejemplo, los Sistemas Contables, que tienen como objetivo brindar soluciones para informatizar la mayoría de los procesos básicos de una organización empresarial: La Gestión Financiera, Logística, Contabilidad, Costos y Procesos, Recursos Humanos, entre otras, pero en su mayoría no se ajustan a las características de la economía cubana, son propietarias, o están desarrolladas sobre tecnologías y herramientas privativas.

La Universidad de las Ciencias Informáticas (UCI) en conjunto al Organismo de la Administración Central del Estado (OACE) se encuentra desarrollando un Sistema Integral de Gestión (en lo adelante CEDRUX).Dicho sistema surge bajo la necesidad de informatizar los procesos económicos dentro de las empresas, CEDRUX está compuesto por los subsistemas: Contabilidad, Costos y Procesos, Caja, Banco, Cobros y Pagos, Inventario y Planificación.

El subsistema Costos y Procesos es el encargado de gestionar las operaciones asociadas a los costos y dentro de este, se encuentra en elaboración el componente Ajuste al Costos por Procesos, que tiene como función ajustar los costos al final de cada período, ya sea por producción o servicios brindados y a partir de una evaluación realizada previamente. Este componente será motivo de estudio y desarrollo durante este trabajo de diploma.

En informática para realizar una aplicación o componente, generalmente se sigue una metodología o modelo que guie el desarrollo, en el curso 2011-2012 se realizó un trabajo de diploma por la ingeniera Doralis Sevilla Reyes (2) guiado por el modelo de desarrollo que propone la Subdirección de Producción del CEIGE donde se generaron los artefactos que se describen a continuación: Descripción de los requisitos funcionales, Diagrama de clase de diseño y Modelo de datos, conllevando al siguiente **problema a resolver**: ¿Cómo lograr un componente informático que permita gestionar el proceso Ajuste al Costo por Procesos en el subsistema Costos y Procesos de CEDRUX, partiendo de los artefactos obtenidos durante el análisis?

Se plantea por tanto como **objeto de estudio:** el proceso de desarrollo de software, especificándose como **campo de acción:** la implementación y validación del componente de Ajuste al Costo por Proceso del subsistema Costos y Procesos.

Para dar solución a la problemática descrita se establece como **objetivo general:** desarrollar el componente Ajuste al Costo por Procesos en el subsistema Costos y Procesos de CEDRUX a partir de la información modelada.

Se trazan como **objetivos específicos**:

- Fundamentar la investigación mediante la elaboración del Marco teórico.
- Diseñar el componente Ajuste al Costo por Procesos.
- Implementar el componente Ajuste al Costo por Procesos.
- Validar el componente Ajuste al Costo por Procesos.

Esta investigación está basada en la siguiente **idea a defender:** con el desarrollo de un componente para la realización del Ajuste al Costo por Procesos en el subsistema Costos y Procesos de CEDRUX y siguiendo las normativas de calidad, se contribuirá a la toma de

decisiones en la gestión de la Contabilidad de Costos en las entidades cubanas una vez puesto en explotación el sistema.

Se establecen las siguientes **tareas de la investigación**:

OE1.

1. Elaborar el marco conceptual de la investigación.
2. Revisar los artefactos generados como resultado de la fase Análisis del proceso Ajuste al Costo por Procesos, realizando análisis valorativo.
3. Analizar el Modelo de desarrollo de software a emplear.
4. Analizar herramientas y tecnologías a utilizar para el desarrollo de la propuesta de solución.

OE2.

1. Elaborar los Diagramas de clases del diseño.
2. Elaborar Descripción del diseño.
3. Elaborar el Modelo de datos de la solución.
4. Describir las clases del modelo.

OE3.

1. Realizar diagrama de componentes.
2. Definir estándares de codificación.
3. Realizar descripción por funcionalidades.
4. Implementar las interfaces de usuario usando Ext Js.
5. Implementar las clases modelos y controladoras haciendo uso del lenguaje PHP.
6. Implementar las clases de acceso a datos favoreciendo la rapidez y seguridad de la transferencia de datos, haciendo uso de la tecnología definida por el proyecto.

OE4.

1. Validar el diseño mediante la aplicación de métricas.

2. Validar el software mediante la aplicación de pruebas de caja blanca y caja negra.

Capítulo 1: Fundamentación teórica

1.1 Introducción del capítulo

El presente capítulo tratará sobre los diferentes sistemas contables que han sido implementados nacional e internacionalmente, a partir de esta base se analiza qué aspectos deben ser tomados para el desarrollo del componente, así como las principales características, ventajas y desventajas. Se expondrán un conjunto de tecnologías y herramientas que se tomarán en cuenta a la hora de realizar el desarrollo del trabajo de diploma. Se abordará sobre los lenguajes de programación a usar, lenguaje de modelado y framework.

1.2 Marco conceptual

El marco conceptual estará centrado en la definición del proceso de Ajuste al Costo por Proceso; que consiste específicamente en ajustar los costos reales al final de cada período, ya sea por concepto de producción o servicios brindados partiendo de una estimación hecha al comienzo y donde se tiene en cuenta la existencia y los destinos de los productos en cada una de las entidades.

1.3 Conceptos básicos asociados al dominio del problema

Costos: es el gasto económico que representa la fabricación de un producto o la prestación de un servicio (3).

Proceso: secuencia ordenada y lógica de actividades, generalmente repetitivas, que se realizan en la organización por una persona, grupo o departamento, con la capacidad de transformar unas entradas en salidas o resultados programados para un destinatario con un valor agregado (4).

Ajustes al Costo: el Ajuste al Costos, también llamado indexación, es el decremento o incremento que sufre un precio en su costo directo por causas o circunstancias de orden económico no previstas y totalmente ajenas a la voluntad de las partes contratantes (4).

1.4 Sistemas contables vinculados al campo de acción

SAP ERP

Es un software desarrollado en la ciudad de Mannheim, Alemania, por antiguos empleados de IBM (International Business Machines). Puede configurar cada uno de los procesos según las necesidades de las empresas con el fin de obtener una ventaja competitiva en su sector. SAP ERP (Servicios, Aplicaciones, Productos) está diseñado para incrementar la eficiencia de la planificación y la gestión de procesos a lo largo de las empresas. Ofrece dos conjuntos de funcionalidades: uno centrado en el soporte a las operaciones y otro centrado en la generación de valor.

Dentro de la primera funcionalidad se permite gestionar operaciones logísticas, satisfacer requisitos de calidad y cumplir con la normativa y estándares de su industria. También cuenta con las herramientas para el desarrollo y la introducción de nuevos productos con cobertura para el ciclo de vida completo del producto. El segundo conjunto de funcionalidades está diseñado para incrementar la eficiencia en los procesos de producción. Permite mejorar la totalidad de las operaciones logísticas relacionadas con la creación y mejora de sus productos y servicios.

Dentro de sus áreas de funcionalidades se encuentran:

- Contabilidad general
- Presupuesto
- Bancos
- Informes financieros

Funciona sobre sistema operativo Windows y está soportado sobre base de datos Oracle, además que está implementado en .NET y está preparado para trabajar con él mediante la web. (5)

Observación: SAP, como aplicación informática da la posibilidad de ejecutar todos los procesos del negocio; incluida la administración, gestión de las relaciones con los clientes, operaciones, seguimiento, análisis y mejora del sistema de control de costes y de la contabilidad analítica. Dentro de sus inconvenientes principales está, que es un software privativo.

ASSETS NS

Es un Sistema de Gestión Integral estándar y parametrizado que permite el control de los procesos de Compras, Ventas, Producción, Taller, Inventario, Finanzas, Contabilidad, Presupuesto, Activos Fijos, Útiles y Herramientas y Recursos Humanos. Es una aplicación cliente-servidor programada en *Visual Basic* 6.0 y Microsoft SQL Server 2000, utilizando adicionalmente Crystal Reports 7.0 para la generación de reportes de salidas.

Como sistema integral todos sus módulos trabajan en estrecha relación, generando automáticamente al Módulo de Contabilidad los Comprobantes de Operaciones por cada una de las transacciones efectuadas, esto permite que se pueda trabajar bajo el principio de Contabilidad al Día. Dispone además de métodos novedosos para administración y planificación de inventarios, así como una amplia gama de análisis y consultas que le permitirán no solo conocer exactamente la situación actual, sino proyectar decisiones futuras.

Es un sistema flexible, amigable, con ayuda en línea que puede ser instalado en una microcomputadora o sobre varias, funcionando en ambiente multiusuario incluidas estaciones remotas. ASSETS NS está diseñado para Multi-Compañía, con una estructura organizativa a varios niveles, en la que podrán existir: Grupo Corporativo, Corporativo, Grupo de Agrupaciones, Agrupación, Almacenes y Centros de Costos. (6)

Observaciones: es un sistema muy usado en el país sobre todo por empresas como el Ministerio de Ciencia, Tecnología y Medio Ambiente (CITMA), pero sin embargo el proceso Ajuste al Costo por Procesos comprende una gama de funcionalidades que no están comprendidas dentro de sus componentes.

Versat-Sarasola

El Versat-Sarasola es un sistema de gestión contable-financiero, representa un ejemplo de sustitución de importaciones en materia de aplicaciones informáticas. El nombre de Versat se origina por la versatilidad y Sarasola, a partir del apellido de un experimentado contador, que fue acreedor al Premio Nacional de Contabilidad. Es un software creado por el país a finales de los noventa, dado que el Ministro del Azúcar (MINAZ) inició la búsqueda de un software integrado para automatizar su actividad económica, contable y financiera.

Sus creadores definen al VERSAT-Sarasola como “un paquete integrado para la gestión económica financiera que permite enviar información eficaz, de forma inmediata, desde lugares apartados, a la vez que ofrece mayor organización, control y disciplina en cada gestión”. Su principal creador, Miguel Cabrera González, es un contador profesional, Licenciado en Economía y ganador del Premio Nacional de Economía 2005 en la especialidad de Contabilidad. (7)

El software incluye 10 módulos:

- Contabilidad General
- Costos y Procesos
- Finanzas
- Caja y Banco
- Inventarios
- Activos Fijos
- Facturación
- Nómina de Salarios
- Planificación
- Configuración
- Complementos.

Costos y Procesos: es un complemento del módulo Contabilidad General, además del registro contable de gastos, incluye dos actividades muy relevantes: el traspaso o distribución de los gastos indirectos hacia los centros de costo directo y el ajuste o costo, según los volúmenes de producción o servicios y sus destinos. Estas actividades están automatizadas.

Permite llevar un registro contable de gastos e incluye actividades para realizar el ajuste a los costos como son:

- El cálculo de los costos de producción y el ajuste correspondiente de forma automatizada.
- Emitir reportes de gastos a todos los niveles de análisis por subelementos y partidas.

Para el ajuste a los costos de producción se diseñan hojas de costos configurables por el usuario de acuerdo con la actividad económica que realiza.

Se caracteriza por ser una aplicación de escritorio. Concebido sobre una plataforma de trabajo Cliente-Servidor lo que permite además su instalación en red por las posibilidades que esta tecnología facilita para el trabajo en un entorno multiusuario. Implementado en Delphi. Trabaja sobre el sistema operativo Windows y presenta como soporte para bases de datos SQL Server 2000. (7)

Observación: a pesar de ser uno de los software más usados por el país, y tener incluido funcionalidades del componente Ajustes al Costo por Procesos este no tiene un desglose detallado de sus funcionalidades y está soportado sobre tecnología privativa.

1.5 Valoración del estado del arte

Después de haber realizado un estudio por los diferentes sistemas contables usados a nivel mundial y nacional, se evidenció la no existencia de un software que tenga implementada la funcionalidad descrita en el problema a resolver, es válido resaltar que las herramientas por las que están soportadas y las licencias de estos sistemas contables son difíciles de obtener, o simplemente no cumplen los requisitos establecidos por el país. Para el desarrollo de la solución se hizo necesario el trabajo con soluciones prácticas, eficientes y sin costos.

1.6 Modelo de desarrollo

En informática para desarrollar una aplicación o componente es recomendado guiarse por una metodología o modelo de desarrollo. El centro tiene como estándar un modelo de desarrollo de software para la realización de todos los proyectos, este es el propuesto por la Subdirección de Producción del CEIGE.

A continuación se recogen las principales características de dicho modelo:

Desarrollo basado en componentes: Lleva a alcanzar un mayor nivel de reutilización de software, aún en contextos distintos de aquellos para los que fue diseñado. Permite que las pruebas sean ejecutadas, probando cada uno antes de probar el conjunto completo de componentes ensamblados. Cuando existe un débil acoplamiento entre componentes, el

desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema, dado que un componente puede ser construido.

Otra característica presente en el modelo de desarrollo es que se encuentra sobre los modelos iterativos e incrementales los cuales disminuyen riesgos y nos ayudan a tener un mejor desarrollo de software ya que estos modelos se basan en la retroalimentación por lo que nos ayudan a tener una mejor arquitectura del software y son muy útiles cuando el usuario tiene más requerimientos. En este modelo aparecen las versiones. Gracias a las versiones se puede aumentar o modificar el software si se necesita.

El modelo incremental: este modelo mantiene la función anterior y aumenta otra, ya que puede ser que el primer incremento no hubiera tenido todos los requerimientos que necesitaba el proyecto.

El modelo iterativo: este modelo en cambio mejora cada versión es decir mejora la función que tiene la versión. (8)

Además está integrado al **modelo de calidad CMMI (Capability Maturity Model Integration) nivel 2**. CMMI (Modelo de Madurez de Capacidad Integrado) pertenece a la familia de modelos desarrollados por el SEI (Software Engineering Institute) para evaluar las capacidades de las organizaciones de ingeniería de sistemas, ingeniería de software, además del desarrollo integrado del producto y del proceso (9).

Como última característica que presenta el modelo de desarrollo de CEIGE es que está **basado en líneas de productos de software**. El centro se encuentra distribuido por varias líneas encargadas de en tareas específicas (Contabilidad, Finanzas, Logística).

A continuación se exponen las diferentes secuencias de actividades a seguir para la construcción y desarrollo de software bajo este modelo. Entre las cuales en el presente trabajo de diploma estarán enfocadas en Diseño, Implementación y Pruebas.

Figure 1. Ciclo de vida de proyectos del CEIGE.

1.7 Lenguajes de programación

PHP (Hypertext Pre-processor) 5.2.6

PHP es un lenguaje de programación interpretado de alto nivel, diseñado originalmente para la creación de páginas web dinámicas y donde el código es ejecutado en el servidor, que es lo que distingue a PHP de la tecnología Java Script.

Dentro de sus características se pueden mencionar:

- Libre y abierto.
- Multiplataforma.
- Soporte para varios servidores web.
- Fácil acceso a bases de datos.
- Abundante documentación y fácil aprendizaje.

Presenta una integración adecuada entre Apache-PHP-MySQL (10).

JavaScript

JavaScript es el lenguaje de programación web del lado del cliente más extendido. Es un lenguaje interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C. Al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de herencia, es más bien un lenguaje basado en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad. Todos los navegadores interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del DOM. JavaScript es un lenguaje con muchas posibilidades, utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes, orientados a objetos mucho más complejos. Con JavaScript se puede crear diferentes efectos e interactuar con los usuarios. JavaScript es soportado por la mayoría de los navegadores como Internet Explorer, Netscape, Opera, Mozilla Firefox, por solo mencionar algunos (11).

HTML (HyperText Markup Language)

Es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un script (por ejemplo JavaScript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

1.8 Sistema gestor de bases de datos

Un sistema gestor de bases de datos es una colección de programas que permiten a los usuarios crear y mantener una base de datos. Sistema software de propósito general que facilita los procesos de definición, construcción y manipulación de la base de datos para distintas aplicaciones. (12)

•Definición de la bases de datos: especificar tipos de datos, estructuras y restricciones.

•Construcción de la bases de datos: almacenar datos.

•Manipulación de la bases de datos: consultar, actualizar el diseño y generar informes

PostgreSQL 8.3

Es el gestor de bases de datos de código abierto más avanzado hoy en día, ofreciendo control de concurrencia multiversión, soportando casi toda la sintaxis SQL (incluyendo subconsultas, transacciones, tipos y funciones definidas por el usuario), contando también con un amplio conjunto de enlaces con lenguajes de programación (incluyendo C, C++, Java, Perl, Tcl y Python). (13)

1.9 Framework

El término framework se refiriere a una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que puede añadirse las últimas piezas para construir una aplicación concreta.

Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones (14).

Ext Js 4.1

Ext. 2.0 es un framework completo y extremadamente avanzado, nace como solución a tareas comunes, pero complejas. Este framework permitirá recordar la programación de entornos visuales, ya que es completamente basado en la Programación Orientada a Objetos (POO). Cada objeto contiene lo típico: propiedades, métodos, eventos entre otros. Ext basa toda su funcionalidad en JavaScript a través de librerías ya muy conocidas: jQuery y

Prototype/Script.aculo.us y un núcleo interno poderoso. En tiempo de ejecución, carga y crea todos los objetos HTML a través del uso intenso de DOM (Modelo de Objetos del Documento) (15).

Doctrine 1.2.1

Doctrine es un mapeador de objetos relacionales (ORM) que posee una poderosa capa de abstracción de base de datos. Una de sus características es la opción de escribir consultas de Base Dato en un objeto apropiado orientado al dialecto SQL (Structured Query Language) y que se le denomina Lenguaje de Consulta de Doctrine o DQL del inglés Doctrine Query Lenguaje, inspirado por el Lenguaje de Consulta de Hibérnate (HQL por sus siglas en inglés). Este proporciona a los desarrolladores una poderosa alternativa al SQL que mantiene la flexibilidad sin requerir duplicación de código innecesario. (16)

Zend 1.7

Zend Framework es un framework de código abierto para el desarrollo de aplicaciones y servicios web con PHP 5. Está implementado utilizando código 100% orientado a objetos. Los componentes tienen poco acoplamiento entre ellos lo cual le permite a los desarrolladores utilizarlos separadamente. Principalmente este framework ofrece una implementación MVC (Modelo Vista Controlador) robusta y de alto rendimiento, abstracción para interactuar con bases de datos. Además, posee otros componentes tales como Zend_Auth y Zend_Acl, los cuales proveen autenticación de usuarios y autorización de acceso a los recursos contra los almacenes de credenciales más comunes. (17)

1.10 Herramientas para el desarrollo

Entorno de desarrollo integrado (Integrated Development Environment (IDE))

Un entorno de desarrollo integrado o en inglés Integrated Development Environment (IDE) es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien, poder utilizarse para varios. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Los IDEs pueden ser aplicaciones por si solas o pueden ser

parte de aplicaciones existentes. Los IDEs proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Java, C#, Delphi, Visual Basic, Object Pascal, Velneo.

NetBeans 7.3

NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento. Sun Microsystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo su patrocinador principal. Hoy en día hay disponibles dos productos: el entorno de desarrollo integrado NetBeans y su Plataforma de mismo nombre. Ambos productos son de código abierto y gratuito para uso tanto comercial como no comercial.

NetBeans es un entorno de desarrollo, 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. Es un producto libre y gratuito sin restricciones de uso. Funciona sobre disimiles sistemas operativos como Open Solaris, Linux, Windows y Mac OS. (18).

Mozilla Firefox 13.0

Es un navegador de Internet, libre y de código abierto, desarrollado por la Corporación Mozilla y un gran número de voluntarios externos. Firefox es un navegador multiplataforma y está disponible en varias versiones de Microsoft Windows, Mac OS, GNU/Linux y algunos sistemas basados en Unix. Su código fuente es publicado bajo una triple licencia GPL/LGPL/MPL. (19)

1.11 Control de versiones

Es la gestión de los diversos cambios que se realizan sobre los elementos de algún producto. Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, así como las posibles especializaciones realizadas. (20)

Subversion 1.7

Subversión es un controlador de versiones empleado en la administración de archivos utilizados en el desarrollo de *software* que permite seguir los cambios de los archivos y directorios a través de copias y renombrados. Presenta integración con el servidor web Apache y existen varias interfaces de Subversión, ya sean programas individuales o interfaces que lo integran en entornos de desarrollo. (20)

1.12 Discrepancias de la tesis, Análisis y diseño del componente Ajuste al Costo por Procesos

A continuación se especificaran las principales deficiencias encontradas en el trabajo de diploma que lleva por título Análisis y Diseño del componente Ajuste al Costo por Procesos. Se evalúan los artefactos generados en dicha investigación, teniendo en cuenta los artefactos definidos en el Modelo de desarrollo CEIGE, una vez analizados estos, se encontraron los siguientes errores:

1.12.1 Modelado de procesos del negocio

En la fase Modelado de procesos del negocio se debieron generar los artefactos necesarios para comprender la ejecución del proceso de negocio en las entidades cubanas y así poder obtener un componente que responda al propósito inicialmente establecido. Sin embargo no se lograron los siguientes artefactos: Descripción de procesos de negocio, Mapa de procesos de negocio y Reglas de negocio; además el Glosario de términos se encuentra desactualizado al no contar con: cuentas de proceso, Centros de costo, Cuentas de inventario y Comprobantes de operaciones.

El contar con el Diagrama de proceso de negocio y el Modelo conceptual no fue suficientemente explicativo debido a que la secuencia de actividades, entradas y salidas, así como los involucrados no quedaron claros al no contar con el artefacto Descripción de Procesos de Negocio.

1.12.2 Requisitos

La fase Requisitos incluye un conjunto de artefactos que describen todas las interacciones que tendrán los usuarios con el software y que responden a los requisitos funcionales del sistema; además se describen los requisitos no funcionales.

En la descripción de los requisitos funcionales Gestionar Destino, Configurar Ajuste por Producto y Realizar Ajuste no queda claro el flujo de eventos; además no se explica el origen de las subcuentas necesarias, los productos y los estados del comprobante.

En el siguiente ejemplo se muestra la descripción del requisito Adicionar cuentas procesos, en el cual se define que para adicionar la cuenta proceso se deben llenar los campos Cuenta de proceso, Subcuenta de saldo inicio de año en Moneda Nacional (MN), Subcuenta de saldo inicio de año en Moneda libremente Convertible (MLC), Subcuenta registra saldo Gasto MN, Subcuenta registra saldo Gasto MLC pero en ningún momento se explica la procedencia de estas cuentas, lo que significa cada una, o la relación que tienen entre sí.

Ejemplo:

Especificación de requisito Adicionar cuentas de procesos.

Descripción textual del requisito

Tabla 1. Especificación de requisito adicionar cuentas de procesos

Precondiciones	El usuario se ha identificado y autenticado ante el sistema y tiene permisos para ejecutar esta acción. Se han seleccionado las opciones Configuración/Ajuste al Costo/General.
Flujo de eventos	
Flujo básico Adicionar cuentas de procesos	
	El usuario una vez que entra al sistema debe entrar en Costos y Proceso/Configuración/Ajuste al Costo/General.
	El usuario en el submenú Cuentas de proceso presiona el botón Adicionar.
	El sistema muestra una ventana para Configurar las cuentas de procesos.

	Cuenta de proceso. Subcuenta de saldo inicio de año en MN. Subcuenta de saldo inicio de año en MLC. Subcuenta registra saldo Gasto MN. Subcuenta registra saldo Gasto MLC.
	El sistema no permite escribir ningún tipo de datos en estos campos ya que solo deben seleccionarse.
	El usuario presiona el botón Aceptar.
	El sistema muestra un mensaje informando que la cuenta se ha configurado correctamente.
	Se presiona el botón Aceptar.
	Concluye el requisito.

1.12.3 Prototipo de interfaz

Los prototipos de interfaz de usuario son otros de los artefactos que se generan en la fase de Requisitos. En los realizados por la Ing. Doralis Sevilla Reyes es necesario hacer cambios debido a que rompen con el estándar de diseño de CEDRUX, como es el caso de la siguiente interfaz:

Ejemplo:

Se crean TabPanel dentro de otros.

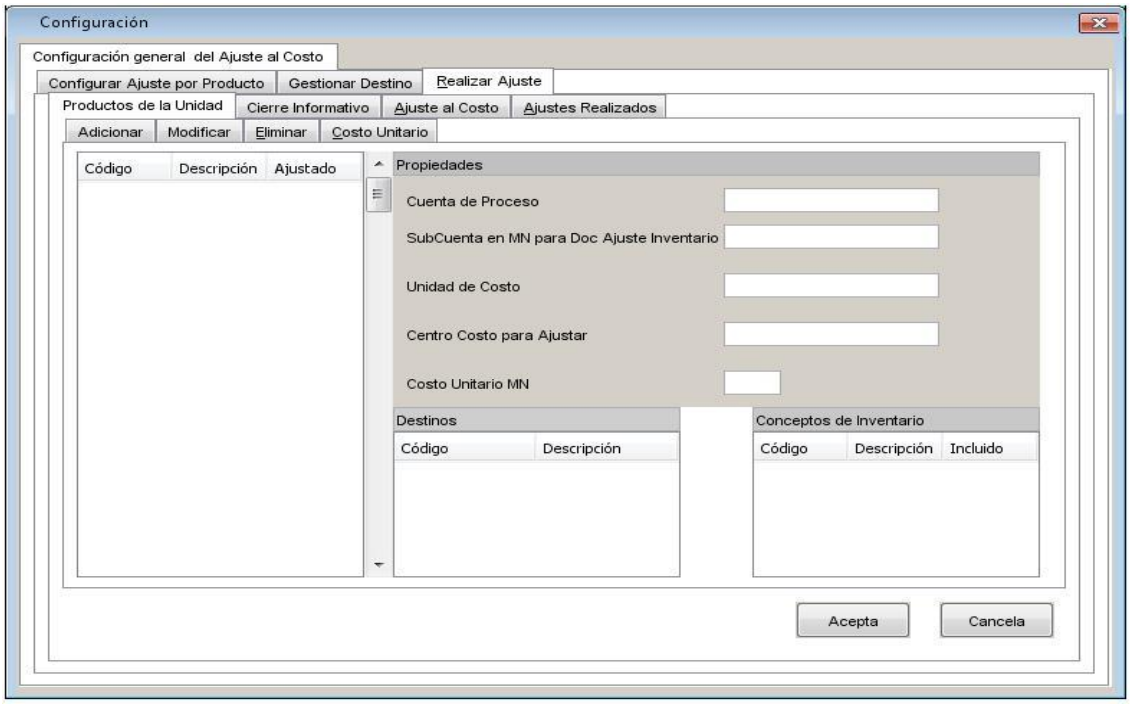


Figure 2. Prototipo de interfaz realizar ajuste.

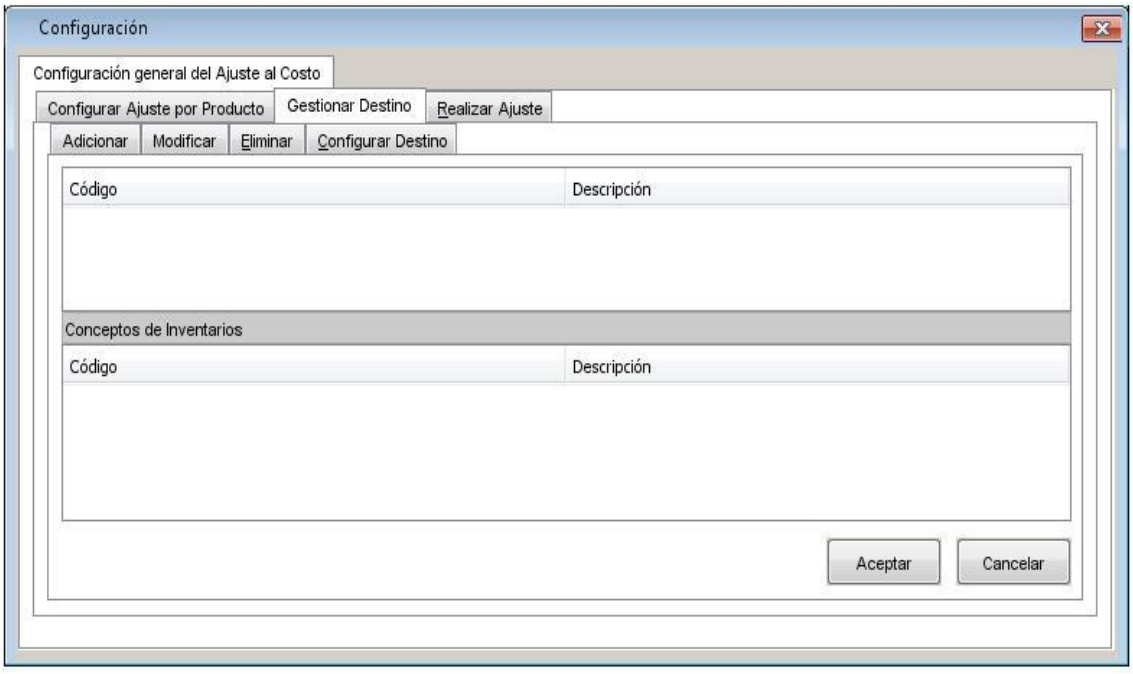


Figure 3. Prototipo de interfaz gestionar destino.

Análisis y diseño

Durante esta fase es modelado el sistema para que soporte todos los requisitos. Esto contribuye a una arquitectura sólida y estable.

1.12.4 Diagrama de clase de diseño

En los levantamientos de requisitos funcionales se tomaron 3 requisitos funcionales principales: Gestionar destino, Configurar ajuste por producto y Realizar ajuste. De ellos se realizó un Diagrama de clase de diseño por cada uno con una clase controladora por cada uno, quedando así 3 clases controladoras para 1 componente. Esto a la hora de programar es algo complicado de manera que es mucho más fácil tener una sola clase controladora relacionada con las tres interfaces pertenecientes a los tres requisitos funcionales principales, rompiendo con el estándar de CEDRUX el cual está programado por cada componente una clase controladora.

A continuación se muestran en la figura 4 y 5 los Diagramas de clase de diseño propuestos por la investigación realizada por la Ing. Doralis Sevilla Reyes.

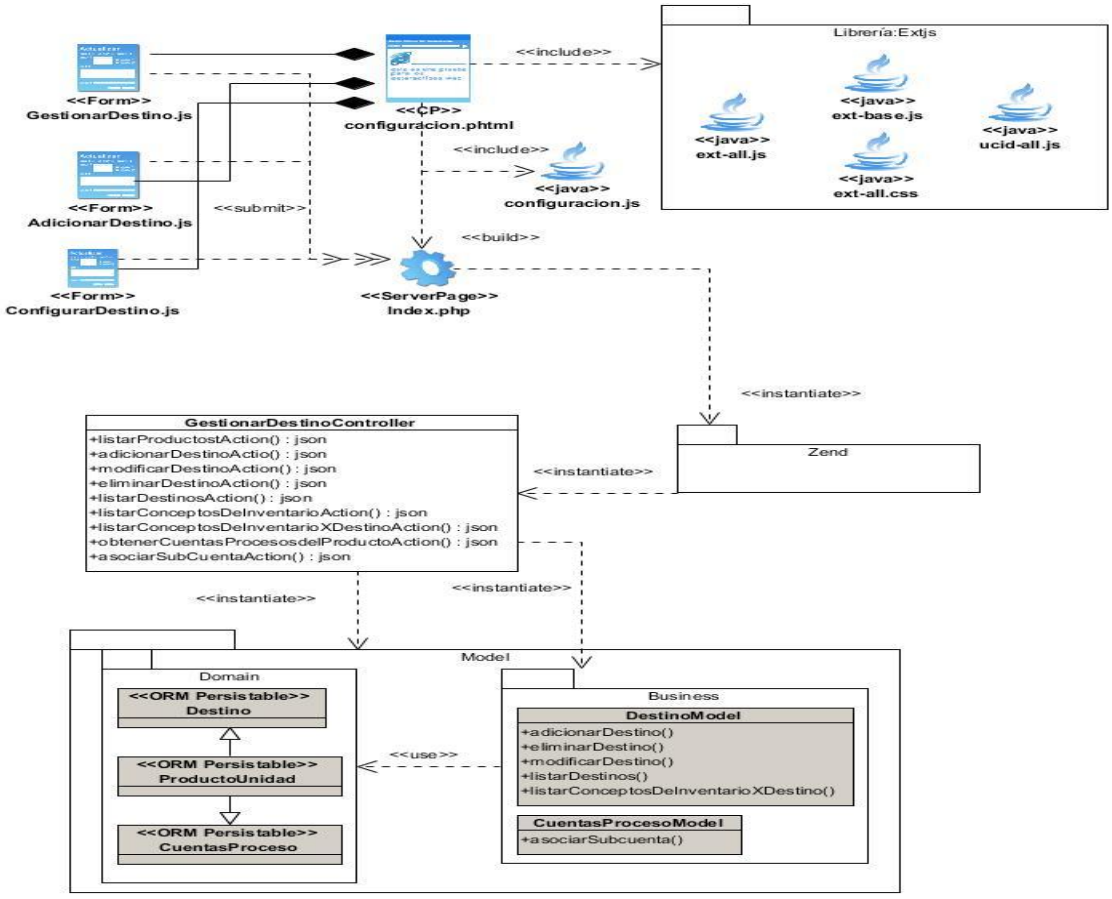


Figure 4. Diagrama de clases de diseño del proceso gestionar destino.

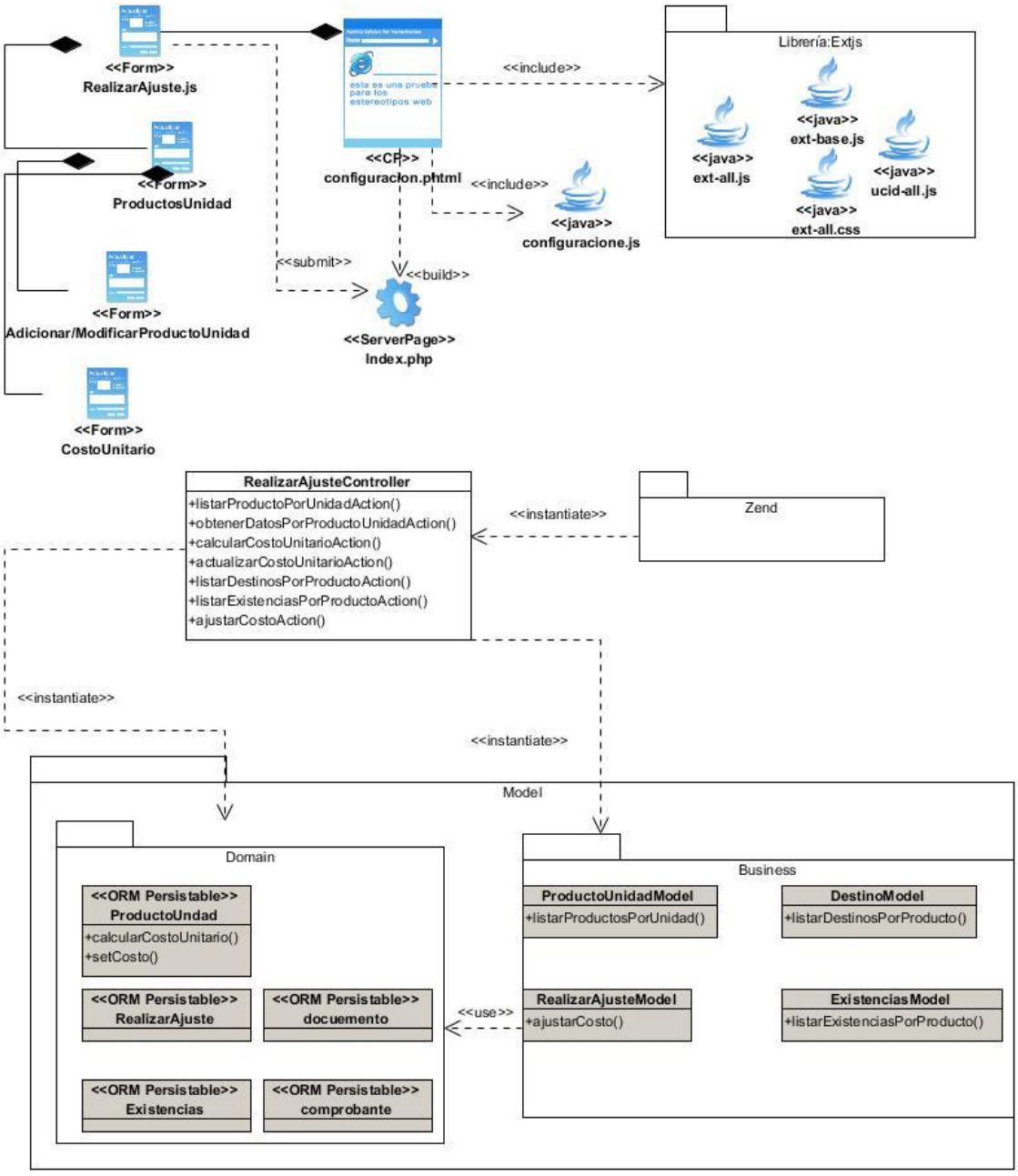


Figure 5. Diagrama de clases de diseño del proceso realizar ajuste.

1.13 Conclusiones del capítulo

Una vez finalizado el presente capítulo se pudo arribar a las siguientes conclusiones:

- El análisis de los sistemas estudiados no permitió obtener información relacionada con la realización del Ajuste al Costo por Procesos, de manera que no hubo aporte en cuanto a funcionalidades e interfaces.
- El ambiente de desarrollo integrado por el modelo utilizado y arquitectura seleccionada, además de los lenguajes, tecnologías y herramientas definidas, brindan la posibilidad de desarrollar una solución que apoye el cumplimiento de los principios de soberanía e independencia tecnológica y posibilite la realización del Ajuste al Costo por Procesos.
- Después de realizar un estudio al trabajo de diploma de la Ing. Doralis Sevilla Reyes se discrepó con diferentes artefactos generados, además que no fueron suficientes para entender el proceso de negocio del componente Ajuste al Costo por Procesos.

Capítulo 2: Diseño e implementación del componente

2.1 Introducción

En el presente capítulo se realiza una descripción de la solución propuesta, para proporcionar un mejor entendimiento del sistema. Se especifican los requisitos funcionales que debe cumplir el componente, contribuyendo directamente en que la solución propuesta satisfaga las necesidades del cliente y los no funcionales que se refieren a las propiedades emergentes de dicha solución. Como parte de la modelación del diseño de la solución se fundamentan los patrones de diseños empleados en su elaboración, el estilo arquitectónico usado y se crean los diagramas que según la notación UML describen las relaciones entre las clases del diseño, el modelo de datos y la comunicación con el resto de las funcionalidades.

Para el desarrollo del proceso de software de la aplicación se hizo necesario visitar a la especialista Rosa Elena Díaz González, Consultora de CANEC (La Consultoría de la Asociación Nacional de Economistas y Contadores de Cuba), donde se explicó el proceso de negocio del componente y con esto se generaron nuevos artefactos.

2.2 Propuesta de solución.

Para entender el proceso de negocio se hizo necesario visitar a una funcionaria (Rosa Elena Díaz González) de la Consultoría de la Asociación Nacional de Economistas de Cuba (CANEC) y una vez entendido este, se propuso una aplicación web que garantice el ajuste de los costos reales de producción, basándose en las ganancias y pérdidas que presenta la empresa. Cuenta con funcionalidades que se encargan de ajustar los costos de producción, generar comprobantes, entre otros.

2.3 Modelo de datos

Un modelo de datos es la descripción de una base de datos. Típicamente un modelo de datos permite describir las estructuras de datos de la base, su tipo, descripción y la forma en que se relacionan, restricciones de integridad entre otros, es factible pensar que un modelo de datos permite describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan esos elementos entre sí (21).

En la siguiente figura se muestra el modelo propuesto para el componente Ajuste al Costo por Procesos, para más información, ver anexo1.

Figure 6. Modelo de datos.

2.4 Descripción de las tablas generadas

A continuación se muestran las descripciones de las tablas del modelo de datos del sistema. Para más información ver anexo 2.

Tabla 2. Diccionario de datos tabla dat_confajusteprod

Nombre: tabla dat_confajusteprod		
Atributo	**Tipo**	**Descripción**

idconfajusteprod	numeric	Llave primaria de la llave
idestructura	varchar	
idfechasistema	date	
idfechaauditoria	date	
idejercicio	numeric	
idperiodo	numeric	Llave foránea
idmoneda	numeric	Llave foránea
idprodterminada	numeric	

2.5 Requisitos funcionales

Un requerimiento es una característica que el sistema debe tener o una restricción que satisfacer para ser aceptada por el cliente. Dentro de estos existen los requisitos funcionales y no funcionales. Los requisitos funcionales describen la iteración entre el sistema y su ambiente independiente de su implementación, el ambiente incluye al usuario y cualquier otro sistema externo que interactúe con el sistema. Los requisitos no funcionales describen aspectos del sistema que son visibles por el usuario que no incluyen una relación directa con el comportamiento funcional del sistema, estos incluyen restricciones como el tiempo de respuesta, la precisión, recursos consumidos, seguridad (22).

A continuación se exponen los diferentes requisitos funcionales.

Tabla 3. Requisitos funcionales

Requisito general Gestionar Destino	***Requisito general Configurar Ajuste por Producto***	***Requisito general Realizar Ajuste***
Adicionar destino Eliminar destino	Adicionar cuentas de procesos. Eliminar cuentas de procesos. Modificar cuentas de procesos.	Mostrar comprobante de ajuste. Mostrar documento de inventario.

	Adicionar producto. Eliminar producto.	Calcular Ajuste. Mostrar datos en existencia. Mostrar datos en destino. Mostrar ajustes realizados

2.6 Requisitos no funcionales.

Usabilidad

El sistema debe de ser fácil de usar para los usuarios que no estén familiarizados con el trabajo web. Además, se brinda la opción de ayuda de las principales funciones del sistema.

Confiabilidad

La información manejada por el sistema debe estar protegida ante el acceso no autorizado y la divulgación indebida.

Rendimiento

Para que la aplicación trabaje sin problemas deberá de tener conectados entre 200 y 300 usuarios, y las PC deberán de tener entre 512MB y 1G de RAM para ejecutar el navegador de manera rápida y sin problemas. Además el tiempo que demora en cargar las interfaces con su contenido se encuentra entre 10 y 15 segundos.

Portabilidad

El sistema debe funcionar tanto en Linux como en Windows XP o Windows 7.

Software

El componente deberá ser desarrollado sobre tecnología PHP 5. La base datos está implementada en PostgreSQL versión 8.4 y utiliza un servidor Web Apache 2.0.

El navegador será Moxilla Firefox en su versión 2.0 o superior.

Hardware

Se requiere tener como servidor una PC de 1 Gb de RAM como mínimo y microprocesador a 3.00GHz, un disco duro de 100GB.

El cliente deberá tener como requisitos mínimos en su PC un microprocesador de 1.40GHz, 512MB de RAM como mínimo.

2.7 Descripción de requisitos funcionales

A continuación se mostrará la descripción del requisito funcional Configurar Destino, para más información sobre los restantes requisitos funcionales, ver anexo 8.

Tabla 4. Especificación del requisito configurar destinos

Conceptos tratados	Conceptos	Atributos
	Destinos	Código, Descripción
	Conceptos de Inventario	Código, Descripción
Precondiciones	Precondiciones	Pre-requisito
	El usuario se ha identificado y autenticado ante el sistema.	Autenticar usuario.
Descripción	En la interfaz Gestionar destinos, se muestra la tabla Conceptos de destinos	
Validaciones	Esta configuración se hace solo una vez al principio, a no ser que haya algún valor que agregar o quitar en períodos posteriores.	
Post-condiciones	Quedan configurados los destinos para realizar los ajustes en cada período.	
Post-requisito	No procede.	

Tabla 5 Especificación del requisito adicionar destinos

Conceptos tratados	Conceptos	Atributos
	Destinos	Código, Descripción
	Conceptos de Inventario	Código, Descripción
Precondiciones	Precondiciones	Pre-requisito
	El usuario se ha identificado y autenticado ante el sistema.	Autenticar usuario.
Descripción	En la interfaz Gestionar destinos, se muestra la tabla conceptos de inventario que son los conceptos por los cuales se seleccionan los destinos para efectuar el ajuste, al seleccionar los conceptos que pasarán a ser Destinos, se activa el botón "+" estando desactivado el botón "-" y se acciona el mismo pasando a la tabla destinos para Ajuste los destinos configurados para el ajuste. Una vez realizada la selección se presiona el botón Guardar. En todo momento se puede cancelar la operación de configurar los destinos accionando el botón Cancelar y se restablecen los valores afectados.	
Validaciones	Esta configuración se hace solo una vez al principio, a no ser que haya algún valor que agregar o quitar en períodos posteriores.	
Post-condiciones	Quedan configurados los destinos para realizar los ajustes en cada período.	
Post-requisito	No procede.	

Tabla 6 Especificación del requisito eliminar destinos

Conceptos tratados	Conceptos	Atributos
	Destinos	Código, Descripción
	Conceptos de iInventario	Código, Descripción
Precondiciones	Precondiciones	Pre-requisito
	El usuario se ha identificado	Autenticar usuario.

	y autenticado ante el sistema.	
Descripción	Se selecciona el destino que desea eliminar, se presiona el botón Eliminar, y se muestra un mensaje de confirmación que el destino se ha eliminado.	
Validaciones	Esta configuración se hace solo una vez al principio, a no ser que haya algún valor que agregar o quitar en períodos posteriores.	
Post-condiciones	No procede.	
Post-requisito	No procede.	

2.8 Diagrama de clases del diseño

El diagrama de clases es el artefacto principal para la disciplina de análisis y diseño. Sirve además para visualizar las relaciones estructurales (herencia, agregación, asociación).

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces (las de java por ejemplo) en una aplicación normalmente contiene la siguiente información (23):

- Clases, asociaciones y atributos.
- Interfaz, con sus operaciones y constantes.
- Métodos.
- Información sobre los tipos de los atributos.
- Navegabilidad.

A continuación se muestra el Diagrama de diseño Ajuste al Costo propuesto para el componente Ajuste al Costo, para más información ver anexo 3.

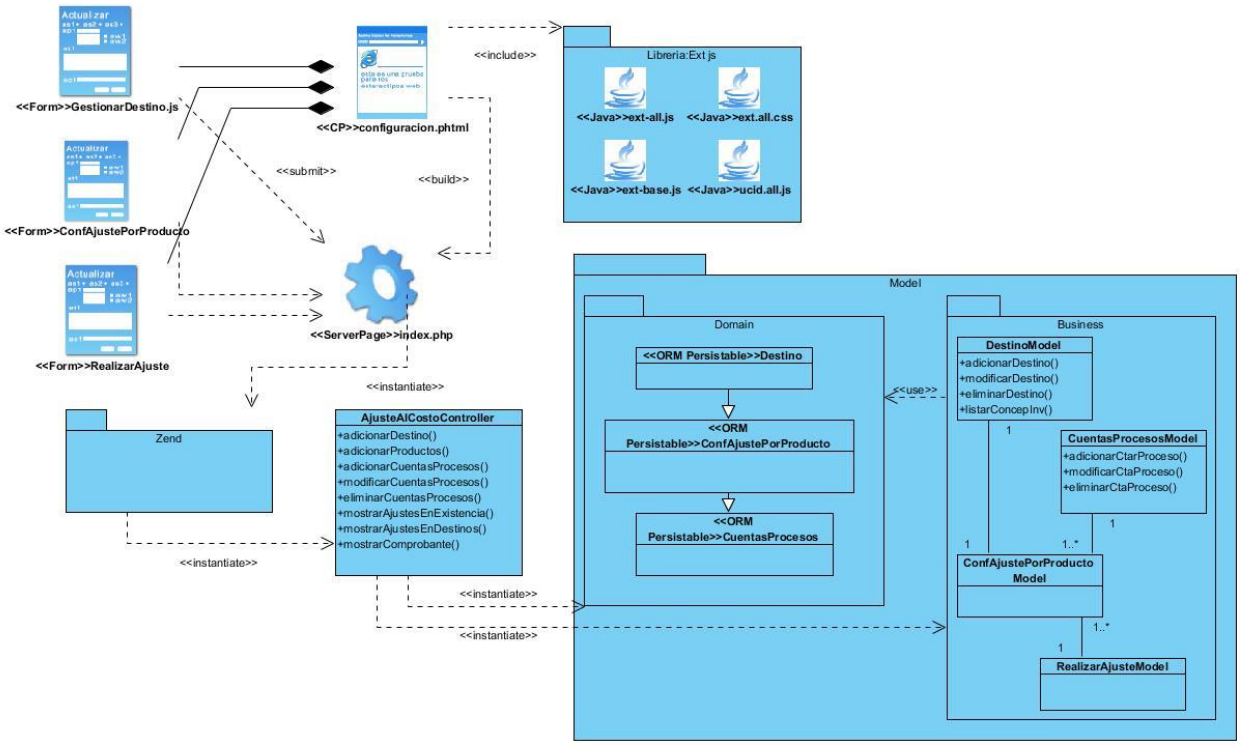


Figure 7. Diagrama de clase de diseño.

2.9 Diagrama de clases

El propósito de este diagrama es el de representar los objetos fundamentales del sistema, es decir, los que percibe el usuario y con los que espera tratar para completar su tarea en vez de objetos del sistema o de un modelo de programación. (23)

A continuación se muestra el diagrama de clases, para más información, ver anexo 4.

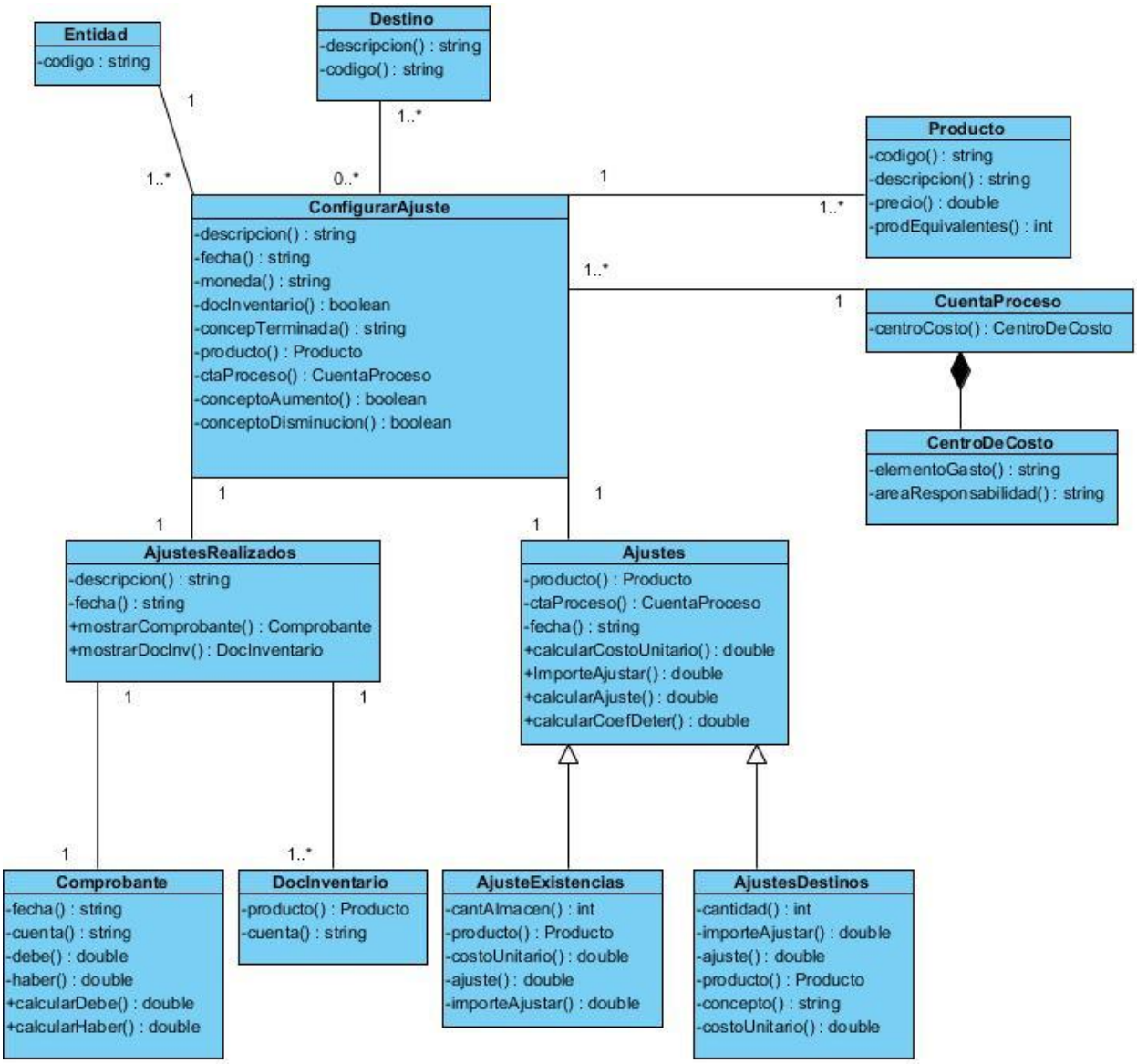


Figure 8. Diagrama de clase.

2.10 Diagrama de secuencia

Un diagrama de secuencia es una forma de diagrama de interacción que muestra los objetos como líneas de vida a lo largo de la página y con sus interacciones en el tiempo representadas como mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida destino. Los diagramas de secuencia son buenos para mostrar qué objetos se comunican con

qué otros objetos y qué mensajes disparan esas comunicaciones. Los diagramas de secuencia no están pensados para mostrar lógicas de procedimientos complejos. (24)

A continuación se muestra una imagen del diagrama de secuencia adicionar cuentas procesos, para más información ver anexos 7.

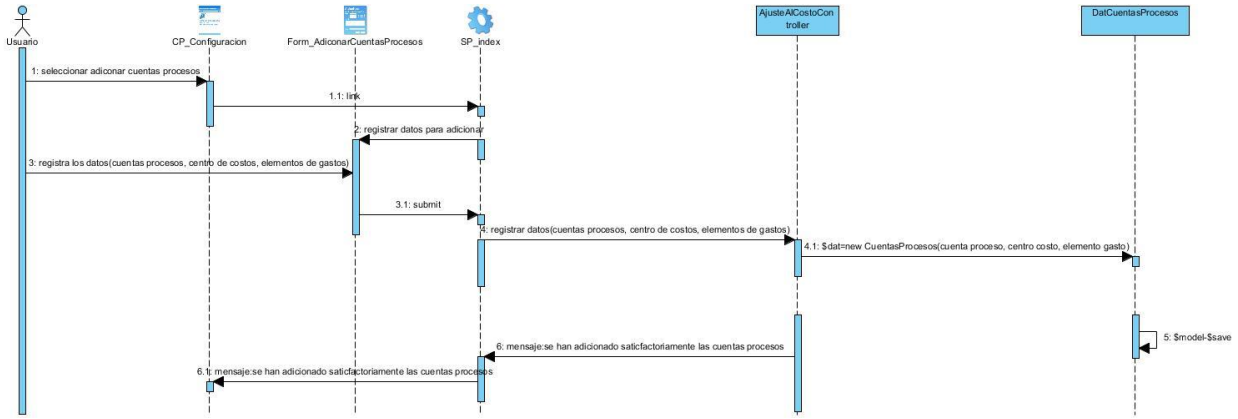


Figure 9. Diagrama de secuencia adicionar cuentas procesos.

2.11 Descripción del proceso Ajuste al Costo por Proceso

Descripción del proceso de negocio.

Primer 1: en la interfaz Gestionar destino, se seleccionan los conceptos de inventarios, a los cuales vas a destinar su producción, estos pueden ser de tipo Insumos, Ventas, Transferencias emitidas.

Paso 2: configurar el ajuste por producto, comienza con la selección del producto al cual se le va a realizar el ajuste. Luego se selecciona la cuenta proceso y la cuenta de gasto por la cual se va a realizar el ajuste.

Paso 3: se muestran las tablas con los ajustes realizados, con su documento de inventario y el comprobante.

2.12 Prototipos de interfaz del componente

A continuación se muestran las principales vistas del componente y una breve explicación de las mismas.

2.12.1 Gestionar destinos

Esta hoja como su nombre lo indica, es para definir los diferentes destinos que tiene la producción; ya sean: Insumos, Ventas, Transferencias emitidas. Una vez adicionado el destino, se selecciona el destino por el cual se realizara el ajuste y se presiona el botón Aceptar.

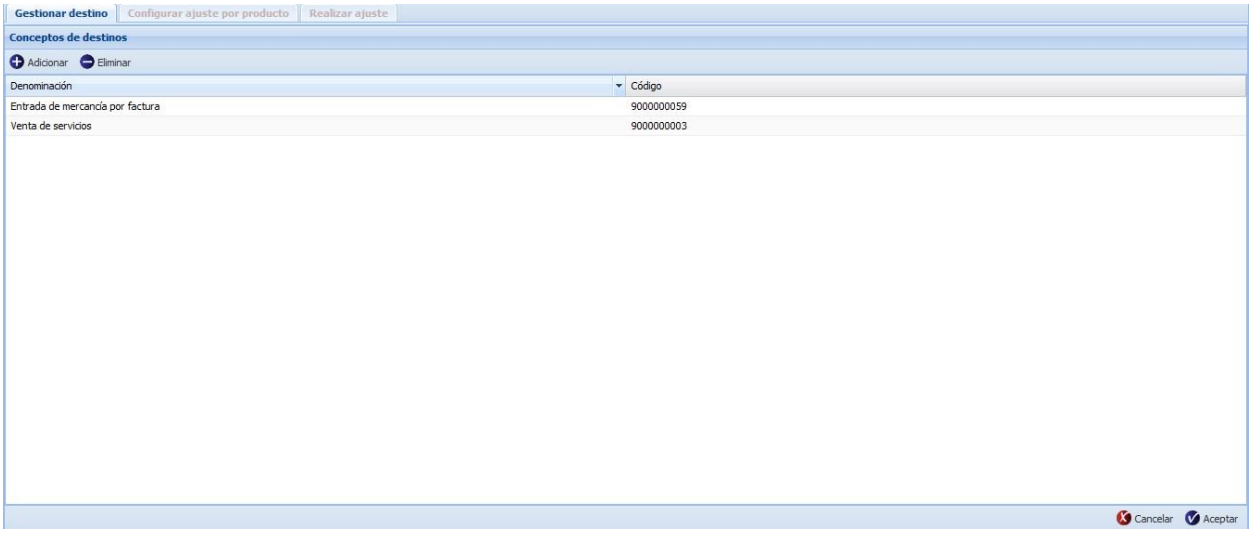


Figure 10. Prototipo de interfaz funcional gestionar destinos.

2.12.2 Configurar ajuste por producto

En la siguiente interfaz se configura el producto que se desea ajustar. Se comienza con la selección del producto, luego se escoge la cuenta proceso y la cuenta de gasto que se le asociará al producto. Los campos moneda y fecha se muestran deshabilitados ya que estos son asignados automáticamente por el sistema. Se presiona el botón Ajustar y se realizan los cálculos relacionados con el ajuste.

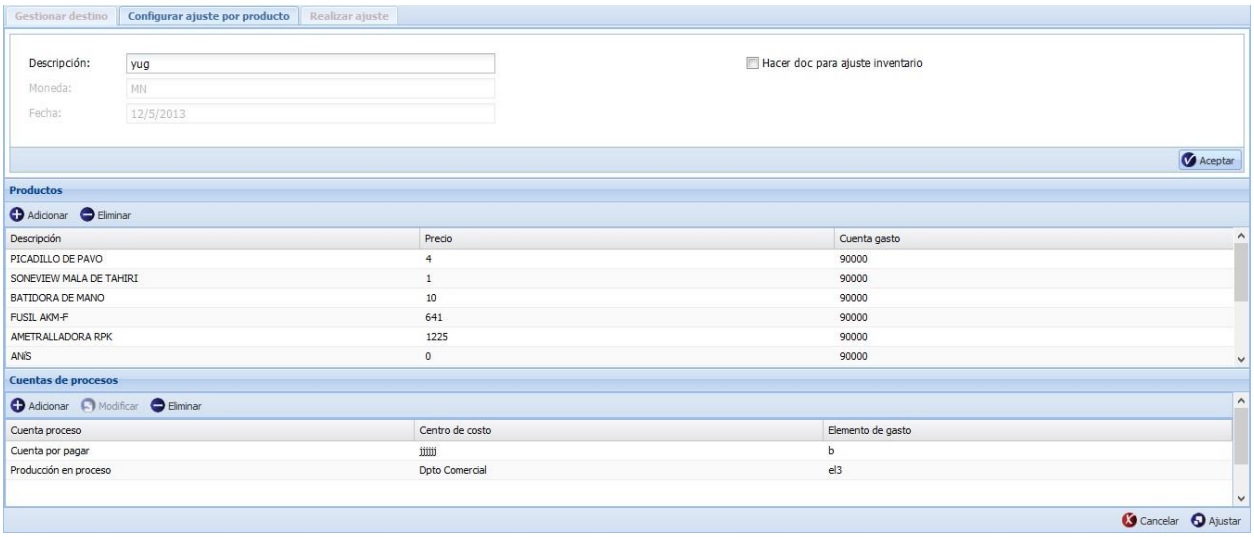


Figure 11. Prototipo de interfaz funcional configurar ajuste por producto.

2.12.3 Realizar ajuste.

La vista de Realizar Ajuste está compuesta por 2 páginas, Cierre informativo figura 12 y Ajustes realizados figura 13.

En la interfaz Cierre informativo se muestran los cálculos realizados, Costo unitario, Importe antes y después del ajuste, Ajuste. Al presionar el botón Aceptar se muestra la interfaz Ajustes realizados que tiene como objetivo mostrar todos los ajustes realizados, con la opción de que al seleccionar un ajuste realizado se muestre su documento de inventario.

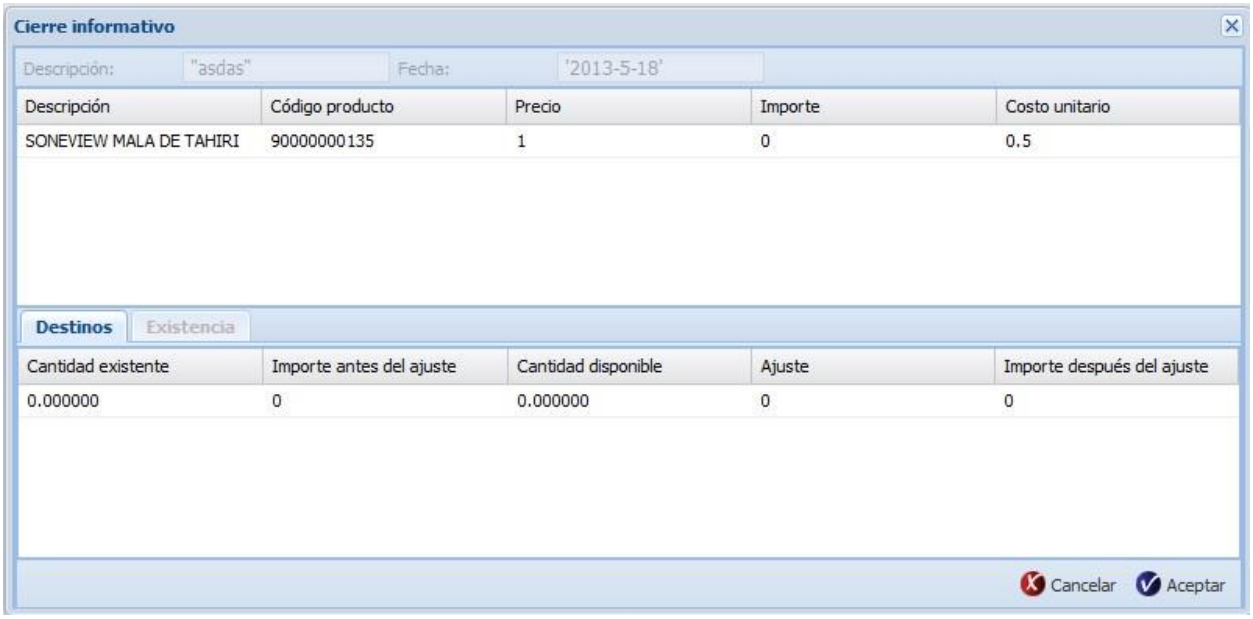


Figure 12. Prototipo de interfaz funcional cierre informativo.

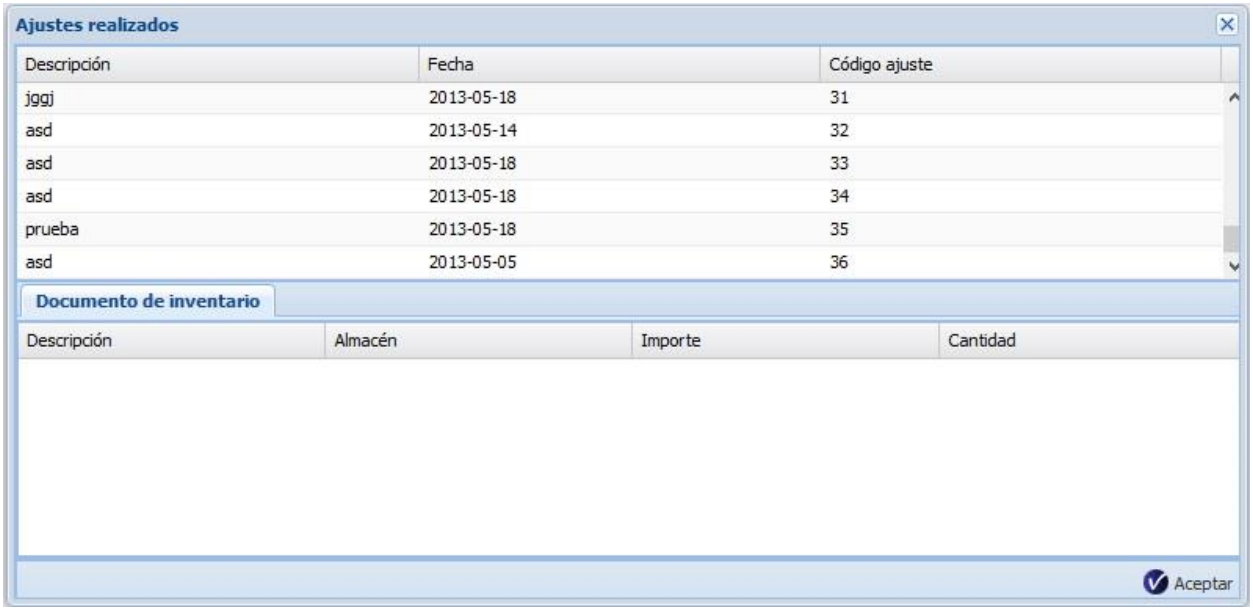


Figure 13. Prototipo de interfaz funcional ajustes realizados.

2.13 Diagrama de componente

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes, además de ayudar a entender mejor el modelo de implementación (25).

A continuación se explica la forma de interacción del componente Análisis Financiero con los componentes Estructura y Composición, Seguridad, Configuración y Estados Financieros, para un mayor entendimiento del diagrama Ver figura 11.

El componente Ajuste al Costo es el encargado de realizar los ajustes de costos al final de cada período. Este consume servicios de los componentes Contabilidad e Inventario. Por otra parte recibe la fecha, el período y el ejercicio contable del componente de Configuración General.

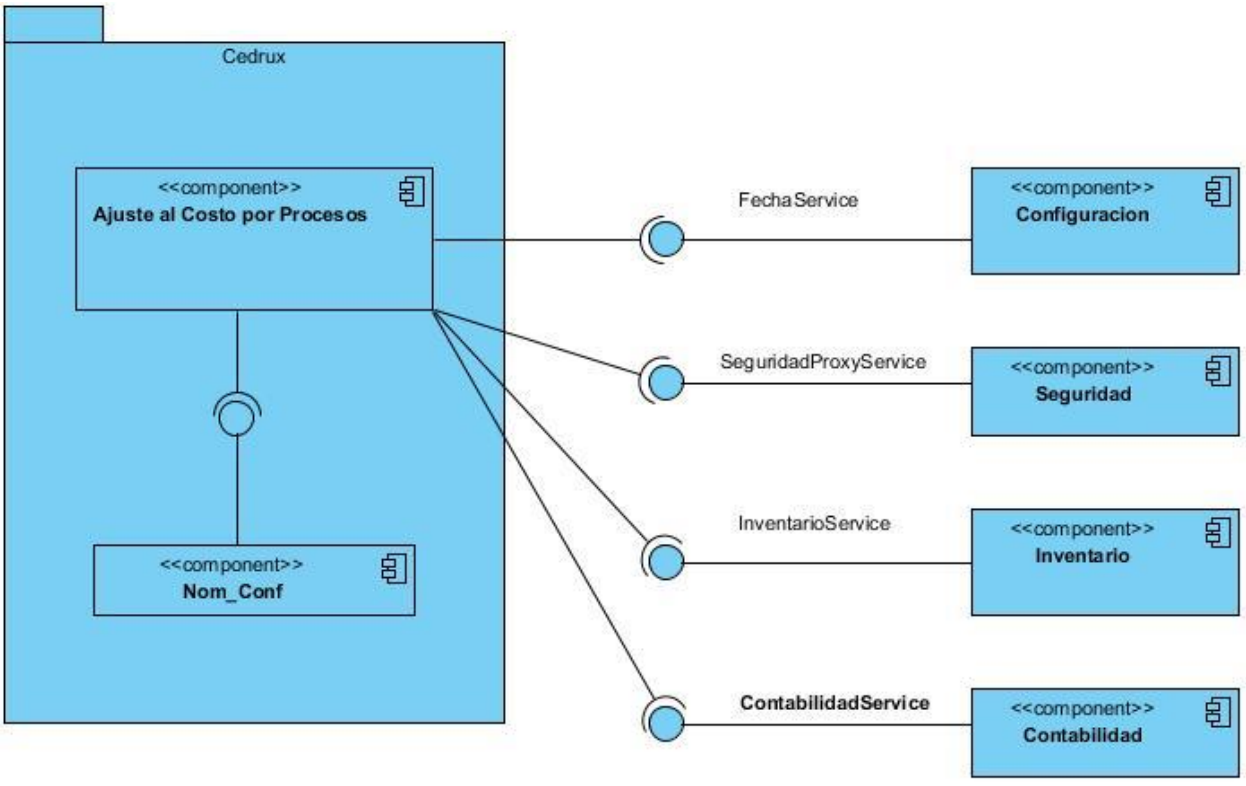


Figure 14. Diagrama de componentes

2.14 Patrones de diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. (26)

GRASP:

En diseño orientado a objetos, GRASP son patrones generales de software para asignación de responsabilidades, es el acrónimo de "GRASP (object-oriented design General Responsibility Assignment Software Patterns)". Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software (27).

Dentro de los patrones GRASP se encuentran:

Experto:

El componente cuenta con clases controladoras, modelos y de entidad que poseen funciones concretas de acuerdo con la información que gestionan. Un ejemplo es la clase DestinoModel la cual será la responsable de efectuar las operaciones que conciernen a las funciones: adicionarDestino, modificarDestino, eliminarDestino, listarDestino, listar conceptos de inventario por destino. Sobre este mismo principio se realiza el diseño de las restantes clases y sus funcionalidades.

```php
<?php
class DatDocinventarioModel extends ZendExt_Model
 {
   public function setUp()
    {
        parent::ZendExt_Model();
    }

   public function Insertar(DatDocinventario $DatDocinventario)
    {
            $DatDocinventario->save();
    }

   public function Actualizar(DatDocinventario $DatDocinventario)
    {
            $DatDocinventario->save();
    }

   public function Eliminar($DatDocinventario)
    {
            $DatDocinventario->delete();
    }
 }
?>
```

Figure 15. Clase DatDocinventarioModel.

Creador:

El patrón de creación hizo posible que el diseño pudiera soportar bajo acoplamiento. Las clases controladoras son responsables de crear el objeto de las modelos y estas a su vez de las entidades. Un ejemplo es la clase AjustePorProductoController crea el objeto de la clase DestinoModel y esta a su vez de las entidades Destino.

```
function eliminarProductosAction() {
    $datos = json_decode(stripslashes($this->_request->getPost('datos')));
    $model = new DatProductoajuste();
    $model->eliminarProducto($datos[0]->idproducto);
    echo json_encode(array("codigo" => "1"));
}
```

Figure 16. Método eliminarProductoAction.

Alta cohesión:

Este patrón fue utilizado en el diseño del componente de manera general; donde se agruparon las clases en dependencia de los requerimientos (GestionarDestino, RealizarAjuste) a los que se les debía dar respuesta, según la premisa de que cada clase debe implementar las operaciones que estén sobre la misma área funcional.

Bajo acoplamiento:

En el modelo de datos se definieron un conjunto de clases persistentes, entre las cuales se establecieron las relaciones necesarias de manera que fueran más independientes y reutilizables para reducir el impacto de los cambios y acrecentar la oportunidad de una mayor productividad.

Patrones GOF

Los patrones de diseños GOF (Gang of Four) son un grupo de patrones creados por autores del libro Design Patterns. Existen 23 patrones GOF, a continuación se explicaran los identificados en el trabajo de diploma (28).

Fachada:

Proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que hace que el subsistema sea más fácil de usar. Su aplicación se puede

observar en la relación que existe entre las clases controladoras y los servicios, lo que permite acceder a métodos que están implementados en otros componentes dentro y fuera del subsistema Costos y Procesos (29).

Cadena de Responsabilidades:

Está concebido que ante la ocurrencia de un error al realizarse una determinada consulta a la base de datos el mismo sea manejado por el modelo, creando una nueva excepción de tipo ZendExt_Exception. Dicha excepción debe ser propagada al controlador, el cual será el encargado de capturarla y enviarla a la vista ya traducida, esta última por su parte mostrará un mensaje al usuario en un lenguaje entendible notificando el error y sin especificar detalles del mismo. De esta manera, se distribuyen las responsabilidades entre las diferentes componentes, evidenciándose por lo tanto el empleo de este patrón (29).

Singleton:

Es un patrón creacional que garantiza que exista una instancia única para una clase y proporciona un punto de acceso global a ella. Los clientes acceden al ejemplar de Singleton únicamente a través del método instance de la clase Singleton. Un ejemplo de su utilización es cuando en la clase controladora, se adiciona un producto este tiene un campo que es el id de estructura el cual se encuentra global en todo el sistema (29).

2.15 Estilo arquitectónico Modelo-Vista-Controlador (MVC).

MVC es un estilo arquitectónico usado principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de los conceptos para que el desarrollo esté estructurado de una mejor manera. Facilita la programación en diferentes capas de forma paralela e independiente. MVC sugiere la separación del software en 3 estratos (30).

Modelo: Esta capa es la representación específica de la información con la cual el sistema opera. Se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas.

Vista: Esta capa presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

Controlador: Este responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y notifica a la vista.

El Modelo es el responsable de:

- Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.
- Definir las reglas de negocio.
- Si se encuentra ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

El Controlador se encarga de:

- Recibir los eventos de entrada.
- Contiene reglas de gestión de eventos. Estas acciones pueden suponer peticiones al modelo o a las vistas.

Las Vistas son responsables de:

- Recibir datos del modelo y mostrarlos al usuario.
- Tienen un registro de su controlador asociado (normalmente porque además lo instancia).
- Pueden dar el servicio de actualización, para que sea invocado por el controlador o por el modelo

Figure 17. Modelo, Vista, Controlador.

La unión entre la capa de presentación y la capa de negocio conocida en el paradigma de la programación por capas, representaría la integración entre vista y su correspondiente controlador de eventos. MVC no pretende discriminar entre la capa de negocio y la capa de presentación pero si pretende separar la capa visual gráfica de su correspondiente programación y modelo, encargado del acceso a datos, algo que mejora el desarrollo y mantenimiento de la vista y el controlador en paralelo, ya que ambos cumplen ciclos de vida muy distintos entre sí.

2.16 Estructura del marco de trabajo CEDRUX

A continuación se presenta la estructura del marco de trabajo CEDRUX, mostrando cómo será organizada la implementación del sistema a desarrollar, de manera que facilite la organización y claridad durante el desarrollo.

Contenido dentro de la carpeta apps:

En la carpeta denominada apps se almacenan los controladores y el modelo de cada una de las funcionalidades a desarrollar.

Figure 18. Contenido de la carpeta de aplicación apps.

- ✓ **comun:** contiene la carpeta recursos y dentro de esta una denominada xml. Esta última incluye a los ficheros: ioc, exception que serán explicados a continuación.
- ✓ **ioc:** es donde se publican los servicios que brinda cada uno de los componentes en cuanto a nombre de clases, funciones y tipo de resultado.
- ✓ **validation:** chequea las precondiciones antes de ejecutar una determinada función en el servidor según el tipo de parámetros, la acción y el usuario que la realice.
- ✓ **exception:** se define el tipo, idioma y la descripción del mensaje que va a ser mostrado cuando se lance una excepción en el servidor.

El componente Ajuste al Costo por Proceso va a contener un conjunto de carpetas que serán especificadas a continuación:

Figure 19. Contenido del componente Ajuste al Costo dentro de la carpeta apps.

En la carpeta **controllers** se encontrarán las clases controladoras encargadas de gestionar las funcionalidades del sistema.

La carpeta **models** se estructura de la siguiente forma:

Figure 20. Contenido de la carpeta models.

Esta carpeta contiene dos carpetas las cuales a su vez agrupan clases y otras carpetas, las mismas serán explicadas a continuación:

- ✓ **bussines:** contiene las clases necesarias para acceder a los datos que persisten en la base de datos.
- ✓ **domain:** contiene las clases generadas por el ORM Doctrine Generator a partir de cada una de las tablas existentes en la base de datos.

Cada una de estas clases mencionadas anteriormente heredará de una clase generada igualmente por el Doctrine Generator las cuales se ubican en otra carpeta dentro de esta llamado generated.

La carpeta **services** incluye todas las clases y funcionalidades de los servicios que va a ofrecer el sistema. Para solicitar un servicio que está en otro dominio, se accede a través de la carpeta services, quien analizará la solicitud e irá a la clase que tiene dicha funcionalidad y la devolverá a services, que a su vez se la entregará al dominio solicitante.

La carpeta **views** contiene las carpetas idioma y scripts, que se encargan de contener el idioma en que se va a mostrar la aplicación y las páginas clientes respectivamente.

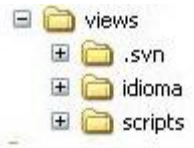


Figure 21. Contenido de la carpeta views.

Contenido dentro de la carpeta web:

Al mismo nivel de la carpeta apps mencionada anteriormente debe existir además una carpeta que contiene las vistas de los subsistemas y componentes. Dicha carpeta se denomina web.

Figure 22. Carpeta de diseño correspondiente al componente Ajuste al Costo.

- ✓ **index:** este fichero incluye la dirección del archivo de configuración y a través de este inicializa la aplicación para que se carguen en la misma un conjunto de componentes necesarios para su funcionamiento. Su código permanece igual para todos los componentes.

La carpeta **views** contendrá los js que se explican a continuación.

- ✓ **js:** comprenderá las clases Java Script necesarias para que el usuario interactúe con el sistema y obtenga los resultados necesarios. Está compuesta por carpetas con los nombres de las funcionalidades del componente.

2.17 Conclusiones del capítulo

Entre los principales resultados alcanzados durante el desarrollo del capítulo se resaltan los siguientes:

- A partir del análisis y el diseño, se logra fundamentar las bases para la implementación del componente.
- El diagrama de clases del diseño permitió conocer la estructura y las relaciones entre las clases que se manejan en el componente, posibilitando a su vez, el mantenimiento y entendimiento de estas, mediante los patrones de diseño definidos en la estructura arquitectónica de la solución.

Capítulo 3: Validación del componente

3.1 Introducción al capítulo

En el presente capitulo se realizará un estudio de los diferentes métodos y tipos de pruebas a llevar a cabo, con el objetivo de garantizar la calidad del sistema y el total cumplimiento de los requisitos establecidos con el cliente.

3.2 Validación del diseño propuesto

Una métrica es un instrumento que cuantifica un criterio y persigue comprender mejor la calidad del producto, estimar la efectividad del proceso y mejorar la calidad del trabajo realizado al nivel del proyecto. Para la evaluación de la calidad del diseño propuesto para el sistema se hizo un estudio de las métricas básicas inspiradas en la calidad del diseño orientado a objeto, en el mismo se abarcan atributos de calidad que permiten medir la calidad del diseño propuesto. Dentro de estos se encuentran:

- **Responsabilidad**: consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
- **Complejidad de implementación**: consiste en el grado de dificultad que tiene implementar un diseño de clases determinado.
- **Reutilización**: consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.
- **Acoplamiento**: consiste en el grado de dependencia o interconexión de una clase o estructura de clase con otras, está muy ligada a la característica de reutilización.
- **Complejidad del mantenimiento**: consiste en el grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error de un diseño de software. Puede influir indirecta, pero fuertemente en los costos y la planificación del proyecto.
- Cantidad de pruebas: consiste en el número o el grado de esfuerzo para realizar las pruebas de calidad del producto diseñado.

3.3 Tamaño operacional de las clases

El tamaño general de una clase se ajusta a realización de un cálculo de sus atributos u operaciones totales. El total de operaciones (operaciones tanto heredadas como privadas de la instancia), que se encapsulan dentro de la clase. El número de atributos (atributos tanto heredados como privados de la instancia), encapsulados por la clase. Los valores grandes para la métrica TOC indican que la clase debe tener bastante responsabilidad, lo que propicia un bajo nivel de reutilización de la clase y complicará la implementación y las pruebas. En general, operaciones y atributos heredados o públicos deben ser ponderados con mayor importancia cuando se determina el tamaño de clase, pues las operaciones y atributos privados, permiten la especialización. También se pueden calcular los promedios para el número de atributos y operaciones de cada clase. Cuanto más pequeño sea el valor promedio para el tamaño, será más viable para que las clases dentro del sistema puedan reutilizarse. Durante la realización de este trabajo se decidió aplicar la métrica para el tamaño de clases en función de sus operaciones. Luego de aplicar dicha métrica a las clases de la solución la misma arrojó los siguientes resultados:

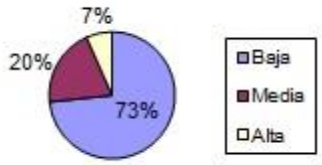


Figure 23. Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Responsabilidad.

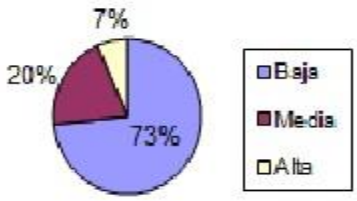


Figure 24. Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Complejidad de implementación.

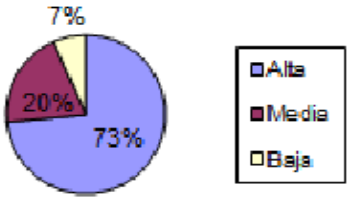


Figure 25. Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Reutilización.

3.4 Relaciones entre clases

El resultado de la aplicación de la métrica RC está dado por el número de relaciones de uso de una clase con otras. La aplicación de dicha métrica permite evaluar atributos como el Acoplamiento, la Complejidad de mantenimiento, la Reutilización y la Cantidad de pruebas. De forma tal que mientras mayor sea las relaciones de uso entre clases mayor será el Acoplamiento, la Complejidad de Mantenimiento y la Cantidad de pruebas, mientras que su reutilización disminuye. Después de aplicar dicha métrica se obtuvieron los siguientes resultados:

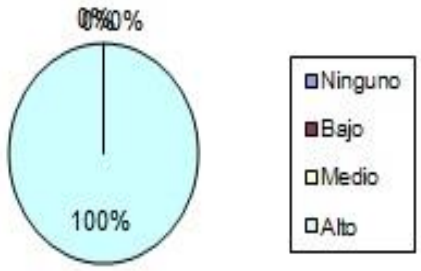


Figure 26. Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Acoplamiento.

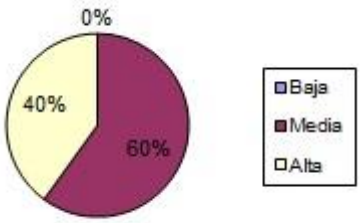


Figure 27. Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Complejidad de Mantenimiento.

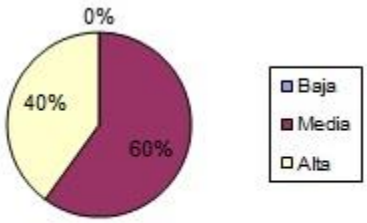


Figure 28. Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Cantidad de Pruebas.

3.5 Pruebas de software

Según la IEEE una prueba es: “Una actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto”. El objetivo fundamental de las mismas es medir el grado en que el software cumple con los requisitos. (31)

Existen dos métodos de pruebas para detectar diferentes tipos de errores. Estos son: pruebas de Caja Negra y Caja Blanca. Según lo definido por Pressman, las pruebas de caja negra se llevan a cabo sobre la interfaz del software. Se trata de demostrar que las funciones del software son operativas, que las entradas se manejan de forma adecuada y que se produce el resultado esperado. Las pruebas de caja blanca se centran en la estructura lógica interna del software. Se basan en un examen detallado de los procedimientos y caminos lógicos del sistema. (32)

Para el presente trabajo fue aplicado el método de caja blanca, descrito a continuación.

3.6 Pruebas de caja blanca

La prueba de caja blanca también conocida como prueba de caja de cristal, se basa en el diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivarlos. Para la solución desarrollada la prueba de Caja Blanca aplicada fue la del camino básico. Esta técnica permite obtener una medida de la complejidad lógica de un diseño y usar esta medida como guía para la definición de un conjunto básico de caminos de ejecución.

A continuación se muestra el código del método ObtenerProductoBaseDato(), al cual se le aplico la prueba de caja blanca.

```
function obtenerProductoBaseDatoAction() {

    $obj = new DatProductoajuste();//1
    $res = $obj->ObtenerProducto();//1
    $idestructura = $this->global->Estructura->idestructura;//1
    $arrayProductos = $this->integrator->logistica->obtenerProductoDadoIdEstructura2($idestructura);//1
    for ($i = 0; $i < count($res); $i++) {//2
        for ($j = 0; $j < count($arrayProductos); $j++) {//3
            if ($arrayProductos[$j]['idproducto'] == $res[$i]['idproducto']) {//4
                $res[$i]['descripcion'] = $arrayProductos[$j]['descripcion'];//5

            }//6
        }//7
    }//8
    echo json_encode($res);//9
    return;//10
}//11
```

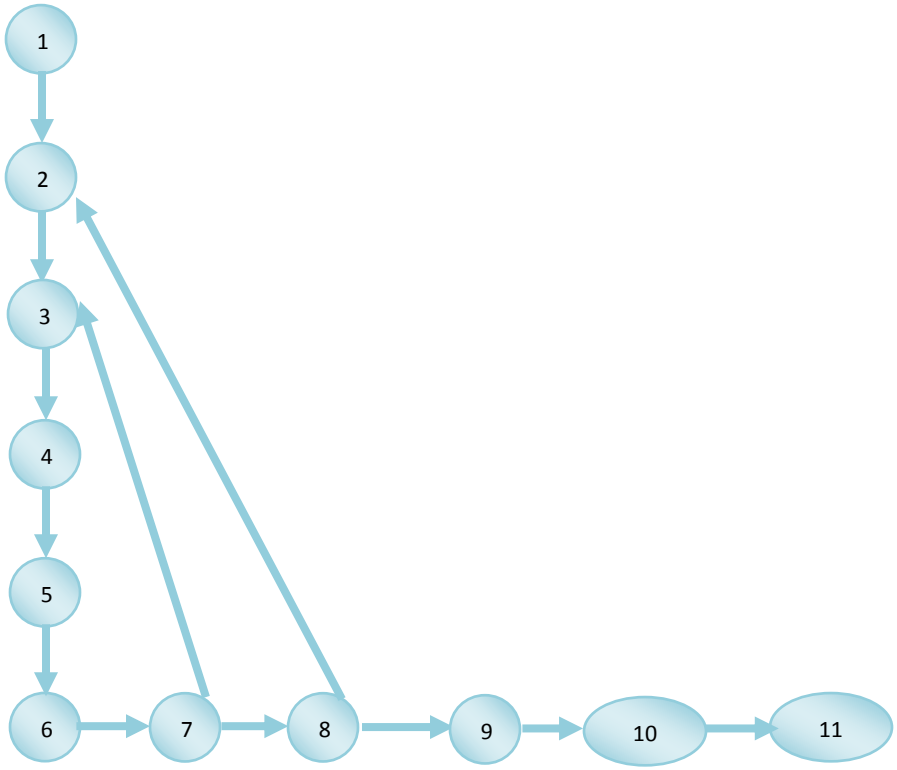
Figure 29. Método ObtenerProductoBaseDatoAction.

Luego del paso anterior, es necesario representar el grafo de flujo asociado al código antes presentado a través de nodos, aristas y regiones, en ese caso:

Nodo: Círculos representados en el grafo de flujo, el cual representa una o más secuencias del procedimiento, un nodo en sí puede representar un proceso, una secuencia de procesos o una sentencia de decisión. Los nodos que no están asociados se utilizan al inicio y final del grafo.

Aristas: Saetas a través de las cuales se unen los Nodos y constituyen el flujo de control del procedimiento.

Regiones: Las regiones son las áreas delimitadas por las aristas y nodos.



Para calcular la complejidad ciclomática del método ObtenerProductoBaseDato fueron utilizadas estas tres formas para lograr una amplia verificación de los resultados.

Las fórmulas para realizar dicho cáculo son:

1. V (G) = (A - N) + 2

V (G) = (12-11) +2

V (G) = 3

Siendo A la cantidad total de aristas del grafo y N la cantidad de nodos.

2. V (G) = P +1

V (G) = 2+1

V (G) = 3

Siendo P la cantidad de nodos predicado (son aquellos de los cuales parten dos o más aristas)

3. V (G) = R

V (G) = 3

Siendo R la cantidad de regiones que posee el grafo.

En cada una de las fórmulas V (G) representa el valor del cálculo. Según los resultados obtenidos en cada uno de estos cálculos se puede concluir que la complejidad ciclomática del código analizado es 3, determinándose a su vez que existen tres caminos posibles por donde puede circular el flujo y que esta misma cantidad representa el límite superior de casos de prueba que se le pueden aplicar a dicho código.

A continuación se muestran los caminos básicos por donde puede circular el flujo.

Camino básico # 1: 1-2-3-4-5-6-7-8-9-10-11

Camino básico # 2: 1-2-3-4-5-6-7-3-4-5-6-7-8-9-10-11

Camino básico # 3:1-2-3-4-5-6-7-8-2-3-4-5-6-7-8-9-10-11

3.7 Pruebas de caja negra o funcional

Conocido también como Prueba de Caja Opaca o Inducida por los Datos, este tipo de pruebas se limita a brindar solo datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo el módulo internamente, es decir, solo trabaja sobre su interfaz externa

Las pruebas de caja negra buscan encontrar errores en cinco categorías (33).

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en acceso a base de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

Dentro de la prueba de caja negra se incluyen las Técnicas de Pruebas que serán descritas a continuación:

Partición de Equivalencia: Divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.

Análisis de Valores Límites: Prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.

Grafos de Causa-Efecto: Permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

A continuación se aplica la prueba de partición de equivalencia como parte de la realización de la prueba de caja negra sobre la interfaz que responde al requisito funcional “adicionar producto”.

La Partición de Equivalencia divide el dominio de entrada de un programa en un número finito de variables de equivalencia. Las variables de equivalencia representan un conjunto de estados válidos y no válidos para las condiciones de entrada de un programa. Se definen dos tipos de variables de equivalencia, las válidas, que representan entradas válidas al programa, y las no válidas, que representan valores de entrada erróneos, aunque pueden existir valores no relevantes a los que no sea necesario proporcionar un valor real de dato.

3.8 Diseño de caso de prueba adicionar producto.

3.8.1 Condiciones de ejecución

- Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
- Se debe seleccionar el subsistema de **Costo y Procesos**.
- Se debe seleccionar la opción **Ajuste al Costo/ Ajuste al Costo**

Tabla 5. Requisitos a probar

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Adicionar Productos.	El sistema debe mostrar un listado	EP 1.1: Adicionar producto.	El usuario dentro del panel de productos,

	de todos los productos que han sido adicionados.		presiona el botón de adicionar El sistema muestra un listado con todos los productos existentes. El usuario selecciona los productos que desea ajustar. Se presiona el botón de aceptar mostrándose el mensaje Los productos se han adicionado correctamente.
		EP 1.2: Cancelar	Se presiona el botón Adicionar. Se seleccionan o no los productos. Se presiona el botón Cancelar.

3.8.3 Descripción de variable

No	Nombre de campo	Tipo	Válido	Inválido	Inválido	Inválido	Inválido
N/	N/A	N/A	N/A	N/A	N/A	N/A	N/A

3.8.4 Juegos de datos a probar

Id del escenario	Escenario	Respuesta del sistema	Resultado de la prueba
EP 1.1	Adicionar productos.	El sistema muestra un listado de los productos existentes.	

Resultados de las pruebas realizadas

Después de aplicados los métodos de prueba Caja Negra y Caja Blanca, se destaca que los resultados obtenidos hasta el momento han sido satisfactorios. Se realizaron pruebas de calidad interna del centro donde las 15 no conformidades detectadas fueron debidamente resueltas, quedando de esta forma la solución liberada y emitiéndose el Acta de liberación, ver anexo 6.

3.9 Conclusiones del capítulo

Al finalizar el presente capítulo vale destacar que:

- Con la aplicación de las métricas destinadas a validar el diseño se pudo constatar que este posee una buena calidad, posibilitando una futura reutilización del código y de las clases en diseños futuros.
- Se obtienen importantes resultados sobre el comportamiento del componente aplicando las pruebas de Caja negra y Caja blanca. Estas permitieron comprobar los requisitos funcionales definidos para el componente a partir de los diseños de casos de pruebas y camino básico, evaluándose la calidad del mismo para lograr satisfacer las necesidades del cliente.

Conclusiones generales

Una vez terminado el presente Trabajo de Diploma se puede concluir que se desarrollaron todas las tareas a fin de cumplir los objetivos propuestos, resaltando que:

- El análisis de los sistemas estudiados no permitió obtener información relacionada con la realización del Ajuste al Costo por Procesos, de manera que no hubo aporte en cuanto a funcionalidades e interfaces.
- El desarrollo de la solución posibilitó la realización del componente Ajuste al Costo por Procesos, sirviendo de apoyo al proceso de toma de decisiones en las entidades empresariales y presupuestadas.
- Las pruebas de Caja negra y Caja blanca permitieron encontrar y corregir los errores no detectados durante la implementación posibilitando cumplir con las especificaciones requeridas y la validación del componente implementado.

Por tanto, al finalizar el presente trabajo, queda informatizada la realización del Ajuste al Costo por Procesos, en correspondencia con los requisitos definidos, alcanzándose de esta manera los objetivos propuestos para dar solución al problema planteado. El desarrollo de esta aplicación tiene gran importancia, porque contribuye al mejoramiento del trabajo de los contadores, aportando un poco más a las entidades empresariales y presupuestadas.

Referencias Bibliográficas

1. **Quevedo Campins VD, Martínez Chong M, González Gutierrez LA, Fernández A.** .. [En línea] 29 de noviembre de 2010. [Citado el: 1 de diciembre de 2012.] http://ccia.cujae.edu.cu/index.php/siia/siia2010/paper/download/772/50.

2. Ing. Sevilla Reyes Doralis.. [En línea] [Citado el: 1 de diciembre de 2012.] http://www.despachomata.com.

3. Costos. ECURED.. [En línea] [Citado el: 1 de diciembre de 20012.] http://www.ecured.cu/index.php/Costos.

4. **ECURED, Proceso.** .. [En línea] [Citado el: 1 de diciembre de 20012.] http://www.ecured.cu/index.php/Proceso.

5. .. [En línea] [Citado el: 6 de noviembre de 2012.] http://www.sap.com/latinamerica/solutions/business-process/enterprise-resource-planning.epx.

6. .. [En línea] [Citado el: 6 de noviembre de 2012.] http://www.assets.co.cu/assets.asp.

7. **ECURED, Versat Sarasola.** .. [En línea] [Citado el: 5 de diciembre de 2012.] http://www.ecured.cu/index.php/Versat_Sarasola.

8. **González, Agustín J.** MODELO ITERATIVO E INCREMENTAL.. [En línea] [Citado el: 15 de 5 de 2013.] http://www.dsic.upv.es/~uml, http://inst.eecs.berkeley.edu/~cs169/.

9. **Institute, Software Engineering.** Que es CMMI? - Software Engineering Institute - Vates SA.. [En línea] [Citado el: 13 de 6 de 2013.] http://www.vates.com/cmmi/que-es-cmmi.html.

10. .. [En línea] [Citado el: 3 de diciembre de 20012.] http://php.ciberaula.com/articulo/PHPoASP/.

11. **JavaScript.** .. [En línea] [Citado el: 5 de diciembre de 2012.] http://www.pergaminovirtual.com.ar/definicion/JavaScript.html.

12. **PÉREZ, Mª TERESA GARZÓN.** .. [En línea] [Citado el: 4 de 1 de 2013.] http://www.csi-csif.es/andalucia/modules/mod_ense/revista/pdf/Numero_30/TERESA_GARZON_1.pdf.

13. **Martinez, Rafael.** PostgreSQL-es.. [En línea] [Citado el: 11 de febrero de 2013.] http://www.postgresql.org.es/node/297.

14. **JJ., Gutiérrez.** ¿Qué es un framework web?.. [En línea] [Citado el: 10 de diciembre de 2013.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

15. Un Vistazo a Ext JS: Introduccion a Ext JS, con algunos ejemplos.. [En línea] [Citado el: 15 de diciembre de 2013.] http://www.desarrolloweb.com/wiki/un-vistazo-a-ext-js.html.

16. **JM., Pérez.** Doctrine2 Parte 1: Introducción, Instalación y Configuración Inicial.. [En línea] [Citado el: 5 de diciembre de 2012.] http://nubedeideasparadesarrollar.blogspot.com/2011/02/doctrine2-parte-1-introduccion.html.

17. **Allen, Rob.** .. [En línea] [Citado el: 20 de marzo de 2013.] http://alemohamad.com/tutorial-zend-framework/.

18. NetBeans.. [En línea] [Citado el: 6 de febrero de 2013.] http://netbeans.org/community/releases/70/.

19. .. [En línea] [Citado el: 25 de marzo de 2013.] http://www.maestrosdelweb.com/editorial/firefox/.

20. Plataforma de Desarrollo de Software Libre (PDSL).. *Manual del Usuario del Sistema de Control de Versiones (SVN).* [En línea] [Citado el: 25 de marzo de 2013.] http://plataforma.cenditel.gob.ve/wiki/ManualUsuarioSvn.

21. concepto de modelo de datos.. [En línea] [Citado el: 2 de febrero de 2013.] http://blogs.ua.es/mu171m3d14/2011/04/17/modelo-de-datos-1%C2%AA-parte/.

22. **JP., Quiroga.** Requisitos funcionales y no funcionales... [En línea] [Citado el: 5 de febrero de 2012.] http://sistemas.uniandes.edu.co/~csof5101/dokuwiki/lib/exe/fetch.php?media=principal:csof5101-requerimientos.pdf.

23. **Quiroga Cerdano G, Zuleta Mayta A, Aguilar Chambi A, Laura Quispe J.** Diagrama de clases de diseño y diagrama de clases. Universidad Salesiana de Bolivia.. [En línea] [Citado el: 20 de febrero de 2013.] http://www.google.com.cu/url?sa=t&rct=j&q=diagramas+de+dise%C3%B1o+definicion&source=web&cd=21&ved=0CHIQFjAU&url=http%3A%2F%2Fvirtual.usalesiana.edu.bo%2Fweb%2Fpractica%2Farchiv%2Fclase1.ppt&ei=ekyQUY3QJ7e24AOWkIHwBw&usg=AFQjCNGl7YWcC2iT-HWHKQJ4cxbnfCQ7Zw.

24. **Tello., Cáceres.** Diagramas de secuencia. Dpto. Ciencias de la Computación. Universidad de Alcalá.. [En línea] [Citado el: 5 de marzo de 2013.] http://www2.uah.es/jcaceres/capsulas/DiagramaSecuencia.pdf.

25. **ITIS, ITIG.** .. [En línea] [Citado el: 15 de 5 de 2013.] http://www.slideshare.net/joshell/diagramas-uml-componentes-y-despliegue.

26. **N., Tedeschi.** ¿Qué es un patron de diseño? Facultad de Ingeniería del Uruguay.. [En línea] [Citado el: 10 de marzo de 2013.] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

27. **Visconti M, Astudillo H.** Fundamentos de la Ingeniería de Software: patrones GRASP.. [En línea] [Citado el: 15 de marzo de 2013.] http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf.

28. **Mestras, Juan Pavón.** Patrones de diseño orientado a objetos.. [En línea] [Citado el: 10 de 5 de 2013.] http://www.fdi.ucm.es/profesor/jpavon/poo/2.14PDOO.pdf.

29. **J., Pavón Mestras.** Patrones de diseño orientado a objetos: Programación orientada a objetos.Facultad de Informática Universidad Complutense de Madrid.. [En línea] [Citado el: 20 de marzo de 2013.] http://www.fdi.ucm.es/profesor/jpavon/poo/2.14PDOO.pdf.

30. **Gómez Baryolo, Oiner, Morejón Borbón, Yoandry y García Tejo, Darien.** *Arquitectura tecnológica para el desarrollo de software..* 2013.

31. **Roger Pressman, IEEE.** .. [En línea] [Citado el: 5 de 20 de 2013.] http://cristinadavila.wordpress.com/2012/05/13/pruebas-en-el-software-segun-pressman-vs-ieee-29119/.

32. **Pressman, Roger.** .. [En línea] [Citado el: 20 de 5 de 2013.] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/rueda_p_jo/capitulo4.pdf.

Anexos

Anexos 1

Figure 30. Modelo de datos.

Anexos 2

Descripción de las tablas generadas

Tabla 1 Diccionario de datos tabla dat_confajusteprod

Nombre: tabla dat_confajusteprod		
Atributo	**Tipo**	**Descripción**
idconfajusteprod	Numeric	Llave primaria de la llave
Idestructura	Varchar	
idfechasistema	Date	

idfechaauditoria	Date	
Idejercicio	Numeric	
Idperiodo	Numeric	Llave foránea
Idmoneda	Numeric	Llave foránea
idprodterminada	Numeric	

Tabla 2 Diccionario de datos tabla dat_confproductodestino

Nombre:dat_confproductodestino		
Atributo	**Tipo**	**Descripción**
idconfproductodestino	Numeric	Llave primaria de la llave
Iddescripcion	Varchar	
Idestructura	Varchar	
idconfajusteprod	Numeric	Llave foránea
Idconfdestino	Numeric	Llave foránea
Idreaccion	Numeric	

Tabla 3 Diccionario de datos tabla dat_confdestino

Nombre:dat_confdestino		
Atributo	**Tipo**	**Descripción**
idconfdestino	Numeric	Llave primaria
Idcuenta	varchar	
Idestructura	varchar	

Tabla 4 Diccionario de datos tabla dat_producto

Nombre:dat_producto		
Atributo	**Tipo**	**Descripción**
Idproducto	varchar	Llave primaria
idconfajusteprod	numeric	Llave foránea
Idcuentainv	varchar	

idcuentagasto	varchar	
idprodequivalente	numeric	
Idestructura	varchar	
Idalmacen	varchar	
Idvalorunitario	numeric	
Idcantidad	numeric	
Idimporte	numeric	

Tabla 5 Diccionario de datos tabla dat_realizarajuste

Nombre:dat_realizarajuste		
Atributo	**Tipo**	**Descripción**
idrealizarajuste	numeric	Llave primaria
idperiodo	varchar	
idconfajusteprod	numeric	Llave foránea
Idejercicio	varchar	
idfechasistema	date	
idfechaauditoria	date	
Idestructura	varchar	
Idreaccion	numeric	

Tabla 6 Diccionario de datos tabla dat_cuentasprocesos

Nombre:dat_cuentasprocesos		
Atributo	**Tipo**	**Descripción**
Idctaprocesos	numeric	Llave primaria
idcuentasprocesos	numeric	
idconfajusteprod	numeric	Llave foránea
Idcentrocosto	varchar	
idarearesponsabilidad	varchar	
idelementogasto	varchar	

Idestructura	varchar	

Tabla 7 Diccionario de datos tabla dat_docinventario

Nombre:dat_docinventario		
Atributo	**Tipo**	**Descripción**
iddocinventario	numeric	Llave primaria
idrealizarajuste	numeric	Llave foranea
Idalmacen	varchar	Las tablas con
Idestructura	varchar	
Idreaccion	numeric	

Anexos 3

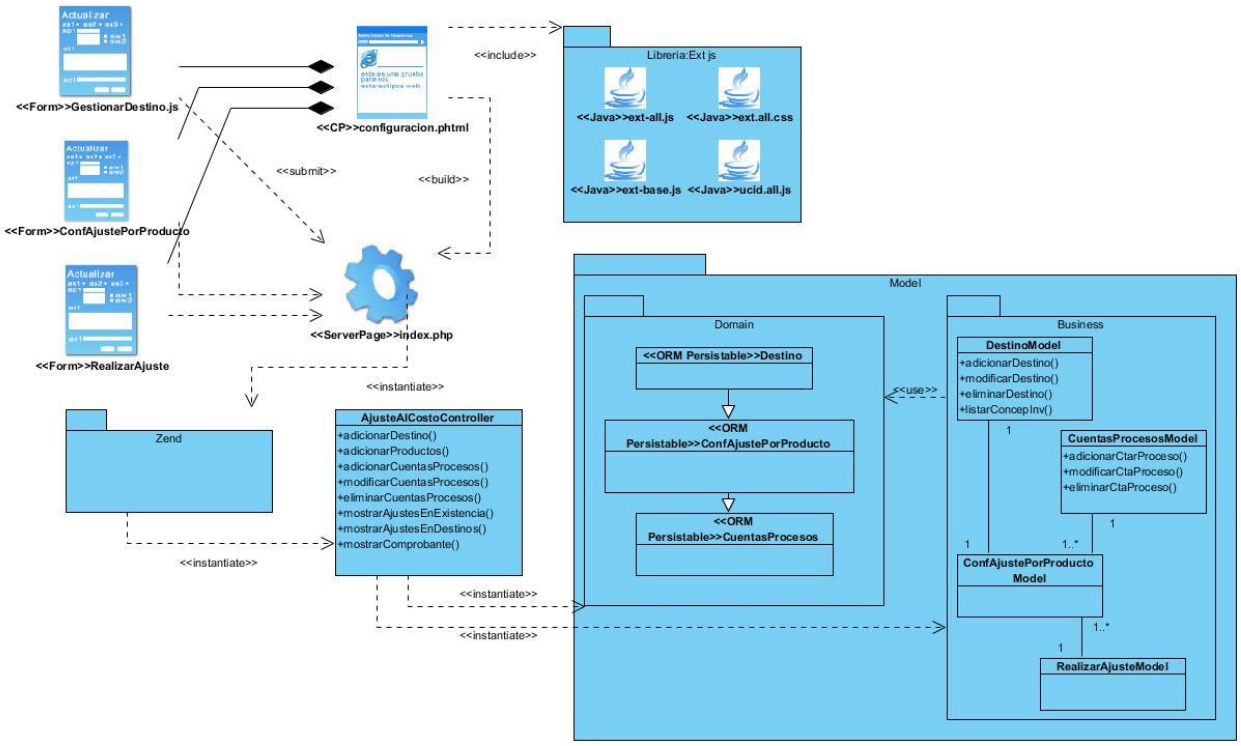


Figure 31. Diagrama de clase de diseño.

Anexos 4

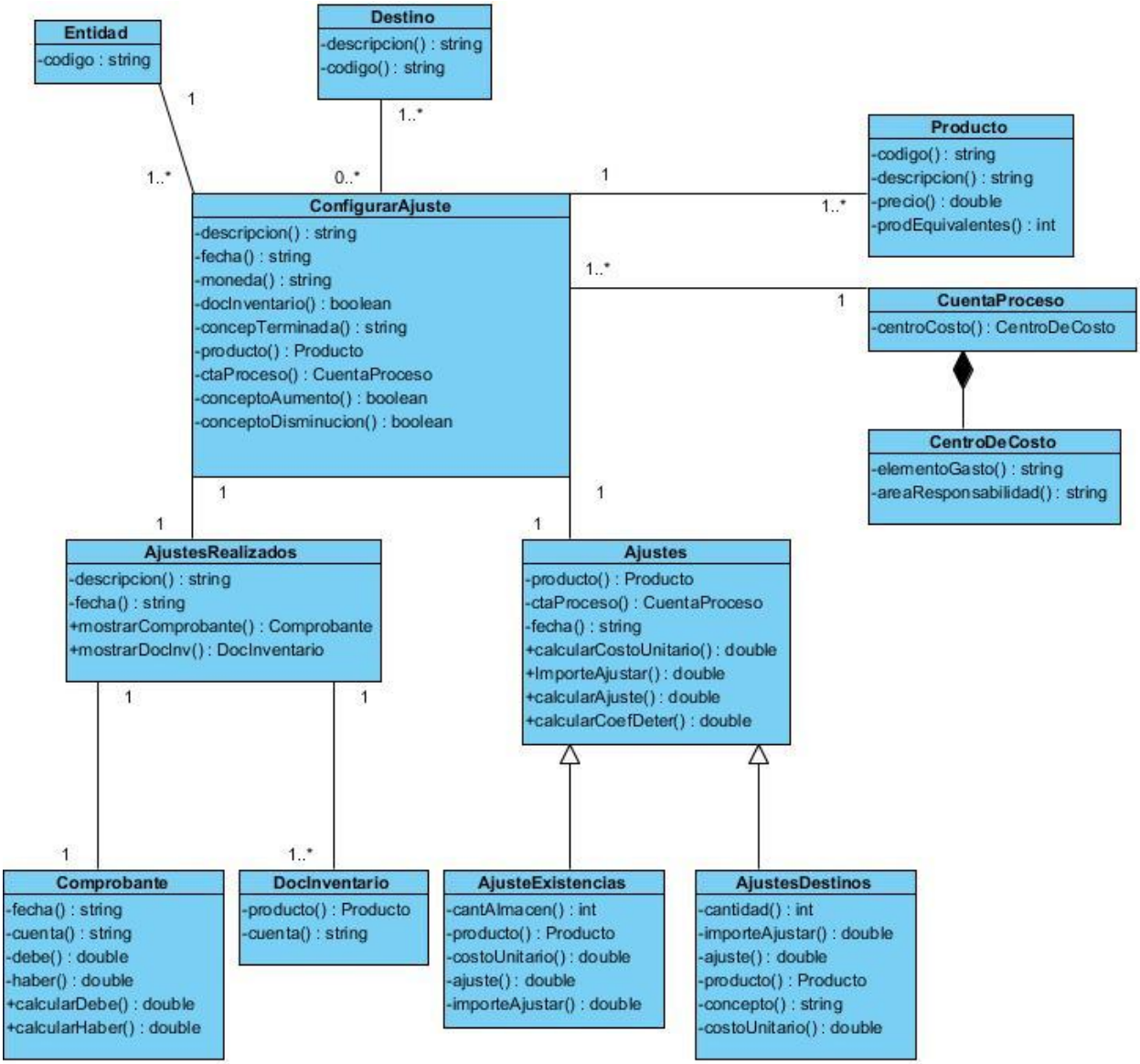


Figure 32. Diagrama de clase.

Anexos 5

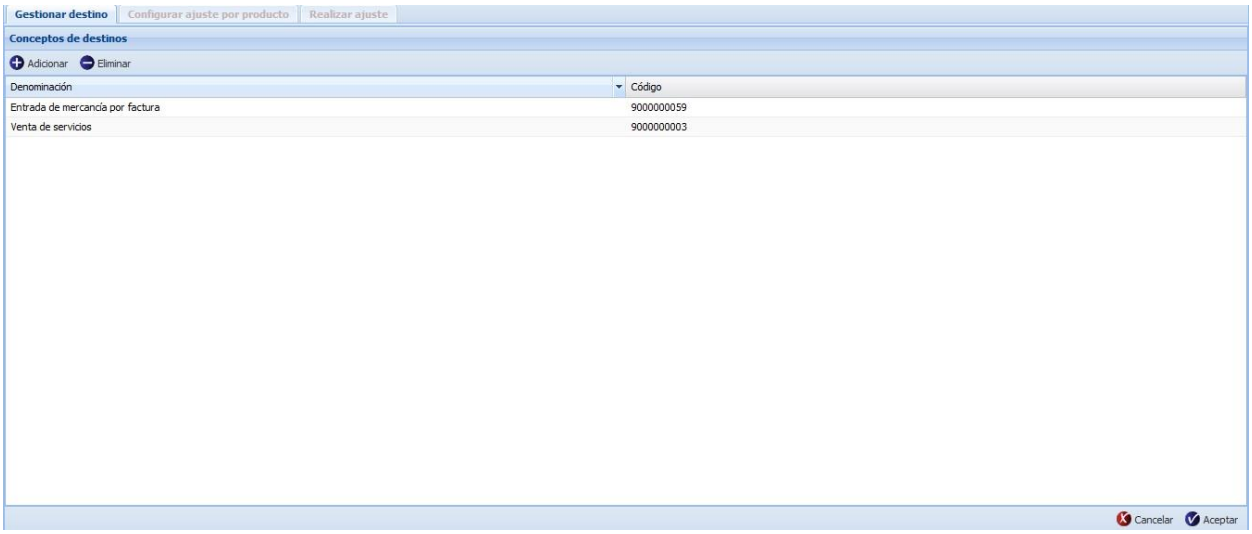


Figure 33. Prototipo de interfaz funcional gestionar destinos.

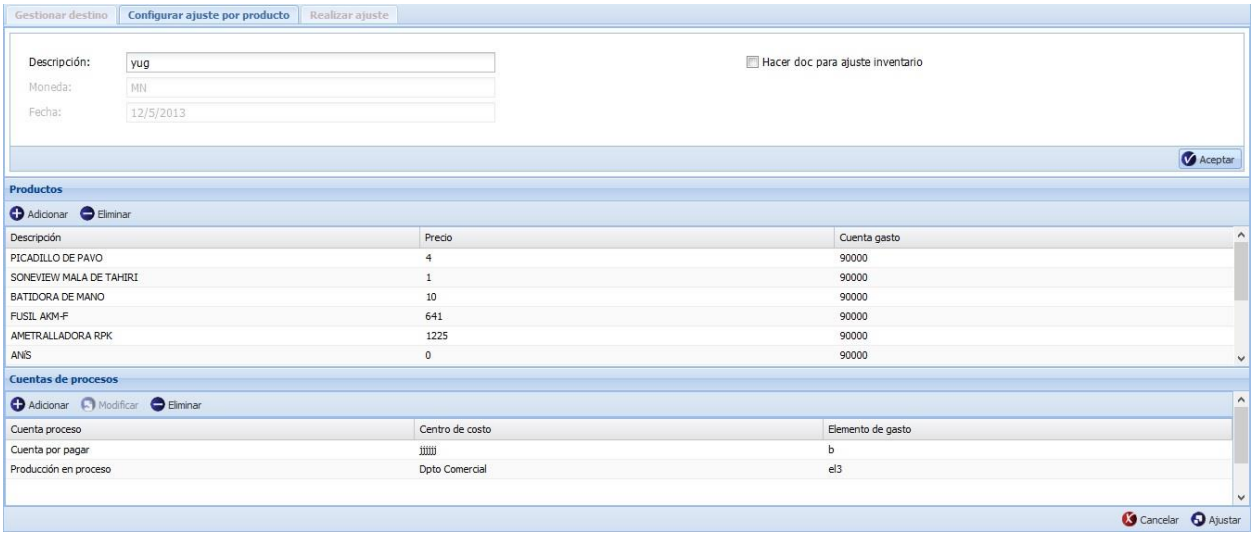


Figure 34. Prototipo de interfaz funcional configurar ajuste por producto.

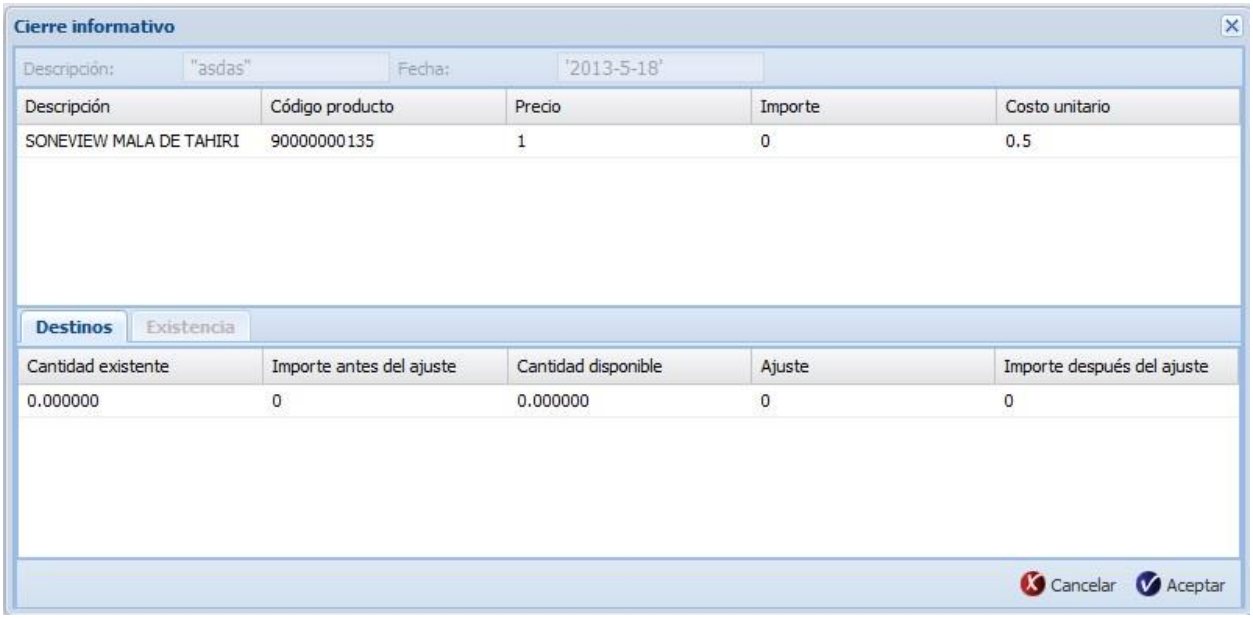


Figure 35. Prototipo de interfaz funcional cierre informativo.

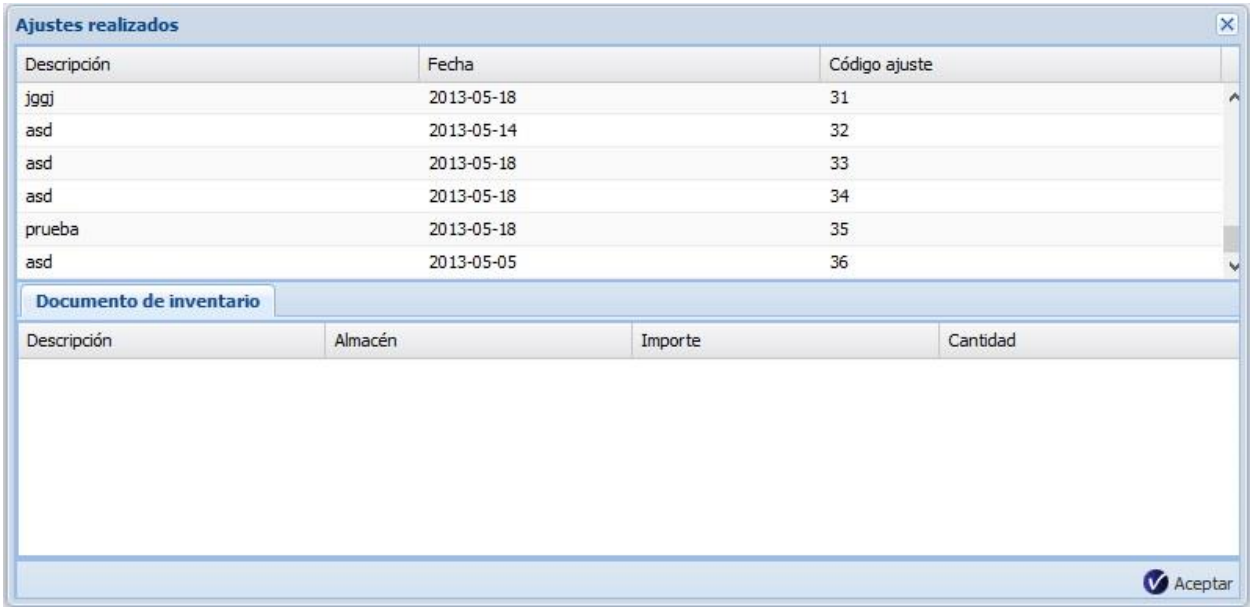


Figure 36. Prototipo de interfaz funcional ajustes realizados.

Anexos 6

ACTA DE LIBERACIÓN DEL COMPONENTE AJUSTE AL COSTO POR PROCESOS.

1 Introducción

1.1 Objetivo

El objetivo de este documento es definir una plantilla para los documentos de contenido que se generen por la empresa.

1.2 Alcance

Está destinado a todos los miembros de la empresa que participen en el uso de la documentación afín de la empresa.

1.3 Definiciones y acrónimos

- **Plantilla:** Documento de ejemplo que sustenta un formato y que describe los lineamientos para la elaboración de documentos similares.

1.4 Referencias

IEEE. 1991. *IEEE Standard Glossary of Software Engineering Terminology.* Spring 1991 Edition. 1991. IEEE Standard 610.12–1990.

2 Datos del producto

Emitida a favor de: Componente ajuste al costo por proceso.
Fecha de emisión del acta: 18/06/2013
Responsable: Ing. Yanay Hernández Sosa
Cargo: Jefa de línea

2.1 Clasificado como:

- Aplicación Web.

2.2 Detalle de los elementos probados y su estado final:

Artefacto	Estado Final
Aplicación	0 No Conformidades
Documentación	0 No Conformidades

2.3 Cantidad de iteraciones:

Para la revisión se emplearon un total de 2 iteraciones para lograr el resultado de 0 (cero) No Conformidad.

Artefacto	Versión	Estado final	Cantidad Iteraciones	Tipos de pruebas realizadas	Fecha de liberación
Aplicación	Versión 1.0	No Conformidades	2 iteraciones total	Pruebas Exploratorias Pruebas Funcionales	18/06/2013

3 Elementos revisados o probados y herramientas utilizadas

Elemento	Herramienta
Complejidad	Estándar de interfaz, Lista de Chequeo Interfaz
Ortografía y redacción	Revisión Técnica

3.1 Cantidad total de horas empleadas y rango de fechas:

Se emplearon un total de 3 horas efectivas de trabajo con la siguiente distribución: 2h (17/Junio/2013), 1 h (18/Junio/2013).

Estructura del equipo de prueba empleado y turnos de trabajo:

Las pruebas se realizaron en un total de 2 turnos de trabajo, en cada turno se contó con 1 probador por parte del equipo de pruebas, la actividad estuvo dirigida por un Jefe de Pruebas.

4 Evaluado por:

4.1 Especialista principal Asignado:

Ing. Giselle Almeida González

4.2 Otro personal especializado participante:

Nombres y Apellidos	Cargo
Ing. Giselle Almeida González	Administrador de Calidad
Lic. Leandro Roberto Reyes Mejías	Especialista General

Anexos 7

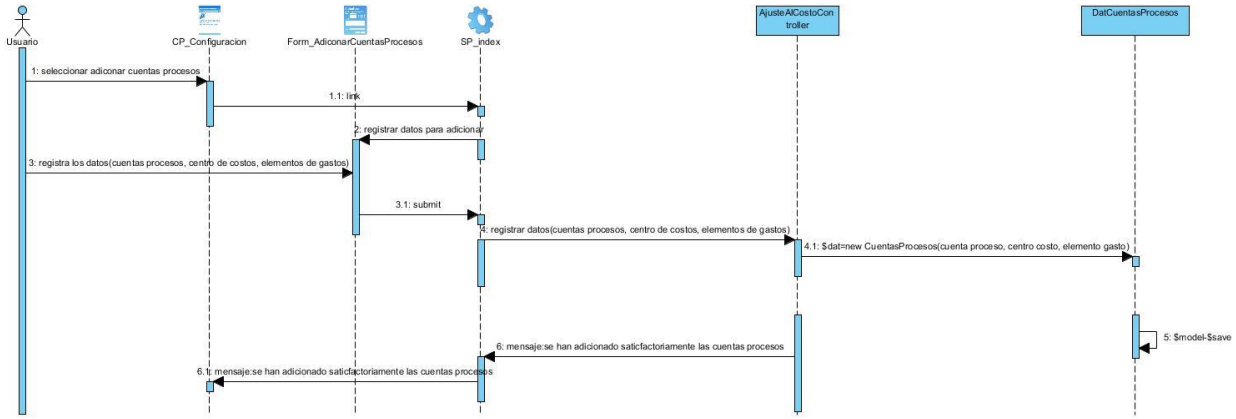


Figure 37. Diagrama de secuencia adicionar cuentas procesos.

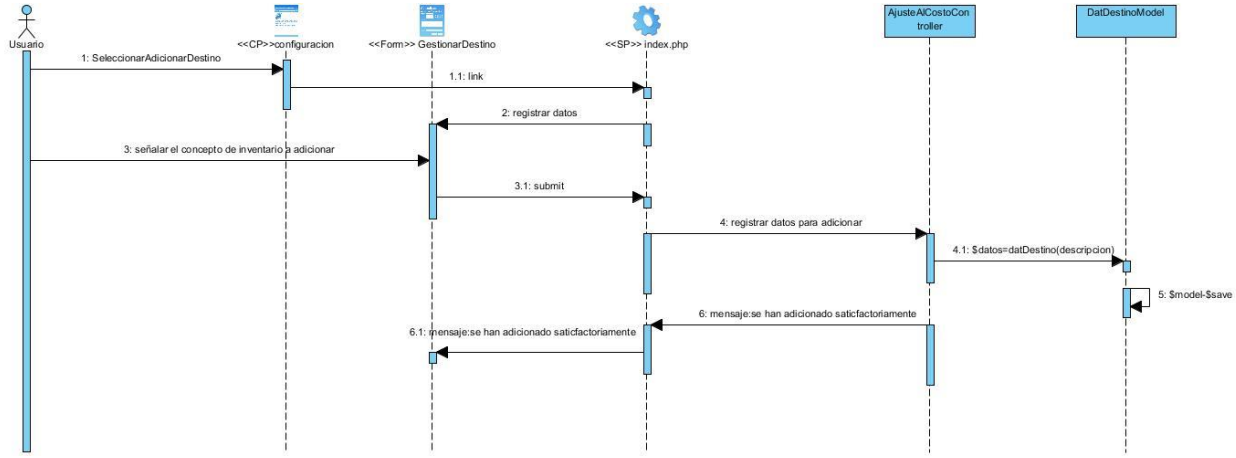


Figure 38. Diagrama de secuencia adicionar destino.

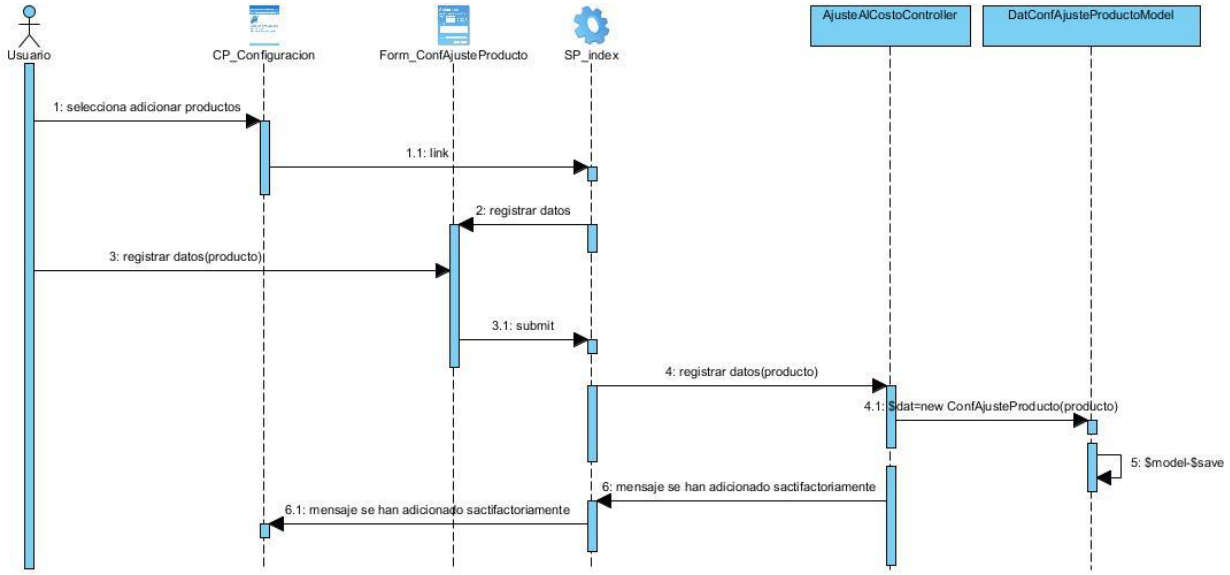


Figure 39. Diagrama de secuencia adicionar productos

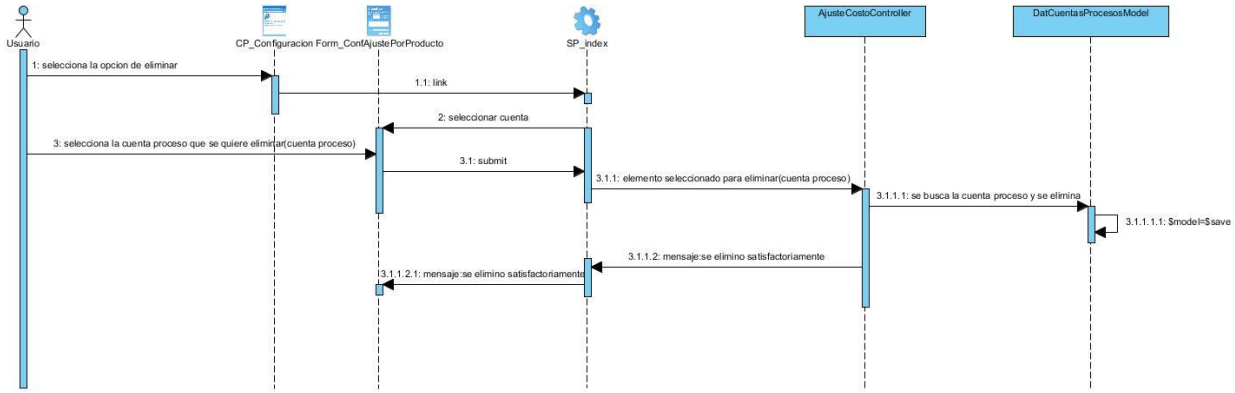


Figure 40. Diagrama de secuencia eliminar cuentas procesos

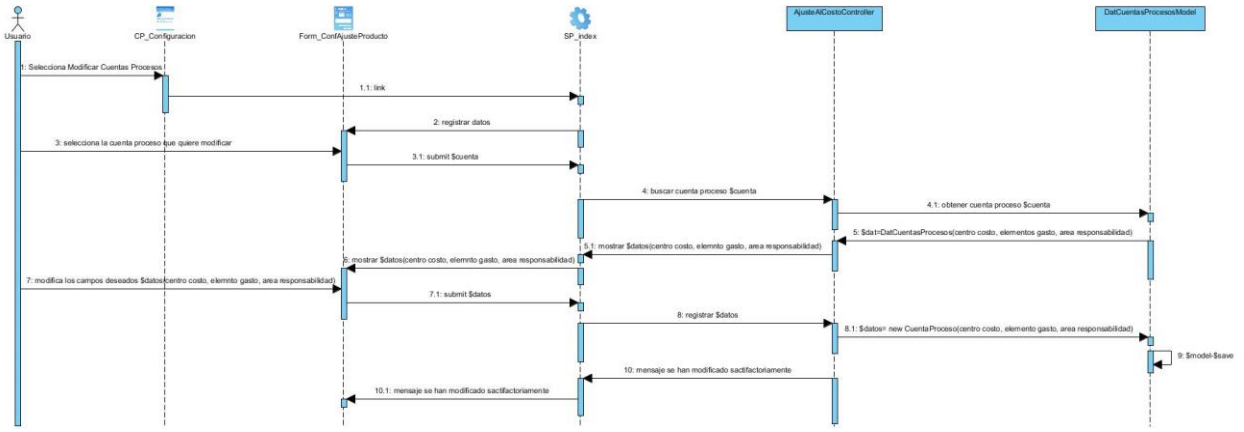


Figure 41. Diagrama de secuencia modificar cuentas procesos

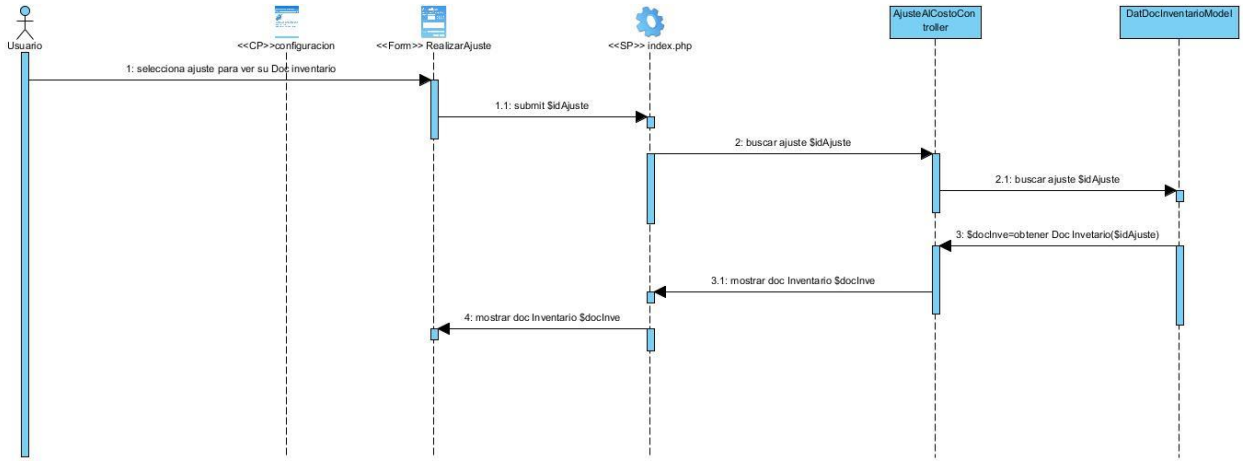


Figure 42. Diagrama de secuencia mostrar documento de inventario

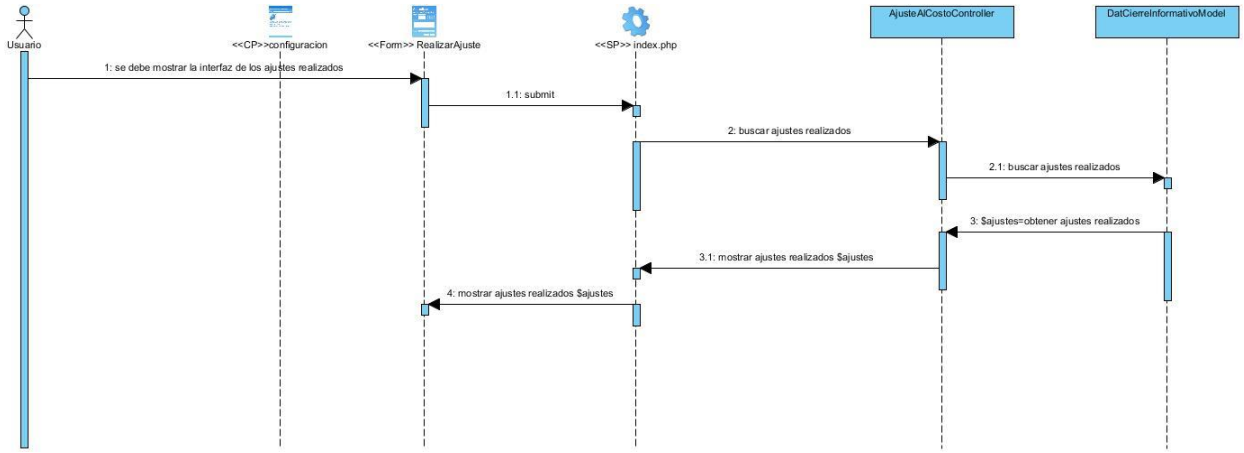


Figure 43. Diagrama de secuencia ajustes realizados.

Anexo 8

1 Condiciones de ejecución

- Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
- Se debe seleccionar el subsistema de **Costo y Procesos**.
- Se debe seleccionar la opción **Ajuste al Costo/ Ajuste al Costo**

1.1 Requisitos a probar

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Eliminar producto.	Se elimina un producto.	EP 1.1: Eliminar un producto.	– Se selecciona el producto a eliminar. – Se presiona el botón **Eliminar.** – Se muestra un mensaje de confirmación. – Se presiona el botón **Aceptar** del mensaje de confirmación. O se presiona el botón Cancelar. – En caso de seleccionar el botón **Aceptar** se muestra un mensaje de información. – Se presiona el botón **Aceptar** del mensaje de información.
		EP1.2: Eliminar un producto que se esté usando.	– Se selecciona el producto a eliminar. – Se presiona el botón **Eliminar.** – Se presiona el botón **Aceptar.** – Se muestra un mensaje de información. – Se presiona el botón **Aceptar** del

		mensaje de información.
EP 1.3:	Cancelar	– Se selecciona el producto a eliminar – Se presiona el botón **Eliminar.** – Se presiona el botón **Cancelar.** – Se mantiene en la ventana producto.

2 Condiciones de ejecución

- Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
- Se debe seleccionar el subsistema de **Costo y Procesos**.
- Se debe seleccionar la opción **Configuración / Ajuste al Costo / Destinos**.

2.1 Requisitos a probar

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Adicionar Destinos	El sistema debe permitir adicionar los destinos después que estos se carguen automáticamente del subsistema inventario	EP 1.1: Adicionar Destino.	– Se selecciona el concepto de inventario el cual se va a destinar, se presiona el botón de aceptar y se muestra un mensaje que se han adicionado los destinos.