Universidad de las Ciencias Informáticas

Facultad 1

Título: Implementación del módulo de gestión de inventario Versión 2.1

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

Autor: Cadete. Alexander García Sánchez

Tutor: Tte. Ing. Lester A. González Gutiérrez

Co-tutor: Tte. Ing. Meylin Martínez Chong

Ciudad de La Habana, 2010.

Año 52 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 firmo la presente a los ____ días del mes de _____ del año _____.

_____ _____

Autor Tutor.

Frase:

El hombre debe transformarse al mismo tiempo que la producción progresa; no realizaríamos una tarea adecuada si fuéramos tan sólo productores de artículos, de materias primas y no fuéramos al mismo tiempo productores de hombres.

DATOS DE CONTACTO

Tutor: Tte. Ing. Lester A. González Gutiérrez.

Correo electrónico: **lagonzalez@uci.cu**

Síntesis del tutor: Graduado de Ingeniero en Ciencias Informáticas en la UCI (2007).

Clasificación: Profesional.

Clasificación del área de desarrollo: SW de gestión.

Co-tutor: Tte. Ing. Meylin Martínez Chong.

Correo electrónico: **meylin@uci.cu**

Síntesis del Co-tutor: Graduado de Ingeniero Informático en la CUJAE, 2005.

Clasificación: Profesional.

Clasificación del área de desarrollo: SW de gestión.

AGRADECIMIENTOS

A mi madre por ser lo más lindo que me ha dado este mundo.

A mis familiares por brindarme apoyo y confianza.

A mis amigos por permanecer a mí lado todos estos años.

A mi vicedecana Matilde Monte de Oca.

A mi tutor (Tte Lester).

A la revolución y a las FAR por brindarme la oportunidad de estudiar en esta universidad (UCI).

DEDICATORIA

En el presente trabajo quiero hacer un humilde reconocimiento a todas aquellas mujeres y hombres que han contribuido en la formación profesional y humana a lo largo de mi vida, la misma que no sería igual sin las enseñanzas y atenciones, tanto en las aulas como fuera; los desvelos, paciencia, el perdón, y el apoyo frente a las adversidades para continuar el camino de frente, sabiendo que una caída implica la oportunidad de soportarla con esperanza, con el constante aprender de los errores propios; así como de la amistad, la cual no puedo definir pero si definir su grandeza y valía. No sería el mismo sin todos aquellos a quienes he conocido, a quienes aprecio y admiro, de quienes he recibido grandes lecciones. Por ello, aseguro que no sería mejor de lo que soy sin ellos. Agradezco en particular:

A la persona más importante en mi vida, mi madre, quien cuyo esfuerzo ha hecho posible este logro.

A mi abuelo (Lalo) y demás familiares.

A mis amigos.

A mi vicedecana Matilde Montes De Oca (mama en la UCI).

A todos muchas gracias, pues en el momento en el que las palabras no son suficientes para expresar lo que el alma desea, rebasa un tono, simplemente queda decir aquello que por su significado extenso y sin límites es, gracias.

RESUMEN

Actualmente la situación general de los procesos de gestión de las entidades presupuestadas y empresariales a escala nacional, está afectada por la existencia de sistemas informáticos que no cumplen con las expectativas de las nuevas tecnologías y la misión del país, en el desarrollo de sus empresas. Proteger los recursos de las empresas así como llegar a la exactitud y confiabilidad de la información económica y contable, ha llevado a la necesidad de mejorar los procesos de gestión de las áreas que las conforman, utilizando plataformas confiables y eficientes.

El propósito de la presente investigación, consiste en la implementación del módulo de gestión de inventario Versión 2.1 del Sistema Integral de Gestión Cedrux[1], con el objetivo de informatizar los procesos de apertura, inventario, devolución del almacén, traspasos entre áreas, documentos de salida, configuración de inventarios y los nomencladores de los medios materiales en correspondencia con los requerimientos de los usuarios. Para esto fue necesario realizar un estudio del estado del arte, donde se analizaron los sistemas existentes, la metodología de desarrollo, las herramientas y tecnologías para la realización de la aplicación. Se realizó un estudio a los objetivos de la organización, el objeto de automatización, la información que se maneja y la propuesta del sistema, procediéndose así a la implementación del sistema. Por último, se realizan las pruebas para validar el sistema existente.

[1] Cedrux: Vocablo formado por la unión de las palabras"Cedro" (fortaleza, resistencia) y "Linux" (tecnología libre).

Contenido

Introducción

En la actualidad la informática y los procesos que se derivan de esta, se manifiestan en el constante desarrollo de la sociedad. Resulta imposible hablar de una empresa donde los principales procesos de producción no sean ejecutados mediante un producto informático. La información es uno de los activos más importantes para empresas y organizaciones, siendo trascendental crear aplicaciones de software cuyo objetivo fundamental sea realizar recuperaciones de datos, lográndose así un acceso rápido y objetivo a la información almacenada.

Hoy en día existen sistemas que facilitan la organización, planificación y actualización de las empresas como es el caso de los ERP (Sistemas de Planificación de Recursos Empresariales), los cuales tienen como inconveniente que su desarrollo es un proceso largo, costoso y complicado, por esta razón requiere de una gran cantidad de desarrolladores. A nivel mundial existen varias herramientas de este tipo como son el caso de: SAP, K2B, Openbravo, CONDOR Enterprise, mientras que en el ámbito nacional podemos mencionar: RODAS XXI, Cedrux, las cuales están destinada para el control de inventario.

Hay que destacar que la gestión de inventarios es un aspecto crítico de la administración de recursos materiales en las empresas. Estos permiten reducir los costes en la actividad empresarial y pueden servir tanto para regular los procesos de producción, como para determinar el nivel de mercancías almacenadas. Es importante señalar que el control de los mismos posibilita la optimización de los tiempos, mantenimiento del nivel competitivo y protección contra aumentos de precios y escasez de materia prima.

En el entorno empresarial cubano la situación de las aplicaciones que viabilizan el control de inventarios está caracterizado por:

- Los sistemas existentes están desarrollados sobre plataformas envejecidas y con poco o ningún criterio de seguridad.
- No existen productos que abarquen la problemática actual de las empresas o instituciones y no soportan mecanismos estándares de integración con otras aplicaciones.
- Los sistemas están desarrollados para un ambiente de escritorio casi ninguno bajo conceptos informáticos multicapa y distribuida en la red.
- No siguen las pollitas actuales del país de migrar hacia software libre.
- La mayoría de estos sistemas tienen procedencia extrajera y no satisfacen todas las necesidades del control de inventario en las entidades.

En el año 2008 la dirección del país propuso la creación de un software de gestión para automatizar los procesos que realizan las entidades, surgiendo en la Universidad de las Ciencias Informáticas el proyecto ERP Cuba donde se desarrolló el módulo Gestión de Inventarios cuyo objetivo es solucionar la problemática existente en las diferentes empresas del país. Al desplegar la primera versión de dicho módulo algunos clientes detectaron una carencia de funcionalidades en diferentes procesos que conforman este subsistema destacándose:

- El proceso de transferencia entre áreas.
- La configuración por tipo de áreas.
- La configuración de operaciones por tipo de áreas
- Nomencladores.

Por lo antes planteado, se puede decir que se hace necesario el desarrollo de una versión informática del módulo gestión de inventario del sistema Cedrux.

La situación anteriormente planteada da lugar al siguiente **problema**: El módulo gestión de inventario desarrollado por el ERP Cuba carece de funcionalidades que permitan un mejor control sobre los medios almacenados.

Luego de lo antes expuesto se definió el **objeto de estudio**: Los procesos de gestión de inventario del sistema Cedrux.

Como **campo de acción** se define: Los procesos de apertura, inventario, recepción, traspasos entre aéreas, configuración de inventarios y los nomencladores del sistema Cedrux.

Para guiar la investigación se planteó la siguiente **idea a defender**: Si se implementan las funcionalidades que le faltan al módulo gestión de inventario del Sistema Cedrux, se obtendrá un producto con mayores prestaciones contribuyendo así a una mejor gestión de inventarios de los medios almacenados.

Para solucionar el problema planteado se propone como **objetivo general** de la investigación: Implementar los nuevos requerimientos a los componentes de inventario del sistema Cedrux.

Se plantean como objetivos específicos.

1. Realizar estado del arte.
2. Realizar el diagrama de componentes, matriz de integración de componentes interna y externa.
3. Implementar los componentes de ajustes, inventarios, transferencia entre áreas, documento de salida, apertura y los nomencladores.
4. Validar el resultado obtenido con pruebas unitarias.

Validar la solución propuesta con pruebas unitarias.

- El **resultado esperado** de este trabajo es un componente funcional, con todos los requerimientos analizados y priorizados por los usuarios funcionales del mismo e integrado con otros módulos y componentes que son necesarios para su correcto funcionamiento y flujo de información. Además, permitirá realizar todas las operaciones de una forma ágil y cumpliendo todo lo establecido.

Los siguientes **métodos teóricos** sustentan la investigación.

Entre los métodos teóricos utilizados se encuentran:

Histórico-Lógico: Se empleó durante la investigación y el estudio de los patrones de diseños, los conceptos, términos y vocabularios propios del objeto de estudio y el campo de acción, se determinó si existe o está en desarrollo un sistema informático similar.

Analítico-Sintético: Permitió integrar y descomponer el conocimiento, descubriendo las relaciones para la utilización de artefactos propuestos, determinando los aspectos esenciales y arribando a conclusiones prácticas y teóricas.

Modelación: *S*e crean abstracciones que explican la realidad, por ejemplo, todos los modelos y diagramas presentados.

El presente documento consta de tres capítulos, desarrollados a partir del estudio realizado. La descripción de los mismos se presenta a continuación:

Capítulo 1. Fundamentación teórica: incluye un estado del arte de las distintas técnicas de programación que existen a nivel internacional, nacional y de la universidad. Las tendencias, técnicas, tecnologías, metodologías, así como de las plataformas de desarrollo que la soportan.

Capítulo 2: Descripción de la solución propuesta: en el segundo capítulo se realizará una valoración crítica del diseño propuesto por el analista, las posibles implementaciones, componentes o módulos ya existentes y que puedan ser rehusados, estrategias de integración, se realizará la descripción de las principales clases que se implementaron.

Capitulo 3. Validación de la solución propuesta: se validó la solución propuesta a través de la realización del diagrama de componentes, realización de los casos de pruebas a los requisitos funcionales y la matriz de integración de componentes interna y externa.

Capítulo 1: Fundamentación teórica

Introducción

En este capítulo se realiza un proceso investigativo de aspectos teóricos y conceptuales relacionados con los procesos de apertura, inventario, devolución del almacén, traspasos entre aéreas, documentos de salida, configuración de inventarios y los nomencladores.

Para la elaboración del trabajo de diploma se aborda el tema de los ERP. Se ofrece una breve panorámica de las tecnologías y tendencias actuales a tener en cuenta para la fabricación del sistema, con el propósito de decidir la tecnología a utilizar.

1.1 Tendencias y tecnologías actuales

1.1.1 ¿Qué es un ERP?

El ERP (Planeación de Recursos de Empresariales) es un sistema integral de gestión empresarial que está diseñado para modelar e informatizar la mayoría de los procesos en una entidad (área de finanzas, comercial, logística, producción, etc.). Su misión es facilitar la planificación de todos los recursos de la entidad.

Lo más destacable de un ERP es que agrupa y ordena toda la información de la entidad en un solo lugar, de este modo cualquier suceso queda a la vista de forma inmediata, posibilitando la toma de decisiones de forma más rápida y segura, acortando los ciclos productivos.

Uno de los módulos gerénciales que existen comúnmente en las empresas y que es integrable a un ERP, es el referente a los procesos del control de inventarios. Este módulo tiene una gran importancia, pues el encargado de controlar los productos en los depósitos de una empresa. Se encarga de examinar la existencia de los productos, así como la entrada y la salida de los mismos del almacén, sistematizando el flujo de mercancía en almacenamiento con el fin de hacer más rentable su posesión y garantizar en cierto grado el éxito de la organización. (1)

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

La mayoría de los ERP tienen en común varios beneficios:

- Solo un sistema para manejar muchos de sus procesos comerciales.
- Integración entre las funciones de las aplicaciones.
- Reduce los costos de gerencia.

- Incrementa el retorno de inversión.
- Fuente de Infraestructura abierta.

1.1.2 Características de los sistemas ERP

Existen tres características principales que distinguen a un sistema de gestión empresarial, de una simple aplicación de gestión.

- **Integración:** El objetivo de un sistema ERP es integrar todos los procesos de la empresa, entendiéndola como una serie de áreas que se relacionan entre sí. Este enfoque permite una mayor eficiencia, reducción de tiempo y costos.
- **Modularidad:** Cada área funcional de la empresa se corresponde con un módulo del sistema de gestión. Estos módulos aunque independientes, comparten información entre sí mediante una base de datos centralizada, lo que facilita la personalización y adaptabilidad por un lado y por otro la facilidad de integración.
- **Adaptabilidad:** Gracias a la modularidad y capacidad de integración de las funcionalidades, un sistema ERP es fácilmente adaptable a las necesidades de cada empresa, permitiendo una total configuración. (2)

1.1.3 ¿Qué es un inventario?

El inventario consiste en verificar físicamente los medios con que cuenta la organización. Su finalidad es llevar a cabo un registro de la existencia, cantidad, características, condiciones de uso, valor de los medios y las personas responsables de su manejo.

Al hacer un inventario hay que tener el mayor cuidado posible, para evitar repeticiones, para que no se incluyan materiales o mercancías que no correspondan. Un inventario es además, una relación de los activos circulantes que posee la entidad en un momento dado y que pueden estar destinados para ser vendidos o como insumos en el proceso productivo. (3)

La gestión de inventario tiene como objetivo:

- Conocer con exactitud la cantidad de medios materiales en almacenamiento de una entidad.
- Llevar el control del uso de los medios materiales, verificando que se mantenga la cantidad y calidad adecuada a las necesidades de la organización.
- Tener el control estricto de las entradas y salidas de los medios materiales del almacén.

- Asignar responsabilidades al personal encargado del uso para garantizar su cuidada y correcta utilización.
- Vigilar el buen uso de los medios, para prevenir reparaciones o reacondicionamientos y así prolongar su utilización.
- Determinar que las existencias físicas inventariadas correspondan al registro en los libros.

1.2 Sistemas existentes

A continuación se realiza un análisis de dos sistemas nacionales y dos extranjeros vinculados a la gestión empresarial.

1.2.1 Gestión de Inventarios (ERP-Cuba)

Es un producto realizado por el ERP-Cuba que se desarrolla en la Universidad de las Ciencias Informáticas (UCI), con el objetivo de resolver los problemas en Ministerio de las Fuerzas Armadas. Es una aplicación web implementado en software libre, multiplataforma, cuenta con diferentes módulos como son: Nomencladores, Ajuste, Inventarios físicos, Configuración, Apertura, Devolución al almacén, Traspasos entre áreas, Documento de salida, Solicitudes, Planes de venta, Distribución. Rompe con todos los estándares que tenía el país en esta rama. Su principal desventaja es que no posee las funcionalidades necesarias para las diferentes empresas del país, por esta razón se decidió realizarle una versión a dicho producto.

1.2.2 Rodas XXI. Versión 3.0

Sistema multiempresa y multiusuario creado por CITMATEL[2] para automatizar la gestión empresarial. Contiene diferentes módulos que pueden usarse integrados o independientes.

El módulo de Inventario de Rodas XXI le permite tener un control detallado de los inventarios de su entidad, realizando en el mismo momento que se registra un movimiento, su contabilización. Se pueden realizar todo tipo de operaciones de entradas y salidas de los almacenes con facilidad en el momento que se desee, generando el documento asociado al movimiento de que se trate de forma automática previa configuración del sistema para ello.

Es posible operar con varios almacenes trabajando cada uno de forma independiente. Este módulo le permitirá además, visualizar información correspondiente a períodos anteriores, tan sólo con cambiar

[2] **CITMATEL:** Empresa de Tecnologías de la Información y Servicios Telemáticos Avanzados.

de período contable a otros anteriores ya cerrados, aunque en dichos períodos no podrá realizar ninguna operación. Esta característica es compartida por todos los módulos de RODAS XXI. (4)

1.2.3 SAP

SAP ERP pone a su disposición gran cantidad de funcionalidades que le permitirán analizar su negocio, optimizar sus finanzas, gestionar sus recursos humanos, operaciones y servicios corporativos. Además, también le proporcionan soporte para cuestiones referentes a la gestión de sistemas de administración de usuarios, gestión de configuración, gestión de datos centralizada y gestión de servicios de web. Es un sistema costoso, que brinda además las posibilidades de mantenimiento.

SAP ERP pone a su disposición gran cantidad de funcionalidades que le permitirán analizar su negocio, optimizar sus finanzas, gestionar sus recursos humanos, operaciones y servicios corporativos.

El subsistema Inventario de SAP permite reducir los costos de almacenamiento, transporte, cumplimiento de pedidos y manipulación de materiales, a su vez mejora el servicio al cliente. Puede mejorar significativamente la rotación de inventario, optimizar el flujo de mercancías y acortar las rutas en su almacén o centro de distribución. Entre los beneficios adicionales de la gestión de inventarios se encuentran la mejora del flujo de caja, la visibilidad y la toma de decisiones.

Para la gestión de almacenes, puede realizarse un seguimiento de la cantidad y el valor de todos sus materiales, realizar un inventario físico y optimizar los recursos del almacén. Los empleados pueden planificar, introducir y documentar movimientos de almacén interno gestionando las entradas y las salidas de mercancías, el almacenamiento, la recogida, el embalaje y los traslados físicos. (5)

1.2.4 CONDOR Enterprise

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

Este brinda mayor autonomía al cliente para efectuar cambios de estructura sin necesidad de la intervención de especialistas, quedando registrados de forma que puedan ser auditables. Incluye la contabilidad multimoneda. (6)

Entre sus módulos se encuentra gestión de inventarios, el cual permite entre otras funcionalidades tener un:

- Control de los esquemas de depósitos y almacenes.

- Realizar inventarios rotativos.
- Serializar los productos.
- Generar transacciones de costeo. (7)

1.3 Valoración crítica de los sistemas estudiados

A partir del estudio realizado a diferentes sistemas de gestión de entidades, tanto nacional como internacional que posean procesos de Control de Inventario, se determinó que las soluciones internacionales (SAP y Condor) usan tecnologías riesgosas debido al bloqueo económico que sufre el país, pues su realización está basada en la plataforma J2EE[3] y su máquina virtual es propiedad de la empresa norteamericana SUN[4] la cual se rige por las leyes de su gobierno. El pago de las patentes o licencias es un aspecto a tener en cuenta, pues el país no se encuentra en condiciones económicas de realizar estos pagos. Otro punto que tenemos que tener en cuenta es que estos sistemas presentan características capitalistas las cuales no son compatibles con el sistema.

La solución nacional (Rodas XXI) está desarrollada sobre software propietario cuando el país está en proceso de emigrar hacia el software libre. Además de ser una aplicación de escritorio, por lo que se hace necesario la instalación en todas las computadoras de la entidad. Sin embargo, con una aplicación web y acceso a la red se tendrá acceso a la aplicación. En el caso de Cedrux se puede decir que esta aplicación cumple con todas las expectativas con que fue realizada, solo se le puede señalar que carece de funcionalidades para su uso en otras entidades.

Los sistemas SAP y Condor realizan los reportes de inventario de forma estática, cuando se tiene que brindar la posibilidad al usuario de filtrar los datos según las características de la recuperación. Por último, el sistema Rodas XXI no realiza informes.

1.4 Proceso de desarrollo de software

Un proceso de desarrollo de software tiene como objetivo la producción eficaz y eficiente de un producto que cumpla con los requisitos del cliente. Este proceso es intensamente intelectual, afectado por la creatividad y juicio de las personas involucradas, aunque un proyecto de desarrollo de software es equiparable en muchos aspectos a cualquier otro proyecto de ingeniería, en el desarrollo de

[3] **J2EE:** Java Platform, Enterprise Edition o Java EE es una plataforma de programación para desarrollar y ejecutar aplicaciones en lenguaje de programación Java.

[4] **SUN:** Stanford University Network, Red de la Universidad de Stanford.

software hay una serie de desafíos adicionales, relativos esencialmente a la naturaleza del producto obtenido. (8)

Modelo de Desarrollo

Con el objetivo de guiar a las personas implicadas en el desarrollo de este software, la dirección del proyecto creó un Modelo de Desarrollo Orientado a Componentes el cual fue desarrollado en colaboración con las Líneas de desarrollo del proyecto ERP-Cuba y de acuerdo con las necesidades que presentaron cada una de ellas y teniendo en cuenta los principales riesgos con los que se cuentan en el proyecto. El modelo propuesto describe la secuencia de actividades de alto nivel para la construcción y desarrollo de soluciones.

Ventajas del Modelo de Desarrollo.

1. Reutilización del software. Conduce a alcanzar un mayor nivel de reutilización de software.
2. Simplifica las pruebas. Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes ensamblados.
3. Simplifica el mantenimiento del sistema. Cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema.
4. Mayor calidad. Dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo. (9)

Este modelo cuenta con los roles mostrados en la figura 1.

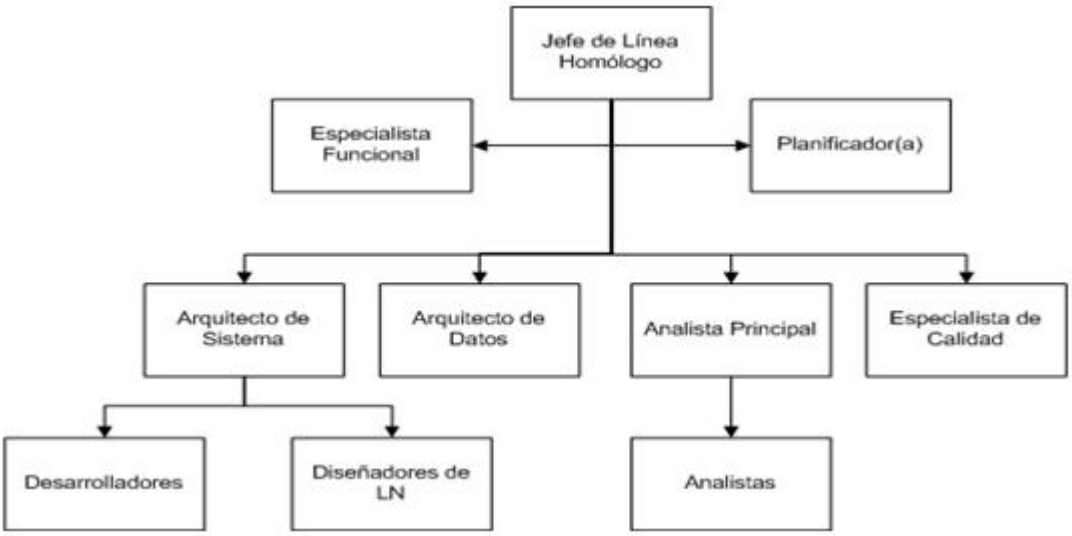

Figura 1 Estructura de las Líneas de desarrollo.

A continuación se muestra la tabla donde aparecen las responsabilidades definidas para cada uno de los roles en el proyecto.

Roles	Responsabilidades
Jefe de Línea de Desarrollo	Responsable de garantizar los cronogramas y compromisos de la línea Supervisar el proceso de desarrollo Organiza y controla el trabajo de los miembros de su línea Controla los indicadores de eficiencia
Planificador	Mantener actualizado el cronograma Mantener actualizada la plantilla de Capital Humano. Planificar y controlar las tareas de los miembros del equipo, según las prioridades Controlar los horarios de trabajo y distribución de máquinas. Llevar las actas de las reuniones y talleres. Controlar los planes de trabajo Individuales
Arquitecto de Sistema	Que se cumplan las políticas y estándares definidos en la Arquitectura. Las decisiones de integración en el proyecto y la Arquitectura del Sistema. Modera el Taller de Diseño.
Arquitecto de Datos	Construye y actualiza el Modelo de Datos, además responde por el manejo y recuperación de la información del mismo
Analista Principal	Dirigir y organizar el trabajo del grupo de analistas de la Línea. Elaborar el Mapa de Procesos de la Línea según los estándares. Participar en la definición y construcción de la Arquitectura de Negocio del ERP.
Especialista de Calidad	Revisar, controlar las normas y estándares que establece el grupo de aseguramiento de la calidad incluyendo el proceso de desarrollo. Guiar al grupo de auditoría y revisiones Coordinar el proceso de diseño de casos de prueba. Coordinar las pruebas de aceptación o liberación.
Especialista Funcional	Participar en las sesiones de trabajo para identificar, describir y validar los procesos de negocio y los requisitos de software Validar, desde el punto de vista funcional, los procesos de negocio y

	requisitos de software Elaborar Casos de Prueba según los estándares establecidos para ello
Analista	Participar en las sesiones de trabajo para identificar, describir y validar los procesos de negocio y los requisitos de software Elaborar la Descripción de Procesos de Negocio, Especificación de Requisitos y Casos de Prueba según los estándares establecidos para ello Participar en el Taller de Diseño
Desarrollador	Diseña y Construye los componentes de software de la línea

Tabla 1 Responsabilidades por roles definidas en el modelo de desarrollo del proyecto ERP Cuba.

Flujos de Trabajo/Artefactos:

- **Inicio**
  - Plan de Iteración (J. de la Línea y Planificador)
  - Plan de Trabajo Individual (Lo elaboran Todos) (Lo controla el planificador y lo evalúa el J. de la Línea)
- **Análisis**
  - Arquitectura de Negocio
    - ✓ Mapa de Proceso (Analista Principal)
    - ✓ Descripción de Procesos de Negocio (Analista)
  - Especificación de Requisitos (Analista)
  - Casos de Prueba (E. Calidad, Analistas y E. Funcionales)
  - Arquitectura del Sistema (Arquitecto de Sistema)
- **Diseño e Implementación**
  - Modelo Conceptual (Arquitecto de Sistema) (Desarrolladores, Analistas)
  - Modelo de Datos (Desarrollador)
  - Diseño de Clases (Desarrollador)
  - Diseño de Interfaz de Usuario (Desarrollador)
  - Release (Arquitecto de Sistema)
- **Prueba**
  - Casos de Prueba (E. de Calidad los refina) (Equipo de Pruebas Central)
  - Registro de no Conformidades (E. de Calidad responsable) (Equipo de Pruebas Central registran las NC)

- **Estabilización e Integración**
    - Historia de la Iteración (J. de la Línea)
    - Informe de Integración (Arquitecto de Sistema)
- **Mantenimiento**
    - Peticiones de Actualizaciones (En teoría es el Dpto. de Implantación)
    - Registro de errores (En teoría es el Dpto. de Implantación

1.5 Herramientas, lenguajes y tecnologías utilizadas

Las herramientas y tecnologías que se describen a continuación en el capítulo, son las utilizadas en la implementación de los componentes. Fue una decisión determinada por el equipo de arquitectura del Centro de Soluciones de Gestión.

1.5.1 Herramientas

Visual Paradigm 6.1

Visual Paradigm para UML es una herramienta UM[5]L profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Entre sus principales características se encuentran:

- Es multiplataforma.
- Soporte de UML versión 2.1.
- Diagramas de Procesos de Negocio - Proceso, Decisión, Actor de negocio, Documento.
- Modelado colaborativo con CVS y Subversion (nueva característica).
- Interoperabilidad con modelos UML2 (meta modelos UML 2.x para plataforma Eclipse) a través de XMI.
- Ingeniería de ida y vuelta.
- Ingeniería inversa - Código a modelo, código a diagrama.
- Generación de código - Modelo a código, diagrama a código.

[5] UML: Lenguaje de modelado unificado

- Editor de Detalles de Casos de Uso - Entorno todo-en-uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso.
- Diagramas EJB - Visualización de sistemas EJB.
- Generación de código y despliegue de EJB´s - Generación de beans para el desarrollo y despliegue de aplicaciones.
- Diagramas de flujo de datos.
- Soporte ORM - Generación de objetos Java desde la base de datos.
- Generación de bases de datos - Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Ingeniería inversa de bases de datos - Desde Sistemas Gestores de Bases de Datos (DBMS) existentes a diagramas de Entidad-Relación.
- Generador de informes para generación de documentación.
- Distribución automática de diagramas - Reorganización de las figuras y conectores de los diagramas UML
- Importación y exportación de ficheros XMI. (10)

Zend Studio para Eclipse 6.0

Zend Studio es uno de los ambientes de desarrollo integrado o *Integrated Development Environment* (IDE) disponible para desarrolladores y agrupa todos los componentes necesarios para el desarrollo de aplicaciones usando PHP. Zend Studio consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Las dos partes se instalan por separado, la del cliente contiene el interfaz de edición y la ayuda. Permite además hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración habrá que disponer de la parte del servidor, que instala Apache y el módulo PHP o, en caso de que estén instalados, los configura para trabajar juntos en depuración. Existen versiones del producto para Windows, Linux y MacOS. Se considera a Zend Studio el entorno IDE más capacitado y con más utilidades cuando es utilizado PHP como lenguaje de programación. Este programa dispone de opciones pensadas con acierto por personas que dominan como nadie la tecnología, de modo que este programa es el idóneo. (11)

pgAdmin III

pgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo una de las más completa y popular con licencia de código abierto. Está escrita en C++ y permite que se pueda usar en Linux, Windows y otros sistemas operativos. Es capaz de gestionar versiones a partir de la PostgreSQL 7.3 ejecutándose en cualquier plataforma.
Está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis así como un editor de código de la parte del servidor. (12)

TortoiseSVN 1.6.2

TortoiseSVN es un cliente gratuito de código abierto para el sistema de control de versiones *Subversion*. Maneja ficheros y directorios a lo largo del tiempo. Los ficheros se almacenan en un *repositorio* central. El repositorio es prácticamente lo mismo que un servidor de ficheros ordinario, salvo que recuerda todos los cambios que se hayan hecho a sus ficheros y directorios. Esto permite que pueda recuperar versiones antiguas de sus ficheros y examinar la historia de cuándo y cómo cambiaron sus datos, y quién hizo el cambio. Esta es la razón por la que mucha gente piensa que Subversion, y los sistemas de control de versiones en general, son una especie de “máquinas del tiempo”. Algunos sistemas de control de versiones también son sistemas de manejo de configuración del software (SCM). Estos sistemas están diseñados específicamente para manejar árboles de código fuente, y tienen muchas características que son específicas para el desarrollo de software - tales como el entendimiento nativo de los lenguajes de programación, o proporcionan herramientas para compilar software. Subversion, sin embargo, no es uno de estos sistemas; es un sistema general que puede ser utilizado para manejar cualquier colección de ficheros, incluyendo código fuente. (13)

1.5.2 Lenguajes

UML (Unified Modeling Language)

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; aun cuando todavía no es un estándar oficial, está apoyado en gran manera por el OMG (*Object Management Group*). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, aspectos concretos

como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

Es importante remarcar que UML es un "lenguaje" para especificar y no un método o un proceso, se utilizan para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para soportar una metodología de desarrollo de software (tal como el Proceso Unificado de Rational), pero no especifica en sí mismo qué metodología o proceso usar. UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas. En todos los procesos basados en UML el concepto de caso de uso desempeña un papel primordial, ya que se emplea para definir los requisitos funcionales del sistema y entorno a ellos se articulan todas las etapas del proceso. (14)

HTML[6]

El lenguaje HTML fue diseñado para estructurar documentos y presentarlos en forma de hipertexto, estableciendo relaciones unidireccionales entre ellos (hipervínculos). Inicialmente fue concebido para visualizar e interconectar el contenido de documentos electrónicos, considerando por ello un pequeño conjunto de etiquetas. Posteriormente, la ventaja que representaba la simplicidad de HTML se convirtió en un inconveniente, ya que su marcado no siempre cubría todos los aspectos de presentación que los usuarios requerían. La solución adoptada fue el desarrollo de extensiones privadas del lenguaje, lo que complicó su estandarización. Las luchas comerciales entre las principales empresas de desarrollo de navegadores web condujeron a un lenguaje HTML que, aunque universalmente utilizado, carece de estandarización real. Desde finales de 1993 existen comités de estandarización, impulsados por el *World Wide Web Consortium20,* que tienden a un modelo único de HTML. (15)

JavaScript

JavaScript es un lenguaje de programación que permite a los desarrolladores crear acciones en sus páginas web, puede ser utilizado por profesionales y para quienes se inician en el desarrollo y diseño

[6] **HTML:** HyperText Markup Language (Lenguaje de Marcas de Hipertexto)

de sitios web. No requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos.
Entre los diferentes servicios que se encuentran realizados con JavaScript en Internet se encuentran: Correo, Chat, Buscadores de Información.
También podemos encontrar o crear códigos para insertarlos en las páginas como: Reloj, Contadores de visitas, Fechas, Calculadoras, Validadores de formularios, Detectores de navegadores e idiomas. (16)

Ajax

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript y XML asíncronos), es una técnica de desarrollo web para crear aplicaciones interactivas. Éstas se ejecutan en el cliente, es decir, en el navegador del usuario, y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma, es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma.

AJAX es una combinación de tres tecnologías ya existentes:
XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.
DOM7 accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y Script, para mostrar e interactuar dinámicamente con la información presentada.
El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto iframe en lugar del XMLHttpRequest para realizar dichos intercambios.
XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML preformateado, texto plano y JSON.
AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente. (17)

Json

JSON es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del Lenguaje de Programación JavaScript. JSON es un formato de texto que es completamente lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias.

El formato JSON es el más adecuado para la respuesta del servidor cuando la acción Ajax debe devolver una estructura de datos a la página que realizó la llamada de forma que se pueda procesar con JavaScript. Este mecanismo es útil por ejemplo cuando una sola petición Ajax debe actualizar varios elementos en la página.

JSON se ha convertido en un estándar en el desarrollo de aplicaciones web. Los servicios web proponen la utilización de JSON en vez de XML para permitir la integración de servicios en el navegador del usuario en vez de en el servidor. El formato JSON es seguramente la mejor opción para el intercambio de información entre el servidor y las funciones JavaScript. (18)

CSS

CSS es la tecnología desarrollada por el World Wide Web Consortium (W3C) con el fin de separar la estructura de la presentación, permite crear páginas web de una manera más exacta, gracias a las CSS el desarrollador es mucho más dueño de los resultados finales de la página, pudiendo hacer muchas cosas que no se podía hacer utilizando solamente HTML, como fuentes, colores, márgenes, líneas, altura, anchura, imágenes de fondo, posicionamiento avanzado y muchos otros temas.

Entre los beneficios concretos de CSS encontramos:

- Control de la presentación de muchos documentos desde una única hoja de estilo.
- Control más preciso de la presentación.

- Aplicación de diferentes presentaciones a diferentes tipos de medios (pantalla, impresión, etc.).
- Numerosas técnicas avanzadas y sofisticadas. (19)

PHP

PHP es el acrónimo de *Hipertext Preprocesor*. Es un lenguaje de programación del lado del servidor, gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación. Se escribe dentro del código HTML, lo que lo hace realmente fácil de utilizar. Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor web. Esto hace que cualquier sistema pueda ser compatible con el lenguaje y significa una ventaja importante.

Algunas de las más importantes capacidades de PHP son: compatibilidad con las bases de datos más comunes, como MySQL, mSQL, Oracle, Informix, y ODBC[8], por ejemplo. Incluye funciones para el envío de correo electrónico, upload de archivos, crear dinámicamente en el servidor imágenes en formato GIF, incluso animadas y una lista interminable de utilidades adicionales.
No es un lenguaje de marcas y la meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. PHP también soporta el uso de otros servicios que usen protocolos como IMAP[9], SNMP[10], POP3[11], HTTP y derivados. También se pueden abrir sockets de red directos (raw sockets) e interactuar con otros protocolos.
Al ser un lenguaje libre dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas: Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de Acrobat Reader), analizar código XML. Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes. (20)

[8] **ODBC:** (Open Database Connectivity) Estándar de acceso a Bases de datos.

[9] **IMAP:** (Internet Message Access Protocol) Protocolo de acceso a mensajes almacenados en un servidor.

[10] **SNMP:** (Simple Network Management Protocol) Protocolo de la capa de aplicación que facilita el intercambio de información de administración.

[11] **POP3:** (Post Office Protocol) Protocolo 3 de Correo, está diseñado para recibir correos, no para enviarlo.

PostgreSQL8.3

PostgreSQL es el servidor de bases de datos de código abierto más utilizado por aquellos programadores que realizan aplicaciones cliente/servidor, complejas o críticas. Es además la alternativa más cercana a MySQL cuando se precisa de operaciones avanzadas como transacciones, procedimientos almacenados, vistas, o cuando se precisa de una base de datos que soporte gran cantidad de información. Este servidor es muy utilizado actualmente y es una opción económica a SQL Server, pues su costo es menor y las prestaciones son similares. El mismo se puede utilizar sobre cualquier sistema operativo, característica que lo pone por encima de SQL Server y a la par con MySQL. Algunas de sus principales características son:

- Es libre.
- Alta concurrencia.
- Amplia variedad de tipos nativos.
- Integridad transaccional.
- Herencia de tablas.
- Replicación que permiten la duplicación de bases de datos maestras en múltiples sitios de réplica.
- Interfaces nativas para ODBC, JDBC, C, C++, PHP, Perl, TCL, ECPG, Python y Ruby.
- Procedimientos almacenados. (21)

1.5.3 Tecnologías

Ningún proceso de desarrollo de software está exento de riesgos, pero si a pesar de esto no se lleva una metodología adecuada el resultado final será la insatisfacción de clientes y desarrolladores. El uso de una arquitectura adecuada es importante, pues define la robustez del sistema y a la futura reutilización del código.

Marcos de trabajo

Los Frameworks (Marcos de Trabajo), no son más que arquitecturas definidas para un determinado dominio de la aplicación que contiene un conjunto de componentes implementados y sus interfaces bien definidas, estos componentes se pueden utilizar, redefinir y crear nuevos componentes.

Zend_Ext Framework

Es un framework de código abierto, diseñado para PHP 5 o superior. Se deriva de Zend Framework por lo que cumple con todas sus características. Posee 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, y se le agregó el IoC para la integración entre los módulos o componentes. Tiene incorporado el ORM Doctrine Framework para trabajo en la capa de abstracción a base de datos y el ExtJS Framework para el desarrollo de las vistas.

Ventajas:

- Comunicación entre sesiones (Global Concept).
- Abstracción de las operaciones transaccionales (TransactionManager).
- Notificaciones declarativas a eventos, software y personas (Trazas).
- Emulación de eventos (Trazas).
- Evaluador de rendimiento (Trazas).
- Extensión de los dominios de tipo (Validador).
- Basada en componentes.
- Principios de integración distribuidos.
- Conectores y configuraciones mediante interfaces bien definidas.
- Integración por IoC y con posibilidad de emulación. (22)

En la siguiente figura se muestra la estructura de Zend Framework.

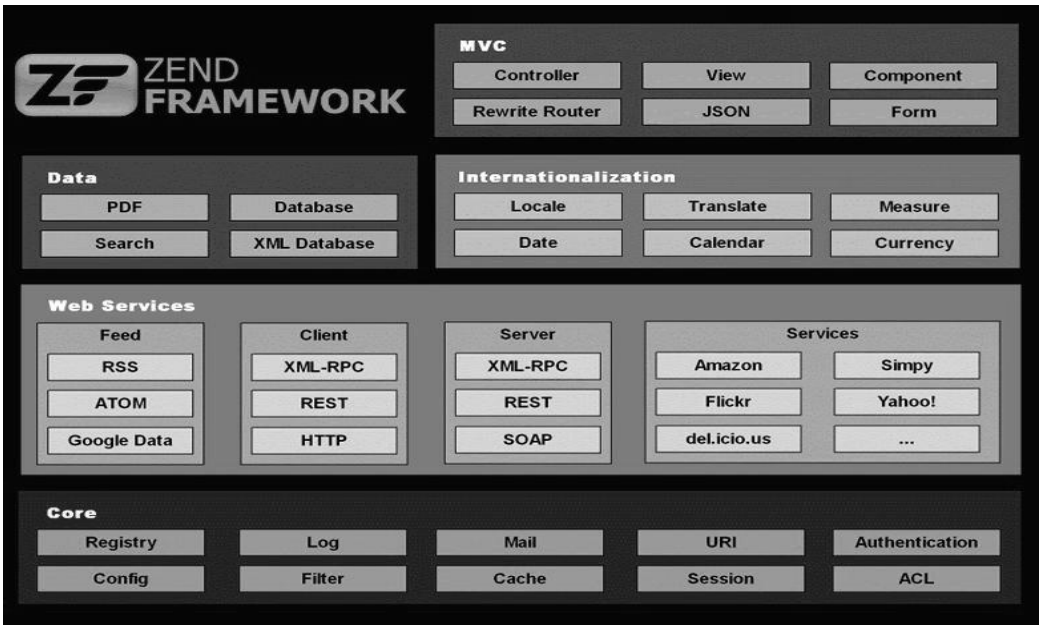

Figura 2 Estructura de Zend Framework

ExtJs

ExtJS es una librería construida con JavaScript que proporciona una interfaz cuya potencia radica en la rica colección de componentes para el diseño de interfaces del lado del cliente. Tiene incluidos la mayoría de los controles de los formularios Web incluyendo tablas 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 de componentes incluye componentes para el manejo de datos, lectura de XML, lectura de datos JSON e implementaciones basadas en AJAX. Presenta el uso de JavaScript con una programación orientada a objetos.

JavaScript es una técnica de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano, posibilitando realizar cambios sobre una página sin necesidad de recargarla, aumentando de esta forma la interactividad, velocidad y usabilidad de la misma. (23)

Algunas tecnologías por la que está compuesto AJAX son:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML, XSLT y JSON, para el intercambio y la manipulación de información.

- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.
- Sus características principales son:
- Mayor rapidez e interactividad, al estilo aplicaciones de escritorio.
- Reduce significativamente la carga de información continua del servidor, actualizando solamente porciones de la página.
- Reduce de manera significativa los tiempos de carga inicial.

Doctrine Framework

El Framework Doctrine es un potente y completo sistema de mapas de relaciones de objetos (Object Relational Mapper, ORM por sus siglas en inglés) para PHP 5.2+ con una base de datos con capas de abstracción incorporada. Además es multiplataforma. (24)

Ventajas:

- Posibilidad de exportar una base de datos existente a sus clases correspondientes y también a la inversa.
- Accede a la base de datos utilizando la programación orientada a objetos a través del patrón Active Record.
- Tiene su propio lenguaje de consultas y trabaja de manera rápida y eficiente.
- Fácil integración a los principales frameworks de desarrollo utilizados actualmente.

En la siguiente figura se muestra la estructura de Doctrine ORM.

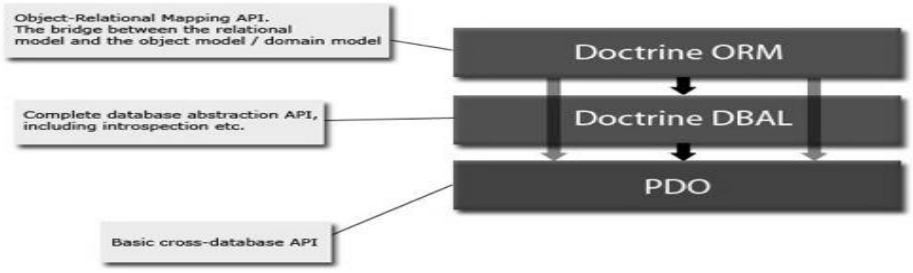

Figura 3 Estructura de Doctrine ORM

UCID Framework

Es el Framework encargado del trabajo con las vistas. Abarca la integración de ExtJs Framework con el sistema incluyendo el integrador de interfaz, el generador de interfaz dinámica y la impresión de documentos. Integra la iconografía, los diferentes temas de escritorio de la aplicación y el multilenguaje.

ExtJs Framework

Es una librería construida con JavaScript que proporciona una interfaz a las famosas librerías de Yahoo, jQuery, y Prototype + Scriptaculous, su potencia radica en la rica colección de componentes para el diseño de GUI's del lado del cliente haciendo uso extensivo de Ajax.

ExtJS es neutral al lenguaje que se use en el servidor. Siempre que el resultado se envíe a la página en el formato adecuado, ExtJS no se preocupará de lo que pase en el servidor. Hay docenas de widgets a escoger en ExtJS, incluyendo composiciones automáticas de páginas, pestañas, menús, barras de herramientas, diálogos, vistas en árbol. Proporciona un selector de nodos DOM extremadamente poderoso llamado DomQuery (puede usarse como una librería independiente, pero en el contexto de ExtJS se usará para seleccionar elementos para poder interactuar con ellos a través de la interfaz Element, contiene mucho de los métodos y propiedades de DOM que se necesitará proporcionando una interfaz conveniente, unificada y multinavegador).
Entre los componentes que esta librería ofrece encontramos cuadros de diálogo, menús, tablas editables, layouts, paneles, pestañas y todo lo necesario para construir atractivos desarrollos al estilo de Web 2.0. (25)

Ventajas:

- La orientación a objetos intensos hará modular todos los scripts.
- El diseño está completamente separado de la funcionalidad.
- Funciones comunes como validación, combox editables, ventanas desplazables (con minimizar y maximizar), grillas editables, son muy fáciles de implementar.
- Buena y amplia documentación, así como también su comunidad.

Desventajas:

- Crear un sistema serio con esta herramienta requiere un previo uso prolongado, ya que se dificulta el manejo de los nuevos objetos en su extensa y bien documentada API. El tiempo de aprendizaje puede llegar a compararse con aprender a programar en un lenguaje nuevo.

- Al estar todo el sitio en JS, no podrá ser accesible para los buscadores, limitando su uso a sistemas y no sitios web.

1.5.4 Arquitectura

La arquitectura de software, llamada también la arquitectura lógica, es el diseño de más alto nivel de la estructura de un sistema, aunque hay varias definiciones y no hay prácticamente ninguna que sea totalmente aceptada. La definición oficial se ha acordado que sea la que brinda el documento de IEEE Std 1471-2000, donde se declara que:

"La Arquitectura del Software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán, y los principios que orientan su diseño y evolución". (27)

Modelo-Vista-Controlador (MVC)

El MVC es un patrón arquitectural aportado por SmallTalk a finales de los sesenta, hoy en día muy difundido en uso en aplicaciones de entorno web. El MVC tiene tres piezas clave que se reparten la responsabilidad de la aplicación:

El modelo (model): Responsable de toda la lógica y estado del dominio de negocio.

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

La vista (view): Responsable de la presentación del dominio de negocio.

- Recibir datos del modelo y lo muestra al usuario.
- Tienen un registro de su controlador asociado (normalmente porque además lo instancia).
- Pueden dar el servicio de "Actualización", para que sea invocado por el controlador o por el modelo (cuando es un modelo activo que informa de los cambios en los datos producidos por otros agentes).

El controlador (controller): Responsable del flujo de control, la navegabilidad y el estado de la entrada del usuario. (28)

- Recibir los eventos de entrada.
- Contiene reglas de gestión de eventos, del tipo "SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas.

En la siguiente figura se muestra la estructura del estilo arquitectónico Modelo-Vista-Controlador.

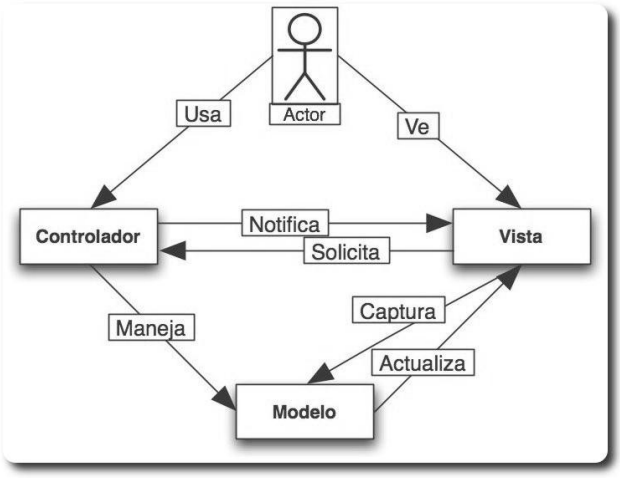

Figura 4 Estructura Modelo-Vista-Controlador

Tanto la vista como el controlador dependen del modelo, el cual no depende de las otras clases. Esta separación permite construir y probar el modelo, independientemente de la representación visual.

Cliente/Servidor

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

- El servidor presenta a todos sus clientes una interfaz única y bien definida.
- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente. (29)

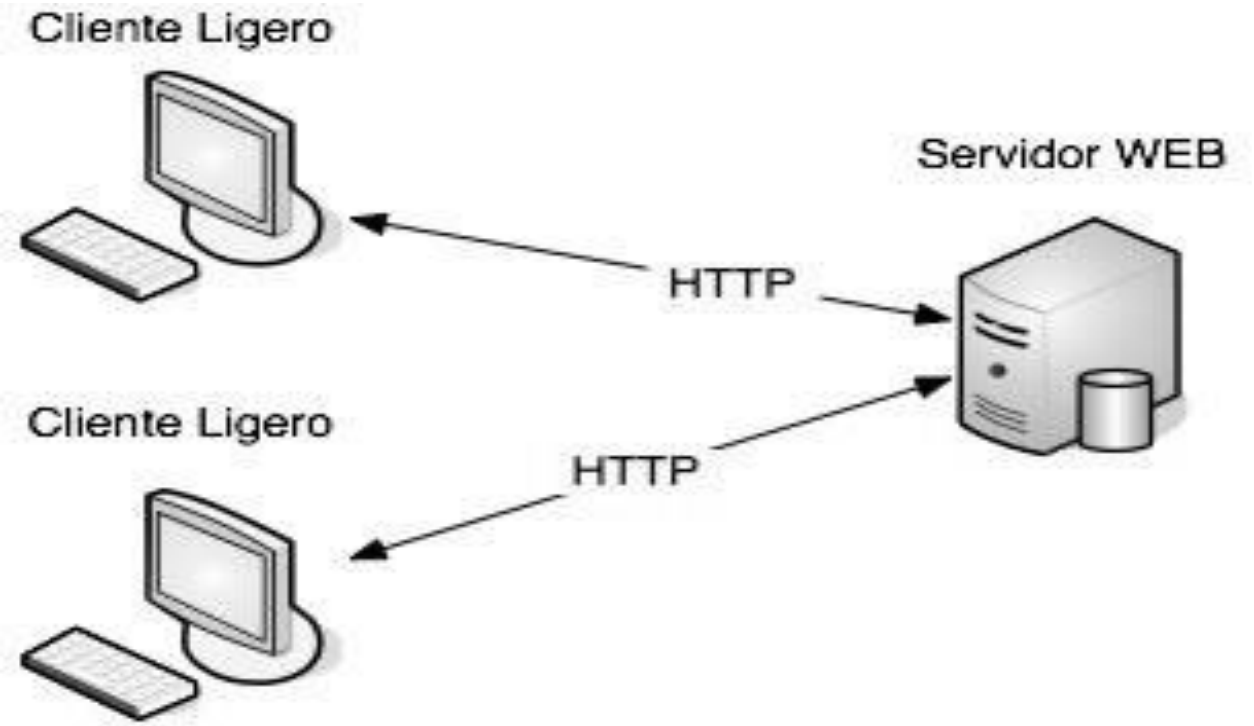

Figura 5. Arquitectura Cliente/Servidor

Inversión de Control

La Inversión de Control (IoC, *del inglés*) es una técnica de implementación en la que se invierte el flujo de programación realizando llamadas a funciones. Se especifican respuestas deseadas a sucesos o solicitudes de datos concretas, recibiendo la información necesaria de parte de alguna entidad o arquitectura externa que realiza la acción de control. Es el propio código del usuario quien es invocado por la librería. En este caso la librería implementa estructuras de alto nivel y el código del usuario las tareas de bajo nivel. (30)

Servidor web utilizado

Un servidor web es un programa que implementa el protocolo HTTP (*Hypertext Transfer Protocol*). Se encarga de atender las peticiones HTTP llevadas a cabo por un cliente y responder con el contenido que el cliente solicita. Este protocolo HTTP está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML. Entre los servidores web más usados a nivel mundial están el Apache, y el *Internet Information Server.*

Debido a las características que presenta el sistema que se desea construir y a la factibilidad del servidor Wamp Server, el mismo ha sido el seleccionado para llevar a cabo el desarrollo de la aplicación.

Wamp Server es una de las alternativas gratuitas más usadas para la gestión de servidores de prueba locales y resulta una completa alternativa al conocido AppServ también de uso libre.

Algunas de las características de Wamp Server son:

- Incluye PhpMyAdmin.
- Servidor Web con soporte para Windows.
- Se pueden instalar todos los CMS disponibles (Alfresco, Joomla!, Drupal, Redmine, Mediawiki, Trac, WordPress, Roller, DokuWiki, Subversion, SugarCRM, Flux, TangoCMS, Dot Clear, bbpress.
- Incluye Apache, MySQL, PHP.
- Fácil instalación y Administración.
- Bajo consumo de Memoria.
- Actualizaciones continuas. (31)

Navegadores

Un navegador, navegador red o navegador web (del inglés, *web browser*) es un programa que permite visualizar la información que contiene una página web (ya esté está alojada en un servidor dentro de la World Wide Web o en uno local).

El navegador interpreta el código, HTML generalmente, en el que está escrita la página web y lo presenta en pantalla permitiendo al usuario interactuar con su contenido y navegar hacia otros lugares de la red mediante enlaces o hipervínculos. (32)

Mozilla Firefox 3.6

Mozilla Firefox es un navegador de Internet libre y de código abierto. Es usado para visualizar páginas web. Incluye corrector ortográfico, búsqueda progresiva, marcadores dinámicos y un sistema de búsqueda integrado que utiliza el motor de búsqueda que desee el usuario. Además, se pueden añadir funciones a través de complementos desarrollados por terceros.

Las características de Mozilla Firefox 3.6 son las siguientes:

- Multiplataforma.
- Cuenta con una protección *antiphishing*, *antimalware* e integración con el antivirus.

- La navegación por pestañas.
- Bloqueador de ventanas emergentes.
- Múltiples Extensiones.
- Incluye un buscador integrado en la interfaz que hace búsquedas en Google.
- Posee gestor de descargas.
- Utiliza el sistema SSL para proteger la comunicación con los servidores web, utilizando fuerte criptografía cuando se utiliza el protocolo HTTPS. (33)

1.5.4 Control de Versiones

Se llama control de versiones a la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, así como las posibles especializaciones realizadas.

Un sistema de control de versiones debe proporcionar:

- Mecanismo de almacenaje de los elementos que deba gestionar (ej. archivos de texto, imágenes, documentación).
- Posibilidad de realizar cambios sobre los elementos almacenados (ej. modificaciones parciales, añadir, borrar, renombrar o mover elementos).
- Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos (normalmente pudiendo volver o extraer un estado anterior del producto). (34)

Subversión 1.4.5

Subversión es un controlador de versiones empleado en la administración de archivos utilizados en el desarrollo de software o contenido. CVS[12] considerado su antecesor es uno de los controladores de versiones más utilizados en proyectos de software libre, sin embargo, a pesar de su amplio uso, el mismo diseño de CVS resultó ineficiente para diversos grupos de usuarios, y ante estas inconformidades se dio inicio al proyecto que hoy es conocido como Subversión, el mismo que ha empezado a socavar el dominio de CVS.

[12] **CVS** Concurrent Versions System.

Ventajas

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.
- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente; Tiene costo de complejidad constante (O (1)) y no lineal (O (n)) como en CVS.
- Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos).
- Maneja eficientemente archivos binarios (a diferencia de CVS que los trata internamente como si fueran de texto). (35)

1.6 Conclusiones parciales

En este capítulo se conocieron diferentes sistemas nacionales e internacionales que abarcan el control de inventario. Se realizó un análisis de las herramientas y tecnologías vigentes para la construcción de aplicaciones sobre la web, que permitió la selección de las mismas para el desarrollo. Se definió por parte de la subdirección de arquitectura del centro de solución de gestión un modelo de desarrollo basados en componentes.

Capítulo 2: Descripción de la solución propuesta

Introducción

En este capítulo se abordarán los principales procesos que se automatizaran. Se mencionarán los documentos de manipulación. Se describe la propuesta de solución para dar una visión general del funcionamiento del software permitiendo así el modelado de los procesos de negocio. Por último, se exponen los requisitos funcionales detectados por los analistas.

2.1 Problema

Al módulo de inventario desarrollado para el ERP Cuba le faltan funcionalidades para realizar una correcta gestión de los medios almacenados.

2.2 Situación problémica

Al desplegar la primera versión del módulo gestión de inventarios algunos clientes detectaron una carencia de funcionalidades en diferentes procesos que conforman este módulo a continuación se mencionan algunas de estas funcionalidades:

- Se debe agregar el proceso de transferencia entre áreas.
- Se debe agregar configuración por tipo de áreas.
- Se debe agregar configuración de operaciones por tipo de áreas
- Se debe agregar nuevos nomencladores.

Por estas razones se hace necesario realizar la Versión 2.1 del Sistema Gestión de Inventario.

2.3 Objetivos de la organización

El principal objetivo de la organización es la integración de nuevas funcionalidades para que el producto no solo cumpla con los requisitos de la empresa, sino de todas las entidades del país, para darle cumplimiento a este objetivo se debe dirigir el trabajo a fin de mantener un registro y control adecuado de los medios materiales a través de la informatización de este proceso, mediante un sistema basado en tecnología web que permita una mejor y mayor seguridad en el almacenamiento y procesamiento de la información, gestionar los recursos necesarios para conocer los medios con que se cuentan, conocer en tiempo real la disponibilidad de recursos para la toma de decisiones en un período de tiempo óptimo y lograr una mejor distribución de los medios materiales.

2.4 Objeto de automatización

Para realizar el proceso inventario se selecciona la hoja de inventario donde desea registrar el inventario, se verifica que se pueda realizar el inventario, se registra el resultado de la verificación realizada, si el activo está en el lugar que le corresponde, el usuario debe marcar el activo como verificado y se registra el inventario, además, de una selección Parcial: (10% de los productos del área deseada) o Total: 100% de los productos del área seleccionada; Para cierre: Sólo manda a contar los productos que se movieron del último cierre a la fecha. Dicho inventario se utiliza para el cierre contable, el sistema pregunta imprimir con existencia o sin existencia, quitar en el reporte el precio y el valor del producto a contar. Cuando se realiza un inventario al cierre, hay que pasar todo lo que haya del producto en el área de predespacho para el área de mercancía para la venta. En el proceso de apertura se deben agregar los campos asociados a los datos del lote y debe implementar el tratamiento a los mismos. Validar que se registre el lote al confirmar el documento. Validar que se ubique el producto al menos en el área al confirmar el documento, así como importar los datos de los productos. En el proceso de configuración, las ubicaciones deben ser hasta el lote del producto. Por último, los nomencladores deben brindar la posibilidad de agregar datos del cliente o proveedor, especificar los tipos de áreas y los tipos de procesos que se deben realizar en dichas áreas.

2.5 Información que se maneja

Los procesos de apertura, recepción, inventario, devolución del almacén, traspasos entre aéreas, configuración de inventarios y los nomencladores del Sistema Cedrux.

2.6 Propuesta de sistema

Al iniciar la sesión el sistema se verificarán los permisos que posee el usuario que inició sesión, permitiendo acceder solo a las entidades y subsistemas a los cuales tenga acceso, así como a las funcionalidades y las distintas acciones definidas según el rol que desempeña. Para llevar a cabo el proceso de Gestión de Inventarios se realizan las siguientes funcionalidades:

1 Gestionar producto al nomenclador.

1.1 Adicionar producto al nomenclador.

1.2 Modificar producto del nomenclador.

Donde se realizan las siguientes operaciones:

1.2.1 El sistema muestra la especialidad a la que pertenece el producto.

1.2.2 El sistema muestra un listado de los productos del nomenclador.

1.2.3 Se selecciona el producto que se desea modificar.

1.2.4 El sistema muestra y permite editar los datos del producto.

1.2.5 Si el producto que se desea modificar pertenece al surtido; se modifican y/o seleccionan los datos del producto: código, código auxiliar, descripción, peso, precio unitario, unidad de medida, número de pieza.

1.2.6 Se seleccionan los siguientes datos del producto: línea del producto, forma farmacéutica, tipo de producto, grupo farmacológico, tipo de producto, tipo de control, nacionalidad, grupo.

1.2.7 Si el producto es un producto terminado, se introducen los siguientes datos: forma de presentación.

1.2.8 Se seleccionan los siguientes datos: categorización.

1.2.9 Si el producto presenta lote, se marca en el checkbox: tiene lote.

1.3 Eliminar producto del nomenclador.

1.4 Buscar producto en el nomenclador.

Donde se realizan las siguientes operaciones:

1.4.1 Se insertan y/o seleccionan los criterios de búsqueda: código, número de pieza, descripción, unidad de medida, activo.

1.4.2 El sistema muestra un listado de los productos que cumplen los criterios de búsqueda especificados. Se muestran los datos del producto: código, código auxiliar, número de pieza, descripción, especialidad, grupo, unidad de medida, activo del producto (estado del producto: activo o no activo), tiene lote, línea de producto, forma farmacéutica, tipo de producto.

1.5 Realizar búsqueda avanzada del producto en el nomenclador.

Donde se realizan las siguientes operaciones:

1.5.1 Se insertan y/o seleccionan los criterios de búsqueda: código, código auxiliar, número de pieza, descripción, línea de producto, grupo farmacológico, forma farmacéutica, tipo de producto.

1.5.2 El sistema muestra un listado de los productos que cumplen los criterios de búsqueda especificados. Se muestran los datos del producto: código, código auxiliar, número de pieza, descripción, especialidad, grupo, unidad de medida, activo del producto (estado del producto: activo o no activo), tiene lote, línea de producto, forma farmacéutica, tipo de producto.

1.6 Listar productos en el nomenclador.

Donde se realizan la siguiente operación:

1.6.1 El sistema muestra un listado de los productos en el nomenclador. El sistema muestra los siguientes datos del producto: código, código auxiliar, número de pieza, descripción, especialidad, grupo, unidad de medida, activo del producto (estado del producto: activo o no activo), tiene lote, línea de producto, forma farmacéutica, tipo de producto.

2 Gestionar planta de producción por entidad.

El sistema permitirá realizar las siguientes acciones:

2.1 Adicionar planta de producción a la entidad.

2.2 Eliminar planta de producción de la entidad.

2.3 Listar plantas de producción de la entidad.

2.4 Buscar planta de producción de la entidad.

2.5 Exportar plantas de producción de la entidad.

Donde se realizan las siguientes operaciones:

2.5.1 El sistema permite seleccionar la ubicación hacia donde se exportará la información, el formato y registrar el nombre.

2.5.2 El sistema exporta los datos.

3 Gestionar línea del producto por entidad.

El sistema permitirá realizar las siguientes operaciones:

3.1 Adicionar línea de producto a la entidad.

3.2 Eliminar línea de producto de la entidad.

3.3 Listar líneas de producto de la entidad.

Donde se realizan la siguiente operación:

3.3.1 El sistema muestra un listado de las líneas de producto que pertenecen a la entidad, incluyendo los datos de la línea de producto: descripción.

4 Gestionar tipo de producto por entidad.

El sistema permitirá realizar las siguientes operaciones:

4.1 Adicionar tipo de producto a la entidad.

4.2 Eliminar tipo de producto de la entidad.

4.3 Listar tipos de producto de la entidad.

4.4 Buscar tipo de producto de la entidad.

4.5 Exportar tipos de producto de la entidad.

5 Gestionar grupo farmacológico por entidad.

El sistema permitirá realizar las siguientes operaciones:

5.1 Adicionar grupo farmacológico a la entidad.

5.2 Eliminar grupo farmacológico de la entidad.

5.3 Listar grupos farmacológicos de la entidad.

5.4 Buscar grupo farmacológico de la entidad.

5.5 Exportar grupos farmacológicos de la entidad.

6 Gestionar forma farmacéutica por entidad.

El sistema permitirá realizar las siguientes operaciones:

6.1 Adicionar forma farmacéutica a la entidad.

6.2 Eliminar forma farmacéutica de la entidad.

6.3 Listar formas farmacéuticas de la entidad.

6.4 Buscar forma farmacéutica de la entidad.

6.5 Exportar formas farmacéuticas de la entidad.

7 Gestionar línea de producto.

El sistema permitirá realizar las siguientes operaciones:

7.1 Adicionar línea de producto.

7.2 Modificar línea de producto.

7.3 Eliminar línea de producto.

7.4 Listar líneas de producto.

7.5 Buscar línea de producto.

7.6 Exportar líneas de productos.

8 Gestionar planta de producción.

El sistema permitirá realizar las siguientes operaciones:

8.1 Adicionar planta de producción.

8.2 Modificar planta de producción.

8.3 Eliminar planta de producción.

8.4 Listar planta de producción.

8.5 Buscar planta de producción.

8.6 Exportar plantas de producción.

9 Gestionar tipo de áreas.

El sistema permitirá realizar las siguientes operaciones:

9.1 Adicionar tipo de área.

9.2 Modificar tipo de área.

9.3 Eliminar tipo de área.

9.4 Listar tipos de áreas.

9.5 Buscar tipo de área.

9.6 Exportar tipos de áreas.

10 Gestionar tipo de producto.

El sistema permitirá realizar las siguientes operaciones:

10.1 Adicionar tipo de producto.

10.2 Modificar tipo de producto.

10.3 Eliminar tipo de producto.

10.4 Listar tipos de producto.

10.5 Buscar tipo de producto.

10.6 Exportar tipos de producto.

11 Gestionar grupo farmacológico.

El sistema permitirá realizar las siguientes operaciones:

11.1 Eliminar grupo farmacológico.

11.2 Listar grupos farmacológicos.

11.3 Buscar grupo farmacológico.

11.4 Exportar grupos farmacológicos.

11.5 Adicionar grupo farmacológico.

11.6 Modificar grupo farmacológico.

Donde se realizan las siguientes operaciones:

11.6.1 Se selecciona el grupo farmacológico a modificar.

11.6.2 El sistema muestra y permite editar los datos del grupo farmacológico.

11.6.3 Se introducen los datos del grupo farmacológico: código, descripción.

11.6.4 El sistema valida.

11.6.5 Si los datos son correctos el sistema los registra.

11.6.7 El sistema confirma el registro de los datos.

12 Gestionar forma farmacéutica.

El sistema permitirá realizar las siguientes operaciones:

12.1 Adicionar forma farmacéutica.

12.2 Modificar forma farmacéutica.

Donde se realizan las siguientes operaciones:

12.2.1 Se selecciona la forma farmacéutica a modificar.

12.2.2 El sistema muestra y permite editar los datos de la forma farmacéutica.

12.2.3 Se introducen los datos de la forma farmacéutica: código, descripción.

12.2.4 El sistema valida.

12.2.5 Si los datos son correctos el sistema los registra.

12.2.6 El sistema confirma el registro de los datos.

12.3 Eliminar forma farmacéutica

12.4 Listar formas farmacéuticas.

12.5 Buscar forma farmacéutica.

12.6 Exportar formas farmacéuticas.

13 Gestionar productos de la transferencia entre áreas.

13.1 El sistema permitirá realizar las siguientes operaciones:

13.2 Adicionar producto a la transferencia entre áreas.

13.3 Eliminar producto de la transferencia entre áreas.

13.4 Buscar producto de la transferencia entre áreas.

14 Gestionar productos entre aéreas.

El sistema permitirá realizar las siguientes operaciones:

14.1 Adicionar transferencia entre áreas.

14.2 Eliminar transferencia entre áreas.

14.3 Listar transferencias entre áreas.

14.4 Consultar transferencia entre áreas.

Donde se realizan las siguientes operaciones:

14.4.1 El sistema permite seleccionar una transferencia entre áreas.

14.4.2 El sistema muestra los siguientes datos del documento: nombre entidad, nombre almacén, número del documento, fecha de emisión, estado del documento, descripción del área origen, descripción del área destino.

14.4.3 Si el producto pertenece a la línea de Materia Prima, Material de envase o Insumo, el sistema muestra el importe total del documento: importe total en MN o en importe total en CUC.

14.4.4 El sistema muestra los siguientes datos del producto: código CUP, número de pieza, descripción, unidad medida, existencia, cantidad, existencia final, precio costo MN, precio costo CUC, importe MN, importe CUC.

14.4.5 Se selecciona el producto para consultar el desglose de lote.

14.5 Buscar transferencia entre áreas.

14.6 Realizar búsqueda avanzada de transferencia.

Donde se realizan las siguientes operaciones:

14.6.1 Se insertan los criterios de búsqueda avanzada: número del documento, año, estado del, documento, descripción del área origen, descripción del área destino, creado por, aprobado por, rango de fecha (desde - hasta).

14.6.2 El sistema muestra un listado de las transferencias entre áreas que cumplen los criterios de búsqueda especificados. Se muestran los datos de la transferencia entre áreas: número del, documento, año, estado del documento, descripción del área origen, descripción del área destino, cantidad de productos, fecha de emisión, creado por, aprobado por.

15 Imprimir transferencia entre áreas.

Donde se realizan las siguientes operaciones:

15.1 El sistema permite seleccionar los elementos que se desean imprimir.

15.2 El sistema imprime los datos (ver formatos de salida).

16 Imprimir listado de transferencias entre áreas.

Donde se realiza la siguiente operación:

16.1 El sistema imprime en un listado los datos (ver formatos de salida) de los documentos listados.

17 Confirmar transferencia entre áreas.

Donde se realizan las siguientes operaciones:

17.1 Se selecciona la transferencia entre áreas a confirmar.

17.2 Se solicita confirmación para confirmar la transferencia entre áreas.

17.3 El usuario ratifica que se confirme la transferencia entre áreas.

17.4 El sistema verifica que se pueda confirmar la transferencia entre áreas.

17.5 El sistema confirma la operación.

18 Cancelar estado de transferencia entre áreas.

Donde se realizan las siguientes operaciones:

18.1 Se selecciona la transferencia entre áreas a cancelar el estado.

18.2 Se solicita confirmación para cancelar el estado de la transferencia entre áreas.

18.3 El usuario confirma que se cancele el estado a la transferencia entre áreas.

18.4 El sistema verifica que se pueda cancelar el estado de la transferencia entre áreas.

18.5 El sistema confirma la operación.

19 Aprobar transferencia entre áreas.

Donde se realizan las siguientes operaciones:

19.1 Se selecciona la transferencia entre áreas a aprobar.

19.2 El usuario aprueba la transferencia entre áreas.

19.3 El sistema verifica que se pueda aprobar la transferencia entre áreas.

19.4 El sistema actualiza la existencia por lote en el área origen y en el área destino:

En el área origen:

Existencia por lote = existencia por lote (antes de la transferencia) – cantidad por lote.

En el área destino:

Existencia por lote = existencia por lote (antes de la transferencia) + cantidad por lote.

19.5 El sistema actualiza la existencia del producto en el área origen y en el área destino:

En el área origen:

Existencia = existencia (antes de la transferencia) – cantidad.

En el área destino:

Existencia = existencia (antes de la transferencia) + cantidad.

20 Gestionar vale de devolución.

El sistema permitirá realizar las siguientes operaciones:

20.1 Adicionar vale de devolución a partir de un vale de entrega.

Donde se realizan las siguientes operaciones:

20.1.1 Se cargan los siguientes datos del documento: nombre de la entidad, nombre del almacén, número del documento, fecha de emisión del documento, estado del documento.

201.2 Se selecciona el vale de entrega a devolver.

20.1.3 Se cargan los siguientes datos del vale de entrega seleccionado: código del centro de costo, descripción del centro de costo, código del elemento de gasto, descripción del elemento de gasto, número de vale de entrega.

20.1.4 Se introducen los siguientes datos del vale de devolución: observaciones.

20.1.5 El sistema valida los datos introducidos.

20.1.6 Si los datos son correctos el sistema los registra.

20.1.7 El sistema confirma el registro de los datos.

20.2 Adicionar vale de devolución.

20.3 Modificar vale de devolución.

Donde se realizan las siguientes operaciones:

20.3.1 Se selecciona el vale de devolución a modificar.

20.3.2 El sistema muestra los siguientes datos del vale de devolución: nombre de la entidad, nombre del almacén, número del documento, fecha de emisión del documento, estado del documento.

20.3.3 Si el vale de devolución no es a partir de un vale de entrega. El sistema muestra y permite editar los siguientes datos: observaciones.

Se selecciona el centro de costo.

20.3.4 El sistema muestra los datos del centro de costo: código, descripción.

20.3.5 El sistema carga los siguientes datos del centro de costo: código del elemento de gasto, descripción del elemento de gasto.

20.3.6 El sistema valida los datos introducidos.

20.3.7 Si los datos son correctos el sistema los registra.

20.3.8 El sistema confirma el registro de los datos.

20.4 Eliminar vale de devolución.

20.5 Listar vales de devolución.

20.6 Consultar vale de devolución.

Donde se realizan las siguientes operaciones:

20.6.1 El sistema permite seleccionar un vale de devolución.

20.6.2 El sistema muestra los siguientes datos del documento: nombre de la entidad, nombre del almacén, número del documento, fecha de emisión del documento, estado del documento.

20.6.3 Si el producto pertenece a la línea de Materia Prima o Insumo, el sistema muestra el importe total del documento: importe Total en MN, importe Total en CUC.

20.6.4 El sistema muestra los siguientes datos de los productos registrados al documento: descripción del área (área donde se encuentre ubicado el producto), código CUP, descripción, número de pieza, unidad de medida, existencia (Existencia del producto en el área de MPV), cantidad (Cantidad reflejada en el vale de entrega), existencia final (Existencia final del producto en el área de MPV después del movimiento), descripción del área, precio costo MN, importe MN, precio costo CUC, importe CUC.

20.7 Buscar vale de devolución.

20.8 Realizar una búsqueda avanzada de vale de devolución.

20.9 Imprimir vale de devolución.

21 Gestionar producto del vale de devolución.

El sistema permitirá realizar las siguientes operaciones:

21.1 Adicionar producto al vale de devolución a partir de un vale de entrega.

Donde se realizan las siguientes operaciones:

21.1.1 Se selecciona el documento al cual se le va a adicionar los productos.

21.1.2 El sistema carga los siguientes datos del documento: nombre de la entidad, nombre del almacén, número del documento, fecha de emisión del documento, estado del documento, código del centro de costo, descripción del centro de costo, código del elemento de gasto, descripción del elemento de gasto, número de vale de entrega.

21.1.3 Si el producto pertenece a la línea de producto Materia Prima, Material de Envase o Insumo, el sistema muestra el importe total del documento: importe total MN, importe total CUC.

21.1.4 El sistema muestra los siguientes datos de los productos incluidos en el vale de entrega seleccionado: código CUP, descripción, unidad de medida, existencia (Existencia del producto en el área de MPV), cantidad (Cantidad reflejada en el vale de entrega), existencia final (Existencia final del producto en el área de MPV después del movimiento), descripción del área, precio costo MN, importe MN, precio costo CUC, importe CUC

21.1.5 Si el producto lleva lote, se selecciona el producto para consultar el desglose de lote.

21.1.6 El sistema valida los datos introducidos.

21.1.7 Si los datos son correctos el sistema los registra.

21.1.8 El sistema actualiza la existencia final del producto: existencia final = existencia – cantidad.

21.1.9 El sistema actualiza la cantidad disponible del producto: cantidad disponible = cantidad disponible – cantidad.

21.1.10 El sistema confirma el registro de los datos.

21.2 Adicionar producto al vale de devolución.

21.3 Modificar producto del vale de devolución.

Donde se realizan las siguientes operaciones:

21.3.1 Si el producto lleva lote, se selecciona el producto a consultar para modificar la cantidad por desglose de lote.

21.3.2 El sistema valida los datos introducidos.

21.3.3 Si los datos son correctos el sistema los registra.

21.3.4 El sistema actualiza los siguientes datos del producto: cantidad = ∑cantidad por lote.

21.3.5 El sistema actualiza la existencia final del producto: existencia final = existencia + cantidad.

21.3.6 El sistema actualiza la cantidad disponible del producto: cantidad disponible = cantidad disponible + cantidad.

21.3.7 Si el producto pertenece a la línea Materia Prima, Material de envase o Insumo, el sistema actualiza el importe total del documento: importe Total en MN= ∑ importe MN de cada producto, importe Total en CUC= ∑ importe CUC de cada producto.

Donde:

Importe MN = cantidad * precio MN.

Importe CUC = cantidad * precio CUC.

21.3.8 El sistema confirma el registro de los datos.

21.4 Eliminar producto del vale de devolución.

21.5 Listar productos del vale de devolución.

21.6 Buscar producto del vale de devolución.

22 Contabilizar vale de devolución.

23 Cancelar estado al vale de devolución.

24 Confirmar vale de devolución.

25 Gestionar configuración de inventarios.

El sistema permitirá realizar las siguientes operaciones:

25.1 Adicionar estructura de ubicación.

25.2 Modificar estructura de ubicación.

Donde se realizan las siguientes operaciones:

25.2.1 Si la estructura de ubicación que se desea modificar es un área; selecciona el área a modificar.

25.2.2 El sistema muestra y permite editar los datos del área.

25.2.3 Se introducen y/o seleccionan los datos del área: código, formato.

25.2.4 Se selecciona en una lista desplegable el tipo de área: tipo de área.

25.2.5 El sistema valida los datos introducidos.

25.2.6 Si los datos son correctos el sistema los registra.

25.2.7 El sistema confirma el registro de los datos.

25.3 Eliminar estructura de ubicación.

25.4 Listar estructuras de ubicación.

26 Relacionar tipos de aéreas y relacionen.

27 Adicionar relación entre tipo de área y operación.

Donde se realizan las siguientes operaciones:

27.1 Se selecciona el tipo de área en una lista desplegable: descripción.

27.2 Se muestra un listado de las operaciones predefinidas para la entidad: operación.

27.3 Se selecciona la o las operaciones a asociar al tipo de área seleccionado.

27.4 El sistema valida los datos introducidos.

27.5 Si los datos son correctos el sistema los registra en una lista de operaciones asociadas al tipo de área seleccionada.

27.6 El sistema confirma el registro de los datos.

28 Eliminar relación de tipo de área y operación.

29 Listar operaciones por tipo de área.

2.6.1 Requerimientos funcionales

- RF 1: Gestionar producto al nomenclador.
  - RF 1.1: Adicionar producto al nomenclador.
  - RF 1.2: Modificar producto del nomenclador.
  - RF 1.3: Eliminar producto del nomenclador.
  - RF 1.4: Buscar producto en el nomenclador.
  - RF 1.5: Realizar búsqueda avanzada del producto en el nomenclador.
  - RF 1.6: Listar productos en el nomenclador.
- RF2: Gestionar planta de producción por entidad.
  - RF 2.1Adicionar planta de producción a la entidad.
  - RF 2.2Eliminar planta de producción de la entidad.
  - RF 2.3Listar plantas de producción de la entidad.
  - RF 2.4Buscar planta de producción de la entidad.
  - RF 2.5Exportar plantas de producción de la entidad.
- RF 3: Gestionar línea del producto por entidad.
  - RF 3.1Adicionar línea de producto a la entidad.
  - RF 3.2Eliminar línea de producto de la entidad.

- RF 3.3Listar líneas de producto de la entidad.

- RF 4: Gestionar tipo de producto por entidad.
  - RF 4.1Adicionar tipo de producto a la entidad.
  - RF 4.2Eliminar tipo de producto de la entidad.
  - RF 4.3Listar tipos de producto de la entidad.
  - RF 4.4Buscar tipo de producto de la entidad.
  - RF 4.5Exportar tipos de producto de la entidad
- RF 5: Gestionar grupo farmacológico por entidad.
  - RF 5.1Adicionar grupo farmacológico a la entidad.
  - RF 5.2Eliminar grupo farmacológico de la entidad.
  - RF 5.3Listar grupos farmacológicos de la entidad.
  - RF 5.4Buscar grupo farmacológico de la entidad.
  - RF 5.5Exportar grupos farmacológicos de la entidad.
- RF 6: Gestionar forma farmacéutica por entidad.
  - RF 6.1: Adicionar forma farmacéutica a la entidad.
  - RF 6.2 Eliminar forma farmacéutica de la entidad.
  - RF 6.3 Listar formas farmacéuticas de la entidad.
  - RF 6.4 Buscar forma farmacéutica de la entidad.
  - RF 6.5 Exportar formas farmacéuticas de la entidad.
- RF 7: Gestionar línea de producto.
  - RF 7.1 Adicionar línea de producto.
  - RF 7.2 Modificar línea de producto.
  - RF 7.3 Eliminar línea de producto.
  - RF 7.4 Listar líneas de producto.
  - RF 7.5 Buscar línea de producto.
  - RF 7.6 Exportar líneas de productos.
- RF 8: Gestionar planta de producción.
  - RF 8.1: Adicionar planta de producción.
  - RF 8.2: Modificar planta de producción.
  - RF 8.3: Eliminar planta de producción.
  - RF 8.4: Listar planta de producción.

- RF 8.5: Buscar planta de producción.
- RF 8.5: Exportar plantas de producción.

➢ RF 9: Gestionar tipo de áreas.
- RF 9.1: Adicionar tipo de área
- RF 9.2: Modificar tipo de área.
- RF 9.3: Eliminar tipo de área.
- RF 9.4: Listar tipos de áreas.
- RF 9.5: Buscar tipo de área.
- RF 9.6: Exportar tipos de áreas.

➢ RF 10: Gestionar tipo de producto.
- RF 10.1: Adicionar tipo de producto.
- RF 10.2: Modificar tipo de producto.
- RF 10.2: Modificar tipo de producto.
- RF 10.3: Eliminar tipo de producto.
- RF 10.4: Listar tipos de producto.
- RF 10.5: Buscar tipo de producto.
- RF 10.6: Exportar tipos de producto.

➢ RF 11: Gestionar grupo farmacológico.
- RF 11.1: Adicionar grupo farmacológico.
- RF 11.2: Modificar grupo farmacológico.
- RF 11.3: Eliminar grupo farmacológico.
- RF 11.4: Listar grupos farmacológicos.
- RF 11.4: Buscar grupo farmacológico.
- RF 11.6: Exportar grupos farmacológicos.

➢ RF 12: Gestionar forma farmacéutica.
- RF 12.1: Adicionar forma farmacéutica.
- RF 12.2: Modificar forma farmacéutica.
- RF 12.3: Eliminar forma farmacéutica.
- RF 12.4: Listar formas farmacéuticas.
- RF 12.5: Buscar forma farmacéutica.
- RF 12.6: Exportar formas farmacéuticas.

- RF 13: Gestionar productos de la transferencia entre áreas.
  - RF 13.1: Adicionar producto a la transferencia entre áreas.
  - RF 13.2: Eliminar producto de la transferencia entre áreas.
  - RF 13.3: Buscar producto de la transferencia entre áreas.
- RF 14: Gestionar productos entre aéreas.
  - RF 14.1: Adicionar transferencia entre áreas.
  - RF 14.2: Eliminar transferencia entre áreas.
  - RF 14.3: Listar transferencias entre áreas.
  - RF 14.4: Consultar transferencia entre áreas.
  - RF 14.5: Buscar transferencia entre áreas.
  - RF 14.6: Realizar búsqueda avanzada de transferencia.
- RF 15: Imprimir transferencia entre áreas.
- RF 16: Imprimir listado de transferencias entre áreas.
- RF 17: Confirmar transferencia entre áreas.
- RF 18: Cancelar estado de transferencia entre áreas.
- RF 19: Aprobar transferencia entre áreas.
- RF 20: Gestionar vale de devolución.
  - RF 20.1: Adicionar vale de devolución a partir de un vale de entrega.
  - RF 20.2: Adicionar vale de devolución.
  - RF 20.3: Modificar vale de devolución.
  - RF 20.4: Eliminar vale de devolución.
  - RF 20.5: Listar vales de devolución.
  - RF 20.6: Consultar vale de devolución.
  - RF 20.7: Buscar vale de devolución.
  - RF 20.8: Realizar una búsqueda avanzada de vale de devolución.
  - RF 20.9: Imprimir vale de devolución.
- RF 21: Gestionar producto del vale de devolución.
  - RF 21.1: Adicionar producto al vale de devolución a partir de un vale de entrega.
  - RF 21.2: Adicionar producto al vale de devolución.
  - RF 21.3: Modificar producto del vale de devolución.
  - RF 21.4: Eliminar producto del vale de devolución.

- RF 21.5: Listar productos del vale de devolución.
- RF 21.6: Buscar producto del vale de devolución.

➢ RF 22: Contabilizar vale de devolución.

1 RF 23: Cancelar estado al vale de devolución.

2 RF 24: Confirmar vale de devolución.

3 RF 25: Gestionar estructura de ubicación.

- RF 25.1: Adicionar estructura de ubicación.
- RF 25.2: Modificar estructura de ubicación.
- RF 25.3: Eliminar estructura de ubicación.

RF 25.4: Listar estructuras de ubicación

➢ RF 26: Relacionar tipos de aéreas y relacionen.

➢ RF 27: Adicionar relación entre tipo de área y operación.

➢ RF 28: Eliminar relación de tipo de área y operación.

➢ RF 29: Listar operaciones por tipo de área.

2.6.2 Requerimientos no Funcionales

➢ Requerimientos de Software.

➢ Requerimientos de Hardware.

➢ Restricciones en el diseño y la implementación.

➢ Requerimientos de apariencia o interfaz externa.

➢ Requerimientos de Usabilidad.

➢ Requerimientos de Soporte.

➢ Requerimientos Legales.

➢ Requerimientos de confiabilidad.

➢ Requerimientos de interfaz Interna.

➢ Requerimientos de Seguridad.

2.7 Conclusiones Parciales

Con la realización de este capítulo se han identificado los procesos que serán objeto de automatización. Se mencionan los documentos procesados para la manipulación de la información. Se planteó la propuesta de solución para dar una visión de cómo funcionará el producto. Por último, podemos decir que se han identificado los requisitos funcionales del software.

Capítulo 3: Implementación y prueba

Introducción

El desarrollo de un software es un proceso largo y costoso que requiere de gran cantidad de personal, por lo que las posibilidades de errores o fallos son innumerables, por lo que debemos realizar alguna actividad que garantice la calidad; las pruebas son un elemento crítico para que se asegure la calidad de un producto informático. Por esta razón se realizan las pruebas unitarias que son las encargadas de verificar el correcto funcionamiento del código. Asegurando así que cada módulo funcione correctamente por separado.

3.1 Implementación

Diagramas de componentes

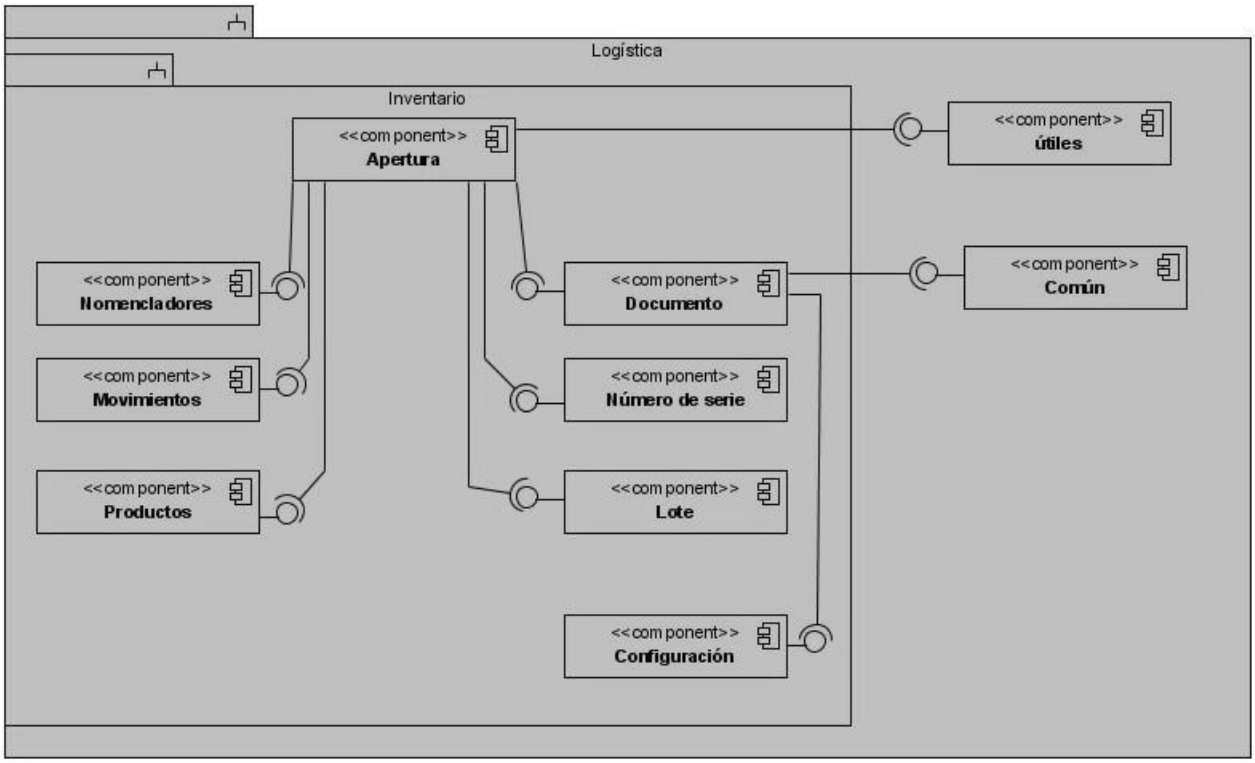

Figura 6. Diagramas de componentes del proceso Apertura.

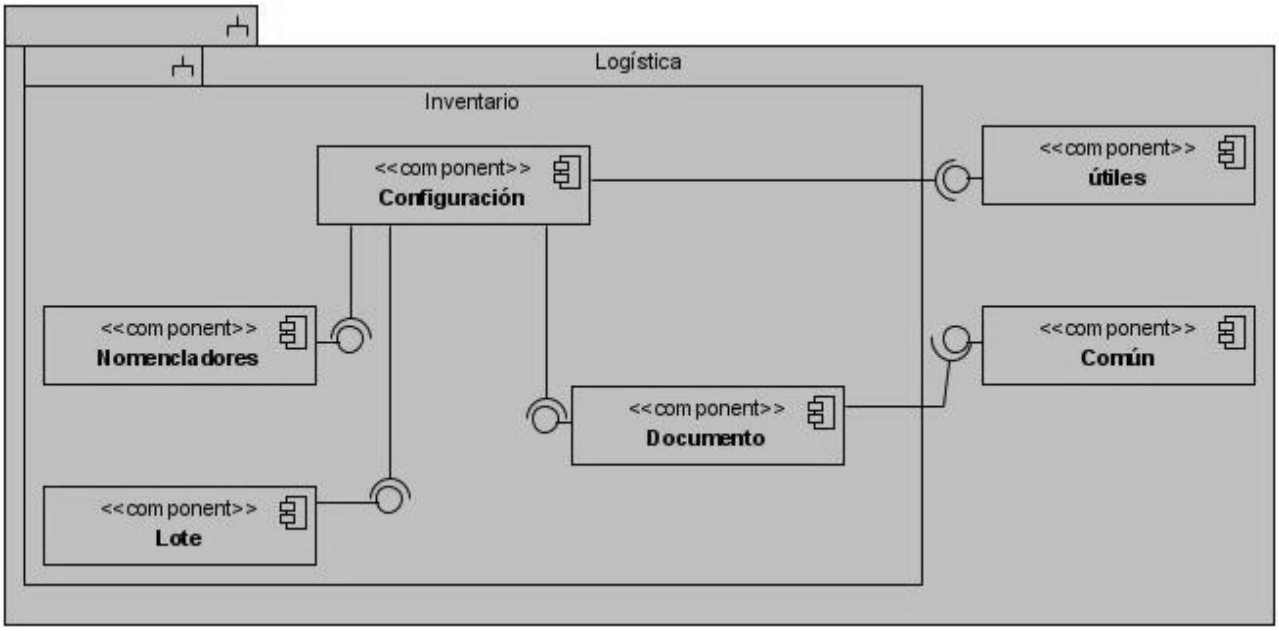

Figura 7. Diagrama de componente del proceso configuración.

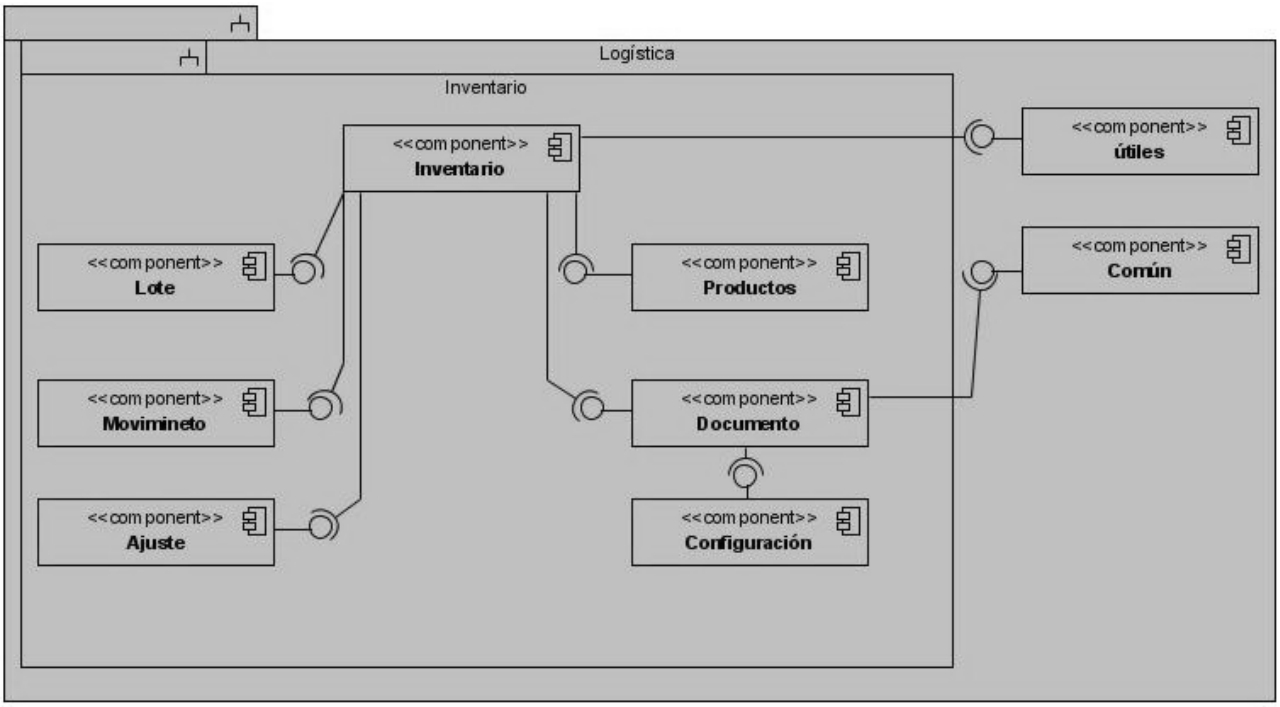

Figura 8. Diagrama de componente del proceso Inventario.

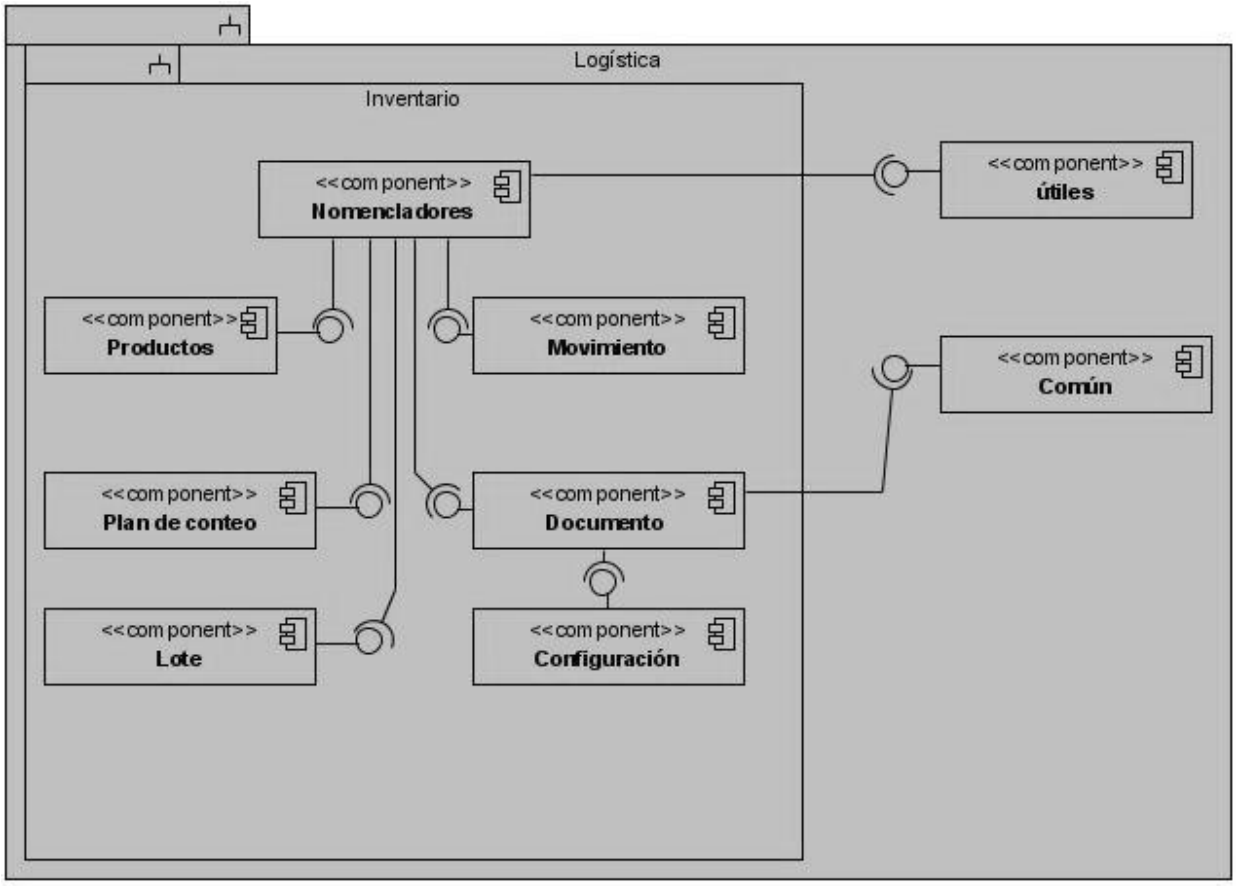

Figura 9. Diagrama de componente del proceso Nomencladores.

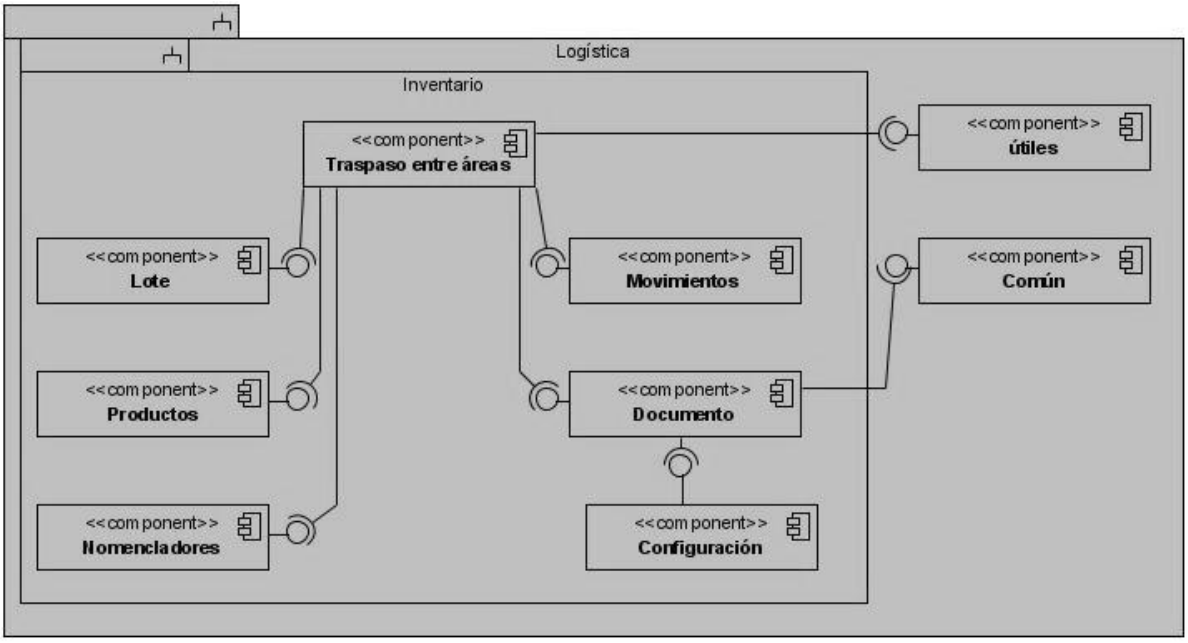

Figura 10. Diagrama de componente del proceso traspaso entre áreas.

3.1.1 Matriz de Integración de componentes interna

	Componentes interno							
Componentes Interos	utilesLogistica	utilesMunicipiones	nomencladoresInventarios	productos	configuracioninventario	ducumentos	movimientos	planconteo
nomencladores	obtenerRuso()	obtenerExisteNom()	obtenerDatMarca(), obtenerElementos(), obtenerComprobarutilizacionElem(), CargarGrupos(), obtenerLineaProducto(), obtenerFormaFarmaceutica(), obtenerCantElem(), obtenerCfgAlmacen().	ProductoConCalidad(), ProductoConDestino(), productosEngrupo(), obtenerProductoDadoIdprod().	obtenerCfgAlmacen().	CantiDconcepto(), Exitedestinofen Baja()	ExisteCalidad().	ExisteGrupoDadoId().

Tabla 2 Matriz de Integración de componentes interna del componente nomenclador.

	Componentes Internos								
Componentes Internos	utilesLogistica	documentos	nroserie	lote	configuracionInventario	utilesMuniciones	movimientos	producto	nomencladoresInventarios
apertura	mostrarCuentas() UbicarProducto() devolverUbicaciones() confirmarInv() verifRecepConfirmar() obtenerUMedidad()	creadoPor() AprobadoPorDocInventario desbloqueardocByIdusuario mostrarNomProductos() AdicionarDocInventario() EncabezadoAdicionarApertura() EncabezadoModInventario() Setalias(), ModificarApertura(), EliminarDocInventario(), EliminarDocInventario(), ,DevolverCronologia(), actualizarcantdisp(), obtenerProducto(), NoAprobarDocInventario(), GetDocumento(), CancelarEstadoDoc(), CambiarUsuario(), BuscarDocFac(),	mostrarNroSeries() GetEquipo(),	MostrarLote()	obtenerCfgAlmacen(), obtenerCfgAlmacenEnt(), ObtenerDocInventario()	obtenerClasifPart(), obtenerGrafEst() creadoPor(), AprobadoPorDocInventario() DesbloqueardocByIdusuario(), mostrarNomProductos()	ObtenerDatos_Pl() BuscarMov() actualizarcronologiamov() BuscarMov() Buscar_ME() BuscarXMovimiento() BuscarProdInventario_Pl(),UpdateEnMovimiento_Pl(),UpdateApertura_Pl() Buscar_Pl(), Existenomprod() Insertar_Pl(), Buscarmovdadoprod(),	Eliminar Producto() contabilizarProd(), obtener prodID() obtener Producto() ExisteProducto() setear(), buscarprodocumento() ,obtener Producto(), AdicionarProducto()	Cargar Categorias()

Tabla 3 Matriz de Integración de componentes interna del componente apertura.

	Componentes Internos							
	nomencladoresInventarios	documentos	lote	productos	configuracionInventario	movimientos	ajusteinventario	utilesLogistica
inventario	devolverGrupo(),	DatosEncabezadoInventario(), EncabezadoModInventario(), AprobadoPorDocInventario(), creadoPor(), BuscarDocInventario(), ObtenerDocInventario(), AdicionarDocInventario(), ActualizarDocInventario(), Setalias(), EliminarDocInventario(), CambiarUsuario(), CancelarEstadoDoc(), ConfirmarDocInventario(), GetDocumento(), AprobarDocInventario(), NoAprobarDocInventario(),	CargarLotesDelProducto(), obtenerLotesParaInventarioDadoId(),	mostrarProductos(), obtenerProducto(), buscarprodocumento(), obtenerprodID(),	obtenerCfgAlmacen(),	ObtenerDatos_Pl(), Cantidad_Pl(), BuscarMov(), Buscar_Pl(), Insertar_Pl(), Actualizar_Pl(), modificarExistenciamov(), ActualizarSinDiferencias_Pl(), BuscarProdInvSinDif_Pl(), BuscarProdInventario_Pl(), Eliminar_P(), Modificar_Pl(), actualizarcronologiamov(),	CrearAjusteInventario(),	devolverUbicaciones(),

Tabla 4 Matriz de Integración de componentes interna del componente inventario.

	Componentes Internos			
Componentes Internos	**Documentos**	**Nomencladores Inventario**	**Estructura de ubicación**	**Utiles de Logística**
Configuración	**obtenerTodosDocumentos() DevolverDatosDocsBloq(),**	**verificarExistenciaNomProd()**	**gestestructubicacion()**	**obtenerUbicAlmDadoIdEstUbic() , obtenerUbicacion(),**

Tabla 5 Matriz de Integración de componentes interna del componente configuración.

	Componentes Internos					
Componentes Internos	Productos	ZendExt_IoC_Inter	Nomencladores de Inventario	Documentos	Movimientos	Configuración de Inventario
Trasferencia entre áreas	mostrarProductos() obtenerProducto () devolverProductos (), buscarprodocumento(),	getInstance()	DameTiposTransferencias(), DameAreasXTipoTransf(), obtenerTipoAreaDadoId(), DameDatosTipoTransferencia(),	BuscarDocTransfAreas(), CargarEncabezDocTransfareas() CargarEncabezDocTransfareasModif() AdicionarDocTransfAreas(), ModificarDocTransferencia(), ConfirmarDocTransfAreas() AprobarDocTransfAreas() CancelarEstadoDocTransf() EliminarDocTransfAreas()	Existencia() BuscarMov() InsertarMov() UpdateMovConfirmar(), BuscarElementoGasto() EliminarMov()	obtenerCfgAlmacen()

Tabla 6 Matriz de Integración de componentes interna del componente traspaso entre áreas.

3.1.2 Matriz de Integración de componentes externa

	Componentes externos		
Componentes Internos	parametros	logistica	Metadatos
nomencladores	BuscarParteFormato(),	obtenerUMedidad(), obtenerUmedidaDadoId(), obtenerCfgAlmacen(),	BuscarPadrePorSubordinacion(), DameEstructura (),

Tabla 7 Matriz de Integración de componentes externa del componente nomenclador.

	Componentes externos		
Componentes Internos	seguridad	logistica	Contabilidad
apertura	getUsersProfile(),	obtenerUMedidad (),	ObtenerCuentaPorId(),

Tabla 8 Matriz de Integración de componentes externa del componente apertura.

	Componentes Externos
Componentes Internos	Parámetros
Configuración	BuscarFormato(), TipoDocumentoById(),

Tabla 9 Matriz de Integración de componentes externa del componente configuración.

	Componentes externos				
Componentes Internos	parametros	logistica	metadatos	seguridad	movimientos
inventario	BuscarMonedaContable(), BuscarParteFormato(),	obtenerAtributos(), obtenerValor(),	DameEstructura() DameAreasPorId()	getUsersProfile(),	Insertar_PI(),

Tabla 10 Matriz de Integración de componentes externa del componente inventario.

	Componentes Externos		
Componentes Internos	ZendExt_IoC_Inter	Seguridad	Parámetros
Transferencia entre áreas	getInstance()	nombreUsuario()	SoloMonedaCont()

Tabla 11 Matriz de Integración de componentes externa del componente traspaso entre áreas.

3.2 Prueba de software

Son el conjunto de técnicas que permiten determinar la calidad de un producto software. Se integran dentro de las diferentes fases del ciclo de vida del software dentro de la ingeniería de software. De esta manera, se ejecuta un programa con el objetivo de descubrir los errores que presenta. La calidad de un sistema software es algo subjetivo que depende del contexto y del objeto que se pretenda conseguir. Para determinar dicho nivel de calidad se deben efectuar unas medidas o pruebas que permitan comprobar el grado de cumplimiento respecto de las especificaciones iníciales del sistema. Una prueba tiene éxito si descubre un error no detectado hasta entonces.

3.2.1 Objetivos de las pruebas

Cuando se le aplican pruebas de calidad a un software, los principales objetivos que se persiguen son los siguientes:

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

El objetivo principal es diseñar pruebas que sistemáticamente reflejen diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo. Si las pruebas se llevan a cabo con éxito se descubrirán errores en el software**,** dándole mayor fiabilidad.

3.2.2 Alcance

Se lleva a cabo la prueba y se evalúan los resultados obtenidos frente a los resultados esperados. Si se descubren datos erróneos implica que hay un error y hay que corregirlo y empieza el proceso de depuración de errores. Se basa en las estructuras de control del diseño procedimental para generar los casos de prueba que:

- Garanticen que se recorren por lo menos una vez todos los caminos independientes de cada módulo.
- Se ejecutan todas las decisiones lógicas en su parte verdadera y en su parte falsa.
- Se recorren todos los bucles.
- Se utilizan las estructuras de datos internas para garantizar su validez.
- Se invierte tiempo y esfuerzo en los detalles de control debido a que:
- Los errores suelen estar en situaciones fuera de las normales.
- A menudo, caminos que se piensa que tienen pocas posibilidades de recorrerse, son recorridos regularmente.
- Los errores tipograficos son aleatorios. Puede que no sean detectados por los procesadores de la sintaxis del lenguaje particular y estar presentes en cualquier camino lógico.

3.2.3 Pruebas de unidad

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

El objetivo de las pruebas unitarias es aislar cada parte del programa y mostrar que las partes individuales son correctas. Proporcionan un contrato escrito que el trozo de código debe satisfacer. Estas pruebas aisladas proporcionan cinco ventajas básicas:

Fomentan el cambio: facilitan que el programador cambie el código para mejorar su estructura, puesto que permiten hacer pruebas sobre los cambios y así asegurarse de que los nuevos cambios no han introducido errores.
Simplifica la integración: permiten llegar a la fase de integración con un grado alto de seguridad de que el código está funcionando correctamente.
Documenta el código: Las propias pruebas son documentación del código puesto que ahí se puede ver cómo utilizarlo.
Separación de la interfaz y la implementación: Dado que la única interacción entre los casos de prueba y las unidades bajo prueba son las interfaces de estas últimas, se puede cambiar cualquiera de los dos sin afectar al otro.
Los errores están más acotados y son más fáciles de localizar: dado que tenemos pruebas unitarias que pueden desenmascararlos.

3.2.4 Pruebas de Caja Blanca

Estas pruebas de caja blanca se realizan sobre las funciones internas de un módulo en concreto, están dirigidas a las funciones internas del programa. El objetivo de la técnica es diseñar casos de prueba para que se ejecuten, al menos una vez, todas las sentencias del programa, y todas las condiciones tanto en su vertiente verdadera como falsa. Entre las técnicas usadas se encuentran; la cobertura de caminos, pruebas sobre las expresiones lógico-aritméticas, pruebas de camino de datos, comprobación de bucles.
En la siguiente tabla se representan las pruebas de caja blanca:

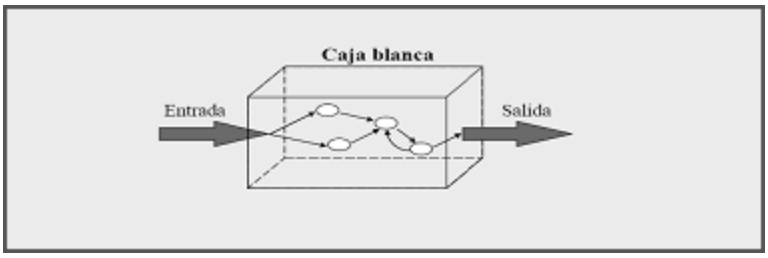

Figura 11 Representación de Pruebas de Caja Blanca.

Entre algunas de las técnicas de prueba de Caja Blanca podemos citar:
Prueba de Condición: es un método de diseño de casos de prueba que ejercita las condiciones lógicas contenidas en el módulo de un programa.

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

Prueba de Bucles: es una técnica de prueba de caja blanca que se centra exclusivamente en la validez de las construcciones de bucles.

Prueba del Camino Básico: esta técnica permite obtener una medida de la complejidad lógica de un diseño y usar esta medida como guía para la definición de un conjunto básico. 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.

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

Los pasos que se siguen para aplicar esta técnica son:

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

Los casos de prueba obtenidos del conjunto básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa.

Para aplicar la técnica del camino básico se debe introducir la notación para la representación del flujo de control, este puede representarse por un grafo de flujo en el cual:

- Cada nodo del grafo corresponde a una o más sentencias de código fuente.
- Todo segmento de código de cualquier programa se puede traducir a un grafo de flujo.
- Se calcula la complejidad ciclomática del grafo.

Para construir el grafo se debe tener en cuenta la notación para las instrucciones.

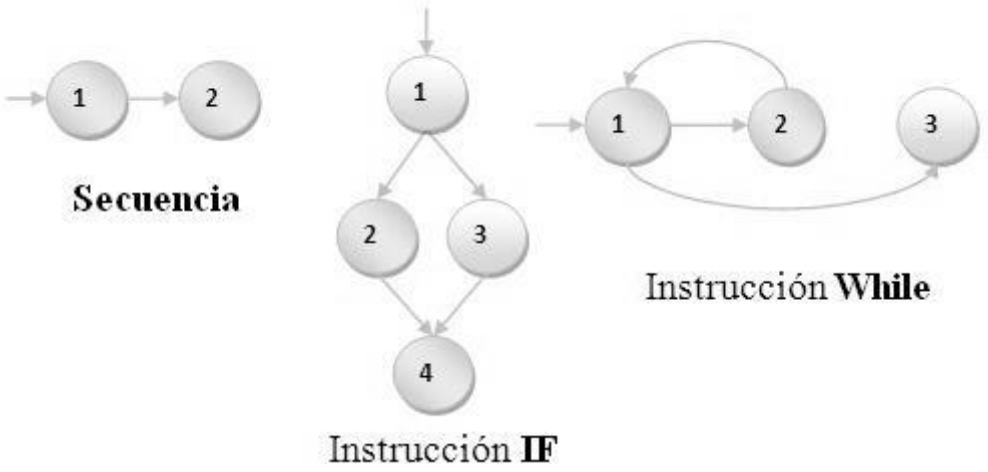

Figura 12 Notación de grafos de flujo para las instrucciones: Secuenciales, If y While.

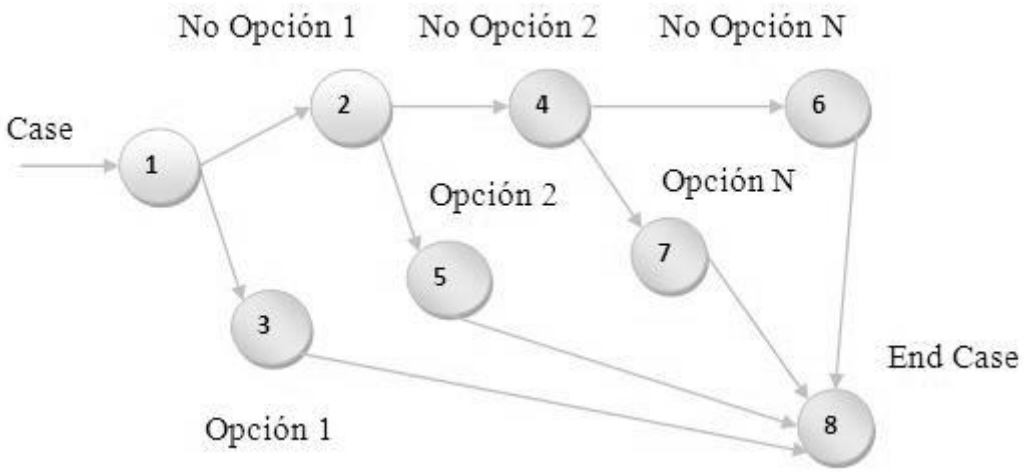

Figura 13 Notación de grafos de flujos para la instrucción Case.

Un grafo de flujo está formado por 3 componentes fundamentales que ayudan a su elaboración, su comprensión y brindan información para confirmar que el trabajo se está haciendo adecuadamente.

Componentes del grafo de flujo:

Nodo: son los círculos representados en el grafo de flujo, el cual representa una o más secuencias del procedimiento, donde un nodo corresponde a una secuencia de procesos o a una sentencia de decisión. Los nodos que no están asociados se utilizan al inicio y final del grafo.

Aristas: son constituidas por las flechas del grafo, son iguales a las representadas en un diagrama de flujo y constituyen el flujo de control del procedimiento. Las aristas terminan en un nodo, aun cuando el nodo no representa la sentencia de un procedimiento.

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

Para realizar la prueba de caja blanca específicamente la prueba del camino básico es necesario calcular antes la complejidad ciclomática del algoritmo o fragmento de código a analizar. A continuación se enumeran las sentencias de código del procedimiento realizado sobre el método cargarencabezadoadicionar ($idestructuracomun, $tipoinventario).

```
public function cargarencabezadoadicionar($idestructuracomun,$tipoinventario)
{
    $inventarioP=  $this->comun->devuelveIddocByidestructura('inventariopLog'); 1
    $inventarioG=  $this->comun->devuelveIddocByidestructura('inventariogLog'); 1
    if ($tipoinventario == 2 || $tipoinventario == 3) 2
        $iddoc = $inventarioP->iddoc; 3
    else 4
        $iddoc = $inventarioG->iddoc; 4
   $datos = $this->pIntegrator->documentos->DatosEncabezadoInventario($idestructuracomun,$iddoc); 4
   return $datos; 5
}
```

Figura 14 Representación del algoritmo cargarencabezadoadicionar($idestructuracomun,$tipoinventario).

En la siguiente figura se muestra el grafo de flujo asociado al código anterior.

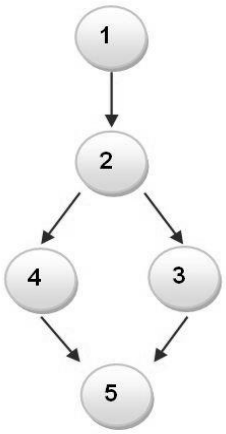

Figura 15 Grafo del algoritmo cargarencabezadoadicionar($idestructuracomun,$tipoinventario).

Cálculo de la complejidad ciclomática a partir de un segmento de código.

La complejidad ciclomática es una métrica de software extremadamente útil pues proporciona una medición cuantitativa de la complejidad lógica de un programa. El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y da un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez.

Fórmulas para calcular complejidad ciclomática:

V (G) = (A – N) + 2

V (G) = (5 - 5) + 2

V (G) = 2

Siendo “A” la cantidad total de aristas y “N” la cantidad total de nodos.

Se puede usar también:

V (G) = P + 1

V (G) = 1 + 1

V (G) = 2

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

V (G) = R

V (G) = 2

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 2, lo que significa que existen a lo sumo dos 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 por los que puede recorrer el flujo.

En la siguiente tabla se muestran los caminos básicos.

Número	Camino básico
1	1,2,3,5
2	1,2,4,5

Tabla 12 Caminos básicos del flujo.

Después de haber extraído los caminos básicos del flujo, se procede a ejecutar los casos de pruebas para este procedimiento, se debe realizar al menos un caso de prueba por cada camino básico.

Para realizar los casos de pruebas es necesario cumplir con las siguientes exigencias:

- **Descripción**: Se hace la entrada de datos necesaria, validando que ningún parámetro obligatorio pase nulo al procedimiento o no sé 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:

Camino 1: [1 – 2 – 3 – 5]

Descripción: Los datos de entrada cumplirán con los siguientes requisitos: El $idestructuracomun, $tipoinventario serán números enteros.

Condición de ejecución: el $tipoinventario = 2 o 3

Entrada: $tipoinventario = 2

Resultados esperados:

Se muestra un mensaje que el documento no se ha pudo conformado.

Caso de prueba para el camino básico 2:

Camino 2: [1 – 2 – 4 – 5]

Descripción: Los datos de entrada cumplirán con los siguientes requisitos: El $idestructuracomun, $tipoinventario serán números enteros.

Condición de ejecución: el $tipoinventario = 4.

Entrada: el $tipoinventario = 4.

Resultados esperados:

Se muestra un mensaje informando de que el documento ha sido conformado.

3.3 Pruebas de Caja Negra.

El método de prueba utilizado para comprobar la funcionalidad del sistema son las pruebas de caja negra. Dichas pruebas, también denominadas pruebas de comportamiento, se centran en los requisitos funcionales del software, o sea, la prueba de caja negra permite al ingeniero del software obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. Estas se centran en lo que se espera de un módulo, es decir, intentan encontrar casos en que el módulo no se atiene a su especificación. Por ello se denominan pruebas funcionales, y el probador se limita a suministrarle datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo el módulo por dentro.

Las pruebas de caja negra intentan encontrar errores en las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación. (26)

3.3.1 Diseño de casos de prueba.

Un caso de prueba detalla una forma de probar el sistema, incluyendo la entrada o resultados con la que se ha de probar y las condiciones bajo las que ha de probarse.

El objetivo del diseño de casos de prueba es comprobar el sistema utilizando el método de pruebas de caja negra, el cual cuenta con varias técnicas de pruebas, pero entre las más conocidas por los probadores se encuentran:

- **Partición equivalente:**

La partición equivalente es un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de forma inmediata una clase de errores que de otro modo requerían la ejecución de muchos casos antes de detectar el error genérico.

El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Una clase de equivalencia representa un conjunto de estados válidos o inválidos para condiciones de entrada. (27)

- **Análisis de valores al límite:**

Esta técnica complementa a la de partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el análisis de valores al límite lleva a la elección de casos de prueba

"en los bordes" de la clase. En lugar de centrarse solamente en las condiciones de entrada, deriva casos de prueba también para el campo de salida. (28)

3.3.2 Diseño de casos de prueba del sistema

Requisito Insertar nueva forma farmacéutica.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Insertar nueva forma farmacéutica.	Se inserta la nueva forma farmacéutica.	EP 1.1: Insertar nueva forma farmacéutica con un valor correcto del código.	- Se introduce el valor correcto del código. -Se inserta la nueva forma farmacéutica.
		EP 1.2: Insertar nueva forma farmacéutica con un valor incorrecto del código.	-Se introducen valores incorrectos del código. -No se inserta la nueva forma farmacéutica.

Tabla 13.1: Caso de prueba: Requisito: Insertar nueva forma farmacéutica.

Descripción de variables.

No	Nombre de campo	Tipo	Válido	Inválido
[1]	Código.	TextField	-Se inserta correctamente el valor del código.	-Se inserta incorrectamente el valor del código. -Se deja el campo código vacío o se introduce un valor de código ya existente.

Tabla 13.2: Descripción de las variables para el caso de prueba del requisito Insertar nueva forma **farmacéutica.**

Juego de datos.

Id del escenario	Escenario	Código	Respuesta del sistema	Resultado de la prueba
EP 1.1:	EP 1.1: Insertar nueva forma farmacéutica con un valor correcto del	V(valor)	- Se inserta la nueva forma farmacéutica.	-Se muestra en pantalla los valores de la nueva forma farmacéutica.

	código.			
EP 1.2:	EP 1.2: Insertar nueva forma farmacéutica con un valor incorrecto del código.	I(valor)	-No se inserta la nueva forma farmacéutica y se lanza un mensaje de error	-El sistema muestra un mensaje que debe insertar los valores del código correctamente en el caso de estar incorrectos y en el caso de que el campo esté vacío debe mostrar un mensaje diciendo que el campo es obligatorio.

Tabla 13.3: Juego de datos de denominación a probar para el caso de prueba del requisito Insertar nueva forma farmacéutica.

Requisito Buscar nueva forma farmacéutica.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Buscar nueva forma farmacéutica.	Busca la nueva forma farmacéutica.	EP 1.1: Buscar nueva forma farmacéutica con valores correctos del código.	-Se introduce el valor correcto del código. -Se busca nueva forma farmacéutica.
		EP 1.2: Buscar nueva forma farmacéutica con valores incorrectos del código.	-Se introducen valores incorrectos del código. -No se ejecuta la acción buscar.
		EP 1.3: Buscar nueva forma farmacéutica con valores correctos de la descripción.	-Se introduce el valor correcto de la descripción. -Se busca nueva forma farmacéutica.
		EP 1.4: Buscar nueva forma farmacéutica con valores incorrectos de la descripción.	-Se introducen valores incorrectos de la descripción. -No se ejecuta la acción

			buscar.

Tabla 14.1: Caso de prueba: Requisito: Buscar nueva forma farmacéutica.

Descripción de variables.

No	Nombre de campo	Tipo	Válido	Inválido
[1]	Código.	TextField	-Se introduce correctamente el valor del código.	-Se inserta incorrectamente el valor del código. -Se deja el campo del código vacío.
[2]	Descripción	TextField	Se introduce correctamente el valor de la descripción.	-Se inserta incorrectamente el valor de la descripción. -Se deja el campo de la descripción vacío.

Tabla 14.2: Descripción de las variables para el caso de prueba del requisito Buscar nueva forma **farmacéutica.**

Juego de datos.

Id del escenario	Escenario	Código	Respuesta del sistema	Resultado de la prueba
EP 1.1:	EP 1.1: Buscar nueva forma farmacéutica con valores correctos del código.	V(valor)	- Se busca nueva forma farmacéutica.	-Se muestra en pantalla la nueva forma farmacéutica.
EP 1.2:	EP 1.2: Buscar nueva forma farmacéutica con valores incorrectos del código.	I(valor)	-No se muestra la nueva forma farmacéutica.	-El sistema muestra un mensaje diciendo que no debe dejar campos vacíos o que el código que se desea buscar no se encuentra en el sistema.
EP 1.3:	EP 1.3: Buscar	V(valor)	-Se busca nueva	-Se muestra en pantalla la

	nueva forma farmacéutica con valores correctos de la descripción.		forma farmacéutica.	nueva forma farmacéutica.
EP 1.4:	EP 1.4: Buscar nueva forma farmacéutica con valores incorrectos de la descripción.	I(valor)	-No se muestra la nueva forma farmacéutica.	-El sistema muestra un mensaje diciendo que no debe dejar campos vacíos o que la descripción que se desea buscar no se encuentra en el sistema.

Tabla 14.3: Juego de datos de denominación a probar para el caso de prueba del requisito Buscar nueva forma farmacéutica.

Requisito Modificar nueva forma farmacéutica.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Modificar nueva forma farmacéutica.	Se modifica la nueva forma farmacéutica.	EP 1.1: Se modifica nueva forma farmacéutica con un valor correcto del código.	-Se introduce el valor correcto del código. -Se modifica la nueva forma farmacéutica.
		EP 1.2: Insertar la nueva forma farmacéutica con un valor incorrecto del código.	-Se introducen valores incorrectos del código. -No se modifica la nueva forma farmacéutica.

Tabla 15.1: Caso de prueba: Requisito: Modificar nueva forma farmacéutica.

Descripción de variables.

No	Nombre de campo	Tipo	Válido	Inválido
[1]	Código.	TextField	-Se introduce correctamente el valor del código.	-Se inserta incorrectamente el valor del código. -Se deja el campo del código vacío.

[2]	Descripción	TextField	Se introduce correctamente el valor de la descripción.	-Se inserta incorrectamente el valor de la descripción. -Se deja el campo de la descripción vacío.

Tabla 15.2: Descripción de las variables para el caso de prueba del requisito: Modificar nueva forma farmacéutica.

Juego de datos.

Id del escenario	**Escenario**	**Código**	**Respuesta del sistema**	**Resultado de la prueba**
EP 1.1:	EP 1.1: la nueva forma farmacéutica con un valor correcto del código.	V(valor)	-Se modifica la nueva forma farmacéutica.	-Se muestra en pantalla los valores de la nueva forma farmacéutica.
EP 1.2:	EP 1.2: Modificar la nueva forma farmacéutica con un valor incorrecto del código.	I(valor)	-No se modifica la nueva forma farmacéutica y se lanza un mensaje de error	-El sistema muestra un mensaje que debe insertar los valores del código correctamente en el caso de estar incorrectos y en el caso de que el campo esté vacío debe mostrar un mensaje diciendo que el campo es obligatorio.

Tabla 15.3: Juego de datos de denominación a probar para el caso de prueba del requisito Modificar nueva forma farmacéutica.

Requisito Eliminar nueva forma farmacéutica.

Nombre del requisito	**Descripción general**	**Escenarios de pruebas**	**Flujo del escenario**
1: Eliminar nueva forma farmacéutica.	Se elimina la nueva forma farmacéutica.	EP 1.1: Eliminar la nueva forma farmacéutica seleccionada.	-Se selecciona la nueva forma farmacéutica a eliminar.

			-Se confirma que realmente se desea eliminar. -Se elimina la nueva forma farmacéutica.

Tabla 16.1: Caso de prueba: Requisito: Eliminar nueva forma farmacéutica.

Nota: [Las celdas de la tabla contienen V, I, o NA. V indica válido, I indica inválido, y NA se debe proporcionar un valor del dato en este caso].

Ver casos de prueba en Anexo II.

3.4 Conclusiones parciales

En el presente capítulo se realiza las matrices de integración de componentes interna y externas a partir del diagrama de componentes. Se realiza el diseño de de los casos de pruebas y las pruebas de caja blanca y caja negra al producto desarrollado.

Conclusiones

A manera de conclusión se plantea que:

- Se analizaron las soluciones existentes detectando deficiencias en las mismas.
- Se definió la metodología de desarrollo, herramientas y tecnologías.
- Se realizó el diagrama de componentes y la matriz de interacción de componente interna y externa, además, se diseñaron los casos de pruebas de los requisitos planteados.
- Se realizó la validación de la solución propuesta mediante el diseño y aplicación de las pruebas de caja blanca para validar la calidad de la solución propuesta arrojando resultados positivos.
- Por todo lo antes mencionado se evidencia el cumplimiento de los objetivos propuesto en el presente trabajo de diploma, lo cual conlleva a un cumplimiento del objetivo general.

Recomendaciones

Las recomendaciones para el presente trabajo son:

- Ampliar las funcionalidades del módulo con nuevos requerimientos que surjan por parte del cliente.
- Llevar los nomencladores existentes a la nueva forma implementada.
- Aplicar las pruebas de calidad a la aplicación.
- Poner al alcance de los estudiantes y profesores este trabajo, como material de estudio para el posterior desarrollo del módulo de gestión de inventario Versión 2.1

Referencia Bibliográfica

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

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

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

4. **RodasXXI.cu.** RodasXXI.cu. [En línea] [Citado el: 12 de 03 de 2010.] http://www.rodasxxi.cu.

5. **España, SAP.** SAP España. [En línea] [Citado el: 12 de 03 de 2010.] http://www.sap.com/spain/solutions/business-suite/erp/index.epx.

6. **Condor.** Condor. [En línea] [Citado el: 12 de 03 de 2010.] http://www.erpsoftwaredownload.com/erpsoftware/condor-enterprise.html.

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

8. **ReDEL .** Proceso de desarrollo de software. [En línea] [Citado el: 14 de 02 de 2010.] http://www.biografica.info/redei/proceso-de-desarrollo-de-software-7.php.

9. GestioPolis.com. [En línea] [Citado el: 14 de 03 de 2010.] http://www.gestiopolis.com/resultados-busqueda.htm?cx=002900072100095058217%3Amp7ncjp0apo&cof=FORID%3A10&ie=ISO-8859-1&q=ventajas]&sa.x=0&sa.y=0&sa=Buscar#921.

10. Visual Parading for UML 6.3. [En línea] [Citado el: 15 de 03 de 2010.] http://www.visual-paradigm.com/support/vpuml/releasenotes/610.jsp.

11. **Zend.com.** .Zend Studio for Eclipse. [En línea] [Citado el: 15 de 03 de 2010.] http://www.zend.com/products/studio/.

12. **ubunto.** ubunto. [En línea] [Citado el: 16 de 03 de 2010.] http://www.ubuntu.com/.

13. TortoiseSVN The coolest Interface to (Sub Version). [En línea] [Citado el: 16 de 03 de 2010.] http://tortoisesvn.net/.

14. **Cova, Rubén Darío.** Html. [En línea] [Citado el: 17 de 03 de 2010.] http://www.geocities.com/covag/defhtml.htm.

15. **Maestrosdelweb.com.** Maestrosdelweb.com. [En línea] [Citado el: 17 de 03 de 2010.] http://www.maestrosdelweb.com/editorial/%C2%BFque-es-javascript.

16. **Webexperto.com. AJAX.** Webexperto.com. [En línea] [Citado el: 17 de 03 de 2010.] http://www.webexperto.com/articulos/articulo.php?cod=223.

17. Json.org. [En línea] [Citado el: 17 de 03 de 2010.] http://www.json.org/json-es.html.

18. maestrosdelweb.com. IntroCSS. [En línea] [Citado el: 19 de 03 de 2010.] http://www.maestrosdelweb.com/editorial/introcss/.

19. dir.PHP. [En línea] [Citado el: 19 de 03 de 2010.] http://www.dirphp.com/.

20. Netpecos.org. PostgreSql. [En línea] [Citado el: 03 de 04 de 2010.] http://www.netpecos.org/docs/mysql_postgres/x15.html.

21. Sencha. [En línea] [Citado el: 11 de 04 de 2010.] http://www.sencha.com/forum/search.php?searchid=316804.

22. Wordpress.com. [En línea] [Citado el: 16 de 04 de 2010.] http://vargasti.wordpress.com/2007/08/06/libreria-extjs/.

23. docrine. [En línea] [Citado el: 22 de 04 de 2010.] http://www.doctrine-project.org/.

24. Gerencia. [En línea] [Citado el: 26 de 04 de 2010.] http://www.emb.cl/gerencia/articulo.mv?sec=12&num=150&searchx=duoc.

25. **Oviedo, Universidad de.** *Definición de una arquitectura software para el diseño de aplicaciones web basadas en tecnologías Java JEE.*

26. Arquitectura orientada a servicio. [En línea] [Citado el: 29 de 04 de 2010.] http://www.emb.cl/gerencia/articulo.mv?sec=12&num=150&searchx=duoc.

27. Java.Lang.NullPointer. [En línea] [Citado el: 29 de 04 de 2010.] http://javalangnullpointer.wordpress.com/2007/05/21/tecnicas-de-programacion-inversion-de-control/.

28. Wampserver. [En línea] [Citado el: 29 de 04 de 2010.] http://www.wampserver.com/.

29. maestrò del web. [En línea] [Citado el: 02 de 05 de 2010.] http://www.maestrosdelweb.com/editorial/%C2%BFcomo-elegir-un-navegador-web/.

30. mozilla europe. [En línea] [Citado el: 02 de 05 de 2010.] http://www.mozilla-europe.org/es/firefox/.

31. Control de versiones con Subversion. [En línea] [Citado el: 03 de 05 de 2010.] http://svnbook.red-bean.com/index.es.html.

Glosario de términos

Algoritmo: Es un conjunto finito de instrucciones o pasos que sirven para ejecutar una tarea o resolver un problema.

Componente: El componente es la unidad de construcción elemental del diseño físico. Las características de un componente son:

Se define según como interactúa con otros.

Encapsula sus funciones y sus datos.

Es reutilizable a través de las aplicaciones.

Puede verse como una caja negra.

Puede contener otros componentes.

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

Implementación: Proceso por el cual se escribe (en un lenguaje de programación), se prueba, se depura y se mantiene el código fuente de un programa informático.

Logística: Conjunto de medios y métodos necesarios para llevar a cabo la organización de una empresa o de un servicio.

Métricas: Medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado.

Módulo: A efectos prácticos, hablar de *programas* es lo mismo que hablar de *productos de software* o hablar de *módulos de software*. Cada *módulo* es una parte del sistema, que se instala y funciona por separado, entrelazándose con otros módulos con los que intercambia información.

Objeto: Entidad provista de un conjunto de propiedades o atributos (datos) y de comportamiento o funcionalidad. Corresponden a los objetos reales del mundo que nos rodea, o a objetos internos del sistema.

Plataforma: Entorno de software común en el cual se desenvuelve la programación de un grupo definido de aplicaciones.

Software: Se refiere a los programas y datos almacenados en un ordenador.

Los programas dan instrucciones para realizar tareas al hardware o sirven de conexión con otro software.

Los datos solamente existen para su uso eventual por un programa.