Universidad de las Ciencias Informáticas
Facultad 3

"Desarrollo del Módulo Centro de balance del subsistema de Planificación del Sistema Integral de Gestión de Entidades CEDRUX."

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

Autores: Charles López Fiol
Grabiel Bueno Sandoval

Tutor: Ing. Mairelys Fernández González
Co-Tutor: Tte. Ing. Yariel Hernández Espino

"Año 53 de la Revolución"

Declaración de autoría

Declaramos que somos los únicos autores del trabajo “Desarrollo del módulo Centro de balance para el subsistema de Planificación del Sistema Integral de Gestión (Cedrux).” y autorizamos a la Facultad 3 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio. Para que así conste firmamos la presente a los ____ días del mes de _____del año _____.

_____ _____

Charles López Fiol

Autor

Grabiel Bueno Sandoval

Autor

_____ _____

Ing. Mairelys Fernández González

Tutor

Tte. Ing. Yariel Hernández Espino

Co-Tutor

Resumen

La gestión de las operaciones de los Centros de balance de la Planificación en las entidades del país se realiza de forma manual. Esto provoca demora en los servicios y pérdida de la información. Para darle solución al problema anterior, se desarrolló el presente trabajo teniendo como objetivo fundamental desarrollar el módulo Centro de balance del subsistema Planificación del Sistema Integral de Gestión de Entidades CEDRUX de forma tal que se perfeccione la gestión de las operaciones de los Centros de balance de la Planificación en las entidades del país. Para cumplir con el objetivo planteado se realiza un estudio de cómo se lleva a cabo el proceso en el país, así como de las herramientas existentes empleadas para la realización del mismo. Se define el Proceso del negocio, identificando las actividades que se realizan en el mismo y quienes las ejecutan mediante la realización de un diagrama de procesos del negocio. Se determinan los requisitos funcionales y se elabora una detallada descripción de los mismos, a partir de las cuales se efectúa el diseño de clases y de la base de datos, entrada fundamental para la fase de implementación. Por último se realiza la validación del diseño con la aplicación de métricas y las pruebas del sistema, con la elaboración de los casos de pruebas para las funcionalidades identificadas aplicadas a las interfaces elaboradas.

Palabras clave: Planificación, Centro de balance, Gestión.

Índice de contenidos

Índice de figuras

Índice de tablas

Introducción

El nuevo siglo ha traído consigo un vertiginoso auge de Internet y de las tecnologías de la información y las comunicaciones (TIC), lo que a su vez ha propiciado el surgimiento de nuevos paradigmas y fundamentalmente el paradigma tecnológico productivo; es decir, lograr un mayor desarrollo en el funcionamiento y productividad de las empresas haciendo uso de los nuevos avances de la ciencia y la técnica para un eficiente manejo y análisis de la información, permitiendo un mejor proceso de toma de decisiones en las empresas y una mayor competencia en el mercado.

La planificación; proceso gradual por el que se establece el esfuerzo necesario para cumplir los objetivos, constituye la base para lograr una mayor organización y control de los recursos. De ella se deriva la planificación financiera, la cual consiste específicamente en alcanzar los objetivos a través del manejo adecuado de las finanzas, mediante la elaboración de un plan en el que se detalla y describe la estrategia de la empresa y se hacen previsiones basadas en los diferentes estados contables y financieros de la misma. Por otra parte existe la planificación de requerimientos de materiales; que no es más que el conjunto de técnicas basadas en el Plan Maestro de Producción donde se tiene en cuenta la información de registros de inventario y los documentos de estructura de los productos con el propósito de generar cronogramas detallados en los que se identifican los artículos específicos, los materiales requeridos para producirlos y así determinar las fechas según los plazos de entrega. Uno de los procesos que se llevan a cabo en la Planificación material y financiera es la gestión de los centros de balance. Este proceso consiste en realizar un análisis de la información relacionada con cada una de las demandas de las diferentes entidades y los productos que se tienen en existencia en los almacenes, así como por concepto de importación y exportación.

En las entidades cubanas se planifica empleando tanto métodos manuales como automatizados. Cuando este proceso se lleva acabo de forma manual resulta lento, engorroso, conlleva a errores matemáticos que repercuten en indisciplinas contables y provoca tanto insuficiencias en el uso y administración de los recursos como la descentralización de la información que se maneja.

La informatización, por su parte, constituye un eslabón fundamental y estratégico para todas aquellas empresas que pretenden desarrollarse. Cuba en pos de su desarrollo informático hace algunos años se inició en el proceso de informatización de la actividad económica, donde la planificación es uno de los procesos que se ha incluido en muchas de estas soluciones informáticas. Sin embargo a la hora

de planificar mediante el uso de herramientas automatizadas no solo de emplean sistemas nacionales sino también sistemas de producción extranjera. Si bien estos productos informáticos ya no presentan las dificultades de la realización del proceso de forma manual, están sujetos a otras deficiencias. El hecho de que muchas de estas aplicaciones estén desarrolladas sobre tecnologías y herramientas privativas, trae consigo que las posibilidades de uso estén limitadas, a lo que se le suma el inconveniente de brindarles el mantenimiento que requieren, obstaculizando la gestión eficiente de la planificación.

Indudablemente hoy no se cuenta en el país con un sistema informático con la capacidad de cumplir con totalidad los requerimientos de funcionalidad, interoperabilidad y seguridad, de manera que pueda ser utilizado por las entidades.

La Universidad de las Ciencias Informáticas en conjunto con el Ministerio de Economía y Planificación, actualmente se encuentran desarrollando el Sistema Integral de Gestión de Entidades(CEDRUX) con el objetivo de darle una solución viable al país en lo que respecta a la planificación de las entidades, que aunque cuenta con el subsistema Planificación no realiza la gestión de las operaciones de los centros de balance.

El presente trabajo de diploma para dar solución a la situación descrita con anterioridad tiene el siguiente **problema a resolver**: ¿Cómo perfeccionar la gestión de operaciones de los Centros de balance de la Planificación en las entidades del país?

Se define por lo tanto como objeto de estudio: Los procesos de la Planificación material y financiera y el campo de acción: La gestión de operaciones de los Centros de balance.

El **objetivo general** del presente trabajo de diploma es: Desarrollar el Módulo Centro de balance del subsistema Planificación del Sistema Integral de Gestión de Entidades CEDRUX, para perfeccionar la gestión de operaciones de los Centros de balance en las entidades del país.

Objetivos específicos:

1. Fundamentar la investigación mediante la elaboración del Marco Teórico.

2. Describir las actividades que requieren informatización mediante el modelado de procesos de negocio.

3. Identificar los requisitos necesarios para el funcionamiento del sistema.

4. Realizar el Diseño y la Implementación de los componentes del módulo Centro de balance.

Idea a defender: Si se desarrolla el Módulo Centro de balance del subsistema Planificación del Sistema Integral de Gestión de Entidades CEDRUX se perfeccionará la gestión de operaciones de los Centros de balance en las entidades del país.

Tareas para cumplir los objetivos:

1. Estudio del estado del arte referente a soluciones informáticas asociadas al área de conocimientos de Centro de balance de la Planificación.

2. Estudio de la Arquitectura del sistema para conocer sus características fundamentales; marco de trabajo, herramientas y tecnologías definidas para el desarrollo.

3. Descripción de los procesos relacionados con la gestión de las operaciones de los Centro de balance.

4. Captura y especificación de los requisitos que corresponden a los procesos de gestión de las operaciones de los Centros de balance.

5. Diseño de los componentes del módulo Centro de balance.

6. Implementación los componentes del módulo Centro de balance.

7. Validación del diseño de los componentes del módulo Centro de balance a través de métricas.

8. Realización de pruebas de unidad a la implementación de los componentes del módulo Centro de balance.

La investigación del presente trabajo de Diploma está sustentada en métodos científicos de corte teórico y empírico. El método teórico empleado para la investigación fue el Analítico-Sintético, que permite analizar la teoría y documentos existentes, para posteriormente realizar la extracción de los elementos más importantes que se relacionan con el objeto de estudio, aplicado en este caso a los procesos de la gestión de operaciones de los Centros de balance, permitiendo esto una mayor concepción de las tareas a resolver.

El método empírico que se utiliza es el método de la observación, con la aplicación del mismo se puede conocer la realidad mediante la percepción directa de los objetos y fenómenos; a través del cual se pudo conocer la esencia de la problemática definida, lo que ayudó al planteamiento del problema científico, además de permitir conocer el proceso definido como objeto de estudio.

Estructura del documento

Capítulo 1: En este capítulo se expone el estado del arte referente al proceso Gestión de Centro de balance de la Planificación en Cuba y el resto del mundo, se realiza la fundamentación teórica del tema. Se mencionan y describen algunos sistemas existentes. Se hace un estudio del Modelo de desarrollo propuesto y de la Arquitectura base definida para el sistema. Por último, se justifican las herramientas y tecnologías a utilizar para el diseño e implementación de los componentes del módulo.

Capítulo 2: En este capítulo se lleva a cabo la modelación de los procesos relacionados con la Gestión de Centro de balance de la Planificación, para lo cual se elabora Modelo conceptual y se realiza el Diagrama de procesos de negocio. Se determinan y especifican los requisitos tanto funcionales como no funcionales con los que debe cumplir el sistema para su correcto funcionamiento y por último se valida la especificación de requisitos.

Capítulo 3: En este capítulo de se elabora el diseño a partir de la Especificación de Requisitos del proceso Gestión de Centro de balance de la Planificación como base para la implementación del mismo. Está orientado a la construcción de la solución en términos de componentes y diagramas de clases de diseño, según la Arquitectura base definida, enfocada al componente y los requerimientos del sistema. Se plantean estándares de la implementación. Se definen las clases que van a contener las funcionalidades del componente, se especifica la estructura de datos a utilizar, son descritas las clases y los algoritmos implementados. Se valida la solución propuesta a través de la realización de pruebas, el mismo se encuentra dividido en dos partes: la explicación detallada de las pruebas de caja blanca que fueron realizadas al código del software y las pruebas de caja negra que se realizaron a la interfaz del mismo.

Posibles resultados

El módulo Centro de balance del subsistema Planificación del Sistema Integral de Gestión de Entidades CEDRUX, de forma tal que cumpla con los requisitos descritos y se integre con los Subsistemas que son necesarios para su correcto funcionamiento.

Capítulo #1: Fundamentación teórica

1.1 Introducción

En este capítulo se muestran los conceptos fundamentales vinculados al problema científico planteado anteriormente, se hace referencia además a algunos elementos importantes relacionados con el proceso de gestión de las operaciones de los Centros de balance. Se caracterizan algunos de los sistemas informáticos que han sido implementados nacional e internacionalmente que comprenden la Planificación material y financiera en las entidades. Se valoran las tendencias actuales en el contexto informático. Se explica el Modelo de desarrollo elaborado por la dirección del proyecto a partir del cual estará orientada la solución y de igual manera se especifican las tecnologías y herramientas a utilizar como elementos clave para el desarrollo de la propuesta de solución a fin de cumplir los requisitos tanto técnicos como funcionales.

1.2 Conceptos relacionados con el problema

En este punto se exponen los conceptos fundamentales afines con el problema en cuestión como factor importante para lograr una mejor comprensión del mismo.

1.2.1 ¿Qué son los ERP?

Los sistemas de Planificación de Recursos de la Empresa (ERP, por sus siglas en inglés, Enterprise Resources Planning), son sistemas integrales de gestión para la empresa. Los mismos están compuestos por diversos módulos que controlan la información de la producción, ventas, compras, logística, contabilidad, recursos humanos, inventarios, marketing, mantenimiento, entre otras los cuales se integran en una sola aplicación, mediante procesos transparentes y en tiempo real en bases de datos relacionales y centralizados. (1)

1.2.2 Planificación

La planificación es un proceso gradual en el que se establecen los esfuerzos necesarios para cumplir los objetivos planteados, de manera que conlleva un estudio y selección del mejor curso de acción a seguir, frente a una variedad de alternativas posibles y factibles de acuerdo a los recursos disponibles. La planificación es además la concepción anticipada de una actividad de acuerdo a una evaluación racional entre fines y medios. (2)

Planificar constituye la base para una mayor organización y control de los recursos, basados en el

principio de eliminar o mitigar al máximo los riesgos, las desventajas y lograr una mayor producción en un determinado período de acuerdo a los recursos con los que cuenta la empresa.

Tipos de Planificación:

La planificación estratégica, es aquella en la que se establecen los objetivos, las estrategias y los planes globales a largo plazo, normalmente son entre 3 y 5 años. Esta actividad es desarrollada por la alta Dirección, que se ocupa de problemas de gran amplitud, tanto en términos de actividad organizativa como de tiempo.

La planificación operativa, es aquella donde se concretan los planes estratégicos y objetivos a un elevado grado de detalles. Así se establecen las tareas a desarrollar para que se cumplan los objetivos y planes a largo plazo. En esa etapa las actividades son un poco más limitadas y oscilan entre un año y 18 meses.

La planificación adaptativa, pretende eliminar las posibles divergencias entre los resultados y los objetivos relacionados con ellos.

Algunos especialistas consideran un nivel intermedio entra la planificación estratégica y la operativa, y que se denomina planificación táctica o de medio plazo. Esta comparte algunas características de cada una de ellas y su misión es conectarlas.

1.2.3 Planificación material y financiera

✓ Planificación de requerimiento de materiales

La planificación de requerimiento de materiales no es más que un conjunto de técnicas basadas en el Plan Maestro de Producción (PMP)[1] de la empresa, donde se tiene en cuenta la información de registro de inventario o estado del inventario[2], que recoge las cantidades de cada una de las referencias que están disponibles o en curso de fabricación y los documentos de estructura de los productos, con el propósito de generar cronogramas detallados en los que se identifican los artículos

[1] Plan maestro detallado de producción, que en base a los pedidos de los clientes y los pronósticos de demanda, determina los productos finales que se deben fabricar y en qué plazos debe tenerse terminados.
[2] Recoge las cantidades de cada una de las referencias de la planta que están disponibles o en curso de fabricación.

específicos, la lista de materiales (BOM)[3] requeridos para producirlos y así determinar las fechas según los plazos de entrega.

✓ Planificación financiera

La planificación financiera consiste específicamente en alcanzar los objetivos planteados por la dirección de la empresa, haciendo un manejo adecuado de las finanzas, para ello se hace necesario la elaboración de un plan en el que se detalla y describe la estrategia a seguir por la empresa y además se hacen previsiones basadas en los diferentes estados contables y financieros de la misma. (3)

1.2.4 Centro de balance

La gestión de las operaciones de los Centros de balance es un proceso llevado a cabo dentro de la Planificación material y financiera, el mismo consiste en realizar un análisis de la información relacionada con cada una de las demandas realizadas por las diferentes entidades que se comportan como órganos demandantes, compararla con lo que se tiene en inventario, los productos pendientes a arribo y lo pactado a través de contratos nacionales, teniendo en cuenta además la necesidad del año actual; siendo esta la diferencia de la planificación del presente año con lo que se ha consumido del inventario del año en curso. Luego se emite un reporte donde se refleja el resultado del balance al Consejo de ministros, siendo este el que tiene la potestad para tomar la decisión más factible. Su objetivo fundamental es lograr una mayor organización y control de los recursos utilizados, para dar cumplimiento a los objetivos trazados por la dirección de la empresa en la fecha fijada.

1.3 Sistemas informáticos vinculados a la Planificación material y financiera

1.3.1 Openbravo ERP

Es un software libre desarrollado por la Universidad de Navarra en España, este posee el módulo de Gestión Económica-Financiera que entre sus funcionalidades tiene la Planificación de las necesidades de aprovisionamiento, además realiza gestión de datos maestros, gestión de aprovisionamientos, gestión de almacenes, gestión de proyectos y servicios, gestión de la producción, gestión comercial y

[3] BOM: Bill of Materials, lista precisa y completa de todos los materiales y componentes que se requieren para la fabricación o montaje del producto final.

de las relaciones con los clientes, finanzas y contabilidad e inteligencia de negocios. Este tiene como desventaja que no incluye la dualidad monetaria, además el módulo que presenta para la planificación, solo lo realiza para proyectos y no para la economía del país. (4)

1.3.2 Open ERP

El programa es Software de licencia libre. Entre sus características están la contabilidad analítica, contabilidad financiera, gestión de almacenes/inventario, gestión de ventas y compras, automatización de tareas, campañas de marketing, ayuda técnica (Helpdesk), y punto de venta, dentro de la construcción misma del software se hace uso intensivo de flujos de trabajo que se puede integrar con los módulos haciendo la modificación de aprobación y en general de cualquier proceso adaptable. El mismo tiene en su contra que fue desarrollado para empresas que siguen una economía capitalista y que su modelo de gestión de procesos es muy diferente al de las empresas cubanas, y tener que incorporar funcionalidades que se adecuen con las características particulares de la economía y la planificación cubana, incurriría en gastos financieros y de recursos humanos aún mayores que los que ocasionaría desarrollar un sistema propio. (5)

1.3.3 SAP R/3

SAP está compuesto por una serie de áreas funcionales o módulos que responden de forma completa y en tiempo real a los procesos operativos de las organizaciones. Comprende cuatro soluciones independientes que brindan soporte a procesos de negocio clave a través de su sistema ERP específico: SAP ERP Finanzas, SAP Administración de Capital Humano, SAP ERP Operaciones y SAP ERP Servicios Generales, SAP ERP Planificación de la Producción (PP). Este último maneja la Planificación de Ventas y Operaciones y Planificación Estratégica de Negocio. Este tiene como desventaja que está Desarrollado con tecnología propietaria, no incluye la dualidad monetaria y que el módulo de Planificación que presenta está dirigido a economías capitalistas, por lo que resulta difícil adaptarlo a las características de la economía cubana, sin tener elevados gastos financieros y de recursos humanos. (6)

1.3.4 VERSAT Sarasola

El VERSAT Sarasola, es un sistema de gestión contable-financiero que representa un ejemplo de sustitución de importaciones en materia de aplicaciones informáticas. Esta herramienta utilizada para la planificación económica, el control y el análisis de gestión incluye 12 módulos: Configuración, Contabilidad general, control de inventarios, Generador de reportes. Control de activos fijos, Costos y

procesos, Finanzas, caja y banco, Contratación y facturación, Planificación económico-productiva, Análisis económico empresarial, Paquetes de gestión, Nóminas de salario. En su conjunto es un sistema integral, que se actualiza constantemente en función de nuevas demandas y viabiliza, sin dudas, la organización y el control en el área económica. Este tiene como desventaja que el módulo de Planificación se ha concebido con una orientación a la esfera presupuestada, aunque no deja de tener posibilidades de generalización en el sector empresarial, además no permite la dualidad de moneda. (7)

1.4 Valoración del estado del arte

Dada la situación económica y financiera en la que se encuentra Cuba en la actualidad, se ha trazado como meta la sustitución de las importaciones de software y los pagos de licencias por concepto de importación, con este objetivo se planteó la necesidad de la creación de aplicaciones nacionales.

A partir del estudio de los sistemas de gestión empresarial se evidencia la no existencia de un software que responda íntegramente a lo que en realidad necesita la economía cubana, presentando diferentes inconvenientes ya sea en la de gestión de las operaciones de los Centros de balance, por las herramientas sobre las que están soportadas, por las licencias de dichos productos o bien porque no son capaces de cumplir con los requisitos establecidos a nivel nacional

Teniendo en cuenta los elementos dados se hace necesaria la creación en el país de una solución informática que satisfaga las necesidades económicas actuales, reduciendo además el tiempo y los recursos financieros en las distintas entidades.

1.5 Tendencias y tecnologías actuales

La industria del software ha mostrado ser una de las áreas más dinámicas y con mayor crecimiento en los últimos años. La evolución hacia a un modelo más racional para los usuarios, con menos costes de licencia, donde se intensifique la prestación de servicios, que reduzca el tiempo de desarrollo e incremente la calidad, viene siendo lo más importante a la hora de desarrollar cualquier tipo de aplicación. A continuación se ofrecerá una valoración sobre algunas de las tendencias y tecnologías que marcan un nivel alto en el mundo del software y que bien contribuye a lo dicho anteriormente:

- ✓ Aplicación Web

Las aplicaciones web son sistemas informáticos que han ganado gran popularidad en la comunidad internauta, debido a las facilidades y funcionalidades que estas ofrecen, siendo las más relevantes:

Compatibilidad Multiplataforma: una misma versión de la aplicación puede correr sin problemas en múltiples plataformas como Windows, Linux, Mac, etc.

Actualización: las aplicaciones web siempre se mantienen actualizadas y no requieren que el usuario deba descargar actualizaciones y realizar tareas de instalación.

Acceso inmediato y desde cualquier lugar: las aplicaciones basadas en tecnologías web no necesitan ser descargadas, instaladas y configuradas. Además pueden ser accedidas desde cualquier computadora conectada a la red en donde se accede a la aplicación.

Menos requerimientos de hardware: este tipo de aplicación no consume (o consume muy poco) espacio en disco y también es mínimo el consumo de memoria RAM en comparación con los programas instalados localmente. Tampoco es necesario disponer de computadoras con poderosos procesadores ya que la mayor parte del trabajo se realiza en el servidor en donde reside la aplicación.

Seguridad en los datos: los datos se alojan en servidores con sistemas de almacenamiento altamente fiables y se ven libres de problemas que comúnmente sufren los ordenadores de usuarios comunes como virus y roturas de disco. (8)

✓ Arquitectura Cliente/Servidor

La arquitectura Cliente/Servidor es una nueva tendencia en el desarrollo de redes, que tiene como objetivo optimizar el uso tanto del hardware como del software, a través de la separación de funciones: el cliente, quien inicia una determinada petición y el servidor, dedicado a responder dichas peticiones.

Puede presentarse como uno o varios clientes y servidores, junto con un sistema operativo y una plataforma de comunicación para formar un sistema cooperativo que permita la computación distribuida, el análisis y la presentación de datos. Un único servidor típicamente sirve a una multitud de clientes, ahorrando a cada uno de ellos el problema de tener la información almacenada localmente. Características de la arquitectura Cliente/Servidor:

El servidor presenta una interfaz única y bien definida a todos sus clientes.

El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.

El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.

Los cambios en el servidor no afectan al cliente.

✓ Software Libre

El término de software libre se le asigna a las aplicaciones informáticas que están libremente disponibles bajo un acuerdo de licencia pública de manera que cualquiera puede adaptarlos y mejorarlos.

Cada vez son más las organizaciones de toda clase que optan por soluciones libres para sus necesidades informáticas, porque los productos libres son generalmente más baratos, más fiables y más fáciles de reparar ante fallas. El software libre ayuda a reducir los costos operativos de las empresas, produce suficientes beneficios en las actividades empresariales como para utilizarlo en aplicaciones determinantes para el negocio, en áreas claves como el control de ingresos y la mejora del servicio a los clientes. Produce además un aumento de la flexibilidad de negocio, el hecho de poder utilizarlo sin pagar una licencia lo convierte en un producto atractivo, la compatibilidad con estándares abiertos, el uso de código sin restricciones permite no quedarse restringidos a un único proveedor.

1.6 Modelo de desarrollo

Para llevar a cabo el desarrollo del sistema CEDRUX, teniendo en cuenta la magnitud del mismo, fue necesario establecer por cada uno de los grupos de trabajo un estándar, así como, una definición precisa de las responsabilidades que implican los diferentes roles que se toman en el desarrollo de la solución.

La propuesta de modelo de desarrollo que se presenta a continuación fue elaborada por el equipo de producción en colaboración con cada una de las Líneas de desarrollo involucradas en el proyecto ERP-Cuba, en correspondencia con las necesidades que presentan y además se tuvieron en cuenta los riesgos fundamentales identificados en el proyecto. (9)

Es un vehículo fundamental para iniciativas informáticas claves, como la modernización de los entornos de aplicación empresariales.

1.6.1 Características del Modelo de desarrollo

✓ Centrado en la arquitectura

La arquitectura determina la línea base y los elementos de software estructurales a partir de los elementos de la arquitectura de negocio. Interviene en la gestión de cambios y diseña la evolución e integración del producto. La arquitectura orienta las prioridades en la producción y resuelve las necesidades tecnológicas y de soporte para el desarrollo.

✓ Orientado a componentes:

Las iteraciones son orientadas según la significación arquitectónica de los componentes, los mismos son abstracciones arquitectónicas de los procesos de negocio y requisitos asociados que modelan, el componente es la unidad de medición y ordenamiento de las iteraciones.

✓ Iterativo e incremental:

Las iteraciones son planificadas y coordinadas con el equipo de arquitectura, los clientes y la alta gerencia. Cada iteración constituye el desarrollo de componentes, los cuales son integrados al término de la iteración, permitiendo de esta manera la evolución incremental del producto.

✓ Ágil y adaptable al cambio:

El desarrollo de las partes formaliza solamente las características principales de la solución, priorizando los talleres y las comunicaciones entre las personas. Los clientes y funcionales están involucrados en el proyecto y poseen parte de la responsabilidad del éxito del mismo. Los cambios son conciliados semanalmente, discutidos y aprobados.

1.7 Arquitectura base

La arquitectura de software es la organización fundamental de un sistema, representada en sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución. Establece los fundamentos para que analistas, diseñadores, programadores, etc. trabajen en una línea común que permita alcanzar los objetivos del sistema, cubriendo todas las necesidades.

Para el desarrollo de la solución se decidió adoptar la propuesta de Arquitectura base definida por la línea de arquitectura del proyecto ERP-Cuba. (10)

Dicha arquitectura está basada en componentes, es además completamente modular y favorece la

reutilización de todos sus elementos, incluyendo los que definen las distintas relaciones entre ellos y tiene como objetivo hacer un uso correcto de software reutilizable, para la construcción de aplicaciones de software mediante el ensamblaje de partes ya existentes.

Un componente por su parte es un fragmento reemplazable de un sistema de software, una unidad de composición con interfaces especificadas contractualmente, que satisface una o varias funcionalidades dentro del contexto de una arquitectura bien definida y puede ser ensamblado con otros fragmentos por medio de una interfaz. Un componente puede contener múltiples objetos, clases y otros componentes. (11)

Dicha arquitectura además tiene una vista de integración entre dichos componentes, donde se establecen las definiciones, estándares, protocolos de comunicación y reglas de intercambio de información. Los componentes pueden lo mismo prestar o consumir servicios, estableciendo una interacción entre ellos, a través del motor de integración, que se lleva a cabo mediante el patrón inversión de control (IOC), el cual forma parte del Zend framework.

El empleo del patrón arquitectónico Modelo-Vista-Controlador o MVC que describe una forma de organizar el código de una aplicación constituye otra de las características de la arquitectura del sistema. Este patrón por su parte propone separar los datos, la interfaz de usuario, y la lógica de control en tres componentes distintos. Los componentes son:

- Modelo: Es el componente encargado del acceso a datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento. Debe dar respuestas a las peticiones del controlador.

- Vista: Define la interfaz de usuario, se encarga de mostrar datos del modelo al usuario. Debe informar al controlador los eventos generados por el usuario.

- Controlador: Debe encargarse de responder a los eventos generados por el usuario y modificar la vista y el modelo.

Es además importante resaltar que la arquitectura definida es híbrida lo que permite la reutilización de código en la programación, la definición de un estándar de desarrollo y aprovechamiento de las horas hombres en función del esfuerzo, mostrando así diversas ventajas para su uso que se mencionan a continuación:

- Desarrollos paralelos: en cada capa.

- Aplicaciones más robustas debido al encapsulamiento.
- Mantenimiento y soporte más sencillo: es más sencillo cambiar un componente que modificar íntegramente una aplicación.
- Mayor flexibilidad: se pueden añadir nuevos módulos para dotar al sistema de nueva funcionalidad.
- Alta escalabilidad: La principal ventaja de una aplicación distribuida y bien diseñada es su buena escalabilidad; es decir, que puede manejar muchas peticiones con el mismo rendimiento simplemente añadiendo más hardware.

1.8 Patrones

¿Qué es un patrón?

Pareja de problema / solución con un nombre, que codifica (estandariza) buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades.

Formato de un patrón.

- Nombre
- Solución
- Problema
- Explicación
- Ejemplo de utilización

1.8.1 Los patrones de casos de uso:

Antes de comenzar el estudio dedicado a los patrones de casos de uso, es necesario aclarar que aunque se haga un análisis general de varios de ellos, los que se decidan utilizar serán adaptados para aplicarlos a los requisitos.

Nombres que revelan la intención (IntentionRevealingName)

El nombre debe reflejar la intención del caso de uso y reflejar un único objetivo e intención que el actor está intentando lograr. Se debe asignar un nombre apropiado que facilite el manejo del caso de uso,

permitiendo tener una vista general del trabajo en su conjunto.

Preciso y Legible (PreciseAndReadable)

Cada caso de uso que se escriba debe exactamente describir una Meta única de manera que la audiencia pueda leerlo y que comunique la suficiente información para su adecuado entendimiento. Los niveles más altos de formalidad en las especificaciones dan a los desarrolladores un sentido falso de seguridad. Nada puede remplazar el diálogo con el cliente.

Escenario más Fragmentos (ScenarioPlusFragments)

El flujo principal debe describir cómo el actor primario logra su meta de una manera honesta. No tiene que ser el posible camino más corto o el único camino exitoso, pero debe ser el normal que deseó para alcanzar la meta, es decir, el que tienden a seguir los usuarios.

Completar una única meta

Este patrón plantea que cada caso de uso debe ser descrito con un objetivo bien definido. Se debe ser consistente entre las metas que se describen.

Condiciones detectables (DetectableConditions)

Un sistema no puede manejar eventos que no pueda detectar, los desarrolladores necesitan conocer que situaciones detectar. Se debe capturar cada posibilidad razonable, de otra manera, el sistema nunca estará capacitado para ejecutar el escenario. Descubrir una condición olvidada después de que el sistema ha introducido servicios es incluso más caro.

Alternativas Exhaustivas, Integras (ExhaustiveAlternatives)

Este patrón establece que un caso de uso puede tener varias alternativas, identificándose en cada caso el flujo normal de eventos y capturando los posibles fallos.

Pasos Nivelados (LeveledSteps)

Pasos excesivamente pequeños hacen el caso de uso largo, difícil de leer y bloquean la visión.

Pasos excesivamente largos pueden enterrar comportamientos importantes. Lo contrario ocasionalmente pasa, que el escritor escribe en un muy alto nivel de abstracción y hace largo el salto en la narrativa, omitiendo acciones claves que los desarrolladores deben saber.

Mezclar niveles de detalle en un escenario es entretener. Ocasionalmente se deben escribir pasos continuos a diferentes niveles de abstracción. Demasiado de esto distrae al lector de lo que se supone que está pasando y le hace difícil la interpretación correcta de las instrucciones.

Adorno, Decoración (Adornments)

Este patrón plantea que el usuario a la hora de leer un caso de uso debe entender cómo el sistema entrega los valores sin preocuparse de detalles de la interfaz de usuario. La idea es crear campos dentro de la plantilla del caso de uso que fuera del texto del escenario apoyen la información auxiliar que es útil asociar con el caso de uso.

1.8.2 Patrones de especificación de requisitos.

Recoger y especificar requisitos no es una actividad sencilla, requiere de técnicas que ayuden a mejorar los modelos clásicos de especificación. Algunos de los patrones de especificación de requisitos son:

Clasificación

Este patrón permite clasificar las especificaciones de forma similar a como se clasifican los requisitos. Pueden ser: puramente textuales y se clasifican como reglas de negocio, restricciones tecnológicas o de diseño, lista de funcionalidades. También las hay gráficas basadas en modelo UML que recogen casos de uso, entidades del dominio o aspectos de navegación de interfaz de usuario.

Derivación

A partir de una determinada necesidad del cliente, se puede obtener una especificación escribiendo o modelando de manera formal lo que dicha necesidad sugiere. No sólo se identifica más rápido y mejor lo que dice el cliente, sino que se transmite mejor al equipo de desarrollo.

Por ejemplo, una regla impuesta por el usuario puede derivarse en una precondición, una especificación; o un cambio en la navegación puede derivarse directamente en una nueva versión de nuestro modelo de interfaz de usuario.

1.8.3 Patrones de Diseño:

El establecimiento de estos patrones comunes es lo que posibilita el aprovechamiento de la experiencia acumulada en el diseño de aplicaciones. (10)

Para hacer un diseño eficiente se tomaron en cuenta un conjunto de patrones, que al ser experiencias de diseñadores expertos, permiten dar solución a problemas de diseño facilitando notablemente el trabajo posterior. Además, se debe tener en cuenta que los marcos de trabajo que se han utilizado llevan implícito la aplicación de gran cantidad de patrones. Aunque el presente trabajo de diploma, no persigue como objetivo un estudio minucioso de los patrones de diseño, a continuación se exponen los patrones de asignación de responsabilidades GRASP (General Responsibility Assignment Software Patterns), entre ellos, los patrones Experto, Controlador, Creador, Bajo acoplamiento y Alta Cohesión:

El patrón **Experto** plantea que se debe asignar la responsabilidad al experto en información, que es la clase que cuenta con la información necesaria para cumplir la responsabilidad.

El patrón **Controlador** establece que se debe asignar la responsabilidad del manejo de los eventos de un sistema a una clase controladora.

El patrón **Creador** ayuda a identificar quién debe ser el responsable de la instanciación o creación de nuevas clases u objetos. La clase podrá crear la nueva instancia si y sólo si tiene en cuenta al menos uno de los siguientes criterios:

- Tiene la información necesaria.
- Usa directamente las instancias creadas del objeto.
- Almacena o maneja varias instancias de la clase.
- Contiene o agrega la clase.

La visibilidad entre la clase creada y la clase creadora es una de las facilidades que se deriva del uso del patrón y además conduce a un bajo acoplamiento, lo cual supone facilidad de mantenimiento y reutilización así como mayor claridad.

El patrón **Bajo Acoplamiento** persigue tener las clases lo menos ligadas entre sí que se pueda. De esta manera, en caso de producirse una modificación en alguna de ellas, se tiene la mínima repercusión posible en el resto de las clases. Con este patrón se potencia la reutilización y se disminuye la dependencia entre las clases. En la perspectiva del diseño orientado a objetos, la cohesión es una medida de cuán relacionadas y enfocadas estén las responsabilidades de una clase.

La aplicación del patrón **Alta Cohesión** facilita un diseño donde las clases tienen responsabilidades

estrechamente relacionadas y no realizan un trabajo enorme.

1.9 Lenguajes de Modelado y Desarrollo

1.9.1 Lenguaje de Modelado

✓ UML

UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos involucrados en un sistema. Este modelado visual es independiente del lenguaje de implementación, los diseños realizados usando UML se pueden implementar en cualquier lenguaje orientado a objetos. Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos. (12)

El vocabulario de UML incluye tres clases de bloques de construcción: elementos, relaciones y diagramas. Los elementos son abstracciones fundamentales de un modelo; las relaciones ligan estos elementos entre sí y los diagramas agrupan colecciones de elementos. (13) Dada estas condiciones que agrupa es el lenguaje utilizado para realizar el diseño del presente trabajo de diploma. Se empleará UML para la representación de las clases y el Modelo de datos en el Diseño, y para mostrar en la Implementación tanto la interacción entre los componentes como el Diagrama de despliegue.

✓ Notación para el Modelado de Procesos de Negocio BPMN

La Notación para el Modelado de Procesos de Negocio -Business Process Modeling Notation (BPMN)- es una notación gráfica estandarizada que permite el modelado de procesos de negocio, la cual fue utilizada para llevar a cabo la modelación del proceso del negocio del balance material del plan. (14)

Su principal objetivo es proveer una notación estándar que sea fácilmente legible y entendible por parte de todos los involucrados e interesados del negocio. Entre estos interesados están los analistas de negocio, los desarrolladores, técnicos y administradores del negocio. El modelado en BPMN se realiza mediante diagramas muy simples con un conjunto de elementos gráficos. Con esto se busca que para los usuarios del negocio y los desarrolladores técnicos sea fácil entender el flujo y el proceso. Se empleará BPMN para modelar los procesos que corresponden a la gestión de operaciones de los Centro de balance.

1.9.2 Lenguajes de programación

✓ Lenguaje del lado del servidor

PHP

PHP Hypertext Pre-processor es software libre, puede ser utilizado en cualquiera de los principales sistemas operativos del mercado, soporta la mayoría de servidores Web en la actualidad y ofrece soporte para unos 20 gestores de bases de datos. Su característica de ser software libre trae como consecuencia que implique menos costes y servidores más baratos que otras alternativas lo que lo convierten en la herramienta ideal para la creación de páginas Web dinámicas. Es además muy rápido, contiene una biblioteca nativa de funciones sumamente amplia e incluida, no requiere definición de tipos de variables ni manejo detallado del bajo nivel, presenta mejoras de rendimiento, es un lenguaje multiplataforma que funciona en todas las plataformas que soporten Apache. Es perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes. Se empleará PHP en su versión 5.2 o superior.

✓ Lenguajes del lado del cliente

Un lenguaje del lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio. Pero la página no se verá bien si el ordenador cliente no tiene instalados los plug-in adecuados. El código, tanto del hipertexto como de los scripts, es accesible a cualquiera y ello puede afectar a la seguridad.

XML

Sigla en inglés de Extensible Markup Language (lenguaje de marcas ampliable), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML. XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades, el mismo no solo es aplicado en la esfera de Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y otros.

JavaScript

JavaScript es un lenguaje interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas Web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C. Al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de Herencia, es más bien un lenguaje basado en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad. Todos los navegadores interpretan el código JavaScript integrado dentro de las páginas Web. Para interactuar con una página

Web se provee al lenguaje JavaScript de una implementación del DOM. JavaScript es un lenguaje con muchas posibilidades, utilizado para crear pequeños programas que luego son insertados en una página Web y en programas más grandes, orientados a objetos mucho más complejos. Con JavaScript se puede crear diferentes efectos e interactuar con los usuarios. JavaScript es soportado por la mayoría de los navegadores como Internet Explorer, Netscape, Opera, Mozilla Firefox, entre otros. (15)

CSS

Es un lenguaje de hojas de estilos (en inglés Cascading Style Sheets) creado para controlar la presentación de documentos estructurados y escritos en HTML y XHTML, aspectos como: el color, el tamaño, el tipo de letra, la separación entre párrafos y la tabulación con la que se muestran los elementos de una lista. El propósito del desarrollo de CSS es separar la estructura y el contenido de la presentación estética en un documento, esto permite un control mayor del documento y sus atributos, convirtiendo al HTML en un documento muy versátil y liviano. “Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo (también llamados "documentos semánticos"). Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.” (16) Entre los beneficios concretos de CSS se encuentran:

1. Control de la presentación de muchos documentos desde una única hoja de estilo.
2. Control más preciso de la presentación.
3. Aplicación de diferentes presentaciones a diferentes tipos de medios (pantalla, impresión, entre otros.) (17)

1.10 Frameworks

Framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un Framework 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.

1.10.1 EXTJS

Es un framework JavaScript del lado del cliente para el desarrollo de aplicaciones Web. Tiene un

sistema dual de licencia: Comercial y Open Source. Este framework puede correr en cualquier plataforma que pueda procesar POST y devolver datos estructurados (PHP, Java, .NET y algunas otras) en tiempo de ejecución carga y crea todos los objetos HTML a través del uso intenso de DOM. Los datos son obtenidos mediante mensajes AJAX a través de XML y/o JSON.

Funcionalidades: Dispone de un conjunto de componentes para incluir dentro de una aplicación web, como:

1. Cuadros y áreas de texto.
2. Campos para fechas.
3. Combos.
4. Radiobuttons y checkboxs.
5. Editor HTML.
6. Elementos de datos (con modos de sólo lectura, datos ordenables, columnas que se pueden bloquear y arrastrar, etc.).
7. Barra de herramientas.
8. También contiene numerosas funcionalidades que permiten añadir interactividad a las páginas HTML, como: Cuadros de diálogo.

Se empleará la versión 2.2.

1.10.2 Zend

Zend Frameworks es un frameworks open source, que está diseñado para PHP 5 y buenas capacidades de ampliación. (18) Presenta como características esenciales un sistema de caché dividido en frontend y backend, de forma que se puedan almacenar en caché diferentes datos como resultados de funciones, páginas completas, etc., y que esta información se almacene en archivos, en memoria, en base de datos, etc. Simplifica la gestión de archivos de configuración y proporciona los componentes que forma la infraestructura del patrón MVC. Se empleará la versión 1.9.7.

1.10.3 Doctrine

Doctrine es un potente y completo sistema ORM (en inglés Object Relational Mapper) para PHP 5.2+ que incorpora una DBL (capa de abstracción a base de datos). Uno de sus rasgos importantes es la

habilidad de escribir opcionalmente las preguntas de la base de datos orientado a objeto. Esto les proporciona una alternativa poderosa a diseñadores de SQL que mantiene un máximo de flexibilidad sin requerir la duplicación del código innecesario. También permite exportar una base de datos existente a sus clases correspondientes y también convierte clases (convenientemente creadas siguiendo las pautas del ORM) a tablas de una base de datos. Se empleará la versión 1.2.2.

1.11 Tecnologías y herramientas de desarrollo

1.11.1 Tecnología AJAX

AJAX acrónimo de Asynchronous JavaScript And XML (en español JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (en inglés Rich Internet Applications). (19) Estas aplicaciones se ejecutan en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones. JavaScript es el lenguaje interpretado (scripting lenguaje) en el que normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante XMLHttpRequest, objeto disponible en los navegadores actuales.

1.11.2 Herramientas CASE

“CASE es una sigla, que corresponde a las iniciales de: Computer Aided Software Engineering; y en su traducción al Español significa Ingeniería de Software Asistida por Computación. Las herramientas CASE representan una forma que permite Modelar los Procesos de Negocios. Un elemento importante conveniente de destacar, es que las herramientas CASE, son eso: "HERRAMIENTAS", y que como tales permiten aumentar la productividad en el desarrollo de un proyecto y como herramientas que son, deben ser aplicadas a una metodología determinada. (20)

- Visual Paradigm

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 rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. (21) Se empleará la versión 6.4.

1.11.3 Herramientas de desarrollo colaborativo:

- Control de versiones

Una versión, revisión o edición de un producto, es el estado en que se encuentra en un momento dado en su desarrollo o modificación. Se llama control de versiones a la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, así como las posibles especializaciones realizadas. Todos los sistemas de control de versiones se basan en disponer de un repositorio, que es el conjunto de información gestionada por el sistema. Este repositorio contiene el historial de versiones de todos los elementos gestionados.

Específicamente se empleará para el desarrollo Sub-versión; software de sistema de control de versiones diseñado específicamente para reemplazar al popular CVS, el cual posee varias deficiencias. Es software libre bajo una licencia de tipo Apache/BSD y se le conoce también como svn por ser ese el nombre de la herramienta de línea de comandos. Una característica importante de Subversión es que, a diferencia de CVS, los archivos versionados no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo. Se estará haciendo uso de la versión 1.4.5.

1.11.4. Desarrollo integrado de desarrollo:

- Zend Estudio para Eclipse

Zend Studio para Eclipse combina la probada tecnología y desarrolladores de PHP de Eclipse Tools (PDT) proyecto para crear el más poderoso IDE para el desarrollo de ricas aplicaciones Web. Zend Studio para Eclipse está diseñado para profesionales que necesitan los desarrolladores de PHP para apoyar el ciclo de vida de toda la aplicación PHP y quieren tomar ventaja de la sofisticación y la extensibilidad del marco de Eclipse y de los ecosistemas. (22) Esta herramienta presenta entre otras las siguientes características:

1. Código refactorización.
2. Nueva generación de código del archivo y magos.
3. Código de Cobertura.
4. PHP Unit pruebas de apoyo.

5. Mejora con PHP Editor avanzado de formato, las nuevas listas de tareas y problemas de vista.
6. Mejora de soporte JavaScript.
7. Acceso al ecosistema de plug-ins de Eclipse.
8. Apoyar el desarrollo de múltiples idiomas.
9. Zend Studio 5.5 Herramientas de Migración.

Se estará haciendo uso de la versión 3.3.

1.11.5 Servidor de aplicaciones web:

- Servidor web Apache

Apache 2.0 es un servidor Web potente, flexible y disponible para distintas plataformas y entornos. Es altamente configurable de diseño modular, posibilitando que los administradores de sitios Web puedan elegir los módulos que serán incluidos y ejecutados en el servidor. Es una tecnología gratuita y de código abierto, lo que proporciona transparencia en todo el proceso de instalación. Es prácticamente universal, por su disponibilidad en multitud de sistemas operativos. Este servidor Web tiene una fácil integración con varios lenguajes de programación como: Java, Perl y especialmente PHP. Dicha relación a dado a lugar el desarrollo de aplicaciones como el APPSERV y XAMPP los cuales instalan el Apache y el PHP configurados para su uso. Se estará haciendo uso de la versión 2.0.

1.11.6 Sistema Gestor de Base de Datos:

Los sistemas de gestión de base de datos (SGBD); son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

- PostgreSQL

PostgreSQL es un servidor de base de datos relacional, libre. Tiene soporte total para transacciones, disparadores, vistas, procedimientos almacenados, almacenamiento de objetos de gran tamaño. Se destaca en ejecutar consultas complejas, consultas sobre vistas, subconsultas y joins de gran tamaño. Permite la definición de tipos de datos personalizados e incluye un modelo de seguridad completo. Como toda herramienta de software libre PostgreSQL tiene entre otras ventajas las de contar con una gran comunidad de desarrollo en Internet, su código fuente está disponible sin costo alguno y algo muy importante es que dicha herramienta es multiplataforma. Fue diseñado para ambientes de alto volumen. Escala muy bien al aumentar el número de CPUs y la cantidad de RAM. Soporta

transacciones y desde la versión 7.0, claves ajenas con comprobaciones de integridad referencial. Tiene mejor soporte para vistas y procedimientos almacenados en el servidor, además tiene ciertas características orientadas a objetos. Se estará haciendo uso de la versión 8.3.

1.11.7 Navegador:

- Mozilla Firefox

Firefox ha sido creado por el proyecto Mozilla. La misión del proyecto Mozilla es preservar la elección y la innovación en Internet utilizando el open source. El apoyo organizativo del proyecto Mozilla es proporcionado por Mozilla Foundation (en los EEUU), Mozilla Europe y Mozilla Japón. (23) Se estará haciendo uso de la versión 3.5

1.12 Conclusiones del capítulo

Luego de un estudio minucioso de diferentes sistemas tanto nacionales como internacionales encargados de la gestión de recursos empresariales se concluye que no existe un sistema capaz de gestionar las operaciones relacionadas con los Centros de balance como parte del proceso de Planificación en las entidades del país y por lo tanto se pretende desarrollar una aplicación web orientada a resolver satisfactoriamente este inconveniente. Dicha aplicación debe posibilitar un análisis de las demandas realizadas por las entidades, para de esta forma llevar a cabo la consolidación de las mismas y efectuar el balance material. El sistema brindará la posibilidad de que los usuarios puedan consolidar las demandas de cada una de las entidades y del propio Centro de balance, y pueda realizar el balance material en el momento que lo desee. Además contará con una interfaz amigable para el usuario, de manera que le sea fácil el manejo de la información, mostrando en todo momento los datos necesarios para llevar a cabo las operaciones.

Es necesaria la utilización de herramientas y tecnologías libres que hoy día están alcanzando un determinado nivel en la industria de software, que pueden contribuir al desarrollo de aplicaciones robustas enfocadas a un modelo más racional para los usuarios, con menos costes de licencia, que reduzca el tiempo de desarrollo e incremente la calidad y ese sentido se presentó la propuesta elaborada por el equipo de arquitectura del proyecto ERP-Cuba para el desarrollo de la solución.

Capítulo #2: Análisis y Diseño

2.1 Introducción

En el presente capitulo se lleva a cabo, en un primer momento la modelación del negocio, teniendo en cuenta que antes de comenzar a desarrollar un software se hace necesario comprender el negocio mediante el estudio de los procesos que en él se desarrollan, en ese sentido se realiza una descripción del mismo y se definen los procesos implicados. Posteriormente se determina y especifican los requisitos funcionales y no funcionales a partir de las expectativas del cliente y lo que desde el punto de vista operacional se debe tener en cuenta en el sistema y por último se diseña la solución como punto de partida para la implementación.

2.2 Modelación del negocio

La modelación del negocio como fase dentro del proceso de desarrollo del software tiene a cargo la comprensión de la estructura y dinámica de la organización en la cual se va a implantar el sistema. Para lograr este propósito el proceso de modelado permite obtener una visión del negocio mediante la definición, descripción y representación de los procesos.

A continuación se realiza la modelación del negocio asociado a la Gestión de operaciones de los Centros de balance.

2.2.1 Descripción del negocio

El primer paso para la realización del Balance material es generación de las demandas de cada una de las entidades, a partir de un modelo ya definido por los planificadores del Ministerio de Economía y Planificación, este presenta la demanda a la entidad que se comporta como Centro de balance, en el mismo llegan a un acuerdo, el propio CB hace una consolidación de las demandas de sus Órganos demandantes y la envía al ministerio de economía y planificación, donde son recibidas dichas demandas, los encargados de realizar el balance en el Ministerio realizan un análisis de las demandas, donde puede revertir el proceso y mandar a reajustar a las entidades demandantes, o puede aceptarlas, en este caso haría la consolidación de las demandas, y se las entregaría al Consejo de Ministros, este haría un análisis de lo demandado, en ese caso se acepta la demanda o se revierte el proceso, mandando a reajustar al Ministerio de Economía y Planificación y este a cada una de sus entidades demandantes.

2.2.2 Diagrama de Proceso de Negocio

Un modelo de procesos de negocio es un conjunto de objetos gráficos, correspondientes a actividades y controles de flujo que definen el orden de ejecución de éstas.

Seguidamente se muestra el diagrama de proceso de negocio correspondiente al proceso Realizar Balance material:

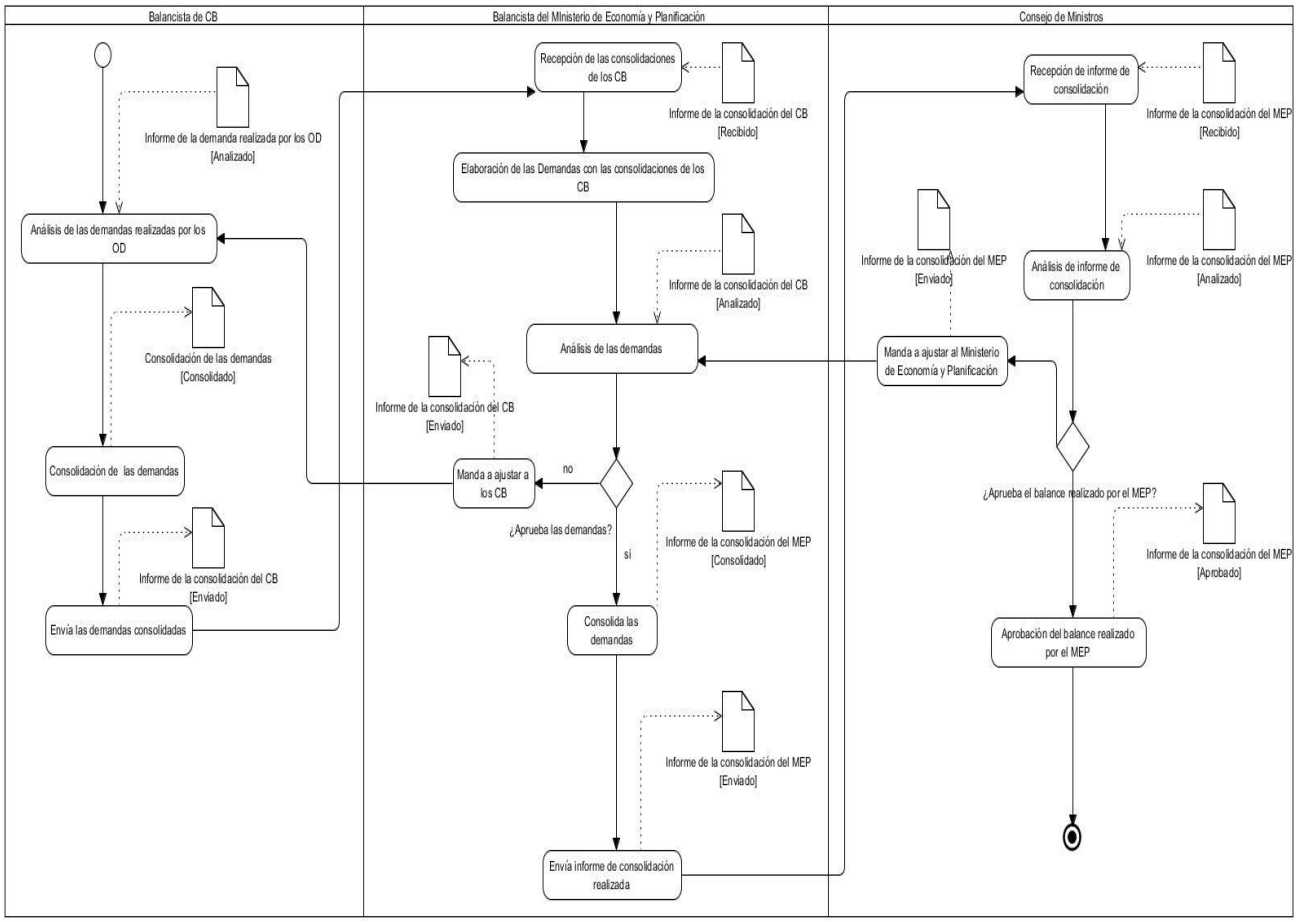

Figura 1: Diagrama del proceso de negocio Realizar Balance material.

2.2.3 Modelo conceptual

El modelo conceptual, permite dominar los principales conceptos con los que se relaciona el módulo a desarrollar y establece las relaciones que existen entre cada uno de ellos. Todos estos conceptos son modelados mediante tablas representando las entidades que lo conforman y sus atributos, estableciendo una representación de los conceptos del dominio del problema.

A continuación se representan los conceptos fundamentales del negocio para un mejor entendimiento del mismo, que servirá como elemento clave para fases posteriores del desarrollo.

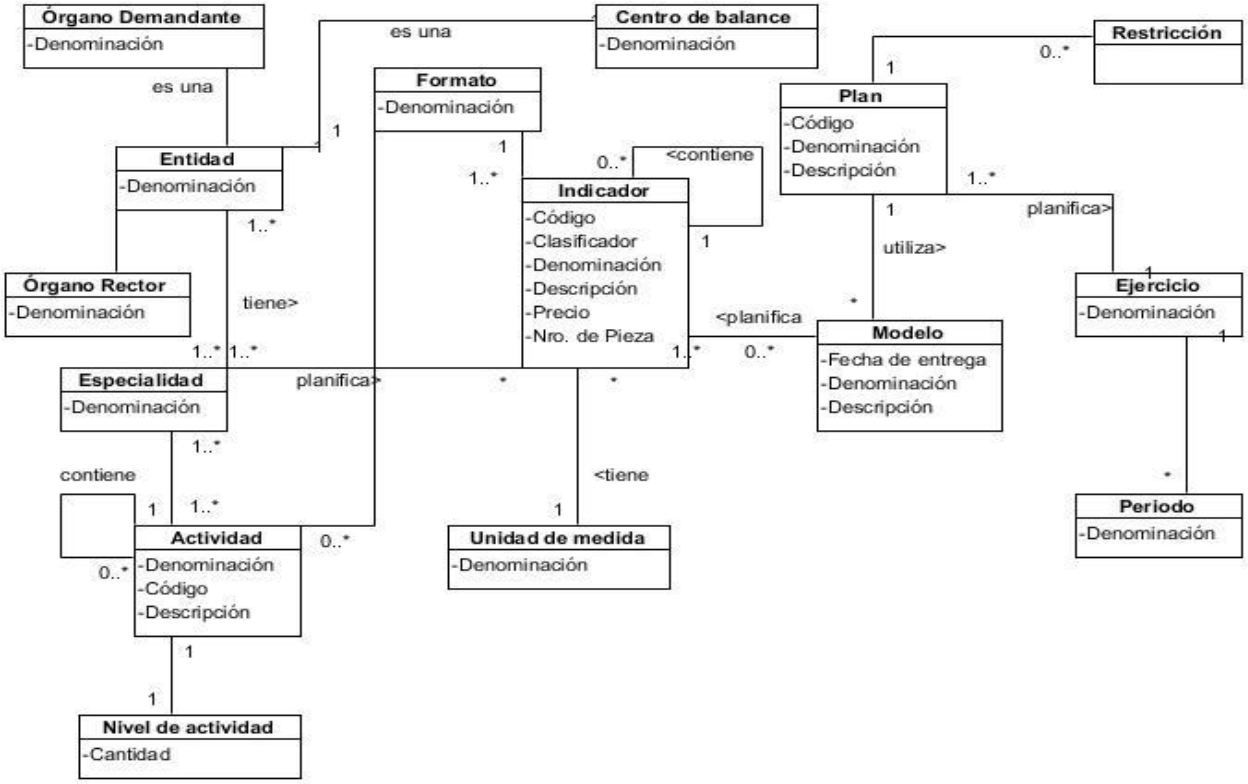

Figura 2: Modelo conceptual

2.2.4 Validación del proceso de negocio

Una vez terminada la identificación y descripción del proceso de negocio se hace imprescindible validarlos para asegurar que el proceso ha sido descrito de manera correcta. En este caso la validación del proceso de negocio Realizar Balance Material fue llevada a cabo mediante un nuevo encuentro con los especialistas funcionales y el analista principal de la línea a modo de Revisión Técnica Formal. En este encuentro se comprobó que la modelación del proceso coincidía con la realización de estas actividades en las diferentes entidades del país.

2.3 Captura de requisitos

La captura de requisitos del sistema está asociada al reconocimiento del problema a partir de cómo lo ve el usuario, a la definición de los datos observables, el contenido de la información y las funciones del software.

2.3.1 Técnicas utilizadas para la captura de requisitos

A continuación se hace referencia a las técnicas empleadas en la actividad de captura de requisitos necesarios para definir las funcionalidades que debe cumplir el sistema de acuerdo con las necesidades del cliente.

- Entrevistas y Cuestionarios

Las entrevistas y cuestionarios se emplean para reunir información proveniente de personas o de grupos. Específicamente esta técnica se aplicó a los diferentes especialistas funcionales tanto del MINFAR como del MEP, para conocer de manera clara y concreta cómo se lleva a cabo la gestión de operaciones de los Centros de balance y a partir de ahí definir las principales actividades a automatizar.

- Sistemas existentes

Esta técnica consiste en analizar distintos sistemas ya desarrollados que están relacionados con el sistema a ser construido. El sistema estudiado fue Contabilidad Financiera al cual se le realizó un análisis a través de interfaces de usuario, observando el tipo de información, cómo esta es manejada en la aplicación y las distintas salidas que produce, con el objetivo de determinar sus principales ventajas y deficiencias en aras de desarrollar un módulo como parte del sistema CEDRUX que reúna las mejores prácticas de estas soluciones informáticas.

- Lluvia de ideas

Reunión de varios interesados en la que todos expresan sus ideas sobre el problema y su solución. La forma en que se llevo a cabo fue mediante la participación de cada implicado dando su idea en reuniones y talleres. Al finalizar la sesión de lluvia de ideas se puedo hacer una recolección de ideas sin duplicidad.

2.3.2 Requisitos del sistema

Los requisitos son la condición o capacidad que tiene que ser alcanzada por un sistema o componente de software para satisfacer un contrato, estándar, u otro documento impuesto formalmente. Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen. A continuación se presentan los requisitos funcionales identificados, su especificación y los requisitos no funcionales que se deben

tener en cuenta para el desarrollo del módulo.

2.3.2.1 Requisitos funcionales del sistema

RF1: Consolidar demanda por modelo de un órgano demandante.

RF2: Consolidar demanda de los órganos demandantes de un Centro de balance.

RF3: Aprobar demanda por indicador.

RF4: Desaprobar demanda por indicador.

RF5: Obtener cobertura total estimada.

RF6: Realizar ajuste por cobertura total estimada.

RF7: Realizar ajuste por economía.

RF8: Realizar ajuste por reserva del ministro.

RF9: Realizar ajuste por nivel de actividad.

Tabla 1: Especificación de requisito Consolidar demanda por modelo de un órgano demandante.

Precondiciones	Debe existir al menos un Centro de balance registrado en el sistema. Debe existir al menos un órgano demandante asociado.
Flujo de eventos	
Flujo básico Listar demanda por modelo de un órgano demandante	
1	El usuario accede al sistema.
2	Si el usuario autenticado es un órgano rector ir al flujo alternativo 2.a El usuario autenticado es un órgano rector, y para el caso de que sea un Centro de balance ir al flujo alternativo 2.b El usuario autenticado es un Centro de balance.
3	Concluye el requisito.
Pos-condiciones	
1	Se listaron los órganos demandantes.
Flujos alternativos	
Flujo alternativo 2.a El usuario autenticado es un órgano rector	
1	Se selecciona el tipo de modelo.

2	Se selecciona el Centro de balance.
3	Se selecciona el órgano demandante.
4	El sistema muestra la consolidación de la demanda de los órganos demandantes del Centro de balance.
Pos-condiciones	
1	N/A
Flujo alternativo 2.b El usuario autenticado es un Centro de balance	
1	Se selecciona el tipo de modelo.
2	Se selecciona el órgano demandante.
3	El sistema consolida los datos y muestra el listado de la demanda del órgano demandante seleccionado.
Pos-condiciones	
1	N/A
Pos-condiciones	
1	N/A
Validaciones	
1	N/A

Conceptos	Órganos demandantes	Visibles en la interfaz: Código No pieza Denominación Necesidad CTE ANA RM Demanda Economía Precio Importe.
Requisitos especiales	N/A	

Asuntos pendientes	N/A

Tabla 2: Especificación del requisito Consolidar demanda de los órganos demandantes de un Centro de balance.

Precondiciones	Debe existir al menos un Centro de balance registrado en el sistema. Debe existir al menos un órgano demandante asociado.
Flujo de eventos	
Flujo básico Consolidar demanda de los órganos demandantes de un Centro de balance	
1	El usuario accede al sistema.
2	Si el usuario autenticado es un órgano rector ir al flujo alternativo 2.a El usuario autenticado es un órgano rector, y para el caso de que sea un Centro de balance ir al flujo alternativo 2.b El usuario autenticado es un Centro de balance.
3	Concluye el requisito.
Pos-condiciones	
1	Se listaron los indicadores que demandan los órganos demandantes.
Flujos alternativos	
Flujo alternativo 2.a El usuario autenticado es un órgano rector	
1	Selecciona el tipo de modelo.
2	Selecciona el Centro de balance.
3	El sistema muestra la consolidación de la demanda de los órganos demandantes del Centro de balance seleccionado.
Pos-condiciones	
1	N/A
Flujo alternativo 2.b El usuario autenticado es un Centro de balance	
1	Selecciona el tipo de modelo.
2	El sistema muestra la consolidación de la demanda de los órganos demandantes del Centro de balance.
Pos-condiciones	
1	N/A
Validaciones	

1	N/A	
Conceptos	Centro de balance.	Visibles en la interfaz: Código No pieza Denominación Necesidad CTE ANA RM Demanda Economía Precio Importe
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 3: Especificación del requisito Aprobar demanda por indicador.

Precondiciones	Debe existir al menos un Centro de balance registrado en el sistema.
Flujo de eventos	
Flujo básico Aprobar demanda por indicador	
1	El usuario accede al sistema.
2	Si el usuario autenticado es un órgano rector ir al flujo alternativo 2.a El usuario autenticado es un órgano rector, y para el caso de que sea un Centro de balance ir al flujo alternativo 2.b El usuario autenticado es un Centro de balance.
3	El sistema confirma la activación de la actividad.
4	Concluye el requisito.
Pos-condiciones	

1	N/A
Flujos alternativos	
Flujo alternativo 2.a El usuario autenticado es un órgano rector	
1	Se selecciona el tipo de modelo.
2	Se selecciona el Centro de balance.
3	Se selecciona el órgano demandante.
4	Se selecciona la demanda.
5	Se oprime el botón aprobar.
Pos-condiciones	
1	N/A
Flujo alternativo 2.b El usuario autenticado es un Centro de balance	
1	Se selecciona el tipo de modelo.
2	Se selecciona el órgano demandante.
3	Se selecciona la demanda.
4	Se oprime el botón aprobar.
Pos-condiciones	
1	N/A
Validaciones	
1	N/A

Conceptos	Órganos demandantes.	Visibles en la interfaz: Código No pieza Denominación Necesidad CTE ANA RM Demanda Economía Precio

	Importe
Requisitos especiales	N/A
Asuntos pendientes	N/A

Tabla 4: Especificación del requisito Desaprobar demanda por indicador.

Precondiciones	Debe existir al menos un Centro de balance registrado en el sistema.
Flujo de eventos	
Flujo básico Desaprobar demanda por indicador	
1	El usuario accede al sistema.
2	Si el usuario autenticado es un órgano rector ir al flujo alternativo 2.a El usuario autenticado es un órgano rector, y para el caso de que sea un Centro de balance ir al flujo alternativo 2.b El usuario autenticado es un Centro de balance.
3	El sistema confirma la desactivación de la actividad.
4	Concluye el requisito.
Pos-condiciones	
1	N/A
Flujos alternativos	
Flujo alternativo 2.a El usuario autenticado es un órgano rector	
1	Se selecciona el tipo de modelo.
2	Se selecciona el Centro de balance.
3	Se selecciona el órgano demandante.
4	Se selecciona la demanda.
5	Se oprime el botón desaprobar.
Flujo alternativo 2.b El usuario autenticado es un Centro de balance	
1	Se selecciona el tipo de modelo.
2	Se selecciona el órgano demandante.
3	Se selecciona la demanda.
4	Se oprime el botón desaprobar.

Pos-condiciones		
1	N/A	
Validaciones		
1	N/A	
Conceptos	Organos demandantes.	Visibles en la interfaz: Código No pieza Denominación Necesidad CTE ANA RM Demanda Economía Precio
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 5: Especificación del requisito Obtener cobertura total estimada.

Precondiciones	Debe existir al menos un Centro de balance registrado en el sistema.
Flujo de eventos	
Flujo básico Obtener cobertura total estimada	
1	Se insertan los datos.
2	El sistema valida los datos introducidos.
3	Si los datos son correctos el sistema los registra, realiza el cálculo de la cobertura total estimada y lo muestra, y para el caso de que los datos sean introducidos incorrectamente debe pasar al flujo alternativo 2.a Información errónea.

4	Concluye el requisito.
Pos-condiciones	
1	Se realizo el cálculo de la cobertura total estimada.
Flujos alternativos	
Flujo alternativo 2.a Información errónea	
1	El sistema señala los datos erróneos y permite corregirlos.
2	El usuario corrige los datos.
3	Volver al paso 2 del flujo básico.
Pos-condiciones	
1	N/A
Validaciones	
1	Se validan los datos según lo establecido en el Modelo conceptual CIG-ERP-N-PLA-i2201.

Conceptos	Centro de balance.	Visibles en la interfaz: Necesidad Inventario PA Contratados
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 6: Especificación del requisito Realizar Ajuste por cobertura total estimada.

Precondiciones	Debe existir al menos un Centro de balance registrado en el sistema. Debe existir al menos un órgano demandante asociado. Obtener cobertura total estimada.
Flujo de eventos	
Flujo básico Realizar ajuste por cobertura total estimada	
1	El usuario selecciona el tipo de modelo.

2	Selecciona el órgano demandante.
3	Selecciona la demanda.
4	El usuario asigna valores a la cobertura total estimada de la demanda seleccionada.
5	El sistema valida (ver validación 1) el dato introducido.
6	Si el dato introducido es correcto el sistema lo registra y actualiza el valor de la cobertura total estimada del Centro de balance. Si el dato ha sido introducido incorrectamente debe ir al flujo 2.a Información errónea.
7	Concluye el requisito.
Pos-condiciones	
1	Se actualiza los valores de la cobertura total estimada.
Flujos alternativos	
Flujo alternativo 2.a Información errónea	
1	El sistema señala los datos erróneos y permite corregirlos.
2	El usuario corrige los datos.
3	Volver al paso 2 del flujo básico.
Pos-condiciones	
1	N/A
Validaciones	
1	Se validan los datos según lo establecido en el Modelo conceptual CIG-ERP-N-PLA-i2201.
Conceptos	Órgano demandante Visibles en la interfaz: Código No pieza Denominación Necesidad CTE ANA RM Demanda Economía Precio

		Importe
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 7: Especificación del requisito Realizar ajuste por Economía.

Precondiciones		N/A
Flujo de eventos		
Flujo básico Realizar Ajuste por Economía		
1	Se debe ingresar el valor asignado por concepto de la economía.	
2	Se valida el dato.	
3	Luego se incorpora el dato al listado de economía perteneciente a la demanda de los órganos demandante.	
4	Concluye el requisito.	
Pos-condiciones		
1	Se ingreso la economía.	
Flujos alternativos		
Flujo alternativo		
1	N/A	
Pos-condiciones		
1	N/A	
Validaciones		
1	Se validan los datos según lo establecido en el Modelo conceptual CIG-ERP-N-PLA-i2201.	
Conceptos	Centro de balance	Visible en la interfaz: Economía
Requisitos especiales	N/A	

Asuntos pendientes	N/A

Tabla 8: Especificación del requisito Realizar ajuste por reserva del ministro.

Precondiciones		N/A
Flujo de eventos		
Flujo básico Realizar ajuste por reserva del ministro		
1	Se debe ingresar el valor asignado por concepto de la reserva del ministro.	
2	Se validan los datos.	
3	Luego se incorpora el dato al listado de la reserva del ministro perteneciente a la demanda de los órganos demandante.	
4	Concluye el requisito.	
Pos-condiciones		
1	N/A	
Flujos alternativos		
Flujo alternativo		
1	N/A	
Pos-condiciones		
1	N/A	
Validaciones		
1	Se validan los datos según lo establecido en el Modelo conceptual CIG-ERP-N-PLA-i2201.	
Conceptos	Centro de balance	Visibles en la interfaz: Economía
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 9: Especificación del requisito Realizar ajuste por nivel de actividad.

Precondiciones	N/A
Flujo de eventos	
Flujo básico Realizar ajuste por nivel de actividad	
1	Se debe ingresar nivel de actividad.
2	Se valida el dato.
3	Luego se incorpora el dato al listado del nivel de actividad perteneciente a la demanda de los órganos demandante.
4	Concluye el requisito.
Pos-condiciones	
1	Se ingreso el nivel de actividad.
Flujos alternativos	
Flujo alternativo	
1	N/A
Pos-condiciones	
1	N/A
Validaciones	
1	Se validan los datos según lo establecido en el Modelo conceptual CIG-ERP-N-PLA-i2201.
Conceptos	Centro de balance Nivel de actividad.
Requisitos especiales	N/A
Asuntos pendientes	N/A

2.3.2.2 Requisitos no funcionales del sistema.

Requerimientos no funcionales: Son restricciones de los servicios o funciones ofrecidas por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requerimientos no funcionales a menudo se aplican al sistema en su totalidad. Normalmente apenas se aplican a características o servicios individuales del sistema. (24)

Los requisitos no funcionales con los que debe contar la aplicación son los establecidos por el equipo de Arquitectura para el producto de ERP (CEDRUX).

De aplicación o Software (SFT)

El sistema requiere como componentes de aplicación:

- Navegador Mozilla Firefox versión 2.2.

Del lado del servidor de Aplicaciones requiere:

- Soporte para PHP5.
- Un servidor web apache versión 2.0.
- Sistema Operativo Ubuntu Server.

Del lado del cliente de BD:

- Ubuntu Server.
- PostgreSQL versión 8.3

Dispositivo o Hardware (HDW): Los dispositivos que se necesitan para el correcto funcionamiento de la aplicación son:

Por el lado cliente:

- Procesador: 1.40 GHZ
- RAM: 256 MB (recomendado 512 Mb)
- Tarjeta de Red: 1

Del lado del servidor de aplicaciones:

- Tarjeta de Red: 1
- Procesador: 3.00 GHZ
- RAM: 1GB
- Disco duro: 160 GB
- UPS: 1
- Lector de CD: 1

Del lado del servidor de Base de datos:

- Tarjeta de Red: 1
- Procesador: 3.00 GHZ
- RAM: 1GB

2.3.3 Trazabilidad de requerimientos

La trazabilidad de requisitos es la habilidad para describir y seguir la vida de un requisito en ambos sentidos, hacia sus orígenes o hacia su implementación, a través de todas las especificaciones generadas. (25) Este proceso permite conocer qué elementos se ven afectados cuando ocurre un cambio en algún otro que tenga relación con el primero, o sea, cuando algún requerimiento o cualquier otro elemento traceable es modificado, todas las relaciones asociadas a ese requerimiento se convierten en sospechosas, en ese caso se deben revisar los cambios y determinar si los elementos asociados deben ser cambiados también.

Para facilitar el trabajo de determinar las relaciones entre los requisitos y el seguimiento de los mismos, se utilizan las matrices de trazabilidad. A continuación se representan las relaciones entre los requisitos definidos anteriormente en la sección: 2.3.2.1 Requisitos funcionales del sistema.

Tabla 10: Matriz de seguimiento de dependencias: Requisitos funcionales del proceso Realizar Balance Material.

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8	RF9
RF1									
RF2									
RF3	X								
RF4	X								
RF5							X	X	X
RF6					X				
RF7									
RF8									
RF9									

2.3.4 Métricas para la verificación de la especificación de requisitos

La especificación de requisitos es un factor principal para el buen funcionamiento del desarrollo de software. De acuerdo con el estándar IEEE 830, se considera que una especificación es de calidad cuando se puede decir que es correcta, no-ambigua, completa, consistente, ordenada por importancia y estabilidad, verificable, modificable y trazable.

Especificación correcta

La corrección no se puede establecer a priori, sino que depende fundamentalmente del usuario final del sistema representado. Quien debe decidir si una especificación es correcta o no es el cliente que solicita el sistema. Por eso la corrección de una especificación debe ser verificada a través de la revisión y aceptación del usuario.

Especificación no ambigua

El equipo de desarrollo que interviene en el proceso de especificación de requisitos suele tener varios puntos de vista. Por este motivo es difícil asegurar la ausencia de ambigüedad en la especificación. Aunque a priori parece un problema insalvable, existen formas de limitar los efectos negativos de la ambigüedad.

Especificación completa

Una especificación es completa si, y sólo si, describe todos los requisitos relevantes para el usuario, incluyendo requisitos asociados con funcionalidad, actuación, restricciones de diseño, atributos o interfaces externas.

Especificación consistente

Los mayores problemas relacionados con la consistencia son los que tienen que ver con las incoherencias lógicas entre distintos requisitos (requisitos incompatibles, incoherentes o mutuamente excluyentes), la repetición de la misma información a lo largo de distintos requisitos (requisitos repetitivos o redundantes) o la referencia en distintos requisitos a ítems que usan la misma palabra para designar conceptos del problema diferentes (incoherencia respecto al dominio del problema).

Especificación organizada

La categorización de los requisitos por orden de importancia es una recomendable práctica que permite establecer prioridades a la hora de abordar el desarrollo. Esta categorización por el atributo

importancia o prioridad es necesaria desde un punto de vista práctico.

Otra posible categorización que resulta interesante es la de la estabilidad de la especificación. El cambio de los requisitos de usuario es algo intrínseco al propio cambio en el problema.

Especificación verificable

Se considera que una especificación es verificable si lo son cada uno de los requisitos constituyentes. A su vez, se considera que un requisito individual es verificable si existe un proceso acotado que permita determinar que el sistema construido satisface lo descrito en el propio requisito.

Una forma de conseguir que los requisitos sean verificables es describirlos con suficiente detalle, o teniendo en cuenta que una de las premisas que se debe cumplir es que sean probados una vez implementados.

Especificación trazable

Una especificación se considera trazable si el origen de cada requisito individual está claro y existe algún mecanismo que permita seguir el impacto de dicho requisito a lo largo del resto de actividades del ciclo productivo. (25)

A continuación se aplicarán métricas para obtener valores cuantitativos y así poder medir el grado en que algunas de estas características se reflejan en las especificaciones realizadas.

Tabla 11: Métricas auxiliares aplicadas a la especificación de requisitos.

	Descripción	Valor
TR	Total de requerimientos evaluados	9
NUI	Número de requisitos para los que todos los revisores tuvieron una misma interpretación	9
RC	Cantidad de requisitos cambiados (insertados, modificados y eliminados)	3
NNV	Número de requisitos no válidos	0
NC	Número de requisitos considerado válidos	9

Tabla 12: Métricas principales aplicadas a la especificación de requisitos.

No	Métrica	Fórmula	Propiedad	Valor
1	Especificidad	Q1 = NUI / RT * 100	No ambigüedad	100%
2	Estabilidad	Q2 = RC / TR *100	Estabilidad	33%
3	Grado de validación	Q3 = NC / (NC + NNV)* 100	Grado de validación	100%

Interpretación:

- Una especificidad de requerimientos de un 100% es una especificación realizada con una alta calidad, con ausencia de ambigüedades, demostrando que todas las personas involucradas en el proceso de revisar los requisitos coincidieron con la interpretación de los mismos.

- La inestabilidad es de un 33%, por lo que las especificaciones se consideran estables para un 67% con respecto a los cambios que se realizaron.

- A partir del resultado que arrojó la métrica Grado de validación se demuestra que todos los requisitos están en estado validado.

2.3.5 Patrones utilizados en el tratamiento de requisitos

Para determinar los requisitos funcionales además de las técnicas utilizadas para su captura se utilizaron patrones de casos de uso adaptados a las necesidades. Se decidió hacer uso de ellos para realizar de forma precisa el tratamiento de los requisitos, logrando una organización en su nomenclatura, descripción y propósito.

Tabla 13: Patrón: El nombre revela la intención.

Patrón	El nombre revela la intención

Problema	Utilizar nombres descriptivos para los casos de uso (requerimientos) es una buena práctica, porque ellos revelan exactamente la intención de cada uno.
Solución	Nombrar los casos de uso (requerimientos) empezando con un verbo y seguido de una frase que refleje su objetivo. Ser conciso pero lo suficientemente descriptivo para capturar su esencia.
Ejemplo	RF3: Aprobar demanda por indicador. RF4: Desaprobar demanda por indicador. RF5: Obtener cobertura total estimada.

Tabla 14 Patrón: Preciso y legible

Patrón	Preciso y legible
Problema	Un caso de uso (requerimientos) debe ser descrito de una manera fácil de entender para los usuarios y el equipo de desarrollo.
Solución	Incluir solo la información necesaria, pero describiendo el comportamiento lo suficientemente claro y preciso de manera que un usuario sin conocimientos pueda completamente entender las consecuencias del comportamiento, sin perderse eventos importantes.
Ejemplo	Para comprender mejor la solución que propone este patrón se aconseja consultar la sección 2.3.2.1 Requisitos funcionales del sistema. Cada una de las especificaciones aquí representadas fueron claras tanto para el cliente como para el equipo de desarrollo.

Tabla 15 Patrón: Completar una única meta

Patrón	Completar una única meta
Problema	Objetivos inadecuados pueden hacer dudar a las personas encargadas de describir los requerimientos a la hora de decidir dónde termina uno y comienza otro.
Solución	Describir cada requisito dirigiéndose hacia un completo y definido propósito.
Ejemplo	Una vez realizada la especificación de requisitos se verificó que cada una tuviera un objetivo claro. Ver sección 2.3.2.1 Requisitos funcionales del sistema.

Tabla 16 Patrón: Condiciones Detectables

Patrón	Condiciones Detectables.
Problema	Los escritores siempre luchan con cuantas y cuales alternativas incluir.
Solución	Incluir solamente condiciones detectables. Reunir condiciones que tengan el mismo efecto neto en el sistema. Cada escenario en el caso de uso (requerimientos) debe comenzar con una acción que el sistema esté capacitado para detectar.
Ejemplo	Requisito: Realizar ajustes por Economía. Flujo básico: 1. Se debe ingresar el valor asignado por concepto de la economía. 2. Se valida el dato. 3. Luego se incorpora el dato al listado de economía perteneciente a la demanda de los órganos demandante. Para comprender mejor la solución que propone este patrón se aconseja consultar la sección 2.3.2.1 Requisitos funcionales del sistema.

Tabla 17: Patrón: Pasos Nivelados.

Patrón	Pasos Nivelados.
Problema	Excesivamente largos o excesivamente pequeños pasos en el caso de uso (requerimientos) oscurecen la meta y hacen el caso de uso difícil de leer y comprender.
Solución	Cada escenario debe tener de tres a nueve pasos. Idealmente son todos a niveles similares un nivel de la abstracción justamente debajo de la meta del caso del uso. Esto no quiere decir que un caso de uso (requerimientos) no pueda tener más de 9 pasos si lo requiere.
Ejemplo	Requisito: Listar demanda por modelo de un órgano demandante. Flujo básico: 1. Se selecciona el modelo. 2. Se selecciona el Centro de balance. 3. Se selecciona el órgano demandante. 4. El sistema consolida los datos y muestra el listado de la demanda del órgano demandante seleccionado. Para comprender mejor la solución que propone este patrón se aconseja consultar la sección 2.3.2.1 Requisitos funcionales del sistema.

En otro orden de demostración, de acuerdo a lo que establece el patrón Clasificación y aplicándolo a la especificación de requisitos, se determinó que las mismas son puramente textuales, constituyendo una lista de funcionalidades.

2.4 Diseño de la solución

Este flujo de trabajo consiste en transformar los requerimientos en el diseño del futuro sistema y adaptarlo para que se corresponda con el entorno de implementación mediante el esbozo de sus funcionalidades, es decir, la traducción de los requisitos a una especificación que describe cómo implementar la solución.

2.4.1 Prototipos funcionales de IU

Los Prototipos de IU son empleados para explorar un diseño de IU factible y cómodo que cumpla los requerimientos, ayudando a cerrar la brecha entre lo que se requiere (expresado a través de la respuesta a los requerimientos) y lo que es factible.

En el diseño de los prototipos funcionales de interfaz de usuario se debe hacer gran énfasis, ya que estos están en la capa de presentación, siendo esta la encargada de interactuar con el usuario por lo que primeramente deben cumplir con los principios de usabilidad y funcionalidad definidos en el estándar de interfaz de usuario para de esta manera ganar la mayor atención posible de los mismos. A continuación de muestran los prototipos funcionales de interfaz de usuario diseñados:

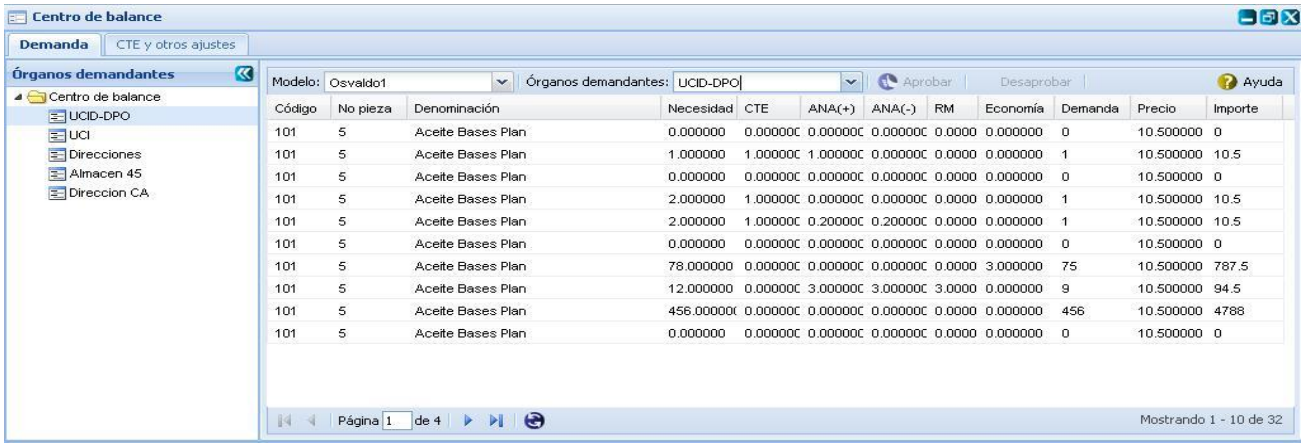

Figura 3: Prototipo funcional de interfaz de usuario (Demanda)

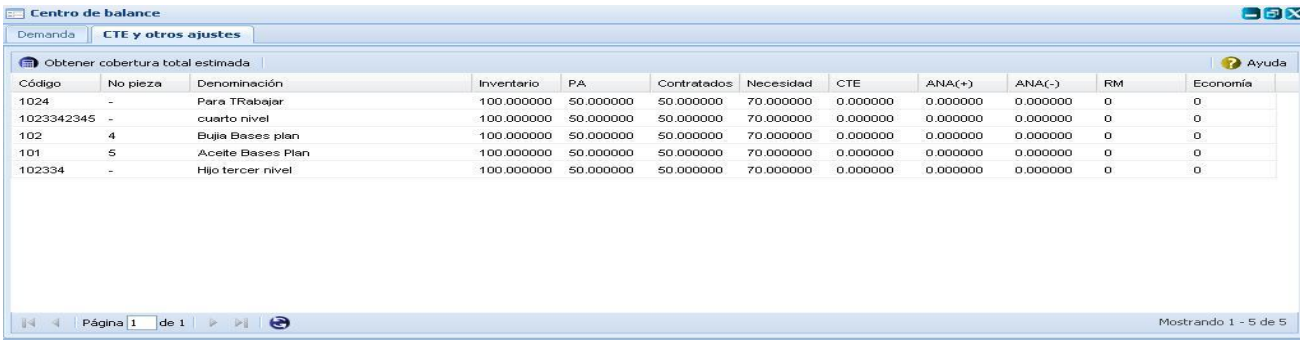

Figura 4: Prototipo funcional de interfaz de usuario (CTE y otros ajustes)

2.4.2 Diagrama de Clases del Diseño

Los diagramas de clases de diseño muestran a través de atributos y métodos la estructura de las clases que después serán escritas en algún lenguaje de programación (PHP en este caso) y constituyen una entrada para los desarrolladores, brindándoles un panorama de lo que contendrá cada clase haciendo más fácil y rápido el trabajo de estos. A continuación se muestra el diagrama de Clases del Diseño:

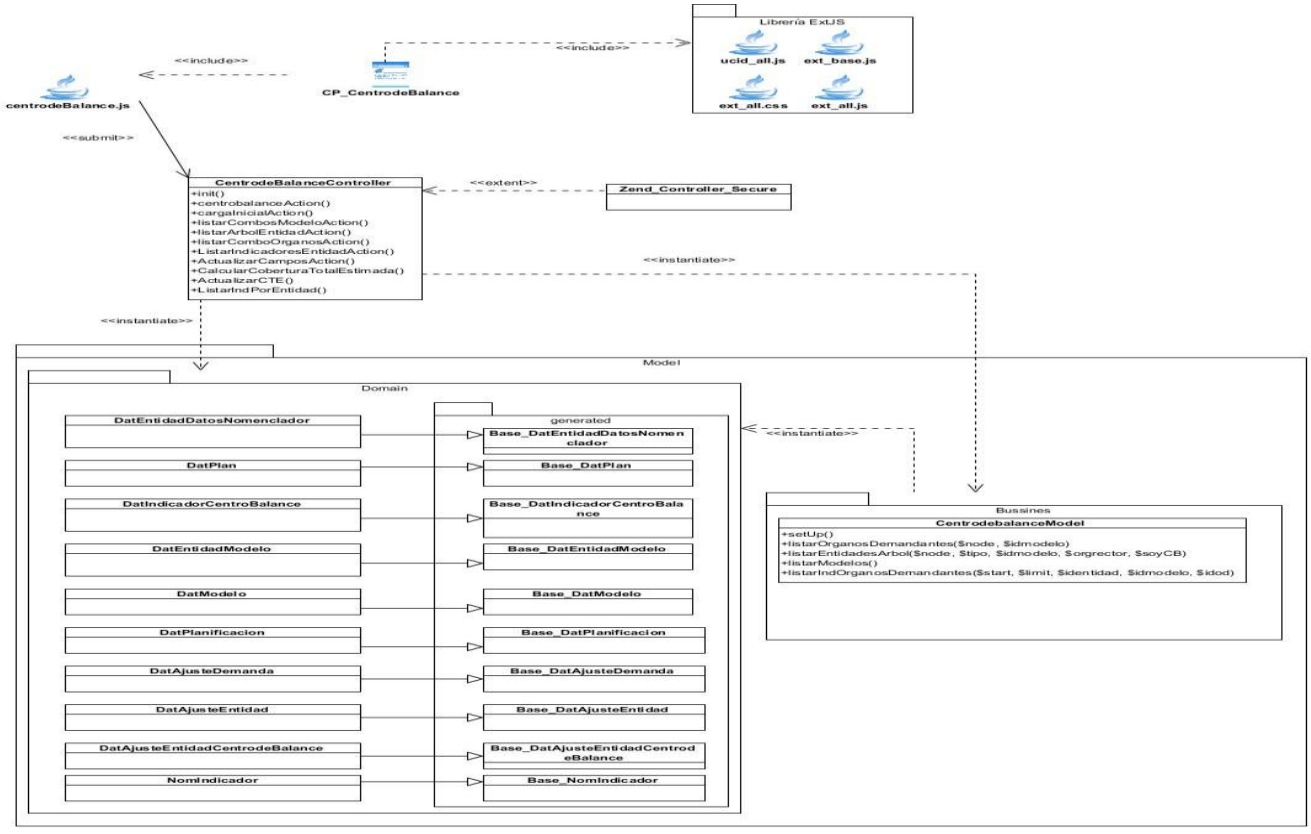

Figura 5: Diagrama de Clases del Diseño

Descripción del diseño de clases del proceso Centro de balance:

Librería Extjs: Contiene los componentes generados a través de la librería JavaScript Extjs.

Centrodebalance.js: Encargada de generar de forma dinámica a través del DOM y utilizando la librería Extjs los componentes. Debe enviar y recibir los datos de la controladora utilizando tecnología AJAX.

Centrodebalance: Responsable de visualizar a través de los js que debe incluir, la información necesaria para realizar las operaciones relacionadas con el Centro de balance. Además de enviar y recibir los datos de la controladora utilizando tecnología AJAX.

SP_CentrodeBalanceController: Clase controladora encargada de efectuar la consolidación del Centro de balance y de sus Órganos Demandantes, la realización del Balance Material y de la Cobertura total estimada.

ZendExt_Controller_Secure: Encargada de gestionar acciones personalizadas y está integrada a la seguridad.

Paquete Model: Encargado de manejar los datos persistentes dentro del componente. Contiene el Bussines y el Domain.

CentrodeBalanceModel: Es la clase encargada de comunicar la clase controladora con el Domain.

ZendExt_Model: Modelo gestor de negocio que permite entre otras funcionalidades iniciar la conexión a la base de datos.

Domain: Permite el acceso a la información que esta almacenada a nivel de datos.

2.4.3 Modelo de datos

El Modelo de datos como artefacto de software describe la representación lógica y física de los datos persistentes usados por la aplicación. El modelo propuesto es una representación de las tablas existentes en la base de datos así como las relaciones entre ellas. El mismo cuenta con 11 tablas que representan de manera general el negocio del componente Centro de balance. Los conceptos de este modelo están dirigidos fundamentalmente al personal informático, no a los usuarios finales.

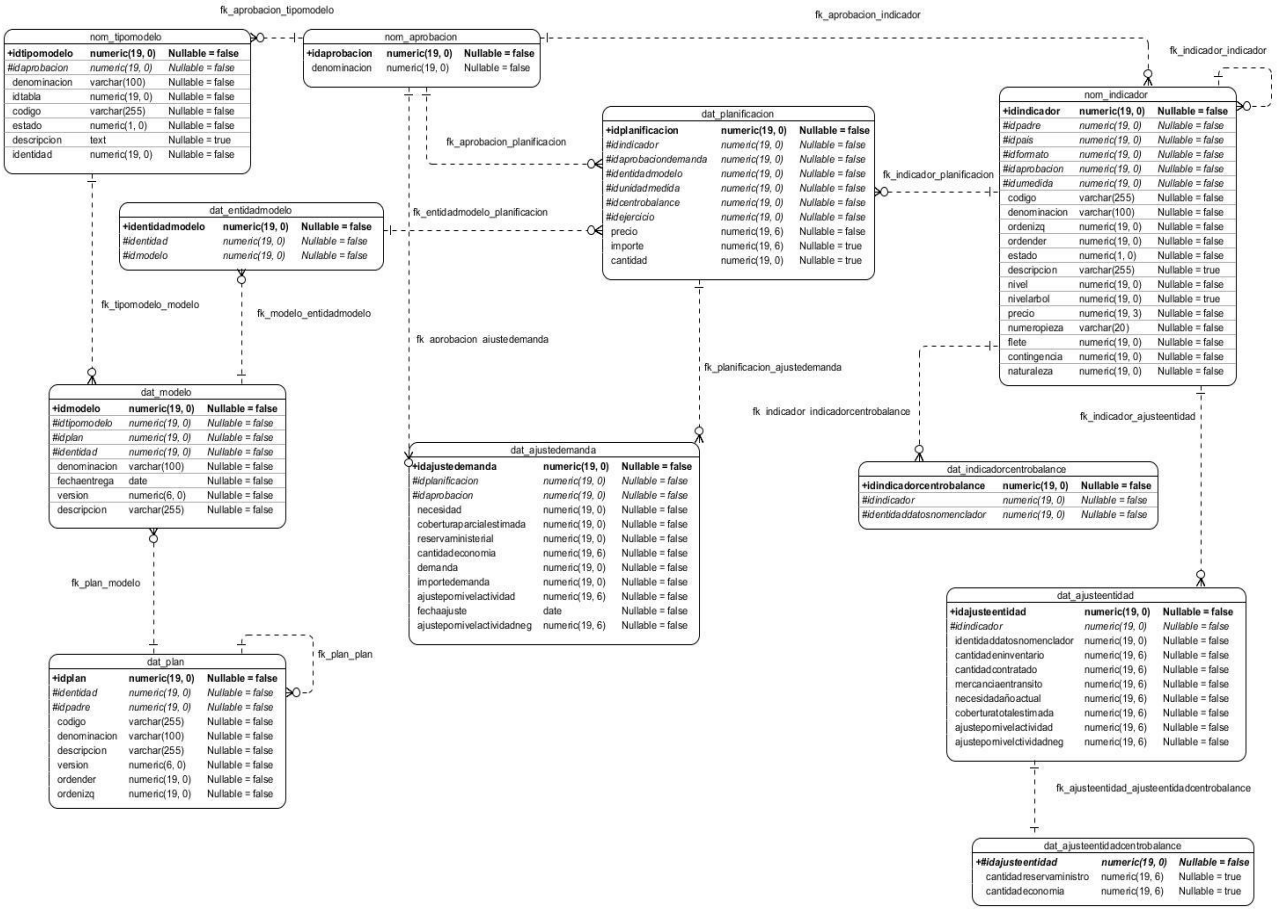

Figura 6: Modelo de datos

2.5 Patrones de diseño utilizados

El diseño fue elaborado basándose en los patrones GRASP, estos constituyen soluciones simples y completas a problemas en específico y comunes del diseño orientados a objetos, los que describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones:

Experto: Dicho patrón es evidenciado en la definición de las clases de acuerdo a las funcionalidades que deben realizar a partir de la información manejada dentro del módulo, como por ejemplo en la clase del Domain DatEntidadModelo, la cual será la responsable de efectuar las operaciones listar las entidades según un modelo y de mostrar lo que tiene una entidad en un modelo. Sobre este mismo principio se realiza el diseño de las restantes funcionalidades.

Creador: Este patrón es adaptable a las clases del paquete Domain, quienes son las encargadas de crear los objetos de tipo Doctrine_Query, para permitir el acceso a la información almacenada a nivel de datos.

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

Controlador: La clase controladora definida: CentrodeBalanceController es un ejemplo de la aplicación de este patrón, la misma tendrá a cargo la responsabilidad de manejar los eventos dentro del módulo.

2.6 Conclusiones del capítulo

Una vez realizadas las tareas definidas para dar cumplimiento los objetivos específicos que tributan a este capítulo, se concluye que se puede dar comienzo a la construcción de la propuesta de solución, teniendo en cuenta que la realización del modelado del negocio, las especificaciones de los requisitos del sistema, la elaboración de los prototipos de interfaz de usuario y el diseño de la solución sustentado en la utilización de patrones, constituyen una entrada válida para las fases posteriores del desarrollo, en este caso, la implementación del modulo Centro de balance.

Capítulo # 3: Implementación y Prueba.

3.1 Introducción

En el presente capítulo serán abordados los temas referentes a la implementación y pruebas del sistema. En el mismo se muestra el modelo de implementación, donde se representan las clases y sus métodos. Se especifica el modelo de pruebas donde se detallan los casos de prueba que componen el sistema, comprobando de esta forma la funcionalidad del sistema.

3.2 Implementación del sistema

La implementación se lleva a cabo luego de un diseño riguroso, aquí se describe cómo como las clases se implementan en términos de componentes. En el proceso de implementación se mantiene como principio la estandarización y reutilización de códigos y componentes, se organizan los componentes de acuerdo con los mecanismos de estructuración y modulación disponibles en el entorno de implementación y en el lenguaje de programación utilizado.

3.2.1 Estándares utilizados

Los estándares de diseño son ciertas pautas que se establecen para lograr la satisfacción de las necesidades y exigencias de los clientes, mediante la uniformidad en el diseño de una aplicación web, que sea compatible con las tecnologías utilizadas y cumpla con todos los objetivos. Los estándares de codificación de programación no están enfocados a la lógica del programa, sino a su estructura y apariencia física para facilitar la comprensión del código y su futuro mantenimiento. La aplicación debe ser implementada siguiendo los estándares de documentación, implementación de interfaz de usuario e implementación de lógica de negocio diseñado para el proceso de desarrollo de software del ERP:

1. Nomenclatura de las clases

Los nombres de las clases comienzan con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación **PascalCasing**, la cual define que los identificadores y nombres de variables, métodos y funciones están compuestos por múltiples palabras juntas, iniciando cada palabra con letra mayúscula y con sólo leerlo se reconoce el propósito de la misma.

Ejemplo: NomIndicador. En este caso el nombre de clase está compuesto por 2 palabras iniciadas

cada una con letra mayúscula.

1.1 Nomenclatura según el tipo de clases:

Clases controladoras: Las clases controladoras después del nombre llevan la palabra:"Controller". Ejemplo: CentrodeBalanceController

Clases de los modelos:

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

Domain (Dominio): Las clases que se encuentran dentro de Domain el nombre que reciben es el de la tabla en la base de datos. Ejemplo: DatPlanificacion.

Generated (Dominio base): Las clases que se encuentran dentro de Generated el nombre comienza con la palabra: "Base" y seguido el nombre de la tabla en la base de datos.

Ejemplo: BaseDatPlan.

2. Nomenclatura de las funcionalidades y atributos:

El nombre a emplear para las funciones y los atributos se escribe con la inicial del identificador en minúscula, en caso de que sea un nombre compuesto se empleará notación **CamelCasing** que es similar a la antes mencionada: **PascalCasing** con la excepción de la primera letra.

Ejemplo de método: actualizarCampos. El nombre de método está compuesto por 2 palabras, la primera en minúsculas y la segunda iniciando con letra mayúscula.

Las principales funcionalidades de las clases controladoras se les pone el nombre y seguida la palabra:"Action" Ejemplo: cargaInicialAction.

3. Nomenclatura de los comentarios:

Los comentarios deben ser lo bastante claros y precisos de forma tal que se entienda el propósito de lo que se está desarrollando. En caso de ser una función complicada se debe comentar para lograr una mejor comprensión del código.

3.2.2 Estructura de datos a utilizar

La estructura de datos es la forma de organizar los valores que puede tomar durante el programa dichos datos, de forma tal que si se le intenta dar un valor fuera del conjunto se produjera un error.

El lenguaje de programación utilizado para llevar a cabo la implementación es PHP que es el que está definido por la línea del ERP, donde se hace uso de variables que tienen información similar y que se procesan de forma semejante, por lo que la estructura más apropiada son los arreglos.

Un arreglo no es más que un conjunto de datos semejante agrupados bajo un mismo nombre, donde el primer elemento está en la posición cero.

En dicho lenguaje de programación existen dos tipos de arreglos: los de índices numéricos y los de índices asociativos, ambos poseen una serie de funciones creadas para ordenarlos por orden alfabético directo o inverso, por claves, para contar el número de elementos que comprende y movernos por dentro de él hacia delante o atrás. A partir de lo antes descrito y por decisión del equipo de arquitectura esta vendría siendo la estructura de datos a utilizar de forma general para el desarrollo en todos los subsistemas.

3.2.3 Representación de la interacción entre los componentes internos del módulo Centro de balance

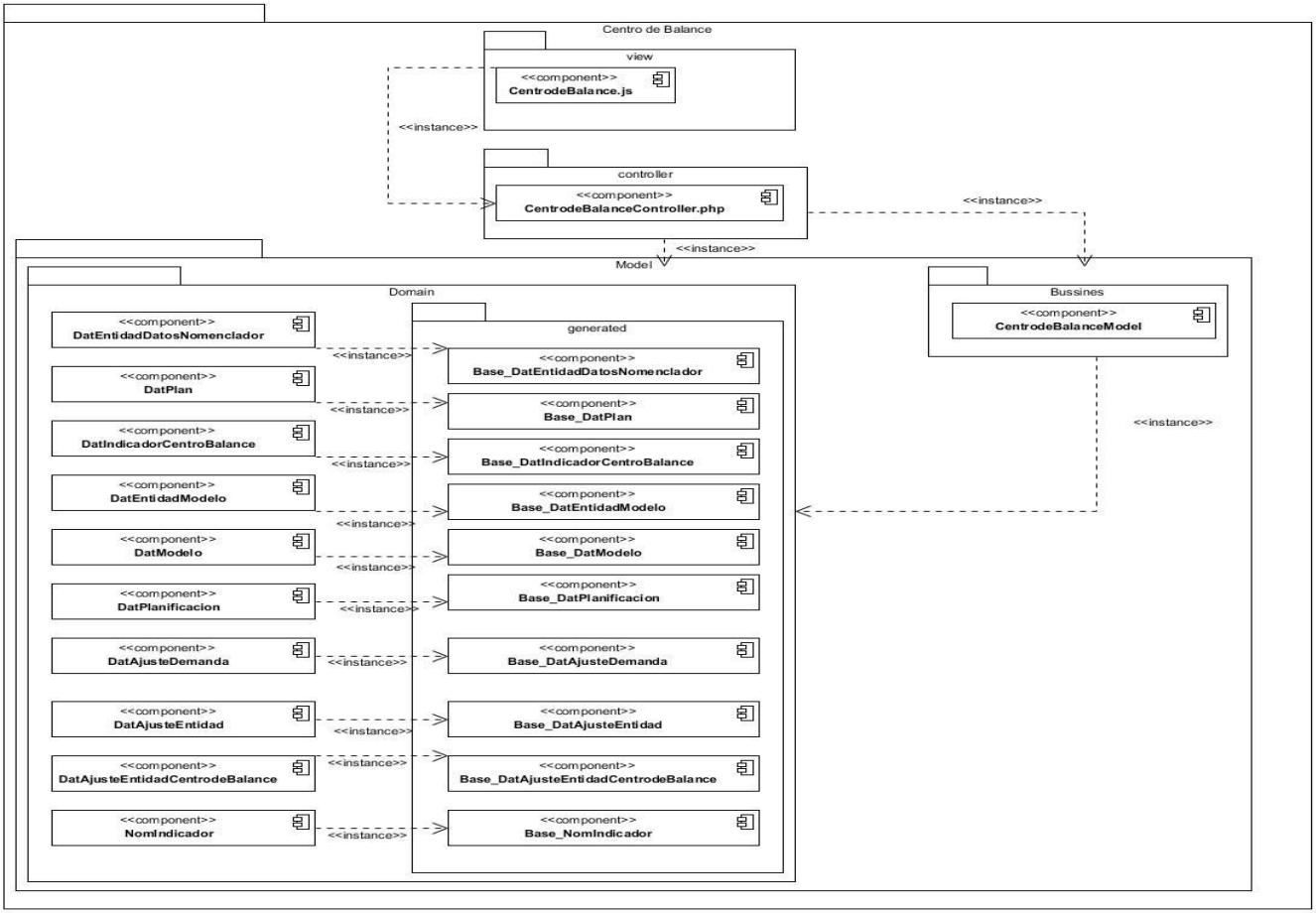

Figura 7: Interacción entre los componentes internos del módulo Centro de balance

3.2.5 Descripción de las clases y funcionalidades del componente

Tabla 18: Descripción de la clase CentrodeBalanceModel.

Nombre: CentrodeBalanceModel.	
Tipo de clase: Bussines.	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Descripción:

listarModelos()	Lista todos los modelos existentes, que se encuentran en la tabla dat_modelo.
listarEntidadesArbol($node, $tipo, $idmodelo, $orgrector, $soyCB)	Muestra un listado con las denominaciones de todos los Centros de balance, si el usuario se loguea como órgano rector. Si el usuario se loguea como Centro de balance se lista él solo. Si el usuario se loguea como órgano demandante se lista él solo.
listarIndOrgDemandantes($start, $limit, $identidad, $idmodelo, $idod)	Muestra un listado con las consolidaciones de las demandas realizadas por el órgano demandante en un modelo determinado en el año actual.
listarOrganosDemandantes($node, $idmodelo)	Lista las entidades que han planificado en el modelo y Centro de balance seleccionados.
actualizarCampos($idajuste, $campo, $valor, $necesidadajuste, $codigoind, $denomind, $idaprobaciondemanda, $precioind, $idindicador, $identidadmodelo, $idejercicio, $fechaajuste)	Se actualizan los datos correspondientes a las demandas.
aprobarDesaprobarDemandaPorIndicador($start, $limit, $orgrector, $arrind, $identidad, $idmodelo, $aprobacion)	Aprobar o desaprobar la(s) demanda(s) seleccionada(s).
ListarIndCbAjustesyCTE($start, $limit, $identidad)	Si se loguea como Centro de balance se muestra un listado con los datos de existencia por indicador que posee el mismo y los que planificó (demandó). Si se loguea como órgano demandante se muestra un listado con los datos de existencia por cada indicador que demandó.

actualizarCamposAjusteEntidad($idajusteentidad, $campo, $valor)	Actualiza los datos modificados en la existencia del indicador seleccionado.
insertarAjusteDemanda($idaprobacion, $necesidad, $cantidadeconomia, $anapos, $ananeg, $reservaministerial, $coberturaparcialest, $fechaajuste, $demanda, $importe, $idind, $identmod, $idejercicio)	Adiciona un nuevo ajuste con los datos actualizados a la tabla dat_ajustedemanda.

Tabla 19: Descripción de la clase CentrodeBalanceController.

Nombre: CentrodeBalanceController.	
Tipo de clase: Controladora	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Descripción:
cargaInicialAction()	Devuelve un conjunto de datos o información relacionada con el proceso de autenticación de cada entidad a la aplicación.
listarComboModelosAction()	Se hace una instancia de la clase CentrodeBalanceModel llamando al método listarComboModelos devolviendo la lista de modelos existentes.
listarArbolEntidadesAction()	Se hace una instancia de la clase CentrodeBalanceModel llamando al método ListarArbolEntidades devolviendo uno o varios Centros de balance o el órgano demandante según cómo se autentique en la aplicación.
	Se hace una instancia de la clase CentrodeBalanceModel llamando al

listarComboOrganosAction()	método listarOrganosDemandantes listando las entidades que han planificado en el modelo seleccionado.
listarIndicadoresEntidadAction()	Se hace una instancia de la clase CentrodeBalanceModel llamando al método listarIndOrganosDemandantes() mostrado un listado con las demandas realizadas por el órgano demandante en el modelo seleccionado y en el año actual.
actualizarCamposAction()	Se hace una instancia de la clase CentrodeBalanceModel llamando al método actualizarCampos() actualizando los campos modificados en el ajuste de las demandas o insertando un nuevo ajuste con los datos modificados en caso de que no exista.
actualizarCamposAjusteEntidadAction()	Se hace una instancia de la clase CentrodeBalanceModel llamando al método actualizarCamposAjusteEntidad() actualizando los datos modificados en la existencia del indicador seleccionado
listarIndCbAjustesyCTEAction()	Se hace una instancia de la clase CentrodeBalanceModel llamando al método listarIndCbAjustesyCTE() mostrando un listado con las existencias por indicador en la entidad logueda.
aprobarDesaprobarDemandaPorIndicadorAction()	Se hace una instancia de la clase CentrodeBalanceModel llamando al método aprobarDesaprobarDemanda() aprobando o desaprobando la(s) demanda(s) seleccionada(s)

Tabla 20: Descripción de la clase DatEntidadDatosNomenclador.

Nombre: DatEntidadDatosNomenclador	
Tipo de clase: Domain	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Descripción:
getIdentidaddatosnomenclador($node)	Devuelve el iddatosnomenclador de una determinada entidad.
getCB()	Devuelve todas las entidades que son Centros de balance en la tabla dat_entidaddatosnomenclador.

Tabla 21: Descripción de la clase NomIndicador.

Nombre: NomIndicador.	
Tipo de clase: Domain	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Descripción:
entidadEsCB($identidad)	Devuelve si una entidad es Centro de balance o no.

Tabla 22: Descripción de la clase DatModelo.

Nombre: DatModelo.
Tipo de clase: Domain

Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Descripción:
devolverModelos()	Devuelve los modelos existentes en la tabla dat_modelo.

Tabla 23: Descripción de la clase DatIndicadorCentroBalance.

Nombre: DatIndicadorCentroBalance.	
Tipo de clase: Domain	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Descripción:
listarIndAprobPorCB($identidaddatosnomenclador)	Devuelve todos los indicadores aprobados de la tabla nom_indicador dada una determinada entidad.

Tabla 24: Descripción de la clase DatEntidadModelo.

Nombre: DatEntidadModelo	
Tipo de clase: Domain	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Descripción:

listarEntMod($idmodelo)	Devuelve todas las entidades dado un determinado modelo.
getIdentidadmodelo($idmodelo, $idod)	Devuelve una entidadmodelo dados un modelo y una entidad.

Tabla 25: Descripción de la clase DatPlanificacion.

Nombre: DatPlanificacion	
Tipo de clase: Domain	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	Descripción:
indEntEjercActual($identidadmodelo, $idindicador)	Devuelve los indicadores de la tabla nom_indicador dados un modelo, una entidad y un ejercicio (año).
dameIndPlanDadoAjuste($idindicador, $identidadmodelo)	Devuelve los identificadores planificados por una entidad dada en un modelo determinado.

3.2.6 Diagrama de Despliegue

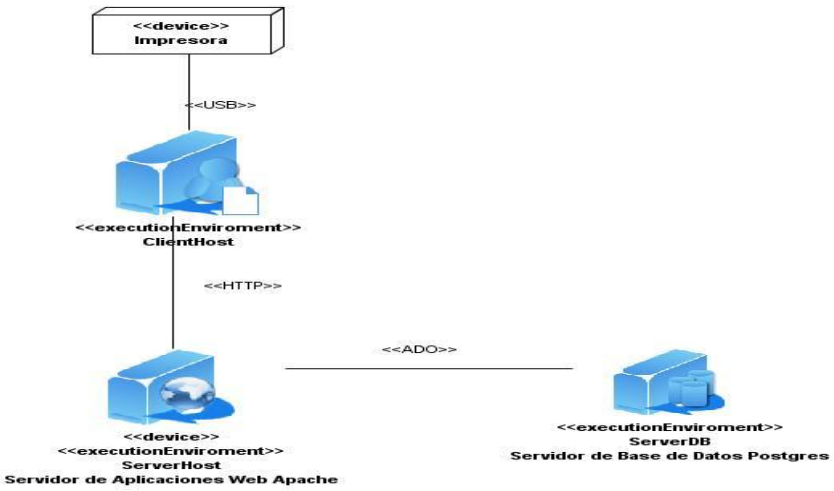

Figura 8: Diagrama de Despliegue.

3.3 Pruebas

Las pruebas del software es la forma que más eficaz que se tiene para garantizar la calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación. Las pruebas aplicadas al sistema resultaron de vital importancia para que el software saliera con la calidad requerida y se exponen a continuación:

3.3.1 Descripción y aplicación de la Prueba de Caja Negra

Las pruebas de caja negra son aquellas que se enfocan directamente en el exterior del módulo, sin importar el código. Son pruebas funcionales en las que se trata de encontrar fallas en la interfaz de usuario. Estas pruebas permiten encontrar:

1. Funciones incorrectas o ausentes.
2. Errores de interfaz.
3. Errores en estructuras de datos o en accesos a las Bases de Datos externas.
4. Errores de rendimiento.
5. Errores de inicialización y terminación.(39)

En la sección de anexos se puede encontrar el diseño de casos de prueba del requisito Obtener cobertura total estimada.

Anexo # 1 Tabla de Requisito a probar: Obtener cobertura total estimada.

Anexo # 2: Descripción de las variables.

Anexo # 3: Juegos de datos a probar.

3.3.2 Descripción y aplicación de la Prueba de Caja Blanca

Las pruebas de caja Blanca son las que se les realizan al software sobre las funciones internas de un módulo, examinándose la lógica interna sin considerar los aspectos de rendimiento.

Tipos de prueba de caja blanca:

- Prueba de Condición: Es un método de diseño de casos de prueba que ejercita las condiciones lógicas contenidas en el módulo de un programa.

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

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

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

Representación del grafo de flujo asociado al código presentado con anterioridad mediante nodos, aristas y regiones:

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

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

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

```php
public function listarEntidadesArbol($node, $tipo, $idmodelo, $orgrector, $soyCB) {
    $i = 0; $identidad = $this->global->Estructura->idestructura; //1
    $datosentidad = $this->integrator->metadatos->DameEstructura($identidad); //1
    $cb = DatEntidaddatosnomenclador::getCB(); //1
    if($orgrector && $node == '0'){ //2
        foreach ($cb as $kr => $da) { //3
            $dat[$i]['id'] = $da['identidad']; //4
            $enti = $this->integrator->metadatos->DameEstructura($da['identidad']); //4
            $dat[$i]['codigo'] = $enti[0]->codigo; //4
            $dat[$i]['leaf'] = true; //4
            $dat[$i]['text'] = $enti[0]->denominacion; //4
            $tipo = 1; //4
            $dat[$i]['tipo'] = $tipo; //4
            $i++; //4
        } //5
    } //6
    }else if(soyCB && $node=='0'){ //7
            $dat[0]['id'] = $identidad; //8
            $dat[0]['codigo'] = $datosentidad[0]->codigo; //8
            $dat[0]['leaf'] = true; //8
            $dat[0]['text'] = $datosentidad[0]->denominacion; //8
            $tipo = 1; //8
            $dat[0]['tipo'] = $tipo; //8
            $datos = $dat; //8
        } //9
        else { //10
            $dat[0]['id'] = $identidad; //10
            $dat[0]['codigo'] = $datosentidad[0]->codigo; //10
            $dat[0]['leaf'] = true; //10
            $dat[0]['text'] = $datosentidad[0]->denominacion; //10
            $tipo = 1; //10
            $dat[0]['tipo'] = $tipo; //10
            $datos = $dat; //10
        }//10
        return $datos; //11
    }
```

Figura 9: Código

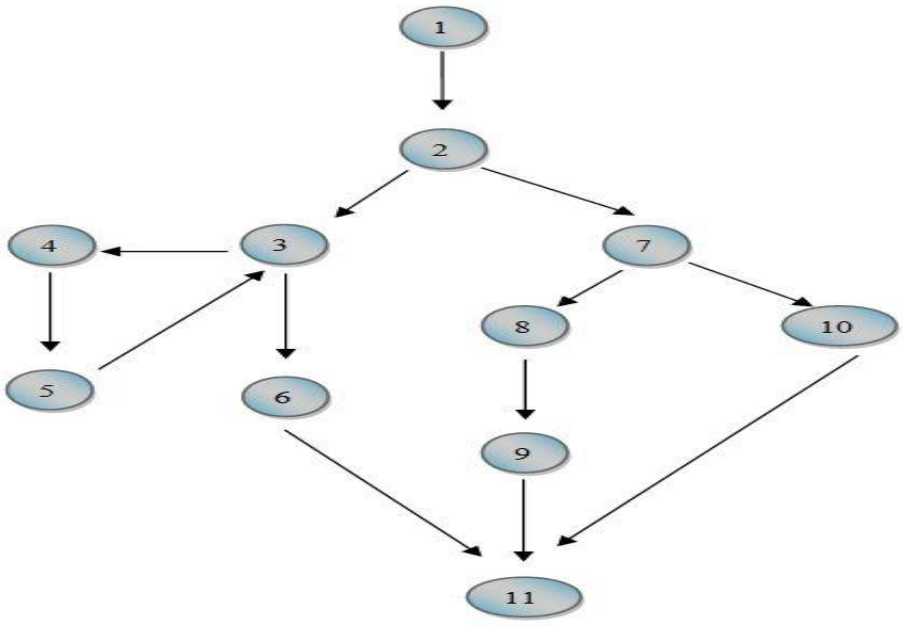

Figura 10: Grafo de flujo

Luego de haber construido el grafo se realiza el cálculo de la complejidad ciclomática mediante las tres fórmulas descritas a continuación, las cuales deben arrojar el mismo resultado para asegurar que el cálculo de la complejidad es correcto.

1. V (G) = (A – N) + 2

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

V (G) = (13 - 11) + 2

V (G) = 4

2. V (G) = P + 1

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

V (G) = 3 + 1

V (G) = 4

3. V (G) = R

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

V (G) = 4

El cálculo efectuado mediante las tres fórmulas ha dado el mismo valor, dando como resultado 4, lo que indica que existen 4 posibles caminos por donde el flujo puede circular, y determina el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez. Seguidamente es necesario representar los caminos básicos por los que puede recorrer el flujo:

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

Para cada camino se realiza un caso de prueba.

Caso de prueba para el Camino básico #1:

Descripción: Los datos de entrada cumplirán con los siguientes requisitos: El node será el identificador

de la entidad, el tipo es el nivel del árbol, el idmodelo es el identificador del plan, el orgrector será un valor booleano(es si la entidad que se logueó es Dirección de Economía), soyCB será un valor booleano (es si la entidad logueada es Centro de balance).

Condición de ejecución: El node tendrá valor cero y el orgrector tendrá valor true.

Entrada: $node = 0, $orgrector = true.

Resultados esperados: Se listan todas las denominaciones de los centros de balance.

Caso de prueba para el Camino básico #2:

Descripción: Los datos de entrada cumplirán con los siguientes requisitos: El node será el identificador de la entidad, el tipo es el nivel del árbol, el idmodelo es el identificador del plan, el orgrector será un valor booleano(es si la entidad que se logueó es Dirección de Economía), soyCB será un valor booleano (es si la entidad logueada es Centro de balance).

Condición de ejecución: El node tendrá valor cero, el orgrector tendrá valor true y el cb cargaría un valor nulo.

Entrada: $node = 0, $orgrector = true.

Resultados esperados: Se muestra un cartel informativo, de que no existen Centros de Balance.

Caso de prueba para el Camino básico #3:

Descripción: Los datos de entrada cumplirán con los siguientes requisitos: El node será el identificador de la entidad, el tipo es el nivel del árbol, el idmodelo es el identificador del plan, el orgrector será un valor booleano(es si la entidad que se logueó es Dirección de Economía), soyCB será un valor booleano (es si la entidad logueada es Centro de balance).

Condición de ejecución: El node tendrá valor cero y el soyCB tendrá valor true.

Entrada: $node = 0, $soyCB = true.

Resultados esperados: Se muestra la denominación del Centro de balance loguiado.

Caso de prueba para el Camino básico #4:

Descripción: Los datos de entrada cumplirán con los siguientes requisitos: El node será el identificador de la entidad, el tipo es el nivel del árbol, el idmodelo es el identificador del plan, el orgrector será un

valor booleano(es si la entidad que se logueó es Dirección de Economía), soyCB será un valor booleano (es si la entidad logueada es Centro de balance).

Condición de ejecución: orgrector tendrá valor false y el soyCB tendrá valor false.

Entrada: $soyCB = true, $orgrector = true.

Resultados esperados: Se muestra la denominación del Órgano Demandante loguiado.

Luego de aplicar los distintos casos de pruebas, se pudo comprobar que el flujo de trabajo de la función es correcto ya que cumple con las condiciones necesarias que se habían planteado.

3.4 Conclusiones del capítulo

En el desarrollo de este capítulo se le dio cumplimiento a la fase de construcción del software generando una primera versión del módulo Centro de balance. Dicha solución posibilita realizar un análisis de las demandas hechas por las entidades, para de esta forma llevar a cabo la consolidación de las mismas y efectuar el balance material. Brinda además, la posibilidad a los usuarios de consolidar las demandas de cada una de las entidades y del propio Centro de balance y la realización del balance material en el momento que lo desee. Cuenta con una interfaz amigable para el usuario, facilitando de esta manera el fácil el manejo de la información, mostrando en todo momento los datos necesarios para llevar a cabo las operaciones.

Conclusiones generales

Una vez terminado el presente trabajo de diploma se puede concluir que se desarrollaron todas las tareas a fin de cumplir los objetivos propuestos.

El módulo brinda la posibilidad a los usuarios de consolidar las demandas de cada una de las entidades y del propio Centro de balance y además de realizar el balance material en el momento deseado. Cuenta con una interfaz amigable para el usuario, de manera que le es fácil el manejo de la información para llevar a cabo las operaciones.

La solución propuesta es novedosa, su importancia radica en la realización de los procesos referentes a la gestión de las operaciones de los Centros de Balance en un único módulo, permitiendo el ahorro de tiempo y recursos, así como el control eficiente de los cambios y la información manipulada, ratificando de esta manera la utilidad y la validez del empleo de las tecnologías informáticas, ganando mayor eficiencia en la planificación productiva del país.

Esta propuesta exhibe valor técnico, donde se destaca la incorporación de principios por los que se mide la factibilidad de un diseño de software, ejemplo: la utilización de patrones que posibilita la reutilización, garantizando la sostenibilidad y mantenimiento del sistema.

Recomendaciones

Teniendo en cuenta el presente trabajo de diploma le dio solución a todas las tareas y objetivos trazados se recomienda:

- ✓ Continuar con el estudio del proceso de Realizar Balance Material con el fin de obtener mayores conocimientos e identificar soluciones comunes para las entidades.
- ✓ Agregar funcionalidades al módulo que permitan una mayor usabilidad del mismo; en este caso: Gestionar comentario para cada una de las demandas.
- ✓ Optimizar los algoritmos para un mejor funcionamiento interno del módulo.
- ✓ Continuar realizando pruebas al módulo para garantizar una mayor calidad del mismo.
- ✓ Implantar la aplicación en varias entidades para validar de forma experimental la utilidad de la misma en la mayor cantidad posibles de escenarios.

Referencias bibliográficas

1. **Molina, Leticia.** SlideShare. *SlideShare.* [En línea] 2009. [Citado el: 10 de Febrero de 2011.] http://www.slideshare.net/lmmmm/que-es-un-erp-y-ejemplos.

2. **Riquelme, María Eugenia.** mailxmail. *mailxmail.* [En línea] 4 de Octubre de 2010. [Citado el: 10 de Febrero de 2011.] http://www.mailxmail.com/curso-administracion-empresas/que-es-planificacion.

3. **Gómez, Giovanny E.** GestioPolis. *GestioPolis.* [En línea] Octubre de 2010. [Citado el: 11 de Febrero de 2011.] http://www.gestiopolis.com/canales/financiera/articulos/no%201/planificacionfinanciera.htm.

4. Openbravo. *Openbravo.* [En línea] [Citado el: 10 de Diciembre de 2010.] http://www.openbravo.com/es/product/erp/procurement-management.

5. COMPLUSOFT. *COMPLUSOFT.* [En línea] [Citado el: 12 de Diciembre de 2010.] http://www.complusoft.es/soluciones/open-erp.

6. SAP. *SAP.* [En línea] [Citado el: 13 de Diciembre de 2010.] http://www.sap.com/mexico/ecosystem/sap_professionals/modules/index.epx.

7. EcuRed. *EcuRed.* [En línea] [Citado el: 12 de Febrero de 2011.] http://www.ecured.cu/index.php/Versat_Sarasola#.C2.BFQu.C3.A9_tiene.3F.

8. **Álvarez, Miguel Ángel.** Desarrolloweb.com. *Desarrolloweb.com.* [En línea] [Citado el: 13 de Febrero de 2011.] http://www.desarrolloweb.com/articulos/392.php.

9. **CEDRUX.** *Documento Especificación de la Arquitectura de Sistema e Integración ERP Cuba.* 2009.

10. **Perera Morales, Jose Raúl.** *Arquitectura de software para sistema gestion de inventarios.* Ciudad de la Habana : s.n., 2007.

11. **ERP-Cuba, Proyecto.** *Arquitectura del ERP.* 2008.

12. **Costilla, Yanisleivi Valdés Fernández y Yoanis.** *Propuesta inicial de un procedimiento, para el modelado de negocio y la gestión de requisitos de proyectos productivos.*

13. **Leyet Fernández, Osmar y Rodríguez Lorenzo, Iosmel.** *Desarrollo de una herramienta generadora de ficheros de mapeo para la persistencia de esquemas de objetos relacionales basada en NHibernate. 2008.*

14. **White, Stephen A.** *IBM Corporation.Introduction to BPMN.*

15. **PC, Territorio.** Territorio PC. *Territorio PC.* [En línea] [Citado el: 15 de Febrero de 2011.] http://www.territoriopc.com/javascript/tutorial_javascript_introduccion.php.

16. **Pérez, Javier Eguíluz.** Librosweb.es. *Librosweb.es.* [En línea] [Citado el: 15 de Febrero de 2011.] http://www.librosweb.es/css.

17. HTML.net. *HTML.net.* [En línea] [Citado el: 16 de Febrero de 2011.] http://es.html.net/tutorials/css/lesson1.asp.

18. **Leopoldo, Carlos.** techtastico. *techtastico.* [En línea] 9 de Marzo de 2006. [Citado el: 9 de Febrero de 2011.] http://techtastico.com/post/glosario-de-negocios/.

19. Abartia Team. *Abartia Team.* [En línea] [Citado el: 2011 de Febrero de 15.] http://www.abartiateam.com/desarrollo-web/200602_uso-de-la-tecnologia-ajax-en-el-desarrollo-web.

20. Herramientas CASE. *Herramientas CASE.* [En línea] [Citado el: 16 de Febrero de 2011.] http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c5/c5.htm.

21. Free Download Manager. *Free Download Manager.* [En línea] [Citado el: 16 de Febrero de 2011.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p.

22. **Almada, Federico.** techtear. *techtear.* [En línea] 22 de Enero de 2008. [Citado el: 20 de Enero de 2011.] http://www.techtear.com/2008/01/22/zend-studio-for-eclipse-desarrollo-profesional-en-php.

23. Mozillla Firefox. *Mozillla Firefox.* [En línea] [Citado el: 13 de Febrero de 2011.] http://www.getfirefox.es/firefox-features.

24. **Sommerville, Ian.** *Ingeniería de Software.* 2005.

25. **Anaya, Victor y Letelier, Patricio.** [En línea] [Citado el: 20 de Febrero de 2011.] http://issi.dsic.upv.es/publications/archives/f-1055508123529/Ideas2003.pdf.

Bibliografía consultada

Rolando Alfredo Hernández, Sayda Coello González. El paradigma cuantitativo de la investigación científica.

SMEToolkit Gestion -de-requerimientos-de-materiales
http://mexico.smetoolkit.org/mexico/es/content/es/185/Gesti%C3%B3n-de-requerimientos-de-materiales

Caja de Herramientas Plan de Negocios para PYME's
http://www.infomipyme.com/Docs/GT/Offline/inicioempresa/PDPP.htm

David Mestre. 2008. [En línea] 2008. [Citado el: 17 de 12 de 2009.] http://davidmaestre.com/2008/01/módulos-de-sap-r3.html.

Departamento de Ingeniería de Sistemas. [En línea] [Citado el: 03 de abril de 2011.]
http://sophia.javeriana.edu.co.

Alvarez, Sara. 2006. Desarrollo web. [En línea] 2006. [Citado el: 12 de febrero de 2011.]
http://www.desarrolloweb.com/articulos/2477.php.

—. 2007. Desarrollo Web. [En línea] 2007. [Citado el: 13 de febrero de 2011.]
http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html.

Bustamante y Paola Hurtado, Diana. 2007. *Patrones GRAPS.* 2007.

Carlos Pes. 2010. Carlospes.com. *Mini Diccionario Informático de Carlos Pes.* [En línea] 2010. [Citado el: 15 de abril de 2011.] http://www.carlospes.com/minidiccionario/manual_de_usuario.php.

CETIC. 2007. Guerrero Gbierno del estado. [En línea] 2007. [Citado el: 02 de abril de 2011.]
http://www.cetic.guerrero.gob.mx/pics/art/articles/113/file.TiposPruebasSoftware.pdf.

—. 2007. Guerrero Gbierno del estado. [En línea] 2007.
http://www.cetic.guerrero.gob.mx/pics/art/articles/113/file.TiposPruebasSoftware.pdf.

Cortés Guzmán, Oscar Hernando. 2004. Willydev. [En línea] 30 de abril de 2004. [Citado el: 25 de marzo de 2011.] http://www.willydev.net/descargas/oguzman-diseno_pruebas.pdf.

Cortés, Oscar Hernando Guzmán. 2004. Willydev. [En línea] 20 de abril de 2004. [Citado el: 15 de abril de 2011.] http://www.willydev.net/descargas/oguzman-diseno_pruebas.pdf.

2008. Doctrine. [En línea] 2008. [Citado el: 02 de abril de 2011.] http://www.doctrine-project.org.

2010. Extjs. [En línea] 2010. [Citado el: 23 de marzo de 2011.] http://www.extjs.com.

Gutierrez Saavedra, Jorge A. 2009. *PATRONES GRAPS. (Patrones de Software para la asignación General de Responsabilidad).* 2009.

2006. SOA. [En línea] 29 de marzo de 2006. [Citado el: 13 de febrero de 2011.] http://arquitecturaorientadaaservicios.blogspot.com/2006/03/pero-qu-es-realmente-soa.html.

2010. PostgreSQL: The world's most advanced open source database Postgresql. [En línea] 2010. [Citado el: 02 de marzo de 2011.] http://www.postgresql.org.

2010. Subversion. [En línea] 2010. [Citado el: 23 de marzo de 2011.] http://subversion.apache.org.

UCID,ERP-CUBA UCI. 2009. *Manual del marco de trabajo.* 2009.

Glosario de términos

- Aplicación web: En la ingeniería software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web (HTML, JavaScript, Java, etc.) en la que se confía la ejecución al navegador.

- Componente: Un componente es una clase de uso específico, que puede ser configurada o utilizada de forma visual desde el entorno de desarrollo.

- Entidad: Empresa, unidad presupuestada u otro tipo de organización similar con una gestión económica, financiera, organizativa, técnica, productiva, comercial, laboral y contractual, con autonomía controlada, en cumplimiento de lo establecido por el Gobierno.

- Framework: Es una estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, 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.

- Herramientas: Es un objeto elaborado a fin de facilitar la realización de una tarea mecánica que requiere de una aplicación correcta de energía. 8. Lenguaje de marcado: Es una forma de codificar un documento que, junto con el texto, incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación.

- Metodología: Se refiere a los métodos de investigación que se siguen para alcanzar una gama de objetivos

- Módulo: Es un software que agrupa un conjunto de subprogramas y estructuras de datos.

- Navegador: Software que permite al usuario recuperar y visualizar documentos de hipertexto desde servidores web a través de Internet.

- Sistema ERP: Es un sistema compuesto por un conjunto de módulos funcionales estándar y que son susceptibles de ser adaptados a las necesidades de cada empresa.

- Sistema gestión: Un sistema de gestión es una estructura probada para la gestión y mejora continua de las políticas, los procedimientos y procesos de la organización.
- Sistema informático: Conjunto de elementos que hacen posible el tratamiento automático de la información.
- Software libre: Es la denominación del software que respeta la libertad de los usuarios sobre su producto adquirido y, por tanto, una vez obtenido puede ser usado, copiado, estudiado, cambiado y redistribuido libremente
- Sistema propietario: Cualquier programa informático en el que los usuarios tienen limitadas las posibilidades de usarlo, modificarlo o redistribuirlo (con o sin modificaciones), o cuyo código fuente no está disponible o el acceso a éste se encuentra restringido.
- Indicador: Notación estándar de cualquier elemento que pueda planificarse, bajo un determinado escenario, el cual se comportará como el elemento base en la planificación.