Universidad de las Ciencias Informáticas
Facultad 3

Desarrollo de la versión 2.0 del módulo Económico para el Sistema de Administración y Economía de la Facultad 3

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

Autor:

Luis Javier Sánchez Hernández

Tutora:

MSc. Ana Marys Garcia Rodríguez

Co-tutor:

Ing. Juan Darién Macías Hernández

La Habana, Julio de 2016

DECLARACIÓN DE AUTORÍA

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

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

Autor:

_____

Luis Javier Sánchez Hernández

Tutora:

_____

MSc. Ana Marys Garcia Rodríguez

Co-tutor:

_____

Ing. Juan Darién Macías Hernández

DATOS DE CONTACTO

Síntesis de la Tutora:
Nombres y apellidos: Ana Marys Garcia Rodríguez.
Correo: agarcia@uci.cu

Profesora Asistente de Ingeniería y Gestión de Software.
Ingeniera en Ciencias Informáticas.
Máster en Calidad de Software.

Síntesis del Co-tutor:
Nombres y apellidos: Juan Darién Macías Hernández.
Correo: macias@uci.cu

Ingeniero en Ciencias Informáticas.

DEDICATORIA

Quisiera dedicar este trabajo especialmente a mis padres los cuales han sido un modelo a seguir durante todo el tiempo que he vivido. Ellos la razón principal de que este momento llegara, pues me alentaron y confiaron en mí a cada momento, siempre estuvieron ahí para aconsejarme en los momentos más difíciles, en los que por un instante llegué a pensar en renunciar a todo lo que había alcanzado.

Dedicarle también este trabajo a mi batallón familiar que siempre me alentaron a seguir sin importar que, a todos ellos muchas gracias.

AGRADECIMIENTOS

Primeramente quisiera agradecer a mi familia por toda la ayuda y apoyo que me ha dado durante todo este tiempo, creo que sin eso nunca lo hubiese logrado.

A Osvaldo q más que primo ha sido un hermano.

Agradecer a Ana Marys la gran tutora… sé que no he sido de los mejores tesistas que has tenido posiblemente el peor, gracias por hacer posible este final, a Macias mi co-tutor que sin él creo que no hubiese escrito una línea de código.

Agradecer también al tribunal y a la oponente, Rosy, siempre ha sido alguien a quien he admirado mucho desde el día en que la conocí, gracias por toda la ayuda que me has brindado en algún momento durante todos estos años.

A todas las amistades que he hecho durante este tiempo, Yordan, Macias, La Poma, la gente de apartamento, a Boris que ha estado en todo momento para brindarme su mano, a Eime, a Ever y todos que son muchos, sería para nunca acabar.

A todos muchas gracias.

RESUMEN

La Universidad de las Ciencias Informáticas tiene como misión la formación de profesionales cualificados y altamente comprometidos con la Revolución. En su estructura organizacional, cada facultad presenta un Vicedecanato de Administración y Servicios que tiene entre sus cometidos la ejecución y control de los procesos económicos. Dado el gran cúmulo de información que se gestiona en el área, se desarrolló el sistema SAEF3 que incluye un módulo Económico, dicho sistema favorece la gestión de los procesos asociados al área económica, pero aún presenta insuficiencias que obstaculizan el control de la información. La presente investigación tiene como objetivo el desarrollo de la versión 2.0 del módulo Económico del SAEF3, para ello se empleó la versión de la metodología Proceso Unificado Ágil establecida por la UCI. La validación de la solución fue desarrollada mediante la aplicación de pruebas de aceptación y se aplicó además la técnica de Iadov, evidenciándose un alto nivel de satisfacción por parte de los clientes.

Palabras clave: control, economía, información, módulo.

ÍNDICE

INTRODUCCIÓN

La economía es la ciencia que estudia los métodos más eficaces para satisfacer las necesidades humanas materiales, mediante el empleo de bienes escasos (RAE 2014). Actualmente constituye la rama que influye en el bienestar social, los avances científicos-técnicos y el medio ambiente, determinando de esta manera el desarrollo de los países. La sociedad se encuentra organizada con mecanismos que permiten adoptar decisiones económicas de forma individual o colectiva, como resultado de su continua evolución. El análisis del entorno económico es vital para conocer en qué condiciones compite una empresa con otras en cualquier nivel, o en qué condiciones se desarrollan los negocios en un sector de la economía del país comparada con otros sectores nacionales e internacionales.

Mientras que la economía es el estudio de cómo las sociedades utilizan recursos escasos para producir bienes valiosos y distribuirlos, la administración es el conjunto de actividades dirigidas al aprovechamiento eficiente y eficaz de estos recursos para el logro de los objetivos. (Orozco et al. 2009). En la gestión administrativa, es esencial la toma de decisiones, cuya calidad está estrechamente asociada a la disponibilidad de información útil para el momento en que se toma la decisión (Davis 2005). Para facilitar la disponibilidad y accesibilidad a la información se han desarrollado diversos sistemas informáticos ajustados a los nuevos paradigmas de las Tecnologías de la Información y las Comunicaciones.

La gestión económico – financiera, se encarga de gestionar administrativamente todos los procesos encaminados a la gestión bancaria, pagaduría, control de la recaudación, tarjetas de crédito, tramitación y pago de indemnizaciones por razón del servicio de los servicios centrales, mediante los registros de presupuesto, ingresos y gastos, además de la elaboración de los estados contables, de la información económica y de las declaraciones fiscales de la universidad.

En la Universidad de las Ciencias Informáticas (UCI) los procesos de administración y gestión económica son dirigidos por el Vicedecanato de Economía y Administración (VDEA) en cada una de las facultades que la integran. El VDEA de la Facultad 3, cuenta con un módulo Económico que integra el Sistema de Administración y Economía (SAEF3), el cual permite establecer un control sobre los procesos asociados al área económica de la facultad; sin embargo, se han identificado insuficiencias que obstaculizan el control de la información, así como la necesidad de incluir nuevas funcionalidades.

Dentro de las insuficiencias identificadas se encuentran:

- El control sobre la asignación de recursos materiales, no permite que los jefes de las áreas consulten el comportamiento de las asignaciones correspondientes a su área. Además, limita la recepción de los recursos solo al jefe del área como único posible receptor.
- El control sobre las solicitudes de materiales a almacenes no brinda información sobre si el abastecimiento de las solicitudes fue total o parcial, ya sea por aprobación del responsable del almacén o por disponibilidad en almacén.
- El control sobre los vales de salida no se realiza a partir de la solicitud previamente elaborada, lo cual dificulta establecer relaciones entre solicitud realizada y vale de salida despachado.
- El control sobre las bonificaciones de pasajes restringe los casos de estudiantes de La Habana y no permite a la facultad manejar casos excepcionales que se presenten.
- El control sobre las ejecuciones del presupuesto no permite visualizar a partir de los vales registrados la ejecución de las partidas, por lo que no se puede constatar si la notificación emitida por la Dirección de Planificación se corresponde con las extracciones realizadas por la facultad.

Como nuevas funcionalidades a implementar se identificaron:

- Gestión de los procesos de entrega y recepción de libros en uso tanto para estudiantes como trabajadores: estos procesos se comenzaron a ejecutar en la facultad en el presente curso, por lo cual no se encuentran implementados en el módulo actual.
- Creación de reportes que permitan establecer análisis de tendencias respecto a las ejecuciones de presupuesto y los planes notificados tanto para la facultad como los centros de desarrollo.

A partir de la problemática anteriormente descrita, se define como problema a resolver: el módulo Económico del SAEF3, presenta insuficiencias en el control de la información asociada a los procesos económicos en la Facultad 3.

Como objeto de estudio se identificó: procesos de gestión económica en universidades.

Se plantea como idea a defender: el desarrollo de la versión 2.0 del módulo Económico para el SAEF3, contribuirá a elevar el control de la información asociada a los procesos económicos en la Facultad 3.

Para dar solución al problema anteriormente planteado se definió el siguiente objetivo general: desarrollar la versión 2.0 del módulo Económico para el SAEF3, de manera que contribuya a elevar el

control de la información de los procesos asociados al área económica de la Facultad 3. Dicho objetivo general se desglosa en los siguientes objetivos específicos:

1. Definir el marco teórico de la investigación mediante el estudio y el análisis de los principales referentes teóricos en torno a los procesos de gestión económica en las universidades.
2. Realizar el diseño e implementación de la solución para obtener los componentes de software de los procesos asociados al área económica de la Facultad 3.
3. Valorar la efectividad de la solución propuesta mediante la realización de pruebas de caja negra, pruebas de aceptación y la aplicación de la técnica Iadov.
4. Validar la variable de la investigación.

Para lograr el cumplimiento de los objetivos especificados se aplicaron los siguientes métodos de investigación:

Teóricos:

- Analítico-Sintético: facilitó el análisis y la comprensión de la documentación consultada, lo cual permitió la realización de un estudio teórico mediante la comparación de sistemas homólogos, para identificar elementos aplicables a la solución.
- Histórico-Lógico: se utiliza para la recopilación de datos históricos sobre el desarrollo de sistemas de administración y gestión económica.

Empíricos:

- Observación: se emplea la obtención de información del entorno a partir de una percepción propia. La observación se aplica durante toda la investigación como método de esclarecimiento.
- Entrevista: mediante el empleo de entrevistas no estructuradas se obtiene el conocimiento necesario para la comprensión del negocio y su funcionamiento.

Se aplicó además la técnica de Iadov para conocer el nivel de satisfacción de los clientes con la solución propuesta.

El trabajo está estructurado en tres capítulos, los cuales se describen a continuación:

Capítulo 1: se realiza un análisis de los referentes teóricos en torno a la gestión económica en las universidades. Además, se hace un estudio de la metodología a emplear para el desarrollo de la propuesta, así como las herramientas y tecnologías aplicables para el desarrollo de la solución.

Capítulo 2: se realiza una descripción de los artefactos generados durante el análisis y diseño del módulo. Se definen los requisitos funcionales y no funcionales estableciendo así las características con las que debe cumplir la solución, y se aplican métricas de validación de requisitos. Se describe el diseño desarrollado especificando los patrones de diseño utilizados y se muestran además, los resultados de la aplicación de las métricas del diseño.

Capítulo 3: se realiza una descripción de los componentes de implementación del módulo, así como de los estándares de codificación empleados. Se especifican los métodos y tipos de pruebas empleados para la verificación y validación de la solución, así como los resultados alcanzados durante la aplicación de la técnica Iadov para medir la satisfacción del cliente, y la validación de la variable de la investigación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se define el marco teórico referencial de la investigación, a partir del análisis de los conceptos asociados al dominio del problema, así como del estudio del estado de arte de los sistemas de gestión económica en las universidades existentes en la actualidad. Además, se describen las tecnologías y herramientas empleadas en el desarrollo del módulo Económico del SAEF3 v2.0.

1.2 Conceptos asociados al dominio del problema

Como conceptos asociados al dominio del problema se tienen: economía, administración y sistema de gestión los cuales son definidos a continuación.

Economía

La economía es una ciencia social que estudia cómo los individuos y sociedades usan o manejan los recursos para satisfacer sus necesidades. Estos recursos pueden ser distribuidos en la producción de bienes y servicios, y en su consumo por diferentes personas de la sociedad (Banco de la República 2015). Por su parte la administración se combina con la economía al dirigir sus esfuerzos hacia el aprovechamiento máximo de estos recursos.

Administración

La administración es el proceso de coordinar las actividades de trabajo mediante la planeación, organización, dirección y control de los recursos para el logro de los objetivos (Orozco et al. 2009).

En la presente investigación se asume el concepto de administración como un conjunto de actividades enfocadas al logro de los objetivos de una organización mediante un correcto control de los recursos con el fin de realizar con eficiencia y eficacia cada proceso, donde los sistemas de gestión juegan un papel importante en la toma de decisiones.

1.3 Análisis de soluciones existentes

En el siguiente epígrafe se realiza un análisis de sistemas homólogos existentes, en busca de una posible solución al problema presentado por el VDEA de la Facultad 3 de la UCI.

1.3.1 SICAP

La Universidad de Valencia, con el fin de mejorar su gestión económica y sus sistemas de información, adquirió en el año 2000 la aplicación informática de contabilidad pública SICAP desarrollada por la empresa TAO (SIUV 2015). Sus objetivos son:

- Mejorar el control de la gestión.
- Aumentar la descentralización de la gestión a nivel de centros y departamentos.
- Establecer un mayor grado de automatización y modernización en la gestión.
- Optimizar los circuitos de información.

Características especiales:

- Base de datos única. La gestión del presupuesto y la contabilidad general son dos "visiones" de la misma información que es única, evitando incoherencias.
- Toda la información bajo el modelo relacional. Ello permite explotarla desde herramientas informáticas como hojas de cálculo o de ayuda a la decisión.
- Definición libre de "Asientos-Patrón" incluida su lógica de tramitación.
- Permite en cada instalación adaptar la normativa contable para que haga cómoda su gestión. Hay adaptaciones a todas las administraciones.
- Permite tanto la gestión centralizada como la gestión descentralizada.

Características funcionales:

- Aplicación modular: dota al sistema contable de las herramientas de control de que carece la contabilidad por partida doble y permite ajustarlas a los requisitos de la instrucción.
- Multi-institución con proceso de consolidación contable entre instituciones.
- Integración de datos externos y carga de datos desde otras aplicaciones: nómina, recaudación.
- Permite trabajar con un nuevo ejercicio sin haber cerrado el anterior.
- Seguimiento de la tramitación presupuestaria de gastos e ingresos.
- Documentación de libre definición asociada a cada posible operación contable.
- Salidas de listados a formatos Access o Excel, salidas gráficas, simulaciones.
- Pre-introducción de terceros.
- Multi-aplicación y registro de facturas.

Características técnicas:

- Servidor de base de datos
- Sistema operativo: AIX

- Gestor de base de datos: ORACLE
- Servidor de terminal Server (10 máquinas)
- Sistema operativo: WINDOWS 2000 ADVANCED SERVER
- Aplicación de contabilidad SICAP + ACCES 97
- Computadoras de usuarios (alrededor de 250)
- Software mínimo: cliente de Terminal Server

1.3.2 UNIVERSITAS XXI - ECONÓMICO

Sistema informático desarrollado por OCU que automatiza la gestión económica de la universidad. UNIVERSITAS XXI - ECONÓMICO está estructurado en componentes que cubren los ámbitos de la contabilidad, gestión económica del gasto e ingreso, tesorería, gestión de la contratación y las compras, fiscalidad, patrimonio, gestión económica de la investigación e intervención (OCU 2016).

RESERVAS DE CRÉDITO

Programa realizado por el SCI que permite a las diferentes unidades de gasto realizar reservas de crédito. Incluye módulos para la petición de la reserva, la comunicación de la misma a la unidad de Gestión Económica y la recepción del código de reserva en caso de ser autorizada.

DOCUMENTOS DE PAGO

Sistema informático desarrollado por el SCI que genera para aquellos servicios que tienen la necesidad de emitir cartas de pago, documentos de pago individualizados mediante un código consecutivo que permite su abono y la posterior identificación del pago.

e-FACTURA

Programa desarrollado por el SCI que permite la digitalización e incorporación automática de los datos de facturas recibidas.

1.3.3 SOROLLA

El proyecto SOROLLA presenta una arquitectura modular, a la vez que integrada, que permite a cada instalación el empleo de los módulos de su interés, sin otra limitación que la establecida por la propia estructura y lógica del sistema, el sistema SOROLLA se desarrolló en 1996 y en el año 2011 se realizó una segunda versión del mismo (DINTEL 2012; PAe 2011)

Las herramientas de gestión contempladas en el proyecto SOROLLA 2 son:

- Un sistema para la tramitación de expedientes de gasto que se realiza a través del módulo *DocumentA* y que alcanza la doble vertiente de la tramitación: administrativa y contable.

- Un sistema de gestión de justificantes de gasto que se instrumenta a través del módulo denominado *Justificantes del Gasto*, que permite la gestión de las cajas pagadoras en lo referente a los pagos que se realizan mediante el sistema de "A justificar", tanto por Acuerdo de Caja Fija como por Pagos a Justificar.
- Un sistema para la elaboración de documentos contables a través del módulo *Docuconta* integrado con los módulos de *DocumentA* y *Justificantes del Gasto*.
- Un sistema de gestión del inmovilizado material instrumentado a través del módulo de gestión de *Inventario* que permite el mantenimiento actualizado tanto desde el punto de vista de los detalles de su composición como en su faceta contable.
- Además de la gestión de comisiones de servicio incorporada al módulo de *Justificantes del Gasto*, dispone de un módulo independiente *SOROnet*, desarrollado bajo tecnología Web que permite a los propios comisionados introducir en el sistema los datos de la comisión, solicitar adelanto si procede, obtener la orden de comisión de servicios y generar la correspondiente liquidación con carácter provisional.

Junto a estas herramientas de gestión, se contemplan otras para el tratamiento de la información:

- Un módulo de elaboración de informes de ejecución presupuestaria, en base a los datos tratados en los sistemas de gestión.
- Un módulo para la elaboración de la documentación de carácter tributario relativa a los pagos realizados a personas físicas o jurídicas y otro correspondiente al resumen anual de retenciones e ingresos.
- Un módulo generador de informes personalizados a través de la vinculación de vistas de la base de datos SOROLLA a una base de datos de tipo ACCESS97/ACCESS2000/ACCESS2003.

Descripción técnica

El sistema SOROLLA ha sido desarrollado bajo la arquitectura cliente-servidor con productos de Microsoft. En la actualidad hay dos formas de instalar SOROLLA dentro de una red:

- Mediante una instalación cliente-servidor pura, es decir, en cada puesto cliente se instalan y por lo tanto se ejecutan los programas que componen el aplicativo.
- Utilizando el servicio Terminal Server de Microsoft, en la que los programas se instalan en un servidor de aplicaciones o incluso en el servidor de datos y en cada puesto cliente se instala la parte cliente de este servicio.

En lo relativo a las características técnicas de los programas y bases de datos empleados en SOROLLA, es válido indicar que la línea estratégica establecida pretende, en la medida de lo posible,

ser independiente de costes de licencias de software y vincular el proyecto a un entorno tecnológico adecuado con una proyección de futuro asegurada.

1.3.4 Sistema Integral de Gestión Económica (S.I.G.E.)

S.I.G.E. da soporte a la gestión económico-administrativa de la Universidad de Córdoba abarcando diferentes áreas relacionadas entre sí: Gestión Económica, Contratación y Patrimonio, Auditoría, Gerencia. Fue desarrollado por personal del Servicio de Informática de la universidad junto con los gestores contables de la misma y se adapta a las necesidades existentes en su universidad (Córdoba 2014).

Funcionalidades que ofrece:

- Pagos: contempla la introducción de justificantes del gasto (facturas), permitiendo una reserva de crédito en el presupuesto implicado desde el momento de la grabación de la misma en el caso de las facturas en firme. Se pueden grabar también los justificantes de los pagos a justificar. Se gestionan las operaciones en divisas y la generación de los documentos contables de pago de justificantes.
- Ingresos: se recogen tanto las liquidaciones como la recaudación de los ingresos permitiendo un control de los ingresos pendientes de aplicar. Se generan los documentos contables correspondientes y permite el control de la evolución de las previsiones de ingresos en el ejercicio.
- Gestión del presupuesto: introducción y actualización del presupuesto inicial de la universidad, modificación del presupuesto por incorporaciones y anulaciones, gestión de expedientes presupuestarios, repartos presupuestarios, certificaciones de las diferentes ayudas recibidas, obtención de información presupuestaria (mayores, balances, diarios, entre otros).
- Tesorería: gestión de los anticipos de caja y transferencias del banco en casa a las unidades de gasto. Consolidación bancaria, transferencias bancarias, emisión de cheques y de pagarés, ordenación del pago y pago de documentos contables. Gestión de pagos por confirming.
- Auditoría y control interno: firmas autorizadas, fiscalización, control y verificación de todas las operaciones realizadas en el sistema.
- Contratación: gestiona los expedientes de contratación que se realizan y todas las fases implicadas en los mismos. Permite generar a partir de las certificaciones, los documentos de pago correspondientes, así como las autorizaciones, disposiciones y obligaciones, contabilizándose y apuntándose directamente en la contabilidad.

- Inventario: mantenimiento de todos los bienes de la universidad, traspasos y traslados de bienes, consolidación de inventario, incorporación directa de los justificantes de compra con las hojas de inventario, gestión de suministros.
- Cierre del ejercicio económico: generación de movimientos, cuentas e informes de cierre.
- Informes y estadísticas: S.I.G.E. incluye todos los listados necesarios para una correcta gestión de la contabilidad universitaria, así como todas las declaraciones, formatos bancarios y libros solicitados por organismos y empresas ajenos a la universidad.

1.3.5 SAEF3, módulo Económico v1.0

La versión 1.0 del módulo Económico de SAEF3, es el sistema actualmente utilizado por el VDEA de la facultad 3 para el manejo y control de los procesos económicos llevados a cabo por la facultad. Dicho módulo tiene como objetivo:

- Control de Prestamos o ayuda económica a alumnos.
- Control de recursos asignados a las áreas de la facultad 3.
- Ejecución y control de la(s) bonificación(es) de pasajes a estudiantes de pasajes a estudiantes.
- Control de ejecución del presupuesto de la facultad 3.
- Generar reportes para análisis y comparación en el tiempo.

A continuación se muestra mediante tablas comparativas el análisis de los sistemas homólogos, atendiendo a tres indicadores que responden a la soberanía tecnológica por la cual aboga nuestro país y además, teniendo en consideración los procesos asociados al área económica que se realizan en el VDEA de la Facultad 3 (Ver Tablas 1 y 2).

Sistemas	Exento de pago	Open Source	Multiplataforma
SICAP	-	-	-
UNIVERSITAS XXI - ECONÓMICO	-	-	✓
SOROLLA 2	✓	-	✓
S.I.G.E.	-	-	✓
SAEF3	✓	✓	✓

Tabla 1. Comparación entre los sistemas

Fuente: (Elaboración propia)

Sistemas / Procesos	SICAP	UNIVERSITAS XXI - ECONÓMICO	SOROLLA	S.I.G.E	SAEF3
Asignación de recursos materiales a las áreas	✓	✓	✓	✓	✓
Solicitud de recursos materiales y vales de salida	✓	✓	✓	✓	✓
Bonificación de pasajes a estudiantes	-	-	-	-	✓
Préstamo y devolución de libros en uso	-	-	-	-	-
Plan y ejecución del presupuesto	✓	✓	✓	✓	✓

Tabla 2. Procesos del VDEA que ejecutan los sistemas

Fuente: (Elaboración propia)

Los sistemas anteriormente estudiados no satisfacen en su totalidad las necesidades de informatización del VDEA de la Facultad 3, SAEF3 constituye el sistema que mayor cantidad de procesos implementa y respecto a los indicadores de soberanía tecnológica abogados por nuestro país es el único que los cumple a cabalidad. Por lo antes expuesto, se considera necesario el desarrollo de la versión 2.0 del módulo Económico para SAEF3.

1.4 Proceso de desarrollo de software

Un proceso de desarrollo de software es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto. Este conjunto de actividades tienen la misión de transformar los requerimientos del usuario en un producto de software; de manera que los integrantes del equipo y todo aquel interesado en el producto final, tengan la misma visión (Jacobson et al. 2000).

1.4.1 Metodología de desarrollo

Una metodología consiste en múltiples herramientas, modelos y métodos para asistir en el proceso de desarrollo de software donde se definen con precisión los artefactos, roles, actividades e involucrados, junto con prácticas y técnicas recomendadas, guías de adaptación de la metodología al proyecto y guías para uso de herramientas de apoyo (Figueroa et al. 2008).

Para la selección de la metodología a emplear se consideraron las características del proyecto, el cual está compuesto por un equipo de desarrollo pequeño, el cliente se encuentra disponible a tiempo completo para el desarrollo de la solución y se cuenta con un tiempo relativamente corto para la entrega del producto final. Por tanto, se consideró emplear una metodología con un enfoque ágil, siendo AUP en su versión UCI la seleccionada para este trabajo.

Metodología AUP versión de la UCI

Se seleccionó la metodología aprobada por la universidad para su actividad productiva de tal forma que se adapte a su ciclo de vida, la cual consiste en una variación de la metodología Proceso Unificado Ágil (AUP por sus siglas en inglés). La variación AUP-UCI queda estructurada en tres fases, que son detalladas a continuación (Sánchez 2014).

En la siguiente tabla se muestran los objetivos de cada una de las fases de la metodología enunciada con anterioridad:

Fases AUP	Fases Variación AUP-UCI	Objetivos de las fases (Variación AUP-UCI)
Inicio	Inicio	Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
Elaboración Construcción Transición	EJecución	En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. Durante esta fase el producto es transferido al ambiente de los usuarios finales o entregado al cliente. Además, en la transición se capacita a los usuarios finales sobre la utilización del software.
	Cierre	En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Figura 1: Ciclo de vida de la Metodología AUP-versión UCI

Fuente: (Sánchez 2014)

La presente investigación se centra en el desarrollo de la fase Ejecución de la metodología.

1.4.2 Ingeniería de requisitos

La ingeniería de requisitos es un conjunto de procesos, tareas y técnicas que permiten la definición y gestión de los requisitos de un producto de modo sistemático (Pressman 2010). La ingeniería de requisitos permite la gestión adecuada de los requisitos de un proyecto de desarrollo software. Además, mejora la capacidad para realizar planificaciones de los procesos de proyectos de desarrollo software puesto que el conocer qué se tiene que desarrollar, permite una efectiva proyección de las actividades, recursos, costos y tiempos del proyecto (Sommerville 2005).

Llevar a cabo de manera adecuada el proceso de ingeniería de requisitos, disminuye la probabilidad de fracaso de un proyecto. Esto se debe a que los requisitos bien definidos permiten conocer de un modo conciso qué debe ser capaz de realizar el software y orienta las actividades, recursos y esfuerzos de manera eficiente permitiendo la disminución de costos y retrasos. Todo ello provoca una mejora en la calidad del software, puesto que los requisitos bien especificados podrán probarse de manera efectiva (Ramírez 2012).

Para el desarrollo de la solución de un software no solo es relevante identificar las características que debe presentar el sistema, sino también establecer la arquitectura base por la cual se rige la solución técnica.

1.4.3 Patrones arquitectónicos

Los patrones arquitectónicos son la respuesta a los problemas de arquitectura de software. Realizan una descripción de los elementos y de la relación existente entre ellos, junto a un grupo de restricciones sobre cómo pueden ser usados. Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones (Buschmann et al. 2007).

1.4.3.1 Patrón arquitectónico Modelo-Vista-Controlador (MVC)

MVC es un estilo arquitectónico usado principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de los conceptos para que el desarrollo esté estructurado de una mejor manera. Facilita la programación en diferentes capas de forma paralela e independiente. MVC sugiere la separación del software en tres estratos: el modelo, la vista y el controlador.

El patrón de llamada y retorno MVC, se ve frecuentemente en aplicaciones web, donde la vista es una página de Lenguaje de Marcado de Hipertexto (HTML por sus siglas en inglés) y el código que provee datos dinámicos a la página. El modelo incluye el Sistema de Gestión de Base de Datos (SGBD) y la lógica de negocio, mientras que el controlador es el responsable de recibir y atender los eventos de entrada desde la vista (Buschmann, Henney and Schmidt 2007; Díaz and Fernández 2012).

1.4.4 Patrones de diseño

Un patrón de diseño es una descripción de clases y objetos que se comunican entre sí para resolver un problema de diseño general en un contexto específico. Cada patrón describe un problema que ocurre una y otra vez, y describe la esencia de la solución a ese problema, de tal modo que pueda

utilizarse dicha solución un millón de veces más, sin siquiera hacerlo de la misma manera dos veces (Larman 2003; Pressman 2010).

Patrones Generales de Asignación de Responsabilidades (GRASP por sus siglas en inglés)

Para el desarrollo del módulo se aplican los patrones de diseño GRASP, los cuales describen los principios fundamentales de la asignación de responsabilidades a objetos. Entre los más conocidos y utilizados para la solución están (Pressman 2010):

- Experto: cada clase dentro del módulo tiene la responsabilidad de utilizar únicamente la información que ella misma posee para realizar la labor para la que fue concebida. Tal es el caso de las clases del modelo que son las encargadas de toda la lógica del acceso a los datos.
- Creador: identifica quien debe ser el responsable de la creación de nuevos objetos o clases, donde la nueva clase deberá ser creada por la clase que tiene toda la información necesaria para realizar la acción, que usa directamente las instancias creadas del objeto, almacena o maneja varias instancias de clase y contiene o agrega la clase.
- Alta cohesión: cada clase que pertenece al modelo tiene como responsabilidad fundamental realizar las labores que solo le competen a ella y que no son desempeñadas por otros elementos del diseño. Dentro de esas labores se encuentra crear las consultas de acceso a los datos. Todas las clases están agrupadas por las funcionalidades que realizan.
- Bajo acoplamiento: las clases que implementan la lógica del negocio y de acceso a datos se encuentran en el modelo, las cuales no tienen asociaciones con las de la vista, lo que proporciona que la dependencia en este caso sea baja.
- Controlador: el módulo tiene clases controladoras que se encargan de atender todas las peticiones y pasar los datos de la misma a las clases del modelo para su procesamiento. Al mismo tiempo es la clase que se encarga de enviar las respuestas a la vista.

Mediante la aplicación de estos patrones se asignan responsabilidades a las clases para que realicen solo las funcionalidades correspondientes a la información que ellas contienen. También se le otorga la responsabilidad de crear instancias de otras clases solamente a aquellas que las contengan y se definen las clases encargadas de controlar el flujo de las operaciones del sistema.

Patrones GoF

Adaptados para solucionar un problema común en particular mediante la descripción de las clases y la comunicación entre objetos; se agrupan en tres categorías: creacionales, estructurales y de comportamiento (Gamma et al. 2002).

Creacionales:

- Inicialización vaga: refleja la delegación de la creación de un objeto o el cálculo de un valor hasta que sea realmente necesitado.

Estructurales:

- Adaptador: permite que la interfaz de una clase existente pueda ser usada desde otra interfaz sin necesidad de cambiar su código. Como ejemplo de ello está el motor de plantillas *Twig* donde cada plantilla redefine los bloques a su beneficio sin necesidad de cambiar la base.

Comportamiento:

- Iterador: provee una manera de acceder a los elementos de un objeto sin necesidad de exponer su representación interna, facilitando las iteraciones y el filtrado sobre colecciones de objetos.

1.4.5 Estándares de codificación

La comunidad detrás de PHP es enorme y muy diversa, por lo que está compuesta de innumerables librerías, *frameworks* y componentes. Así que es común que combinar varios proyectos PHP diferentes en una misma aplicación. Por eso es importante que los diferentes proyectos utilicen, en la medida de lo posible, el mismo estilo para escribir su código.

El *Framework Interop Group* (antes conocido como el *PHP Standards Group*) ha propuesto y aprobado una serie de recomendaciones de estilo, conocidas como PSR-0, PSR-1 y PSR-2.

Las recomendaciones son simplemente un conjunto de normas que los proyectos PHP más importantes como Drupal, Symfony, phpBB, Silex y Laravel han empezado a adoptar.

El código fuente de Symfony sigue los estándares de codificación PSR-1 y PSR-2 que fueron definidos por la comunidad PHP (Symfony 2015).

Por otra parte, el proyecto PHP Coding Standards Fixer publicado por Fabien Potencier permite modificar automáticamente el código fuente de cualquier proyecto para que cumpla cualquiera de las normas anteriores. En cuanto a los nombres de variables, clases y funciones, la recomendación consiste en utilizar exclusivamente el idioma inglés. Los comentarios de código sí que se pueden escribir en el idioma que mejor entiendan los programadores del equipo de desarrollo (Refsnes-Data 2016).

1.4.6 Calidad de software

La calidad del software se define como el grado con que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario (IEEE 1990). Pressman, incorpora a esta definición un elemento importante referente a la medición del producto, pues establece que la calidad del software proporciona un valor medible para los que lo producen y lo usan que se pueden distinguir entre requerimientos y necesidades (Pressman 2010).

La calidad del software debe medirse en cada una de las etapas de desarrollo para evitar errores que constituyan muy costosos de implementar en etapas de un avance en la implementación.

Verificación y validación de software

La verificación y validación (V&V) de software abarca un conjunto de actividades encaminadas a verificar si el producto construido cumple correctamente con los requisitos establecidos, también evalúan si el producto satisface el cumplimiento de las necesidades del cliente, si el software desarrollado es el correcto. Entre estas actividades se encuentran: auditorías de calidad y configuración, monitoreo de desempeño, revisiones técnicas, estudio de viabilidad, revisiones de documentación y bases de datos, análisis de algoritmos y pruebas de desarrollo, calidad, aceptación e instalación. Las V&V también son definidas como los procesos de chequeo y análisis para asegurar que el software desarrollado cumpla con las especificaciones y funcionalidades esperadas por el cliente, estos procesos son realizados durante todo el ciclo del desarrollo del software (Pressman 2010; Sommerville 2005).

Métricas para la validación de requisitos

En el desarrollo del software los requisitos constituyen el eslabón fundamental para el desarrollo de las etapas posteriores de producción y de ellos depende en gran medida el éxito o fracaso de un proyecto, por ello es importante que exista una garantía de que estén correctamente especificados y que en realidad satisfagan las necesidades del cliente que solicita el software. La validación de los requisitos debe estar orientada a certificar que los requisitos definidos en el cumple con los siguientes atributos de calidad (Davis et al. 1993; Garcia et al. 2013).

No ambiguo:

Un requisito es ambiguo si y solo si tiene una única posible interpretación. Este constituye uno de los atributos más importantes, múltiples interpretaciones de un requisito pueden causar desacuerdos entre clientes y desarrolladores.

Una manera de medir este atributo es a partir del porcentaje de requisitos interpretados de una única manera.

Este atributo puede ser medido de la siguiente manera:

$$Q_1 = n_{ui} / n_r$$

Donde

n_{ui} = número de requisitos con una única interpretación, n_r = número total de requisitos documentados

Cuando el valor resultante se aproxima a 0, se interpreta como que cada requisito tiene múltiples interpretaciones; cuando el valor se aproxima a 1, se interpreta que cada requisito tiene una única interpretación. Debido a que este atributo es muy crítico para el éxito de un proyecto se recomienda que Q1 = 1.

Comprensible:

Un requisito es comprensible cuando todos los lo leen (clientes, usuarios, desarrolladores) comprenden el significado de todos los requisitos con un mínimo de explicación.

Este atributo puede ser medido de la siguiente manera:

$$Q_2 = n_{ui} / n_r$$

Donde

n_{ui} = número de requisitos que todos los revisores entienden, n_r = número total de requisitos

Correcto:

Un requisito es correcto si y solo si todos los requisitos representan una necesidad de función del sistema a construir

Este atributo puede ser medido de la siguiente manera:

$$Q_3 = n_c / n_r$$

Donde

n_c = número de requisitos correctos, n_r = número total de requisitos

Métricas para la validación del diseño

“La medición es fundamental para cualquier disciplina de ingeniería y la ingeniería de software no es una excepción. La medición nos permite tener una visión más profunda proporcionando un mecanismo para la evaluación objetiva” (Pressman 2010).

Las métricas de software se refieren a un amplio elenco de mediciones para el software, la medición se puede aplicar al proceso del software con el intento de mejorarlo sobre una base continua, se pueden utilizar para ayudar a la estimación, el control de la calidad, la evaluación de productividad y el control de proyectos (Pressman 2010).

En el libro de métricas del diseño realizado por Lorenz y Kidd, dividen las métricas basadas en clases en cuatro categorías: tamaño, herencia, valores internos y valores externos. Las métricas orientadas a tamaños para una clase se centran en cálculos de atributos y de operaciones para una clase individual, y promedian los valores para el sistema orientado a objetos en su totalidad. Las métricas basadas en herencia se centran en la forma en que se reutilizan las operaciones a lo largo y ancho de la jerarquía de clases. Las métricas para valores internos de clase examinan la cohesión y asuntos relacionados con el código, y las métricas orientadas a valores externos examinan el acoplamiento y la reutilización (Lorenz and Kidd 1994; Pressman 2010; Sánchez 2010).

Chidamber y Kemerer también realizan una propuesta para medir la calidad del diseño, entre ellas: Métodos Ponderados por Clases (MPC), Árbol de Profundidad de Herencia (APH), Numero de Descendientes (ND), Acoplamiento entre Clases Objeto (ACO), Relaciones entre Clases (RC) y Carencia de Cohesión de los Métodos (CCM) (Chidamber and Kemerer 1994; Pressman 2010).

Para el contexto de la investigación se consideró oportuno aplicar la métrica TC de Lorenz y Kidd, para visualizar si se distribuyen correctamente las asignaciones de responsabilidades entre las clases, y de Chidamber y Kemerer las métricas RC para evaluar el acoplamiento entre las clases y CCM para verificar la cohesión entre las mismas.

Estrategia de pruebas de software

Una estrategia de prueba de software integra las técnicas de diseño de casos de pruebas en una serie de pasos bien planificados que dan como resultado una correcta construcción del software. La estrategia describe los pasos que hay que llevar a cabo como parte de la prueba, el diseño de los casos de pruebas, la ejecución de las pruebas y la agrupación y evaluación de los datos resultantes. (Pressman 2010).

Métodos de prueba

- Pruebas de caja negra: se centran en verificar el cumplimiento de los requisitos funcionales del software. Permiten obtener un conjunto de entradas que ejerciten completamente todos los requisitos funcionales de un programa. Intenta encontrar errores de las siguientes categorías:
    - Funciones incorrectas o ausentes
    - Errores de interfaz
    - Errores en estructuras de datos o en accesos a las bases de datos externas
    - Errores de rendimiento
    - Errores de inicialización y terminación
- Pruebas de caja blanca: se centran en evaluar la ejecución por lo menos una vez de cada sentencia del programa. Estas pruebas deben garantizar como mínimo que:

- Se ejerciten por lo menos una vez todos los caminos independientes para cada módulo.
- Se ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsas.
- Se ejerciten las estructuras internas de datos para asegurar su validez.

El software debe cumplir con ciertos requerimientos de calidad los cuales se encuentran en estrecha correspondencia con los Requisitos No Funcionales (RNF) del sistema. Para verificar que el sistema cumpla con estos atributos de calidad se aplican diferentes tipos de pruebas

Tipos de prueba

Entre los tipos de pruebas que pueden aplicarse se encuentran (Hetzel 1993)

- **Funcionalidad:** habilidad del software para realizar el trabajo deseado.
- **Usabilidad:** habilidad del software para satisfacer al usuario.
- **Portabilidad:** habilidad del software para correr en diferentes entornos informáticos.

El principal objetivo del flujo de pruebas es evaluar la calidad del producto a través de la búsqueda y documentación de errores, lo que permite verificar hasta q punto parecen funcionar las funcionalidades del software de acuerdo con las especificaciones, además los datos recogidos durante las pruebas proporcionan una buena indicación de la fiabilidad del software y de alguna manera indican la calidad del software como un todo.

Mientras que la comprobación de que el producto se realizó correctamente es de vital importancia, también es importante comprobar que el producto realizado es el correcto, para lo cual es aconsejable la aplicación de técnicas que permitan medir la satisfacción del cliente con respecto al producto.

Técnica Iadov

La técnica Iadov permite medir la satisfacción del cliente con un producto, se compone de cinco preguntas claves: tres cerradas y dos abiertas, las cuales se reformulan en la investigación para valorar el grado de satisfacción de los clientes sobre un tema en específico. Una vez establecidas las preguntas se conforma el cuadro lógico de Iadov y el número resultante de la interrelación de las tres preguntas, indica la posición de los sujetos en la escala de satisfacción. La escala de satisfacción está dada por los criterios (Kuzmina 1970).

1. Máxima satisfacción.
2. Más satisfecho que insatisfecho.
3. No definida.

4. Más insatisfecho que satisfecho.
5. Máxima insatisfacción.
6. Contradictoria.

Pregunta cerrada 3	Pregunta cerrada 1								
	No			No sé			Sí		
	Pregunta cerrada 2								
	Sí	No sé	No	Sí	No sé	No	Sí	No sé	No
Me gusta mucho	1	2	6	2	2	6	6	6	6
No me gusta mucho	2	2	3	2	3	3	6	3	6
Me da lo mismo	3	3	3	3	3	3	3	3	3
Me disgusta más de lo que me gusta	6	3	6	3	4	4	3	4	4
No me gusta nada	6	6	6	6	4	4	6	4	6
No sé qué decir	2	3	6	3	3	3	6	3	4

Tabla 3: Cuadro lógico de Iadov

Fuente: (Kuzmina 1970)

Para obtener el índice de satisfacción grupal (ISG) se trabaja con los diferentes niveles de satisfacción que se expresan en la escala numérica que oscila entre +1 y - 1 de la siguiente forma:

Índice de satisfacción	**Escala**
Máxima satisfacción	+1
Más satisfecho que insatisfecho	0,5
No definido y contradictorio	0
Más insatisfecho que satisfecho	-0,5
Máxima insatisfacción	-1

Tabla 4: Escala de satisfacción de Iadov

Fuente: (Kuzmina 1970)

La satisfacción grupal (ISG) se calcula por la siguiente fórmula:

$$\text{ISG} = \frac{A(+1) + B(+0{,}5) + C(0) + D(-0{,}5) + E(-1)}{N}$$

Donde:

A representa el número de sujetos con índice individual 1

B representa el número de sujetos con índice individual 2

C representa el número de sujetos con índice individual 3 o 6

D representa el número de sujetos con índice individual 4
E representa el número de sujetos con índice individual 5
N representa el número total de sujetos del grupo

1.5 Herramientas y tecnologías a utilizar

Para el desarrollo de la segunda versión de la aplicación se utilizarán como herramientas y lenguajes de programación:

1.5.1 Servidor web: Apache server

Apache uno de los servidores web más populares en la actualidad. Es el servidor web que más sitios en internet maneja actualmente.
Entre las características más relevantes de Apache se encuentran:

- Apache es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a al Protocolo de Transferencia de Hipertexto (HTTP).
- Multiplataforma.
- Modular: puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la Aplication Programming Interface (API) de programación de módulos, para el desarrollo de módulos específicos.
- Extensible: gracias a ser modular se han desarrollado diversas extensiones entre las que destaca HyperText Preprocessor (PHP), un lenguaje de programación del lado del servidor.

1.5.2 Lenguajes de programación

Debido a que la aplicación está orientada a la web y la arquitectura utilizada en la misma es la arquitectura Cliente/Servidor, se utilizara para la programación del lado del cliente el lenguaje interpretado HTML en su versión 5 y para la programación del lado del servidor PHP ya que este es un lenguaje que el propio servidor interpreta y ejecuta.

HTML5

HTML5 es mantenido por la World Wide Web Consortium (W3C), es utilizado para la creación de páginas web, en su quinta versión incorpora algunas nuevas etiquetas para lograr una estructura más lógica y funcional en las páginas web. HTML5 está definido en base al Document Object Model (DOM), la representación interna de una web con la que trabaja un navegador, dejando de lado la representación real y definiendo a la vez un estándar HTML y XHTML (Refsnes-Data 2016).

Hojas de Estilos en Cascada (CSS)

Para la confección del visual, se empleó Boostrap en su versión 3, ya que es la mejor forma de separar el contenido de la página de su presentación lo cual mejora la accesibilidad de la página desde disímiles dispositivos y facilita el mantenimiento y perfeccionamiento de dicha página (Refsnes-Data 2016).

PHP 5.4.4-14

PHP es un lenguaje utilizado para la programación del lado del servidor, es interpretado y utilizado para la creación de páginas web dinámicas, posee una gran librería de funciones y una amplia documentación para su utilización.

PHP es un lenguaje de código abierto, multiplataforma y de alto rendimiento, posee gran capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, un escaso consumo de recursos y es simple y de fácil aprendizaje para programadores principiantes, pero a su vez, ofrece muchas características avanzadas para los programadores profesionales (Refsnes-Data 2016).

1.5.3 Entorno de Desarrollo Integrado (IDE)

PhpStorm 8

Trae consigo tecnologías emergentes para ayudar a disfrutar el desarrollo web con un entendimiento profundo del código y soporte avanzado para ambientes remotos. PhpStorm ofrece un editor enriquecido e inteligente para PHP que realmente comprende el código y entiende profundamente su estructura, soportando PHP 5.3, 5.4, 5.5 & 5.6 para proyectos modernos y legados. El IDE provee finalización inteligente de código, señalamiento de sintaxis, configuración extendida de formato de código, chequeo de errores al instante, plegado de código, soporte para mezclas de lenguajes y más (JetBrains 2016)

1.5.4 Sistema Gestor de Bases de Datos (SGBD)

Un SGBD es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones, permite definir datos a distintos niveles de abstracción, la manipulación de dichos datos, así como garantizar la integridad y seguridad de dichos datos.

PostgreSQL v9.2

PostgreSQL es un SGBD relacional, libre y orientado a objetos, es multiplataforma, confiable, con gran estabilidad y escalabilidad, posee gran control de concurrencia y funcionalidades que lo destacan como uno de los SGBD más potentes en la actualidad (PostgreSQL 1996).

Dentro de sus principales ventajas se encuentran:

- Soporta distintos tipos de datos.
- Puede ser instalado un número ilimitado de veces sin temor de sobrepasar la licencia.
- Posee estabilidad y confiabilidad legendaria.
- Soporte nativo para los lenguajes más populares del medio: PHP, C++, Perl, Python, entre otros.
- Extensiones para alta disponibilidad, nuevos tipos de índices, datos especiales, minería de datos, entre otros.

PgAdmin 3 v1.14.2-2

PgAdmin 3 es una aplicación multiplataforma diseñada para el manejo y diseño de base de datos, responde a las necesidades de cualquier usuario, desde una simple consulta en SQL hasta el desarrollo de bases de datos complejas. Su interfaz gráfica soporta todos los componentes de PostgreSQL y permite una fácil administración del mismo, la aplicación también incluye un editor señalamientos de sintaxis SQL, un editor de código server-side, un agente de SQL/batch/shelljobscheduling y soporte para motores de replicación Slony-I. Permite realizar conexiones al servidor utilizando puertos TCP/IP o Dominios Unix, y el encriptado mediante SSL para protección (pgAdmin 2016; PostgreSQL 1996).

1.5.5 Marco de Trabajo (Framework)

Symfony 2

Es un framework para el desarrollo de aplicaciones web mediante PHP5, su diseño permite una mayor optimización de las aplicaciones, así como una gran organización de la misma lo que facilita el proceso de creación y mantenimiento de las aplicaciones. Es un framework multiplataforma, de alto rendimiento, usabilidad, extensibilidad, así como flexibilidad por lo que es compatible con la mayoría de los gestores de base de datos y permite el desarrollo por capas pues está basado en el patrón arquitectónico MVC, posee una arquitectura interna completamente desacoplada, lo que permite reemplazar o eliminar fácilmente aquellas partes que no encajan en el proyecto (Potencier 2016).

jQuery1.9

JQuery es un nuevo tipo de librería JavaScript que simplifica el recorrido de documentos HTML, el manejo de eventos, animaciones e interacciones Ajax para un rápido desarrollo de aplicaciones web. Esta nueva versión del framework es mucho más liviana y limpia que sus versiones anteriores pues muchas de sus funcionalidades ya obsoletas han sido retiradas del mismo, así como la corrección de bugs existentes (JQuery-Fundation 2016).

1.5.6 Herramientas CASE (Computer Aided Software Engineering)

Visual Paradigm 8.0

Es una herramienta CASE profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML (Visual-Paradigm 2016).

Lenguaje Unificado de Modelado (UML)

Se selecciona UML 2.0 como lenguaje de modelado pues presenta un conjunto de herramientas para modelar sistemas orientados a objetos. Permite visualizar, especificar, construir y documentar el sistema a desarrollar cada uno de los de artefactos durante las primeras fases del ciclo de vida del software. También incluye características conceptuales como procesos de negocios, funciones del sistema, esquemas de bases de datos (Rumbaugh et al. 2000).

1.6 Conclusiones parciales

En el presente capítulo se plantearon los principales conceptos asociados al objeto de estudio mediante lo cual se posibilitó tener una mejor visión del problema, partiendo del estudio de soluciones existente y la comparación entre estas. Se puede afirmar que las mismas no brindan una solución completa al problema existente en el VDEA de la Facultad 3, lo que sirvió de base para un mejor entendimiento de los sistemas estudiados y para un mayor análisis de la problemática planteada, aportando ideas para el diseño y desarrollo del sistema.

Dadas las características del equipo de desarrollo y el entorno de producción, se decidió el empleo de la metodología AUP en su variación UCI. Además, se consideró oportuno aplicar métricas para la validación de la calidad de los requisitos y el diseño. Como parte de la estrategia de pruebas se determinó la aplicación de pruebas de funcionalidad, usabilidad y portabilidad, así como el método de prueba caja negra. Para medir la satisfacción del cliente respecto a la solución, se decide aplicar la técnica de Iadov.

Para el desarrollo de la solución se determinó el empleo de herramientas y tecnologías como: servidor web apache server, HTML5, PHP5.4.4-14, PhpStorm 8, PostgreSQL v9.2, pgAdmin3 v1.14.2-2, Symfony 2.5.6, Bootstrap 3, jQuery1.9, Visual Paradigm 8.0 y UML.

CAPÍTULO 2: MÓDULO ECONÓMICO DEL SAEF3 V 2.0

2.1 Introducción

El presente capítulo refleja el resultado del levantamiento de los requisitos funcionales a implementar, así como los no funcionales. Se realiza además la descripción de los artefactos del análisis y el diseño de la solución.

2.2 Diagrama de procesos de negocio

Para lograr un mayor entendimiento de los procesos de negocios de VDEA, se desarrollaron los diagramas de procesos del negocio los cuales permiten visualizar de forma gráfica la secuencia lógica de las actividades de cada proceso.

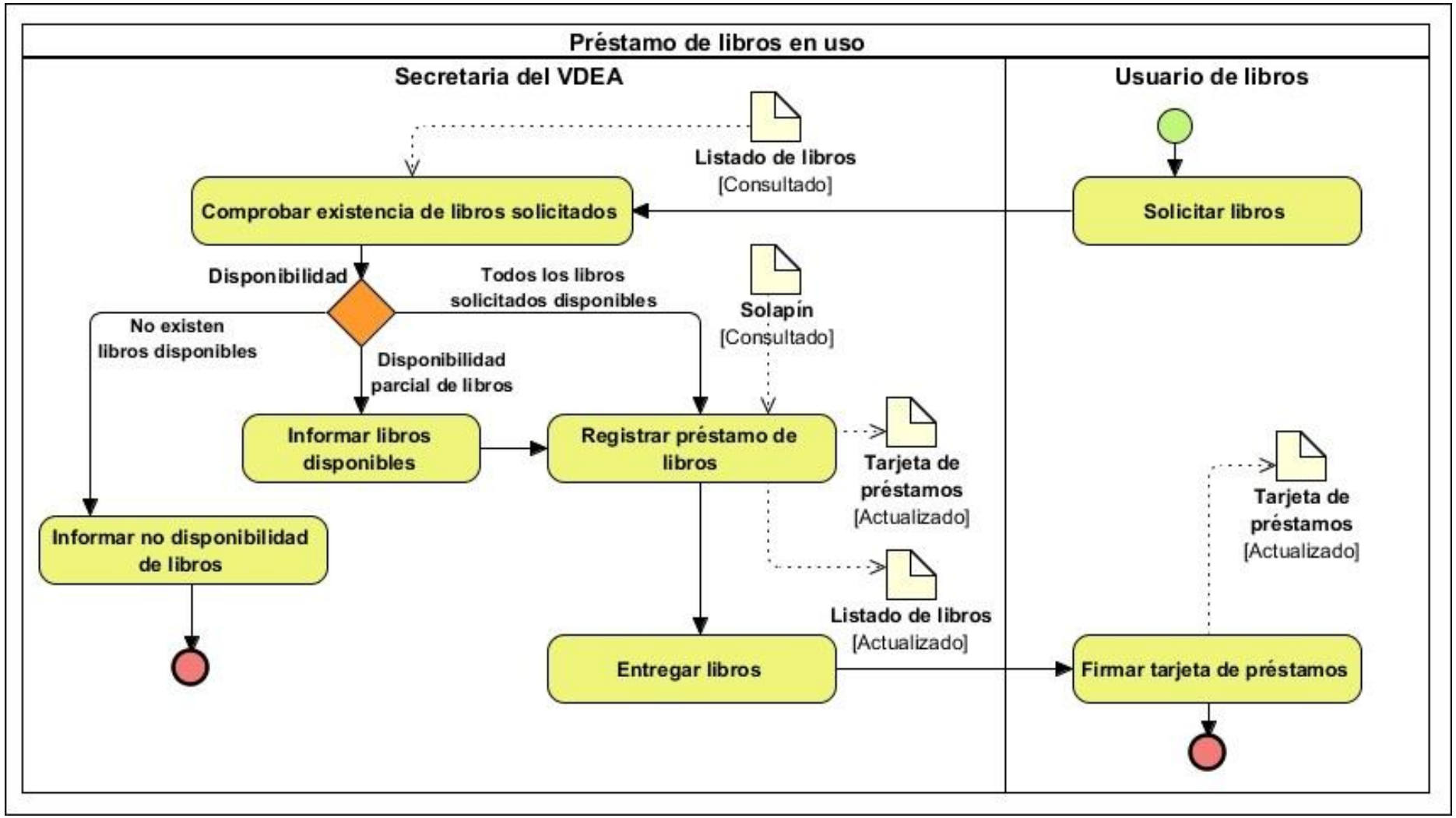


Figura 2: Préstamo de libros en uso

Fuente: (Elaboración propia)

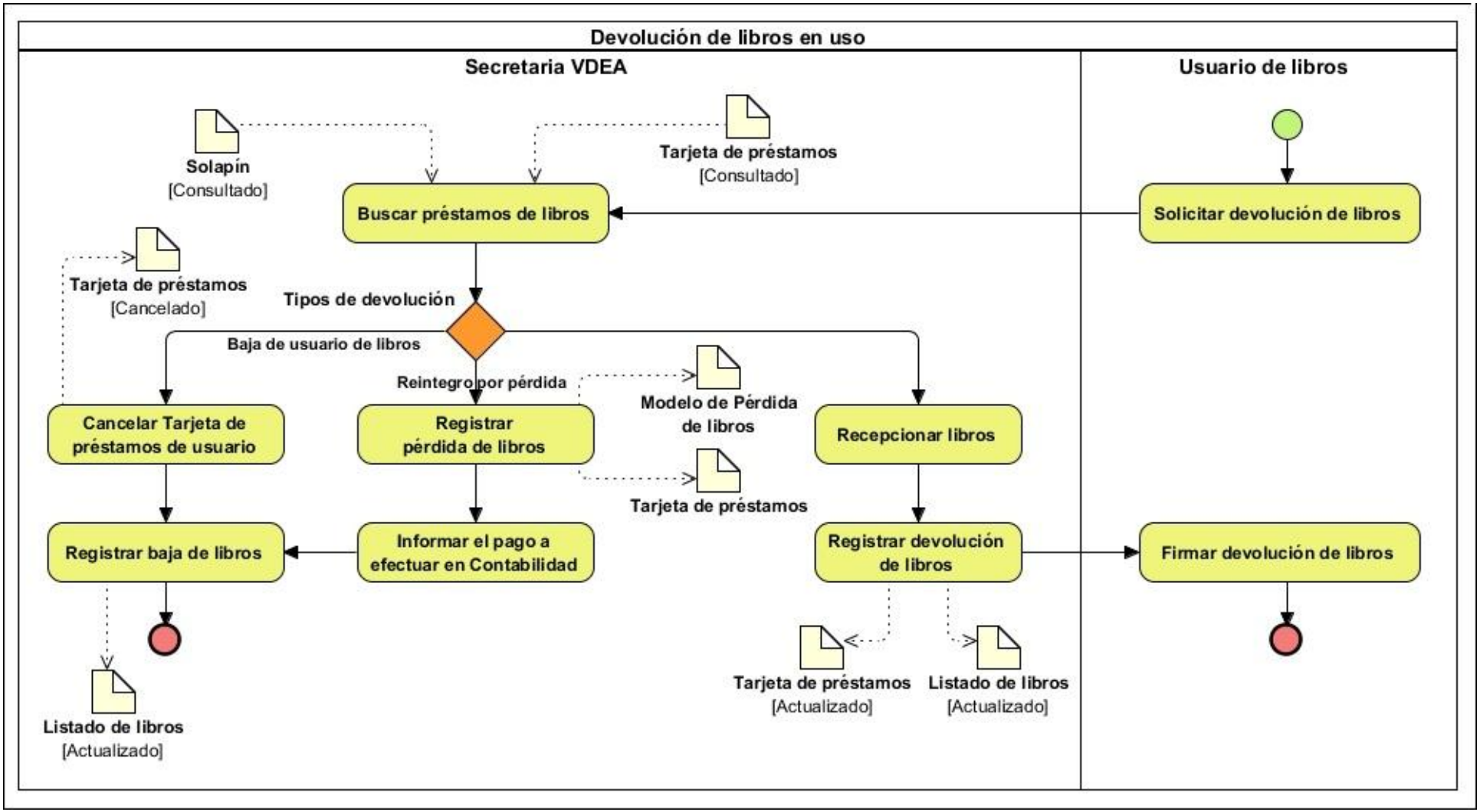


Figura 3: Devolución de libros en uso

Fuente: (Elaboración propia)

2.3 Descripción de la solución

Como solución a la problemática planteada se propone la implementación de 15 nuevas funcionalidades en el módulo Económico del sistema SAEF3 en su segunda versión, así como la modificación de 42 ya existentes. Los requisitos se agruparon usando la técnica de agrupamiento por objetivos.

2.3.1 Requisitos funcionales

Requisitos funcionales a modificar:

- RF1 Gestionar solicitud de recursos materiales.
  - RF1.1 Solicitar recursos materiales.
  - RF1.2 Modificar solicitud de recursos materiales.
  - RF1.3 Consultar solicitud de recursos materiales.
  - RF1.4 Listar solicitudes de recursos materiales.
  - RF1.5 Eliminar solicitud de recursos materiales.
  - RF1.6 Registrar aprobación de recursos materiales solicitados.
- RF2 Gestionar asignación de recursos materiales.
  - RF2.1 Asignar recursos materiales.
  - RF2.2 Modificar asignación de recursos materiales.
  - RF2.3 Consultar asignación de recursos materiales.
  - RF2.4 Listar asignaciones de recursos materiales.
  - RF2.5 Eliminar asignación de recursos materiales.
- RF3 Gestionar vale de salida.
  - RF3.1 Adicionar vale de salida.
  - RF3.2 Modificar vale de salida.
  - RF3.3 Consultar vale de salida.
  - RF3.4 Listar vales de salida.
  - RF3.5 Eliminar vale de salida.
- RF4 Gestionar bonificación de pasaje a estudiantes.
  - RF4.1 Adicionar bonificación de pasaje a estudiantes.
  - RF4.2 Modificar bonificación de pasaje a estudiantes.
  - RF4.3 Consultar bonificación de pasaje a estudiantes.
  - RF4.4 Listar bonificaciones de pasajes a estudiantes.
  - RF4.5 Eliminar bonificación de pasaje a estudiantes.
- RF5 Gestionar partida de presupuesto.
  - RF5.1 Adicionar partida de presupuesto.

- RF5.2 Modificar partida de presupuesto.
- RF5.3 Consultar partida de presupuesto.
- RF5.4 Listar partidas de presupuesto.
- RF5.5 Eliminar partida de presupuesto.

- RF6 Generar reportes
  - RF6.1 Obtener cantidad de recursos materiales solicitados en un rango de fechas determinado.
  - RF6.2 Obtener cantidad de solicitudes canceladas.
  - RF6.3 Listar solicitudes aprobadas parcialmente.
  - RF6.4 Obtener cantidad de un recurso material asignado por rango de fechas y por áreas.
  - RF6.5 Obtener monto de recursos materiales abastecidos por rango de fechas y por almacén.
  - RF6.6 Obtener histórico de bonificaciones de pasaje por estudiante en un rango de fechas.
  - RF6.7 Obtener histórico de bonificaciones de pasaje en un rango de fechas.
  - RF6.8 Obtener bonificaciones de pasaje por provincia y municipio.
  - RF6.9 Obtener partidas sobregiradas por centro de costo.
  - RF6.10 Obtener partidas ejecutadas parcialmente por centro de costo.
  - RF6.11 Obtener histórico de partidas por rango de años.
  - RF6.12 Obtener Solicitud de materiales Vs. Aprobación/Extracción.
  - RF6.13 Obtener tendencias de desviaciones en ejecución de partidas.
  - RF6.14 Obtener control de presupuesto por partidas.

Nuevas funcionalidades a implementar:

- RF7 Gestionar libros de la facultad.
  - RF7.1 Adicionar libro.
  - RF7.2 Modificar libro.
  - RF7.3 Consultar libro.
  - RF7.4 Listar libro.
  - RF7.5 Eliminar libro.

- RF8 Gestionar los préstamos de libros de la facultad
  - RF8.1 Adicionar préstamo de libro.
  - RF8.2 Modificar préstamo de libro.
  - RF8.3 Consultar préstamo de libro.
  - RF8.4 Listar préstamo de libro.
  - RF8.5 Eliminar préstamo de libro.

- RF9 Gestionar devolución de libros de la facultad
  - RF9.1 Adicionar devolución de libro.
  - RF9.2 Modificar devolución de libro.
  - RF9.3 Consultar devolución de libro.
  - RF9.4 Listar devolución de libro.
  - RF9.5 Eliminar devolución de libro.

2.3.2 Requisitos no funcionales

Funcionalidad
Seguridad

- El sistema SAEF3 v2 maneja la seguridad de acceso y administración de usuarios mediante el otorgamiento de privilegios y roles.
- Todo uso de las funcionalidades del módulo requiere la autenticación de los usuarios.
- El módulo concederá acceso a cada usuario autenticado solo a las funciones que le estén permitidas, de acuerdo a la configuración del sistema.

Eficiencia
Utilización de recursos

- El módulo interactuará con impresoras para imprimir los diferentes documentos que genere la aplicación como respuesta a las funcionalidades del sistema.

Instalabilidad

- El sistema podrá ser instalado en el ambiente especificado en los requisitos tecnológicos para servidores.

Hardware
Servidor de aplicaciones web

- Procesador: 3.00 GHZ
- RAM: 3GB
- Disco duro: 160 GB

Servidor de base de datos

- RAM: 2GB
- Disco duro: 160 GB

PC Cliente

- Procesador: 1.40 GHZ
- RAM: 1GB
- Tarjeta de Red: 1

2.4 Prototipos de interfaz de usuario

RF Gestionar libros de la facultad

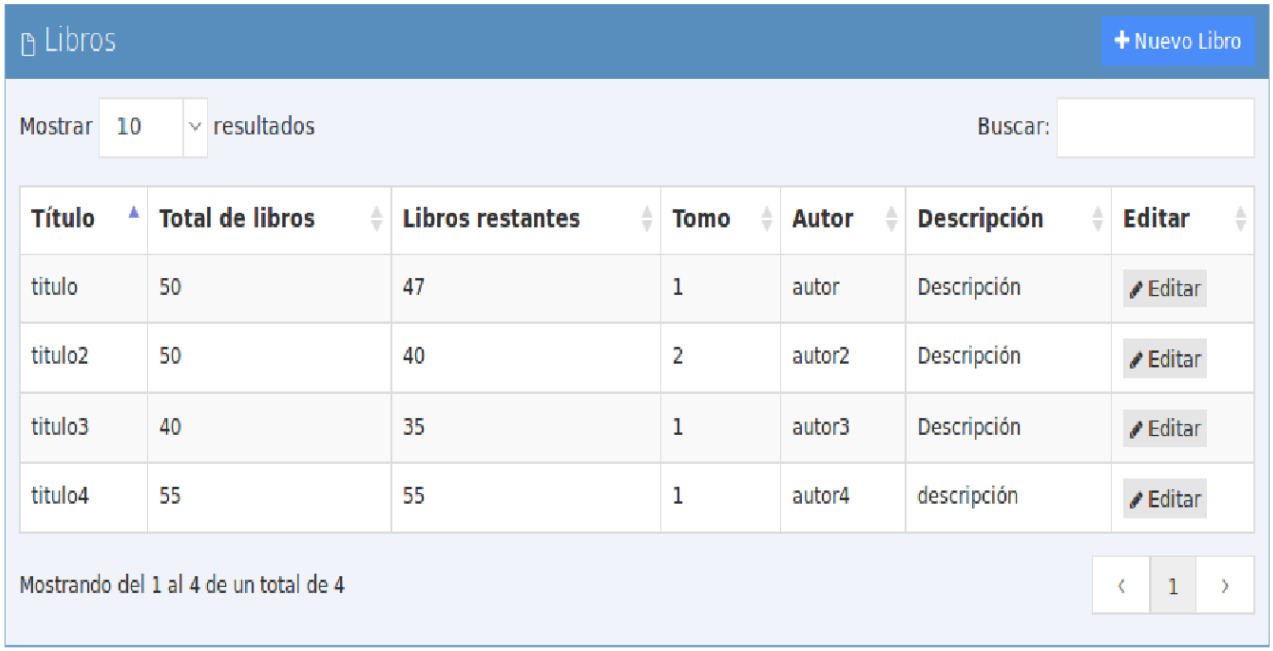


Figura 4: Prototipo de interfaz de usuario “Listar de libros”.

Fuente: (Elaboración propia)

Figura 5: Prototipo de interfaz de usuario” Insertar libros”.

Fuente: (Elaboración propia)

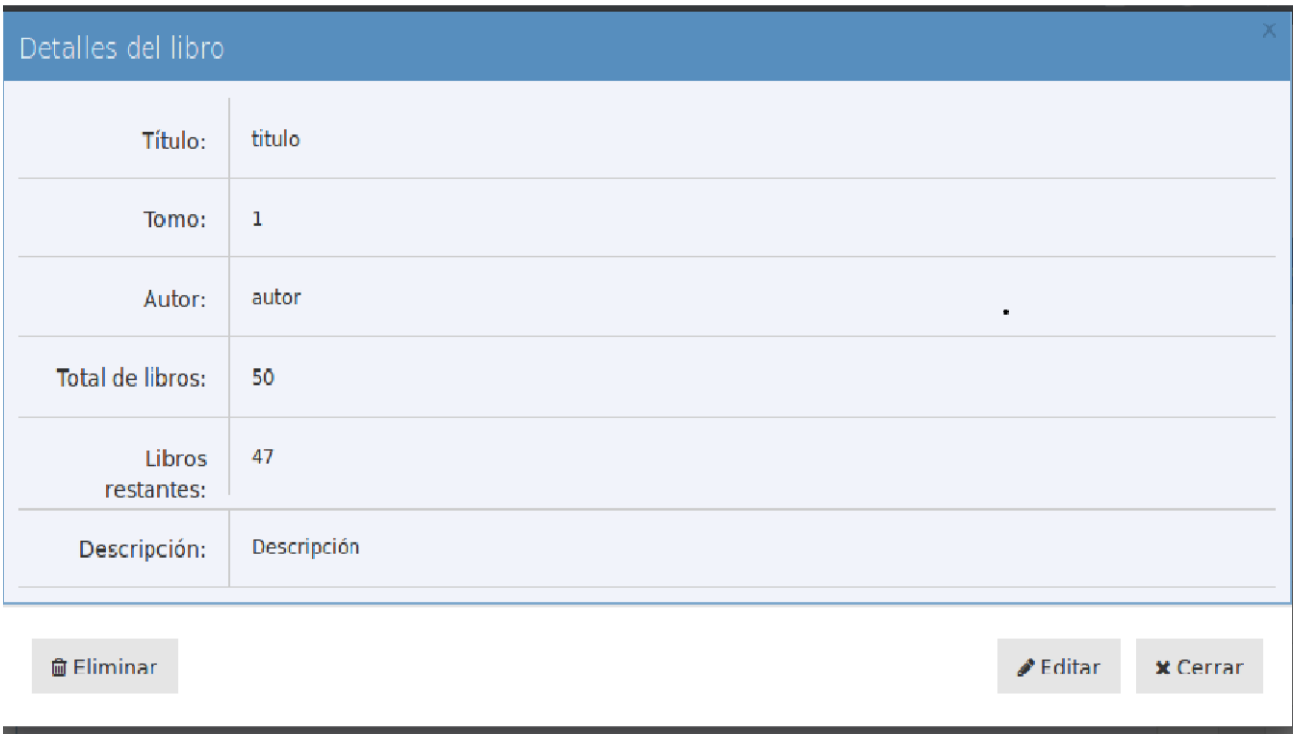


Figura 6: Prototipo de interfaz de usuario" Ver datos de un libro"

Fuente: (Elaboración propia)

2.4.1 Validaciones de requisitos

Para realizar la validación de los requisitos se tuvo en cuenta no solo el criterio de los desarrolladores, sino también el criterio del cliente y de sus homólogos, para un total de 9 personas. Los resultados obtenidos se muestran a continuación.

Atributos de calidad	Resultado	Interpretación
No ambiguo	Q_1=57/57=1	Todos los requisitos tienen una única interpretación.
Comprensible	Q_2=57/57=1	Todos los requisitos son comprendidos.
Correcto	Q_3=57/57=1	Todos los requisitos representan una necesidad de función del sistema a construir.

Tabla 5: Aplicación de las métricas de validación de requisitos

Fuente: (Elaboración propia)

A partir del uso de las métricas de validación de requisitos se evidencia que hubo una total concordancia entre los desarrolladores de los requisitos con respecto al cumplimiento de los atributos

de calidad, obteniéndose que en su totalidad los requisitos son no ambiguos, correctos, y comprensibles.

2.5 Diseño de la solución

En el presente epígrafe se especifican los patrones del diseño aplicados a la solución, el diagrama de clases del diseño y el modelo de datos obtenido.

2.5.1 Patrones de diseño

A continuación, se muestran algunos de los patrones estudiados para la realización del diseño del sistema.

Patrones GRASP

El patrón controlador y el creador se reflejan en las clases controladoras, ya que son las encargadas de atender todas las peticiones de las vistas y pasar los datos a las clases del modelo, así como la responsabilidad de identificar la creación de los nuevos objetos por las clases que contienen la información necesaria para realizarla. En todas las clases de la lógica del negocio se muestra una alta cohesión y un bajo acoplamiento ya que cada una de ellas posee el trabajo de realizar las labores que solo le competen a ella sin asociaciones con la vista, favoreciendo que la dependencia sea baja.
En la siguiente figura se evidencia la utilización de los patrones de asignación de responsabilidades, como es el caso del experto en la clase CE_ReintegroPasaje y EReintegroPasajeRepository. El patrón controlador y creador se refleja en la clase CC_ReintegroPasaje, ya que es la encargada de atender todas las peticiones de las vistas y pasar los datos a las clases del modelo.

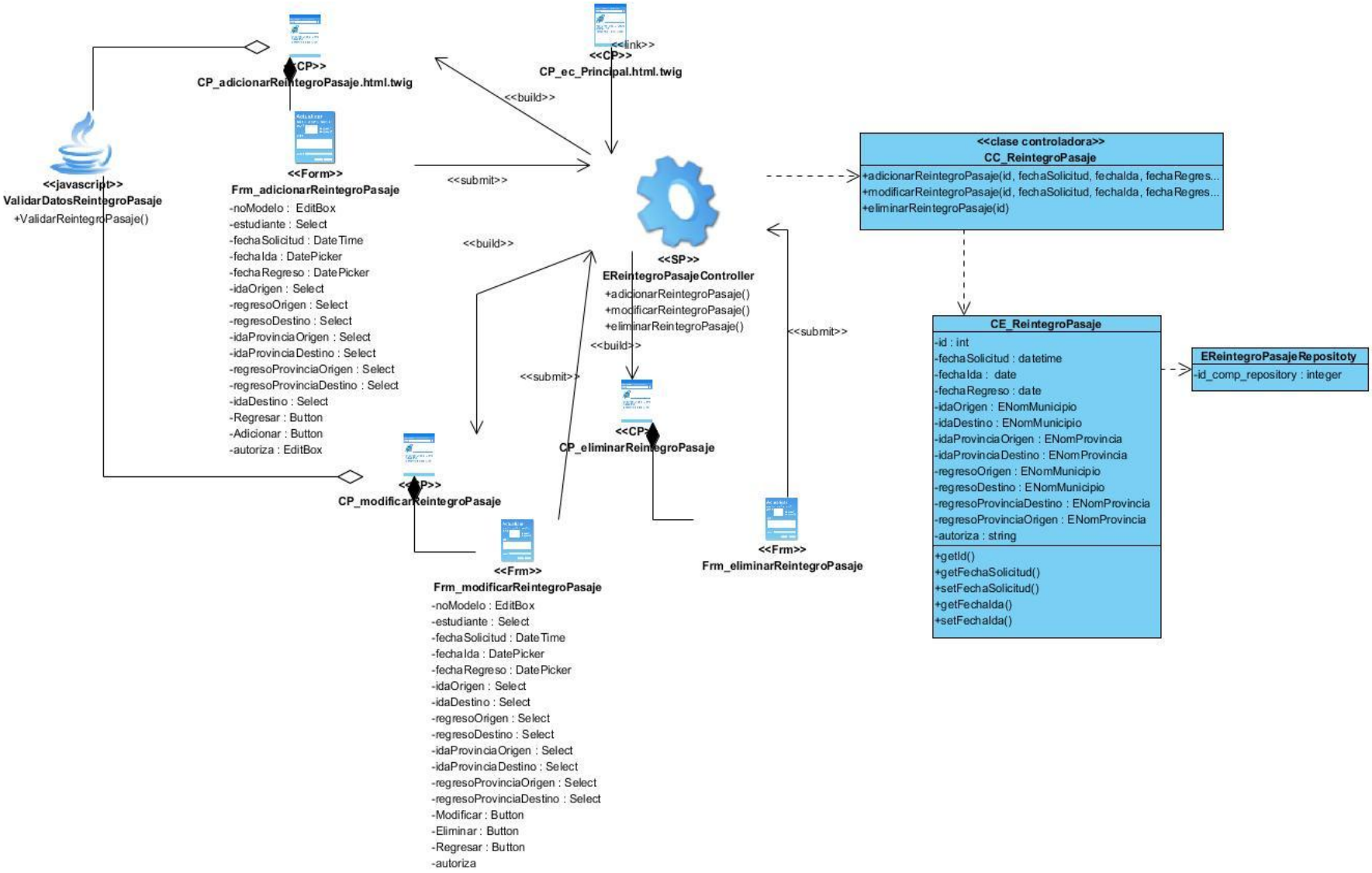


Figura 7: Diagrama de clases del diseño de Bonificación de pasajes a estudiantes

Fuente: (Elaboración propia)

Patrones GoF

Estructurales: Adaptador

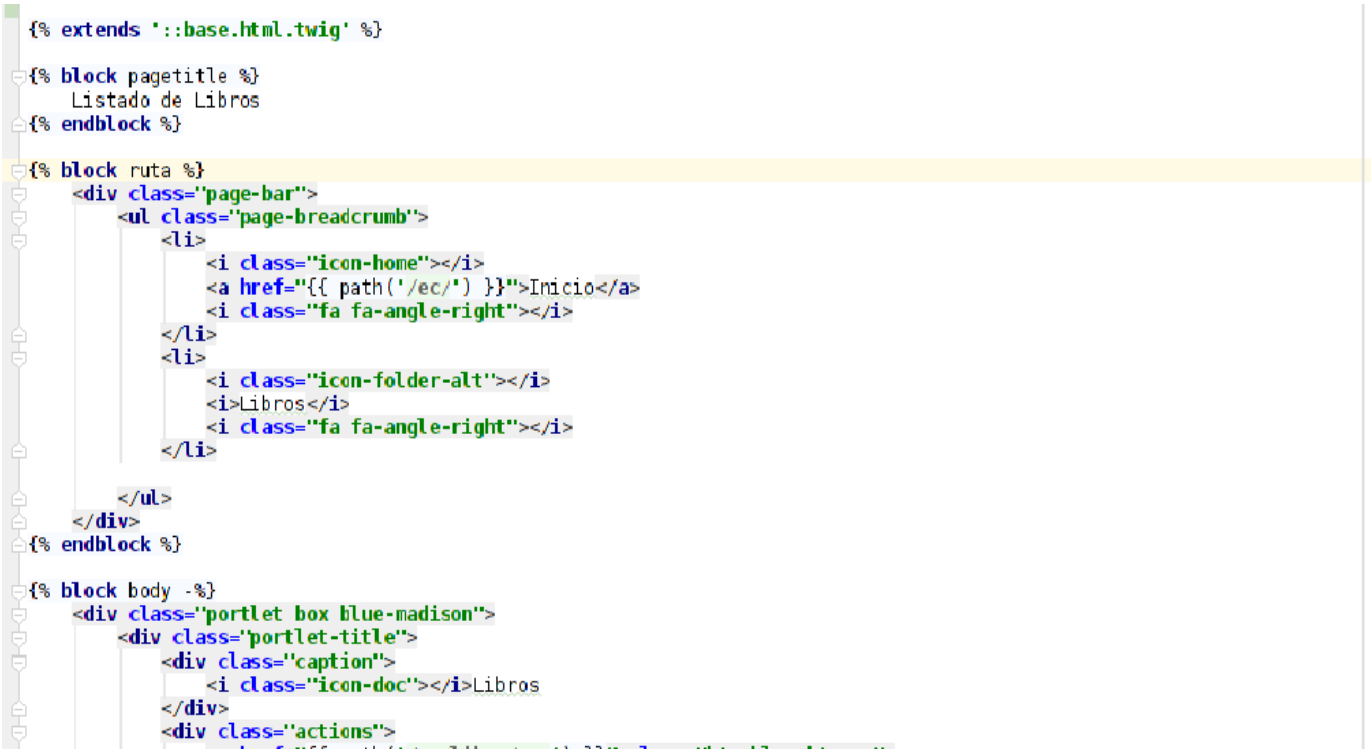
La vista del módulo implementa el uso de plantillas Twig, las cuales contienen variables o expresiones que se reemplazan por valores cuando se evalúa la plantilla, y las etiquetas controlan la lógica de la plantilla.

```
<!DOCTYPE html>
<!--[if IE 8]>
<html class="ie8 no-js"> <![endif]-->
<!--[if IE 9]>
<html class="ie9 no-js"> <![endif]-->
<!--[if !IE]><!-->
<html>
<!--<![endif]-->

<head>
    <meta charset="utf-8"/>
    <title>Facultad 3 | Administración</title>

    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta content="width=device-width, initial-scale=1.0" name="viewport"/>
    <meta http-equiv="Content-type" content="text/html; charset=utf-8">

    <!-- BEGIN GLOBAL MANDATORY STYLES -->
    <link href="{{ asset('assets/global/plugins/font-awesome/css/font-awesome.min.css') }}" rel="stylesheet"
          type="text/css"/>
    <link href="{{ asset('assets/global/plugins/simple-line-icons/simple-line-icons.min.css') }}" rel="stylesheet"
          type="text/css"/>
    <link href="{{ asset('assets/global/plugins/bootstrap/css/bootstrap.min.css') }}" rel="stylesheet" type="text/css"/>
    <link href="{{ asset('assets/global/plugins/uniform/css/uniform.default.css') }}" rel="stylesheet" type="text/css"/>


    <!-- BEGIN PAGE LEVEL STYLES -->
    <link href="{{ asset('assets/admin/pages/css/blog.css') }}" rel="stylesheet" type="text/css"/>
    <link href="{{ asset('assets/admin/pages/css/news.css') }}" rel="stylesheet" type="text/css"/>
    <link href="{{ asset('assets/global/plugins/select2/select2.css') }}" rel="stylesheet" type="text/css"/>
    <link href="{{ asset('assets/global/plugins/datatables/extensions/Scroller/css/dataTables.scroller.min.css') }}"
          rel="stylesheet" type="text/css"/>
    <link href="{{ asset('assets/global/plugins/datatables/extensions/ColReorder/css/dataTables.colReorder.min.css') }}"
          rel="stylesheet" type="text/css"/>
    <link href="{{ asset('assets/global/plugins/datatables/plugins/bootstrap/dataTables.bootstrap.css') }}"
```

Figura 8: Clase base.html.twig

Fuente: (Elaboración propia)

La herencia de plantillas permite crear un “esqueleto” de plantilla base (la clase base.html.twig cumple la función de esqueleto) que contiene todos los elementos comunes del módulo y define bloques que las plantillas descendientes pueden sustituir. Se sustituyen los bloques definidos dentro de las etiquetas **{% block**… **%}** … **{% endblock %}**.



```
{% extends '::base.html.twig' %}

{% block pagetitle %}
    Listado de Libros
{% endblock %}

{% block ruta %}
    <div class="page-bar">
        <ul class="page-breadcrumb">
            <li>
                <i class="icon-home"></i>
                <a href="{{ path('/ec/') }}">Inicio</a>
                <i class="fa fa-angle-right"></i>
            </li>
            <li>
                <i class="icon-folder-alt"></i>
                <i>Libros</i>
                <i class="fa fa-angle-right"></i>
            </li>

        </ul>
    </div>
{% endblock %}

{% block body -%}
    <div class="portlet box blue-madison">
        <div class="portlet-title">
            <div class="caption">
                <i class="icon-doc"></i>Libros
            </div>
            <div class="actions">
```

Figura 9: Clase ListarLibros.html.twig

Fuente: (Elaboración propia)

La clase ListarLibros.html.twig (figura 7), extiende de la clase base.html.twig, redefiniendo los bloques pagetitle, ruta, body y scripts.

Comportamiento: Iterador

El uso de este patrón se evidencia en la clase controladora **ESalidaController**, deonde se hace uso de la variable **array** para acceder a los elementos del objeto **json,** facilitando las iteraciones para la obtención de cada elemento contenido en el objeto.

```php
public function createAction(Request $request)
{
    $entity = new ESalida();
    $form = $this->createCreateForm($entity);
    $form->handleRequest($request);

    if ($form->isValid()) {
        $em = $this->getDoctrine()->getManager();
        $json = $request->request->get('input_materiales');
        $array = json_decode($json);

        $em->persist($entity);
        $em->flush();

        foreach ($array as $material){
            $mate = new ESalidaMaterial();
            $el_material = $em->getRepository('Fac3AppBundle:EMaterial')->find($material->material);
            $mate->setMaterial($el_material);
            $mate->setCantidad($material->cantidad);
            $mate->setSalida($entity);
            $em->persist($mate);
            $em->flush();
        }

        return $this->redirect($this->generateUrl('/esalida/'));
    }
    $em = $this->getDoctrine()->getManager();
    $materiales = $em->getRepository('Fac3AppBundle:EMaterial')->findAll();
```

Figura 10: Clase ESalidaController.php

Fuente: (Elaboración propia)

Creacional: Inicialización vaga

El patrón de inicialización vaga se utiliza para delegar la creación de los objetos material y centro en el sistema, así como el cálculo de sus valores hasta que sean realmente necesitados.

```php
/**
 * Constructor
 */
public function __construct()
{
    $this->organizaciones = new \Doctrine\Common\Collections\ArrayCollection();
    $this->nucleofamiliar = new \Doctrine\Common\Collections\ArrayCollection();
    $this->prestamoAnt = new \Doctrine\Common\Collections\ArrayCollection();
    $this->estudianteAP = new \Doctrine\Common\Collections\ArrayCollection();
    $this->estado = new \Doctrine\Common\Collections\ArrayCollection();
}

/**
```

Figura 11: Ejemplo de aplicación del patrón de inicialización vaga

Fuente: (Elaboración propia)

2.5.2 Modelo de datos

El modelado y diseño de la base de datos tiene como objetivo generar un conjunto de entidades y las relaciones entre ellas, que permitan el almacenamiento de la información con un mínimo de redundancia, manteniendo su integridad y facilitando la recuperación para su consulta. Para ello se utilizó el Modelo Entidad-Relación (MER) compuesto por 38 tablas, de ellas 13 nomencladores y el resto destinadas al almacenamiento de la información correspondiente al dominio del problema. A continuación, se muestra dicho diagrama.

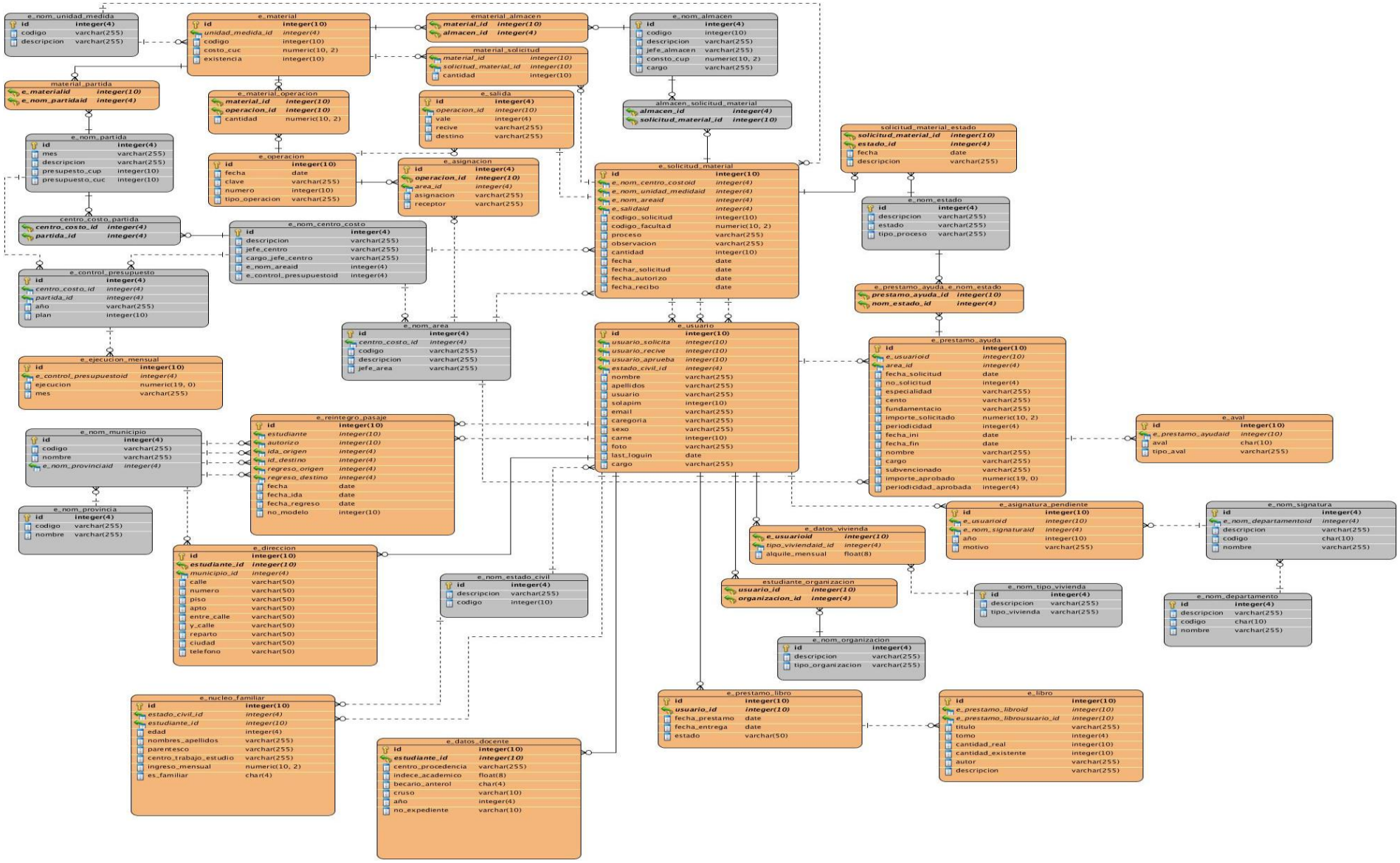


Figura 12: Diagrama MER. Modelo de datos

Fuente: (Elaboración propia)

2.5.3 Verificación del diseño

Para la validación del diseño de la solución se emplearon las métricas de Relaciones ente clases (RC), Tamaño de las clases (TC) y Carencia de Cohesión de los Métodos (CCM) aplicadas a los indicadores: responsabilidad, complejidad de implementación y reutilización, clasificándolos en altos, medios o bajos.

Métrica	Resultados del análisis
RC	La aplicación de la métrica arrojo resultados favorables en cuanto a acoplamiento y complejidad de implementación con valores por encima del 70%, en el caso de la reutilización se alcanzó un 70% y la cantidad de prueba se obtuvo un 75% lo que conlleva a la conclusión de que la calidad del diseño escogido es aceptable.
CCM	Luego de revisar detenidamente cada diagrama de diseño se pudo constatar que, en la mayoría de las clases, a lo sumo existe un método que accede a un mismo atributo. Esto arroja un valor CCM =2. Exceptuando la clase *EPrestamoAyuda* donde existen 7, para un CCM = 7. Se llega a la conclusión de que las clases del diseño poseen un alto grado de cohesión debido a que, de un total de 30, solo una posee valor medio y el resto posee valor bastante pequeño.
TC	Luego de la aplicación de la métrica los resultados obtenidos fueron satisfactorios pues más del 60% de las clases presentó baja responsabilidad y complejidad de implementación y solo el 10% presentó bajo grado de reutilización de un total de 30 clases.

Tabla 6: Resultado de la aplicación de las métricas del diseño

Fuente: (Elaboración propia)

2.6 Conclusiones parciales

- En el presente capítulo se describieron las características del sistema, se identificaron los requisitos funcionales y no funcionales, realizándose la especificación de los funcionales a través de historias de usuario. Mediante la aplicación de las métricas de validación de requisitos se evidenció que los mismos eran en su totalidad: no ambiguos, comprensibles y correctos.
- Se realizó el modelado del diseño, evidenciándose la aplicación de los patrones del diseño. La aplicación de las métricas del diseño TC, RC y CCM confirmó la calidad del diseño realizado para facilitar la implementación del módulo, comprobándose que el mismo posee baja responsabilidad y complejidad, lo cual favorece la reutilización, la alta cohesión y el bajo acoplamiento.

CAPÍTULO 3 IMPLEMENTACIÓN Y ANÁLISIS DE LOS RESULTADOS

3.1 Introducción

En el presente capítulo se muestra el diagrama de componentes que estructura la implementación, así como la descripción de los estándares de codificación empleados durante la implementación del módulo además del tratamiento de errores. Se realiza además el análisis de los resultados obtenidos en la validación de la solución y las variables de la investigación.

3.2 Aspectos relevantes de la implementación

A continuación, se muestra el diagrama de componentes desarrollado, los estándares de codificación empleados y el tratamiento de errores.

3.2.1 Diagrama de componentes

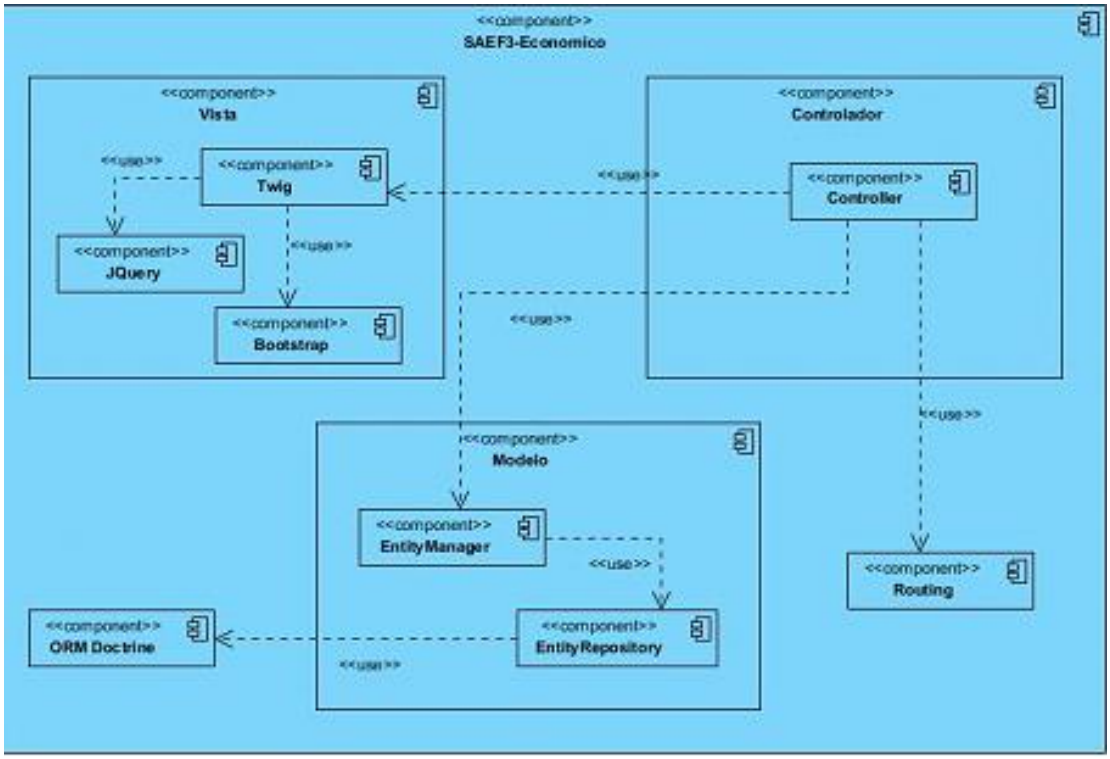


Figura 13: Diagrama de componentes del módulo Económico versión 2.0 del SAEF3

Fuente: (Elaboración propia)

3.2.2 Estándares de codificación

Symfony 2.6, el framework empleado para el desarrollo del módulo Económico, presenta estándares de codificación definidos en los documentos *PSR-0*, *PSR-1*, *PSR-2* and *PSR-4* de la siguiente manera (Symfony 2015):

Estructura

- Añade un solo espacio después de cada delimitador coma;
- Añade un solo espacio alrededor de los operadores (==, &&, ...);
- Añade una coma después de cada elemento del arreglo en un arreglo multilínea, incluso después del último;
- Añade una línea en blanco antes de las declaraciones return, a menos que el valor devuelto solo sea dentro de un grupo de declaraciones (tal como una declaración if);
- Usa llaves para indicar la estructura del cuerpo de control, independientemente del número de declaraciones que contenga;
- Define una clase por archivo — esto no se aplica a las clases ayudante privadas, de las cuales no se tiene la intención de crear una instancia desde el exterior y por lo tanto no les preocupa la norma PSR-0;
- Declara las propiedades de clase antes que los métodos;
- Declarar primero los métodos públicos, después los protegidos y por último los privados.

Convenciones de nomenclatura

- Utiliza mayúsculas intercaladas —sin guiones bajos— en nombres de variable, función, método o argumentos;
- Usa guiones bajos para nombres de opción y nombres de parámetro;
- Utiliza espacios de nombres para todas las clases;
- Prefija las clases abstractas con Abstract. Por favor, ten en cuenta que algunas de las primeras clases de Symfony2 no siguen esta convención y no se han rebautizado por razones de compatibilidad hacia atrás. No obstante, todas las nuevas clases abstractas tienen que seguir esta convención de nomenclatura;
- Sufija las interfaces con Interface;Sufijo las características con Trait;
- Sufija las excepciones con Exception;
- Utiliza caracteres alfanuméricos y guiones bajos para los nombres de archivo;

Documentación

- Añade bloques PHPDoc a todas las clases, métodos y funciones; omite la etiqueta @return si el método no devuelve nada; las anotaciones @package y @subpackage no se utilizan.

Licencia

- Symfony se distribuye bajo la licencia MIT, y el bloque de la licencia tiene que estar presente en la parte superior de todos los archivos PHP, antes del espacio de nombres.

3.2.3 Tratamiento de errores

Para asegurar el tratamiento de los errores producidos tanto por el sistema como por los usuarios se utilizan las validaciones provenientes de la librería de jQuery, estas validaciones son utilizadas para la notificación de los errores ocurridos en las vistas, de esta forma se garantiza la correcta entrada de datos en los campos de los formularios y no lleguen al servidor ningún dato inconsistente o incorrecto. Por otra parte el marco de trabajo Symfony contiene un conjunto de procedimientos que garantizan validaciones básicas a nivel de usuario.

3.3 Validación de la solución

A continuación se muestran los resultados obtenidos de la aplicación de las pruebas aplicadas al módulo.

4.3.1 Pruebas de caja negra

Las pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. Estas pruebas se basan en las especificaciones del módulo a ser probadas Su comportamiento se evalúa mediante sus entradas y las salidas obtenidas a partir de ellas. Sin embargo, el estudio de todas las posibles entradas y salidas sería impracticable, es por ello que se selecciona un conjunto sobre las que se realizan las pruebas (Juristo, y otros, 2004). Para la aplicación de las pruebas de caja negra se seleccionó la técnica de Particiones de Equivalencia.

Durante la aplicación de las pruebas en una primera iteración se detectaron 22 no conformidades las cuales fueron corregidas, durante una segunda iteración fueron detectadas 9 no conformidades, las cuales fueron corregidas dando paso a una tercera iteración en la cual no se identificaron ninguna no conformidad.

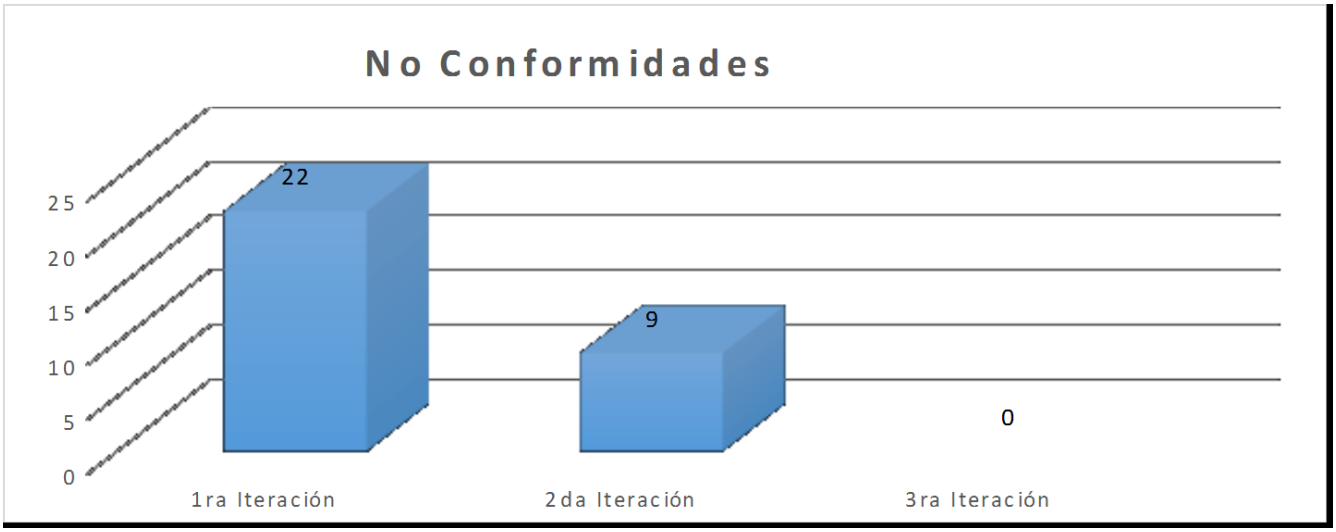


Figura 14: Resultados por iteraciones de pruebas

Fuente: (Elaboración propia)

A continuación se muestra un ejemplo de caso de prueba efectuado al requisito Adicionar Libro.

Escenario	Descripción	Título	Tomo	Autor	Total de libros	Libros restantes	Descripción	Respuesta del sistema
EC 1.1 Adicionar Libro	Se inserta un nuevo libro	NA	NA	NA	NA	NA	NA	El sistema muestra un mensaje indicando que se debe llenar cada uno de los campos
		vacío	vacío	vacío	vacío	vacío	vacío	
		V	V	NA	V	V	V	El sistema muestra un mensaje indicando que se debe llenar cada uno de los campos faltantes
		Matemática Discreta	1	vacío	50	50	Matemática discreta para segundo año	
		V	V	V	V	V	V	El sistema inserta el nuevo libro y muestra un mensaje indicando que se insertaron correctamente
		Matemática Discreta	1	Colectivo de Autores	50	50	Matemática discreta para segundo año	
		V	NA	V	V	V	V	El sistema muestra un mensaje indicando que en ese campo solo se pueden insertar números
		Matemática Discreta	I	Colectivo de Autores	50	50	Matemática discreta para segundo año	

Tabla 7: Caso de prueba para requisito Adicionar libro

Fuente: (Elaboración propia)

3.3.2 Tipos de pruebas

Se aplicaron tres tipos de pruebas:

-Funcionalidad: esta prueba se aplicó mediante las pruebas de caja negra para verificar el correcto funcionamiento del módulo.

-Portabilidad: el módulo fue probado en diferentes plataformas (Nova 4, Windows 7 y 8.1), mostrando un correcto desempeño en cada una de ellas.

-Usabilidad: el cliente mostró su satisfacción con respecto al producto mediante la aplicación de varias pruebas, la cual se refleja además, en los resultados de la aplicación de la técnica Iadov.

Para valorar la satisfacción del cliente con respecto al control de la información se aplicó la técnica de Iadov. Como clientes fungieron los VDEA de cada una de las facultades de la universidad y dos trabajadores del VDEA de la Facultad 3, para un total de 9 personas durante la valoración. Como resultado se obtuvo:

¿Considera factible utilizar la versión 2.0 del módulo Económico del SAEF3 propuesto, para ejecutar los procesos asociados al área económica del VDEA?	¿Considera que el uso de la primera versión del módulo Económico satisface las necesidades de control de la información a pesar de las insuficiencias detectadas en él?								
	No			No sé			Sí		
	¿Considera que la versión 2.0 del módulo Económico refleja mejoras sobre el control de la información asociada al VDEA con respecto a la versión anterior?								
	Sí	No sé	No	Sí	No sé	No	Sí	No sé	No
Me gusta mucho	1	2	6	2	2	6	6	6	6
No me gusta mucho	2	2	3	2	3	3	6	3	6
Me da lo mismo	3	3	3	3	3	3	3	3	3
Me disgusta más de lo que me gusta	6	3	6	3	4	4	3	4	4
No me gusta nada	6	6	6	6	4	4	6	4	6
No sé qué decir	2	3	6	3	3	3	6	3	4

Tabla 8: Cuadro lógico de Iadov para evaluar la solución

Fuente: (Elaboración propia)

IMPLEMENTACIÓN Y ANÁLISIS DE LOS RESULTADOS

Para obtener el índice de satisfacción grupal (ISG) se trabaja con los diferentes niveles de satisfacción que se expresan en la escala numérica que oscila entre +1 y - 1 de la siguiente forma:

Índice de satisfacción	Escala
Máxima satisfacción	+1
Más satisfecho que insatisfecho	0,5
No definido y contradictorio	0
Más insatisfecho que satisfecho	-0,5
Máxima insatisfacción	-1

Tabla 9: Índice de satisfacción de Iadov

Fuente: (Kuzmina 1970)

De manera que el ISG = 0.94

Los resultados de la satisfacción individual según las categorías empleadas fueron los siguientes:

Nivel de satisfacción	Cantidad	%
Máxima satisfacción	8	88.8
Más satisfecho que insatisfecho	1	11.2
No definida	0	0.0

Tabla 10: Resultado de aplicación de la técnica Iadov

Fuente: (Elaboración propia)

Al procesar las encuestas en el cuadro lógico de Iadov se obtuvo como resultado un grado de satisfacción grupal de 0.94; lo cual se traduce en una clara satisfacción con el uso de la versión 2.0 del módulo Económico para el SAEF3.

Con respecto al criterio de que si la versión 2.0 del módulo Económico contribuye a la mejora del control de la información en el VDEA tuvo una concordancia de un 100%. Igual concordancia se evidenció ante la manifestación de los participantes en hacer uso de la versión 2.0 del módulo Económico del SAEF3 para el desarrollo de los procesos asociados al VDEA. Mientras que un 11.2% no sabía si sería oportuno seguir utilizando la versión anterior del módulo Económico.

Las preguntas abiertas formuladas fueron las siguientes:

- ¿Qué valoraciones le sugiere a la versión 2.0 del módulo Económico del SAEF3 respecto al control y disponibilidad de la información asociada al VDEA?

- ¿Qué elemento(s) le gustaría adicionaría a la solución propuesta?

Como valoraciones positivas obtenidas en respuestas a las preguntas abiertas, se obtuvieron las siguientes:

- La versión 2.0 del módulo Económico permite un control más eficiente sobre los procesos con respecto a la primera versión del mismo.
- La versión 2.0 facilita la obtención de reportes para una gestión proactiva desde el VDEA.
- La versión 2.0 facilita la obtención de nuevos reportes, los cuales no eran posibles en la primera versión del módulo Económico.
- Favorece la privacidad hacia los estudiantes en un tema delicado como lo es el proceso de solicitud de préstamos o ayudas económicas.
- Establece un mejor control sobre el proceso de solicitud de los recursos materiales así como sobre la asignación de los mismos a las distintas áreas de la facultad

Con la aplicación de la técnica de Iadov se validó la fortaleza de la propuesta del desarrollo de la versión 2.0 del módulo Económico a través del grado de satisfacción del cliente, el cual evidencia mediante los criterios emitidos por el cliente con respecto a la nueva versión del módulo Económico.

La aplicación de la técnica de Iadov aportó información significativa respecto al grado de satisfacción del cliente. Los resultados obtenidos y los criterios emitidos validan la fortaleza de la propuesta, reflejándose una valoración muy positiva del cliente con la solución.

4.4 Validación de las variables de la investigación

Para validar la variable de la investigación (tabla 10), se analizaron las dimensiones de las mismas. Estas dimensiones fueron seleccionadas a seleccionando las dimensiones a partir del concepto de control de la información.

Variables	Dimensiones	Módulo Económico del SAEF3	Módulo Económico versión 2.0 del SAEF3
Control de la información	Fiscalización	Actualmente los jefes de áreas no pueden hacer juicio cierto de si los recursos que son asignados a sus respectivas áreas son los que realmente se encuentran en el sistema.	Una vez que se le es asignado uno o varios recursos a un área, el jefe de dicha área tiene la posibilidad de verificar cuales y cuantos recursos le fueron asignados.
	Comprobación	El control de los préstamos de libros a estudiantes y trabajadores de la facultad se hace de forma manual, siendo muy engorroso el manejo de la información, lo cual provoca pérdida de información por parte del personal que la maneja o por deterioro, pues que toda la información se encuentra en formato duro.	Los préstamos de los libros son registrados en el sistema, lo que posibilita una mayor organización de la información, se tiene un mayor control sobre los libros existentes en la facultad y sus disponibilidad.
		El control del presupuesto se realiza anualmente, lo que imposibilita la verificación de las partidas inejecutadas y sobregiradas y de forma que no puedan ser corregidas hasta que sea realizada la ejecución del presupuesto.	El control del presupuesto se lleva a cabo de manera semiautomática, pues a medida que se registran los vales de salida de un almacén el módulo actualizara la ejecución de las partidas relacionadas con cada uno de los materiales contenidos en el vale de salida, permitiendo de esta forma la comprobación exacta de cada ejecución, también se brinda la posibilidad de corrección de los valores de las ejecuciones de ser esto requerido.

Tabla 11: Validación de la variable de la investigación

Fuente: (Elaboración propia)

3.5 Conclusiones parciales

- El uso de los estándares de codificación en la implementación permitió ganar en legibilidad, claridad y mayor entendimiento del código. El tratamiento de errores evitó la posibilidad de que datos incorrectos fuesen introducidos en el sistema a través de los diferentes campos de los formularios, lo cual fue posible mediante el uso de determinados métodos de validación.
- La aplicación de pruebas de caja negra validó el correcto funcionamiento del módulo, las cuales arrojaron como resultado que luego de tres iteraciones la solución estuviese libre de no conformidades.
- La aplicación de la técnica Iadov evidenció la satisfacción del cliente respecto al control de la información asociada a la solución, donde se constató un índice de satisfacción grupal de un 0.94 lo que refleja una clara satisfacción por parte del cliente con la versión 2.0 del módulo Económico del SAEF3.

CONCLUSIONES GENERALES

- El estudio de sistemas homólogos corroboró que los mismos no brindan una solución completa para el problema existente en el VDEA de la Facultad 3, lo que evidenció la necesidad de desarrollar la versión 2.0 del módulo Económico para SAEF3. Para ello se aplicó la metodología AUP en su variación UCI y se empleó el framework Symfony en su versión 2.5.6, utilizando Bootstrap 3 y JQuery 1.9.
- La realización del modelado del negocio facilitó la comprensión de las nuevas funcionalidades del módulo desarrollado, para lo cual se identificaron requisitos funcionales y no funcionales, aplicando métricas de validación a los funcionales lo cual evidenció que los mismos eran correctos, comprensibles y no ambiguos.
- El uso de las métricas TC, RC y CCM arrojó como resultado que el diseño evidenciaba una baja responsabilidad y complejidad de implementación.
- La utilización de los estándares de codificación propuestos por los desarrolladores de Symfony permitió una implementación organizada y de fácil comprensión para posibles mantenimientos.
- Con la aplicación de pruebas de caja negra, se verificó el correcto funcionamiento de la solución, obteniéndose en un principio varias no conformidades las cuales fueron corregidas posteriormente. La aplicación de la técnica de Iadov, permitieron medir el nivel de satisfacción del cliente respecto a la versión 2.0 del módulo Económico, obteniéndose un elevado grado por parte de los mismos.
- La validación de las variables de la investigación corroboró mejoras respecto al control de la información asociada a los procesos del área económica del VDEA de la Facultad 3.

RECOMENDACIONES

- Hacer extensivo el uso del módulo Económico del SAEF3 al resto de los Vicedecanatos de Economía y Administración de la UCI, para visualizar los resultados que arroja y futuras mejoras.

BibliografíaREFERENCIAS BIBLIOGRÁFICAS

BANCO DE LA REPÚBLICA, S. C. Definición y funciones de la economía. In. Colombia, 2015, vol. 2016.

BUSCHMANN, F., K. HENNEY AND D. C. SCHMIDT *Pattern-Oriented Software Architecture: A Pattern Language For Distributed Computing*. Edtion ed., 2007. ISBN 8126513004.

CHIDAMBER, S. R. AND C. F. KEMERER A Metrics Suite for Object Oriented Design. Software Engineering, IEEE Transactions, 1994.

CÓRDOBA, U. D. Sistema Integral de Gestión Económica. In. Córdoba, 2014, vol. 2016.

DAVIS, A., S. OVERMYER, K. JORDAN, J. CARUSO, et al. Identifying and measuring quality in a software requirements specification. In *Software Metrics Symposium, 1993. Proceedings., First International*. 1993, p. 141-152.

DAVIS, A., Y OTROS. Identifying and Measuring Quality in a Software Requirements Specification. In., 2005.

DÍAZ, G. Y. AND R. Y. FERNÁNDEZ Patrón Modelo-Vista-Controlador. Telemática. Revista Digital de las Tecnologías de la Información y las Comunicaciones, 2012, 11(1).

DINTEL Sistema para la Gestión. DINTEL, 2012.

FIGUEROA, R. G., C. J. SOLÍS AND A. A. CABRERA. Metodologías tradicionales vs. Metodologías ágiles. 2008.

GAMMA, E., R. HELM, R. JOHNSON AND J. VLISSIDES. Patrones de diseño. Elementos de Software orientado a objetos reutilizables. In.: Addison-Wesley, 2002.

GARCIA, A. M. R., G. H. DARIAS AND B. Y. G. PÉREZ Validación de requisitos con expertos funcionales. Informática 2013, 2013.

HETZEL, B. *The Complete Guide to Software Testing*. Edtion ed., 1993. ISBN 978-0471565673.

IEEE. 610-1990 - IEEE Standard Computer Dictionary: Compilation of IEEE Standard Computer Glossaries. In. IEEE Xplore Digital Library: IEEE Computer Society, 1990.

JACOBSON, I., G. BOOCH AND J. RUMBAUGH *El Proceso Unificado de Desarrollo de Software*. edited by ADDISON-WESLEY. Edtion ed., 2000.

JETBRAINS. PhpStorm Lightning-smart PHP IDE. In., 2016.

JQUERY-FUNDATION. JQuery, write less do more. In., 2016.

KUZMINA, N. V. *Metódicas Investigativas de la actividad pedagógica*. Edtion ed. Leningrado, 1970.

LARMAN, C. *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado*. Edtion ed. Madrid: Person Educación, 2003.

LORENZ, M. AND J. KIDD *Object-Oriented Software Metrics*. Edtion ed., 1994. ISBN 9780131792920.

OCU. Oficina de Cooperación Universitaria. In *Universitas XXI - Económico*. 2016, vol. 2016.

OROZCO, E. S., J. ALCANTAR, J. CARRO SUÁREZ, O. FERNANDO CASTELLANOS, et al. *Inteligencia Empresarial. Qué y cómo*. Edtion ed. La Habana: IDICT, 2009. ISBN 978-959-234-070-1.

PAE. Administración Presupuestaria. In M.D.H.Y.A. PÚBLICAS. *Sistema Informático de Apoyo a la Gestión Económica de los Centros Gestores Públicos versión 2*. España, 2011, vol. 2016.

PGADMIN. pgAdmin PostgreSQL Tools. In., 2016, vol. 2016.

POSTGRESQL, E. D. D. D. Manual del usuario de PostgreSQL. T. LOCKHART, 1996.

POTENCIER, F. Symfony. In.: SesionLabs, 2016, vol. 2016.

PRESSMAN, R. *Software Engineering. A Practtitioner´s Approach*. Edtion ed. New York: McGraw-Hill, 2010. ISBN 978-0-07-337 597 -7.

RAE. Diccionario de la lengua española. Edición del Tricentenario. In.: DRAE, 2014, vol. 2016.

Bibliografía

RAMÍREZ, F. A. CLASIFICACIONES DE TIPOS DE REQUISITOS PARA LA MEJORA DEL PROCESO DE DESARROLLO DEL SOFTWARE. 2012.
REFSNES-DATA. w3schools.com. In., 2016, vol. 2016.
RUMBAUGH, J., I. JACOBSON AND G. BOOCH *El Lenguaje Unificado de Modelado*. Edtion ed.: Addison Wesley, 2000. ISBN 84-7829-037-0.
SÁNCHEZ, A. M. G. Evaluación de métricas de calidad del software sobre un programa Java. 2010.
SÁNCHEZ, T. R. Metodología de desarrollo para la Actividad productiva de la Universidad de las Ciencias Informáticas. La Habana: 2014.
SIUV. Servicio de Informática Universidad de Valencia. In. Valencia: SIUV, 2015, vol. 2016.
SOMMERVILLE, I. *Ingeniería del Software*. Edtion ed.: Pearson. Addison Wesley, 2005. ISBN 84-7829-074-5.
SYMFONY. Symfony. In., 2015.
VISUAL-PARADIGM. Visual Paradigm. In., 2016, vol. 2016.

ANEXOS

Historias de usuario

Número: 1	**Nombre del requisito:** Adicionar libro.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 2 horas
Riesgo en Desarrollo: Bajo	**Tiempo Real:** 2 horas

Descripción:

El podrán registrar todos los datos correspondientes a los libros. En la vista s contendrá los campos **Título,** en el cual se registrará el título del libro, en el campo **Tomo** se registrará el tomo del libro, en el campo **Autor** el autor del libro. Los próximos campos son **Total de libros** y **Libros restantes** los cuales contendrán los libros existentes en el almacén de la facultad, uno representará el total de libros de un tipo que tiene la facultad (total de libros) y el otro los libros de un tipo que existen actualmente en el almacén de la facultad. En el campo Descripción se podrá introducir un comentario respecto al libro ingresado.

La vista cuenta con dos botones para realizar las acciones de aceptar el ingreso del libro (**Aceptar**) o regresar al listado de libros (**Regresar al listado**).

Observaciones:

Si no se introducen todos los datos del libro no se podrá completar la operación, no pueden quedar campos en blanco.

Número: 2	**Nombre del requisito:** Editar Libro.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Media	**Tiempo Estimado:** 2 horas
Riesgo en Desarrollo: Bajo	**Tiempo Real:** 3 horas

Descripción:

Permite modificar los datos de un libro en específico mostrando los mismos campos de la historia de usuario 1 con los datos originales.

Observaciones: Si no se introducen todos los datos del libro no se podrá completar la operación, no pueden quedar campos en blanco.

Número: 3	**Nombre del requisito:** Mostrar Libro.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Media	**Tiempo Estimado:** 2 horas
Riesgo en Desarrollo: Bajo	**Tiempo Real:** 1 horas
Descripción: Al hacer clic sobre uno de los libros mostrados en el listado de libros el sistema debe mostrar todos los datos de los libros en una vista. Esta vista brindará la opción mediante el botón **Editar** de redireccionar al usuario a la vista de modificar el libro o mediante el botón **Eliminar** eliminar dicho libro y posee un botón para cerrar la vista y pasar nuevamente al listado de libros.	
Observaciones:	

Número: 4	**Nombre del requisito:** Eliminar Libro.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Media	**Tiempo Estimado:** 2 horas
Riesgo en Desarrollo: Bajo	**Tiempo Real:** 2 horas
Descripción: El sistema brinda la posibilidad al usuario de eliminar un libro mediante la vista de mostrar libro.	
Observaciones:	

Número: 5	**Nombre del requisito:** Listar Libros.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Media	**Tiempo Estimado:** 2 horas
Riesgo en Desarrollo: Baja	**Tiempo Real:** 2 horas

Descripción: El sistema muestra en la vista principal de los libros el listado de todos los libros ingresados en el sistema.
Observaciones:

Número: 6	**Nombre del requisito:** Adicionar Préstamo de Libros.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 2 horas
Riesgo en Desarrollo: Media	**Tiempo Real:** 2 horas
Descripción: Se podrá seleccionar en la vista el nombre del estudiante o profesor al que se le realizará el préstamo del libro mediante el campo **Préstamo a** este campo tiene la opción de buscar el nombre de la persona al ingresar dicho nombren el campo. Mediante el campo **Libros** registrarán los libros que vallan a ser prestados seleccionando cada uno de ellos, este campo al igual que el anterior, brinda la posibilidad de acortar la lista de libros al escribir parte del nombre del libro. Se seleccionará la fecha del préstamo a través del campo **Fecha**, en el campo **Estado** se reflejará el estado del libro, el cual trae por defecto el estado de prestado. La vista cuenta con dos botones para realizar las acciones de aceptar el ingreso del préstamo (**Aceptar**) o regresar al listado de préstamos (**Regresar al listado**).	
Observaciones: Si no se introducen todos los datos del libros no se podrá completar la operación, no pueden quedar campos en blanco.	

Número: 7	**Nombre del requisito:** Editar Préstamo de libro.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Media	**Tiempo Estimado:** 2 horas
Riesgo en Desarrollo: Baja	**Tiempo Real:** 2 horas
Descripción: Permite modificar los datos de un préstamo en específico mostrando los mismos campos de la historia de usuario 6 con los datos originales.	
Observaciones: Si no se introducen todos los datos del préstamo no se podrá completar la operación, no pueden quedar campos en blanco.	

Número: 8	**Nombre del requisito:** Mostrar Préstamo de libro.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Media	**Tiempo Estimado:** 4 horas
Riesgo en Desarrollo: Media	**Tiempo Real:** 4 horas
Descripción: Al hacer clic sobre uno de los préstamos mostrados en el listado de préstamos el sistema debe mostrar el nombre, la categoría y el número de solapín del plan persona a la que se le realizó el préstamo. Esta vista brindará la opción mediante el botón **Modificar** cambiar el estado del libro una vez que él mismo haya sido entregado, el cual se actualizará mediante el botón **Actualizar**. La vista también brinda la posibilidad de eliminar un libro del préstamo mediante el botón **Eliminar**. Mediante el botón **Aceptar** se podrá regresar el listado de los préstamos.	
Observaciones: Se debe actualizar el estado del préstamo una vez que se haya cambiado para guardar el cambio.	

Número: 9	**Nombre del requisito:** Listar Préstamos de Libros.
Programador: Luis Javier Sánchez Hernández	**Iteración Asignada:** 1
Prioridad: Media	**Tiempo Estimado:** 2 horas
Riesgo en Desarrollo: Baja	**Tiempo Real:** 2 horas
Descripción: El sistema muestra en la vista principal de los préstamos el listado de todos los préstamos realizados.	
Observaciones:	

Anexo 2

Diagrama de clases:

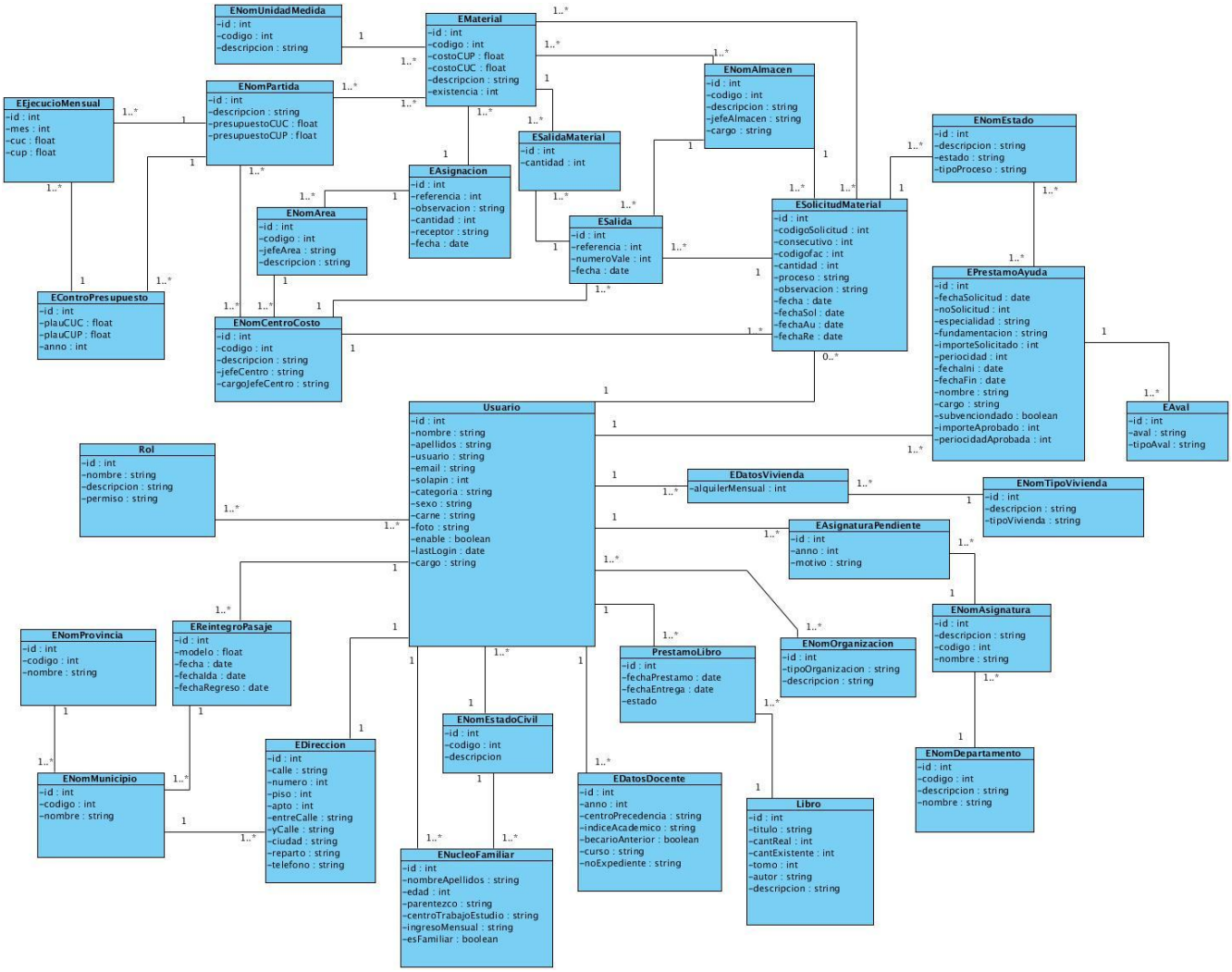


Figura 15: Diagrama de clases

Fuente: (Elaboración propia)

Diagramas de clases con estereotipos web

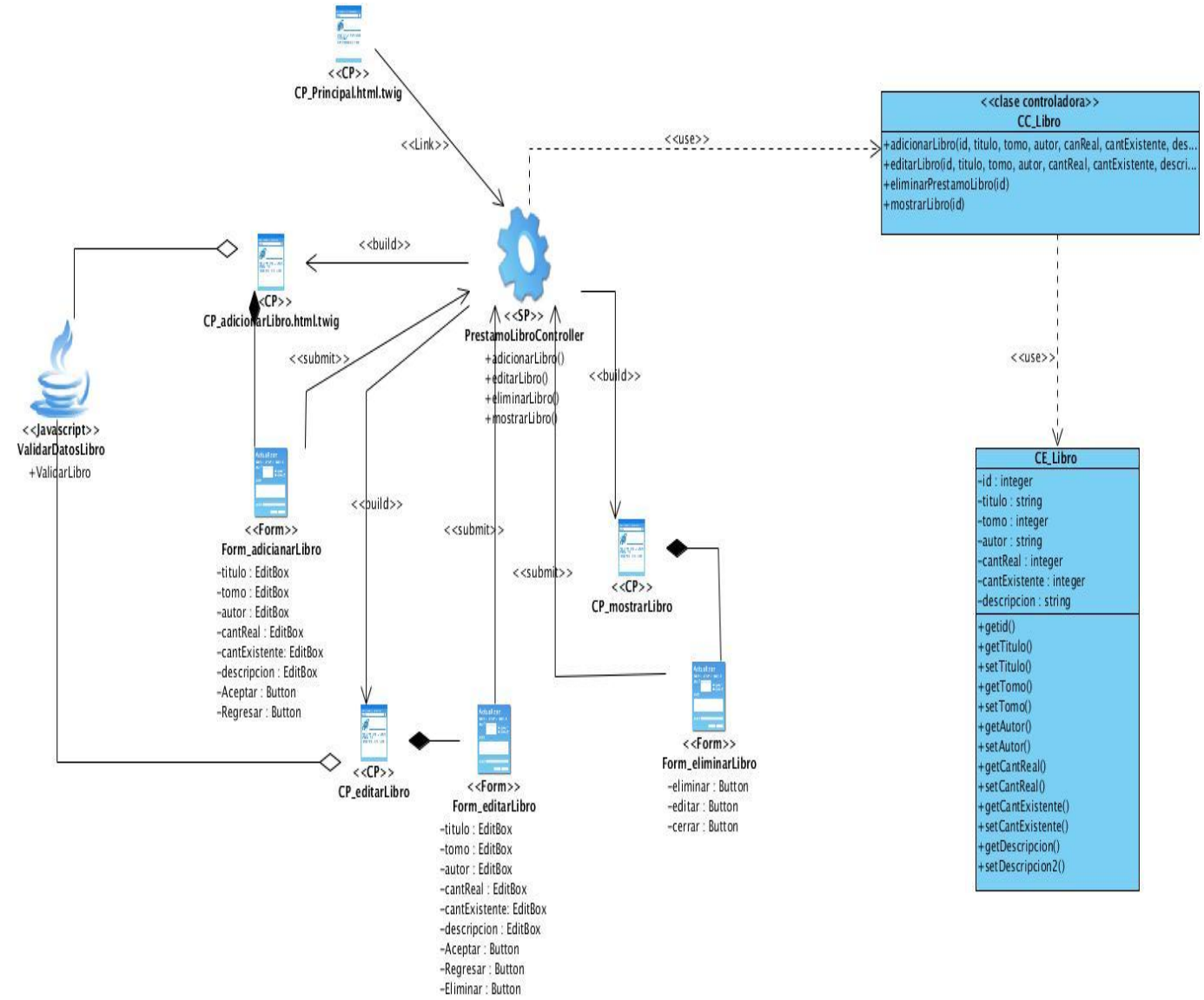


Figura 16: Libros de la facultad

Fuente: (Elaboración propia)

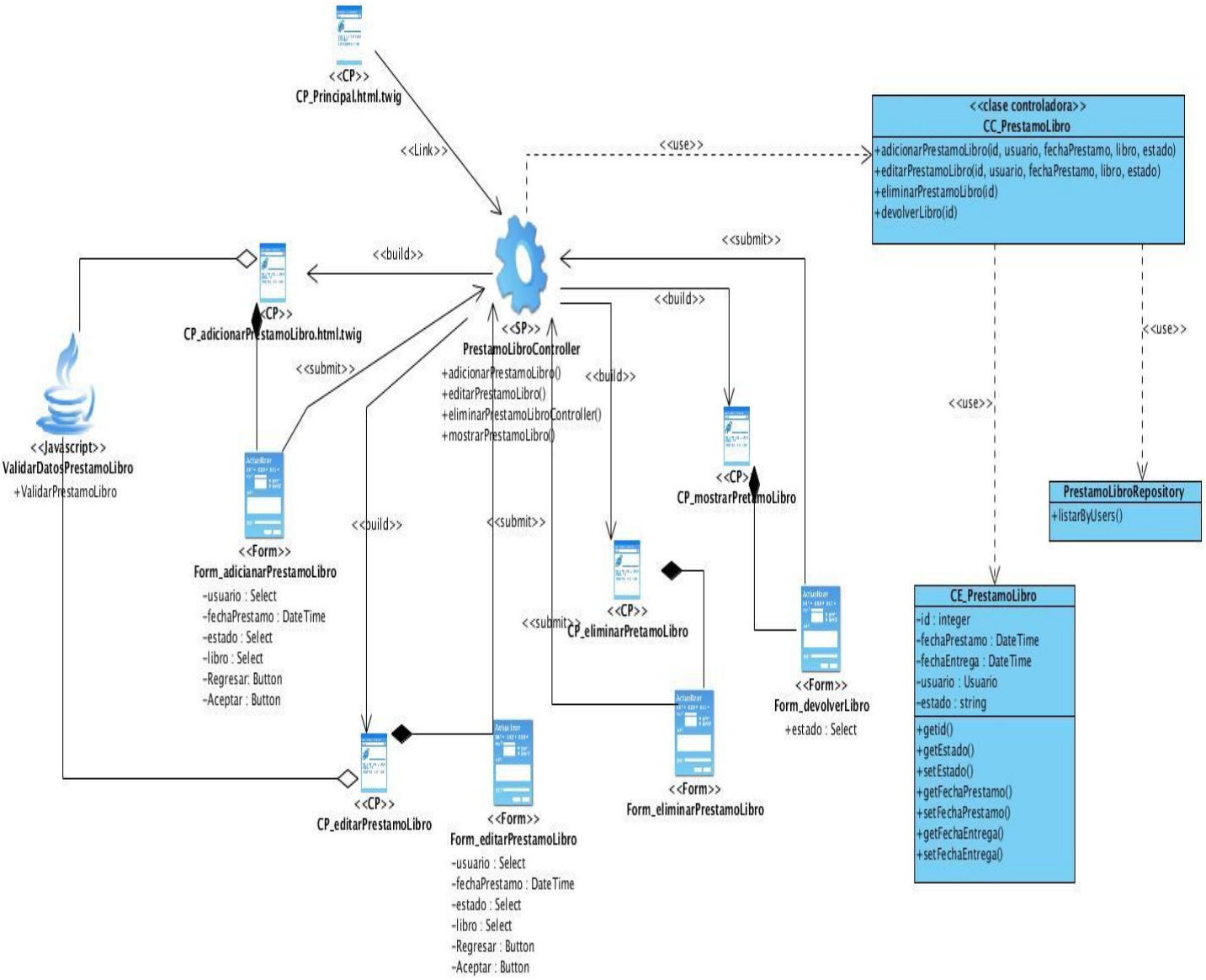


Figura 17: Préstamo y devolución de libros en uso

Fuente: (Elaboración propia)