Universidad de las Ciencias Informáticas

Facultad 1

Sistema de Gestión de Movimiento de Productos de la Unión de Empresas de Recuperación de Materias Primas.

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

Autores: Imer Olivera Valcárcel

Mario Claro Arcos

Tutor: Ing. Marianny Hernández Batista

Co-Tutor: Ing. Iran Roberto Rodríguez Moreno

Ciudad de La Habana, 2010

"Año 52 de la Revolución"

"La revolución es algo que se lleva en el alma, no en la boca para vivir de ella."

Ernesto Guevara de la Serna.

DECLARACIÓN DE AUTORÍA

Imer Olivera Valcárcel y Mario Claro Arcos declaramos que somos los autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Imer Olivera Valcárcel	Mario Claro Arcos
_____	_____
Firma del Autor	Firma del Autor
Ing. Marianny Hernández Batista	Ing. Iran Roberto Rodríguez Moreno
_____	_____
Firma del Tutor	Firma del Co-Tutor

AGRADECIMIENTOS

A mi familia, la cual me ha brindado todo su apoyo y amor, sin ellos no hubiera llegado a convertirme en la persona que soy hoy.
A mi novia y amiga Marisabel por todo su apoyo, cariño, comprensión y dedicación.
A la Revolución Cubana por brindarme la posibilidad de estudiar en esta maravillosa escuela.
A mi compañero de tesis por su motivación en el desarrollo del trabajo de diploma.
A los tutores por todo el apoyo y dedicación a lo largo de la elaboración de esta tesis.
A todos mis compañeros y amigos con los cuales he pasado los momentos más maravillosos de mi vida.
A todas las personas que de una forma u otra han contribuido en mi desarrollo profesional.

Imer Olivera Valcárcel.

Agradecer a las personas más especiales en mi vida; a mi mamá, mi abuela y a Rafe por el apoyo, sacrificio y confianza que tuvieron en mí.
A toda mi gran familia que me ha brindado siempre ayuda y apoyo, especialmente a mi papá, mi abuelo, mis hermanos, mis tíos y tías por la confianza depositada, a mis primos por haber compartido siempre conmigo, a todos por ser tan importantes para mí.
A mi novia Rosy primero por su cariño, también por la comprensión, dedicación y confianza que ha sabido darme y por todo lo lindo que ha sido estar juntos.
A mis amigas y amigos por haber sido siempre absolutamente los mejores.
A mi compañero de tesis por su perseverancia, motivación y paciencia en el desarrollo del trabajo de diploma.
A los tutores por brindarnos su ayuda profesional incondicionalmente.
A todas las personas que me enseñaron a transitar en el camino por la búsqueda del conocimiento.
A la Revolución Cubana por ofrecerme la oportunidad de formarme como profesional.

Mario Claro Arcos.

DEDICATORIA

Este trabajo de tesis se lo dedico a mi familia y especialmente a la persona que ha sido capaz de formarme como persona y como un hombre con convicciones sólidas y revolucionarias, dándome una buena educación e inculcando los valores más valiosos de los cuales hoy dispongo, me refiero a Carlos Manuel Valcárcel Peña mi abuelo "pipo".

Imer Olivera Valcárcel.

A toda la gente que me quiere y me apoya…

Mario Claro Arcos.

RESUMEN

Las Tecnologías de la Información y las Comunicaciones, actualmente son elementos fundamentales para la superación y desarrollo de una institución. Por eso, los países desarrollados basan su crecimiento en la aplicación y la programación estratégica de las herramientas computacionales; han definido políticas que los inducirán a su permanencia en el dinamismo mundial de los próximos años. Ante el nuevo entorno económico mundial los países emergentes están obligados a preparar profesionales en el área de la informática y las telecomunicaciones, capaces de enfrentar los retos que se tienen hoy en día. La Unión de Empresas de Recuperación de Materias Primas es la organización estatal rectora en Cuba de la recuperación, procesamiento y comercialización de materias primas en el territorio nacional. A esta organización se le hace cada día más difícil controlar la información que genera el movimiento de productos que se lleva a cabo en las diferentes áreas de trabajo; esto se debe a la poca automatización con la que cuenta. Es muy difícil que la empresa adquiera una ventaja competitiva si no se logra un uso más eficiente de la tecnología y, por supuesto, optimizar la gestión del negocio. Por tal motivo el objetivo principal del presente trabajo de diploma se centra en la implementación de un sistema que gestione adecuadamente los procesos de movimiento de productos en la empresa. El documento recoge un estudio sobre otros sistemas de gestión empresarial en el ámbito nacional e internacional, se detallan las tecnologías, lenguajes y herramientas utilizadas, la descripción de la propuesta de solución, información detallada de las clases y la validación de la solución mediante las pruebas que propone el marco de trabajo de pruebas Lime que incluye Symfony.

Palabras clave: gestión, movimiento de productos, empresa.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

El reciclaje [1] surge al tratar de preservar los recursos naturales no renovables y como una manera de aprovechar los residuos con valor. Desde hace más de dos siglos, la industria de los metales implementó sistemas de reciclaje de chatarra que se recolecta para posteriormente fundirla y convertirla en una pieza nueva. Otra es la industria del papel, cuyo reciclaje data desde principios del siglo pasado cuando los japoneses lo reutilizaban convirtiéndolo en pulpa. El reciclado del vidrio existe desde antes de Cristo, pero en el caso de los plásticos, aunque son reciclables, no se les había dado valor para este fin hasta los años 1970. (Márquez, 2007)

Antes del triunfo de la Revolución la recuperación de materias primas en la Isla se desarrollaba de manera espontánea y dispersa, y no tenía el carácter masivo. La estrategia cubana después de 1959 fue concebida y proyectada por Ernesto Guevara de la Serna siendo Ministro de Industrias y se origina, cuando se crea mediante la Resolución 21-1272 del 7 de noviembre de 1961 la Empresa Consolidada de Recuperación de Materias Primas. El Che concibió, organizó y estimuló, con su ejemplo personal, esta operación como parte de su estrategia para el desarrollo económico y social del país, mostrando su aguda visión en torno a la protección del medio ambiente. En 1981 se convierte en la Unión de Empresas de Recuperación de Materias Primas, momento a partir del cual se fortalece la actividad económica y tecnológicamente.

En la actualidad la Unión de Empresas de Recuperación de Materias Primas permanece en la rama de la siderurgia, hoy con el nombre de Ministerio de la Industria Sideromecánica, cuenta con 26 empresas clasificadas en: 15 empresas recuperadoras, una en cada provincia del país incluyendo el municipio especial Isla de la Juventud, 9 empresas especializadas y 2 exportadoras – importadoras. La renovación e incorporación de nuevas tecnologías y equipos utilizados en los procesos de recuperación son parte de la nueva estrategia del perfeccionamiento empresarial y se decide incluir las Tecnologías de la Información y las

[1] Es el proceso mediante el cual productos de desecho, son nuevamente utilizados y se produce ante la perspectiva del agotamiento de recursos naturales.

Comunicaciones (TIC[2]). Cuba ha defendido siempre el concepto de que el uso masivo de las TIC no es un fin sino una herramienta poderosa para lograr el desarrollo.
Los procesos de gestión[3] del movimiento de productos que se realizan en la Unión de Empresas de Recuperación de Materias Primas son llevados a cabo por tres áreas de trabajo, que se presentan a continuación:

- Ferroso[4]
- No ferroso
- No metálico

Estos procesos actualmente ocurren de la siguiente forma:

Ferroso

En la Unión de Empresas de Recuperación de Materias Primas, los procesos de movimiento de los productos en esta área se gestionan mediante la aplicación Concilia, desarrollada en lenguaje de bajo nivel MS-DOS[5], la cual no realiza óptimamente los procesos, incluso comete errores y mantiene una baja seguridad, implicando esto la pérdida de información. No cumple con las funcionalidades requeridas y no existe un servicio de soporte por parte del proveedor de la aplicación.

No ferroso

Los procesos de gestión del movimiento de productos en esta área se realizan mediante la aplicación SI – Estadístico 451, desarrollada en el lenguaje de bajo nivel MS–DOS. No cumple

[2] Agrupan los elementos y las técnicas utilizadas en el tratamiento y la transmisión de las informaciones, principalmente de informática, internet y telecomunicaciones.

[3] Del latín *gesĭo* y hace referencia a la acción y al efecto de gestionar o de administrar. Se trata, por lo tanto, de la concreción de diligencias conducentes al logro de un negocio o de un deseo cualquiera.

[4] Que es de hierro o que lo contiene.

[5] Siglas de *Microsoft Disk Operating System*, es un sistema operativo perteneciente a la familia DOS comercializado por Microsoft para el IBM PC.

con las funcionalidades y la seguridad mínima requerida, posee un interfaz poco agradable al usuario y no existe soporte por parte del proveedor de la aplicación.

No metálico

En esta área los procesos de movimiento de los productos se efectúan manualmente en plantillas Excel[6] y por teléfono se pide toda la información relacionada, por lo que resulta muy incómodo y extenuante para los encargados de registrar dichos procesos.

Debido al gran volumen de información que se maneja en la Unión de Empresas de Recuperación de Materias Primas, que actualmente se gestionan con herramientas propietarias y no cumplen con las funcionalidades requeridas; podemos determinar que no existe una aplicación de software que de solución al conjunto de funcionalidades técnicas y actividades dirigidas al manejo de toda la información vinculada al movimiento de productos de la empresa.

Por las insuficiencias planteadas se formula el siguiente **problema científico:** ¿Cómo mejorar la gestión del movimiento de productos de la Unión de Empresas de Recuperación de Materias Primas?

En consecuencia el **objeto de estudio** de la presente investigación son los procesos de gestión del movimiento de productos y el **campo de acción**: los procesos de gestión del movimiento de productos de la Unión de Empresas de Recuperación de Materias Primas.

Para llevar a cabo este trabajo se plantea como **objetivo general:** realizar la implementación del Sistema de Gestión de Movimiento de Productos de la Unión de Empresas de Recuperación de Materias Primas. Del objetivo general se derivan los siguientes **objetivos específicos:**

- Realizar la fundamentación teórica.
- Realizar un estudio crítico del análisis y diseño propuesto.

[6] Aplicación para manejar hojas de cálculo.

- Obtener un sistema que gestione el movimiento de productos de la Unión de Empresas de Recuperación de Materias Primas.
- Validar la propuesta de solución.

Conociendo lo anteriormente reflejado se plantea la siguiente **hipótesis:**[7] la implementación del Sistema de Gestión de Movimiento de Productos de la Unión de Empresas de Recuperación de Materias Primas contribuirá a la mejora de la gestión del movimiento de productos.

Variable independiente

- Sistema de Gestión de Movimiento de Productos de la Unión de Empresas de Recuperación de Materias Primas.

Variable dependiente

- Proceso de Gestión de Movimiento de Productos de la Unión de Empresas de Recuperación de Materias Primas.

Para darle cumplimiento a los objetivos trazados se decide desarrollar las siguientes **tareas de la investigación científica:**

- Definir el diseño teórico metodológico de la investigación.
- Estudiar el estado del arte.
- Estudiar las herramientas propuestas por los analistas.
- Estudiar el análisis y diseño propuesto por los analistas.
- Evaluar el análisis y diseño propuesto por los analistas.
- Construir los artefactos correspondientes.
- Implementar el sistema.
- Definir la estrategia de pruebas a seguir.
- Diseñar los casos de prueba.

[7] Proposición aceptable (o conjunto de proposiciones) que ha sido formulada a través de la recolección de información y datos.

- Realizar las pruebas al sistema.
- Evaluar el resultado de las pruebas.

Se utilizaron como **métodos de investigación científica:**

Métodos teóricos:

Analítico – sintético: Permite la separación de un todo en sus partes o en sus elementos constitutivos. Se apoya en que para conocer un fenómeno es necesario descomponerlo en sus partes y luego sintetizar, esto no es más que la unión de los elementos para formar un todo.

Análisis histórico – lógico: Se empleó para el estudio y profundización de la evolución de las aplicaciones lo que facilitó la indagación de soluciones al problema planteado.

Modelación: Se utilizó durante la elaboración del Sistema de Movimiento de Productos de la Unión de Empresas de Recuperación de Materias Primas porque se hizo necesario explicarle al cliente mediante modelos, cómo darle solución al problema para saber si cumple con sus necesidades.

Métodos empíricos:

Observación: Permitió conocer directamente los detalles del funcionamiento de las aplicaciones existentes y de los procesos que se desarrollan en las áreas donde se gestionan el movimiento de productos, así como las operaciones efectuadas por parte de los usuarios de las aplicaciones.

Estudio documental: Se empleó en la consulta de documentos, entendiéndose este término, en sentido amplio, como todo material de índole permanente, es decir, al que se puede acudir como fuente o referencia en cualquier momento o lugar, sin que se altere su naturaleza o sentido, para que aporte información o rinda cuentas de una realidad o acontecimiento a nuestro trabajo.

Entrevista: Se realizó a directivos para obtener información relacionada con los procesos, lo que contribuyó al perfeccionamiento de la propuesta realizada.

Posibles resultados: Un sistema que gestione el movimiento de productos de la Unión de Empresas de Recuperación de Materias Primas.

El trabajo de diploma está estructurado de la siguiente forma:

Capítulo 1 - Fundamentación Teórica: Abarca los principales conceptos manipulados en el transcurso de la investigación y una breve referencia al estado del arte de las herramientas utilizadas en el mundo para dar solución a problemas similares.

Capítulo 2 - Descripción y Análisis de la Solución Propuesta: Se orienta a la implementación del sistema; así como componentes o módulos[8] que puedan ser rehusados. Además de las descripciones de clases y algoritmos complejos desarrollados o utilizados en la investigación.

Capítulo 3 - Validación de la Solución Propuesta: Se realizan los diseños de las pruebas unitarias y funcionales que permiten validar la solución propuesta, se describen los valores utilizados para las pruebas y finalmente se realiza la evaluación de la ejecución de las mismas y de los resultados obtenidos.

Además de conclusiones, recomendaciones, referencias bibliográficas, bibliografía y anexos.

[8] Componente autocontrolado de un sistema, dicho componente posee una interfaz bien definida hacia otros componentes.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se abordan los principales conceptos que son utilizados en el transcurso de la investigación. Se realiza el análisis sobre la actualidad del problema y de soluciones existentes que entran en el entorno del tema a investigar; así como un estudio que incluye varias de las tecnologías actuales para el desarrollo de software[9], las principales herramientas y las metodologías[10] que serán usadas en la implementación del sistema que se va a desarrollar.

1.2 ¿Qué es un software de gestión?

Las aplicaciones o software de gestión son aquellas diseñadas para sustituir uno o varios procedimientos, tanto comerciales como administrativos, que habitualmente realiza una persona o institución de forma presencial, permite realizar al cliente los mismos procedimientos de forma no presencial y disminuir el esfuerzo empleado para los mismos. El proceso de información comercial constituye la mayor de las áreas de aplicación del software de gestión.

[9] Equipamiento lógico o soporte lógico de una computadora digital, y comprende el conjunto de los componentes necesarios para hacer posible la realización de tareas específicas.

[10] Del griego *metà* "más allá", *odòs* "camino" y *logos* "estudio", hace referencia al conjunto de procedimientos basados en principios lógicos, utilizados para alcanzar una gama de objetivos que rigen en una investigación científica.

1.3 Sistemas de gestión empresarial

Los sistemas de gestión empresarial son las herramientas que permiten a una compañía automatizar e integrar la mayor parte de los procesos de su negocio, compartir datos, producir y acceder a la información en tiempo real. Uno de los elementos clave para una organización es la mejora del flujo y procesamiento de la información y que el acceso a ésta se realice de manera rápida e interrelacionada.

Algunas de las ventajas que ofrece la implementación de un sistema de gestión empresarial son:

- Integración de la información y mejora de la comunicación entre diferentes áreas.
- Información disponible e inmediata para la toma de decisiones.
- Mejoras en la productividad y en la gestión de órdenes de compra.
- Reducción de los costos por compras, transporte, logística y mantenimiento.
- Mejora en los tiempos de respuesta.
- Rápida adaptación a los cambios.
- Escalabilidad[11] del sistema.
- Seguridad definida por el usuario para el manejo de información.
- Reducción de inventarios.

[11] Propiedad deseable de un sistema, una red o un proceso, que indica su habilidad para extender el margen de operaciones sin perder calidad.

1.4 Sistemas de control de almacenes y ventas.

1.4.1 Sistemas internacionales para el control de almacenes y ventas.

Internacionalmente, muchas empresas, han adquirido un auge considerable dentro del mercado empresarial pues las compañías están obligadas a ser más competitivas, por lo que se hace necesario la optimización de sus flujos internos de información y sus relaciones comerciales externas. Aprovechando las tecnologías, dichos flujos se han ido informatizando, y hoy en día se pueden acceder a través de internet. Numerosas empresas tienen sistemas de control de almacenes y de ventas, facilitando el manejo de toda la información relacionada. Estos sistemas tienen características en común como son:

- Se adaptan a diversos tipos de establecimientos.
- Las gestiones se almacenan en la base de datos del sistema.
- Facturación de las salidas.

En internet se encuentran varios ejemplos de estos tipos de aplicaciones, entre las cuales se encuentran:

Trazabilidad 3000:

Creado por Cea Ordenadores, una empresa creada en 1984, que se dedica a la creación de software en español. Dicho software facilita la gran tarea de controlar un almacén (las entradas y salidas de productos, la facturación y recibos, los datos de clientes, pedidos, proveedores, transportistas e inventarios).
El programa gestiona toda esta información de manera eficaz, asociando cada artículo con los datos con los que está relacionado. De esta manera, resulta más fácil realizar el seguimiento de cualquier producto.
Trazabilidad 3000 dispone de una base de datos en la que ordena la información a la que se accede fácilmente a través de la interfaz del programa. Ésta se encuentra organizada por

secciones: clientes, proveedores, artículos, lotes, *stock*[12], facturación, transferencias entre almacenes, inventario.

Es un pequeño, sencillo, práctico y funcional programa diseñado para llevar un adecuado control del inventario. Incorpora funciones para el registro de productos, ventas y compras, impresión de diversos reportes y varias otras de utilidad como la posibilidad de realizar copias de seguridad, y personalizar algunos aspectos operativos.

Gestión 3000:

Creado también por la empresa Cea Ordenadores. Gestión 3000 es una herramienta de gestión de facturación que abarca los campos de: compras, ventas, control de almacén y contabilidad. El programa cuenta con diversos módulos por separado que permiten controlar todos los aspectos de la facturación y contabilidad del negocio:

- Facturas.
- Gestión de almacén.
- Control de clientes.
- Proveedores.
- Control de ventas.

El módulo de ventas incluye el ciclo completo de ventas, desde la creación de un presupuesto hasta el proceso automático de facturación. Los cuatro documentos básicos de ventas son:

- Presupuesto de venta.
- Pedido de venta.
- Albarán [13]de venta.

[12] Existencia o reserva de algo disponible para un uso futuro.

[13] Palabra de origen árabe *AL BORHAN*, es un documento mercantil que acredita la entrega de un pedido.

- Factura de venta.

El ciclo está perfectamente integrado y es bidireccional, es decir, permite partiendo de un presupuesto generar múltiples pedidos de venta, expedir cada pedido con múltiples albaranes de venta y finalmente generar una factura o varias que incluyan los múltiples albaranes de venta que un cliente pueda tener, agrupándolos por obra o su número de pedido.
El módulo de gestión de almacén permite la gestión de los ficheros relacionados con el control de un almacén. Buscando la sencillez y facilidad de personalización este módulo incluye la siguiente funcionalidad:

Ficha de almacén: Cada artículo dispone de una rejilla donde se pueden ver todo el movimiento de compras, ventas, regularización e inventario.

Existencia: Se gestiona por cada artículo su existencia, cantidad pendiente de servir, cantidad pendiente de recibir y cantidad disponible.

Inventario valorado permanente: Cada movimiento de almacén guarda la existencia final que resulta del movimiento anterior y de la entrada o salida del movimiento.

Inventario según cantidad contada: Este sistema permite que cuando se graba un movimiento de inventario, se fija para esa fecha y hora la cantidad contada, de tal forma que la cantidad del movimiento se calcula de forma dinámica en función de la existencia del movimiento anterior y la cantidad contada.

Los sistemas antes mencionados pueden ser visitados a través de estas direcciones:

- Trazabilidad 3000: http://www.ceaordenadores.com
- Secop PRO: http://www.sistemaspaez.com
- Gestión 3000: http://www.ceaordenadores.com

Al visitar las aplicaciones se ha observado que estos sistemas son altamente funcionales a la hora de cumplir sus objetivos específicos, pero no sería conveniente utilizarlos en la Unión de Empresas de Recuperación de Materias Primas porque poseen funcionalidades que no se

necesitan y están desarrollados con tecnologías privativas con un costo muy elevado para la adquisición de las licencias.

1.4.2 Sistemas de control de almacenes y ventas en Cuba.

En Cuba también se encuentran sistemas que controlan los medios en los almacenes y las ventas realizadas en la empresa, de ellos se pueden citar:

Rodas XXI:

Sistema multiempresa y multiusuario creado por Tecnologías de la Información y Servicios Telemáticos (CITMATEL) para la automatización de la gestión empresarial. Contiene diferentes módulos que pueden usarse integrados o independientes:

- Contabilidad.
- Efectivo caja y banco.
- Nóminas.
- Activos fijos tangibles.
- Inventarios.
- Ventas.
- Facturación.
- Finanzas.

El módulo de inventario permite tener un control detallado de los inventarios de la 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. Es posible trabajar con varios almacenes y a su vez cada uno de ellos de forma independiente. En este módulo se hacen los cierres diarios cada vez que se realizan movimiento. En las opciones de informes es posible ver el cuadre diario y una variedad de informes de utilidad como el submayor de inventario, documentos emitidos tanto

de entrada como de salida, la existencia y el movimiento de un producto o un grupo de productos, el importe por proveedores así como la existencia por cuentas, resumida o detallada. (RodasXXI, 2007)

En el módulo de ventas, al generar el listado, se puede enviar directamente por fax o correo electrónico. Los documentos se pueden imprimir en formato de pedido o factura pro forma. Consulta de cartera de pedidos y envío de mercancías**.** La previsión de entrega de mercancías se realiza desde los pedidos. Permite entregas parciales y completas, individualizándolas o agrupando distintos pedidos en el mismo albarán de entrega. Si se desea que el documento que acompañe a la mercancía sea la factura, y no el albarán, se genera una factura directa, que puede ser bloqueada opcionalmente y se contabiliza sin pasar por albarán.

Siscont5:

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

- Efectivo en caja y bancos.
- Inventarios.
- Ventas.
- Facturación.
- Activos fijos tangibles[14].
- Nóminas.
- Contabilidad.

[14] Que puede tocarse.

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

El módulo de inventario maneja toda la información referida al submayor de inventarios de la entidad, garantizando el cuadre permanente con las respectivas cuentas de la contabilidad general. El sistema está preparado para controlar el saldo de cada material en dos monedas, a partir de los procedimientos vigentes en el país en cuanto a política monetaria. Permite el tratamiento de las contabilizaciones de forma transaccional, por resúmenes diarios o de forma personalizada según se defina por parámetros. (SISCONT5, 2007)

SABIC[15]

Es un sistema diseñado y desarrollado por la Dirección de Sistemas Automatizados del Banco Central de Cuba para satisfacer las necesidades de procesamiento de datos de bancos e instituciones no bancarias, utilizando los medios técnicos de computación disponibles en el mercado. Este sistema ha sido adaptado a los requerimientos de las operaciones propias del Banco Central y ha sido desarrollado para que los empleados que hagan uso de él puedan tramitar sus operaciones y realizar sus consultas sin necesidad de acudir a los archivos ni a la actividad manual. De esta forma, se aumenta la seguridad, la eficiencia del trabajo y la productividad de los trabajadores.

Entre sus principales características, está la contabilización en tiempo real (permite mantener actualizados los ficheros contables) y la contabilización multimoneda (permite registrar los activos y pasivos en las monedas orígenes sin tener que realizar en el momento del registro las conversiones de monedas, lo cual aumenta la exactitud de la información sobre la posición financiera de la institución). Además, las operaciones contables se pueden realizar a través de transacciones tipificadas que generan los asientos contables de forma automática. Teniendo en cuenta lo dinámico que resulta la actividad bancaria, este sistema está concebido modularmente. El módulo central es uniforme para todas las aplicaciones, se encarga del control de accesos, actualización de ficheros y los procesos de inicio y cierre del día contable.

[15] Sistema automatizado para la Banca Internacional de Comercio.

Los módulos restantes son adaptados a los requerimientos de la actividad específica de cada institución, siendo el módulo de transacciones el que se encarga de dar una entrada uniforme y coherente de los datos de la contabilidad en el sistema y es el que identifica las operaciones de cada entidad. Esta modularidad tiene la intención de facilitar la adaptabilidad y el mantenimiento del sistema para garantizar su evolución; o sea, se puede modificar cualquier módulo sin que los demás se vean afectados. (Tamargo)

A pesar de que en algunas empresas cubanas se utilizan varios software nacionales y extranjeros para el control y gestión de los recursos materiales y que realizan los procesos de gestión de inventario y control de ventas, se detecta que su desempeño no es el mejor; esto se debe a que:

- No se ajustan a las normativas de la Unión de Empresas de Recuperación de Materias Primas.
- Han sido creados para resolver los problemas de determinadas entidades.
- Se caracterizan por abordar solamente parte del problema de la gestión de la empresa o la unidad presupuestada.

1.4.3 Sistemas de control de almacenes y ventas en la UCI.

En la UCI, se cuenta con un sistema de gestión integral llamado Asset NS.

Asset NS:

Es un sistema integral modular concebido para el control de la actividad económica empresarial. Permite realizar, controlar y contabilizar todas las transacciones relacionadas con el proceso de compra – venta de productos y servicios, los cobros, pagos y anticipos asociados a los mismos, recursos humanos y nóminas, los activos fijos, útiles y herramientas de su entidad. Es un sistema que facilita el uso de la parametrización para adaptarse a las exigencias de cada cliente en particular, en la emisión de varios reportes que tendrán la forma y el contenido que el usuario les defina.

Posibilita el control del inventario de múltiples almacenes y la generación automática de comprobantes de operaciones, ofrece un control estricto de las existencias, reservas y disponibilidad de productos, así como de las cuentas por cobrar y pagar debidas a la facturación y recepción de productos.

Es un sistema flexible, amigable, con ayuda en línea, posee pantallas de entradas de datos con opciones fáciles de interpretar y ejecutar. El sistema ha sido ideado para facilitar el procesamiento rápido de la información, mostrando los resultados por pantalla e imprimiendo las páginas que el usuario decida. Su arquitectura cliente-servidor fue seleccionada para su diseño por la robustez, flexibilidad, seguridad y la rapidez de esta tecnología por la ejecución de varias tareas simultáneamente, así como poder atender un elevado número de usuarios a la vez, empleando el tiempo libre del servidor en ejecutar otras tareas automáticamente, como por ejemplo, hacer copias de seguridad de sus datos. El sistema comprende los siguientes módulos:

- Compras.
- Ventas.
- Devoluciones.
- Control de inventario.
- Cobros y pagos.
- Anticipos.
- Activos fijos.
- Útiles y herramientas.
- Recursos humanos y nóminas.
- Contabilidad.
- Auditorías contables.
- Comunicaciones.

Realizado el análisis de los sistemas automatizados se puede concluir que los mismos poseen un gran número de características que le permiten solucionar eficientemente un amplio espectro de problemas, pero están elaborados con herramientas y tecnologías propietarias lo

que ofrece una desventaja, teniendo presente que el objetivo del país es migrar a software libre para evitar los grandes costos de los software propietarios. Al analizar estos sistemas se decide por parte del equipo de desarrollo tomar algunos aspectos positivos que ayuden a la confección de la propuesta de solución:

- Los procesos de registrar las ventas y productos incluyen todas las características de estos, que constituye una manera de normalizar los términos que serán utilizados y brindan la posibilidad de actualizar estos datos ya sea creándolos, modificándolos o eliminándolos.
- Los procesos de ventas e inventarios permiten obtener información rápida y oportuna que facilita la toma de decisiones y el intercambio de información.

1.5 Tendencias y tecnologías actuales

Para la realización del trabajo, se ha realizado una investigación sobre las tendencias y las tecnologías más usadas actualmente a lo largo del mundo, que pudiesen ser útiles. Con este previo estudio se podrá comenzar un trabajo con el cual aprovechar al máximo la optimización del sistema, navegar lo más rápido posible o hacerle mucho más fácil la navegación al cliente, siendo necesarios como requerimientos mínimos el estar conectado a una red y estar trabajando en una computadora perteneciente a la misma.

1.6 Aplicaciones web.

Es una aplicación informática distribuida cuya interfaz de usuario es accesible desde un cliente web, normalmente, un navegador web. Una de las tendencias que más llama la atención son las aplicaciones web que utilizan un servicio proveniente de internet *World Wide Web*[16]. Viendo la preferencia perseguida por la humanidad, se ha trazado como tarea fundamental el utilizar y observar que internet está poblado de grandes aplicaciones web para ejercer

[16] Red Global Mundial o Red de Amplitud Mundial, es un sistema de documentos de hipertexto enlazados y accesibles a través de internet.

negocios electrónicos, ya sea con bienes lucrativos o no; además para el uso y empleo de la información, la recreación y bienes de interés.

Al observar que los sistemas más empleados para el manejo de información son las aplicaciones web, se decidió entonces para la realización del sistema la elaboración de la misma en una aplicación web. Esta favorecería con creces la cantidad de usuarios que se podrían conectar a través de los servicios web. A continuación se presentan algunas de sus ventajas:

- Ahorra tiempo: Se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.
- No hay problemas de compatibilidad: Basta tener un navegador mínimamente actualizado para poder utilizarlas.
- Actualizaciones inmediatas: Como el software lo gestiona el propio desarrollador, cuando nos conectamos estamos usando siempre la última versión que haya lanzado.
- Consumo de recursos bajo: Dado que toda (o gran parte) de la aplicación no se encuentra en nuestro ordenador, muchas de las tareas que realiza el software no consumen recursos nuestros porque se realizan desde otro ordenador.
- Multiplataforma: Se pueden usar desde cualquier sistema operativo porque sólo es necesario tener un navegador.
- Portables: Es independiente del ordenador donde se utilice porque se accede a través de una página web (sólo es necesario disponer de acceso a internet).
- La disponibilidad: Suele ser alta porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo. Los virus no dañan los datos porque éstos están guardados en el servidor de la aplicación.
- Colaboración: Gracias a que el acceso al servicio se realiza desde una única ubicación es sencillo el acceso y compartición de datos por parte de varios usuarios.

- Los navegadores: Ofrecen cada vez más y mejores funcionalidades para crear Aplicaciones de Internet Enriquecidas (RIA[17]).

1.7 Arquitectura cliente - servidor.

1.7.1 ¿Qué es una arquitectura?

Una arquitectura es un entramado de componentes funcionales que, aprovechando diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios informáticos. Se debe señalar que para seleccionar el modelo de una arquitectura, hay que partir del contexto tecnológico y organizativo del momento y, que la arquitectura cliente - servidor requiere una determinada especialización de cada uno de los diferentes componentes que la integran.

1.7.2 ¿Qué es un cliente?

Es el que inicia un requerimiento de servicio. El requerimiento inicial puede convertirse en múltiples requerimientos de trabajo a través de redes LAN[18] o WAN[19]. La ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente.

1.7.3 ¿Qué es un servidor?

Es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LAN o WAN, para proveer

[17] Acrónimo de *Rich Internet Applications*, tienen la mayoría de las características de las aplicaciones tradicionales.

[18] Del inglés *Local Area Network*, es la interconexión de varias computadoras y periféricos. Su extensión está limitada físicamente a un edificio o a un entorno de 200 metros, con repetidores podría llegar a la distancia de un kilómetro.

[19] Es una red punto a punto, es decir, red de paquete conmutado. Pueden usar sistemas de comunicación vía satélite o de radio.

de múltiples servicios a los clientes y ciudadanos tales como impresión, acceso a bases de datos, fax[20] y procesamiento de imágenes.

Ventajas de la arquitectura cliente-servidor:

- El servidor no necesita tanta potencia de procesamiento, parte del proceso se reparte con los clientes.
- Se reduce el tráfico de red considerablemente.

1.7.4 Técnicas y tecnologías del lado del cliente.

Son aquellas que se utilizan para desarrollar programas que se ejecutarán en los mismos, entiéndase por cliente a los navegadores: Firefox, Internet Explorer, Opera, Safari entre otros. Estos son los encargados de visualizar toda la información solicitada por los usuarios, el navegador es capaz de leer dicha información, ya que este interpreta código HTML y algunos lenguajes *script*[21] como son VBScript [22]y JavaScript.

1.7.4.1 CSS

Es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar el contenido y su presentación; y es imprescindible para crear páginas web complejas.
Dicho lenguaje se utiliza para definir el aspecto de todo el contenido, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista. (Pérez, 2008)

Ventajas:

- Obliga a crear documentos HTML/XHTML bien definidos y con significado completo.

[20] Sistema que permite transmitir a distancia por la línea telefónica, escritos o gráficos.

[21] Conjunto de instrucciones. Permiten la automatización de tareas, creando pequeñas utilidades.

[22] Abreviatura de *Visual Basic Script Edition*, es un lenguaje interpretado. Su sintaxis refleja su origen como variación del lenguaje de programación Visual Basic.

- Mejora la accesibilidad del documento.
- Reduce la complejidad de su mantenimiento.
- Permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Desventajas:

- Incompatibilidad entre diferentes navegadores, por lo que si algo se ve excelente en un navegador es posible que en otro no se vea igual, o que varíe la visualización incluso en las diferentes versiones de un mismo navegador.

1.7.4.2 HTML

Es el lenguaje utilizado para representar documentos en la *World Wide Web*. Además de texto normal incluye también, elementos multimedia (gráficos, vídeo, audio) y existen enlaces que permiten acceder a otras partes del documento o a otro sitio cualquiera de internet. La descripción se basa en especificar en el texto la estructura lógica del contenido (títulos, párrafos de texto normal, enumeraciones, definiciones y citas), así como los diferentes efectos que se quieren dar (cursiva, negrita, o un gráfico determinado) y dejar que luego la presentación final de dicho hipertexto se realice por un programa especializado. (Santamaría)
Otra característica muy importante de este lenguaje es que es portable, es decir, se pueden visualizar las páginas con cualquier sistema operativo y por supuesto también crearlas.

1.7.4.3 JavaScript

JavaScript es el lenguaje *scripting* por excelencia, es decir, es un lenguaje basado en *scripts*. Posee una sintaxis similar a la del lenguaje Java y el lenguaje C, aunque no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de herencia. Está destinado al desarrollo de aplicaciones web como complemento del HTML. (Abraham, 2007)

Ventajas:

- No requiere tiempo de compilación.
- Los *scripts* pueden desarrollarse en un período de tiempo relativamente corto.
- Posee características de interfaz, que son gestionados por el navegador y por el código HTML.

- Los programas JavaScript tienden a ser pequeños y compactos, no requieren mucha memoria ni tiempo adicional de transmisión.
- Es independiente de la plataforma de hardware o sistema operativo, siempre y cuando exista un navegador con soporte JavaScript.

Desventajas:

- El número de métodos integrados es insuficiente para gestionar documentos y ventanas.
- El código del *script* debe descargarse completamente antes de poderse ejecutar.
- No es posible ocultar el código fuente.

1.7.5 Tecnologías del lado del servidor.

Son aquellos programas o servicios que corren en un servidor remoto y que brindan funcionalidades al sistema.

1.7.5.1 PHP[23]

Es un lenguaje de programación utilizado para la creación de sitios web. PHP es un acrónimo recursivo. (PHP, 2009)
PHP es un lenguaje *script* interpretado en el lado del servidor, utilizado para la generación de páginas web dinámicas y ejecutadas en un servidor. PHP no necesita ser compilado para ejecutarse y genera así una página HTML para ser mostrada al cliente. Para su funcionamiento necesita tener instalado Apache o *Internet Information Server* (IIS) con las librerías de PHP. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas.

- Soporta una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle, Microsoft SQL Server[24], Informix[25], entre otras.

[23] Inicialmente se llamó *Personal Home Page*. Surgió en 1995, desarrollado por PHP Group.

[24] Es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional.

- Ofrece una solución simple y universal para las paginaciones dinámicas de la web.
- Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
- Con PHP se puede hacer cualquier cosa que podemos realizar con un *script* de Interfaz de Entrada Común (CGI), como el procesamiento de información en formularios, foros de discusión, manipulación de *cookies* y páginas dinámicas.
- PHP corre en cualquier plataforma utilizando el mismo código fuente.
- Muchas interfaces distintas para cada tipo de servidor. PHP actualmente se puede ejecutar bajo Apache, IIS, AOLServer[26].
- Puede conectarse con muchos motores de bases de datos tales como Microsoft SQL, Oracle, Informix y PostgreSQL, destacando su conectividad con MySQL.
- Con respecto a la rapidez, PHP generalmente es utilizado como módulo de Apache, lo que lo hace extremadamente veloz.
- Es capaz de leer y manipular datos desde diversas fuentes, incluyendo datos que pueden ingresar los usuarios desde formularios HTML.
- Tiene la capacidad de expandir su potencial utilizando su enorme cantidad de módulos.
- Posee una amplia documentación en su web oficial de internet.
- Pertenece a la alternativa de código abierto.
- Permite las técnicas de programación orientada a objetos.
- Permite crear los formularios para la web.
- Cuenta con una biblioteca sumamente amplia para facilitar su funcionalidad.

[25] Fue concebido y diseñado por Roger Sippl a finales de los años 1970. La compañía Informix fue fundada en 1980 y durante parte de los años 1990 fue el segundo sistema de bases de datos más popular después de Oracle.

[26] Servidor web de código abierto.

1.8 Gestor de base de datos

Una base de datos no es más que un conjunto de información almacenada en memoria auxiliar que permite acceso directo y un conjunto de programas que manipulan esos datos. Los sistemas de gestión de bases de datos son un tipo de software que funciona como interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, un lenguaje de manipulación de datos y un lenguaje de consulta.

1.8.1 PostgreSQL 8.3

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS), es una derivación libre del proyecto Postgres, esta versión incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional, pero a pesar de esto PostgreSQL no es un sistema de gestión de bases de datos puramente orientado a objetos.
Una de sus principales características es que soporta distintos tipos de datos, además de los del tipo base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes, cadenas de bits [27] y permite la creación de tipos propios.

Ventajas:

- Posee una gran escalabilidad. Es capaz de ajustarse al número de unidades centrales de procesamiento (CPU) y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta.
- Implementa el uso de *rollback*, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz.

[27] Acrónimo de *Binary digit* (dígito binario). Un bit es un dígito del sistema de numeración binario.

- Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.
- El único costo asociado a él, es el de conocerlo pues su código fuente está disponible bajo la más liberal de las licencias del código abierto: la licencia BSD (Distribución de Software Berkeley), que permite usarlo modificarlo y distribuirlo en productos comerciales o no comerciales, sin costo alguno.
- Requiere pocos recursos de hardware y la simplificación del proceso de administración de licencias de software, que no es necesario cuando se usa software libre.
- PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.
- Puede lidiar con gran volumen de datos.

1.9 Marcos de Trabajo

En el desarrollo de software, un marco de trabajo es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un marco de trabajo puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. (Gutiérrez)

Un marco de trabajo simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un marco de trabajo proporciona estructura al código fuente, donde el desarrollador es obligado a crear código más legible y más fácil de mantener, lo cual permite la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

1.9.1 Symfony 1.4.3

Symfony es un marco de trabajo que tiene como objetivo fundamental automatizar los patrones más utilizados en la elaboración de sistemas web; además de obligar a clarificar a los programadores el código, establece un estándar de código legible, encapsula operaciones

complejas en simples líneas de código, ahorrando mucho más tiempo a la hora de mostrar datos directamente de la base de datos. (Potencier, 2008)

Este marco de trabajo está realizado o implementado bajo el lenguaje PHP 5, ha sido utilizado en varias aplicaciones web obteniéndose de esta manera resultados satisfactorios. Es compatible con la mayoría de gestores de base de datos existentes, se puede comentar sobre MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. También otorga facilidades en diferentes plataformas. (Potencier, 2008)

Symfony provee una abstracción de la base de datos, que permite una mayor facilidad de obtención de los datos haciendo a la vista y a las acciones independientes del gestor de base de datos, la vista tiene de *helpers* que facilitan el trabajo para los diseñadores en el código HTML de la aplicación, aunque es necesario crear una nueva vista, mantiene al modelo y al controlador original; se encarga de mantener alejados a la vista y al modelo de los detalles del protocolo utilizado. (Potencier, 2008)

Características particulares de Symfony:

- Escalable: Symfony es infinitamente escalable si se disponen de los recursos necesarios. Yahoo lo utiliza para programar aplicaciones con 20 millones de usuarios y 12 idiomas, además posee un centenar de complementos desarrollados por la comunidad.
- Probado y testeado: Symfony ha sido probado con éxito durante varios años en aplicaciones muy diferentes. Desde sitios web con millones de usuarios hasta otros miles de sitios pequeños y medianos, obteniendo en cada uno de ellos excelentes resultados.
- Soporte: Symfony sigue una política de tipo LTS[28]. Las versiones estables se mantienen durante 3 años sin cambios pero con una continua corrección de los errores conocidos.

[28] Brinda soporte durante más tiempo del habitual, 3 años para la versión de escritorio y 5 para la de servidores.

- Licencia: Symfony utiliza una licencia MIT, con la que puedes hacer aplicaciones web comerciales, gratuitas y/o de software libre.
- Compromiso: La empresa que ha creado Symfony (*Sensio Labs*) no genera ingresos del marco de trabajo, sino de las aplicaciones que hacen con él.
- Código: Desde su primera versión Symfony ha sido creado exclusivamente para PHP 5, desechando la versión PHP 4 (que ha sido declarada obsoleta recientemente).
- Seguro: Se puede controlar hasta el último acceso a la información e incluye por defecto protección contra ataques XSS [29]y CSRF[30].
- Documentado: Se trata del marco de trabajo PHP mejor documentado: tutoriales de hasta 250 páginas y libros gratuitos de casi 500 páginas. Además, los libros están completamente traducidos al español.
- Calidad: Su código fuente incluye más de 8.000 pruebas unitarias y funcionales.
- Internacionalización: Se pueden crear aplicaciones en varios idiomas. La internacionalización está integrada en el marco de trabajo, funciona bien, es muy completa y está probada en aplicaciones reales.

1.9.2 JQuery 1.3

JQuery es una biblioteca o marco de trabajo de JavaScript, creada inicialmente por John Resig, permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM[31], manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web.

JQuery es libre y de código abierto, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código.

[29] Tipo de inseguridad informática o agujero de seguridad basado en la explotación de vulnerabilidades del sistema de validación de HTML incrustado.

[30] Fragmento de datos malicioso de un sitio web en el que comandos no autorizados son transmitidos por un usuario en el cual el sitio web confía.

[31] Interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML y XML.

Es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

Características:

- Selección de elementos DOM.
- Eventos.
- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- AJAX.
- Soporta extensiones.
- Varias utilidades como obtener información del navegador, operar con objetos y arreglos.
- Compatible con los navegadores Firefox 2.0+, Internet Explorer 6+, Safari 2.0.2+ y Opera 9+.

1.10 Metodología para el desarrollo del software

Muchas veces el proceso de desarrollo de software resulta riesgoso y se convierte en una tarea difícil hallar el modo de controlar su curso de principio a fin. El problema principal radica en cómo coordinar todas las actividades que comprende el desarrollo de un proyecto de software, sobre todo si se trata de un proyecto de gran envergadura. De modo que se torna imprescindible contar con una forma organizada y adecuadamente estructurada para trabajar. Se necesita un proceso que integre las múltiples fases del desarrollo, un método común, un proceso que: (EVA)

- Proporcione una guía para ordenar las actividades de un equipo.
- Dirija las tareas de cada desarrollador por separado y del equipo como un todo.
- Especifique los artefactos que deben desarrollarse.
- Ofrezca criterios para el control y la medición de los productos y actividades del proyecto.

1.10.1 Rational Unified Process

Es un proceso de desarrollo de software, cuyos modelos y artefactos se expresan en el Lenguaje Unificado de Modelado. RUP es una metodología robusta que puede ser adaptado a proyectos de mayor o menor complejidad, aplicable a diferentes esferas y ajustable a las necesidades de cada organización. Se trata de un proceso iterativo e incremental debido a que se basa en la evolución de prototipos ejecutables que se muestran a los usuarios y clientes.
Se caracteriza por ser centrado en la arquitectura porque establece refinamientos sucesivos de una arquitectura ejecutable, construida como un modelo evolutivo de manera que no se afecte de forma significativa ante posibles modificaciones, para lograr finalmente una arquitectura comprensible, adaptable y robusta. Por último, RUP está dirigido por los casos de uso, pues guía el desarrollo del proyecto manteniendo como un aspecto de vital importancia la satisfacción del usuario y no sólo teniendo en cuenta las funcionalidades del sistema sino que permite controlar el proceso de desarrollo del proyecto al mismo tiempo que es elaborado, quedando conformada, a su vez, una guía para posteriores mejoras del producto.
En RUP se han agrupado las actividades en grupos lógicos definiéndose 9 flujos de trabajo principales. Los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como de apoyo.

Flujos de trabajo:

- Modelamiento del negocio: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
- Requerimientos: Define qué es lo que el sistema debe hacer, para lo cual se Identifican las funcionalidades requeridas y las restricciones que se imponen.
- Análisis y diseño: Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.

- Implementación: Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán, la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
- Prueba: Busca los defectos a lo largo del ciclo de vida.
- Instalación: Produce *release*[32] del producto y realiza actividades (empaque, instalación y asistencia a usuarios) para entregar el software a los usuarios finales.
- Administración del proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
- Administración de configuración y cambios: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, entre otros.
- Ambiente: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

Fases:

- Conceptualización (Concepción o Inicio): Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.
- Elaboración: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del mismo, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo con el alcance definido.

[32] Versión candidata a definitiva o candidata para el lanzamiento, comprende un producto final, preparado para publicarse como versión definitiva a menos que aparezcan errores que lo impidan.

- Construcción: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtiene uno o varios *release* del producto que han pasado las pruebas. Se ponen estos *release* a consideración de un subconjunto de usuarios.

- Transición: El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores. (EVA)

1.11 Lenguaje de modelado

1.11.1 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. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.
Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional o RUP), pero no especifica en sí mismo qué metodología o proceso usar.

1.12 Herramienta de modelado

1.12.1 Visual Paradigm 6.4

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

documentación. La herramienta también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. (1)

1.13 Servidor de aplicaciones web

Un servidor web sirve de contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante HTTP[33].

1.13.1 Apache 2.2.6

Es un servidor de páginas web de código abierto multiplataforma y modular, se desarrolla dentro del proyecto HTTP Server (httpd) de la *Apache Software Foundation*.
Presenta entre otras características, mensajes de errores altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración. Se ha adaptado siempre a una gran variedad de entornos a través de su diseño modular.
Este diseño permite a los administradores de sitios web elegir que características van a ser incluidas en el servidor seleccionando, que módulos se van a cargar, ya sea al compilar o al ejecutar el servidor. (Ciberaula, 2006)

Ventajas:

- Trabaja sobre múltiples plataformas.
- Incluye módulos que se cargan de forma dinámica.
- Soporta CGI, Perl, PHP.
- Soporte para bases de datos.
- Soporte SSL [34]para transacciones seguras.

[33] Protocolo de transferencia de hipertexto, es el protocolo usado en cada transacción de la *World Wide Web*.

[34] Es un protocolo criptográfico que proporciona comunicación segura por una red, comúnmente internet.

- Incluye soporte para *host* virtuales.
- Soporta HTTP 1.1.
- Código abierto.
- Rápido.
- Eficiente.

1.14 Herramientas a utilizar

1.14.1 NetBeans IDE 6.8

Es una aplicación de código abierto diseñada para el desarrollo de aplicaciones fácilmente portables entre las distintas plataformas, haciendo uso de la tecnología Java.
Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, colaboración entre varias personas, creación de aplicaciones compatibles con teléfonos móviles, resaltado de sintaxis y por si fuera poco sus funcionalidades son ampliables mediante la instalación de paquetes. Además, soporta PHP 5.3 y Symfony.

1.14.2 DBDesigner 4.0.5.6

Es un sistema totalmente visual de diseño de bases de datos, que combina características y funciones profesionales con un diseño simple, muy claro y fácil de usar, a fin de ofrecer un método efectivo para gestionar bases de datos.
Permite administrar la base de datos, diseñar tablas, hacer peticiones SQL manuales y mucho más, como ingeniería inversa en MySQL, Oracle, MSSQL y otras bases de datos ODBC, modelos XML y soporte para la función arrastrar y soltar.
El programa dispone además de una interfaz profesional y de detallados manuales de uso.
DBDesigner es un sistema de diseño visual de base de datos que integra diseño de base de datos, modelado, creación y mantenimiento en un entorno único, sin fisuras.

Combina características profesionales y una interfaz de usuario clara y sencilla para ofrecer la manera más eficiente para gestionar sus bases de datos.
DBDesigner se compara con productos similares como Oracle Designer, IBM Rational Rose, *Computer Associates's* de ERwin y DataArchitect theKompany's, pero es un proyecto de código abierto disponible para Microsoft Windows 2k/XP y Linux KDE / Gnome.

1.14.3 EMS PostgreSQL Manager 4.1.0.7

Es una poderosa herramienta gráfica para la administración y desarrollo del servidor de bases de datos PostgreSQL. PostgreSQL Manager soporta todas las nuevas características de PostgreSQL incluyendo espacios de tablas, argumentos nombrados en funciones y otras más. Ofrece una gran variedad de herramientas poderosas para usuarios avanzados, tales como el diseñador visual de base de datos, el constructor visual de consultas y un poderoso editor de objetos binarios para satisfacer todas sus necesidades. PostgreSQL Manager cuenta con una nueva y avanzada interfaz gráfica de usuario con un sistema asistente bastante descriptivo, tan claro en su uso que ni un principiante se podrá confundir.

Características principales:

- Nueva interfaz gráfica de usuario último modelo.
- Ágil navegación y administración de base de datos.
- Administración sencilla de todos los objetos PostgreSQL.
- Herramientas de manipulación de datos avanzada.
- Administración efectiva de seguridad.
- Excelentes herramientas visuales y de texto para elaboración de consultas.
- Acceso al servidor PostgreSQL a través del protocolo HTTP.
- Conexión vía reenvío de puerto local a través del túnel SSH[35].
- Impresionantes opciones de exportación e importación de datos.
- Poderoso diseñador visual de base de datos.

[35] Protocolo para acceder a máquinas remotas a través de una red.

- Asistentes fáciles de usar que realizan mantenimiento de tareas de PostgreSQL.

1.15 Conclusiones

Existen múltiples lenguajes, metodologías y tecnologías para el desarrollo de aplicaciones. El lenguaje de programación seleccionado es PHP, como gestor de base de datos PostgreSQL y servidor web Apache. Los marcos de trabajo para desarrollar la aplicación son JQuery y Symfony. Para agilizar el proceso de desarrollo, NetBeans IDE 6.8 y se decide realizar el sistema sobre la base de la metodología RUP.

CAPÍTULO 2: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA

2.1 Introducción

En el presente capítulo se aborda: la valoración crítica del diseño propuesto por el analista, los patrones de diseño utilizados, análisis de posibles implementaciones, la estructura de la aplicación así como los componentes que son reutilizados, información detallada de las clases desarrolladas, la descripción de la solución propuesta y los modelos de implementación.

2.2 Valoración crítica del diseño propuesto por el analista

Los analistas y diseñadores del sistema han realizado una propuesta de acuerdo con el desarrollo que se quiere llevar a cabo. Primero en el momento de implementar el sistema y luego a la hora de ponerlo en funcionamiento.
En el diseño propuesto se definen las clases necesarias para un correcto funcionamiento del sistema así como los atributos y métodos que deben tener las mismas, mostrando una idea clara de lo que se debe implementar. Juntamente con una serie de diagramas de interacción, que explican la relación entre las clases y como son llamados los métodos y sentencias dentro de cada una de ellas, generando así toda la información necesaria para conocer el orden de implementación de las acciones.

2.3 Patrones de diseño implementados

El desarrollo del sistema utilizando el marco de trabajo Symfony proporciona ventajas significativas para los desarrolladores de software. El marco de trabajo mencionado es capaz de fusionar buenas prácticas de trabajo por sí mismo, de forma que los desarrolladores no

tengan que preocuparse por implementar varios de los patrones arquitectónicos y de diseño más utilizados en la actualidad, ya que Symfony los implementa.

2.3.1 Patrones GRASP

Creador: Todos los módulos del sistema tienen una clase actions.class.php que contiene las acciones que hacen al sistema funcional. En esta clase las acciones se encargan de crear los objetos de las clases que representan las entidades, evidenciando de este modo que la clase actions.class.php es el "*creador*" de las entidades.

Experto: Se evidencia este patrón puesto que Doctrine es la librería externa que utiliza Symfony para realizar su capa de abstracción al modelo de datos, encapsulando toda la lógica de los datos y generando las clases con funcionalidades comunes de las entidades. Por tanto, cada clase creada por Doctrine a partir de una entidad es experta en manejar su información.

Controlador: Todas las peticiones web son manejadas por un solo controlador frontal (frontend.php), que es el punto de entrada único de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la URL entrada por el usuario.

Alta Cohesión: Symfony permite la asignación de responsabilidades con alta cohesión, por ejemplo la clase actions.class.php tiene la responsabilidad para definir las acciones sobre las plantillas y colabora con otras para realizar diferentes operaciones y crear objetos, está formada por diferentes funcionalidades que se encuentran estrechamente relacionadas, proporcionando que el software sea flexible frente a grandes cambios.

Bajo Acoplamiento: La clase actions.class.php hereda solamente de sfActions para lograr un bajo acoplamiento de clases.

2.3.2 Patrones GOF

Singlenton (Instancia Única): Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Es el caso del controlador frontal, donde hay una llamada a la función sfContext::getInstance() que garantiza que siempre se acceda a la misma instancia.

Abstrac Factory (Fábrica Abstracta): Se utiliza este patrón al trabajar con objetos de distintas familias de manera que no se mezclen entre sí, haciendo transparente el tipo de familia concreta que se esté usando. Cuando el marco de trabajo necesita, por ejemplo crear un nuevo objeto, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea.

Decorator (Decorador): Añade funcionalidad a una clase, dinámicamente. El archivo layout.php, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el *Layout* decorando la misma.

Composite (Objeto compuesto): Permite tratar objetos compuestos como si de uno simple se tratase. Sirve para construir objetos complejos a partir de otros más simples y similares entre sí, gracias a la composición recursiva y a una estructura en forma de árbol. Esto simplifica el tratamiento de los objetos creados, ya que al poseer todos ellos una interfaz común, se tratan todos de la misma manera.

2.4 Modelo Vista Controlador (MVC)

Symfony está basado en un patrón clásico del diseño web conocido como arquitectura MVC, que está formado por 3 niveles:

- El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- La vista transforma el modelo en una página web que permite al usuario interactuar con ella.

- El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista), por lo que se consigue un mantenimiento más sencillo de las aplicaciones. Por ejemplo, una misma aplicación debe ejecutarse tanto en un navegador estándar como en un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, correo electrónico). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación.

Contenido de cada capa en Symfony:

La capa del Modelo

- Abstracción de la base de datos
- Acceso a los datos

La capa de la Vista

- Vista
- Plantilla
- *Layout*

La capa del Controlador

- Controlador frontal

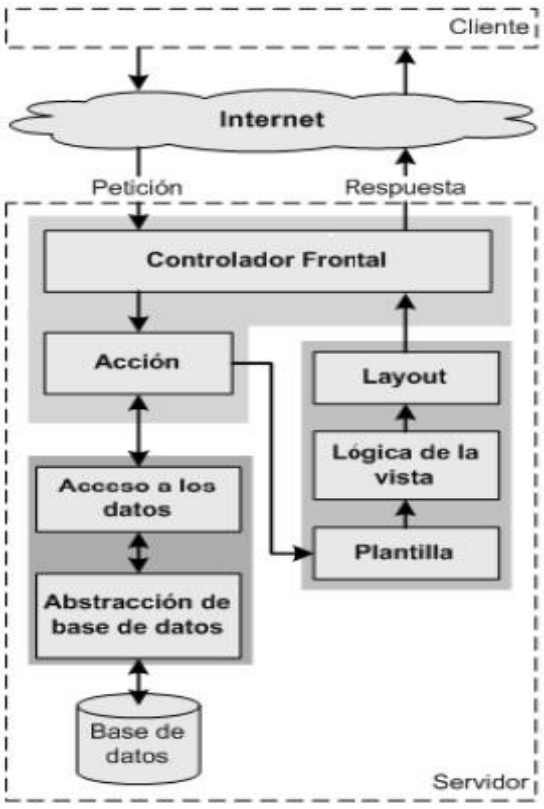

Figura 1. Flujo de trabajo de Symfony

2.5 Análisis de posibles implementaciones y componentes que son reutilizados

Los complementos permiten agrupar todo el código diseminado por diferentes archivos y reutilizar este código en otros proyectos. Además, permiten encapsular clases, filtros, *mixins*[36], ayudantes, archivos de configuración, tareas, módulos, esquemas y extensiones para el modelo y archivos estáticos.

Dentro de los complementos utilizados se cuenta con:

SfDoctrineGuardPlugin: Es un complemento de Symfony que proporciona características de autenticación y autorización por encima de la función de seguridad estándar de Symfony. Brinda el modelo (usuarios, grupos y objetos de permiso) y los módulos (sfGuardAuth, sfGuardGroup, sfGuardPermission y sfGuardUser) facilitando el control de la seguridad de la aplicación.

SfAdminDashPlugin: Brinda una interfaz de escritorio para la aplicación backend.

SfJqueryReloadedPlugin: Ayuda mucho en el manejo de JQuery en la aplicación.

2.6 Estructura de GEMPRO[37]

Symfony proporciona una estructura en forma de árbol de archivos para organizar de forma lógica todos esos contenidos, además de ser consistente con la arquitectura MVC utilizada y con la agrupación proyecto / aplicación / módulo. Cada vez que se crea un nuevo proyecto, aplicación o módulo, se genera de forma automática la parte correspondiente de esa estructura. Además, la estructura se puede personalizar completamente, para reorganizar los archivos y directorios o para cumplir con las exigencias de organización de un cliente.

2.6.1 Estructura de la raíz

La raíz de GEMPRO contiene los siguientes directorios:

[36] Clase que ofrece cierta funcionalidad para ser heredada por una subclase.

[37] Sistema de Gestión de Movimiento de Productos.

```
apps/
  frontend/
  backend/
cache/
config/
data/
  sql/
lib/
  model/
  filter/
  form/
  vendor/
log/
plugins/
test/
  bootstrap/
  unit/
  functional/
web/
  css/
  images/
  js/
  uploads/
```

La siguiente tabla describe los contenidos de estos directorios.

Directorio	Descripción
apps/	Contiene un directorio por cada aplicación de GEMPRO (frontend para la parte pública y backend para la administración).
cache/	Contiene la versión cacheada de la configuración y (si está activada) la versión cacheada de las acciones y plantillas de GEMPRO. El mecanismo de cache utiliza los archivos de este directorio para acelerar la respuesta a las peticiones web. Cada aplicación contiene un subdirectorio que

	guarda todos los archivos PHP y HTML preprocesados.
config/	Almacena la configuración general de GEMPRO.
data/	En este directorio se almacenan los archivos relacionados con los datos, como por ejemplo el esquema de una base de datos, el archivo que contiene las instrucciones SQL para crear las tablas e incluso un archivo de bases de datos de SQLite.
lib/	Almacena las clases y librerías externas. Se suele guardar todo el código común a todas las aplicaciones de GEMPRO. El subdirectorio model/ guarda el modelo de objetos de GEMPRO.
log/	Guarda todos los archivos de log [38]generados por Symfony. También se puede utilizar para guardar los logs del servidor web, de la base de datos o de cualquier otro componente de GEMPRO. Symfony crea un archivo de log por cada aplicación y por cada entorno.
plugins/	Almacena los complementos instalados en GEMPRO.
test/	Contiene las pruebas unitarias y funcionales escritas en PHP y compatibles con el marco de trabajo de pruebas de Symfony.
web/	La raíz del servidor web. Los únicos archivos accesibles desde Internet son los que se encuentran en este directorio.

Tabla 2.1: Directorios en la raíz de GEMPRO

2.6.2 Estructura de cada aplicación

Las dos aplicaciones de GEMPRO tienen la misma estructura de archivos y directorios:

```
apps/
  [ nombre de la aplicación ]/
    config/
    i18n/
    lib/
    modules/
    templates/
```

A continuación se describen los subdirectorios de las aplicaciones.

[38] Registro oficial de eventos durante un rango de tiempo en particular.

Directorio	Descripción
config/	Contiene un montón de archivos de configuración creados con YAML. Aquí se almacena la mayor parte de la configuración de la aplicación, salvo los parámetros propios del marco de trabajo. También es posible redefinir en este directorio los parámetros por defecto si es necesario.
i18n/	Contiene todos los archivos utilizados para la internacionalización de la aplicación, sobre todo los archivos que traducen la interfaz. La internacionalización también se puede realizar con una base de datos, en cuyo caso este directorio no se utilizaría.
lib/	Contiene las clases y librerías utilizadas exclusivamente por la aplicación.
modules/	Almacena los módulos que definen las características de la aplicación.
templates/	Contiene las plantillas globales de la aplicación, es decir, las que utilizan todos los módulos. Por defecto contiene un archivo llamado layout.php, que es el *layout* principal con el que se muestran las plantillas de los módulos.

Tabla 2.2: Subdirectorios de cada aplicación de GEMPRO

2.6.3 Estructura de los módulos

Cada aplicación contiene varios módulos. Cada módulo tiene su propio subdirectorio dentro del directorio modules y el nombre del directorio es el que se elige durante la creación del módulo.

Esta es la estructura de directorios típica de un módulo:

```
apps/
  [nombre de la aplicación]/
    modules/
      [nombre del módulo]/
        actions/
          actions.class.php
        config/
        lib/
        templates/
          indexSuccess.php
```

La siguiente tabla describe los subdirectorios de un módulo.

Directorio	Descripción
actions/	Contiene un único archivo llamado actions.class.php y que corresponde a la clase que almacena todas las acciones del módulo. También es posible crear un archivo diferente para cada acción del módulo.
config/	Puede contener archivos de configuración adicionales con parámetros exclusivos del módulo.
lib/	Almacena las clases y librerías utilizadas exclusivamente por el módulo.
templates/	Contiene las plantillas correspondientes a las acciones del módulo. Cuando se crea un nuevo módulo, automáticamente se crea la plantilla llamada indexSuccess.php.

Tabla 2.3: Subdirectorios de cada módulo

2.6.4 Estructura del sitio web

```
web/
  css/
  images/
  js/
  uploads/
```

Directorio	Descripción
css/	Contiene los archivos de hojas de estilo creados con CSS (archivos con extensión .css).
images/	Contiene las imágenes del sitio con formato .jpg, .png o .gif.
js/	Contiene los archivos de JavaScript con extensión .js.
uploads/	Se almacenan los archivos subidos por los usuarios. Aunque normalmente este directorio contiene imágenes, no se debe confundir con el directorio que almacena las imágenes del sitio (images/). Esta distinción permite sincronizar los servidores de desarrollo y de producción sin afectar a las imágenes subidas por los usuarios.

Tabla 2.4: Subdirectorios habituales en la carpeta web

2.7 Descripción de la solución propuesta

2.7.1 Autenticar usuario

El usuario ingresa sus credenciales y selecciona la opción entrar, se verifica en la base de datos si los datos introducidos son correctos, se asignan las credenciales y funcionalidades, así como el nivel de acceso a los recursos dentro del sistema.

2.7.2 Configuración

Se muestra al usuario el panel de configuración del sistema, donde se tiene un formulario en el que se podrá definir el título que aparecerá en el sistema, así como el correo del administrador de la aplicación y la posibilidad de habilitar o deshabilitar el servicio del sistema.

2.7.3 Gestionar usuarios

Para gestionar un usuario se selecciona en el menú de GEMPRO el vínculo Usuarios, se visualiza un listado con los usuarios existentes brindando la posibilidad de modificarlos, eliminarlos o crearlos. Para crear un nuevo usuario se selecciona el vínculo Nuevo, mostrándose un formulario donde se introduce en un campo de texto el nombre del nuevo usuario, luego se procede a escribir la contraseña del mismo, posteriormente se selecciona el rol de dicho usuario y los permisos de navegación a los que este tendrá acceso. Para editar un usuario se selecciona el vínculo Editar, luego se muestra un formulario donde aparecen los campos a modificar. Para eliminar un usuario se selecciona el vínculo Eliminar, en todas estas acciones se guardan los datos y se muestra el listado de usuarios actualizado.

2.7.4 Gestionar grupos

Para gestionar un grupo de usuarios se selecciona en el menú de GEMPRO el vínculo Grupos, se visualiza un listado con los grupos existentes brindando la posibilidad de modificarlos, eliminarlos o crearlos. Para crear un nuevo grupo se selecciona el vínculo Nuevo, mostrándose un formulario donde se introduce en un campo de texto el nombre del nuevo grupo, posteriormente se procede a escribir una descripción del mismo, después se seleccionan los usuarios a los cuales se le asignarán dicho grupo y los permisos de navegación a los que este grupo tendrá acceso. Para editar un grupo se selecciona el vínculo Editar, mostrándose un formulario donde aparecen los campos a modificar. Para eliminar un grupo se selecciona el vínculo Eliminar, en todas estas acciones se guardan los datos y se muestra el listado de los grupos actualizado.

2.7.5 Gestionar permisos

Para gestionar un permiso se selecciona en el menú de GEMPRO el vínculo Permisos, se visualiza un listado con los permisos existentes brindando la posibilidad de modificarlos, eliminarlos o crearlos. Para crear un nuevo permiso se selecciona el vínculo Nuevo, mostrándose un formulario donde se introduce en un campo de texto el nombre del nuevo permiso, posteriormente se procede a escribir una descripción del mismo, después se selecciona el grupo al cual va a estar asociado dicho permiso y los usuarios que tendrán asignados el mismo. Para editar un permiso se selecciona el vínculo Editar, mostrándose un formulario donde aparecen los campos a modificar. Para eliminar un permiso se selecciona el vínculo Eliminar, en todas estas acciones se guardan los datos y se muestra el listado de los permisos actualizado.

2.7.6 Gestionar empresa

Para gestionar una empresa se selecciona en el menú de GEMPRO el vínculo Empresa, se visualiza un listado con las empresas existentes brindando la posibilidad de modificarlas, eliminarlas o crearlas. Para crear una nueva empresa se selecciona el vínculo Nuevo, mostrándose un formulario donde se introduce en un campo de texto el código de la empresa, posteriormente se procede a seleccionar el organismo al cual pertenece, después se procede a escribir una descripción de la misma y la dirección donde esta ubicada. Para editar una empresa se selecciona el vínculo Editar, mostrándose un formulario donde aparecen los campos a modificar. Para eliminar una empresa se selecciona el vínculo Eliminar, en todas estas acciones se guardan los datos y se muestra el listado de las empresas actualizado.

2.7.7 Gestionar organismo

Para gestionar un organismo se selecciona en el menú de GEMPRO el vínculo Organismo, se visualiza un listado con los organismos existentes brindando la posibilidad de modificarlos, eliminarlos o crearlos. Para crear un nuevo organismo se selecciona el vínculo Nuevo, mostrándose un formulario donde se introduce en un campo de texto el código del organismo y posteriormente se procede a escribir una descripción del mismo. Para editar un organismo

se selecciona el vínculo Editar, mostrándose un formulario donde aparecen los campos a modificar. Para eliminar un organismo se selecciona el vínculo Eliminar, en todas estas acciones se guardan los datos y se muestra el listado de los organismos actualizado.

2.7.8 Gestionar moneda

Para gestionar una moneda se selecciona en el menú de GEMPRO el vínculo Moneda, se visualiza un listado con las monedas existentes brindando la posibilidad de modificarlas, eliminarlas o crearlas. Para crear una nueva moneda se selecciona el vínculo Nueva Moneda, mostrándose un formulario donde se introduce en un campo de texto el nombre de la moneda, así como su símbolo y valor. Para editar una moneda se selecciona el vínculo Editar, mostrándose un formulario donde aparecen los campos a modificar. Para eliminar una moneda se selecciona el vínculo Eliminar, en todas estas acciones se guardan los datos y se muestra el listado de las monedas actualizado.

2.7.9 Gestionar producto

Para gestionar un producto se selecciona en el menú de GEMPRO el vínculo Producto, se visualiza un listado con los productos existentes brindando la posibilidad de modificarlos, eliminarlos o crearlos. Para crear un nuevo producto se selecciona el vínculo Nuevo, mostrándose un formulario donde se introduce en un campo de texto el código del producto. Posteriormente se procede a escribir una descripción del mismo, después se selecciona el tipo de producto al cual pertenece, se introduce en un campo de texto el precio de compra y venta así como el tipo de moneda con la cual se comercia este producto. Para editar un producto se selecciona el vínculo Editar, mostrándose un formulario donde aparecen los campos a modificar. Para eliminar un producto se selecciona el vínculo Eliminar, en todas estas acciones se guardan los datos y se muestra el listado de los productos actualizado.

2.7.10 Gestionar solicitud

Para gestionar una solicitud se selecciona en el menú de GEMPRO el vínculo Solicitud, se visualiza un listado con las solicitudes realizadas hasta el momento brindando la posibilidad de modificarlas, eliminarlas o crearlas. Para realizar una nueva solicitud se selecciona el vínculo

Nuevo, mostrándose un formulario donde se selecciona la empresa que está realizando dicha solicitud así como el producto solicitado, posteriormente se introduce en un campo de texto la cantidad requerida así como la unidad básica de medida requerida. Para editar una solicitud se selecciona el vínculo Editar, mostrándose un formulario donde aparecen los campos a modificar. Para eliminar una solicitud se selecciona el vínculo Eliminar, en todas estas acciones se guardan los datos y se muestra el listado de las solicitudes actualizado. También brinda la funcionalidad de obtener reportes en dependencia del criterio de búsqueda especificado por el usuario.

2.8 Diagrama de componentes

Un diagrama de componentes es un diagrama tipo del Lenguaje Unificado de Modelado. Representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema.

Debido a que estos son más parecidos a los diagramas de casos de usos son utilizados para modelar la vista estática y dinámica de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente, se realizan por partes. Cada diagrama describe un apartado del sistema.

En él se situarán librerías, tablas, archivos, ejecutables y documentos que formen parte del sistema.

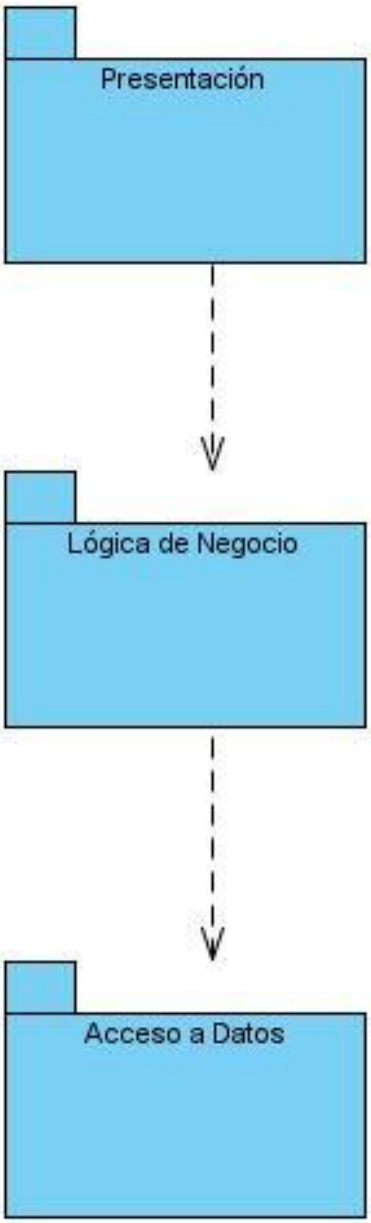

Figura 2. Diagrama de componentes general.

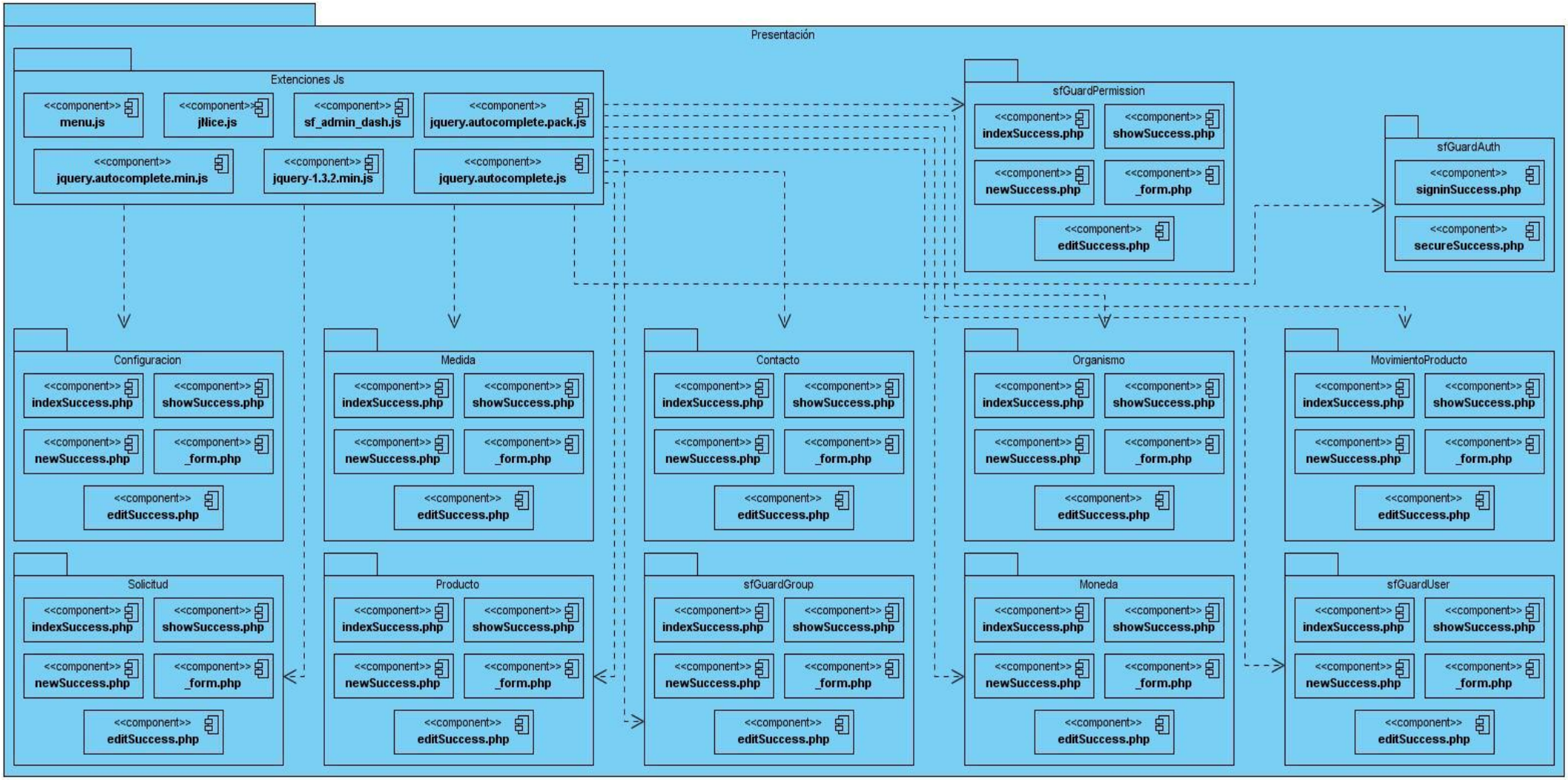

Figura 3. Diagrama de componentes de la capa presentación.

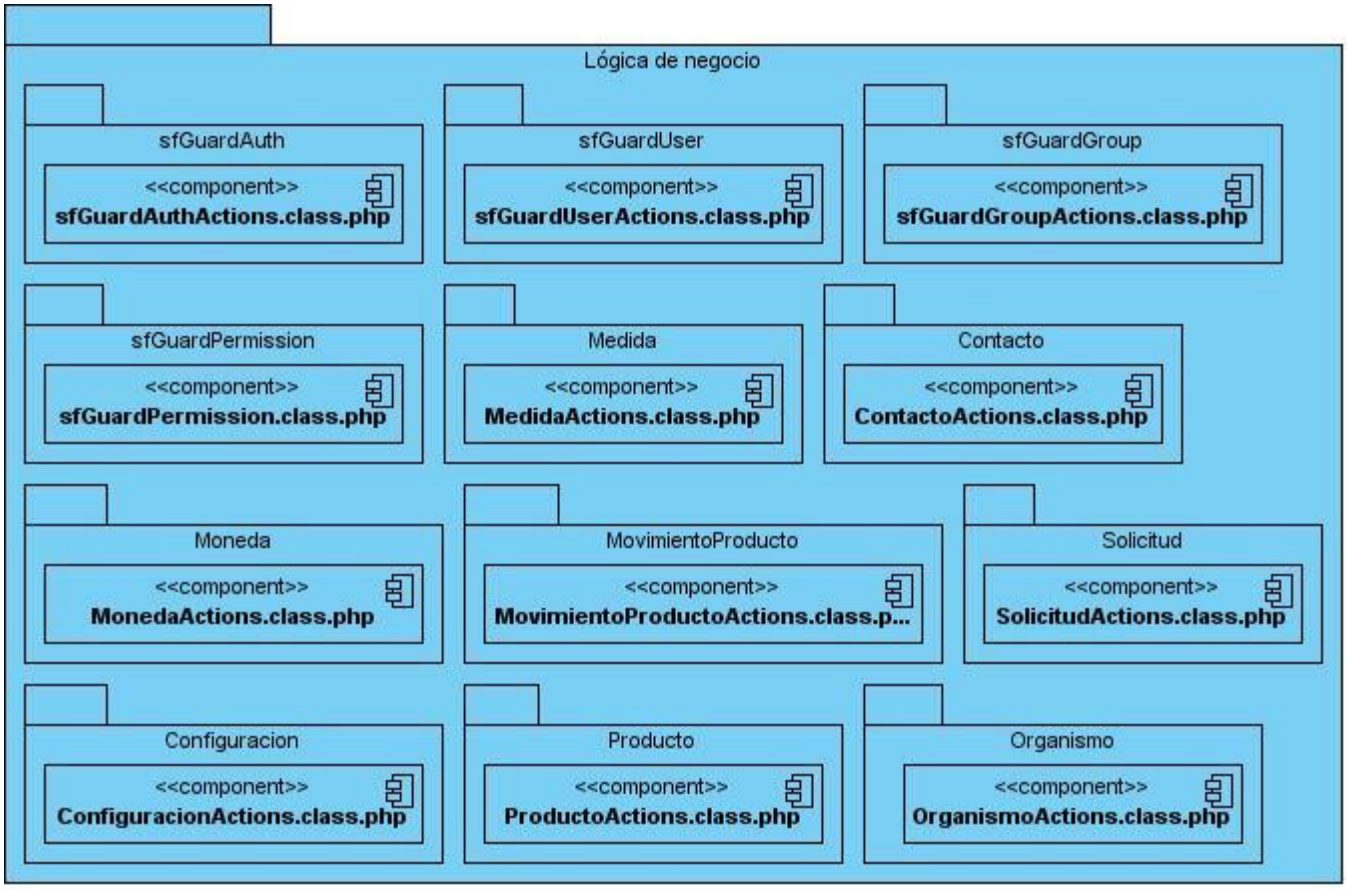

Figura 4. Diagrama de componentes de la capa lógica de negocio.

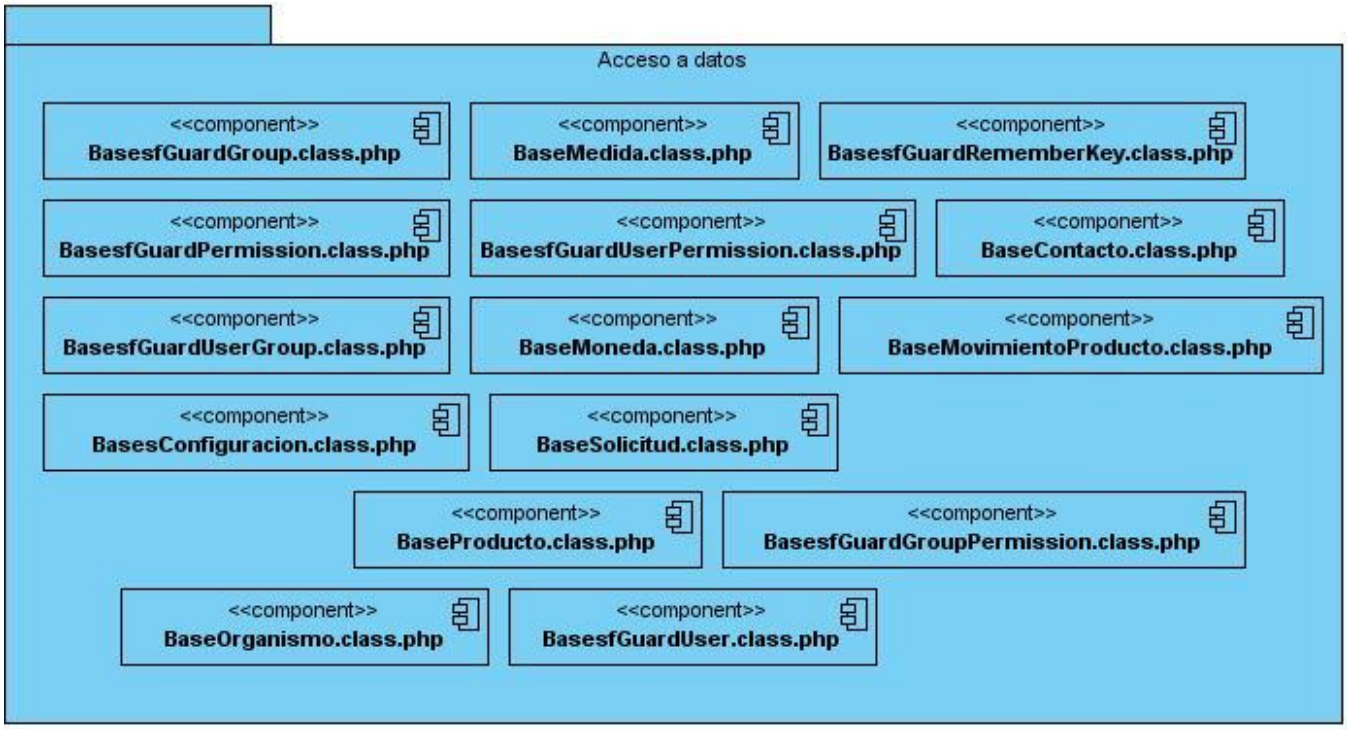

Figura 5. Diagrama de componentes de la capa de acceso a datos.

2.9 Modelo de despliegue

El modelo de despliegue describe la distribución física del sistema, muestra la distribución de los componentes de software entre los distintos nodos de cómputo. Permite comprender la correspondencia entre la arquitectura software y la arquitectura hardware.

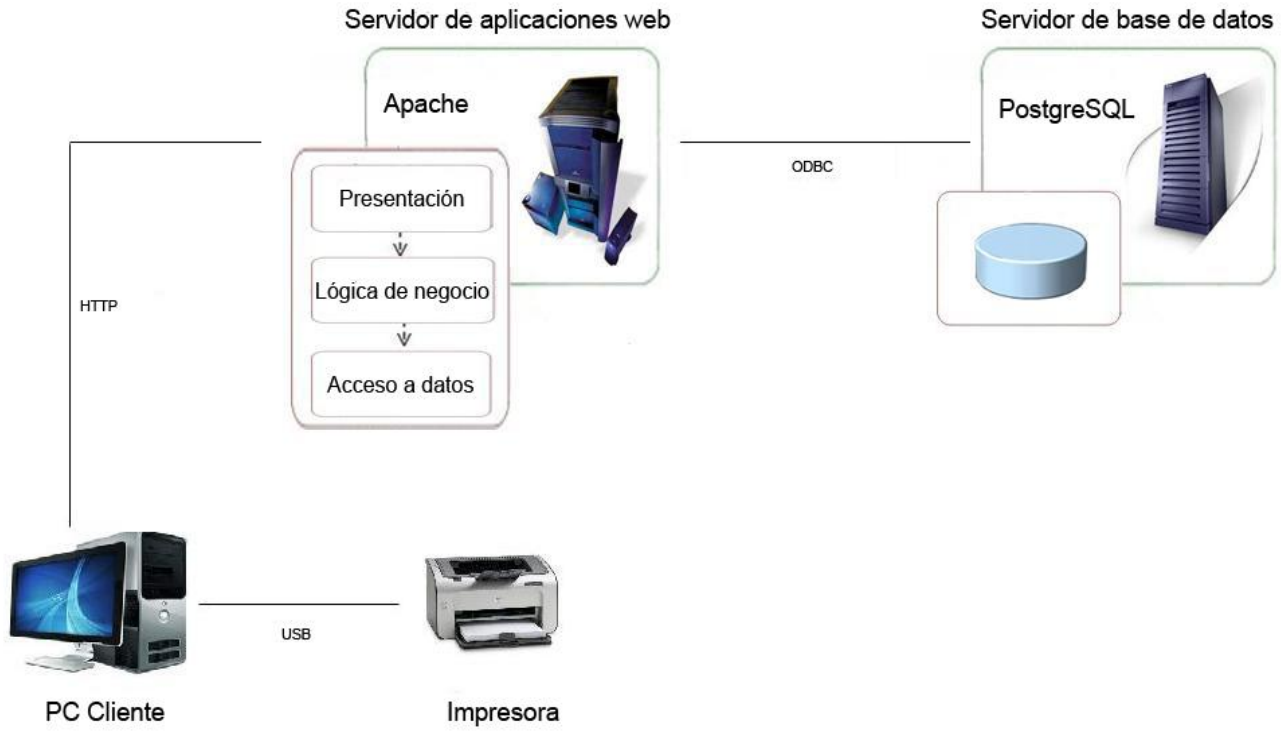

Figura 6: Diagrama de Despliegue.

2.10 Estándares de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo

armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar la implementación del sistema se establece un estándar de codificación para asegurarse de que todos los programadores del mismo trabajen de forma coordinada.

La legibilidad del código fuente repercute directamente en lo bien que un programador comprende un sistema de software. La mantenibilidad del código es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento. Aunque la legibilidad y la mantenibilidad son el resultado de muchos factores, una faceta del desarrollo de software en la que todos los programadores influyen especialmente es en la técnica de codificación.

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

Al conformar un estándar de codificación deben tenerse en cuenta varios criterios entre los cuales se pueden mencionar:

- Nombres representativos para variables, controles y procedimientos.
- Indentación (sangrías) y espacios apropiados en el código.
- Documentación del código (poner comentarios para aclarar).
- Procedimientos coherentes.

Nombres representativos*:* Cuando se definan nombres de variables, controles y procedimientos deben hacerse de forma representativa con el propósito de facilitar el entendimiento.

Indentación y espacios apropiados en el código: Visualizar un código fuente puede resultar complicado pero haciendo uso de la indentación se obtiene una mejor visibilidad pues se muestran las líneas que están sujetas a otras.

Documentación del código: Es una buena práctica documentar aquellas secciones de código más complicadas e inusuales. Al definir las variables y arreglos se deben documentar para que su función pueda ser comprendida posteriormente.

Procedimientos coherentes*:* No debe añadírsele demasiada complejidad a los procedimientos pues si manejan muchas tareas resulta natural que sean difíciles de entender y tengan una alta probabilidad de ocurrencia de errores.
A continuación se especifican los criterios de estandarización de código utilizados para el desarrollo de GEMPRO.

Indentación:

Se realiza siempre con 2 espacios en blanco; nunca se utilizan los tabuladores. La razón es que los tabuladores se muestran con distinta anchura en función del editor de texto utilizado, y porque el código que mezcla tabuladores con espacios en blanco es bastante difícil de leer.

```
11  class SolicitudeActions extends sfActions
12  {
13    public function executeIndex(sfWebRequest $request)
14    {
15      $this->pager = Doctrine::getTable('Solicitude')->getSolicitudePages($request->getParameter('page', 1));
16      $this->solicitudes = Doctrine::getTable('Solicitude')
17        ->createQuery('a')
18        ->execute();
19    }
```

Figura 7: Código indentado.

Saltos de línea:

- Añadir un salto de línea después del cierre de los paréntesis de los parámetros.
- Añadir un salto de línea después de un punto y coma, cuando termina la sentencia.

Espacios y líneas en blanco:

- Usar espacios en blanco para mejorar la legibilidad del código.

- Usar espacios en blanco en ambos lados del operador de símbolos, después de comas y después de las declaraciones.
- Usar líneas en blanco para separar trozos de código.

Longitud de la línea:

- Evitar las líneas de más de 80 caracteres cuando se supere esta cifra se debe reorganizar el código usando algún principio descrito anteriormente.

Apariencia de clases y funciones:

- Para las clases se establece *UpperCamelCase*[39].

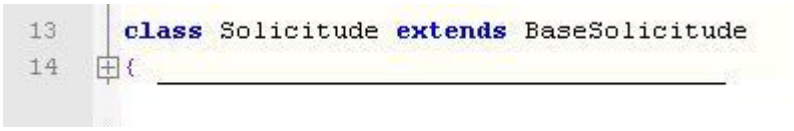

Figura 8: Ejemplo de clase que usa UpperCamelCase.

- Para las funciones se establece *lowerCamelCase*[40].

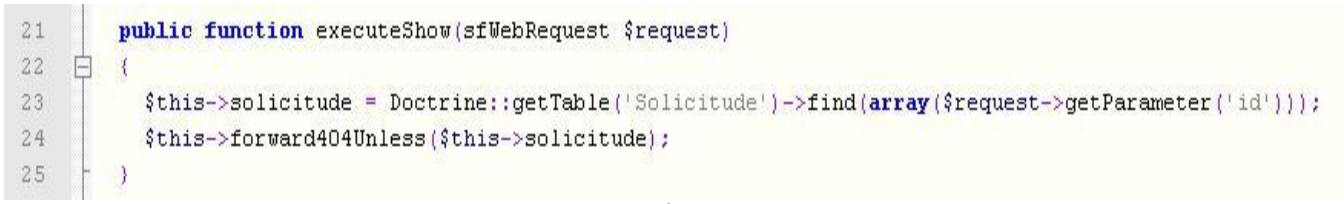

Figura 9: Ejemplo de función que usa lowerCamelCase.

Apariencia de atributos:

- Los atributos deben ser escritos en minúsculas y su nombre debe guardar relación con el valor que almacena.

$descripcion = $objeto->getDescripcion();

[39]Cuando la primera letra de cada una de las palabras es mayúscula.

[40]Igual que la aclaración anterior con la excepción de que la primera letra es minúscula.

2.11 Conclusiones

Al finalizar este capítulo queda implementado el sistema y descritas todas las clases utilizadas, cumpliendo con los objetivos específicos planteados y gran parte de las tareas propuestas para la elaboración del sistema. A partir de una breve descripción de cómo tener acceso a las funcionalidades implementadas y el sistema estar completamente funcional, se concluye expresando que se debería pasar a la etapa de pruebas y realizar los ensayos necesarios para asegurar que el sistema esté libre de no conformidades y con la debida calidad para ser entregado al cliente.

CAPÍTULO 3: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

3.1 Introducción

El desarrollo del software implica una serie de actividades de producción en las que las posibilidades de que aparezcan fallos humanos son muy grandes. Los errores pueden presentarse debido a especificaciones erróneas e imperfectas de los requisitos, uso indebido de las estructuras de datos, errores al integrar módulos, entre otras causas. Dado a la imposibilidad humana de trabajar y comunicarse de forma perfecta, el desarrollo del software ha de ir acompañado de una actividad que garantice la calidad.

Las pruebas es una actividad en la cual un sistema o componente es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente. En este capítulo se realiza la validación a la solución propuesta, con el objetivo de comprobar la eficiencia de las clases u operaciones utilizadas para dar respuesta a los distintos requisitos planteados por el cliente.

3.2 Pruebas de software

La calidad de un sistema está determinada, entre otras cosas, por la coincidencia entre lo que se programó y los requisitos establecidos en la primera fase. Para comprobar el grado de cumplimiento de estos requisitos se usan las pruebas del sistema. Estas definen un conjunto amplio de acciones de comprobación que abarcan todas las características que determinan la calidad de un software. Se comprueban las funcionalidades diseñando casos de prueba que definen cómo proceder. Estos casos de prueba incluyen los juegos de datos a usar que son los válidos o esperados y los no válidos o no esperados por el programa. Además, establecen los resultados a alcanzar en correspondencia de la lógica del programa y los datos

ingresados. Describen las condiciones generales en las que se deben aplicar las pruebas para obtener los objetivos propuestos. El objetivo de los casos de prueba es forzar al máximo el sistema en los puntos críticos para encontrar fallos y detectar defectos. Las pruebas se deben aplicar durante todo el ciclo de vida del software e invariablemente se le debe dedicar una gran parte del esfuerzo total del desarrollo. Se deben planificar correctamente desde el inicio y establecer qué hacer, cómo hacer, quién va a hacer y en qué condiciones hacer las comprobaciones. Es beneficioso que los desarrolladores prueben su producto pero que no falte la mano de terceras personas que no intervinieron en el proyecto directamente, ya que así se detecta mayor cantidad de fallas. (Pressman, 2005)

Se deben escoger los tipos de prueba que se adapten mejor al sistema que se va a probar. Para esto se debe tener en cuenta el lenguaje de programación, el proceso de desarrollo, las características de los desarrolladores, el tipo de funcionalidad que se implementa, la plataforma en que se ejecutan los procesos, los errores más importantes, si la aplicación es de escritorio o web y si realiza conexiones a bases de datos entre otras observaciones.

3.3 Objetivos de las pruebas

El objetivo de las pruebas de un programa es detectar todo posible mal funcionamiento, un error puede ser costoso de reparar mientras más avanza la etapa del ciclo de vida del software. Para minimizar estos riesgos se crean baterías de pruebas que serán de mayor calidad en la medida de cuantos menos errores queden por descubrir tras haberlas pasado y viceversa, si un programa aún tiene muchos fallos tras haberla superado, se dirá que esta es de poca calidad. Si se pudiera probar un programa con todos los posibles datos de entrada, se tendría una batería de pruebas perfecta, pues no hay lugar para las sorpresas. Lamentablemente, casi nunca es posible probar con todos los casos. En consecuencia, se necesita un criterio para elegir qué casos se prueban.

3.4 Automatización de pruebas

Cualquier programador con experiencia en el desarrollo de aplicaciones web conoce de sobra el esfuerzo que supone probar correctamente la aplicación. Crear casos de prueba,

ejecutarlos y analizar sus resultados es una tarea tediosa. Además, es habitual que los requisitos de la aplicación varíen constantemente, con el consiguiente aumento del número de versiones de la aplicación y la refactorización continua del código. En este contexto, es muy probable que aparezcan nuevos errores. Este es el motivo por el que la automatización de pruebas es una recomendación, aunque no una obligación, útil para crear un entorno de desarrollo satisfactorio. Los conjuntos de casos de prueba garantizan que la aplicación hace lo que se supone que debe hacer. Incluso cuando el código interno de la aplicación cambia constantemente, las pruebas automatizadas permiten garantizar que los cambios no introducen incompatibilidades en el funcionamiento de la aplicación. Además, este tipo de pruebas obliga a los programadores a crear pruebas en un formato estandarizado y muy rígido que pueda ser procesado por un marco de trabajo de pruebas.

En ocasiones, las pruebas automatizadas pueden reemplazar la documentación técnica de la aplicación, ya que ilustran de forma clara el funcionamiento de la aplicación. Un buen conjunto de pruebas muestra la salida que produce la aplicación para una serie de entradas de prueba, por lo que es suficiente para entender el propósito de cada método. Symfony aplica este principio a su propio código. El código interno del marco de trabajo se valida mediante pruebas automáticas. (Potencier, 2008)

3.5 El marco de trabajo Lime

En el ámbito de PHP existen muchos marcos de trabajo para crear pruebas, siendo los más conocidos PHPUnit y SimpleTest. Symfony incluye su propio marco de trabajo llamado Lime. Se basa en la librería Test::More de Perl. Lime es más eficiente que otros marcos de trabajo de pruebas de PHP y tiene las siguientes ventajas:

- Ejecuta los archivos de prueba en un entorno independiente para evitar interferencias entre las diferentes pruebas. No todos los marcos de trabajo de pruebas garantizan un entorno de ejecución "limpio" para cada prueba.
- Las pruebas de Lime son fáciles de leer y sus resultados también lo son. En los sistemas operativos que lo soportan, los resultados de Lime utilizan diferentes colores para mostrar de forma clara la información más importante.

- Symfony utiliza Lime para sus propias pruebas, por lo que el código fuente de Symfony incluye muchos ejemplos reales de pruebas unitarias y funcionales.
- Está escrito con PHP, es muy rápido y está bien diseñado internamente. Consta únicamente de un archivo, llamado lime.php y no tiene ninguna dependencia. (Potencier, 2008)

3.6 Pruebas unitarias

Las pruebas unitarias permiten probar, como su nombre lo indica, cada unidad independiente del software. Actúan esencialmente sobre el código fuente y sobre los elementos básicos de la interfaz de cada módulo. Pressman plantea que los casos de prueba que se generan durante las pruebas de unidad deben estar encaminados a verificar los siguientes elementos:

- Interfaz: Se prueba la interfaz del módulo para asegurar que la información fluye de forma adecuada hacia y desde la unidad de programa que está siendo probada.
- Estructuras de datos locales: Se examinan las estructuras de datos locales para asegurar que los datos que se mantienen temporalmente, conservan su integridad durante todos los pasos de ejecución del algoritmo.
- Condiciones límite: Se prueban las condiciones límites para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones de procesamiento.
- Caminos independientes: Se ejercitan todos los caminos independientes (caminos básicos) de la estructura de control con el fin de asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez.
- Camino de manejo de errores: Se prueban todos los caminos de manejo de errores.

Con las pruebas unitarias es posible aislar una parte del código de manera que pueda ser analizado. Un ejemplo de esto es evaluar las funciones o métodos, a los cuales se les realiza una entrada de datos para obtener los datos de salida correctos. Este tipo de pruebas valida la forma en la que las funciones y métodos trabajan en cada caso particular.

3.7 Pruebas funcionales

Se denominan pruebas funcionales, a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido creados.

Es común que este tipo de pruebas sean desarrolladas por analistas de pruebas con apoyo de algunos usuarios finales, esta etapa suele ser la última etapa de pruebas. A este tipo de pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra, ya que los analistas de pruebas, no enfocan su atención a como se generan las respuestas del sistema. El enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y en los de salida, esto generalmente se define en los casos de prueba preparados antes del inicio de las pruebas.

Al realizar pruebas funcionales lo que se pretende es ponerse en los pies del usuario, usar el sistema como él lo usaría, sin embargo, el analista de pruebas debe ir mas allá que cualquier usuario, generalmente se requiere apoyo de los usuarios finales ya que ellos pueden aportar mucho en el desarrollo de casos de prueba complejos, enfocados básicamente al negocio, posibles particularidades que no se hayan contemplado adecuadamente en el diseño funcional, el analista de pruebas debería dar fuerza a las pruebas funcionales.

Generalmente los usuarios realizan las pruebas con la idea que todo debería funcionar, a diferencia del analista de pruebas que tiene más bien una misión destructiva, su objetivo será encontrar alguna posible debilidad y si la llega a ubicar, se esforzará porque deje de ser pequeña y posiblemente se convertirá en un gran error, cada error encontrado por el analista de pruebas es un éxito, y se debería considerar como tal.

Los sistemas que han pasado por pruebas unitarias tienen un menor tiempo de pruebas funcionales, este comportamiento es obvio, ya que las pruebas unitarias permiten encontrar los errores más evidentes y fáciles de corregir. Si un sistema llega a la etapa de pruebas funcionales con demasiados errores críticos y/o bloqueantes, se debería devolver el sistema a la etapa de pruebas unitarias ya que resulta muy poco productivo realizar pruebas funcionales con sistemas inestables. El avance es demasiado lento y el analista de pruebas no podrá

apoyar mucho en la resolución de los errores ya que en esta etapa solo se centra la atención en las entradas y salidas, y no en la lógica intermedia. (Oré, 2009)

3.8 Pruebas de aceptación

El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema que determine su aceptación, desde el punto de vista de su funcionalidad y rendimiento.

Las pruebas de aceptación son definidas por el usuario del sistema y preparadas por el equipo de desarrollo, aunque la ejecución y aprobación final corresponden al usuario.

La validación del sistema se consigue mediante la realización de pruebas de caja negra que demuestran la conformidad con los requisitos, el cual define las verificaciones a realizar y los casos de prueba asociados. Dicho plan está diseñado para asegurar que se satisfacen todos los requisitos funcionales especificados por el usuario teniendo en cuenta también los requisitos no funcionales relacionados con el rendimiento, seguridad de acceso al sistema, a los datos y procesos, así como a los distintos recursos del sistema.

Después de realizada una reunión entre el equipo de desarrollo y el cliente, el mismo aprueba el producto dándole la categoría de satisfactorio y altamente funcional para la Unión de Empresas de Recuperación de Materias Primas.

3.9 Pruebas unitarias realizadas

En Symfony las pruebas unitarias se guardan en el directorio test/unit/. Además, Symfony utiliza la convención de nombrar las pruebas mediante el nombre de la clase que prueban seguido de la palabra Test. (Potencier, 2008)

Para ejecutar el conjunto de pruebas, se utiliza la tarea test:unit desde la línea de comandos. El resultado de esta tarea en la línea de comandos es muy explícito, lo que permite localizar fácilmente las pruebas que han fallado y las que se han ejecutado correctamente.

En primer lugar, se instancia el objeto lime_test. Las siguientes pruebas unitarias consisten en una llamada al método de la instancia de lime_test. El último parámetro de estos métodos siempre es una cadena de texto opcional que se utiliza como resultado del método.

A continuación se muestra uno de los casos de prueba, para ver los demás dirigirse a los Anexos.

#	Funcionalidad	Tipo de Prueba	Recibe	Esperado	Resultado
2	executeSignin()	ok	'administrador','admin'	ok	ok
3	executeSignin()	is	'administrador', 'admin'	Not ok	Not ok
4	executeSignin ()	isa_ok	'administrador', 'admin', ' sfGuardAuthActions'	ok	ok

Tabla 3.5: Clase sfGuardAuthActions.

3.9.1 Métodos para las pruebas unitarias

El objeto lime_test dispone de un gran número de métodos para las pruebas, como se muestra en la tabla 6.

Método	Descripción
comment($mensaje)	Muestra un comentario, pero no ejecuta ninguna prueba.
ok($prueba, $mensaje)	Si la condición que se indica es *true*, la prueba tiene éxito.
is($valor1, $valor2, $mensaje)	Compara 2 valores y la prueba pasa si los 2 son iguales (==).
isnt($valor1, $valor2, $mensaje)	Compara 2 valores y la prueba pasa si no son iguales.
like($cadena, $expresionRegular, $mensaje)	Prueba que una cadena cumpla con el patrón de una expresión regular.
unlike($cadena, $expresionRegular, $mensaje)	Prueba que una cadena no cumpla con el patrón de una expresión regular.
cmp_ok($valor1, $operador, $valor2, $mensaje)	Compara 2 valores mediante el operador que se indica.
isa_ok($variable, $tipo, $mensaje)	Comprueba si la variable que se le pasa es del tipo que se indica.
isa_ok($objeto, $clase,	Comprueba si el objeto que se le pasa es de la clase que se indica.

$mensaje)	
can_ok($objeto, $método, $mensaje)	Comprueba si el objeto que se le pasa dispone del método que se indica.
is_deeply($array1, $array2, $mensaje)	Comprueba que 2 arreglos tengan los mismos valores.
include_ok($archivo, $mensaje)	Valida que un archivo existe y que ha sido incluido correctamente.
fail()	Provoca que la prueba siempre falle (es útil para las excepciones).
pass()	Provoca que la prueba siempre se pase (es útil para las excepciones).
todo()	Cuenta como si fuera una prueba (es útil para las pruebas que todavía no se han escrito).

Tabla 3.6: Métodos para las pruebas unitarias.

3.10 Pruebas funcionales realizadas

En Symfony las pruebas funcionales se guardan en el directorio test/functional/. Symfony dispone de un objeto especial, llamado sfBrowser, que actúa como un navegador que está accediendo a una aplicación, pero sin necesidad de utilizar un servidor web real (y sin la penalización de las conexiones HTTP). Este objeto permite el acceso directo a los objetos que forman cada petición (los objetos petición, sesión, contexto y respuesta). Symfony también dispone de una extensión de esta clase llamada sfTestBrowser, que está especialmente diseñada para las pruebas funcionales y que tiene todas las características de sfBrowser.
Una prueba funcional suele comenzar con la inicialización del objeto del navegador para pruebas. Este objeto permite realizar una petición a una acción de la aplicación y permite verificar que algunos elementos están presentes en la respuesta. (Potencier, 2008)
Para ejecutar una prueba funcional, se utiliza la tarea test: functional de la línea de comandos de Symfony. Los argumentos que se indican a la tarea son el nombre de la aplicación y el nombre de la prueba omitiendo el sufijo Test.php.

A continuación se muestra uno de los casos de prueba, para ver los demás dirigirse a los Anexos.

Funcionalidad	Escenario	Clases válidas	Clases inválidas	Resultado Esperado	Resultado de la prueba
1. Autenticar usuario	**EC 1.1 Autenticar usuario**	Introducir el nombre de usuario y la contraseña correctamente.		Permitir la navegación en el sistema después de asignar los privilegios de acuerdo al rol del usuario.	Permite la navegación al sistema después de asignar los privilegios de acuerdo al rol que posee dicho usuario.
			Introducir el nombre de usuario y/o la contraseña incorrectamente.	Mostrar mensaje de error en caso de que los datos introducidos sean incorrectos.	Muestra un mensaje de error notificando que hay datos introducidos incorrectamente.

Tabla 3.7: Escenarios de las funcionalidades.

3.10.1 Métodos para las pruebas funcionales

La clase sfBrowser incluye métodos que simulan la navegación que se realiza en cualquier navegador tradicional:

Método	Descripción
get()	Obtiene una URL.
post()	Envía datos a una URL.
call()	Realiza una llamada a una URL (se utiliza para los métodos PUT y DELETE).
back()	Vuelve a la página anterior almacenada en el historial.
forward()	Va a la página siguiente almacenada en el historial.
reload()	Recarga la página actual.
click()	Pulsa sobre un enlace o un botón.

select()	Selecciona un *radiobutton* [41] o un *checkbox*[42].
deselect()	Deselecciona un *radiobutton* o un *checkbox.*
restart()	Reinicia el navegador.

Tabla 3.8: Métodos que simulan la navegación

La clase sfBrowser también incluye métodos para configurar el comportamiento del navegador:

Método	Descripción
`setHttpHeader()`	Establece el valor de una cabecera HTTP.
`setAuth()`	Establece las credenciales de la autenticación básica.
`setCookie()`	Establece una *cookie*.
`removeCookie()`	Elimina una *cookie*.
`clearCookie()`	Borra todas las *cookies* actuales.
`followRedirect()`	Sigue una redirección.

Tabla 3.9: Métodos para configurar el comportamiento del navegador

Los métodos para pruebas se incluyen en otra clase llamada sfTestFunctional y que utiliza como argumento de su constructor un objeto de tipo sfBrowser. La clase sfTestFunctional delega las pruebas en objetos de tipo tester.

El tester request incluye métodos para realizar la introspección y probar los objetos de tipo sfWebRequest:

Método	Descripción
isParameter()	Comprueba el valor de un parámetro de la petición.
isFormat()	Comprueba el formato de la petición.
isMethod()	Comprueba el método utilizado.
hasCookie()	Comprueba si la petición incluye una *cookie* con el nombre indicado.
isCookie()	Comprueba el valor de una *cookie*.

Tabla 3.10: Métodos para realizar la introspección

[41] Elemento de la interfaz gráfica que permite al usuario hacer una única selección de un conjunto de opciones.

[42] Elemento de la interfaz gráfica que permite al usuario hacer selecciones múltiples de un conjunto de opciones.

También existe un tester response que incluye los métodos equivalentes para los objetos de tipo sfWebResponse:

Método	Descripción
checkElement()	Comprueba si un selector CSS sobre la respuesta cumple el criterio indicado.
isHeader()	Comprueba el valor de una cabecera.
isStatusCode()	Comprueba el código de estado de la respuesta.
isRedirected()	Comprueba si la respuesta actual es en realidad una redirección.

Tabla 3.11: Métodos equivalentes para los objetos de tipo sfWebResponse

3.11 Mejoras en el proceso de gestión

Con el desarrollo de GEMPRO el proceso de gestión del movimiento de productos se ha centralizado en un sistema, logrando la integración de la información y mejora de la comunicación entre las diferentes áreas. Se obtiene información disponible e inmediata para la toma de decisiones. Con la automatización de este proceso se disminuye el tiempo empleado en la realización de las actividades que anteriormente se desarrollaban de forma manual en el área de trabajo de los productos no metálicos y en las de los productos ferrosos y no ferrosos por los sistemas Concilia y SI – Estadístico 451 respectivamente, logrando más confianza y credibilidad en los datos manejados por los usuarios a los cuales se brinda una interfaz visual agradable y nuevas funcionalidades, ampliando las posibilidades de uso de las TIC al desarrollar un sistema que será accesible por todas las empresas pertenecientes a la Unión de Empresas de Recuperación de Materias Primas, reduciéndose los gastos por concepto de transporte y material de oficina. Todas las mejoras antes mencionadas traen consigo la reducción del tiempo empleado en la gestión del movimiento de productos contribuyendo así a la mejora de la misma.

3.12 Conclusiones

Con el desarrollo de este capítulo se pudo apreciar la importancia que tiene el proceso de prueba en el desarrollo de software, para el buen entendimiento del mismo se analizaron diferentes aspectos como el significado de las pruebas de software y sus objetivos, se realizó

una descripción de las pruebas de unidad y funcionales. Por último, se puede asegurar que las pruebas realizadas con el marco de trabajo de prueba Lime que incluye Symfony, posibilitó la corrección de defectos y permitieron comprobar diferentes métodos utilizados.

CONCLUSIONES GENERALES

Una vez finalizada la investigación científica se concluye que:

- El estudio de la documentación obtenida en las iteraciones anteriores realizadas al Sistema de Gestión de Movimiento de Productos, permitió un mejor entendimiento de los procesos realizados en la Unión de Empresas de Recuperación de Materias Primas que facilitaron, en gran medida, la implementación.
- Se automatizan las actividades que anteriormente los trabajadores, realizaban de forma manual disminuyendo el tiempo de ejecución de las mismas.
- Se genera información confiable y precisa para la toma de decisiones.
- Se obtienen los resultados esperados en las pruebas unitarias y funcionales realizadas garantizando que los requerimientos fueron cumplidos y que el sistema es estable.

RECOMENDACIONES

Una vez vencidos los objetivos de esta investigación, teniendo en cuenta las experiencias obtenidas a lo largo del desarrollo, se recomienda:

- Trabajar en el desarrollo de nuevas funcionalidades al sistema.
- Desarrollar un manual de usuario para familiarizar al cliente en el manejo del sistema.

REFERENCIAS BIBLIOGRÁFICAS

1- Sitio de Descargas de Software. [En línea] [Citado el: 13 de Febrero de 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/

2- Abraham. (2007). *Java Hispano*. Recuperado el 21 de Enero de 2010, de http://www.javahispano.org/

3- *Ciberaula*. (2006). Recuperado el 05 de Febrero de 2010, de http://linux.ciberaula.com/articulo/linux_apache_intro/

4- *EVA.* (s.f.). Recuperado el 04 de Febrero de 2010, de http://eva.uci.cu/mod/resource/view.php?id=2241

5- Gutiérrez, J. (s.f.). *Lenguajes y Sistemas Informáticos.* Recuperado el 02 de Febrero de 2010, de http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf

6- Márquez, L. (12 de Marzo de 2007). *Ambiente Plástico*. Recuperado el 10 de Diciembre de 2009, de http://www.ambienteplastico.com/suscriptores/article_704.php

7- Pérez, J. E. (2008). *Libros Web*. Recuperado el 10 de Enero de 2010, de http://www.librosweb.es/ajax/index.html

8- *PHP*. (27 de Marzo de 2009). Recuperado el 29 de Enero de 2010, de http://mx.php.net/manual/es/history.php.php

9- Potencier, F. (2008). *librosweb.* Recuperado el 03 de Febrero de 2010, de http://www.librosweb.es/symfony_1_1

10- Pressman, R. (2005). *Ingeniería del software. Un enfoque práctico.* La Habana: Félix Varela.

11- *Programación Web*. (s.f.). Recuperado el 01 de Febrero de 2010, de http://www.programacionweb.net/articulos/articulo/?num=184

12- *RodasXXI*. (2007). Recuperado el 13 de Diciembre de 2009, de http://www.rodasxxi.cu/inventario.php

13- Santamaría, F. (s.f.). *Fesabid 98*. Recuperado el 20 de Enero de 2010, de http://fesabid98.florida-uni.es/Comunicaciones/f_santamaria/f_santamaria.htm

14- *SISCONT5.* (2007). Recuperado el 16 de Diciembre de 2009, de http://siscont.tm.minbas.cu/Docs/Documentos/Manuales/Inventario%20SISCONT5.pdf

15-Tamargo, L. C. (s.f.). *La Revista del Empresario Cubano*. Recuperado el 06 de Enero de 2010, de http://www.betsime.disaic.cu/secciones/tec_feb_02.htm

16-*todoexpertos*. (2009). Recuperado el 01 de Febrero de 2010, de http://www.todoexpertos.com/categorias/tecnologia-e-internet/bases-de- datos/oracle/

BIBLIOGRAFÍA

Fabien Potencier, F. Z. (2008). *Symfony la guía definitiva.*

Pressman, R. (2005). *Ingeniería del software. Un enfoque práctico.*

Ordenadores, C. (s.f.). *Cea Ordenadores*. Obtenido de http://www.ceaordenadores.com/

RodasXXI. (s.f.). *RodasXXI*. Obtenido de http://www.rodasxxi.cu/inventario.php

SISCONT5. (s.f.). *SISCONT5.* Obtenido de http://siscont.tm.minbas.cu/Docs/Documentos/Manuales/Inventario%20SISCONT5.pdf

DesarrolloWeb.com. (s.f.). *DesarrolloWeb.com.* Obtenido de Manual completo de HTML: http://www.desarrolloweb.com/manuales/21/

Miguel Angel Alvarez, R. A. (s.f.). *DesarrolloWeb.com*. Obtenido de http://www.desarrolloweb.com/manuales/34/

PostgreSQL. (s.f.). Obtenido de http://www.postgresql.cl/

Foundation, A. S. (2010). *The Apache Software Foundation*. Obtenido de http://www.apache.org/

Corporation, M. (2010). *Internet Information server*. Obtenido de http://www.iis.net/

EllisLab, I. (s.f.). *Manual de CodeIgniter.* Obtenido de http://codeigniter.com/user_guide/

Rubén Alvarez, P. R. (s.f.). *Programación en ASP.* Obtenido de www.desarrolloweb.com/manuales/8/

Miguel Angel Alvarez, J. M. (s.f.). *Qué es cada tecnología.* Obtenido de http://www.desarrolloweb.com/manuales/15/

IBM. (s.f.). *IBM Rational Unified Process.* Obtenido de http://www-01.ibm.com/software/awdtools/rup/

Gracia, J. (2005). *IngenieroSoftware.com.* Obtenido de http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php

visual-paradigm.com. (s.f.). Obtenido de http://www.visual-paradigm.com/

Object Management Group, I. (s.f.). *Unified Modeling Language*. Obtenido de http://www.uml.org/

Málaga, G. d. (s.f.). *BIBLIOTECA VIRTUAL de Derecho, Economía y Ciencias Sociales*. Obtenido de http://www.eumed.net/libros/2007c/306/METODOS%20GENERALES%20DE%20LA%20INVESTIGACION%20CIENTIFICA.htm

Inc., Z. T. (s.f.). *ZendFramework*. Obtenido de http://manual.zfdes.com/

Alvarez, S. (s.f.). *Manual de iniciación a la programación.* Obtenido de http://www.desarrolloweb.com/manuales/74/

ZonaOracle.com. (s.f.). *Comunidad Oracle en español.* Obtenido de http://www.zonaoracle.com/