Universidad de las Ciencias Informáticas

Facultad 3

Título: "Desarrollo del módulo de Distribución y Pedido de productos para el Sistema Integral de Gestión Cedrux."

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

Autores:
Marieni Arencibia Álvarez.
Jorge Yosmiel Jiménez Quintana.

Tutor:
Ing. Judiht García Rodríguez.
Ing. Leodanny Wuilber Polanco Garay.

La Habana, 2014
"Año 56 de la Revolución"

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 firmamos la presente a los ____ días del mes de _____ del año _____.

Marieni Arencibia Álvarez	Jorge Yosmiel Jiménez Quintana
_____	_____
Firma del Autor	Firma del Autor
Ing. Judiht García Rodríguez	Ing. Leodanny Wuilber Polanco Garay
_____	_____
Firma del Tutor	Firma del Tutor

Y si alguna de las cosas que decimos las explota el enemigo
y nos producen profunda vergüenza,
¡¡bienvenida sea la vergüenza!!...
¡¡bienvenida sea la pena!!,
si sabemos convertir la vergüenza en fuerza,
si sabemos convertir la vergüenza en espíritu de trabajo,
si sabemos convertir la vergüenza en dignidad,
si sabemos convertir la vergüenza en moral"
Fidel Castro Ruz.

Agradecimientos

Jorge Yosmiel Jimenez Quintana:

Le agradezco a la Revolución por la oportunidad de formarme como ingeniero. A los profesores que tanto han aportado durante toda la carrera. Al tribunal por ser tan exigente y obligarnos a esforzarnos tanto en la realización del trabajo. A los profes del proyecto que tanto hemos molestado con dudas y preguntas. A los tutores que tanto nos han ayudado y que han perdido tanto el sueño por nosotros. A los amigos que han estado siempre ahí cuando los he necesitado, a los de siempre a los que conozco desde antes de entrar a la universidad, a los que he conocido durante estos cinco años de alegría, los del aula, los del edificio, los del equipo de pelota, los del futbol, a los profesores que no nos pudieron ganar nunca, a mi traductor simultáneo que me ayudó muchísimo. A mi compañera de tesis que es la mejor que he tenido y que estoy seguro que solo he ella es capaz de resistir mi carácter, a su familia también por la gran ayuda, a mi familia que siempre me ha apoyado, a mi mamá que aunque nunca se lo digo la quiero con la vida y ella ha dado todo para que yo esté aquí hoy y sea la persona que soy, a ella las gracias y las felicidades porque ella siempre ha dicho que yo soy su carrera y entonces esta es nuestra graduación, ahora me corresponde a mi hacer por ella y espero ser tan buen hijo como ella es de buena como madre. También agradecer a alguien que aunque ya no esté siempre me cuida y sé que está conmigo en los momentos difíciles. A todos los que de una forma u otra han aportado a este triunfo en mi vida y espero que estemos siempre juntos.

Marieni Arencibia Alvarez:

Le agradezco a la Revolución por la oportunidad de cumplir mi sueño de ser informática. A los profesores que estuvieron presentes a lo largo de mi carrera enseñándome todo lo necesario para llegar hasta aquí. A mis amigos porque sin ellos este día no significaría nada, gracias por estar cuando se les necesitaba. Gracias también a las personas que no creyeron en mi porque me dieron más fuerza para seguir adelante y cumplir mis metas. A mi mamá gracias por estar para mí siempre, por darme los consejos más útiles cuando en el momento oportuno. A mi papá gracias por enseñarme a trabajar con una computadora desde que tenía tres años, no importa si eran viejas porque desde entonces mi sueño fue ser como tú, gracias por ser mi ídolo. A mi hermana, porque todo lo que he hecho es por ella para darle un buen ejemplo. A todos los que de una forma u otra han aportado a este triunfo en mi vida y espero que estemos siempre juntos.

Dedicatoria

Dedicamos este trabajo a nuestras familias por el apoyo brindado y por creer siempre en nosotros. A nuestros amigos por ser un apoyo en los momentos buenos y malos. Y a los que de una forma u otra han sido testigos y cómplices de este sueño.

Resumen

En el campo de la gestión empresarial los inventarios gestionan un conjunto de procesos entre los que se encuentran la distribución y pedido de productos. Estos procesos se utilizan para el abastecimiento de las líneas de producción así como la distribución directa a clientes y proveedores.

En algunas empresas cubanas estos procesos se realizan de forma manual. Los pedidos son entregados por escrito, teléfono o correo electrónico a las entidades proveedoras por la entidad cliente. Hecho que puede verse afectado por el deterioro o pérdida de dichas solicitudes.

Motivo por el cual en esta investigación se estudian herramientas utilizadas para la gestión de Inventarios, como son los Sistemas Integrales de Gestión. Con el fin de lograr un mejor entendimiento de la gestión de los procesos de distribución y pedido de productos, se realiza un análisis de los principales conceptos. Estudio que permitió llevar a cabo el modelado del negocio y describir además los procesos de negocio de la investigación así como definir todos los requisitos funcionales y no funcionales necesarios para la gestión de la distribución y pedido de productos.

Se generaron además otros artefactos necesarios para el diseño de la solución que permitieron llevar a cabo la implementación, generando los archivos de código con los estándares de codificación definidos por el proyecto. Validándose el correcto funcionamiento de la solución a través de las pruebas de caja negra y caja blanca.

Palabras clave: Cedrux, Distribución, Inventario, Pedido.

Índice

Índice de figuras

Índice de tablas

Introducción

En la Universidad de las Ciencias Informáticas en lo adelante UCI, específicamente en el Centro de Informatización de Entidades (CEIGE) se trabaja en el desarrollo de un Sistema Integral de Gestión (SIG) denominado Cedrux, pensado y ajustado a las necesidades de las entidades empresariales presupuestadas cubanas. Donde actualmente se centran en lograr posibles mejoras a sus funcionalidades e incorporar nuevos requisitos.

"Un SIG integra y automatiza muchas de las prácticas de negocio asociadas con los aspectos operativos o productivos de una empresa. El propósito fundamental de un SIG es otorgar apoyo a los clientes del negocio, brindándole tiempos rápidos de respuesta a sus problemas así como un adecuado manejo de información que permita la toma oportuna de decisiones y una disminución de los costos totales de operación." (1)

En el campo de la gestión empresarial, el inventario registra el conjunto de todos los bienes propios y disponibles para la venta a los clientes, considerados como activos corrientes. Los bienes de una entidad empresarial que son objeto de inventario son las existencias que se destinan a la venta directa o aquellas destinadas internamente al proceso productivo como materias primas, productos inacabados, materiales de embalaje o envasado y piezas de recambio para mantenimiento que se consuman en el ciclo de operaciones. (2)

Los Inventarios en la distribución se utilizan para el abastecimiento de las líneas de producción, así como la distribución directa a clientes y proveedores, generando un enlace con el recibo de los materiales, la transportación interna y la distribución a clientes. Esta clasificación de inventario se forma a partir de la necesidad de mover los productos de un lugar a otro. Los inventarios de distribución se establecen según la localización y forma de transporte, y dependen de la estructura y diseño de la empresa.

Actualmente el manejo de los procesos de distribución y pedido de productos no forma parte de la informatización brindada por el sistema Cedrux, debido a los numerosos procesos que debe informatizar un SIG y al hecho de que dicho sistema no cuenta con un cliente específico que defina claramente sus necesidades.

Los procesos de distribución y pedido de productos están estrechamente relacionados entre sí, debido a que la distribución puede ser realizada basándose en los planes de consumo o en los

pedidos que realizan determinados clientes. Donde expresan la cantidad de los productos que desean adquirir, así como de la disponibilidad de tiempo que precisan para la entrega del mismo.

Para que los productos y más importante, sus beneficios lleguen al pueblo es necesario que existan métodos de pedido y distribución adecuados. La planificación de la distribución de los productos que circulan en Cuba se efectúa de forma manual o a través de software propietario. Estos últimos brindan una solución parcial, implicando considerables gastos en el pago de licencias que pueden alcanzar sumas entre $990.00 y $1610.00 USD por usuario, más 18% del precio del producto por mantenimiento anual; más el IVA dependiendo del tipo de licencia adquirida. Además no realizan un adecuado manejo de la información en todas las áreas de las entidades, lo que trae consigo que en ocasiones se realice un ineficiente proceso de pedido y distribución de productos; provocando que los clientes no estén satisfechos por la tardanza y el deficiente manejo de sus pedidos.

Como se mencionaba anteriormente el proceso de pedido de productos se realiza de forma manual. Para ello es necesario realizar solicitudes de dichos productos por escrito, la cuáles tienen que ser entregadas en las entidades proveedoras por algún representante de la entidad cliente. Este tipo de pedidos puede verse afectado por el deterioro o pérdida de dichas solicitudes. Además pueden efectuarse por vía telefónica, correo electrónico y fax, las cuales se pueden ver afectadas por el mal funcionamiento de los servicios; provocando lentitud en la respuesta a estos pedidos. La información resultante de estos medios de comunicación trae consigo la acumulación de pedidos, en los cuales sería muy complejo evitar la duplicidad de los mismos, viéndose afectado el proceso de distribución.

Es preciso señalar que la mayoría de las dificultades presentadas en las entidades cubanas, para llevar a cabo la gestión de los procesos de distribución y pedido de productos están dadas por un inadecuado manejo de la información.

Por todo lo anteriormente expuesto se plantea como **problema a resolver**: ¿Cómo contribuir a la gestión de los procesos de distribución y pedido de productos en las entidades cubanas que utilicen Cedrux?

Se puede definir entonces como **objeto de estudio** los procesos de distribución y pedido de productos.

Enmarcándose el **campo de acción** la informatización de los procesos de distribución y pedido de productos en el Sistema Integral de Gestión Cedrux.

Para darle solución al problema se plantea como **objetivo general** desarrollar un módulo de distribución y pedido de productos para el Sistema Integral de Gestión Cedrux.

A partir del objetivo general se desglosan los siguientes **objetivos específicos:**

- Elaborar la fundamentación teórica de la investigación sobre el estudio realizado de los procesos de distribución y pedido así como su relación con los Sistemas Integrales de Gestión.
- Realizar el modelado del negocio asociado a los procesos de distribución y pedidos de productos.
- Generar los artefactos correspondientes a la disciplina de requisitos del módulo de distribución y pedido de productos.
- Realizar el análisis y diseño del módulo de distribución y pedido de productos.
- Implementar las funcionalidades correspondientes al módulo de distribución y pedido de productos.
- Validar la solución del módulo realizado a partir de pruebas internas.

Para guiar el desarrollo de este trabajo se plantea como **idea a defender** con el desarrollo del módulo de distribución y pedido de productos para el Sistema Integral de Gestión Cedrux se contribuirá a la gestión de dichos procesos en las entidades cubanas que utilicen Cedrux.

Con el objetivo de proveer las bases necesarias de la investigación se hizo necesaria la utilización de diversos **métodos de investigación**.

Métodos teóricos:

- **Analítico – sintético**: Permitió el estudio y análisis de los diferentes conceptos y definiciones lo cual facilitó la extracción de los elementos más importantes relacionados con los procesos de distribución y pedido de productos del subsistema Inventario de Cedrux. De esta forma se logró entender el tema, facilitando su estudio y definiendo una estrategia para llegar al resultado final con mayor facilidad.

- **Análisis Histórico – lógico**: Permitió constatar teóricamente la evolución del proceso de distribución y pedido de productos en los diferentes Sistemas Integrales de Gestión, sus características y ventajas, para evaluar posibles mejoras a incluir en el sistema.

- **Modelación:** Con este método se logró un mayor entendimiento del problema, pues permite crear abstracciones que explican la realidad a través de los diferentes modelos y diagramas obtenidos.

Esta investigación se desglosa en tres capítulos, en los cuales se abordan de forma simplificada los siguientes contenidos:

Capítulo I Fundamentación teórica: En este capítulo se realiza un análisis de los procesos de distribución y pedido de productos dentro de los sistemas de planificación de recursos empresariales. Al mismo tiempo se aluden y detallan algunos sistemas, tanto nacionales como internacionales que implementan los procesos de distribución y pedido de productos. Finalmente se justifican las herramientas y tecnologías a utilizar para el diseño y la implementación de los módulos pedido y distribución.

Capítulo II Análisis y diseño del sistema: Durante la realización de este capítulo se describen los procesos de negocio, se especifícan los requisitos funcionales y no funcionales del sistema. Además se genera el Modelo Conceptual en el cual quedan expuestos los conceptos fundamentales y las relaciones entre ellos. Se realiza el diagrama de clases del análisis con el fin de sentar las bases para una adecuada implementación de la solución. Finalmente se definen los patrones de diseño a utilizar, obteniéndose el diagrama de clases del diseño.

Capítulo III Implementación y Prueba: A partir de los resultados del análisis y diseño de la solución se lleva a cabo la implementación de los módulos de distribución y pedido de productos generando los ficheros de código fuente así como los scripts. Conjuntamente son definidos los estándares de codificación que se utilizan para la obtención de un código legible que facilite el entendimiento del mismo. Además se realiza la validación de la solución propuesta a través de las pruebas de caja negra con el fin de eliminar las no conformidades del sistema y así mejorar la calidad del mismo.

Capítulo I Fundamentación teórica

1.1. Introducción

En este capítulo se realiza un estudio detallado de los sistemas integrales de gestión empresarial más utilizados a nivel nacional e internacional. Se recopila información detallada acerca de cómo estos sistemas realizan la gestión de pedidos y distribución de productos, así como de las tecnologías utilizadas para el desarrollo de los mismos. Para un mejor entendimiento de la problemática se hace alusión a los conceptos fundamentales que se refieren a lo largo del presente trabajo de diploma, así como se realiza un análisis de las ventajas y desventajas que proporciona a las entidades cubanas el uso de un sistema de este tipo. En este capítulo además se presenta un estudio donde se reflejan las principales características de las herramientas, tecnologías, lenguajes y modelo de desarrollo que se utilizan para el desarrollo de la solución propuesta.

1.2. Conceptos generales

Sistema Integrado de Gestión (SIG)

Según Cesar Camisón, Doctor en Ciencias Económicas y Empresariales, un **Sistema Integrado de Gestión (SIG)** es una plataforma común para unificar los sistemas de gestión de la organización en distintos ámbitos en uno sólo, recogiendo en una base documental única los antes independientes manuales de gestión, procedimientos, instrucciones de trabajo, documentos técnicos y registros, realizando una sola auditoría y bajo un único mando que centraliza el proceso de revisión por la dirección. (3)

Según Dra. Rosa Mayelín Guerra Bretaña y MSc. María del Carmen Meizoso el **SIG** es una apuesta indispensable que permite una gestión transversal en materias sensibles para la empresa, sus trabajadores y la sociedad. La realización de las soluciones organizativas de manera independiente una de otra, crea un sistema de dirección dividido, lo que se trata es de ver las interrelaciones para construir un sistema único de dirección en la empresa donde se vayan incorporando coherentemente las nuevas soluciones organizativas, para elevar la eficacia y la eficiencia en la toma de decisiones a corto y a largo plazo. (4)

En la presente investigación luego del análisis de los principales conceptos expresados anteriormente, se llega a la conclusión que un SIG es un sistema que permite automatizar e

integrar los procesos que se realizan en las diferentes áreas de una empresa permitiéndole un adecuado manejo de la información en todas las ramas.

Planificación de Recursos Empresariales (ERP)

Los sistemas de Planificación de Recursos Empresariales, ERP por sus siglas en inglés, son Sistemas Integrales de Gestión que además muestran características de adaptabilidad y modularidad. (5)

Los ERP pueden intervenir en el control de muchas actividades de negocios, tales como ventas, compras, entregas, pagos, producción, contabilidad, administración de inventarios y de recursos humanos. Funcionan en todo tipo de empresas. Para ello, integran todos los departamentos funcionales de la empresa, herramientas de mercadotecnia y administración estratégica en un solo sistema. (6)

Distribución

Distribución es la acción y efecto de distribuir (dividir algo entre varias personas, dar a algo el destino conveniente, entregar una mercancía). El término, que procede del latín *distributio*, es muy habitual en el comercio para nombrar al reparto de productos. (8)

Las decisiones sobre las distribuciones se toman teniendo en cuenta las necesidades de los compradores y los usuarios. Para ello se deben conocer las respuestas de los consumidores a las preguntas: ¿Dónde, cuándo, qué, cómo, cuánto compran? Por medio de la investigación de mercados la empresa se informa sobre sus preferencias y las satisface llevando los productos demandados, en la forma deseada, al lugar solicitado, la cantidad requerida en el momento indicado.

La distribución, en este trabajo, es el proceso que consiste en hacer llegar físicamente el producto al consumidor. Dicha distribución puede ser realizada basándose en los planes de consumo o en pedidos realizados por determinados clientes.

La distribución sin pedido se basa en los planes de consumo que están estrechamente relacionados con los circuitos de distribución a los que pertenece cada cliente, dichos circuito tiene asociados un conjunto de productos.

La distribución con pedido consiste en actividades que resultan de la cumplimentación de órdenes de pedido del cliente, a la vez que se asegura el máximo valor de la cadena de suministro y servicio al cliente. La distribución debe cumplir con los siguientes Requisitos:

- Necesidades del cliente.
- Requisitos del cliente.
- Información del pedido
- Plazos de entrega.

Los Sistemas de Distribución permiten a las empresas distribuidoras gestionar y controlar el proceso en el cuál se realizan los pedidos y la distribución de ellos en todo momento durante el ciclo de distribución. Gestionando así la red de distribución de cada entidad relacionada, ya sean clientes o proveedores.

Pedido

Petición que un cliente hace a un proveedor para que este suministre los bienes o servicios solicitados.

Una **orden de compra** o **nota de pedido** es un **documento** que el comprador entrega al vendedor para **solicitar ciertas mercaderías**. En él se detallan la cantidad a comprar, el tipo de producto, el precio, las condiciones de pago y otros datos importantes para la operación comercial. Por lo general, la orden de compra menciona el lugar y fecha de emisión, el nombre y domicilio del comprador y del vendedor, datos impositivos, detalles de las mercaderías pedidas y condiciones de pago y entrega. Es importante que se aclare que dicho documento no es válido como **factura**. (9)

1.3. Sistemas informatizados existentes

Se hizo necesario realizar un estudio de los Sistemas Integrales de Gestión con el propósito de identificar posibles mejoras a incluir en el subsistema Inventario de CEDRUX. Para acometer este estudio se tuvieron en cuenta los sistemas más utilizados tanto, a nivel nacional como internacional que gestionan los procesos de distribución y pedido de productos.

1.3.1. Sistemas utilizados en Cuba

Stock Mistral

Es una herramienta útil para facilitar la toma de decisiones. Está diseñado para disminuir los niveles de stock[1], reducir los recursos inmovilizados e incrementar la liquidez en la empresa. Posee herramientas que crean, en tiempo real, almacenes virtuales a diferentes niveles, con las existencias y movimientos de sus dependencias, lo cual permite organizar el proceso de compras, y distribución. La consulta de esta información se realiza sobre sitios web ajustándose a los procedimientos internos de cada cliente.

Stock Mistral brinda una solución parcial a los problemas presentes en el proceso de distribución que se realiza en el país dado que sólo realiza las distribuciones sin pedido y está desarrollado sobre plataformas de software propietario. Aún así este sistema permitió estudiar a fondo las distribuciones sin pedido basadas en los planes de consumo de cada entidad.

SAP R/3

SAP R/3 es un sistema de gestión integral que opera utilizando el principio cliente/servidor aplicado a varios niveles. Es altamente modular y se aplica fundamentalmente por medio del software, de forma que los modos de interacción entre los diversos clientes y servidores puedan ser controlados. Incluye los módulos de contabilidad y análisis financiero, recursos humanos, manufactura, logística, ventas y distribución. Se encuentra actualmente en explotación en nuestro país en la Empresa de Telecomunicaciones ETECSA.

Además de estas soluciones estándares, el ambiente de desarrollo de SAP y su sistema de información, provee a los clientes de herramientas para el desarrollo y adaptación del sistema a los requisitos individuales (personalización). Proporciona un amplio rango de servicios lo cual es una de las causas del éxito del sistema R/3. El sistema SAP R/3 es un sistema integrado, esto significa que una vez que la información es almacenada está disponible a través de todo el sistema, facilitando el proceso de transacciones y el manejo de información. (10)

Este sistema a pesar de ser una solución bastante completa no es accesible para todas las empresas de nuestro país ya que sólo es compatible con sistemas operativos propietarios como Windows. Sin embargo este sistema cuenta con un módulo de Ventas y Distribución el cual

[1] Conjunto de recursos ociosos que tienen un valor económico y que están pendientes de ser vendidos

permitió el estudio detallado de las distribuciones por pedidos y sin pedidos. Facilitó además el estudio y entendimiento de la rentabilidad de los circuitos de distribución. Del mismo modo el análisis de la herramienta SAP aporta una clara visión de cómo este sistema realiza la gestión de la información y permite la integración de los pedidos y la distribución, brindando un adecuado seguimiento de los pedidos y manejo de las distribuciones.

1.3.2. Sistemas utilizados internacionalmente

VirtualWork ERP

VirtualWork ERP implementa el sistema de Planificación de Recursos Empresariales ERP web para gestionar la distribución de los productos en la ciudad de Copiapo y otras ciudades próximas, ubicadas en Chile. El sistema permite realizar el proceso de distribución logrando un mejor control del inventario. (11) En la actualidad VirtualWork ofrece aplicaciones para diversos mercados, entre los que podemos resaltar:

- Logísticas y Distribución.
- Gestión Automotriz.
- Administración de proyectos.
- Desarrollo a Solicitud del Cliente.

VirtualWork incluye más de 460 módulos distintos entre los que podemos resaltar:

- Administración de Producción.
- Gestión de Inventario.
- Gestión de Distribución.
- Administración de políticas Comerciales.
- Recurso Humanos.
- Facturación Electrónica.
- Contabilidad.

VirtualWork incluye varios módulos para gestionar la distribución como son:

- Políticas comerciales.
- Toma de pedidos.
- Pedidos remotos a través de equipos móviles.

- Ruteo de pedidos.

Este sistema aunque trabaja en varias esferas, no abarca todas las áreas de las empresas cubanas. VirtualWork está creado para acceder al servidor central que se encuentra en Chile a través de una conexión a internet, esta vía aún no es accesible para las empresas de nuestro país. Este cuenta con un módulo de distribución que recibe los pedidos y se preparan para su posterior distribución, se asignan los pedidos a los cuales se les puede dar respuesta y se realiza la distribución teniendo en cuenta los que serán despachados en un mismo transporte; una variante acertada la cual ayudo a definir la propuesta a implementar en la solución de esta investigación como circuito de distribución y áreas MPV.

SOLUFLEX ERP

SOLUFLEX ERP se creó basado en las empresas de Distribución masiva de productos, específicamente con el objetivo de cumplir con los requisitos y necesidades de cada una de las áreas comerciales y cuenta con el soporte tecnológico necesario para controlar las áreas críticas de ese sector. Es un sistema Integrado de Gestión Administrativa ideal para Negocios en pleno crecimiento, como medianas y grandes empresas. Opera en una red informática dentro de un Negocio, este sistema trabaja con una base de datos unificada donde las áreas de la empresa pueden compartir información evitando la redundancia de funciones y el doble registro de información al utilizar sistemas que usan datos aislados e independientes. (12)

Éste ERP contiene una gran cantidad de módulos entre los que se encuentran:

Módulo Comercial

- Toma de Pedidos (Pre-Venta).
- Manejo de Zonas/Rutas/Módulos/Recorrido.
- Distribución Masiva por Zonas/Rutas/Módulos/Recorrido.
- Registro de Ventas.
- Ranking de Productos.

El sistema se encarga de la venta y distribución, registrando datos de la ubicación de los Clientes como la Zona/Ruta/Módulo/Frecuencia de Visita/Recorrido para la Fuerza de Ventas. Además también se encarga de la Toma de Pedidos, y Facturación Automática de Pedidos, que garantizan un correcto funcionamiento en el proceso de la gestión de la distribución de los productos.

Módulo Logística

- Solicitudes de Compra – Abastecimiento.
- Selección de Proveedores.
- Órdenes de Compra.
- Ingresos Almacén con Orden de Compra.
- Movimiento de Inventarios.
- Maestro de Almacenes.
- Maestro de Proveedores.
- Maestro de Productos.

SOLUFLEX ERP es un gran competidor a nivel mundial por el gran alcance que tiene dentro de las empresas, pero su principal desventaja es que solo está disponible para plataformas Microsoft y es necesario adquirir la licencia correspondiente para la utilización del mismo.

SAP

SAP es uno de los ERP más utilizados mundialmente, tiene un gran número de funciones y áreas cubiertas dentro de las empresas; han desarrollado un programa de socios que garantiza la experta implementación en todo el mundo. Debido al gran número de funciones, SAP puede llevar más tiempo para ponerlo en práctica. (13)

Para garantizar un servicio al cliente consistente y de máxima calidad, puede gestionar eficazmente tanto los pedidos de pequeño volumen como los de gran volumen, comprobar la disponibilidad de productos y realizar un seguimiento de los pedidos. (13)

Este sistema a pesar de ser una solución bastante completa no es accesible para todas las empresas de nuestro país ya que es software propietario e implica altos pagos por conceptos de licencias. Pero aun así se utilizó como objeto de estudio ya que incluye funcionalidades claves para gestionar los Circuitos de Distribución, así como procesos de distribución y pedido de productos bastante cercanos a los que se realizan en las entidades cubanas.

1.3.3. Resultados de los sistemas estudiados

Al analizar los sistemas de distribución y pedidos de productos utilizados en Cuba y el mundo, se concluye que estos no son soluciones accesibles para la gran mayoría de las empresas de nuestro país, primeramente constituyen software propietarios los cuales incurren en gastos por

concepto de licencia. Cabe destacar que estos sistemas brindan en su gran mayoría soluciones parciales las cuales pudieran ser explotadas en nuestro país y que sirvieron de gran ayuda para el desarrollo de esta investigación, pero estas no pueden ser integradas al Sistema Integral de Gestión Cedrux por las herramientas y tecnologías utilizadas para el desarrollo de las mismas.

Como se mencionaba anteriormente el análisis de estos sistemas posibilitó evaluar diversos elementos fundamentales en la gestión de los procesos de distribución y pedido de productos. Permitió estudiar a fondo las distribuciones sin pedidos basadas en los planes de consumo de cada entidad, así como hacer un detallado análisis de las distribuciones por pedidos. Facilitó el entendimiento de la rentabilidad de los circuitos de distribución así como el análisis de herramientas que aportan una clara visión de cómo se realiza la gestión de la información que permite integrar los pedidos y las distribuciones.

El estudio antes realizado aportó elementos fundamentales para la informatización de estos procesos con el fin de obtener un módulo de Distribución y Pedido de Productos que pueda satisfacer las necesidades de las entidades cubanas que utilicen Cedux.

1.4. Modelo de desarrollo

Modelo de Desarrollo de Software del centro CEIGE:

El ciclo de vida de los proyectos de CEIGE (Figura 1) tiene en cuenta las actividades de cada una de las fases y áreas de procesos que plantea el nivel dos de CMMI establecido en la UCI. Esta abarca el total de acciones que se realizan en las distintas líneas de desarrollo para la elaboración del servicio o producto final, sin embargo, se debe adaptar a las características particulares del proyecto que puede que no ejecute determinada disciplina, así como la elaboración de determinados artefactos del total aquí definido.(14)

Figura 1 Ciclo de vida de proyectos del CEIGE.

Estudio preliminar:

Se llevan a cabo las actividades relacionadas con la planeación del proyecto a un alto nivel. Se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto y realizar estimaciones de tiempo, esfuerzo y costo, y decidir si se ejecuta o no el proyecto.

Modelado del negocio:

Es la fase destinada a comprender los procesos de negocio de la organización. Se comprende cómo funciona el negocio que se desea automatizar para tener garantías de que el software desarrollado va a cumplir su propósito.

Requisitos:

El esfuerzo principal en la fase de Requisitos es desarrollar un modelo del sistema que se va a construir. Incluye un conjunto de artefactos que describen todas las interacciones que tendrán los usuarios con el software y que responden a los requisitos funcionales del sistema. Se especifican los requisitos funcionales y no funcionales. Actividades de la disciplina Requisitos.

Análisis y diseño:

Durante esta fase es modelado el sistema para que soporte todos los requisitos. Esto contribuye a una arquitectura sólida y estable que se convierte en un plano para la próxima fase. Los artefactos generados en esta etapa son más formales y específicos de una implementación. En caso de llevarse a cabo la reutilización de componentes software ya desarrollados, durante esta fase se ajusta el modelado existente a los requisitos actuales.

Implementación:

A partir de los resultados del análisis y diseño se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ejecutables y similares. Al reutilizar componentes software ya implementados se lleva a cabo el desarrollo necesario para ajustar a los requisitos actuales y posteriormente realizar la integración de los componentes.

Pruebas internas:

Durante esta fase se desarrollan las pruebas del grupo de calidad del centro verificando el resultado de la implementación. Permite identificar posibles errores en la documentación y el software, es decir requisitos que el producto debería cumplir y que aún no los cumple.

Pruebas de liberación:

Se aplican pruebas diseñadas e implementadas por el Laboratorio Industrial de Pruebas de software a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.

A continuación se presentan los artefactos que serán generados al finalizar cada una de las disciplinas del modelo de desarrollo utilizado para guiar esta investigación, en correspondencia con los intereses del proyecto al cual tributa la misma:

Modelado del Negocio:

- Modelo conceptual.
- Glosario de términos.
- Mapa de procesos de negocio.
- Descripción de procesos de negocio.
- Reglas de negocio.

Requisitos:

- Especificación de requisitos de software.
- Descripción de requisitos.

Análisis y diseño:

- Diseño de casos de prueba.
- Modelo de datos.
- Diccionario de datos.
- Diagrama de componentes.
- Diagrama de clases del diseño.
- Diagramas de interacción.

Implementación:

- Ficheros de código.

Pruebas Internas:

- Acta de liberación.

1.5. Herramientas y tecnologías a utilizar

Para llevar a cabo el desarrollo del módulo distribución y pedido de productos se hace necesaria la descripción de un conjunto de tecnologías y herramientas, las mismas fueron definidas por el proyecto al cual tributa esta investigación. Además, el módulo que se propone como solución en la investigación se integra al Sistema Integral de Gestión Cedrux específicamente en el subsistema de Inventario, por lo cual es necesaria la compatibilidad entre las tecnologías a utilizar.

1.5.1. Herramienta CASE

Las herramientas CASE (*Computer Aided Software Engineering*), Ingeniería de Software Asistida por Ordenador, son el conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores para aumentar la calidad del software reduciendo el

esfuerzo, el costo y el tiempo. Además de estructurar la documentación asociada a los artefactos generados, facilitan el desarrollo de software desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas.

Visual Paradigm 8.0 para el Modelado UML

Se seleccionó por parte del proyecto el Visual Paradigm para el modelado UML por ser un producto de calidad que soporta aplicaciones web en varios idiomas, siendo fácil de instalar y actualizar, además de tener compatibilidad entre todas sus ediciones. Otra de sus ventajas es la disponibilidad en múltiples sistemas operativos como Windows, Linux, Unix.

Es una herramienta UML[2] profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Ayuda a una rápida construcción de aplicaciones de gran calidad, mejoras y a un menor costo de producción. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Forma parte del IDE de Eclipse. Está diseñado para desarrollar software con Programación Orientada a Objetos, permite reducir la duración del ciclo de desarrollo brindando ayuda a los arquitectos, analistas, diseñadores y desarrolladores. Busca también automatizar tareas tediosas que pueden distraer a los desarrolladores. Dentro de sus características fundamentales están: (15)

- Multiplataforma: Soportado en plataformas Java para Sistemas Operativos Windows, Linux y Mac.
- Interoperabilidad: Intercambia diagramas UML y modelos con otras herramientas. Soporta exportar e importar a XML y archivos Excel. Importa archivos de proyectos de *Rational Rose*. Integración con Microsoft Office Visio.

Servidor Web

Un servidor Web es un programa que está diseñado para transferir hipertextos, páginas Web o páginas HTML (*Hyper Text Markup Language*): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música. Además sirve contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante HTTP. (16)

2 UML: (UnifiedModelingLanguage) Lenguaje Unificado de Modelado

Apache 2.2.9

Entre sus principales características se encuentran:

- Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- Es una tecnología gratuita de código fuente abierto.
- Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.
- Es altamente configurable en cuanto a la creación y gestión de logs[3]. (16)

1.5.2. Gestor de bases de datos

Sistema Gestor de Base Datos: Los sistemas de gestión de bases de datos o SGBD (en inglés *Database Management System*, abreviado DBMS) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos. (17)

PostgreSQL8.3

Es un sistema de gestión de bases de datos relacional orientada a objetos, libre y multiplataforma, publicado bajo la licencia BSD (*Berkeley Software Distribution*). Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una sola empresa sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo. Dicha comunidad es denominada el PGDG (PostgreSQL *Global Development Group*). (18)

1.5.3. Framework de desarrollo

Un marco de trabajo es una estructura de software compuesta por componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework o marco de trabajo se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta, puede incluir soporte de programas, bibliotecas y un lenguaje de *scripting* entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. El marco de trabajo que se

3Logs: Registro de errores.

empleará para la realización de este trabajo es Sauxe, el cual posee lenguajes, *frameworks* y tecnologías que facilitarán la implementación del sistema que se desea obtener. (19)

Sauxe 2.0

El Marco de Trabajo Sauxe ha sido estructurado de manera tal que facilite la reutilización de los diferentes componentes y que sea de fácil entendimiento para todos los programadores que desarrollen sobre el mismo. Sauxe utiliza el framework Ext, para implementar la capa de presentación, se apoya en Zend_Ext, una extensión de Zend Framework para el desarrollo de la lógica del negocio y para la gestión de los datos utiliza Doctrine. Este utiliza como patrón arquitectónico Modelo Vista Controlador (MVC). También tiene como propósito insertar la programación orientada a aspectos así como la inversión de controles. Además Sauxe implementa mecanismos de autenticación y autorización, mecanismos de mensajería y control de excepciones, gestión y configuración dinámica de precondiciones, pos-condiciones y validaciones de variables, gestión y configuración de flujos de trabajo, visualización de las funcionalidades de un sistema, entre otras funcionalidades. (19)

ExtJS Framework 2.2

Es un framework JavaScript del lado del cliente para el desarrollo de aplicaciones Web. ExtJS posee una librería inmensa que permite configurar las interfaces Web de manera semejante a aplicaciones desktop, incluye la mayoría de los controles de los formularios Web basándose en *Grids* para mostrar datos y elementos semejantes a la programación desktop como los formularios, paneles, barras de herramientas, menús y muchos otros. Dentro de su librería contiene componentes para el manejo de datos, lectura de XML, lectura de datos JSON e implementaciones basadas en AJAX. Usar un motor de *render* como ExtJS permite entre otros beneficios un balance entre Cliente – Servidor, la carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo; una comunicación asíncrona. En este tipo de aplicación el motor de *render* puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente se dé cuenta, además de que facilita eficiencia de la red, el tráfico de red puede disminuir al permitir que la aplicación elija que información desea transmitir al servidor y viceversa, sin embargo la aplicación que haga uso de la pre -carga de datos puede que revierta este beneficio por el incremento del tráfico. A continuación se muestra la descripción de las clases pertenecientes a este framework. (20)

Doctrine Framework 1.2.1

Doctrine es un potente y completo sistema ORM (*object relational mapper*) para PHP 5.2+ con un DBAL (*data base abstraction layer*) incorporado, el mismo cuenta con disimiles funcionalidades, una de ellas es la que brinda 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 y trabaja de manera rápida y eficiente. (21)

Zend Framework 1.7

Es un framework para el desarrollo de aplicaciones Web y servicios Web con PHP. Brinda soluciones para construir sitios web modernos, robustos y seguros. Además es *Open Source* y trabaja con PHP 5. Zend Framework utiliza el estilo MVC (Modelo Vista Controlador) 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. Cuenta con un importante mecanismo de manejo de controladores y vistas. (22)

Dentro de sus principales características están:

- Proporciona los componentes que forman la infraestructura del patrón MVC.
- Proporciona una capa de acceso a base de datos, construida sobre PDO [4] pero ampliándola con diferentes características.
- Proporciona mecanismos de filtrado y validación de entradas de datos.
- Permite convertir estructuras de datos PHP a JSON y viceversa, para su utilización en aplicaciones AJAX.
- Proporciona capacidades de búsqueda sobre documentos y contenidos.
- Permite consumir y proveer servicios web.
- Incluye objetos de las diferentes bases de datos, por lo que es extremadamente simple para consultar la base de datos.
- Completa documentación y pruebas de alta calidad.

[4] PDO: (PHP Data Objects) Capa de abstracción de acceso a datos para PHP

- Robustas clases para autenticación y filtrado de entrada.

Proporciona un sistema de caché dividido en *frontend y backend*, de forma que se puedan almacenar en caché diferentes datos como resultados de funciones, páginas completas, entre otros. (11)

Zend_Ext Framework

Es un framework *Open Source*, que está diseñado para php5 y buenas capacidades de ampliación. Es elaborado a partir de Zend Framework cumpliendo con todas sus características. Este trae de novedoso un controlador vertical para el control de las acciones realizadas por las vistas hacia el controlador, un motor de reglas para las validaciones en el servidor. Se le incluyó el IoC[5] para la comunicación entre los módulos o componentes, la integración con el ORM Doctrine Framework para trabajo en la capa de abstracción a base de datos, el ExtJS Framework para el desarrollo de las vistas.

1.5.4. IDE de desarrollo NetBeans 6.9

El ambiente de desarrollo (Development Environment) es algo imprescindible en la producción de software. Es donde se definen el conjunto de herramientas y tecnologías (frameworks), versiones a usar y su integración, que intervienen en un proceso de desarrollo de software. Un entorno de desarrollo integrado o *Integrated Development Environment* (IDE), en inglés, es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien, poder utilizarse para varios. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Los IDEs pueden ser aplicaciones por si solas o pueden ser parte de aplicaciones existentes. Estos proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Java, C#, Delphi, Visual Basic y Object Pascal. Es posible que un mismo IDE pueda funcionar con varios lenguajes de programación. (23)

1.5.5. Lenguajes de programación y etiquetado

En la actualidad existen numerosos lenguajes de programación para desarrollar la web, tanto lenguajes de lado cliente como de lado servidor. Cuando se habla de lenguajes del lado cliente,

[5]Inversión de control (*Inversion of Control* en inglés, IoC) sucede cuando es la biblioteca la que invoca el código del usuario.

son aquellos capaces de ser digeridos directamente por el navegador sin necesidad de un pre - tratamiento, entre los lenguajes utilizados se encuentran los siguientes:

PHP 5

PHP (*Hypertext Preprocessor*) es un lenguaje interpretado, diseñado especialmente para desarrollo web y puede ser incluido dentro de código HTML. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. A continuación se muestran algunas de las principales ventajas que ofrece PHP. (24)

Librerías incluidas: PHP fue diseñada para trabajar sobre la web, por ello trae un conjunto muy amplio de funciones para ser utilizadas en diferentes tareas relacionadas con la web. Se puede conectar con bases de datos, conectar a web *services*, parsear XML, enviar e -mail, generar PDF, generar imágenes, etc. Basadas en estas librerías existen clases implementadas para facilitar el trabajo de los desarrolladores. Otro punto es que hay desarrolladores que agregan librerías especializadas para extender las funcionalidades de PHP. (24)

Javascript

Es un lenguaje interpretado creado por BrendanEich en la empresa Netscape Communications. El código Javascript puede ser integrado dentro de las páginas web. El Consorcio *World Wide Web* (W3C) para evitar incompatibilidades diseñó un estándar para la modelación de objetos del documento, más conocido como DOM (en inglés *Document Object Model*). Entre sus ventajas está su scripting seguro y fiable y contradictoriamente su debilidad es su visibilidad al alcance de cualquier usuario.

AJAX

Acrónimo en inglés de Asynchronous JavaScript And XML (JavaScript y XML asíncronos). Es una técnica de desarrollo Web para crear aplicaciones interactivas mediante la combinación de tres tecnologías ya existentes: HTML (o XHTML) y Hojas de Estilo en Cascada (CSS) para presentar la información. DOM y JavaScript, para interactuar dinámicamente con los datos. XML y Extensible *Stylesheet Language Transformations* (XSLT), para intercambiar y manipular datos de manera no sincronizada con un servidor Web (aunque las aplicaciones AJAX pueden usar otro

tipo de tecnologías, incluyendo texto llano, para realizar esta labor). Todos los navegadores Web usados por las aplicaciones AJAX soportan las tres tecnologías mencionadas anteriormente. Entre estos se incluyen: Mozilla Firefox, Internet Explorer, Safariy Opera. (25)

1.6. Herramientas de Desarrollo Colaborativo

SVN (SubVersion) 1.7.6

SVN (SubVersion) es un sistema de control de versiones, que mantiene los registros de todos los cambios que se han realizado a los archivos de un software, lo que permite el trabajo de distintos desarrolladores en un mismo proyecto, esta herramienta es muy usada por los programadores de software libre. Existen dos razones fundamentales para su uso: (26)

- Gestiona las modificaciones durante el desarrollo.
- Permite que varias personas trabajen sobre los mismos ficheros.

1.7. Conclusiones parciales

Una vez finalizado este capítulo, se arribaron a las siguientes conclusiones:

- Como resultado de los estudios e investigaciones realizados en este capítulo, quedan claramente expuestos los principales conceptos relacionados con Distribución, Pedido, Inventario y su relación con los Sistemas Integrales de Gestión.
- Se realizó un estudio de los sistemas utilizados actualmente en las entidades cubanas para la gestión de los procesos de distribución y pedido de productos permitiendo obtener una valoración crítica de los mismos.
- Se definió como metodología de desarrollo a utilizar para guiar el desarrollo de la investigación el Modelo de Desarrollo del Centro CEIGE. Definiéndose además las tecnologías y herramientas necesarias para el desarrollo de la solución.

Capítulo II Análisis y Diseño del sistema

2.1. Introducción

En este capítulo se realiza un estudio y análisis de la situación problemática a fin de lograr un mejor entendimiento de la misma. Para esto se realiza la identificación de todos los conceptos que intervienen en el proceso de negocio, los cuales quedan relacionados en el modelo conceptual. También se definen los procesos que necesitan ser informatizados y se modelan los procesos de negocio especificando los requisitos de software y describiendo los requisitos funcionales y no funcionales del sistema.

2.2. Modelo Conceptual

Un Modelo Conceptual es un artefacto de la disciplina de análisis, construido con las reglas de UML, donde es ilustrado con un grupo de diagramas de estructura estática donde no se define ninguna operación. La designación de modelo conceptual ofrece la ventaja de subrayar fuertemente una concentración en los conceptos del dominio, no en las entidades del software. El mismo puede mostrarnos conceptos, asociaciones entre conceptos y atributos de conceptos. Modelo Conceptual puede utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema. Es utilizado por el analista como un medio para comprender el sector de negocios al cual el sistema va a servir. (28)

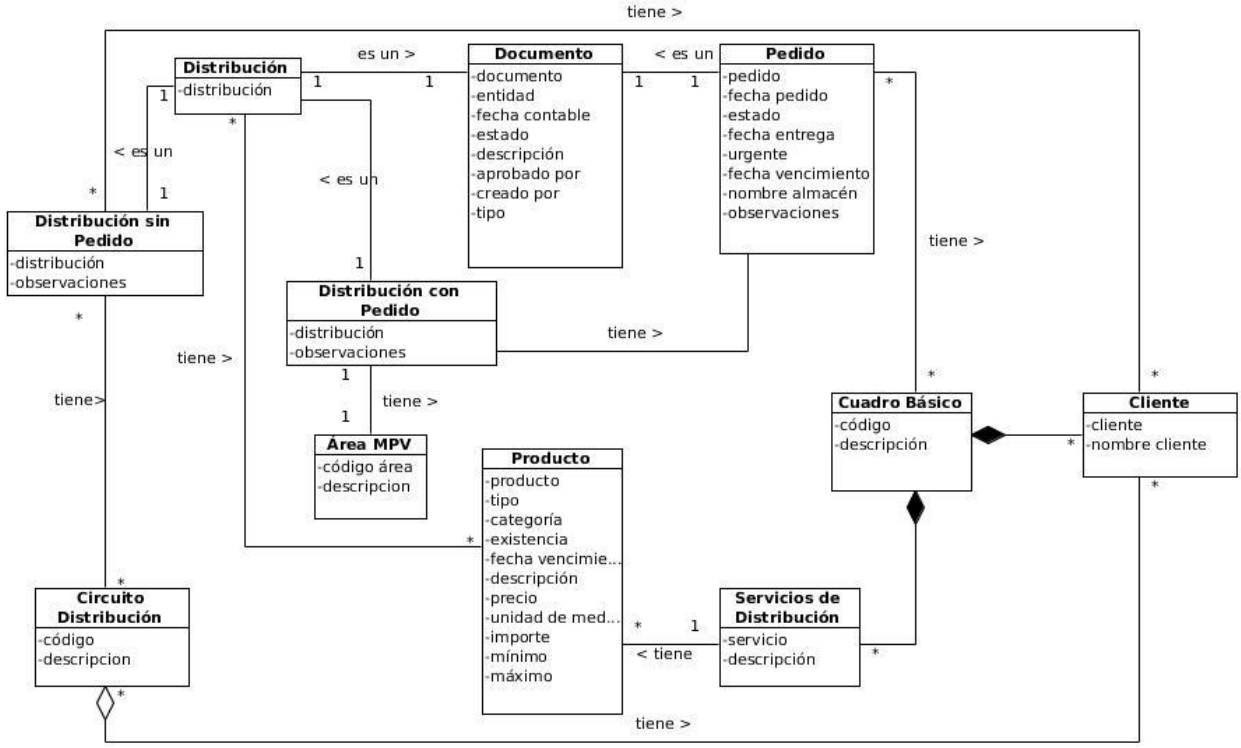

Figura 2. Modelo Conceptual.

2.2.1.Definición de los conceptos fundamentales

Clientes: Entidad o persona que accede a un producto o servicio por medio de una transacción financiera (dinero) u otro medio de pago. Para el caso de la Conciliación material (mediante transferencias entre almacenes) el cliente es el almacén receptor.

Documento: Documento que da origen a algún movimiento de productos dentro del almacén.

Área MPV: Lugar o depósito donde se encuentran los productos y por donde se realizan los movimientos, amparados los mismos por documentos primarios.

Servicio: Servicios que se atienden en la entidad.

Pedido: El pedido es la operación de solicitud de productos de un proveedor determinado.

Producto: Los materiales de módulo son aquellos activos de los cuales puede hacer posesión un cliente permanentemente.

Cuadro Básico: Es el conjunto de relaciones entre clientes y servicios.

Distribución: Distribuciones de productos que se realizan en la entidad.

Circuito de Distribución: Es el conjunto de clientes agrupados para distribuir en la misma acción.

2.3. Modelado del Negocio

Es la disciplina destinada a comprender los procesos de negocio de la organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito. (14) Esta disciplina permite tener una visión del negocio mediante la definición, descripción y representación de los procesos.

2.3.1.Descripción del Proceso del Negocio

Un proceso de negocio es un conjunto de actividades o procedimientos que cumplen un objetivo específico y la descripción de estos procesos permite un mayor entendimiento y disponibilidad para el desarrollo de dicha actividad. (29)

Objetivo	Realizar un pedido
Evento(s) que lo genera(n)	**N/A**
Pre condiciones	**N/A**
Marco legal	**N/A**
Clientes internos	**N/A**
Clientes externos	**N/A**
Entradas	**N/A**
Flujo de eventos	
Flujo básico <Nombre del flujo básico>	
	La entidad cliente se dirige a la entidad proveedora realizar el pedido de los productos que desea adquirir.
	Se define el cliente que va a realizar los pedidos.
	Se especifica si el pedido es de urgencia o no.
	Se define la fecha de vencimiento del pedido.
	Se anotan observaciones en caso de existir.
	Se seleccionan los productos a incluir en el pedido (Se seleccionan los productos de los servicios que se incluyen en el pedido).
	Se especifica la cantidad que se desea adquirir por cada producto.
	Se procede a guardar el pedido, a través de lo cual se obtiene el documento pedido.
	Una vez archivado dicho documento se procede a confirmar los datos del pedido.
	Si estos son correctos se aprueba el pedido.
	Concluye en proceso.
Pos-condiciones	

	Se creó el documento de pedido con los servicios y productos seleccionados.
Salidas	
Documento de Pedido	
Flujos paralelos	
N/A	
Pos-condiciones	
N/A	
Salidas	
NA	
Flujos alternos	
Ir al paso 3	
Pos-condiciones	
El pedido se crea sin fecha de vencimiento.	
Salidas	
N/A	
Asuntos pendientes	

Tabla 1. Descripción del proceso de negocio Realizar Pedido.

Ver en el Anexo #1 las restantes descripciones de procesos de negocio.

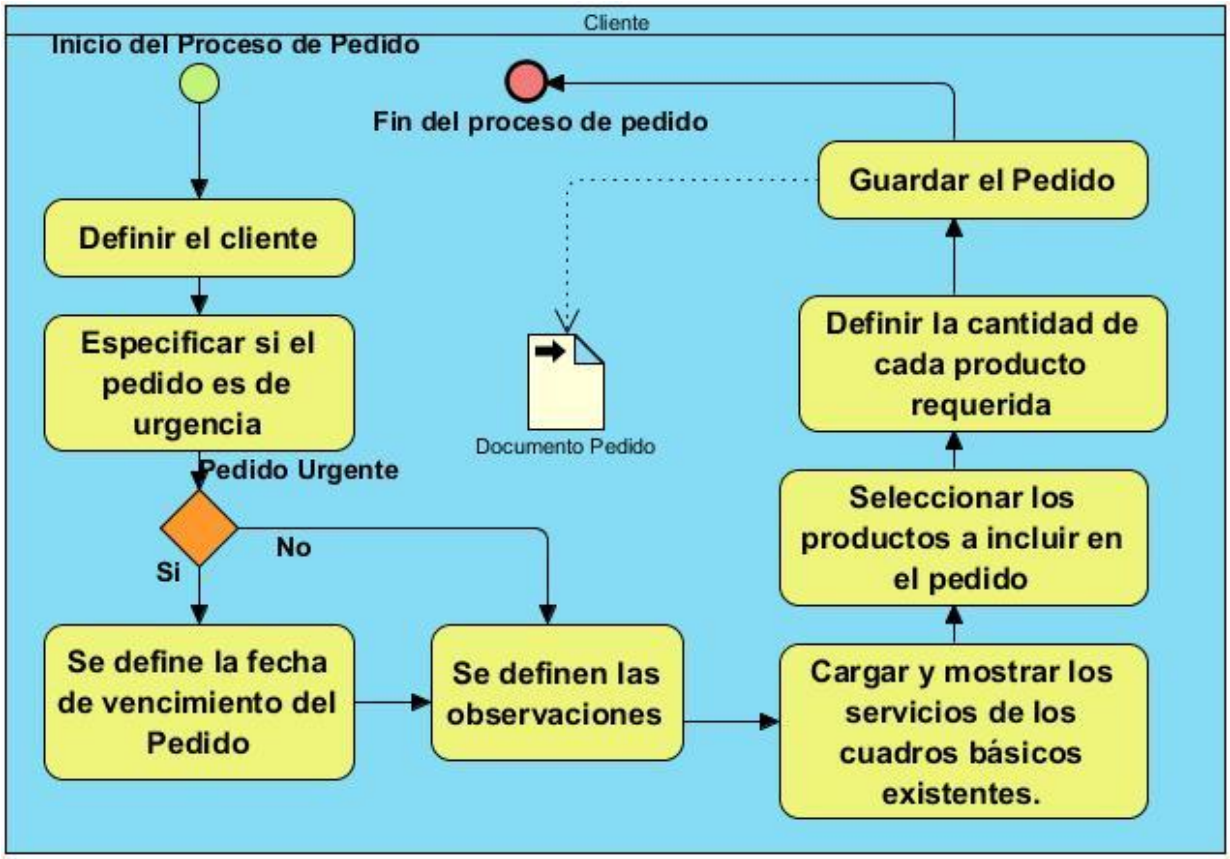

Figura 3. Diagrama del procesos de negocio realizar pedido

Ver en el Anexo #2 las restantes imágenes de diagramas de procesos de negocio.

2.3.2.Mapa de Procesos de Negocio

Un proceso de negocio permite realizar un análisis del problema existente a fin de comprender la estructura y la dinámica de la entidad que se está informatizando, entender los problemas actuales de la organización e identificar las posibles mejoras, asegurar que los usuarios finales y desarrolladores tengan un entendimiento común de la organización y derivar los requisitos del sistema. A continuación se presentan los procesos fundamentales de la investigación y la relación que existe entre ellos a través del mapa de procesos.

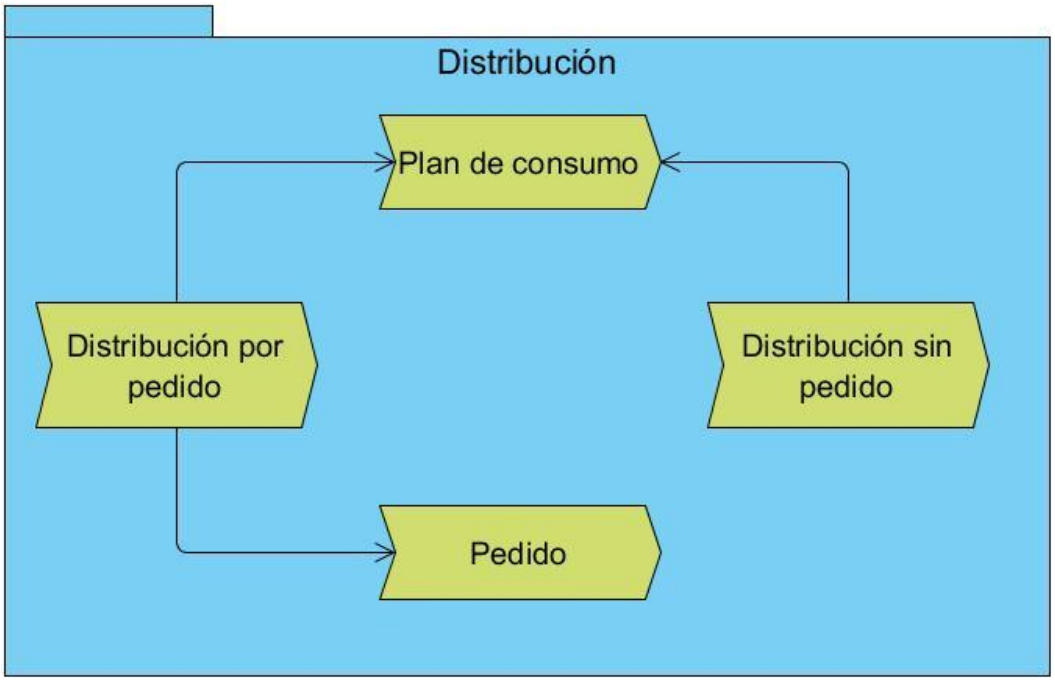

Figura 4. Mapa de procesos del negocio.

2.4. Requisitos

"*Los Requisitos son una especificación de lo que debe ser implementado. Estos son descripciones de cómo el sistema se debe comportar, de las propiedades y atributos del mismo. Deben ser una restricción del proceso de desarrollo del sistema*". (29)

2.4.1.Técnicas para la captura de requisitos

Entrevistas: con esta técnica se obtuvo una descripción global de cada uno de los procesos, permitiendo identificar las actividades específicas que se realizan en cada uno de ellos, además de realizar una modelación de los mismos.

Sistemas existentes: esta técnica fue utilizada durante el análisis de varios sistemas existentes, tal es el caso de SAP, Virtual Work, SOLUFLEX, Stock Mistral y SAP R/3 analizando en todos los casos como se gestionan los procesos de pedido y distribución en estos sistemas, tomando de ellos todas las características que se apegan a las necesidades de las entidades cubanas.

Revisión Documental: Esta técnica se aplicó realizando un análisis y revisión a la documentación que se describe de los procesos de negocio relacionados con los pedidos y la distribución de los productos, para derivar de estos los requisitos que debe tener el sistema.

2.4.2. Requisitos funcionales

RF1-Gestionar pedido

RF1.1 Adicionar pedido.
RF1.2 Modificar pedido.
RF1.3 Eliminar pedido.
RF1.4 Listar pedidos.
RF1.5 Buscar pedido.
RF1.6 Confirmar pedido.
RF1.7 Aprobar pedido.
RF1.8 Cancelar estado del pedido.

RF2-Gestionar distribución.

RF2.1 Adicionar distribución por pedido.
RF2.2 Adicionar distribución sin pedido.
RF2.3 Modificar distribución por pedidos.
RF2.4 Modificar distribución sin pedidos.
RF2.5 Eliminar distribución.
RF2.6 Listar distribuciones.
RF2.7 Buscar distribución.

RF2.8 Confirmar distribución.

RF2.9 Aprobar distribución.

RF2.10 Cancelar estado de la distribución.

RF3-Gestionar pedidos de la distribución.

RF3.1 Adicionar pedidos a la distribución.

RF3.2 Eliminar pedido de la distribución.

RF3.3 Listar pedidos de la distribución.

RF4-Gestionar planes y otros datos de distribución.

RF4.1 Adicionar planes y otros datos de distribución.

RF4.2 Modificar planes y otros datos de distribución.

RF4.3 Eliminar planes y otros datos de distribución.

RF4.4 Listar planes y otros datos de distribución.

RF5-Gestionar productos de la distribución sin pedidos.

RF5.1 Adicionar productos a la distribución sin pedidos.

RF5.2 Eliminar productos de la distribución sin pedidos.

RF5.3 Listar productos de la distribución sin pedidos.

RF6-Gestionar producto del pedido.

RF6.1 Adicionar producto al pedido.

RF6.2 Modificar producto del pedido.

RF6.3 Eliminar producto del pedido.

RF6.4 Listar productos del pedido.

RF7-Gestionar configuración.

RF7.1 Adicionar configuración.

RF7.2 Modificar configuración.

RF7.3 Eliminar configuración.

RF7.4 Listar configuración.

2.4.3. Requisitos no funcionales

Un requisito no funcional es, en la ingeniería de software, un requisito que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos. (30)

- **Software**

Para el cliente:

- Navegador Mozilla Firefox.
- Sistema operativo Linux o Windows.

Para el servidor:

- Sistema operativo Windows o Linux.
- Servidor Apache 2.0 o superior con módulo PHP 5.0 disponible, este debe estar configurado con las extensiones PDO y PDO_pgsql.
- Servidor de base de datos PostgreSQL v4.1.0.1 o superior.

- **Hardware:**

Para el cliente:

- Procesador Pentium IV a 3.00Ghz.
- 512 MB de memoria RAM.
- Tarjeta de red.
- Impresora.

Para el servidor:

- Procesador Pentium IV a 1GHz.
- 1Gb de memoria RAM.
- Tarjeta de red.

- **Rendimiento (REN)/ Disponibilidad**

Un usuario accede a cualquier interfaz de la aplicación y esta debe estar disponible en un período de 0.1 a 0.2 segundos.

Se solicita un recurso al sistema, que puede ser ligero (capacidad por debajo de 2 MB), medio (capacidad de 2 a 10 MB) o complejo (más de 10 MB de capacidad) en un servidor de 1 GB de memoria RAM y se recibe en un período de 0.1 a 0.7, de 0.8 a 2.0 y de 3.0 a 5.0 segundos respectivamente.

Se intercambian datos con el sistema, ya sea mediante acciones de inserción, búsqueda, eliminación o modificación de datos, en un servidor de 1 GB de memoria RAM y se recibe la notificación de la acción realizada en un período de 0.1 a 0.7 segundos.

- **Seguridad (SEG)**

La seguridad en el Sistema Integral de Gestión Cedrux se gestiona a través del sistema Acaxia desarrollado sobre el marco de trabajo Sauxe. Acaxia gestiona la seguridad en un entorno de varias aplicaciones y garantiza temas como:

- Compartimentación de la información.
- Administración de conexiones.
- Entornos multi-temas.
- Entornos multi-lenguaje.
- Entornos multi-entidad.
- Autenticación.
- Autorización.
- Administración de perfiles.
- Auditoría.

Acaxia brinda además tres tipos de integración para reutilizar todas las ventajas que ofrece entre las cuales se encuentra la integración a nivel de servicios web que da la posibilidad de que sistemas de arquitecturas diferentes se comuniquen con Acaxia mediante servicios web. La integración a nivel de servicios internos en la cual es preciso que las aplicaciones sean

desarrolladas sobre el mismo marco de trabajo, logrando así consumir sus servicios y la integración a nivel de interfaz que permite mostrar diferentes aplicaciones desarrolladas en distintas plataformas web en el portal principal.

2.4.4. Descripción de Requisitos

Descripción Textual del Requisito Adicionar pedido

Precondiciones	El usuario ha sido validado.
Flujo de eventos	
Flujo básico Adicionar pedido	
	Se cargan los siguientes datos del documento: Nombre de la entidad Código de la entidad Número del documento Fecha de emisión del documento Estado del documento
	Se selecciona la entidad cliente (se busca la entidad cliente por el nombre).
	El sistema muestra los siguientes datos de la entidad cliente seleccionada: Nombre de entidad cliente
	Si el pedido es urgente, se marca el atributo: Es urgencia
	Si el pedido es de urgencia se insertan los siguientes datos: Fecha de vencimiento
	Se introducen los siguientes datos del documento: Observaciones
	Se adicionan uno o más servicios al pedido.
	El sistema muestra los productos asociados a esos servicios escogidos.
	Se selecciona de los productos mostrados los que se van a adicionar en el pedido y la cantidad de cada uno.
	El sistema valida (ver validación 1, 2, 3, 4) los datos introducidos.
	Si los datos son correctos el sistema los registra.
	El sistema confirma el registro de los datos.
	Concluye el requisito.
Pos-condiciones	
	Se registró en el sistema un nuevo pedido.
Flujos alternativos	
Flujo alternativo 4.a El pedido no es urgente	
	Ir al paso 6 del flujo básico.
Pos-condiciones	
	N/A
Flujo alternativo 10.a Información errónea	
	El sistema señala los datos erróneos y permite corregirlos.
	El usuario corrige los datos.
	Volver al paso 10 del flujo básico.
Pos-condiciones	
	N/A

Flujo alternativo 10.b Información incompleta		
El sistema señala los datos vacíos y permite corregirlos.		
El usuario corrige los datos.		
Volver al paso 10 del flujo básico.		
Pos-condiciones		
N/A		
Flujo alternativo *.a El usuario cancela la acción		
Concluye el requisito.		
Pos-condiciones		
No se registran los datos.		
Validaciones		
Se validan los datos según lo establecido en el Modelo conceptual <<Referencia al modelo conceptual en cuestión>>.		
Se valida que si ya existe un pedido para el mismo cliente donde el estado en la entidad suministradora tiene valor de: “No procesado” (no ejecutado en distribución), el sistema no permite adicionar un nuevo pedido mostrando el mensaje de aviso: “Ya existen un pedido (número del pedido) no anulado ni servido para el cliente (nombre del cliente). Por favor añade los productos al pedido existente o anúlelos.”. En caso de que este configurado “dar seguimiento al pedido por reemplazamiento del existente”, el sistema Cancela el documento existente (guardando la observación: “Cancelado por reemplazamiento”) para dar paso a la creación del nuevo pedido. Validar que el pedido existente a cancelar no tenga productos que estén en proceso en Distribución.		
Se valida que si el pedido es una urgencia y si ya existen un pedido para el mismo cliente, el sistema obvia la validación 3.		
Se valida que se haya seleccionado al menos un servicio, en caso de haber configurado que si se va a mostrar los cuadros básicos en pedido.		
Conceptos	Pedido	Visibles en la interfaz: nombre de la entidad nombre del almacén número del documento fecha de emisión fecha de vigencia estado del documento observaciones Utilizados internamente: N/A
	Entidad cliente	Visibles en la interfaz: nombre de entidad cliente código de entidad cliente Utilizados internamente: dirección entidad cliente tipo de cliente cuenta bancaria de entidad cliente cuenta de la sucursal de entidad cliente NIT entidad cliente Utilizados internamente: N/A

	Servicio	Visibles en la interfaz: Utilizados internamente: N/A
	Documento	Visibles en la interfaz: nombre de la entidad nombre del almacén número del documento fecha de emisión del documento estado del documento observaciones Utilizados internamente: N/A
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 2. Descripción del requisito adicionar pedido.

Prototipo elemental de la interfaz gráfica de usuario:

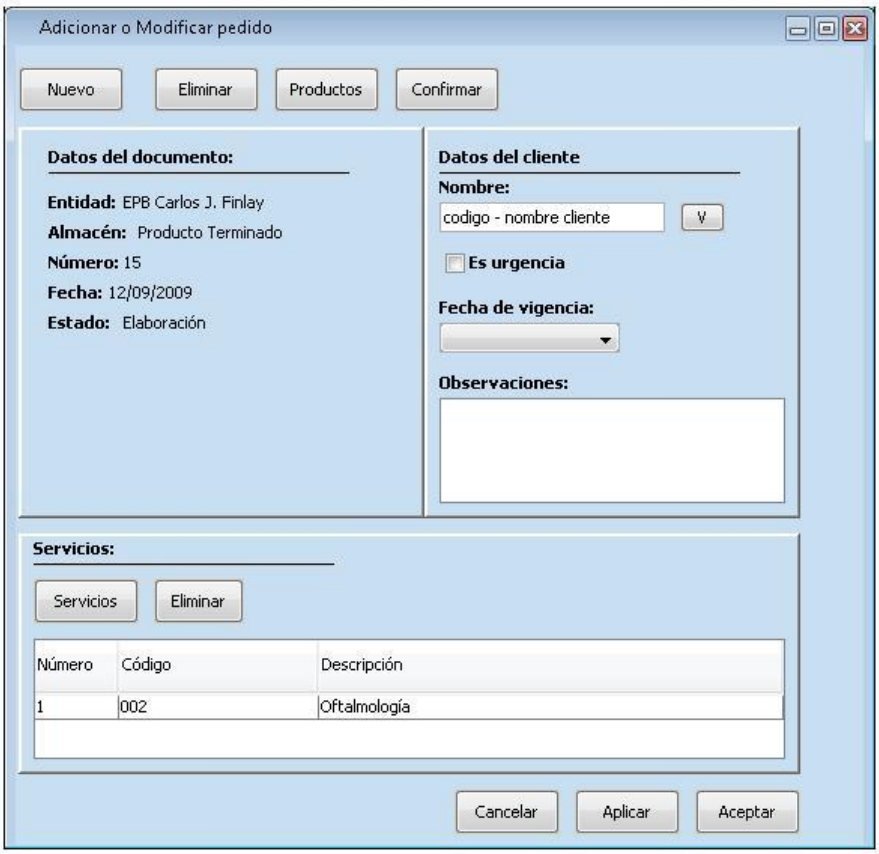

Figura 5. Prototipo de interfaz de usuario adicionar pedido.

Ver en el Anexo #3 las restantes descripciones de requisitos.

2.4.5.Técnicas de Validación de Requisitos

La validación de requisitos tiene como objetivo fundamental demostrar que estos satisfacen las necesidades de los clientes o usuarios finales de las aplicaciones, lo cual es muy importante debido a que los errores en el documento de requisitos, puede conducir a importantes costes al repetir el trabajo cuando son descubiertos durante el desarrollo o después que es sistema está en uso. El costo de arreglar un problema en los requisitos haciendo un cambio en el sistema es mucho más costoso que reparar los errores de diseño o de codificación. A razón de esto un cambio en los requisitos normalmente significa que el diseño y la implementación del sistema también deben cambiar y que este debe probarse nuevamente. (29)

Para la validación de requisitos pueden utilizarse, en conjunto o de forma individual, varias técnicas de validación de requisitos. Durante el desarrollo de los módulos de pedido y distribución de productos se utilizaron las siguientes técnicas de Validación de Requisitos.

Revisión de requisitos: Los requisitos son analizados sistemáticamente por un equipo de revisores.

Prototipado: El método del prototipado consiste en construir una maqueta del futuro sistema software a partir de los requisitos recogidos en la especificación. Esta maqueta será evaluada por el cliente y usuarios para comprobar su corrección y completitud.

Generación de Casos de Prueba: Los requisitos deben poder probarse. Si las pruebas para estos se conciben como parte del proceso de validación, a menudo revela los problemas en los requisitos. Si una prueba es difícil o imposible de diseñar, normalmente significa que los requisitos serán difíciles de implementar y deberían ser considerados nuevamente. Desarrollar pruebas para los requisitos del usuario antes de que se escriba el código es una parte fundamental de la programación.

2.5. Modelo de Datos

Un modelo de datos permite describir las estructuras de datos de la base (el tipo de los datos que incluye la base y la forma en que se relacionan), las restricciones de integridad (las condiciones que los datos deben cumplir para reflejar correctamente la realidad deseada) y las operaciones de

manipulación de los datos (agregado, borrado, modificación y recuperación de los datos de la base). (31)

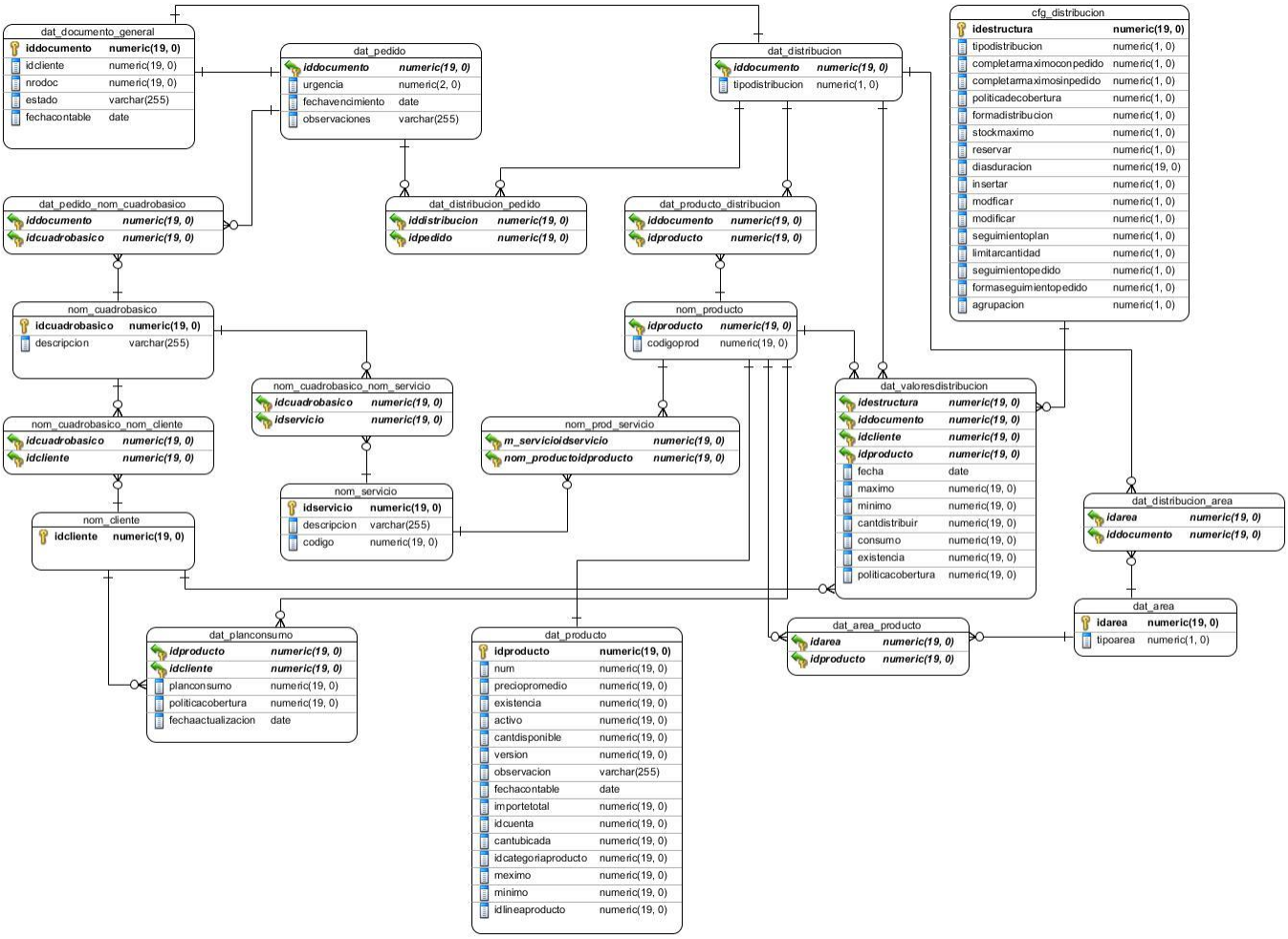

Figura 6. Modelo de Datos.

2.5.1.Diccionario de Datos

dat_documento: Tabla que almacena los datos de los documentos que serán tratados en la entidad.

dat_pedido: Tabla que almacena los datos de los pedidos realizados por las entidades.

dat_pedido_nom_cuadrobasico: En esta tabla se crea una relación con nom_cuadrobasico a través del iddocumento.

nom_cuadrobasico: En esta tabla se almacenan las relaciones entre los servicios y los productos.

nom_cuadrobasico_nom_cliente: En esta tabla se crea una relación con nom_cliente a través del idcuadrobasico.

nom_cliente: Se almacenan los clientes de las entidades.

nom_cuadrobasico_nom_servicio: En esta tabla se crea una relación con nom_servicio a través del idcuadrobasico.

nom_servicio: Se almacenan los diferentes servicios que pueden ser brindados.

nom_producto: Se almacenan todos los productos existentes.

nom_prod_servicio: En esta tabla se crea una relación con nom_servicio.

2.6. Diagrama de Componentes

Un diagrama de componentes permite visualizar con más facilidad la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de las interfaces.

Un diagrama de componentes representa a un componente real: un componente de software. Dicho componente puede ser accedido a través de su interfaz, esta relación es conocida como *realización.* También puede suceder que un componente acceda a los permisos de otro, cuando esto sucede se utiliza una *interfaz de importación*. (32)

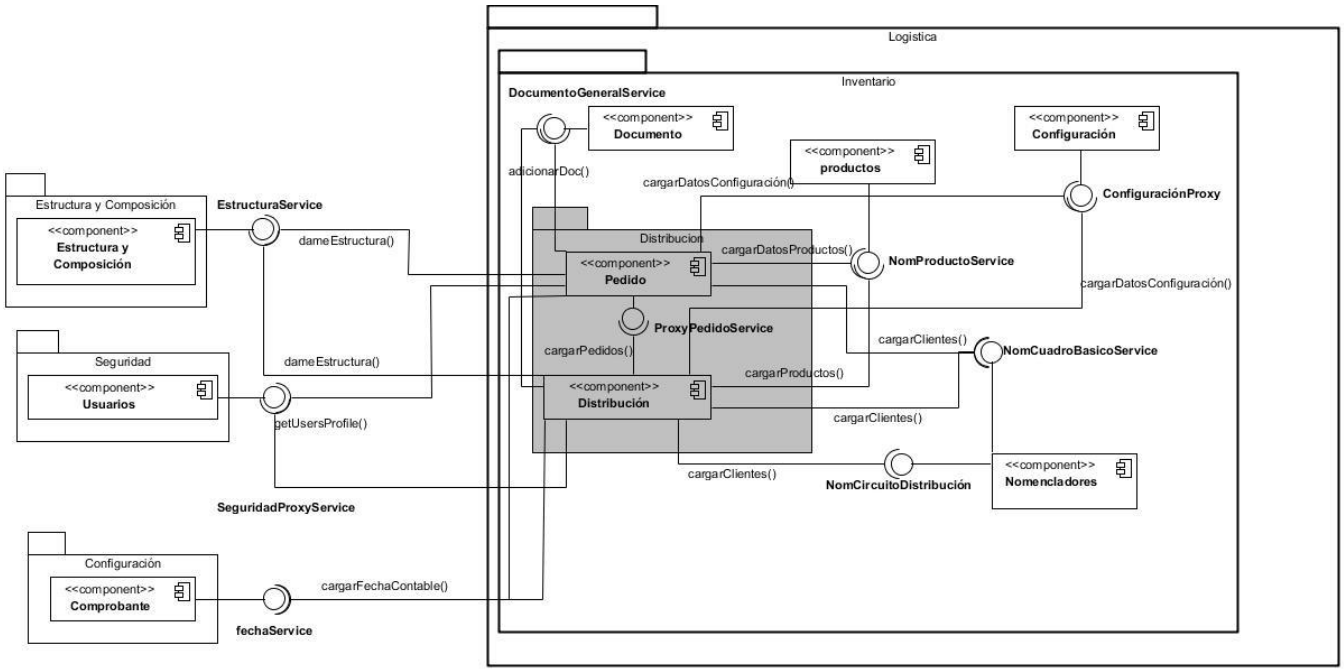

Figura 7. Diagrama de componente de los módulos Distribución y Pedido de Productos.

2.7. Modelado de Diseño

2.7.1. Patrones arquitectónicos utilizados para el diseño de la solución

Modelo Vista Controlador

EL patrón arquitectónico utilizado fue MVC dado que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres capas diferentes. La aplicación de este patrón se evidencia en los diagramas de clases del diseño, donde en las clases del modelo (ClaseModel) representadas, se representa la lógica de negocio. La vista (clases .js y las páginas .phtml) transforma el modelo en una página Web que permite al usuario interactuar con ella. El controlador (ClaseController) se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista. (28)

El patrón *Model-View-Controller* (MVC) divide una aplicación interactiva en tres componentes. El modelo contiene la funcionalidad central y los datos. Las vistas despliegan la información al usuario. Los controladores se ocupan de las entradas del usuario. Las vistas y controladores juntos forman la interfaz de usuario.

Arquitectura basada en componentes

Sauxe, marco de trabajo sobre el cual fue desarrollado el sistema Cedrux, se rige por una arquitectura basada en componentes de forma horizontal, que tiene como objetivo hacer un uso correcto del software reutilizable.

Un componente es un fragmento reemplazable de un sistema de software, una unidad de composición con interfaces especificadas contractualmente, que satisface una o varias funcionalidades dentro del contexto de una arquitectura bien definida y puede ser ensamblado con otros fragmentos por medio de una interfaz. El sistema Cedrux está formado por varios subsistemas los cuales cuentan con una serie de componentes que trabajan de forma independiente y pueden ser reutilizados por otros subsistemas. Cada componente implementa el patrón arquitectónico y de diseño Modelo-Vista-Controlador (MVC).

2.7.2.Patrones de diseño utilizados

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

de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. (28)

Patrones de principios generales para asignar responsabilidades (GRASP):

Los patrones GRASP [6] describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones. Los patrones no se proponen descubrir ni expresar nuevos principios de la ingeniería del software. Todo lo contrario, intentan codificar el conocimiento, las expresiones y los principios ya existentes que cuanto más trillados y generalizados, mucho mejor. (28)

A continuación se detallan los patrones GRASP que serán utilizados en el diseño de la solución:

Experto en Información: Este patrón consiste en asignar una responsabilidad al experto en información que no es más que la clase que cuenta con la información necesaria para cumplir dicha responsabilidad. Responde al problema ¿Cuál es el principio fundamental en virtud del cual se asignan las responsabilidades en el diseño orientado a objetos?

Experto es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él se pretende expresar simplemente la "intuición" de que los objetos hacen cosas relacionadas con la información que poseen. (28)

Creador: Consiste en crear una nueva instancia por la clase que: Tiene la información necesaria para realizar la creación del objeto, usa directamente las instancias creadas del objeto, almacena o maneja varias instancias de la clase o contiene o agrega la clase. Responde al problema ¿Quién debería ser responsable de crear una nueva instancia?

El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. (28)

6 GRASP es un acrónimo que significa General Responsibility Assignment Software Patterns

Alta Cohesión: Asignar responsabilidades de modo que una clase no realice trabajos que debieron ser delegados a otros objetos y evitar el alto grado de abstracción. Responde al problema ¿Cómo mantener la complejidad dentro de límites manejables?

Grady Booch señala que se da una alta cohesión funcional cuando los elementos de un componente (clase, por ejemplo) "colaboran para producir algún comportamiento bien definido". (28)

Bajo Acoplamiento: Diseñar con el objetivo de tener las clases lo menos ligadas entre sí que se pueda. De forma tal que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. Responde al problema ¿Cómo dar soporte a las bajas dependencias y al incremento de la reutilización?

El Bajo Acoplamiento es un principio que debemos recordar durante las decisiones de diseño: es la meta principal que es preciso tener presente siempre. Es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño. (28)

Controlador: Consiste en garantizar que los procesos del dominio sean manejados por los la capa de los objetos del dominio y no por la interfaz. Responde a la pregunta ¿Quién debería encargarse de atender un evento en el sistema?

Este patrón se pone de manifiesto en la clase controladora GestpdidosController, la cual tiene la responsabilidad de manejar los eventos dentro del componente.

Indirección: Asigna la responsabilidad a un objeto intermedio para que medie entre otros componentes y servicios, y estos no terminen directamente acoplados. El intermediario crea una indirección entre el resto de los componentes o servicios. Este patrón se pone de manifiesto entre las clases controladoras, las clases modelos y las clases de acceso a datos.

Patrones de la Pandilla de los Cuatro (Gang of Four)

Los patrones Gof fueron definidos en *Design Patterns* obra pionera en la que se presentan un total de 23 patrones de gran utilidad para la fase de diseño. Estos patrones son conocidos actualmente como Pandilla de los Cuatro (*Gabg of Four* por sus siglas en inglés), debido a que el libro fue escrito por cuatro autores.

Fachada: Consiste en proporcionar una interfaz unificada de alto nivel que, representando a todo un subsistema, facilite su uso. La “fachada” satisface a la mayoría de los clientes, sin ocultar las funciones de menor nivel a aquellos que necesiten acceder a ellas. Al separar al cliente de los componentes del subsistema, se reduce el número de objetos con los que el cliente trata, facilitando así el uso del subsistema. Se promueve un acoplamiento débil entre el subsistema y sus clientes, eliminándose o reduciéndose las dependencias. Su aplicación se puede observar en el uso de los servicios y su relación con las clases controladoras, la cual le permite acceder a métodos que están implementados en otros componentes dentro y fuera del subsistema Inventario.

Cadena de Responsabilidad: Es un patrón de comportamiento que permite establecer la línea que deben llevar los mensajes para que los objetos realicen la tarea indicada. Este patrón es utilizado en el tratamiento de excepciones. Este se pone de manifiesto cuando ante la ocurrencia de un error al realizarse una determinada consulta a la base de datos, el mismo sea manejado por el Modelo, creando una nueva excepción de tipo ZendExt_Exception. Dicha excepción debe ser propagada al Controlador, el cual será el encargado de capturarla y enviarla a la Vista ya traducida, siendo esta última la que mostrará un mensaje al usuario en un lenguaje entendible notificando el error y sin especificar detalles del mismo. De esta manera se distribuyen las responsabilidades entre los diferentes componentes.

Singleton (Instancia Única): Es un patrón creacional que garantiza que una clase solo tenga una instancia, y proporciona un punto de acceso global a ella. Este se pone de manifiesto tanto en las clases controladoras como en las modelo.

2.7.3. Diagramas de clases del diseño

Un diagrama de clases de diseño (DCD) representa las especificaciones de las clases e interfaces software en una aplicación. Entre la información general encontramos:

- Clases, asociaciones y atributos.
- Interfaces, con sus operaciones y constantes.
- Métodos.
- Información acerca del tipo de los atributos.
- Navegabilidad.
- Dependencias.

Las clases de diseño de los DCD muestran las definiciones de las clases software en lugar de los conceptos del mundo real, además de las asociaciones y atributos básicos, el diagrama se amplía para representar, por ejemplo, los métodos de cada clase, información del tipo de los atributos y navegación entre los objetos.

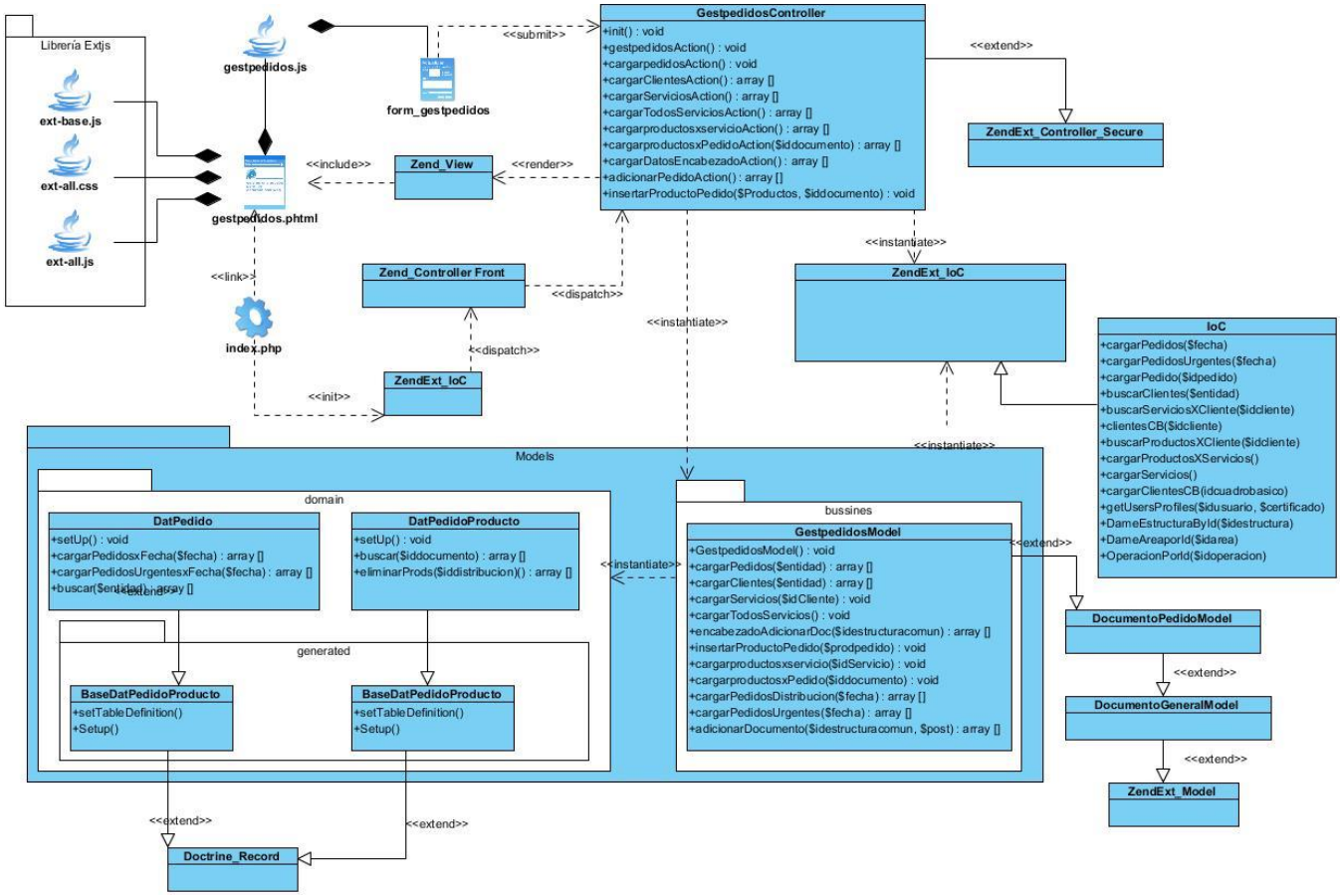

Figura 8. Diagrama de clases del diseño de gestionar pedidos.

Ver en el Anexo #4 las restantes imágenes de los diagramas de clases del diseño.

2.7.4. Diagramas de Secuencias

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

El diagrama de secuencias consta de objetos que se representan del modo usual: rectángulos con nombre (subrayado), mensajes representados por líneas continuas con una punta de flecha y el tiempo representado como una progresión vertical. (32)

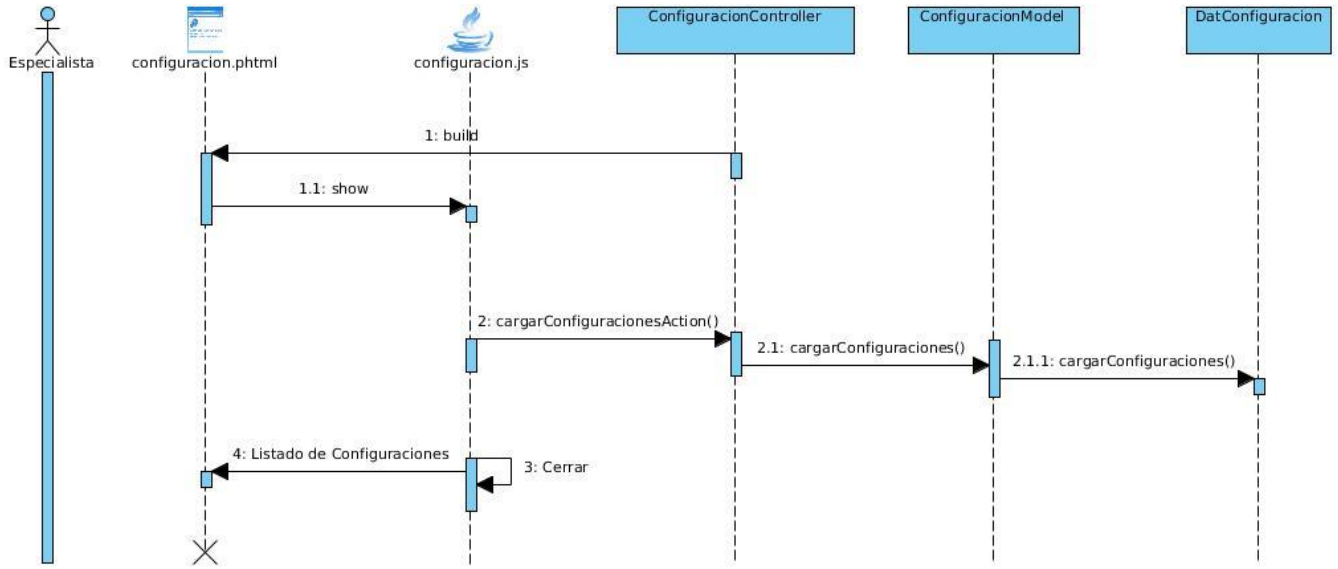

Figura 9. Diagrama de secuencia del requisito listar configuraciones.

Ver en el Anexo #5 las restantes imágenes de los diagramas de secuencia.

2.8. Validación del diseño

El diseño está enfocado a convertir los requisitos del cliente en un modelo que al ser implementado, se obtenga el producto deseado. Generalmente constituye el punto de partida para el desarrollo de software una vez especificados los requisitos del sistema. Realizar una validación del mismo para verificar su calidad y flexibilidad, garantiza una buena base para la implementación. Con este objetivo se utilizan un conjunto de métricas de software orientadas a determinar, entre otros aspectos, qué características del modelo de diseño se pueden estimar para comprobar que el sistema será fácil de implementar en cuanto a organización. Lorenz y Kidd dividen las métricas basadas en clases en cuatro categorías: tamaño, herencia, valores internos y valores externos. Las métricas orientadas a tamaños para una clase se centran en cálculos de atributos y de operaciones, para luego promediar los valores para el sistema en su totalidad. Las métricas basadas en herencia se centran en la forma en que se reutilizan las operaciones en la jerarquía de clases. Las métricas para valores internos de clase examinan la cohesión y asuntos relacionados con el código así como las métricas orientadas a valores externos examinan el acoplamiento y la reutilización (33)

Entre ellas se seleccionaron las métricas Tamaño Operacional de Clase (TOC) y Relación entre Clases (RC) para validar los módulos distribución y pedido de productos.

Tamaño Operacional de Clase (TOC):

Esta métrica se determina por el número total de operaciones que están encapsuladas dentro de la clase. Grandes valores de esta medida muestran que la clase puede tener demasiada responsabilidad, lo cual reducirá la reusabilidad de la misma y complicará su implementación. Por otro lado, en cuanto menor sea el valor medio para el tamaño más probable es que las clases tengan menos responsabilidad y complejidad y más nivel de reutilización.

Fueron utilizados para la evaluación de las clases umbrales para el tamaño general *(Ver tabla 3)* y la responsabilidad, la complejidad y la reutilización de las clases como atributos de calidad *(Ver tabla 4).*

Atributo	Categoría	Criterio
Responsabilidad	Baja	<= PO
	Media	Entre PO y 2 * PO
	Alta	> 2 * PO
Complejidad implementación	Baja	<= PO
	Media	Entre PO y 2 * PO
	Alta	> 2 * PO
Reutilización	Baja	> 2 * PO
	Media	Entre PO y 2 * PO
	Alta	<= PO

Tabla 3. Atributos de calidad que afecta esta prueba.

Nombre	Descripción
Responsabilidad	Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
Complejidad de implementación	Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
Reutilización	Un aumento del TOC implica una disminución del grado de reutilización de la clase.

Tabla 4. Modo en que son afectados los atributos de calidad.

Representación en % de la incidencia de los resultados obtenidos en cada atributo de calidad

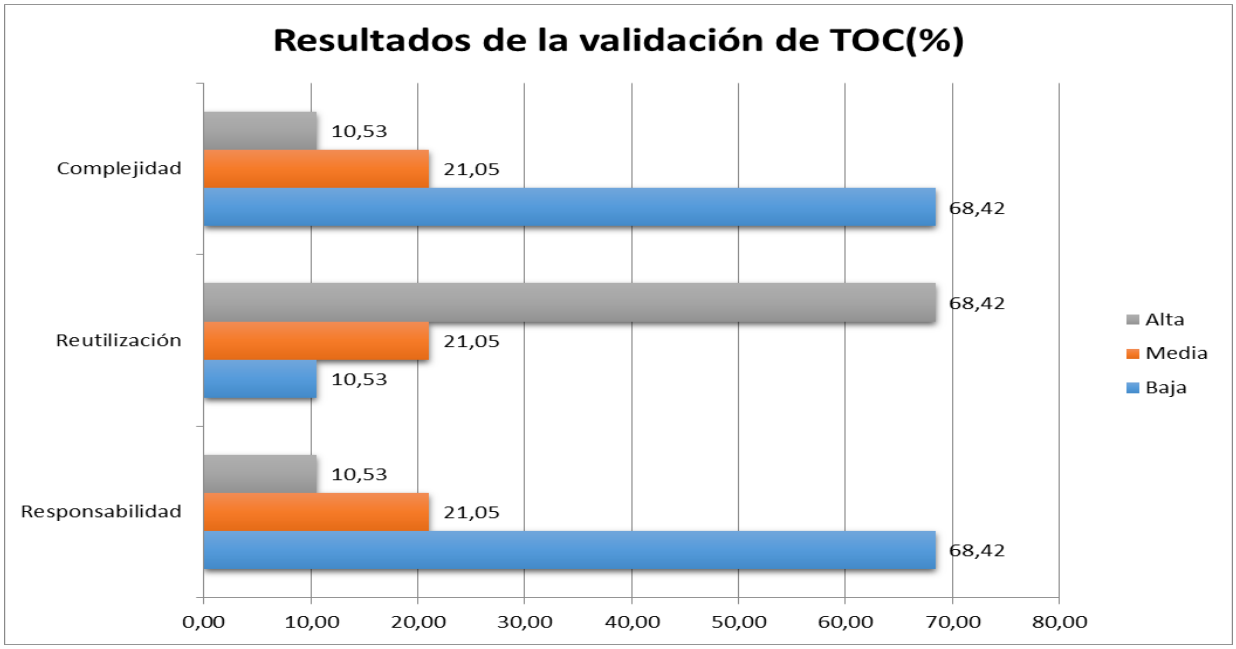

Figura 10. Resultados de la validación de "Tamaño Operacional de Clase"

Análisis de los resultados obtenidos en la evaluación de la métrica TOC

Durante la evaluación de la métrica TOC los resultados obtenidos demuestran que los atributos de calidad de las clases se encuentran en un nivel satisfactorio, evidenciándose un porciento elevado de posibilidad de reutilización de las clases, proporcionado por un 68.42% de baja responsabilidad de clases y de complejidad de implementación.

Relación entre Clases (RC):

Esta métrica está dada por el número de relaciones de uso de una clase con otra y evalúa los siguientes atributos de calidad. *(Ver tabla 5)*

Atributo de calidad	Modo en que lo afecta
Acoplamiento	Un aumento del RC implica un aumento del acoplamiento de la clase.
Complejidad de mantenimiento	Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
Reutilización	Un aumento del RC implica una disminución en el

	grado de reutilización de la clase.
Cantidad de pruebas	Un aumento del RC implica un aumento de la cantidad de pruebas de unidad necesarias para probar una clase.

Tabla 5. Atributos de calidad y el modo en que son afectados.

Para la evaluación de las clases fueron utilizados umbrales para el acoplamiento, complejidad de mantenimiento, la reutilización y cantidad de pruebas. *(Ver Tabla 6)*

Atributo	Categoría	Criterio
Responsabilidad	Ninguna	0
	Baja	1
	Media	2
	Alta	> 2
Complejidad implementación	Baja	<= PO
	Media	Entre PO y 2 * PO
	Alta	> 2 * PO
Reutilización	Baja	> 2 * PO
	Media	Entre PO y 2 * PO
	Alta	<= PO
Reutilización	Baja	<= PO
	Media	Entre PO y 2 * PO
	Alta	> 2 * PO

Tabla 6. Atributos de calidad y el modo en que son afectados.

Representación en % de la incidencia de los resultados obtenidos en cada atributo de calidad

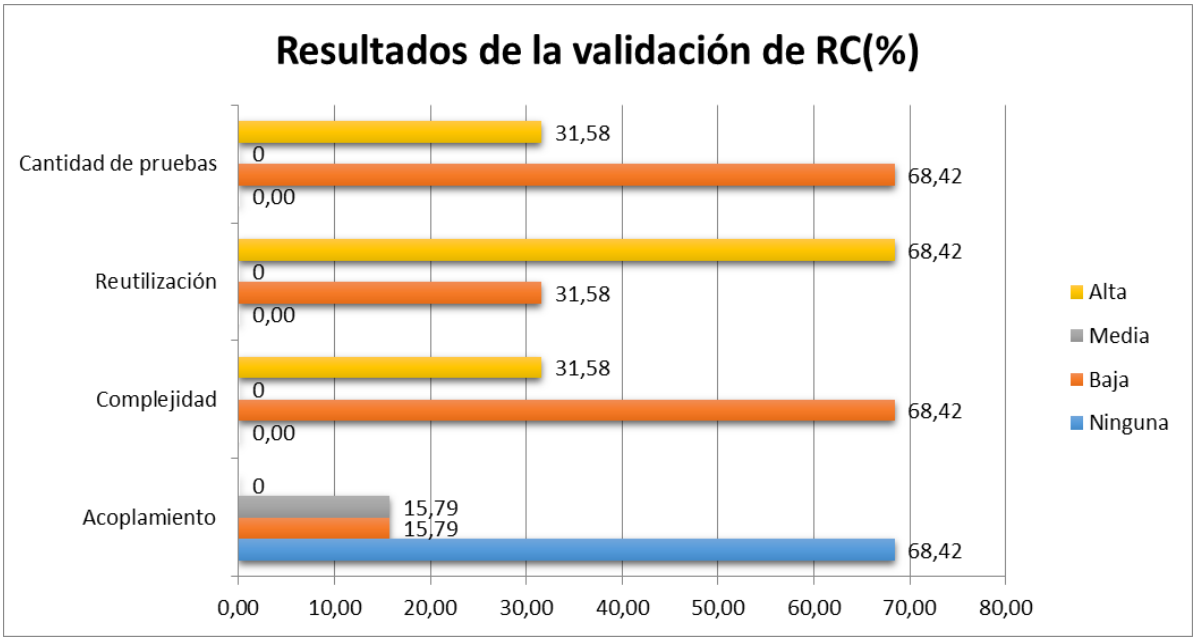

Figura 11. Resultados de la validación "Relaciones por Clases"

Análisis de los resultados obtenidos en la evaluación de la métrica RC

Luego de aplicarse la métrica de diseño RC y obtenidos los resultados, se puede concluir que el diseño propuesto tiene una calidad aceptable teniendo en cuenta que el 100 % de las clases empleadas poseen menos de 3 dependencias de otras clases, lo que lleva a evaluaciones positivas de los atributos de calidad involucrados como son el bajo acoplamiento, complejidad de mantenimiento, cantidad de pruebas y alta reutilización todos con un 68.42 % de positividad.

Matriz de inferencia de indicadores de calidad

La matriz de inferencia permite evaluar positiva o negativamente los resultados obtenidos de las relaciones atributo/métrica en una escala numérica, donde el valor 1 representa un resultado “positivo” y el valor 0 “negativo”. Si la métrica no influye en el atributo de calidad la relación es considerada como nula y es representada con un guion simple (-). Al promediar por cada atributo las relaciones no nulas, se obtiene un resultado general que al ubicarse en el rango de evaluación (valor <=0.4: “Mala”, valor >0.4 y valor<0.7: “Regular”, valor>=0.7: “Buena”) permite arribar a conclusiones sobre la calidad del diseño propuesto. Teniendo en cuenta que los resultados arrojados con la aplicación de las métricas fueron positivos para todos los atributos, la matriz de la inferencia queda elaborada de la siguiente manera. *(Ver tabla 7)*

Atributos\Métricas	TOC	RC	Promedio
Responsabilidad	1	-	1
Complejidad de implementación	1	-	1
Reutilización	1	1	1
Acoplamiento	-	1	1
Complejidad de mantenimiento	-	1	1
Cantidad de prueba	-	1	1

Tabla 7. Matriz de Inferencia de indicadores de calidad.

El promedio general es 1 y teniendo en cuenta el rango de evaluación se concluye que los módulos distribución y pedido de productos presenta un diseño aceptable.

2.9. Conclusiones parciales

Con el desarrollo del presente capítulo se alcanzaron los siguientes resultados:

- Se realizó la descripción de los procesos de negocio y se elaboró el correspondiente modelo conceptual en el cual se relacionan los conceptos fundamentales de la investigación.
- Se elaboró el diagrama de componentes evidenciando de forma general la estructura del sistema y el comportamiento de los servicios que estos proporcionan.
- Se identificaron y describieron los requisitos funcionales del módulo, así como los requisitos no funcionales especificando los criterios necesarios para operar el sistema.
- Se elaboraron los artefactos correspondientes a la disciplina de Análisis y Diseño como son el Modelo de Datos, diagrama de Clases del Análisis, diagramas de Clases del Diseño y diagramas de Secuencia, así como la enunciación de los patrones a utilizar.
- Se realizó la validación del diseño aplicando las métricas Tamaño Operacional de Clases y Relaciones entre Clases los cuales permiten comprobar el nivel de calidad del sistema.

Capítulo III Implementación y Pruebas

3.1 Introducción

En el presente capítulo se abordaran los aspectos más importantes relacionados con la implementación del sistema y la validación del mismo. También se valida a través de las pruebas de caja negra si el software realizado se corresponde con los requisitos funcionales planteados en la disciplina de requisitos.

3.2 Implementación

A partir de los resultados del análisis y diseño se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ejecutables y similares. Para llevar a cabo dicha implementación se utilizó como IDE de Desarrollo el NetBeans y los lenguajes de programación PHP y Javascript, siguiendo además estándares de codificación para la implementación de la especificación de requisitos descritas. Al reutilizar componentes software ya implementados se lleva a cabo el desarrollo necesario para ajustar a los requisitos actuales y posteriormente realizar la integración de los componentes.

3.3 Estándares de Codificación

Un estándar de codificación comprende todos los aspectos de la generación de código, dicho código debe reflejar un estilo armonioso, como si un único programador fuese el responsable de escribir el código de todo el proyecto, logrando así que cuando realice el mantenimiento de un sistema de software creado anteriormente, el estándar de codificación defina cómo operar con la base de código existente. La legibilidad del código fuente repercute directamente en lo bien que un programador comprende un sistema de software. La mantenibilidad del código es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento. Para el desarrollo de los módulos de pedido y distribución de productos, se utilizarán algunos de los estándares y normas de codificación propuestos como parte de la Línea de Arquitectura determinada para el proyecto ERP Cuba, los cuales se muestran a continuación:

- **Notación PascalCasing:** En este caso los identificadores y nombres de las variables, métodos y funciones están compuestos por múltiples palabras juntas, iniciando cada palabra con letra mayúscula.

- **Notación CamelCasing:** es parecido al PascalCasing con la excepción que la letra inicial del identificador no debe estar en mayúscula.

Identación en el código (Tabs o espacios): Son aquellos tabs o espacios que se ponen delante de las líneas de código, los cuales se han utilizado en este trabajo para lograr una estructura más organizada y legible que permita un mejor entendimiento del código a otros programadores que necesiten consultarlo en el futuro.

```
foreach ($result as $k => $re) {
    if ($datos['iddoc'] == $iddocs['combustible']->iddoc) {
        $res[$k]['idcliente'] = $re['idcliente'];
    }
    if ($re['DatInventario']['tipoinventario']){
        switch ($re['DatInventario']['tipoinventario']) {
            case '1':$res[$k]['dtipoinventario'] = 'Apertura';
                break;
            case '2':$res[$k]['dtipoinventario'] = 'Parcial Mensual';
                break;
            case '3':$res[$k]['dtipoinventario'] = 'Parcial';
                break;
            case '4':$res[$k]['dtipoinventario'] = 'General';
                break;
        }
```

Figura 12. Estándar de Codificación Identación en el código.

Cabecera de Archivo: Se han insertado cabecera a los archivos de código con el fin de dejar claro que tipo de operaciones se realizan en la clase, quien la implementó, la versión, el nombre del archivo y a que paquete pertenece.

```
/**
 * GestpedidosModel
 * Modelo de Gestión de Inventario Físico
 *
 * @package Inventario
 * @subpackage Inventario
 * @author Marieni y Jorge
 * @copyright ERP Cuba
 * @version 1.1
 */
```

Figura 13. Estándar de Codificación Cabecera de Archivo.

Nomenclatura de Clases: Se utiliza la notación PascalCasing, la cual define que se escribe la primera letra del nombre de la clase con mayúscula y el resto con minúscula. Si el nombre es compuesto las demás palabras se escriben juntas y comenzando con mayúscula.

```
class GestcircuitodistribucionController extends ZendExt_Controller_Secure {
```

Figura 14. Estándar de Codificación Nomenclatura de Clases.

Nomenclatura según el tipo de Clases: La nomenclatura según el tipo de clases consiste en agregar al final de cada nombre de clase una palabra que identifique el tipo de clase. Por ejemplo a las clases del modelo se le agrega *"Model"* y a las controladoras se les agrega el *"Controller"*.

```
class GestpedidosController extends ZendExt_Controller_Secure {
```

Figura 15. Estándar de Codificación Nomenclatura según el tipo de Clases.

Nomenclatura de Funciones: El nombre de las funciones se escribe en minúscula y si dicho nombre es compuesto las palabras se escriben juntas y con la primera letra en mayúscula exceptuando la primera palabra.

```
public  function insertarProductoPedido $Productos, $iddocumento){
    //$iddocumento=$this->_request->getPost('iddocumento');
    //print_r($iddocumento);die("jjn");
```

Figura 16. Estándar de Codificación Nomenclatura de Funciones.

Nomenclatura de Variables: La nomenclatura utilizada en las variables (Figura 13) es similar a la nomenclatura usada en las funciones, solo se diferencia en que para comenzar una variable se le antepone el símbolo "$" el cual es predefinido por el lenguaje PHP.

```
$entidad = $this->global->Estructura->idestructura;
$categorias = $this->objModel->cargarcircuitosdist($entidad);
$cantidad = count($categorias);
```

Figura 17. Estándar de Codificación Nomenclatura de Variables.

3.4 Validación y Verificación de la solución propuesta

Durante y luego del proceso de implementación, el software debe ser comprobado para verificar que cumple con su especificación y realiza las funcionalidades esperadas. La verificación y validación es el nombre dado a estos procesos de análisis y prueba, procesos que tienen lugar en cada etapa del proceso de software, que comienza con las revisiones de los requisitos y continúa con revisiones del diseño e inspecciones de código, hasta las pruebas del producto.

Cualquier producto de ingeniería puede ser probado de una de estas formas:

1. Conociendo la funcionalidad específica para la cual fue diseñado el producto, se pueden llevar a cabo pruebas que demuestren que cada función es completamente operativa.

2. Conociendo el funcionamiento del producto se pueden desarrollar pruebas que aseguren que "todas las piezas encajen", o sea, que la operación interna se ajusta a las especificaciones y que todos los componentes internos se han comprobado de forma adecuada.

Las primeras son conocidas como prueba de caja negra y las segundas como pruebas de caja blanca. En el presente trabajo se utilizaran las pruebas de caja blanca y caja negra para llevar a cabo la validación de la solución.

Prueba de caja negra: Se refiere a las pruebas que se llevan a cabo sobre la interfaz del software, por lo que los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta, así como que la integridad de la información externa se mantiene. Esta prueba examina algunos aspectos del modelo fundamentalmente del sistema sin tener mucho en cuenta la estructura interna del software.

Pressman plantea en 1998 que estas pruebas permiten encontrar:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las Bases de Datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

En la preparación de los casos de prueba son necesarios un conjunto de datos que ayuden en la ejecución de estos y que permitan que el sistema se ejecute en todas sus variantes, los datos utilizados pueden ser válidos o inválidos siempre en consecuencia de si lo que se desea hallar es un error o probar una funcionalidad. Los datos se escogen atendiendo a las especificaciones del problema, sin importar los detalles internos del programa, a fin de verificar que el programa corra bien.

Según Pressman público en el año 2000, para desarrollar la prueba de caja negra existen varias técnicas, entre ellas están:

1. Técnica de la Partición de Equivalencia: esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.

2. Técnica del Análisis de Valores Límites: esta técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.

3. Técnica de Grafos de Causa-Efecto: es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

En este caso se utilizará la Partición de Equivalencia que es una de las más efectivas pues permite examinar los valores válidos e inválidos de las entradas existentes en el software, descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se dirige a la definición de casos de pruebas que descubran clases de errores, reduciendo así el número de casos de prueba a desarrollar.

3.4.1 Diseño de Casos de Prueba

Condiciones de ejecución:

- Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
- Se debe seleccionar el subsistema de Logística.

- Se debe seleccionar la opción **Inventario / Distribución / Adicionar Distribución por Pedido.**
- Se debe haber adicionado al menos un Cuadro Básico.
- Se debe haber adicionado al menos un Pedido.
- Debe estar configurado la forma de distribuir: por Prioridad del cliente, Equitativamente, o Mixta, o que el usuario decida entre estas variantes en el momento de distribuir.

Requisito a probar:

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Adicionar Distribución por Pedido	El sistema debe permitir adicionar una distribución por pedido seleccionando la fecha de los pedidos a visualizar, las áreas MPV y los circuitos de distribución.	EP 1.1: Adicionar una Distribución por pedido seleccionando la fecha de los pedidos a visualizar, las áreas MPV y los circuitos de distribución correctamente sin procesar las urgencias.	- Se muestra la interfaz Gestionar Distribución por Pedido - Se introduce la fecha de pedidos a visualizar y las observaciones. - Se seleccionan las áreas MPV que se desean incluir en la distribución. - Se seleccionan los circuitos a incluir en la distribución. - Se presiona el botón Aceptar. - Se muestra un mensaje de confirmación preguntando si se desean procesar las urgencias. - Se presiona la opción No. - Se muestra una ventana emergente con los pedidos que pueden ser

			seleccionados. - Se seleccionan los pedidos a incluir en la distribución. - Se presiona el botón Aceptar. - Si los datos son correctos el sistema los registra. - El sistema confirma el registro de los datos.
		EP 1.2: Adicionar una Distribución por pedido seleccionando la fecha de los pedidos a visualizar, las áreas MPV y los circuitos de distribución correctamente procesando las urgencias.	- Se muestra la interfaz Gestionar Distribución por Pedido - Se introduce la fecha de pedidos a visualizar y las observaciones. - Se seleccionan las áreas MPV que se desean incluir en la distribución. - Se seleccionan los circuitos a incluir en la distribución. - Se presiona el botón Aceptar. - Se muestra un mensaje de confirmación preguntando si se desean procesar las urgencias. - Se selecciona la opción Sí. - Si los datos son correctos el sistema los registra. - El sistema confirma el registro de los datos.
		EP 1.3: Cancelar	- Se muestra la interfaz Gestionar Distribución por

			Pedido - Se introduce o no la fecha de pedidos a visualizar y las observaciones. - Se seleccionan o no las áreas MPV que se desean incluir en la distribución. - Se seleccionan o no los circuitos a incluir en la distribución. - Se presiona el botón Cancelar.

Tabla 8. Escenarios de prueba del requisito añadir distribución por pedido.

Ver en el Anexo #6 los restantes diseños de casos de pruebas.

Descripción de la variable:

No	Nombre de campo	Tipo	Valor nulo	Descripción
1	Fecha de los pedidos a visualizar	Lista Desplegable	No	Lista de objetos
2	Áreas MPV	Panel de Selección	No	Lista de objetos
3	Circuitos de distribución	Panel de Selección	No	Lista de objetos
4	Observaciones	Cuadro de Texto	Si	Letras y número (no permite Caracteres especiales)

Tabla 9. Descripción de las variables.

Juego de Datos a Probar

Id del escenario	Escenario	Observaciones	Áreas MPV	Fecha	Circuitos	Respuesta del sistema	Resultado de la prueba

EP 1.1	Adicionar una Distribución por pedido seleccionando la fecha de los pedidos a visualizar, las áreas MPV y los circuitos de distribución correctamente sin procesar las urgencias.	N/A	MPV1	10/10/2013	Lisa	El sistema muestra un mensaje de información. "Los datos de la distribución por pedido fueron introducidos correctamente"	
EP 1.2	Adicionar una Distribución por pedido seleccionando la fecha de los pedidos a visualizar, las áreas MPV y los circuitos de distribución correctamente procesando las urgencias.	N/A	MPV2	12/11/2013	Mariano	El sistema muestra un mensaje de información. "Los datos de la distribución por pedido fueron introducidos correctamente"	
EP 1.3	Cancelar	N/A	N/A	N/A	N/A	Se cancela la operación y se cierra la interfaz.	

Tabla 10. Juego de datos a probar.

3.5 Resultados de las Pruebas de Caja Negra

Con el objetivo de comprobar que las funcionalidades del Módulo Distribución y Pedido de Productos funcionan de acuerdo a las especificaciones de los requisitos, se llevan a cabo pruebas de caja negra

a través de la técnica de la Partición de Equivalencia con el objetivo de reducir la cantidad de diseño de casos de prueba a uno por requisito. En la primera iteración realizada se detectaron 23 no conformidades.

Iteración	Ortografía	Funcionales	Validación
1	3	17	3
2	2	3	3
3	0	0	0

Tabla 11. Cantidad de no conformidades por iteración.

Prueba de caja blanca: Permiten inspeccionar la estructura interna del programa. Se diseñan casos de prueba para examinar la lógica del programa. Es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivar casos de prueba que garanticen que:

- Se ejercitan todos los caminos independientes de cada módulo.
- Se ejercitan todas las decisiones lógicas.
- Se ejecutan todos los bucles.
- Se ejecutan las estructuras de datos internas.

En este caso se utilizará la prueba del **camino básico** la cual es una técnica de prueba de la caja blanca propuesta por Tom McCabe. Esta técnica permite obtener una medida de la complejidad lógica de un diseño y usar esta medida como guía para la definición de un conjunto básico. 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.(34)

Para la aplicación de esta técnica se siguieron los siguientes pasos:

Paso 1. Enumerar las sentencias de código del procedimiento.

```
function adicionarPedidoAction() {

        $idestructuracomun = $this->global->Estructura->idestructura;                                   //1

        $post = $this->_request->getPost();                                                             //1

        $respuesta = $this->objModel->adicionarDocumento($idestructuracomun, $post);                   //1
        $array = stripslashes($this->_request->getPost('arrayProductos'));                              //1
        $Productos = json_decode($array);                                                               //1
        if($Productos){                                                                                 //2
        foreach ($Productos as $p) {                                                                    //3
            $Prod = new DatPedidoProducto();                                                            //4
            $Prod->iddocumento = $respuesta['iddocumento'];                                             //4
            $Prod->idprod = $p[0];                                                                      //4
            $Prod->cantprod = $p[1];                                                                    //4
            $this->objModel->insertarProductoPedido($Prod);                                             //4
        }
            echo ("{'codMsg':1,'mensaje':'El pedido ha sido adicionado satisfactoriamente.'}");         //5
        }else                                                                                           //6
            echo ("{'codMsg':3,'mensaje':'El pedido no ha sido adicionado.'}");                         //7

    }                                                                                                   //8
```

Figura 18. Segmento de código a analizar.

La *figura 18* muestra una funcionalidad de la clase GestpedidosModel.php a partir de la cual se enumeran las sentencias, contribuyendo así a determinar la cantidad de nodos que tendrá el grafo asociado a este método.

Paso 2. A partir del código fuente se dibuja un grafo de flujo asociado.

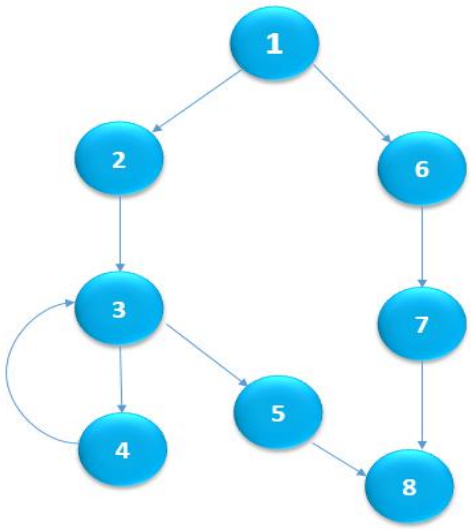

Figura 19. Grafo del método adicionarPedidoAction().

El grafo representado (Figura 19) permite observar fácilmente la diversidad de caminos que se pueden recorrer al ejecutar el código al cual pertenece este grafo. La representación gráfica del código a través del grafo ayuda a determinar la cantidad de aristas, regiones, nodos y nodos predicados del grafo.

Paso 3. Se calcula la complejidad ciclomática del grafo.

Fórmulas para calcular la complejidad ciclomática del grafo:

V (G) = (A – N) + 2

V (G) = (9 - 8) + 2

V (G) = 3 Siendo "A" la cantidad total de aristas y "N" la cantidad total de nodos.

Se puede usar también:

V (G) = P + 1

V (G) = 2 + 1

V (G) = 3

Siendo "P" la cantidad total de nodos predicados (son los nodos de los cuales parten dos o más aristas).

V (G) = R

V (G) = 3

Siendo "R" la cantidad total de regiones, para cada formula "V (G)" representa el valor del cálculo.

El cálculo efectuado mediante las tres fórmulas ha dado el mismo valor, por lo que se puede plantear que la complejidad ciclomática del código es de 3, lo que significa que existen a lo sumo tres posibles caminos por donde el flujo puede circular, este valor representa el límite mínimo del número total de casos de pruebas para el procedimiento tratado. Seguidamente es necesario representar los caminos básicos (Tabla 12) por los que puede recorrer el flujo.

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

Tabla 12. Caminos Básicos.

Luego de haber extraído tres caminos básicos se ejecutan los casos correspondientes a cada uno de estos caminos. Para realizar los casos de pruebas es necesario cumplir con un grupo de normas.

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

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

Entrada: se muestran los parámetros que entran al procedimiento

Resultados Esperados: se expone el resultado que se espera que devuelva el procedimiento.

Caso de prueba para el camino básico 1 [1-6-7-8]:

Descripción: Los datos de entrada son atributos de pedido.

Condición de ejecución: $Productos==null

Entrada: $observaciones, $fechavencimiento, $urgencia, $cliente, $Productos,

Resultados Esperados: Se adiciona el documento satisfactoriamente.

Resultados: No se adiciona el documento.

Salida: -1

Caso de prueba para el camino básico 1 [1-2-3-5-8]:

Descripción: Los datos de entrada son atributos de pedido.

Condición de ejecución: $Productos!=null

Entrada: $observaciones, $fechavencimiento, $urgencia, $cliente, $Productos,

Resultados Esperados: Se adiciona el documento satisfactoriamente.

Resultados: Se adiciona el documento satisfactoriamente.

Salida: 1

Caso de prueba para el camino básico 1 [1-2-3-4-3-5-8]:

Descripción: Los datos de entrada son atributos de pedido.

Condición de ejecución: $Productos!=null

Entrada: $observaciones, $fechavencimiento, $urgencia, $cliente, $Productos,

Resultados Esperados: Se adiciona el documento satisfactoriamente.

Resultados: Se adiciona el documento satisfactoriamente.

Salida: 1

3.6 Conclusiones parciales

Al finalizar el presente capítulo se puede apreciar que:

- Se definieron los estándares de codificación a utilizar durante la implementación del módulo de distribución y pedido de productos.
- Se aplicaron las pruebas internas tanto de caja negra como de caja blanca, utilizando la técnica partición de equivalencia y camino básico respectivamente evaluándose de forma general la calidad del módulo desarrollado.

Conclusiones Generales

- Como resultado de los estudios e investigaciones realizados en este capítulo, quedan claramente expuestos los conceptos relacionados con Distribución, Pedido, Inventario así como su relación con los Sistemas Integrales de Gestión. Realizándose además una valoración crítica de los sistemas que se utilizan actualmente en las entidades cubanas para la gestión de los procesos de distribución y pedido de productos.
- Se generaron los artefactos correspondientes a la disciplina de modelado del negocio donde se relacionan los principales conceptos de la investigación.
- Se identificaron y describieron un total de 36 requisitos funcionales del módulo de distribución y pedido de productos, así como los requisitos no funcionales.
- Se elaboraron los artefactos correspondientes a la disciplina de Análisis y Diseño validándose este último mediante la aplicación de métricas.
- Se implementaron las funcionalidades correspondientes al módulo de distribución y pedido de productos aplicándose los estándares de codificación definidos en la investigación.
- Se aplicaron las pruebas internas, caja negra y caja blanca, utilizando la técnica partición de equivalencia y camino básico; validando tanto la interfaz como el adecuado funcionamiento interno del software.

Recomendaciones

Se recomienda para la futura incorporación de esta investigación:

- La integración del módulo de distribución y pedido de productos con el módulo de despacho del subsistema Inventario con el fin de dar seguimiento a los pedidos y planes de consumo logrando así ejecutar el ciclo completo del proceso de distribución.

Referencias Bibliográficas

1. *ERP (Enterprise Resource Planning). [En línea] [Citado el: 29 de 01 de 2014.] http://oracle.abast.es/oracle_erp.shtml.*

2. *Gemeil, Prof.Dr.Ing.Manuel Torres. Logística Tomo II. Berlín y Ciudad de La Habana: Universitaria, 2004.*

3. *Mailxmail- Curso de Modelos de Implantacion de Gestion de la Calidad Total del Sistema Integrado Gestión. [En línea] [Citado el: 16 de 11 de 2013.] http://www.mailxmail.com/curso-modelos-implantacion-gestion-calidad-total-sistema-integrado-gestion/concepto-sistema-integrado-gestion-sig.*

4. *Los Sistemas Integrados de Gestion. [En línea] [Citado el: 16 de 11 de 2013.] http://www.gestiopolis.com/administracion-estrategia/los-sistemas-integrados-de-gestion.htm.*

5. *Centro de Investigación de Ciencias Administrativas y Gerenciales. [En línea] [Citado el: 24 de 05 de 2014.] http://www.publicaciones.urbe.edu/index.php/cicag/article/viewArticle/1517/2929.*

6. *Gonzalez Brito, Henry Raúl y Lezcano Lozada, Yuniesky. Sistema de Inventario Participativo de la UCI. Ciudad de la Habana: UCI: s.n., 2005.*

7. *del Toro Ríos, José Carlos y González Brito, Henry Raúl. Documento Visión. Proyecto ERP-Cuba. La Habana: s.n., 2009.*

8. *Definiciones. [En línea] [Citado el: 17 de 11 de 2013.] http://definicion.de/distribucion.*

9. *Definición de. [En línea] [Citado el: 17 de 11 de 2013.] http://definicion.de/orden-de-compra/#ixzz2wwEy8a7W.*

10. *ERP SAP R/3. [En línea] [Citado el: 25 de 11 de 2013.] http://www.oocities.org/espanol/emoly188/que_es_sap_r3.htm.*

11. *Implementar sistemas erp para distribución | Sistema erp 100% web. [Accedido: 29-ene-2014]. [En línea] [Citado el: 29 de 01 de 2014.] http://www.virtualwork.cl/?p=304.*

12. *SoluFlex ERP, a quién está dirigirido, ERP, MRP I, MRP II, CRM, ERP Peruano, textiles, Constructoras, Servicios de la Mineria | Soluflex ERP. [En línea] [Citado el: 29 de 01 de 2014.] http://www.soluflex.com.pe/soluflex-erp.*

13. *SAP, SAP España - Software ERP | Sistema Enterprise ResourcePlanning (ERP) de. [En línea] [Citado el: 25 de 11 de 2013.] http://global.sap.com/spain/solutions/business-suite/erp/index.epx.*

14. *Obregón, Ing. William González. Modelo de desarrollo de software. La Habana: s.n., 2012.*

15. *López, Patricia. Ingeniería de Software, Herramienta CASE Visual Paradigm. Cantabri: s.n.*

16. *Servidor de Aplicación, Apache. [En línea] [Citado el: 04 de 11 de 2013.] http://www.digitallearning.es/blog/apache-servidor-web-configuracion-apache2-conf/.*

17. *PgAdmin. [En línea] [Citado el: 20 de 12 de 2013.] http://www.pgadmin.org/visualtour16.php.*

18. *PostgreSQL. [En línea] [Citado el: 02 de 11 de 2013.] http://www.postgresql.org.es/sobre_postgresql.*

19. *Pupo, Yanisleydi Cañete. Libro de ayuda del marco de Trabajo Sauxe, en su versión 2.0. Universidad de las Ciencias Informática, Ciudad de la Habana: s.n., 2010.*

20. *Ext JS. [En línea] [Citado el: 25 de 11 de 2013.] http://es.scribd.com/doc/100415898/SenchaExt-JS.*

21. *Doctrine. [En línea] [Citado el: 22 de 03 de 2014.] http://www.doctrine-project.org/projects/orm.html.*

22. *Zend Framework. [En línea] [Citado el: 12 de 01 de 2014.] http://www.maestrosdelweb.com/editorial/guia-zend/.*

23. *NetBeans IDE. [En línea] [Citado el: 15 de 12 de 2013.] http://netbeans.org/features/index.html.*

24. *PHP. [En línea] [Citado el: 01 de 12 de 2013.] http://ivancamayo.files.wordpress.com/2010/09/php1.pdf.*

25. *AJAX. [En línea] [Citado el: 13 de 01 de 2014.] http://elmasterdelaweb.wikispaces.com/file/view/QUE+ES+AJAX.pdf.*

26. *Subversion. [En línea] [Citado el: 05 de 12 de 2013.] http://www.lyx.org/WebEs.HowToUseSVN.*

27. *Mozilla Firefox. [En línea] [Citado el: 13 de 12 de 2013.] https://www.mozilla.org/es-CL/firefox/features/.*

28. *Hall, Larman - Prentice. UML y Patrones. Introducción al análisis y diseño orientado a objetos.*

29. *Sawyer, Sommerville and. 1997.*

30. *Servicios Web de la UCI | Directorio de Servicios Web UCI. [En línea] [Citado el: 08 de 11 de 2013.] http://uddi.uci.cu/.*

31. *Modelo de datos. [En línea] [Citado el: 23 de 03 de 2014.] http://definicion.de/modelo -de-datos/.*

32. *Schmuller, Joseph. Aprendiendo UML en 24 Horas. 1999.*

33. *LORENZ, MARK y KIDD, JEFF. Object-Oriented Software Metrics. Englewood Cliffs. Nueva Jersey: s.n., 1994.*

34. *Pressman, Roger. Ingeniería del Software. "Un enfoque práctico. [Documento] 2005.*

Bibliografía

1. *ERP (Enterprise Resource Planning). [En línea] [Citado el: 29 de 01 de 2014.] http://oracle.abast.es/oracle_erp.shtml.*

2. *Gemeil, Prof.Dr.Ing.Manuel Torres. Logística Tomo II. Berlín y Ciudad de La Habana: Universitaria, 2004.*

3. *Mailxmail- Curso de Modelos de Implantacion de Gestion de la Calidad Total del Sistema Integrado Gestión. [En línea] [Citado el: 16 de 11 de 2013.] http://www.mailxmail.com/curso-modelos-implantacion-gestion-calidad-total-sistema-integrado-gestion/concepto-sistema-integrado-gestion-sig.*

4. *Los Sistemas Integrados de Gestion. [En línea] [Citado el: 16 de 11 de 2013.] http://www.gestiopolis.com/administracion-estrategia/los-sistemas-integrados-de-gestion.htm.*

5. *Centro de Investigación de Ciencias Administrativas y Gerenciales. [En línea] [Citado el: 24 de 05 de 2014.] http://www.publicaciones.urbe.edu/index.php/cicag/article/viewArticle/1517/2929.*

6. *Gonzalez Brito, Henry Raúl y Lezcano Lozada, Yuniesky. Sistema de Inventario Participativo de la UCI. Ciudad de la Habana: UCI: s.n., 2005.*

7. *del Toro Ríos, José Carlos y González Brito, Henry Raúl. Documento Visión. Proyecto ERP-Cuba. La Habana: s.n., 2009.*

8. *Definiciones. [En línea] [Citado el: 17 de 11 de 2013.] http://definicion.de/distribucion.*

9. *Definición de. [En línea] [Citado el: 17 de 11 de 2013.] http://definicion.de/orden-de-compra/#ixzz2wwEy8a7W.*

10. *ERP SAP R/3. [En línea] [Citado el: 25 de 11 de 2013.] http://www.oocities.org/espanol/emoly188/que_es_sap_r3.htm.*

11. *Implementar sistemas erp para distribución | Sistema erp 100% web. [Accedido: 29-ene-2014]. [En línea] [Citado el: 29 de 01 de 2014.] http://www.virtualwork.cl/?p=304.*

12. *SoluFlex ERP, a quién está dirigirido, ERP, MRP I, MRP II, CRM, ERP Peruano, textiles, Constructoras, Servicios de la Mineria | Soluflex ERP. [En línea] [Citado el: 29 de 01 de 2014.] http://www.soluflex.com.pe/soluflex-erp.*

13. *SAP, SAP España - Software ERP | Sistema Enterprise ResourcePlanning (ERP) de. [En línea] [Citado el: 25 de 11 de 2013.] http://global.sap.com/spain/solutions/business-suite/erp/index.epx.*

14. *Obregón, Ing. William González. Modelo de desarrollo de software. La Habana: s.n., 2012.*

15. *López, Patricia. Ingeniería de Software, Herramienta CASE Visual Paradigm. Cantabri: s.n.*

16. *Servidor de Aplicación, Apache. [En línea] [Citado el: 04 de 11 de 2013.] http://www.digitallearning.es/blog/apache-servidor-web-configuracion-apache2-conf/.*

17. *PgAdmin. [En línea] [Citado el: 20 de 12 de 2013.] http://www.pgadmin.org/visualtour16.php.*

18. *PostgreSQL. [En línea] [Citado el: 02 de 11 de 2013.] http://www.postgresql.org.es/sobre_postgresql.*

19. *Pupo, Yanisleydi Cañete. Libro de ayuda del marco de Trabajo Sauxe, en su versión 2.0. Universidad de las Ciencias Informática, Ciudad de la Habana: s.n., 2010.*

20. *Ext JS. [En línea] [Citado el: 25 de 11 de 2013.] http://es.scribd.com/doc/100415898/SenchaExt-JS.*

21. *Doctrine. [En línea] [Citado el: 22 de 03 de 2014.] http://www.doctrine-project.org/projects/orm.html.*

22. *Zend Framework. [En línea] [Citado el: 12 de 01 de 2014.] http://www.maestrosdelweb.com/editorial/guia-zend/.*

23. *NetBeans IDE. [En línea] [Citado el: 15 de 12 de 2013.] http://netbeans.org/features/index.html.*

24. *PHP. [En línea] [Citado el: 01 de 12 de 2013.] http://ivancamayo.files.wordpress.com/2010/09/php1.pdf.*

25. *AJAX. [En línea] [Citado el: 13 de 01 de 2014.] http://elmasterdelaweb.wikispaces.com/file/view/QUE+ES+AJAX.pdf.*

26. *Subversion. [En línea] [Citado el: 05 de 12 de 2013.] http://www.lyx.org/WebEs.HowToUseSVN.*

27. *Mozilla Firefox. [En línea] [Citado el: 13 de 12 de 2013.] https://www.mozilla.org/es-CL/firefox/features/.*

28. *Hall, Larman - Prentice. UML y Patrones. Introducción al análisis y diseño orientado a objetos.*

29. *Sawyer, Sommerville and. 1997.*

30. *Servicios Web de la UCI | Directorio de Servicios Web UCI. [En línea] [Citado el: 08 de 11 de 2013.] http://uddi.uci.cu/.*

31. *Modelo de datos. [En línea] [Citado el: 23 de 03 de 2014.] http://definicion.de/modelo -de-datos/.*

32. *Schmuller, Joseph. Aprendiendo UML en 24 Horas. 1999.*

33. *LORENZ, MARK y KIDD, JEFF. Object-Oriented Software Metrics. Englewood Cliffs. Nueva Jersey: s.n., 1994.*

34. *Pressman, Roger. Ingeniería del Software. "Un enfoque práctico. [Documento] 2005.*

35. *EcuRed. [En línea] [Citado el: 11 de 02 de 2014.] http://www.ecured.cu/index.php/Modelo_de_dominio.*

36. *ERP - Sistemas de Gestión. [En línea] [Citado el: 14 de 11 de 2013.] http://www.adpime.com/ERP/Es_ERP_intro.htm.*

37. *Modelo de Dominio. [En línea] http://synergix.wordpress.com/2008/07/10/modelo-de-dominio.*

38. *Angelica. [En línea] [Citado el: 20 de Abril de 2012.] http://delfin.mxl.uabc.mx/~angelica/Metricas.pdf.*

39. *Analysis and Design Software. [En línea] http://analysisanddesignsoftware.blogspot.com/2010/10/analisis-de-requrimientos.html.*

Glosario de Términos

[A]

AJAX: Acrónimo en inglés de Asynchronous JavaScript And XML («JavaScript y XML asíncronos»).

[C]

CASE: (Computer Aided Software Engineering), Ingeniería de Software Asistida por Ordenador.

Cedrux: Vocablo formado por la unión de las palabras “Cedro” (fortaleza, resistencia) y “Linux” (tecnología libre).

CSS: (Cascading Style Sheets) Hojas de Estilo en Cascada.

[D]

DOM: (Document Object Model) Modelo en Objetos para la representación de Documentos.

DBAL: (database abstraction layer).

[E]

ERP: (Enterprise Resource Planning) Sistemas de Planificación de Recursos Empresariales.

[F]

Framework: Denota la infraestructura sobre la cual se reúnen un conjunto de lenguajes, herramientas y servicios que simplifican el desarrollo de aplicaciones en entorno de ejecución distribuido.

[G]

GUIs: (Graphics Users Interfaces) Interfaces Gráficas de Usuarios.

[H]

HTML: (Hyper Text Markup Language) Lenguaje de Marcas de Hipertexto.

[I]

IoC: Inversión de control (Inversion of Control en inglés, IoC) sucede cuando es la biblioteca la que invoca el código del usuario.

[J]

JSON: (JavaScript Object Notation) Notación de Objetos de JavaScript.

[O]

ORM: (Object Relational Mapper) Mapeo de Objeto-Relacional.

[P]

PDO: (PHP Data Objects) Capa de abstracción de acceso a datos para PHP.

POO: (Objects Oriented Programming) Programación orientada a objetos.

[S]

SGBD: (en inglés Database Management System, abreviado DBMS). Sistema Gestor de Bases de datos

Software: Es un programa o aplicación de que permite a los usuarios el control o realización de varias tareas y que hacen que el trabajo sea más cómodo, rápido y eficiente.

Subsistema: Son las partes que forman un sistema. Cada sistema está compuesto de subsistemas, los cuales a su vez son parte de otros subsistemas.

Servidor: En informática, un servidor es una computadora que, formando parte de una red, provee servicios a otras computadoras denominadas clientes.

[U]

UML: (Unified Modeling Language) Lenguaje Unificado de Modelado.

[X]

XSLT: (Stylesheet Language Transformations)

XML: (Extensible Markup Language) Lenguaje de Etiquetado Extensible.

[W]

Web: En informática, la World Wide Web, cuya traducción podría ser Red Global Mundial o "Red de Amplitud Mundial", es un sistema de documentos de hipertexto enlazados y accesibles a través de Internet.