“IMPLEMENTACIÓN DEL MÓDULO DESPACHO DEL SUBSISTEMA INVENTARIO DEL SISTEMA INTEGRAL DE GESTIÓN CEDRUX”

Trabajo de Diploma para optar por el Título de Ingeniería en Ciencias Informáticas

Autor(es): Kenia Riverón Ovalle

Yaidel Rodríguez Blanco

Tutora: Lic. Arismayda Dorado Risco

Ciudad de La Habana, Cuba

Junio del 2009

PENSAMIENTO

''La vida de un individuo tiene sentido siempre que ayude a hacer la vida de otro ser viviente más noble y hermosa''

Albert Einstein

DECLARACIÓN DE AUTORÍA

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

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

_____ _____

Yaidel Rodríguez Blanco Kenia Riverón Ovalle

_____

Lic. Arismayda Dorado Risco

DATOS DE CONTACTO

Tutora: Lic. Arismayda Dorado Risco

Clasificación: Profesional

Clasificación del área de desarrollo: Dirección de Producción 4

Síntesis de la Tutora: Licenciada, cuatro años de graduada

AGRADECIMIENTOS

…de Kenia:

A mi familia por sentir orgullo de mí.

A Yuniel por su amor, ayuda y comprensión.

A Yvon y a Ortiz por portarse como mis padres.

A Taty y a su familia por su apoyo y ser mi familia.

A Eloi y a Alichín por preocuparse por mí y ayudarme.

A nuestro tutor por su apoyo incondicional y confianza.

A Lugo, Yuri, Modesto y Saili por su ayuda en todo momento.

A mis primas por darme el aliento de sentir como si estuviera en casa.

A Edgar por su apoyo incondicional, quien me impulso a seguir la carrera.

A Roexcy, Yady, Pache, Albeto, pepe por ayudarme y preocuparse por mí.

A mis tías y tíos, por su preocupación y dedicación en todos estos años de estudio.

A mi compañero de tesis por toda su paciencia y confianza al realizar la tesis conmigo.

A mis amigas de mi infancia, que aun siguen siendo mi mano derecha, en especial Deylis.

A las tías del edificio, principalmente a Albis, Arianna, Dibelkis y Maira, por su procupación por mí.

A Yohana, Anais, Frank y en especial a Iveisis, fueron quienes me dieron aliento de seguir estudiando.

A las personas del grupo en especial a Jose, Thais, Enrique, Mariluz, Yvon, Ida, Ariana, Malena, Lisi, Sahyli, Yadira, Yuliet, Roberto.

A mis padres y a mi hermano quien siempre supieron darme el frente, los que se preocuparon día a día por mí, en especial a mi mamá a quien se lo debo todo en esta vida.

A aquellas personas que considero mi familia por siempre estar a mi lado en las buenas y en las malas.

A todas las personas que se preocuparon de una forma u otra por mi bienestar.

A todos ellos mucha gracias.

…de Yaidel:

A mis padres, por todo lo que me enseñaron, por su aliento en los momentos más difíciles, por estar siempre pendientes y por ser los principales artífices de que yo haya llegado hasta aquí.

A mi hermana, por estar siempre pendiente, por los consejos y estar siempre ahí.

A mi tutora, por el tema propuesto, por aguantarnos en estos meses, por los regaños, los cuales siempre eran bien recibidos y por su ayuda incondicional al trabajo.

A mi compañera de tesis, a la que tendré siempre presente.

A Yuriel y Lugo por romperse la cabeza junto a mi lado.

A Yuniel, por su ayuda, opiniones y arrastrar con nosotros.

A mis compañeros de aula, las mejores que una persona puede desear: Ariana (mi peluquera), Ivón, Idaliana, Lisandra, Sahily, Any, Leyanis, Annier y Abelito.

A mis amigos Juan Carlos, Jairol, Gustavo, el Nino, Toledo, Ramiro, Alejandro, Carlos Mario, Malena, y Rega, a los que llevaré siempre conmigo.

A mis amigos del IPVCE, que aunque nos separen kilómetros siempre hemos estado en contacto y estaremos: Lester, Lisandra (mi negra), Yaliet, Fiallo, Aliet y Neni.

… a todos, GRACIAS, por sus valiosas sugerencias y recomendaciones para llevar a feliz término esta tarea…

DEDICATORIA

…de Kenia

A mi mamita, por ser mi guía, por su apoyo y amor.
A mi herma Kemil, por soportarme siempre.
A Eloi por confiar en mí.
A mi papá, creo haber cumplido tu sueño de hacerme alguien en la vida.
A la memoria de mi abuela Eloisa, decía que yo era la única que servía en la familia.

…de Yaidel:
A mi hermano Alexey, ésta es la tuya "cuñao".
A mi mamá y mi papá, por estar siempre a mi lado dando lo mejor de ellos.
A mi sobrino.
A mi hermana, la cual con su ejemplo me ha guiado hasta aquí.

RESUMEN

Debido a los grandes avances tecnológicos y como consecuencia del gran volumen de información que se genera actualmente, las entidades cubanas han tratado de nutrirse al máximo con el uso de la informática para el desarrollo interno de las mismas.

Como base y sostén de la economía nacional, las empresas cubanas son las encargadas de llevar el control y la gestión de los productos que se producen e importan en el país. Para ello tienen asociadas a las mismas, los almacenes; quienes constituyen un eslabón esencial en los flujos físicos de mercancías desde un origen a un destino, y junto a la gestión de pedidos y compra, la producción, la gestión de inventario, el transporte, la distribución e incluso el reciclaje, integran un sistema logístico que es vital para el funcionamiento de la Economía. Como parte del funcionamiento del almacén se encuentran los procesos de Despacho. Encargándose el Despacho de la gestión de las operaciones de salida del almacén por criterios de venta, transferencias, vales de entrega.

El objetivo que persigue el presente trabajo de diploma es llevar a cado la implementación de las funcionalidades detectadas por los analistas. Para lograr esto fue necesario analizar las herramientas para generar los artefactos. Se hizo necesario también estudiar la documentación que fue entregada por los clientes y usuarios finales con el fin de comprender los procesos de negocio y una posterior captura de requisitos. Por último se obtuvo un producto funcional que arrojó muy buenos resultados por las validaciones realizadas.

PALABRAS CLAVES

ALMACÉN, DESPACHO, PRODUCTO

Índice de Figuras

Índice de Tablas

Contenido

INTRODUCCIÓN

En el mundo de hoy la mayoría de las operaciones y actividades giran en torno a una computadora. La informática soporta importantes y continuos cambios que repercuten en la sociedad y en el desarrollo del mundo empresarial. Las Tecnologías de la Información y las Comunicaciones (TIC), producto de sus avances y como integrantes de esta ciencia, han posibilitado el incremento rápido del desarrollo de la economía y la sociedad. Estos avances vienen dados gracias a la ambición de conocimiento del ser humano, a su naturaleza creadora, y son aplicados en gran medida en el desarrollo de la economía y la sociedad mediante la producción de software. Esta actividad demanda cada vez más un mayor estudio y una mejor planificación porque cada vez se solicitan programas más complejos y con mayor calidad. Los sistemas de software se especializan en negocios determinados. Recientemente se está estilando desarrollar un sistema que constituye un paquete de software empresarial que integra todas las áreas de la organización, estos sistemas se denominan sistema de Planeación de Recursos Empresariales (*Enterprise Resource Planning*, ERP). Este sistema garantiza la centralización de la información de una empresa. Implementar un sistema de ERP es un proceso largo, costoso, complejo y que requiere de gran cantidad de desarrolladores. Es por esto que es más factible dividir su desarrollo en módulos que representan las distintas áreas de la empresa, de forma que se viabilice su proceso de desarrollo. Uno de estos módulos abarca la Gestión de los procesos del Despacho, el cual se encarga de los procesos para el control de inventario, que agrupa la apertura, movimientos, posteo, operaciones contables, cierre, de los productos que se encuentran en almacenados, además de brindar toda la información necesaria para la entidad. Las creación de los documentos que soportan las salidas por venta, transferencias o conduces son el centro de las responsabilidades en las cuales se centra el despacho.

La Gestión de Despacho permite que todos los procesos conlleven a mejor control de los productos de una organización. Se encarga de evaluar los procedimientos de salida de sus bienes, es decir, controla su existencia con el fin de hacer más rentable su posesión y garantizar en cierto grado el éxito de la organización.

Cuba cuenta con algunos sistemas para el despacho de los productos en los almacenes, tanto nacionales como internacionales. Sin embargo, no existe un sistema único para implantar en las empresas cubanas que cumpla con las necesidades del cliente y reúna la mayor cantidad de funcionalidades posibles, por lo

que se pretende implementar un sistema ERP cubano que pueda ser utilizado por cualquier empresa y que permita centralizar toda la información en la misma.

Además, se ha trazado como meta informatizar la sociedad para valerse de las ventajas que se derivan del uso de la tecnología y favorecer el desarrollo económico y social. Con la evolución hacia el uso de las Tecnologías de la Información y las Comunicaciones (TIC), se necesitan realizar transformaciones referidas a la gestión empresarial, administrativa y de calidad.
La Universidad de las Ciencias Informáticas (UCI) fue creada en el 2002 con el propósito de preparar recursos humanos que contribuirán con la informatización del país. Es una Universidad productiva donde los estudiantes y sus conocimientos van encaminados a la producción de software, por lo que se le dio la tarea de desarrollar un sistema integral de gestión para las entidades del país. Esta tarea conlleva a un proceso en el cual es necesario optar por técnicas, métodos, herramientas, que permitan desarrollarlo con la mayor calidad y organización posible.

Dada esta situación problemática, se define como **Problema científico:**
¿Cómo garantizar la existencia de una herramienta para uso común del proceso de Despacho de productos en almacenamiento dentro del entorno empresarial cubano?

Para dar solución al problema planteado se hace necesario realizar un **estudio** de:
Proceso de Gestión de Almacenes.

Profundizando en:
Proceso para el Despacho de los productos en almacenamiento del país.

Como **Objetivo general** de la investigación**:**
Implementar el proceso de Despacho de los productos en almacenamiento para el entorno empresarial cubano.

La investigación se basa en la **Hipótesis** de que**:**

Si se realiza un módulo que permita el Despacho y control de los productos en almacén, se garantiza la existencia de una herramienta automatizada para uso común en el entorno empresarial cubano.

Para dar cumplimiento al objetivo general se derivan **Objetivos específicos** para un mejor seguimiento los mismos son:

- Analizar los procesos de Despacho, los sistemas que existen actualmente para su gestión, así como las herramientas que se utilizarán para el desarrollo de la solución.
- Implementar un módulo para el Despacho de los productos en almacenamiento.
- Validar la solución propuesta.

Como **Tareas:**

- Análisis de los procesos de inventario físico.
- Análisis de los sistemas existentes para el control de inventarios.
- Análisis de las tecnologías, lenguajes y herramientas propuestas para el desarrollo de la aplicación.
- Análisis de los artefactos entregados por los analistas para los procesos de inventario físico.
- Implementación de las interfaces a partir del prototipo entregado por los analistas.
- Implementación de la capa de negocio que dé respuesta a los requisitos propuestos por los analistas.
- Implementación de la capa de acceso a datos.
- Implementación de las validaciones y excepciones.
- Realización de pruebas de unidad.

La estructura del documento se divide en 3 capítulos:

Capítulo 1: Se exponen los conceptos fundamentales relacionados con el tema de investigación, se realiza una caracterización de la metodología de desarrollo de software empleada y se describen los principales aspectos del lenguaje y herramientas.

Capítulo 2: Se realiza una valoración crítica del diseño propuesto por los analistas, dado la importancia del mismo para la implementación, se analizan las componentes ya existentes que pueden ser rehusados. Se efectúa la selección de las estructuras de datos apropiadas para la implementación de estos algoritmos y se realiza la descripción de las clases más importantes con sus operaciones.

Capítulo 3: Se desarrolla un análisis y diseño de las pruebas de unidad que permiten validar la solución propuesta. Se realiza una descripción de estas teniendo en cuenta (Objetivo, Alcance, Tipo de prueba y detalles de la misma). Se efectúa la descripción de los valores utilizados para las prueba, una evaluación de la ejecución de estas y de los resultados obtenidos.

Capítulo 1 Fundamentación Teórica

1.1 Introducción

En este capítulo se tratan los principales conceptos relacionados con los Sistemas de Planificación de Recursos Empresariales, en específico los relacionados con el proceso de despacho dentro de la gestión de almacenes.

Se da una panorámica general del caso de Cuba, fundamentado en la necesidad de la migración a Software Libre, además de la línea que se sigue actualmente por las empresas que se auxilian de software ERP para la gestión de sus procesos.

Se fundamenta de manera general la propuesta de solución basada en aplicaciones Web, analizando el lenguaje a utilizar, así como las herramientas de apoyo al proceso de desarrollo del software. En cada caso se deja claro el por qué de su selección y las principales ventajas que brinda.

1.2 Planeación de Recursos Empresariales (ERP)

La Planeación de Recursos Empresariales, es una forma de utilizar la información a través de la organización de forma más proactiva en áreas claves como: fabricación, compras, administración de inventario y cadena de suministros, control financiero, administración de recursos humanos, logística y distribución, ventas, mercadeo y administración de relaciones con clientes. La aplicación de las técnicas de ERP permite ganar en organización y control dentro de la empresa.

Como se mencionó anteriormente, la Planeación de Recursos Empresariales puede abarcar muchas áreas dentro de la entidad. Algunas de esas áreas tienen especial significación, debido a su importancia y los recursos que manejan.

1.3 Inventario. Gestión de Almacenes

La gestión de almacenes es una de las tareas que se realizan dentro del control de inventario y se puede definir como: el proceso logístico que trata la recepción, almacenamiento y movimiento dentro de un mismo almacén, hasta el punto de consumo de cualquier material, materias primas, semielaborados, terminados, así como el tratamiento e información de los datos generados.

Fig. 1 Proceso de Gestión de almacenes

Esta tarea engloba otros subprocesos de singular vitalidad e importancia. A continuación se describe uno de ellos.

1.3.1 Despacho.

En el proceso de gestión de almacenes se realizan muchas tareas y una de ellas es el despacho de lo medios en almacenamiento, el cual se puede definir como el conjunto de actos y formalidades a la salida de los productos de nuestros almacenes.

Desde hace algunos años comenzaron a surgir sistemas informáticos dirigidos a automatizar los procesos de la planificación de recursos empresariales. Dichos sistemas son conocidos como sistemas ERP.

1.4 Sistemas ERP

"*Un sistema ERP es una herramienta que ayuda a integrar todos los procesos del negocio y a optimizar los recursos disponibles*" (Ortuño, 2009).

Con un sistema integrado que posee una interfaz amigable, las barreras de información entre los diferentes sistemas y departamentos desaparecen. Toda la empresa, sus sistemas y procesos de planificación y organización de recursos pueden reunirse bajo la misma protección, para beneficiar a toda la organización. Este es precisamente el objetivo fundamental de los sistemas ERP, el beneficio de la organización.

Los objetivos principales de los sistemas ERP son:

- Optimizar los procesos empresariales.
- Lograr un acceso a toda la información de forma confiable, precisa y oportuna (integridad de datos).
- Dar la posibilidad de compartir información entre todos los componentes de la empresa.
- Eliminar datos y operaciones innecesarias de reingeniería.

Características de los sistemas ERP

El propósito fundamental de un ERP es otorgar a los clientes del negocio tiempos rápidos de respuesta a sus problemas, así como un eficiente manejo de información que permita la toma oportuna de decisiones y disminución de los costos totales de operación. A continuación se describen las características.

- Integrales: Todos los departamentos o áreas de una empresa se relacionan entre sí, permitiendo de esta forma controlar los distintos procesos de la entidad, donde la salida de un proceso puede ser la entrada de otro.
- Modulares: Se dividen en módulos, que son las áreas que se integran como un todo para el manejo óptimo de la información. Estos módulos se instalan según las necesidades del cliente.
- Adaptables: Son diseñados para ser configurables según lo que cada empresa necesite.
- Base de datos centralizada.
- Sus componentes interactúan entre sí consolidando todas las operaciones.
- En un sistema ERP los datos se ingresan sólo una vez y deben ser consistentes, completos y comunes.

- Las empresas que lo implanten suelen tener que modificar alguno de sus procesos para alinearlos con los del sistema ERP. Este proceso se conoce como Reingeniería de Procesos, aunque no siempre es necesario.

Beneficios del uso de los sistemas ERP

Varios son los puntos de vista en cuanto a los diferentes beneficios que se esperan en una implementación de un ERP, así como los impactos que este tendrá en la organización.

Es importante mencionar que las diferentes marcas creadoras de software ERP (SAP, Oracle.) tienen sus beneficios característicos. A continuación se presentan algunos de los beneficios que podrían adquirirse al implementar cualquiera de ellos:

- Sistema para manejar muchos de sus procesos comerciales.
- Integra las funciones de las aplicaciones.
- Reduce los costos de gerencia.
- Incrementa el retorno de inversión.
- Provee acceso en tiempo real a operaciones y datos financieros.
- Moderniza las estructuras administrativas.
- Permite transacciones de la información más rápidas.
- Incrementa las oportunidades de ventas.
- Mejora la calidad y la satisfacción a los clientes.
- Mide los resultados continuamente.
- La seguridad de la información está contemplada dentro del ERP, para la protección contra crímenes externos, como espionaje industrial y crimen interno, como por ejemplo malversación.

Desventajas y obstáculos de los sistemas ERP

- El éxito depende en las habilidades y la experiencia de la fuerza de trabajo, incluyendo la educación y cómo hacer que el sistema trabaje correctamente. Muchas compañías reducen costos reduciendo entrenamientos. Los propietarios de pequeñas empresas están menos capacitados, lo que significa que el manejo del sistema ERP es operado por personal que no está capacitado para el manejo del mismo.

- Cambio de personal: las compañías pueden emplear administradores que no están capacitados para el manejo del sistema ERP de la compañía empleadora, proponiendo cambios en las prácticas de los negocios que no están sincronizados con el sistema.

- Los ERP son vistos como sistemas muy rígidos y difíciles de adaptarse al flujo específico de los trabajadores y el proceso de negocios de algunas compañías, este punto se cita como una de las principales causas de falla.

- Muchos de los eslabones integrados necesitan exactitud en otras aplicaciones para trabajar efectivamente. Una compañía puede lograr estándares mínimos, y luego de un tiempo los datos sucios (datos inexactos o no verificados) reducirán la confiabilidad de algunas aplicaciones.

- Una vez que el sistema esté establecido, los costos de los cambios son muy altos (reduciendo la flexibilidad y las estrategias de control).

- La resistencia en compartir la información interna entre departamentos puede reducir la eficiencia del software.

1.5 Estudio del Estado del Arte

1.5.1 Sistemas ERP nacionales usados en el entorno empresarial cubano

A partir de la fuerte tendencia a nivel internacional del uso de los sistemas ERP, nuestro país ha introducido en diferentes ramas de la industria y la economía, algunos de estos sistemas. Las alternativas

que se ejemplifican corresponden a sistemas nacionales certificados, con características de desarrollo en plataformas totalmente propietarias y aplicaciones de escritorio.

Los motores de bases de datos utilizados son MS ACCESS, MS SQL SERVER y ORACLE en diferentes versiones.

Se describe a continuación, una síntesis de funcionalidades generales:

1.5.1.1 Versat-Sarasola

El programa Versat-Sarasola, sistema cubano de contabilidad confiable, permite enviar información eficaz, de forma inmediata, desde lugares apartados, a la vez que ofrece mayor organización, control y disciplina en cada gestión.

Fue este el primer sistema de contabilidad cubano certificado, en cuya evaluación participaron el Ministerio de Finanzas y Precios, consultorías internacionales y el organismo encargado de la seguridad informática. Es un sistema económico integrado. Constituido por 12 módulos:

- Configuración y seguridad.
- Contabilidad general y de gastos.
- Costos y procesos.
- Análisis económico empresarial.
- Control de activos fijos.
- Finanza y caja.
- Planificación y presupuestos.
- Control de inventarios.
- Pago de salario.
- Paquete de gestión.
- Contratación.
- Facturación.

Actualmente lo utilizan alrededor de 200 entidades de varias provincias.

Características

- Es una aplicación de escritorio.
- Implementado en Delphi.
- Trabaja sobre el sistema operativo Windows.
- Soporte para bases de dato SQL Server 2000.

1.5.1.2 CONDOR.

Sistema automatizado de alta complejidad y seguridad que abarca todos los aspectos del proceso contable de una entidad, tales como la dualidad de moneda y el pago por resultados. Está formado por varios Módulos:

- Activos Fijos.
- Contabilidad General.
- Nóminas.
- Control de Inventarios.
- Condexce.
- Recursos Humanos.

Este brinda mayor autonomía al cliente para efectuar cambios de estructura sin necesidad de la intervención de especialistas, quedando registrados de forma que puedan ser auditables. Incluye la contabilidad multimonedas.

1.5.1.3 RODAS XXI Versión 3.0

Sistema multiempresa y multiusuario creado por CITMATEL (Centro de Servicios Telemáticos Avanzados del Ministerio de Ciencia, Tecnología y Medio Ambiente) para la automatización de la gestión empresarial. Contiene diferentes módulos que pueden usarse integrados o independientes:

- Contabilidad.
- Efectivo Caja y Banco.
- Nóminas.
- Activos Fijos Tangibles.

- Inventarios.
- Cobros y Pagos.
- Facturación.
- Finanzas.
- Tele-cobranza.

Además, cuenta con el módulo Administrador, que brinda mayor integralidad al sistema y garantiza facilidades adicionales durante su instalación y explotación.

1.5.1.4 SISCONT5

El sistema se aviene a las definiciones y conceptos del Ministerio de la Industria Básica, aunque por las acciones contables y financieras que permite puede ser utilizado en otras entidades nacionales.

Está formado por varios Módulos:

- Efectivo en Caja y Bancos.
- Inventarios.
- Cobros y Pagos.
- Facturación.
- Activos Fijos Tangibles.
- Nóminas.
- Contabilidad.

Puede ser explotado en régimen monousuario y multiusuario. Se define para monoentidad y multientidad, para esta última existe el control de su acceso para las entidades en un mismo equipo de cómputo como servidor.

1.5.2 Sistemas ERP internacionales usados en el entorno empresarial cubano

1.5.2.1 ERP SAP/3

Es un sistema integrado de gestión que permite controlar todos los procesos que se llevan a cabo en una empresa, a través de módulos. En su nombre la **R** significa procesamiento en tiempo real y el número 3 se

refiere a las tres capas de la arquitectura de procesos: bases de datos, servidor de aplicaciones y cliente. Es un software propietario, producido por la empresa SAP, la quinta compañía más grande de software a nivel mundial. A continuación una figura que representa las capacidades del SAP/R3.

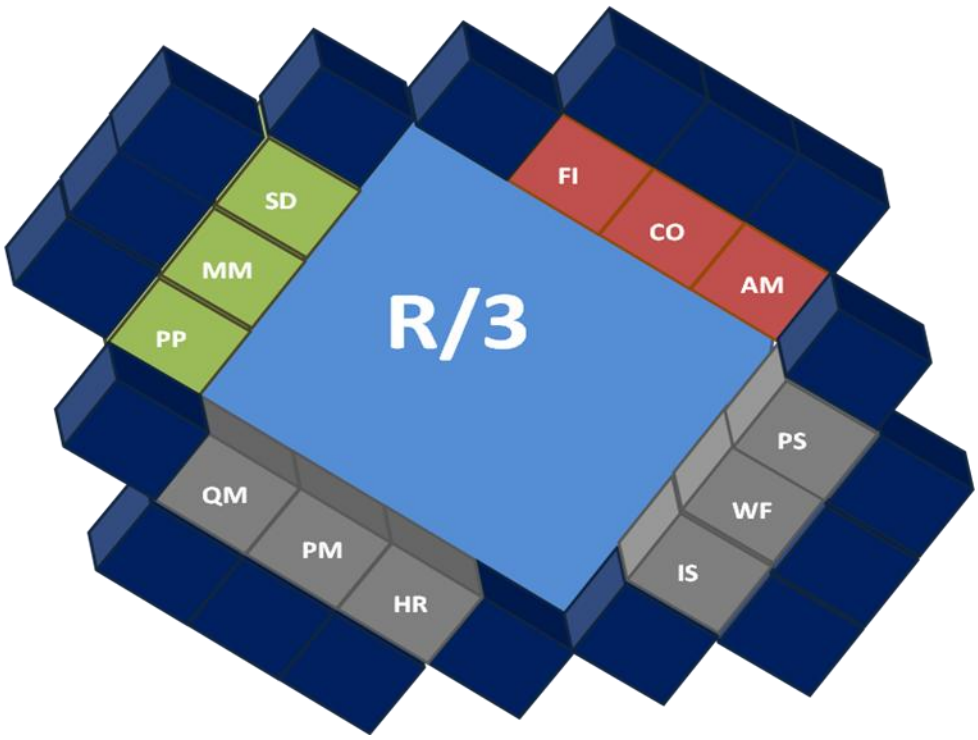


Fig. 2 ERP SAP/3

Finanzas

- FI: (Financial) Finanzas.

Submódulos:

- GL (General Ledger) Contabilidad general.
- AP (Accounts Payable) Cuentas por pagar.
- AR (Accounts Recivable) Cuentas por cobrar.

- CO: (Controling) Contabilidad de costos.
- AM (Assets Management) Administración de activos.
- CA (Contract Agreement) Gestión de contratos.

Ventas y Distribución

- SD: (Sales and Distribution) Ventas y Distribución.

Submódulos:

- LETRA (Logistic Execution Transport) Logística y ejecución de Transportes.
- LIS (Logistic Information System) Sistema de información de logística.

Almacenes e Inventarios

- MM: (Materials Management) Gestión de Materiales.

Submódulos:

- WM (Warehouse Management) Gestión de Almacenes.
- IM (Inventory Management) Gestión de Inventarios.

Producción

- PP: (Production Planning) Planificación de la producción.

Submódulos:

- PM (Plant Maintenance) Control de Piso.
- PI (Product Information) Gestión de Fórmulas.
- QM (Quality Management) Aseguramiento de calidad.
- E&HS (Enviroment and Healt Security) Gestión del medio ambiente.

Recursos Humanos

- HR (Human Resources) Recursos Humanos.

Submódulos:

- PA (Personal Administration) Administración de personal.
- PD (Personal Development) Desarrollo de Personal.
- PY (Payroll) Nómina.

Tecnología

- BC Basis Components.

Submódulos:

- STMS Sistema de Corrección y Transporte.
- ABAP Lenguaje nativo de SAP R/3 para programar.

1.5.3 Conclusiones del estudio del Estado del Arte

Una vez analizadas las propuestas de ERP cubanos e internacionales en el entorno libre y propietario y no haberse encontrado alguno adaptable, ya sea por las características de Cuba o las necesidades del cliente, se pudo concluir la necesidad de desarrollar uno propio.

Para realizar un software, del tipo que este sea, es importante definir un conjunto de elementos que ayudan a su desarrollo de manera correcta y tributan a su calidad final.

1.6 Metodologías de desarrollo de software

En todo proceso de desarrollo de software siempre puede existir el riesgo que este sea difícil de organizar y controlar, de ahí surge la necesidad de utilizar una Metodología de Desarrollo de Software, para así evitar resultados impredecibles y detección tardía de errores."*En un proyecto de desarrollo de software la metodología define Quién debe hacer, Qué, Cuándo y Cómo debe hacerlo, constituye la columna vertebral del proceso de desarrollo de software"* (Cataldi, 2000). Utilizar una metodología es de gran importancia para los desarrolladores, ya que permite, entre muchas otras cosas, mantener un orden y estructura sobre cómo hacer el software.

1.6.1 Metodología a seguir: Proceso Unificado de Desarrollo (RUP)

El Proceso Unificado es una de las metodologías que rige el proceso de desarrollo de software. "*Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software"* (Booch, 2000). RUP como marco de trabajo genérico, puede ser aplicado a cualquier sistema de software, a cualquier producto que se éste desarrollando y en cualquier área u organización de producción.

El proceso unificado plantea un proceso de desarrollo de software basado en componentes que ayuda a mejorar la productividad del equipo de trabajo, definiendo actividades, roles y responsabilidades dentro del equipo de desarrollo, desde los jefes de proyectos a los analistas y desde los desarrolladores a los probadores. Define el cómo y el cuándo se realizará cada actividad definida hasta alcanzar la meta planteada. Utiliza el Lenguaje Unificado de Modelado (UML) para la construcción de todos los esquemas de un sistema de software. Además, se encarga de verificar la calidad del software y controla los cambios que podrían realizarse. Ahora bien, existen características fundamentales que tiene RUP, las cuales son:

- Dirigido por casos de uso
- Centrado en la arquitectura
- Iterativo e Incremental
- Adaptable a proyectos de largo plazo.
- Destaca la importancia de lograr una buena captura de requisitos.

Los elementos expresados anteriormente fundamentan la selección de esta metodología de desarrollo para el proceso actual.

1.7 Desarrollo de aplicaciones Web

"*Una aplicación web es una aplicación informática distribuida cuya interfaz de usuario es accesible desde un cliente web, normalmente un navegador*" (Software, 2004).

Una aplicación Web es un sistema Web (servidor Web, red, protocolo, navegador) donde la entrada del usuario (entrada de datos y navegación) afecta el estado del negocio. Su arquitectura general es la de un sistema cliente/servidor. Las aplicaciones Web implementan lógica de negocios y su uso cambia el estado del negocio. Se puede decir que normalmente instalar una aplicación Web consiste en configurar los componentes del lado del servidor en la red y no es necesaria una instalación o configuración en el lado

cliente. El protocolo principal de comunicación en una aplicación Web es HTTP (Hyper Text Transfer Protocol, Protocolo de Transferencia de Hipertexto), el cual funciona normalmente desconectado, es decir, el cliente hace una petición al servidor, este la procesa y le devuelve el resultado, terminando la comunicación entre estos.

Las aplicaciones Web tienen varias ventajas sobre los programas de software descargables tradicionales. Estas son las principales (Graham, 2001):

- Compatibilidad multiplataforma**.** Las aplicaciones Web tienen un camino mucho más sencillo para la compatibilidad multiplataforma que las aplicaciones de software descargables. Varias tecnologías incluyendo Java, Flash, ASP y Ajax permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales.
- Actualización**.** Las aplicaciones basadas en Web están siempre actualizadas con el último lanzamiento sin requerir que el usuario tome acciones pro-activas.
- Inmediatez de acceso. Las aplicaciones basadas en Web no necesitan ser descargadas, instaladas y configuradas.
- Menos requerimientos de memoria. Las aplicaciones basadas en Web tienen muchas más razonables demandas de memoria RAM de parte del usuario final que los programas instalados localmente. Al residir y correr en los servidores del proveedor, a esas aplicaciones basadas en Web usa en muchos casos la memoria de las computadoras que ellos corren, dejando más espacio para correr múltiples aplicaciones del mismo tiempo sin incurrir en frustrantes deterioros en el rendimiento.
- Múltiples usuarios concurrentes. Las aplicaciones basadas en Web pueden ser utilizadas por múltiples usuarios al mismo tiempo. No hay más necesidad de compartir pantallas cuando múltiples usuarios pueden ver e incluso editar el mismo documento de manera conjunta.

1.7.1 Lenguajes de programación Web

Una vez determinado el tipo de aplicación a desarrollar es importante seleccionar el lenguaje de desarrollo. Los lenguajes de programación son herramientas que permiten crear programas y software, representan en forma simbólica y en manera de un texto los códigos que podrán ser leídos por una persona, en este caso se utiliza PHP.

1.7.1.1 Programación del lado del servidor: PHP

"El PHP es un lenguaje de script incrustado dentro del HTML. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas" (Heredia, 2001).

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas Web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting), pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+. PHP es un acrónimo recursivo que significa Hypertext Pre-processor (inicialmente PHP Tools, o, Personal Home Page Tools).

Ventajas de PHP

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables.
- Tiene manejo de excepciones (desde PHP5).

1.7.1.2 Programación del lado del cliente: JavaScript

JavaScript 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 nuestros usuarios. Javascript es soportado por la mayoría de los navegadores como Internet Explorer, Netscape, Opera, Mozilla Firefox, entre otros.

1.7.2 Tendencias actuales en el desarrollo de Aplicaciones Web

A pesar de las numerosas ventajas de los lenguajes de programación, como por ejemplo PHP, desarrollar hoy en día de la manera conocida como “tirar código a mano” no es para nada factible.

Una de las tendencias más generalizadas es el uso de pequeñas aplicaciones pre-elaboradas que sirven como base a otras aplicaciones más complejas, incluyen un conjunto de funcionalidades básicas útiles e implementan, por lo general, estilos y patrones arquitectónicos que organizan y facilitan la programación. Dichas aplicaciones son conocidas como frameworks y a continuación se describen algunas de las más usadas sobre PHP.

1.7.2.1 Zend Framework

Zend Framework es uno de los más usados para PHP y utiliza el estilo MVC como base de su funcionamiento. Es fácilmente integrable a las aplicaciones debido a su composición y a que contiene diferentes clases de gran utilidad, como por ejemplo en la búsqueda dinámica de ficheros a incluir o utilizar. Por lo anteriormente descrito, se propone su uso en la implementación del software.

Los frameworks, como aplicaciones de software que son, deben cumplir con requerimientos de arquitectura, semejante al resto. En este caso se encuentran los patrones de diseño y Zend Framework incluye en su implementación algunos de ellos.

Vista:

Implementa el patrón Decorator en la clase Zend_View, encargada de asignarle responsabilidades a objetos de manera dinámica y configurarlos con nuevos atributos.
Controlador:

Zend Framework tiene implementado el patrón Front Controller que implica que todas las solicitudes son dirigidas a un único script PHP que se encarga de instanciar al controlador frontal y redirigir las llamadas.

Además tiene una instancia única del controlador frontal disponible mediante el patrón Singleton para lograr una vía de entrada única a las solicitudes.

Modelo:

ZF provee una API para el acceso a dato conformada por un conjunto de clases que implementan los patrones Factory, Table Data Gateway y Row Data Gateway.

Por supuesto todos los patrones de diseño anteriormente mencionados implican características y comportamiento específico en los componentes de la arquitectura por lo que deben ser tenidos muy en cuenta.

A continuación se describen las clases Zend Framework:

Clase	Descripción
Zend_Controller_Action	De esta clase deben heredar todos los controladores de la aplicación, en ella se incluyen numerosas funcionalidades comunes.
vistaController	Representa el controlador del Caso de Uso en cuestión.
modelController	Es un intermediario entre el controlador y la clase del modelo. No debe heredar de Zend_Controller_Action, incluye las principales funciones para el manejo de los datos.

Tabla 1 Descripción clases Zend Framework

1.7.2.2 Zend_Ext Framework

Es un framework open Source, que está diseñado para PHP 5 y tiene buenas capacidades de ampliación. Es elaborado a partir de Zend Framework cumpliendo con todas sus características. Este trae de novedoso un controlador vertical para las acciones realizadas por las vistas hacia el controlador, un motor de reglas para las validaciones en el servidor, se le incluyó el IoC para la comunicación entre los módulos o componentes. Presenta integración con el ORM (Mapeador de Objetos Relacionales) Doctrine Framework para trabajo en la capa de abstracción a base de datos y el ExtJs Framework para el desarrollo de las vistas.

1.7.2.3 Doctrine Framework

Doctrine es un potente y completo sistema ORM para PHP con un DBAL (database abstraction layer, capa de abstracción de datos) incorporado.

Entre muchas otras cosas tiene la posibilidad de exportar una base de datos existente a sus clases correspondientes y también a la inversa, es decir, convertir clases (convenientemente creadas siguiendo las pautas del ORM) a tablas de una base de datos.

Su principal ventaja radica en poder acceder a la base de datos utilizando la programación orientada a objetos (POO) debido a que doctrine utiliza el patrón Active Record para manejar la base de datos, tiene su propio lenguaje de consultas (DQL, Doctrine Query Languaje) y trabaja de manera rápida y eficiente.

1.7.2.4 IoC

IoC es un concepto junto a unas técnicas de programación en las que el flujo de ejecución de un programa se invierte respecto a los métodos de programación tradicionales, en los que la interacción se expresa de forma imperativa, haciendo llamadas a procedimientos (procedure calls) o funciones. Tradicionalmente el programador especifica la secuencia de decisiones y procedimientos que pueden darse durante el ciclo de vida de un programa mediante llamadas a funciones. En su lugar, en la inversión de control, se especifican respuestas deseadas a sucesos o solicitudes de datos concretas, dejando que algún tipo de entidad o arquitectura externa lleve a cabo las acciones de control que se requieran en el orden necesario y para el conjunto de sucesos que tengan que ocurrir. El flujo habitual se da cuando es el código del usuario quien invoca a un procedimiento de una librería. La inversión de control sucede cuando es la

librería la que invoca el código del usuario. Típicamente sucede cuando la librería es la que implementa las estructuras de alto nivel y es el código del usuario el que implementa las tareas de bajo nivel.

1.7.2.5 ExtJS

No sólo es posible encontrar frameworks que apoyan la programación del lado del servidor, una correcta y sencilla interfaz de usuario que sea personalizable brinda grandes ventajas y posibilita ganar en aceptación. ExtJS es un framework para JavaScript muy utilizado en el desarrollo de aplicaciones Web con AJAX (Ver epígrafe 1.7.2.7). Tiene una librería inmenza que permite configurar las interfaces Web de manera semejante a aplicaciones desktop .

Trae incluidos la mayoría de los controles de los formularios Web incluyendo grids para mostrar datos y elementos semejantes a la programación desktop, como los formularios, paneles, barras de herramienta, menus y muchos otros. Dentro de su librería de componentes incluye componentes para el manejo de datos, lectura de XML, lectura de datos JSON (ver epígrafe 1.7.2.6) e implementaciones basadas en AJAX. Presenta el uso de JavaScript con una programación orientada a objetos.

A continuación se describen las clases EXTJS:

Clase	Descripción
ext-base	Encargada del manejo de las solicitudes y respuestas, trabajo con ajax y manejo de componentes de EXT. Está incluida en el paquete original.
ext-all	Es la encargada de la creación de los componentes visuales de la vista. Está incluida dentro de las clases que trae EXT JS.
Vista	Representa la vista que se muestra al usuario.
js_vista	Fichero js con las funciones Java Script asociadas a la vista. Aquí se establece la referencia a las clases de EXT.

Tabla 2 Descripción clases EXTJS

Entre las bondades de este framework se tiene:

- Capas.
- Grid.
- PlantillasX.
- Vistas de Datos.
- Sirve de puente entre las librerías JS más usadas (Prototype, JQuery, YUI). Debido a que se inició como una extensión de YUI, esta presenta una cierta ventaja de compatibilidad respecto a las otras dos.

Es soportado por los navegadores:

- Internet Explorer 6+
- FireFox 1.5+ (PC, Mac)
- Safari 2+
- Opera 9+ (PC, Mac)

1.7.2.6 JSON.

JSON (JavaScript Object Notation) es un formato sencillo para intercambiar datos. Consiste básicamente en un array asociativo de JavaScript que se utiliza para incluir información del objeto. JSON ofrece dos grandes ventajas para las interacciones AJAX: es muy fácil de leer en JavaScript y puede reducir el tamaño en bytes de la respuesta del servidor.

El formato JSON es el más adecuado para la respuesta del servidor cuando la acción AJAX debe devolver una estructura de datos a la página que realizó la llamada, de forma que se pueda procesar con JavaScript. Este mecanismo es útil por ejemplo cuando una sola petición AJAX debe actualizar varios elementos en la página. Para actualizar varios elementos, la respuesta AJAX podría consistir en una única cabecera JSON, mediante algunas pocas instrucciones de JavaScript se puede interpretar la respuesta del servidor y actualizar varios elementos de la página de forma seguida. El protocolo HTTP permite que

el objeto JSON se pueda enviar como una cabecera de la respuesta. Como la respuesta no tiene ningún contenido, la acción envía solo la cabecera de forma inmediata. De esta forma, se evita completamente la capa de la vista y es muy rápido pero además con una respuesta más pequeña. JSON se ha convertido en un estándar en el desarrollo de aplicaciones Web. Los servicios Web proponen la utilización de JSON en vez de XML para permitir la integración de servicios en el navegador del usuario en vez de en el servidor. El formato JSON es seguramente la mejor opción para el intercambio de información entre el servidor y las funciones JavaScript**.**

1.7.2.7 AJAX

AJAX, acrónimo de JavaScript asíncrono y XML, es una técnica de desarrollo Web para crear aplicaciones interactivas. Se ejecuta en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma. AJAX es una combinación de tres tecnologías ya existentes:

- XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.
- Modelo de Objetos de Documento (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
- El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor Web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto iframe en lugar del XMLHttpRequest para realizar dichos intercambios.
- XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML preformateado, texto plano y JSON

DHTML, LAMP o SPA, AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de estas que trabajan conjuntamente.

1.7.2.8 CSS

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

y su presentación presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo (también llamados documentos semánticos). Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes. Mientras que el lenguaje HTML/XHTML se utiliza para marcar los contenidos, es decir, para designar lo que es un párrafo, lo que es un titular o lo que es una lista de elementos, el lenguaje CSS se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista.

Las ventajas de utilizar CSS son:

- Control centralizado de la presentación de un sitio Web completo con lo que se agiliza de forma considerable la actualización del mismo.
- Los Navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio Web, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario. Por ejemplo, para ser impresa, mostrada en un dispositivo móvil, o ser leída por un sintetizador de voz.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

1.7.3 IDEs para el desarrollo

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 GUI. Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. En algunos lenguajes, un IDE puede funcionar como un sistema en tiempo de ejecución, en donde se permite

utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto. A continuación se detallan las características del IDE que se utiliza.

1.7.3.1 Zend Studio

Zend Studio o Zend Development Environment es un completo entorno integrado de desarrollo para el lenguaje de programación PHP. Está escrito en Java, y está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux.

Junto con su contraparte Zend Platform, son la propuesta de Zend Technologies para el desarrollo de aplicaciones Web utilizando PHP, actuando Zend Studio como la parte cliente y Zend Platform como la parte servidora. Se trata en ambos casos de software comercial, pero presenta numerosas características ventajosas como son:

- No requiere la instalación previa de PHP ni del entorno de ejecución de Java.
- Soporte para PHP 4 y PHP 5.
- Resaltado de sintaxis, autocompletado de código, ayuda de código y lista de parámetros de funciones y métodos de clase.
- Plegado de código (comentarios, bloques de phpDoc, cuerpo de funciones y métodos e implementación de clases).
- Inserción automática de paréntesis y corchetes de cierre.
- Emparejamiento (matching) de paréntesis y corchetes (si se sitúa el cursor sobre un paréntesis (corchete) de apertura (cierre), Zend Studio localiza el correspondiente paréntesis (corchete) de cierre (apertura)).
- Detección de errores de sintaxis en tiempo real.
- Funciones de depuración: Botón de ejecución y traza, marcadores, puntos de parada (breakpoints), seguimiento de variables y mensajes de error del intérprete de PHP. Permite también la depuración en servidores remotos (requiere Zend Platform).
- Instalación de barras de herramientas para Internet Explorer y Mozilla Firefox (opcional).
- Soporte para gestión de grandes proyectos de desarrollo.
- Manual de PHP integrado.
- Soporte para control de versiones usando CVS o Subversion (a elección del desarrollador).

- Cliente FTP integrado.
- Soporte para navegación en bases de datos y ejecución de consultas SQL.

Zend Studio fue diseñado para usarse con el lenguaje PHP, sin embargo ofrece soporte básico para otros lenguajes Web, como HTML, JavaScript y XML.

1.7.4 Herramientas CASE

Las Herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) son las aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas herramientas contribuyen de manera directa en todos los aspectos del ciclo de vida del desarrollo del software, en tareas como el diseño del proyecto, cálculo de costes, generación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores, entre otras. Cada una de estas herramientas persigue nueve objetivos principales:

- Mejorar la productividad en el desarrollo y mantenimiento del software.
- Aumentar la calidad del software.
- Mejorar el tiempo y coste de desarrollo y mantenimiento de los sistemas informáticos.
- Mejorar la planificación de un proyecto
- Aumentar la biblioteca de conocimiento informático de una empresa ayudando a la búsqueda de soluciones para los requisitos.
- Apoyar en documentación, generación de código, pruebas de errores y gestión del proyecto.
- Ayuda a la reutilización del software, portabilidad y estandarización de la documentación.
- Gestión global en todas las fases de desarrollo de software con una misma herramienta.
- Facilita el uso de las distintas metodologías propias de la ingeniería del software.

1.7.4.1 Herramienta seleccionada: Visual Paradigm

Visual Paradigm 6.0 es una herramienta CASE orientada a UML, soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, prueba y despliegue. Permite realizar diferentes tipos de diagramas de clases. Visual Paradigm genera toda la documentación de lo que se hace, cumpliendo con estándares establecidos. Brinda la posibilidad de generar código a partir de los diagramas para plataformas como .Net, Java y PHP, así como obtener diagramas a partir de código. Esta

es precisamente una gran ventaja, puesto que el sistema será desarrollado en PHP. El análisis textual es una técnica útil y práctica para la captura de los requisitos del sistema y la identificación de las clases candidatas, Visual Paradigm es una de las pocas herramientas CASE que soporta el análisis textual. Tiene disponibilidad en múltiples plataformas.

1.7.5 Sistemas de Gestión de Base de Datos

Los Sistemas de gestión de base de datos (SGBD) son un prototipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan, facilitándole al usuario las herramientas que le permitan manipular o manejar de manera clara, sencilla y ordenada un conjunto de datos. Estos sistemas tienen la ventaja de permitir la facilidad de manejo de grandes volúmenes de información a gran velocidad en muy poco tiempo, no hay probabilidad de duplicidad de información y permiten la comprobación de información en el momento de introducir la misma. Brindan seguridad de información (acceso a usuarios autorizados), de modificaciones, inclusiones y consulta.

No se concibe la realización de un sistema informático que no maneje información y al hacerlo es muy importante la utilización de un SGBD para hacerlo de manera eficiente.

1.7.5.1 PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en el proyecto POSTGRES, de la universidad de Berkeley. Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido más tarde en otros sistemas de gestión comerciales. PostgreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos y funciones.

Tiene más de 15 años de activo desarrollo y arquitectura probada que se ha ganado una muy buena reputación por su confiabilidad e integridad de datos, funciona en todos los sistemas operativos importantes, incluyendo Linux, UNIX y Windows. Tiene soporte total para transacciones, disparadores, vistas, procedimientos almacenados (en múltiples lenguajes) y almacenamiento de objetos de gran tamaño. Se destaca en ejecutar consultas complejas, consultas sobre vistas, subconsultas y joins de gran tamaño. Permite la definición de tipos de datos personalizados. Incluye un modelo de seguridad completo. Permite distribuir una base de datos en distintos discos. Es altamente escalable tanto en la cantidad de datos que puede manipular como en la cantidad de usuarios concurrentes que puede atender.

Se pueden enumerar otras características importantes:

- Soporte nativo para los lenguajes más populares: PHP, C, C++, Perl, Python.
- Soporte de protocolo de comunicación encriptado por SSL.
- Extensiones para alta disponibilidad, nuevos tipos de índices, datos espaciales, minería de datos.

1.7.6 Servidor Web

Un servidor Web es un programa que sirve datos en forma de páginas Web, hipertextos o páginas HTML: textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonidos. La comunicación de estos datos entre cliente y servidor se hace por medio un protocolo, concretamente del protocolo HTTP. Con esto, un servidor Web se mantiene a la espera de peticiones HTTP, que son ejecutadas por un cliente HTTP; lo que se conoce como un navegador Web.

1.7.6.1 Apache

El servidor HTTP Apache es un servidor Web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux.), Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Muchas aplicaciones Web están diseñadas asumiendo como ambiente de implantación a Apache, o que utilizarán características propias de este servidor Web. Apache es el componente de servidor Web en la popular plataforma de aplicaciones LAMP, junto a MySQL y los lenguajes de programación PHP/Perl/Python (y ahora también Ruby).

Teniendo en cuenta los elementos anteriormente mencionados, se selecciona Apache en su versión 2.* para el soporte de la aplicación.

1.7.7 Otras Herramientas de apoyo al desarrollo.

1.7.7.1 SVN (SubVersion)

Subversion es un software de sistema de control de versiones diseñado específicamente para reemplazar al popular CVS, el cual posee varias deficiencias. Es software libre bajo una licencia de tipo Apache/BSD y se le conoce también como svn por ser ese el nombre de la herramienta de línea de comandos. Una característica importante de Subversion es que, a diferencia de CVS, los archivos versionados no tienen

cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo.

1.7.7.2 TortoiseSVN

TortoiseSVN es un cliente Subversion, implementado como una extensión al shell de Windows. Es software libre liberado bajo la licencia GNU GPL.

Características:

- Integración con el shell de Windows.
- Puede ser usado sin un entorno de desarrollo.
- Pequeñas imágenes decoran los íconos de los archivos mostrando qué archivos o directorios necesitan ser enviados al repositorio.
- Disponible en 28 idiomas diferentes.
- Maneja el mostrar la diferencia de documentos de Office tales como los creados con Microsoft Word.

1.7.8 Navegadores Web

Un navegador Web o explorador Web es una aplicación software que permite al usuario recuperar y visualizar documentos de hipertexto, comúnmente descritos en HTML, desde servidores Web de todo el mundo a través de Internet. Cualquier navegador actual permite mostrar o ejecutar gráficos, secuencias de vídeo, sonido, animaciones y programas diversos además del texto y los hipervínculos o enlaces. En este caso el explorador usado es Mozilla Firefox.

1.7.8.1 Mozilla Firefox

Uno de los navegadores más utilizados actualmente es el Mozilla Firefox. Este permite abrir por defecto las nuevas páginas Web en pestañas. Cada una de esas pestañas tiene su propio botón de cerrado. Restaurar sesión es una de sus ventajas, si Firefox tiene que reiniciarse o cerrarse, cuando se inicie de nuevo aparecerán nuevamente las páginas mostradas antes de cerrarse. Posee un corrector ortográfico integrado para evitar que se cometan errores en las entradas de información que haga el usuario. Tiene una sugerencia de búsqueda que se va desplazando a medida que se introduce el texto a buscar. El bloqueador de Firefox avisa cuando se bloquean ventanas emergentes mediante una barra informativa o un icono en la parte inferior de la pantalla. Firefox mantiene protección contra programas espías,

impostores y spammers, usando el poder de una comunidad apasionada que trabaja 24 horas al día, 7 días a la semana. La protección antiphishing lleva la seguridad de Firefox a un nuevo nivel. Cuando es encontrada una página Web sospechosa de fraude, Firefox advierte al usuario y ofrece una página de búsqueda para encontrar la página Web buscada.

El programa es multiplataforma y está disponible en versiones para Microsoft Windows, Mac OS X y GNU/Linux. El código fuente de Firefox está disponible libremente bajo la triple licencia de Mozilla como un programa libre y de código abierto. Estas y muchas otras posibilidades brinda este navegador, por lo que se propone su uso para interactuar con el sistema (Ver epígrafe 2.2.1).

1.7.9 Arquitectura

La arquitectura de software es uno de los grandes temas actuales que han dominado prácticamente la década del 90. El objetivo principal de la Arquitectura del Software es aportar elementos que ayuden a la toma de decisiones y, al mismo tiempo, proporcionar conceptos y un lenguaje común que permitan la comunicación entre los equipos que participen en un proyecto.

La definición oficial de Arquitectura del Software es la IEEE 1471-2000:

"*La arquitectura de software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución*".

1.7.9.1 Arquitectura Cliente/Servidor.

La arquitectura cliente/servidor es una forma de dividir el desarrollo de sistemas de información en procesos independientes separando la interfaz de usuario (nivel de presentación) de la gestión de la información (nivel de gestión de datos). Se denomina cliente al proceso que inicia el diálogo o solicita los recursos y servidor al proceso que responde a las solicitudes. Por lo general las aplicaciones Web funcionan bajo esta arquitectura.

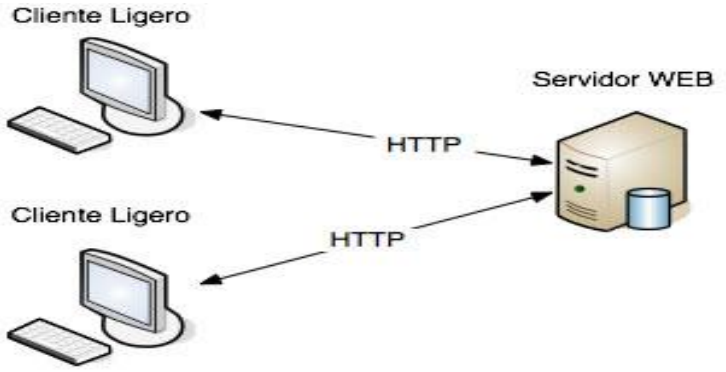


Fig. 3 Arquitectura Cliente /Servidor

En este modelo el servidor contiene los recursos a compartir para varios usuarios, y el cliente es la parte que permanece individual para cada usuario.

Entre las principales características de la arquitectura cliente/servidor se pueden destacar las siguientes:

- El servidor presenta a todos sus clientes una interfaz única y bien definida.
- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente.

Ventajas

- Centralización del control: los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema.
- Escalabilidad: se puede aumentar la capacidad de clientes y servidores por separado. Se reduce el tráfico de red considerablemente. Idealmente, el cliente se comunica con el servidor utilizando un protocolo de alto nivel de abstracción.

1.7.9.2 Modelo–Vista–Controlador (MVC)

Es un estilo basado en un patrón de diseño que plantea la separación de diferentes clases en dependencia de la función que realizan de modo tal que sea posible manejar dinámicamente la forma en

que se procesan solicitudes y se gestiona la manera en que se muestran resultados al usuario final. En otras palabras separa la presentación del dominio de la aplicación. A simple vista ya hay ventajas.

Es un principio que utilizan muchos frameworks para basar su funcionamiento, la idea de Don´t call us, we´ll call you (No nos llame, nosotros lo llamaremos a usted). Esa idea ha hecho que los frameworks que implementan MVC se puedan usar sencillamente implementando interfaces o extendiendo de una clase abstracta que brindan.

A continuación se muestra una representación del funcionamiento del patrón MVC que es la base del estilo arquitectónico del mismo nombre. Representa una manera muy sencilla de gestionar la presentación de datos a los usuarios finales de la aplicación.

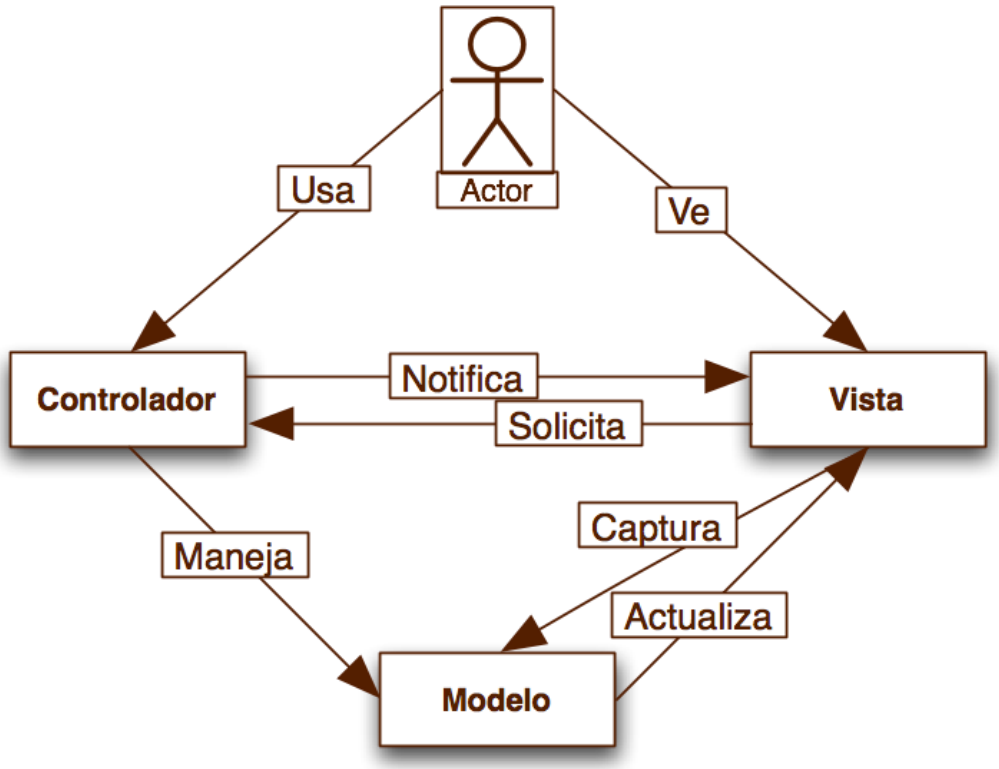


Fig. 4. Modelo - Vista – Controlador

Precisamente es utilizando este estilo, que es implementada la aplicación propuesta, basados fundamentalmente, en el uso de un framework sobre MVC para PHP.

1.8 Estándares de codificación

Las convenciones o estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código. La utilización de los estándares permite reducir el tiempo de desarrollo, el coste y el esfuerzo. Es por ello que para este proyecto se establece un estándar de codificación para asegurar de que todos los programadores del proyecto trabajen de forma coordinada. A continuación se explican los estándares utilizados.

1.8.1 Estándares de Nomenclatura

1. Nomenclatura de las clases

   Los nombres de las clases comienzan con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing. Con sólo leerlo se reconoce el propósito de la misma.

   Ejemplo: GestionarUsuario

2. Nomenclatura según el tipo de clases

   - Clases controladoras

   Las clases controladoras después del nombre llevan la palabra Controller.

   Ejemplo: GestionarUsuarioController

   - Clases de los modelos

     - Business (Negocio)
       Las clases que se encuentran dentro de Business después del nombre llevan la palabra Model.
       Ejemplo: GestorDenDespachoModel

   - Clases del framework

Como parte del marco de desarrollo de Zend existe el Zend_Loader (Cargador) que, además del cumplimiento de ciertas normas para la nomenclatura de las clases garantiza que, a partir de una ruta de inclusión, este sea el responsable de la inclusión de los recursos requeridos en el proceso.

Ejemplo:

Los nombres de las clases contienen la dirección donde se encuentran, de la siguiente forma, si se tiene una clase llamada Condado en la siguiente estructura Cuba/VC/SantaClara/Condado.php entonces un identificador de la misma debe ser Cuba_VC_SantaClara_Condado lo que garantiza que a través de una casuística particular el cargador localice estos recursos.

1.8.2 Nomenclatura de las funciones

El nombre a emplear para las funciones se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto se empleará notación CamelCasing, y con sólo leerlo se reconoce el propósito de la misma.

Ejemplo:

insertarProducto

- En caso de ser una acción de la clase controladora se le pone el nombre y seguida la palabra Action.
  Ejemplo: insertarProductoAction

1.8.3 Nomenclatura de las variables

El nombre a emplear para las variables se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto se empleará notación CamelCasing, y comenzando con un prefijo según el tipo de datos.

Ejemplo: arrProducto

1.8.4 Prefijos para los tipos de datos

Los prefijos a utilizar en la creación de variables serán los siguientes:

Tipos de Datos	Prefijos
Arreglos	Arr
Objetos	Obj
Enteros	Int
Cadena	Str
Float	Flt
Boolean	Boo

Tabla 3 Prefijos para tipos de datos

1.9 Conclusiones Parciales

- Tras un análisis detallado de los procesos asociados al despacho en almacenes, sus características esenciales, los elementos que identifican a los ERP, así como el tipo de sistema informático que se utiliza como apoyo en estos casos y las tendencias mundiales para su desarrollo, se tienen elementos consistentes de aspectos esenciales a tener en cuenta durante la implementación.
- El estudio de las principales características del lenguaje a utilizar, la tecnología afín, los frameworks, sienta las bases necesarias para pasar a una etapa donde se comience a hacer realidad el software, es decir, se estará en condiciones de codificar y por último, probar el resultado.

Capítulo 2: Descripción de la solución propuesta

2.1 Introducción

Teniendo todos los elementos necesarios, se comienza entonces la etapa de cierre del proceso de desarrollo: la implementación. Si bien en este punto no es posible tener claro cómo será el software y cómo funcionará realmente, sí se tienen presentes las características de sus componentes esenciales, la manera en que se relacionan y cómo interactúan unos con otros.

Es muy importante en este momento realizar una profunda valoración del diseño propuesto por los analistas del sistema, exponiendo las principales ventajas y deficiencias del mismo; fundamentalmente los componentes físicos derivados de las clases. Además, se necesitan analizar las posibilidades que proporcionan los frameworks y librerías utilizadas en la programación de la aplicación, sus características y componentes, con el objetivo de facilitar la modelación.

Una vez que se identifican los elementos anteriormente mencionados, se está en condiciones de realizar el modelo de implementación, basado en los diagramas de componentes, que describen cada componente físico del software y su relación con los demás. Elementos fundamentales de este diagrama son los subsistemas que integran el sistema general y sus partes esenciales. Todos estos elementos se describen entre los epígrafes a continuación.

2.2 Valoración crítica de los artefactos generados por los analistas

El trabajo realizado por los analistas, permitió un mejor entendimiento de los procesos y funcionalidades, facilitando una buena comprensión del problema. De esta forma se garantizó una correcta identificación de las clases y funcionalidades a implementar. A partir de la descripción detallada de los requisitos funcionales entregados, se traza la estrategia de trabajo para el comienzo de la implementación.

2.2.1 Especificación de los requisitos del software

La Ingeniería de Requerimientos cumple un papel primordial en el proceso de producción de software, ya que enfoca un área fundamental: la definición de lo que se desea producir. Su principal tarea consiste en la generación de especificaciones correctas que describan con claridad, sin ambigüedades, en forma consistente y compacta, el comportamiento del sistema; de esta manera, se pretende minimizar los problemas relacionados al desarrollo de sistemas.

Los requerimientos pueden dividirse en requerimientos funcionales y no funcionales. Los requerimientos funcionales definen las funciones que el sistema será capaz de realizar, describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Por otra parte, los requerimientos no funcionales tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad y estándares.

Luego del análisis realizado, se identificaron las siguientes funcionalidades o requisitos funcionales:

- R1- Aprobar autorización de entrega.
- R2- Buscar autorización de entrega y documentos de salida.
  - R2.1- Buscar documentos de entrega.
  - R2.2- Búsqueda avanzada de documentos de entrega.
- R3- Cancelar asignaciones de la autorización de entrega.
- R4- Gestionar autorización de entrega.
  - R4.1- Adicionar autorización de entrega.
  - R4.2- Adicionar clientes a la autorización de entrega.
  - R4.3- Eliminar cliente de la autorización de entrega.
  - R4.4- Modificar clientes de la autorización de entrega.
  - R4.5- Eliminar autorización de entrega.
  - R4.6- Modificar autorización de entrega.
  - R4.7- Confirmar autorización de entrega.
  - R4.8- Cancelar el estado de la autorización de entrega.
  - R4.9- Cumplimiento de la autorización de entrega.
  - R4.10- Ver estado de ejecución de la autorización de entrega.
- R5- Gestionar documento de salida.
  - R5.1- Generar conduce.
  - R5.2- Modificar conduce.
  - R5.3- Eliminar conduce.

- R5.4- Confirmar conduce.
- R5.5- Cancelar estado del conduce.
- R5.6- Contabilizar conduce.
- R5.7- Generar vale de entrega o devolución.
- R5.8- Modificar vale de entrega o devolución.
- R5.9- Eliminar vale de entrega o devolución.
- R5.10- Confirmar vale de entrega o devolución.
- R5.11- Cancelar estado del vale de entrega o devolución.
- R5.12- Contabilizar vale de entrega o devolución.
- R5.13- Generar transferencia.
- R5.14- Modificar transferencia.
- R5.15- Eliminar transferencia.
- R5.16- Confirmar trasferencia.
- R5.17- Cancelar el estado de la trasferencia.
- R5.18- Contabilizar transferencia.
- R5.19- Generar transferencia.
- R5.20- Modificar transferencia.
- R5.21- Eliminar transferencia.
- R5.22- Confirmar trasferencia.
- R5.23- Cancelar el estado de la trasferencia.
- R5.24- Contabilizar transferencia.

- R6- Gestionar productos autorizados.
  - R6.1- Adicionar productos a la autorización de entrega.
  - R6.2- Eliminar productos de la autorización de entrega.
  - R6.3- Registrar productos del informe de recepción.
  - R6.4- Modificar cantidades del producto autorizado.
  - R6.5- Buscar productos de la autorización de entrega.
- R7- Gestionar productos del documento de salida.
  - R7.1- Registrar cantidades a entregar.
  - R7.2- Modificar cantidades a entregar.

  - R7.3- Buscar productos del documento de salida.
- R8- Importar autorización de entrega.
- R9- Visualizar autorización de entrega y documentos de salida.
  - R9.1- Visualizar documentos de entrega.

Para hacer posible el cumplimiento de los requerimientos antes descritos de manera eficiente, así como para lograr una mejor aceptación por parte de los clientes, se identificaron algunas capacidades y características (requisitos no funcionales) que debe tener el sistema.

- Apariencia o interfaz externa:
  - El sistema debe tener una interfaz fácil de usar y amigable para que pueda ser utilizada sin mucho entrenamiento por el usuario.
  - Empleo de imágenes identificadas con el negocio donde se implantará el sistema y colores agradables a la vista, siendo estos claros.
- Usabilidad:
  - El sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora.
- Rendimiento:
  - Los tiempos de respuesta y velocidad de procesamiento de la información serán rápidos, no mayores de 5 segundos para las actualizaciones y 20 para las recuperaciones.
- Portabilidad:
  - El sistema debe ser multiplataforma, importante el correcto funcionamiento en Linux y Windows.
- Seguridad:
  - Autenticación (Contraseña de acceso).
  - Autorización (Atribución a los usuarios respecto a sus funciones de trabajo).
  - Implementación de auditoría (Registrar la confirmación de cada operación efectuada por el usuario que afecte los registros contables).
  - La atención al sistema incluyendo, el mantenimiento de las bases de datos así como la salva de la información se realizarán de forma centralizada por el administrador.

- Políticos culturales:
  El sistema solo podrá ser utilizado en territorio cubano y por las entidades autorizadas.
- El producto debe respetar los términos empleados normalmente por los especialistas en el tema de la esfera que se automatiza.
- Legales:
  - El sistema está avalado por los tres documentos rectores emitidos en el país para la certificación y validación de los sistemas contables:
  - La Resolución Conjunta de los ministerios de Finanzas y Precios de fecha 8.04.04.
  - La Resolución 340 del Ministerio de Finanzas y Precios de fecha 8.12.04.
  - La Resolución No. 12 del Ministerio de la Informática y las Comunicaciones de fecha 24.01.05.

- Software:

  Para el cliente:
  - Navegador Mozilla Firefox.
  - Sistema operativo Linux, Windows 98 o superior.

  Para el servidor:
  - Sistema operativo Windows Advanced Server (2000 o superior) o Linux en cualquiera de sus distribuciones.
  - Un servidor Apache 2.0 o superior con módulo PHP 5.0 disponible, este debe estar configurado con la extensión pgsql incluida.
  - Un servidor de base de datos PostgreSQL 8.0 o superior.
- Hardware:

  Para el servidor:
  - Requerimientos mínimos: Procesador Pentium III a 1GHz de velocidad de procesamiento y 1Gb de memoria RAM (Memoria de Acceso Aleatorio de sus siglas en ingles Random Access Memory).
  - Al menos 40Gb de espacio libre en disco duro.
  - Tarjeta de red.

  Para el cliente:

  - Requerimientos mínimos: Procesador Pentium II a 133Mhz con 128 Mb de memoria RAM.
  - Tarjeta de red.
- Restricciones para el diseño e implementación:
  - Emplear como servidores Web y de bases de datos Apache y PostgreSQL respectivamente.
  - Utilizar como lenguaje del lado del servidor al PHP 5.0 o superior y del lado del cliente el JavaScript.

2.3 Propuesta de solución

Luego de analizar detenidamente lo anteriormente expuesto, se concluye que es factible implementar una aplicación Web, basada en el uso del estilo arquitectónico Modelo – Vista – Controlador, usando el lenguaje de programación PHP y como soporte para los datos el Sistema Gestor de Bases de Datos PostgreSQL. Se propone utilizar además, para facilitar el trabajo, el Zend Framework, desarrollado en PHP bajo el estilo arquitectónico MVC y muy utilizado en la actualidad. Dicha aplicación, automatiza los procesos que integran el despacho en los almacenes dentro del entorno empresarial cubano y la documentación asociada a su desarrollo sirve de base para futuras implementaciones, mantenimientos o mejoras al software desarrollado con este mismo propósito.

2.4 Procesos objeto de automatización

Para el proceso de entrega de medios materiales se quiere automatizar la elaboración y aprobación de Planes de Abastecimiento y la entrega de sus productos por diferentes criterios (genérico, subgenérico, específico y surtido), así como la generación de los documentos de entrega para dichos planes.

También se desea informatizar la recuperación donde se obtenga el estado del cumplimiento de las Autorizaciones de Entrega en la entidad suministradora. Además buscar por diferentes criterios dichas autorizaciones.

2.5 Componentes o módulos ya existentes

A continuación se describen algunos componentes que favorecieron la implementación de este trabajo.

2.5.1 Componente Documento

Dentro de los componentes que se utilizan en la implementación del módulo de Despacho se encuentra el componente documento, el cual tiene como función la de gestionar los documentos, siendo este el padre de los documentos de despacho. Dichos documentos son iguales a documento padre, pero con diferentes particularidades.

2.5.2 Componente Producto

Es el componente encargado de la gestión de los productos dentro del almacén, ya sea los que están nomenclados, como los nuevos que se incluyen. Este servicio brinda todos los servicios necesarios para operar sobre los productos, que a través de los movimientos se relacionan con los documentos. Los productos pueden ser equipos, no equipos o munición, también tienen categorías según la calidad del mismo, además se registra la nacionalidad, el tipo de control, el número de pieza y otros atributos que son importantes para los almacenes.

2.5.3 Componente Movimiento

El componente Movimiento tiene una función muy importante, ya que es el encargado de la unión entre el documento y los productos. Dicho componente facilita todas las operaciones que son necesarias para operar con los movimientos de ese documento. Un movimiento es una copia que se le hace al producto para no afectar los valores reales del mismo y una vez contabilizado el documento serán afectados los valores de producto con los del movimiento. Dentro del componente de movimiento se encuentra también el movimiento de los equipos, para en el caso de la baja, la recepción y la apertura incluirle los números de series al producto que sea equipo.

2.5.4 Cliente

El componente cliente cumple una función muy importante, ya que es el encargado del manejo de todos los componentes de la empresa. Dicho componente facilita todas las operaciones que son necesarias para operar con los mismos.

2.5.5 Nomenclador de conceptos de planes

Este nomenclador permite realizar órdenes de despachos por concepto, es decir se llama concepto a cualquier despacho de medios materiales, previamente definida por la entidad distribuidora. Se podría poner como ejemplo de concepto a Batalla de Ideas, Marcha o Destacado.

2.5.6 Clases

A continuación se describen de las clases principales:

Nombre: GestordendespachoController	
Tipo de clase: Controladora	
Atributo	**Tipo**
$model	GestordendespachoModel
$comun	comunModel
$bloqueo	GestBloqueoModel
Para cada responsabilidad	
Nombre.	**Descripción.**
init()	Constructor de la clase.
insertarproductosordenAction()	Acción para insertar los productos al documento.
cargarclientereporte108AAction()	Función para devolver un arreglo de clientes según 108A.
cancelarestadoAction()	Acción para cancelar estado del documento.
verestadoejecucionAction()	Acción para ver el estado en ejecución del documento.
eliminardespachoAction() modificardespachoAction() adicionarordendespAction()	Acción para la gestión del documento.
aprobarAction()	Acción para aprobar el documento.
aprobarcancelarAction()	Acción para aprobar la cancelación del documento.
modificarclientesordenAction()	Acción para modificar los clientes de la orden.
importarordendespachoAction()	Acción para importar datos de una orden de despacho.
verificarclientesAction()	Acción para verificar que los clientes importados existan en el nomenclador de cliente.
cancelarasignacionesAction()	Acción para cargar los productos que tienen las cantidades ordenadas mayor que la cantidades entregadas para el cancelar asignaciones del 108.

cargarasignacionesAction()	Acción para cargar los productos que tienen las cantidades ordenadas mayor que la cantidades entregadas para el cancelar asignaciones del 108ª.
cargararbolclientesAction()	Acción para cargar los clientes para el cancelar asignaciones del 108ª.
insertarcantcancelarAction()	Acción para insertar la cantidad a cancelar.
cargaruserfechaAction()	Acción para en el cancelar asignaciones ver si el rol es Jefe especialidad mostrar la fecha de vencimiento editable sino no. Se retorna 1 ó 2.
cargarimportetotalAction()	Acción para cargar el importe total de una orden o de un cliente.
confirmarcancelarasigAction()	Acción para confirmar la cancelación.
constgridclienteplanabastAction()	Acción para construir el Grid editable con las columnas dinámicas del Plan de Abastecimiento.
cargarprodplanabastAction()	Acción para cargar los productos para el plan de abastecimiento.
adicionarinformeAction()	Acción para adicionar un nuevo informe.
selclienteAction()	Acción para seleccionar los clientes al plan de abastecimiento.
cantdisponibleAction()	Acción para conocer la cantidad de productos disponibles.
adicionarproddespachoAction()	Acción para adicionar productos a la orden.
buscardocelabprepAction()	Acción para buscar los documentos en las fases de Elaborados y Preparados.
addprodplanabastAction()	Acción para adicionar los productos al plan de abastecimiento.
insertarprodplanabastAction()	Acción para insertar productos al plan de abastecimiento.
eliminarprodplanAction()	Acción para eliminar los productos del documento.

Tabla 4 Descripción de la clase GestordendespachoController

Nombre: GestordendespachoModel	
Tipo de clase: Auxiliar	
Atributo	**Tipo**
$data	ReaderModel

$mensaje	String
$comun	comunModel
Para cada responsabilidad	
Nombre.	**Descripción.**
GestordendespachoModel()	Constructor de la clase.
Cargardatosencabdespacho()	Acción para cargar los datos del encabezado.
adicionarOrdenDesp()	Acción para adicionar una orden de despacho.
eliminarOrden()	Acción para eliminar una orden de despacho.
CargarDatosEncabezadoModificar()	Acción cargar los datos del encabezado cuando voy a modificar.
modificarOrden()	Acción para modificar la orden.
Modificarclientesorden()	Acción para modificar los clientes relacionados con la orden.
Eliminarclienteorden()	Acción para eliminar un cliente de una orden de despacho.
Adicionarclienteorden()	Acción para adicionar clientes a una orden de despacho.
Confirmardespacho()	Acción para realizar la confirmación del documento de despacho.

Tabla 5 Descripción de la clase GestordendespachoModel

2.6 Descripción de la implementación

2.6.1 Diagrama de Despliegue

El diagrama consiste en uno o más nodos (elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos), dispositivos (nodos estereotipados con una capacidad de procesamiento), y conectores entre nodos, y entre nodos y dispositivos. El modelo de despliegue también mapea procesos dentro de estos elementos de procesamiento, permitiendo la distribución del comportamiento a través de los nodos que son representados. También se utiliza para visualizar la distribución de los componentes de software en los nodos físicos. A continuación se muestra el diagrama de despliegue propuesto:

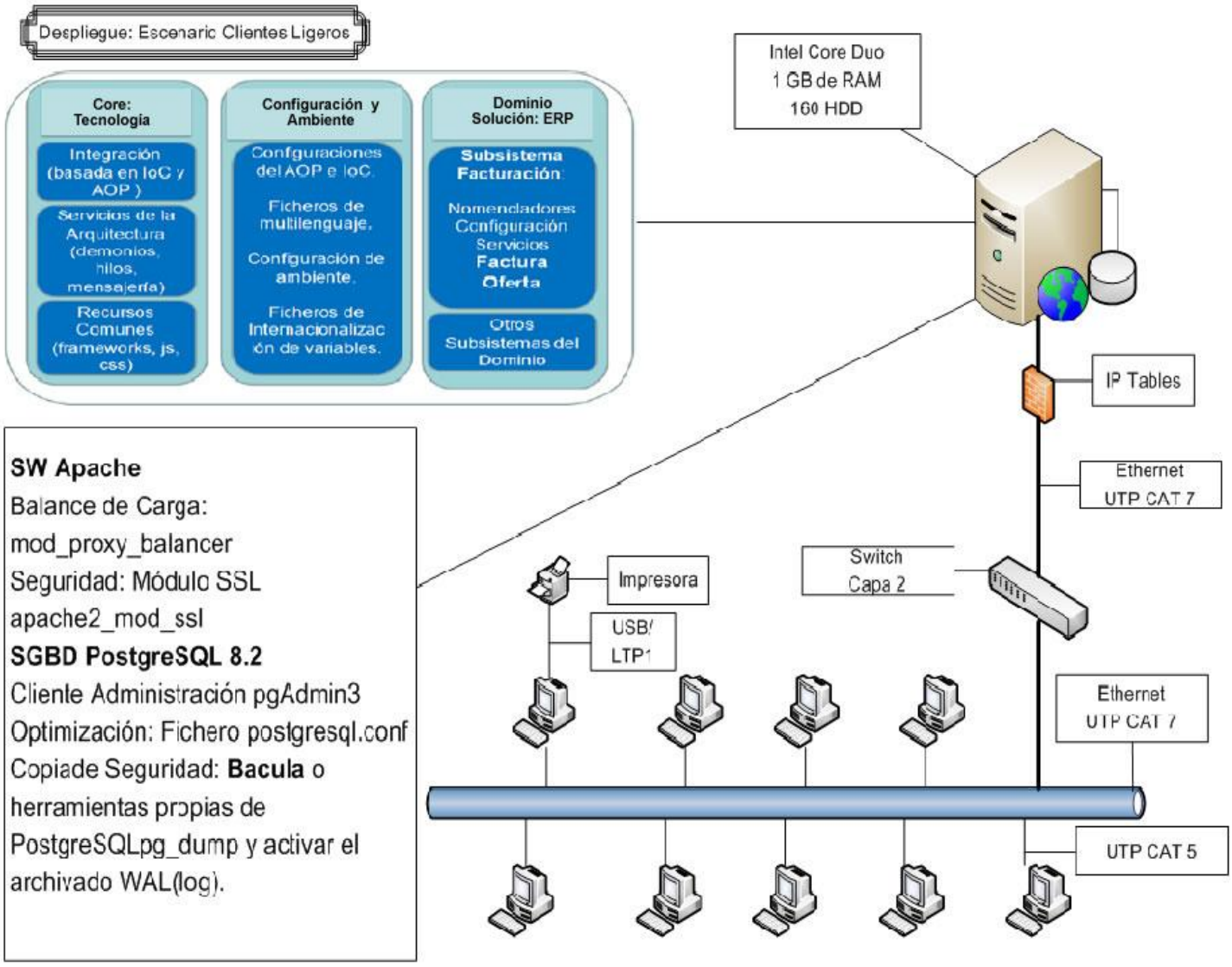


Fig. *5.* Diagrama de Despliegue

2.6.2 Diagrama de Componentes

Se representa como un grafo de componentes de software unidos por medio de relaciones de dependencia, pudiendo mostrarse las interfaces que estos soporten. Es el conjunto de ficheros interrelacionados entre sí para lograr la completa funcionalidad del sistema. Se representa un diagrama de

componentes general, ver figura #6, donde se ubican los paquetes y subsistemas por cada uno de los elementos de la arquitectura escogida para el diseño de la aplicación.

A continuación se muestra la vista general del sistema, distribuido por cada uno de sus paquetes de componentes representando las relaciones que se establecen entre ellos.

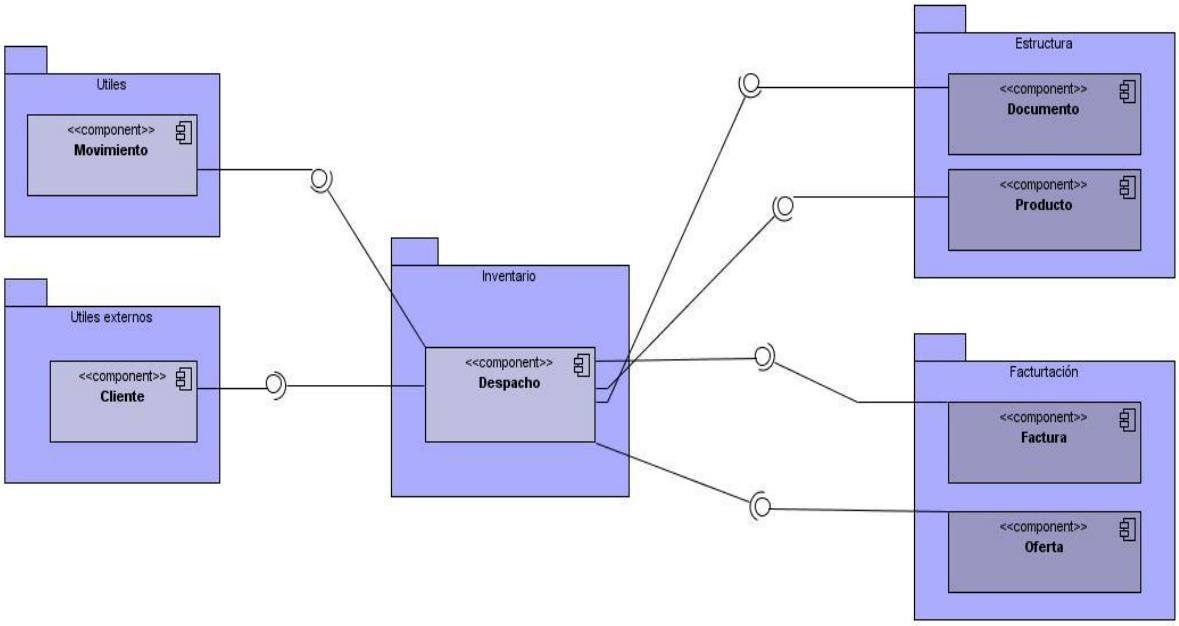


Fig. 6. Diagrama de Componentes General

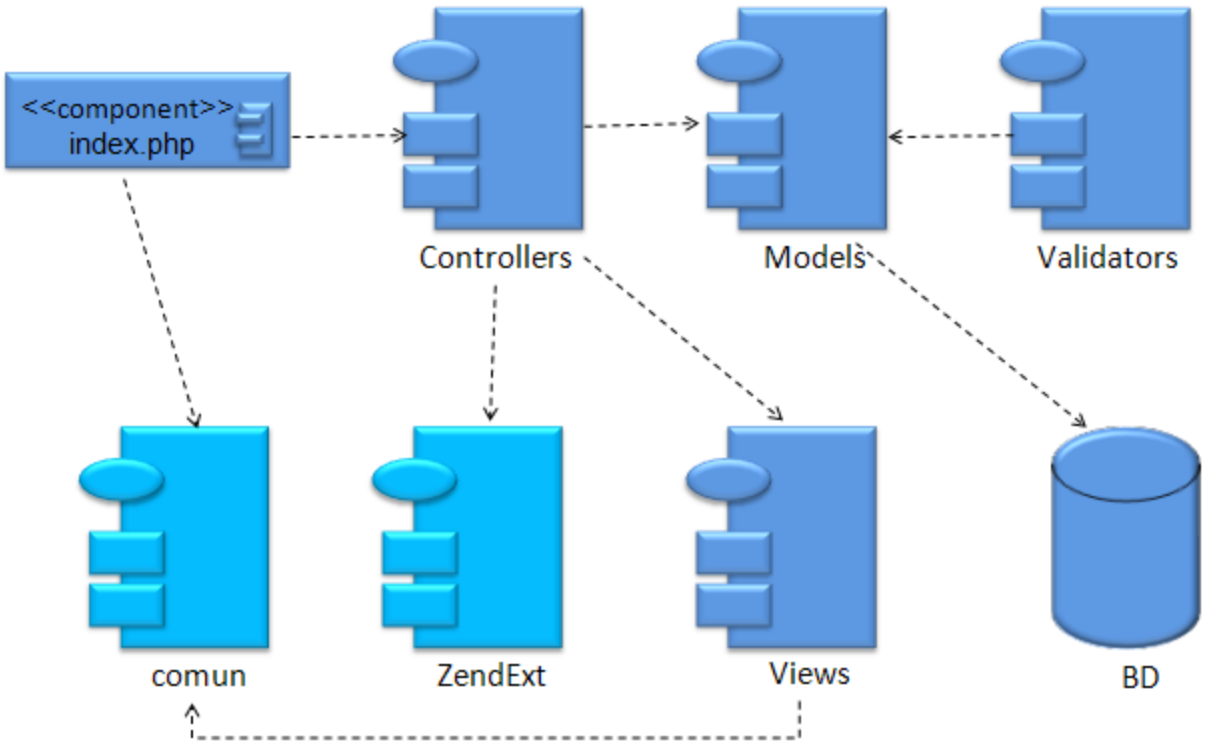


Fig. 7 Diagrama de Componentes (Despacho)

2.6.3 Integración entre componentes

MVC es un patrón de diseño de arquitectura de software que se usa, principalmente, en una aplicación que maneja 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, facilitando la programación en diferentes capas de manera paralela e independiente. Dada las ventajas que presenta dicho patrón se decidió utilizarlo en la implementación de la aplicación. Dicho patrón posibilita un trabajo más seguro, rápido y eficiente; pues antes se programaba de una manera engorrosa, lo que hacía a la aplicación más lenta e insegura.

Consta de cuatro nodos de integración, el que encuentra entre la vista y el controlador, el que está entre el controlador y el modelo, el que vincula el modelo con el framework doctrine y el que se encuentra entre el doctrine y la base de datos.

Todo el código dentro un mismo componente utiliza llamadas a métodos o eventos de forma directa. La comunicación entre diferentes módulos y componentes se realiza mediante llamadas a la inversión de control. El IoC especifica respuestas deseadas a sucesos o solicitudes de datos concretas, dejando que otro módulo o componente lleve a cabo las acciones de control que se requieran en el orden necesario y para el conjunto de sucesos que tengan que ocurrir.

Cada componente tiene su registro de los datos de los módulos en un fichero xml que será mapeado por el framework para el funcionamiento del mismo, dicho fichero tiene por nombre IoC y registra las funcionalidades que ofrecen los métodos de las clases control de los componentes del sistema. La base de datos es accedida de forma directa mediante controladoras y los componentes rehusados son integrados mediante interfaces sencillas, garantizando así una total integración de las capas en el sistema.

2.7 Estrategia para la captura de errores

Como todo sistema, para garantizar su correcto funcionamiento se debe tener en cuenta el tratamiento de errores. Hacer captura de las excepciones que son lanzadas por el sistema para advertir al usuario de que el software no funciona correctamente y debe ser cambiado, el sistema captura todas aquellas excepciones que son lanzadas y se le da un tratamiento para que el sistema no colapse.

Se sabe de antemano que, a la hora de registrar un nuevo usuario, por poner un ejemplo, al introducir algún dato que esté incorrecto, el sistema debe ser capaz de advertir al usuario, en caso contrario, esta excepción es lanzada por el sistema. De esta manera se da solución a las problemáticas de los lanzamientos de excepciones que se dan en el sistema según las peticiones del usuario, utilizando los try - catch, funciones que posibilitan hacer capturas de errores, capturando los tipos de errores lanzados y mostrando de manera visible al usuario mensajes de confirmación, los cuales le darán una idea de qué anda mal y cómo solucionarlo.

Mediante la interfaz Web se impedirá que el usuario asuma un papel activo en la introducción de la información, para esto se contará con cuadros de opción, menú de selección lo cual facilitará la entrada de datos. La información que requiera ser adicionada por el usuario se validará mediante funciones que garanticen que sea válida y que el cuadro de texto no esté vacío si es obligatorio llenarlo. Si hay un error en la información le saldrá al usuario un mensaje en pantalla indicándole el error, al oprimir Aceptar el mensaje desaparece y el usuario podrá seguir introduciendo los datos en el formulario. También se validarán las opciones correspondientes a la extracción o modificación de datos del servidor de base datos. Si se desea eliminar algún elemento de la BD se preguntará al usuario si está seguro de realizar dicha acción, al igual que cuando desee modificar alguna información, antes de actualizarla se le preguntará si desea realizarla o no. Así se logra que se realicen las operaciones que se desean y que se rectifique al cometer un error. A continuación se muestran algunos de estos mensajes:

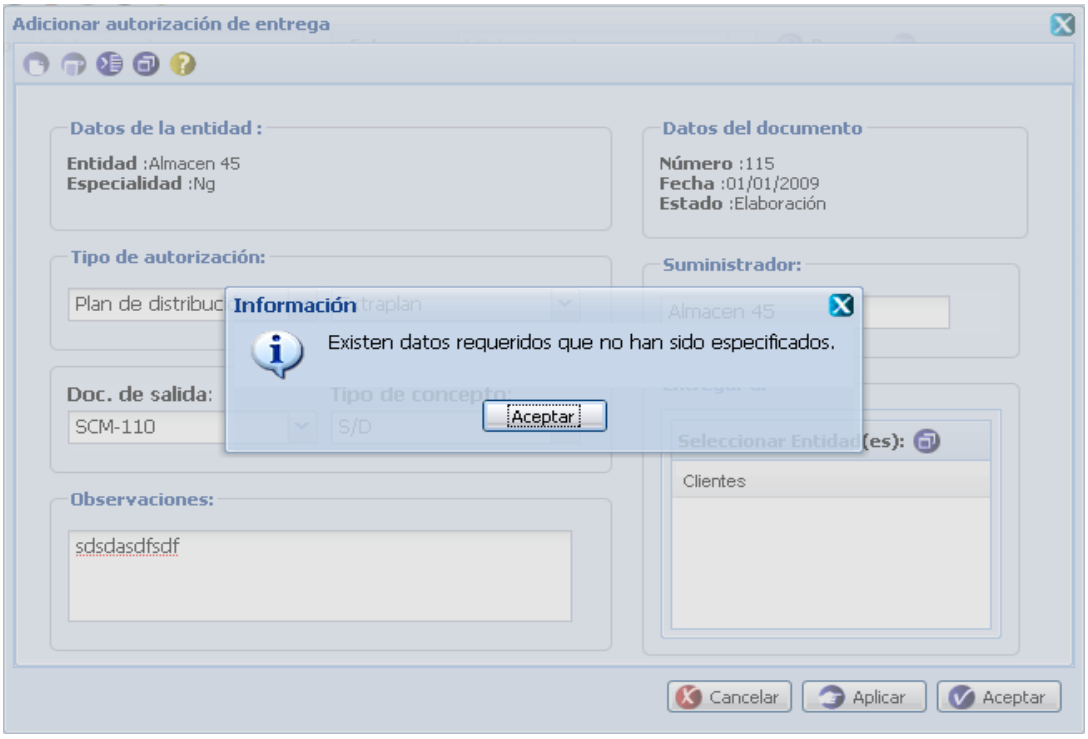


Fig.9 Error encontrado al crear una orden de entrega

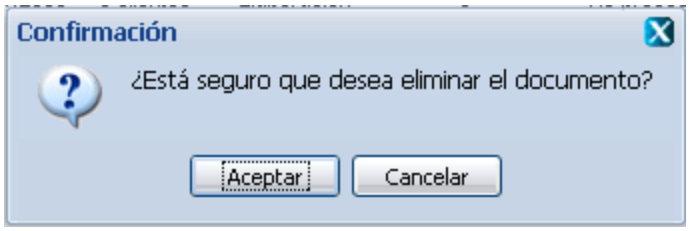


Fig. 10 Confirmación para realizar el borrado

En caso de que el usuario cometa un error y presione la opción de eliminar documento, el sistema le pide que confirme la acción.

2.8 Resultado esperado: Módulo de Despacho

Como resultado del proceso de implementación se obtiene una versión funcional del Módulo de Despacho. La aplicación es primera liberación del software, que como característica principal tiene que cumple con los requerimientos críticos, en otras palabras, las funcionalidades básicas que satisfacen los requerimientos del cliente.

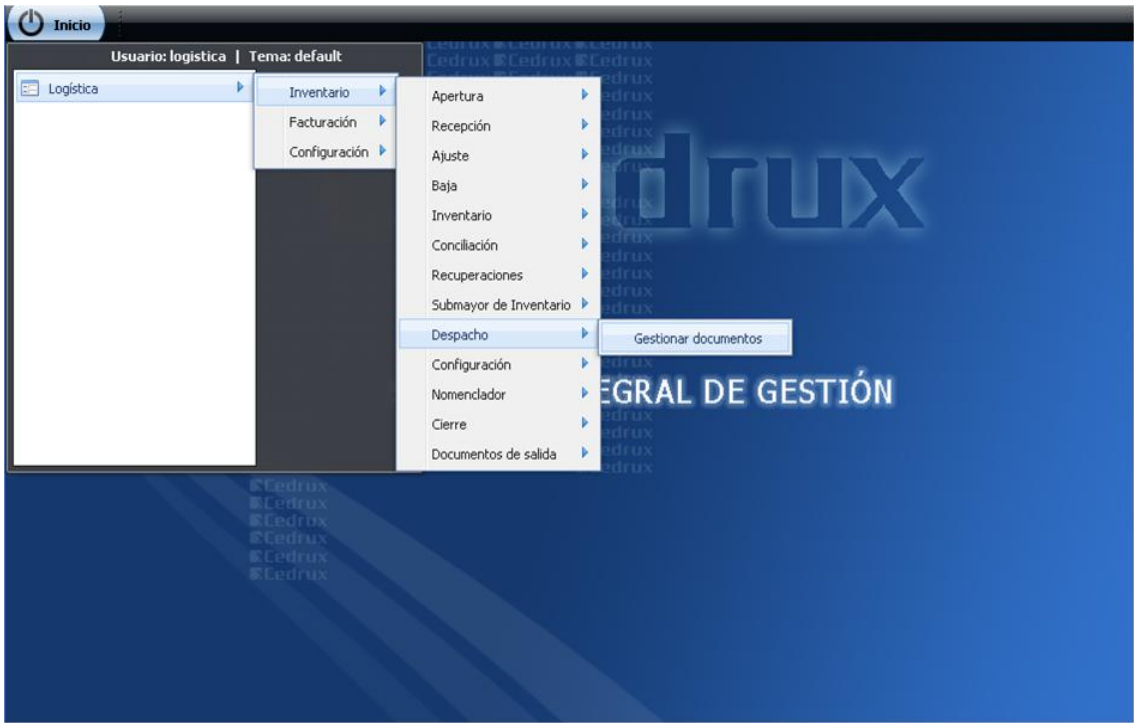


Fig. 11 Vista General del Sistema (entrada a despacho)

2.8.1 Características esenciales

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

- Realizar Orden de entrega
- Realizar Plan de abastecimiento
- Importar orden de despacho
- Crear reportes
- Adicionar Clientes/Productos a los documentos
- Eliminar documentos
- Modificar documentos

A continuación se muestran algunas figuras para visualizar de mejor forma lo descrito anteriormente:

Fig. 12 Crear Orden de Entrega

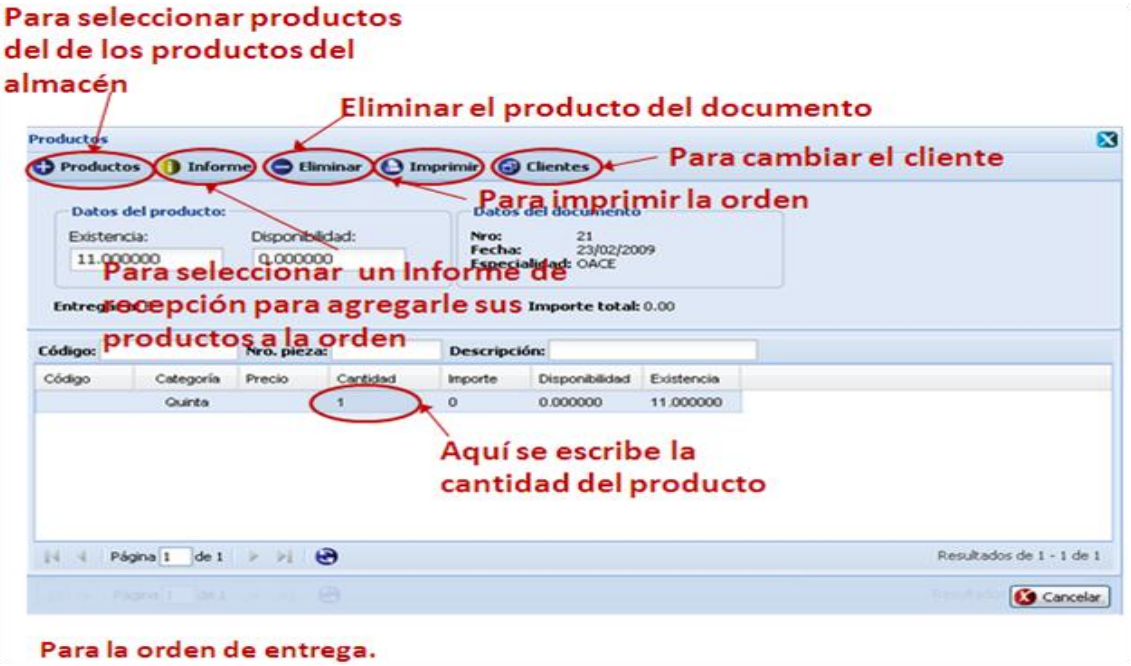


Fig. 13 Seleccionar productos para Orden de Entrega

Para más detalle acerca del sistema, ver Anexo II.

2.9 Aporte práctico

El Sistema, en su fase inicial, presenta algunas de las funcionalidades básicas requeridas por el cliente, pero, aún así, mejora en gran medida el proceso normal de la manera en que se desarrolla hoy. La automatización de los procesos de despacho en almacenes juega un papel importante y es un marcado avance en la automatización general de la gestión de almacenes en las entidades cubanas.

El módulo en cuestión permite disminuir los tiempos de respuesta, optimiza el proceso de gestión de la información derivada del despacho e incrementa su seguridad.

2.10 Conclusiones Parciales

Una vez que se parte de los principales artefactos entregados por analistas, diseñadores y arquitectos, de su revisión y estudio; cuando además se representan los diagramas de componentes y a partir de ellos se realiza la implementación del software, es posible obtener como resultado un prototipo funcional del sistema que cumple con las características básicas necesarias para satisfacer los requerimientos primarios del usuario final.

Capítulo 3 Validación de la solución propuesta

3.1 Introducción

A la hora del desarrollo de un software, existe una alta posibilidad de que aparezcan errores, ya sea por un uso indebido de la estructura de datos, por una mala especificación de los requisitos o en el momento de enlazar componentes. Para dar solución a todos estos problemas surge un nuevo proceso dentro del ciclo de desarrollo del software: el proceso de pruebas. Esta fase o proceso de prueba, por la cual pasa el software, es un elemento crítico para garantizar una alta calidad del producto.

Las pruebas y validación de los resultados no se realizan una vez acabado el software, sino que deben ejecutarse en cada una de las etapas de desarrollo. Es fundamental medir la cobertura de las pruebas, es decir, la determinación de cuándo se han realizado las suficientes pruebas. Si se siguen encontrando errores cada vez que se procesa el programa, las pruebas deben continuar.

En este capítulo se describe la validación de la solución propuesta, de forma que se pueda comprobar la eficiencia de las clases y operaciones utilizadas para satisfacer las necesidades del cliente, para ello se describen los tipos de prueba que verifican la validez de las funcionalidades del módulo de Despacho.

3.2 Pruebas de software

La prueba es una actividad fundamental en muchos procesos de desarrollo, incluyendo el software. De manera general, se puede decir que la prueba de software permite al desarrollador determinar si el producto generado satisface las especificaciones establecidas. Así mismo, una prueba de software permite detectar la presencia de errores que pudieran generar salidas o comportamientos inapropiados durante su ejecución.

De acuerdo a la IEEE el concepto de prueba se define como:

"*Una actividad en la cual un sistema o componente es ejecutado bajo condiciones especificas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente*".

Otro concepto importante a tomar en consideración es el emitido por Pressman en el libro de Ingeniería del Software en su edición de 1998, que plantea lo siguiente:

"La prueba del software es un elemento crítico para la garantía de calidad del software y representa una revisión de las especificaciones, del diseño y de la codificación".

Teniendo en cuenta las definiciones anteriores se puede concluir que la prueba de software es una actividad en la cual el sistema es ejecutado bajo condiciones específicas para demostrar que no tiene la madurez necesaria para ser implantado. Dentro de las actividades para obtener un software con la madurez necesaria están:

- Revisiones*:* consiste en que cada integrante del equipo de desarrollo revisa el producto que va generando.
- Inspecciones*: e*s el trabajo de revisar cada producto por parte de colegas.
- Validaciones*:* es el cliente quien revisa el producto para decir si cumple con sus necesidades.

Esta definición implica que, se considera una prueba exitosa, si se demuestran deficiencias en el software. Las fallas pueden ser en el código o en el modelado, en dependencia del tipo de pruebas que se le apliquen al software.

3.2.1 Objetivo de las pruebas

Dentro de los objetivos fundamentales que se persiguen al aplicarles pruebas a un software se encuentran los siguientes:

- Brindar un mayor nivel de confiabilidad en los productos que se van generando.
- Detectar fallas o errores.
- Aumentar la calidad del producto final.

Los objetivos anteriores cambian la idea que normalmente se tiene cuando se plantea que una prueba exitosa es aquella donde no se detectan errores. El objetivo principal es diseñar pruebas que sistemáticamente reflejen diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y

esfuerzo. Si las pruebas se llevan a cabo con éxito se descubrirán errores en el software**,** dándole mayor fiabilidad. Es importante tener en cuenta una frase de Pressman:

"La prueba no puede asegurar la ausencia de defectos; solo puede demostrar que existen defectos en el software" (Pressman, 2002).

3.3 Modelos de pruebas

3.3.1 Pruebas de unidad

Las pruebas de unidad son pruebas llevadas a cabo por los implementadores sobre las unidades mínimas desarrolladas por ellos, estas unidades pueden ser clases, métodos, propiedades, componentes. Se prueban separadas unas de otras, esto básicamente se hace durante la implementación del software.

Para realizar estas pruebas es necesario establecer una serie de reglas que sirven como objetivos, las cuales son:

- La prueba es un proceso de ejecución de un programa con la intención de descubrir errores.
- Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.
- Una prueba tiene éxito si descubre un error no detectado hasta entonces.

Las pruebas de unidad están asociadas a las pruebas de caja blanca, aunque para realizarlas es necesario probar el flujo de datos desde la interfaz del componente. Si los datos no se comportan correctamente al ser introducidos en la aplicación, todas las demás pruebas no tienen sentido. Las pruebas de caja blanca son aplicadas a componentes representados en el modelo de implementación para verificar que los flujos de control y de datos están cubiertos, y que estos funcionen como se espera.

3.3.2 Técnicas de diseño de pruebas

Existen tres enfoques principales para el diseño de casos de prueba:

1. El enfoque estructural o de caja blanca: que se basa en un minucioso examen de los detalles procedimentales del código a evaluar, por lo que es necesario conocer la lógica del programa.

2. El enfoque funcional o de caja negra: que realiza pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa. No es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar.

3. El enfoque aleatorio: consiste en utilizar modelos (en muchas ocasiones estadísticos) que representen las posibles entradas al programa para crear a partir de ellos los casos de prueba de caja negra.

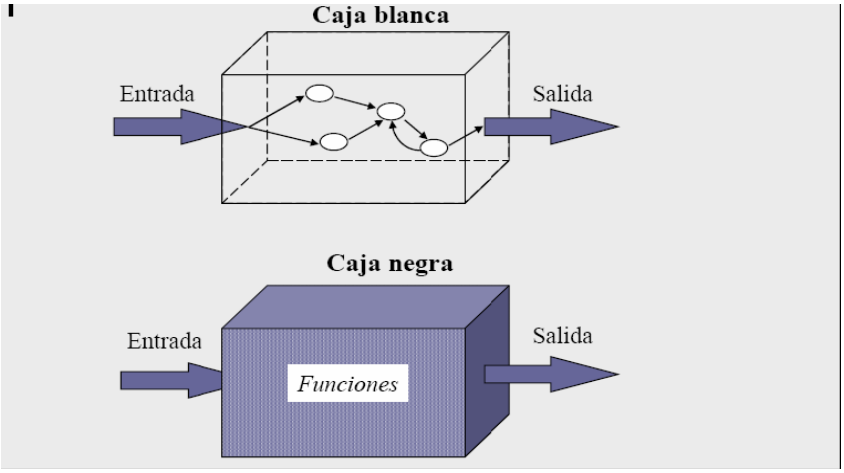


Fig. *8*. Representación de pruebas de Caja Blanca y Caja Negra

La Figura 8 representa gráficamente la filosofía de las pruebas de caja blanca y caja negra. Como se puede observar para las pruebas de caja blanca se necesita conocer los detalles procedimentales del código, mientras que para las de caja negra únicamente se necesita saber el objetivo o funcionalidad que el código ha de proporcionar.

3.3.3 Pruebas de caja blanca

A este tipo de técnicas se le conoce también como Técnicas de Caja Transparente o de Cristal. Este método se centra en cómo diseñar los casos de prueba atendiendo al comportamiento interno y la estructura del programa. Se examina así la lógica interna del programa sin considerar los aspectos de rendimiento.

El objetivo de la técnica es diseñar casos de prueba para que se ejecuten, al menos una vez, todas las sentencias del programa, y todas las condiciones tanto en su vertiente verdadera como falsa.
En resumen, mediante las pruebas de la caja blanca se pueden obtener casos de prueba que:

- Garanticen que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo, programa o método.
- Ejerciten todas las decisiones lógicas en las vertientes verdadera y falsa.
- Ejecuten todos los bucles en sus límites operacionales.
- Ejerciten las estructuras internas de datos para asegurar su validez.

Algunas de las técnicas de prueba de Caja Blanca son:

- Prueba de Condición
- Prueba de Flujo de Datos
- Prueba de Bucles.
- Prueba del Camino Básico

Esta técnica permite obtener una medida de la complejidad lógica del diseño y usar esta medida como guía para la definición de un camino básico.

La idea es derivar casos de prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control. Para obtener dicho conjunto de caminos independientes se construye el Grafo de Flujo asociado y se calcula su complejidad ciclomática.

Los pasos que se siguen para aplicar esta técnica son:

1. A partir del diseño o del código fuente, se dibuja el grafo de flujo asociado.
2. Se calcula la complejidad ciclomática del grafo.
3. Se determina un conjunto básico de caminos independientes.
4. Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

Los casos de prueba derivados del conjunto básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa. A continuación se muestra un ejemplo donde se calcula la complejidad ciclomática.

```
public function modificarclientesorden($orden,$idclientes,$opciones){
 $orden['iddoc']  = ($orden['torden'] == 1)?'108':'108A'; 1
 if($orden['iddoc'] == '108' ){  2
  if($idclientes['add']) 3
   {
    $this->adicionarclienteorden($orden,$idclientes['add'],$opciones); 4
   }
  if($opciones ){ 5
   if($orden['planextraplan'] != 2) 6
    $clientes = $this->pIntegrator->movimientos->GetClientes_PO($orden['idorden']); 7
   else $clientes = $this->pIntegrator->movimientos->GetClientes_PP($orden['idorden']); 8
    if($clientes[0]->idcliente){ 9
    $datcliente[0]['idcliente']= $idclientes['add']  ; 10
    $datcliente[0]['denom']= $this->pIntegrator->utilesLogistica->getClientesNombre($idclientes['add'] ); 10
 }
} 11
}else{
 if($orden['pant']) { 12
  $listaprod = $this->pIntegrator->movimientos->BuscarListProdDoc_PO($orden['idorden']); 13
  if($listaprod[0]->idorden){   14
   foreach($listaprod as $reg) { 15
   $datosproducto = $this->pIntegrator->productos->obtenerProducto()->devolverProductos('',$reg->idproducto);16
   $this->pIntegrator->movimientos->Insertar_PP($reg->idorden,0,$datosproducto[0]['idprod'], 16
   $reg->idcliente,0);   16
   $res = $this->pIntegrator->productos->actualizarcantdisp($reg->cantidad,$reg,'+'); 16
   } 17
  }
    $this->pIntegrator->movimientos->Eliminar_PO($orden['idorden']);  18
  }  19
   if($idclientes['add'] ){ 20
    $idclientesadd = split(',',$idclientes['add']); 21
    foreach($idclientesadd as $unidad){ 22
     $this->adicionarclienteorden($orden,$unidad,$opciones); 23
    } 24
   } 25
   if($idclientes['elim']){ 26
    $idclienteselim = split(',',$idclientes['elim']); 27
    foreach($idclienteselim as $unidad){ 28
     $this->eliminarclienteorden($orden,$unidad,$opciones); 29
    }  30
   }31
 }

 $datcliente['ok'] = true; 32
 return $datcliente; 32
}  33
```

Fig. 9 Función modificarclientesorden

Para el grafo de la función anterior, ver Anexo III.

V(G) = P + 1
V(G) = 11 + 1
V(G) =12

V(G) = A - N + 2
V(G) = 43 – 33 + 2
V(G) = 12

V(G) = R
V(G) = 12

La complejidad ciclomática brinda la posibilidad de conocer la cantidad de caminos que presenta el grafo construido, en este caso se puede llegar a la conclusión que son 12 los caminos por los que, desde el nodo inicial se puede llegar al nodo final. A continuación se verá un ejemplo completo que ilustra este proceso.

3.3.4 Aplicaciones de las pruebas caja blanca

En los siguientes subepígrafes se le realizarán pruebas de caja blanca a algunas de las principales funcionalidades implementadas. Se harán pruebas con juegos de datos reales para buscar resultados satisfactorios, comprobando todos los nodos del árbol.

3.3.4.1 Función CargarDatosEncabezadoModificar.

```
public function CargarDatosEncabezadoModificar($iddocumento,$idestructuracomun,$opciones)
{
 $datos = $this->pIntegrator->documentos->CargarDatosEncabezadoModificarDesp($idestructuracomun,$iddocumento); 1
 $datcliente=array(); 1
 if($datos->tplan != 5)2
 $clientes =   $this->pIntegrator->movimientos->GetClientes_PO($iddocumento); 3
 else $clientes = $this->pIntegrator->movimientos->GetClientes_PP($iddocumento); 4
 if($clientes[0]->idcliente) 5
 {
     $k=0;6
     foreach ($clientes as $client)7
     {
         $res =  $this->pIntegrator->utilesLogistica->getClientesNombre($client->idcliente);8
         $datcliente[$k]['idcliente']= $client->idcliente;8
         $datcliente[$k]['denom']= $res;  8
         $k++;8
     }9
 }
 if($opciones && $opciones!= 'Mod')10
 $datos->importe = $this->ImporteTotal($iddocumento);11
 return  $datos =array('datos'=>$datcliente,'datosd'=>$datos);  12
}13
```

Fig. 10 Función CargarDatosEncabezadoModificar

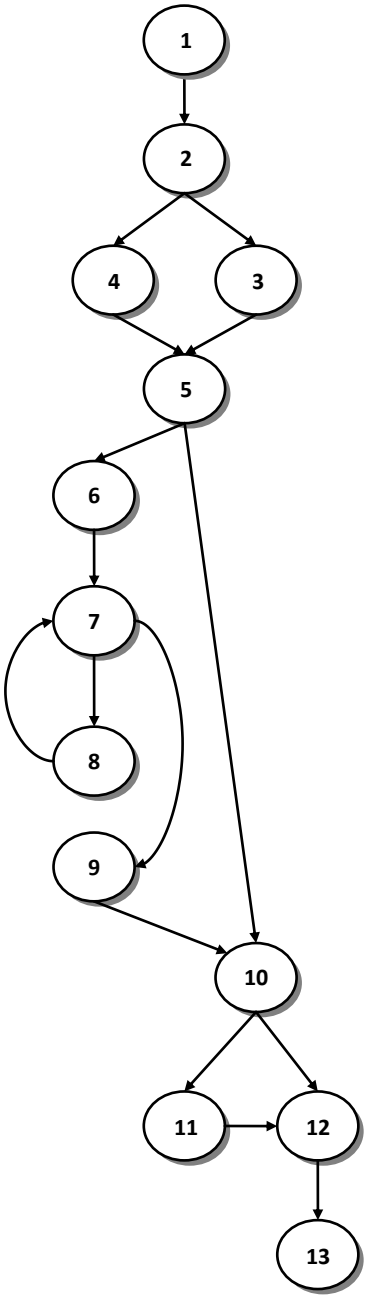


Fig. 11 Grafo asociado a la función anterior

V(G) = P + 1
V(G) = 4 + 1
V(G) =5

V(G) = A - N + 2
V(G) = 16 – 13 + 2
V(G) = 5

V(G) = R
V(G) = 5

Camino 1: 1-2-4-5-10-12-13
Camino 2: 1-2-3-5-10-12-13
Camino 3: 1-2-3-5-10-11-12-13
Camino 4: 1-2-4-5-6-7-9-10-11-12-13
Camino 5: 1-2-4-5-6-7-8-7-9-10-11-12-13

Caso de prueba para el camino básico 1.
Descripción:

- Los datos de entrada cumplirán con los siguientes requisitos:

  El documento con el que se trabajará será un Plan de Distribución que tendrá como tipo de plan '5', no existirán clientes adicionados al documento y la orden será la de Modificar.

Condición de ejecución:

- El iddocumento será 100
- El idestructuracomun será 1002050
- La opción entrada será la de "Modificar"

Entrada:

- $iddocumento=100
- $idestructuracomun=1002050
- $opciones='Modificar'

Resultados esperados:

Se espera que se adicione el iddocumento en la base de datos y se muestre el documento.

Caso de prueba para el camino básico 2

Descripción:

- Los datos de entrada cumplirán con los siguientes requisitos:

  El documento con el que se trabajará será un Plan de Distribución que tendrá como tipo de plan '5', no existirán clientes adicionados al documento y la orden será la de Modificar.

Condición de ejecución:

- El iddocumento será 100
- El idestructuracomun será 1002050
- La opción entrada será la de "Modificar"

Entrada:

- $iddocumento=100
- $idestructuracomun=1002050
- $opciones='Modificar'

Resultados esperados:

Se espera que se adicione el iddocumento en la base de datos y se muestre el documento.

Caso de prueba para el camino básico 3

Descripción:

- Los datos de entrada cumplirán con los siguientes requisitos:

El documento con el que se trabajará será un Plan de Distribución que tendrá como tipo de plan '5', no existirán clientes adicionados al documento y la orden será la de Adicionar.

Condición de ejecución:

- El iddocumento será 100
- El idestructuracomun será 1002050
- La opción entrada será la de "Adicionar"

Entrada:

- $iddocumento=100
- $idestructuracomun=1002050
- $opciones='Adicionar'

Resultados esperados:

Se espera que se adicione el iddocumento en la base de datos, se calcule el importe total de los productos asociados al documento y se muestra el documento.

Caso de prueba para el camino básico 5
Descripción:

- Los datos de entrada cumplirán con los siguientes requisitos:

El documento con el que se trabajará será un Extraplan y la orden será la de Adicionar.

Condición de ejecución:

- El iddocumento será 100
- El idestructuracomun será 1002050
- La opción entrada será la de "Adicionar"

Entrada:

- $iddocumento=100
- $idestructuracomun=1002050

- $opciones='Adicionar'

Resultados esperados:

Se espera que se adicione el iddocumento en la base de datos, se calcule el importe total de los productos asociados al documento y se muestra el documento.

3.3.4.2 Función adicionarOrdenDesp

```
public function adicionarOrdenDesp($datos,$idclientes,$sess)
 {
 if($datos['iddoc']=='108') 1
 {
    $entrega = $this->integrator->parametros->TipoDocumentoByAlias('entregaLog');  2
    $res = $this->pIntegrator->movimientos->Insertar_PO($a->idorden,0,0,substr($idclientes,0,-1));  2
 }
 else
 {
    $entrega = $this->integrator->parametros->TipoDocumentoByAlias('plandistribLog'); 3
    $a =  $this->pIntegrator->documentos->AdicionarDocOrd($sess['idestructuracomun'],$sess['alias'],
    $entrega->iddoc,$datos['iddoccg'],$datos['observaciones'],$datos['idconcepto'],$datos['planextraplan'],
    $datos['idsuministrador']);  3
    $entregara = substr($idclientes,0,strlen($idclientes)-1); 3
    $entregara = split(',',$entregara);3
    if($entregara)  4
    {
      foreach($entregara as $unidad) 5
      {
        if($datos['planextraplan'] != 2) 6
          $this->pIntegrator->movimientos->Insertar_PO($a->idorden,0,0,$unidad); 7
        else
        $this->pIntegrator->movimientos->Insertar_PP($a->idorden,0,0,$unidad);  8
      }  9
    }
 }
 $a->ok = true;   10
 return $a ;     10
 } 11
```

Fig. 12 Función adicionarOrdenDesp

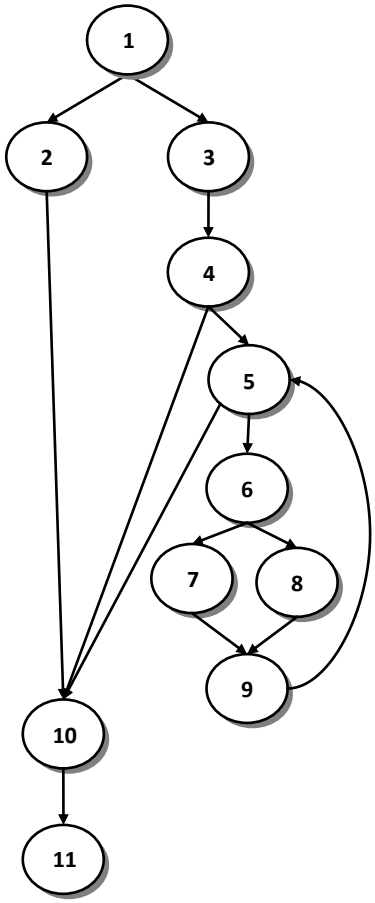


Fig. 13 Grafo asociado a la función anterior

V(G) = P + 1
V(G) = 4 + 1
V(G) = 5

V(G) = A - N + 2
V(G) = 14 – 11 + 2
V(G) = 5

V(G) = R
V(G) = 5

Camino 1: 1-2-10-11
Camino 2: 1-3-4-10-11
Camino 3: 1-3-4-5-6-7-9-5-10-11
Camino 4: 1-3-4-5-6-8-9-10-11
Camino 5: 1-3-4-5-10-11
Caso de prueba para el camino básico 1.
Descripción:

- Los datos de entrada cumplirán con los siguientes requisitos:

El documento que se adicionará será una Orden de Entrega, el cual tiene como identificador '108'.

Condición de ejecución:

- El iddocumento será 108
- El idclientes será 1002, 1003, 2321, 2501
- Los datos que conformarán a la variable datos serán iddoccg=901056, observaciones='Todo bien', idconcepto=2675, planextraplan='2'.
- Los datos utilizados por el componente externo sesión será el idestructuracomun=0014278 y el alias='alias'.

Entrada:

- $iddocumento=108
- $idclientes[1002, 1003, 2321, 2501]
- $datos=[$iddoccg=901056, $observaciones='Todo bien', $idconcepto=2675, $planextraplan='2']
- $sess=[$idestructuracomun=0014278, $alias='alias']

Resultados esperados:

La orden se adiciona a la base de datos, correspondiendo a una Orden de Entrega, asignándole un identificador, así como el identificador del cliente.

Caso de prueba para el camino básico 2
Descripción:

- Los datos de entrada cumplirán con los siguientes requisitos:

  El documento que se adicionará será un Plan de Distribución, el cual tiene como identificador '108a', pero a la hora de adicionarle los datos necesarios para conformar la orden, los clientes que de la anterior no serán adicionados al documento.

Condición de ejecución:

- El iddocumento será 108a
- El idclientes será 1002, 1003, 2321, 2501
- Los datos que conformarán a la variable datos serán iddoccg=901056, observaciones='Todo bien', idconcepto=2675, planextraplan='2'.
- Los datos utilizados por el componente externo sesión será el idestructuracomun=0014278 y el alias='alias'.

Entrada:

- $iddocumento=108a
- $idclientes[1002, 1003, 2321, 2501]
- $datos=[$iddoccg=901056, $observaciones='Todo bien', $idconcepto=2675, $planextraplan='2']
- $sess=[$idestructuracomun=0014278, $alias='alias']

Resultados esperados:

La orden se adiciona a la base de datos, correspondiendo a una Plan de Distribución con los datos necesarios, menos los clientes encargados de la orden.

Caso de prueba para el camino básico 3

Descripción:

- Los datos de entrada cumplirán con los siguientes requisitos:

El documento que se adicionará será un Plan de Distribución, el cual tiene como identificador '108a', el tipo de plan será '3'.

Condición de ejecución:

- El iddocumento será 108a
- El idclientes será 1002, 1003, 2321, 2501
- Los datos que conformarán a la variable datos serán iddoccg=901056, observaciones='Todo bien', idconcepto=2675,planextraplan='3'
- Los datos utilizados por el componente externo sesión será el idestructuracomun=0014278 y el alias='alias'.

Entrada:

- $iddocumento=108a
- $idclientes[1002, 1003, 2321, 2501]
- $datos=[$iddoccg=901056, $observaciones='Todo bien', $idconcepto=2675, $planextraplan='2']
- $sess=[$idestructuracomun=0014278, $alias='alias']

Resultados esperados:

La orden es adicionada a la base de datos de forma correcta, sin demora y no han ocurrido errores en la ejecución del código.

Caso de prueba para el camino básico 4

Descripción:

- Los datos de entrada cumplirán con los siguientes requisitos:

El documento que se adicionará será un Plan de Distribución, el cual tiene como identificador '108a', el tipo de plan será '2'.

Condición de ejecución:

- El iddocumento será 108a
- El idclientes será 1002, 1003, 2321, 2501
- Los datos que conformarán a la variable datos serán iddoccg=901056, observaciones='Todo bien', idconcepto=2675,planextraplan='3'
- Los datos utilizados por el componente externo sesión será el idestructuracomun=0014278 y el alias='alias'.

Entrada:

- $iddocumento=108a
- $idclientes[1002, 1003, 2321, 2501]
- $datos=[$iddoccg=901056, $observaciones='Todo bien', $idconcepto=2675, $planextraplan='2']
- $sess=[$idestructuracomun=0014278, $alias='alias']

Resultados esperados:

La orden es adicionada a la base de datos de forma correcta, sin demora y no han ocurrido errores en la ejecución del código.

3.3.5 Resultados

Las pruebas de caja blanca fueron complementadas con revisiones cruzadas por parte de otros programadores, efectuándose durante toda la fase de implementación. Estas revisiones posibilitaron la detección de errores comunes, los cuales hubieran sido causa de retraso en implementaciones posteriores.

Se realizaron varias iteraciones de pruebas de caja blanca a las principales funcionalidades del sistema. En las primeras iteraciones las funciones presentaban cierto grado de errores, entre los que se pueden destacar:

- Utilización de código innecesario para darle solución a un problema.
- La declaración de variables innecesariamente.

Al realizar las iteraciones posteriores de la aplicación de las pruebas diseñadas y expuestas anteriormente, luego de erradicar los errores encontrados en la ejecución de la primera iteración, los resultados mejoraron considerablemente, como se muestran en la siguiente figura:

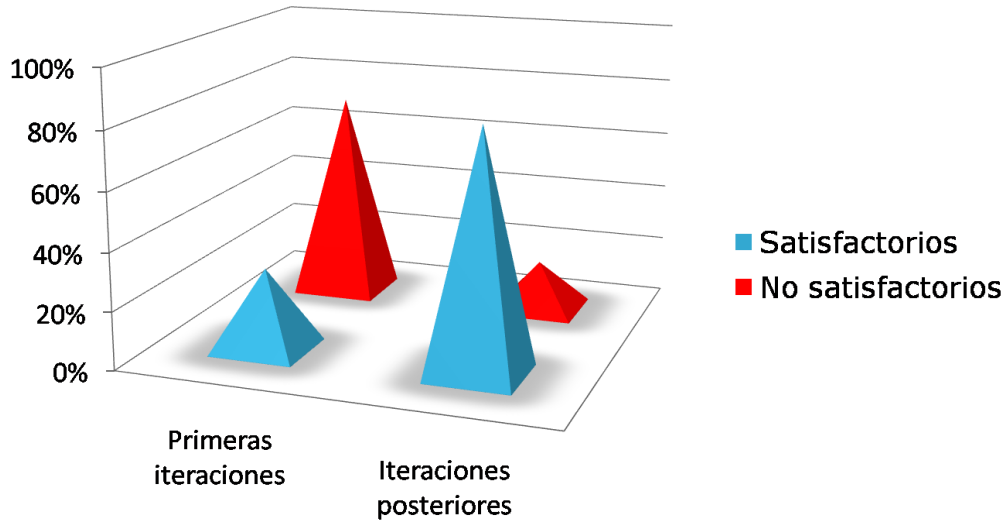


Fig. 14 Resultado de las pruebas

3.4 Conclusiones Parciales

La realización de las pruebas de caja blanca a los principales requerimientos del software permitió comprobar que:

- Si el programa en cada de una de sus variantes llegó al resultado esperado.
- Las condicionales tomaban el valor adecuado en el momento preciso.
- Los mensajes, tanto de errores previstos como de oraciones aclaratorias que se deben mostrar al cliente, lo hacían en el momento justo. También se probó si el programa luego de emitir un error mantenía la estabilidad requerida.

CONCLUSIONES

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

- Se realizó un análisis de los sistemas descubriendo deficiencias en los mismos.
- Se implementaron los requisitos funcionales planteados por los analistas reduciendo el tiempo de elaboración y ejecución del despacho en las entidades del país.
- Se aplicaron pruebas al sistema para validar la calidad de la solución propuesta mediante las pruebas del camino básico arrojando resultados positivos.

RECOMENDACIONES

Las recomendaciones propuestas para la continuidad del presente trabajo son:

- Arreglar las no conformidades detectadas en las pruebas pilotos con el objetivo de refinar la solución.
- Ampliar las funcionalidades del módulo con los nuevos requerimientos que surjan por necesidades del cliente.
- Poner al alcance de todos, este documento, como material de estudio, guía y apoyo para su posterior continuación.

GLOSARIO DE TÉRMINOS

LAMP: la integración de Linux, Apache, MySQL y PHP, es una herramienta muy utilizada para la creación de servicios web.

SVN: Sistema de Control de Versiones.

CVS: Sistema Concurrente de Versiones.

YUI: Interfaz de Usuario de Yahoo.

API: Interfaz de Programación de Aplicaciones. Es el conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

IoC: Inversión de Contro.

Framework: Conjunto de APIs y herramientas destinadas a la construcción de un determinado tipo de aplicaciones de manera generalista.

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

Caso de Prueba: es especificar una forma de probar el sistema, incluyendo las entradas con las que se ha de probar, los resultados esperados y las condiciones bajo las que ha de probarse.

Grafo de Flujo: está formado por tres componentes fundamentales que ayudan a su elaboración, comprensión y brinda información para confirmar que el trabajo se está haciendo adecuadamente. Los componentes son:

Nodo: Cada círculo representado se denomina nodo del Grafo de Flujo, el cual representa una o más secuencias procedimentales. Un solo nodo puede corresponder a una secuencia de procesos o a una sentencia de decisión. Puede ser también que hayan nodos que no se asocien, se utilizan principalmente al inicio y final del grafo.

Aristas: Las flechas del grafo se denominan aristas y representan el flujo de control, son análogas a las representadas en un diagrama de flujo. Una arista debe terminar en un nodo, incluso aunque el nodo no represente ninguna sentencia procedimental.

Nodo Predicado: Nodo del cual salen varias aristas.

Regiones: Las regiones son las áreas delimitadas por las aristas y nodos. También se incluye el área exterior del grafo, contando como una región más. Las regiones se enumeran y la cantidad de regiones es equivalente a la cantidad de caminos independientes del conjunto básico de un programa.

La complejidad ciclomática V(G), define el número de caminos independientes del conjunto básico de un programa y da un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez. Un camino independiente es cualquier camino del programa que introduce por lo menos un nuevo conjunto de sentencias de procesamiento o una nueva condición. El camino independiente se debe mover por lo menos por una arista que no haya sido recorrida anteriormente.

Hay tres formas fundamentales de calcular la complejidad:

1. V(G) = número de regiones del grafo de flujo.

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

donde: *A* es el número de aristas del grafo y *N* es el número de nodos.

3. V(G) = *P* + 1

donde: *P* es el número de nodos predicado contenidos en el grafo *G*.

Protocolo: Conjunto de reglas que gobiernan el intercambio de datos entre entidades dentro de una red. Es el lenguaje común que utilizan los ordenadores para hablar y entenderse entre sí.

Almacén: Lugar que lleva a cabo los movimientos de productos, ya sea las entradas, salidas e inventario de los mismos. Sirve como centro regulador del flujo de mercancías entre la disponibilidad y la necesidad de fabricantes, comerciantes y consumidores.

Autorización de entrega: Contiene los diferentes documentos que vienen de inventario (orden de entrega y plan de distribución).

Orden de entrega: Ordena al almacén las entregas de mercancías para la venta y productos terminados a los clientes, con destino a la comercialización.

Plan de distribución: Su objetivo es comunicar a los almacenes en los diferentes niveles las asignaciones hechas a las unidades subordinadas.

Producto: Es cualquier objeto que puede ser ofrecido a un mercado que pueda satisfacer un deseo o una necesidad. Sin embargo, es mucho más que un objeto físico. Es un completo conjunto de beneficios o

satisfacciones que los consumidores perciben cuando compran; es la suma de los atributos físicos, psicológicos, simbólicos y de servicio.

BIBLIOGRAFÍA

Alvarez, Miguel Angel. 2003. Evaluando Zend Studio. [En línea] 2003. http://www.cienciasmisticas.com.ar.

Andrés, María Mercedes Marqués. 2001. Clasificación de los sistemas de gestión de bases de datos. [En línea] 2001. http://www3.uji.es/~mmarques/f47/apun/node38.html.

Arquitectura Cliente Servidor Web. [En línea] http://www.scribd.com/doc/14168211/2009-01-Arquitectura-Cliente-Servidor-Web.

Booch, Jacobson y Rumbaugh. 2000. *El Proceso Unificado de Desarrollo de software.* s.l. : Addison-Wesley, 2000.

Camy, Lázaro Issi. 2002. *Javascript.* 2002.

Cano, Fernando P. Najera. 2009. Un cliente de Subversion para Windows. [En línea] 4 de abril de 2009. http://tortoisesvn.net/docs/release/TortoiseSVN_es/index.html.

2008. Características principales de un ERP. [En línea] 2 de junio de 2008. http://definanzas.com/2008/06/02/caracteristicas-erp/.

Cataldi, Zulma. 2000. Metodología de diseño, desarrollo y evaluación de software educativo. [En línea] 2000. http://laboratorios.fi.uba.ar/lsi/cataldi-tesisdemagistereninformatica.pdf.

Cestero, José M. 2008. Servidores web en la empresa, ventajas y posibilidades . [En línea] 25 de octubre de 2008. http://www.tecnologiapyme.com/software/servidores-web-en-la-empresa-ventajas-y-posibilidades.

Ciberaula. Introducción, definición y evolución de PHP. [En línea]

Collins-Sussman, Ben. Control de versiones con Subversion. [En línea] http://svnbook.red-bean.com/index.es.html.

2008 . Crear JSON con PHP. [En línea] 23 de noviembre de 2008 . http://blog.unijimpe.net/crear-json-con-php/.

Curso de JavaScript. [En línea] http://www.cienciasmisticas.com.ar.

CURSOS SOBRE DESARROLLO DE APLICACIONES EN ARQUITECTURA . *CLIENTE/SERVIDOR USANDO TECNOLOGÍAS WEB.* [En línea] http://iier.isciii.es/cur_web/.

Española, W3 Oficina. 2008. Guía Breve de CSS . [En línea] 9 de enero de 2008. http://www.w3c.es/divulgacion/guiasbreves/HojasEstilo.

Espinoza, Humberto. 2005. Una Alternativa de DBMS Open Source. [En línea] 2005. http://www.lgs.com.ve/pres/PresentacionES_PSQL.pdf.

2006. Firefox, el navegador que está haciendo época. [En línea] 8 de marzo de 2006. http://www.positivoynegativo.com/2006/03/08/39/.

Good, Robin. 2005 . Beneficios De Las Aplicaciones Basadas En Web Y El Anuncio De Microsoft De La Era “En Vivo”. [En línea] 2 de Noviembre de 2005 . http://www.google.com.cu/search?hl=es&q=ventajas+de+aplicaciones+Web&btnG=Buscar+con+Google&meta=&aq=f&oq=.

Graham, Paul. 2001. The Road Ahead. [En línea] September de 2001. http://www.paulgraham.com/road.html.

Heredia, Cristian Van Der y Herminio. 2001. Introducción al PHP. [En línea] 23 de Mayo de 2001. http://www.maestrosdelweb.com/editorial/phpintro/.

Hernández, Carlos de la Rosa. 2008. *Análisis y Diseño de los módulos Inventario y Administración del proyecto ERP Cubano.* Ciudad de La Habana : s.n., 2008.

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

Mozos, Ramón Coloma. 2007. Visual Paradigm, "La herramienta" . [En línea] 26 de noviembre de 2007. http://bikermon-informatico.blogspot.com/2007/11/visual-paradigm-la-herramienta.html.

Negrete, Fernando Atanasio. 2003. Apache + PHP + MySql + PhpMyAdmin como módulo de Apache. [En línea] 19 de octubre de 2003. http://www.maestrosdelweb.com/editorial/phpmysqlap/.

Objetivos de ERP. [En línea] http://www.erp-recycling.org/351.0.html.

Ortuño, Miguel Ángel. 2009. Criterios y Costes de implantación de un ERP. [En línea] 15 de enero de 2009. http://download.microsoft.com/download/d/c/0/dc0e04eb-ddfc-4845-8efc-88ccc1d19fcd/Costes_Criterios_Implantacion_ERP.pdf.

Pantaleo, Guillermo G. Arquitectura de Software. [En línea] http://materias.fi.uba.ar/7572/.

Pérez, Javier Eguíluz. *Introducción a CSS.* pág. 223 .

Perissé, Marcelo Claudio. 2001. *Proyecto Informáticos Metodología Estructurada Simplificada.* 2001.

Pressman, Roger S. 2002 . *Ingeniería del Software .* s.l. : McGraw-Hill, 2002 .

Prueba de software. [En línea] http://lsi.ugr.es/~ig1/docis/pruso.pdf.

2005. Qué es ERP y SAP. [En línea] 16 de noviembre de 2005. http://www.nocturnabsas.com.ar/forum/informes/1301-que-erp-y-sap.html.

Referencia Netscape de Javascript . [En línea] http://www.geocities.com/v.iniestra/javascript/index.html.

Sánchez, Federico Plancarte. 2005. PLANEACIÓN DE RECURSOS EMPRESARIALES (ERP). [En línea] marzo de 2005. http://www.gestiopolis.com/recursos4/docs/ger/planerp.htm.

Software, Grupo de Ingeniería del. 2004. Introducción a las Aplicaciones Web. [En línea] octubre de 2004. http://www.lsi.us.es/docencia/get.php?id=352.

Soluciones de Software ERP. [En línea] http://www.erpsoftware.com.mx/.

SUPINFO, Escuela de Ingenieros. 2008. Lenguajes de programación. [En línea] 16 de octubre de 2008. http://es.kioskea.net/contents/langages/langages.php3.

Torossi, Gustavo. El Proceso Unificado de Desarrollo de Software. [En línea] http://www.chaco.gov.ar/UTN/disenodesistemas/apuntes/oo/ApunteRUP.pdf.

Tutorial Básico de PHP. [En línea] http://www.find-script.com/scripts/PHP.

Yuniel. 2009. Revista Atix - Software Libre Argentina. *Diseño Avanzado de Aplicaciones Web. Uso de EXT, Zend Framework y Doctrine.* [En línea] Opentelematics.org, 9 de marzo de 2009. [Citado el: 15 de marzo de 2009.] http://softwarelibre.org.bo/esteban/files/86/226/atix08.pdf.

ANEXO I

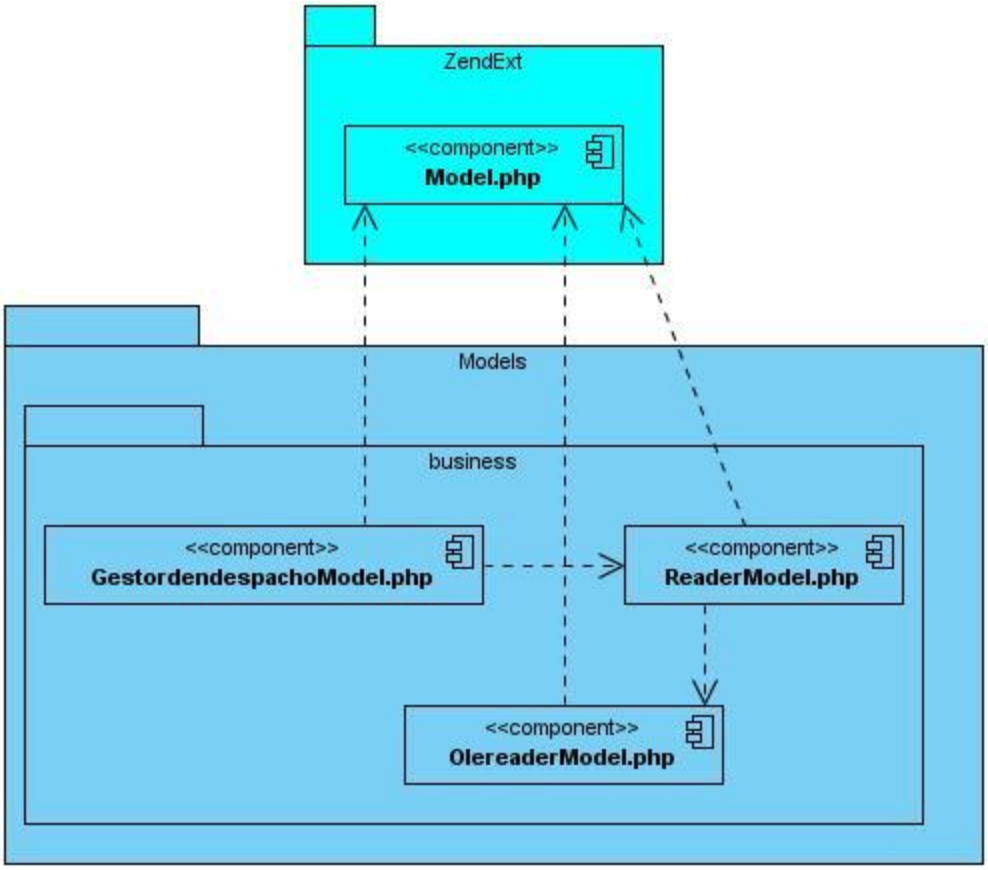


Fig. 15 Paquete Models

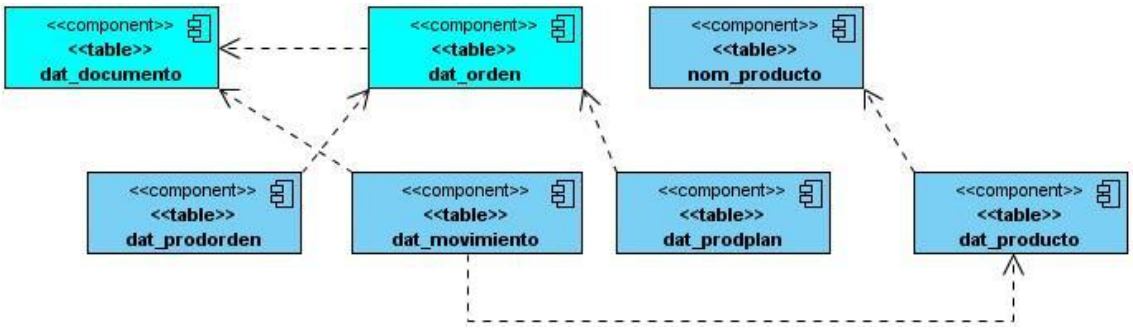


Fig. 16 Paquete Base de Datos

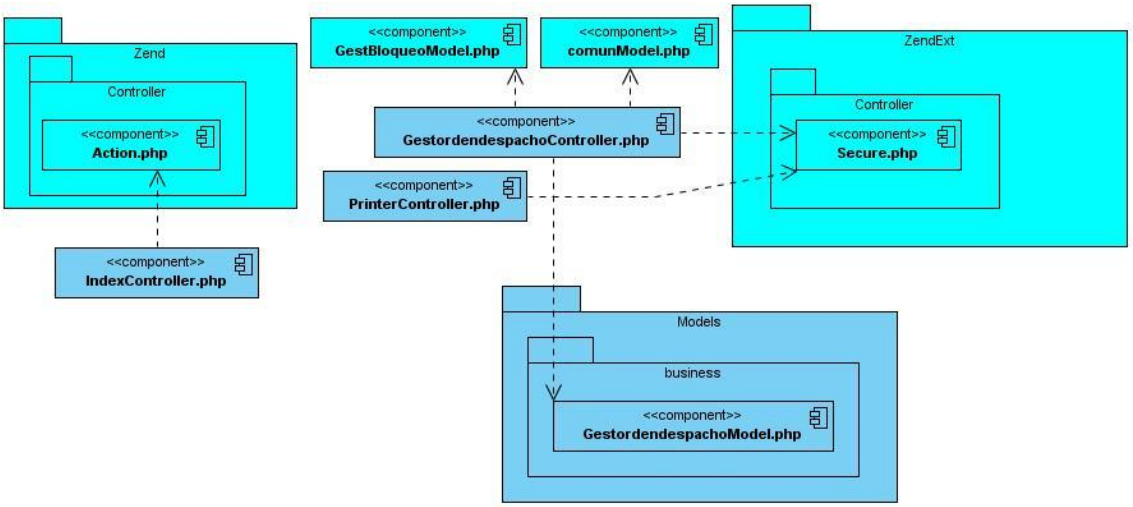


Fig. 17 Paquete Controllers

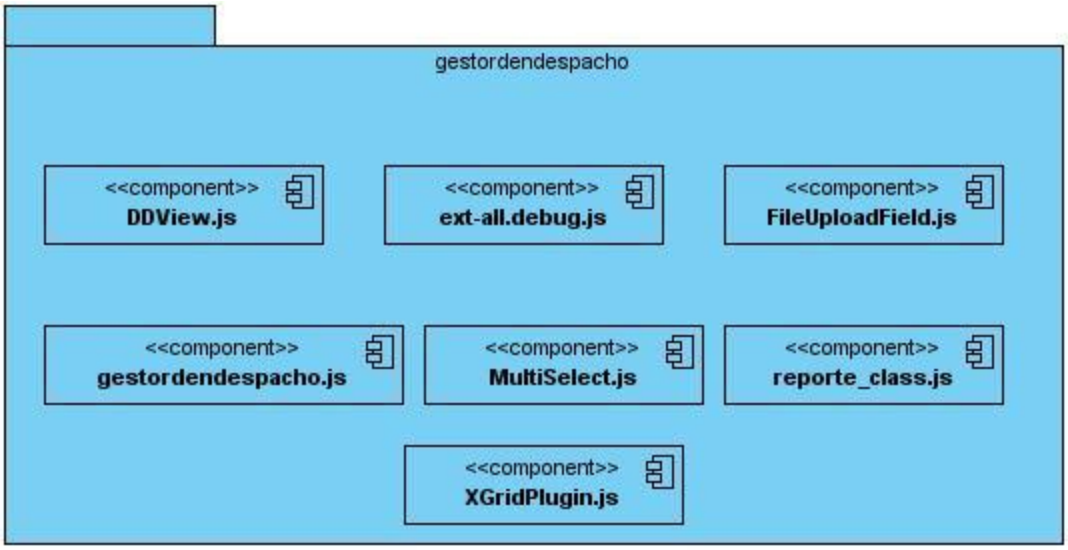


Fig. 18 Paquete de Java scritp

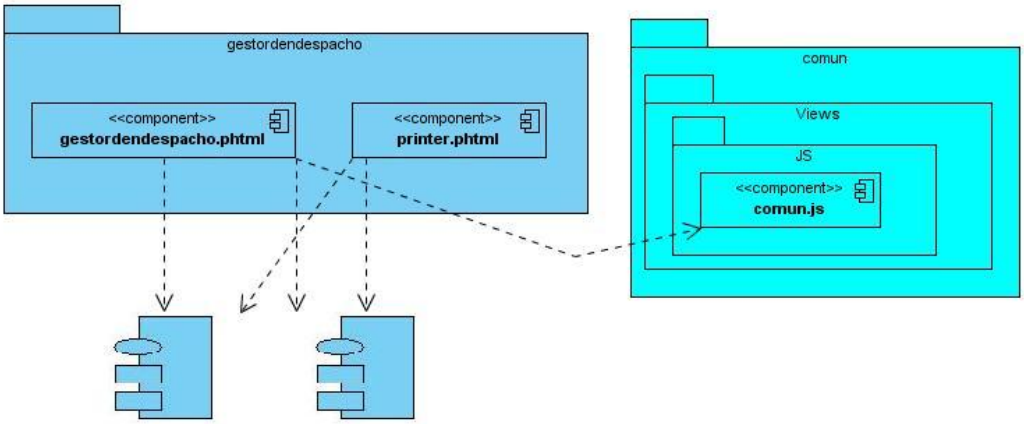


Fig. 19 Paquete Scritp

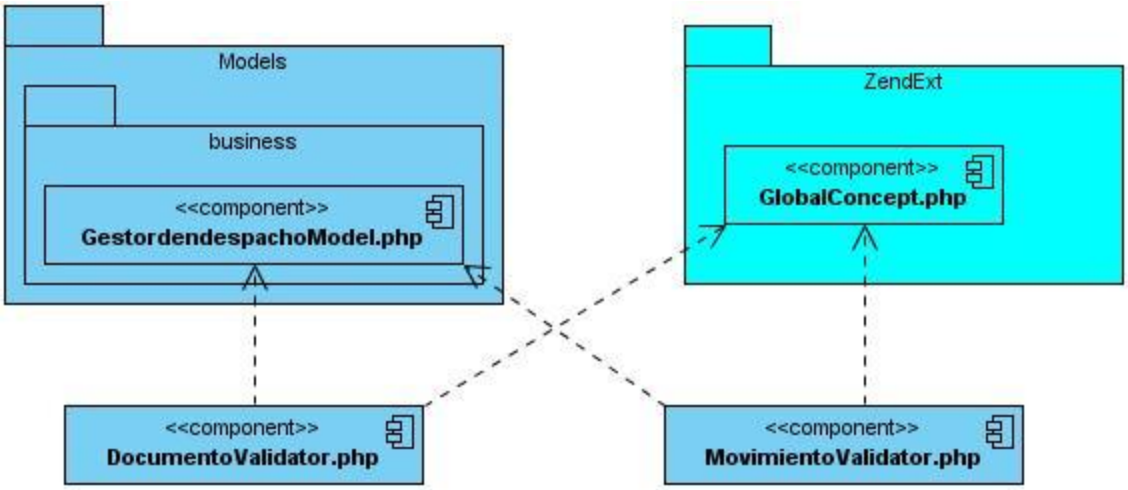


Fig. 20 Paquete Validators

Fig. 21 Paquete Css

ANEXO II

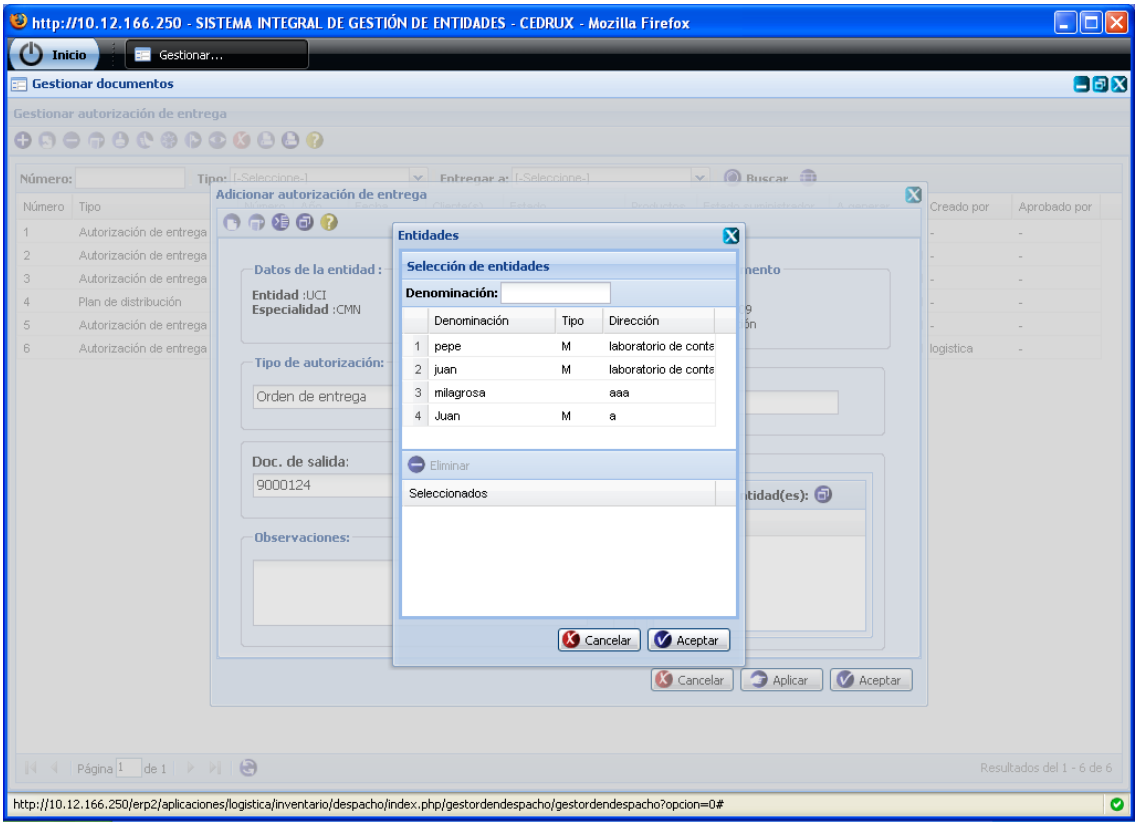


Fig. 22 Adicionar Cliente (Orden de Entrega)

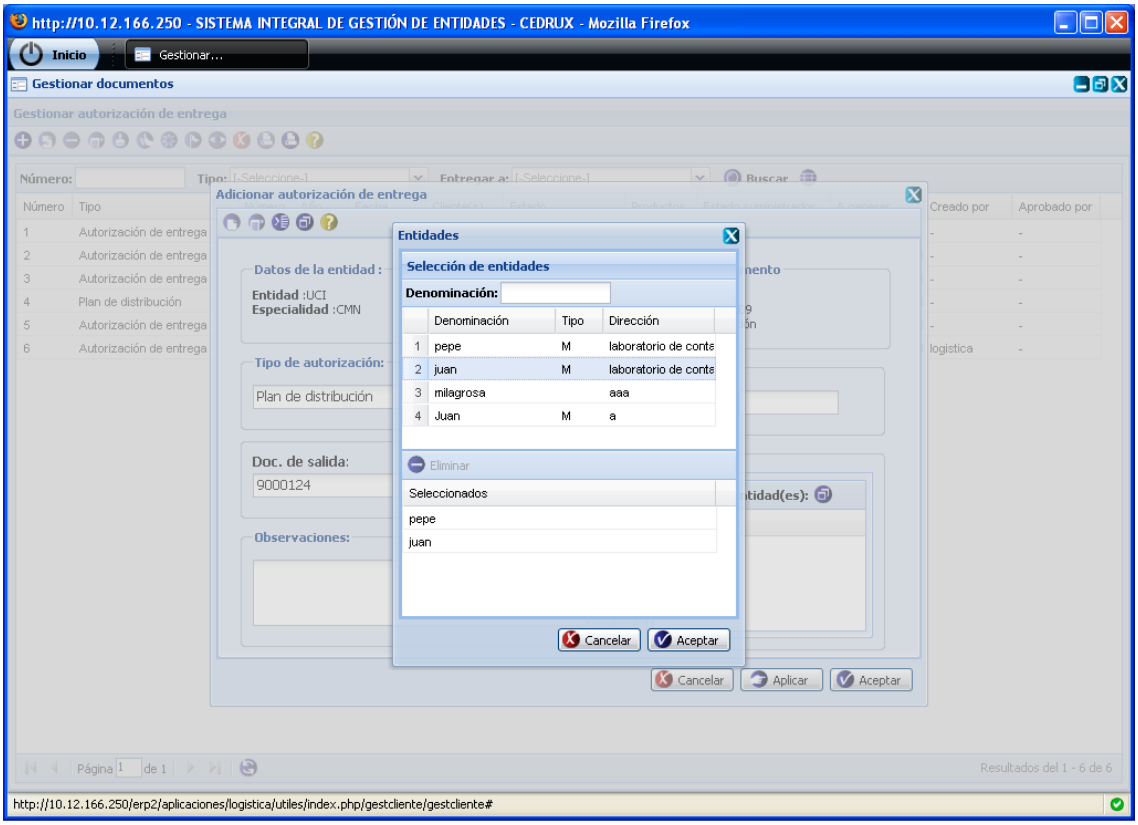


Fig. 23 Adicionar Cliente (Plan de Distribución)

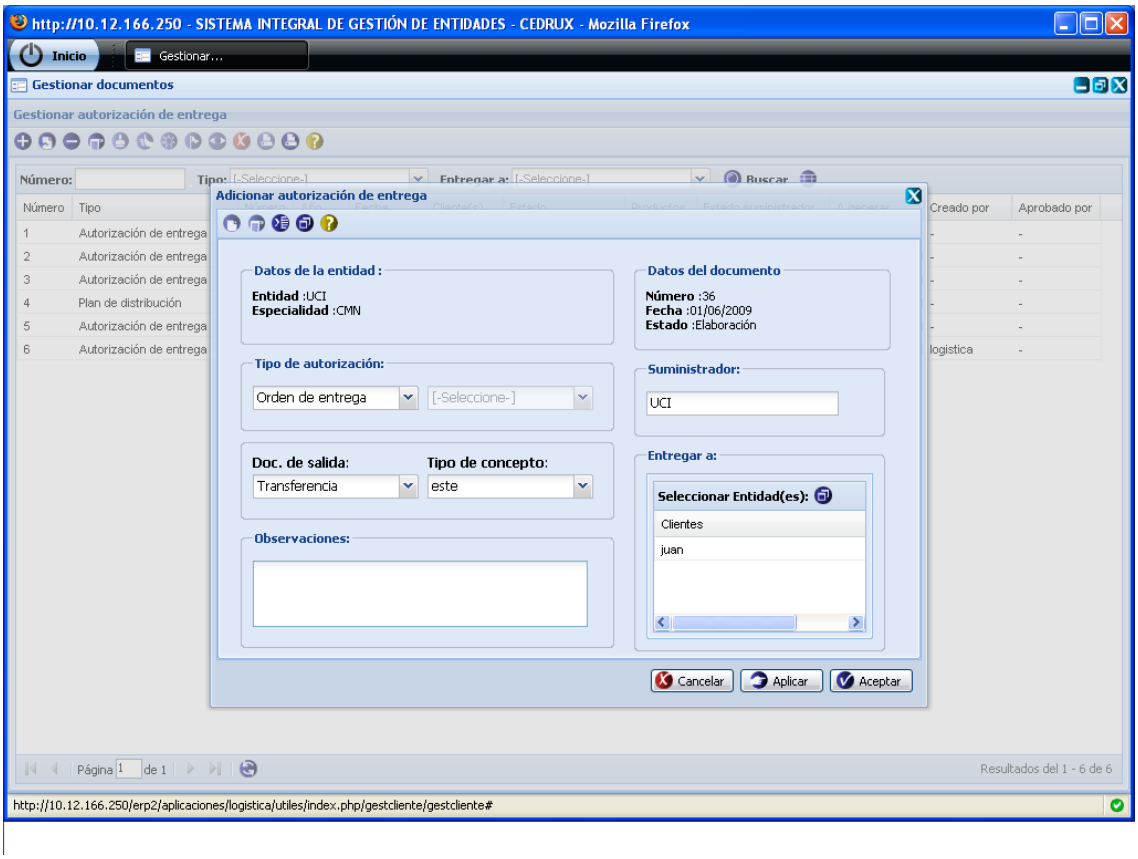


Fig. 24 Adicionar Nuevo Documento (Orden de Entrega)

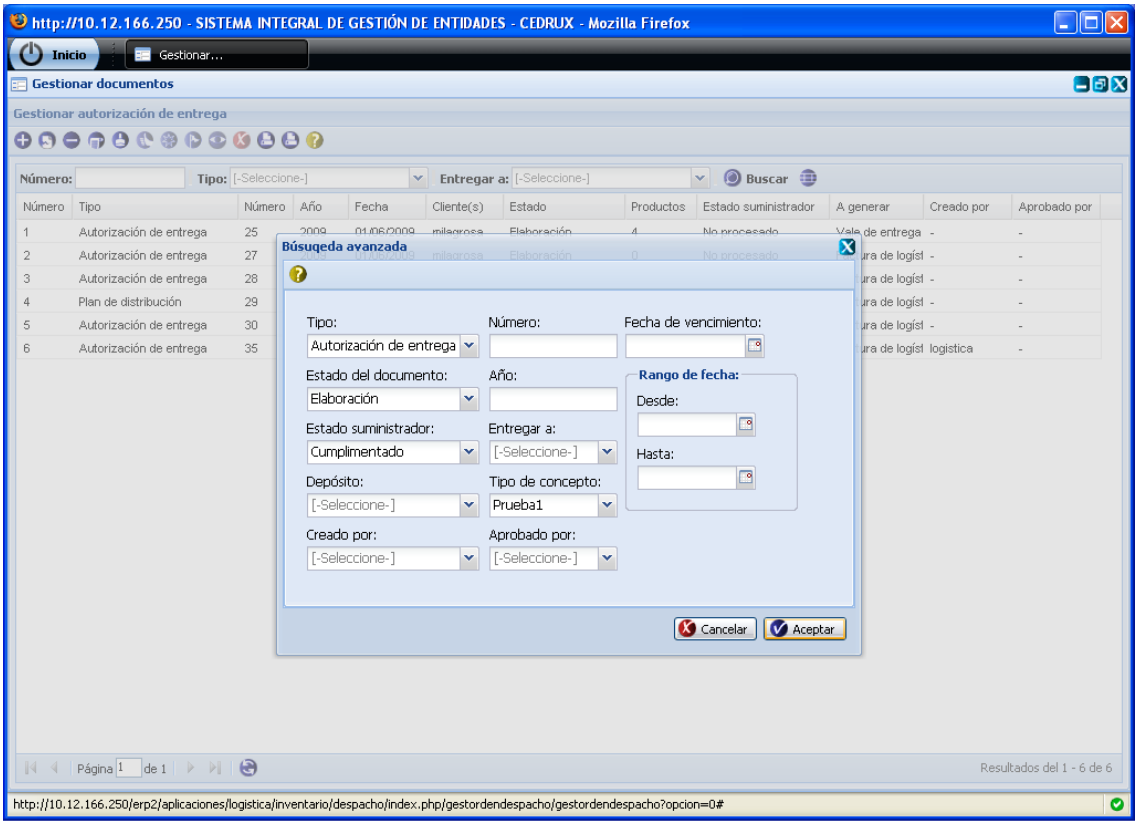
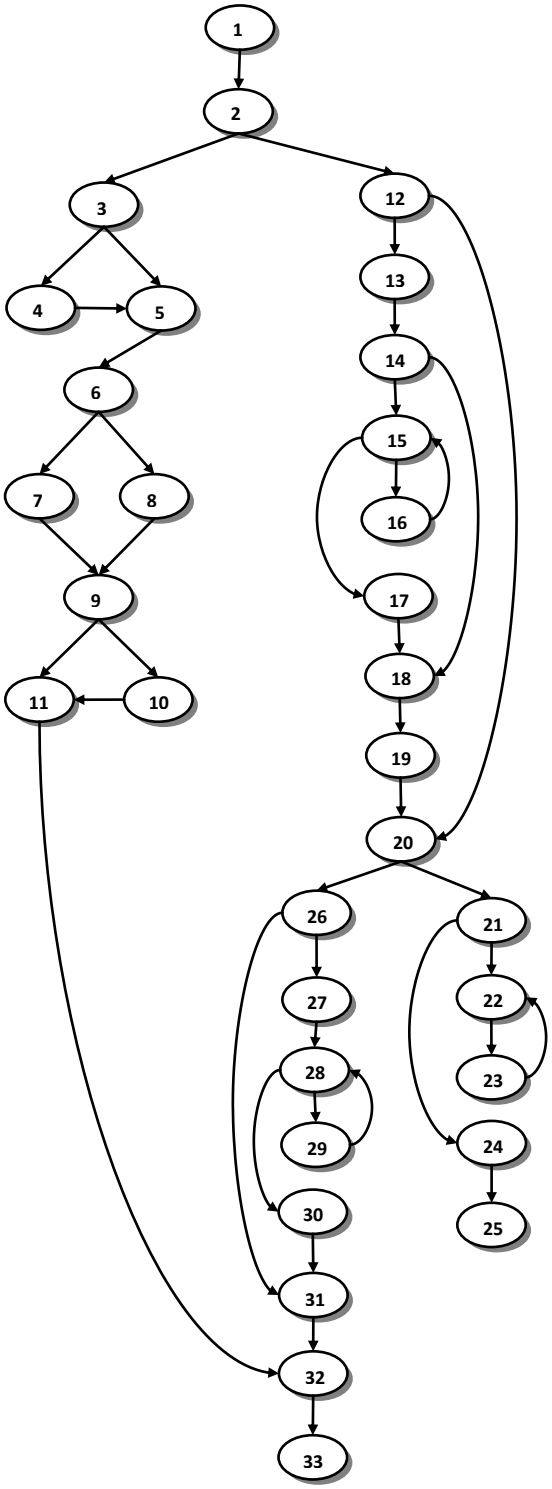


Fig. 25 Búsqueda avanzada (por diferentes criterios)

ANEXO III


1
2
3
12
4
5
13
14
6
15
7
8
16
9
17
18
11
10
19
20
26
21
27
22
28
23
29
24
30
25
31
32
33