Universidad de las Ciencias Informáticas

Facultad 4

Análisis y Diseño del subsistema Préstamos del proyecto Modernización del Sistema Bancario Cubano.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Liset González Polanco

Yadian Guillermo Pérez Betancourt

Tutores: Ing. Yoan Antonio López Rodríguez

Ing. Yosbel Rivero Alonso

La Habana, Cuba
Mayo 2009

FRASE

"El futuro de Cuba tiene que ser necesariamente un futuro de hombres de ciencia, de hombres de pensamiento"

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores del trabajo titulado: *Análisis y Diseño del subsistema Préstamos del proyecto Modernización del Sistema Bancario Cubano* y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Liset González Polanco	Yadian Guillermo Pérez Betancourt
Yoan Antonio López Rodríguez	Yosbel Rivero Alonso

DATOS DE CONTACTO

Ing. Yoan Antonio López Rodríguez.

Graduado de Ingeniero en Ciencias Informáticas en el 2008. Título de Oro. Actualmente imparte clases de Teleinformática II en la Facultad 4, integra el proyecto Banco Nacional y se desempeña como analista y Jefe del módulo Préstamos en el mismo.

Ing. Yosbel Rivero Alonso

Graduado de Ingeniero en Ciencias Informáticas en el 2008. Título de Oro. Cumplió misión en Venezuela. Actualmente imparte clases de Seguridad Informática en la Facultad 4, integra el proyecto Banco Nacional y se desempeña como analista y Jefe del módulo Comercio Internacional en el mismo.

AGRADECIMIENTOS

De Liset:

Una de las partes más complejas y que más me anudan la garganta y me hacen sollozar es los agradecimientos y la dedicatoria, en un momento pasan por mi mente los buenos y malos recuerdos de estos 5 años que parecen que se fueron pronto, pero no, fueron 1825 días vividos; lejos de la casa de los mimos de mami, papi, mima y pipo.

Debo darles las gracias a todos los jóvenes que dejaron atrás sus sueños y lucharon por una Patria para todos, donde todo aquel que tuviese el interés y el deseo pudiese estudiar, a mi Comandante en jefe Fidel Castro Ruz por todo, por la UCI, a mi familia por ser tan incondicional y por darme tantas fuerzas, por ayudarme y guiarme por ser tan unidos, por ser mi Familia

Antes de comenzar quiero decir: Mamita (Magalys Elaine Polanco Izada), que decirte a ti que estudiaste matemática, programación y física conmigo, que siempre me has dada tu ayuda y consejo, por tu desvelos, en fin mamita, por poner tus sueños junto a los míos, Papito (Javier de Jesús González Ávila), no te pongas celosito tu eres un papito único y que junto a mi mamita me han dado el apoyo que siempre he necesitado, recuerdas con el número 2, Lili mi hermanita querida gracias por cuidar a mi mamita y a mi papito en todo esto tiempo que apenas he podido estar con ellos.

A pipo (Gilberto Edilberto Polanco Zayas-Bazán) gracias por confiar en mi, cuando yo creo nadie confiaba, por tu apoyo y tu confianza esa que nunca podré traicionar que me impulsaba antes de cada clase dominar el contenido y a mima (Caridad Bárbara Izada Manzolo), a Silvia Odelta Ávila Ávila y a Gilberto, con mis abuelitos que he compartido mis sueños y que me encanta su compañía después de almuerzo en el portal hablábamos del pasado, del presente y también nos aventuramos a predecir el futuro.

A mis tías que son como mi mamá con las cuales he contado en momentos difíciles y que está de más decir que importantes para mí; tía Sonia (Sonia Polanco Izada), tía Nana (Iliana Elena Polanco Izada), a mis tíos Alexis Martínez y Salvita (Salvador Lorenzo Echavarría), y mis primitas que son además hermanitas Seilita y Gladiolo. Y ya a los de Menéndez y a los restantes de Bartle porque creo que estoy realizando un árbol genealógico de agradecimientos.

Quiero darle las gracias a Yadian Guillermo Pérez Betancourt, por su amistad, por su cariño, por su ayuda, por su comprensión, por estar siempre que lo he necesitado y por las veces que estaba sin yo pedírselo, en fin por pensar en mí.

De Yadian:

A mi mamita querida Gertrudis, fuente de inspiración, sin su ayuda y educación hoy yo no estaría aquí, a Pedro mi papá por confiar en mí y apoyarme en cada momento, a mi hermana por ayudar a mi mamá durante todos estos años que no he podido estar con ella.
A mi novia Liset por ser mi confidente, mi amiga, por saber comprenderme y apoyarme en todo.
A mis abuelos, tíos y primos, en fin en mi familia.
A todos los que siempre han creído en mí y esperan ansiosos este momento

A la revolución por hacer realidad este bello sueño que es la UCI y por dejarme formar parte de ella.
A nuestro Fidel por ser mi ejemplo, mi guía.
A los profesores de la facultad, y a los que en ella laboran Yoisi, Gricel, Idania, a mis compañeros del proyecto y a nuestros tutores por sus consejos, por su ayuda, y por estar pendiente en el correcto desempeño de esta tesis.
Gracias a todos.

DEDICATORIA

De Liset:

A mi mamita Magalys, a mi papito Javier, a Lili y a mis abuelitos Cari y Gilberto por siempre apoyarme y confiar en mí.

De Yadian:

A mi mamá, a mi papá Pedro, a mi hermana, a la memoria de mi abuelo Pedro, a mis sobrinitos, A todos los que siempre creyeron en mí.

RESUMEN

El presente trabajo aborda el análisis y diseño del subsistema Préstamos del proyecto Modernización del Sistema Bancario Cubano, el mismo se ajusta a los lineamientos arquitectónicos establecidos dentro del proyecto. Se describen las técnicas y herramientas utilizadas, así como la aplicación de patrones dentro del flujo de desarrollo de software siguiendo la metodología y procesos dictados por el Proceso Unificado de Desarrollo. Los artefactos se generaron usando como lenguaje de modelado el Lenguaje Unificado de Modelado y auxiliados por el Visual Paradigm como herramienta de Ingeniería de Software Asistida por Computadora.

Mediante el flujo de trabajo de análisis y diseño se obtuvieron artefactos entendibles que viabilizan a los implementadores la automatización de una plataforma capaz de satisfacer las necesidades de los clientes. Para garantizar la calidad de los artefactos generados se aplicaron métricas, que arrojaron resultados positivos.

Palabras claves.

Análisis, diseño, procesos bancarios, préstamos.

TABLA DE CONTENIDOS

INTRODUCCIÓN

Una consecuencia del desarrollo y la complejidad alcanzados por la actividad económica en las últimas décadas, ha sido el perfeccionamiento de los sistemas informáticos, así como de su evolución. Los bancos por su parte, controlan la vida económica del mundo y su actuación estratégica se ha potenciado en las últimas décadas con el uso de los sistemas informáticos, los cuales han contribuido a lograr una mayor eficiencia en la toma de decisiones y en el resto de sus operaciones.

Actualmente el entorno en que se desarrolla la actividad bancaria, exige un constante esfuerzo de mejoras tales como: el rediseño de procesos, la mejora de la productividad, la reducción de costos y el alcance de una buena calidad para mejorar la satisfacción de los clientes. Con este propósito en el Banco Nacional de Cuba se ha utilizado el sistema informático SABIC, que a su vez ha funcionado como sistema contable.

Con el transcurso de los años se han realizado nuevas versiones de este sistema, las cuales se han ido instalando en una serie de entidades bancarias cubanas, a pesar de ello el Banco Nacional de Cuba continúa trabajando con la versión inicial del SABIC. La plataforma tecnológica sobre la que está basado este sistema contiene un conjunto de limitaciones entre las que se encuentra la mono tarea, que está directamente asociada a la utilización del MS-DOS como sistema operativo, por lo cual no puede llevar a cabo de una manera efectiva y menos compleja la toma de decisiones, la obtención de los reportes y el resto de sus operaciones. Además resulta extremadamente engorroso, trabajar sobre varias plataformas. Entre las áreas bancarias que se ven afectadas se encuentra el área de Préstamos en la cual se centra el desarrollo de este trabajo.

Debido a las limitaciones que enfrenta el sistema del Banco Nacional de Cuba, se determina la necesidad de realizar un nuevo sistema informático que resuelva esas dificultades. Dándole la tarea a la Universidad de las Ciencias Informáticas (UCI), y en específico a la Facultad 4 de realizarlo.

Una vez definidos los requerimientos funcionales, surge la necesidad de obtener en un lenguaje entendible para todos (el análisis) y realizar en un lenguaje más técnico (el diseño) la propuesta de la solución, por lo que se define como ***problema a resolver:*** *¿Cómo transformar las necesidades de los clientes a un lenguaje entendible por los desarrolladores, que permita la posterior implementación?*

El ***objeto de estudio*** del presente trabajo se enfocará hacia *Los procesos de Préstamos en las entidades bancarias* y el ***campo de acción*** *los procesos de Préstamos en el Banco Nacional de Cuba.*

Para responder al problema anteriormente planteado, los autores se trazaron como **objetivo general** *Realizar el análisis y diseño del subsistema Préstamos del proyecto Modernización del Sistema Bancario Cubano que permita darle cumplimiento al levantamiento de requisitos realizado.*

Para cumplir lo planteado, se llevaron a cabo cuatro **Tareas**:

1. *Obtención del marco teórico conceptual*.
2. *Obtención del diseño del subsistema Préstamos.*
3. *Utilización de buenas prácticas y patrones en el modelo diseño.*

Y como cuarta y última tarea, sería obtener el **posible resultado;** *Del subsistema Préstamos del proyecto Modernización del Sistema Bancario Cubano, obtener el modelo de diseño.*

Para lograr la comprensión y claridad de los contenidos de la investigación realizada se ha estructurado el documento en tres capítulos.

En el Capítulo 1: Fundamentación teórica, usted podrá encontrar una serie de aspectos relacionados con la necesidad de un nuevo sistema informático para el Banco Central de Cuba, en específico para el área de Préstamos. Además se argumenta sobre las herramientas, metodologías y tecnologías a utilizar.

En el Capítulo 2: Análisis y diseño, se muestran los artefactos generados en el análisis y diseño de la propuesta de solución, se argumentan algunas de las tantas terminologías que se utilizan. Además se abordan los patrones a utilizar y una explicación de la arquitectura propuesta en el proyecto Modernización del Sistema Bancario Cubano.

En el Capítulo 3: Validación de la solución propuesta, se aplican una serie de métricas de diseño orientado a objeto y se analizan los resultados para determinar la calidad del diseño realizado para el Subsistema de Préstamos del proyecto Modernización del Sistema Bancario Cubano.

Para cada capítulo se ofrecen sus conclusiones parciales y al final del documento se exponen las conclusiones generales, las recomendaciones y el glosario de términos.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En el presente capítulo se define el conjunto de actividades que guían los esfuerzos de los autores que tienen la misión de ya transformadas las necesidades de los usuarios en requerimientos, realizar el análisis y el diseño para la posterior implementación del Subsistema de Préstamos del proyecto Modernización del Sistema Bancario Cubano. Incluyéndose además, un estudio relacionado con los sistemas contables en entidades bancarias y las herramientas a utilizar con el fin de lograr de forma productiva los objetivos propuestos.

1.2. Ingeniería de Software

Durante los primeros años de la informática, no existía una disciplina en el desarrollo del software, la programación era considerada un "arte", y no se controlaba que los ingenieros utilizaran de forma consistente los métodos. El proceso de desarrollo de un sistema se realizaba sin planificación alguna (1). La evolución y perspectivas de la Ingeniería del Software posteriormente, desde mediados de la década de los 60 hasta finales de los 70 se caracterizaron por el establecimiento del software como un producto que se desarrollaba para una distribución general. En esta época nació lo que se conoce como el mantenimiento del software que se realiza cuando cambian los requisitos de los usuarios y se hace necesaria la modificación del mismo. El esfuerzo necesario para realizar este mantenimiento era en la mayoría de los casos tan elevado que se hacía imposible llevarlo adelante.

En la actualidad en la Industria de Software hay tendencia al crecimiento del volumen y complejidad de los productos, los proyectos están excesivamente tardes, se exige mayor calidad y productividad en menos tiempo y hay insuficiente personal calificado; por lo que se puede decir que la fallas de los proyectos de software se deben a:

- Planificación irreal: Los usuarios piden un sistema para hoy que tenga costo 0 y los ingenieros no son capaces de enfrentar un plan porque no están entrenados para usar métodos de planificación y frecuentemente, las estimaciones no se basan en datos reales.

- Mala calidad del trabajo: Las prácticas pobres de Ingeniería, la carencia de métricas adecuadas de calidad y las decisiones de los directivos guiadas por una planificación irreal; traen como consecuencias tiempos de pruebas impredecibles, productos con muchos defectos, demoras en la aceptación de los usuarios y una extensa garantía de servicio y reparaciones. Una pobre calidad afecta la planificación y torna ineficiente el proceso de prueba.
- Personal inadecuado: En múltiples ocasiones el personal asignado a un proyecto se incorpora tarde, no cubre las necesidades en cuanto a cantidad y calidad. Como consecuencia el trabajo se demora o descuida, es ineficiente y sufre la moral del equipo. Con independencia del plan, los proyectos deben comenzar en tiempo y con todo el personal.
- Cambios no controlados: Es importante recordar que siempre ocurren cambios en los requerimientos, que los planes del proyecto se basan en el alcance del trabajo conocido, que los cambios siempre requieren más trabajo, sin planes detallados los equipos no pueden estimar el efecto o magnitud de los cambios y que si los equipos no controlan cada cambio, se pierde gradualmente el control del plan del proyecto (2).

Para enfrentar esta situación las empresas requieren desarrollar o adquirir una disciplina en el desarrollo del software y controlar que los ingenieros usen de forma consistente los nuevos métodos. Según Pressman 2005[1], la Ingeniería de Software es una tecnología multicapa en la que se pueden identificar: los métodos (indican cómo construir técnicamente el software), el proceso (es el fundamento de la Ingeniería de Software, es la unión que mantiene juntas las capas de la tecnología) y las herramientas (soporte automático o semiautomático para el proceso y los métodos)**. [Anexo, figura 1]**

La industria del software debe asentar cada vez más sus bases en la Ingeniería de Software, proporcionando la investigación continua para la práctica y estandarización de sus políticas, con el objetivo de que sus productos se realicen con calidad y que respondan a los requerimientos definidos.

1.2.1. Metodologías de Desarrollo de Software

Las metodologías se desarrollan con el objetivo de dar solución a los problemas existentes en la producción de software, que cada vez son más complejos. Estas engloban procedimientos, técnicas, documentación y herramientas que se utilizan en la creación de un producto de software.

Una metodología es un proceso. Un proceso de desarrollo de software es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto (3).

Este conjunto de actividades, en el proceso de desarrollo de software, es el que proponen Jacobson, Rumbaugh y Booch, tiene la misión de transformar los requerimientos del usuario en un producto de software; de manera que los integrantes del equipo y todo aquel que pueda estar interesado en el producto final, tengan la misma visión.

Las piedras angulares del proceso de desarrollo del software son: el proyecto, las personas y el producto; siendo las características del cliente, el entorno de desarrollo y las condiciones del negocio, elementos que influyen en el proceso. Existe una estrecha relación entre personas, proyecto, producto y proceso. Estos términos son conocidos como las cuatro "P" en el desarrollo de software:

Personas: Los principales autores de un proyecto software.

Proyecto: Elemento organizativo a través del cual se gestiona el desarrollo de software. El resultado de un proyecto es una versión de un producto.

Producto: Artefactos que se crean durante la vida del proyecto, como los modelos, código fuente, ejecutables y documentación.

Proceso: Un proceso de ingeniería de software es una definición del conjunto completo de actividades necesarias que guían los esfuerzos de los desarrolladores, para transformar los requisitos de usuario en un producto.

No existe una metodología de software universal. Las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigen que el proceso sea configurable (3).

Si tomamos como criterio las notaciones utilizadas para especificar artefactos producidos en actividades de análisis y diseño, podemos clasificar las metodologías en dos grupos: Metodologías Estructuradas y Metodologías Orientadas a Objetos (3). Por otra parte, considerando su filosofía de desarrollo aparece el apelativo Metodologías Tradicionales o Ágiles.

Dentro de las metodologías más utilizadas hoy día se pueden citar: RUP, XP y MSF, de ellas Rational Unified Process (RUP) se adapta mejor a los proyectos de largo plazo, dividiendo el desarrollo de software en cuatro fases desarrolladas iterativamente y abarcando seis disciplinas ingenieriles y tres de apoyo; Extreme Programing (XP) por su parte se recomienda para proyectos cortos y tiene por particularidad

contar con el usuario final como parte del equipo; Microsoft Solution Framework (MSF) se adapta por su parte a cualquier tipo de proyecto y se centra en los modelos de procesos y de equipo (4).
Para el desarrollo de este proyecto se ha seleccionado la metodología RUP debido al alcance del proyecto y a que la misma aplica muchas de las mejores prácticas del desarrollo de software moderno, enfocadas a la producción de software con calidad.

1.2.1.1. Rational Unified Process (RUP)

El Proceso Unificado de Desarrollo (RUP-Rational Unified Process) es un proceso pensado en dos dimensiones. El mismo se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo concluye con una versión del producto para los clientes. Cada ciclo consta de cuatro fases. Cada fase termina con un hito (5).
Como RUP es un proceso, en su modelación define como sus principales elementos:

Trabajadores (“quién”)	Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.
Actividades (“cómo”)	Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.
Artefactos (”qué”)	Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.
Flujo de actividades (“Cuándo”)	Secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.

El ciclo de vida de RUP se caracteriza por:

- Dirigido por casos de uso: Los casos de uso (CU) reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso (cómo se llevan a cabo).
- Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. RUP se desarrolla mediante iteraciones, comenzando por los CU relevantes desde el punto de vista de la arquitectura. El modelo de arquitectura se representa a través de vistas en las que se incluyen los diagramas de UML.
- Iterativo e Incremental: Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Por ejemplo, una iteración de elaboración centra su atención en el análisis y diseño, aunque refina los requerimientos y obtiene un producto con un determinado nivel, pero que irá creciendo incrementalmente en cada iteración.

Cuenta con 4 fases, 9 flujos de trabajos, los primeros seis son conocidos como flujos de ingeniería y los tres últimos como de apoyo. **[Anexo, Figura 2]**

La fase de Elaboración tiene como principal finalidad completar el análisis de los Casos de Uso y definir la arquitectura del sistema, en esta fase se encuentran los flujos de trabajo análisis y diseño los cuales tienen entre sus objetivos:

- Transformar los requerimientos en un diseño de cómo va a ser implementado el sistema,
- Evolucionar hacia una arquitectura del software robusta.
- Adaptar el diseño para que coincida con el ambiente de implementación, diseñando el sistema con un enfoque hacia el rendimiento.

1.2.2. Herramienta CASE a utilizar

El acrónimo CASE en inglés significa Computer Aided Software Engineering y según su traducción en español es Ingeniería de Software Asistida por Computadoras.

1.2.2.1. Visual Paradigm 6.1

Visual Paradigm es una herramienta que ofrece un entorno de creación de diagramas usando las notaciones UML 2.0, con un diseño centrado en casos de uso y enfocado además al negocio, lo cual genera un software de alta calidad. Esta herramienta fue creada para el ciclo completo de desarrollo del software que lo automatiza y acelera, permitiendo la captura de requisitos, análisis, diseño e implementación, también proporciona características tales como generación del código, ingeniería inversa y generación de informes. Permite escribir toda la especificación de un caso de uso sin necesidad de utilizar una herramienta externa como editor de texto, utilizando plantillas que se encuentran definidas, o que pueden ser creadas por los usuarios. Está diseñada para usuarios interesados en sistemas de software de gran escala, apoya los estándares más recientes de la notación UML. A pesar de todo es importante destacar que la licencia de Visual Paradigm es muy restringida.

Debido a las funcionalidades brindadas por el Visual Paradigm al permitir crear diagramas usando las notaciones UML y BPMN y por ser una herramienta multiplataforma que se integra fácilmente con varios Entorno de desarrollo integrado (IDEs), se decidió por parte de la dirección del proyecto usarla como herramienta durante todo el proceso de modelado.

1.2.3. Lenguaje de modelado

Un lenguaje de modelado de objetos se puede definir como el conjunto estandarizado de símbolos y de modos de disponerlos para modelar (parte de) un diseño de software orientado a objetos.

1.2.3.1. Notación Unified Modeling Language (UML)

UML es un lenguaje, que permite modelar analizar y diseñar sistemas orientados a objetos. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad (5). Ofrece un estándar para describir un panorama del sistema modelo, incluyendo aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables y

aspectos conceptuales como los procesos de negocios y funciones del sistema. UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software.

El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. La finalidad de los diagramas es presentar diversas perspectivas de un sistema, a las cuales se les conoce como modelo.

Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos. Es importante recalcar que UML no es una guía para realizar el análisis y diseño orientado a objetos, es decir, no es un proceso. UML es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos.

1.2.4. Framework a utilizar

Un framework es un término muy utilizado últimamente en el campo de la informática, se utiliza para referirse a un conjunto de bibliotecas, que se utilizan para implementar la estructura de un modelo para una aplicación. Esto se realiza con el objetivo de promover la reutilización de código, posibilitando que no sea necesario perder tiempo en reinventar la rueda.

Existen diferentes tipos de framework, para diferentes propósitos, algunos orientados al desarrollo de aplicaciones web, o para un determinado sistema operativo o lenguaje. En un sentido muy amplio el framework que se utiliza determina la arquitectura del software.

1.2.4.1. Spring Framework

Los frameworks dan la posibilidad de definir la forma de realización de aplicaciones de software, dando soporte y simplificando parte de la complejidad. Spring; es un framework que tiene el objetivo de facilitar la construcción de aplicaciones Java, presenta un entorno diseñado para aumentar la productividad, liberando al desarrollador de tareas repetitivas, ayudándolo a hacer diseños más consistentes. Se puede utilizar en cualquier tipo de aplicación, y es ligero por el mínimo impacto que tiene en las aplicaciones.

Spring se basa en la técnica Inversión de Control (IoC), técnica que promueve el bajo acoplamiento a partir de la inyección de dependencias (DI) entre los objetos y una implementación de desarrollo según el paradigma de Orientación a Aspectos (AOP) que presenta una estructura simplificada para el desarrollo y utilización de aspectos (módulos multiple object crosscutting).

El paradigma de Orientación a Aspectos (AOP) complementa la Programación Orientada a Objetos (POO), proponiendo otra manera de pensar sobre la estructura de un programa. Mientras que la POO descompone las aplicaciones en una jerarquía de objetos, la AOP descompone los programas en aspectos o preocupaciones. Dichas preocupaciones se convierten en servicios del sistema, separados de la lógica de negocio y que se ejecutan de manera trasversal a la funcionalidad base, lo que proporciona una definición de responsabilidades superior. Spring básicamente es un contenedor, que se encarga de gestionar, administrar el ciclo de vida y de la configuración de las clases de la aplicación. Es una aplicación open source, lo cual implica que no tiene ningún costo, ni se necesita licencia para utilizarlo, dando la libertad de incursionar en la utilización de esta aplicación, además de que está disponible todo el código fuente de este framework en el paquete de la instalación.

En sentido general Spring no intenta reinventar la rueda, sino que integra las diferentes tecnologías existentes en un solo framework posibilitando un desarrollo más sencillo y eficaz.

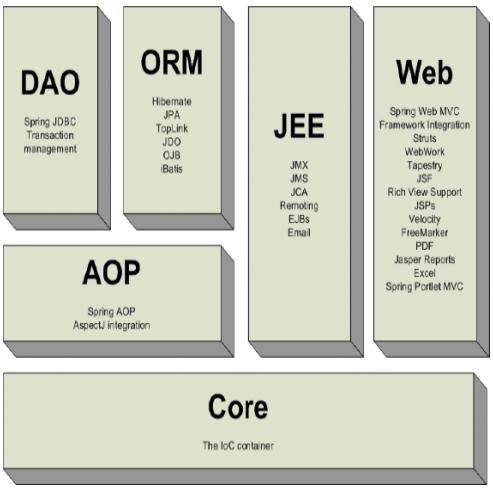


Figura 3. Arquitectura del framework Spring.

- Core: Como su nombre indica, es el núcleo de Spring. Permite técnicas de Inversión del Control (IoC) como la inyección de dependencias.
- AOP: Proporciona una implementación de programación orientada a aspectos, permitiendo definir puntos de corte e interceptores.

- DAO: Proporciona el acceso a una capa de abstracción JDBC (Java Database Connectivity) con una forma de administrar transacciones desde el módulo AOP como es el caso de Hibernate y JDO.
- ORM: Provee capas de integración para APIs de mapeo objeto-relacional.
- Web: posibilita determinadas características apropiadas para el desarrollo de aplicaciones web e integración con otros frameworks (Struts, JSF, Tapestry, etc).
- JEE: acceso e interacción con servicios enterprise.

1.2.4.2. Spring web flow

Spring web flow forma parte del paquete de desarrollo de aplicaciones web de Spring. Se caracteriza por permitir la navegación múltiple y compleja permitiendo guiar al usuario a través de una serie de pasos para completar una transacción de aplicación y por ser una plataforma web de alto nivel, además incrementa la productividad, calidad y facilidad para realizar pruebas en el proceso de desarrollo.

1.2.4.3. Hibernate

Hibernate es una solución ORM (*Object-Relational Mapping*) para Java, es open source (código abierto) y la licencia está eximida de costo. *Hibernate* busca solucionar el problema de la diferencia entre el modelo orientado a objetos y el usado en las bases de datos modelo relacional mediante archivos declarativos. Está diseñado para ser flexible en cuanto al esquema de tablas utilizado, para poder adaptarse a su uso sobre una base de datos ya existente. También tiene la funcionalidad de crear la base de datos a partir de un modelo objetual existente. Cabe mencionar que provee con Hibernate Query Language (HQL) una poderosa vía de comunicación entre el programador y la base de datos, al dotarlo de un lenguaje invariante con respecto a esta y además sintácticamente muy parecido al Structured Query Language (SQL).

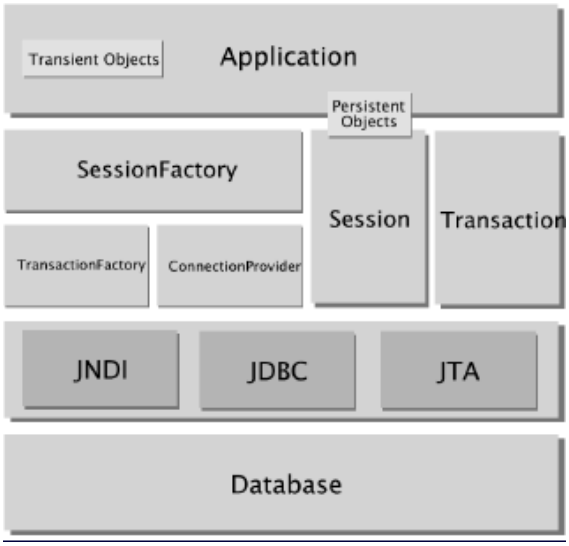


Figura 4: Arquitectura de Hibernate.

1.2.5. Lenguaje de programación y entorno de desarrollo integrado (IDE)

1.2.5.1. Lenguaje de programación Java

Java es un lenguaje de desarrollo orientado a objetos, multiplataforma. Proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets y establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas. Es un lenguaje interpretado y compilado, esto quiere decir que su código fuente se transforma en algo similar al código máquina, los bytecodes y a la vez estos bytecodes se pueden ejecutar directamente sobre cualquier máquina a la cual se hayan portado el intérprete y el sistema de ejecución en tiempo real (run-time). Además es considerado un lenguaje robusto, fiable y seguro, para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución.

Soporta la sincronización múltiple de hilos de ejecución (*multithreading*), lo cual da la posibilidad que un hilo puede ocuparse de interactuar con el cliente y otro de realizar una operación. Otra de las tantas características que lo hacen idóneo para su utilización, es que se utiliza para dos tipos de programas, aplicaciones independientes y applets.

1.2.5.2. Entorno de desarrollo integrado Eclipse

Eclipse es un entorno de desarrollo integrado (IDE) de código abierto independiente de una plataforma, es una aplicación de cliente enriquecido, emplea *plug-ins* para proporcionar toda su funcionalidad, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Eclipse es, en el fondo, únicamente un armazón sobre la que se pueden montar herramientas de desarrollo. La arquitectura de *plug-ins* permite, además de integrar diversos lenguajes, introducir otras aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo, tales como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, entre otros.

1.3. Sistema Bancario

En la introducción del capítulo se señalaba “la Banca determina y controla la vida económica del mundo” eso va relacionado con que toda actividad industrial, comercial y personal está influenciada por la Banca, desde el momento en que un negocio se proyecta, se ve la necesidad de contar con los servicios del banco. Los bancos por su parte constituyen los entes indiscutibles del sistema bancario, realizan operaciones como prestatarios y prestamistas de crédito, reciben y concentran en forma de depósitos los capitales captados para ponerlos a disposición de quienes puedan hacerlos fructificar.

En la actualidad el sistema bancario enfrenta retos diariamente, entre los cuales está el que implica relacionarse con millones de clientes, basándose en la planificación de captura de datos individuales y el cálculo de un importante conjunto de variables y perfiles de propensión que conforman una base de conocimientos, destinada a personalizar tanto la oferta como el trato otorgados a cada cliente. Otro de los retos es que para todo banco, independientemente de su tamaño, es imprescindible superar la presión a la que se enfrentan a diario para lograr disparar sus beneficios y ser más efectivo. Además es muy competitiva y compleja la red que se ha creado entre los Bancos, Cajas y demás Servicios Financieros que existen, incluso las nuevas generaciones y futuros clientes, totalmente integrados a las nuevas tecnologías, también fuerzan a las entidades financieras a innovar, integrar tecnología y ser cada día más competitivas y eficientes.

En resumen la situación que enfrenta el sistema bancario exige contar con un buen sistema informático capaz de gestionar toda la información necesaria y ayudar en el desarrollo de las operaciones**.** Para desarrollar un sistema capaz de informatizar las actividades contables y el resto de las operaciones de un

banco, es imprescindible conocer a fondo su funcionamiento. Desde un inicio el banco tiene la función de captar pasivos a clientes, los cuales de una manera bien planificada se utilizan en la adquisición de intereses para la entidad. De la misma manera se realizan un grupo de operaciones tales como préstamos, depósitos, transferencias, una serie de pagos y otras operaciones. Además de las muchas negociaciones, ya sea entre clientes cercanos o a disímiles distancias que también son apoyadas por los bancos. En cuanto a la estructura física de un banco, se puede plantear que cada uno se estructura de la manera más conveniente, contando con una serie de departamentos, en los cuales lleva a cabo las operaciones de una manera coordinada.

1.3.1. Proceso Bancario

Los procesos bancarios se pueden comprenden a partir del estudio de los objetivos generales y específicos que tiene un banco como entidad financiera. La atención personalizada a los clientes y la búsqueda incesante de ingresos, resultan algunos de ellos, además de promover su actividad como instrumento, para el desarrollo del Comercio Exterior y sostener las relaciones con otros bancos.

Las operaciones bancarias son todas aquellas operaciones de crédito practicadas por un banco de manera profesional. Las mismas se clasifican en activas, pasivas y neutras. Las operaciones activas son aquellas en las que el banco otorga el crédito, como ejemplos de ellas se tienen los préstamos y los descuentos. Las operaciones pasivas son aquellas en las que el banco recibe dinero de clientes y paga intereses por esas prestaciones, como ejemplos de operaciones pasivas se tienen las Cuentas Corrientes, las de Ahorro, a Plazo Fijo y Cédulas Hipotecarias. Por su parte las operaciones neutras o accesorias son aquellas donde el banco no recibe ni otorga crédito, como las operaciones de mediación, donde el banco solo sirve de intermediario (7). Todas esas operaciones en su conjunto hacen posible que un banco mantenga su estabilidad como entidad financiera.

1.3.1.1. Préstamos Bancarios

Todo préstamo se efectúa entre un prestamista, quien da a préstamo el dinero, y un prestatario, quien lo recibe, originando una deuda de este último ante el primero. Los bancos y otras instituciones financieras asumen generalmente el papel de prestamistas, captando recursos que luego ofrecen a los interesados. También pueden prestarse bienes físicos, aunque en este caso suele hablarse por lo regular de un

contrato de arrendamiento, donde el pago del alquiler correspondiente sustituye a los intereses. Los prestatarios son generalmente empresas que requieren recursos de capital para mantener, desarrollar o ampliar sus actividades; personas que desean disponer de sumas relativamente grandes -con respecto a sus ingresos- para la adquisición de bienes, o gobiernos que buscan recursos para el pago de sus compromisos más allá del límite de sus ingresos corrientes. La economía moderna se basa en gran medida en la existencia de un enorme número de préstamos de diversos tipos y magnitudes. Ellos serían imposibles si no existiesen instituciones que, como los bancos, realizan la función especializada de concentrar y hacer circular el capital**.**

1.3.2. Sistema Bancario Cubano

El sistema bancario cubano ha sufrido con el transcurso de los años una gran modernización: durante 1995 y 1996 el esfuerzo principal se concentró en la automatización de las 500 sucursales y demás oficinas bancarias, instalando en la totalidad de ellas redes locales con computadoras personales en prácticamente todos los puestos de trabajo. Durante 1997 y 1998 el centro de atención estuvo en la interconexión de las sucursales a la Red Pública de Transmisión de Datos. A finales de 1998 los bancos cubanos que realizan transacciones internacionales quedaron conectados a SWIFT. Actualmente en Cuba se cuenta con nueve bancos: Banco Central de Cuba (BCC), Banco Nacional de Cuba (BNC), Banco de Crédito y Comercio (BANDEC), Banco Popular de Ahorro (BPA), Banco Financiero Internacional, S.A. (BFI), Banco Internacional de Comercio S.A. (BICSA), Banco Metropolitano S.A., (BM), Banco de Inversiones, S.A. y Banco Exterior de Cuba, además existen oficinas de representación de bancos extranjeros en Cuba, que operan bajo licencia de BCC.

Entre los bancos cubanos podemos citar el Banco Nacional de Cuba (BNC), el cual una vez liberado de las funciones de banca central y de rector del sistema bancario, continúa existiendo con el carácter de banco comercial, autorizado a ejercer funciones inherentes a la banca universal, teniendo además la función de registro, control, servicio y atención de la deuda externa que el Estado y el propio banco han contraído con acreedores extranjeros con la garantía del Estado. La situación presente en el sistema bancario señala la necesidad imperiosa de usar una herramienta que les permita conocer, controlar, analizar resultados y tomar decisiones que los sitúen en la competencia, papel este de la Contabilidad de la Gestión Bancaria, que no es más que un sistema que se ocupa de la captación, medición y valoración

de la circulación interna de la empresa y a la vez suministra a los diferentes directivos de la organización, la información suficiente y relevante para la toma de decisiones.

1.4. Sistemas Contables

Para comprender la importancia de un sistema contable se debe conocer el significado de contabilidad: la misma registra, clasifica y resume en forma propia y en términos monetarios, las operaciones económicas que acontecen en una entidad y por medio de ella, se interpretan los resultados obtenidos. No constituye un fin en sí misma, sino que representa un medio para llegar a uno o más fines. Se plantea que la contabilidad como ciencia tiene su objeto en el estudio cualitativo y cuantitativo del patrimonio, tanto en su aspecto estático como dinámico, con la finalidad de lograr la dirección adecuada de los bienes que la integran. Por su parte para los sistemas contables se presentan una serie de definiciones tales como: sistemas de recopilación, registro, procesamiento y reporte de todas las transacciones financieras; conjunto de reglas, principios, mecanismos, cuentas, procedimientos, libros y registros de contabilidad, enlazados y relacionados de tal manera entre sí, que permitan analizar, comprobar, asentar y resumir las operaciones practicadas, con el mínimo de esfuerzo y el máximo de precisión; conjunto de principios y reglas que facilitan el conocimiento y la representación adecuada de la empresa y de los hechos económicos que afectan a la misma**.**

Todo sistema contable tiene tres funciones fundamentales, la primera consiste en crear un registro sistemático de la actividad diaria de la entidad, esta actividad se realiza a través de la ejecución de diferentes transacciones que tienen un soporte documental y estos documentos se expresan en términos monetarios y son la base de las inscripciones en los libros de contabilidad. La segunda función consiste en clasificar la información, el registro completo de todas las actividades de una entidad implica un gran volumen de datos, de ahí la necesidad de clasificar la información en grupos o categorías. Como tercera función de un sistema contable se tiene resumir la información, pues para que la información pueda ser utilizada por quienes toman decisiones, esta debe estar muy resumida y suministrar con claridad y precisión la situación financiera de la entidad en un momento determinado, así como los resultados de las operaciones (8). Los tres pasos anteriores constituyen los medios que se utilizan para lograr la información contable. El proceso contable involucra además la comunicación de esta información a los jefes y especialistas interesados en la misma, así como a los usuarios externos. El análisis e

interpretación de esta información orienta las decisiones a los diferentes niveles de dirección, que es en última instancia el objetivo final de la contabilidad.

1.4.1. Sistemas contables en entidades bancarias

Para la contabilidad en entidades bancarias, no existe un modelo único de contabilidad de gestión, sino que cada institución debe fijar sus fines y sus objetivos, debe aclarar para qué quiere el modelo de gestión y adaptarlo o adecuarlo a su organización, para ello deberá considerar factores claves tales como su tamaño, la diversificación del negocio y los mercados geográficos donde está presente (9). Actualmente el mundo de los negocios avanza a pasos agigantados, y este movimiento arrollador va de la mano con los cambios que surgen en la tecnología, entre ellos: las nuevas demandas de información, los cambios sociales, culturales y económicos existentes en este nuevo entorno. La gestión bancaria depende totalmente de la gestión de su información contable, por ello la necesidad de contar con buenos sistemas informáticos, que permitan llevar a cabo una gestión rápida y eficaz de las operaciones.

1.4.1.1. SABIC en el Banco Nacional de Cuba (BNC)

En particular, el Banco Nacional de Cuba, constituye una de las entidades bancarias en la cual, la contabilidad es su actividad fundamental y para ejecutar sus funciones requiere del procesamiento de un gran volumen de datos, por ello ha venido utilizando el sistema contable SABIC en el control y dinamismo de sus operaciones. El SABIC (Sistema Automatizado para la Banca Internacional de Comercio) es un sistema diseñado y desarrollado por la Dirección de Sistemas Automatizados del Banco Central de Cuba para satisfacer las necesidades de procesamiento de datos de bancos e instituciones no bancarias, utilizando los medios técnicos de computación disponibles en el mercado.

Este sistema ha sido adaptado a los requerimientos de las operaciones propias del Banco Nacional y ha sido desarrollado para que los empleados que hagan uso de él, puedan tramitar sus operaciones y realizar sus consultas sin necesidad de acudir a los archivos ni a la actividad manual -algo que no ha sido logrado totalmente-, aumentando la seguridad, la eficiencia del trabajo y la productividad de los trabajadores. Entre las características fundamentales del sistema SABIC se encuentran: la contabilización multimoneda, que permite contabilizar los activos y pasivos en sus monedas de origen, sin tener que realizar las conversiones correspondientes, lo cual aumenta en exactitud la información sobre la posición financiera

de la entidad; la contabilización en tiempo real, que no es más que la actualización de los ficheros contables justo en el momento de hacer las operaciones; la transaccionalidad, con la cual las operaciones pueden realizarse usando transacciones tipificadas que crean asientos automáticamente.

Otra característica importante es la modularidad, pues permite que además de los módulos que trae el sistema y que pueden ser enriquecidos, puedan incorporarse otros módulos nuevos, según las demandas que tenga cada entidad en particular.

A pesar de las grandes ventajas que brinda el sistema SABIC, tiene como plataforma FoxPro sobre MS-DOS, corriendo en un servidor NOVELL 3.12 que no está preparado para dar acceso remoto a los usuarios, no permite conexiones remotas. No es posible además obtener reportes de salida dinámicos y no es posible crearlos por la plataforma donde corre. Por otro lado el sistema SABIC no es compatible con Windows que es el sistema operativo que tienen instalado los trabajadores del BNC (10).

1.4.1.1.1. Nueva versión del SABIC para el Banco Central de Cuba (BCC)

Debido a las limitaciones que presenta el sistema SABIC se decidió realizar en el Banco Central una nueva versión para un ambiente cliente-servidor. Se escogió como lenguaje para programar al cliente a Visual FoxPro y como gestor de base de datos SQLServer.

Todas las tablas que utilizaba la versión anterior se integraron en una Base de Datos que se traspasó para conformar la Base de Datos SQL actual y se estudiaron todos los procedimientos que incluían acceso a dichas tablas, de modo que, el mayor conjunto de ellos pasaran a conformar procedimientos almacenados dentro de la base de datos.

El hecho de tener estos procedimientos almacenados le brinda al sistema una gran rapidez pues el tráfico en la red se limita sólo, a la solicitud de ejecución del procedimiento, retornándose la respuesta correspondiente (11). A pesar de existir esta nueva versión del SABIC que mejora la situación del Banco Central, sus características no se ajustan a las del Banco Nacional.

1.4.1.3. Otros sistemas contables

1.4.1.3.1. Enterprise Resorces Planing (SAP)

SAP es un ERP muy usado en el mundo especialmente en los bancos, posee funcionalidades muy útiles para los procesos que se llevan a cabo, una de ellas es que permite a empleados y clientes comunicarse a través de call centers, portales Web, personalmente y a través de otros canales de comunicación. Ofrece datos actualizados de los clientes a partir de todas las áreas de negocio, permite crear programas de marketing específicos y medir su efectividad, ayuda a monitorizar y a gestionar el ciclo completo de una cuenta o contrato, facilita el establecimiento de intereses, costes y fechas valores.

Por otra parte SAP procesa de manera eficaz balances de cuentas, concentración de efectivo, extractos y otras tareas claves, analiza la rentabilidad de los canales de distribución, de las unidades organizativas, de los productos y de los clientes, permite una gestión total de los riesgos bancarios tanto para la gestión comercial como no comercial, permite planificar el negocio a través de la gestión de activos y pasivos, posibilita una administración y mantenimiento de los contratos, así como la gestión de los procesos contables en las diferentes áreas. A pesar de todas las posibilidades que brinda el ERP SAP, en Cuba se pretende informatizar la mayor cantidad de entidades posibles con software cubano, sin necesidad de pagarlo al extranjero y con mayor confiabilidad. Por ello en esta investigación se estudian las opciones que brinda el SAP con el objetivo de aplicar esos conocimientos en el desarrollo del módulo para el sistema del Banco Nacional cubano.

1.5. Conclusiones Parciales

En el presente capítulo se expone un análisis de los sistemas bancarios y de los sistemas contables existentes en la actualidad, además se aborda la necesidad de un nuevo sistema para el Banco Nacional de Cuba, que se ocupe del área de Préstamos. Se puede consultar sobre la metodología de desarrollo de software, el lenguaje de modelado, y las restantes herramientas a utilizar para la realización del análisis y diseño del Subsistema de Préstamos del proyecto Modernización del Sistema Bancario Cubano.

CAPÍTULO 2: ANÁLISIS Y DISEÑO

2.1. Introducción

En el presente capítulo se realiza el análisis y diseño de la propuesta de solución. Se muestran los artefactos generados y se mencionan alguna de las tantas terminologías que se utilizan. Es importante destacar que aunque RUP contempla Análisis y Diseño en la misma disciplina por estar muy relacionadas, son actividades diferentes con artefactos diferentes.

Cabe señalar que en este capítulo se hace alusión a los requisitos que fueron capturados en la tesis de los autores Yoan Antonio López Rodríguez y Yulier Matías León “, titulada Definición de los requerimientos funcionales del módulo tesorería, préstamos y depósitos del proyecto banco nacional”, debido a que en el análisis se debe conseguir una comprensión más precisa de los requisitos, refinarlos y estructurarlos.

2.2. Análisis

“Durante el análisis, se analizan los requisitos que se describen en la captura de requerimientos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que nos ayude a estructurar el sistema entero, -incluyendo su arquitectura” (5).

De forma general el análisis consiste en transformar los requisitos funcionales en un diseño de clases, en el cual se vean las relaciones e interacciones que existe entre las clases. Teniendo presente en este proceso una arquitectura robusta, que permita adaptar el sistema al entorno de implementación que se está desarrollando. Además en este flujo se obtiene una visión del sistema que se preocupa de ver que hace, de tal forma que se preocupa solo por los requisitos funcionales. Los cuales se listan a continuación:

1 RF1. Gestionar préstamo.
   1.1. Registrar préstamo.
   1.2. Actualizar préstamo.
   1.3. Buscar préstamos concedidos
   1.4. Consultar Préstamo Concedido.
2 RF2. Gestionar vencimientos de préstamos.
   2.1 Buscar vencimientos de préstamos.

2.2 Consultar vencimientos de préstamos.

2.3 Liquidar vencimientos de préstamos.

2.4 Poner vencido un vencimiento de préstamos.

3 RF3. Gestionar solicitud de préstamo.

3.1. Registrar solicitud de préstamo.

3.2. Eliminar solicitud de préstamo.

3.3. Consultar solicitud de préstamo.

3.4. Actualizar solicitud de préstamo.

3.5. Buscar solicitud de préstamo.

4 RF4. Gestionar contrato de préstamo.

4.1. Actualizar contrato de préstamo.

4.2. Eliminar contrato de préstamo.

4.3. Buscar contrato de préstamo.

4.4. Consultar contrato de préstamo

4.5. Registrar contrato de préstamo

Cabe recalcar que en el presente trabajo los autores del mismo participaron ocupando los roles de diseñador y arquitecto.

El diseñador es el responsable de diseñar partes del sistema, teniendo en cuenta las restricciones de los requerimientos, la arquitectura y el proceso de desarrollo para el proyecto. Este es el encargado de identificar y definir las responsabilidades, funcionamientos, atributos, y relaciones de los elementos del diseño. Asegura que el diseño es consistente con la arquitectura del software.

La persona que desempeñe este rol debe contar con sólidos conocimientos en cuanto a los requisitos del sistema, la arquitectura del mismo, las técnicas de diseño de software, incluyendo el análisis orientado objeto y el lenguaje unificado de modelado, las tecnologías en las que el sistema será implementado y las líneas bases del proyecto.

El arquitecto de software es responsable de la arquitectura del software incluyendo las decisiones técnicas que restringen el diseño e implementación general del proyecto, se ocupa de la integridad del modelo de análisis y por la arquitectura del modelo de análisis. Debe identificar y documentar los aspectos arquitectónicamente significativos de las vistas de requerimientos, diseño, implementación y despliegue.

2.2.1. Modelo del análisis

El lenguaje que se utiliza en el análisis se basa en un modelo de objetos conceptual, que se denomina modelo del análisis.

El modelo de análisis contribuye a refinar los requisitos, así como a razonar sobre los aspectos internos del sistema. A pesar de que el modelo del análisis hay un refinamiento de los requisitos, no se toman en cuenta aspectos que afectan al sistema como el lenguaje de programación, la plataforma en la que se ejecutará la aplicación, los componentes prefabricados o reusables de otras aplicaciones…el objetivo del análisis es comprender perfectamente los requisitos del software y no precisar cómo se implementará la solución.

El modelo del análisis es descrito por el lenguaje del desarrollador, presenta una vista interna del sistema, es estructurado por clases y paquetes estereotipados; proporciona la estructura a la vista interna. No debería contener redundancias, inconsistencias, entre requisitos. Esboza cómo llevar a cabo la funcionalidad dentro del sistema incluida la funcionalidad significativa para la arquitectura; sirve como una primera aproximación del diseño. Define realizaciones de casos de uso, y cada una de ellas representa el análisis de un caso de uso del modelo de casos de uso.

2.2.1.1. Clases del análisis

Las clases del análisis se centran en los requisitos funcionales y son evidentes en el dominio del problema porque representan conceptos y relaciones del dominio. Tienen atributos y entre ellas se establecen relaciones de asociación, agregación / composición, generalización / especialización y tipos asociativos. RUP propone clasificar a las clases en:

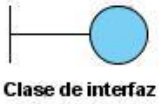
NOMBRE	CARATERÍSTICAS	REPRESENTACIÓN
Entidad	Modelan información que posee larga vida y que es a menudo persistente.	
Interfaz	Modelan la interacción entre el sistema y sus actores.	
Control	Coordinan la realización de uno o unos pocos casos de uso coordinando las actividades de los objetos que implementan la funcionalidad del caso de uso.	

Tabla 1. Estereotipos de las clases del análisis.

2.2.1.2. Realización de los casos de uso en el análisis

La realización de los casos de usos en el análisis es una colaboración como parte del modelo de análisis en el que se muestra cómo se ejecuta un determinado caso de uso en términos de clases del análisis y de sus objetos del análisis en interacción (5). [Ver anexos Descripción de Casos de Usos]

2.2.1.3. Diagrama de clases del análisis

Un Diagrama de clases del análisis es un artefacto en el que se representan los conceptos en un dominio del problema. Representa las cosas del mundo real, no de la implementación automatizada de estas cosas.

A continuación los diagramas de clases del análisis:

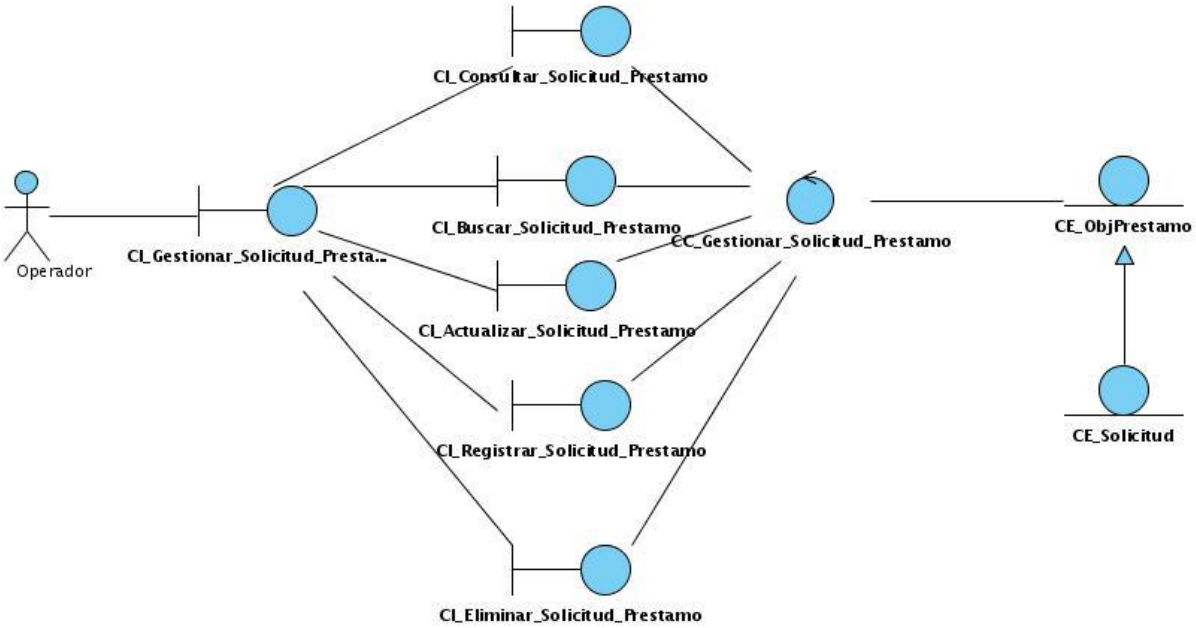


Figura 5. Gestionar solicitud de préstamo.

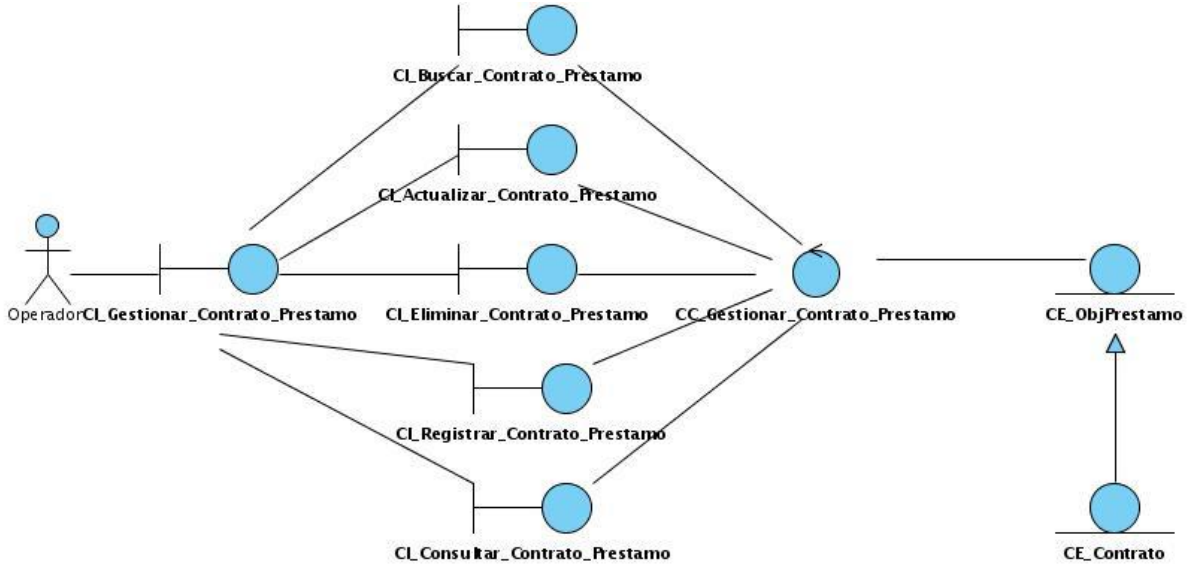


Figura 6. Gestionar contrato de préstamo.

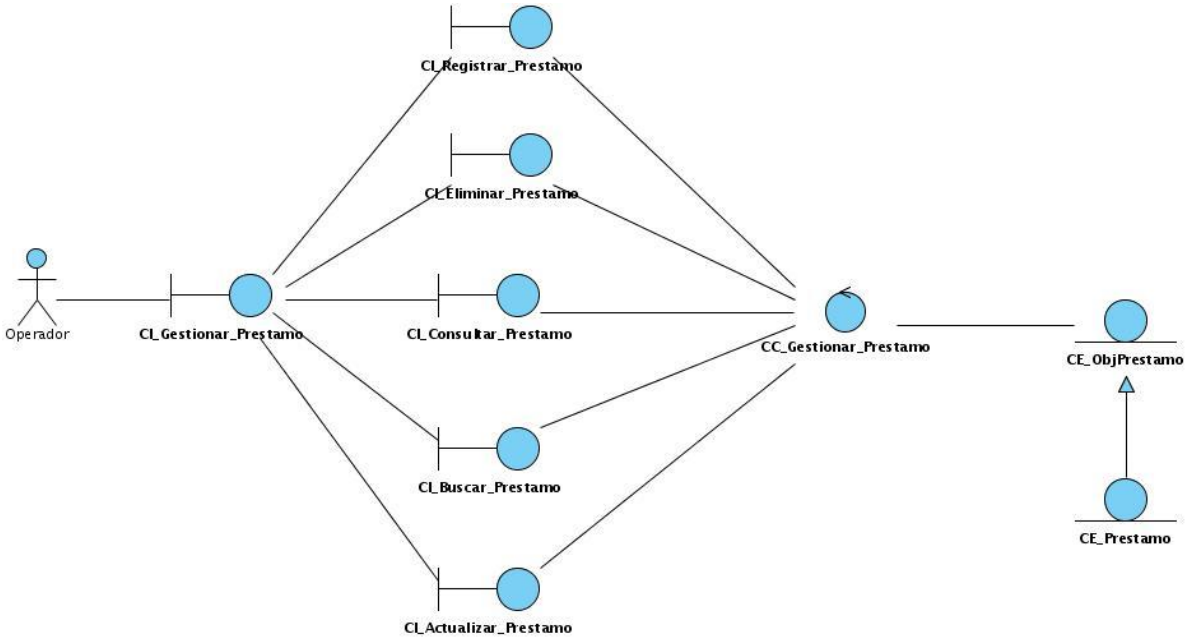


Figura 7. Gestionar préstamo.

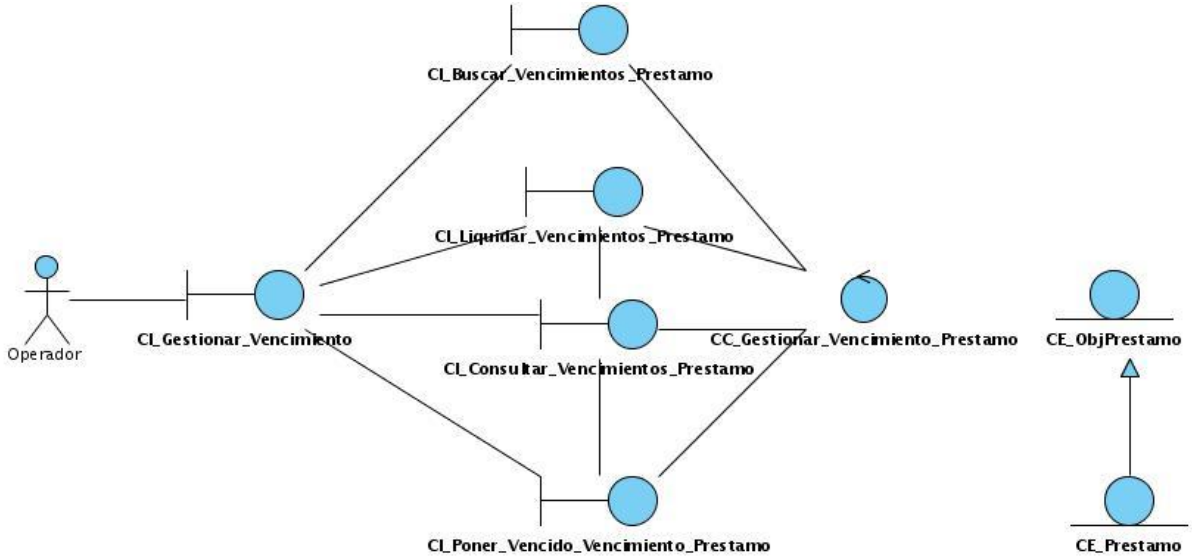


Figura 8. Gestionar vencimiento de préstamo.

2.2.1.4. Diagrama de interacción (colaboración)

Un diagrama de colaboración muestra una interacción organizada en torno a los objetos que efectúan operaciones. Es parecido a un diagrama de objetos que muestra los objetos y los enlaces existentes entre ellos que se necesitan para implementar una operación de nivel más elevado.

Una colaboración modela los objetos y los enlaces significativos dentro de una interacción. Los objetos y los enlaces son significativos solamente en el contexto proporcionado por la interacción. Un rol describe un objeto, y un rol en la asociación describe un enlace dentro de una colaboración.

Un diagrama de colaboración muestra los roles en la interacción en una disposición geométrica. Los mensajes se muestran como flechas, ligadas a las líneas de la relación, que conectan a los roles. La secuencia de mensajes, se indica con los números secuenciales que preceden a las descripciones del mensaje.

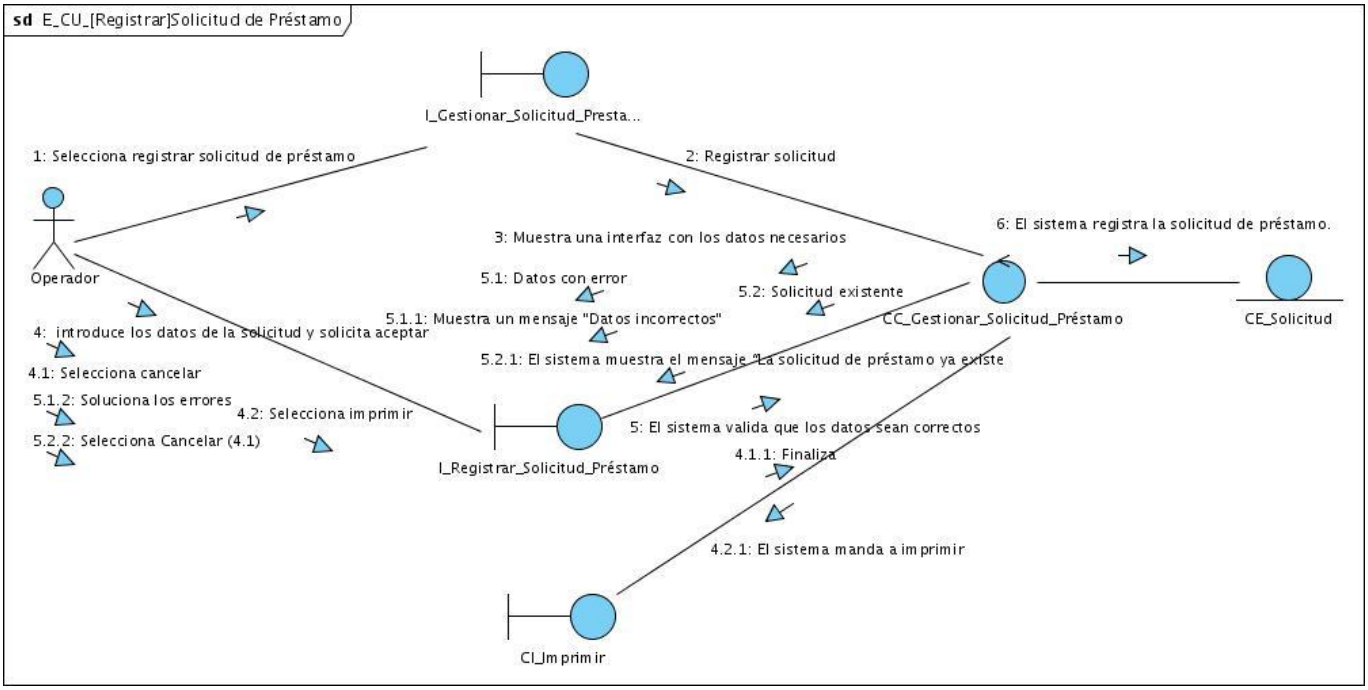


Figura 9. Diagrama de colaboración CU Registrar solicitud.

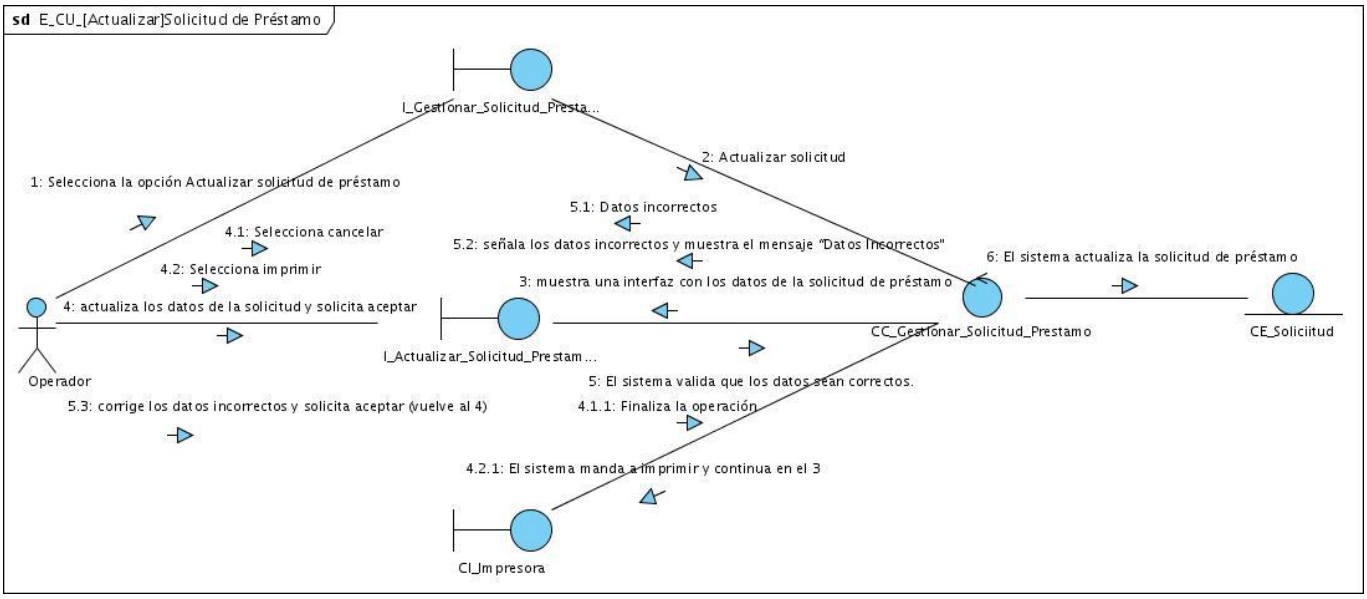


Figura 10. Diagrama de colaboración CU Actualizar solicitud.

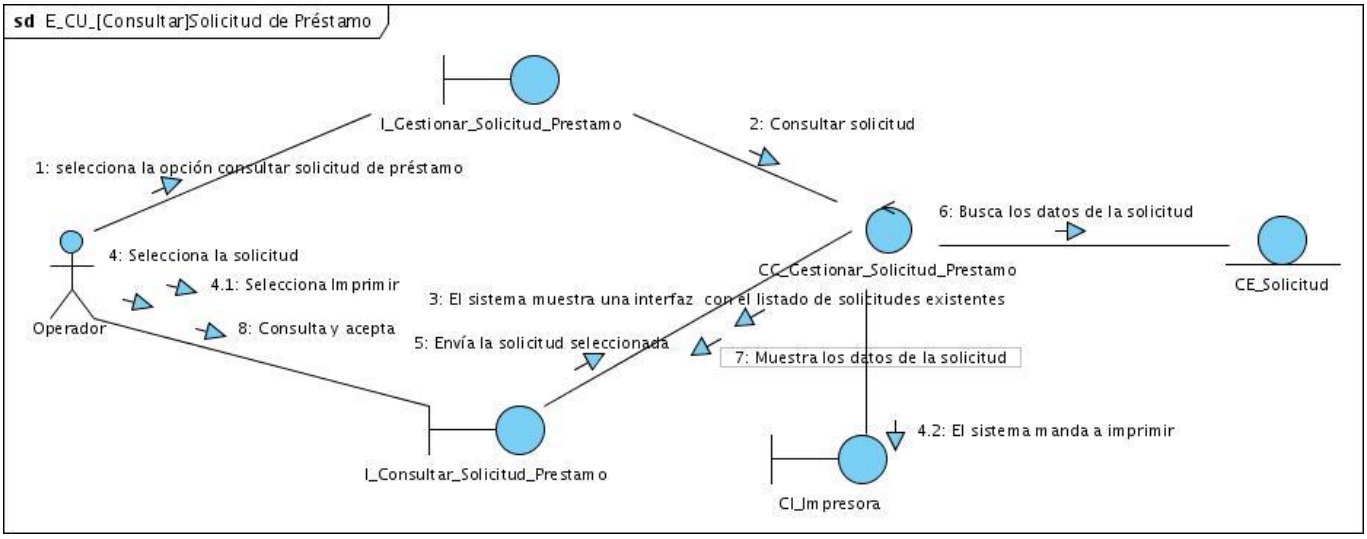


Figura 11. Diagrama de colaboración CU Consultar solicitud.

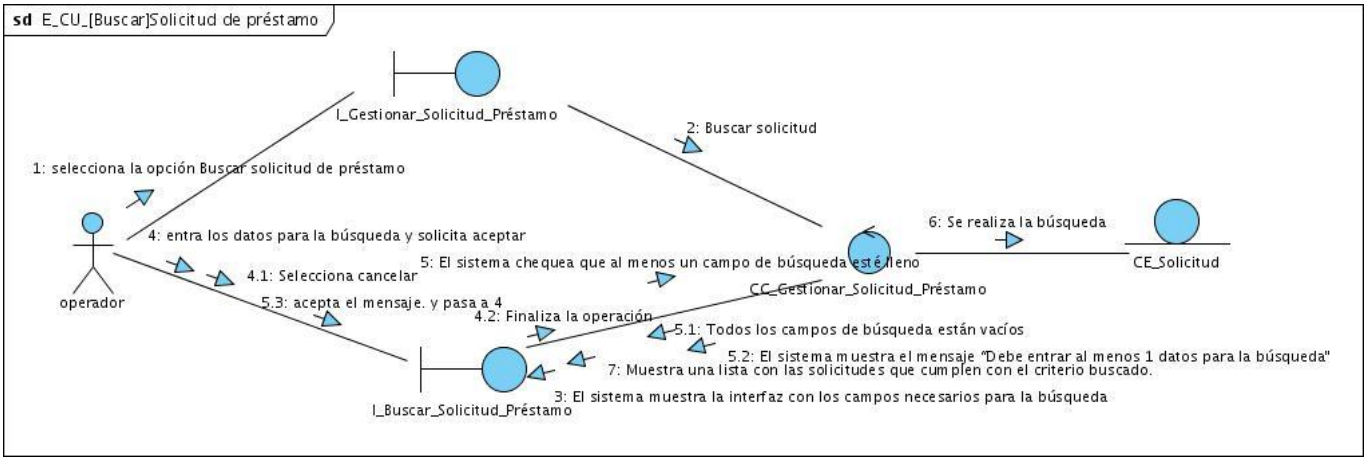


Figura 12. Diagrama de colaboración CU Buscar solicitud.

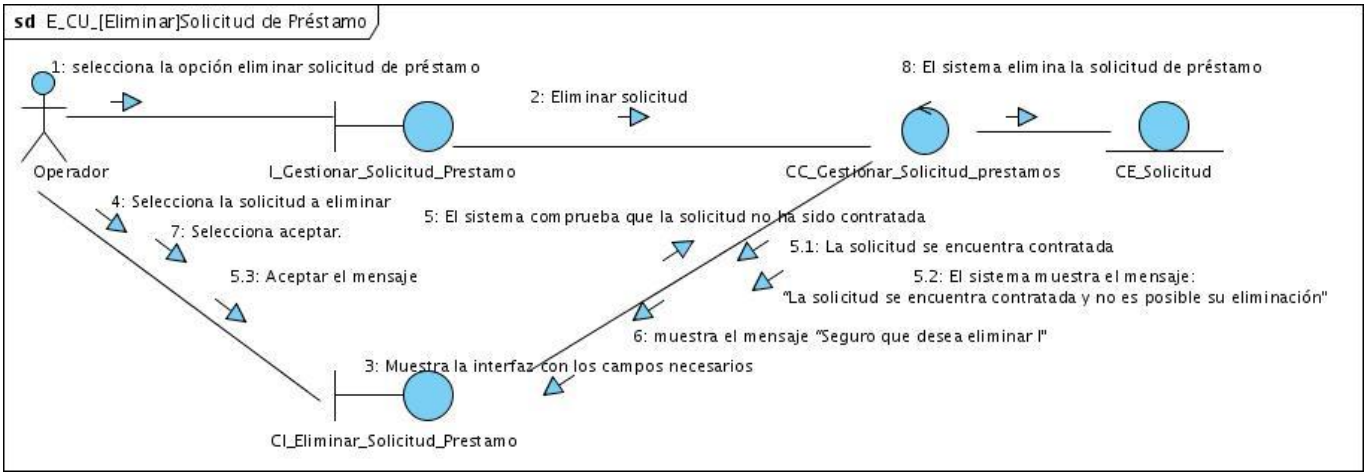


Figura 13. Diagrama de colaboración CU Eliminar solicitud.

2.3. Diseño

RUMBAUGH, JACOBSON y BOOCH, en el libro "El lenguaje unificado de modelado", señalan que el diseño es: "*esa etapa de un sistema que describe cómo se implementará el sistema, en un nivel lógico sobre código real. En el diseño, las decisiones estratégicas y tácticas se toman para resolver los requisitos funcionales y de calidad requeridos de un sistema. Los resultados de esta etapa son representados por*

los modelos a nivel de diseño, especialmente la vista estática, vista de la máquina de estados, y vista de interacción. Contrastar con: análisis, diseño, implementación, y despliegue (6).

De forma general en el diseño se modela el sistema, contribuyendo a una arquitectura estable y sólida, para que soporte todos los requisitos, tanto funcionales como no funcionales. El diseño posibilita una entrada apropiada y un punto de partida para las actividades de la implementación, descompone los trabajos de implementación en partes más manejables que pueden ser llevados a cabos por diferentes equipos de desarrollo.

El diseño del sistema contribuye a una arquitectura estable y sólida en la creación del modelo de implementación. El modelo de diseño se puede utilizar para visualizar la implementación y para soportar las técnicas de programación gráfica.

2.3.1. Descripción de la arquitectura propuesta

2.3.1.1. Módulos, Componentes y Subsistemas.

Los Componentes son un conjunto de funcionalidades comunes que serán reutilizados por otros módulos del sistema. Estos componentes en algunas ocasiones se comportarán como módulos visuales en el sistema, y en otras ocasiones solamente recogerán funcionalidades del negocio.

Los Módulos agruparán un conjunto de Casos de Uso relacionados con uno o más procesos bancarios estrechamente relacionados.

Los Subsistemas agruparán un conjunto de Módulos que estén relacionados con los procesos que ejecutan.

Los Subsistemas, Módulos y Componentes se definirán según las funcionalidades identificadas en el levantamiento de requisitos. Con el objetivo de lograr un sistema flexible, robusto y adaptable, se definirán los componentes genéricos que encapsulen procesos o actividades generales y luego se definirán otros módulos que respondan a procesos específicos utilizando estos componentes.

Diseño de las capas lógicas

Para ganar en organización en el desarrollo y en el despliegue del sistema, se agruparán los Módulos y Componentes por Subsistema, cada Subsistema tendrá uno o más Módulos y/o Componentes

estrechamente relacionados con las funcionalidades que ejecutan. Los Módulos y/o Componentes estarán separados por diferentes capas lógicas según la naturaleza de los mismos.

Las capas lógicas definidas son:

Capa de Presentación:

Esta capa estará dividida en dos partes. Una subcapa del lado del servidor, encargada de recibir todos los pedidos de la interfaz de usuario, controlar el flujo de presentación del sistema y enviar las respuestas correspondientes a la interfaz de usuario. La otra subcapa estará en el cliente, utilizándose los componentes visuales de Java Script para manejar los eventos y validaciones del lado del cliente. La subcapa colocada en el lado del servidor estará relacionada con la capa de Negocios y de Dominio.

Capa de Negocios:

Esta capa está divida en dos subcapas principales sin dejar de incluir otras que se necesiten y que estén relacionadas con el negocio. En la Fachada se expondrán todas las funcionalidades que la capa de presentación necesitará. Esta capa invocará métodos de la subcapa de Desarrollo del negocio. En la capa de Desarrollo del negocio se implementará el negocio de los módulos en cuestión, y de aquí se accederá de ser necesario a la Capa de Acceso a Datos, a otras Capas de Negocios y/o a la Capa de Dominio.

Capa de Acceso a Datos:

En esta capa se implementarán los métodos encargados en interactuar con el gestor de Base de Datos. Esta capa tendrá solamente dependencia con la Capa de Dominio.

Capa de Dominio:

En esta capa se declararán todas las clases que representan entidades del negocio. Estas clases de dominio estarán presentes en todas las capas anteriormente descritas. A continuación se muestra una figura con la estructura de las capas lógicas descritas anteriormente.

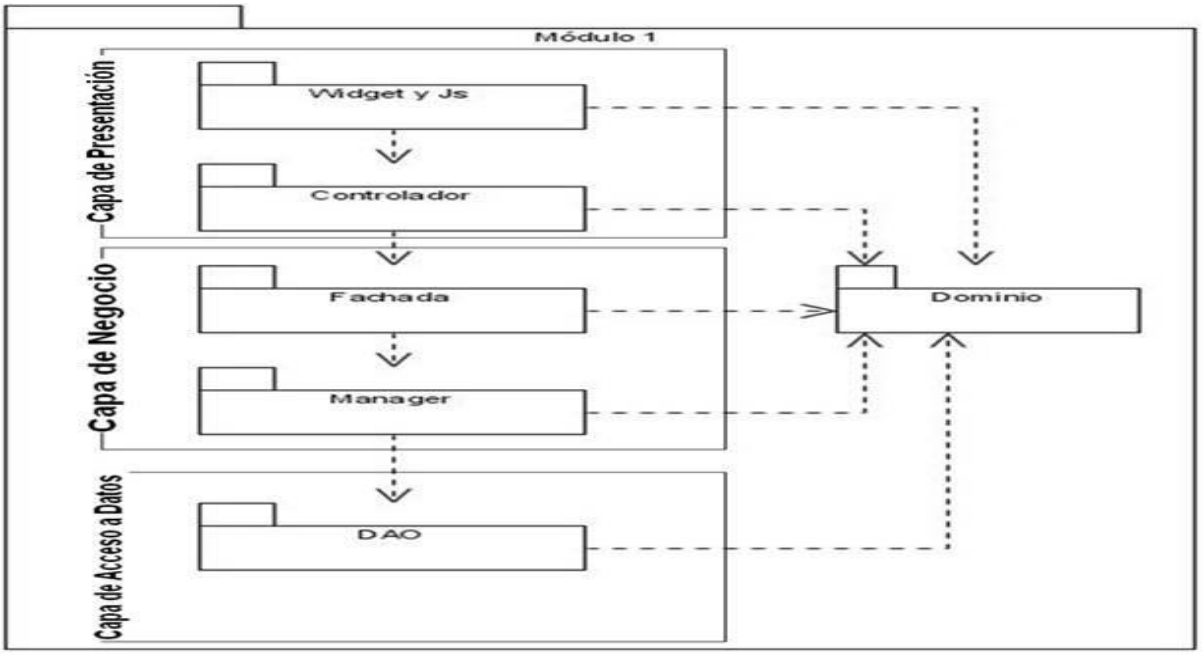


Figura 14. Estructura de las capas lógicas del sistema.

Se muestra otra figura con las capas lógicas y los frameworks que se utilizarán en cada una.

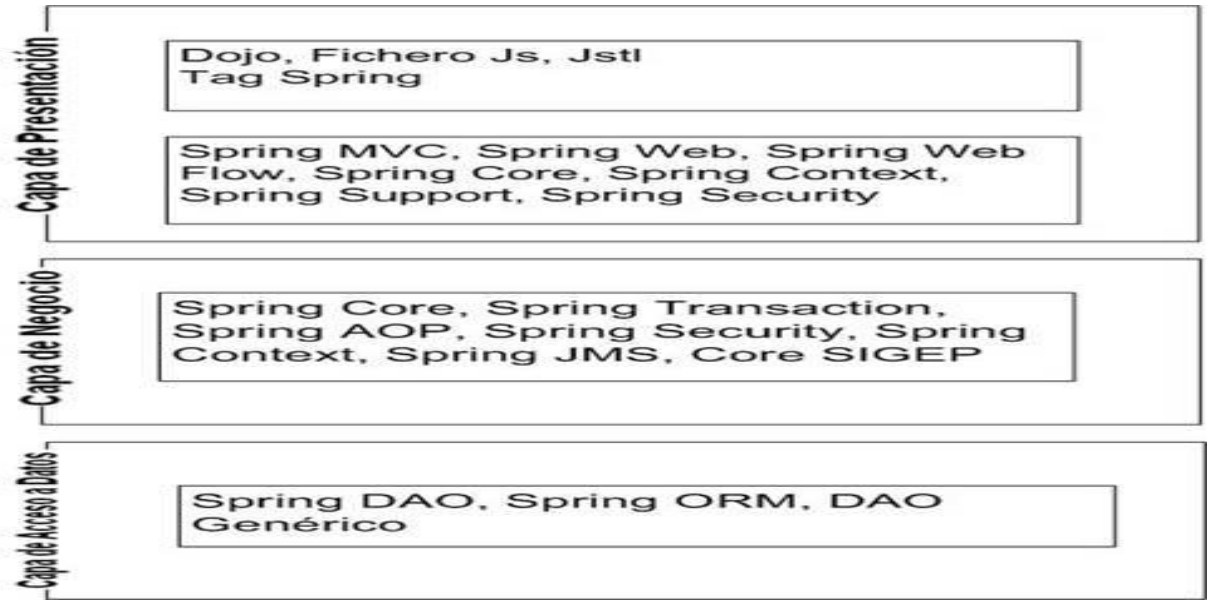


Figura 15. Relación entre capas lógicas y los frameworks.

2.3.1.1. Comunicación entre Módulos y Componentes

Las dependencias entre módulos y componentes están dadas por la necesidad de ejecutar en un módulo, funcionalidades que estén en otro módulo. El módulo que necesite una funcionalidad desarrollada en otro módulo, invocará dicha funcionalidad. A estas funcionalidades se les invocará a través de la capa de negocio del módulo. Por tanto, las dependencias entre módulos se realizarán a través de las capas del negocio.

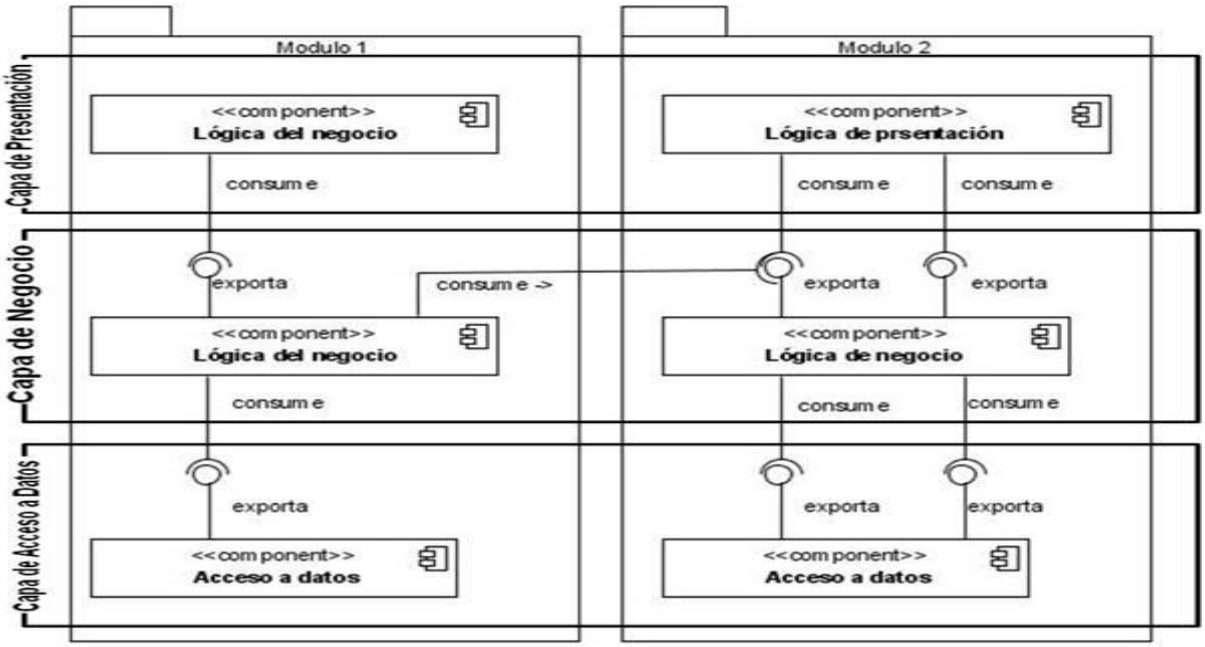


Figura 16. Dependencias entre módulos.

2.3.1.2. Patrones a utilizar

Cada patrón describe un problema que ocurre una y otra vez en nuestro ambiente, y luego describe el núcleo de la solución a ese problema, de tal manera que puedes usar esa solución un millón de veces más, sin hacer jamás la misma cosa dos veces.

2.3.1.4.1. Patrones de asignación de Responsabilidades GRASP

En los patrones GRASP[1] se codifican algunos de los principios, que se aplican al preparar los diagramas de interacción.

- Experto.

La aplicación de este patrón permite a cada clase desarrollar las tareas que pueden realizar según la información que poseen.

- Creador.

Permite crear instancias de otras clases en correspondencia con la responsabilidad dada. Con esto se logró conservar el encapsulamiento ya que los objetos logran valerse de su propia información para realizar lo que se les pide.

- Bajo acoplamiento.

Este patrón soluciona el inconveniente de dar soporte a una dependencia escasa y a un aumento de la reutilización.

- Alta cohesión.

Este patrón es utilizado para mantener la complejidad dentro de los límites manejables.

El diseño obtenido cumple con los patrones de Bajo acoplamiento y Alta cohesión permitiendo la colaboración entre los elementos del diseño, sin verse afectados la reutilización de los mismos y el entendimiento de estos cuando se encuentran aislados.

La creación de clases controladoras facilitó realizar las operaciones del sistema, debido a que estas operaciones reflejan los procesos de la empresa o dominio y no es factible manejarse en la capa de interfaz o presentación.

2.3.1.4.2. Patrones Estructurales

- Facade.

El propósito de utilizar este patrón es proveer de una interfaz unificada y sencilla que haga de intermediaria entre un cliente y una interfaz o grupo de interfaces más complejas. Este patrón permite que una biblioteca de software sea más fácil de usar y entender. Esto es posible porque el facade implementa

[1] ***G****eneral* ***R****esponsibility* ***A****ssignment* ***S****oftware* ***P****atterns*((patrones generales de software para asignar responsabilidades)*, son patrones generales de software para asignación de responsabilidades, aunque se considera que más que patrones, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.*

métodos convenientes para tareas comunes, puede reducir la dependencia de código externo en los trabajos internos, permitiendo así más flexibilidad en el desarrollo de sistemas.

2.3.1.4.3. Patrones Comportamiento

- Command.

El objetivo de utilizar este patrón es tener parametrizados los objetos por las acciones que realizan. Este patrón permite especificar, administrar y ejecutar solicitudes en tiempos distintos.

Puede guardar un estado que permita deshacer la ejecución del comando. Soporta la capacidad de generar bitácoras que permitan la recuperación del estado en caso de que el sistema falle. Facilita la estructuración del sistema en torno a operaciones de alto nivel construidas con base en operaciones primitivas o de bajo nivel. Un comando nos desliga el objeto invocador del objeto receptor. Permite que las acciones sean objetos de primera clase y se puedan agrupar comandos de uso frecuente en comandos compuestos.

2.3.1.4.4. Patrón de acceso a datos (DAO)

- DAO

La utilización de este patrón permitirá acceder a la fuente de datos y encapsular los objetos clientes, ocultando tanto la fuente como el modo de acceder a ella. Los DAOs deben implementar los métodos del interface (InterfaceDAO) que declaran. Pero además pueden implementar otros métodos que no están en el interfaz. DAO permite el acceso a reglas de validación, esto es posible porque tiene capacidad de especificar relaciones entre tablas.

2.3.2. Modelo del diseño

El modelo del diseño es un modelo de objetos que describe la realización física de los casos de usos centrándose en como los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas en el entorno de implementación, tienen un impacto en el sistema a considerar y sirve de abstracción a la implementación y al código fuente del sistema. Contiene protocolo, cápsula, realización de casos de usos, señales, eventos, subsistema de diseño, paquetes de diseño, interfaces, clases del diseño, clases de prueba, diseño de pruebas.

2.3.2.1. Clases del diseño

Una clase de diseño es una construcción similar a una clase en la implementación del sistema.

Extensiones de UML para Web

 Server Page (página servidora): representa la página Web que tiene código, que se ejecuta en el servidor.

 Client Page (página cliente): una instancia de Página Cliente es una página Web, con formato HTML.

 Form (formulario): colección de elementos de entrada que son parte de una página cliente.

<<Build>>: representa una asociación especial que relaciona las páginas cliente con las páginas servidor.

<<Link>>: expresa las asociaciones más comunes entre las páginas, en este caso la del hipervínculo.

<<Submit>>: es la relación que se crea siempre entre una página servidor y un formulario.

2.3.2.2. Realización de los casos de uso en el diseño

[Ver anexos Descripción de Casos de Usos]

2.3.2.2.1. Diagramas de clases del diseño

Los diagramas de clases son los más utilizados en el modelado de sistemas orientados a objetos. Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas.

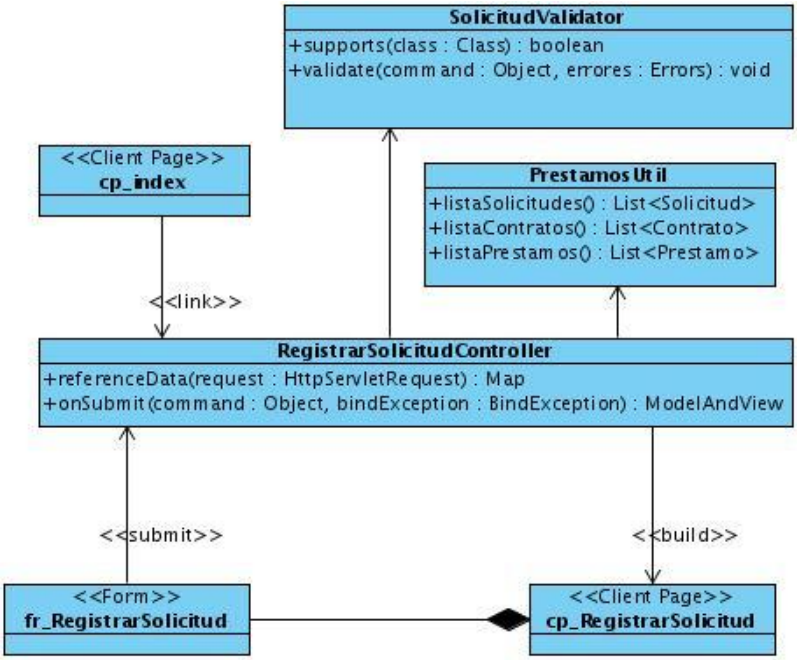


Figura 17. Diagrama de clases del diseño CU Registrar solicitud. Capa de presentación.

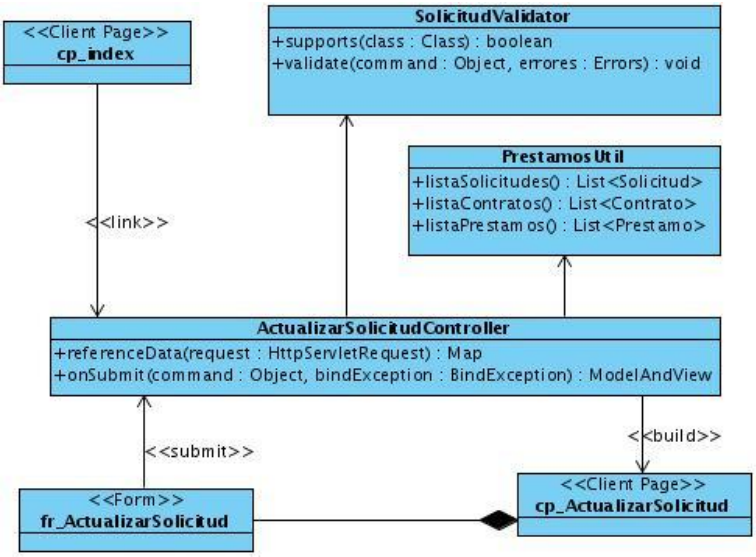


Figura 18. Diagrama de clases del diseño CU Actualizar solicitud. Capa de presentación.

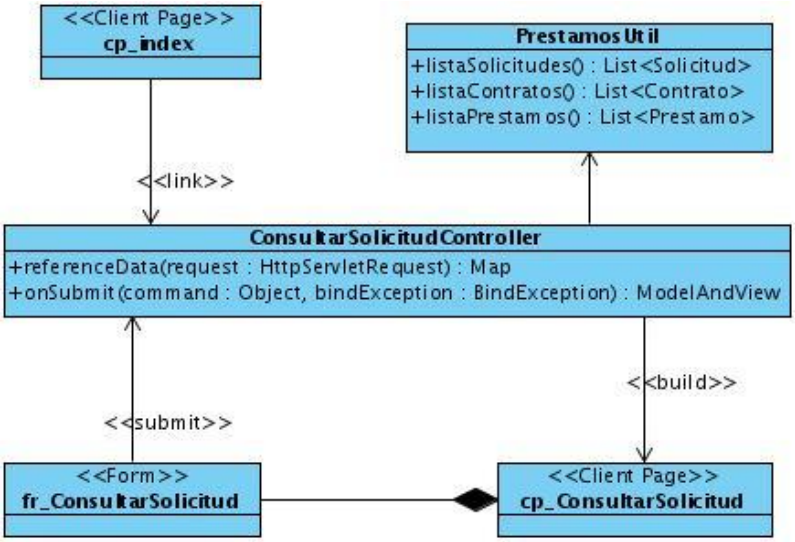


Figura 19. Diagrama de clases del diseño CU Consultar solicitud. Capa de presentación.

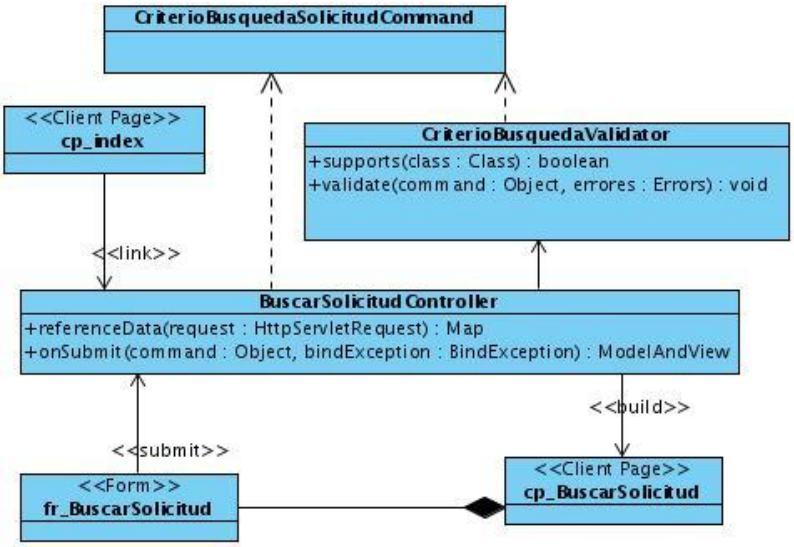


Figura 20. Diagrama de clases del diseño CU Buscar solicitud. Capa de presentación.

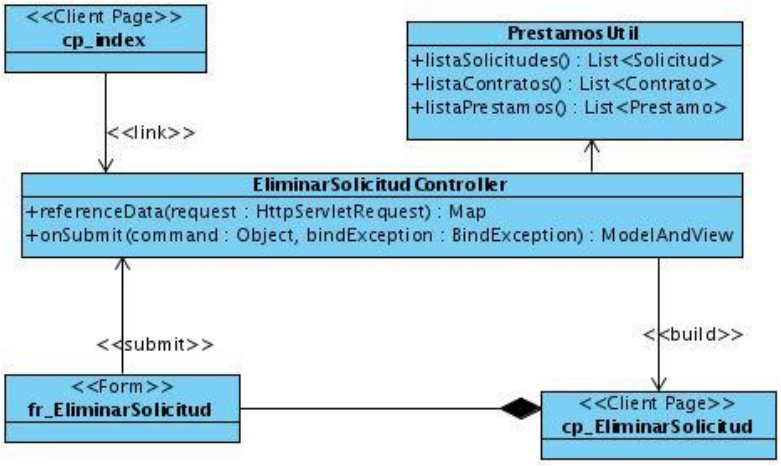


Figura 21. Diagrama de clases del diseño CU Eliminar solicitud. Capa de presentación.

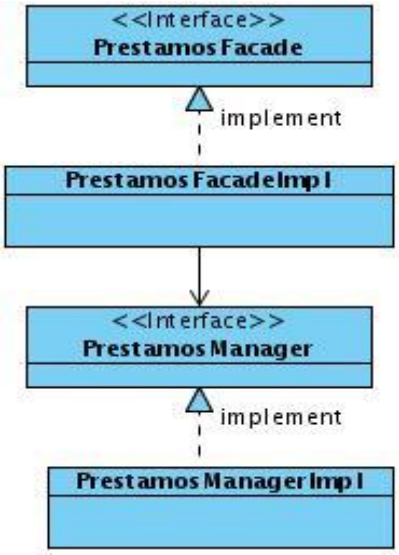


Figura 22. Diagrama de clases del diseño capa de negocio.

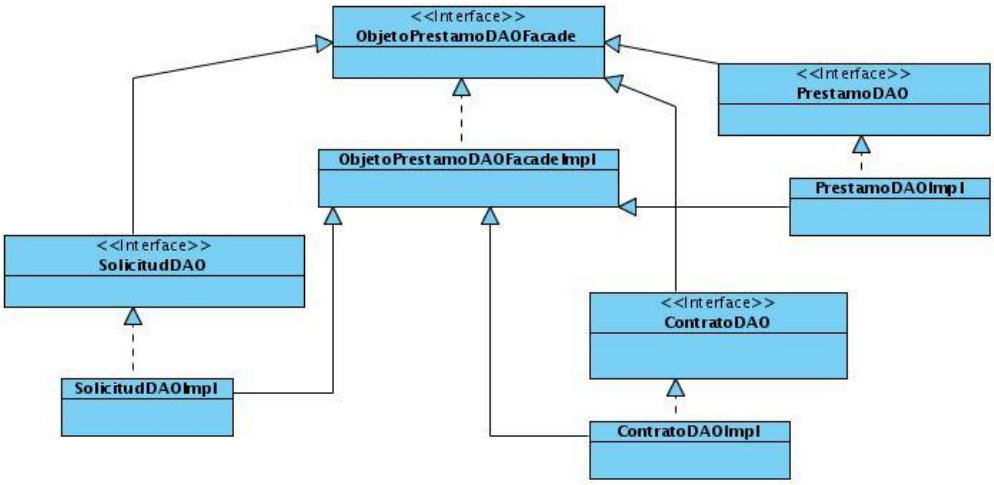


Figura 23. Diagrama de clases del diseño del las clases más representativas. Capa de acceso a datos.

Descripción de las principales clases de la lógica de negocio y de acceso a datos

PrestamosFacade: Interfaz de la capa de lógica de negocio que contiene las funcionalidades relacionadas con el subsistema, brindadas a la capa de presentación y a la vez le sirve de fachada.

PrestamosFacadeImpl: Clase que implementa las funcionalidades definidas en la interfaz "*PrestamosFacade*". En esta no se implementa ninguna lógica de negocio, simplemente se limita a delegar las responsabilidades a la clase Manager correspondiente.

PrestamosManager: Interfaz que define las funcionalidades de lógica de negocio especificas para el subsistema Préstamos.

PrestamosManagerImpl: Interfaz que implementa los métodos de lógica de negocio definidos en la interfaz "*PrestamosManager*". Se encarga de implementar las funcionalidades brindadas por la fachada.

ObjPrestamoDAOFacade: Es una interfaz que agrupa funcionalidades comunes relacionadas con los métodos de acceso a datos en el subsistema para las solicitudes, contratos y prestamos.

ObjPrestamoDAOFacadeImpl: Esta clase implementa la interfaz "*ObjPrestamoDAOFacade"*. En esta clase no se implementa ninguna lógica de acceso a datos, ella sólo se encarga de delegar esas responsabilidades a los DAOs**.**

SolicitudDAO: Es una interfaz que brinda las funcionalidades correspondientes al manejo del acceso a datos para las solicitudes de préstamos.
SolicitudDAOImpl: Esta clase implementa la interfaz "*SolicitudDAO"*. En ella se implementa el manejo del acceso a datos correspondiente a las solicitudes de préstamos.
ContratoDAO: Es una interfaz que brinda las funcionalidades correspondientes al manejo del acceso a datos para los contratos de préstamos.
ContratoDAOImpl: Esta clase implementa la interfaz "*ContratoDAO"*. En ella se implementa el manejo del acceso a datos correspondiente a los contratos de préstamos.
PrestamoDAO: Es una interfaz que brinda las funcionalidades correspondientes al manejo del acceso a datos para los prestamos.
PrestamoDAOImpl: Esta clase implementa la interfaz "*PrestamoDAO"*. En ella se implementa el manejo del acceso a datos correspondiente a los préstamos.

2.3.2.2.2. Diagramas de interacción (secuencia)

Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes (mensaje simple, síncrono, asíncrono, y/o de retorno). Este tipo de diagrama se destaca por tener la línea de vida y el foco de control que representa el período de tiempo durante el cual un objeto ejecuta una acción. A continuación los Diagramas de secuencias por escenarios del caso de uso.

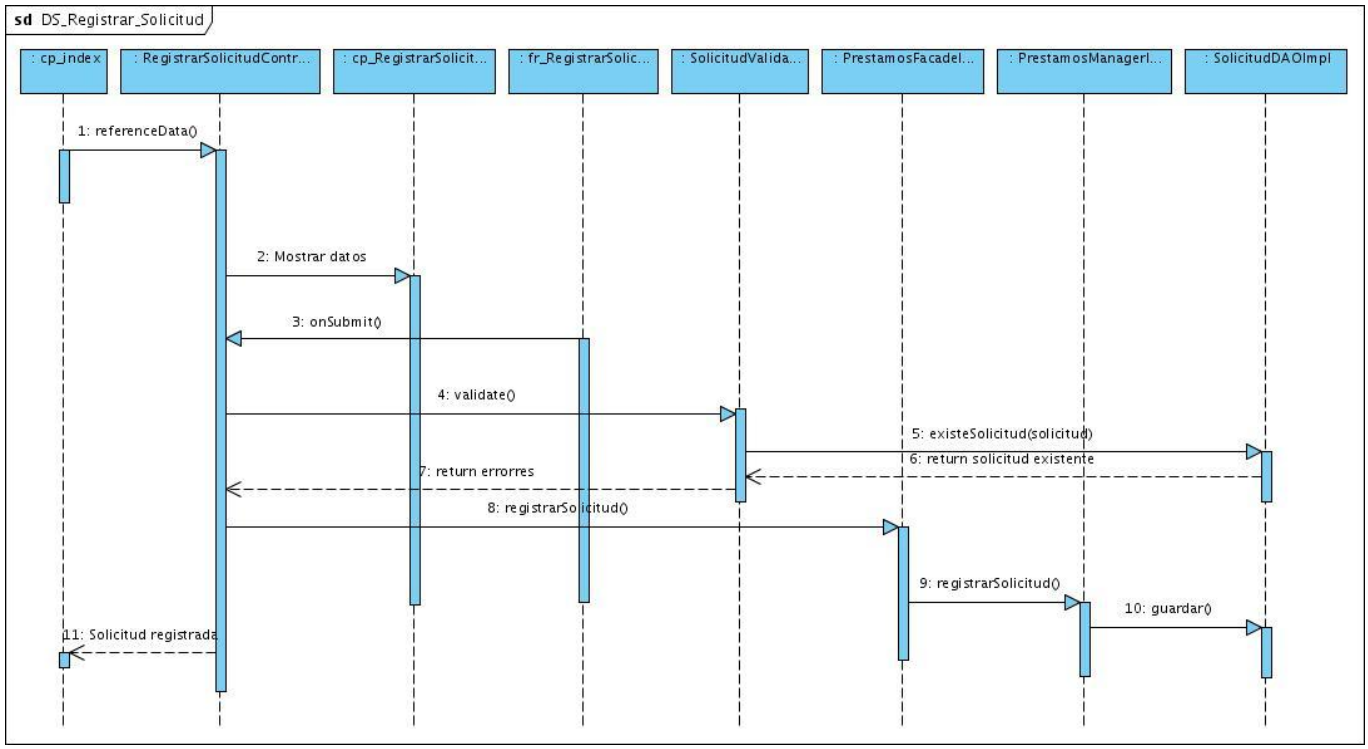


Figura 24. Diagrama de secuencia CU Registrar solicitud. Flujo básico.

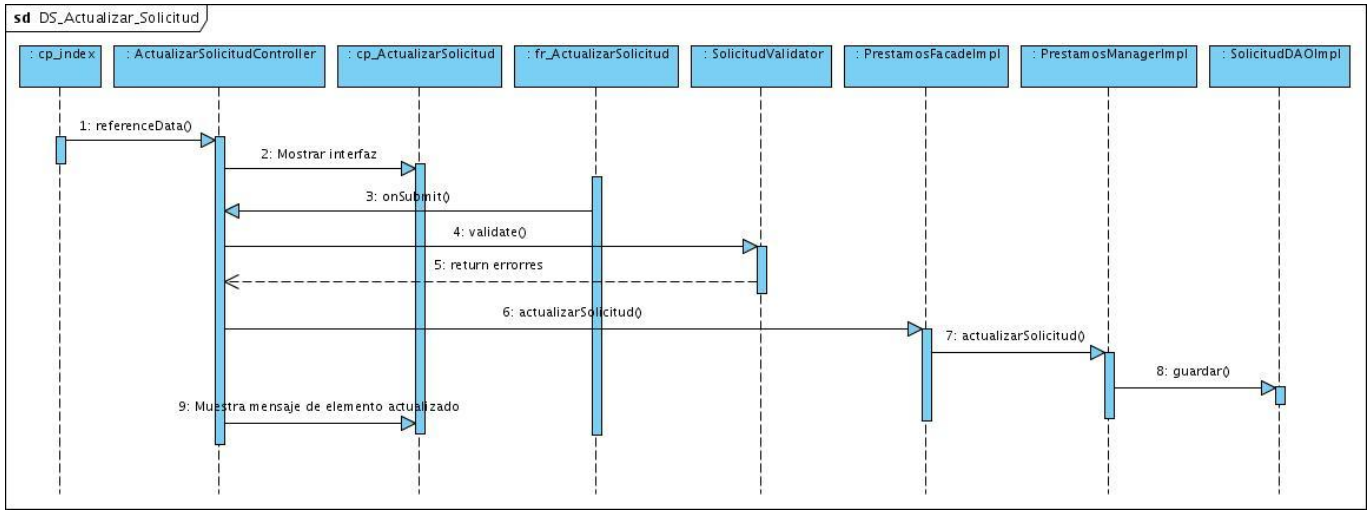


Figura 25. Diagrama de secuencia CU Actualizar solicitud. Flujo básico.

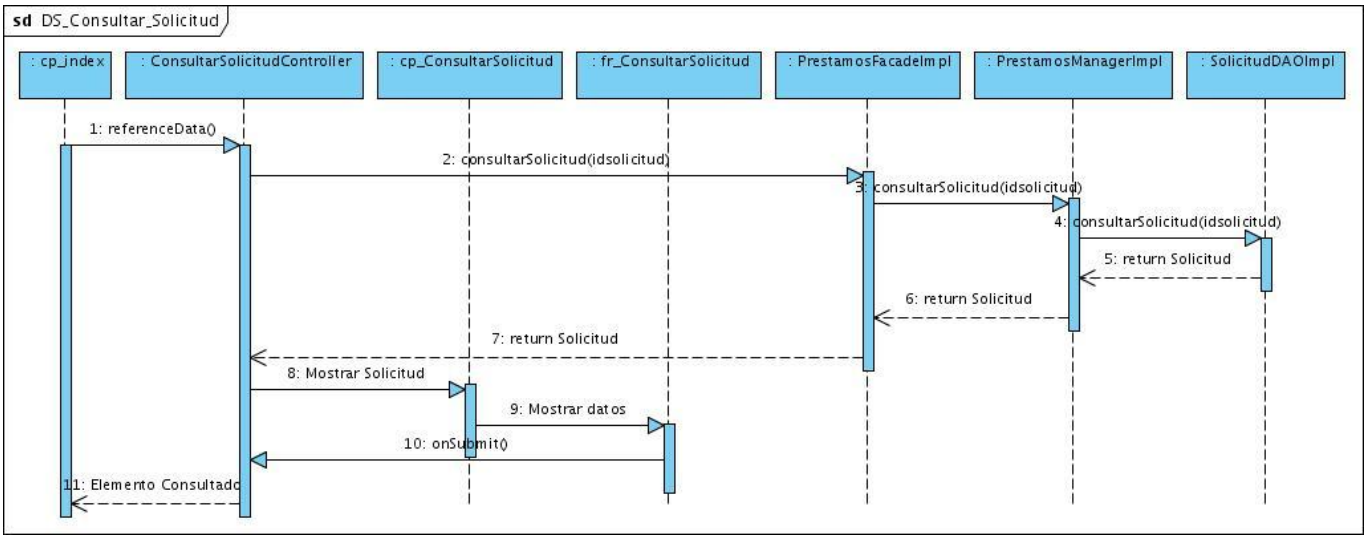


Figura 26. Diagrama de secuencia CU Consultar solicitud. Flujo básico.

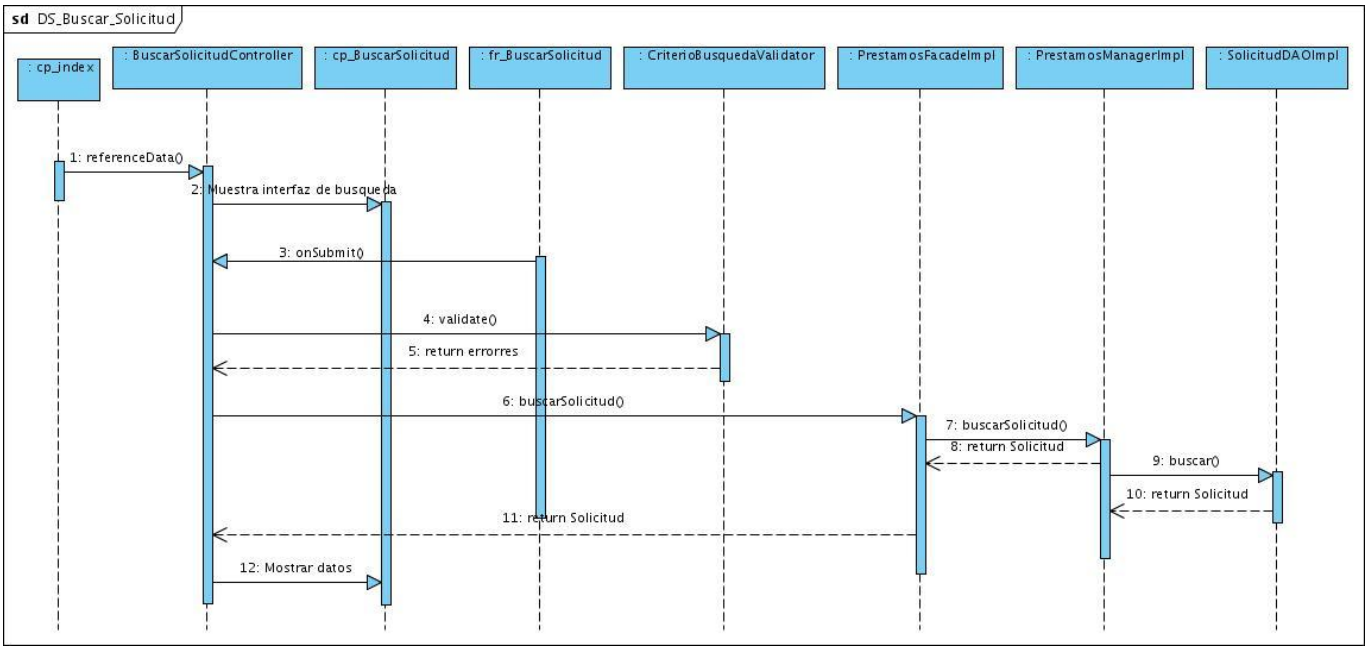


Figura 27. Diagrama de secuencia CU Buscar solicitud. Flujo básico.

2.3.3. Modelo de datos

El modelo de datos es usado para describir la representación lógica y física de la información persistente manejada por el sistema. A continuación el modelo de datos:

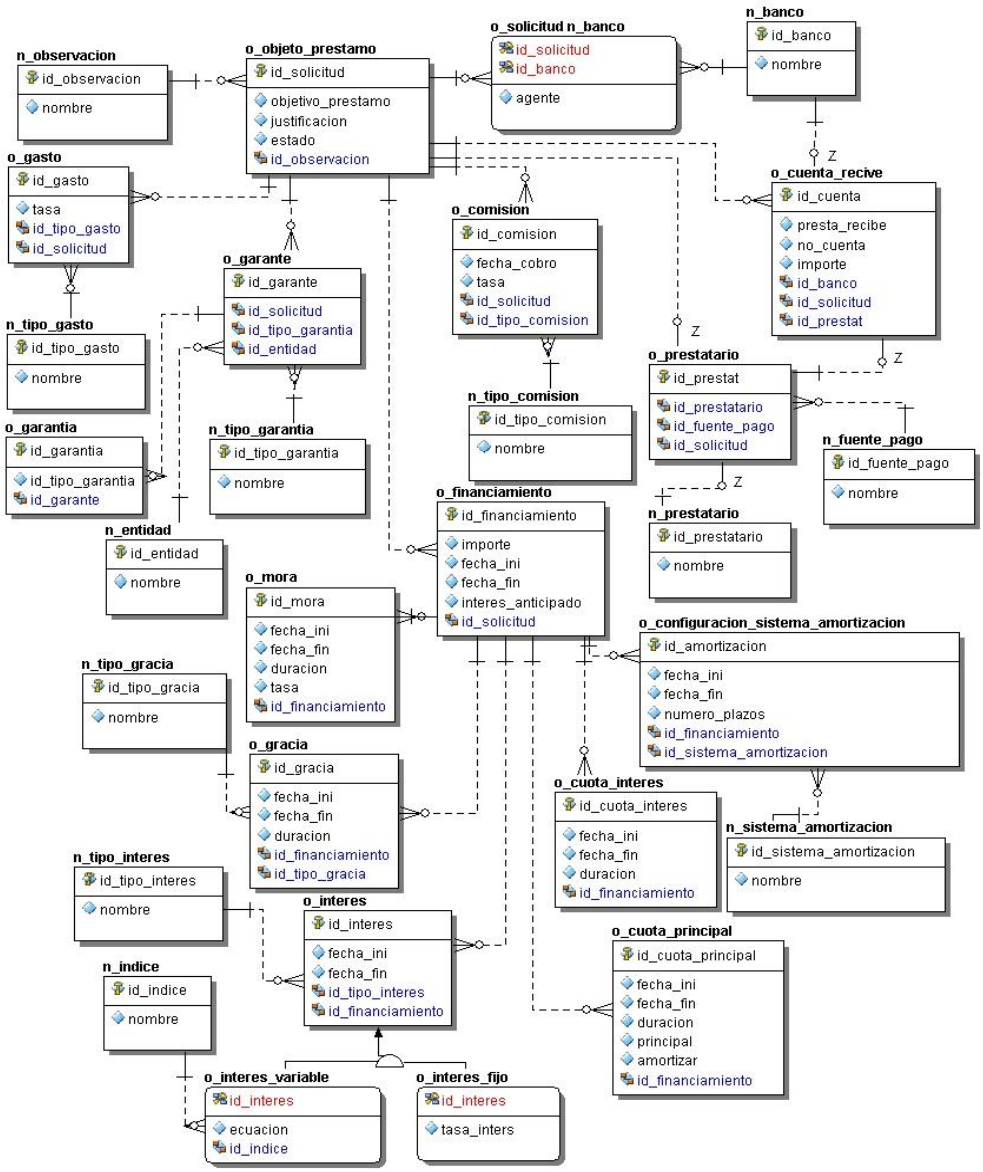


Figura 28. Modelo de datos.

2.4. Conclusiones parciales

En el capítulo se muestran los artefactos generados en el flujo de trabajo de análisis y diseño, además se incluye una breve descripción de los mismos, con el objetivo de que se comprenda perfectamente los requisitos del software. Posibilitando la correcta transformación de los mismos a un diseño que indique como debe ser implementado el software. También se encuentra información referente a los patrones utilizados y a la arquitectura.

CAPÍTULO 3: Validación de la solución propuesta.

3.1. Introducción

La medición es fundamental para cualquier disciplina de ingeniería, y la ingeniería del software no es una excepción. La medición nos permite tener una visión más profunda, proporcionando un mecanismo para la evaluación objetiva. (12)

En el presente capítulo se aplican un conjunto de métricas de diseño orientado a objeto y se analizan los resultados para determinar la calidad del diseño.

3.2. Métricas para el Modelo de Diseño

Las métricas para diseño proporcionan al diseñador una mejor visión interna, ayudan a que el diseño evolucione a un nivel superior de calidad. Para la aplicación de estas métricas Berard [Laranjeira ´90] define cinco características fundamentales.

1. Localización.
2. Encapsulamiento.
3. Ocultamiento de la información.
4. Herencia.
5. Técnicas de abstracción de objetos.

3.2.1. Métricas orientadas a Clases

Una clase es la unidad principal de todo sistema OO. Por lo que las medidas y métricas para una clase individual, la jerarquía de clases, y las colaboraciones de clases resultan sumamente valiosas para un ingeniero de software que necesite estimar la calidad de un diseño.

3.2.1.1. Métricas propuestas por Lorenz y Kidd

Lorenz y Kidd en su libro[2] dividen las métricas basadas en clases en cuatro categorías: tamaño, herencia, valores internos y valores externos. Las métricas orientadas a tamaños para una clase se centran en cálculos de atributos y de operaciones para una clase individual, y promedian los valores para el sistema en su totalidad. Las métricas basadas en herencia se centran en la forma en que se reutilizan las operaciones a lo largo y ancho de la jerarquía de clases. Las métricas para valores internos de clase examinan la cohesión y asuntos relacionados con el código, y las métricas orientadas a valores externos examinan el acoplamiento y la reutilización.

De las métricas propuestas por Lorenz y Kidd se aplicaran dos al diseño propuesto.

3.2.1.1.1. Tamaño de Clase (TC)

Lorenz y Kidd proponen que para medir el tamaño de clases se deben tener los siguientes aspectos en cuenta:

- ✓ Total de operaciones, ya sean las de la clase o las heredadas de las clases padres o interfaces que implementen.
- ✓ Cantidad de atributos, al igual que el anterior, tanto los de ella, como los de los padres.
- ✓ Promedio de operaciones y atributos para el sistema completo.

Para evaluar las métricas son necesarios los valores de los umbrales. Las medidas o umbrales para los parámetros de calidad han sido una polémica a nivel mundial en el diseño de sistemas. Algunos especialistas plantean umbrales para estas métricas según se muestra en la tabla 2, estos fueron los aplicados en el diseño de este sistema.

[2] *Lorenz, M. et al, 1994] Lorenz, M. & Kidd, J. "Object-Oriented Software Metrics". Prentice Hall. 1994. Texto recomendado para introducirse en la medición de atributos de entidades desarrolladas con metodología orientada a objetos.*

Nro. de operaciones y/o atributos	
TC	Umbral
Pequeño	<= 20
Medio	>20 y <=30
Grande	>30

Tabla 2. Umbrales para TC

Al aplicar la métrica TC al diseño propuesto se obtuvieron los siguientes resultados:
Para un total de 61 clases existentes en el diseño, se obtuvo un promedio de 2.7 atributos por clases y 7.6 operaciones, como se muestra en la tabla 3.

Total de Clases	Operaciones	Atributos
61	7.6	2.7

Tabla 3 Cantidad de clases de Diseño, operaciones y atributos promediados.

De acuerdo con los umbrales propuestos en la tabla 2 se obtuvo que para un total de 61 clases que tiene el diseño, 57 son de tamaño pequeño y 4 son de tamaño medio, como se muestra en la tabla 4, lo que representa el 7 % de clases medianas y el 93 % de clases pequeñas.

Cantidad de clases	Umbral	Tamaño
57	<= 20	Pequeño
4	>20 y <=30	Medio

Tabla 4. Cantidad de clases por tamaño.

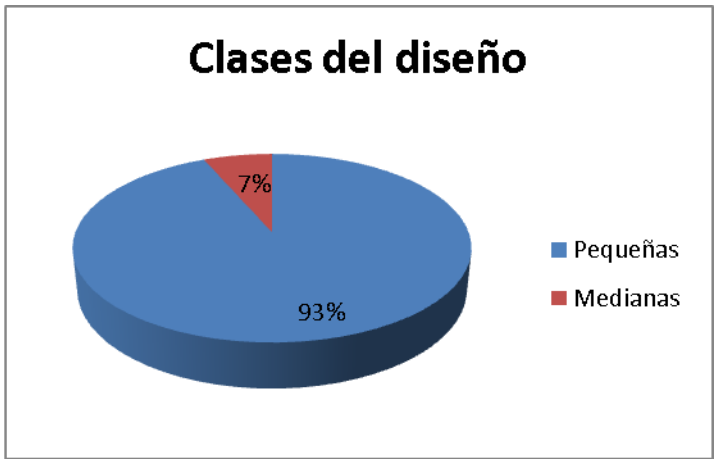


Figura 29. Representación de las clases según TC

Estos valores demuestran que los indicadores de calidad reutilización, implementación y responsabilidad no se ven afectados.

3.2.1.1.2. Número de Operaciones redefinidas por una subclase (NOR)

Para aplicar esta métrica es necesario conocer el número de clases que redefinen operaciones heredadas de otras clases. En la tabla 5 se muestran los valores obtenidos para el diseño propuesto.

Total de clases del diseño	Total de clases que redefinen operaciones
61	11

Tabla 5. Total de clases que redefinen operaciones.

Si se calculamos el porciento que representa el total de clases que redefinen funciones, nos da como resultado que el 18 % de las clases existentes en el diseño redefinen operaciones.

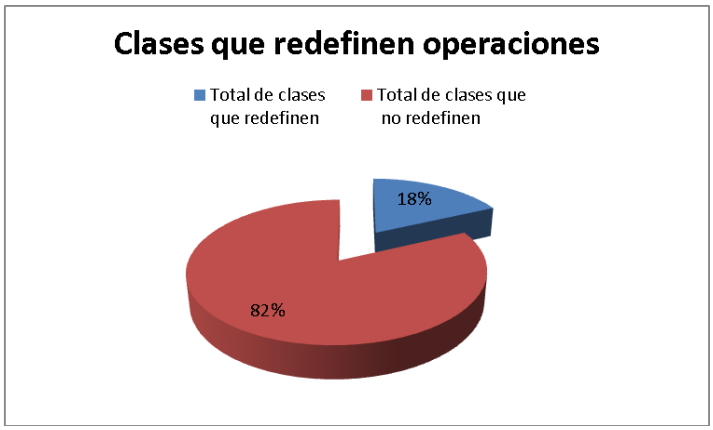


Figura 30. Total de clases que redefinen operaciones.

Se puede observar que el valor obtenidos para la aplicación de la norma NOR al diseño propuesto es aceptable, esto trae consigo que no se afecte la calidad del diseño y puede ser llevado a pruebas o modificado sin dificultad alguna.

3.2.1.2. Familia de métricas propuestas por Chidamber & Kemerer

Los conjuntos de métricas propuestos por Chidamber y Kemerer han sido uno de los más referenciados, las que son conocidas normalmente como la serie de métricas CK. Chidamber y Kemerer proponen seis métricas basadas en clases para sistemas orientados a objetos:

- Métodos ponderados por clase
- Profundidad árbol de herencia
- Número de descendientes
- Acoplamiento entre clases
- Respuesta para una clase
- Carencia de cohesión en los métodos

Estas métricas permiten conocer hasta qué punto están bien definidas las clases y el sistema, lo cual tiene un impacto directo en la mantenibilidad del mismo, tanto por la comprensión de lo desarrollado como por la dificultad de modificarlo con éxito.

De estas seis se aplicaran dos al diseño propuesto.

3.2.1.2.1. Árbol de profundidad de herencia (APH)

Esta métrica está definida como la longitud máxima desde el nodo hasta la raíz del árbol. A medida que crece el APH, es más probable que las clases de niveles inferiores hereden muchos métodos. Esto da lugar a posibles dificultades cuando se intenta predecir el comportamiento de una clase. Una jerarquía de clases profunda (con un valor grande de APH) lleva también a una mayor complejidad de diseño. Por el lado positivo, los valores grandes de APH implican que se pueden reutilizar muchos métodos, lo que debe ser considerado como un elemento a favor de la mantenibilidad.

En la tabla 6 se muestran los umbrales o medidas recomendados por algunos especialistas, estos fueron los aplicados en el diseño de este sistema.

Nivel de jerarquía de clases	
APH	Umbral
Sencilla	<= 5
Compleja	>5

Tabla 6. Umbral para la Métrica APH

Al aplicar esta métrica al diseño propuesto, según los umbrales definidos, se obtiene como resultado que el APH presenta valor 4 como se muestra en la tabla 7, pues solo hay presencia de herencia entre las clases y las interfaces de las cuales heredan sus funciones y las redefinen, lo cual está dentro del umbral definido para determinar que el diseño es sencillo, por lo que evita que exista algún problema si se desea predecir el comportamiento de una clase y no es difícil de dar mantenimiento.

Total de clases	APH
61	4

Tabla 7. Resultados aplicando la métrica APH

3.2.1.2.2. Número de descendiente (NDD)

Las subclases que son inmediatamente subordinadas a una clase son denominadas descendientes. A medida que crece el número de descendientes, se incrementa la reutilización, pero también es cierto, que

a medida que crece NDD, la abstracción representada por la clase predecesora puede verse diluida y existe la posibilidad de que algunos de los descendientes no sean realmente miembros propios de la clase predecesora.

Nro. de clases	Nro. de descendientes
23	1
1	2

Tabla 8. Cantidad de clases por cantidad de descendientes

Como se puede observar en la tabla 8, existen 23 clases con NDD igual a 1 y solo una con 2 descendientes de un total de 61 clases. Los valores de NDD alcanzados para el diseño propuesto son pequeños lo que permite que cada descendiente sea realmente miembro de la clase predecesora y también se reduce la cantidad de pruebas necesarias para ejercitar cada uno de los miembros de la jerarquía.

3.3. Conclusiones parciales

En este capítulo se aplicaron métricas para la evaluación del diseño obtenido, dándole solución al objetivo planteado. El diseño propuesto no presenta una alta complejidad permitiendo que las pruebas no sean complejas y que el resto de los módulos puedan ser integrados sin muchas dificultades, además presenta un bajo acoplamiento pues la profundidad de los niveles de herencia están acorde con los umbrales. Se puede concluir que el diseño obtenido posee una calidad aceptable, facilitando la continuación eficiente del desarrollo en etapas posteriores.

CONCLUSIONES

Una vez concluido el estudio de los procesos de Préstamos Bancario, se realizo el análisis y diseño de dicho subsistema, lo cual permitirá su posterior implementación.

Como resultados del análisis se obtuvieron los diagramas de colaboración y los de las clases del análisis.

Para la realización del diseño se tuvieron en cuenta algunos patrones para modelar el sistema, lo que permitió generar artefactos empleando buenas prácticas, necesarias para desarrollar un software con mayor robustez. Entre los artefactos generados en el diseño se pueden mencionar: diagramas de secuencia, diagramas de clases del diseño y el modelo de datos, imprescindibles para el buen desarrollo de una solución informática.

De forma general los resultados obtenidos a partir del análisis y diseño del sistema son positivos, tomando como referencia la aplicación de métodos y métricas para validar la solución del mismo.

RECOMENDACIONES

1. Profundizar en el estudio del proceso de Préstamos en otras entidades bancarias para una futura versión ampliada del sistema.
2. Realizar los restantes flujos de trabajo que propone la metodología utilizada, RUP, llegando a implementar las funcionalidades que hemos propuesto para el subsistema de Préstamos del Proyecto Modernización del Sistema Bancario Cubano, lo cual tributará a un incremento de la eficiencia y calidad en el Banco Nacional de Cuba.

REFERENCIA BIBLIOGRÁFICA

1 Menéndez-Barzanallana Asensio Rafael. Informática Aplicada a la Gestión Pública. Facultad de Derecho. Capítulo 1. Ingeniería del software. Introducción, 2008. [Citado el: 26 de abril 2009] http://www.um.es/docencia/barzana/IAGP/IAGP2-Ingenieria-software-introduccion.html#BM3

2 Fernández Pérez Yudid. Enfoque administrativo de la Ingeniería de Software, 2007. [Citado el: 26 de Abril de 2009.] http://www.monografias.com/trabajos-pdf/enfoque-administrativo-ingenieria-software/enfoque-administrativo-ingenieria-software.pdf.

3 Departamento Central de Ingeniería y Gestión de Software. Material para la conferencia 1, 2008. [Citado el: 26 de abril del 2009] http://teleformacion.uci.cu/mod/resource/view.php?id=4598&subdir=/Materiales_Basicos_Conf_1.

4 Mendoza Sánchez, María A. Metodologías De Desarrollo De Software. Informatizate. 2004. [Citado el 20 de abril del 2009] http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html.

5 Jacobson Ivar, Rumbaugh James y Booch, Grady. El proceso Unificado de desarrollo de software, 2000 [Citado el: 4 de marzo del 2009] http://bibliodoc.uci.cu/pdf/reg00060.pdf

6 RUMBAUGH JAMES, I. J., GRADY BOOCH. El lenguaje unificado de modelado. Manual de Referencia, 2001. [Citado el: 7 de enero 2009] http://bibliodoc.uci.cu/pdf/reg03050.pdf

7 Toca Ramos Lázaro Alberto. Estudio de la Contabilidad General. La Habana: Félix Varela, 2006.

8 Filcun Juan Carlos. 2006. Curso: Conocimiento básicos de contabilidad, 2006. [Citado el 11 de noviembre 2008] http://www.mailxmail.com/curso/empresa/basicocontabilidad/capitulo5.htm.

9 Martinez Vilches, Ramón. NUEVOS RETOS DE LA CONTABILIDAD DE GESTION EN LAS ENTIDADES BANCARIAS .1993 [Citado el: 6 de noviembre 2008] HTTP://WWW.OBSERVATORIO-IBEROAMERICANO.ORG/PAISES/SPAIN/ART%C3%ADCULOS%20DIVERSOS%20SOBRE%20CONTABILIDAD%20DE%20GESTI%C3%B3N/NUEVOS%20RETOS%20CONT%20GEST%20EN%20ENTID%20BANCARIAS%20-%20VILCHES.HTM

10 Cerezal Tamargo, Lourdes. La revista del empresario cubano. La contabilidad en una nueva tecnología. [Citado el 6 de septiembre 2008] http://www.betsime.disaic.cu/secciones/tec_feb_02.htm

11 Milián Sardiña, Lic. Beatriz, y otros. Principales características del Sistema Contable del Banco Central de Cuba. Octubre de 1998. [Citado el 1 de octubre 2008] http://www.cemla.org/pdf/pub-di-sis-mb.pdf.

12 Pressman, Roger S. 2005. Ingeniería del Software: Un enfoque práctico, 2005. [Citado el 3 de abril 2009] http://bibliodoc.uci.cu/pdf/reg02689.pdf

BIBLIOGRAFÍA CONSULTADA

Larman Graig. UML y Patrones,1999 [citado 30 de abril 2009]. http://bibliodoc.uci.cu/pdf/reg00061.pdf.

JAMES RUMBAUGH, I. J., GRADY BOOCH. El lenguaje unificado de modelado. Manual de Referencia, 2001. http://bibliodoc.uci.cu/pdf/reg03050.pdf

Bauer, F. 1997. Software engineering: a practitioner's approach. s.l. : Mc Graw-Hil, 1997. BCC. Evolución del Sistema Bancario y Financiero a partir de 1995. [Citado el: 24 de Mayo de 2008.] http://www.cubagob.cu/des_eco/banco/espanol/sistema_bancario/bcc.htm.

Bravo Santos, Crescencio y García Rubio, Félix Oscar. Ingeniería del Software. Metodologías de Desarrollo de Software. [En línea] http://alarcos.inf-cr.uclm.es/per/fgarcia/isoftware/doc/tema3_1xh.pdf.

Cerezal Tamargo, Lourdes. La revista del empresario cubano. La contabilidad en una nueva tecnología. [En línea] http://www.betsime.disaic.cu/secciones/tec_feb_02.htm.

Cruzado Nuño, Ignacio, García del Egido, Jorge y Portugal Alonso, Javier. Metodologías Orientadas a Objetos. [En línea] http://pisuerga.inf.ubu.es/icruzado/tfc/Metodologias.pdf.

Díaz-Antón, María Gabriela, Pérez, María Angélica y Grimmán, Anna C. 2002. PROPUESTA DE UNA METODOLOGÍA DE DESARROLLO DE SOFTWARE EDUCATIVO BAJO UN ENFOQUE DE CALIDAD SISTÉMICA. [En línea] 2002. http://www.academia-interactiva.com/ise.pdf.

Diccionario de economía y finanzas. [Citado el: 24 de agosto de 2008.] http://www.eumed.net/cursecon/dic/P8.htm.

Fernández Pérez, Yudid. 2007. Enfoque administrativo de la Ingeniería de Software. 2007. [Citado el: 14 de Mayo de 2008.] http://www.monografias.com/trabajos-pdf/enfoque-administrativo-ingenieria-software/enfoque-administrativo-ingenieria-software.pdf.

Pressman, Roger S. 2005. Ingeniería del Software: Un enfoque práctico, 2005. http://bibliodoc.uci.cu/pdf/reg02689.pdf

Sixto del Llano, Janet y Moreno Moya, Luis Kedir. La Contabilidad, concepto, clasificaciones y principios. [Citado el: 24 de Mayo de 2008.] http://www.educatur.nh.co.cu/cursos/pleste/contabilidad/Tema%202/Tema%202.clase%203.htm.

Borrás Francisco, López Míriam. La Contabilidad de Gestión en Cuba.1996 http://www.observatorio-iberoamericano.org/Libro%20-%20La%20contab%20de%20gestión%20en%20Latinoamérica/Cuba.htm

ANEXOS

Figura 1. La Ingeniería de Software es una tecnología multicapa.

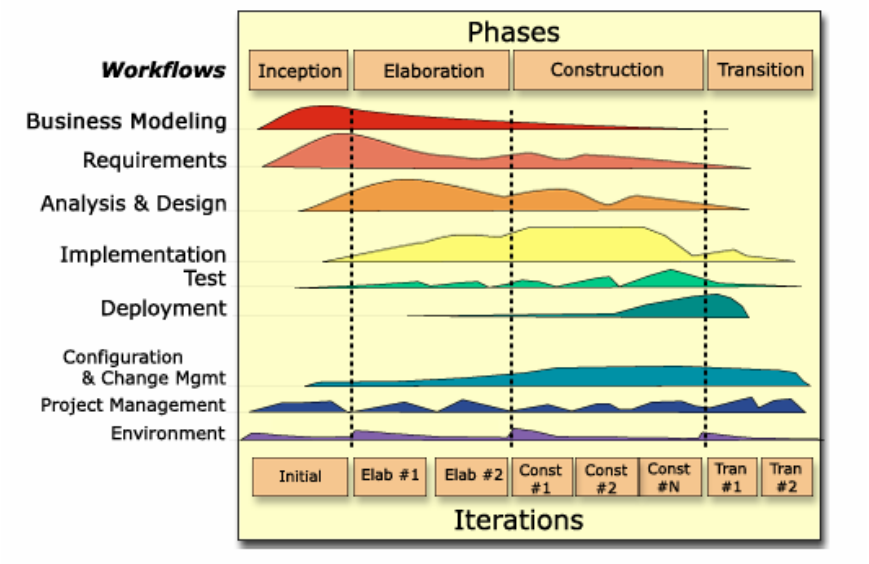


Figura 2. Metodología Rational Unified Process (RUP)

Descripción de casos de usos

Caso de Uso:	Registrar solicitud de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción Registrar solicitud de préstamo.	2. El sistema muestra una interfaz con los campos necesarios para registrar la solicitud.
3. El Operador de la DGN introduce los datos de la solicitud incluyendo los del financiamiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.* *Ver Flujos Alternos Sección Imprimir.*	4. El sistema valida que los datos sean correctos. *En caso de:* *c) Error en los datos entrados.* *Ver Flujos Alternos Sección Datos Incorrectos.* *d) Solicitud de préstamo ya exista en el sistema.* *Ver Flujos Alternos Sección Solicitud de Préstamo Existente.*
	5. El sistema registra la solicitud de préstamo. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.

	2. Continúa el flujo normal de eventos.
Sección Datos Incorrectos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema señala los datos incorrectos y muestra el mensaje “Datos Incorrectos”.
2. El Operador de la DGN corrige los datos incorrectos y solicita aceptar.	3. El sistema pasa a la acción 4 del flujo normal de eventos.
Sección Solicitud de Préstamo Existente	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje “La solicitud de préstamo ya existe”.
2. El Operador de la DGN acepta el mensaje y solicita cancelar.	3. Finaliza el caso de uso.

Caso de Uso:	Actualizar solicitud de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción Actualizar solicitud de préstamo.	2. El sistema muestra una interfaz con los datos de la solicitud de préstamo seleccionada.
3. El Operador de la DGN actualiza los datos de la solicitud incluyendo los del financiamiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.*	4. El sistema valida que los datos sean correctos. *En caso de:* *c) Error en los datos entrados.* *Ver Flujos Alternos Sección Datos Incorrectos.*

b) Selecciona imprimir. *Ver Flujos Alternos Sección Imprimir.*	
	5. El sistema actualiza la solicitud de préstamo. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.
	2. Continúa el flujo normal de eventos.
Sección Datos Incorrectos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema señala los datos incorrectos y muestra el mensaje “Datos Incorrectos”.
2. El Operador de la DGN corrige los datos incorrectos y solicita aceptar.	3. El sistema pasa a la acción 4 del flujo normal de eventos.

Caso de Uso:	Eliminar solicitud de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción eliminar solicitud de préstamo.	2. El sistema comprueba que la solicitud no ha sido contratada y muestra el mensaje “Seguro que desea eliminar la solicitud de

	préstamo del sistema. *En caso de:* *a) La solicitud se encuentra contratada.* *Ver Flujos Alternos Sección Solicitud Contratada.*
3. El Operador de la DGN solicita aceptar. *En caso de:* *b) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.*	4. El sistema elimina la solicitud de préstamo. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Solicitud Contratada	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje “La solicitud se encuentra contratada y no es posible su eliminación”.
2. El Operador de la DGN aceptar el mensaje.	3. Finaliza el caso de uso.

Caso de Uso:	Buscar solicitud de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción Buscar solicitud de préstamo.	2. El sistema muestra la interfaz con los campos necesarios para la búsqueda.
3. El Operador de la DGN entra los datos para	4. El sistema chequea que al menos un

la búsqueda y solicita aceptar. *En caso de:* *a) Selecciona cancelar Ver Flujos Alternos Sección Cancelar.*	campo de búsqueda esté lleno. *En caso de:* *b) Todos los campos de búsqueda estén vacíos.* *Ver Flujos Alternos Sección Campos de Búsqueda Vacíos.*
	5. El sistema realiza la búsqueda y muestra una lista con las solicitudes que cumplen con el criterio buscado. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Campos de Búsqueda Vacíos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje "Debe entrar los datos para la búsqueda".
2. El Operador de la DGN acepta el mensaje.	
3. El Operador de la DGN llena el campo de búsqueda y solicita buscar.	4. El sistema pasa a la acción 5 del flujo normal de eventos.

Caso de Uso:	Consultar solicitud de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción	2. El sistema muestra una interfaz con los

consultar solicitud de préstamo.	datos de la solicitud de préstamo seleccionada incluyendo los datos del financiamiento.
3. El Operador de la DGN consulta los datos de la solicitud y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.* *Ver Flujos Alternos Sección Imprimir.*	4. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir. 2. Continúa el flujo normal de eventos.

Caso de Uso:	Registrar contrato de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción Registrar contrato de préstamo.	2. El sistema muestra una interfaz con los campos necesarios para registrar el contrato.

3. El Operador de la DGN introduce los datos del contrato incluyendo los datos del financiamiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.* *Ver Flujos Alternos Sección Imprimir.*	4. El sistema valida que los datos sean correctos. *En caso de:* *c) Error en los datos entrados.* *Ver Flujos Alternos Sección Datos Incorrectos.* *d) Contrato de préstamo ya exista en el sistema.* *Ver Flujos Alternos Sección Contrato de Préstamo Existente.*
	5. El sistema registra el contrato de préstamo. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.
	2. Continúa el flujo normal de eventos.
Sección Datos Incorrectos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema señala los datos incorrectos y muestra el mensaje “Datos Incorrectos”.
2. El Operador de la DGN corrige los datos incorrectos y solicita aceptar.	3. El sistema pasa a la acción 4 del flujo normal de eventos.
Sección Contrato de Préstamo Existente	
Acción del Actor	**Respuesta del Sistema**

	1. El sistema muestra el mensaje “El contrato de préstamo ya existe”.
2. El Operador de la DGN acepta el mensaje.	3. Finaliza el caso de uso.

Caso de Uso:	Actualizar contrato de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción actualizar contrato de préstamo.	2. El sistema muestra una interfaz con los datos del contrato seleccionado.
3. El Operador de la DGN actualiza los datos del contrato incluyendo los del financiamiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.* *Ver Flujos Alternos Sección Imprimir.*	4. El sistema valida que los datos sean correctos. *En caso de:* *c) Error en los datos entrados.* *Ver Flujos Alternos Sección Datos Incorrectos.*
	5. El sistema actualiza el contrato de préstamo. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso

Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir. 2. Continúa el flujo normal de eventos.
Sección Datos Incorrectos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema señala los datos incorrectos y muestra el mensaje “Datos Incorrectos”.
2. El Operador de la DGN corrige los datos incorrectos y solicita aceptar.	3. El sistema pasa a la acción 4 del flujo normal de eventos.

Caso de Uso:	Eliminar contrato de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción eliminar contrato de préstamo.	2. El sistema comprueba que el contrato no ha sido contabilizado y muestra el mensaje “Seguro que desea eliminar el contrato de préstamo del sistema”. *En caso de:* *a) El contrato se encuentra contabilizado.* *Ver Flujos Alternos Sección Contrato Contabilizado.*
3. El Operador de la DGN solicita aceptar. *En caso de:* *b) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.*	4. El sistema elimina el contrato de préstamo. Finaliza el caso de uso.

Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Contrato Contabilizado	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje “El contrato se encuentra contabilizado y no es posible su eliminación”.
2. El Operador de la DGN acepta el mensaje.	3. Finaliza el caso de uso.

Caso de Uso:	Buscar contrato de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción Buscar contrato de préstamo.	2. El sistema muestra la interfaz con los campos necesarios para la búsqueda.
3. El Operador de la DGN entra los datos para la búsqueda y solicita aceptar. *En caso de:* *a) Selecciona cancelar Ver Flujos Alternos Sección Cancelar.*	4. El sistema chequea que al menos un campo de búsqueda esté lleno. *En caso de:* *b) Todos los campos de búsqueda estén vacíos.* *Ver Flujos Alternos Sección Campos de Búsqueda Vacíos.*
	5. El sistema realiza la búsqueda y muestra una lista con los contratos que cumplen con el criterio buscado. Finaliza el caso de uso.

Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Campos de Búsqueda Vacíos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje "Debe entrar los datos para la búsqueda".
2. El Operador de la DGN acepta el mensaje.	
3. El Operador de la DGN llena el(los) campo(s) de búsqueda y solicita buscar.	4. El sistema pasa a la acción 4 del flujo normal de eventos.

Caso de Uso:	Consultar contrato de préstamo
Actores:	Operador de la DGN
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de la DGN selecciona la opción Consultar contrato de préstamo.	2. El sistema muestra una interfaz con los datos del contrato de préstamo seleccionado.
3. El Operador de la DGN consulta los datos del contrato incluyendo los del financiamiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.* *Ver Flujos Alternos Sección*	4. Finaliza el caso de uso.

Imprimir.	
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.
	2. Continúa el flujo normal de eventos.

Caso de Uso:	Registrar préstamo
Actores:	Operador de préstamos
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de préstamos selecciona la opción Registrar préstamo.	2. El sistema muestra una interfaz con los campos del préstamo.
3. El Operador de préstamos inserta los datos del préstamo incluyendo los del financiamiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.* *Ver Flujos Alternos Sección Imprimir.* *c) Selecciona Ver Transacción.* *Ver Flujos Alternos Sección Ver*	4. El sistema valida los datos entrados. *En caso de:* *d) Error en los datos entrados.* *Ver Flujos Alternos Sección Datos Incorrectos*

Transacción.	
	5. El sistema registra el préstamo afectando la contabilidad. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.
	2. Continúa el flujo normal de eventos.
Sección Datos Incorrectos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema señala los campos incorrectos y muestra el mensaje “Datos Incorrectos”.
2. El Operador de Préstamos corrige los datos incorrectos y solicita aceptar.	3. El sistema pasa a la acción 4 del flujo normal de eventos.
Sección Ver Transacción	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra la transacción contable generada.
2. El Operador de Préstamos acepta.	3. El sistema pasa a la acción 4 del flujo normal de eventos.

Caso de Uso:	Actualizar préstamo

Actores:	Operador de Préstamos
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de Préstamos selecciona la opción Actualizar préstamo.	2. El sistema muestra una interfaz con los campos necesarios para actualizar el préstamo.
3. El Operador de Préstamos introduce los cambios incluyendo los del financiamiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.* *Ver Flujos Alternos Sección Imprimir.* c) *Seleccionar Ver Transacción.* *Ver Flujos Alternos Sección Ver Transacción.*	4. El sistema valida que los datos sean correctos. *En caso de:* *d) Error en los datos entrados.* *Ver Flujos Alternos Sección Datos Incorrectos.*
	5. El sistema actualiza el préstamo. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.

	2. Continúa el flujo normal de eventos.
Sección Datos Incorrectos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema señala los datos incorrectos y muestra el mensaje “Datos Incorrectos”.
2. El Operador de Préstamos corrige los datos incorrectos y solicita aceptar.	3. El sistema pasa a la acción 4 del flujo normal de eventos.
Sección Ver Transacción	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra la transacción contable generada.
2. El Operador de Préstamos aceptar.	3. El sistema pasa a la acción 4 del flujo normal de eventos.

Caso de Uso:	Consultar Préstamo Concedido
Actores:	Usuario
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción Consultar préstamo.	2. El sistema muestra una interfaz con los datos del préstamo seleccionado.
3. El Usuario consulta los datos del préstamo incluyendo los del financiamiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.*	4. Finaliza el caso de uso.

Ver Flujos Alternos Sección Imprimir.	
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.
	2. Continúa el flujo normal de eventos.

Caso de Uso:	Buscar préstamos concedidos
Actores:	Usuario
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción Buscar préstamos concedidos.	2. El sistema muestra la interfaz con los campos necesarios para la búsqueda.
3. El Operador de la DGN entra los datos para la búsqueda y solicita aceptar. *En caso de:* *a) Selecciona cancelar Ver Flujos Alternos Sección Cancelar.*	4. El sistema chequea que al menos un campo de búsqueda esté lleno. *En caso de:* *b) Todos los campos de búsqueda estén vacíos.* *Ver Flujos Alternos Sección Campos de Búsqueda Vacíos.*
	5. El sistema realiza la búsqueda y muestra una lista con los que cumplen con el criterio buscado. Finaliza el caso de uso.

Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Campos de Búsqueda Vacíos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje "Debe entrar los datos para la búsqueda".
2. El Operador de la DGN acepta el mensaje.	
3. El Operador de la DGN llena el(los) campo(s) de búsqueda y solicita buscar.	4. El sistema pasa a la acción 4 del flujo normal de eventos.

Caso de Uso:	Buscar vencimientos de préstamo
Actores:	Operador de Préstamos
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de Préstamos selecciona la opción Buscar vencimientos de préstamos.	2. El sistema muestra una interfaz con los campos para la búsqueda.
3. El Operador de Préstamos introduce los datos de la búsqueda y solicita aceptar. *En caso de:* *a) Solicite cancelar.* *Ver Flujos Alternos Sección Cancelar*.	4. El sistema verifica que se hayan llenado los campos de búsqueda. *En caso de:* *b) Todos los campos se encuentren vacíos.* *Ver Flujos Alternos Sección Campos Vacíos.*
	5. El sistema realiza la búsqueda y muestra la lista de los vencimientos. Finaliza el caso de uso.

Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Campos Vacíos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje “Deben llenarse los campos par la búsqueda”.
2. El Operador de Préstamos acepta el mensaje.	
3. El Operador de Préstamos introduce los datos para la búsqueda y solicita aceptar.	4. El sistema pasa a la acción 4 del flujo normal de eventos.

Caso de Uso:	Consultar vencimiento de préstamo
Actores:	Operador de Préstamos
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Operador de Préstamos selecciona la opción Consultar vencimiento de préstamo.	2. El sistema muestra una interfaz con los datos del vencimiento de préstamo seleccionado.
3. El Operador de Préstamos consulta los datos del vencimiento y solicita aceptar. *En caso de:* *a) Selecciona cancelar.* *Ver Flujos Alternos Sección Cancelar.* *b) Selecciona imprimir.*	4. Finaliza el caso de uso.

Ver Flujos Alternos Sección Imprimir. c) *Selecciona liquidar.* *Ver Flujos Alternos Sección Liquidar.* d) *Selecciona pasarlo a vencido.* *Ver Flujos Alternos Sección Pasar Vencimiento a Vencido.*	
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.
	2. Continúa el flujo normal de eventos.
Sección Liquidar	
Acción del Actor	**Respuesta del Sistema**
	1. Se inicia el caso de uso Liquidar vencimiento.
Sección Pasar Vencimiento a Vencido	
Acción del Actor	**Respuesta del Sistema**
	1. Se inicia el caso de uso Pasar vencimiento a vencido.

Caso de Uso:	Liquidar vencimiento de préstamo
Actores:	Operador de Préstamos
Flujo Normal de Eventos	

Acción del Actor	Respuesta del Sistema
1. El Operador de Préstamos selecciona la opción Liquidar vencimiento de préstamo.	2. El sistema verifica la fecha de su vencimiento y muestra la interfaz con la transacción generada. *En caso de:* *a) La fecha de vencimiento sea mayor a la actual.* *Ver Flujo Alterno Sección Liquidación Cancelada*
3. El Operador de Préstamos solicita aceptar. En caso de: *b) Selecciona cancelar.* *Ver Flujo Alterno Sección Cancelar.* c) *Selecciona imprimir.* *Ver Flujos Alternos Sección Imprimir.*	4. El sistema registra la transacción en la contabilidad. Finaliza el caso de uso.
Flujos Alternos	
Sección Cancelar	
Acción del Actor	**Respuesta del Sistema**
	1. Finaliza el caso de uso
Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.
	2. Continúa el flujo normal de eventos.
Sección Liquidación Cancelada	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje” Aún no se

	puede liquidar el vencimiento".
2. El Operador de Préstamos acepta el mensaje.	3. Finaliza el caso de uso.

Caso de Uso:	Poner vencido un vencimiento de préstamo	
Actores:	Operador de Préstamos	
Flujo Normal de Eventos		
Acción del Actor		**Respuesta del Sistema**
1. El Operador de Préstamos selecciona la opción poner en estado vencido un vencimiento de préstamo.		2. El sistema verifica la fecha de su vencimiento y muestra la interfaz con la transacción generada. *En caso de:* *a) La fecha de vencimiento sea mayor a la actual.* *Ver Flujo Alterno Sección Acción Cancelada.*
3. El Usuario solicita aceptar. En caso de: *b) Selecciona cancelar.* *Ver Flujo Alterno Sección Cancelar.* c) *Selecciona imprimir.* *Ver Flujos Alternos Sección Imprimir.*		4. El sistema registra la transacción en la contabilidad. Finaliza el caso de uso.
Flujos Alternos		
Sección Cancelar		
Acción del Actor		**Respuesta del Sistema**
		1. Finaliza el caso de uso

Sección Imprimir	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema manda a imprimir.
	2. Continúa el flujo normal de eventos.
Sección Acción Cancelada	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra el mensaje “Aún no se puede pasar a vencido el vencimiento”.
2. El Operador de Préstamos acepta el mensaje.	3. Finaliza el caso de uso.

GLOSARIO DE TÉRMINOS

Acreedores Aquella persona física o jurídica legítimamente facultada para exigir el pago o cumplimiento de una obligación contraída por dos partes con anterioridad.

Amortización: es un término económico y contable, referido al proceso de distribución en el tiempo de un valor duradero. Adicionalmente se utiliza como sinónimo de depreciación en cualquiera de sus métodos.

Amortización de un pasivo: La obligación de devolver un préstamo recibido de un banco es un pasivo, cuyo importe se va reintegrando en varios pagos diferidos en el tiempo. La parte de capital (o principal) que se cancela en cada uno de esos pagos es una amortización.

Amortización de un activo: Existen varios métodos de amortización, tanto de activos inmovilizados (cuotas fijas, crecientes, decrecientes...). Se trata de técnicas aritméticas para repartir un importe determinado, el valor a amortizar, en varias cuotas, correspondientes a varios periodos.

Desde el punto de vista lingüístico la expresión **depreciación** es más apropiada para reflejar la pérdida de valor de los activos materiales (también llamados bienes de uso). Sin embargo, las normas contables de algunos países eligen la expresión **amortización**.

Amortización desde el punto de vista contable

Amortizar significa considerar que un determinado elemento del activo fijo empresarial ha perdido, por el mero paso del tiempo, parte de su valor. Para reflejar contablemente este hecho, y en atención al método contable de partida doble, hay que: 1º Dotar una amortización, es decir, considerar como pérdida del ejercicio la disminución del valor experimentado. 2º Crear una cuenta negativa en el activo del balance, que anualmente vería incrementado su saldo con la indicada disminución del valor del bien. De esta forma todo elemento del activo fijo de la empresa vendría reflejado por dos cuentas, una positiva, que recogería el valor de su adquisición u obtención, y otra negativa (llamada de Amortización Acumulada), en la cual se indica lo que vale de menos como consecuencia del paso del tiempo.

Banco Agente: En los créditos sindicados, figura encargada de realizar la gestión y administración del crédito y de estar en contacto con el prestatario, de forma que es el único que está facultado para recibir los intereses y devolución del principal. En numerosas ocasiones el banco agente coincide con el banco director.

En una operación de préstamo sindicado, entre los bancos prestamistas se designa a uno de ellos como

banco agente para proteger y gestionar sus intereses comunes. También es el banco designado por una empresa para asegurar el cumplimiento de determinadas obligaciones financieras, tanto por operaciones realizadas en el propio país como en el extranjero.

Comisiones: es el porcentaje que te pagan por vender algún producto. La comisión es una parte, calculada en Porcentaje o Monto especifico, sobre un valor total, que se origina en consideración de pago por gestión de venta o participación en la comercialización de un producto o servicio.

Configuración de la mora: La configuración del estado de mora opera también en el devengar de la cláusula penal desde el vencimiento del término, sobre todo tratándose de cláusula penal moratoria. La configuración de la mora del acreedor requiere la reunión de los requisitos de fondo: la posibilidad del deudor de cumplir la prestación, y la falta de aceptación de la prestación en el tiempo por parte del acreedor o su omisión de prestar la cooperación de su parte que sea indispensable para permitir el cumplimiento.

Cuenta: Es el elemento básico y central en la contabilidad y en los servicios de pagos. Las cuentas suponen la clasificación de todas las transacciones comerciales que tiene una empresa o negocio. Se refiere al nombre debidamente codificado o numerado que se da a los valores que posee la empresa. La cuenta facilita el registro de las operaciones contables en los libros de contabilidad, representa bienes, derechos y obligaciones de los que dispone una empresa en una fecha determinada.

Garantes:

1. Que garantiza, que da garantía.
2. Persona que garantiza una cosa o avala a alguien.

Interés Fijo: Este tipo de transacción paga una tasa de interés acordada que se mantiene constante durante el término de la negociación. Los intereses fijos muchas veces se encuentran en los bonos, así como también en las hipotecas con tasa fija.

Interés Variables: Tipo de interés fijado de acuerdo con un tipo de referencia (interbancario). El tipo de interés varía al alza o a la baja según varíe el tipo de referencia en vez de ser siempre el mismo como en el caso del interés fijo.

Liquidez: Facilidad con la cual un activo puede convertirse en dinero. Los activos comprenden el efectivo, que es perfectamente líquido, y otros que resultan gradualmente menos líquidos: divisas, valores, depósitos a corto y largo plazo, cuentas de resultado acreedor, bienes de consumo duraderos, bienes de

capital, metales preciosos, obras de arte, entre otros. El grado de liquidez de cada uno de estos activos se mide por la facilidad de convertirlo en dinero efectivo.

Prestamista: Persona que da dinero a préstamo

Prestatario: Persona que recibe el dinero y se endeuda respecto del prestamista.

Persona que recibe una cantidad de dinero con la obligación de devolverlo, junto a los intereses acordados, al cabo de un tiempo fijado.

Sujeto pasivo del contrato de préstamo. El que recibe el dinero o la cosa en préstamo.

Período de Gracia: El periodo de gracia, o período gratuito, es la cantidad de días que tiene que pagar su factura en su totalidad sin incurrir en un gasto financiero. Si no paga el saldo en su totalidad, generalmente no obtiene un período de gracia sobre el saldo al mes siguiente. La mayoría de las tarjetas de crédito no proporcionan un período de gracia sobre adelantos en efectivo y transferencia de saldos. Los períodos de gracia de la tarjeta de crédito son la cantidad de tiempo dada por un acreedor al consumidor para pagar apagado una compra antes de que se apliquen las cargas de interés.

Tipo de cambio: Precio de una moneda en términos de otra. Los tipos de cambios resultan una importante información que orienta las transacciones internacionales de bienes, capital y servicios.

Tasa de interés: Porcentaje que se cobra como interés por una suma determinada. Las tasas de interés suelen denominarse activas cuando nos referimos a las que cobran los bancos y otras instituciones financieras que colocan su capital en préstamos a las empresas y los particulares, y pasivas, cuando nos referimos al interés que pagan dichas instituciones al realizar operaciones pasivas, es decir, cuando toman depósitos de ahorro o a plazo fijo.