



UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN

Diseño e Implementación de una Aplicación Web para la Automatización de los Procesos de la Bolsa del Libro

Trabajo Especial de Grado
presentado ante la ilustre
Universidad Central de Venezuela
por las Bachilleres:

Br. Andreina Jiménez
Br. Joselyn Oviedo

Para optar al título de Licenciado en Computación

Tutor: Prof. Jaime A. Parada D.

Caracas, Venezuela
Septiembre 2011

Agradecimientos y Dedicatoria

Andreina Jiménez

Agradezco en primer lugar a Dios, por permitirme lograr este triunfo y ser mi guía en todos los momentos de mi vida.

Quiero además agradecer y dedicar este Trabajo Especial de Grado a mi familia, especialmente a mi madre Carmen Arelis Quintero, por su apoyo incondicional a lo largo de la carrera y en todas las etapas de mi vida, por brindarme su cariño, su protección, sus consejos y por ser uno de los pilares fundamentales de mi educación, guiándome por el camino del bien e incentivándome siempre para lograr mis metas. Este logro es tuyo mamá, ¡TE AMO!.

A mis tíos, Carlos Rafael Quintero, Rosa Margarita Quintero y Marisol Peña, quienes fueron partícipes de este recorrido y me aconsejaron y apoyaron cuando más lo necesitaba.

A mis abuelos, Rosalina Capote y Carlos Alberto Quintero, por el amor que me ofrecen día a día y por formar parte fundamental de mi crecimiento.

A mi novio y prometido Warlys Suárez, por aconsejarme sabiamente, por darme todas las fuerzas necesarias para vencer los obstáculos que se presentaron en el camino, por brindarme su apoyo, motivación, comprensión y tolerancia en los momentos más difíciles y por compartir este éxito a mi lado, simplemente por formar parte de mi vida y llenarla de amor y felicidad a cada instante. Este logro es tuyo mi príncipe bello, ¡TE AMO Cunquero!.

A mi compañera de Trabajo Joselyn Oviedo, por la amistad brindada durante los últimos años de la carrera y por confiar en mí para emprender esta meta juntas, y lograrla satisfactoriamente.

A mis amigos, Glendy Sulbarán y Gabriel Plaza, quienes formaron parte primordial en el desarrollo del Trabajo Especial de Grado, ayudándome de manera desinteresada y brindándome todos sus conocimientos, los cuales me permitieron emprender el rumbo hacia la meta final.

A mi tutor, el Profesor Jaime Parada, por haber dirigido este proyecto, por el apoyo y confianza que ha tenido en mi trabajo y por sus orientaciones, ayuda, atención y correcciones que permitieron el progreso y perfeccionamiento del Trabajo Especial de Grado.

A los profesores Andrés Sanoja y Sergio Rivas, por su gran disposición y contribución para el avance del Trabajo Especial de Grado.

A Marlene Carrizalez, por todas las horas de dedicación brindadas, por su paciencia, su apoyo y constante disponibilidad y colaboración para sacar adelante este Trabajo.

Agradecimientos y Dedicatoria

Joselyn Oviedo

Primeramente, agradezco a Dios por sobre todas las cosas, siendo mi fortaleza y mi guía a lo largo de mi vida. Además de permitirme lograr este gran sueño y anhelo.

A mi madre Gloria Oviedo, por su comprensión, dedicación, amor y confianza en toda mi carrera. Por ser esa madre luchadora de la que toda mi vida estaré inmensamente orgullosa y por creer en mí para llegar a cumplir esta meta. Tu eres parte de este logro mami y también es tuyo ¡Te Amo mami!.

A mi Esposo, por ser ese pilar fundamental en mi carrera, por su inmenso e invaluable apoyo, por ser mi guía y mi fuerza en los momentos más difíciles y creer en mí ciegamente para lograr esta meta. Por darme tanto amor en estos años de mi vida y hacerme tan feliz. Tu eres parte de esta meta y también es tuya ¡Te Amo infinitamente!.

A mis madrinas Lethy Oviedo y Antonia Contreras, por sus sabios consejos y creer en mí para llegar a este logro.

A mi compañera de tesis Andreina Jiménez, por creer en lograr esta meta juntas, por su amistad, y por compartir conmigo este gran trabajo y este gran momento.

A mis amigos, que coseche en la universidad, gracias por su amistad y esos buenos momentos vividos.

A nuestro tutor Jaime Parada, por su respaldo, colaboración y confianza en nuestro trabajo de grado.

A los profesores Andrés Sanoja y Sergio Rivas por su incondicional apoyo, dedicación y tiempo prestado a lo largo de este trabajo.

A Marlene Carrizalez, por su tiempo y colaboración para llevar a cabo la realización y culminación de este trabajo.

A Glendy Sulbarán, Gabriel Plaza y Jorge Sánchez por su colaboración y apoyo brindado en este trabajo.

Índice general

Introducción	1
1. Planteamiento del Problema	3
1.1. Título	3
1.2. Planteamiento del Problema	3
1.3. Objetivos	4
1.3.1. Objetivo General	4
1.3.2. Objetivos Específicos	4
1.4. Importancia y Justificación	4
1.5. Solución del Problema	5
1.6. Arquitectura utilizada	6
2. Marco Conceptual	8
2.1. Bolsa del Libro	8
2.1.1. Servicios que ofrece la Bolsa del Libro	9
2.1.2. Procesos involucrados en la Bolsa del Libro	10
2.2. Sistemas anteriores	15
2.2.1. Sistema de Información de la Bolsa del Libro de la Facultad de Ciencias (UCV)	16
2.2.2. Desarrollo de un Sistema de Software Orientado a Objetos para una Bolsa del Libro	16
2.2.3. Desarrollo de Aplicaciones de Comercio Electrónico usando Software Libre. Caso de estudio: Bolsa del Libro de la Facultad de Ciencias de la UCV	17
3. Marco Tecnológico	18
3.1. Ruby On Rails	18
3.1.1. Ventajas del lenguaje Ruby	18
3.1.2. Framework Rails	19
3.2. REST	19
3.2.1. Servicios Web de Amazon	19
3.3. MySQL	20
3.3.1. Ventajas	20
3.4. Subversion	20
3.4.1. Características básicas	20
3.5. AJAX	21
3.5.1. Ventajas	21
3.6. Programación Extrema (XP)	21

3.6.1. Prácticas	21
3.6.2. Valores	22
3.6.3. Actividades	22
4. Marco Aplicativo	24
4.1. Adaptación de la Metodología XP	24
4.1.1. Actores y Responsabilidades	24
4.1.2. Metáfora del Sistema	25
4.1.3. Adaptación de las fases de XP	26
Conclusiones	83
Recomendaciones	84
Bibliografía	85
Apéndice	86

Índice de figuras

1.1. Arquitectura Cliente/Servidor de 3 capas	6
2.1. Estructura Organizativa y Procesos asociados a la Biblioteca Alonso Gamero	9
2.2. Ejemplo de Listado de Libros existentes por título	14
2.3. Ejemplo de Listado de Deudores	14
2.4. Ejemplo de Listado de Inventario	15
2.5. Ejemplo de Listado de Libros prestados a otros usuarios	15
4.1. Metáfora del Sistema	25
4.2. Descripción del Proceso de Adquisición de Libros	26
4.3. Descripción del Proceso de Alquiler de Libros	27
4.4. Descripción del Proceso de Solvencias	28
4.5. Descripción del Proceso de Desincorporación de Libros	29
4.6. Listados	30
4.7. Modelo de Datos	32
4.8. Vista Inicial del Módulo de Alquiler desde el perfil de Estudiante	34
4.9. Vista de Error generado en el Módulo de Alquiler desde el perfil de Estudiante cuando el Estudiante es deudor	35
4.10. Vista del Documento PDF del Comprobante de Alquiler	35
4.11. Vista Inicial del Módulo de Alquiler desde el perfil del Personal Administrativo	36
4.12. Vista desde donde se efectúa la Devolución de libros por parte del Personal Administrativo	36
4.13. Código del método para la generación del comprobante PDF de Alquiler (Parte 1).	37
4.14. Código del método para la generación del comprobante PDF de Alquiler (Parte 2).	38
4.15. Código del método para realizar la búsqueda de los libros a alquilar.	38
4.16. Código de la vista Index para realizar la búsqueda de los libros a alquilar.	39
4.17. Código de la vista Javascript para realizar la búsqueda de los libros a alquilar.	39
4.18. Código del condicional en el Controlador para realizar la selec- ción de los libros a alquilar.	40
4.19. Código de la vista Index para realizar la selección de los libros a alquilar.	40
4.20. Vista del catálogo de libros desde el perfil de Administrador	43

4.21. Vista para la creación de un nuevo libro	43
4.22. Vista donde se muestran en detalle los datos de un libro	44
4.23. Vista del catálogo de ejemplares del personal Administrativo . .	44
4.24. Código definido para realizar la búsqueda de un libro en Amazon.	45
4.25. Código de la vista Nuevo para la creación de un libro.	45
4.26. Código de la vista JavaScript para la creación de un libro.	45
4.27. Código del archivo de migración AddAttachmentsImagenToLibro.	46
4.28. Código para definir las validaciones de Paperclip.	46
4.29. Parte del código de la vista de Nuevo Libro con el formulario multipart.	47
4.30. Código que permite mostrar la imagen adjuntada a un libro. . .	47
4.31. Vista donde el estudiante debe seleccionar el tipo de solvencia que requiere	48
4.32. Vista del Documento PDF del Comprobante de Solvencia de tipo Retiro	49
4.33. Vista del Documento PDF del Comprobante de Solvencia de tipo Grado	49
4.34. Vista de la Administración de Solvencias	50
4.35. Código del método para la generación del comprobante PDF de Solvencia de tipo Retiro o Cambio (Parte I).	51
4.36. Código del método para la generación del comprobante PDF de Solvencia de tipo Retiro o Cambio (Parte II).	51
4.37. Código del método para la generación del comprobante PDF de Solvencia de tipo Grado (Parte I).	52
4.38. Código del método para la generación del comprobante PDF de Solvencia de tipo Grado (Parte II).	52
4.39. Vista de la selección de los tipos de listados	54
4.40. Vista del listado de libros adquiridos	54
4.41. Vista del documento PDF del listado de libros adquiridos . . .	55
4.42. Código del método selección.	55
4.43. Código para elegir un tipo de listado.	56
4.44. Código para la creación del listado de libros alquilados.	56
4.45. Parte del Código del método para la generación del documento PDF del Listado de Libros Adquiridos	57
4.46. Vista del Módulo de Usuarios	59
4.47. Parte del código creado para la visualización del Módulo de Usua- rios.	59
4.48. Vista del Módulo de Configuraciones	61
4.49. Parte del código que genera la tabla de las Configuraciones. . . .	61
4.50. Vista de creación de un Periodo Lectivo	63
4.51. Vista donde aparecen todos los Periodos Lectivos creados . . .	63
4.52. Código para la creación de un Periodo Lectivo.	64
4.53. Parte del Código para visualizar la tabla de Periodos Lectivos. .	64
4.54. Vista de la página principal de la Bolsa del Libro	66
4.55. Vista de la página de inicio desde el Perfil de Administrador .	66
4.56. Vista de la opción de Cambiar Clave	67
4.57. Vista de la opción de Olvido de Clave	67

4.58. Vista de la Tabla de Auditorias de Olvido de Clave	68
4.59. Métodos del Módulo de Autenticación	69
4.60. Métodos de Autenticación del Controlador Sesión	70
4.61. Métodos de Autenticación del Modelo Usuario	70
4.62. Código para realizar el cambio de clave	71
4.63. Código del método de olvido de clave	71
4.64. Código que aplica los permisos de Administrador en el sistema .	72
4.65. Código del método usuario_puede	73
4.66. Código del condicional para verificar si el usuario puede visu- alizar el botón Alquileres y el enlace de la opción Mostrar	73
4.67. Código del condicional para verificar el perfil del usuario actual e imprimirlo en pantalla	74
4.68. Vista de la página que permite realizar la actualización de la información de la página principal	75
4.69. Listado de libros agotados en la Bolsa del Libro	76
4.70. Código para generar el editor de texto enriquecido NicEdit . . .	76
4.71. Código del método libros_agotados	77
4.72. Vista del Módulo de Auditorias	78
4.73. Parte del código que genera la tabla de las Auditorias.	79
4.74. Vista principal de la Funcionalidad de Carga Masiva	80
4.75. Código que guarda el archivo de los estudiantes en el sistema (Parte I).	81
4.76. Código que guarda el archivo de los estudiantes en el sistema (Parte II).	81

Índice de cuadros

4.1. Esquema de actores y roles que desempeñan	25
4.2. Formato de registro para una Historia de Usuario	30
4.3. Esquema de planificación de cada iteración	31
4.4. Formato de registro de Prueba de Aceptación	33
4.5. Esquema del Módulo de Pre-Alquiler y Alquiler con su MVC asociado	33
4.6. Pruebas de Aceptación de la Iteración 0	41
4.7. Esquema del Módulo de Libros y Ejemplares con su MVC asociado	42
4.8. Pruebas de Aceptación de la Iteración 1	47
4.9. Esquema del Módulo de Solvencias con su MVC asociado	48
4.10. Pruebas de Aceptación de la Iteración 2	53
4.11. Esquema del Módulo de Listados con su MVC asociado	53
4.12. Pruebas de Aceptación de la Iteración 3	58
4.13. Esquema del Módulo de Usuarios con su MVC asociado	58
4.14. Pruebas de Aceptación de la Iteración 4	60
4.15. Esquema del Módulo de Configuraciones con su MVC asociado	60
4.16. Pruebas de Aceptación de la Iteración 5	62
4.17. Esquema del Módulo de Periodos Lectivos con su MVC asociado	62
4.18. Pruebas de Aceptación de la Iteración 6	65
4.19. Esquema del Módulo de Autenticación con su MVC asociado	65
4.20. Pruebas de Aceptación de la Iteración 7	74
4.21. Esquema de la Funcionalidad de Actualizaciones con su MVC asociado	75
4.22. Pruebas de Aceptación de la Iteración 8	77
4.23. Esquema del Módulo de Auditorias con su MVC asociado	78
4.24. Pruebas de Aceptación de la Iteración 9	79
4.25. Esquema de la Funcionalidad de Carga Masiva con su MVC asociado	79
4.26. Pruebas de Aceptación de la Iteración 10	82

Resumen

La Bolsa del Libro de la Facultad de Ciencias de la Universidad Central de Venezuela, lleva a cabo sus diversos procesos de manera manual. Como ejemplo de estos procesos tenemos: el Alquiler de Libros, la generación de Solvencias, la adquisición y desincorporación de Libros y la elaboración de Listados.

En este Trabajo Especial de Grado se presentan los resultados de la automatización de los procesos mencionados anteriormente, mediante la implementación de una Aplicación Web que consta de nueve Módulos y dos funcionalidades. Esta Aplicación se desarrolló utilizando una adaptación de la metodología ágil “Programación Extrema” (XP). Entre las tecnologías usadas se destaca el framework de desarrollo de Aplicaciones Web Rails 2.3.8, el lenguaje de programación Ruby 1.8, la técnica REST, el Sistema Manejador de Base de Datos MySQL 2.8.1, la tecnología AJAX y el software para el control de versiones Subversion.

La creación de los Módulos de la Aplicación Web aporta un agregado significativo a la Bolsa del Libro, automatizando y agilizando sus servicios. En el siguiente Trabajo, para cada uno de los Módulos creados, se explica en detalle su proceso de desarrollo.

Palabras claves: Bolsa de Libro, Aplicación Web, Automatización.

Introducción

Actualmente, el uso de la Web ha crecido y es de vital importancia para la sociedad, ya que se ha convertido en un medio indispensable dentro del campo de la información y las comunicaciones, por lo cual, se ha tomado en cuenta como una herramienta muy poderosa para el desarrollo de aplicaciones que faciliten la realización de las actividades cotidianas.

La Bolsa del Libro de la Facultad de Ciencias de la Universidad Central de Venezuela, utiliza un sistema de trabajo que no aprovecha las tecnologías existentes, ya que muchas de las tareas diarias son realizadas de forma manual, invirtiendo mucho tiempo en efectuarlas.

El objetivo del presente Trabajo es presentar el desarrollo de una Aplicación Web para la automatización de los procesos de la Bolsa del Libro, que contribuye a mejorar la calidad y eficiencia de los mismos, permitiendo que su ejecución sea sistematizada vía Web.

Para la realización de la Aplicación Web, se realizó un levantamiento de información de los distintos procesos y servicios de la Bolsa del Libro y se estudió en detalle el flujo de actividades que se realizan en cada uno, lo cual permitió implementar un sistema hecho conforme a las necesidades y requerimientos existentes. También se realizó una adaptación de la metodología ágil “Programación Extrema” (XP), la cual se caracteriza por su simplicidad, comunicación, retroalimentación y refactorización de código.

La visión general de cada uno de los capítulos que se presentan en el siguiente Trabajo Especial de Grado es la siguiente:

Capítulo I: Planteamiento del Problema, en el cual se expone el problema, se describen los objetivos propuestos, la importancia y justificación, la solución del problema y la arquitectura utilizada para el desarrollo de la Aplicación Web.

Capítulo II: Marco Conceptual, donde se describen los aspectos referentes a la Bolsa del Libro de la Facultad de Ciencias de la Universidad Central de Venezuela, desde su perfil, los servicios ofrecidos, los procesos que intervienen para llevar a cabo las tareas cotidianas y los sistemas anteriores que se han propuesto para la automatización de la Bolsa del Libro.

Capítulo III: Marco Tecnológico, donde se describen las tecnologías utilizadas, tales como el Lenguaje de Programación Ruby, el Framework de desarrollo de Aplicaciones Web Rails, la técnica REST (Representational State Transfer), el Sistema Mane-

jador de Bases de Datos MySQL, la tecnología AJAX (Asynchronous JavaScript And XML) y la metodología de desarrollo de software utilizada (XP).

Capítulo IV: Marco Aplicativo, en el que se especifican el modelo de datos, la adaptación de las fases de XP y las actividades realizadas en cada una de las iteraciones que conforman el desarrollo del sistema.

Para finalizar con la estructura del Trabajo Especial de Grado se presentan las conclusiones, las recomendaciones y algunas referencias bibliográficas y digitales consultadas durante el desarrollo del documento y la Aplicación.

Capítulo 1

Planteamiento del Problema

En este capítulo se proporciona una explicación acerca de la situación actual de la Bolsa del Libro, se enumeran los objetivos propuestos, se describe la importancia, la justificación y la solución del problema, y se presenta una breve explicación de la arquitectura utilizada para el desarrollo de la Aplicación Web.

1.1. Título

Diseño e Implementación de una Aplicación Web para la Automatización de los Procesos de la Bolsa del Libro.

1.2. Planteamiento del Problema

Actualmente, la Bolsa del Libro de la Facultad de Ciencias de la Universidad Central de Venezuela, utiliza un sistema de trabajo manual para llevar a cabo las diferentes actividades, relacionadas a los servicios ofrecidos a la comunidad estudiantil. Entre esas actividades se destacan: el préstamo semestral de libros, la emisión de solvencias, el registro y la desincorporación de libros y la generación de listados relacionados con la información de los libros, estudiantes y otros usuarios del servicio.

Una de las principales causas que afecta de manera negativa la ejecución de las tareas mencionadas anteriormente es el tiempo que se invierte en poder realizarlas, ya que actualmente existe una sola persona encargada de efectuarlas. Además algunas tareas del flujo de trabajo involucran a varios entes externos de la Bolsa del Libro, que no responden rápidamente a las solicitudes realizadas, lo que trae como consecuencia la acumulación de trabajo y el retraso en la ejecución de dichas tareas.

Por tal razón se desarrolló una Aplicación Web, para mejorar el desempeño general de los procesos y servicios involucrados en la Bolsa del Libro.

1.3. Objetivos

Los objetivos propuestos en el Trabajo Especial de Grado se especifican a continuación:

1.3.1. Objetivo General

Desarrollar una Aplicación Web que facilite el control y la ejecución de los procesos que se llevan a cabo en la Bolsa del Libro.

1.3.2. Objetivos Específicos

- Realizar el levantamiento de información de los procesos que se realizan en la Bolsa del Libro.
- Adaptar la metodología de desarrollo XP a la implementación de la Aplicación Web.
- Diseñar e implementar el modelo de la Base de Datos, que permita reflejar la información existente en la Bolsa del Libro.
- Diseñar las interfaces de usuario correspondientes a los Módulos de la Aplicación Web, utilizando los lineamientos establecidos para su desarrollo.
- Implementar los Módulos que permitan llevar a cabo los servicios que ofrece la Bolsa del Libro.
- Someter a pruebas de verificación la Aplicación Web realizada para comprobar su funcionamiento.

1.4. Importancia y Justificación

En la Bolsa del Libro no se cuenta con un sistema automatizado que facilite la ejecución de los procesos, sino que por el contrario, se efectúan de manera manual, sin darle uso a los recursos tecnológicos con los que se cuentan hoy en día, invirtiendo mucho tiempo en la realización de los mismos. Por tal razón, surgió la iniciativa de crear una Aplicación Web que permite automatizar dichos procesos, mejorando la ejecución de las tareas administrativas y logrando que los estudiantes puedan realizar el alquiler semestral de libros, entre otras acciones, de manera cómoda, haciendo uso de un computador.

Es importante mejorar esos procesos, ya que de esa manera se contribuye al ahorro de tiempo, dedicación y esfuerzo, minimizando los posibles errores que puedan existir durante las actividades realizadas manualmente, agilizando todas las tareas administrativas y disminuyendo la carga laboral al personal administrativo, además de permitir que el estudiante pueda acceder a los servicios ofrecidos por la Bolsa del Libro, desde cualquier lugar donde haya conexión y acceso a Internet.

1.5. Solución del Problema

Se desarrolló una Aplicación Web para la automatización de las actividades de la Bolsa del Libro, facilitando y agilizando los procesos y servicios involucrados, con el fin de efectuarlos de forma eficiente, manteniendo el registro y control de la información asociada con los libros y los usuarios del servicio y permitiendo al personal administrativo y a los estudiantes, realizar acciones con la Bolsa del Libro desde Internet.

Dicha Aplicación Web, consta de nueve Módulos y dos funcionalidades: Módulo de Autenticación, Módulo de Pre-Alquiler y Alquiler, Módulo de Solvencias, Módulo de Libros y Ejemplares, Módulo de Listados, Módulo de Auditorias, Módulo de Configuraciones, Módulo de Usuarios, Módulo de Periodos Lectivos, funcionalidad de Actualizaciones y funcionalidad de Carga Masiva.

El primer Módulo (Autenticación) permite el ingreso del Personal de la Bolsa del Libro (Administrador y Empleados) y de los Estudiantes de la Facultad de Ciencias al sistema de la Bolsa del Libro. A través del segundo Módulo (Pre-Alquiler y Alquiler), el Estudiante puede realizar la pre-solicitud de varios Ejemplares para posteriormente efectuar la solicitud de alquiler de los Ejemplares que desee y el Administrador maneja dichas solicitudes y la devolución de los Ejemplares. El tercer Módulo (Solvencias), permite al Estudiante realizar la solicitud del tipo de solvencia que requiera y así el Administrador recibe dicha solicitud para posteriormente imprimir el comprobante de solvencia generado y entregárselo al Estudiante.

En el cuarto Módulo (Libros y Ejemplares) el Personal Administrativo tiene el manejo de los Libros que se encuentran en la Bolsa del Libro y puede manipular todos los Ejemplares que le pertenecen a un determinado Libro. El Estudiante de acuerdo al manejo de los Libros observa los mismos a través de un Catálogo de Libros. En el quinto Módulo (Listados), el Administrador genera los reportes (Listado de Libros Adquiridos, Listado de Libros Alquilados, Listado de Libros por Título, Listado de Libros Excluidos, Listado de Inventario, Listado de Cartelera, Listado de Deudores y Listado de Multados) que sean necesarios.

En el sexto Módulo (Auditorias), el Personal Administrativo tiene el histórico de los elementos que han sido eliminados del sistema (Alquileres, Libros, Ejemplares y Solvencias). El séptimo Módulo (Configuraciones) permite al Administrador controlar variables de acuerdo a las necesidades que se presenten para un determinado momento, tales como: cantidad de libros a pre-alquilar y alquilar en un semestre particular, cantidad de días disponibles para que el Estudiante retire en la Bolsa del Libro los libros que alquiló, entre otras.

En el octavo Módulo (Usuarios), el Administrador tiene el manejo de los Estudiantes que están relacionados con la Bolsa del Libro. En el noveno Módulo (Periodos Lectivos) el Personal Administrativo crea el periodo académico correspondiente a un semestre en particular, con el fin de asociar los alquileres, solvencias, listados y usuarios a ese periodo.

La funcionalidad de Actualizaciones, permite al Personal Administrativo realizar ac-

tualizaciones de la información del préstamo rental de libros en la página principal del sistema y la funcionalidad de Carga Masiva permite al Personal Administrativo realizar la carga semestral del archivo con los estudiantes para un semestre específico.

Además se cuenta con tres Perfiles de Usuario para acceder a la Aplicación Web: Administrador, Empleado y Estudiante. El Administrador y el Empleado lo constituyen el Personal Administrativo de la Bolsa del Libro, donde el Administrador puede realizar cualquier acción (Mostrar, Editar, Eliminar) dentro del sistema, mientras que el Empleado tiene una restricción en las acciones que puede realizar en el sistema. Los Estudiantes lo constituyen toda la comunidad estudiantil de la Facultad de Ciencias.

1.6. Arquitectura utilizada

La Aplicación Web se desarrolló utilizando el esquema de la figura 1.1, el cual muestra la Arquitectura Cliente/Servidor de 3 capas: Capa de Presentación, Capa de Negocio y Capa de Datos. La información de esta sección está basada en [7].

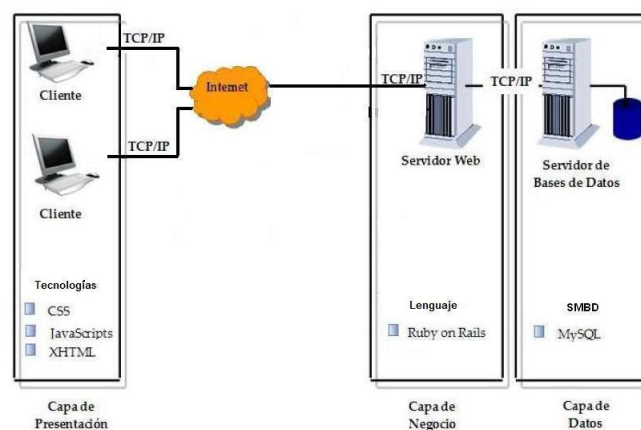


Figura 1.1: Arquitectura Cliente/Servidor de 3 capas

- **Capa de Presentación:** Esta capa está compuesta por los Clientes (Administrador, Empleado, Estudiantes) que solicitan datos al servidor Web, a través de Internet. Las tecnologías manejadas en esta capa son: CSS, JavaScript, XHTML, entre otras. Además ensambla la información que será enviada en código HTML al navegador Web de los usuarios, y recibe los requerimientos que estos solicitan.
- **Capa de Negocio:** Esta capa es la que mantiene la sincronización entre la Capa de Presentación y la Capa de Datos, se encarga de recibir directamente las solicitudes de los Clientes. Además contiene la Aplicación, que en este caso, está realizada con el framework de desarrollo de Aplicaciones Web Ruby On Rails.

- **Capa de Datos:** Esta capa es la encargada de realizar la gestión de Datos, interactúa directamente con la Base de Datos a fin de recuperar o almacenar la información que le solicita la Capa de Negocio en la misma. Estas tareas son realizadas por el Sistema Manejador de Base de Datos MySQL.

Capítulo 2

Marco Conceptual

La finalidad de este capítulo es presentar las bases conceptuales que sirven de fundamento para el desarrollo de este Trabajo Especial de Grado. En la primera sección, se hace una descripción de la organización a la cual se le implementó la Aplicación Web. En la segunda sección se hace referencia a sistemas anteriores que fueron diseñados para la automatización de las actividades administrativas de la Bolsa del Libro.

2.1. Bolsa del Libro

El Servicio Rental de la Bolsa del Libro de la Facultad de Ciencias de la Universidad Central de Venezuela, es un servicio bibliotecario estudiantil autogestionario que consiste en el alquiler semestral de un número fijo de textos a los estudiantes de la Facultad de Ciencias. Esta información de la Bolsa del Libro está basada en [18].

La Bolsa del Libro fue creada en Octubre de 1969 por la Doctora y Profesora de la Escuela de Química Dora Turk de García Banus [18], con el fin de promover los beneficios estudiantiles.

Actualmente, la Bolsa del Libro funciona adscrita a la Biblioteca de la Facultad de Ciencias “Alonso Gamero”, cuenta con una existencia aproximada de 525 títulos y 4150 ejemplares y está dirigida por la Licenciada en Biología Marlene Carrizalez. Esta información de la Bolsa del Libro y la siguiente, está basada en [19].

La Bolsa del Libro tiene como *objetivo principal* **apoyar a la comunidad de la Facultad de Ciencias en el proceso de formación académica a través del préstamo rental de libros.**

Tiene como *objetivos específicos*:

- Permitir a los estudiantes contar con los libros de texto durante todo el semestre, mediante el pago de una módica cuota.
- Servir de auxiliar a la Biblioteca, siendo mayor la capacidad de préstamo de textos básicos para los estudiantes.

- Fomentar la responsabilidad en el estudiante, quien debe cuidar el texto para devolverlo en buenas condiciones al finalizar el préstamo.

2.1.1. Servicios que ofrece la Bolsa del Libro

La figura 2.1 muestra la estructura organizativa y los procesos asociados a la Biblioteca “Alonso Gamero” [1]. En ella se observa que la Bolsa del Libro de la Facultad de Ciencias, es uno de los servicios ofrecidos por la Biblioteca Alonso Gamero y a su vez, la misma proporciona tres servicios a la comunidad estudiantil: Alquiler de Libros, Venta de Libros y Emisión de Solvencias. La información de esta sección está basada en [19].

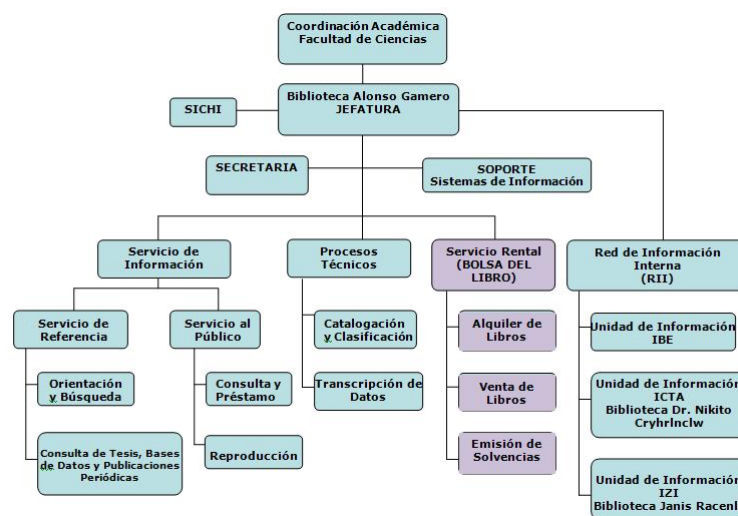


Figura 2.1: Estructura Organizativa y Procesos asociados a la Biblioteca Alonso Gamero

Cada uno de los servicios pertenecientes a la Bolsa del Libro se describe a continuación:

1. **Alquiler de Libros:** Consiste en el alquiler semestral de libros a los estudiantes de la Facultad de Ciencias. El estudiante que ingresa por primera vez a la Facultad tiene la posibilidad de alquilar un máximo de 3 textos y el estudiante regular puede alquilar un máximo de 4 textos, ambos a partir del segundo día de préstamo, ya que el primer día de préstamo sólo se permite alquilar un libro.
2. **Venta de Libros:** Consiste en la venta de los libros existentes en la Bolsa del Libro que han sido desincorporados del servicio, con previa autorización de las autoridades respectivas y la venta de los libros editados por la Facultad de Ciencias.
3. **Emisión de Solvencias:** Consiste en la entrega de un comprobante de solvencia a los estudiantes de la Facultad de Ciencias que lo requieran para solicitud de grado, equivalencias, notas certificadas, traslados, retiro, reincorporación, cambio de Escuela o Facultad.

2.1.2. Procesos involucrados en la Bolsa del Libro

En la Bolsa del Libro intervienen varios procesos que manejan y controlan la información de los estudiantes beneficiarios del servicio durante un periodo lectivo. Uno de los procesos principales es el alquiler de libros, el cual permite ayudar a los estudiantes pertenecientes a las distintas Escuelas de la Facultad que lo requieran y en algunos casos a personal administrativo.

Otros procesos que se destacan son: la generación de solvencias, la adquisición de libros y la desincorporación de libros. Además en la Bolsa del Libro se manejan varios tipos de listados como: listado de libros existentes por título, listado de deudores, listado de inventario y listado de libros prestados a otros usuarios.

Es importante resaltar que estos procesos, involucran la intervención de otros entes, que están en constante comunicación con la Bolsa del Libro, como la División de Control de Estudios, la Coordinación Académica, la Biblioteca y la Caja. A continuación se describen los procesos mencionados.

■ Proceso de Alquiler

En el proceso de Alquiler, la Bolsa del Libro publica en cartelera la lista de libros y precios, el proceso y los horarios de alquiler. El estudiante dependiendo del día realiza una determinada actividad.

Alquiler en el primer día: En el primer día, a cada estudiante por orden de llegada se le entrega una tarjeta de usuario numerada, para que la llene con sus datos y haga la solicitud correspondiente de un solo libro. Luego dependiendo del rango de numeración que presente su tarjeta, el estudiante tendrá una hora fijada por la Bolsa del Libro para ser atendido.

En ese lapso de tiempo el estudiante se dirige a la cartelera, consulta los datos del libro que desea alquilar (cota, autor, título y precio correspondiente), llena la tarjeta y vuelve a la Bolsa del Libro. En la Bolsa del Libro, espera ser llamado según la numeración de su tarjeta y entrega su constancia de inscripción del semestre actual previamente sellada en Control de Estudios y su carnet estudiantil. Estos documentos serán comparados con las listas de deudores de libros y de dinero (multados) para verificar la situación del estudiante (solvente, deudor o multado).

En caso que el estudiante sea deudor, deberá devolver él o los libros que posea para poder alquilar uno nuevo. En caso contrario entregará la tarjeta y se verificará si el libro que solicita está disponible, en dicho caso, el estudiante cancela el monto del libro y se le entrega el libro junto con la normativa de la Bolsa del Libro. Luego el alquiler será registrado en *Microsoft Access*. Si el libro no se encuentra disponible, se le sugiere un nuevo libro que corresponda a su requerimiento.

Alquiler en el segundo y tercer día: En el segundo y tercer día, la persona encargada de la Bolsa del Libro les pregunta a los estudiantes que se encuentran en la fila si ya alquilaron libros, y en caso de ser positiva su respuesta, solo se les entrega un número, para posteriormente llamarlos según el orden de numeración. Luego, el estudiante consulta en cartelera los datos del libro que desea alquilar y entrega su constancia de inscripción y su carnet estudiantil. Una vez que el estudiante esté dentro de la Bolsa del Libro se le entrega su tarjeta de usuario, y debe llenarla con los datos de los libros que va a solicitar, cancelando el monto del libro allí mismo y continuando el proceso como en el primer día de alquiler.

En caso de que el estudiante no haya alquilado libros aún, se le entrega una tarjeta de usuario que debe rellenar y un número. Luego debe consultar en cartelera los datos del libro que desea alquilar e igualmente será atendido en la Bolsa del Libro según el orden de numeración, y el proceso continúa como el del primer día de alquiler.

Alquiler en otros días y en el curso intensivo (verano): El estudiante debe consultar en cartelera los datos del libro que desea alquilar y es atendido en la Bolsa del Libro por orden de llegada. Luego, entrega su constancia de inscripción junto con su carnet estudiantil y se cumple el proceso de verificación de solvencia como en el primer día de alquiler. Después, se le entrega al estudiante una tarjeta de usuario, la cual debe llenar con sus datos y los datos del libro que va a solicitar y el proceso continúa como en el primer día de alquiler, cancelando el monto del libro en la Bolsa del Libro si se trata del alquiler en verano; y cancelando el monto en la caja si se trata del alquiler en otros días. En este último caso el estudiante entrega el recibo de pago en la Bolsa de Libro.

Finalizado el semestre el estudiante deberá devolver el(los) libro(s) dentro de un límite de tiempo fijado previamente por la Bolsa del Libro, si pasada esa fecha el usuario no ha devuelto los libros, adquiere la condición de deudor y al momento de la entrega deberá cancelar el cien por ciento del costo del alquiler, pues de lo contrario quedará suspendido para el siguiente préstamo rental y no podrá inscribirse en el siguiente semestre, según un acuerdo con la Directora de la División de Control de Estudios.

■ Proceso de Solvencias

En este proceso, el estudiante se dirige a la Bolsa del Libro y hace la solicitud de solvencia, entrega su carnet estudiantil y luego el personal de la Bolsa del Libro verifica en *Microsoft Access* si el estudiante es deudor de libros del semestre actual. Si el estudiante es deudor, se busca en el fichero la tarjeta de usuario para asignar el costo asociado; el estudiante debe cancelar su deuda para que se genere su solvencia.

Si no es deudor de libros del semestre actual, el personal de la Bolsa del Libro revisa la carpeta de deudores de semestres anteriores y multados, para verificar si

el estudiante está en una de esas condiciones. Si es deudor, se verifica el libro que tiene en alquiler y se indica la deuda, y luego de ser cancelada, se verifica el tipo de solvencia a ser generada. Si no es deudor de semestres anteriores, se emite el tipo de solvencia que sea requerida.

Si la solvencia es por grado, se entrega al estudiante una lista de libros a donar. En este proceso el estudiante elige el libro que más se ajuste a su presupuesto o preferencia y hace la donación del mismo; luego el personal de la Bolsa del Libro le entrega un formato que debe llenar con los datos personales (nombre y apellido, cédula de identidad y escuela), fecha de donación y datos del Libro (título, autor y edición), el cual luego se coloca en el libro para pasar la información a un cuaderno de donación. El libro es registrado en el cuaderno de donación y luego en la carpeta de libros existentes. Posteriormente se registra la solvencia en una carpeta que contiene datos del estudiante, fecha de la solicitud y motivo del retiro y se genera la solvencia.

Si la solvencia es por retiro o cambio, se registra la solvencia en una carpeta y luego se genera la solvencia.

■ **Proceso de Adquisición de Libros**

La adquisición de libros en la Bolsa del Libro se lleva a cabo a través de diferentes vías: por Compra de Ingresos Propios, por Coordinación Académica y por donación de Grado o de otras Instituciones.

Adquisición por Compra de Ingresos Propios: Para la adquisición de libros por esta vía, se realiza una lista de los libros más demandados, luego se llama al proveedor y se pide el presupuesto. Posteriormente se realiza una requisición con la lista de libros, que se entrega junto con el presupuesto en la Biblioteca, luego estos documentos son enviados a la Coordinación Académica para esperar su aprobación. Si la adquisición es aprobada, se llama al proveedor y se hace el pedido de los libros. Cuando el pedido es recibido, se registran los libros en la carpeta de libros existentes, se actualiza la lista de inventario y luego se actualiza la lista en *Microsoft Excel* y posteriormente en *Microsoft Access*.

Adquisición por Coordinación Académica y por donación de otras Instituciones: En este caso la adquisición de libros se efectúa mediante dos vías, por Coordinación Académica o por donación de otras Instituciones. Una vez que los libros sean donados y se encuentren en la Bolsa del Libro, son registrados en la carpeta de libros existentes, se actualiza la lista de inventario y luego se actualiza la lista en *Microsoft Excel* y posteriormente en *Microsoft Access*.

Adquisición por donación de Grado: En este tipo de adquisición, el estudiante se dirige a la Bolsa del Libro y solicita la lista de libros a donar, escoge un libro y luego que realiza la donación del texto, el mismo es registrado en el cuaderno de

donación y luego en la carpeta de libros existentes, se actualiza la lista de inventario y luego se actualiza la lista en *Microsoft Excel* y posteriormente en *Microsoft Access*.

■ Proceso de Desincorporación de Libros

En el proceso de desincorporación de libros, en la Bolsa del Libro se revisan todos los libros y se determina el tipo de desincorporación, que puede ser de dos maneras: por Deterioro y por Donación.

Desincorporación por Deterioro: En este tipo de desincorporación se seleccionan los libros deteriorados y luego se verifica el estado del deterioro, si el libro está muy deteriorado, se desincorpora el libro de la Bolsa del Libro y se registra la desincorporación en la carpeta de libros existentes y luego en *Microsoft Excel*; en caso contrario, se verifica si el libro es demandado, en dicho caso, se manda a reparar, de lo contrario, se consulta con la Comisión de Biblioteca el destino del libro, si el mismo se encuentra muy estropeado se vende, sino se envía a la Biblioteca y se registra la desincorporación en la carpeta de libros existentes y luego en *Microsoft Excel*.

Desincorporación por Donación: Al realizar este tipo de desincorporación, se determina la cantidad de cada libro, luego se seleccionan los libros con mayor cantidad de ejemplares y poca demanda, después se realiza un listado por Escuela de libros a donar y se entrega el listado en la Dirección de la Escuela, donde se espera la respuesta de aprobación para donar el libro. Si la respuesta es positiva, se donan algunos ejemplares y se registra la desincorporación en la carpeta de libros existentes y luego en *Microsoft Excel*; en caso contrario, se dejan los libros en la Bolsa del Libro.

■ Generación de Listados

La Bolsa del Libro maneja varios tipos de listados como lo son: el listado de libros existentes por título, el listado de deudores, el listado de inventario y el listado de libros prestados a otros usuarios.

Listado de Libros existentes por título: Este listado se maneja por Escuela, donde se busca el título de los libros que existen dentro de la Bolsa del Libro, además se registran nuevas adquisiciones de ejemplares. Para realizar un listado de este tipo se revisa la lista de inventario y se realiza una sumatoria de los títulos de interés. En la figura 2.2 se muestra un ejemplo del Listado de Libros existentes por título.

UNIVERSIDAD CENTRAL DE VENEZUELA
 FACULTAD DE CIENCIAS
 BOLSA DEL LIBRO ISBN 6N-205-N/0-9

HOJAS DE REGISTRO BIBLIOGRÁFICO
 EDITORIAL: PEARSON - AVENI
 CIUDAD: MADRID - ESPAÑA

AUTOR: STALLINGS WILLIAM
 TÍTULO: COMUNICACIONES Y REDES DE COMPUTADORES

Fecha de adquisición	Cota	Edición	Imp / Reimp/ Año	Valor Bs.	Librería	Donación / Compra	Ref. Factura	Observaciones
09 NOV 05	CE.39.1	7 ^{ma}	---	106.000	Librería CA	Donado por - Book City - libros de texto		
09 NOV 05	CE.39.2	7 ^{ma}	---	106.000		COMPRA	00994	
09 NOV 05	CE.39.3	7 ^{ma}	---	106.000		COMPRA	00994	
09 NOV 05	CE.39.4	7 ^{ma}	---	106.000		COMPRA	00994	

Figura 2.2: Ejemplo de Listado de Libros existentes por título

Listado de Deudores: En esta lista se registran aquellos estudiantes que se encuentran deudores con la Bolsa del Libro. A partir de esta lista se generan reportes semestralmente. Para realizar un listado de este tipo, se debe entrar a *Microsoft Access*, generar un reporte de estudiantes deudores y unificar el reporte con la lista de deudores de semestres anteriores. Luego el listado es pasado a *Microsoft Excel* y posteriormente se entrega a la División de Control de Estudios en digital y en papel. En la figura 2.3 se muestra un ejemplo del Listado de Deudores.

COMPUTACIÓN

10504405	BARBOZA G RUTH N	C5.10.17
19734434	CASTILLO BRICEÑO ILVANY ALEJANDRA	C6.39.16
15440826	GÓMEZ C RICHARD A	M1.26.17
11566123	GONZALEZ C JUAN G	C9.17.2
13952941	GRIJALVO BLANCO ADRIAN JOSE	C5.12.8 / C2.35.2
14049146	LÓPEZ TOUSSAINT JOHANNA	M2.36.13
17384974	ORDAZ ABREU ANDRES EDUARDO	C4.22.23

ESTUDIANTES MULTADOS PARA EL SEM II-10

		MONTO (Bs)
13845233	AVILA V GUSTAVO	5.000
16700732	DELGADO KEVIN	12.000
18713368	NARANJO C. NESTOR MANUEL	6.000
16869738	OVALLES M GILBERTO	9.000

Figura 2.3: Ejemplo de Listado de Deudores

Listado de Inventario: Para realizar un listado de este tipo, luego de finalizado el semestre, se hace un conteo de los libros que existen en físico en la Bolsa del Libro, para luego publicar la lista de libros en alquiler. Este resultado es comparado con la lista de deudores y la lista de libros prestados a otros usuarios, y debe coincidir con la lista de libros existentes para así tener el inventario completo. Posteriormente se cargan los títulos de los libros en el listado a publicar y se publica la lista de libros de alquiler en la cartelera. En la figura 2.4 se muestra un ejemplo del Listado de Inventario.

Cota	Autor	Título	Edición Cent
C2.20.0	SANCHIS-MORALES	PROGRAMACIÓN CON EL LENGUAJE PASCAL	1
C2.22.0	TREMBLAY-BUNT	PASCAL ESTRUCTURADO	1
C2.23.0	AGUILAR LUIS	PROGRAMACIÓN EN TURBO PASCAL	1

Figura 2.4: Ejemplo de Listado de Inventario

Listado de Libros prestados a otros usuarios: En esta lista se refleja el personal administrativo que se encuentra deudor de libros con la Bolsa del Libro. Para realizar un listado de este tipo se busca el archivo de *Microsoft Excel* correspondiente, se visualiza el listado de usuarios con libros prestados y se hace un filtro avanzado para generar el listado de deudores. En la figura 2.5 se muestra un ejemplo del Listado de Libros prestados a otros usuarios.

	A	B	C	D	E	F	G	H	I	J	K
1	APellidos y Nombres	CEDUL	DEPENDENC	PERSONA	TELEFON	COTA	AUTOR	TITULO	EDICIC	F. Ref.	F. Dev.
2	Alvarez, Marisela	5.564.211	Biblioteca AG	Administrativo	4155544	B7.26.4	Trudy, Mc Kee	Bioquímica		18/04/2006	29/11/2006
3	Alvarez, M Adonahis Arlette	9.958.083	Computacion		6051061	M5.12.2	Lehmann, Charles	Geometria Analitica		30/01/2006	
4	Alvarez, Juan Francisco	9.907.516	Quimica	Docent-Postg	0416-4092131	Q4.45.7	H. B Callen	Termodinamica	1ra	19/10/2006	10/04/2007
5	Arteaga R. Carlos	12.394.479	Matemáticas		6621896	M7.4.1	Seymour, Lipschutz	Topologia General		10/11/2004	20/11/2006
6	Arteaga R. Carlos	12.394.479	Matemáticas		6621896	M7.8.1	Kelly John	Topologia General		10/11/2004	20/11/2006
7	Arteaga R. Carlos	12.394.479	Matemáticas		6621896	C4.26.1	Mendenhall	Probabilidad y Estadística		10/11/2004	20/11/2006
8	Arteaga R. Carlos	12.394.479	Matemáticas		6621896	C4.2.15	Spiegel Murray	Estadística		10/11/2004	20/11/2006
9	Bianco Manuel	14.564.799	Quimica	Proyecto Amaz	0416-8158182	Q3.50.10	Marcano / Cortes	Fundamentos de Quím Anal		06/07/2010	
10	Bramble M. Evelyn	18.961.900	Letras	PCI	3214832	B1.12.2	Solomons	Biología		09/11/2006	27/02/2007
11	Bramble M. Evelyn	18.961.900	Letras	PCI	3214832	B1.9.6	Curtis, Helena	Biología		15/12/2006	23/02/2007
12	Brito A. Jesús A.	5.695.281	Control de Estuc	Administrativo	0414-2776272	Q3.14.1	Rakoff	Química Orgánica Fundam		17/03/2010	
13	Brito A. Jesús A.	5.695.281	Control de Estuc	Administrativo	0414-2776272	Q3.18.37	Shriner / Fuson	Identificación de Comp		17/03/2010	

Figura 2.5: Ejemplo de Listado de Libros prestados a otros usuarios

2.2. Sistemas anteriores

Esta parte del capítulo se enfoca en describir a nivel general los sistemas anteriores que se han propuesto para la automatización de la Bolsa del Libro de la Facultad de Ciencias de la UCV, que forman parte de Trabajos Especiales de Grado para optar a la *Licenciatura en Computación*, tomando en cuenta sus principales características.

2.2.1. Sistema de Información de la Bolsa del Libro de la Facultad de Ciencias (UCV)

Fue un Sistema de información diseñado por Marcela Tuati y Francisco García [17] en Abril de 1998 para el control administrativo de la Bolsa del Libro, sustentado en las filosofías y tecnologías de Reingeniería de Procesos y Automatización de Flujos de Negocios como *Workflow*, con un enfoque Orientado a Objetos en un ambiente Cliente/Servidor.

Características

- Para el esbozo de las pantallas del sistema se tomó como base el diseño de las aplicaciones en Windows 95 en cuanto a barras de desplazamiento, barras de menús, entre otros.

En una entrevista realizada a la Licenciada Marlene Carrizalez, indicó que este Sistema de información para el control administrativo de la Bolsa del Libro, nunca llegó a ser implantado, desconociendo las razones exactas.

2.2.2. Desarrollo de un Sistema de Software Orientado a Objetos para una Bolsa del Libro

Fue un Sistema de información interactivo diseñado por Lewis Hernández [15] en Octubre de 1998, para la administración, el control y el mantenimiento de la información asociada con los libros y los usuarios del servicio de la Bolsa del Libro.

Características

- Se utilizaron los métodos de desarrollo de Sistemas Orientados a Objetos *Object Oriented Software Engineering* (OOSE) y *Object Modeling Technique* (OMT).
- La aplicación se implementó mediante la utilización de *Visual Basic* 4.0 bajo el ambiente de Windows.

Según una entrevista realizada a la Licenciada Marlene Carrizalez, indicó que este sistema automatizado logró implantarse correctamente en la Bolsa del Libro, durando aproximadamente tres semestres en funcionamiento, pero luego el sistema presentaba errores constantemente, por lo cual no pudo seguir en funcionamiento.

2.2.3. Desarrollo de Aplicaciones de Comercio Electrónico usando Software Libre. Caso de estudio: Bolsa del Libro de la Facultad de Ciencias de la UCV

Fue una Aplicación de Comercio Electrónico diseñada por María Araujo y Cidalía Pereira [16] en Septiembre de 2007, creada para responder a los requerimientos de administración de la Bolsa del Libro y automatizar el proceso del alquiler y compra de libros permitiendo el pago en línea del servicio a los estudiantes de la Facultad de Ciencias.

Características

- Se utilizó la Metodología *eXtreme Programming* (XP) y Modelación Ágil.
- Se utilizaron las siguientes herramientas de Software Libre: PHP, Apache y MySQL.

En una entrevista realizada a la Licenciada Marlene Carrizalez, señaló que para esta aplicación de comercio electrónico se realizaron todos los trámites y convenios necesarios con el Banco de Venezuela para poder llevar a cabo el pago en línea, pero no se pudo contar con un servidor. En la presentación pública del Trabajo Especial de Grado, se presentaron ciertos errores, por lo cual sus desarrolladoras se comprometieron a solventar las fallas para dejar el sistema operativo en la Bolsa del Libro, pero posteriormente no se pudo implantar el sistema, ya que las mismas perdieron el contacto con la UCV una vez graduadas.

Capítulo 3

Marco Tecnológico

En este capítulo se describen las herramientas tecnológicas utilizadas en el desarrollo de la aplicación Web, tales como Ruby, Rails, REST, MySQL, Subversion y AJAX; y además se hace una breve descripción de la metodología de desarrollo de software usada (XP).

3.1. Ruby On Rails

Ruby es un lenguaje de programación dinámico, que posee una librería base de clases con una rica API. Se inspira en otros lenguajes de programación como Lisp, Smalltalk y Perl, pero usa una sintaxis que puede ser entendible por los programadores de C y Java [11].

El lenguaje fue creado en Japón por Yukihiro Matsumoto, quien empezó a trabajar en Ruby el 24 de Febrero de 1993, y lo presentó al público en el año 1995 [22].

3.1.1. Ventajas del lenguaje Ruby

Algunas características particulares del lenguaje Ruby, basadas en [11], que indican sus ventajas y beneficios son:

- Es un lenguaje orientado a objetos, moderno, flexible y productivo.
- Es altamente extensible.
- Es multiplataforma.
- Se integra a la perfección con las últimas tecnologías como Sistemas Manejadores de Bases de Datos (MySQL, PostgreSQL, SQLite, Oracle, SQL Server y DB2), con XML y HTML.
- Es software libre.

3.1.2. Framework Rails

Rails es un framework de aplicaciones Web de código abierto escrito en el lenguaje de programación Ruby, siguiendo el paradigma de la arquitectura Modelo Vista Controlador (MVC). Trata de combinar la simplicidad con la posibilidad de desarrollar aplicaciones del mundo real escribiendo menos código que con otros frameworks y con un mínimo de configuración. Se distribuye a través de RubyGems, que es el formato oficial de paquete (gemas) y el canal de distribución de librerías y aplicaciones Ruby. Esta información de Rails y la siguiente, está basada en [9].

Fue creado por David Heinemeier Hansson, empleado de la empresa 37 signals, mientras trabajaba en una herramienta de gestión de proyectos llamada Basecamp, siendo presentado por primera vez al público en Julio de 2004. La versión 1.0 fue liberada a principios del año 2006.

Ser ágil es una de las premisas de Rails, es una tecnología que está siempre evolucionando hacia el desarrollo de aplicaciones cada vez más robustas pero manteniendo siempre la facilidad que lo caracteriza y desde su concepción, ha sido apoyado por un creciente número de desarrolladores hasta convertirse en un framework de programación Web sólido, escalable, bien documentado, preparado para el entorno empresarial y con una concienciación sobre la seguridad.

3.2. REST

La Transferencia de Estado Representacional (*Representational State Transfer*) o REST, es una técnica para el diseño de Servicios Web, para las aplicaciones que solicitan y manipulan recursos en la Web utilizando los métodos estándar de HTTP: GET, POST, PUT y DELETE [14]. Este nuevo estilo ha supuesto una nueva opción de estilo de uso de los Servicios Web.

3.2.1. Servicios Web de Amazon

El *Amazon Web Services* (AWS) es un conjunto de servicios informáticos a distancia (también llamados Servicios Web), que ofrece Amazon.com a través de Internet. Entre los servicios ofrecidos por *Amazon Web Services* se encuentra *Amazon E-Commerce Service* (ECS) [5].

Se utilizó el Servicio Web *E-Commerce de Amazon* (ECS) para realizar la descarga desde la Aplicación Web de los libros disponibles en el catálogo de libros de Amazon, haciendo uso de un buscador donde se introduce el código ISBN del libro, que corresponde al código ASIN (Número de Identificación Estándar de Amazon) del mismo. El ECS es una API que permitió tener acceso a los datos de Amazon y se accedió a través del protocolo REST [6].

3.3. MySQL

MySQL es un Sistema Manejador de Base de Datos Relacional (SMBDR), con la fuente de código abierto más popular, que se desarrolla y distribuye con el apoyo de la corporación Oracle. La información de esta sección y de la siguiente, está basada en [2].

3.3.1. Ventajas

Las principales ventajas de MySQL se describen a continuación:

- Es muy rápido al momento de resolver las consultas solicitadas por los usuarios.
- Posee un buen enfoque en cuanto a seguridad.
- Tiene soporte multi-usuario y multi-hilo.
- Ofrece licenciamiento.
- Es multiplataforma.

3.4. Subversion

Subversion es un sistema de control de versiones libre y de código fuente abierto.

Maneja archivos y directorios a través del tiempo. El repositorio es como un servidor de archivos ordinario, excepto porque recuerda todos los cambios hechos a los archivos y directorios. Esto permite recuperar versiones antiguas de los datos, o examinar el historial de cambios de los mismos. Subversion puede acceder al repositorio a través de redes, lo que le permite ser usado por personas que se encuentran en distintos computadores, fomentando la colaboración. La información de esta sección y de la siguiente, está basada en [21].

3.4.1. Características básicas

A continuación se describen las características de Subversion:

- Implementa un versionado de directorios.
- Posee un historial de versiones.
- Realiza envíos atómicos.
- Crea un versionado de metadatos.
- Efectúa una manipulación consistente de datos.

3.5. AJAX

AJAX (*Asynchronous JavaScript And XML*) es una tecnología de desarrollo Web para crear aplicaciones interactivas mediante la combinación de tres tecnologías ya existentes: HTML, DOM y XML. La información de esta sección y la siguiente, está basada en [10].

3.5.1. Ventajas

Existen diversas razones para utilizar AJAX, entre las cuales se destacan:

- Está basado en estándares abiertos.
- Proporciona usabilidad en la páginas web.
- Es válido en cualquier plataforma y navegador.
- Está adoptado por los grandes de la tecnología Web.
- Es independiente del lenguaje de programación Web que se utilice.

3.6. Programación Extrema (XP)

La programación extrema es una metodología ágil para el desarrollo de proyectos informáticos, que trata de dar solución a los problemas de la ingeniería del software. Los objetivos de XP son: la satisfacción del cliente, tratando de dar al cliente el software que él necesita y cuando lo necesita y potenciar al máximo el trabajo en grupo. La información de esta sección y las siguientes, está basada en [13].

3.6.1. Prácticas

Las doce prácticas de la programación extrema tienen su origen en prácticas bien conocidas en la ingeniería del software:

- **El juego de la Planificación:** Poner en producción las características más importantes.
- **Versiones Pequeñas:** Periódicamente, se producen nuevas versiones del sistema.
- **Metáfora del Sistema:** Descripción general del sistema.
- **Diseño Simple:** El sistema se diseña con la máxima simplicidad posible.
- **Pruebas Continuas:** Los clientes especifican pruebas funcionales.
- **Refactorización:** Modificar la forma del código sin cambiar su funcionamiento.

- **Programación por parejas:** La programación se realizan de a dos programadores por computadora.
- **Posesión Colectiva del Código:** Cualquier programador puede cambiar cualquier parte del sistema en cualquier momento.
- **Integración continua:** Los cambios se integran en el código base varias veces por día.
- **Semana laboral de 40 horas:** Cada trabajador trabaja no más de 40 horas por semana.
- **Cliente en el Sitio:** El equipo de desarrollo tiene acceso todo el tiempo al cliente.
- **Estándares de Codificación:** Todo el código debe estar escrito de acuerdo a un estándar de codificación.

3.6.2. Valores

La programación extrema se apoya en cuatro valores fundamentales:

- **Comunicación:** Para ser efectiva, debe involucrar a todos los participantes en el proyecto.
- **Simplicidad:** Se simplifica el diseño para agilizar el desarrollo y facilitar el mantenimiento.
- **Retroalimentación:** El cliente debe estar integrado en el proyecto.
- **Coraje:** Para trabajar correctamente con metodologías ágiles se requiere de mucho coraje.

3.6.3. Actividades

A continuación se describen las actividades de XP. La información de esta sección está basada en [20].

- **Planificación:** Se definen entre el conjunto de desarrolladores y usuarios del sistema una serie de Historias de Usuario que describen las funcionalidades requeridas para el software que se construirá. El desarrollo se divide en iteraciones, cada iteración comienza con un plan para que se elijan las Historias de Usuario a desarrollar.
- **Diseño:** Los diseños deben ser sencillos, si alguna parte del sistema es de desarrollo complejo, lo apropiado es dividirla en varias. Si hay fallos en el diseño o malos diseños, estos deben ser corregidos cuanto antes.

- **Codificación:** XP argumenta que lo más importante para el desarrollo del producto de software es codificar. Codificar en XP es hacer diagramas que generaran código, scripts para una Aplicación Web o código para una Aplicación que necesita compilarse.
- **Pruebas:** Hay que asegurarse de que todo lo que se hace funcione correctamente. Para ello, lo mejor es desarrollar las pruebas desde el momento que se conocen las Historias del Usuario.

Capítulo 4

Marco Aplicativo

En este capítulo se describen los aspectos más importantes del proceso de desarrollo de la Aplicación Web de la Bolsa del Libro, destacando los actores y responsabilidades, la metáfora del sistema y la adaptación de las fases de XP.

4.1. Adaptación de la Metodología XP

A continuación se describen los aspectos relacionados a la adaptación del proceso XP que se realizó durante el desarrollo de la Aplicación Web de la Bolsa del Libro de la Facultad de Ciencias. La información de esta sección está basada en [13].

4.1.1. Actores y Responsabilidades

Los actores son todas las personas involucradas en el desarrollo del proyecto, los cuales a su vez cumplen distintos roles o responsabilidades según su importancia y nivel de participación. A continuación se destacan los roles existentes en el presente proceso de desarrollo:

- **Cliente:** Determina la funcionalidad que se pretende alcanzar en cada iteración y define las prioridades de implementación según el valor de negocio que aporta cada historia.
- **Programador:** Es responsable de implementar las historias solicitadas por el cliente. Además, estima el tiempo de desarrollo de cada historia para que el cliente pueda asignarle prioridad dentro de alguna iteración.
- **Seguidor:** Su función se centra en realizar las pruebas de integración al sistema del código provisto por los programadores y de verificar el correcto funcionamiento de la aplicación.

- **Gerente:** Es responsable del proceso general. Se encarga de apoyar y de guiar a las personas del equipo en poner en marcha las 12 prácticas.

En el cuadro 4.1 se muestra el esquema de actores y los roles que representan.

	Cliente	Programador	Seguidor	Gerente
Joselyn Oviedo		X	X	
Andreina Jiménez		X	X	
Jaime Parada	X			X
Marlene Carrizalez	X			

Cuadro 4.1: Esquema de actores y roles que desempeñan

4.1.2. Metáfora del Sistema

Con el propósito de poder brindarles a los usuarios de la Bolsa del Libro (Personal Administrativo y Estudiantes) la posibilidad de administrar los distintos procesos de préstamo rental de libros y realizar acciones con la Bolsa del Libro desde Internet, se pone a su disposición el Sistema Automatizado de la Bolsa del Libro.

Principalmente se permite a los Estudiantes realizar actividades como: Pre-Solicitud de Alquileres, Solicitud de Alquileres y Solvencias y Consulta del Catálogo de Libros. Adicionalmente, el Personal Administrativo puede realizar: Administración de Pre-Alquileres y Alquileres, Administración de Solvencias, Administración de Libros y Ejemplares, Generación de Listados y Manejo de Configuraciones, Auditorias, Actualizaciones, Periodos Lectivos, Usuarios y Carga Masiva.

La metáfora de la Aplicación Web de la Bolsa del Libro se muestra en la figura 4.1.

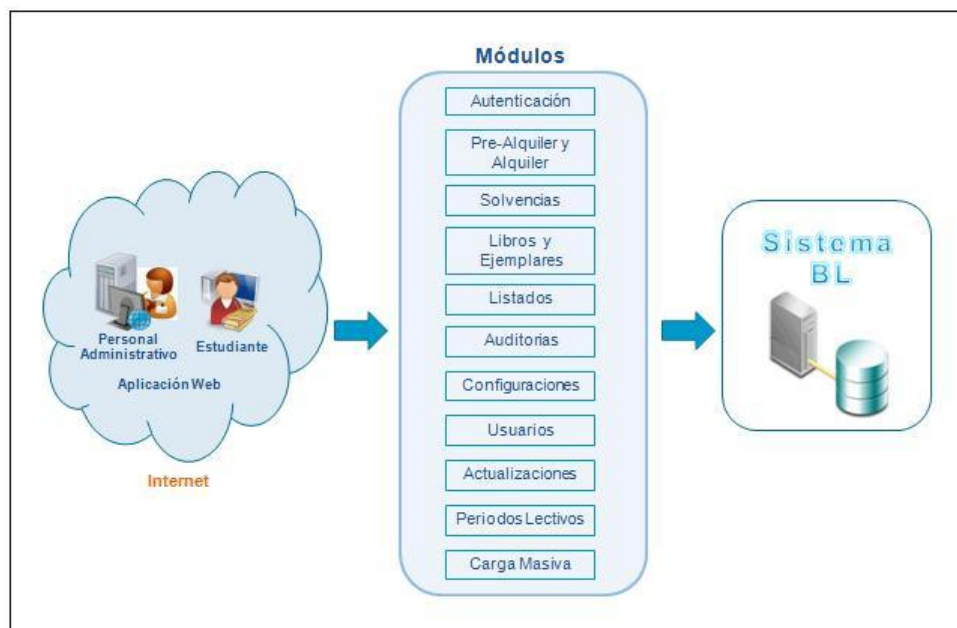


Figura 4.1: Metáfora del Sistema

4.1.3. Adaptación de las fases de XP

La metodología XP está compuesta por cuatro fases fundamentales las cuales fueron tomadas en cuenta para el desarrollo de la Aplicación Web para la Bolsa del Libro. A continuación se describe su adaptación:

■ Exploración

En esta fase, se llevaron a cabo varias reuniones y conversaciones con la Administradora de la Bolsa del Libro (Marlene Carrizalez), que permitieron definir una lista de Historias de Usuario, como lo propone la metodología XP, pero en principio esto no fue suficiente para entender la situación actual de la Bolsa del Libro y todos los procesos y servicios que allí se llevan a cabo. Por lo cual, se realizaron Diagramas de Actividades que reflejan los procesos que se realizan a diario dentro de la Bolsa del Libro, como artefactos adicionales para complementar de manera adecuada la metodología XP.

Estos procesos son: El Alquiler de Libros, La Generación de Solvencias, la Adquisición de Libros, la Desincorporación de Libros y la Generación de Listados. A continuación en las Figuras 4.2, 4.3, 4.4, 4.5 y 4.6, se muestran cada uno de ellos con sus Diagramas de Actividades correspondientes.

Proceso de Adquisición de Libros

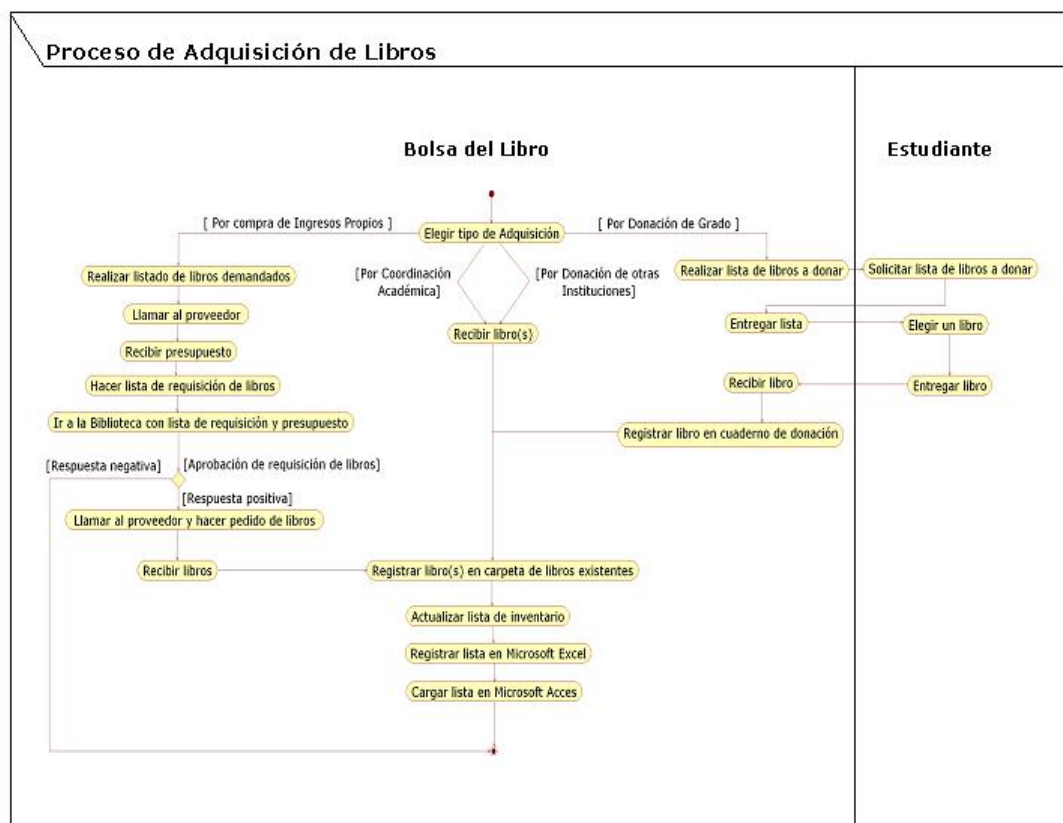


Figura 4.2: Descripción del Proceso de Adquisición de Libros

Proceso de Alquiler de Libros

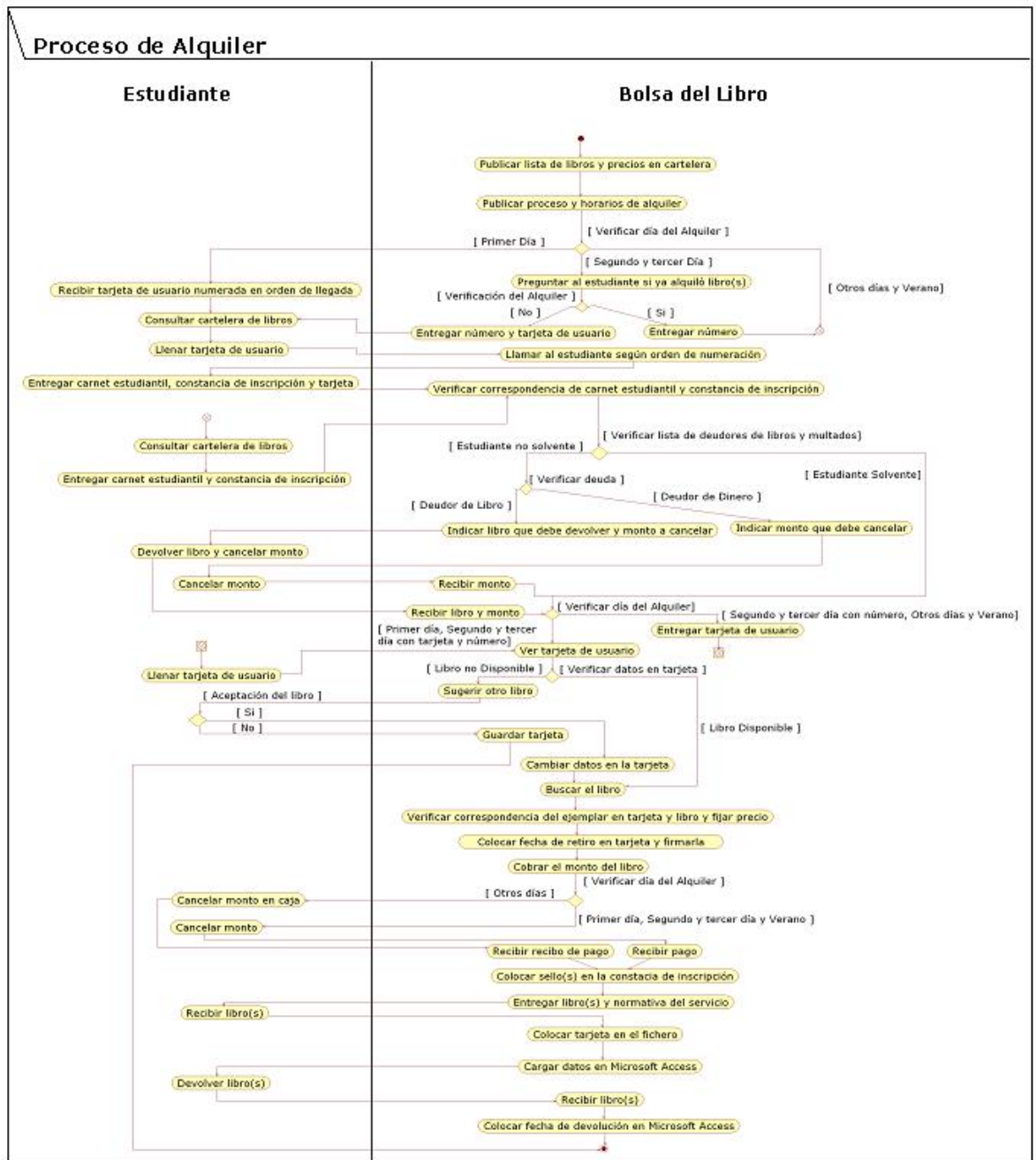


Figura 4.3: Descripción del Proceso de Alquiler de Libros

Proceso de Solvencias

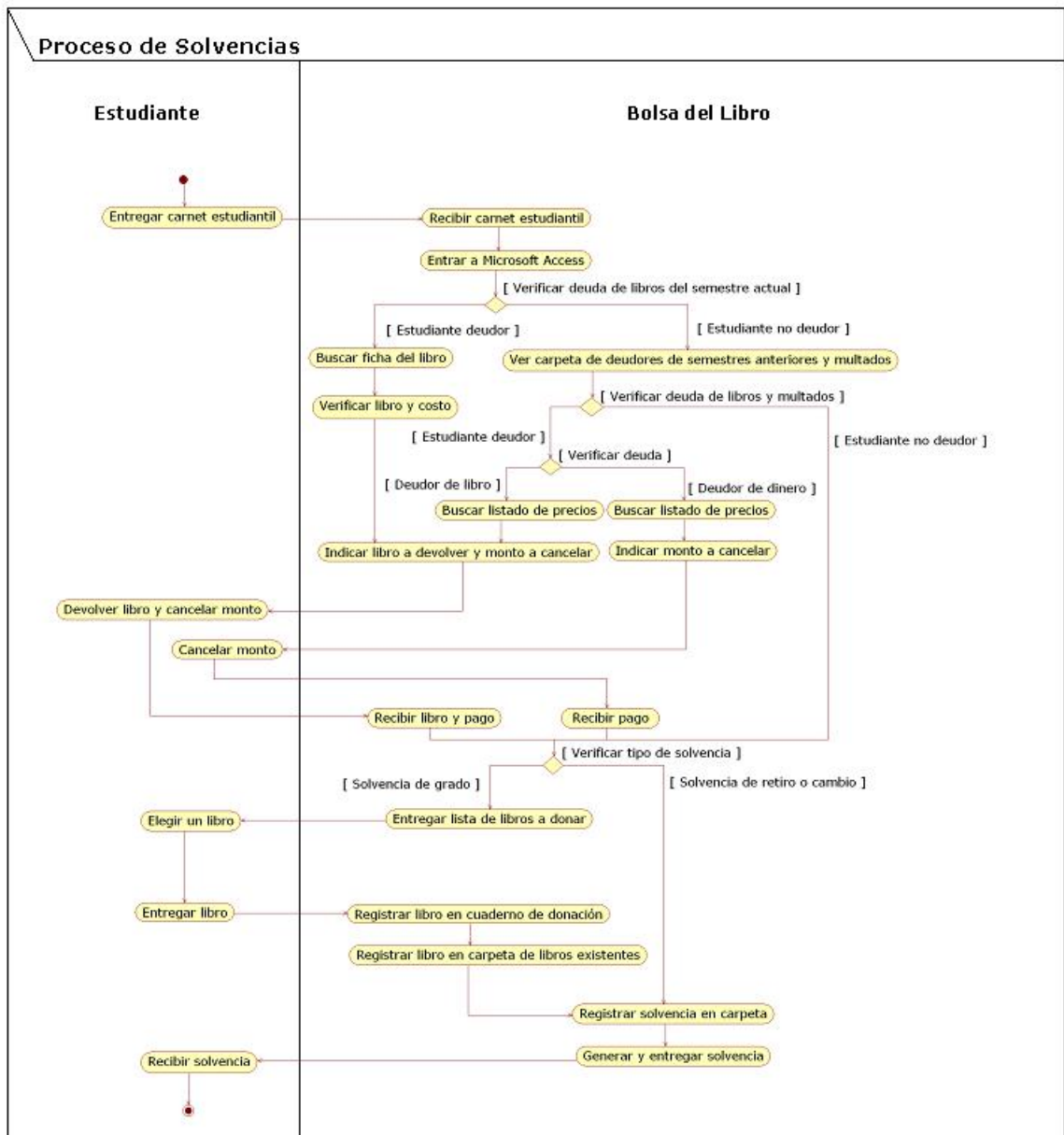


Figura 4.4: Descripción del Proceso de Solvencias

Proceso de Desincorporación de Libros

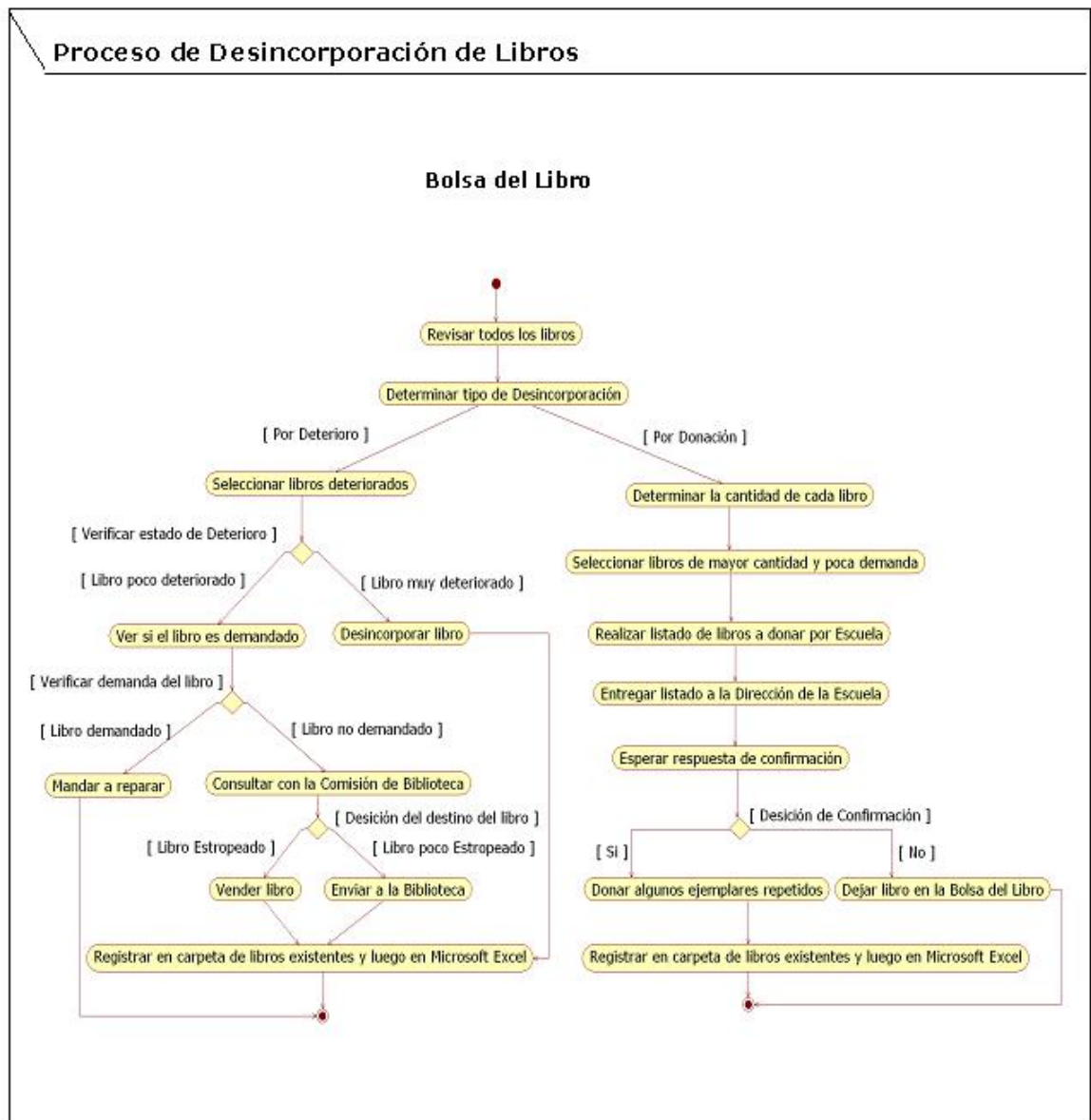


Figura 4.5: Descripción del Proceso de Desincorporación de Libros

Generación de Listados

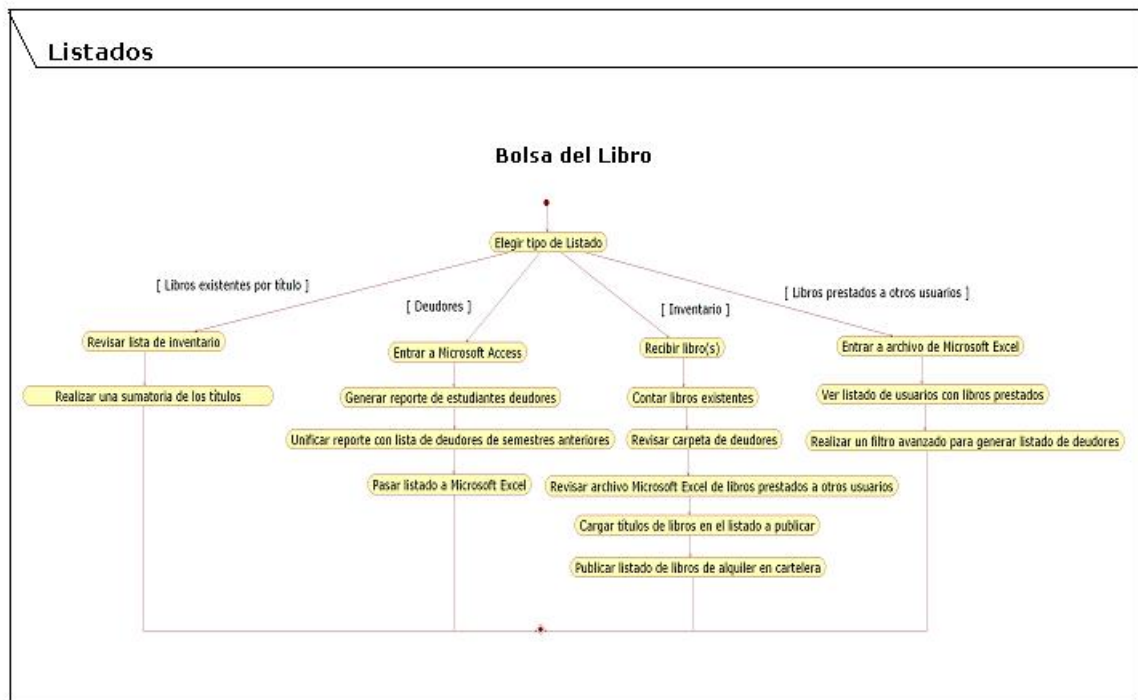


Figura 4.6: Listados

Es importante destacar, que el formato utilizado para cada Historia de Usuario es el siguiente: un número que sirve de identificador, un nombre, una prioridad (alta, media o baja) una estimación del tiempo y una breve descripción sobre la Historia de Usuario, dicho formato se muestra a continuación en el cuadro 4.2:

Número: –	Nombre: –
Prioridad: –	Tiempo Estimado: –
Decripción: –	

Cuadro 4.2: Formato de registro para una Historia de Usuario

Las Historias de Usuario levantadas con el cliente fueron 47 (cuarenta y siete) en total, y cada una de ellas se muestra a detalle en la sección de Apéndice.

■ Planificación

En esta fase, las programadoras nos reunimos con la Administradora de la Bolsa del Libro (Marlene Carrizalez) para establecer la prioridad de desarrollo de cada historia.

El desarrollo de estas historias se dividió en iteraciones, cada iteración comenzó con un plan donde se eligieron las Historias de Usuario a desarrollar.

A cada iteración se le estableció una fecha de inicio y una fecha de culminación por parte del equipo de desarrollo.

Durante el desarrollo de las iteraciones surgieron cambios, que fueron adaptados o mejorados rápidamente sin que se afectara la planificación.

En el cuadro 4.3 se muestra el esquema que se utilizó al inicio de cada iteración, el cual contiene el número de la iteración, una descripción, el número y nombre de cada historia, el tiempo estimado y las fechas de inicio y fin de la iteración.

Número de la Iteración	
Descripción	
Historias de Usuario	
Tiempo Estimado	
Fecha Inicio - Fin	

Cuadro 4.3: Esquema de planificación de cada iteración

El proyecto y su desarrollo, comprende un conjunto de 11 (once) iteraciones. El periodo para llevar a cabo dicho proyecto y su documentación correspondiente fue desde el 01 de Diciembre de 2010 hasta el 15 de Julio de 2011.

Es importante mencionar, que a lo largo del desarrollo del proyecto, las programadoras tuvimos una serie de entrevistas con el cliente (Administradora de la Bolsa del Libro), para recaudar toda la información necesaria de la Bolsa del Libro, cada una de estas entrevistas está plasmada a detalle en la sección de Apéndice.

■ Iteraciones

En esta fase, primeramente se realizó el diseño de la Base de Datos de la Bolsa del Libro, y posteriormente su implementación, para la cual se utilizó el Sistema Manejador de Base de Datos MySQL versión 2.8.1.

La estructura de la Base de Datos está compuesta por 24 (veinticuatro) tablas denominadas: Alquileres, Auditorias, Auditorias_Correos, Actualizaciones, Contactos, Configuraciones, Condiciones, Detallealquileres, Detalle_prealquileres, Ejemplares, Escuelas, Libros, Prealquileres, Permisos, Periodo_lectivos, Status, Schema_migrations, Simple_captcha_data, Solvencias, Roles, Rolespermisos, Usuarios, Usuarios-roles y Ubicaciones.

A continuación se muestran las tablas en detalle en la figura 4.7:

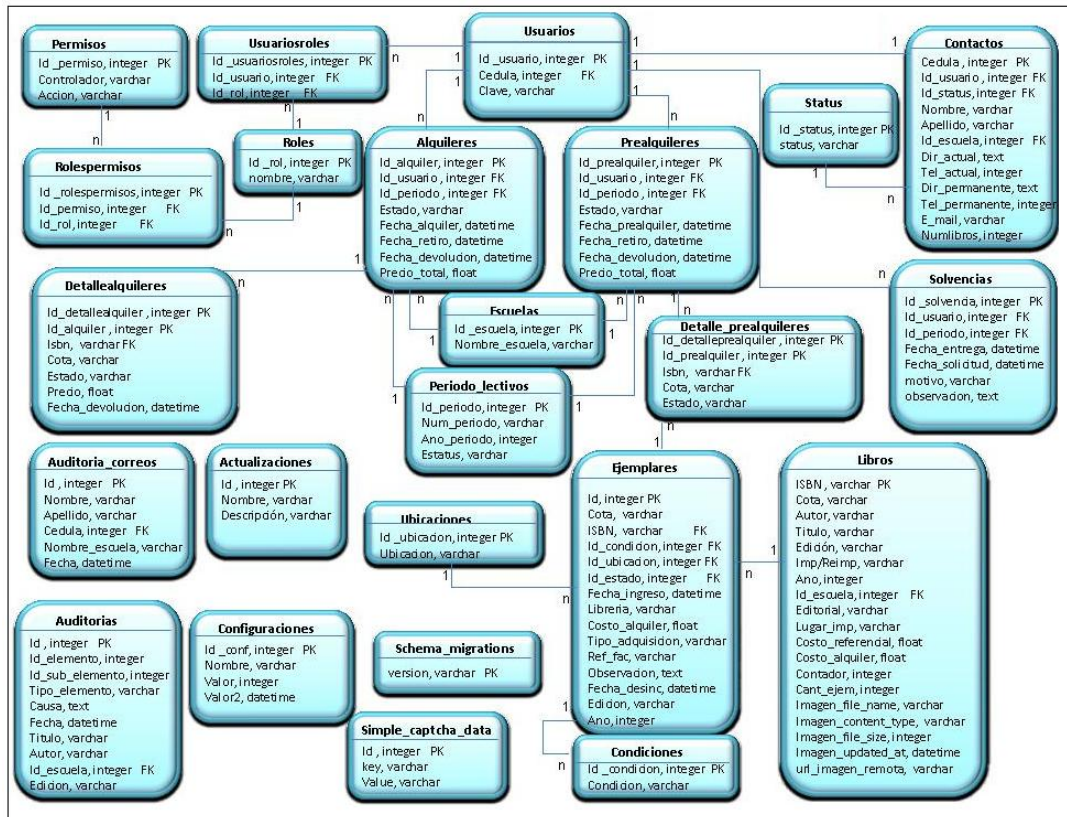


Figura 4.7: Modelo de Datos

Luego, por cada iteración se realizaron prototipos de interfaces sencillas que definieron las funcionalidades de cada Módulo de la Aplicación Web, para tener una visión clara de lo que el cliente esperaba. Estos prototipos de interfaces de la Aplicación Web se mostraron al cliente manteniendo conversaciones con él, que permitieron determinar algunos cambios en las mismas.

Para desarrollar el código de las principales Historias de Usuario se adoptó la programación en pareja, siempre trabajando en una misma iteración. De cada iteración se asignaron diferentes tareas por Historias de Usuario a cada programadora, realizando frecuentes integraciones de los códigos creados por cada una de las programadoras, a fin de no ocasionar problemas de compatibilidad, ni de interfaz. La integración del código se realizó de forma continua a través del sistema de control de versiones (Subversion).

Además se efectuaron **Pruebas de Aceptación** al final de cada iteración, las cuales fueron especificadas por el cliente y se enfocaron en las características generales y la funcionalidad del sistema. En estas pruebas el cliente verificó el adecuado funcionamiento, a fin de comprobar que sus requerimientos fueron cumplidos satisfactoriamente. Ver cuadro 4.4:

No. Caso Prueba	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido	Motivo de la Falla
-	-	-	-	-	-

Cuadro 4.4: Formato de registro de Prueba de Aceptación

A continuación, se muestran cada una de las iteraciones a detalle.

Iteración 0: Módulo de Pre-Alquiler y Alquiler

A continuación en el cuadro 4.5 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Pre-Alquiler y Alquiler.

Módulo(s)	Modelo(s)	Vistas	Controlador(es)
Alquiler	alquilere.rb detallealquilere.rb	buscar.rjs edit.html.erb index.html.erb new.html.erb show.html.erb mensajes.html.erb	alquileres_controller.rb detallealquileres_controller
Pre-Alquiler	prealquilere.rb detalle_prealquilere.rb	buscar.rjs mensajes.html.erb edit.html.erb index.html.erb new.html.erb show.html.erb	prealquileres.rb detalle_prealquileres.rb

Cuadro 4.5: Esquema del Módulo de Pre-Alquiler y Alquiler con su MVC asociado

La planificación de las Historias de Usuario para esta iteración fue organizada de la siguiente manera:

Iteración 0	
Descripción	Creación de la interfaz de usuario general y el Módulo de Pre-Alquiler y Alquiler.
Historias de Usuario	<ol style="list-style-type: none"> 1. Visualizar la Aplicación Web. 2. Visualizar los Pre-Alquileres y Alquileres en el sistema. 3. Buscar los libros que el estudiante desea pre-alquilar y alquilar. 4. Seleccionar los libros a pre-alquilar y alquilar. 5. Guardar un alquiler en el módulo de Alquiler. 6. Descargar el comprobante de alquiler en formato PDF. 7. Visualizar el estado de solvencia de los estudiantes en el módulo de Pre-Alquiler y Alquiler. 8. Registrar la devolución de un libro en el módulo de Alquiler. 9. Visualizar los diferentes estados de los usuarios.
Tiempo Estimado	26 días
Fecha Inicio - Fin	01/12/2010 - 10/01/2011

El desarrollo de las vistas asociadas consistió en la agregación de todas las interfaces necesarias para visualizar las acciones que pueden ser realizadas por los estudiantes y por el personal administrativo de la Bolsa del Libro. El diseño de las interfaces de todos los módulos está basado en los lineamientos establecidos por Shneiderman [8].

Para que los estudiantes realicen el pre-alquiler y el alquiler de los libros, se creó una vista donde aparecen todos los libros disponibles en la Bolsa del Libro, de manera que el estudiante puede seleccionar los libros que desee pre-alquilar o alquilar según sea el caso.

Para la devolución de los ejemplares alquilados, se diseñó una vista en la cual el personal administrativo puede observar los libros que tiene alquilados un estudiante en particular. En dicha vista también se puede indicar la devolución de los ejemplares haciendo uso de los enlaces *Devolver Ejemplar*, *Pagar ejemplar* y *Devolver y Pagar Ejemplar*.

En la figura 4.8 se observa la Vista Inicial del Módulo de Pre-Alquiler y Alquiler desde el perfil de Estudiante, donde se destacan el buscador de libros (para que el estudiante encuentre con facilidad el libro que requiere) y los libros disponibles para realizar el alquiler.

Home -- Estudiante Cerrar Sesión

Alquiler
Catálogo de Libros
Solventas

Alquiler de Libros

Buscar por: Título

	ISBN	COTA	TITULO	AUTOR	COSTO DE ALQUILER
<input type="checkbox"/>	0582306574	12.90.0	Friends Starter Global Activity Book (FRND)	Olivia Date	21.0
<input type="checkbox"/>	970306111X	Q2.90.0	Química - 9ª Edición	RAYMOND CHANG	45.8
<input type="checkbox"/>	9780596536178	c2.89.0	The Ruby Programming Language	David FlanaganYukhiro Matsumoto	30.2

Figura 4.8: Vista Inicial del Módulo de Alquiler desde el perfil de Estudiante

En el caso de que el estudiante sea deudor con la Bolsa del Libro, es decir, que devolvió el libro pasada la fecha de devolución de libros establecida, el estudiante no puede realizar un nuevo alquiler, hasta que no cancele su deuda con la Bolsa del Libro. Además si el estudiante es deudor, eso también lo convierte en estudiante multado, debiendo cancelar el cien por ciento del costo del alquiler del libro. En este caso el sistema genera un error, indicando al usuario que no puede realizar un alquiler debido a su estado. Ver figura 4.9



Figura 4.9: Vista de Error generado en el Módulo de Alquiler desde el perfil de Estudiante cuando el Estudiante es deudor

Además, el estudiante puede descargar su comprobante de alquiler, después de realizar el alquiler respectivo. Cabe destacar que en relación al formato PDF del comprobante de alquiler se diseñó un encabezado el cual es común para el resto de los comprobantes. Ver figura 4.10



Figura 4.10: Vista del Documento PDF del Comprobante de Alquiler

De manera simultánea, por cada alquiler realizado por algún estudiante, las solicitudes de alquileres se muestran en una vista correspondiente al Personal Administrativo. Ver figura 4.11

Universidad Central de Venezuela
Facultad de Ciencias
Biblioteca Alonso Gamero
Bolsa del Libro

Historia | Servicios | Normativas | Horarios

Home -- Administrador Cerrar Sesión

Actualizaciones
Admin.Alquileres
Admin.Libros
Admin.Ejemplares
Admin.Prealquiler
Admin.Solvencias
Admin.Usuarios
Auditorias
Configuraciones
Listados
Periodo Lectivo

Alquileres Realizados

Buscar por: Cedula

« Anterior 1 2 3 4 Siguiente »

ID ALQ	NOMBRE	APELLIDO	CEDULA	PERIODO LECTIVO	ESTADO	FECHA ALQUILER	FECHA RETIRO	PRECIO TOTAL	
16	Joselyn	Oviedo	18110941	I - 2011	devuelto sin multar	15/05/2011	13/05/2011	66.8	Mostrar Editar Eliminar
17	Joselyn	Oviedo	18110941	I - 2011	devuelto y pagado	15/05/2011	14/05/2011	66.8	Mostrar Editar Eliminar
18	Joselyn	Oviedo	18110941	I - 2011	devuelto sin multar	16/05/2011	16/05/2011	66.8	Mostrar Editar Eliminar
19	Andreina	Jimenez	18141182	I - 2011	devuelto y pagado	16/05/2011		66.8	Mostrar Editar Eliminar

« Anterior 1 2 3 4 Siguiente »

Figura 4.11: Vista Inicial del Módulo de Alquiler desde el perfil del Personal Administrativo

También se puede visualizar el detalle del alquiler realizado por un estudiante, a través de la opción *Mostrar*, donde se observan los libros que alquiló el estudiante. Luego de que el estudiante realice la devolución del ejemplar, el Personal Administrativo debe registrar en el sistema dicha devolución. Ver figura 4.12

Datos de el/los Libro(s)

[Devolver Ejemplar](#) | [Pagar Ejemplar](#) | [Devolver y Pagar Ejemplar](#)

Cota	12.90.03
Autor	Olivia Date
Título	Friends Starter Global Activity Book (FRRID)
Edición	Edición No definida
Isbn	0582306574
Costo de alquiler	21.0
Estado	devuelto sin multar
Fecha Alquiler	15/05/2011
Fecha Retiro	13/05/2011
Fecha Devolucion	15/05/2011

Datos de el/los Libro(s)

[Devolver Ejemplar](#) | [Pagar Ejemplar](#) | [Devolver y Pagar Ejemplar](#)

Cota	q2.90.03
Autor	PAYMOND CHANG
Título	Química - 9ª Edición
Edición	Edición No definida

Figura 4.12: Vista desde donde se efectúa la Devolución de libros por parte del Personal Administrativo

Con respecto a la codificación en esta iteración, se realizó un método en el controlador de alquileres para generar el comprobante PDF de alquiler y darle formato al mismo, llamado **comprobante_de_alquiler**, donde primeramente se inicializan las variables **alquilere**, **detalquileres** y **contactoinfo** con los valores necesarios que permiten obtener la información del alquiler y del usuario que realiza el alquiler.


```

tabla.columns["costo_alquiler"] = PDF::SimpleTable::Column.new("costo_alquiler") { |col|
  #col.font_size = 4
  col.width = 80
  col.heading = "COSTO DE ALQUILER"
  col.heading.justification = :center
  col.justification = :center
}

data = []

@detallealquileres.each do |detallealquiler|
  data << {
    "cota" => detallealquiler.cota,
    "titulo" => to_utf16(detallealquiler.libro.titulo),
    "autor" => to_utf16(detallealquiler.libro.autor),
    "costo_alquiler" => detallealquiler.libro.costo_alquiler
  }
end

tabla.data.replace data
tabla.render_on(pdf)

pdf.text "\n\n\n"
pdf.text "<b>\nPRECIO TOTAL: <b> #{@alquilere.precio_total}" + " "*16, :justification => :right, :font_size => 12

pdf.save_as(nombre_completo_archivo_pdf)
f = File.open(nombre_completo_archivo_pdf, 'rb')
contenido = f.read
f.close

send_data(contenido, { :type => "application/pdf", :filename => "alquiler_#{@contactoinfo[:cedula]}_
#{@alquilere.periodo_lectivo.num_periodo}-#{@alquilere.periodo_lectivo.ano_periodo}.pdf"})

end

```

Figura 4.14: Código del método para la generación del comprobante PDF de Alquiler (Parte 2).

Para que el estudiante pueda realizar la búsqueda de los libros a alquilar se creó un método llamado **buscar** en el controlador, donde se reciben los parámetros que el usuario pasa por la vista, los cuales son: el tipo de búsqueda y la búsqueda que desea realizar. Luego, se realiza un ciclo donde se verifican los parámetros y de acuerdo a ellos se muestra el resultado correspondiente.

El código de este buscador en el controlador se muestra en la figura 4.15:

```

def buscar

  @tit = params[:query]
  @tipo = params[:busquedas]
  @tit = @tit.strip unless @tit.nil?
  if @tit==''
    @exito = 0
  else
    if @tipo.to_i == 1
      @campo = "titulo"
    elsif @tipo.to_i == 2
      @campo = "autor"
    elsif @tipo.to_i == 3
      @campo = "cota"
    end
    @hay = Libro.all(:conditions => [{"UPPER(#{@campo}) LIKE ?", "%#{@tit.upcase}%"])
    if @hay.length > 0
      @exito = 1
    else
      @exito = 2
    end
  end
end
end

```

Figura 4.15: Código del método para realizar la búsqueda de los libros a alquilar.

En la vista se utilizó un formulario de ruby para realizar los botones que representan el buscador y las opciones de búsqueda y de acuerdo a lo seleccionado por el usuario se llama al método **buscar** que se encuentra en el controlador, explicado anteriormente. El código de la vista se muestra en la figura 4.16:

```
<div id="baef">
  <%= form_remote_tag :url => url_for(:action => "buscar") do%>
    <b><%= "Buscar por:" %></b>
    <%= select_tag "busquedas",
      " <option value = 1>Titulo</option>
      <option value = 2>Autor</option>
      <option value = 3>Cota </option>" %>

    <%= text_field_tag "query" %>
    <%= submit_tag "Buscar" %>
  <%= end%>
</div>
```

Figura 4.16: Código de la vista Index para realizar la búsqueda de los libros a alquilar.

Además, se realizó una vista JavaScript para verificar el éxito de la búsqueda, si la búsqueda es satisfactoria se realiza de acuerdo a los requerimientos correspondientes, en caso contrario se refleja mediante una ventana un mensaje de alerta. El código de la vista JavaScript se muestra en la figura 4.17:

```
if @exito==1
  if @tipo.to_i == 1
    @campo = "titulo"
  elsif @tipo.to_i == 2
    @campo = "autor"
  elsif @tipo.to_i == 3
    @campo = "cota"
  end
  page.redirect_to(:action => "index",:el Tit => @tit,:elcampo => @campo)
else
  if @exito==0
    page.alert("Debe ingresar un título")
  else
    page.alert("No existen libros con ese título")
  end
end
end
```

Figura 4.17: Código de la vista Javascript para realizar la búsqueda de los libros a alquilar.

Para la realización de los checkbox que permiten la selección de los libros a alquilar, se hizo en el controlador un método llamado **select_item**, el cual verifica mediante un ciclo el estado de la sesión, es decir, cuando el estudiante entra a la vista en una variable llamada **checkeados** se crea un nuevo arreglo donde se guardan los items seleccionados. Este método además va limpiando el arreglo, es decir, cuando el estudiante entra a una sesión no deben haber checkbox marcados y una vez que finaliza la sesión se borran los checkbox. El código se muestra en la figura 4.18:


```

def select_item
  if session[:arreglo_chequeados].nil?
    @chequeados = Array.new
  else
    @chequeados = session[:arreglo_chequeados]
  end

  if @chequeados.include?(params[:id_isbn])
    @chequeados.delete(params[:id_isbn])
  else
    @chequeados << params[:id_isbn]
  end
  session[:arreglo_chequeados]= @chequeados
end

```

Figura 4.18: Código del condicional en el Controlador para realizar la selección de los libros a alquilar.

En la vista Index se realizó un ciclo donde primero se verifica que el estudiante haya seleccionado al menos un libro, en este caso, ya hay libros seleccionados, por lo cual se guardan los ISBN de los libros en la variable **chequeados** y luego se llama a la acción **select_item** explicada anteriormente. El código se muestra en la figura 4.19:

```

<%= @libros.each do |libro| %>
<tr>

  <tr class="<%=cycle('list-line-odd', 'list-line-even') %>">
  <td>
    <%= if libro.cant Ejem !=0%>
    <%= check_box_tag 'to_check[]',
      'yes',
      (@chequeados && @chequeados.include?(libro.isbn.to_s)),
      :onclick => remote_function(:url => url_for(:controller =>:alquileres,:action => :select_item,:id_isbn=>libro.isbn) %>
    <%= @sihay= 1%>
    <%=end%>
  </td>

    <td><%=h libro.isbn %></td>
    <td><%=h libro.cota %></td>
    <td><%=h libro.titulo %></td>
    <td><%=h libro.autor %></td>
    <td><%=h libro.costo_alquiler %></td>
  </tr>
</tr>
<%= end %>

```

Figura 4.19: Código de la vista Index para realizar la selección de los libros a alquilar.

Las pruebas realizadas para la verificación de todo lo desarrollado, consistieron en la utilización de las interfaces creadas y la visualización de las mismas con el propósito de corroborar que fueran acordes a lo solicitado y que contaran con el diseño característico del sistema. Luego fue generado un comprobante de alquiler, utilizando la interfaz previa. El resultado de las pruebas fue satisfactorio. A continuación se muestran las pruebas de aceptación realizadas en la presente iteración.

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
1	Alquiler	Proporcionar los datos necesarios para generar el comprobante de alquiler.	El sistema debe generar el comprobante de alquiler una vez realizada la selección de los libros a alquilar, solicitar el periodo académico, los datos personales del estudiante que realizó el alquiler y la información del libro alquilado.	El sistema solicita los datos, tal cual como se esperaba, generando el comprobante de alquiler de manera satisfactoria.
2	Alquiler	Verificar el formato PDF del comprobante de alquiler generado.	El comprobante debe contener el encabezado, el título, el periodo académico, el texto donde se especifican los datos del estudiante, los datos de los libros alquilados, la fecha de emisión y el precio total del alquiler.	El formato PDF del comprobante de alquiler generado posee todos los datos esperados.
3	Alquiler	Verificar el correcto funcionamiento de la devolución de los ejemplares.	Una vez realizada la devolución de los ejemplares, los mismos deben estar de nuevo disponibles para alquilar y la cantidad de ejemplares del libro asociado debe incrementar.	La devolución de los ejemplares funcionó adecuadamente, incrementándose los valores necesarios para tal fin.
4	Alquiler	Comprobar si después de realizar un alquiler, se actualiza la lista de Administración de Alquileres del personal administrativo.	Después de que un estudiante realice un alquiler, la solicitud de ese alquiler debe aparecer en la lista de Administración de Alquileres del personal administrativo.	Efectivamente se actualizó la lista de Administración de Alquileres del personal administrativo, luego de que un estudiante realizó un alquiler.

Cuadro 4.6: Pruebas de Aceptación de la Iteración 0

Iteración 1: Módulo de Libros y Ejemplares

A continuación en el cuadro 4.7 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Libros y Ejemplares.

Módulo(s)	Modelo(s)	Vistas	Controlador(es)
Libros	libro.rb	buscar.rjs buscar2.rjs create.rjs edit.html.erb buscar_amazon.html.erb buscar_otro_buscador.html.erb index.html.erb new.html.erb show.html.erb	libro_controller.rb
Ejemplares	ejemplare.rb	buscar.rjs create.rjs edit.html.erb index.html.erb new.html.erb show.html.erb	ejemplare_controller.rb

Cuadro 4.7: Esquema del Módulo de Libros y Ejemplares con su MVC asociado

La planificación de las Historias de Usuario para esta iteración fue organizada de la siguiente manera:

Iteración 1	
Descripción	Desarrollo del Módulo de Libros y Ejemplares
Historias de Usuario	10. Visualizar los Libros que se encuentran en la Bolsa del Libro. 11. Incorporar al sistema un libro disponible en Amazon. 12. Asignar una imagen a un libro. 13. Asignar a un libro sus ejemplares correspondientes. 14. Visualizar los Ejemplares que existen en el sistema. 15. Realizar acciones sobre los libros y ejemplares. 16. Visualizar los campos obligatorios para la creación de libros y ejemplares. 17. Buscar un libro. 18. Buscar un ejemplar.
Tiempo Estimado	24 días
Fecha Inicio - Fin	11/01/2011 - 15/02/2011

Para la creación de las vistas del módulo de Libros, se diseñó una interfaz sencilla, donde aparecen los atributos más importantes de un libro: ISBN, cota, título, autor y cantidad de ejemplares. Además se muestra la imagen que identifica a cada libro (correspondiente a la carátula de dicho libro). La siguiente vista corresponde al catálogo de libros disponibles en la Bolsa del Libro, y la misma puede ser visualizada por el personal administrativo desde su página respectiva. Ver figura 4.20

Home -- Administrador Cerrar Sesión

Administración de Libros

Buscar por: Título

« Anterior 1 2 Siguiente »

	ISBN	COTA	TITULO	AUTOR	CANTIDAD DE EJEMPLARES	
	0582306574	f2.90.0	Friends Starter Global Activity Book (FRND)	Olivia Date	3	Mostrar Editar Eliminar
	8408082914	B2.10.0	Chile y la isla de Pascua (Country Guide) (Spanish Edition)	Planet	1	Mostrar Editar Eliminar
	8495787148	P2.10.0	Los Tres Pasos: UN Relato Imprescindible Para Definir Nuestra Verdadera Pasión Laboral (Spanish Edition)	Arrie Warren	0	Mostrar Editar Eliminar
	970106111X	Q2.90.0	Química - 9ª Edición	RAYMOND CHANG	3	Mostrar Editar Eliminar

Figura 4.20: Vista del catálogo de libros desde el perfil de Administrador

Para incorporar un nuevo libro en el sistema, se creó una vista donde se puede elegir entre incorporarlo haciendo uso de un buscador que verifica si el libro solicitado se encuentra disponible en Amazon, donde en caso afirmativo se descarga el libro y sus datos principales para luego guardarlo en el sistema, o incorporarlo de manera manual, rellenando los datos del libro a través de un formulario. Ver figura 4.21

NUEVO LIBRO
Formulario para la creación de un Libro

Buscar:

ISBN *
Código ISBN del libro

COTA *
Código del Ejemplar

Título *
Título completo del libro

Autor *
Nombre del autor

Edición *
Edición del libro

(Re)Impreso *
Impreso o Reimpreso

Año *
Año de edición

Editorial *
Nombre de la editorial

Lugar Impresión
Lugar de impresión del libro

Costo referencial
Costo del libro

Escuela
Escuela a la que pertenece el libro

Imagen
Imagen del libro

Costo Alquiler *
Costo de alquiler del libro

Figura 4.21: Vista para la creación de un nuevo libro

Adicionalmente se crearon vistas para las opciones de *Mostrar* y *Editar* un libro para los fines de Mostrar el detalle de un libro en particular (imagen, ISBN, cota, título, autor, edición, impresión/reimpresión, año, escuela, editorial, lugar de impresión, costo referencial, costo de alquiler y cantidad de ejemplares) y realizar la edición de los datos del libro. Ver figura 4.22



Figura 4.22: Vista donde se muestran en detalle los datos de un libro

En cuanto a los ejemplares, se creó una vista donde se muestran todos los ejemplares creados por libro con la cota genérica asociada, ISBN, título, autor, costo de alquiler y ubicación (estante, prestado, en reparación, etc.). Ver figura 4.23



Figura 4.23: Vista del catálogo de ejemplares del personal Administrativo

Con respecto a la codificación en esta iteración, se definió un método llamado **buscar_amazon** en el controlador, para realizar la búsqueda de los libros en Amazon. Este método recibe el ISBN que el usuario introduce por la vista y lo guarda en una variable llamada **data**, luego se conecta a Amazon a través de las claves de acceso (**access_key_id** y **secret_key** creadas al momento de abrir la cuenta) para buscar el libro que pertenece a

dicho ISBN. De los libros se buscan los atributos imagen, título, autor, ISBN, edición y editorial en caso de que esos datos se encuentren en Amazon, de lo contrario, se coloca en su lugar la frase *NO DEFINIDO* para determinar que no se encontró el atributo buscado. Ver figura 4.24

```
def buscar_amazon
  @data = params[:isbn_buscar]

  begin
    Amazon::Ecs.configure do |options|
      options[:aws_access_key_id] = 'AKIAI2372MHDUEUTLORA'
      options[:aws_secret_key] = '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'
    end
    res = Amazon::Ecs.item_search(@data, {:response_group => 'Medium', :sort => 'salesrank'})
    res.items.each do |item|
      @libro = Libro.new
      @libro.id_escuela = 1

      item.get('asin')
      item.get('itemattributes/title')
      # or return Amazon::Element instance
      atts = item.search_and_convert('itemattributes')
      if !item.get_hash.nil?
        @libro.imagen_url = item.get_hash("smallimage")[:url] || "IMagen No definido"
        if @libro.imagen_url=""
          @libro.imagen_url = "/images/libros/libro.jpg"
          if item.get_hash=""
            flash[:notice] = "@libro.imagen_url"
          end
        end
      end
      @libro.imagen_url = item.get_hash("smallimage")[:url] || "IMagen No definido"
    rescue
    end
  end if
end
```

Figura 4.24: Código definido para realizar la búsqueda de un libro en Amazon.

En la Vista de nuevo libro, se utilizó un formulario de ruby que representa los botones de búsqueda. Este formulario recibe el ISBN del libro que el administrador desea buscar y luego si el administrador oprime el botón llamado Amazon, la acción llama al método **buscar2**. Ver figura 4.25

```
<div id="buscador_nuevo_libro">

  <% form_remote_tag :url => url_for(:action => "buscar2") do%>
    <b>Buscar:</b>
    <input id="isbn_buscar" name="isbn_buscar" type="text" style="float:none;margin:0;" />
    <form method="post" action="/backend/libros/buscar2" class="button-to"><div><input type="submit" value="Amazon"
    <%end%>
  </div>
</div>
```

Figura 4.25: Código de la vista Nuevo para la creación de un libro.

Además, se realizó una vista JavaScript llamada **buscar2** para verificar que el administrador está introduciendo los datos correctamente y enviar la acción al método **buscar_amazon** que se encuentra en el controlador, explicado anteriormente. Ver figura 4.26

```
if @exito
  page.redirect_to(:action => "buscar_amazon",:isbn_buscar => @data)
else
  page.alert("Debe ingresar un ISBN")
end
```

Figura 4.26: Código de la vista JavaScript para la creación de un libro.

Para permitir al Personal Administrativo la posibilidad de adjuntar una imagen a un libro se utilizó la gema Paperclip [12]. Se creó un archivo de migración llamado `AddAttachmentsImagenToLibro` para agregar cuatro campos nuevos al modelo `Libro` (`imagen_file_name`, `imagen_content_type`, `imagen_file_size` e `imagen_updated_at`). El código del archivo de migración se muestra a continuación en la figura 4.27:

```
class AddAttachmentsImagenToLibro < ActiveRecord::Migration
  def self.up
    add_column :libros, :imagen_file_name, :string
    add_column :libros, :imagen_content_type, :string
    add_column :libros, :imagen_file_size, :integer
    add_column :libros, :imagen_updated_at, :datetime
  end

  def self.down
    remove_column :libros, :imagen_file_name
    remove_column :libros, :imagen_content_type
    remove_column :libros, :imagen_file_size
    remove_column :libros, :imagen_updated_at
  end
end
```

Figura 4.27: Código del archivo de migración `AddAttachmentsImagenToLibro`.

Luego se indicó en el modelo `Libro` que se va a usar un archivo adjunto con nombre `imagen` para cada libro, se especificó la ruta donde se van a guardar las imágenes de los libros y se definió una imagen por defecto para los libros a los cuales no se le adjunte imagen (`libro.jpg`).

Además se validó el tamaño (5 megabytes) del archivo adjunto (`imagen`) y que la imagen perteneciera a los tipos `jpeg`, `png` y `jpg`. El código para definir las validaciones de Paperclip se muestra en la figura 4.28:

```
# Paperclip
has_attached_file :imagen, :url => "/images/libros/:id/:basename.:extension",
                          :path => ":rails_root/public/images/libros/:id/:basename.:extension",
                          :default_url => "/images/libros/libro.jpg"

# Validaciones de Paperclip
validates_attachment_size :imagen, :less_than => 5.megabytes
validates_attachment_content_type :imagen, :content_type => ['image/jpeg', 'image/png', 'image/jpg']
before_validation :descargar_imagen_remota, :if => :imagen_url_provided?
```

Figura 4.28: Código para definir las validaciones de Paperclip.

Después se creó un código en la vista de `Nuevo Libro` para indicarle a Rails que se utilizará un formulario *multipart*, que es necesario para subir las imágenes en un formulario HTML y se definió el botón para adjuntar la imagen al libro. Este código se muestra en la figura 4.29:

```

<% form_for @libro, :url => ( :action => "create"), :html => ( :multipart => true ) do |f| %>
<label>Imagen
<span class="small">Imagen del libro</span>
</label>
<input id="libro_imagen" name="libro[imagen]" size="12" type="file" />

```

Figura 4.29: Parte del código de la vista de Nuevo Libro con el formulario multipart.

Finalmente, para mostrar la imagen en la vista necesaria se utilizó el método url del objeto Paperclip. El código para mostrar la imagen se puede observar en la figura 4.30:

```

<%=image_tag libro.imagen.url, :class => 'list-image'%>

```

Figura 4.30: Código que permite mostrar la imagen adjuntada a un libro.

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración.

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
5	Libros y Ejemplares	Verificar el correcto funcionamiento al descargar un libro desde Amazon.	El sistema debe buscar el libro especificado en el buscador a través de su ISBN, si el libro se encuentra en Amazon, debe descargar toda la información referente al mismo, de lo contrario, debe generar un error.	El sistema se comportó como se esperaba, descargando el libro con sus datos en caso de encontrarlo en Amazon y generando un error en caso contrario.
6	Libros y Ejemplares	Verificar el correcto funcionamiento al crear los ejemplares de un libro en particular.	Se especifica en el sistema el número de ejemplares que se requieren crear de un libro, indicando el ISBN y la cota genérica del libro y se debe generar la cantidad de ejemplares indicada, cada ejemplar con una cota distinta.	El sistema permitió la creación de los ejemplares de un libro indicando la cantidad específica de los mismos sin ningún problema.
7	Libros y Ejemplares	Comprobar si funciona la carga de imágenes para los libros.	Al momento de creación de un libro de forma manual, al seleccionar <i>Adjuntar imagen</i> , se debe guardar la imagen seleccionada para identificar el libro que se estaba creando.	Después de seleccionar la imagen deseada para identificar el libro, una vez guardado el libro, el mismo se muestra con la imagen establecida.

Cuadro 4.8: Pruebas de Aceptación de la Iteración 1

Iteración 2: Módulo de Solvencias

A continuación en el cuadro 4.9 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Solvencias.

Módulo	Modelos	Vistas	Controladores
Solvencias	solvencia.rb periodo_lectivo.rb usuario.rb delete_old_post.rb	edit.html.erb index.html.erb show.html.erb guardasol.html.erb mensaje.html.erb solicita.html.erb	solvencias_controller.rb application_controller.rb

Cuadro 4.9: Esquema del Módulo de Solvencias con su MVC asociado

La planificación de las Historias de Usuario para esta iteración fue organizada de la siguiente manera:

Iteración 2	
Descripción	Creación del Módulo de Solvencias.
Historias de Usuario	19. Seleccionar el tipo de solvencia requerida. 20. Visualizar mensaje de estado de solvencia al solicitar una solvencia. 21. Visualizar las solicitudes de solvencia. 22. Realizar acciones con las solicitudes de solvencia. 23. Buscar la solvencia deseada. 24. Descargar el comprobante de solvencia en formato PDF.
Tiempo Estimado	12 días
Fecha Inicio - Fin	16/02/2011 - 10/03/2011

Para el diseño de las vistas de esta iteración, primero se creó una vista donde el estudiante debe seleccionar el tipo de solvencia que requiere de una lista desplegable y seguidamente se le indica a través de un mensaje la fecha en la cual debe retirar su solvencia en la Bolsa del Libro. Además tiene la opción de descargar su comprobante de solvencia, pero de igual forma debe dirigirse a la Bolsa del Libro para que su solvencia sea sellada y así tenga validez. Ver figura 4.31



Figura 4.31: Vista donde el estudiante debe seleccionar el tipo de solvencia que requiere

La constancia de solvencia generada, luego de que el estudiante seleccione el tipo de solvencia, tiene los datos correspondientes a la solvencia que solicitó el estudiante (motivo y escuela), en caso de que el tipo seleccionado sea Retiro o Cambio; y los datos personales del estudiante. Ver figura 4.32

PAG. 1 DE 1

UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
BIBLIOTECA ALONSO GAMERO
BOLSA DEL LIBRO

CONSTANCIA DE SOLVENCIA
PERIODO: 1 - 2011

FECHA: 20/07/2011

APELLIDOS Y NOMBRES: Jiménez Andreina
CEDULA DE IDENTIDAD: 18141182

Por medio de la presente, se hace constar que el Br. **Jiménez Andreina**; Cédula de Identidad No. **18141182**.
ESTA SOLVENTE CON ESTA OFICINA.

Se expide esta solvencia a solicitud de la parte interesada a los 20 días del mes de julio de 2011.

ESCUELA: COMPUTACION
MOTIVO: RETIRO TOTAL

Por: La Bolsa del Libro

CUIDAD UNIVERSITARIA DE CARACAS - PATRIMONIO MUNDIAL DE LA HUMANIDAD

NOTA: Esta solvencia tiene validez por 5 (cinco) días hábiles.

Figura 4.32: Vista del Documento PDF del Comprobante de Solvencia de tipo Retiro

Si el estudiante selecciona la solvencia de tipo Grado, se genera un comprobante como el que se muestra en la figura 4.33:

PAG. 1 DE 1

UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
BIBLIOTECA ALONSO GAMERO
BOLSA DEL LIBRO

CONSTANCIA DE SOLVENCIA
PERIODO: 1 - 2011

FECHA: 20/07/2011

APELLIDOS Y NOMBRES: Jiménez Andreina
CEDULA DE IDENTIDAD: 18141182

Por medio de la presente, se hace constar que el Br. **Jiménez Andreina**; Cédula de Identidad No. **18141182**.
ESTA SOLVENTE CON ESTA OFICINA Y HA CUMPLIDO CON EL REQUISITO DE DONACION.

Se expide esta solvencia a solicitud de la parte interesada a los 20 días del mes de julio de 2011.

Por: La Bolsa del Libro

CUIDAD UNIVERSITARIA DE CARACAS - PATRIMONIO MUNDIAL DE LA HUMANIDAD

NOTA: Esta solvencia tiene validez por 10 (diez) días hábiles.

Figura 4.33: Vista del Documento PDF del Comprobante de Solvencia de tipo Grado

Por cada solvencia solicitada por los estudiantes, se crea una lista de solvencias solicitadas que puede manejar el personal Administrativo, donde aparecen todos los datos

correspondientes a la solvencia y los datos del estudiante que la solicitó. Ver figura 4.34

NOMBRE	APELLIDO	CEDULA	PERIODO_LECTIVO	FECHA SOLICITUD	FECHA ENTREGA	MOTIVO	
Joselyn	Oviedo	18110941	I - 2011	17/05/2011	24/05/2011	CAMBIO DE UNIVERSIDAD	Mostrar Editar Eliminar
Andreina	Jimenez	18141182	I - 2011	18/05/2011	25/05/2011	CAMBIO DE ESCUELA	Mostrar Editar Eliminar

Figura 4.34: Vista de la Administración de Solvencias

Con respecto a la codificación en esta iteración, para la generación de los comprobantes PDFs de solvencia, se crearon dos métodos en el controlador de solvencias, llamados **comprobante_de_solvencia** y **comprobante_de_solvencia_grado**, debido a que existen dos tipos de comprobantes de solvencia: uno para el caso de que la solvencia sea por retiro (total o permanente) o cambio (escuela, facultad, universidad) y otro para el caso de que la solvencia sea por grado.

En estos métodos, en primer lugar se inicializan las variables **solvencia** y **contactoinfo** con los valores necesarios que permiten obtener la información de la solvencia y del usuario que solicita la solvencia.

Luego se crea el nuevo documento PDF y se asignan los márgenes y las coordenadas correspondientes para la ubicación del texto que va en el encabezado del documento, las imágenes de la Universidad Central de Venezuela y de la Facultad de Ciencias respectivamente, el título del documento, el periodo académico para el cual fue emitido el comprobante, la fecha de emisión del mismo y los datos del estudiante que solicita la solvencia (el nombre, el apellido y la cédula de identidad).

Después se imprime la Escuela del Estudiante que solicita la solvencia y el Motivo de la solicitud (retiro o cambio), sólo en el caso de que la solvencia solicitada sea de tipo retiro o cambio.

Finalmente, se especifican los días de validez de la solvencia y se define el nombre del comprobante de solvencia.

Dichos métodos se muestran a continuación en las figuras 4.35, 4.36, 4.37 y 4.38:

```
def comprobante_de_solvensia

  @solvensia = Solvensia.find(params[:id_solvensia])
  # @contacto = Contacto.find(params[:id])
  # @escuela = Escuela.find(params[:id])

  @contacto_info = @solvensia.usuario.contacto

  nombre_completo_archivo_pdf = "SOLVENCIA_PDF"
  pdf = PDF::Writer.new

  # CODIGO PARA GENERAR EL PDF

  pdf.margins_pt(40,20,20,20)
  pdf.select_font('Times-Roman')

  pdf.start_page_numbering(500, 760, size=9, pos = nil, pattern = "PAG. <PAGENUM> DE <TOTALPAGENUM>", starting = nil)

  pdf.text "", :font_size => 10, :justification => :center
  pdf.text "UNIVERSIDAD CENTRAL DE VENEZUELA", :justification => :center
  pdf.text "FACULTAD DE CIENCIAS", :justification => :center
  pdf.text "BIBLIOTECA ALONSO GAMERO", :justification => :center
  pdf.text "BOLSA DEL LIBRO", :justification => :center

  pdf.add_image_from_file "#{RAILS_ROOT}/public/images/ucv.jpg", 30, 673, 85
  pdf.add_image_from_file "#{RAILS_ROOT}/public/images/ciencias.jpg", 515, 650, 60

  pdf.text "\n\n"
  pdf.text "<b>\n\nCONSTANCIA DE SOLVENCIA</b>", :justification => :center, :font_size => 14
  pdf.text "<b>PERIODO: #{@solvensia.periodo_lectivo.num_periodo} - #{@solvensia.periodo_lectivo.ano_periodo}</b>", :justification => :center, :font_size => 11

  pdf.text "\n\n"

  pdf.add_text 455, 610, "<b>FECHA:</b> " + Time.now.strftime("%d/%m/%Y") #x,y

  pdf.text "\n\n"
  pdf.text to_utf16("<b>\n\nAPELLIDOS Y NOMBRES:</b> #{@contacto_info[:apellido]} #{@contacto_info[:nombre]}") + " *14, :justification => :right, :font_size => 12
  pdf.text "<b>CEDULA DE IDENTIDAD:</b> #{@contacto_info[:cedula]}") + " *14, :justification => :right, :font_size => 12

  pdf.text "\n\n\n\n"
end
```

Figura 4.35: Código del método para la generación del comprobante PDF de Solvensia de tipo Retiro o Cambio (Parte I).

```
pdf.text to_utf16("Por medio de la presente, se hace constar que el Br. <b> #{@contacto_info[:apellido]} #{@contacto_info[:nombre]}</b> Cédula de Identidad No. <b> #{@contacto_info[:cedula]}</b>, <b> ESTA SOLVENTE CON ESTA OPTICINA.") + " *14, :justification => :justify

pdf.text "\n\n"

pdf.text to_utf16("Se expide esta solvensia a solicitud de la parte interesada a los #{Time.now.strftime("%d")} dias del mes de #{Time.now.strftime("%B")} de #{Time.now.strftime("%Y")}.", :justification => :justify

pdf.text "\n\n"
pdf.text "<b>\n\nESCUELA:</b> #{@contacto_info[:escuela[:nombre_escuela]]", :justification => :left

pdf.text "<b>\n\nMOTIVO:</b> #{@solvensia[:motivo]}", :justification => :left

pdf.text "\n\n\n\n"

pdf.text "_____", :justification => :center

pdf.text "<b>Por: La Bolsa del Libro</b>", :justification => :center

pdf.text "\n\n\n\n"

pdf.text "CIUDAD UNIVERSITARIA DE CARACAS - PATRIMONIO MUNDIAL DE LA HUMANIDAD", :justification => :center, :font_size => 8

pdf.text "\n\n\n\n"

pdf.text to_utf16("NOTA: Esta solvensia tiene validez por 5 (cinco) dias hábiles.", :justification => :justify, :font_size => 11

# FIN DEL CODIGO PARA GENERAR EL PDF
pdf.save_as(nombre_completo_archivo_pdf)
f = File.open(nombre_completo_archivo_pdf, 'rb')
contenido = f.read
f.close
send_data(contenido, { :type => "application/pdf", :filename => "solvensia_#{@contacto_info[:cedula]}_#{@solvensia.periodo_lectivo.num_periodo} - #{@solvensia.periodo_lectivo.ano_periodo}.pdf" })

end
```

Figura 4.36: Código del método para la generación del comprobante PDF de Solvensia de tipo Retiro o Cambio (Parte II).

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración.

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
8	Solvencias	Proporcionar los datos necesarios para generar el comprobante de solvencia.	El sistema debe generar el comprobante de solvencia una vez realizada la selección de la solvencia que requiere el estudiante, solicitar el periodo académico, los datos personales del estudiante que realizó la solvencia y la información correspondiente a la solicitud.	El sistema solicita los datos, tal cual como se esperaba, generando el comprobante de solvencia de manera satisfactoria.
9	Solvencias	Verificar el formato PDF de los comprobantes de solvencia.	Los comprobantes deben contener el encabezado, el título, el periodo académico, el texto donde se especifican los datos del estudiante, los datos de la solvencia solicitada y la fecha de emisión.	El formato PDF del comprobante de solvencia generado posee todos los datos esperados.
10	Solvencias	Verificar el funcionamiento del buscador	Una vez seleccionado el atributo para realizar la búsqueda, la misma debe reflejarse en el sistema.	Efectivamente, la búsqueda se reflejó en el sistema.

Cuadro 4.10: Pruebas de Aceptación de la Iteración 2

Iteración 3: Módulo de Listados

A continuación en el cuadro 4.11 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Listados.

Módulo	Modelos	Vistas	Controladores
Listados	listado.rb	adquiridos.html.erb alquilados.html.erb cartelera.html.erb deudores.html.erb ejemp_excluidos.html.erb excluidos.html.erb inventario.html.erb librosalquilados.html.erb mensaje.xls.html.erb most_ejemplar.html.erb multados.html.erb portitulo.html.erb solicita.html.erb	listados_controller.rb

Cuadro 4.11: Esquema del Módulo de Listados con su MVC asociado

La planificación de las Historias de Usuario es la siguiente:

Iteración 3	
Descripción	Creación del Módulo de Listados.
Historias de Usuario	25. Seleccionar un tipo de listado. 26. Visualizar el listado seleccionado. 27. Buscar por atributos en cada listado. 28. Descargar el listado requerido en formato PDF. 29. Descargar el listado de libros alquilados en formato EXCEL.
Tiempo Estimado	11 días
Fecha Inicio - Fin	11/03/2011 - 30/03/2011

Para el diseño de las vistas de esta iteración, primero creamos una vista, en la cual el personal Administrativo puede seleccionar algún tipo de listado a través de una lista desplegable donde aparecen los listados más comunes que son requeridos en el día a día. Ver figura 4.39




Figura 4.39: Vista de la selección de los tipos de listados

Luego de seleccionar el listado requerido, y generarlo a partir de la opción *Generar*, el mismo se muestra en una tabla, donde aparecen los atributos más importantes que destacan a cada tipo de listado. Ver figura 4.40




Figura 4.40: Vista del listado de libros adquiridos

En la vista del listado generado aparece la opción de *Descargar listado*, donde se puede descargar el documento en formato PDF del listado generado. Ver figura 4.41



UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
BIBLIOTECA ALONSO GAMERO
BOLSA DEL LIBRO

PAG. 1 DE 2



LIBROS ADQUIRIDOS

FECHA: 20/07/2011

TIPO DE ADQUISICION	COTA	AUTOR	TITULO	EDICION	ESCUELA
COMPRA INGRESOS PROPIOS	M1.20.04	JOHNSONBAUGH RICHARD	MATEMATICAS DISCRETAS 6ª EDICION - 2005	Edicion No definida	MATEMATICA
COMPRA INGRESOS PROPIOS	M1.20.02	JOHNSONBAUGH RICHARD	MATEMATICAS DISCRETAS 6ª EDICION - 2005	Edicion No definida	MATEMATICA
COMPRA INGRESOS PROPIOS	M1.20.01	JOHNSONBAUGH RICHARD	MATEMATICAS DISCRETAS 6ª EDICION - 2005	Edicion No definida	MATEMATICA
COMPRA INGRESOS PROPIOS	c4.45.03	Gabriel Garcia Márquez	Cien Años de Soledad	6	BIOLOGIA
COMPRA INGRESOS PROPIOS	c2.89.03	David Flanagan/Yukihiko Matsumoto	The Ruby Programming Language	1	COMPUTACION
COMPRA INGRESOS PROPIOS	c2.89.02	David Flanagan/Yukihiko Matsumoto	The Ruby Programming Language	1	COMPUTACION
COMPRA INGRESOS PROPIOS	B2.10.03	Planet	Chile y la isla de Pascua (Country Guide) (Spanish Edition)	2da Edicion	BIOLOGIA
COMPRA INGRESOS PROPIOS	M1.20.03	JOHNSONBAUGH RICHARD	MATEMATICAS DISCRETAS 6ª EDICION - 2005	Edicion No definida	MATEMATICA

Figura 4.41: Vista del documento PDF del listado de libros adquiridos

Con respecto a la codificación en esta iteración, se creó en el controlador un método llamado **selección**, que guarda en una variable el tipo de listado recibido de la vista. En este método, una vez tomado el parámetro a través de un ciclo, se empieza a comparar el listado seleccionado y se envía a la acción correspondiente, la cual retorna el listado elegido. Ver figura 4.42

```

def seleccion
  @tipo = params[:Tipo_Listado]
  if @tipo=="Listado_Libros_Adquiridos"
    redirect_to :action => "adquiridos"
  else if @tipo=="Listado_Libros_Titulo"
    redirect_to :action => "portitulo"
  else if @tipo=="Listado_Libros_Alquilados"
    redirect_to :action => "alquilados"
  else if @tipo=="Listado_Libros_Excluidos"
    redirect_to :action => "excluidos"
  else if @tipo=="Listado_Ejemplares_Excluidos"
    redirect_to :action => "ejemp_excluidos"
  else if @tipo=="Listado_Inventario"
    redirect_to :action => "inventario"
  else if @tipo=="Listado_de_Cartelera"
    redirect_to :action => "cartelera"
  else if @tipo=="Listado_Deudores"
    redirect_to :action => "deudores"
  else if @tipo=="Listado_Multados"
    redirect_to :action => "multados"
  end
end
end
end
end
end
end
end
end

```

Figura 4.42: Código del método selección.

Además, en la vista Solicita se recibe a través de un formulario de selección el tipo de listado que el administrador desea generar y lo envía al método selección que se encuentra

en el controlador, explicado anteriormente. Ver figura 4.43

```

<div id="buscador_nuevo_libro">
  <${ form_tag :action => "seleccion" do}>
    <b>Solvensias:</b>
    <select name="Tipo_Listado">
      <option value="-1"> Seleccione... </option>
      <option value="Listado_Libros_Adquiridos"> Listado de Libros Adquiridos </option>
      <option value="Listado_Libros_Titulo"> Listado de Libros por Titulo </option>
      <option value="Listado_Libros_Alquilados"> Listado de Libros Alquilados </option>
      <option value="Listado_Libros_Excluidos"> Listado de Libros Excluidos </option>
      <option value="Listado_Ejemplares_Excluidos"> Listado de Ejemplares Excluidos </option>
      <option value="Listado_Inventario"> Listado de Inventario </option>
      <option value="Listado_de_Cartelera"> Listado de Cartelera </option>
      <option value="Listado_Deudores"> Listado de Deudores </option>
      <option value="Listado_Multados"> Listado de Multados </option>
    </select>
    <form method="post" action="/backend/listados/seleccion" class="button-to"><div><input type="submit" value="Generar"
      style="float:none;margin:0;" /></div></form>
  <end>
</div>

```

Figura 4.43: Código para elegir un tipo de listado.

Por otra parte, se crearon varios métodos donde se establecieron las condiciones correspondientes para generar cada tipo de listado, uno de ellos, es el Listado de Libros Alquilados. Para la creación de este listado, se realizó en el controlador un método llamado **alquilados**, en el que primero se guardan los parámetros de búsqueda que se reciben de la vista, luego en una variable llamada **config** se recibe el valor de la variable **num_paginas** que representa el número de elementos que está determinado visualizar en la vista. Por último, se realiza el despliegue de ejemplares alquilados que cumplen los parámetros correspondientes. Ver figura 4.44

```

def alquilados
  titulo = params[:eltit]
  @elcamp = params[:elcampo]
  if @elcamp.nil?
    @elcamp = "cota"
  end
  @config = Configuracion.first(:conditions => ["nombre = 'num_paginas'"])
  @ejemplares = EjemplaresLibro.paginate(:page => params[:page], :per_page => @config.valor,:conditions =>
    ["id_ubicacion = 2 and UPPER(#{@elcamp}) LIKE ?", "%#{@titulo}%"])
end

```

Figura 4.44: Código para la creación del listado de libros alquilados.

Se realizaron varios métodos en el controlador de listados para generar cada listado en formato PDF y darle formato al mismo, uno de ellos es el método **listado_adquiridos**, que genera el documento PDF del listado de libros adquiridos. En este método primeramente se inicializa la variable **ejemplares** con el valor necesario que permite obtener los atributos que aparecen en cada columna del listado.

Luego se crea el nuevo documento PDF y se asignan los márgenes y las coordenadas correspondientes para la ubicación del texto que va en el encabezado del documento, las imágenes de la Universidad Central de Venezuela y de la Facultad de Ciencias respectivamente, el título del documento y la fecha de emisión del mismo.

Después se crea una tabla con seis columnas (tipo_adquisición, cota, autor, título, edición, nombre_escuela) que contiene toda la información del listado de libros adquiridos (tipo de adquisición del libro, cota del ejemplar del libro, autor del libro, título del libro, edición del libro y nombre de la escuela a la que pertenece el libro).

Finalmente, se define el nombre del documento PDF del listado de libros adquiridos. Dicho método se muestra a continuación en la figura 4.45:

```

tabla = PDF::SimpleTable.new
tabla.heading_font_size = 9
tabla.font_size = 9
tabla.bold_headings = true
tabla.show_lines = :inner
tabla.show_headings = true
tabla.shade_rows = :none
tabla.orientation = :right
tabla.position = :top
tabla.column_order = ["tipo_adquisicion", "cota", "autor", "titulo", "edicion", "nombre_escuela"]
tabla.columns["tipo_adquisicion"] = PDF::SimpleTable::Column.new("tipo_adquisicion") { |col|
  #col.font_size=4
  col.width = 100
  col.heading = "TIPO DE ADQUISICION"
  col.heading_justification = :left
  col.justification = :left
}
tabla.columns["cota"] = PDF::SimpleTable::Column.new("cota") { |col|
  #col.font_size=4
  col.width = 60
  col.heading = "COTA"
  col.heading_justification = :left
  col.justification = :left
}
tabla.columns["autor"] = PDF::SimpleTable::Column.new("autor") { |col|
  #col.font_size=4
  col.width = 110
  col.heading = "AUTOR"
  col.heading_justification = :left
  col.justification = :left
}
tabla.columns["titulo"] = PDF::SimpleTable::Column.new("titulo") { |col|
  #col.font_size=4
  col.width = 130
  col.heading = "TITULO"
  col.heading_justification = :left
  col.justification = :left
}
tabla.columns["edicion"] = PDF::SimpleTable::Column.new("edicion") { |col|
  #col.font_size=4
  col.width = 70
  col.heading = "EDICION"
  col.heading_justification = :left
  col.justification = :left
}
  tabla.columns["nombre_escuela"] = PDF::SimpleTable::Column.new("nombre_escuela") { |col|
  #col.font_size=4
  col.width = 90
  col.heading = "ESCUELA"
  col.heading_justification = :left
  col.justification = :left
}
data = [] #de la tabla
@ejemplares.each do |ejemplare|
  data << {
    "tipo_adquisicion" => ejemplare.tipo_adquisicion,
    "cota" => ejemplare.cota,
    "autor" => to_utf16(ejemplare.autor),
    "titulo" => to_utf16(ejemplare.titulo),
    "edicion" => ejemplare.edicion,
    "nombre_escuela" => ejemplare.nombre_escuela,
  }
end
tabla.data.replace data
tabla.render_on(pdf)

#FIN DEL CODIGO PARA GENEREAR EL PDF

pdf.save_as(nombre_completo_archivo_pdf)
f = File.open(nombre_completo_archivo_pdf, 'rb')
contenido = f.read
f.close

send_data(contenido, {:type => "application/pdf", :filename=>"adquiridos.pdf"})

end

```

Figura 4.45: Parte del Código del método para la generación del documento PDF del Listado de Libros Adquiridos .

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración.

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
11	Listados	Verificar que se generen correctamente los listados seleccionados.	El sistema debe generar la vista con el tipo de listado que seleccionó el personal administrativo.	Después de que el personal administrativo eligiera el tipo de listado deseado, el sistema generó el listado correcto según la selección.
12	Listados	Verificar que el buscador por atributo de los listado funcione adecuadamente.	Después de generar un listado, al realizar la búsqueda de un atributo del mismo, se debe obtener el listado filtrado por esa búsqueda.	Se generaron los listados indicados después de efectuar la búsqueda de atributos en los mismos.

Cuadro 4.12: Pruebas de Aceptación de la Iteración 3

Iteración 4: Módulo de Usuarios

A continuación en el cuadro 4.13 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Usuarios.

Módulo	Modelos	Vistas	Controladores
Usuarios	usuario.rb alquilere.rb prealquilere.rb contacto.rb solvencia.rb rol.rb statu.rb escuela.rb usuario_email.rb usuariosrole.rb	edit.html.erb index.html.erb new.html.erb show.html.erb enviar_email.html.erb enviar_nueva_clave.html.erb cambiar_clave.html.erb inicio.html.erb olvido_clave.html.erb	contactos_controller.rb alquileres_controller.rb auditoria_correos.rb carga_masiva_controller.rb prealquileres_controller.rb solvencias_controller.rb principal_controller.rb sesion_controller.rb application_controller.rb

Cuadro 4.13: Esquema del Módulo de Usuarios con su MVC asociado

La planificación de las Historias de Usuario es la siguiente:

Iteración 4	
Descripción	Desarrollo del Módulo de Usuarios.
Historias de Usuario	30. Buscar usuarios. 31. Crear un nuevo usuario. 32. Realizar acciones sobre un usuario.
Tiempo Estimado	6 días
Fecha Inicio - Fin	31/03/2011 - 06/04/2011

Con respecto al diseño de las vistas en esta iteración, se le permite a la parte administrativa visualizar los usuarios de la Bolsa del Libro desde la página principal, en una tabla donde aparecen datos como: cédula de identidad, nombre, apellido, status (solvente, deudor), dirección actual, teléfono actual, e-mail y escuela. Ver figura 4.46



Figura 4.46: Vista del Módulo de Usuarios

De manera general, en cuanto a la codificación, para la visualización de los usuarios del sistema, en la vista Index por ejemplo, se manejan todos los atributos que se van a visualizar acerca de los usuarios del sistema, en este caso se manejan el nombre, apellido, cédula, dirección actual, teléfono actual, correo electrónico, nombre de la escuela a la que pertenece el usuario, el rol del usuario (indica si es estudiante, administrador o empleado) y el estatus del usuario que puede ser: deudor, solvente, usuario y multado si el usuario es estudiante. En caso de que el usuario sea administrador o empleado se tiene el estatus *No Aplica*. Ver figura 4.47:

```

<%= @contactos.each do |contacto| %>
  <tr>
    <td><%=h contacto.cedula %></td>
    <td><%=h contacto.nombre %></td>
    <td><%=h contacto.apellido %></td>
    <%= if contacto.usuario.roles.collect{|r|r.nombre}.include?('Estudiante') %>
      <td><%=h contacto.statu.status %></td>
    <%= else %>
      <td>No aplica</td>
    <%= end %>

    <%=#*td><%=h contacto.statu.status %>
    <td><%=h contacto.dir_actual %></td>
    <td><%=h contacto.tel_actual %></td>
    <td><%=h contacto.e_mail %></td>
    <td><%=h contacto.escuela.nombre_escuela %></td>

    <td class="list-actions">
      <%= if controller.usuario_puede?(usuario_actual, 'backend/contactos/', 'show') %>
        <%= link_to 'Mostrar', backend_contactos_mostrar_url(contacto) %>
      <%= end %>

      <%= if controller.usuario_puede?(usuario_actual, 'backend/contactos/', 'edit') %>
        <%= link_to 'Editar', backend_contactos_editar_url(contacto) %>
      <%= end %>

      <%= if controller.usuario_puede?(usuario_actual, 'backend/contactos/', 'destroy') %>
        <%= link_to 'Eliminar', backend_contactos_eliminar_url(contacto), :confirm => "%Está seguro que desea Eliminar el Usuario?" %> </td>
    <%= end %>
  </tr>
</%=>

```

Figura 4.47: Parte del código creado para la visualización del Módulo de Usuarios.

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración.

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
13	Usuarios	Comprobar el adecuado funcionamiento del Módulo de Usuarios.	Cuando un usuario pasa a tener estado deudor o solvente, esos cambios de estado deben verse reflejados en el módulo de Usuarios.	Después de que un usuario realiza ciertos alquileres y solicita solvencias, efectivamente su estado cambia en la tabla de los Usuarios, según sea el caso.
14	Usuarios	Verificar el funcionamiento del buscador.	Cuando se seleccione un criterio de búsqueda para buscar los usuarios, se debe obtener la búsqueda indicada.	El sistema realizó la búsqueda de usuarios de manera satisfactoria, una vez seleccionado el criterio de búsqueda.

Cuadro 4.14: Pruebas de Aceptación de la Iteración 4

Iteración 5: Módulo de Configuraciones

A continuación en el cuadro 4.15 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Configuraciones.

Módulo	Modelo(s)	Vista(s)	Controlador(es)
Configuraciones	configuracione.rb	buscar.rjs edit.html.erb index.html.erb new.html.erb show.html.erb	configuraciones_controller.rb

Cuadro 4.15: Esquema del Módulo de Configuraciones con su MVC asociado

La planificación de las Historias de Usuario es la siguiente:

Iteración 5	
Descripción	Creación del Módulo de Configuraciones
Historias de Usuario	33. Visualizar las Configuraciones. 34. Realizar cambios sobre los valores de las configuraciones del sistema.
Tiempo Estimado	5 días
Fecha Inicio - Fin	07/04/2011 - 13/04/2011

El módulo de configuraciones lleva el mismo diseño de los módulos anteriores, donde la información se muestra en tablas, en este caso, en la tabla de las Configuraciones se pueden visualizar las diferentes variables que pueden ser manipuladas por el personal Administrativo para ser establecidas en el sistema de acuerdo a su valor. Algunas de las variables que se pueden configurar son: la cantidad de días para buscar el libro en la Bolsa

del Libro, la fecha de inicio y de culminación del pre-alquiler de libros, la paginación de todas las tablas del sistema, entre otras. Ver figura 4.48.

The screenshot shows the 'Configuraciones' (Configurations) module. The header includes the library name 'Biblioteca Alonso Gamero Bolsa del Libro' and navigation links: 'Historia | Servicios | Normativas | Horarios'. A user is logged in as 'Administrador'. The main content area features a search bar and a table of configurations.

nombre	Valor	Valor de Fechas	
num libros	2		Mostrar Editar
cant dias	15		Mostrar Editar
num prealquiler	2		Mostrar Editar
flope devolucion		14/05/2011	Mostrar Editar

Footer information: Av. Los Ilustres, Los Chaguaramos, Facultad de Ciencias Universidad Central de Venezuela, Caracas ZP 1040, Apartado Postal 20513, Teléfonos: (0212) 6051159.

Figura 4.48: Vista del Módulo de Configuraciones

En cuanto a la codificación, para la visualización de las configuraciones del sistema, en la vista Index se manejaron los atributos de las configuraciones que se deben visualizar como lo son el nombre de la variable de configuración, el valor uno y el valor dos. Es importante destacar que el valor uno representa el valor numérico de la variable de configuración y el valor dos representa el valor de configuración de una fecha. Posteriormente, se controlan las distintas acciones que pueden manejar los usuarios administrador y empleado. Ver la figura 4.49:

```

<% @configuraciones.each do |configuracione| %>
  <tr>
    <td><%=h configuracione.nombre %></td>
    <td><%=h configuracione.valor %></td>
    <td><%=h configuracione.valor2 %></td>
    <td class="list-actions">
      <% if controller.usuario_puede?(usuario_actual, 'backend/configuraciones/', 'show') %>
      <%= link_to 'Mostrar',backend_configuraciones_mostrar_url(configuracione) %>
      <% end %>

      <% if controller.usuario_puede?(usuario_actual, 'backend/configuraciones/', 'edit') %>
      <%= link_to 'Editar',backend_configuraciones_editar_url(configuracione) %>
      <% end %>

      <% if controller.usuario_puede?(usuario_actual, 'backend/configuraciones/', 'destroy') %>
      <%= link_to 'Eliminar',backend_configuraciones_eliminar_url(configuracione), :confirm => "¿Está seguro que desea
      Eliminar la configuración?"%></td>
    <% end %>
  </tr>
<% end %>

```

Figura 4.49: Parte del código que genera la tabla de las Configuraciones.

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración.

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
15	Configuraciones	Verificar si al establecer las configuraciones los cambios se ven reflejados en el sistema.	Una vez que el Personal Administrativo establezca alguna configuración, el cambio debe reflejarse en el sistema.	Después de establecer las configuraciones, los cambios en cuanto a esa configuración se pueden observar en el sistema.

Cuadro 4.16: Pruebas de Aceptación de la Iteración 5

Iteración 6: Módulo de Periodos Lectivos

A continuación en el cuadro 4.17 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Periodos Lectivos.

Módulo	Modelos	Vistas	Controladores
Periodos Lectivos	periodo_lectivo.rb alquilere.rb prealquilere.rb solvenca.rb	edit.html.erb index.html.erb new.html.erb show.html.erb	periodo_lectivos_controller.rb alquileres_controller.rb solvenca_controller.rb prealquileres_controller.rb

Cuadro 4.17: Esquema del Módulo de Periodos Lectivos con su MVC asociado

La planificación de las Historias de Usuario es la siguiente:

Iteración 6	
Descripción	Desarrollo del Módulo de Periodos Lectivos.
Historias de Usuario	35. Crear un nuevo periodo lectivo. 36. Realizar acciones sobre un periodo lectivo. 37. Buscar un periodo lectivo en particular.
Tiempo Estimado	6 días
Fecha Inicio - Fin	14/04/2011 - 21/04/2011

Para el diseño de las vistas de esta iteración, los periodos lectivos son creados a partir de un formulario donde se especifica el número del periodo y el año correspondiente a dicho periodo. Además se puede elegir el estatus del periodo lectivo (Abierto o Cerrado). Ver figura 4.50



Figura 4.50: Vista de creación de un Periodo Lectivo

Sólo debe existir en un momento determinado un periodo lectivo con estatus Abierto, dicho periodo es el periodo lectivo actual del semestre.

A medida que se vayan creando periodos lectivos, los mismos aparecen en una tabla donde se pueden visualizar todos los periodos lectivos creados con anterioridad. Ver figura 4.51



Figura 4.51: Vista donde aparecen todos los Periodos Lectivos creados

Con respecto a la codificación, para la creación de un periodo lectivo se definió en el controlador un método llamado **create**. Primero, se guarda el valor que se recibe de

la vista en una variable llamada **periodo lectivo**, luego se guarda en una variable llamada **num** el periodo lectivo abierto en el sistema para ser manejado en los alquileres y las solvencias. Después, se guardan todos los valores del periodo lectivo en una variable llamada **número** y posteriormente se realizan comparaciones para verificar primero si se intenta crear un periodo abierto ya existiendo uno en el sistema con ese mismo estatus y segundo se verifica si se intenta crear un periodo cerrado y que ya se encuentra creado en el sistema. En caso de que el periodo haya sido creado con éxito se visualiza un mensaje y luego se direcciona a la vista mostrar. Ver la figura 4.52:

```
def create
  @periodo_lectivo = PeriodoLectivo.new(params[:periodo_lectivo])
  @num = PeriodoLectivo.all(:conditions => ["estatus = 'ABIERTO'"])
  @numero = PeriodoLectivo.all(:conditions => ["ano_periodo = #{@periodo_lectivo.ano_periodo} and num_periodo =
  '#{@periodo_lectivo.num_periodo}' and estatus = '#{@periodo_lectivo.estatus}'"])
  if @num.length>0 and @periodo_lectivo.estatus=="ABIERTO" then
    flash[:notice] = "ERROR: Ya existe un periodo ABIERTO"
    render :action => "new"
  else if @numero.length>0 then
    flash[:notice] = "ERROR: Ya ese periodo fue Cerrado"
    render :action => "new"
  else
    respond_to do |format|
      if @periodo_lectivo.save
        flash[:notice] = "periodo creado exitosamente"
        format.html { redirect_to backend_periodo_lectivos_mostrar_url(@periodo_lectivo) }
        format.xml { render :xml => @periodo_lectivo, :status => :created, :location => @periodo_lectivo }
      else
        format.html { render :action => "new" }
        format.xml { render :xml => @periodo_lectivo.errors, :status => :unprocessable_entity }
      end
    end
  end
end
end
end
```

Figura 4.52: Código para la creación de un Periodo Lectivo.

Para visualizar los periodos lectivos, se manejan en la vista index los atributos correspondientes, en este caso el número de periodo, el año del periodo y el estatus. Luego se controla el acceso a las acciones por parte del administrador y los empleados. Ver la figura 4.53:

```
<table>
  <tr>
    <th>NUMERO DE PERIODO</th>
    <th>ANO DEL PERIODO</th>
    <th>ESTADO</th>
  </tr>
  <% @periodo_lectivos.each do |periodo_lectivo| %>
    <tr>
      <td><%=h periodo_lectivo.num_periodo %></td>
      <td><%=h periodo_lectivo.ano_periodo %></td>
      <td><%=h periodo_lectivo.estatus %></td>
      <td class="list-actions">
        <% if controller.usuario_puede?(usuario_actual, 'backend/periodo_lectivos/', 'show') %>
          <%= link_to 'Mostrar',backend_periodo_lectivos_mostrar_url(periodo_lectivo) %>
        <end%>
        <% if controller.usuario_puede?(usuario_actual, 'backend/periodo_lectivos/', 'edit') %>
          <%= link_to 'Editar',backend_periodo_lectivos_editar_url(periodo_lectivo) %>
        <end%>
        <% if controller.usuario_puede?(usuario_actual, 'backend/periodo_lectivos/', 'destroy') %>
          <%= link_to 'Eliminar',backend_periodo_lectivos_eliminar_url(periodo_lectivo), :confirm => "¿Está seguro que desea Eliminar el Periodo?"%> </td>
        <end%>
      </td>
    </tr>
  <% end %>
</table>
```

Figura 4.53: Parte del Código para visualizar la tabla de Periodos Lectivos.

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración:

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
16	Periodos Lectivos	Verificar que solo pueda existir un periodo lectivo con estatus abierto.	El sistema debe generar un error si se intenta crear un periodo lectivo con estatus abierto, cuando ya se tenga un periodo lectivo abierto.	El sistema generó el error esperado.
17	Periodos Lectivos	Verificar que el buscador por atributo de los periodos lectivos funcione adecuadamente.	Al realizar la búsqueda de un atributo del periodo lectivo, se deben obtener todos los periodos lectivos que coincidan con esa búsqueda.	Se generaron los periodos lectivos indicados después de efectuar la búsqueda de atributos en los mismos.

Cuadro 4.18: Pruebas de Aceptación de la Iteración 6

Iteración 7: Módulo de Autenticación

A continuación en el cuadro 4.19 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Autenticación.

Módulo	Modelos	Vistas	Controladores
Autenticación	usuario.rb rol.rb rolespermiso.rb permiso.rb usuariosrole.rb	cambiar_clave.html.erb index.html.erb inicio.html.erb olvido_clave.html.erb enviar_email.html.erb enviar_nueva_clave.html.erb	application_controller.rb principal_controller.rb sesion_controller.rb alquileres_controller.rb prealquileres_controller.rb solvencias_controller.rb

Cuadro 4.19: Esquema del Módulo de Autenticación con su MVC asociado

La planificación de las Historias de Usuario es la siguiente:

Iteración 7	
Descripción	Creación del Módulo de Autenticación.
Historias de Usuario	38. Cambiar la clave de acceso al sistema. 39. Enviar correo electrónico a los usuarios indicando su clave de acceso al sistema. 40. Tener un enlace de olvido de clave.
Tiempo Estimado	7 días
Fecha Inicio - Fin	22/04/2011 - 18/05/2011

Para la página principal de la Bolsa del Libro, se estableció el mismo diseño que tienen las páginas del personal Administrativo y los estudiantes, con el mismo encabezado y pie.

Es importante destacar que, en dicha página se indica la información correspondiente al préstamo rental de libros. Ver figura 4.54



Figura 4.54: Vista de la página principal de la Bolsa del Libro

En la página de inicio de la Bolsa del Libro se pueden visualizar todas las opciones disponibles que se tienen según sea el usuario autenticado. Ver figura 4.55



Figura 4.55: Vista de la página de inicio desde el Perfil de Administrador

Además se diseñó una vista para que los usuarios puedan efectuar el cambio de su clave de acceso al sistema, para mayor seguridad. Ver figura 4.56

The screenshot shows a web page titled "Bolsa del Libro" with a navigation menu including "Historia", "Servicios", "Normativas", and "Horarios". The page is for an "Administrador" user. The main heading is "Cambio de Clave". Below it, the instruction reads "A continuación introduzca los siguientes datos:". A form box titled "Datos para el cambio de Clave" contains three input fields: "Clave Actual:", "Clave Nueva:", and "Confirmación Clave Nueva:". Each field has a small asterisk to its right. Below the fields is a note "* Campos obligatorios." and an "Aceptar" button. The footer contains the address: "Av. Los Ilustres, Los Chaguaramos, Facultad de Ciencias Universidad Central de Venezuela, Caracas ZP 1040, Apartado Postal 20513, Teléfonos: (0212) 6051159".

Figura 4.56: Vista de la opción de Cambiar Clave

También, se tiene una vista para los usuarios que han olvidado su clave de acceso al sistema, donde se solicita la Cédula de Identidad del usuario y posteriormente se envía un correo al usuario indicando su nueva clave para ingresar al sistema. Ver figura 4.57

The screenshot shows a web page titled "Bolsa del Libro" with a navigation menu including "Historia", "Servicios", "Normativas", and "Horarios". The page is for a "Home" user. The main heading is "Olvido de Clave". Below it, the instruction reads "A continuación introduzca los siguientes datos:". A form box titled "Datos para el olvido de Clave" contains one input field labeled "Cédula:". Below the field is an "Aceptar" button. The footer contains the address: "Av. Los Ilustres, Los Chaguaramos, Facultad de Ciencias Universidad Central de Venezuela, Caracas ZP 1040, Apartado Postal 20513, Teléfonos: (0212) 6051159".

Figura 4.57: Vista de la opción de Olvido de Clave

Para tener el registro de los usuarios que han olvidado su clave de acceso al sistema, se diseñó una vista, donde aparecen en una tabla todos los usuarios que han presionado el enlace de *Olvide mi clave*. Ver figura 4.58

Universidad Central de Venezuela
Facultad de Ciencias
Biblioteca Alonso Gamero
Bolsa del Libro

Historia | Servicios | Normativas | Horarios

Cerrar Sesión

Home-- Administrador

Actualizaciones
Alquileres
Libros
Ejemplares
Prealquileres
Solvencias
Usuarios
Auditorias
Configuraciones
Listados
Periodo Lectivo
Carga Masiva
Aud. Olvido Clave

Auditorias Olvido Clave

Buscar por: Nombre

« Anterior 1 2 3 4 5 6 7 8 9 10 11 Siguiente »

NOMBRE	APELLIDO	CEDULA	ESCUELA	FECHA	
Andreina	Jimenez	18141182	COMPUTACION	11/07/2011	Mostrar
Joselyn	Oviedo	18110941	COMPUTACION	13/07/2011	Mostrar
Pedro	Linares	17145897	FISICA	15/07/2011	Mostrar
Maria	Lopez	15874485	BIOLOGIA	16/07/2011	Mostrar

« Anterior 1 2 3 4 5 6 7 8 9 10 11 Siguiente »

Av. Los Ilustres, Los Chaguaramos, Facultad de Ciencias Universidad Central de Venezuela
Caracas ZP 1040, Apartado Postal 20513
Teléfonos: (0212) 6051159

Figura 4.58: Vista de la Tabla de Auditorias de Olvido de Clave

En cuanto a la codificación, para realizar la autenticación en el sistema, se creó un módulo de Autenticación, donde se definieron todos los métodos necesarios. Los métodos creados, son para permitir que el usuario actual inicie sesión en el sistema y pueda cerrar la sesión en el sistema.

Si el inicio de sesión falla por algún motivo, por ejemplo, que el usuario actual coloque en el cuadro de inicio de sesión, la clave, la cédula de identidad o las letras captcha incorrectas, se direcciona a la página principal del sistema, de lo contrario, se crea una nueva sesión en el sistema. Las letras captcha se implementaron con la gema simple_captcha [3].

Los métodos del módulo de Autenticación se muestran en la figura 4.59

```

module Autenticacion
  ##### iniciar sesion #####

  def inicio_sesion?
    (!!usuario_actual)
  end

  def usuario_actual
    @usuario_actual ||= (logueado_por_sesion) unless @usuario_actual == false
  end

  def usuario_actual=(nuevo_usuario)
    session[:usuario_cedula_actual] = nuevo_usuario ? nuevo_usuario[:cedula] : nil
    @usuario_actual = nuevo_usuario || false
  end

  def logueado_por_sesion
    session[:usuario_cedula_actual]
    self.usuario_actual = Usuario.find_by_cedula(session[:usuario_cedula_actual]) if session[:usuario_cedula_actual]
  end

  ##### Fin iniciar sesion #####

  ##### cerrar sesion #####

  def redirect_to_or_default(default)
    redirect_to(session[:retornar_a] || default)
    session[:retornar_a] = nil
  end

  def desloguear_manteniendo_sesion!
    @usuario_actual = false
    session[:usuario_cedula_actual] = nil
  end

  def desloguear_eliminado_sesion!
    desloguear_manteniendo_sesion!
    reset_session
  end

  ##### Fin cerrar sesion #####

end

def acceso_denegado
  if inicio_sesion?
    redirect_to index_principal_url
    return false
  else
    redirect_to sessions_nuevo_url
  end
end
end

```

Figura 4.59: Métodos del Módulo de Autenticación

En el controlador sesión, se definieron los métodos **crear** y **eliminar**. El método **crear** permite crear una nueva sesión en el sistema, recibiendo como parámetros la cédula de identidad y la clave del usuario actual. Se verifica que la cédula de identidad, la clave y las letras captcha colocadas por el usuario actual sean correctas, para redireccionar a la página de inicio del sistema, de lo contrario se redirecciona a la página principal y se imprime un mensaje de error. El método **eliminar**, permite que el usuario visualice un mensaje cuando cierre la sesión en el sistema. Los métodos crear y eliminar se muestran en la figura 4.60

```

def crear
  @usuario = Usuario.autenticar(params[:cedula], params[:clave], params)
  @exito = (@usuario && simple_captcha_valid? and escuela_pertenece(params[:licenciatura]))

  if @exito
    $usua = params[:cedula]
    self.usuario_actual = @usuario

    redirect_to :controller => "principal", :action => "inicio"
  else
    flash[:error] = "Datos Incorrectos para #{@usuario}"
    redirect_to :controller => "principal"
  end
end

def eliminar
  desloguear_eliminado_session!
  flash[:notice] = "Hasta pronto, ha salido del sistema."
  redirect_to_or_default("/")
end

```

Figura 4.60: Métodos de Autenticación del Controlador Sesión

En el modelo usuario, se definieron todos los métodos necesarios que permiten guardar un nuevo usuario en la Base de Datos, autenticar al usuario actual que intente iniciar sesión en el sistema, verificar que la clave ingresada por el usuario sea válida y encriptar la clave de acceso al sistema del usuario. Los métodos de autenticación del modelo usuario se muestran en la figura 4.61

```

def self.guardar(attr,options={})
  clave_por_defecto = attr[:cedula]
  usuario = new(attr)
  usuario.cedula = attr[:cedula]
  usuario[:clave] = self.encriptar(clave_por_defecto,attr[:cedula])
  usuario.save
  usuario
end

def self.autenticar(cedula, clave,opciones={})
  return nil if (cedula.blank? || clave.blank?)
  u = find_by_cedula(cedula)
  exito = (u && u.clave_valida?(clave))
  exito ? u : false
end

def clave_valida?(clave)
  self.clave.eql?(encriptar(clave))
end

def encriptar(clave,generador=cedula)
  self.class.encriptar(clave,generador)
end

def self.encriptar(clave,generador)
  Digest::SHA1.hexdigest("--1--#{clave}--#{generador}--0--")
end

```

Figura 4.61: Métodos de Autenticación del Modelo Usuario

Se efectuó un método llamado **realizar_cambiar_clave** en el controlador principal, en el cual se solicita al usuario que desea realizar el cambio de la clave de acceso al sistema su clave actual, la nueva clave y la confirmación de la nueva clave.

Luego se verifica si la clave actual ingresada por el usuario es válida y si la clave nueva coincide con la confirmación de la misma y además está compuesta por seis dígitos.

Si el usuario realiza el cambio de clave de manera adecuada, se encripta la nueva clave, se guarda en la Base de Datos y se imprime un mensaje de éxito, en caso contrario, se imprime un mensaje de error.

El método realizar_cambiar_clave se muestra en la figura 4.62.

```
def realizar_cambiar_clave
  clave_actual = params[:clave_actual]
  clave_nueva = params[:clave_nueva]
  confirmacion_clave_nueva = params[:confirmacion_clave_nueva]

  if usuario_actual.clave_valida?(clave_actual)
    if clave_nueva == confirmacion_clave_nueva
      if clave_nueva.size >= 6
        usuario_actual.clave = usuario_actual.encriptar(clave_nueva)
        usuario_actual.save
        flash[:notice] = "Su Clave fue cambiada exitosamente"
        redirect_to(:controller => 'principal', :action => 'inicio')
      else
        flash[:error] = "La clave debe tener más de 6 dígitos"
        redirect_to(:controller => 'principal', :action => 'cambiar_clave')
      end
    else
      flash[:error] = "Su Clave Nueva no coincide con la Clave Nueva de Confirmación"
      redirect_to(:controller => 'principal', :action => 'cambiar_clave')
    end
  else
    flash[:error] = "Su Clave Actual es incorrecta. Intente nuevamente"
    redirect_to(:controller => 'principal', :action => 'cambiar_clave')
  end
end
```

Figura 4.62: Código para realizar el cambio de clave

Se creó un método llamado **olvido_clave** en el controlador principal, donde se verifica si la cédula de identidad introducida por el usuario es válida, para generar una clave aleatoria, encriptarla y almacenarla en la Base de Datos.

Posteriormente se envía un correo electrónico al usuario, indicando su nueva clave de inicio de sesión en el sistema. Luego se crea una nueva entrada en la tabla de Auditorias de Olvido de clave con los datos del usuario y la fecha en la cual el usuario presionó el enlace de *Olvide mi clave*. Ver figura 4.63

```
def olvido_clave
  if params[:cedula].nil?
    render(:layout => 'otro')
    return
  else
    begin
      #raise "Error forzado"
      redirect_to(:layout => 'otro')
      raise "Debe colocar una Cédula" if params[:cedula].size == 0
      flash.delete(:error)
      @clave_plana_aleatoria = (rand * 9999999).ceil # genera la clave aleatoria
      @usuario_encontrado = Usuario.find_by_cedula(params[:cedula])
      raise "La Cédula no se encuentra en el Sistema" if @usuario_encontrado.nil?
      flash.delete(:error)
      @usuario_encontrado.clave = @usuario_encontrado.encriptar(@clave_plana_aleatoria)
      @usuario_encontrado.save
      UsuarioEmail.deliver_enviar_nueva_clave(@usuario_encontrado, @clave_plana_aleatoria)
      flash[:notice] = "Su nueva clave de acceso al Sistema ha sido enviada a su correo"

      AuditoriaCorreo.new( #bitacora que crea una nueva entrada en la tabla aud.olvido clave
        :nombre => @usuario_encontrado.contacto.nombre,
        :apellido => @usuario_encontrado.contacto.apellido,
        :cedula => @usuario_encontrado.cedula,
        :nombre_escuela => @usuario_encontrado.contacto_escuela.nombre_escuela,
        :fecha => Time.now
      ).save
    rescue Exception => e
      puts e, e.backtrace
      flash[:error] = e.to_s #todos los errores caen aqui, los de la excepcion
    end
  end
end
```

Figura 4.63: Código del método de olvido de clave

Además se definieron los perfiles o roles de Usuario, para que una vez iniciada la sesión en el sistema, dependiendo del usuario que haya ingresado, aparezcan ciertas opciones y se restrinjan las acciones que pueden realizar los usuarios de acuerdo a su permisología.

Para definir y aplicar los permisos de todos los perfiles de Usuario (Administrador, Empleado y Estudiante) en el sistema, se creó un arreglo de pares de la forma [controlador, acción] para cada uno de los perfiles, de esta forma se especifican las acciones que pueden realizar los usuarios sobre los controladores que corresponden a las opciones del sistema.

En la figura 4.64, se muestra el código que aplica los permisos de Administrador en el sistema.

```
# APLICAR PERMISOS

r = Rol.find_by_nombre('Administrador') # 9918157
[
  # esto es un arreglo de pares de la forma ['controlador', accion]

  ['backend/actualizaciones/', 'index'],
  ['backend/actualizaciones/', 'show'],
  ['backend/actualizaciones/', 'edit'],

  ['backend/alquileres/', 'index'],
  ['backend/alquileres/', 'show'],
  ['backend/alquileres/', 'edit'],
  ['backend/alquileres/', 'destroy'],
  ['backend/alquileres/', 'comprobante_de_alquiler'],

  ['backend/libros/', '*'],

  ['backend/ejemplares/', '*'],

  ['backend/prealquileres/', 'index'],
  ['backend/prealquileres/', 'show'],
  ['backend/prealquileres/', 'edit'],
  ['backend/prealquileres/', 'destroy'],

  ['backend/solvencias/', '*'],

  ['backend/contactos/', 'index'],
  ['backend/contactos/', 'show'],
  ['backend/contactos/', 'edit'],
  ['backend/contactos/', 'new'],

  ['backend/auditorias/', 'index'],
  ['backend/auditorias/', 'show'],
  ['backend/auditorias/', 'edit'],

  ['backend/auditoria_correos/', 'index'],

  ['backend/configuraciones/', 'index'],
  ['backend/configuraciones/', 'show'],
  ['backend/configuraciones/', 'edit'],

  ['backend/listados/', 'solicita'],
  ['backend/listados/', 'adquiridos'],
  ['backend/listados/', 'alquilados'],
  ['backend/listados/', 'cartelera'],
  ['backend/listados/', 'deudores'],
  ['backend/listados/', 'excluidos'],
  ['backend/listados/', 'inventario'],
  ['backend/listados/', 'multados'],
  ['backend/listados/', 'portitulo'],
  ['backend/listados/', 'ejemp_excluidos'],
  ['backend/listados/', 'most_ejemplar'],

  ['backend/periodo_lectivos/', '*'],

  ['backend/carga_masiva/', 'index'],
  ['backend/carga_masiva/', 'guarda'],

].each{|par|
  r.permisos << Permiso.new(:controlador => par[0], :accion => par[1])
}
```

Figura 4.64: Código que aplica los permisos de Administrador en el sistema

Además, se realizó el método `usuario_puede` en el controlador `application`, para permitir al usuario realizar una acción sobre un controlador, dependiendo del rol (Administrador, Estudiante o Empleado) que tenga, siempre y cuando tenga el permiso para hacerlo. Ver figura 4.65

```
def usuario_puede?(usuario, controlador, accion = nil)
  begin
    usuario.roles.each{|rol|
      rol.permisos.each{|permiso|
        if(permiso.controlador == controlador && accion == nil)
          return true
        end
        if(permiso.controlador == controlador && permiso.accion == '*')
          return true
        end
        if(permiso.controlador == controlador && permiso.accion == accion)
          return true
        end
      }
    }
  rescue Exception => e
    puts e, e.backtrace
    return false
  end
end
```

Figura 4.65: Código del método `usuario_puede`

Para que en el sistema aparecieran los enlaces correspondientes a las acciones que puede realizar cada usuario y los botones de las opciones que tiene disponibles dependiendo de su permisología, se verifica si el usuario actual tiene permiso sobre el controlador para imprimir el botón correspondiente a una opción del sistema y si el usuario actual puede realizar la acción sobre el controlador dependiendo de los permisos aplicados, se imprime el enlace de la acción. En la figura 3.60 se muestra el condicional para verificar si el usuario puede visualizar el botón Alquileres y el enlace de la opción Mostrar. Ver figura 4.66

```
<% if controller.usuario_puede?(usuario_actual, 'backend/alquileres/') %>
  <%= button_to "Alquileres",url_for(:controller => "backend/alquileres/", :action => :index) %>
<% end %>

<% if controller.usuario_puede?(usuario_actual, 'backend/alquileres/', 'show') %>
<u><%= link_to "Mostrar",backend_alquileres_mostrar_url(alquilere, :id_alquiler => alquiler.id_alquiler) %></u> &nbsp;&nbsp;&nbsp;
<% end %>
```

Figura 4.66: Código del condicional para verificar si el usuario puede visualizar el botón Alquileres y el enlace de la opción Mostrar

Por último, para que el usuario pueda visualizar en el sistema el nombre de su perfil correspondiente, se creó un condicional que se colocó en los layouts necesarios, donde se verifica el rol (Administrador, Estudiante o Empleado) del usuario actual y se imprime el rol identificado. Ver figura 4.67


```

<td class="contenido-td" colspan=3><!-- inicio TD barra superior -->
  <div id=home>
    <u><%= link_to 'Home', url_for(:controller => "principal/", :action => :inicio)%></u>
  </div>
  <%= begin %>
    <%= if usuario_actual.roles.collect{|r|r.nombre}.include? "Administrador" %>
      -- Administrador
    <%= elsif usuario_actual.roles.collect{|r|r.nombre}.include? "Empleado" %>
      -- Empleado
    <%= elsif usuario_actual.roles.collect{|r|r.nombre}.include? "Estudiante" %>
      -- Estudiante
    <%= end %>
    <%= rescue %>
    <%= end %>
  </td><!-- fin TD barra superior -->

```

Figura 4.67: Código del condicional para verificar el perfil del usuario actual e imprimirlo en pantalla

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración:

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
18	Autenticación	Comprobar que un usuario pueda ingresar al sistema colocando como clave su cédula de identidad.	Una vez que el usuario rellene todos los campos del cuadro de inicio de sesión correctamente, se debe permitir el acceso al sistema.	El usuario pudo ingresar al sistema.
19	Autenticación	Verificar que un usuario no pueda ingresar al sistema si rellena los campos de forma incorrecta.	Cuando el usuario intente ingresar al sistema colocando un dato erróneo y si le falta rellenar algún campo, no se permite el acceso al mismo.	El usuario no pudo ingresar al sistema si colocó los datos erróneos en el inicio de sesión o le faltó rellenar algún campo.
20	Autenticación	Verificar el correcto funcionamiento del enlace de cerrar sesión.	Si el usuario presiona el enlace de cerrar sesión, el sistema debe direccionar al usuario a la página principal.	Al cerrar sesión en el sistema, se direcciona a la página principal.
21	Autenticación	Comprobar el correcto funcionamiento del enlace de cambiar clave.	Si el usuario intenta cambiar la clave, se debe efectuar el cambio satisfactoriamente siempre y cuando los datos proporcionados sean correctos. De lo contrario se deben generar errores de acuerdo a la falla que haya tenido el usuario al intentar cambiar su clave. Luego el usuario debe poder ingresar al sistema con su nueva clave.	El sistema realiza el cambio de clave de manera satisfactoria, generando errores si algún campo es erróneo y permitiendo el acceso a la aplicación con la nueva clave.

Cuadro 4.20: Pruebas de Aceptación de la Iteración 7

Iteración 8: Funcionalidad de Actualizaciones

A continuación en el cuadro 4.21 se mencionan los Modelos, las Vistas y los Controladores asociados con la Funcionalidad de Actualizaciones.

Funcionalidad	Modelos	Vistas	Controladores
Actualizaciones	actualizacione.rb	edit.html.erb index.html.erb	actualizaciones_controller.rb

Cuadro 4.21: Esquema de la Funcionalidad de Actualizaciones con su MVC asociado

La planificación de las Historias de Usuario es la siguiente:

Iteración 8	
Descripción	Creación de la funcionalidad de Actualizaciones.
Historias de Usuario	41. Editar la información de la página principal. 42. Editar la información del correo electrónico que es enviado a los estudiantes para indicar su clave de acceso al sistema. 43. Visualizar el listado de libros agotados en la página principal. 44. Visualizar el listado de libros disponibles en la Bolsa del Libro en la página principal.
Tiempo Estimado	8 días
Fecha Inicio - Fin	19/05/2011 - 29/05/2011

En cuanto al diseño, para que el personal Administrativo pueda realizar las actualizaciones de información del préstamo rental de libros en la página principal, se realizó una vista en la cual se colocó un editor de texto enriquecido Javascript/AJAX llamado Nic-Edit [4] que permite hacer la modificación del texto que aparece en la página principal de la Bolsa del Libro. Ver figura 4.68



Figura 4.68: Vista de la página que permite realizar la actualización de la información de la página principal

Los listados de libros agotados y de catálogo de libros disponibles en la Bolsa del libro que aparecen en la página principal, siguen el mismo formato de las listas de solicitudes de alquiler y solicitudes de solvencias que se muestran en los módulos de Alquiler y Solvencias respectivamente. Ver figura 4.69



Universidad Central de Venezuela
Facultad de Ciencias
Biblioteca Alonso Gamero
Bolsa del Libro

Historia | Servicios | Normativas | Horarios

Principal

Libros Agotados

TITULO	AUTOR	EDITORIAL	COSTO DE ALQUILER
Chile y la isla de Pascua (Country Guide) (Spanish Edition)	Planet	Lonely Planet	10.2
Los Tres Pasos: UN Pelato Imprescindible Para Definir Nuestra Verdadera Pasión Laboral (Spanish Edition)	Arnie Warren	Empresa Activa	50.9

Av. Los Ilustres, Los Chaguaramos, Facultad de Ciencias Universidad Central de Venezuela
Caracas ZP 1040, Apartado Postal 20513
Teléfonos: (0212) 6051159

Figura 4.69: Listado de libros agotados en la Bolsa del Libro

En cuanto a la codificación, para hacer uso de un editor de texto enriquecido en el sistema, se utilizó el editor de texto enriquecido Javascript/AJAX NicEdit. Se colocó el código indicado en la página oficial de NicEdit para cargar el editor, que permite agregar el editor de texto enriquecido NicEdit en la vista donde se añada dicho código. En ese código se define un script que utiliza la función `nicEditors.allTextAreas`, la cual permite reemplazar el campo de tipo text area que se encuentre en la vista donde se coloca el código por un editor NicEdit [4]. En este caso, el código se insertó en la vista de editar una actualización.

El editor NicEdit se utiliza para modificar la información de la página principal del sistema y la información del correo electrónico que se envía a los estudiantes para indicarles su clave de inicio de sesión en el sistema. En la figura 4.70 se observa el código para generar el editor de texto enriquecido NicEdit.

```
<script src="http://js.nicedit.com/nicEdit-latest.js" type="text/javascript"></script>
<script type="text/javascript">bklLib.onDomLoaded(nicEditors.allTextAreas);</script>

<script type="text/javascript" src="http://js.nicedit.com/nicEdit-latest.js"></script> <script type="text/javascript">
//
bklLib.onDomLoaded(function() { nicEditors.allTextAreas() });
//]]&gt;
&lt;/script&gt;</pre>
</div>
<div data-bbox="170 790 876 808" data-label="Caption">
<p>Figura 4.70: Código para generar el editor de texto enriquecido NicEdit</p>
</div>
<div data-bbox="141 826 909 896" data-label="Text">
<p>Para visualizar los libros que ya están agotados en el sistema, se creó un método llamado <b>libros_agotados</b> en donde primero se maneja el buscador de la vista, luego se maneja en la variable <b>config</b> el número de elementos que se desea visualizar y por último, en una variable llamada <b>libros</b> se guardan todos los libros cuya cantidad de ejemplares</p>
</div>
```

es igual a cero. Ver figura 4.71

```
def libros_agotados
  @config = Configuracione.first(:conditions => ["nombre = 'num_paginas'"])

  @libros = Libro.paginate(:page => params[:page], :per_page => @config.valor, :conditions => ["cant_ejem = 0"])
  render(:layout => 'otro')
end
```

Figura 4.71: Código del método libros_agotados

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración:

No.	Funcionalidad	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
22	Actualizaciones	Comprobar que los cambios realizados al hacer una actualización de información se muestren en la página principal.	Una vez que el personal administrativo edite la información de la página principal haciendo uso del editor de texto enriquecido, dicha información debe actualizarse en la página principal.	La información de la página principal se pudo actualizar de manera satisfactoria.
23	Actualizaciones	Comprobar si el listado de libros agotados es correcto.	Después de que un libro no posea ejemplares disponibles para el alquiler, ese libro es el que debe aparecer en el listado de libros agotados.	Efectivamente cuando un libro se quedó sin ejemplares disponibles para alquilar, el libro apareció en el listado de libros agotados.
24	Actualizaciones	Verificar si el listado de libros disponibles es correcto.	Se deben mostrar en la página principal todos los libros que aparecen en el catálogo de libros del personal administrativo, ya que son los libros que han sido creados en el sistema.	En la página principal, al presionar el enlace de Catálogo de Libros, se muestran todos los libros de la Bolsa del Libro.

Cuadro 4.22: Pruebas de Aceptación de la Iteración 8

Iteración 9: Módulo de Auditorias

A continuación en el cuadro 4.23 se mencionan los Modelos, las Vistas y los Controladores asociados con el Módulo de Auditorias.

Módulo	Modelo(s)	Vista(s)	Controlador(es)
Auditorias	auditoria.rb	buscar.rjs edit.html.erb index.html.erb new.html.erb show.html.erb	auditorias_controller.rb

Cuadro 4.23: Esquema del Módulo de Auditorias con su MVC asociado

La planificación de las Historias de Usuario es la siguiente:

Iteración 9	
Descripción	Desarrollo del Módulo de Auditorias.
Historias de Usuario	45. Visualizar los elementos eliminados en el sistema. 46. Buscar un elemento eliminado en el módulo Auditorias.
Tiempo Estimado	4 días
Fecha Inicio - Fin	01/06/2011 - 06/06/2011

Para el diseño, en este módulo, se realizó una vista donde se muestra una tabla en la cual aparecen todos los elementos que han sido eliminados del sistema por alguna razón (alquileres, libros, ejemplares, solvencias). Ver figura 4.72



Figura 4.72: Vista del Módulo de Auditorias

Con respecto a la codificación, para visualizar la tabla auditorias, en la vista index se manejan atributos, que corresponden a los elementos que han sido eliminados del sistema. Los elementos eliminados pueden ser: Alquileres, Solvencias, Libros y Ejemplares. Ver la figura 4.73:

```

<%= @auditorias.each do |auditoria| %>
  <tr>
    <td><%=h auditoria.id_elemento %></td>
    <td><%=h auditoria.id_sub_elemento %></td>
    <td><%=h auditoria.tipo_elemento %></td>
    <td><%=h auditoria.causa %></td>
    <td><%=h auditoria.fecha %></td>
    <td class="list-actions">
      <%= if controller.usuario_puede?(usuario_actual, 'backend/auditorias/', 'show') %>
      <%= link_to 'Mostrar',backend_auditorias_mostrar_url(auditoria) %>
      <%= end %>

      <%= if controller.usuario_puede?(usuario_actual, 'backend/auditorias/', 'edit') %>
      <%= link_to 'Editar',backend_auditorias_editar_url(auditoria) %>
      <%= end %>

      <%= if controller.usuario_puede?(usuario_actual, 'backend/auditorias/', 'destroy') %>
      <%= link_to 'Eliminar',backend_auditorias_eliminar_url(auditoria), :confirm => "¿Está seguro que desea Eliminar?"%></td>
      <%= end %>
    </td>
  </tr>
<%= end %>

```

Figura 4.73: Parte del código que genera la tabla de las Auditorias.

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración:

No.	Módulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
25	Auditorias	Comprobar el correcto funcionamiento del módulo de Auditorias.	Una vez que el Personal Administrativo elimine un elemento en el sistema (alquiler, solvencia, libro, ejemplar), ese elemento se debe incorporar en la tabla de auditorias.	Después de eliminar los elementos, los mismos aparecieron en la tabla de auditorias.
26	Auditorias	Verificar el funcionamiento del buscador.	Cuando se realice la búsqueda de los elementos eliminados, deben aparecer solo los elementos que fueron especificados en el buscador.	El sistema muestra los elementos buscados de manera adecuada, una vez seleccionado el criterio de búsqueda.

Cuadro 4.24: Pruebas de Aceptación de la Iteración 9

Iteración 10: Funcionalidad de Carga Masiva

A continuación en el cuadro 4.25 se mencionan los Modelos, las Vistas y los Controladores asociados con la Funcionalidad de Carga Masiva.

Funcionalidad	Modelos	Vistas	Controladores
Carga Masiva	carga.rb	guarda.html.erb index.html.erb	carga_masiva_controller.rb

Cuadro 4.25: Esquema de la Funcionalidad de Carga Masiva con su MVC asociado

La planificación de las Historias de Usuario es la siguiente:

Iteración 10	
Descripción	Desarrollo de la funcionalidad de Carga Masiva.
Historias de Usuario	47. Realizar una Carga Masiva de estudiantes.
Tiempo Estimado	5 días
Fecha Inicio - Fin	17/06/11 - 23/06/11

Para el diseño de la funcionalidad de Carga Masiva, se realizó una interfaz sencilla, donde el personal Administrativo de la Bolsa del Libro puede realizar la carga del archivo de los estudiantes en formato CSV (del inglés comma-separated values) para un semestre determinado, presionando el botón *Examinar* para adjuntar el archivo en el sistema y luego oprimiendo el botón *Procesar*, para que se llene automáticamente la tabla con los estudiantes de un semestre en particular en el Módulo de Usuarios. Ver figura 4.74



Figura 4.74: Vista principal de la Funcionalidad de Carga Masiva

Con respecto a la codificación, para guardar semestralmente el archivo de los estudiantes en formato CSV (del inglés comma-separated values) en el sistema, se creó en el controlador un método llamado **Guarda**, donde primero se verifica que en la ruta que se indica por la vista se tenga un nombre de archivo, luego en una variable llamada **ruta** se guarda el archivo que se desea cargar y después se abre y se lee por líneas, donde se va a determinar que los campos van a ser separados por punto y coma.

Posteriormente, se crea un nuevo usuario y un nuevo contacto para guardar el campo cédula y sucesivamente se van cargando en la tabla contactos el apellido, nombre, correo electrónico, escuela y el estatus del estudiante (que en el momento de la carga siempre va a ser solvente).

Por último a esos usuarios ya cargados se les envía un correo electrónico informándoles que su clave de acceso al sistema es su cédula de identidad.

En caso de cargar un usuario que ya se encuentre en el sistema se arroja un mensaje de error, de lo contrario, el mensaje de carga es exitoso. Ver las figuras 4.75 y 4.76:

```
def guarda
  @error=0
  #@tipo = params[:archivo]
  if params[:archivo].nil? or params[:archivo].empty?
    flash[:notice] = "DEBE INTRODUCIR UN NOMBRE DE ARCHIVO"
  else
    #@ruta = "/home/joselyn/Escritorio"
    @ruta = Carga.save(params[:archivo])
    #@cadena = File.open(@ruta.to_s+"/"+#@tipo.to_s).read rescue ""
    cadena = File.open(@ruta.to_s).read rescue ""
    puts líneas = cadena.split("\n")
    líneas.each do |línea|
      campos = línea.split(";")
      unless campos.nil?
        u = Usuario.new
        c = Contacto.new
        u.cedula = campos[1]
        c.cedula = campos[1]
        c.apellido = campos[2]
        c.nombre = campos[3]
        c.e_mail = campos[4]
      end
    end
  end
end
```

Figura 4.75: Código que guarda el archivo de los estudiantes en el sistema (Parte I).

```
@esc = Escuela.first(:conditions => ["nombre_escuela= '#{campos[0]}'"])
c.id_escuela = @esc.id_escuela
c.id_status = 1
c.numlibros = 0
u.clave = campos[1]
u.contacto = c
@conta = Contacto.all(:conditions => ["cedula= '#{campos[1]}'"])
@usuario = Usuario.all(:conditions => ["cedula= '#{campos[1]}'"])
if @conta.length==0 and @usuario.length==0
  @usu=Usuario.guardar({:cedula => u.cedula})
  c.id_usuario = @usu.id_usuario
  c.save
  UsuarioEmail.deliver_enviar_email(c.e_mail)
  @usurol = Usuariosrole.new(:id_usuario => @usu.id_usuario, :id_rol => 3)
  @usurol.save
else
  @error=1
end
end

end

if @error==1 then
  flash[:notice] = "ERROR: EL ARCHIVO QUE INTENTA CARGAR TIENE CLAVES DUPLICADAS"
else
  flash[:notice] = "CARGA EXITOSA"
end
end
```

Figura 4.76: Código que guarda el archivo de los estudiantes en el sistema (Parte II).

A continuación se muestran las pruebas de aceptación realizadas en la presente iteración:

No.	Funcionalidad	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido
27	Carga Masiva	Comprobar el correcto funcionamiento de la funcionalidad de Carga Masiva.	Después de adjuntar el archivo con los estudiantes para un semestre determinado en el sistema, se debe llenar automáticamente la tabla de usuarios en el módulo de Usuarios.	Efectivamente después de adjuntar el archivo con los estudiantes para un semestre determinado en el sistema, se llenó automáticamente la tabla de usuarios en el módulo de Usuarios.

Cuadro 4.26: Pruebas de Aceptación de la Iteración 10

■ Producción

En esta fase, se realizaron una serie de pruebas, antes de colocar cada funcionalidad en el ambiente de desarrollo del cliente, por lo cual se aplicaron **Pruebas Funcionales** por parte de las desarrolladoras para probar que el sistema, cumpliera con las funciones específicas para el cual fue creado.

Estas pruebas también son llamadas **Pruebas de caja negra**, ya que se enfocaron en determinar cómo se generaban las respuestas del sistema, haciendo un análisis de los datos de entrada y de los datos de salida.

Conclusiones

El presente Trabajo Especial de Grado tuvo como finalidad la automatización del Servicio de Préstamo Rental de Libros de la Facultad de Ciencias, mejorando la realización de los servicios que allí se ofrecen.

En el desarrollo se utilizó un lenguaje dinámico e interpretado (Ruby) sobre un framework destacado en la actualidad (Rails), lo que permitió que la realización del Trabajo Especial de Grado simplificara la solución presentada, haciendo uso de las convenciones manejadas por el framework. Además, se utilizó una adaptación de la metodología ágil “Programación Extrema” (XP), la cual nos facilitó trabajar de forma organizada al permitirnos agrupar los requerimientos en un conjunto de iteraciones que se fueron desarrollando progresivamente.

Es importante destacar que, el modelado consistió en crear nueve Módulos, los cuales son: Alquiler, Solvencias, Libros y Ejemplares, Listados, Configuraciones, Periodos Lectivos, Usuarios, Autenticación y Auditorias; además de dos funcionalidades como lo son: Actualizaciones y Carga Masiva, representando todos estos componentes un paso muy importante dentro del proyecto Bolsa del Libro, ya que lo acerca cada vez más a su puesta en producción, y a su consolidación como una herramienta fundamental para el Servicio de Préstamo Rental de Libros de la Facultad de Ciencias.

El sistema realizado en este Trabajo Especial de Grado fue el resultado del estudio exhaustivo de los diversos servicios que presta la Bolsa del Libro. Esto se logró gracias a la constante comunicación del equipo de desarrollo con el personal administrativo de la Bolsa del Libro, a fin de garantizar que los requerimientos solicitados correspondieran exactamente con las funcionalidades reflejadas en el sistema final.

El desarrollo de la aplicación, se hizo siguiendo un modelo abierto y flexible que permitiese realizar mantenimientos correctivos, adaptativos y/o perfectivos, sin que se vean afectadas en gran medida las funcionalidades y estructuras ya existentes. De esta manera, el sistema está desarrollado en módulos y pensado en que va a evolucionar con los cambios que sean necesarios.

Para finalizar podemos afirmar que el desarrollo de la aplicación, permitió automatizar los principales servicios ofrecidos en la Bolsa del Libro, logrando satisfacer las necesidades actuales de la misma, permitiendo sentar las bases para desarrollos futuros sobre la aplicación y dejando un valioso aporte a nuestra casa de estudios.

Recomendaciones

Un vez culminado el desarrollo de la Aplicación Web, es de suma importancia dejar sentadas las bases que permitan el diseño de nuevas funcionalidades y la realización de futuras mejoras de las funcionalidades ya creadas, por lo cual se sugiere:

- Añadir nuevas funcionalidades al Módulo de Pre-Alquiler, permitiendo que en el mismo se efectúe la asignación de los ejemplares solicitados por los estudiantes, tomando en cuenta un estudio socio-económico que facilite dicha asignación.
- Implementar una funcionalidad que permita al usuario Administrador la edición de la información que aparece en los enlaces de *Historia, Servicios, Normativas y Horarios*.
- La implementación de un nuevo buscador en el Módulo de Libros y Ejemplares, para que se tenga otra alternativa de descarga de Libros a través de la Web.
- Agregar nuevas funcionalidades al Módulo de Pre-Alquiler y Alquiler, para contemplar en el Sistema Automatizado el caso cuando un estudiante se encuentra suspendido del préstamo rental de Libros.
- Implementar el Módulo de Venta de Libros.
- Realizar las validaciones pertinentes en cuanto al préstamo de Libros a otros usuarios (personal administrativo, profesores, entre otros).
- Generación de documentos PDFs a partir de la búsqueda realizada en cada listado.
- Realizar pruebas de Stress a la Aplicación Web.

Adicionalmente, después de la realización de este Trabajo, consideramos importante incluir el uso de los Diagramas de Actividades como artefactos adicionales, que aportan información relevante a la metodología XP.

Bibliografía

- [1] Facultad de ciencias universidad central de venezuela biblioteca alonso gamero. <http://biblioteca.ciens.ucv.ve/>. (Consultado el 27 de Julio de 2010).
- [2] Mysql 5.0 reference manual. <http://dev.mysql.com/doc/refman/5.0/en/what-is-mysql.html>. (Consultado el 03 de Agosto de 2010).
- [3] Simple captcha. http://expressica.com/simple_captcha/. (Consultado el 15 de Marzo de 2011).
- [4] What is nicedit? <http://nicedit.com>, 2008. (Consultado el 04 de Marzo de 2011).
- [5] ¿qué es aws? <http://aws.amazon.com/what-is-aws/>, 2010. (Consultado el 05 de Agosto de 2010).
- [6] Tutorial: muestra del servicio web de amazon. [http://msdn.microsoft.com/es-es/library/ms498501\(office.12\).aspx](http://msdn.microsoft.com/es-es/library/ms498501(office.12).aspx), 2010. (Consultado el 30 de Septiembre de 2010).
- [7] Gómez Andrés and Ania Ignacio. *Introducción a la Computación*. Cengage Learning, first edition, December 2008.
- [8] Shneiderman Ben and Plaisant Catherine. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison Wesley, fifth edition, March 2009.
- [9] Ediger Brad. *Avanced Rails*. O'Reilly Media, first edition, December 2007.
- [10] Fowler Chad. *Rails Recipes*. Pragmatic Bookshelf, 2006.
- [11] Flanagan David and Matsumoto Yukihiro. *The Ruby Programming Language*. O'Reilly Media, first edition, January 2008.
- [12] Samoza Gabriel. Cómo adjuntar archivos a un modelo en rails. <http://viarails.wordpress.com/2009/11/29/como-adjuntar-archivos-a-un-modelo-en-rails/>, 2010. (Consultado el 05 de Febrero de 2011).
- [13] Beck Kent and Andres Cynthia. *Extreme Programming Explained: Embrace Change*. Addison Wesley, second edition, November 2004.
- [14] Richardson Leonard and Ruby Sam. *RESTful Web Services*. O'Reilly Media, first edition, May 2007.

- [15] Hernández Lewis. Desarrollo de un sistema de software orientado a objetos para una bolsa del libro. T.E.G., Octubre 1998.
- [16] Araujo María and Pereira Cidalía. Desarrollo de aplicaciones de comercio electrónico usando software libre. caso de estudio: Bolsa del libro de la facultad de ciencias de la ucv. T.E.G., Septiembre 2007.
- [17] Tuati Marcela and García Francisco. Sistema de información de la bolsa del libro de la facultad de ciencias (ucv). T.E.G., Abril 1998.
- [18] Carrizalez Marlene. Bienvenida a la cohorte 2009 facultad de ciencias – ucv. Presentación de Bienvenida, Septiembre 2009.
- [19] Carrizalez Marlene. Bolsa del libro. Tríptico de la Bolsa del Libro, 2009.
- [20] Pressman Roger S. *Ingeniería del Software. Un enfoque práctico*. McGraw-Hill Interamericana, sixth edition, 2006.
- [21] Ruby Travis, Swicegood. *Pragmatic Version Control*. Pragmatic Bookshelf, 2008.
- [22] Matsumoto Yukihiro. *Ruby in a Nutshell*. O'Reilly Media, November 2001.

Apéndice

Apéndice 1: Historias de Usuario

Número: 1	Nombre: Visualizar la Aplicación Web.
Prioridad: Alta	Tiempo Estimado: 5 días
Descripción: Permitir al personal administrativo de la Bolsa del libro la visualización de la aplicación web.	
Número: 2	Nombre: Visualizar los Pre-Alquileres y Alquileres en el sistema.
Prioridad: Alta	Tiempo Estimado: 4 días
Descripción: Permitir al personal administrativo la visualización de los alquileres realizados y a los estudiantes realizar la solicitud de Pre-alquiler y Alquiler de libros.	
Número: 3	Nombre: Buscar los libros que el estudiante desea pre-alquilar y alquilar.
Prioridad: Alta	Tiempo Estimado: 3 días
Descripción: Tener un buscador que facilite a los estudiantes la búsqueda de los libros que desean alquilar o prealquilar.	
Número: 4	Nombre: Seleccionar los libros a pre-alquilar y alquilar.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Tener una opción que permita a los estudiantes seleccionar los libros que deseen pre-alquilar y alquilar.	
Número: 5	Nombre: Guardar un alquiler en el módulo de Alquiler.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Permitir que los alquileres realizados por los estudiantes se guarden en el sistema.	
Número: 6	Nombre: Descargar el comprobante de alquiler en formato PDF.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro y a los estudiantes la descarga del comprobante de alquiler en formato PDF.	
Número: 7	Nombre: Visualizar el estado de solvencia de los estudiantes en el módulo de Pre-Alquiler y Alquiler.
Prioridad: Alta	Tiempo Estimado: 3 días
Descripción: Permitir a los estudiantes la visualización de su estado de solvencia con la Bolsa del Libro.	

Número: 8	Nombre: Registrar la devolución de un libro en el módulo de Alquiler.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro registrar en el sistema los libros devueltos por los estudiantes.	
Número: 9	Nombre: Visualizar los diferentes estados de los usuarios.
Prioridad: Alta	Tiempo Estimado: 3 días
Descripción: Permitir al personal administrativo visualizar el estado de solvencia de los estudiantes con la Bolsa del Libro.	
Número: 10	Nombre: Visualizar los Libros que se encuentran en la Bolsa del Libro.
Prioridad: Alta	Tiempo Estimado: 4 días
Descripción: Permitir tanto a los estudiantes como al personal administrativo visualizar los libros que se ofertan en la Bolsa del Libro.	
Número: 11	Nombre: Incorporar al sistema un libro disponible en Amazon.
Prioridad: Alta	Tiempo Estimado: 5 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la incorporación de un libro disponible en Amazon a través de un buscador.	
Número: 12	Nombre: Asignar una imagen a un libro.
Prioridad: Alta	Tiempo Estimado: 3 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro asignar o cambiar la imagen a un libro.	
Número: 13	Nombre: Asignar a un libro sus ejemplares correspondientes.
Prioridad: Alta	Tiempo Estimado: 3 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la asignación de los ejemplares que le corresponden a un determinado libro.	
Número: 14	Nombre: Visualizar los Ejemplares que existen en el sistema.
Prioridad: Alta	Tiempo Estimado: 4 días
Descripción: Permitir al personal administrativo visualizar todos los ejemplares con su respectiva información de un libro en particular.	
Número: 15	Nombre: Realizar acciones sobre los libros y ejemplares.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Permitir al administrador de la Bolsa del Libro realizar las acciones de Mostrar, Editar y Eliminar sobre los libros y ejemplares que se encuentran en el sistema.	
Número: 16	Nombre: Visualizar los campos obligatorios para la creación de libros y ejemplares.
Prioridad: Alta	Tiempo Estimado: 1 día
Descripción: Permitir al personal administrativo de la Bolsa del Libro la visualización de los campos obligatorios al momento de crear un libro y un ejemplar en el sistema.	

Número: 17	Nombre: Buscar un libro.
Prioridad: Alta	Tiempo Estimado: 1 día
Descripción: Permitir tanto al personal administrativo de la Bolsa del Libro como a los estudiantes buscar un libro que se encuentra en el sistema a través de un buscador.	

Número: 18	Nombre: Buscar un ejemplar.
Prioridad: Alta	Tiempo Estimado: 1 día
Descripción: Permitir al personal administrativo de la Bolsa del Libro buscar un ejemplar que se encuentra en el sistema a través de un buscador.	

Número: 19	Nombre: Seleccionar el tipo de solvencia requerida.
Prioridad: Alta	Tiempo Estimado: 1 día
Descripción: Permitir al estudiante la selección de la solvencia que requiera.	

Número: 20	Nombre: Visualizar mensaje de estado de solvencia al solicitar una solvencia.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Permitir al estudiante visualizar su estado de solvencia al solicitar una solvencia.	

Número: 21	Nombre: Visualizar las solicitudes de solvencia.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la visualización de las solicitudes de solvencia realizadas por los estudiantes.	

Número: 22	Nombre: Realizar acciones con las solicitudes de solvencia.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Permitir al administrador de la Bolsa del Libro realizar las acciones de Mostrar, Editar y Eliminar sobre una solvencia.	

Número: 23	Nombre: Buscar la solvencia deseada.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Tener un buscador que facilite al personal administrativo de la Bolsa del Libro la búsqueda de las solvencias realizadas por los estudiantes.	

Número: 24	Nombre: Descargar el comprobante de solvencia en formato PDF.
Prioridad: Alta	Tiempo Estimado: 3 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro y a los estudiantes la descarga del comprobante de solvencia en formato PDF.	

Número: 25	Nombre: Seleccionar un tipo de listado.
Prioridad: Alta	Tiempo Estimado: 1 día
Descripción: Permitir al personal administrativo de la Bolsa del libro la selección del listado que requiera.	

Número: 26	Nombre: Visualizar el listado seleccionado.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del libro la visualización del listado seleccionado.	

Número: 27	Nombre: Buscar por atributos en cada listado.
Prioridad: Alta	Tiempo Estimado: 2 días
Descripción: Facilitar al personal administrativo de la Bolsa del Libro un buscador que permita realizar búsquedas por atributos de un listado particular.	

Número: 28	Nombre: Descargar el listado requerido en formato PDF.
Prioridad: Alta	Tiempo Estimado: 3 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la descarga del listado que requiera en formato PDF.	

Número: 29	Nombre: Descargar el listado de libros alquilados en formato EXCEL.
Prioridad: Alta	Tiempo Estimado: 3 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la descarga del listado de libros alquilados en formato EXCEL.	

Número: 30	Nombre: Buscar usuarios.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: Tener un buscador que facilite al personal administrativo de la Bolsa del Libro la búsqueda de usuarios por nombre y por cédula de identidad.	

Número: 31	Nombre: Crear un nuevo usuario.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la creación de un nuevo usuario.	

Número: 32	Nombre: Realizar acciones sobre un usuario.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: Permitir al administrador de la Bolsa del Libro realizar las acciones de Mostrar y Editar sobre un usuario.	

Número: 33	Nombre: Visualizar las Configuraciones.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la visualización de las configuraciones que se encuentran validadas en el sistema.	

Número: 34	Nombre: Realizar cambios sobre los valores de las configuraciones del sistema.
Prioridad: Media	Tiempo Estimado: 3 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro cambiar los valores de las configuraciones que se encuentran validadas en el sistema.	

Número: 35	Nombre: Crear un nuevo periodo lectivo.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la creación de un periodo lectivo.	

Número: 36	Nombre: Realizar acciones sobre un periodo lectivo.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: Permitir al administrador de la Bolsa del Libro realizar las acciones de Mostrar, Editar y Eliminar sobre un periodo lectivo.	

Número: 37	Nombre: Buscar un periodo lectivo en particular.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: Tener un buscador que facilite al personal administrativo de la Bolsa del Libro la búsqueda de periodos lectivos.	

Número: 38	Nombre: Cambiar la clave de acceso al sistema.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: Permitir a los usuarios cambiar su clave de acceso al sistema.	

Número: 39	Nombre: Enviar correo electrónico a los usuarios indicando su clave de acceso al sistema.
Prioridad: Media	Tiempo Estimado: 3 días
Descripción: Enviar un correo electrónico a los usuarios para notificarles su clave de acceso al sistema.	

Número: 40	Nombre: Tener un enlace de olvido de clave.
Prioridad: Media	Tiempo Estimado: 2 días
Descripción: En la página principal tener un enlace para los usuarios que hayan olvidado su clave de acceso al sistema.	

Número: 41	Nombre: Editar la información de la página principal.
Prioridad: Baja	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro actualizar la información referente al servicio de préstamo rental en la página principal.	

Número: 42	Nombre: Editar la información del correo electrónico que es enviado a los estudiantes para indicar su clave de acceso al sistema.
Prioridad: Baja	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro actualizar la información el correo electrónico que es enviado a los estudiantes para indicar su clave de acceso al sistema.	

Número: 43	Nombre: Visualizar el listado de libros agotados en la página principal.
Prioridad: Baja	Tiempo Estimado: 2 días
Descripción: Permitir la visualización del listado de libros agotados en la página principal.	

Número: 44	Nombre: Visualizar el listado de libros disponibles en la Bolsa del Libro en la página principal.
Prioridad: Baja	Tiempo Estimado: 2 días
Descripción: Permitir la visualización del listado de los libros disponibles en la Bolsa del Libro en la página principal.	

Número: 45	Nombre: Visualizar los elementos eliminados en el sistema.
Prioridad: Baja	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro la visualización de las auditorias del sistema.	

Número: 46	Nombre: Buscar un elemento eliminado en el módulo de Auditorias.
Prioridad: Baja	Tiempo Estimado: 2 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro buscar un elemento que ha sido eliminado del sistema.	

Número: 47	Nombre: Realizar una Carga Masiva de estudiantes.
Prioridad: Baja	Tiempo Estimado: 5 días
Descripción: Permitir al personal administrativo de la Bolsa del Libro cargar el listado de estudiantes que le suministra Control de Estudios semestralmente en el sistema.	

Apéndice 2: Entrevistas Realizadas

No.	Fecha	Puntos a tratar
1	15/06/10	<ul style="list-style-type: none"> - Entender el funcionamiento del Servicio del Préstamo Rental de Libros de manera general. - Obtener los primeros requerimientos: generación de documento para solvencia de libros, generación de listado semestral de estudiantes deudores, consultar información de los estudiantes (por Cédula de Identidad y Nombre).
2	21/06/10	<ul style="list-style-type: none"> - Identificar los procesos principales del Servicio del Préstamo Rental de Libros: alquiler de libros, solicitud de solvencias, generación de listados. - Conversar acerca de los requerimientos principales: incorporación y desincorporación de libros, consulta de libros vía web, cargar listado de estudiantes proveniente de Control de Estudios.
3	23/06/10	<ul style="list-style-type: none"> - Investigar los tipos de Solvencias (Retiro Temporal, Retiro Total, Cambio de Escuela, Cambio de Facultad, Grado). - Investigar los tipos de listados (deudores, libros alquilados, libros excluidos, libros existentes por título, libros adquiridos). - Investigar los tipos de búsqueda (cota, título, cédula de identidad, apellido, nombre, escuela). - Conversar acerca de los sistemas anteriores creados para la automatización de la Bolsa del Libro (antecedentes).
4	07/07/10	<ul style="list-style-type: none"> - Corrección de los diagramas de actividades de los procesos involucrados en la Bolsa del Libro. - Recoger material informativo acerca de la Bolsa del Libro.
5	09/07/10	<ul style="list-style-type: none"> - Corrección de los diagramas de actividades de los procesos involucrados en la Bolsa del Libro. - Entendimiento del flujo de trabajo asociado a cada proceso.
6	12/07/10	<ul style="list-style-type: none"> - Levantamiento de la información necesaria para la creación de las primeras historias de usuario: consulta del listado de libros de alquiler, autenticación de estudiantes sólo para alquilar libros no para realizar la consulta de los mismos, realizar pre-alquiler de libros antes del alquiler, selección del tipo de solvencia a solicitar, registrar la devolución del libro, realizar la incorporación y desincorporación de libros, realizar búsquedas de libros y estudiantes.
7	21/07/10	<ul style="list-style-type: none"> - Aclarar dudas acerca de las Tesis anteriores para la automatización de la Bolsa del libro. - Aclarar inquietudes referentes al material bibliográfico de la Bolsa del Libro.
8	23/07/10	<ul style="list-style-type: none"> - Recabar información acerca de la Base de Datos actual utilizada en la Bolsa del Libro (Lista de deudores y Multados, Listado de libros existentes, préstamo de libros). - Establecer los atributos de las tablas de la Base de Datos para el Sistema propuesto. - Aclarar dudas acerca de la Base de Datos. - Obtención de material informativo en digital de la Bolsa del Libro (documentos y presentaciones).
9	15/09/10	<ul style="list-style-type: none"> - Creación del resto de las historias de usuario: generación de documento PDF de comprobante de alquiler, documento PDF de constancia de solvencia y documento PDF para cada tipo de listado, edición de la información de libros y ejemplares, realizar búsquedas de alquileres, libros, ejemplares, solvencias y usuarios, actualización del listado de libros agotados y el listado de libros disponibles, entre otras.

No.	Fecha	Puntos a tratar
10	29/09/10	- Aclarar dudas acerca de las historias de usuario.
11	14/10/10	- Enseñar las primeras interfaces del sistema propuesto. - Correcciones de las interfaces y la información correspondiente a la Bolsa del Libro.
12	03/11/10	- Correcciones de las primeras funcionalidades del sistema propuesto. - Aclarar dudas acerca del flujo de trabajo involucrado en los procesos principales de la Bolsa del Libro. - Correcciones de la presentación del Seminario.
13	10/11/10	- Aclarar dudas acerca de los procesos que intervienen en la Bolsa del Libro.
14	17/11/10	- Correcciones del documento de Seminario.
15	08/12/10	- Enseñar el sistema propuesto con nuevas funcionalidades: selección de los libros a alquilar desde el Perfil de Estudiante, creación y edición de libros desde el Perfil de Administrador, entre otras. - Correcciones del sistema.
16	27/01/11	- Enseñar las funcionalidades del Módulo de Pre-Alquiler y Alquiler y el Módulo de Libros y Ejemplares del sistema propuesto. - Correcciones del Módulo de Pre-Alquiler y Alquiler y del Módulo de Libros y Ejemplares. - Correcciones de la información que aparece en los enlaces de Historia, Servicios, Normativa y Horarios.
17	16/02/11	- Mostrar avances de la Aplicación Web. - Correcciones de los Módulos creados: Alquiler, Libros y Ejemplares.
18	24/02/11	- Enseñar las funcionalidades del Módulo de Solvencias. - Correcciones del Módulo de Solvencias.
19	16/03/11	- Enseñar funcionalidades del Módulo de Listados. - Correcciones del Módulo de Listados.
20	04/04/11	- Correcciones y cambios sobre el Módulo de Libros y Ejemplares.
21	21/04/11	- Mostrar funcionalidades de los Módulos de Usuarios, Configuraciones y Periodos Lectivos. - Correcciones de los Módulos de Usuarios, Configuraciones y Periodos Lectivos.
22	03/05/11	- Enseñar funcionalidades del Módulo de Autenticación desde la página de la Aplicación Web en Internet. - Mostrar la funcionalidad de Actualizaciones desde la página de la Aplicación Web en Internet. - Mostrar las funcionalidades de todos los Módulos creados hasta el momento desde la página de la Aplicación Web en Internet. - Correcciones de los módulos correspondientes.
23	17/05/11	- Enseñar el resto de las funcionalidades de Actualizaciones desde la página de la Aplicación Web en Internet. - Mostrar las funcionalidades del Módulo de Auditorias. - Realizar correcciones de la funcionalidad de Actualizaciones y el módulo de Auditorias.
24	24/05/11	- Correcciones de la permisología para el Perfil de Usuario Empleado. - Determinar que acciones puede realizar un Empleado en el sistema para cada uno de los Módulos creados. - Realizar correcciones de manera general de todos los Módulos del sistema.
25	25/05/11	- Correcciones y cambios en el Módulo de Alquiler.

No.	Fecha	Puntos a tratar
26	08/06/11	<ul style="list-style-type: none">- Mostrar la Aplicación Web con algunos de los cambios efectuados, tomando en cuenta las correcciones realizadas en la reunión anterior.- Correcciones en la funcionalidad de Actualizaciones.- Correcciones del menú de opciones del sistema desde el perfil administrativo en general.- Correcciones de la información del enlace de Historia.
27	15/06/11	<ul style="list-style-type: none">- Buscar correcciones de la información que aparece en el mensaje que se muestra al estudiante cuando solicita una solvencia.- Buscar correcciones de la información del correo electrónico que es enviado a los estudiantes para informarle su clave de inicio de sesión en el sistema.
28	23/06/11	<ul style="list-style-type: none">- Observaciones del sistema en general.
29	06/07/11	<ul style="list-style-type: none">- Mostrar algunas correcciones realizadas en el sistema que faltaban revisar.
30	29/07/11	<ul style="list-style-type: none">- Mostrar las últimas funcionalidades que se realizaron en el sistema.