Universidad de las Ciencias Informáticas

Facultad 8

Título: Sistema Web de Gestión de Préstamos externos de libros de literatura en la biblioteca de la Universidad de las Ciencias Informáticas (UCI).

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

Autor(es): Roimel Rafael Aguilar Ortiz.
Yadira Entenza Escobar.
Tutor(es): Ing. Ariel Orta Hernández.
Ing. Jorge Hernández Roselló.

Ciudad de La Habana

Junio, 2009

"Año del 50 Aniversario del triunfo de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaro que somos autores de este trabajo y autorizamos a la Biblioteca de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con el mismo.

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

Roimel Rafael Aguilar Ortiz **Yadira Entenza Escobar**

_____ _____

Firma del Autor Firma del Autor

Ing. Ariel Orta Hernández **Ing. Jorge Hernández Roselló**

_____ _____

Firma del Tutor Firma del Tutor

“El verdadero hombre no mira de qué lado se vive mejor, sino de qué lado está el deber”.

José Martí.

Dedicatoria

A nuestros padres, hermanos, familiares y amigos…

Agradecimientos Comunes

A nuestro Comandante en Jefe y a la Revolución por proporcionarnos la oportunidad de estudiar en un centro como este.

A la Universidad de las Ciencias Informáticas por formarnos como profesionales.

A nuestros tutores Ing. Ariel Orta Hernández e Ing. Jorge Hernández Roselló por su paciencia y dedicación.

A todos los profesores que de una forma u otra nos enseñaron a transitar en el camino por la búsqueda del conocimiento.

A nuestros amigos, a todos los que nos ayudaron……

Gracias

Agradecimientos

A mami, por estar siempre cuando la necesité y depositar toda su confianza en mí.

A mis compañeros de aula, por compartir tantos momentos que quedarán para siempre en mi memoria.

A mi familia, por estar siempre pendiente de mis estudios y a todos que de una forma u otra han hecho realidad este sueño……..

Yadira

A mis padres Rafael Ernesto Aguilar Pavón y Maritza Ortiz Brito que me dieron su apoyo en todo momento de mi carrera y siempre confiaron en mi.

A mis hermanas Madelaine Aguilar Ortiz y Marialis Aguilar Ortiz que no dejaron de darme tanto amor y cariño como necesité.

A Jose Carmenate Menéses quien ha sido uno más de mis hermanos.

Roimel

Resumen

Este trabajo propone el desarrollo de un sistema Web de gestión de préstamos externos de libros de literatura en la biblioteca de la Universidad de las Ciencias Informáticas (BiUCI). Actualmente nuestros estudiantes y profesores tienen la posibilidad de acudir a la BiUCI y solicitar el préstamo externo por un periodo de 7 días. Las bibliotecarias de dicha universidad para poder ofrecer este servicio tienen que llevar un control estricto de cada préstamo efectuado, debido al gran número de lectores de nuestra universidad, este trabajo se les hace engorroso pues actualmente todo este proceso se desarrolla a mano. En la actualidad, existe una página Web que registra todos los materiales de lectura existentes en la BiUCI, pero no específica si estos pueden ser solicitados por la comunidad estudiantil. Con la realización de esta investigación se propone poner en las manos de las bibliotecarias del centro un sistema que le ayude a administrar de forma general todo el proceso de préstamos externo de libros de literatura. Como también conocer si el material que busca está en disposición. Este documento recoge todo el trabajo realizado para la elaboración del sistema antes mencionado, incluyendo el estado del arte de aplicaciones con similares objetivos existentes en el mundo, el estudio y definición de las características del sistema, así como la planificación, diseño, codificación y pruebas del producto.

Índice

Índice de Tablas.

Índice de Figuras

Introducción:

Las bibliotecas guardan dentro de ella, los valores textuales de la sabiduría adquirida por los seres humanos durante su evolución. Este centro de enseñanza es la fuente de conocimientos a la que todos los individuos deben tener acceso de forma adecuada. Es un órgano vital que alimenta las mentes de todos aquellos interesados en adquirir conocimientos, refrescar sus ideas, aclarar dudas y hasta despejar sus mentes.

La Universidad de Las Ciencias Informáticas cuenta con una biblioteca. Dicho centro presta servicios de apoyo a la docencia, al estudio y a la investigación. Su objetivo básico es satisfacer las necesidades de información de la comunidad universitaria en general. Esta biblioteca cuenta con una o varias personas encargadas de la administración de los procesos que allí se desarrollan para brindar los mejores servicios a los lectores. Uno de estos procesos es el de préstamo externo de libros de literatura con los que cuenta la institución. El proceso de préstamos externos resulta muy engorroso, tanto para los prestatarios como para las bibliotecarias ya que se gestiona totalmente a mano.

Por lo planteado, la Dirección de La Biblioteca de la UCI (BiUCI) decidió informatizar este proceso como vía para alcanzar mejoras en la eficiencia y control.

Por tanto el **problema** a resolver queda formulado, a modo de interrogante, de la siguiente forma: ¿Cómo mejorar el proceso de préstamo externo de libros de literatura en la BiUCI?

El **objeto de estudio** lo constituye el proceso de préstamo externo de libros de literatura.

El **campo de acción** que abarca este trabajo es el proceso de préstamo externo de libros de literatura en la BiUCI.

Nuestra idea a defender consiste en que al contar con un sistema que controle el proceso de préstamo externo de libros de literatura en la BiUCI, se podrían alcanzar mejoras en la eficiencia y control.

El **objetivo general** de este trabajo es: Construir un sistema que automatice el proceso de préstamo externo de libros de literatura a estudiantes, profesores y trabajadores en la BiUCI.

De este objetivo general se desprenden **objetivos específicos** como:

- Estudiar como funciona el proceso de préstamo externo a estudiantes y profesores de la UCI.
- Realizar análisis y diseño de la aplicación.
- Automatizar un sistema que garantice la administración del proceso de préstamo externo de libros de literatura en la BiUCI.

Para el cumplimiento de estos objetivos se propone la realización de las siguientes **Tareas**:

- Investigar con los trabajadores de la BiUCI, el proceso que se lleva a cabo para realizar un préstamo externo de libros de literatura.
- Realizar un estudio detallado de las tendencias y tecnologías existentes para el manejo de los datos de préstamos en las bibliotecas.
- Investigar las formas de que el sistema brinde información detallada a las bibliotecarias con respecto a los préstamos efectuados.
- Estudiar, comparar y seleccionar las metodologías y herramientas existentes para el desarrollo de aplicaciones.
- Diseñar la Base de Datos.
- Desarrollar el análisis, diseño e implementación con la metodología y herramienta seleccionada.

1.1 Introducción.

En el presente capítulo se realizará un estudio acerca de cómo ocurre el proceso de préstamo externo de libros de literatura en la BiUCI, en el mismo se estudiarán los sistemas similares existentes, además se presentarán las tendencias y tecnologías actuales y se realizará una selección de aquellas que serán utilizadas durante el desarrollo del producto.

1.2 Estado del Arte.

Dedicar un espacio al tema de la lectura, los libros y lo que de ellos se puede aprender y transmitir, se ha convertido en una premisa necesaria para quienes tienen conciencia de la importancia de lo que los libros nos puedan proporcionar.

Los fundamentos y principios de los seres humanos, se basan en conocimientos adquiridos por sus propias experiencias, la educación es el reflejo de estos conocimientos. La preparación teórica prepara a las personas para el manejo de las situaciones prácticas que se presentan a lo largo de su vida diaria. El desarrollo adecuado de una sociedad bien organizada depende de una educación satisfactoria de todos los individuos que en ella se desenvuelven, para ello es de suma importancia, el poder contar con fuentes de información y cultura que permitan promover el aumento de la sabiduría y de los conocimientos de todos los eslabones que componen dicha sociedad.

Sistemas que gestionan los préstamos externos de libros en las bibliotecas a nivel Internacional.

En la actualidad en el ámbito internacional existen muchos sitios que publican como ocurre el proceso de préstamos externos en las bibliotecas, en dichos sitios se pueden encontrar información de vasta necesidad para todo aquel personal involucrado en este proceso; ejemplos de estos sitios son: www.elgranerocomun.net y el Sistema Integrado de Automatización de Bibliotecas (SIAB); también se pueden encontrar sitios, en los cuales sus principales servicios se definen en como presentar catálogos de diferentes géneros como son www.unicornio.biblored.org.co; www.biblioteca.universia.net; www.metabase.net, www.bilbao.net, entre otros.

A pesar de existir todos estos sitios antes mencionados, ninguno de estos sitios realiza préstamo de libros; sino que se puede encontrar una gran variedad de servicios como información de lo que ofrecen, secciones infantiles, secciones locales y catálogo el cual nos dice en qué biblioteca en el mundo puedo encontrar un determinado libro.

Sistemas que gestionan los préstamos externos de libros en las bibliotecas a nivel Nacional.

A nivel nacional, se cuenta con una gran cantidad de portales en los que se puede encontrar cualquier información; como por ejemplos: El Sitio Web de la Cultura de Manzanillo www.crisol.cult.cu y la, Biblioteca Zaira Rodríguez Ugido www.filosofia.cu, también se puede apreciar el portal http://www.bnjm.cu ¨Biblioteca Nacional José Martí¨, en la cual se brinda la posibilidad de interactuar con la cultura, la ciencia, la historia y el arte, la Biblioteca Nacional Médica cuenta también con un portal http://www.sld.cu/sitios/bmn. A pesar de existan varios portales cubanos en beneficio de la información bibliotecaria, no se cuenta con uno que preste el servicio de préstamos de libros en línea.

Sistemas que gestionan los préstamos externos de libros en la biblioteca de la UCI.

En la universidad de las Ciencias Informáticas se cuenta con el portal http://biblioteca.uci.cu . Este brinda gran cantidad de información; pero no contiene funcionalidades que permitan realizar préstamos de libros de literatura. Además de este portal la universidad, cuenta con varios sitios publicados en internet como http://biblioteca.filosofia.cu/, http://www.bvv.sld.cu/.

¿Por qué esta propuesta?

La razón fundamental de esta propuesta es dotar a las bibliotecarias de la BiUCI con una herramienta que les permita automatizar el proceso que se lleva a cabo en el préstamo externo de un libro de literatura, soportada esta herramienta en una plataforma Web.

1.3 Arquitectura Cliente/Servidor.

¿Qué es una arquitectura?

Una arquitectura es un entramado de componentes funcionales que, aprovechando diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios informáticos. Se debe señalar que para seleccionar el modelo de una arquitectura, hay que partir del contexto

tecnológico y organizativo del momento y, que la arquitectura Cliente/Servidor requiere una determinada especialización de cada uno de los diferentes componentes que la integran.

¿Qué es un cliente?

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

¿Qué es un servidor?

Es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LANs o WANs, para proveer de múltiples servicios a los clientes y ciudadanos tales como impresión, acceso a bases de datos, fax, procesamiento de imágenes, etc.

Ventajas de la arquitectura cliente-servidor:

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

1.4 Técnicas y Tecnologías del lado del Cliente.

Las técnicas y tecnologías del lado del cliente son aquellas que se utilizan para desarrollar programas que se ejecutaran en los mismos, entiéndase por cliente a los navegadores: Firefox, Internet Explorer, Opera, Safari entre otros. Estos son los encargados de visualizar toda la información solicitada por los usuarios, dicha información el navegador es capaz de leerla ya que este interpreta código HTML y algunos lenguajes script como son VBScript y Java Script.

1.4.1 CSS (Hojas de Estilo en Cascada)

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación; y es imprescindible para crear páginas web complejas.

Dicho lenguaje se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista. (Eguíluz Pérez, 2008)

Las ventajas de utilizar CSS son:

- Obliga a crear documentos HTML/XHTML bien definidos y con significado completo.
- Mejora la accesibilidad del documento.
- Reduce la complejidad de su mantenimiento.
- Permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Desventaja en el uso del CSS

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

1.4.2 HTML (HiperTexto Markup Language)

El HTML es el lenguaje utilizado para representar documentos en la WWW (World Wide Web). Además de texto normal incluye también, elementos multimedia (gráficos, vídeo, audio) y existen enlaces (links) que permiten saltar a otras partes del documento o a otro sitio cualquiera de Internet. La descripción se basa en especificar en el texto la estructura lógica del contenido (títulos, párrafos de texto normal, enumeraciones, definiciones, citas, etc.), así como los diferentes efectos que se quieren dar (cursiva, negrita, o un gráfico determinado) y dejar que luego la presentación final de dicho hipertexto se realice por un programa especializado. (Santamaría, 2007)

Otra característica muy importante de este lenguaje es que es portable, es decir, se pueden visualizar las páginas con cualquier sistema operativo y, por supuesto también crearlas.

1.4.3 AJAX (Asynchronous JavaScript and XML)

AJAX es una técnica de desarrollo Web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios

sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma. (Eguíluz Pérez, 2008)

AJAX es una combinación de cuatro tecnologías ya existentes:

- XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.
- Document Object Model (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
- El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor Web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto iframe en lugar del XMLHttpRequest para realizar dichos intercambios.
- XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML preformateado, texto plano y JSON.

AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente.

1.4.4 JavaScript

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

Ventajas:

- No requiere tiempo de compilación.
- Los scripts pueden desarrollarse en un período de tiempo relativamente corto.
- Posee características de interfaz, que son gestionados por el navegador y por el código HTML.
- Los programas JavaScript tienden a ser pequeños y compactos, no requieren mucha memoria ni tiempo adicional de transmisión.

- Es independiente de la plataforma de hardware o sistema operativo, siempre y cuando exista un navegador con soporte JavaScript.

Desventajas:

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

Técnicas y tecnologías del lado del cliente seleccionadas

En el estudio realizado anteriormente sobre las técnicas y tecnologías del lado del cliente, se llego a la conclusión de que en el desarrollo de dicha aplicación se utilizara las siguientes herramientas: HTML, CSS y JavaScript; dicha elección fue realizada teniendo en cuenta las ventajas que estas tecnologías nos pueda proporcionar en el desarrollo de la investigación.

1.5 Tecnologías del lado del servidor.

Las tecnologías del lado del servidor son aquellos programas o servicios que corren en un servidor remoto y que brindan funcionalidades al sistema.

1.5.1 ASP (Active Server Pages)

ASP dentro de esta tecnología el mayor inconveniente es que se trata de un sistema propietario que es usado nativamente sólo por Microsoft Internet Information Server (IIS). ASP es un lenguaje más lento y pesado que PHP, y también menos estable. Algunas de las ventajas de dicho lenguaje consisten en que debido a que usa principalmente VBScript, es relativamente simple tratar con el lenguaje si ya se conoce cómo programar en Visual Basic. El soporte de este lenguaje también se encuentra habilitado por defecto en el servidor IIS, facilitando su instalación y ejecución. Los componentes integrados en ASP son bastante limitados, de modo que si necesita usar características "avanzadas", como interactuar con servidores FTP, necesita comprar componentes adicionales. ASP es un sistema con nula portabilidad pues requiere necesariamente de un servidor Windows, con todas las implicaciones de alto costo, poca flexibilidad y escasa seguridad que estos equipos conllevan. (Anónimo, 2001)

1.5.2 JSP

Basado en Java, orientado a objetos, multiplataforma. Es un software de Sun Microsystems Inc, con gran experiencia y actualizaciones habituales y con un proyecto de desarrollo libre a partir de la versión 1.2. Existe una gran comunidad de Java pero más orientada a aplicaciones de escritorio que a aplicaciones Web. JSP hace gran énfasis en los componentes y no tanto en los scripts, permitiendo obtener un buen rendimiento y escalabilidad. Permite el trabajo con múltiples BD, sencillo de implementar con MySQL. Es necesario que disponga de, por ejemplo, TomCat para poder ejecutarlo**.** Dos de las grandes desventajas de JSP es, por una parte, una excesiva complejidad y por otra (paradójicamente) una aproximación elemental. (Anónimo, 2002)

La excesiva complejidad deriva de enfocar cualquier problema como un EJB (enterprise java bean), lo que muchas veces lleva a consumir grandes cantidades de tiempo y dinero. La aproximación elemental radica en que muchas empresas usan JSP como ASP, es decir, como una manera de hacer sus páginas dinámicas pero sin integrarlas a todos los módulos Java.

1.5.3 PHP (acrónimo de "PHP: Hypertext Preprocessor")

PHP (Hypertext Preprocessor) es un lenguaje de "código abierto" interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor. Se trata de un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios web. Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite; lo cual permite la creación de Aplicaciones Web muy robustas. (Anónimo, 2001)

Ventajas.

- Es un lenguaje multiplataforma, funciona tanto para Unix (con Apache) como para Windows (con Microsoft Internet Information Server).
- Similar en sintaxis a C y a PERL.
- Soporta en cierta medida la orientación a objeto. Clases y herencia.
- El análisis léxico para recoger las variables que se pasan en la dirección lo hace PHP de forma automática. Evitando que el usuario tenga que separar las variables y sus valores.
- Se puede incrustar código PHP con etiquetas HTML.
- Excelente soporte de acceso a base de datos.

- La comprobación de que los parámetros son válidos se hace en el servidor y no en el cliente (como se hace con javascript) de forma que se puede chequear que no se reciban solicitudes adulteradas.
- Es software libre.
- Se puede obtener en la web y su código está disponible bajo la licencia GPL.

Desventajas:

- Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.

Lenguaje del lado del servidor seleccionado (PHP)

Con el estudio de las técnicas y tecnologías del lado del servidor realizado con anterioridad, se llegó a la conclusión de que el lenguaje a utilizar en la implementación de la aplicación propuesta será PHP; ya que entre las principales características de dicho lenguaje es que es multiplataforma y es software libre.

1.6 Gestor de Base de Datos

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

1.6.1 MySQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario. Funciona en diferentes plataformas. Posee un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite verificación basada en el host. Las contraseñas son seguras porque todo el tráfico de contraseñas está encriptado cuando se conecta con un servidor. Soporta a grandes bases de datos. Un índice puede usar prefijos de una columna para los tipos de columna CHAR, VARCHAR, BLOB, o TEXT. MySQL Server tiene soporte para comandos SQL para chequear, optimizar, y reparar tablas. (ProgramacionWeb, 2003_2009)

Ventajas:

- La velocidad a la hora de realizar las operaciones, lo que le hace uno de los gestores que ofrecen mayor rendimiento.
- Su bajo consumo lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Las utilidades de administración de este gestor son envidiables para muchos de los gestores comerciales existentes, debido a su gran facilidad de configuración e instalación.
- Tiene una probabilidad muy reducida de corromper los datos, incluso en los casos en los que los errores no se produzcan en el propio gestor, sino en el sistema en el que está.
- El conjunto de aplicaciones Apache-PHP-MySQL es uno de los más utilizados en Internet en servicios de foro y de buscadores de aplicaciones.

Desventajas:

- Carece de soporte para transacciones, rollback y subconsultas.
- El hecho de que no maneje la integridad referencial, hace de este gestor una solución pobre para muchos campos de aplicación, sobre todo para aquellos programadores que provienen de otros gestores que sí que poseen esta característica.
- No es viable para su uso con grandes bases de datos, a las que se acceda continuamente, ya que no implementa una buena escalabilidad.

1.6.2 Oracle

Oracle es un sistema de bases de datos relacional que se destaca por ofrecer soporte de transacciones, estabilidad, escalabilidad, además de ser multiplataforma. Por estas características se le considera como uno de los más completos. Las últimas versiones han sido certificadas para poder trabajar sobre plataforma Linux (Expertos, 2009). Su principal defecto es su enorme precio, que es de varios miles de euros, además de criticársele la política de suministro de parches de seguridad desde el 2005, lo cual ha incrementado el nivel de exposición de los usuarios en lo que a seguridad respecta.

Ventajas:

- Posee igual interacción en todas las plataformas (Windows, Unix, Macintosh y Mainframes). Ya que más del 80% de los códigos internos de Oracle son iguales a los establecidos en todas las plataformas de Sistemas Operativos.

- Oracle soporta bases de datos de todos los tamaños, desde severas cantidades de bytes y gigabytes en tamaño.
- Soporte de transacciones.
- La realización de backups (copias de seguridad) es fácil.
- Es eficaz y eficiente.
- Tiene una amplia gama de herramientas para operar con la Base de Datos tanto como usuario y como administrador.

Desventajas:

- Elevado precio comercial.
- La seguridad de la plataforma, y las políticas de suministro de parches de seguridad, modificadas a comienzos de 2005 y que incrementan el nivel de exposición de los usuarios.
- Exige una gran cantidad de recursos de la máquina donde se encuentre instalado.

1.6.3 Microsoft SQL Server

Microsoft SQL Server es también un sistema de bases de datos relacional, entre sus características se destacan el soporte de transacciones, su escalabilidad, estabilidad y seguridad. Incluye un potente entorno grafico de administración. Este sistema constituye la alternativa de Microsoft a otros potentes gestores como Oracle, PostgreSQL o MySQL. Su principal dificultad radica en que no es multiplataforma, solo esta disponible en Sistemas Operativos de Microsoft. (Microsoft)

Entre sus ventajas están:

- Soporte de transacciones.
- Gran estabilidad.
- Gran seguridad.
- Escalabilidad.
- Soporta procedimientos almacenados.
- Incluye también un potente entorno gráfico de administración, que permite el uso de comandos Lenguaje de Definición de Datos (DDL) y Lenguaje de Manipulación de Datos (DML) gráficamente.
- Permite trabajar en modo cliente-servidor donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información.

- Además permite administrar información de otros servidores de datos.

Desventajas:

- Tiene soporte solamente en el sistema operativo Windows.
- Es un software de licencia propietaria.

1.6.4 PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS), es una derivación libre (OpenSource) del proyecto POSTGRES, esta versión incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional, pero a pesar de esto PostgreSQL no es un sistema de gestión de bases de datos puramente orientado a objetos. (Superuser, 2003)

Unas de sus principales características es que soporta distintos tipos de datos, además de los del tipo base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...), cadenas de bits, etc. y permite la creación de tipos propios. (Quiñones, 2004)

Ventajas:

- Posee una gran escalabilidad. Es capaz de ajustarse al número de Unidades Centrales de Procesamiento (CPUs) y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta.
- Implementa el uso de rollback, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz.
- Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.
- El único costo asociado a él, es el de conocerlo pues su código fuente está disponible bajo la más liberal de las licencias del OpenSource: la licencia BSD, que permite usarlo modificarlo y distribuirlo en productos comerciales o no comerciales, sin costo alguno.
- Requiere pocos recursos de hardware y la simplificación del proceso de administración de licencias de software, que no es necesario cuando se usa software libre.
- PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.
- Puede lidiar con gran volumen de datos.

Desventajas:

- Tiene un límite de 8K por fila, aunque se puede aumentar a 32K, con una disminución considerable del rendimiento.
- Es lento.

Gestor de Base Datos seleccionado (MySQL)

Para el desarrollo del sistema se realizó un detallado estudio de los Gestores de Base de Datos, el cual permitió que se pudiera realizar una elección certera. En dicho estudio se llegó a la conclusión de que el Gestor que se utilizará es MySQL, por las siguientes razones:

- Es un sistema gestor de bases de datos “Open Source”.
- Presenta múltiples motores de almacenamiento, permitiendo al usuario escoger el que sea más adecuado para cada tabla de la base de datos.
- Agrupación de transacciones, reuniendo múltiples transacciones de varias conexiones para incrementar el número de transacciones por segundo.
- MySQL está escrito en una mezcla de C y C++.

Además de las facilidades antes mencionadas se decidió escoger este gestor pues es muy usado en la universidad por ser muy rápido en aplicaciones de este tipo.

1.7 Frameworks.

En el desarrollo de software, un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros softwares para ayudar a desarrollar y unir los diferentes componentes de un proyecto. (Gutiérrez, 2008)

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

1.7.1 CodeIgniter

CodeIgniter es un buen framework, utilizado por una gran comunidad de usuarios. Construido para codificadores PHP que necesitan una herramienta de desarrollo fácil para crear aplicaciones web simples y elegantes.
Entre sus características podemos encontrar su compatibilidad con PHP 4 y PHP 5, incorpora el modelo MVC, soporte para múltiples bases de datos, plantillas, validaciones, no requiere instalación, podemos encontrar una librería con un gran número de clases. (Rivera, 2007)

1.7.2 CakePHP

CakePHP es un Framework similar a CodeIgniter de desarrollo rápido. Es una estructura de librerías y clases para programar aplicaciones web. Su base es el Framework de Ruby on Rails. Nos brinda la posibilidad de interactuar con las base de datos, usando ActiveRecord. Incorpora el patrón MVC, compatible con PHP4 y PHP5, URLs amigables, Soporta AJAX, incluye caching, validación. (Anónimo)

1.7.3 Zoop Framework

Zoop es un Framework PHP Orientado a Objeto basado en el modelo MVC, sus desarrolladores lo caracterizan por ser rápido, eficiente y fácil destinado para programadores.

Requiere PHP 4.3.10 o superior, además podemos contar con librerías para “PEAR”. Cuenta con soporte e integración con AJAX, caching, validación, sistemas de plantillas Smarty, creación de PDF, plantillas para el envío de correo electrónico usando SMTP, cuenta con compatibilidad con múltiples base de datos.

1.7.4 Symfony Framework

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas

estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. (Zaninotto, y otros, 2009)

Symfony está desarrollado completamente con PHP5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows.

Características de Symfony

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales, adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Según el estudio realizado sobre los frameworks para php. Pudimos ver las posibilidades que estos nos brindan para el desarrollo de cualquier sistema web. En la investigación realizada, se seleccionó el Symfony por las ventajas que este nos proporcionaba; pero en el transcurso de la misma, percibimos que teníamos que dedicarle mucho tiempo en el estudio del framework seleccionado y esto conllevaría a un atraso en el desarrollo de la aplicación. Por lo que decidimos no utilizar ningún framework, ya que nuestro sistema, por lo sencillo y pequeño que es no lo necesita.

1.8 Servidor de aplicaciones Web

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

1.8.1 IIS

IIS (Internet Information Server), es una serie de servicios para los ordenadores que funcionan con Windows. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS. Este servicio convierte a un ordenador en un servidor de Internet o Intranet es decir que en las computadoras que tienen este servicio instalado se pueden publicar páginas web tanto local como remotamente (servidor web).

El servidor web se basa en varios módulos que le dan capacidad para procesar distintos tipos de páginas, por ejemplo Microsoft incluye los de Active Server Pages (ASP) y ASP.NET. También pueden ser incluidos los de otros fabricantes, como PHP o Perl.

Versiones

- IIS 1.0, Windows NT 3.51 Service Pack 3.
- IIS 2.0, Windows NT 4.0.
- IIS 3.0, Windows NT 4.0 Service Pack 3.
- IIS 4.0, Windows NT 4.0 Option Pack.
- IIS 5.0, Windows 2000.
- IIS 5.1, Windows XP Professional.
- IIS 6.0, Windows Server 2003 y Windows XP Profesional x64 Edition.
- IIS 7.0, Windows Vista (Solo Bussines) y Windows Server 2008.

1.8.2 Thttpd

Thttpd es un servidor web de código libre disponible para la mayoría de las variantes de Unix. Se caracteriza por ser simple, pequeño, portátil, rápido, y seguro, ya que utiliza los requerimientos mínimos de un servidor HTTP. Esto lo hace ideal para servir grandes volúmenes de información estática. Se caracteriza por ser:

- *Simple*, ya que sólo maneja el mínimo necesario para poner en práctica el protocolo HTTP, algunas veces un poco más que el mínimo.
- *Pequeño*, porque también tiene un pequeño tamaño de período de explotación, ya que esto no se divide en dos partes y es muy cuidadoso sobre la asignación de memoria.
- *Portátil*, pues se compila limpiamente sobre la mayoría de sistemas operativos, expresamente incluyendo FreeBSD, SunOS 4, Solaris 2, BSD/OS, GNU/Linux, OSF.
- *Rápido*, porque en el empleo típico es sobre todo más rápido que los mejores servidores "destacados" (Apache), y bajo la carga extrema es mucho más rápido.
- *Seguro*, porque se extiende a grandes longitudes para proteger el servidor Web contra ataques de otros sitios.

Ventajas

- El administrador puede decidir restringir la transferencia de archivos de imagen JPEG a en la mayor parte de 20 KB por segundo.
- Los promedios de carga se caen debido a la reducción de la transferencia gráfica.

Desventaja

- No posee las mismas aplicaciones que se pueden obtener de un software estándar como lo es el Apache.

1.8.3 Lighttpd

Lighttpd es un servidor web diseñado para ser rápido, seguro, flexible, y fiel a los estándares. Está optimizado para entornos donde la velocidad es muy importante, y por eso consume menos CPU y memoria RAM, incluso en servidores con 512MB funcionará de maravilla. Por todo lo que ofrece, lighttpd es apropiado para cualquier servidor que tenga problemas de carga. Este servidor es software libre y se distribuye bajo la licencia BSD. Funciona en GNU/Linux y UNIX de forma oficial. Para Microsoft Windows actualmente hay una distribución conocida como Lighttpd For Windows mantenida por Kevin Worthington. También permite comunicarse con programas externos mediante FastCGI o SCGI, que son mejoras al CGI original (también soportado). De esta forma, se pueden usar programas en prácticamente cualquier lenguaje de programación. Dicho servidor se caracteriza por:

- Alojar varios dominios en la misma IP (Virtual hosting).
- Soporta varios lenguajes de programación (PHP, Ruby, y otros).
- Consumo de memoria constante.
- Redirecciones HTTP y reescrituras de URL.
- También permite otros sistemas de notificación de eventos como Kqueue y epoll.
- Muestra un listado de ficheros cuando se entra a un directorio sin index.html.
- Permite módulos externos.
- Acepta parte de WebDAV.

1.8.4 Apache

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

Características principales:

- Trabaja sobre múltiples plataformas (Unix, Linux, MacOSX, Vms, Win32, OS2, etc.).
- Incluye módulos que se cargan de forma dinámica.
- Soporta CGI, Perl, PHP.
- Soporte para Bases de datos.
- Soporte SSL para transacciones seguras.
- Incluye soporte para host virtuales.
- Soporta HTTP 1.1.
- Código Abierto.
- Rápido.
- Eficiente.

1.8.5 Servidor de aplicaciones web seleccionado (Apache)

Luego de hacerse un estudio profundo de los servidores web, se decidió que para el entorno a desarrollar el mejor servidor web es el Apache por las siguientes razones:

Ventajas

- Ayuda en la mejora del posicionamiento.
- Este servidor junto con el módulo mod_rewrite puede convertirse en una herramienta muy jugosa para crear páginas con enlaces amigables para los buscadores,;
- Es un software libre.
- Multiplataforma.
- Open Source.
- Modular.
- Extensible.
- Presenta mensajes de error altamente configurables.

1.9 Metodologías de desarrollo de software.

Todo desarrollo de software se torna riesgoso y difícil de controlar, pero si no se utiliza una metodología, existe una mayor exposición a obtener clientes y desarrolladores insatisfechos con el resultado. Para evitar estos incidentes se debe llegar a un acuerdo formal con el cliente, al inicio del proyecto, de tal manera que cada cambio o modificación no perjudique el desarrollo del mismo, así como escoger una metodología adecuada para el desarrollo del software que sirva como guía para realizar de forma disciplinada y eficiente el producto deseado. (Escobar, 1997)

1.9.1 Programación Extrema (XP)

Programación Extrema es una metodología de desarrollo que se encuentra dentro de las llamadas metodologías ágiles en las que se da máxima prioridad a la obtención de resultados. XP no se basa en principios nuevos, sino que todas o casi todas sus características ya se conocen dentro de la ingeniería del Software, las cuales se complementan para minimizar los típicos problemas que pueden surgir en todo desarrollo de proyectos software. Ésta metodología promueve la idea de que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista

que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. (Castillo, 2007)

Las características fundamentales de esta metodología son:

- Desarrollo iterativo e incremental.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
- Programación en parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto.
- Frecuente integración del equipo de programación con el cliente o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento.
- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto.
- Simplicidad en el código: Es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo en caso necesario, que realizar algo complicado y quizás nunca utilizarlo.

1.9.1.1 Fases de XP

1. Planificación del proyecto.

Historias de usuario: Las historias de usuario tienen la misma finalidad que los casos de uso pero con algunas diferencias: Constan de 3 ó 4 líneas escritas por el cliente en un lenguaje no técnico sin hacer mucho hincapié en los detalles; no se debe hablar ni de posibles algoritmos para su implementación ni de diseños de base de datos adecuados, etc. Son usadas para estimar tiempos de desarrollo de la parte de la aplicación que describen. También se utilizan en la fase de pruebas, para verificar si el programa cumple con lo que especifica la historia de usuario. Cuando llega la hora de implementar una historia de usuario, el cliente y los desarrolladores se reúnen para concretar y detallar lo que tiene que hacer dicha historia. El tiempo de desarrollo ideal para una historia de usuario es entre 1 y 3 semanas.

Release planning: .Después de tener ya definidas las historias de usuario es necesario crear un plan de publicaciones, en inglés "Release plan", donde se indiquen las historias de usuario que se crearán para cada versión del programa y las fechas en las que se publicarán estas versiones. Un "Release plan" es una planificación donde los desarrolladores y clientes establecen los tiempos de implementación ideales de las historias de usuario, la prioridad con la que serán implementadas y las historias que serán implementadas en cada versión del programa.

Iteraciones Todo proyecto que siga la metodología XP se ha de dividir en iteraciones de aproximadamente 3 semanas de duración. Al comienzo de cada iteración los clientes deben seleccionar las historias de usuario definidas en el "Release planning" que serán implementadas. También se seleccionan las historias de usuario que no pasaron el test de aceptación que se realizó al terminar la iteración anterior. Estas historias de usuario son divididas en tareas de entre 1 y 3 días de duración que se asignarán a los programadores.

Velocidad del proyecto: La velocidad del proyecto es una medida que representa la rapidez con la que se desarrolla el proyecto; estimarla es muy sencillo, basta con contar el número de historias de usuario que se pueden implementar en una iteración; de esta forma, se sabrá el cupo de historias que se pueden desarrollar en las distintas iteraciones. Usando la velocidad del proyecto controlaremos que todas las tareas se puedan desarrollar en el tiempo del que dispone la iteración. Es conveniente reevaluar esta medida cada 3 ó 4 iteraciones y si se aprecia que no es adecuada hay que negociar con el cliente un nuevo "Release Plan".

Programación en pareja: La metodología XP aconseja la programación en parejas pues incrementa la productividad y la calidad del software desarrollado. El trabajo en pareja involucra a dos programadores trabajando en el mismo equipo; mientras uno codifica haciendo hincapié en la calidad de la función o método que está implementando, el otro analiza si ese método o función es adecuado y está bien diseñado. De esta forma se consigue un código y diseño con gran calidad.

Reuniones diarias. Es necesario que los desarrolladores se reúnan diariamente y expongan sus problemas, soluciones e ideas de forma conjunta. Las reuniones tienen que ser fluidas y todo el mundo tiene que tener voz y voto.

2. Diseño.

Diseños simples: La metodología XP sugiere que hay que conseguir diseños simples y sencillos. Hay que procurar hacerlo todo lo menos complicado posible para conseguir un diseño fácilmente entendible e impleméntable que a la larga costará menos tiempo y esfuerzo desarrollar.

Glosarios de términos: Usar glosarios de términos y un correcta especificación de los nombres de métodos y clases ayudará a comprender el diseño y facilitará sus posteriores ampliaciones y la reutilización del código.

Riesgos: Si surgen problemas potenciales durante el diseño, XP sugiere utilizar una pareja de desarrolladores para que investiguen y reduzcan al máximo el riesgo que supone ese problema.

Funcionalidad extra: Nunca se debe añadir funcionalidad extra al programa aunque se piense que en un futuro será utilizada. Sólo el 10% de la misma es utilizada, lo que implica que el desarrollo de funcionalidad extra es un desperdicio de tiempo y recursos.

Refactorizar. Refactorizar es mejorar y modificar la estructura y codificación de códigos ya creados sin alterar su funcionalidad. Refactorizar supone revisar de nuevo estos códigos para procurar optimizar su funcionamiento. Es muy común rehusar códigos ya creados que contienen funcionalidades que no serán usadas y diseños obsoletos. Esto es un error porque puede generar código completamente inestable y muy mal diseñado; por este motivo, es necesario refactorizar cuando se va a utilizar código ya creado.

Tarjetas C.R.C. El uso de las tarjetas C.R.C (Class, Responsabilities and Collaboration) permiten al programador centrarse y apreciar el desarrollo orientado a objetos olvidándose de los malos hábitos de la programación procedural clásica.

Las tarjetas C.R.C representan objetos; la clase a la que pertenece el objeto se puede escribir en la parte de arriba de la tarjeta, en una columna a la izquierda se pueden escribir las responsabilidades u objetivos que debe cumplir el objeto y a la derecha, las clases que colaboran con cada responsabilidad.

3. Codificación.

El cliente es una parte muy importante en el equipo de desarrollo; su presencia es indispensable en las distintas fases de XP. A la hora de codificar una historia de usuario su presencia es aún más necesaria. No olvidemos que los clientes son los que crean las historias de usuario y negocian los tiempos en los que serán implementadas. Antes del desarrollo de cada historia de usuario el cliente debe especificar detalladamente lo que ésta hará y también tendrá que estar presente cuando se realicen los test que verifiquen que la historia implementada cumple la funcionalidad especificada.

La codificación debe hacerse ateniendo a estándares de codificación ya creados. Programar bajo estándares mantiene el código consistente y facilita su comprensión y escalabilidad.

4. Pruebas.

Uno de los pilares de la metodología XP es el uso de test para comprobar el funcionamiento de los códigos que vayamos implementando.

El uso de los test en XP es el siguiente:

Se deben crear las aplicaciones que realizarán los test con un entorno de desarrollo específico para test.

Hay que someter a tests las distintas clases del sistema omitiendo los métodos más triviales. Se deben crear los test que pasarán los códigos antes de implementarlos; en el apartado anterior se explicó la importancia de crear antes los test que el código.

Un punto importante es crear test que no tengan ninguna dependencia del código que en un futuro evaluará. Hay que crear los test abstrayéndose del futuro código, de esta forma aseguraremos la independencia del test respecto al código que evalúa.

El uso de los test es adecuado para observar la refactorización. Los test permiten verificar que un cambio en la estructura de un código no tiene porqué cambiar su funcionamiento.

Test de aceptación. Los test mencionados anteriormente sirven para evaluar las distintas tareas en las que ha sido dividida una historia de usuario. Para asegurar el funcionamiento final de una determinada historia de usuario se deben crear "Test de aceptación"; estos test son creados y usados por los clientes para comprobar que las distintas historias de usuario cumplen su cometido.

Al ser las distintas funcionalidades de nuestra aplicación no demasiado extensas, no se harán test que analicen partes de las mismas, sino que las pruebas se realizarán para las funcionalidades generales que debe cumplir el programa especificado en la descripción de requisitos.

1.9.2 El Proceso Unificado de Modelado (RUP)

El Proceso Unificado es un proceso de software genérico que puede ser utilizado para una gran cantidad de tipos de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de competencia y diferentes tamaños de proyectos. Dicho proceso divide en 4 fases el desarrollo del software:

- Inicio: El Objetivo en esta etapa es determinar la visión del proyecto.
- Elaboración: En esta etapa el objetivo es determinar la arquitectura óptima.
- Construcción: En esta etapa el objetivo es llegar a obtener la capacidad operacional inicial.
- Transición: El objetivo es llegar a obtener el release del proyecto.

Cada una de estas etapas es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes. (Anónimo)

El ciclo de vida que se desarrolla por cada iteración, es llevado bajo dos disciplinas:

Disciplina de Desarrollo:

- Ingeniería de Negocios: Entendiendo las necesidades del negocio.
- Requerimientos: Trasladando las necesidades del negocio a un sistema automatizado.
- Análisis y Diseño: Trasladando los requerimientos dentro de la arquitectura de software.
- Implementación: Creando software que se ajuste a la arquitectura y que tenga el comportamiento deseado.
- Pruebas: Asegurándose que el comportamiento requerido es el correcto y que todo lo solicitado está presente.

Disciplina de Soporte:

- Configuración y administración del cambio: Guardando todas las versiones del proyecto.
- Administrando el proyecto: Administrando horarios y recursos.
- Ambiente: Administrando el ambiente de desarrollo.
- Distribución: Hacer todo lo necesario para la salida del proyecto.

Es recomendable que a cada una de estas iteraciones se les clasifique y ordene según su prioridad, y que cada una se convierte luego en un entregable al cliente. Esto trae como beneficio la retroalimentación que se tendría en cada entregable o en cada iteración.

Los elementos del RUP son:

- Actividades: Son los procesos que se llegan a determinar en cada iteración.
- Trabajadores: Son las personas involucradas en cada proceso.
- Artefactos: Un artefacto puede ser un documento, un modelo, o un elemento de modelo.

1.9.2.1 Características del Proceso Unificado.

Dirigido por casos de uso: Los casos de uso 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. Un caso de uso es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado importante. Representan requerimientos funcionales. Los casos de uso guían la arquitectura del sistema y la arquitectura del sistema influye en la selección de los casos de uso.

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. La arquitectura se refleja en los casos de uso pues cada producto tiene tanto una función como una forma, ninguna es suficiente por sí sola.

Iterativo e Incremental: RUP propone dividir el trabajo en partes más pequeñas o miniproyectos, donde cada mini proyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto. Cada iteración se realiza de forma planificada es por eso que se dice que son miniproyectos.

Una iteración es una secuencia de actividades con un plan establecido y criterios de evaluación las cuales están estrechamente relacionadas, cuyo resultado es una versión del software.

Metodología de desarrollo seleccionada (XP)

Para obtener una mejor organización, eficiencia y eficacia en el desarrollo de dicha aplicación se realizó un estudio detallado de dos de las metodologías más usadas a nivel mundial: XP y RUP; después de haber realizado la investigación se llegó a la conclusión de que se utilizará la metodología ágil XP, ya que dicha metodología:

Ventajas

- Atribuye una mayor satisfacción por parte del programador.
- Genera una menor taza de errores.
- Tiene una programación organizada.
- Es recomendable solo para proyectos a corto plazo.

1.10 Herramientas CASE.

Visual Paradigm

Es una herramienta CASE que ofrece un entorno de creación de diagramas para UML, diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad; uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación; capacidades de ingeniería directa (versión profesional) e inversa; modelo y código que permanece sincronizado en todo el ciclo de desarrollo; disponibilidad de múltiples versiones, para cada necesidad; disponibilidad de integrarse en los principales IDEs; disponibilidad en múltiples plataformas, y muy útil para la generación de código fuente en PHP, también con el Paradigm se generan script de las tablas de salidas para las clases persistentes.

Beneficios:

- Navegación intuitiva entre el modelo visual y el código.
- Poderosa herramienta de generación de PDF/HTML a partir de diagramas UML.
- Sincronización entre el código fuente y el modelo en tiempo real o bajo demanda.
- Entorno visual de modelado superior.
- Soporte para toda la notación UML.

- Sofisticados y automáticos diagramas de capas.
- Análisis de textos.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de integrarse en los principales IDEs.
- Disponibilidad en múltiples plataformas.

Conclusiones.

En este capítulo se detallaron las condiciones y problemas que rodean el objeto de estudio; y a través de los conceptos y definiciones planteadas, se determinaron las condiciones específicas que envuelven al mismo. Se analizaron las características de diferentes herramientas, para la creación de un software o aplicación así como algunas metodologías.

Después de este análisis y la fundamentación realizada, el lenguaje de programación que se utilizará es PHP, HTML, CSS y JavaScript para el control del las diferentes funciones a realizar en el cliente, como gestor de base de datos se estableció el MySQL.

Se empleará la metodología XP, como lenguaje de modelado UML y la herramienta que se usará como entorno de creación de diagramas para UML el Visual Paradigm. Una vez conocidas las herramientas óptimas y los conceptos a utilizar se puede empezar a desarrollar la propuesta de sistema.

Capítulo 2: Solución Propuesta

2.1 Introducción.

En este capítulo se explica cómo funciona el proceso de préstamos externos de libros de literatura en la BiUCI. Se describe además la solución propuesta y se planifica su desarrollo, haciendo hincapié en la planificación de las iteraciones, plan de entrega y las fases iniciales del proceso de ingeniería de software.

2.2 Descripción de la solución propuesta.

La herramienta a diseñar en la presente investigación, debe permitir a los administradores del sistema, gestionar los préstamos externos de los libros de literatura existentes en la biblioteca de la universidad y gestionar los lectores que en dicha biblioteca soliciten un libro, mediante una interfaz gráfica, sencilla y amigable. Además será capaz de mostrar en una interfaz, el reporte de todos los préstamos efectuados en un mes para cada facultad.

Este proceso funcionará de la siguiente manera: Los estudiantes, profesores o trabajadores de la Universidad de las Ciencias Informáticas acuden a la biblioteca del centro con el objetivo de solicitar un préstamo. Una vez allí tienen la obligación de buscar en el catálogo el libro que desean y ver el identificador que tiene. Luego se dirige a la persona encargada de realizar el préstamo y con el identificador encontrado lo solicita. La compañera verifica que el libro en cuestión esté disponible y que el lector no tenga préstamos pendientes. Si no hay problemas, procede a la captura de los datos del lector y registra en la base de datos el préstamo.

2.3 Desarrollo del Sistema Web de Gestión de Préstamos externos de libros de literatura.

El desarrollo del sistema para la gestión de préstamos externos, es una parte de todos los servicios que se prestan en la BiUCI, el cual constituye un aporte para el trabajo que realizan las bibliotecarias en dicho centro. Este sistema brindará la posibilidad de mostrar o eliminar los préstamos activos que existan hasta una fecha determinada. Así como poder actualizar, insertar, o eliminar de la base de datos a los lectores, y administradores. El sistema mostrará un reporte con el resultado de un conteo los préstamos efectuados

en el mes por cada una de las facultades y dividido por categorías, a estudiantes, a profesores y a los trabajadores de forma general.

2.3.1 Fase de Planificación del Sistema.

La metodología programación extrema tiene 4 fases fundamentales: Planificación, Diseño, Codificación y Prueba; de la cual solo se ha hará referencia a la primera en el presente capítulo.

XP plantea la planificación como un diálogo entre el desarrollador y el cliente los cuales llegarán a un consenso con respecto al alcance del proyecto, la prioridad a la hora de implementar las funcionalidades requeridas, la composición de las versiones y la fecha de entrega de las mismas. La metodología plantea una serie de liberaciones parciales al terminar cada iteración para retroalimentar el proceso de codificación y probar si se están cumpliendo los requisitos del cliente.
Los principales artefactos de esta fase son las historias de usuario, que sirven de guía a todo el proceso de desarrollo. Son utilizadas para especificar los requisitos del software y las escriben los propios clientes según su percepción de las necesidades del sistema. Las historias de usuario proporcionan los detalles sobre la estimación del riesgo y tiempo que llevará la implementación de determinadas funcionalidades. Son una vista a muy alto nivel del costo y esfuerzo del proyecto que se desea desarrollar.

2.3.1.1 Historias de Usuarios.

Para la implementación del sistema se realiza la redacción de las historias de usuarios según las necesidades del cliente.

Las historias de usuarios a implementar son:

1. Gestionar Préstamos.
2. Gestionar Lectores.
3. Reporte de Estadísticas mensual.
4. Gestionar Administradores.

Historia de Usuario	
Fecha:	**Versión:** 1.0
Número: 1	**Dependencia:** HU2
Nombre de la HU: Gestionar Préstamos.	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Descripción: El sistema dará la posibilidad de crear un nuevo préstamo, así como buscar los préstamos existentes y de ellos poder visualizar sus datos; también permitirá eliminar un préstamo en específico.	
Observaciones: El sistema deberá tener acceso a la base de datos.	

Tabla 1: HU 1

Historia de Usuario	
Fecha:	**Versión:** 1.0
Número: 2	**Dependencia:** No
Nombre de la HU: Gestionar Lectores.	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Descripción: El sistema dará la posibilidad de mostrar todos los lectores existentes visualizando así sus datos personales. También permitirá insertar, buscar o eliminar un lector determinado.	
Observaciones: El sistema deberá tener acceso a la base de datos.	

Tabla 2: HU 2

Historia de Usuario	
Fecha:	**Versión:** 1.0
Número: 3	**Dependencia:** HU1
Nombre de la HU: Reporte de Estadísticas.	
Prioridad en Negocio: Media	**Riesgo en Desarrollo:** Medio
Descripción: El sistema mostrará un reporte generado para un mes en específico, el cual visualizará la estadística de los préstamos efectuados por cada una de las facultades. También permitirá eliminar las estadísticas.	
Observaciones: El sistema deberá tener acceso a la base de datos.	

Tabla 3: HU 3

Historia de Usuario	
Fecha:	**Versión:** 1.0
Número: 4	**Dependencia:** No
Nombre de la HU: Gestionar Administradores.	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Medio
Descripción: El sistema mostrará los administradores, dando la posibilidad de eliminar uno existente o insertar uno nuevo, también le permitirá al administrador modificar sus datos, así como cambiar su contraseña.	
Observaciones: El sistema deberá tener acceso a la base de datos.	

Tabla 4: HU 4

2.3.1.2 Planificación de las Historias de Usuarios.

Para planificar acertadamente el desarrollo de la aplicación, es necesario estimar de forma ideal el tiempo que les tomará a los programadores la codificación de cada una de las historias de usuario. Las condiciones ideales se dan cuando se escribe el código sin distracciones y con una dedicación de tiempo completo. La estimación debe dar un máximo de 2 semanas, si es necesario más tiempo debe considerarse una división de la historia de usuario en partes más pequeñas. Como es lógico, debido a las

características del equipo de desarrollo y de la UCI en general, el tiempo de estimación ideal de las historia del sistema, puede extenderse un poco más por cargas extra docentes y un margen de fallo debido a actividades imprevistas.

A partir de la prioridad de las historias se decide cuales de ellas se implementarán en las primeras iteraciones, las funcionalidades críticas del sistema deben ser codificadas en iteraciones tempranas del ciclo.

No	Nombre de la HU	Prioridad	Riesgo	Esfuerzo (Días)	Iteración
1	Gestionar Préstamos.	Alta	Medio	15	2
2	Gestionar Lectores.	Alta	Medio	15	1
3	Reporte de Estadísticas.	Media	Medio	3	3
4	Gestionar Administradores.	Alta	Medio	8	1

Tabla 5: Plan de Entrega

2.3.1.3 Plan de Entregas.

Iteración 1: Se propone codificar las historias de usuario, que proveen las funcionalidades del sistema: #2" Gestionar Lectores" y #4."Gestionar Administradores".

Iteración 2: Se codificará una historia de prioridad alta y riesgo medio, con mucha interacción con el usuario: #1" Gestionar Préstamos".

Iteración 3: Última iteración, se desarrollarán las historias de menos complejidad y que representan un interés especial para el cliente: # 3" Reporte de Estadísticas".

2.3.2 Modelación de las historias de usuario.

Las historias de usuarios son el principal artefacto en la metodología XP pues describen los requerimientos que debe cumplir. Por éste motivo y para ganar claridad en el proceso de desarrollo y comprensión del negocio del problema a resolver es muy importante modelarlas apoyándose en diagramas que ilustren el funcionamiento del sistema.

La herramienta objetivo de la presente investigación será descrita utilizando técnicas de modelado ágil y lenguaje UML para asegurar la total comprensión de las historias de usuario.

2.3.2.1 Iteración 1

En la primera iteración se debe implementar las características principales del sistema, para poder liberar una primera versión o prototipo funcional al cliente. Como la lógica del negocio de la herramienta que se quiere implementar está en las historias #2 y #4, estas fueron las que se seleccionaron para realizarlas en la primera iteración.

La historia #4, permitirá gestionar los administradores del sistema, permitiendo al usuario inicializar dicha aplicación. (Ver: ***¡Error! No se encuentra el origen de la referencia.***; ***¡Error! No se encuentra el origen de la referencia.***)

Tarea de Ingeniería.	
Número de la Tarea: 1	**Número de la HU:** 4
Nombre tarea: Insertar administradores.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 1/04/09	**Fecha Fin:** 3/04/09
Programador Responsable: Yadira Entenza Escobar.	
Descripción: El sistema será capaz de insertar un nuevo administrador, guardando sus datos en la base datos.	

Tabla 6: Tarea 1

Tarea de Ingeniería.	
Número de la Tarea: 2	**Número de la HU:** 4
Nombre tarea: Eliminar administradores.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 6/04/09	**Fecha Fin:** 8/04/09
Programador Responsable: Roimel R. Aguilar Ortiz.	
Descripción: El sistema eliminará de la base datos al administrador escogido.	

Tabla 7: Tarea 2

Tarea de Ingeniería.	
Número de la Tarea: 10	**Número de la HU:** 4
Nombre tarea: Modificar administradores.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 6/05/09	**Fecha Fin:** 8/05/09
Programador Responsable: Roimel R. Aguilar Ortiz.	
Descripción: El sistema modificará en la base datos al administrador escogido y los campos modificados.	

Tabla 8: Tarea 10

La historia #2 se dividió en tres tareas de la ingeniería fundamentales, que proveerán las funcionalidades básicas de insertar, actualizar y eliminar lectores. (Ver: ***¡Error! No se encuentra el origen de la referencia.***; ***¡Error! No se encuentra el origen de la referencia.***)

Tarea de Ingeniería.	
Número de la Tarea: 3	**Número de la HU:** 2
Nombre tarea: Insertar lectores.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 13/04/09	**Fecha Fin:** 15/04/09
Programador Responsable: Yadira Entenza Escobar.	
Descripción: El sistema mostrará un formulario con los campos necesarios para la insertar un lector, luego procederá a almacenar estos datos en la base de datos.	

Tabla 9: Tarea 3

Tarea de Ingeniería.	
Número de la Tarea: 4	**Número de la HU:** 2
Nombre tarea: Mostrar lectores.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 16/04/09	**Fecha Fin:** 18/04/09
Programador Responsable: Roimel R. Aguilar Ortiz.	
Descripción: El sistema dará la posibilidad al usuario de ver todos los datos del lector en cuestión.	

Tabla 10: Tarea 4

Tarea de Ingeniería.	
Número de la Tarea: 5	**Número de la HU:** 2
Nombre tarea: Eliminar lectores.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 20/04/09	**Fecha Fin:** 22/04/09
Programador Responsable: Roimel R. Aguilar Ortiz.	
Descripción: El sistema eliminará de la base de datos al lector escogido por el usuario.	

Tabla 11: Tarea 5

2.3.2.2 Iteración 2

En la segunda iteración se implementará una historia de usuario que interactúa directamente con el usuario por lo que se le realizarán pruebas de aceptación. Dicha historia proveerá las funcionalidades básicas de crear, buscar y terminar un préstamo realizado en la biblioteca.

(Ver: ***¡Error! No se encuentra el origen de la referencia.***; ***¡Error! No se encuentra el origen de la referencia.***; ***¡Error! No se encuentra el origen de la referencia.***)

Tarea de Ingeniería.	
Número de la Tarea: 6	**Número de la HU:** 1
Nombre tarea: Realizar préstamos.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 23/04/09	**Fecha Fin:** 25/04/09
Programador Responsable: Roimel R. Aguilar Ortiz.	
Descripción: El sistema mostrará una pequeña interfaz en la cual se debe introducir el solapín y el ID del libro y luego se procederá a almacenar estos datos en la base de datos.	

Tabla 12: Tarea 6

Tarea de Ingeniería.	
Número de la Tarea: 7	**Número de la HU:** 1
Nombre tarea: Buscar préstamos.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 27/04/09	**Fecha Fin:** 29/04/09
Programador Responsable: Roimel R. Aguilar Ortiz.	
Descripción: El sistema en dependencia de un criterio búsqueda (solapín, morosos, fecha), mostrará un listado de todos los prestamos realizados hasta ese momento.	

Tabla 13: Tarea 7

Tarea de Ingeniería.	
Número de la Tarea: 8	**Número de la HU:** 1
Nombre tarea: Terminar préstamos.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 30/04/09	**Fecha Fin:** 2/05/09
Programador Responsable: Yadira Entenza Escobar.	
Descripción: El sistema eliminará de la base de datos el préstamo elegido por el usuario.	

Tabla 14: Tarea 8

2.3.2.3 Iteración 3

En la tercera iteración se implementará una historia de usuario que no interactúa con el usuario por lo que no se le realizarán pruebas de aceptación. Dicha historia proveerá las funcionalidades que permitirán mostrar la información concerniente a los préstamos realizados en un en mes específico en la biblioteca, esta información será mostrada en pantalla de una forma sencilla de asimilar para el usuario de la aplicación. (Ver: ***¡Error! No se encuentra el origen de la referencia.*** *)*

Tarea de Ingeniería.	
Número de la Tarea: 9	**Número de la HU:** 3
Nombre tarea: Estadísticas mensuales.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 4/05/09	**Fecha Fin:** 6/05/09
Programador Responsable: Roimel R. Aguilar Ortiz.	
Descripción: El sistema contará en la base de datos todos los préstamos efectuados, que este puede ser para un mes en específico o de forma general, por cada una de las facultades.	

Tabla 15: Tarea 9

Tarea de Ingeniería.	
Número de la Tarea: 11	**Número de la HU:** 3
Nombre tarea: Buscar estadísticas.	
Tipo de Tarea: Desarrollo Desarrollo/Corrección/Mejora/Otra(especificar)	
Fecha Inicio: 9/05/09	**Fecha Fin:** 10/05/09
Programador Responsable: Roimel R. Aguilar Ortiz.	
Descripción: El sistema mostrará en una interfaz, las estadísticas; en dependencia del criterio de búsqueda (mes, año) que se desee.	

Tabla 16: Tarea 11

Conclusiones

Durante el transcurso del presente capítulo se realizó una descripción de la solución general propuesta, se definieron y redactaron las historias de usuario así como las tareas de la ingeniería correspondientes a cada una de ellas, se estima un tiempo ideal de 30 días necesario para implementar la herramienta. Se definieron las pruebas de aceptación requeridas para garantizar que la implementación de cada una de las historias sea realizada correctamente y que el sistema se comporta tal y como se espera que lo haga.
Además se generaron los diagramas UML considerados indispensables para asegurar la completa comprensión y claridad del sistema antes de comenzar a codificar las historias obtenidas.

Capítulo 3: Diseño, Codificación y Pruebas.

3.1 Introducción

En el presente capítulo se describe el proceso de diseño de la herramienta de gestión de préstamos de libros en la BiUCI a través de varios artefactos propuestos por la metodología XP. Como exige dicha metodología, la herramienta ha sido implementada de forma iterativa, obteniendo al culminar de cada iteración un producto funcional que debe ser probado y mostrado al cliente para incrementar la visión de los desarrolladores con la opinión de este. Además se detallan las tres iteraciones llevadas a cabo durante la etapa de construcción del sistema, exponiéndose fundamentalmente las Tarjetas CRC y el modelado en lenguaje UML de la arquitectura del sistema, las tareas generadas por cada historia de usuario, así como las pruebas de aceptación efectuadas sobre el sistema.

3.2 Diseño

La metodología XP sugiere que hay que conseguir diseños simples y sencillos. Por lo que se procura hacerlo todo lo menos complicado posible para conseguir un diseño fácilmente entendible e implementable que a la larga costará menos tiempo y esfuerzo al desarrollar.

3.3 Diseño de la Arquitectura del Sistema

Durante el ciclo de vida de desarrollo de un software utilizando la metodología XP la fase de diseño se realiza justo después que se termina la fase de planificación. Durante esta fase se comienza a definir el diseño arquitectónico del software partiendo de las historias de usuario de la primera iteración y de la metáfora, un elemento que según Kent Beck el creador de la metodología, guía al equipo de proyecto durante todo el desarrollo del programa y que además sustituye en gran parte de lo que sería la arquitectura del sistema en modelos más pesados o tradicionales.

La arquitectura de la herramienta o de cualquier proyecto guiado por XP debe ser sencilla y responder al principio de *"cuál es la solución más simple capaz de funcionar"*. Debe ser una estructura que organiza la lógica del sistema de manera que al realizar cambios en partes no provoca cambios en otros componentes, a menos que sea absolutamente necesario. Debe enfocarse en cómo resolver los problemas identificados en el momento en que se diseña, sin tener en cuenta posibles variaciones o

requerimientos futuros, los cuales serán enfrentados y agregados al diseño en el momento en que sean detectados.

3.3.1 Metáfora del sistema propuesto

La metáfora creada por Kent Beck es una historia simple de cómo funciona todo el sistema compartida por todo el equipo de desarrollo. Su misión es describir el software de manera que los desarrolladores y clientes que trabajan en conjunto puedan comunicarse fácilmente a la hora de referirse al sistema, o una parte de éste. Cuando el sistema es pequeño la metáfora es extremadamente sencilla.
Teniendo en cuenta los parámetros anteriormente expuestos, la metáfora del sistema sería: Herramienta para gestionar los préstamos externos de libros de literatura en la biblioteca de la UCI.

3.3.2 Arquitectura

Para definir correctamente la arquitectura de un software es necesario que se identifiquen los elementos arquitectónicamente significativos (clases, módulos, objeto, interfaces) y las relaciones que existen entre ellos. En el siguiente ciclo de vida del proyecto se generará solamente los diagramas y artefactos estrictamente necesarios para garantizar una total comprensión entre el equipo de desarrollo y el cliente.

Para garantizar que los cambios o nuevas funcionalidades añadidas puedan ser asimilados por la arquitectura del sistema, esta estará formada por capas (presentación, lógica del negocio y acceso a datos) organizadas por responsabilidades.
Se propone para describir la arquitectura de la herramienta, desarrollar los diagramas de clases y diagramas de componentes, los cuales proveerán tanto una vista conceptual como física del funcionamiento del sistema.

3.4 Tarjetas CRC

Las tarjetas CRC son técnicas de modelado creadas para ayudar a los desarrolladores de software a crear diseños de clases orientados a responsabilidades. Dichas tarjetas constan de tres secciones, nombre de la clase, responsabilidades y colaboradores. La sección responsabilidades se lista cada una de las funciones o tareas que debe ser capaz cumplir un objeto de dicha clase mientras que la sección colaboradores contiene otras clases del diseño que pueden colaborar para proveer datos o funcionalidades.

La metodología XP estipula. En toda la bibliografía consultada. El uso de las tarjetas CRC como un artefacto obligatorio durante el desarrollo de un proyecto. Debido a los beneficios que aporta en cuanto a la comunicación y transparencia del desarrollo. La mayoría de los autores recomiendan que se encuentren en un lugar público. A la vista de todo el equipo. Donde todos los programadores puedan contribuir a mejorar y refinar el diseño resultante; así como verificar que diferentes clases no dupliquen responsabilidades o que una interfiera con las funcionalidades de la otra; problemas muy frecuentes a la hora de desarrollar una arquitectura bajo paradigmas de orientación a objetos.

Historia Usuario: Gestionar Lectores.	
Funcionalidades	**Colaboraciones(HU)**
Buscar Lectores. Mostrar Lectores. Eliminar Lectores.	Gestionar Préstamos. Reporte Estadísticas.

Tabla 17: Tarjeta CRC HU Gestionar Lectores

Historia Usuario: Gestionar Préstamos.	
Funcionalidades	**Colaboraciones (HU)**
Crear Préstamos. Buscar Préstamos. Mostrar Préstamos. Eliminar Préstamos.	Gestionar Lectores. Reporte de Estadísticas.

Tabla 18: Tarjeta CRC HU Gestionar Préstamos

Historia Usuario: Reporte Estadísticas.	
Funcionalidades	**Colaboraciones(HU)**
Buscar Estadísticas. Mostrar Estadísticas.	Gestionar Préstamos.

Tabla 19: Tarjeta CRC HU Reporte Estadísticas

3.5 Diagramas de clases del diseño

Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Estos son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará.

En el caso propuesto se ha creado un diseño compuesto por capas utilizando un Patrón Modelo Vista Controlador (MVC), para desacoplar la interfaz gráfica de los modelos que implementan la lógica del negocio y el acceso a datos.

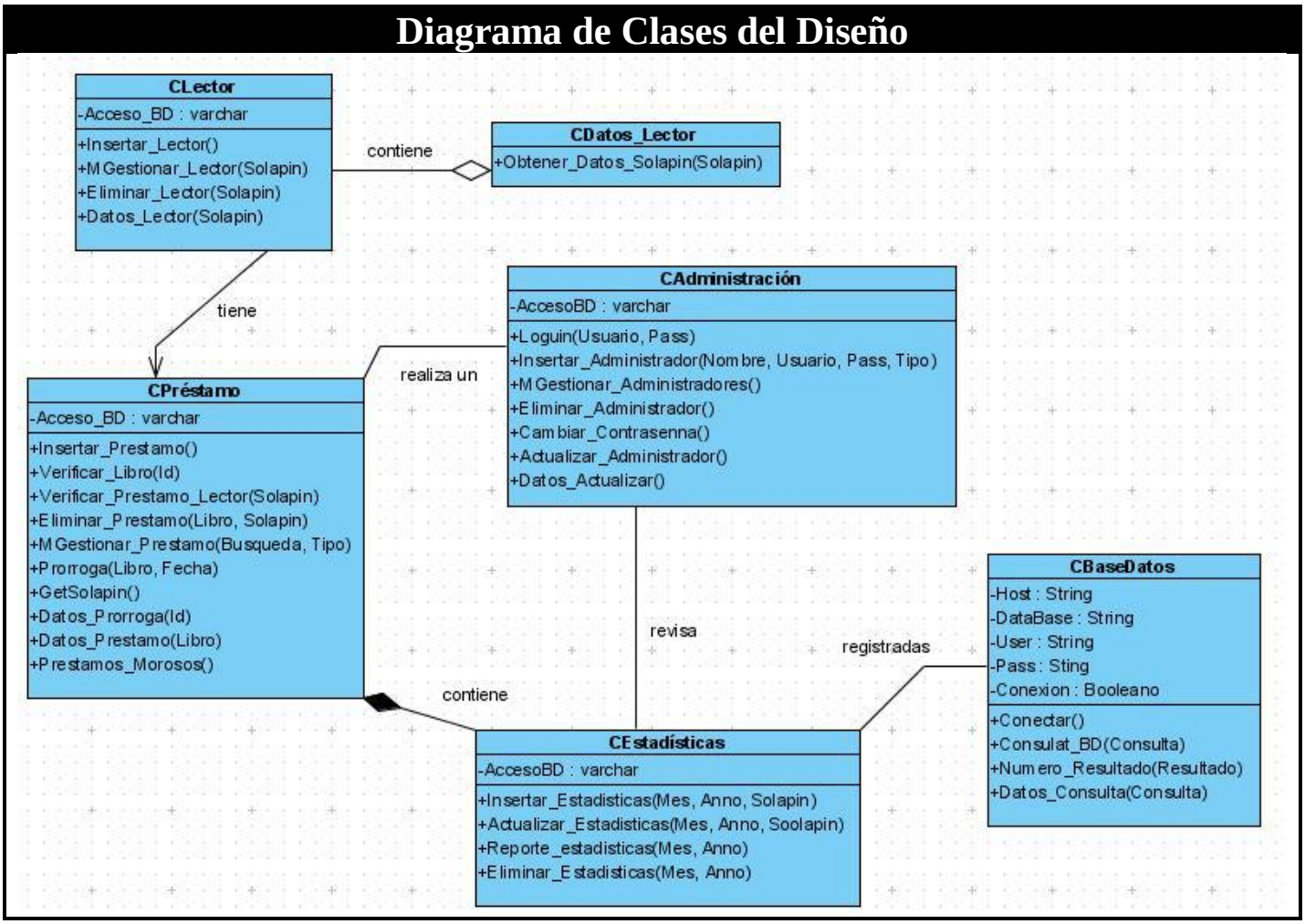

Figura 1: Diagrama de Clases del Diseño

3.6 Interfaz de Usuario

- **Iniciar Sesión**: Interfaz mediante la cual el administrador se autentica en el sistema.

Figura 2: Interfaz "Iniciar Sesión"

- **Estadísticas:** Interfaz mediante la cual los administradores del sistema, podrán ver los préstamos realizados en la biblioteca, especificados por facultad y por quien son realizados: si son estudiantes, profesores o trabajadores. Además permitirá buscar los préstamos por un criterio de búsqueda, que en este caso será por una fecha determinada.

Figura 3: Interfaz "Estadísticas"

- **Crear préstamos:** Esta interfaz permitirá agregar al sistema un nuevo préstamo, teniendo en cuenta de que el solapín del lector no puede estar en la base datos y que el ID del libro no este activo.

Figura 4: Interfaz "Crear Préstamos"

- **Buscar préstamo:** Esta interfaz permitirá mediante un criterio de búsqueda acceder a otra página mostrando todos los préstamos realizados hasta ese momento según el criterio escogido.

Figura 5: Interfaz "Buscar Préstamos"

- **Gestionar préstamos:** Mostrará un listado con todos los préstamos realizados. Además permitirá mostrar los datos del préstamo y eliminar un préstamo de la base datos.

Figura 6: Interfaz "Gestionar Préstamos"

- **Buscar lectores:** Esta interfaz permitirá. Mediante un criterio de búsqueda. Acceder a otra página mostrando todos los lectores que han realizado préstamos hasta ese momento según el criterio escogido.

Figura 7: Interfaz "Buscar Lectores"

- **Gestionar lectores:** Mostrará un listado con todos los lectores registrados en la base datos. Además permitirá mostrar los datos del lector y eliminar un lector de la base datos.

Figura 8: Interfaz "Gestionar Lectores"

- **Insertar administrador:** Esta interfaz permitirá registrar un nuevo administrador a la base datos.

Figura 9: Interfaz "Insertar Administrador"

- **Buscar administrador:** Esta interfaz permitirá mediante un criterio de búsqueda acceder a otra página mostrando todos los administradores registrados en la base datos. Según el criterio escogido.

Figura 10: Interfaz "Buscar administrador"

- **Gestionar administradores:** Mostrará un listado con todos los administradores registrados en la base datos. Además permitirá actualizar los datos de un administrador y eliminar un administrador de la base datos.

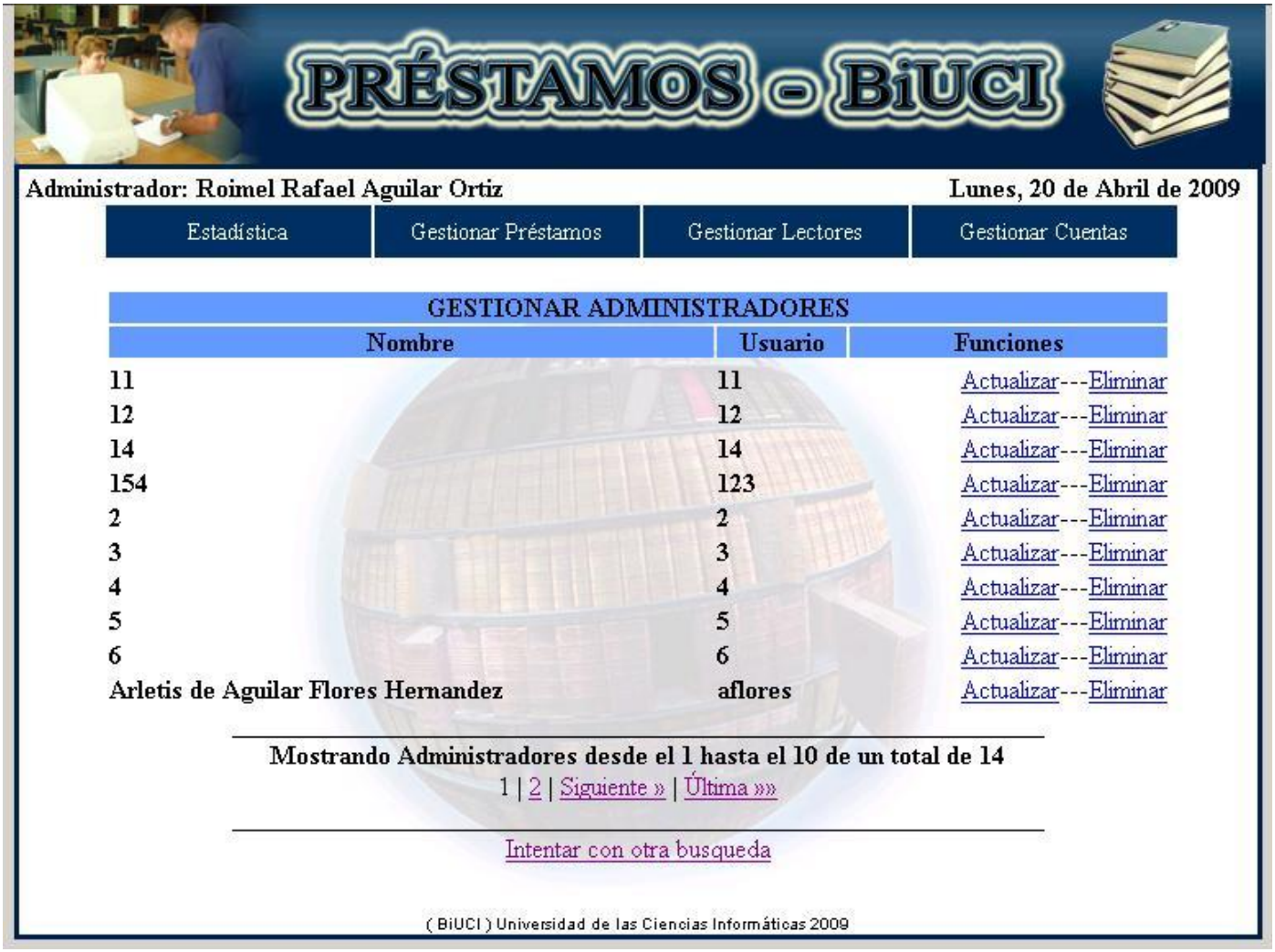

Figure 11: Interfaz "Gestionar Administradores"

3.7 Diseño de la Base de Datos

Una de las tareas más importantes a la hora de construir una aplicación Web es la base de datos, ya que uno de sus objetivos fundamentales es brindar la persistencia al modelo que se describe en el capítulo.
El modelo de datos del problema en cuestión posee un nivel de complejidad bajo, producto a que la aplicación es pequeña.

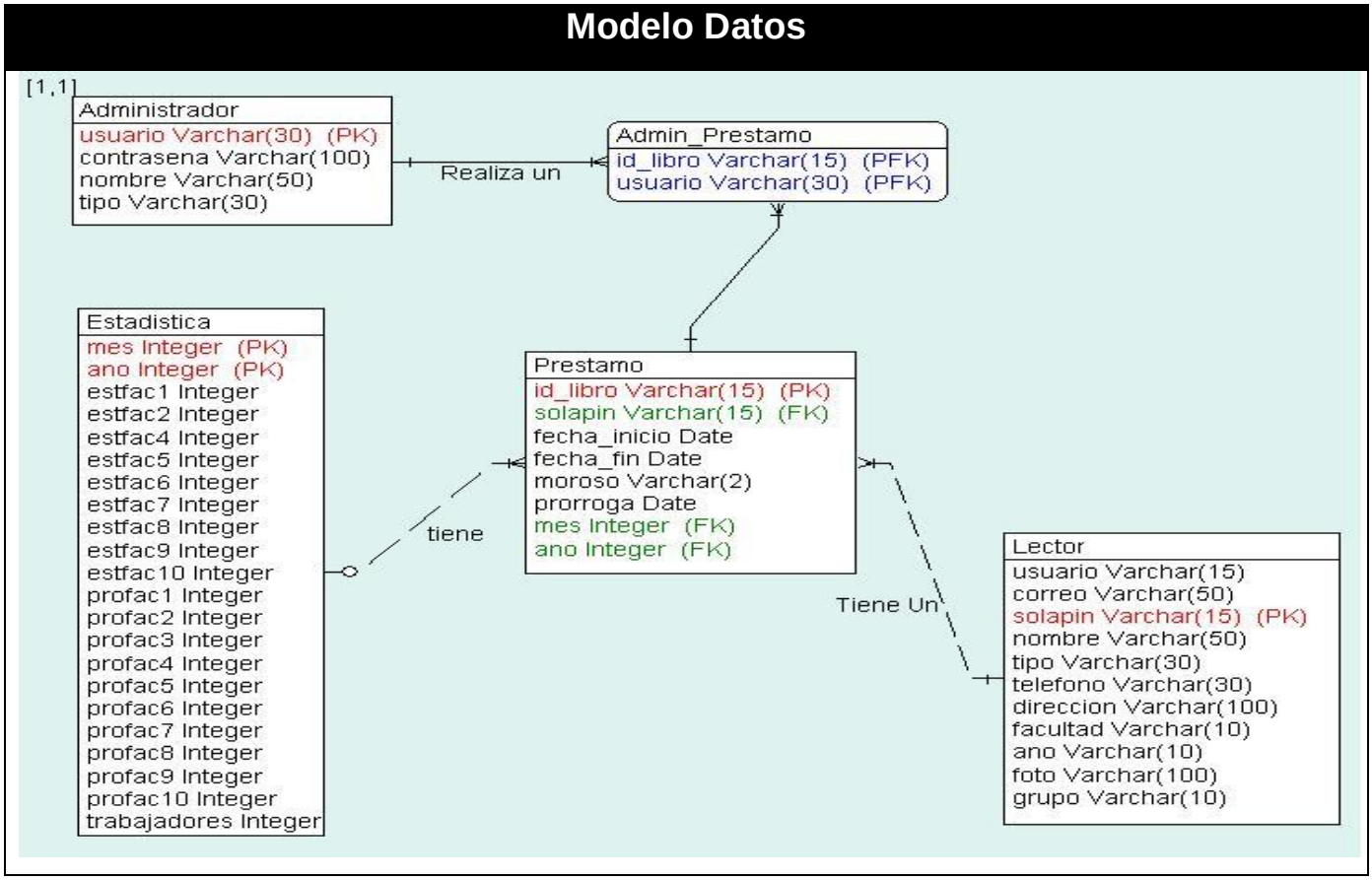

Tabla 20: Modelo Datos

3.8 Codificación

Durante el transcurso de las iteraciones se realiza la implementación de las historias de usuario que fueron seleccionadas en cada una de ellas. Al principio de estas se lleva a cabo una revisión al plan de iteraciones y se modifica en caso de ser necesario. Como parte del plan se descomponen estas historias en tareas de ingeniería.

Teniendo en cuenta la planificación realizada en el capítulo anterior, se llevaron a cabo tres iteraciones de desarrollo sobre el sistema, obteniéndose un producto con funcionalidad en cada una de ellas. A continuación se detallan cada una de las iteraciones.

3.8.1 Iteración 1

En esta iteración se implementaron las historias de usuario con mayor prioridad para el usuario, con el objetivo de obtener una versión del producto con algunas de las funcionalidades más importantes para ser mostrado al cliente.

Historias de Usuario	Tiempo de Implementación(Días)	
	Estimación	Tiempo Real
Gestionar lectores	15	5
Gestionar administradores	8	4

Tabla 21: Estimación de la iteración 1

3.8.2 Iteración 2

En esta iteración se implementó una historia de usuario con una prioridad alta para el usuario.

Historia de Usuario	Tiempo de Implementación(Días)	
	Estimación	Tiempo Real
Gestionar préstamos	15	4

Tabla 22: Estimación de la iteración 2

3.8.3 Iteración 3

En esta iteración se implementó la historia de usuario de prioridad media para el cliente. Esta tiene la finalidad de proporcionarle un ambiente afable y cómodo para el usuario. Al finalizar se cuenta con un producto listo para poner en funcionamiento.

Historia de Usuario	Tiempo de Implementación(Días)	
	Estimación	Tiempo Real
Reporte estadísticas	3	2

Tabla 23: Estimación de la iteración 3

3.8.4 Diagrama de despliegue

Los diagramas de despliegue muestran nodos, conexiones, componentes y objetos. Los nodos representan objetos físicos con recursos computacionales como procesadores y periféricos; pueden mostrarse como una clase (e.g. una familia de procesadores) o una instancia, por lo que su nombre sigue la misma sintaxis establecida para clases y objetos. Las conexiones son asociaciones de comunicación entre los nodos, y se etiquetan con un estereotipo que identifica el protocolo de comunicación o la red utilizada. Los componentes son archivos de código ejecutable, que residen y se ejecutan dentro de un nodo; se pueden representar relaciones de dependencia entre los componentes que, de manera similar a las dependencias entre paquetes, corresponden al uso de servicios.

En este caso la aplicación se encuentra hospedada en un servidor Web y esta se comunica con un sistema de gestión de base de datos (MySQL).

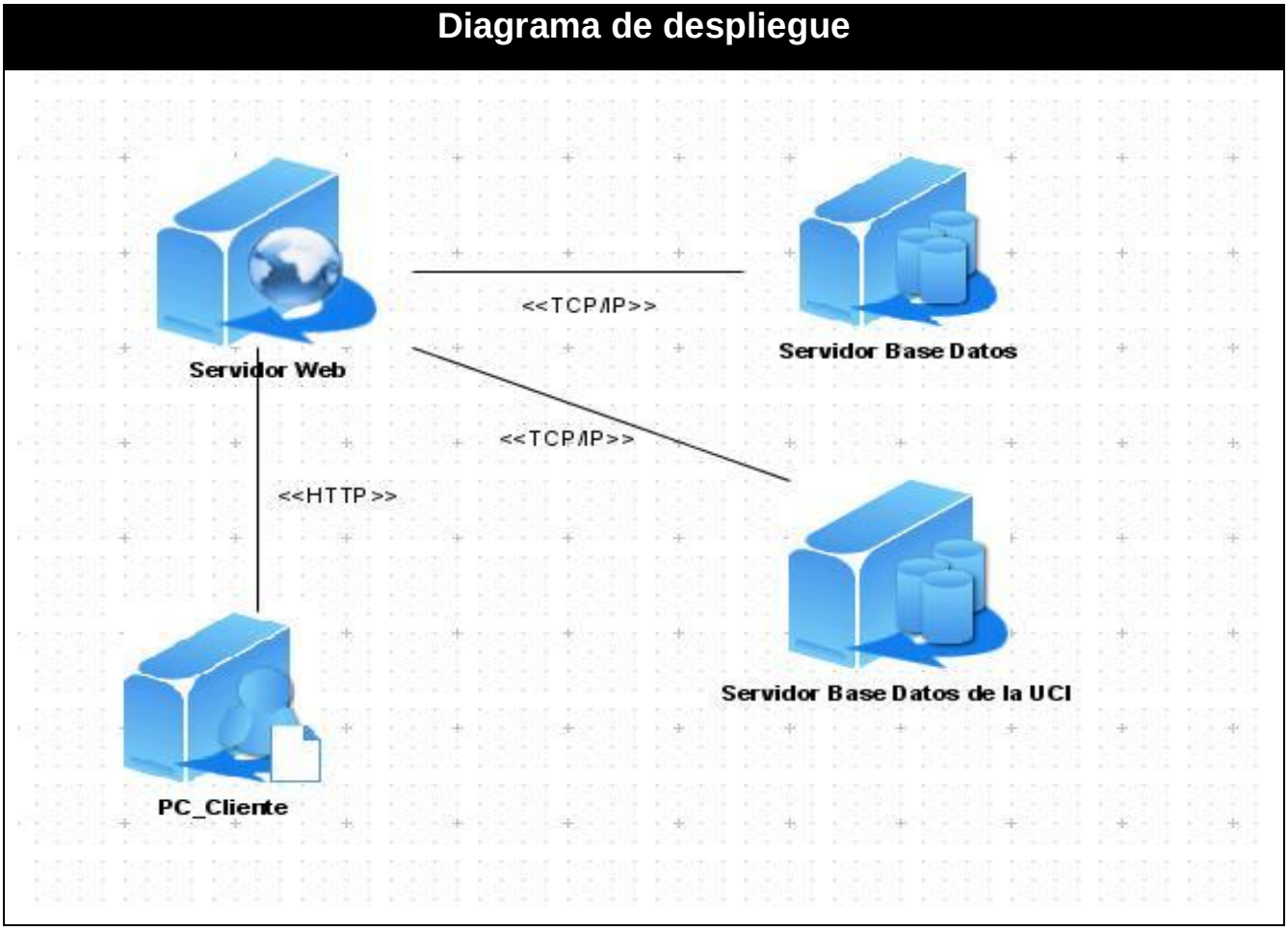

Figura 12: Diagrama de Despliegue

3.9 Pruebas

En XP uno de los pilares más importantes es el proceso de pruebas, en el cual los desarrolladores prueban tanto como sea posible. Mediante esta filosofía se reduce el número de errores no detectados, así como el tiempo de introducción de éste en el sistema y su detección. Todo esto permite una mejor calidad en los productos desarrollados y la seguridad de los programadores a la hora de introducir algún cambio o modificación en el sistema.

La metodología XP divide las pruebas en dos grupos: las pruebas unitarias, que son realizadas por los programadores, encargadas de verificar el código; y las pruebas de aceptación, destinadas a verificar si al final de cada iteración se obtuvo la funcionalidad que se requiere, además de comprobar que dicha funcionalidad sea lo que el cliente quiere.

3.9.1 Pruebas de Aceptación

3.9.1.1 Iteración 1

Las pruebas de aceptación diseñadas por el equipo de trabajo para verificar las funcionalidades de la segunda historia de usuario están enfocadas mayormente en comprobar la capacidad de la herramienta de insertar y eliminar un lector. Con respecto a la historia #4 se comprobará la correcta inserción, modificación y eliminación de los administradores del sistema. Ambas tienen una alta interacción con el usuario de la herramienta por lo que es necesario chequear la manera en que la herramienta cumpla con las exigencias del cliente.

Casos de pruebas: Historia de Usuario #2.

Pruebas de Aceptación.	
Número del Caso de Prueba: 1	**Número de la HU:** 2
Nombre del Caso de Prueba: Insertar lectores.	
Entradas: Datos del lector.	
Resultado Esperado: Que se registre el lector en la base de datos.	
Observaciones: El lector no debe estar registrado en la base de datos.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 24: Caso de Prueba "Insertar lectores"

Pruebas de Aceptación.	
Número del Caso de Prueba:2	**Número de la HU:** 2
Nombre del Caso de Prueba: Mostrar lectores.	
Entradas: Solapín.	
Resultado Esperado: Que el sistema muestre un listado con los lectores registrados en la base de datos, con el criterio de búsqueda escogido.	
Observaciones: El lector debe estar registrado en la base datos.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 25: Caso de Prueba "Mostrar lectores"

Pruebas de Aceptación.	
Número del Caso de Prueba:3	**Número de la HU:** 2
Nombre del Caso de Prueba: Eliminar lectores.	
Entradas: Solapín.	
Resultado Esperado: El sistema deberá eliminar de la base de datos al lector escogido	
Observaciones: El lector debe estar registrado en la base datos.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 26: Caso de Prueba "Eliminar lectores"

Casos de Pruebas: Historia de Usuario # 4.

Pruebas de Aceptación.	
Número del Caso de Prueba:4	**Número de la HU:** 4
Nombre del Caso de Prueba: Insertar Administrador.	
Entradas: Datos del administrador.	
Resultado Esperado: El sistema deberá almacenar en la base de datos los datos introducidos y crear la cuenta de administración correspondiente. Dándole a este los privilegios de administración.	
Observaciones: No debe existir otro administrador con el mismo usuario.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 27: Caso de Prueba "Insertar administradores"

Pruebas de Aceptación.	
Número del Caso de Prueba:5	**Número de la HU:** 4
Nombre del Caso de Prueba: Modificar Administrador.	
Entradas: Datos del administrador.	
Resultado Esperado: El sistema deberá almacenar en la base de datos los nuevos datos introducidos.	
Observaciones: El administrador deberá existir en la base datos.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 28: Caso de Prueba "Modificar administrador"

Pruebas de Aceptación.	
Número del Caso de Prueba:6	**Número de la HU:** 4
Nombre del Caso de Prueba: Eliminar Administrador.	
Entradas: Usuario.	
Resultado Esperado: El sistema deberá eliminar de la base de datos al administrador seleccionado.	
Observaciones: El administrador deberá existir en la base datos.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 29: Caso de Prueba "Eliminar administrador"

3.9.1.2 Iteración 2

Las pruebas de aceptación diseñadas por el cliente del sistema para verificar esta funcionalidad están enfocadas mayormente en comprobar la capacidad del sistema de crear, buscar y terminar un préstamo.

Casos de Pruebas: Historia de Usuario #1.

Pruebas de Aceptación.	
Número del Caso de Prueba:7	**Número de la HU:** 1
Nombre del Caso de Prueba: Buscar préstamos.	
Entradas: Solapín, fecha, morosos o por todos los criterios de búsqueda.	
Resultado Esperado: Mostrar los préstamos activos.	
Observaciones:	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 30: Caso de Prueba "Buscar préstamos"

Pruebas de Aceptación.	
Número del Caso de Prueba: 8	**Número de la HU:** 1
Nombre del Caso de Prueba: Realizar préstamos.	
Entradas: Datos del préstamo.	
Resultado Esperado: Que se registre el préstamo en la base de datos.	
Observaciones: El ID del libro escogido para el préstamo tiene que estar disponible y el lector debe estar registrado en la base de datos anteriormente.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 31: Caso de Prueba "Realizar préstamos"

Pruebas de Aceptación.	
Número del Caso de Prueba: 9	**Número de la HU:** 1
Nombre del Caso de Prueba: Terminar préstamo.	
Entradas: Solapín.	
Resultado Esperado: Que se elimine el préstamo en la base de datos.	
Observaciones:	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 32: Caso de Prueba "Terminar préstamos"

3.9.1.3 Iteración 3

Las pruebas de aceptación diseñadas para esta iteración están basadas en comprobar la capacidad del sistema, al emitir un reporte de estadísticas, teniendo en cuenta un mes en específico.

Casos de Pruebas: Historias de Usuario #3.

Pruebas de Aceptación.	
Número del Caso de Prueba: 10	**Número de la HU:** 3
Nombre del Caso de Prueba: Reporte Estadísticas.	
Entradas: Mes del que se desea ver el reporte.	
Resultado Esperado: El sistema será capaz de mostrar un listado con la cantidad de préstamos realizados hasta el momento por cada una de las facultades, siendo así para estudiantes, profesores y trabajadores.	
Observaciones:	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 33: Caso de Prueba "Reporte estadísticas"

Conclusiones

En el presente capítulo se abordó la fase de diseño de la herramienta, se obtuvo la metáfora que va a describir el sistema y se crearon las Tarjetas CRC de todas las clases que intervendrán en la solución con lo que se hizo una exploración de las relaciones y colaboraciones entre ellas, lo cual contribuyó a lograr un buen diseño. Se obtuvo una arquitectura capaz de asimilar variación de requisitos apoyada en la programación modular y un diseño en 3 capas lógicas descrito detalladamente mediante un Diagrama de Clases y un Diagrama de Componentes. Se creó además un prototipo de interfaz gráfica que reúne las mejores características de las herramientas similares estudiadas.

Conclusiones

La gestión de la información hoy en día se ha convertido en tarea de todos. El rápido avance de las tecnologías en el campo de la informática y su gran aceptación a nivel mundial hace que cada día se le de más importancia a los hábitos de lectura de forma digital, así como al uso de las tecnologías de la informática en general, debido a la gran cantidad de usuarios que le han dado su preferencia a este sector.

Durante el desarrollo de esta investigación se realizó un estudio detallado que permitió tener un conocimiento de la situación actual y las tendencias de los sistemas de gestión de préstamos externos de libros. Se demostró la necesidad de desarrollar un sistema que fuese capaz de automatizar los procesos que se llevan a cabo en la realización de préstamos de libros en las bibliotecas. Con este trabajo se presenta una aplicación Web que permite crear préstamos; así como buscar, eliminar y mostrar estos. Además genera un reporte de estadísticas con los préstamos efectuados en la biblioteca.

El desarrollo del trabajo utilizando la metodología XP permitió documentar el mismo desde el comienzo, lo que facilitará su estudio, permitiendo de esta forma, una comprensión más rápida y fácil de la concepción general del sistema. La versión del sistema obtenida, constituye una base para la integración con otro módulo de servicios prestados en la biblioteca de la Universidad de las Ciencias Informáticas, así como para futuras versiones del sistema.

Recomendaciones

Como resultado del proceso de investigación y realización de la aplicación han surgido ideas que serían recomendables tener en cuenta para un futuro perfeccionamiento del sistema, a continuación se listan las mismas:

- Agregar un módulo, para la gestión de un catálogo más detallado, que el existente.
- Incluir un módulo; donde los estudiantes, profesores y trabajadores de la Universidad de las Ciencias Informáticas puedan solicitar los préstamos de forma on-line.

Referencias Bibliográficas

Metodología XP. *oness.sourceforge.net.* [En línea] http://oness.sourceforge.net/proyecto/html/ch05.html.

Abraham. 2007. javahispano. *javahispano.com.* [En línea] 10 de 09 de 2007. http://www.javahispano.org/contenidos/es/mysql_vs_postgresql_cuando_emplear_cada_una_de_ellas_11.

Anonimo. *sistemas.fsl.fundacite-merida.gob.ve.* [En línea] http://sistemas.fsl.fundacite-merida.gob.ve/docman/view.php/84/420/compracion_de_framework_mvc.pdf.

Anónimo. El Proceso Unificado de Desarrollo de Software (RUP). *El Proceso Unificado de Desarrollo de Software (RUP).* [En línea] http://yaqui.mxl.uabc.mx/~molguin/as/RUP.htm.

—. 2007. Maestros del Web. *Maestros del Web.* [En línea] 5 de 06 de 2007. http://www.maestrosdelweb.com/editorial/herramientas-adecuadas-para-el-diseno-y-desarrollo-de-un-sitio-web/.

—. www.slideshare.net. *www.slideshare.net.* [En línea] http://www.slideshare.net/alexmerono/sistemas-gestores-de-bases-de-datos.

—. www.ulpgc.es. *www.ulpgc.es.* [En línea] http://www.ulpgc.es/otros/tutoriales/JavaScript/cap1.htm. cakephp. *www.cakephp.org.* [En línea] http://www.unixmexico.org/modules.php?name=News&file=article&sid=1754.

Castillo, Oswaldo, Figueroa, Daniel y Sevilla, Hector. Programación Extrema. *Programación Extrema.* [En línea] http://programacionextrema.tripod.com/index.htm.

2006. Ciberaula. *Ciberaula.* [En línea] 2006. http://linux.ciberaula.com/articulo/linux_apache_intro/.

2001. desarrolloweb. *desarrolloweb.* [En línea] 9 de 05 de 2001. http://www.desarrolloweb.com/articulos/393.php.

2001. desarrolloweb. *desarrolloweb.* [En línea] 9 de 05 de 2001. http://www.desarrolloweb.com/articulos/392.php.

2002. desarrolloweb. *desarrolloweb.* [En línea] 8 de 07 de 2002. http://www.desarrolloweb.com/articulos/831.php.

Eguíluz Pérez, Javier. 2008. Libros Web. *Libros Web.* [En línea] 26 de 10 de 2008. http://www.librosweb.es/ajax/index.html.

—. 2008. Libros Web. *Libros Web.* [En línea] 30 de 12 de 2008. http://www.librosweb.es/javascript/capitulo1.html.

—. 2008. Libros Web. *Libros Web.* [En línea] 29 de 10 de 2008. http://www.librosweb.es/css/index.html.

Escobar, Yanvary. 1997. monografias.com. *monografias.com.* [En línea] 1997. http://www.monografias.com/trabajos39/desarrollo-del-software/desarrollo-del-software2.shtml.

Flores, Antonio. www.formaselect.com. *www.formaselect.com.* [En línea] http://www.formaselect.com/curso/experto-en-sql-server-2000/Introduccion-a-SQL-Server%202000.pdf.

Gutiérrez, Javier J. www.lsi.us.es. *www.lsi.us.es.* [En línea] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

HERRAMIENTAS DE DESARROLLO DE INGENIERIA DE SW PARA LINUX. **Giraldo, Luis y Zapata, Yuliana. 2005.** 2005, pág. 11.
microsoft. *www.microsoft.com.* [En línea] http://www.microsoft.com/spain/medianaempresa/products/sql/evaluate.mspx.

2007. Mononeurona. *Mononeurona.org.* [En línea] 29 de 04 de 2007. http://www.mononeurona.org/pages/display/150.

ProgramacionWeb. 2003_2009. www.programacionweb.net. *www.programacionweb.net.* [En línea] 2003_2009. http://www.programacionweb.net/articulos/articulo/?num=184.

Quiñones, Ernesto. www.apesol.org. *www.apesol.org.* [En línea] http://www.postgresql.org.pe/articles/introduccion_a_postgresql.pdf.

Rivera, Volkan. 2007. Tecnología y negocios. [En línea] 2 de 07 de 2007. http://www.volkanrivera.com/esp/?p=24.

Santamaría, Fernando. Fesabid 98. *Fesabid 98.* [En línea] http://fesabid98.florida-uni.es/Comunicaciones/f_santamaria/f_santamaria.htm.

Schweitzer, Albert. 2000. Netware Sistemas. *Netware Sistemas.* [En línea] 2000. http://www.netwareinformatica.com.ar/index.asp.

sidar. *www.sidar.org.* [En línea] http://www.sidar.org/recur/desdi/mcss/manual/intro.php.

Superuser. 2003. soporte.tiendalinux.com. *soporte.tiendalinux.com.* [En línea] 31 de 05 de 2003. http://soporte.tiendalinux.com/portal/Portfolio/postgresql_ventajas_html.

2009. todoexpertos. *www.todoexpertos.com.* [En línea] 2009. http://www.todoexpertos.com/categorias/tecnologia-e-internet/bases-de- datos/oracle/

Zaninotto, François y Potencier, Fabien. 2009. Libros Web. *Libros Web.* [En línea] 27 de 01 de 2009. http://www.librosweb.es/symfony_1_1/.

Bibliografía

Zaninotto, François y Potencier, Fabien. Libros Web. [Online] http://www.librosweb.es/symfony_1_1/

Abraham. Javahispano.com. [Online]
http://www.javahispano.org/contenidos/es/mysql_vs_postgresql_cuando_emplear_cada_una_de_ellas_11

2008. AJAX un nuevo acercamiento a aplicaciones web. [En línea] 6 de 05 de 2008.
http://www.uberbin.net/archivos/internet/ajax-un-nuevo-acercamiento-a-aplicaciones-web.php.

Andrei, Zeev, Rasmus, Jim y otros. *Manual de PHP.*

Bakken, Sæther, Schmid, Stig. y Egon. Manual de Php. PHP Documentation Group. [En línea]
http://www.php.net/docs.php.

Beck, K. y Fowler, M. 2000. Planeando en Programación Extrema. [Online] 2000.
Construcción de una página para la Web.
El Proceso Unificado de Desarrollo de Software (RUP). [Online]
http://yaqui.mxl.uabc.mx/~molguin/as/RUP.htm.

Instalar Apache, Internet Information Server (IIS) y MySQL en Windows. [Online]
http://www.cristalab.com/tutoriales/128/instalar-apache-internet-information-server-iis--y-mysql-en-windows.
Introducción al JavaScript . [Online] http://www.ulpgc.es/otros/tutoriales/JavaScript/cap1.htm.

Pérez, Javier Eguíluz. *Intrdoucción a JavaScript.*
—. *Introducción a Ajax.*
—. *Introducción a CSS.*

XP. A gentle introduction. [Online] http://www.extremeprogramming.org.

Glosario de Términos

ASP: Del inglés Active Server Pages es una tecnología del lado servidor de Microsoft para páginas web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Server (IIS). La tecnología ASP está estrechamente relacionada con el modelo tecnológico de su fabricante. Intenta ser solución para un modelo de programación rápida.

AJAX: Acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications).

BiUCI: Biblioteca de la Universidad de las Ciencias Informáticas.

CCS: Las hojas de estilo en cascada (en inglés Cascading Style Sheets) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML).

Framework: Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software, para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

HTML: Siglas de HyperText Markup Language (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

JSP: Es un acrónimo de Java Server Pages, que en castellano vendría a decir algo como Páginas de Servidor Java. Es, pues, una tecnología orientada a crear páginas web con programación en Java.

Plugin: “Parche” para un programa que le añade características nuevas.

Release: Nueva versión de una aplicación informática.

Anexos

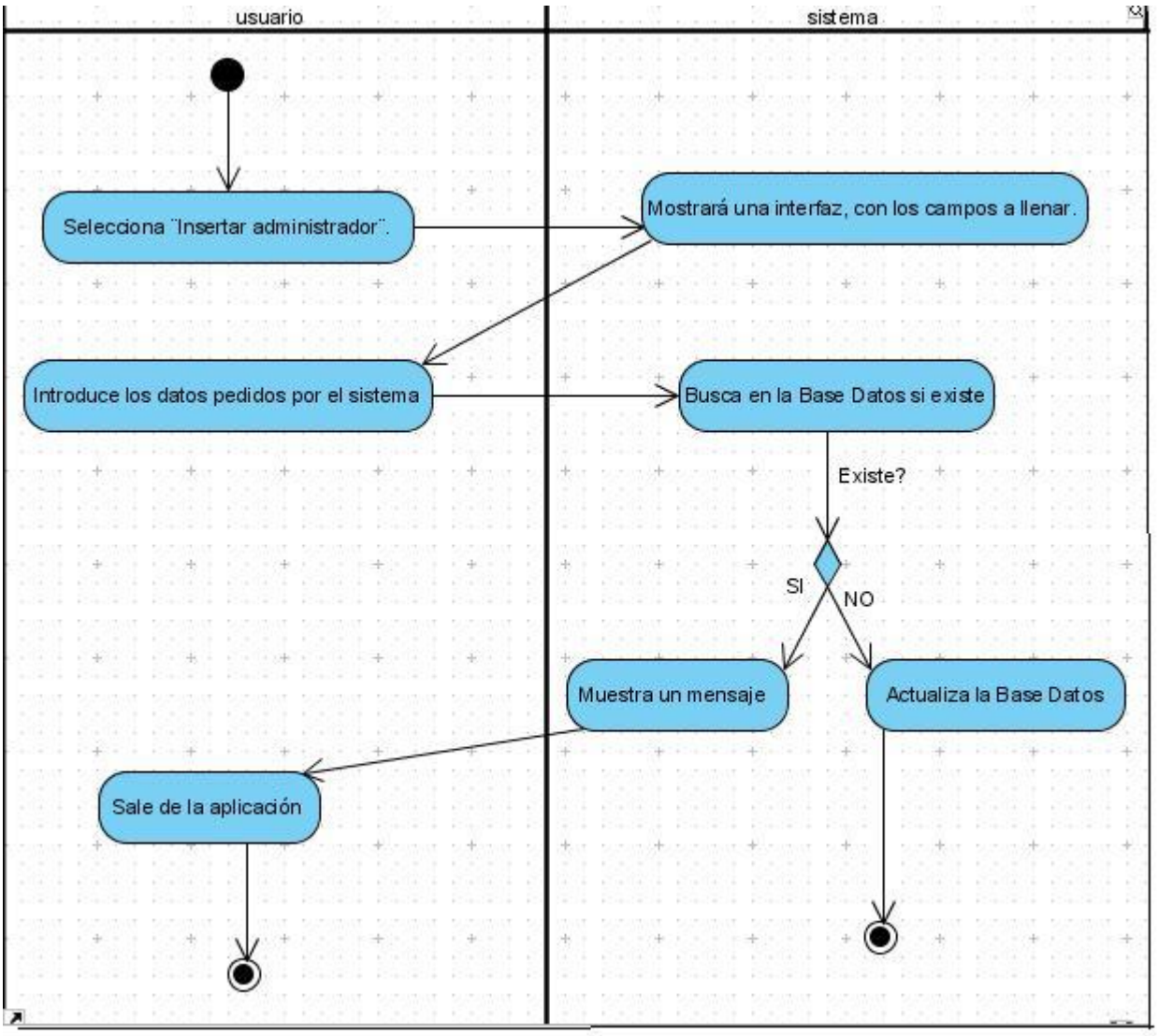

Figura 13: DS "Insertar administrador"

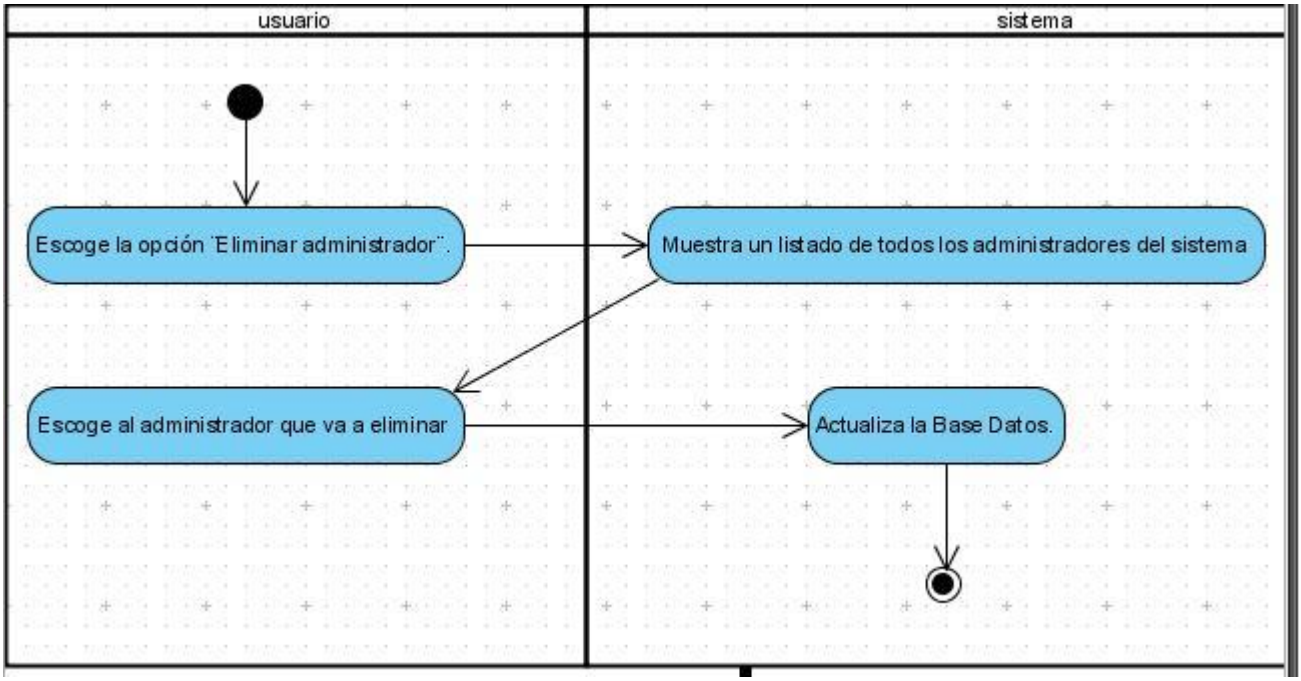

Figure 14: DS "Eliminar administrador"

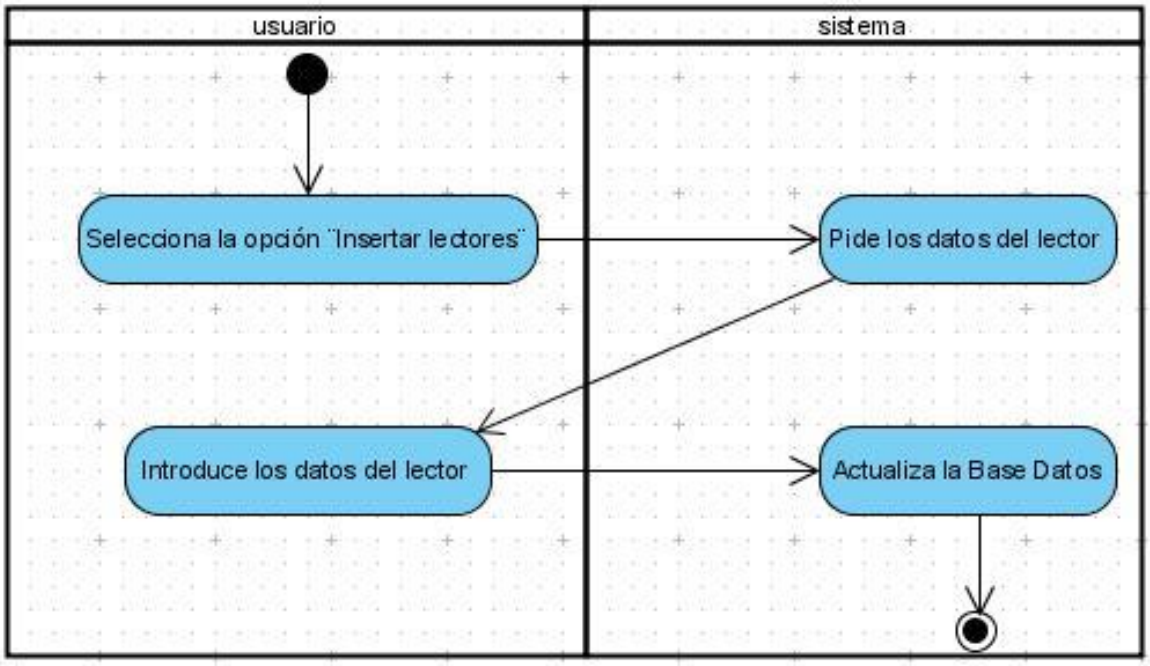

Figure 15: DS "Insertar lectores"

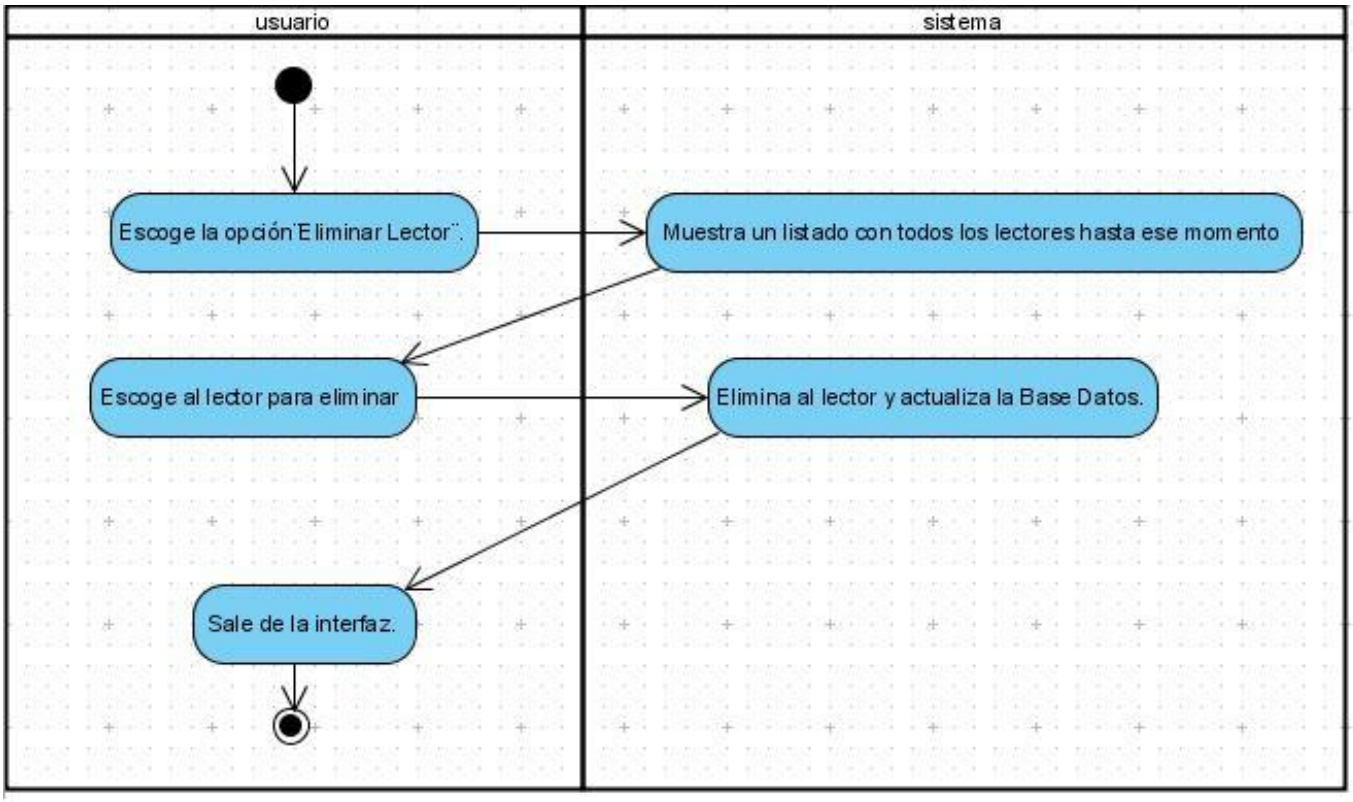

Figure 16: DS "Eliminar lectores"

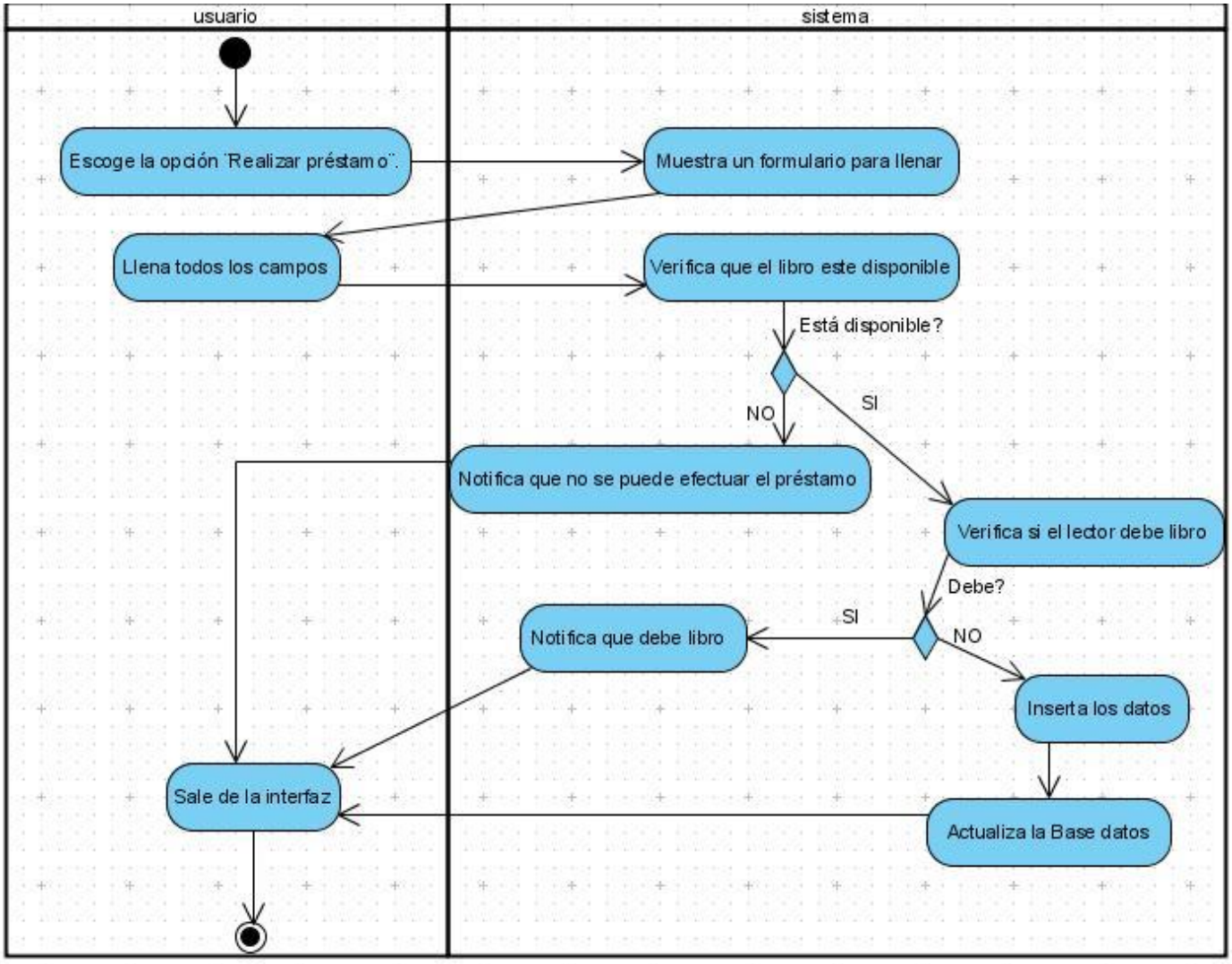

Figure 17: DS "Realizar préstamos"

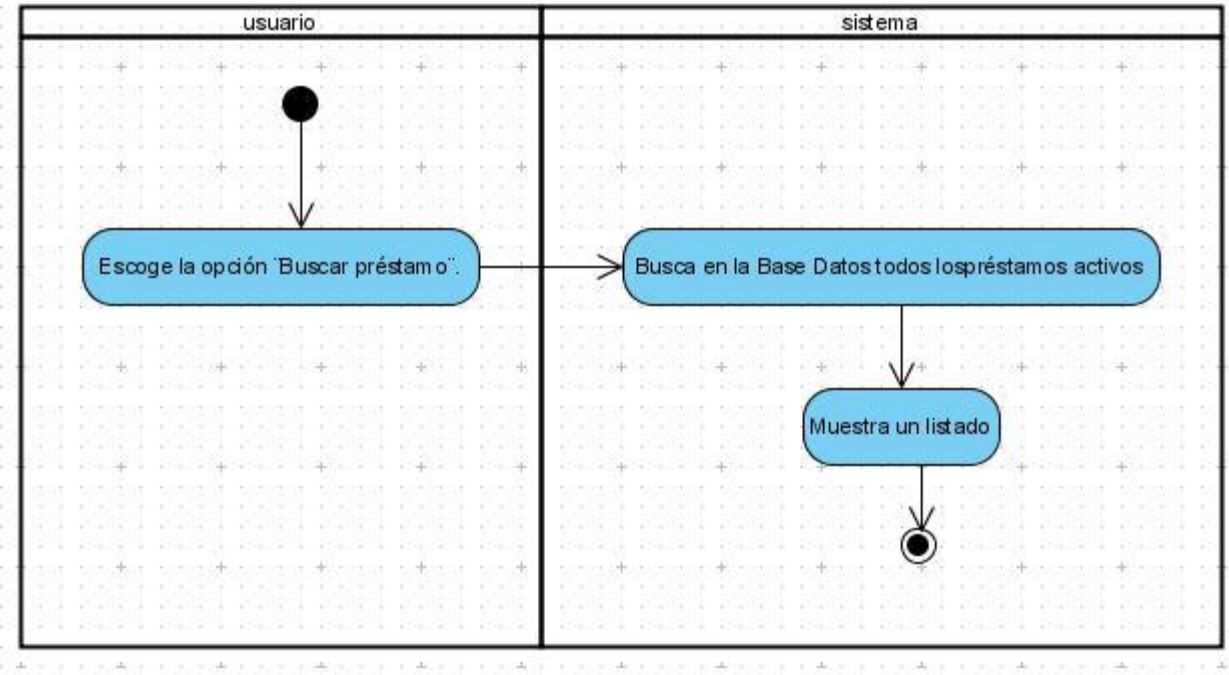

Figura 18: DS "Buscar préstamos"

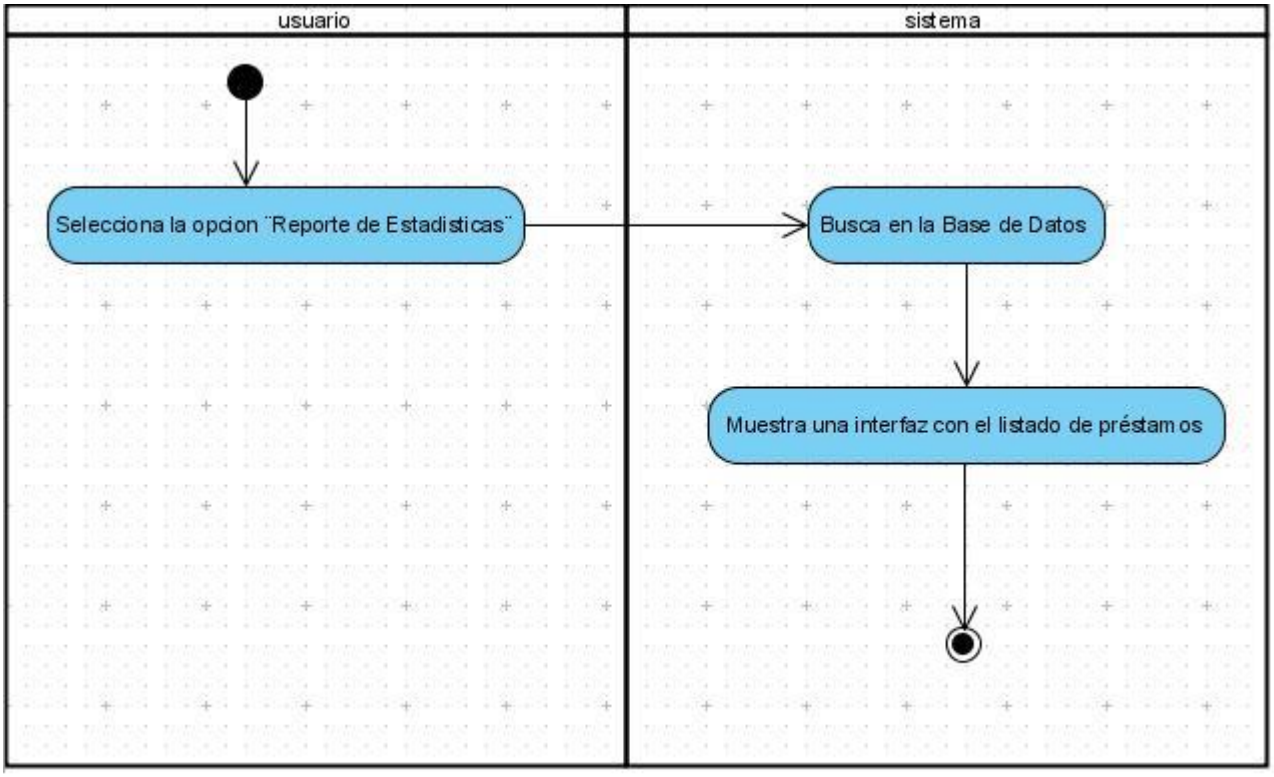

Figura 19: DS "Reporte estadísticas"

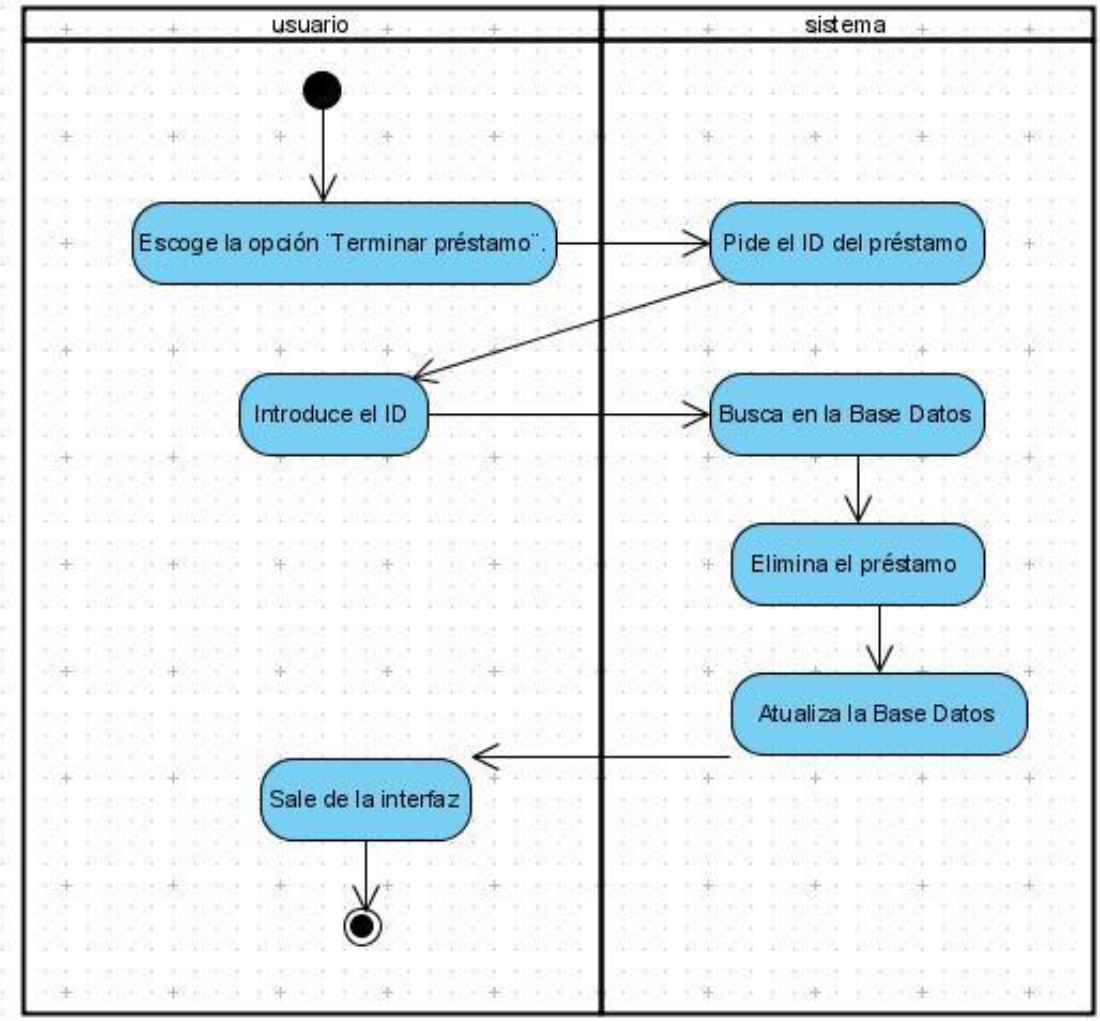

Figura 20: DS "Terminar préstamos"