

**Universidad de las Ciencias Informáticas
Facultad 2**

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

**Diseño e Implementación del Módulo Circulación del
Sistema ABCD 3.0**

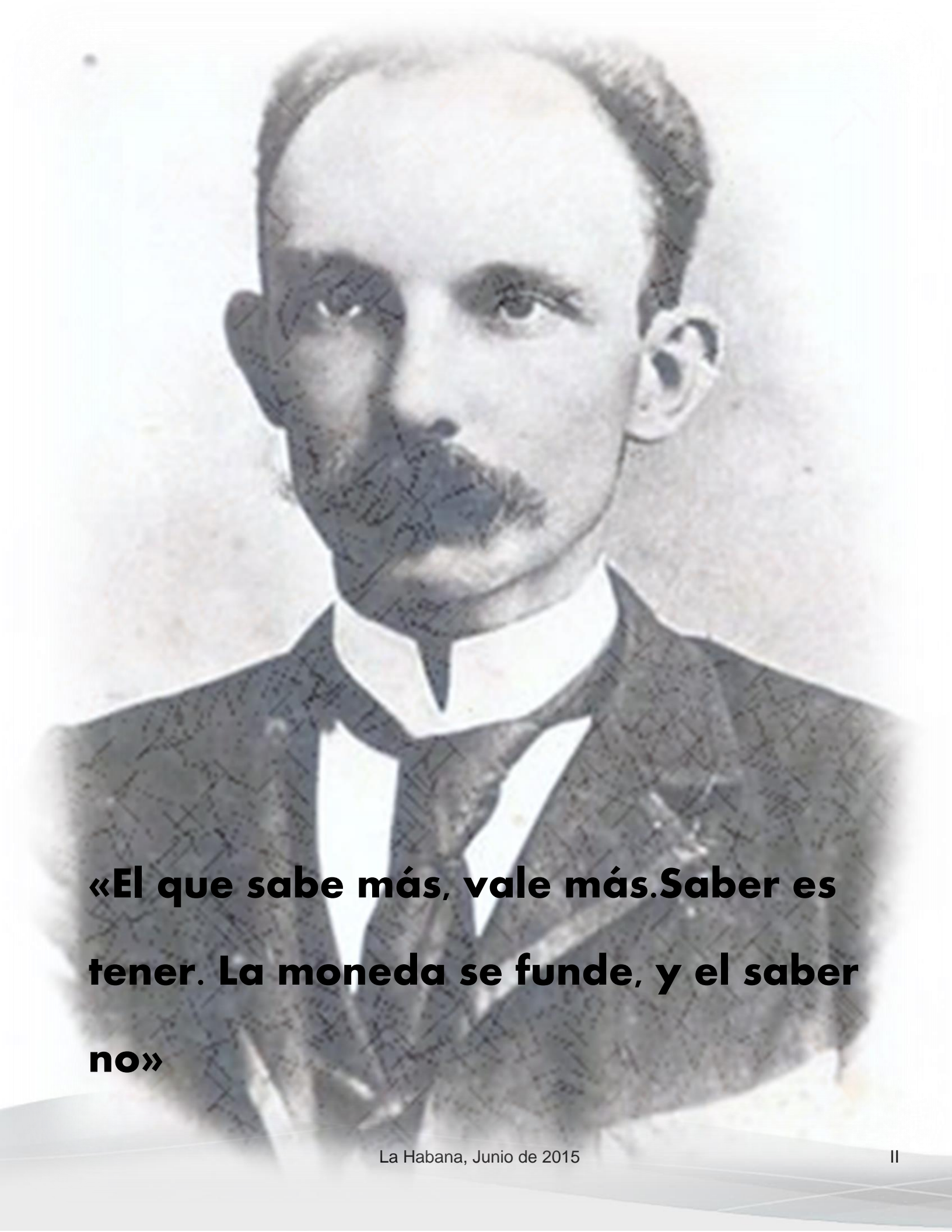
Autores: Abel Antonio Sánchez Rodríguez
Jorge Carlos Salazar Morales

Tutores: Ing. José Enrique Ximelis

Co-Tutor: Ing. Yanet Crespo Diaz
Ing. José Rolando Lafaurié Olivares

La Habana, junio de 2015

“Año 57 de la Revolución”



«El que sabe más, vale más. Saber es tener. La moneda se funde, y el saber no»

De Jorge Carlos

Agradezco infinitamente a mi madre por significar todo para mí y ser el único motivo por el cual hoy estoy en este lugar, alcanzando una meta que me propuse desde hace ya algún tiempo. A esa persona en la cual mi madre ha encontrado un compañero para apoyarse: Rafael, te agradezco inmensamente por estar presente. A mi familia, sobre todo a mi tía Ileana por preocuparse esporádicamente por mí, cuando me extrañaba porque no iba a la casa. A mi primas Anaelys, Lendys, Orisel, Belinda y a mis primitos Brian y William por existir y hacerme los momentos cotidianos de la vida, grandes historias para contar. A la gente del barrio, que desde segundo año me están preguntando cuando es que llegaría este día, no sé para qué. Gracias a la familia de mi compañero de tesis, a Zoraida, a Papo y al Javie, gracias por tanta caldosa.

A todas las personas con las que pude contar para echar el partidito de fútbol de por las tardes. A mi gente del “Real Barça”, que son una de las cosas fundamentales que me llevo de esta experiencia. Gracias a ustedes por haberme dado la idea de fundar un equipito que tuvo como primer nombre “Amiguitos de Messi”; logrando luego convertirse en lo que es hoy, más que una familia. Muchas gracias a todos los que formaron parte de este team ganador. Agradecer a todas las personas que compartieron conmigo cada edición de los Juegos Mella, tanto en la Facultad 2, como en la 6, quisiera haber podido hacer más de lo que demostré... pero principalmente, gracias al equipo de la Facultad 7, liderado por Pochet cuando alcanzamos el segundo lugar, es uno de los momentos que nunca en la vida voy a olvidar. A mis compañeros de aula que fueron cómplices de todas mis malacrianzas y disparates. Agradecer a mis compañeros de apartamento, que hace mucho que dejaron de serlo para convertirse en los hermanos que nunca tuve.... Gracias al Sevi, el Buty, al Yosva, y negro que tengo por compañero de tesis.

Muchas gracias a esas mujeres especiales, con las cuales he vivido experiencias increíbles; gracias Yisel por ser mi hermanita y mi nada al mismo tiempo. Gracias a Rosa, aunque siempre nos fajábamos por boberías, en el fondo sabes que te quiero muchísimo y que cuentas conmigo para casi todo. No alcanzan los agradecimientos para ustedes dos.

Siempre recordaré a esa pareja que son Suinni y Magdiel. Los adoro con la vida negros, y recuerden que seguiré viniendo solo para verlos.

Gracias a mi primo Damián, por estar ahí cuando me hacía falta y por ser mi ejemplo de superación a seguir. Gracias a Nirving, por ser la persona que cada día me recuerda que puedo ser mejor y que confíe

más en mí. Gracias por ser mi amigo y mi hermano, por escucharme y aconsejarme tanto en estos últimos tiempos de locura que he tenido. Sin ti no sé ni que hubiera hecho, tú sabes por qué. Gracias a mi padre, que aunque nunca en la vida ha estado ahí, para aconsejarme, dame su opinión, verme crecer, verme sonreír, bailar, o simplemente darme un beso; le debo agradecer lo único que ha hecho por mí en toda su vida, gracias por tu aporte.

En general, gracias a mis tutores, sobre todo a Yanet por su paciencia y poder de entendimiento; también a los que me conocen en el proyecto. Muchas gracias a todos con los que he compartido y espero poder tener la posibilidad de seguir escribiendo esta historia con ustedes como protagonistas.

De Abel

Gracias a mi madre por ser el motivo principal por el cual hoy soy ingeniero. A mi familia, a mi papá que aunque esté lejos, sé que ahora mismo está muy orgulloso de mi y de la persona en la que me he convertido. Gracias a mi hermano Javier, a Papo, a todos mis amigos del barrio, a Leonel y Martica, a mi abuelo y a mi novia por estar ahí para mí en todo momento.

Agradecer eternamente a mis compañeros de aula por soportarme, a mis hermanos del apartamento que desde hace rato son parte también de mi familia, Osvaldo, Sevilla, Yosbani, mi compañero de tesis, el Negro y a todos los demás con los que he compartido en la Universidad.

Gracias a todos los que integran el proyecto, a mis tutores, sobre todo a Yanet, que ha sabido lidiar con estos dos animalitos; y a toda la gente del proyecto. Gracias a todas las personas que han compartido conmigo en estos 5 años, son parte de esta gran experiencia que espero poder contar con mucha alegría. Gracias a todos por estar presentes.

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos al Centro de Informatización de Gestión Documental (CIGED) de la Universidad de las Ciencias Informáticas (UCI) a hacer uso del mismo en su beneficio.

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

Abel Antonio Sánchez Rodríguez

Jorge Carlos Salazar Morales

Firma del autor

Firma del autor

Ing. José Enrique Ximelis

Ing. Yanet Crespo Diaz

Firma del tutor

Firma del tutor

DATOS DE CONTACTO

Ing. José Enrique Ximelis Alvarez

Actualmente trabaja en el Centro de Informatización de la Gestión Documental (CIGED), en el Departamento de Desarrollo de Aplicaciones, ocupando el cargo de Especialista “B” en ciencias informáticas.

Correo electrónico: jeximelis@uci.cu

Ing. Yanet Crespo Diaz

Actualmente trabaja en el Centro de Informatización de la Gestión Documental (CIGED), en el Departamento de Desarrollo de Componentes, ocupando el cargo de recién graduada en adiestramiento (NS).

Correo electrónico: ycdias@uci.cu

RESUMEN

En Cuba, a partir de la introducción de las Tecnologías de la Información y las Comunicaciones, se han revolucionado considerablemente los servicios brindados, tales como la televisión, telefonía móvil y el internet. Una de las instituciones beneficiadas con estas tecnologías, son las bibliotecas públicas, las cuales se encargan de gestionar los materiales disponibles para el consumo de información de los usuarios. Actualmente existen deficiencias en el sistema de gestión bibliotecaria que se utiliza en las bibliotecas universitarias del país, específicamente en los procesos de préstamo de materiales bibliográficos. Es por esta razón que el objetivo de la presente investigación es desarrollar un módulo que contribuya a unificar la gestión de información en el proceso de préstamos de materiales bibliográficos, en el sistema de Automatización de Bibliotecas y Centros de Documentación (ABCD) en su versión 3.0.

En este trabajo se realiza un estudio detallado de los Sistemas Integrados de Gestión Bibliotecaria (SIGB), enfocados principalmente en el proceso de préstamo de materiales bibliográficos. Se especifican cada uno de los problemas existentes en la versión anterior del sistema a desarrollar. Se definen las tareas realizadas a lo largo de la investigación, las herramientas y tecnologías que se utilizaron para el desarrollo del sistema en cuestión y además la metodología de desarrollo de software que se utilizó para estructurar, planificar y controlar el proceso de desarrollo de la propuesta de solución. Se presentan artefactos de ingeniería de software que especifica la metodología para los flujos de diseño e implementación del Módulo Circulación del Sistema ABCD 3.0. Se presentan los resultados de las pruebas funcionales y de integración, correspondientes a las pruebas de caja negra con el fin de detectar y corregir errores existentes, dando lugar a la obtención de una solución con una mejor calidad.

PALABRAS CLAVE

Gestión bibliotecaria, gestión de información, proceso de préstamo, materiales bibliográficos.

TABLA DE CONTENIDOS

INTRODUCCIÓN	6
CAPÍTULO 1: FUNDAMENTOS TEÓRICOS	10
1.1 Conceptos fundamentales.....	10
1.2 Sistemas existentes.....	11
1.3 Metodología y Entorno de desarrollo	14
1.3.1 Metodología Rational Unified Process (RUP).....	14
1.3.2 Lenguaje de modelado UML v2.0	15
1.3.3 Herramienta CASE (Visual Paradigm v8.0).....	16
1.3.4 Framework RAP v3.0.....	16
1.3.5 Lenguajes de desarrollo (Java v1.7)	16
1.3.6 Entorno de desarrollo integrado (Eclipse Kepler Release v4.3)	16
1.3.7 Gestor de base de datos (Postgres v9.3.5).....	17
1.3.8 Servidor de aplicaciones (Virgo v3.6.3).....	18
1.3.9 Framework de desarrollo Equinox (OSGi v3.8.2)	18
1.3.10 Spring Framework v3.1	18
Conclusiones del capítulo.....	19
CAPÍTULO 2: CARACTERÍSTICAS Y DISEÑO DE LA SOLUCIÓN	20
2.1 Reglas del Negocio.....	20
2.1.1 Actores del Sistema	20
2.1.2 Descripción de casos de uso	21
2.2 Diseño de la solución.....	23
2.2.1 Diagrama Entidad Relación	23
2.2.2 Diagrama de Clase del Diseño	26
2.2.3 Diagrama de Secuencia.....	27
2.2.4 Patrón de arquitectura.	29
2.2.5 Patrones de Diseño utilizados.....	34
Conclusiones de capítulo.....	35
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA DE LA SOLUCIÓN	37
3.1 Implementación	37
3.1.1 Clases de dominio	37
3.1.2 Clases controladoras	37

3.1.3. Clases servicio.....	38
3.1.4 Clases de acceso a datos.....	38
3.2 Estándares de codificación.....	38
3.3 Tratamiento de errores.....	40
3.4 Diagramas de componentes.....	41
3.5 Diagrama de despliegue.....	41
3.6. Funcionalidades.....	42
3.6.1 Registrar sanción.....	42
3.6.2 Consultar usuario de préstamo.....	43
3.7 Pruebas.....	44
3.7.1 Tipo de prueba.....	44
3.7.2 Diseño de caso de prueba.....	47
3.7.3 Resultado de las pruebas.....	57
Conclusiones del capítulo.....	59
CONCLUSIONES.....	61
RECOMENDACIONES.....	62
REFERENCIAS.....	63
BIBLIOGRAFÍA.....	65

ÍNDICE DE FIGURAS

Figura 1: Estructura de RUP	15
Figura 2: Diagrama Entidad-Relación del Módulo Circulación del Sistema ABCD 3.0	24
Figura 3: Diagrama de clase del diseño (Gestionar Transacción).....	26
Figura 4: Diagrama de Secuencia correspondiente a Adicionar Transacción	28
Figura 5: Estructura de la arquitectura n capas (30)	30
Figura 6: Ejemplo de la utilización de la arquitectura n capas (Capa de presentación del diagrama de clase del diseño Gestionar Transacción)	32
Figura 7: Ejemplo de la utilización de la arquitectura n capas (Capa de negocio del diagrama de clase del diseño Gestionar Transacción).....	33
Figura 8: Ejemplo de la utilización de la arquitectura n capas (Capa de datos del diagrama de clase del diseño Gestionar Transacción)	34
Figura 9: Ejemplo de utilización del patrón experto	34
Figura 10: Ejemplo de utilización del patrón creador	35
Figura 11: Ejemplo de utilización de estándares de codificación (bloque for).	39
Figura 12: Ejemplo de utilización de estándares de codificación (bloque if).....	39
Figura 13: Ejemplo de utilización del bloque try/catch para el tratamiento de errores.	40
Figura 14: Diagrama de componente correspondiente al CU Gestionar Préstamo	41
Figura 15: Diagrama de Despliegue del Sistema.....	42
Figura 16: Interfaz correspondiente a la funcionalidad "Registrar Sanción"	43
Figura 17: Interfaz correspondiente a la funcionalidad "Consultar usuario de préstamo"	44
Figura 18: Librerías y bundles utilizadas por el módulo Circulación.....	46
Figura 19: Servicios definidos en el archivo osgi.beam.definition.xml.....	47
Figura 20: Gráfico correspondiente al resultado de las pruebas funcionales realizadas.	57
Figura 21: Gráfico correspondiente al resultado de las pruebas de integración realizadas.	59

ÍNDICE DE TABLA

Tabla 1: Tabla comparativa entre los sistemas estudiados.....	13
Tabla 2: Actores del Sistema.....	20
Tabla 3: Descripción de Casos de Uso.....	21
Tabla 4: Caso de prueba "Consultar Transacciones".....	47

INTRODUCCIÓN

Hablar hoy día de las bibliotecas, es hablar de instituciones que han sufrido una serie de transformaciones y que no ha sido ajena al panorama de la introducción de las Tecnologías de la Información y las Comunicaciones (TICs). *“Al respecto, desde el desarrollo e implementación de las TIC, se han dado cambios en cuanto a los procesos de comunicación, lectura y transferencia de información. Sin embargo, es importante acotar que estos cambios son dados por las personas, al transformar su mentalidad y cambiar sus prácticas, pues la herramienta por sí sola no logra alcanzar niveles de desarrollo”.* (1) *“La biblioteca digital surge dentro de la sociedad de la información, y esta para desarrollarse de forma democrática y sostenible, necesita una biblioteca pública vigorosa, actualizada, operativa y al alcance de todos”.* (2)

Las bibliotecas institucionales y públicas se encargan de llevar la gestión de los medios en el proceso de entrada y salida de los libros, documentos, manuales y artículos. La automatización de bibliotecas podría definirse como el uso de herramientas y técnicas informáticas en las tareas bibliotecarias.

“Los Sistemas Integrados de Gestión Bibliotecaria (SIGB), surgieron en las bibliotecas más avanzadas hacia finales de los 70 y principios de los 80. Estos sistemas utilizan una arquitectura única para gestionar los procesos básicos de las bibliotecas como la catalogación, el préstamo, las adquisiciones, y los presupuestos, y a veces, otras tareas afines pero quizás menos importantes como el préstamo inter-bibliotecario y otras funciones de tipo administrativo”. (3)

Actualmente en las bibliotecas universitarias del país, se cuenta con el sistema Automatización de Bibliotecas y Centros de Documentación (ABCD) en su versión 1.2, el cual, posibilita la gestión de todos los materiales y contenidos disponibles en la biblioteca, registra las reservaciones realizadas por los diferentes usuarios que acuden a la biblioteca con la necesidad de consultar algún material disponible en la institución. Además de realizar las operaciones mencionadas, se encarga de la gestión de las sanciones que se les aplican a los usuarios que incumplen con los períodos de reservación.

Sin embargo, este sistema cuenta con algunas limitaciones como la incorrecta validación en los formularios de entrada de datos, mientras que en una misma interfaz, el usuario puede encontrarse con lenguajes diferentes. Por otra parte, este sistema no fue implementado por especialistas en el tema, utilizando herramientas y tecnologías que no potencian al máximo este tipo de aplicaciones. Este sistema cuenta con los módulos EmpWeb y Circulación para gestionar los procesos de préstamo, entre los cuales no existe interoperabilidad, dando lugar a imperfecciones en la definición del flujo de préstamo, por lo que se le hace engorroso al usuario realizar cualquier transacción en el sistema. La existencia de estos módulos para realizar los procesos de préstamo, afecta la

escalabilidad del sistema en el momento de brindarle soporte y mantenimiento, dificultando el crecimiento continuo del mismo. Esta aplicación cuenta además con la deficiencia de que no se permite realizar préstamos sobre los objetos que son de tipo revista.

Con el fin de corregir las deficiencias mencionadas anteriormente, se le asignó como tarea a la Universidad de las Ciencias Informáticas (UCI), el desarrollo de la versión 3.0 de ABCD.

Por la situación planteada anteriormente, se identifica como **problema a resolver**: ¿cómo unificar los procesos gestión de préstamo de materiales bibliográficos del Sistema ABCD 3.0?

Por esta razón, se define como **objeto de estudio** el proceso de préstamo de materiales bibliográficos, enmarcado en el **campo de acción**, los procesos de préstamo de materiales bibliográficos dentro de los sistemas de gestión bibliotecaria.

Para resolver el problema identificado se propone el siguiente **objetivo general**, desarrollar las funcionalidades asociadas al Módulo Circulación del Sistema ABCD 3.0, para unificar el proceso de préstamo de materiales bibliográficos.

Para cumplir con el objetivo general de la investigación se trazaron los siguientes **objetivos específicos**:

- Realizar un estudio de los sistemas integrados de gestión bibliotecaria, con el fin de caracterizar este tipo de soluciones, para lograr obtener la fundamentación teórica necesaria para el desarrollo de la investigación, además de caracterizar la metodología, herramientas y tecnologías utilizadas para realizar el diseño e implementación del módulo.
- Realizar el diseño de la solución propuesta, mediante el modelado y confección de los artefactos correspondientes con los flujos definidos por la metodología seleccionada, empleando los patrones de diseño pertinentes.
- Realizar la implementación del sistema empleando los estándares de codificación definidos por el equipo de proyecto y realizar las pruebas funcionales y de integración para la detección y corrección de errores, para contribuir a la validación del sistema.

Para dar solución al problema científico planteado y cumplir los objetivos propuestos, se definen las siguientes **tareas de la investigación**:

- Análisis de los principales conceptos asociados al proceso de préstamo de materiales bibliográficos para obtener la base teórica necesaria en el desarrollo de la solución.
- Estudio y caracterización de los sistemas existentes que gestionan préstamos de materiales bibliográficos, para determinar particularidades de cada uno de los procesos llevados a cabo en estos sistemas.
- Estudio y análisis de la metodología, herramientas y tecnologías definidas para el diseño e

implementación del Módulo de Circulación.

- Análisis y aplicación de los patrones de diseño y de arquitectura durante el proceso de diseño e implementación de la solución para eliminar problemas comunes en el desarrollo del módulo.
- Elaboración de los diagramas de clases del diseño del Módulo de Circulación para visualizar las relaciones entre las clases que involucran el sistema.
- Implementación de las clases diseñadas para obtener las funcionalidades definidas anteriormente.
- Diseño de los casos de pruebas del Módulo Circulación para comprobar la correcta implementación de cada uno de los requisitos definidos.
- Ejecución de las pruebas de software para la detección y corrección de errores.

Para el desarrollo de la investigación se utilizaron diversos **métodos científicos**. Dentro de los **métodos teóricos** utilizados, se encuentran:

Histórico – Lógico: se utiliza en la investigación para determinar los antecedentes históricos relacionados con los sistemas de gestión bibliotecaria, posibilitando el análisis de la trayectoria de estos sistemas para comprender lógicamente cuales son las tendencias actuales.

Inductivo – Deductivo: se realizó la revisión y análisis del comportamiento y las características del proceso de gestión bibliotecaria, para así poder deducir conclusiones sobre casos particulares que puedan ser verificados en la práctica.

Analítico-Sintético: facilita el resultado del trabajo con el análisis de gran cantidad de bibliografía sobre el proceso de gestión bibliotecaria, permitiendo sintetizar las características generales.

Estructuración del trabajo de diploma

Capítulo.1 Fundamentos teóricos: se realiza la fundamentación teórica necesaria y la definición de conceptos que apoyan este trabajo de diploma y que resultan de interés en la presente investigación. Se presenta un estudio de los Sistemas Integrados de Gestión Bibliotecaria (SIGB), dirigido específicamente a los procesos de préstamo de materiales bibliográficos. Se realiza un análisis de la metodología de desarrollo de software, herramientas, tecnologías y lenguajes a utilizar para el desarrollo de la solución propuesta.

Capítulo.2 Características y diseño de la solución: se realiza la descripción de los casos de uso definidos, de los actores que interactúan con el sistema, además de una breve caracterización de los elementos que componen la solución propuesta. Se identifica la arquitectura y los patrones de

diseño utilizados, reflejados directamente en los diagramas de clase de diseño presentados. Además, se muestran diagramas de secuencias para identificar las inter-relaciones entre las clases definidas y el diagrama de entidad-relación.

Capítulo.3 Implementación y Prueba de la solución: se realiza la implementación de las clases para obtener las funcionalidades definidas empleando los estándares de codificación, además de presentarse los diagramas de componente y el modelo de despliegue. Se describen las pruebas de software realizadas para la detección y corrección de errores.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

Introducción

En el presente capítulo se definen los conceptos fundamentales relacionados con los sistemas de gestión bibliotecaria, con el fin de establecer la base teórica necesaria para el desarrollo de la investigación. Además se realiza un estudio del estado del arte utilizando los métodos científicos de investigación, con el fin de analizar algunos de los sistemas integrados de gestión bibliotecaria existentes a nivel mundial, para determinar características particulares de cada uno de ellos. Se caracterizan también las herramientas, tecnologías y la metodología de desarrollo de software que se utilizaron para la implementación del sistema.

1.1 Conceptos fundamentales.

Para una mayor comprensión del presente trabajo investigativo, se analizaron los conceptos asociados al proceso de préstamo de materiales bibliográficos, definido como objeto de estudio de la investigación.

Según la Real Academia Española (RAE), un **proceso** es un *“conjunto de fases sucesivas de un fenómeno natural o de una operación artificial”*. (4)

La Dra. Inmaculada Blaya, directora de servicios, perteneciente a la Oficina de Gestión y Control de la Calidad de la Universidad Miguel Hernández de Elche, España, define un **proceso** como un *“conjunto de actuaciones, decisiones, actividades y tareas que se encadenan de forma secuencial y ordenada para conseguir un resultado que satisfaga plenamente los requerimientos del cliente al que va dirigido”*. Además de un *“conjunto de actividades secuenciales que realizan una transformación de una serie de entradas en las salidas deseadas añadiendo valor”*. (5)

Luego de estudiado algunos conceptos relacionados con los procesos, los autores de esta investigación concluyen que un proceso no es más que un conjunto de pasos organizados y relacionados entre sí, que tienen como objetivo final, conseguir un resultado satisfactorio luego de transformar un conjunto de entradas y agregarle valor.

Según la comunidad catalográfica, se concibe el **registro bibliográfico** como *“una agregación de datos que se asocian con las entidades descritas en los catálogos bibliotecarios y en las bibliografías nacionales. Dichos datos describen, representan y posibilitan la organización de materiales textuales, musicales, audiovisuales, cartográficos, gráficos y tridimensionales”*. (6) Los autores de esta investigación entienden por registro bibliográfico los datos que permiten organizar de forma eficiente los medios que se encuentran en la institución, que pueden ser consumidos por los usuarios de la biblioteca.

El licenciado en Literatura Española, Luis Ángel García Melero, define un **Sistema Integrado de Gestión Bibliotecaria** (SIGB) como *"un conjunto organizado de recursos humanos que utilizan dispositivos y programas informáticos, adecuados a la naturaleza de los datos que deben procesar, para realizar procesos y facilitar los servicios que permiten alcanzar los objetivos de la biblioteca: almacenar de forma organizada el conocimiento humano contenido en todo tipo de materiales bibliográficos para satisfacer la necesidades informativas, recreativas y de investigación de los usuarios"*. (7)

Teniendo en cuenta el análisis realizado, los autores de este trabajo concluyen que un SIGB es un programa informático destinado a administrar todas las funciones realizadas por una biblioteca. Esto se logra con la implementación de servicios que le posibiliten a los usuarios el fácil acceso al conocimiento almacenado en forma de información en la institución.

Según la Real Academia Española (RAE), un préstamo es la *"acción y efecto de prestar, entregar algo a alguien para que lo devuelva"*. (4) Debido al análisis de conceptos relacionados con los préstamos, los autores concluyen que el **préstamo de materiales bibliográficos** no es más que el proceso de entregarle al usuario los medios solicitados por un periodo determinado, con el fin de que este pueda acceder a la información contenida en el medio y luego entregarlo a la institución. Este proceso se realiza con una serie de condiciones definidas previamente, definiendo así las consecuencias de la pérdida de los medios prestados.

1.2 Sistemas existentes.

Para realizar la investigación es muy importante saber que se ha realizado previamente en relación a los sistemas de automatización de bibliotecas y centros de documentación, para así tener una base general que posibilite un mejor entendimiento del funcionamiento de los mismos y lograr cumplir satisfactoriamente las necesidades sociales de la población del consumo organizado de información.

A nivel mundial han sido elaboradas y diseñadas varias soluciones informáticas, con la finalidad de gestionar los procesos de materiales bibliográficos. A continuación se muestran algunas de estas herramientas, donde se hace énfasis específicamente en los elementos que integran dichas soluciones relacionadas con los procesos de préstamo, dado que es relevante para la investigación:

1.2.1 Aleph 500

Es un sistema de gestión bibliotecaria que cuenta con algunos de los siguientes módulos:

- Préstamo
- Préstamo inter-bibliotecario
- Ejemplares

- Búsqueda (OPAC)
- Revistas (8)

La política de préstamo tiene en cuenta el estatus del usuario definido por la biblioteca, el calendario y la ubicación de los ejemplares para determinar la fecha y hora de vencimiento. El sistema verifica instantáneamente las sanciones del usuario y las solicitudes de los ejemplares asegurando el manejo adecuado de los materiales. *“ALEPH 500 permite administrar ejemplares de préstamo corto para periodos tan breves como 30 minutos y permite préstamos cortos fijos y dinámicos. Los usuarios pueden hacer reservas avanzadas de ejemplares incluidos en un calendario definido por la biblioteca”.* (9)

Las funciones de reserva y lectura de cursos, proporcionan acceso tanto a materiales de reserva de cursos tradicionales como electrónicos. ALEPH 500 soporta un sub-módulo de Sala de Lectura que registra los materiales prestados a usuarios para uso en la misma.

1.2.2 Janium

“Permite gestionar todos los servicios de la biblioteca en una forma centralizada y optimiza el flujo de trabajo. Además, garantiza el acceso a la información y la posibilidad de actualizar la plataforma de manera sencilla y eficaz. Puede trabajar en las más diversas plataformas sin disminuir sus capacidades”. (10) El módulo de circulación permite definir políticas de circulación distintas e independientes para cada biblioteca. A través de este módulo se pueden realizar:

- *Préstamos, devoluciones, y renovación de materiales.*
- *Impresión de tickets de préstamo y devolución.*
- *Registro de usuarios.*
- *Gestión de sanciones y multas.*
- *Envío de notificaciones vía correo electrónico, impresas, o vía teléfono móvil (sms) etc.* (11)

Desde el Catálogo al público, el usuario puede enviar una solicitud de préstamo del material que se encuentra disponible para su consulta. Esta función permite al usuario enviar la solicitud al mostrador de préstamo para que le sea facilitado el préstamo de un material determinado. *“Cuando el usuario envía una solicitud, el material queda bloqueado por un tiempo determinado (que es configurable en minutos u horas) para que el usuario pueda pedirlo en el mostrador de préstamo”.* (11)

1.2.3 Koha

“Es un SIGB, desarrollado en Nueva Zelanda por la empresa Katipo Communications en el año 1999. Es un software de código abierto liberado bajo Licencia Pública General (GPL), mantenido por

un grupo de desarrolladores de distintos países. Presenta los módulos básicos OPAC, catalogación, préstamo de documentos, administración del sistema (relacionados específicamente con el préstamo de materiales bibliográficos)". (12)

Características del módulo circulación:

- Préstamos, devoluciones y renovaciones de materiales.
- Impresión de recibos de préstamo y devolución.
- Gestión de reservas.
- Gestión de transferencias.
- Gestión de Multas monetarias y administrativas.
- Cada biblioteca dentro del sistema puede tener políticas de circulaciones distintas e independientes.
- Informes de circulación.
- Seguimiento de préstamo en sala.
- Módulo de auto préstamo interno.
- Utilidad de circulación offline. (13)

Además del análisis realizado anteriormente, los autores de este trabajo definieron varios indicadores a evaluar, siendo estos: la posibilidad de modificación del código fuente, la correcta definición de los flujos de préstamo, la correcta internacionalización del sistema y si se pueden integrar con las tecnologías y herramientas definidas por el proyecto para el diseño e implementación de la solución.

Tabla 1: Tabla comparativa entre los sistemas estudiados

Indicadores	ABCD 1.2	Aleph	Janium	Koha
Modificación del código fuente	Si	No	No	Si
Correcta definición de los flujos de préstamo	No	Si	Si	Si
Correcta internacionalización	No	Si	Si	Si
Integración tecnológica	No	No	No	No

Una vez analizados estos indicadores se determinó que existen varios sistemas destinados a la automatización de bibliotecas y centros de documentación a nivel mundial, pero ninguno de los sistemas analizados anteriormente proporcionan una solución factible para el problema planteado anteriormente. Por este motivo se propuso realizar la implementación del módulo Circulación del Sistema ABCD 3.0.

1.3 Metodología y Entorno de desarrollo

En este epígrafe se realiza una caracterización de la metodología definida para guiar el proceso de desarrollo de la solución propuesta, definiendo específicamente las fases entre las cuales se enmarca la investigación y las disciplinas desarrolladas. Además, se caracterizan las herramientas y tecnologías definidas para realizar el diseño e implementación del módulo Circulación del Sistema ABCD 3.0.

1.3.1 Metodología Rational Unified Process (RUP)

El Proceso de Desarrollo Unificado es un proceso de ingeniería de software que proporciona un enfoque disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo de software. Su objetivo es garantizar la producción de software de alta calidad que satisfaga las necesidades de sus usuarios finales, dentro de un previsible cronograma y presupuesto. *“RUP identifica 6 buenas prácticas con las que define una forma efectiva de trabajar para los equipos de desarrollo de software.*

- *Gestión de requisitos*
- *Desarrollo de software iterativo*
- *Desarrollo basado en componentes*
- *Modelado visual (usando UML)*
- *Verificación continua de la calidad*
- *Gestión de los cambios” (14)*

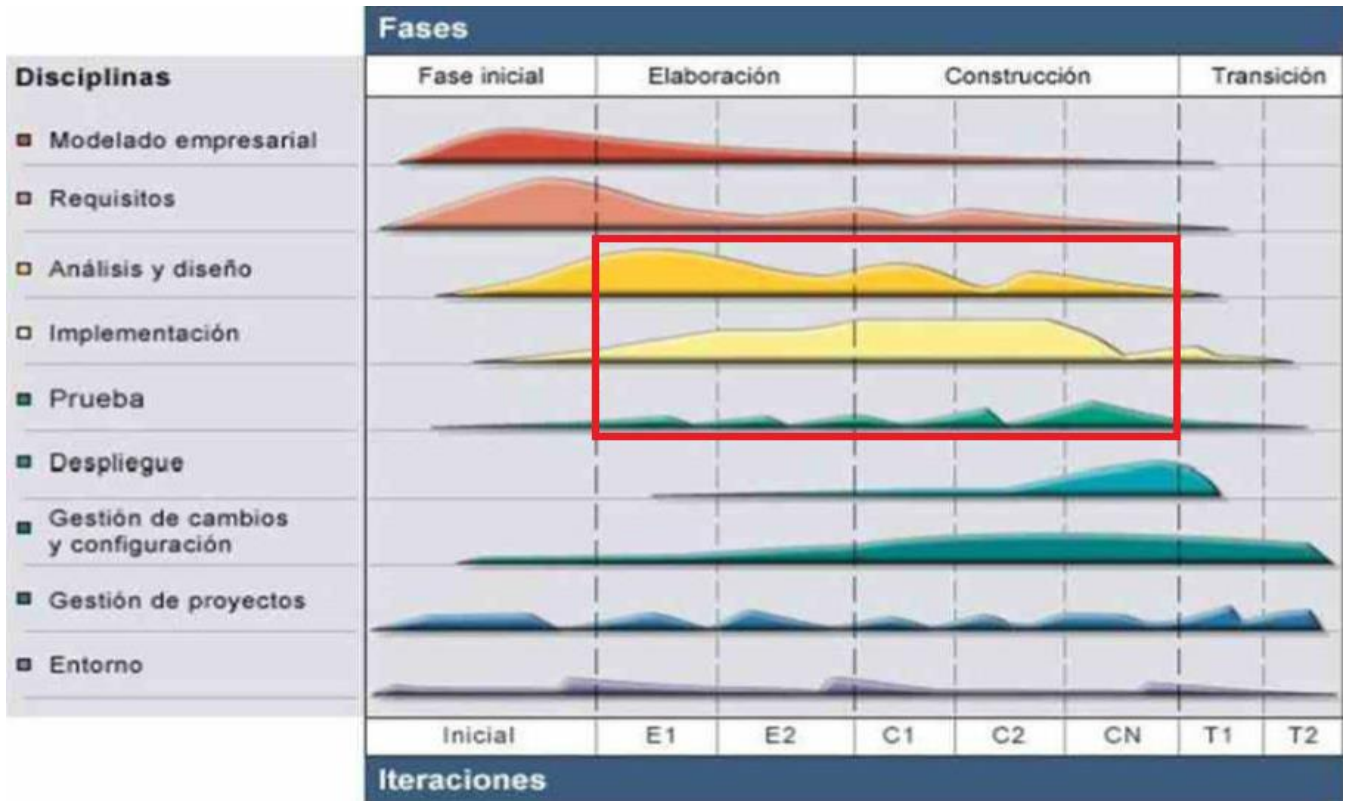


Figura 1: Estructura de RUP

La investigación se centró solamente en las fases de elaboración y construcción, utilizando los artefactos generados anteriormente en las disciplinas de modelado de negocio, definición de requisitos y análisis, brindados por el equipo del proyecto. Los artefactos generados fueron utilizados para realizar las disciplinas de diseño, implementación y prueba.

1.3.2 Lenguaje de modelado UML v2.0

Se utiliza el lenguaje de modelado UML (Unified Modeling Language) que prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos y describe la semántica esencial de estos diagramas y los símbolos en ellos utilizados. Con el lenguaje UML, los diseñadores sólo tienen que aprender una única notación que vale para los diferentes aspectos del diseño y construcción de un hipertexto. *“Se puede emplear también para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, organizaciones del mundo real, etc. Específicamente se utiliza en la investigación para modelar los diagramas de clase de diseño, diagramas de secuencia y diagrama de componentes.*

El UML unido a un gestión de calidad, evita malos entendidos y entrega ciertas precauciones en la evolución y mantención de programas. Especialmente en lo referente a los requerimientos asociados al levantamiento y diseño funcional de un sistema”. (15)

1.3.3 Herramienta CASE (Visual Paradigm v8.0)

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. *“Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. También proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. Presenta licencia gratuita y comercial. Es fácil de instalar y actualizar y compatible entre versiones”.* (16)

1.3.4 Framework RAP v3.0

Provee una caja de herramientas integrada con tecnologías probadas como la Open Services Gateway Initiative (Iniciativa de Pasarela de Servicios Abiertos (OSGI)). *“Permite realizar una aplicación completamente en java, reutilizar código y beneficiarse de herramientas de entornos de desarrollo. RAP puede ejecutarse en cualquier navegador relevante sin requerir ningún add-on. Brinda la posibilidad de realizar aplicaciones de forma modular, las cuales pueden consumir y brindar servicios hacia otros módulos sin afectar su desarrollo. Posee una herramienta para el diseño de las interfaces de usuario y a su vez una herramienta para realizar las pruebas de aceptación”.* (17)

1.3.5 Lenguajes de desarrollo (Java v1.7)

Java es un lenguaje de programación simple, orientado a objetos, robusto, seguro, de hilos múltiples y dinámicos. El lenguaje en sí mismo toma mucha de la sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel como punteros. *“Los programas escritos en el lenguaje Java pueden ejecutarse en cualquier tipo de hardware. Dentro de sus principales ventajas se encuentra la de ser multiplataforma. Con Java se pueden programar páginas web dinámicas, con accesos a bases de datos”.* (18)

1.3.6 Entorno de desarrollo integrado (Eclipse Kepler Release v4.3)

Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés), o plataforma de integración de herramientas diseñada para ser extendida de forma indefinida a través de plug-ins. *“Dicha plataforma no tiene un lenguaje específico de desarrollo, sino que es un IDE genérico que puede utilizar disímiles lenguajes de programación para la implementación de las aplicaciones. Es un programa que brinda poderosas herramientas para la implementación de aplicaciones modulares, incluyendo además el lenguaje Java usando el plug-in JDT, en la distribución estándar del IDE”.* (19)

Características principales:

- Perspectivas, editores y vistas

- Gestión de proyectos
- Depurador de código
- Extensa colección de plug-ins como:
 - Data Tools Platform
 - Eclipse Git Team Provider
 - Eclipse Java Development Tools
 - Eclipse Java EE Developer Tools
 - JavaScript Development Tools
 - Maven Integration for Eclipse
 - Mylyn Task List
 - Eclipse Plug-in Development Environment
 - Remote System Explorer
 - Eclipse XML Editors and Tools (20)

1.3.7 Gestor de base de datos (Postgres v9.3.5)

Es un sistema de gestión de bases de datos de código abierto que utiliza el modelo cliente/servidor y usa multiprocesos en vez de multi-hilos para garantizar la estabilidad del sistema. *“Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia Berkeley Software Distribution (BSD) y con su código fuente disponible libremente”.* (21)

Algunas características:

- Copias de seguridad en caliente (Online/hot backups)
- Unicode
- Juegos de caracteres internacionales
- Regionalización por columna
- Multi-Version Concurrency Control (MVCC)
- Múltiples métodos de autenticación
- Acceso encriptado vía SSL
- Completa documentación (21)

1.3.8 Servidor de aplicaciones (Virgo v3.6.3)

Servidor de aplicaciones basado en OSGi, originalmente desarrollado por Spring DN Server y actualmente mantenido por la Fundación Eclipse. Virgo soporta el despliegue de los bundles¹ de OSGi y no modifica el funcionamiento de las aplicaciones web, además de que permite compartir librerías y servicios.

Características principales:

- Consola de administración – permite implantar y gestionar los artefactos, examinar los paquetes, ya sea en el sistema en vivo o de un error de resolución.
- Extensión de la consola de Equinox – permite gestionar Virgo y los artefactos desplegados.
- Planes – permite definir los artefactos que componen una aplicación, opcionalmente haciendo la aplicación atómica para empatar el artefacto y ciclos de vida, y con ámbito de aislar a la aplicación de otras aplicaciones.
- Aprovisionamiento – suministra automáticamente las dependencias de una aplicación que incluye paquetes, planes, el plan de archivos (PAR) y configuraciones, tanto de los repositorios locales y remotos.
- Contenedor web – soporta archivos WAR con todas sus dependencias en WEB-INF / lib, y Web Application Bundles, que importan sus dependencias a través de OSGi.
- Región de usuario – aísla el núcleo de las aplicaciones instaladas por el usuario y permite a los administradores centrarse en artefactos de aplicaciones y sus dependencias sin ver las del núcleo.
- Despliegue en caliente – desplegar artefactos a Virgo copiándolos en el directorio de recogida, ya sea en el archivo o de forma despiezada, como alternativa a la implementación a través de la consola de administración. (22)

1.3.9 Framework de desarrollo Equinox (OSGi v3.8.2)

“Es la referencia de implementación para las especificaciones del marco de trabajo OSGi. Es un framework que le permite a los desarrolladores implementar una aplicación como un conjunto de bundles utilizando una infraestructura de servicios común”. (23)

1.3.10 Spring Framework v3.1

“Spring Framework proporciona un modelo de programación y configuración completa para las aplicaciones empresariales modernas basadas en Java en cualquier tipo de plataforma de despliegue. Un elemento clave de Spring es el apoyo de infraestructura a nivel de aplicación: se

¹ Archivos que contienen datos locales específicos.

centra en la "fontanería" de las aplicaciones empresariales para que los equipos puedan centrarse en la lógica de negocios a nivel de aplicación, sin ataduras innecesarias a los entornos de despliegue específicos Spring DM se utiliza para realizar la inyección de dependencias. OSGI Service Platform determina una arquitectura común para proveedores de servicios, desarrolladores, para desarrollar, desplegar y trabajar con servicios de manera coordinada". (25)

Características:

- Inyección de dependencias.
- Programación orientada a aspectos incluyendo la gestión de transacciones declarativas de Spring.
- Apoyo fundacional para JDBC, JPA, JMS.
- Aplicación web Spring MVC y un amplio marco de servicios web.
- Capacidad de desplegar varias versiones de un mismo módulo de manera concurrente.
- Instalación, eliminación y actualización dinámica en el entorno de ejecución.
- Búsqueda y utilización de servicios ofrecidos por otros módulos desplegados en el sistema.

Conclusiones del capítulo

En el presente capítulo, se profundizó en el conocimiento de los conceptos fundamentales relacionados con los procesos de préstamo de materiales bibliográficos y los sistemas integrados de gestión bibliotecaria, necesarios para la obtención de la fundamentación teórica y comprensión de la presente investigación. Se realizó un estudio detallado de los sistemas integrados de gestión bibliotecaria, enfocados principalmente en los procesos de préstamo de materiales bibliográficos, determinando así similitudes y diferencias entre los mismos. Se efectuó un análisis de la metodología de desarrollo de software, herramientas y tecnologías propuestas por el grupo de arquitectura del proyecto ABCD para desarrollar el módulo.

CAPÍTULO 2: CARACTERÍSTICAS Y DISEÑO DE LA SOLUCIÓN

Introducción

En el presente capítulo se muestra el diseño de la solución propuesta presentando todos los artefactos generados, dígame diagramas de clase de diseño, diagramas de secuencia, diagrama de componentes y diagrama entidad-relación, para obtener una mejor comprensión de las relaciones que se establecen entre cada uno de los elementos del negocio. También se puntualizan las características fundamentales de dicho módulo, realizando una breve descripción de los casos de uso que representan cada una de las funcionalidades definidas.

Propuesta de solución.

2.1 Reglas del Negocio.

En el Módulo Circulación del sistema ABCD 3.0, se gestiona el proceso de préstamo de materiales bibliográficos.

- El sistema permite consultar el estado actual e histórico de los usuarios de préstamo registrados, permitiendo visualizar los datos de las transacciones, sanciones y reservaciones asociadas a los mismos. La misma acción puede realizarse con los objetos de préstamo, pero solamente consultando los datos de las sanciones, transacciones y reservaciones históricas asociadas.
- Los operarios del sistema pueden efectuar préstamos sobre los materiales que se encuentren disponibles en la institución bibliotecaria a partir de las reservaciones realizadas por los usuarios o mediante la presentación directa de la persona en la institución sin una reservación previa.
- Permite realizar renovaciones y devoluciones sobre los préstamos realizados anteriormente.
- Los préstamos de los materiales bibliográficos serán realizados verificando las reglas de circulación definidas por la institución bibliotecaria.
- Los usuarios pueden ser sancionados con multas o suspensiones, especificando el motivo de la misma e indicando el monto a pagar, en el caso de que la sanción sea de tipo “Multas”.

2.1.1 Actores del Sistema

Tabla 2: Actores del Sistema

Actor	Objetivo
Gestor de Transacciones	Se encarga de gestionar las transacciones del sistema. Puede asociar objetos y usuarios de préstamo a las transacciones,

	además de listar las transacciones históricas y actuales de los usuarios y objetos de préstamo.
Consultor de Transacciones	Puede consultar los datos de las transacciones que coincidan con los criterios de búsqueda introducidos.
Gestor de Usuario de Préstamo²	Puede registrar usuarios de préstamo en el sistema asociándole primeramente una persona. Además de modificar, editar o eliminar los usuarios registrados. Puede visualizar los datos del estado actual de los usuarios listando sus transacciones, reservas y sanciones actuales. Puede visualizar los datos del historial de los usuarios listando sus transacciones, sanciones y reservas históricas.
Consultor de Usuario de Préstamo	Puede consultar los datos de los usuarios de préstamo que coincidan con los criterios de búsqueda introducidos.
Consultor de Objetos de Préstamo	Puede consultar los datos de los objetos de préstamo que coincidan con los criterios de búsqueda introducidos.
Gestor de Sanciones	Puede gestionar las sanciones del sistema, asociando primero el usuario de préstamo que incurrió. Además de modificar, editar o eliminar las sanciones registradas en el sistema.
Consultor de Sanciones	Puede consultar los datos de las sanciones que coincidan con los criterios de búsqueda introducidos.
Consultor de Reservas	Puede consultar los datos de las reservas que coincidan con los criterios de búsqueda introducidos.
Gestor de Acceso	Se encarga de gestionar el acceso de los usuarios al sistema.

2.1.2 Descripción de casos de uso

Se realiza una breve descripción de los casos de los casos de uso definidos en el documento de Especificación de Casos de Uso del Módulo Circulación (27) para entender el comportamiento de cada una de las funcionales a desarrollar.

Tabla 3: Descripción de Casos de Uso

Caso de Uso	Descripción
-------------	-------------

² Usuarios del sistema a los cuales se le realizan préstamos de los medios existentes en la institución.

Capítulo 2: Características y Diseño de la Solución

Gestionar Usuario de Préstamo	Permite registrar, visualizar, eliminar o editar un Usuario de Préstamo.
Consultar Usuarios de Préstamos	Permite consultar los datos de los usuarios de préstamo que coinciden con los criterios de búsqueda.
Asociar Usuarios de Préstamo	Permite buscar y asociar un Usuario de Préstamo.
Mostrar Estado Actual³ de un Usuario	Permite visualizar las Transacciones ⁴ , Reservaciones y Sanciones activas del Usuario de Préstamo.
Mostrar Historial⁵ de un Usuario	Permite visualizar las Transacciones, Reservaciones y Sanciones históricas del Usuario de Préstamo.
Ver Objeto de Préstamos⁶	Permite visualizar los datos del Objeto de Préstamos.
Consultar Objetos de Préstamos	Permite consultar listado de Objetos de Préstamos que coinciden con los criterios de búsqueda.
Asociar Objeto de Préstamos	Permite buscar y asociar objetos de préstamo.
Mostrar Historial del Objeto de Préstamos	Permite visualizar las Transacciones, Reservaciones y Sanciones históricas asociadas al Objeto de Préstamo.
Gestionar Transacciones	Permite prestar, renovar, devolver, eliminar y visualizar los datos de la Transacción.
Consultar Transacciones	Permite consultar el listado de las transacciones que coinciden con los criterios de búsqueda.
Listar Transacciones actuales de un usuario	Permite listar las Transacciones activas de un Usuario de Préstamo.
Listar Transacciones históricas de un usuario	Permite listar Transacciones históricas de un Usuario de Préstamo.
Listar Transacciones históricas del Objeto de Préstamo	Permite listar las Transacciones históricas del Objeto de Préstamo.
Consultar Reservación	Permite consultar el listado de las reservaciones que coinciden con los criterios de búsqueda.
Listar Reservaciones Actuales de un Usuario	Permite listar las Reservaciones Actuales de un Usuario.

³ Se refiere a las transacciones, sanciones y reservaciones que están activas. No han rebasado el rango de fecha establecido.

⁴ Préstamos de los medios disponibles en la institución.

⁵ Se refiere a las transacciones, sanciones y reservaciones que están inactivas. Han rebasado el rango de fecha establecido.

⁶ Se refiere a los medios disponibles en la institución, sobre los cuales se realizan los préstamos.

Listar Reservaciones Históricas de un Usuario	Permite listar las Reservaciones Históricas de un Usuario.
Gestionar Sanción	Permite registrar, visualizar, editar y eliminar las Sanciones y registrar el pago de Sanciones de tipo: "Multa".
Consultar Sanciones	Permite consultar los datos de las sanciones que coinciden con los criterios de búsqueda.
Listar Sanciones actuales del usuario	Permite listar las Sanciones actuales del usuario.
Listar Sanciones Históricas del Usuario	Permite listar Sanciones Históricas del Usuario
Listar Sanciones Históricas asociada a un Objeto de Préstamo	Permite listar las sanciones históricas asociadas a un objeto de préstamo seleccionado.

2.2 Diseño de la solución

A continuación se muestran los elementos referentes al diseño de la propuesta de solución, específicamente se muestra el diagrama entidad-relación, donde se definen cada una de las relaciones establecidas entre las clases que componen el módulo. Luego se presenta el diagrama de clases del diseño donde se muestra la interacción entre las clase definidas, evidenciando además los patrones arquitectónicos y de diseño utilizados. Se presentan además diagramas de secuencia, donde se muestran paso a paso las interacciones entre el usuario y el sistema en el momento de realizar los procesos definidos.

2.2.1 Diagrama Entidad Relación

Un diagrama Entidad-Relación es un artefacto generado donde se representan las entidades relevantes de un sistema de información así como sus interrelaciones y propiedades. A continuación se muestra el diagrama entidad relación perteneciente al módulo propuesto, evidenciando las relaciones entre cada una de las clases que lo componen, y una breve descripción de las principales entidades del sistema.

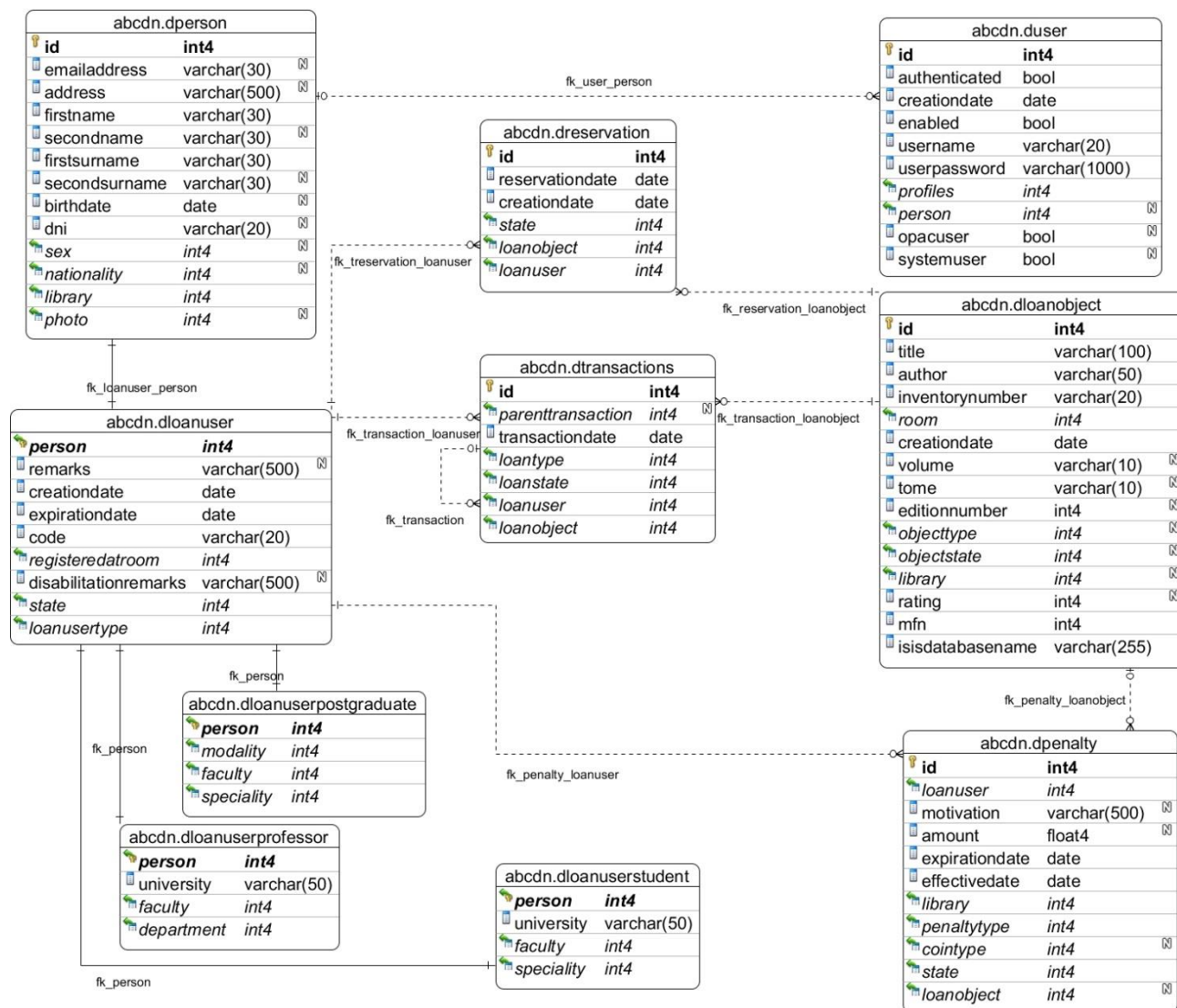


Figura 2: Diagrama Entidad-Relación del Módulo Circulación del Sistema ABCD 3.0

- La entidad **nomencladores** se utiliza para identificar el tipo de moneda, tipo de sanción, tipo de usuario de préstamo, tipo de objeto de préstamo, estado de usuario del sistema y estado de la sanción. Estos nomencladores se encuentran divididos en dos grupos: administrables y no administrables. Los administrables son los nomencladores a los cuales se les puede agregar o eliminar más opciones; como el tipo de moneda. Los no administrables son los que se encuentran predefinidos y no sufren cambios; ejemplo de este tipo de nomenclador es el tipo de sanción, que puede ser “Multa” o “Suspensión”. No es parte del alcance de la investigación, pero dicha clase incide directamente sobre las entidades usuario de préstamo, sanción, transacción y objeto de préstamo.

Capítulo 2: Características y Diseño de la Solución

- Un **objeto de préstamo** puede estar asociado a varias reservaciones, transacciones y sanciones.
- Una **reservación** está asociada a un usuario de préstamo y a un objeto de préstamo.
- Un **usuario de préstamo** es cualquier persona del sistema y puede relacionarse con varias reservaciones, sanciones y transacciones.
- Una **transacción** está asociada a un usuario de préstamo y a uno o varios objetos de préstamo. Estas pasan a ser transacciones históricas desde el momento en que expira la fecha límite de la transacción, o actuales si no se ha cumplido esta condición.
- Una **sanción** está asociada a un usuario de préstamo y puede o no estar relacionada con un objeto de préstamo. Estas pasan a ser históricas desde el momento que expira la fecha límite de la sanción o actuales si no se ha cumplido esta condición.

2.2.2 Diagrama de Clase del Diseño

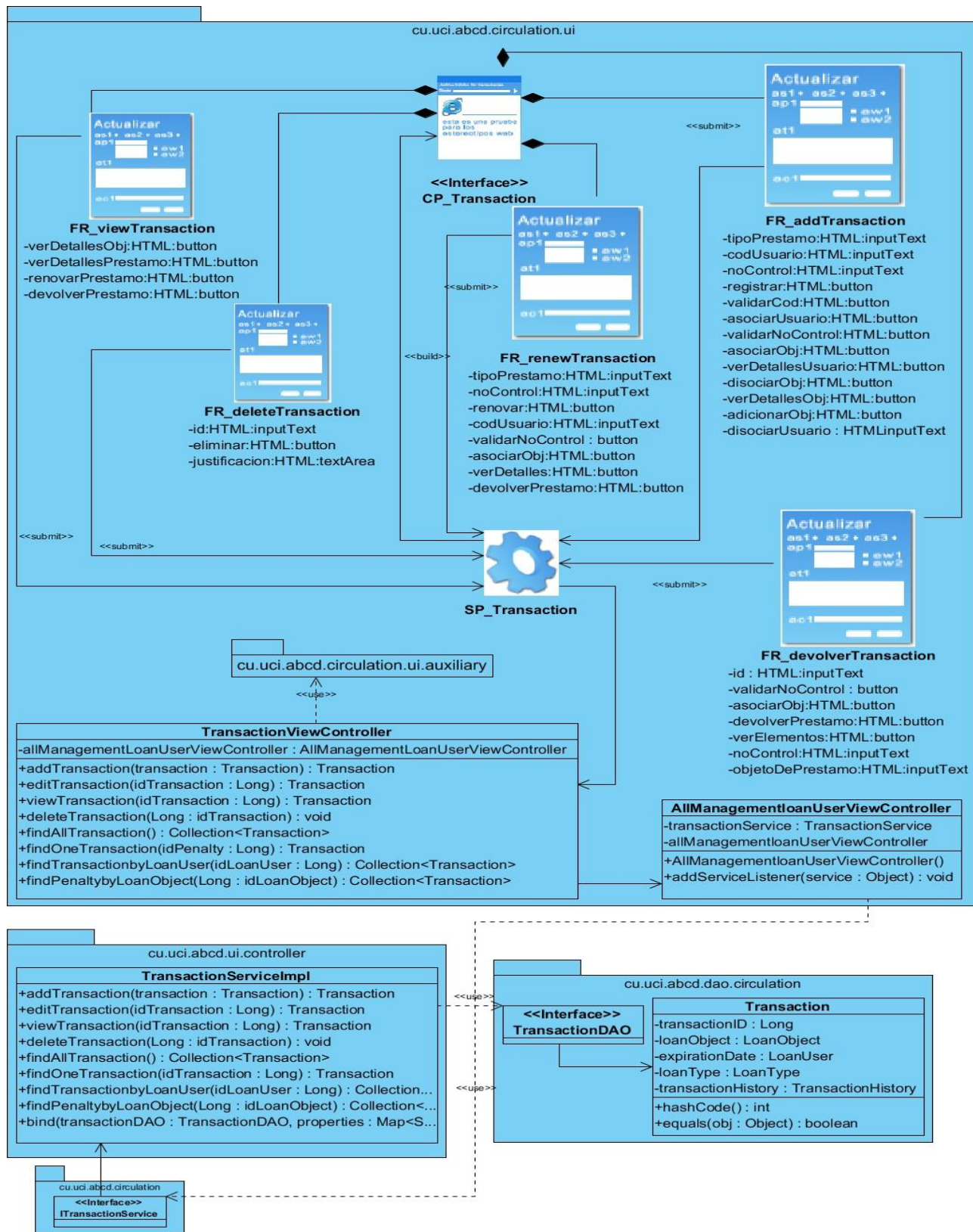


Figura 3: Diagrama de clase del diseño (Gestionar Transacción)

Capítulo 2: Características y Diseño de la Solución

En la figura anterior se muestra el diagrama de clase del diseño correspondiente al caso de uso gestionar transacción, donde se reflejan cada una de las capas que componen la arquitectura del sistema y la relación entre los elementos que componen cada capa. ([Ver Anexos 1: Diagramas de clase del diseño](#))

2.2.3 Diagrama de Secuencia

El diagrama de secuencia muestra paso a paso las interacciones del usuario con el sistema, además de presentar las respuestas del mismo a cada petición del usuario. También, se muestran las interrelaciones entre las clases definidas. “Muestran la secuencia de mensajes durante un escenario concreto”.

- *En la parte superior se muestran los objetos que intervienen.*
- *La dimensión temporal se indica verticalmente.*
- *Las líneas verticales indican el período de vida de cada objeto.*
- *El paso de mensajes se indica con flechas horizontales u oblicuas.*
- *La realización de una acción se indica con rectángulos sobre las líneas de actividad del objeto que realiza la acción. (28)*

En este caso se muestra el diagrama de secuencia correspondiente al proceso de adicionar una transacción. Como primer paso, el usuario selecciona la opción que le permite adicionar una transacción al sistema, luego introduce los datos del usuario de préstamo relacionado y después de ser validados los datos por el sistema, lo asocia a la transacción. Posteriormente, el usuario asocia uno o más objetos de préstamo e introduce los datos de la transacción.

Los datos son validados, verificando que se hayan introducido correctamente. En el caso de no realizarse correctamente esta operación, el sistema muestra un mensaje indicando la existencia de campos vacíos o incorrectos; si esto no sucede, se procede a la persistencia de la información en la base de datos y luego, se le indica al usuario que la transacción se ha adicionado exitosamente.

([Ver Anexos 2: Diagramas de secuencia](#))

Capítulo 2: Características y Diseño de la Solución

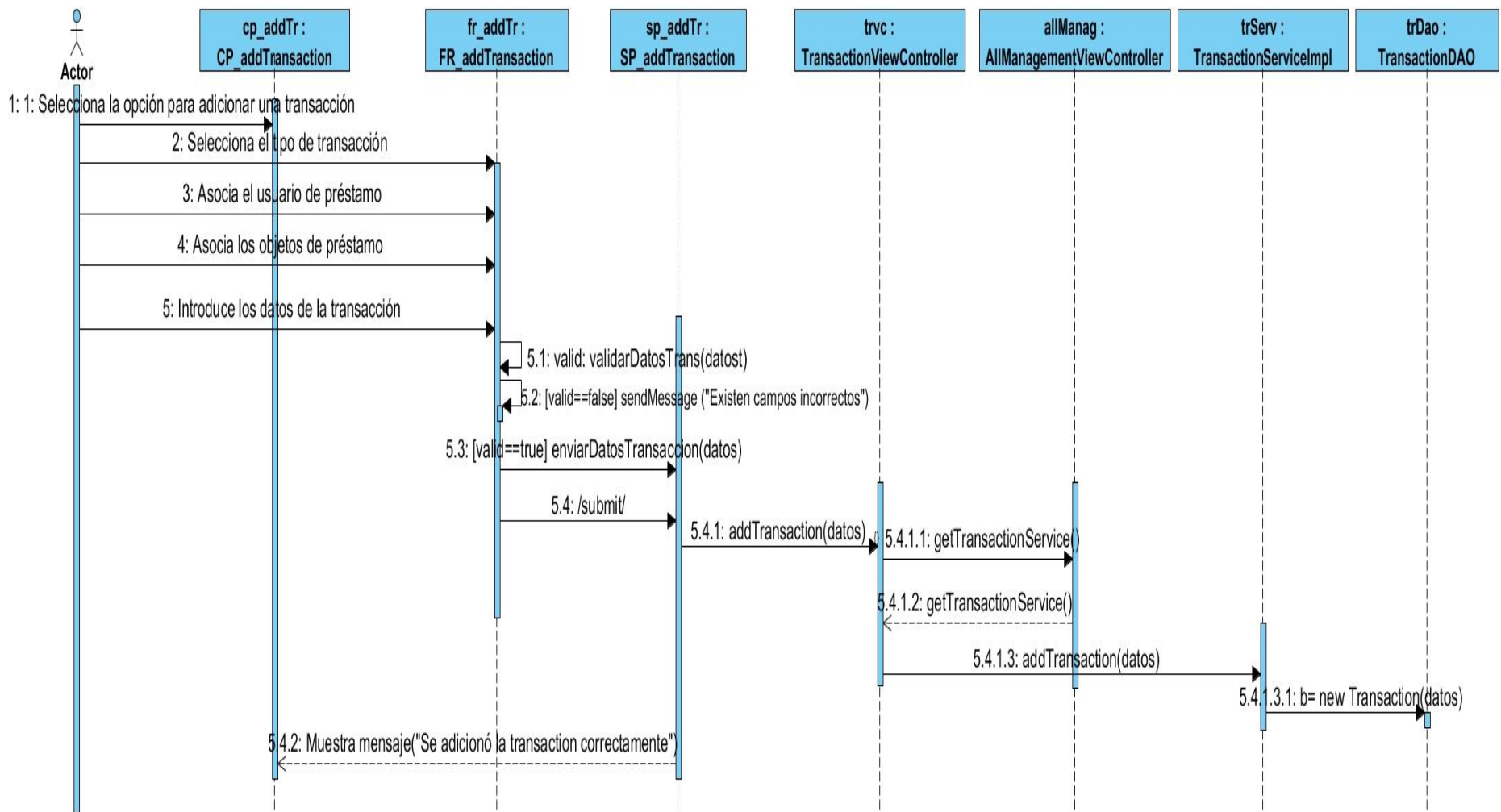


Figura 4: Diagrama de Secuencia correspondiente a Adicionar Transacción

2.2.4 Patrón de arquitectura.

Arquitectura n Capas

El estilo arquitectural en capas se basa en una distribución jerárquica de los roles y las responsabilidades para proporcionar una división efectiva de los problemas a resolver. Los roles indican el tipo y la forma de la interacción con otras capas y las responsabilidades la funcionalidad que implementan. (29)

Esta arquitectura está conformada por una capa de presentación, que contiene las interfaces donde se muestran los datos a los usuarios, además de los servicios y funcionalidades que brinda la aplicación. En la capa de aplicación, se encuentran distribuidos los bundles y componentes. En esta capa es donde se realizan las validaciones y se establecen las reglas del negocio. Como tercera capa que forma parte de la arquitectura, se encuentra la capa de datos, encargada de realizar la persistencia de los datos utilizando el sistema gestor de bases de datos (SGBD): PostgreSQL 9.3.5. Por otra parte, se cuenta con una capa transversal, encargada de la implementación de la seguridad y el control de acceso.

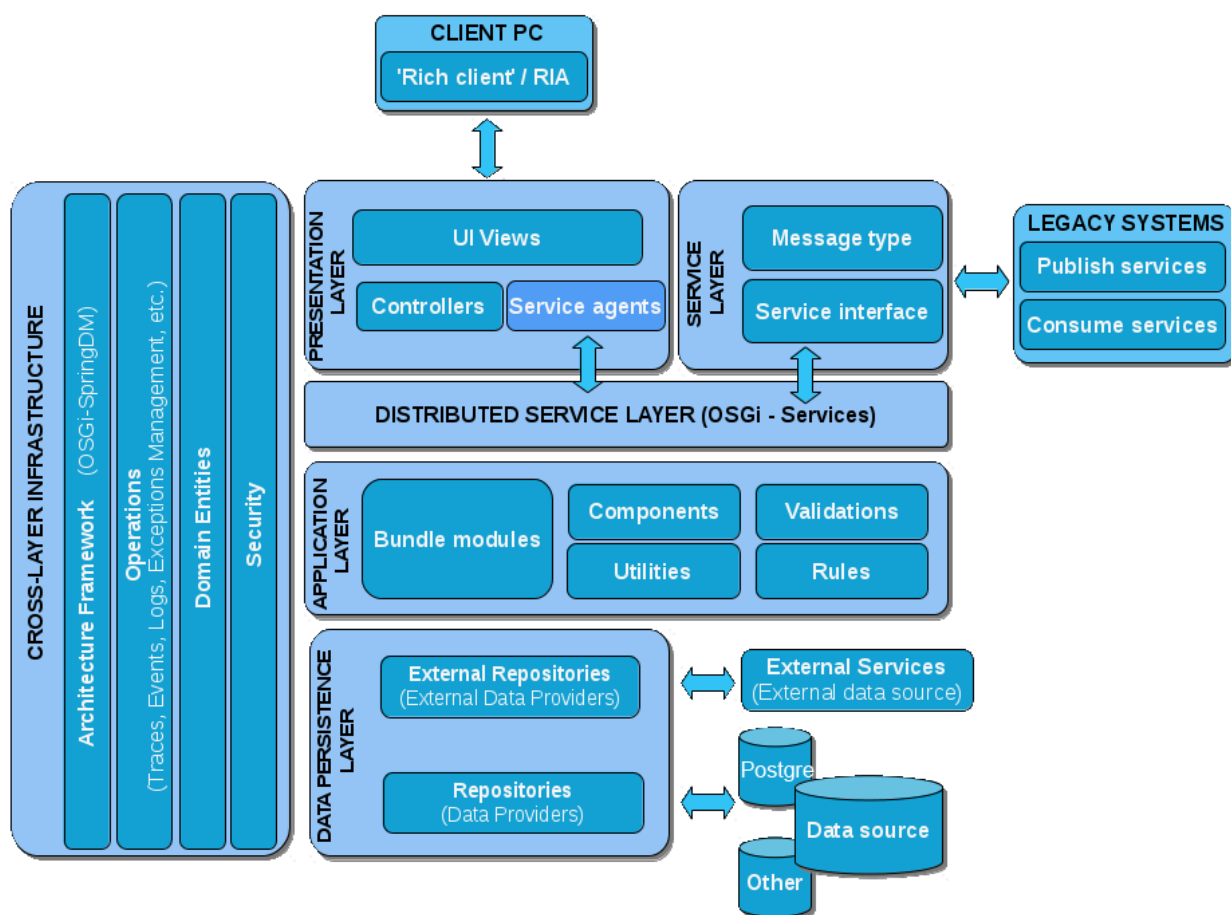


Figura 5: Estructura de la arquitectura n capas (30)

Características

- Descomposición de los servicios de forma que la mayoría de interacciones ocurre solo entre capas vecinas.
- Las capas de una aplicación pueden residir en la misma máquina o pueden estar distribuidos entre varios equipos.
- Los componentes de cada capa se comunican con los otros componentes a través de interfaces bien conocidas.
- Cada nivel agrega las responsabilidades y abstracciones del nivel inferior. (29)

Principios clave

- Muestra una vista completa del modelo y a la vez proporciona suficientes detalles para entender las relaciones entre capas.

Capítulo 2: Características y Diseño de la Solución

- No realiza ninguna suposición sobre los tipos de datos, métodos, propiedades y sus implementaciones.
- Separa de forma clara la funcionalidad de cada capa.
- Cada capa contiene la funcionalidad relacionada solo con las tareas de esa capa.
- Las capas inferiores no tienen dependencias de las capas superiores.
- La comunicación entre capas está basada en una abstracción que proporciona un bajo acoplamiento entre capas. (29)

Beneficios

- Abstracción ya que los cambios se realizan a alto nivel y se puede incrementar o reducir el nivel de abstracción que se usa en cada capa del modelo.
- Aislamiento ya que se pueden realizar actualizaciones en el interior de las capas sin que esto afecte al resto del sistema.
- Rendimiento ya que distribuyendo las capas en distintos niveles físicos se puede mejorar la escalabilidad, la tolerancia a fallos y el rendimiento.
- Testeabilidad ya que cada capa tiene una interfaz bien definida sobre la que realizar las pruebas y la habilidad de cambiar entre diferentes implementaciones de una capa.
- Independencia ya que elimina la necesidad de considerar el hardware y el despliegue así como las dependencias con interfaces externas. (29)

Capa de presentación: es la interfaz donde se le muestran los datos al usuario (también se la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio. (31)

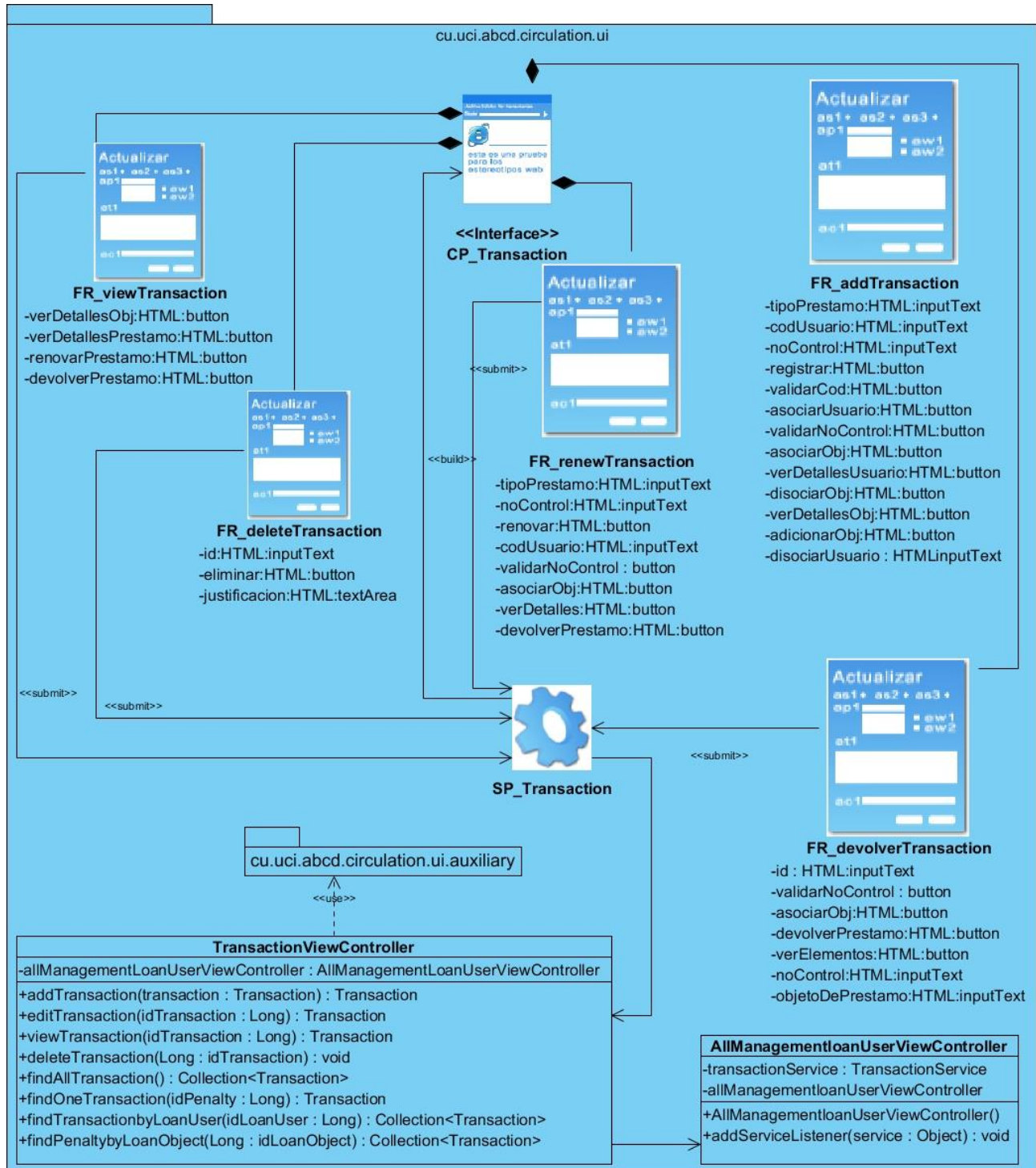


Figura 6: Ejemplo de la utilización de la arquitectura n capas (Capa de presentación del diagrama de clase del diseño Gestionar Transacción)

Capa de negocio: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta

capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación. (31)

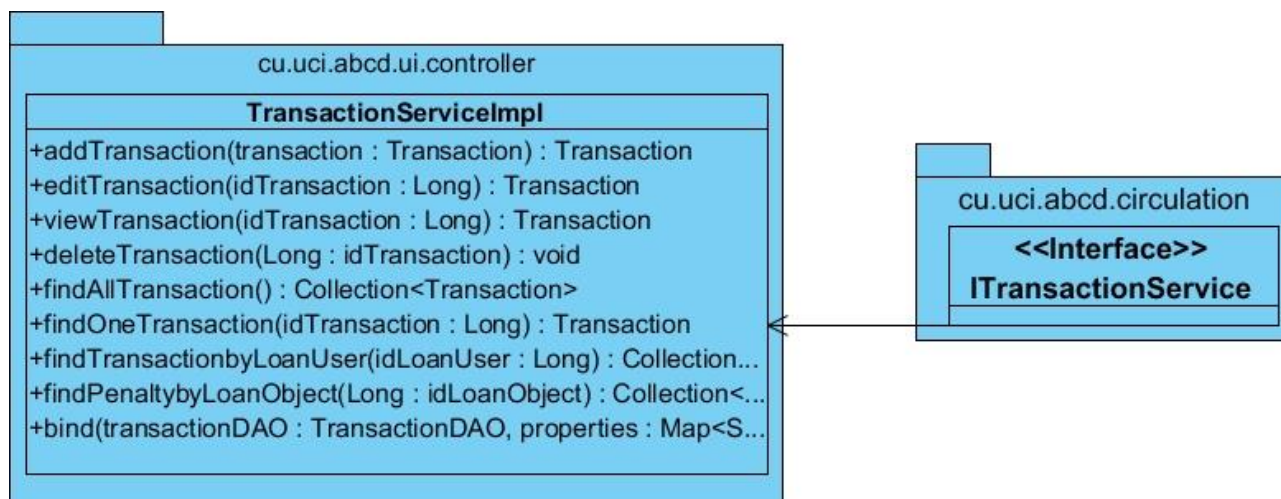


Figura 7: Ejemplo de la utilización de la arquitectura n capas (Capa de negocio del diagrama de clase del diseño Gestionar Transacción)

Capa de datos: es donde residen los datos y es la encargada de acceder a los mismos.

Está formada por un gestor de base de datos que realiza todo el almacenamiento de datos, recibe solicitudes de almacenamiento o recuperación de información desde la capa de negocio. Todas estas capas pueden residir en un único computador, si bien lo más usual es que haya una multitud de computadoras en donde reside la capa de presentación (son los clientes de la arquitectura cliente/servidor). Las capas de negocio y de datos pueden residir en el mismo computador, y si el crecimiento de las necesidades lo aconseja se pueden separar en dos o más computadoras. Así, si el tamaño o complejidad de la base de datos aumenta, se puede separar en varias computadoras los cuales recibirán las peticiones del computador en que resida la capa de negocio. Si, por el contrario, fuese la complejidad en la capa de negocio lo que obligase a la separación, esta capa de negocio podría residir en uno o más computadores que realizarían solicitudes a una única base de datos. En sistemas muy complejos se llega a tener una serie de computadores sobre los cuales corre la capa de negocio, y otra serie de computadores sobre los cuales corre la base de datos. (31)

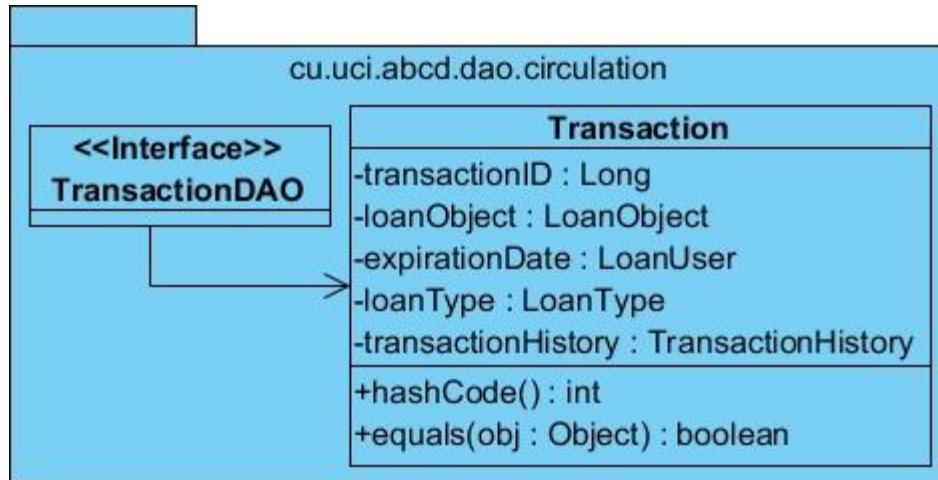


Figura 8: Ejemplo de la utilización de la arquitectura n capas (Capa de datos del diagrama de clase del diseño Gestionar Transacción)

2.2.5 Patrones de Diseño utilizados.

Patrones generales de software para asignación de responsabilidades (GRASP)

- Experto

Asignar una responsabilidad al experto en información; la clase que tiene la información necesaria para llevar a cabo la responsabilidad. (32)

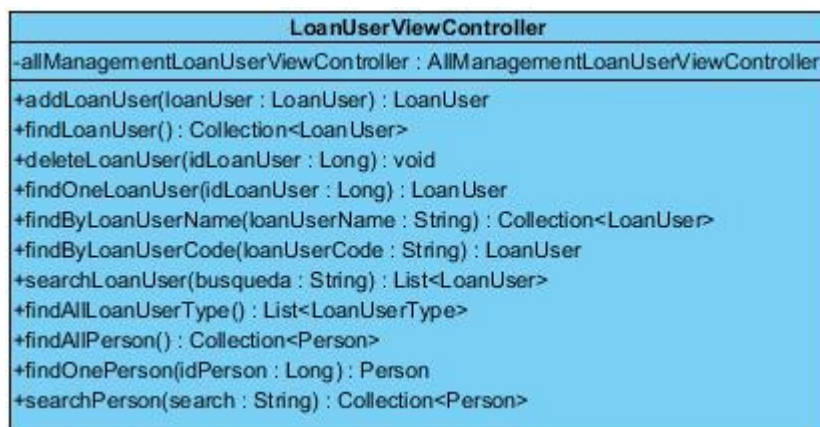


Figura 9: Ejemplo de utilización del patrón experto

- Creador

Crear una nueva instancia por la clase que:

- Tiene la información necesaria para realizar la creación del objeto.
- Usa directamente las instancias creadas del objeto.
- Almacena o maneja varias instancias de la clase.
- Contiene o agrega la clase. (32)

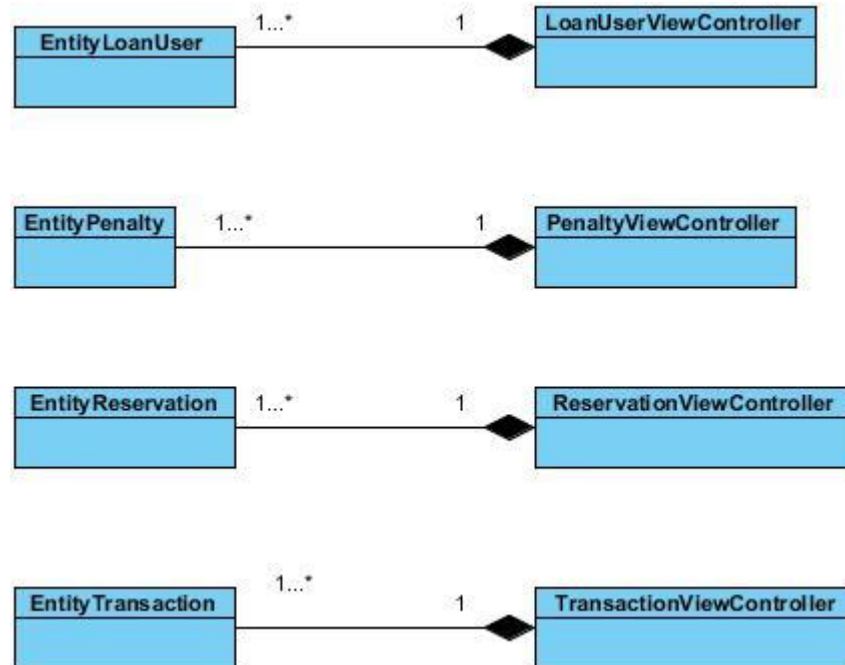


Figura 10: Ejemplo de utilización del patrón creador

- Bajo Acoplamiento

Diseñar con el objetivo de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. (32)

- Alta Cohesión

Asignar responsabilidades de manera que la información que almacena una clase sea coherente y esté relacionada con la clase. (32)

The gang of four (GOF)

- Proxy

Se basa en la idea de un proxy web, que proporciona recursos. Permite tener más de un controlador en una misma interfaz. (33)

Conclusiones de capítulo

En el presente capítulo de la investigación se expusieron los elementos que componen el diseño de la solución mediante la presentación de los artefactos generados a lo largo de la investigación, para lograr un mejor entendimiento de las relaciones que se establecen entre las clases definidas para la implementación del módulo. Se exhibieron elementos que ejemplifican la interacción entre el usuario

Capítulo 2: Características y Diseño de la Solución

y el sistema en el momento de realizar los procesos definidos, y además quedaron evidenciados los patrones que se utilizaron para la implementación del módulo.

Capítulo 3: Implementación y prueba de la solución

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA DE LA SOLUCIÓN

Introducción

En el presente capítulo se abordan todos los elementos referentes a la etapa de implementación de la solución propuesta; definiendo primeramente cada una de las clases de dominio, clases controladoras, clases de servicio y clases de acceso a datos correspondientes al módulo en cuestión. Luego se presentan los diagramas de componentes del sistema y de despliegue, además de los diferentes casos de prueba que han sido verificados como parte de las pruebas de software para la validación del módulo. Se utilizó específicamente la **técnica de partición de equivalencia** correspondiente a las **pruebas de caja negra**.

3.1 Implementación

A continuación se presentan cada una de las clases definidas en el sistema, incluyendo una breve descripción del objetivo de las mismas.

3.1.1 Clases de dominio

EntityLoanObject.java: Clase que contiene todos los datos referentes a los objetos de préstamo.

EntityLoanUser.java: Clase entidad que contiene los datos de los usuarios de préstamo.

EntityPenalty.java: Clase que contiene los datos referentes a las sanciones.

EntityReservation.java: Clase que contiene los datos referentes a las reservaciones.

EntityTransaction.java: Clase que contiene todos los datos referentes a las transacciones.

3.1.2 Clases controladoras

AllManagementLoanUserController.java: Clase que se encarga de controlar todos los objetos (proxy).

LoanObjectViewController.java: Clase que se encarga de controlar todos los objetos de tipo objeto de préstamo.

LoanUserController.java: Clase que se encarga de controlar todos los objetos de tipo usuario de préstamo.

PenaltyViewController.java: Clase que se encarga de controlar todos los objetos de tipo sanción.

ReservationViewController.java: Clase que se encarga de controlar todos los objetos de tipo reservación.

Capítulo 3: Implementación y prueba de la solución

TransactionViewController.java: Clase que se encarga de controlar todos los objetos de tipo transacción.

3.1.3. Clases servicio

ILoanObjectService.java: Clase que se encarga de brindar los servicios referentes a los objetos de préstamo.

ILoanUserService.java: Clase que se encarga de brindar los servicios referentes a los usuarios de préstamo.

IPenaltyService.java: Clase que se encarga de brindar los servicios referentes a las sanciones.

IPersonService.java: Clase que se encarga de brindar los servicios referentes a las personas.

IReservationService.java: Clase que se encarga de brindar los servicios referentes a las reservaciones.

ITransactionService.java: Clase que se encarga de brindar los servicios referentes a las transacciones.

3.1.4 Clases de acceso a datos

LoanUserDAO.java: Clase encargada de acceder a los datos relacionados con los usuarios de préstamo en la base de datos.

LoanUserTypeDAO.java: Clase encargada de acceder a los datos relacionados con los tipos de usuarios de préstamo en la base de datos.

PenaltyDAO.java: Clase encargada de acceder a los datos relacionados con las sanciones en la base de datos.

ReservationDAO.java: Clase encargada de acceder a los datos relacionados con las reservaciones en la base de datos.

TransactionDAO.java: Clase encargada de acceder a los datos relacionados con las transacciones en la base de datos.

3.2 Estándares de codificación

Un estándar de codificación es un conjunto de reglas que guían a los desarrolladores con el fin de lograr escribir un código fuente que sea entendible, y así no presentar grandes problemas en el momento de brindarle mantenimiento al sistema. Estos estándares facilitan una mayor organización y limpieza en el código.

Capítulo 3: Implementación y prueba de la solución

A continuación se describen los estándares de codificación a utilizar para la implementación del módulo:

- Los nombres de las clases comienzan con la primera letra en mayúscula y el resto con minúscula, en caso de ser una palabra compuesta se empleará la notación PascalCasing. Ejemplo: Penalty.java, LoanUser.java.
- Los nombres de los métodos y los atributos de las clases comienzan con la primera letra en minúscula, en caso de que sea un nombre compuesto se empleará la notación CamelCasing. Ejemplo: addPenalty(), name.
- Para realizar los comentarios de código, los mismos estarán delimitados por “//” o “/*...*/”.
- El inicio y fin de los bloques será de dos espacios en blanco desde la instrucción anterior para el inicio y fin de bloque {}. Esto se utilizará también para el caso de las instrucciones **if**, **else**, **for**, **while**, **do while**, **switch**. No se debe utilizar el tabulador ya que puede variar según la configuración que tenga dicha tecla en la computadora. El inicio y cierre de ámbito deben estar alineados debajo de la declaración a la que pertenecen y deben evitarse si hay sólo una instrucción. “{“ “}”

Ejemplo:

```
for (LoanUser loanUser : list) {  
    EntityConsultLoanUser aux = new EntityConsultLoanUser(loanUser);  
    entityLoanUsersList.add(aux);  
}
```

Figura 11: Ejemplo de utilización de estándares de codificación (bloque for).

```
if (today.get(Calendar.MONTH) <= birth.get(Calendar.MONTH)) {  
    if (today.get(Calendar.MONTH) == birth.get(Calendar.MONTH)) {  
        if (today.get(Calendar.DATE) > birth.get(Calendar.DATE)) {  
            factor = -1; // Aun no celebra su cumpleaños  
        }  
    } else {  
        factor = -1; // Aun no celebra su cumpleaños  
    }  
}
```

Figura 12: Ejemplo de utilización de estándares de codificación (bloque if).

Capítulo 3: Implementación y prueba de la solución

3.3 Tratamiento de errores

Es inevitable contar con la presencia de errores durante la ejecución, pero los mismos pueden ser disminuidos mediante los procesos de gestión de errores, logrando aumentar así la calidad de los sistemas desarrollados. Una de las técnicas más utilizadas para el tratamiento de errores es el lanzamiento y captura de excepciones, implementando los bloques **try/catch** en los fragmentos de código necesarios.

```
try {
    Calendar birth = new GregorianCalendar();
    Calendar today = new GregorianCalendar();
    int age = 0;
    int factor = 0;
    Date birthDate = new SimpleDateFormat("dd-MM-yyyy").parse(datetext);
    Date currentDate = new Date(); // current date
    birth.setTime(birthDate);
    today.setTime(currentDate);
    if (today.get(Calendar.MONTH) <= birth.get(Calendar.MONTH)) {
        if (today.get(Calendar.MONTH) == birth.get(Calendar.MONTH)) {
            if (today.get(Calendar.DATE) > birth.get(Calendar.DATE)) {
                factor = -1; // Aun no celebra su cumpleaños
            }
        } else {
            factor = -1; // Aun no celebra su cumpleaños
        }
    }
    age = (today.get(Calendar.YEAR) - birth.get(Calendar.YEAR)) + factor;
    return age;
} catch (ParseException e) {
    return -1;
}
```

Figura 13: Ejemplo de utilización del bloque try/catch para el tratamiento de errores.

3.4 Diagramas de componentes

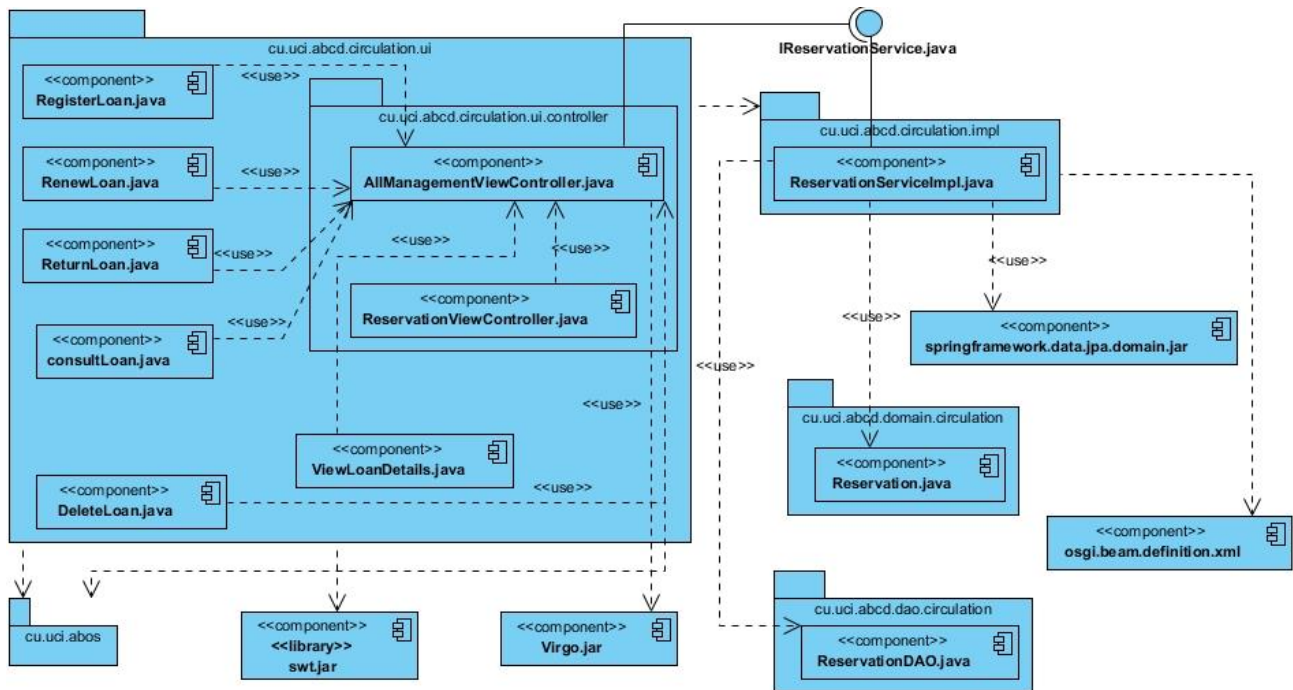


Figura 14: Diagrama de componente correspondiente al CU Gestionar Préstamo

En el presente diagrama de componente se refleja la integración de cada una de las herramientas utilizadas para la implementación de la solución propuesta. Para la presentación de los datos al usuario, los formularios y otros componentes se relacionan con la librería RAP, mientras que para realizar la persistencia de la información en la base de datos, se utiliza Spring Framework. El bundle que contiene la implementación de los servicios, depende directamente del bundle que contiene los elementos del dominio y a su vez utiliza la clase de acceso a datos para la persistencia o recuperación de los mismos en la base de datos.

3.5 Diagrama de despliegue

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Dicho diagrama muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos. (34) El módulo desarrollado responde al modelo cliente servidor y puede ser utilizada en cualquier sistema operativo; pero debe tener soporte de instalación de la Máquina Virtual de Java (JVM) con JRE 7.0, debe reconocer la codificación utf-8. Debe usarse una versión del navegador Mozilla/Firefox igual o inferior a la 30.0, y

Capítulo 3: Implementación y prueba de la solución

el navegador debe tener habilitado el soporte para Cookies y Java Script. La interfaz externa está diseñada para verse en una resolución igual a 1024x768. Se necesita un servidor de aplicaciones con Virgo Tomcat 3.6.3.

- PC Cliente (1.1GHz, 512MB RAM, Mozilla Firefox, Windows, Linux, Mac OS).
- ServidorApp_ABCD3.0 (3.0GHz, 16GB RAM, Mozilla Firefox, Windows, Linux, Mac OS, 1TB Almacenamiento).
- ServidorBD_Postgres (3.0GHz, 16GB RAM, Mozilla Firefox, Windows, Linux, Mac OS, 1TB Almacenamiento).

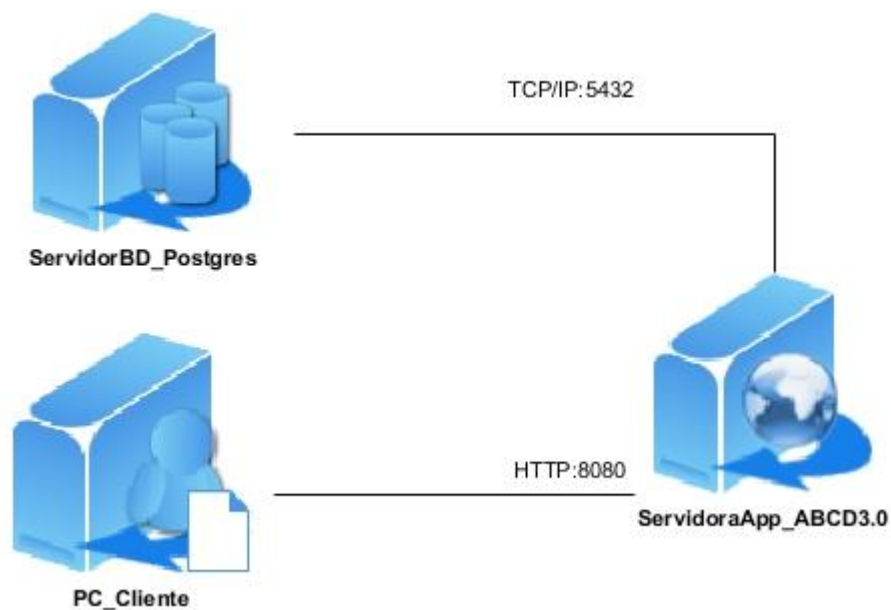


Figura 15: Diagrama de Despliegue del Sistema

3.6. Funcionalidades

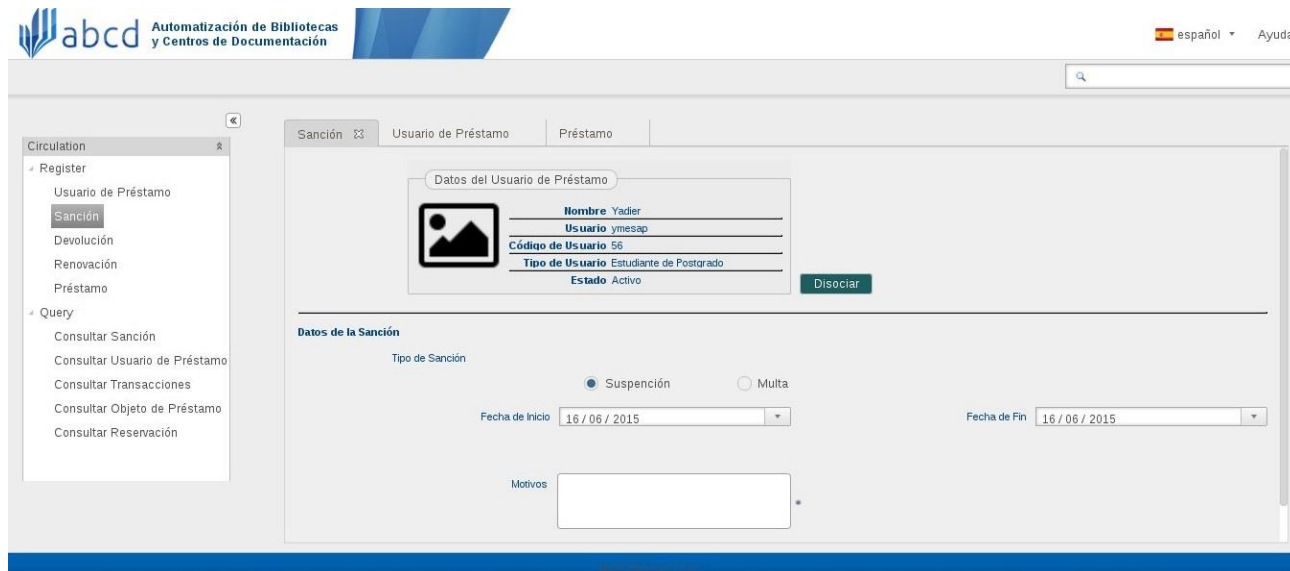
A continuación se muestran algunas de las interfaces del sistema y las funcionalidades asociadas a las mismas. ([Ver Anexos 3: Funcionalidades](#))

3.6.1 Registrar sanción

Esta funcionalidad permite registrar una sanción asociada a un usuario de préstamo, especificando los motivos por los cuales le ha sido impuesta dicha sanción. Puede ser una multa o suspensión,

Capítulo 3: Implementación y prueba de la solución

dependiendo de las características de la incidencia. Luego se introducen los datos pertinentes de la sanción dependiendo de si es una multa o suspensión.



The screenshot shows the ABCD web application interface. On the left is a navigation menu with categories like 'Circulation', 'Register', and 'Query'. The 'Sanción' option is highlighted. The main content area has three tabs: 'Sanción', 'Usuario de Préstamo', and 'Préstamo'. The 'Sanción' tab is active, displaying a form for recording a sanction. The form is divided into two sections: 'Datos del Usuario de Préstamo' and 'Datos de la Sanción'. The 'Datos del Usuario de Préstamo' section includes a profile picture placeholder and fields for 'Nombre' (Yadier), 'Usuario' (ymesap), 'Código de Usuario' (56), 'Tipo de Usuario' (Estudiante de Postgrado), and 'Estado' (Activo). A 'Disociar' button is located to the right of these fields. The 'Datos de la Sanción' section includes a 'Tipo de Sanción' section with radio buttons for 'Suspensión' (selected) and 'Multa'. Below this are 'Fecha de Inicio' and 'Fecha de Fin' dropdown menus, both set to '16 / 06 / 2015'. A 'Motivos' text area is at the bottom.

Figura 16: Interfaz correspondiente a la funcionalidad "Registrar Sanción"

3.6.2 Consultar usuario de préstamo

Luego de acceder a la opción que le permite consultar los usuarios de préstamo, el sistema muestra una interfaz que contiene un formulario que especifica los criterios de búsqueda. El usuario introduce y selecciona los criterios de búsqueda y se procede a mostrar los elementos que coinciden con dichos criterios.

Capítulo 3: Implementación y prueba de la solución

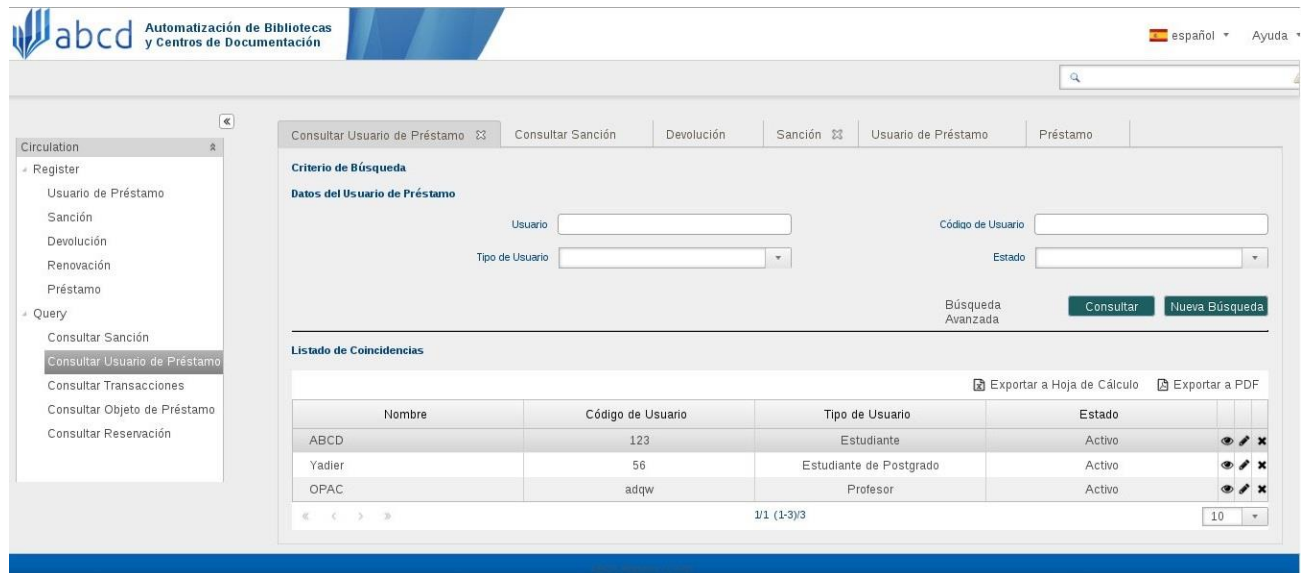


Figura 17: Interfaz correspondiente a la funcionalidad "Consultar usuario de préstamo"

3.7 Pruebas

Las pruebas son un elemento crítico para la calidad del software. La importancia de los costos asociados a los errores, promueve la definición y la aplicación de un proceso de pruebas minuciosas y bien planificadas. Las pruebas permiten validar y verificar el software, entendiendo como validación del software, el proceso, externo al equipo de desarrollo, que determina si el software satisface los requisitos, y verificación como el proceso interno que determina si los productos de un fase satisfacen las condiciones de dicha fase. (35)

El sistema puede probarse de dos formas: a) conociendo la función específica para la cual fue diseñado; y b) conociendo el funcionamiento del producto. El primer enfoque se centra en las llamadas pruebas de caja negra, mientras que el segundo se basa en las pruebas de caja blanca.

3.7.1 Tipo de prueba

En la presente investigación se realizan dos tipos de pruebas para determinar la calidad del software:

- Pruebas funcionales:

Se asegura el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. Este tipo de pruebas están basadas en técnicas de caja negra, que es, verificar la aplicación y sus procesos internos mediante la interacción con la aplicación y analizar los resultados. Se realiza específicamente la técnica de

Capítulo 3: Implementación y prueba de la solución

partición de equivalencia correspondiente a las pruebas de caja negra, verificando por casos de prueba el cumplimiento de todos los requisitos definidos para la implementación del módulo. (35)

- Pruebas de integración:

Las pruebas de integración se realizan para verificar que exista una correcta comunicación entre los módulos que componen el sistema final. Se utilizan para verificar que los componentes del sistema funcionan correctamente actuando en conjunto. En el caso de la investigación presentada, el objetivo es verificar la integración directa entre los módulos OPAC, Adquisición y Seguridad, permitiendo comprobar que el módulo Circulación pueda consumir correctamente los servicios necesarios para su funcionamiento.

El objetivo final de las pruebas de integración, es detectar errores de programación entre los componentes del sistema y errores de interoperabilidad. Uno de los errores de programación más frecuentes en estos casos son: el abrazo mortal o “deadlock”, impidiendo la ejecución de los procesos por no obtener las respuestas necesarias para continuar. Se utiliza la estrategia de prueba incremental, específicamente la integración ascendente, partiendo de los módulos que se encuentran jerárquicamente al final, utilizando controladores de prueba a fin de coordinar la entrada y la salida de los casos de prueba. (35)

- En el archivo **menu.xml** encontrado en el directorio **servidor\abcdconfig\menu\menu.xml**, se definen los identificadores que representan a las interfaces agrupadas en el menú. En la implementación de estas interfaces se utilizan los mismos identificadores definidos en el archivo, para que estos sean mostrados correctamente.
- En la carpeta “**resources**” localizada en el directorio **servidor\abcdconfig\resources**, se encuentran las imágenes e íconos del sistema utilizados para la construcción del SIGB ABCD versión 3.0.
- La carpeta “**usr**” localizada en el directorio **servidor\repository\usr**, contiene los componentes del servidor de aplicaciones, necesarios para el funcionamiento del sistema. Dichos componentes permiten que los módulos tengan un correcto rendimiento de forma unitaria, y al integrarse los módulos que son dependientes necesitan de la estabilidad funcional de uno para la ejecución del otro.

Capítulo 3: Implementación y prueba de la solución

```
cu.uci.abcd.circulation 1.0.0
cu.uci.abcd.dao.circulation [1.0.0,1.0.0]
cu.uci.abcd.dao.common [1.0.0,1.0.0]
cu.uci.abcd.dao.management.library [1.0.0,1.0.0]
cu.uci.abcd.domain.circulation [1.0.0,1.0.0]
cu.uci.abcd.domain.common [1.0.0,1.0.0]
cu.uci.abcd.domain.management.library [1.0.0,1.0.0]
cu.uci.abcd.management.report [1.0.0,1.0.0]
cu.uci.abcd.management.security [1.0.0,1.0.0]
cu.uci.abos.core.ui [1.0.0,1.0.0]
cu.uci.abos.l10n [1.0.0,1.0.0]
cu.uci.abos.l10n.circulation [1.0.0,1.0.0]
cu.uci.abos.l10n.platform [1.0.0,1.0.0]
cu.uci.abos.l10n.util [1.0.0,1.0.0]
cu.uci.abos.platform.util [1.0.0,1.0.0]
cu.uci.abos.ui.api [1.0.0,1.0.0]
cu.uci.abos.util.api [1.0.0,1.0.0]
cu.uci.abos.validation.ui [1.0.0,1.0.0]
cu.uci.abos.widget.CompoundGroup [1.0.0,1.0.0]
cu.uci.abos.widgets.grid [1.3.0,1.3.0]
cu.uci.abos.widgets.paginator [1.0.0,1.0.0]
javax.persistence [2.0.4,2.0.4]
javax.persistence.criteria [2.0.4,2.0.4]
```

Figura 18: Librerías y bundles utilizadas por el módulo Circulación.

El archivo `osgi.beam.definition.xml` se encuentran los servicios publicados por el módulo.

Capítulo 3: Implementación y prueba de la solución

```

cu.uci.abcd.circulation.ui | x osgi-bean-definition.xml |
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">

  <!-- Creacion del controlador de la presentacion -->

  <!-- Loan User -->
  <bean id="proxy"
    class="cu.uci.abcd.circulation.ui.controller.AllManagementLoanUserViewController" />

  <!-- Creacion de la factoria de presentacion -->
  <bean id="consultLoanUser" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />
  <bean id="consultSanction" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />
  <bean id="consultLoanObject" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />

  <bean id="registerLoanUser" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />
  <bean id="registerSantion" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />
  <bean id="registerLoan" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />

  <bean id="consultTransaction" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />
  <bean id="consultReservation" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />

  <bean id="registerRenew" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />
  <bean id="registerReturn" class="cu.uci.abos.core.ui.ContributorFactoryImpl" />

</beans>

```

Figura 19: Servicios definidos en el archivo osgi.beam.definition.xml

3.7.2 Diseño de caso de prueba

Los casos de pruebas fueron definidos por cada uno de los casos de uso definidos anteriormente por el analista del proyecto. Se realizaron 3 iteraciones para verificar que en cada uno de los escenarios definidos en los casos de prueba se obtuvieran los resultados satisfactorios.

Caso de prueba: **Consultar Transacciones (Iteración 3)**

Tabla 4: Caso de prueba "Consultar Transacciones"

Escenario	Respuesta del sistema	Flujo central	Resultado esperado
			X

Capítulo 3: Implementación y prueba de la solución

<p>EC 1.1 Accede a Consultar Transacciones</p>	<p>Brinda la posibilidad de introducir los datos elementales de búsqueda:</p> <ul style="list-style-type: none"> • Título • No de Control • Usuario • Código de usuario <p>Seleccionar:</p> <ul style="list-style-type: none"> • Tipo de objeto • Tipo de usuario • Tipo de préstamo • Estado <p>Permite:</p> <ul style="list-style-type: none"> • Realizar una búsqueda a partir de los datos mostrados • Realizar una búsqueda avanzada con otros datos • Salir de la vista actual • Realizar una nueva búsqueda 	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p>	
			X

Capítulo 3: Implementación y prueba de la solución

EC 1.2 Consultar Transacciones

Brinda la posibilidad de introducir los datos elementales de búsqueda:

- Título
- No de Control
- Usuario
- Código de usuario

Seleccionar:

- Tipo de objeto
- Tipo de usuario
- Tipo de préstamo
- Estado

Permite:

- Realizar una búsqueda a partir de los datos mostrados
- Realizar una búsqueda avanzada con otros datos

- Salir de la vista actual
- Realizar una nueva búsqueda

Validas los datos introducidos.

Muestra un resumen de los datos de las Transacciones coincidentes:

- Título
- No de Control
- Tipo de Objeto
- Tipo préstamo
- Estado
- Usuario
- Tipo de Usuario
- Código de usuario

Permite:

- Ordenar el listado de manera ascendente o descendente por todos los campos del listado

- Visualizar un elemento de la lista mostrada

- Exportar a PDF
- Exportar a Hoja de Cálculo

Y permite en dependencia de los permisos que posea el usuario y del estado de la Transacción:

- Devolver (Si la transacción tiene estado "Prestado" y el usuario tiene los permisos en el sistema)
- Renovar (Si la transacción tiene estado "Prestado", el usuario tiene los permisos en el sistema y el Usuario de Préstamo puede renovar)
- Eliminar (Si el usuario tiene los permisos en el sistema)

1- Accede a la acción "**Transacciones**", perteneciente a la opción "**Consultar**", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.

2- Introduce los datos que considere para realizar una búsqueda.

3- Selecciona el botón "**Buscar**" que permite realizar una búsqueda a partir de los datos introducidos.

Capítulo 3: Implementación y prueba de la solución

<p>EC 1.3 Exportar a PDF</p>	<p>Permite exportar a PDF los datos mostrados, ver: CP Exportar a PDF.</p>	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- Introduce los datos que considere para realizar una búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.</p> <p>4- Selecciona la opción que le permite exportar los datos mostrados a PDF.</p>	<p>X</p>
<p>EC 1.4 Exportar a Hoja de Cálculo</p>	<p>Permite exportar a Hoja de Cálculo los datos mostrados, ver: CP Exportar a Hoja de Cálculo.</p>	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- Introduce los datos que considere para realizar una búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.</p> <p>4- Selecciona la opción que le permite exportar los datos mostrados como Hoja de Cálculo.</p>	<p>X</p>
			<p>X</p>

Capítulo 3: Implementación y prueba de la solución

<p>EC 1.5 Realizar una nueva búsqueda.</p>	<p>Borra de los criterios de búsqueda los datos introducidos y/o seleccionado.</p> <p>Oculto el listado resultado, en caso de que se haya generado una consulta con resultados.</p>	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- Introduce los datos que considere para realizar una búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.</p> <p>4- El actor selecciona la opción que le permite realizar una nueva búsqueda.</p>	
<p>EC 1.6 Ordenar Listado</p>	<p>Reordena y muestra la lista de coincidencias ordenadas de manera ascendente o por el campo seleccionado.</p>	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- Introduce los datos que considere para realizar una búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a</p>	<p>X</p>

Capítulo 3: Implementación y prueba de la solución

		<p>partir de los datos introducidos.</p> <p>4- El actor selecciona la opción que le permite ordenar el listado de manera ascendente o descendente por un campo del listado.</p> <p>5- Reordena y muestra la lista de coincidencias ordenadas de manera ascendente o por el campo seleccionado.</p>	
EC 1.7 Búsqueda Avanzada	<p>Muestra todos los criterios de búsqueda básica y brinda la posibilidad de seleccionar los datos asociados a los criterios avanzados.</p> <p>Del Objeto de Préstamo:</p> <ul style="list-style-type: none"> • Autor <p>Del Usuario de Préstamo:</p> <ul style="list-style-type: none"> • Primer Nombre • Segundo Nombre • Primer Apellido • Segundo Apellido <p>De la Transacción:</p> <ul style="list-style-type: none"> • Rango de fechas <p>o Desde</p> <p>o Hasta</p> <p>Permite:</p> <ul style="list-style-type: none"> • Acceder a realizar una búsqueda básica 	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- El actor selecciona la opción que le permite realizar una búsqueda avanzada.</p> <p>3- Introduce los datos que considere para realizar una búsqueda.</p> <p>4- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.</p>	<p>X</p>
			X

Capítulo 3: Implementación y prueba de la solución

<p>EC 1.8 Regresar a la Búsqueda Básica</p>	<p>Ocultar los criterios de la búsqueda avanzada.</p>	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- El actor selecciona la opción que le permite mostrar más criterios de búsqueda y posteriormente selecciona la opción que le permite regresar a la búsqueda básica.</p>	
<p>EC 1.9 Campos Vacíos.</p>	<p>Muestra un mensaje de información: "Debe especificar un criterio de búsqueda".</p> <p>Muestra un indicador sobre el/los campos vacíos.</p>	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- No introduce ni selecciona ningún dato como criterio de búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.</p>	<p>X</p>
<p>EC 1.10 No existen coincidencias.</p>	<p>Muestra un mensaje de información: "No se encontraron coincidencias".</p>	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- Introduce los datos que considere para realizar una búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a</p>	<p>X</p>

Capítulo 3: Implementación y prueba de la solución

		partir de los datos introducidos.	
EC 1.11 Visualizar Elemento	Muestra los detalles del elemento, ver: CP Gestionar Transacciones	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- Introduce los datos que considere para realizar una búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.</p> <p>4- El actor selecciona la opción que le permite visualizar un elemento de la lista mostrada.</p>	X
			X

Capítulo 3: Implementación y prueba de la solución

EC 1.12 Renovar Transacción	Permite renovar una transacción, ver: CP Gestionar Transacción.	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- Introduce los datos que considere para realizar una búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.</p> <p>4- El actor selecciona la opción que le permite renovar.</p>	
EC 1.13 Devolver Transacción	Permite devolver una transacción, ver: CP Gestionar Transacción.	<p>1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.</p> <p>2- Introduce los datos que considere para realizar una búsqueda.</p> <p>3- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.</p> <p>4- El actor selecciona la opción que le permite devolver.</p>	X
			X

Capítulo 3: Implementación y prueba de la solución

EC 1.14 Eliminar Datos Listados	Permite eliminar los datos del Objeto de Préstamo, ver: CP Gestionar Transacciones	<ol style="list-style-type: none">1- Accede a la acción "Transacciones", perteneciente a la opción "Consultar", ubicada en el menú lateral izquierdo perteneciente al módulo Circulación.2- Introduce los datos que considere para realizar una búsqueda.3- Selecciona el botón "Buscar" que permite realizar una búsqueda a partir de los datos introducidos.4- El actor selecciona la opción que le permite eliminar los datos de una Transacción.	
---------------------------------	---	--	--

Capítulo 3: Implementación y prueba de la solución

3.7.3 Resultado de las pruebas

- Pruebas funcionales:

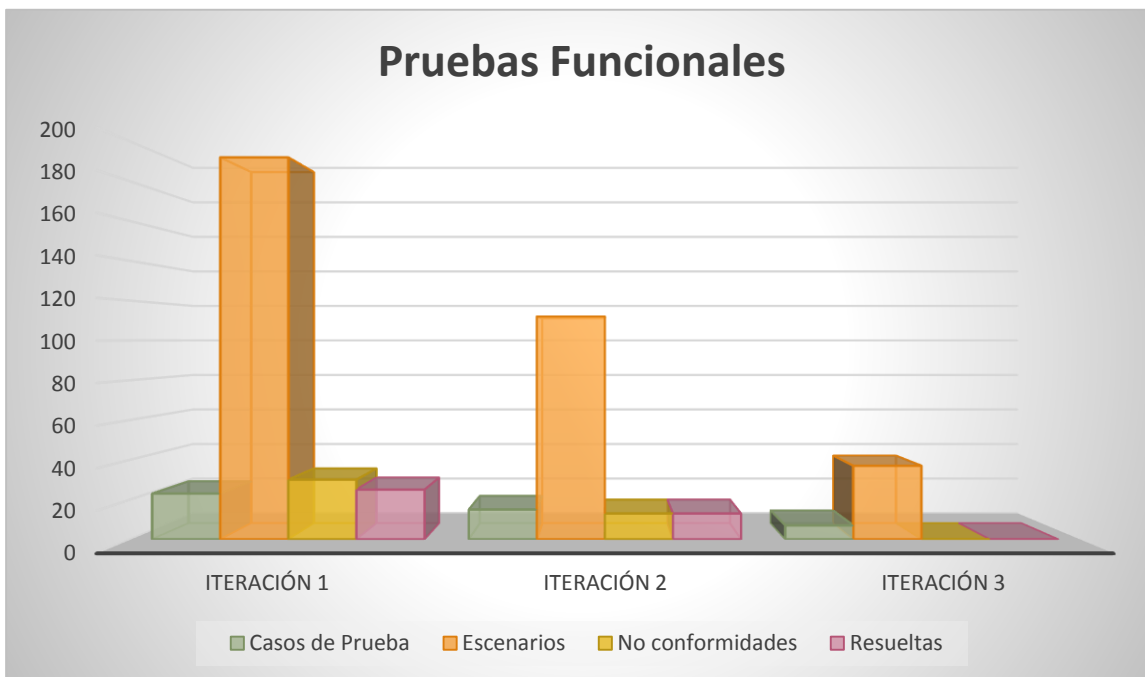


Figura 20: Gráfico correspondiente al resultado de las pruebas funcionales realizadas.

Iteración 1:

23 casos de pruebas, 192 escenarios de prueba, 30 no conformidades, 25 no conformidades resueltas. Entre las no conformidades encontradas se presentaron:

- No se muestra el mensaje “el usuario no es válido”.
- Permite prestar más objetos que la cantidad definida en las reglas de préstamo.
- No muestra correctamente las transacciones históricas asociadas al usuario seleccionado.
- No muestra el indicador sobre los campos vacíos.
- No muestra el indicador sobre los campos con datos introducidos incorrectamente.
- Permite adicionar un préstamo sin seleccionar el tipo de préstamo.
- No permite devolver una transacción.
- No permite renovar una transacción correctamente.
- No permite realizar el pago de una sanción.

Capítulo 3: Implementación y prueba de la solución

Iteración 2:

15 casos de pruebas, 112 escenarios de prueba, 13 no conformidades, 13 no conformidades resueltas. Entre las no conformidades encontradas se presentaron:

- No se muestra el mensaje “el usuario no es válido”.
- No muestra el indicador sobre los campos que contienen datos no validos al registrar un usuario de préstamo.
- No muestra el indicador sobre los campos que contienen datos no validos al registrar una transacción.
- No muestra el indicador sobre los campos vacíos al registrar una transacción.
- No muestra el mensaje “Debe especificar un criterio de búsqueda” al consultar los usuarios de préstamo.
- No muestra el mensaje “No existen coincidencias” al consultar las sanciones.
- No limpia los campos al presionar el botón “Realizar Nueva Búsqueda” al consultar los usuarios de préstamo.

Iteración 3:

18 casos de pruebas, 125 escenarios de prueba, 0 no conformidades, 0 no conformidades resueltas.

Al realizarse la tercera iteración de las pruebas funcionales, se eliminan las no conformidades existentes en el módulo implementado.

- Pruebas de integración:

Al realizarse las pruebas de integración se obtuvieron los siguientes resultados:

Capítulo 3: Implementación y prueba de la solución

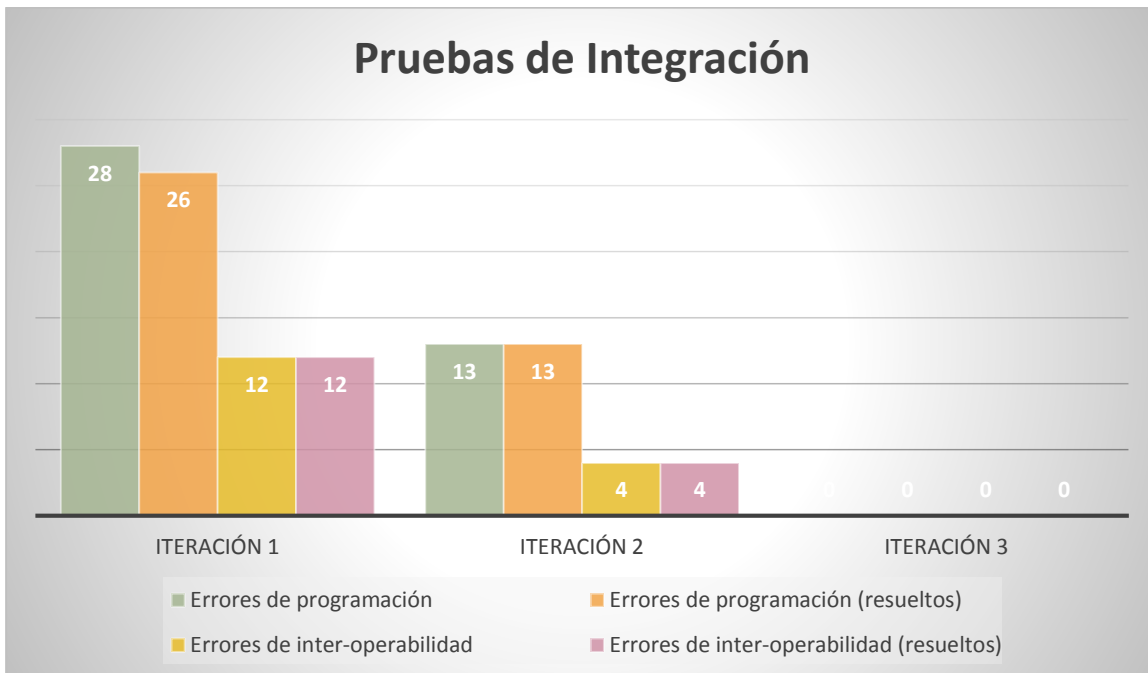


Figura 21: Gráfico correspondiente al resultado de las pruebas de integración realizadas.

Luego de realizada la tercera iteración de las pruebas de integración, se llegó a la conclusión de que el sistema se encuentra correctamente integrado. Esto es debido a que no existen errores de programación, ni errores de inter-operabilidad, permitiendo esto que los módulos interactúen correctamente como un todo.

Conclusiones del capítulo

En este capítulo se presentaron las clases implementadas en el módulo, divididas en clases de servicio, clases del dominio, clases de acceso a datos y clases controladoras. Se emplearon los estándares de codificación para lograr tener un mejor entendimiento sobre el código en el momento de brindarle soporte y mantenimiento al módulo, o simplemente para realizar un cambio en cualquier parte de la implementación. Se utilizó el tratamiento de errores para aumentar la calidad del sistema y como mecanismo para manejar los diferentes escenarios negativos de la solución propuesta.

Se presentó el diagrama de componentes, permitiendo visualizar las dependencias entre los bundles, librerías y componentes que integran el sistema; además del diagrama de despliegue donde se especifican los requerimientos que deben tener cada uno de los nodos y la forma en la que dichos elementos se comunican.

Capítulo 3: Implementación y prueba de la solución

Se mostraron algunas de las interfaces relacionadas con las funcionalidades definidas y las pruebas de software realizadas para la detección de no conformidades.

CONCLUSIONES

- El estudio de los sistemas existentes para realizar el proceso de préstamo de materiales bibliográficos, demuestra que los mismos no brindan solución al problema de la investigación, por lo que fue necesario una propuesta del módulo Circulación para el Sistema ABCD 3.0. Se estudiaron y definieron las herramientas, tecnologías y metodología necesarias para el diseño e implementación de la solución propuesta, para obtener la fundamentación teórica necesaria para llevar a cabo el desarrollo de la investigación.
- Se mostraron los elementos que componen el diseño de la solución mediante la presentación de los artefactos generados a lo largo de la investigación, quedando evidenciados los patrones utilizados para la posterior implementación del sistema.
- Se realizó la implementación de la aplicación, empleando los estándares de codificación definidos. Se presentaron los artefactos asociados a este capítulo de la investigación y además se realizaron las pruebas de software para la detección y corrección de errores.

RECOMENDACIONES

- Se propone confeccionar un manual de usuario donde se describa paso a paso cada una de las funcionalidades implementadas en el módulo.
- Realizar la implementación del envío de notificaciones vía correo electrónico, impresas o vía teléfono móvil (sms).
- Se propone implementar la posibilidad de pago de las multas mediante las tarjetas magnéticas, para que así el usuario en cuestión no tenga que personarse en la institución para realizar el pago de las mismas.

Referencias Bibliográficas

REFERENCIAS

1. Ochoa Gutierrez, Jadier. IFLA World Library and Information Congress. [En línea] 2 de Julio de 2012. [Citado el: 30 de Diciembre de 2014.] <http://conference.ifla.org/past-wlic/2012/147-gutierrez-es.pdf>.
2. *Bibliotecas digitales y sociedad de la información*. Lacruz, María del Carmen Agustín. Zaragoza : s.n., 1998.
3. Lozano, Tony. *La gestión del cambio en las bibliotecas electrónicas*. Granada : s.n., 2002.
4. Diccionario de la Real Academia Española (DRAE). [En línea] 2014. [Citado el: 13 de Enero de 2015.] <http://www.rae.es/recursos/diccionarios/drae>.
5. Blaya, Inmaculada. Slideshare. [En línea] 9 de Mayo de 2006. [Citado el: 12 de Abril de 2015.] <http://s3.amazonaws.com/ppt-download/gestionprocesos-1217005093737113-9.ppt?response-content-disposition=attachment&Signature=zAvmlVWJ9WO52iHmJvdIE0%2Ffr%2Fs%3D&Expires=1429816116&AWSAccessKeyId=AKIAIA7QTBOH2LDUZRTQ>.
6. *Aplicación de los "Requisitos funcionales de los registros bibliográficos" (FRBR) en los catálogos en línea*. González, Yolanda Martín y Hilario, Ana B. Ríos. 2005.
7. Melero, Garcia LA. *Automatización de bibliotecas*. Madrid : Arco/Libros, 1999.
8. *Introducción a un nuevo Aleph*. Ponsati Obiols, Agnès. s.l. : ExLibris, 2000.
9. Slideshare, Aleph 500. [En línea] <http://es.slideshare.net/dolbac/aleph-500>.
10. Janium. [En línea] <http://www.janium.com/>.
11. Janium. [En línea] <http://www.janium.com/modulos-janium/>.
12. *Sistemas integrales para la automatización de bibliotecas basados en software libre*. Navarrete, Óscar Arriola y Yáñez, Katya Butrón. Ciudad de La Habana : s.n., 24 de Octubre de 2008.
13. Koha Latino. [En línea] [Citado el: 12 de Febrero de 2015.] http://kohalatinoinfo.com/koha_circulacion.html.
14. Kruchten, Philippe. *The Rational Unified Process*. Boston : Addison-Wesley, 2004.
15. UML. [En línea] [Citado el: 12 de Febrero de 2015.] <http://www.uml.org/>.
16. Visual Paradigm. [En línea] [Citado el: 30 de Diciembre de 2014.] <http://www.visual-paradigm.com/>.
17. Eclipse. [En línea] <http://eclipse.org/rap/>.
18. [En línea] [Citado el: 25 de Abril de 2015.] <https://srgperez.wordpress.com/2009/06/10/diferencias-y-similitudes-entre-java-y-c/>.

Referencias Bibliográficas

19. Genbeta: Desarrollo de Software. [En línea] <http://www.genbetadev.com/herramientas/eclipse-ide>.
20. Eclipse. [En línea] The Eclipse Foundation, 2015. [Citado el: 12 de Mayo de 2015.] <http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/keplerr>.
21. Martínez, Rafael. Portal en español sobre PostgreSQL. [En línea] http://www.postgresql.org.es/sobre_postgresql.
22. Eclipse. Virgo Server. [En línea] <http://eclipse.org/virgo/>.
23. Eclipse. [En línea] The Eclipse Foundation, 2015. [Citado el: 12 de Mayo de 2015.] <http://www.eclipse.org/equinox/documents/quickstart.php>.
24. JUnit. [En línea] 4 de Diciembre de 2014. [Citado el: 23 de Mayo de 2015.] <http://junit.org/>.
25. Spring. *Spring*. [En línea] Pivotal Software, 12 de Enero de 2015. [Citado el: 13 de Mayo de 2015.] <http://docs.spring.io/spring/docs/3.2.13.RELEASE/spring-framework-reference/htmlsingle/>.
26. *Evaluación y análisis de rendimiento de los frameworks de persistencia Hibernate y Eclipselink*. Mauro Callejas Cuervo, Diego Iván Peñalosa Parra, Andrea Catherine Alarcón Aldana. 2011, Ventana Informática.
27. Cabrera Medina, Andy. *0114_ESPECIFICACIÓN DE CASOS DE USO*. 2014.
28. Grady Boosh, Jim Rumbaugh, Ivar Jacobson. *El lenguaje unificado de modelado*. 1991.
29. Guía de Arquitectura en N-Capas. [En línea] Octubre de 2012. [Citado el: 12 de Enero de 2015.] <http://arquitecturancapas.blogspot.com/>.
30. Proyecto, Equipo del. *Arquitectura N Capas del Sistema ACBD 3.0*. 2014.
31. Academia.EDU. [En línea] http://www.academia.edu/10102692/Arquitectura_de_n_capas.
32. Grosso, Andres. Patrones GRASP. [En línea] <http://www.practicadesoftware.com.ar/2011/03/patrones-grasp/>.
33. Patrones GOF. [En línea] <http://www.godtic.com/blog/2012/11/15/patrones-de-diseno-gof/>.
34. Sparxs System. [En línea] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.
35. Pressman. *Pruebas de Software*. 2002.

BIBLIOGRAFÍA

1. Ochoa Gutierrez, Jadier. IFLA World Library and Information Congress. [En línea] 2 de Julio de 2012. [Citado el: 30 de Diciembre de 2014.] <http://conference.ifla.org/past-wlic/2012/147-gutierrez-es.pdf>.
2. *Bibliotecas digitales y sociedad de la información*. Lacruz, María del Carmen Agustín. Zaragoza : s.n., 1998.
3. Lozano, Tony. *La gestión del cambio en las bibliotecas electrónicas*. Granada : s.n., 2002.
4. Diccionario de la Real Academia Española (DRAE). [En línea] 2014. [Citado el: 13 de Enero de 2015.] <http://www.rae.es/recursos/diccionarios/drae>.
5. Blaya, Inmaculada. Slideshare. [En línea] 9 de Mayo de 2006. [Citado el: 12 de Abril de 2015.] <http://s3.amazonaws.com/ppt-download/gestionprocesos-1217005093737113-9.ppt?response-content-disposition=attachment&Signature=zAvmlVWJ9WO52iHmJvdIE0%2Ffr%2Fs%3D&Expires=1429816116&AWSAccessKeyId=AKIAIA7QTBOH2LDUZRTQ>.
6. *Aplicación de los "Requisitos funcionales de los registros bibliográficos" (FRBR) en los catálogos en línea*. González, Yolanda Martín y Hilario, Ana B. Ríos. 2005.
7. Melero, Garcia LA. *Automatización de bibliotecas*. Madrid : Arco/Libros, 1999.
8. *Introducción a un nuevo Aleph*. Ponsati Obiols, Agnès. s.l. : ExLibris, 2000.
9. Slideshare, Aleph 500. [En línea] <http://es.slideshare.net/dolbac/aleph-500>.
10. Janium. [En línea] <http://www.janium.com/>.
11. Janium. [En línea] <http://www.janium.com/modulos-janium/>.
12. *Sistemas integrales para la automatización de bibliotecas basados en software libre*. Navarrete, Óscar Arriola y Yáñez, Katya Butrón. Ciudad de La Habana : s.n., 24 de Octubre de 2008.
13. Koha Latino. [En línea] [Citado el: 12 de Febrero de 2015.] http://kohalatinoinfo.com/koha_circulacion.html.
14. Kruchten, Philippe. *The Rational Unified Process*. Boston : Addison-Wesley, 2004.
15. UML. [En línea] [Citado el: 12 de Febrero de 2015.] <http://www.uml.org/>.
16. Visual Paradigm. [En línea] [Citado el: 30 de Diciembre de 2014.] <http://www.visual-paradigm.com/>.
17. Eclipse. [En línea] <http://eclipse.org/rap/>.
18. [En línea] [Citado el: 25 de Abril de 2015.] <https://srgperez.wordpress.com/2009/06/10/diferencias-y-similitudes-entre-java-y-c/>.

19. Genbeta: Desarrollo de Software. [En línea] <http://www.genbetadev.com/herramientas/eclipse-ide>.
20. Eclipse. [En línea] The Eclipse Foundation, 2015. [Citado el: 12 de Mayo de 2015.] <http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/keplerr>.
21. Martínez, Rafael. Portal en español sobre PostgreSQL. [En línea] http://www.postgresql.org.es/sobre_postgresql.
22. Eclipse. Virgo Server. [En línea] <http://eclipse.org/virgo/>.
23. Eclipse. [En línea] The Eclipse Foundation, 2015. [Citado el: 12 de Mayo de 2015.] <http://www.eclipse.org/equinox/documents/quickstart.php>.
24. JUnit. [En línea] 4 de Diciembre de 2014. [Citado el: 23 de Mayo de 2015.] <http://junit.org/>.
25. Spring. *Spring*. [En línea] Pivotal Software, 12 de Enero de 2015. [Citado el: 13 de Mayo de 2015.] <http://docs.spring.io/spring/docs/3.2.13.RELEASE/spring-framework-reference/htmlsingle/>.
26. *Evaluación y análisis de rendimiento de los frameworks de persistencia Hibernate y Eclipselink*. Mauro Callejas Cuervo, Diego Iván Peñalosa Parra, Andrea Catherine Alarcón Aldana. 2011, Ventana Informática.
27. Cabrera Medina, Andy. *0114_ESPECIFICACIÓN DE CASOS DE USO*. 2014.
28. Grady Boosh, Jim Rumbaugh, Ivar Jacobson. *El lenguaje unificado de modelado*. 1991.
29. Guía de Arquitectura en N-Capas. [En línea] Octubre de 2012. [Citado el: 12 de Enero de 2015.] <http://arquitecturancapas.blogspot.com/>.
30. Proyecto, Equipo del. Arquitectura N Capas del Sistema ACBD 3.0. 2014.
31. Academia.EDU. [En línea] http://www.academia.edu/10102692/Arquitectura_de_n_capas.
32. Grosso, Andres. Patrones GRASP. [En línea] <http://www.practicadesoftware.com.ar/2011/03/patrones-grasp/>.
33. Patrones GOF. [En línea] <http://www.godtic.com/blog/2012/11/15/patrones-de-diseno-gof/>.
34. Sparxs System. [En línea] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.
35. Pressman. *Pruebas de Software*. 2002.
36. *Los sistemas integrados de gestión bibliotecaria: estructuras de datos y recuperación de información*. Anegón, Félix de Moya. Madrid : ANABAD, D.L., 1994.
37. Universidad de Girona. [En línea] <http://ima.udg.edu/~sellares/EINF-ES2/Present1011/MetodoPesadesRUP.pdf>.
38. *Tecnología de la Información y las Comunicaciones*. s.l. : Consejo Privado de Competitividad, 2014-2015.

39. Selenium Project. *Selenium Documentation*. 2010.

