UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 4

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

Título: Sistema para la Gestión de préstamos de implementos deportivos en la Universidad de las Ciencias Informáticas.

Autor:

Ariel Nuviola Rodríguez

Tutora:

Ing. Ana María Álvarez

La Habana 2012

"Año 54 de la Revolución"

Declaración de autoría

Declaración de Autoría.

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

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

_____

Ariel Nuviola Rodríguez

Autor

_____

Ing. Ana María Álvarez

Tutora

Dedicatoria

A mi madre por dedicar su vida en mi formación y por esperar este momento para ver realizado este sueño que es de todos.

A mi hermanita Ainad por haber estado ahí para escucharme y por servirme de guía. Lo mejor del mundo para ti y mi sobrinito Malcolm.

A todas esas personas que forman parte de mi vida y que sienten tanto como yo este éxito.

Agradecimientos

A mi madre, a mi hermana, a mi padrastro y a mis tías Yamilé y Sara por saber guiarme por el camino correcto, por convertirme en la persona que soy, por estar y quererme en todo momento, por el apoyo que me dan, por tener la confianza plena en mí, por ser los capaces de mantenerme en pie y por darme las fuerzas para seguir adelante.

A mi amigo Jose Luis Barrueta por el apoyo que me ha dado incondicionalmente durante el desarrollo del trabajo, y por estar siempre ahí.

A mi tía Nora que siempre ha estado pendiente de mí.

A mi tutora Ana María por haberme brindado toda la ayuda posible y por confiar en mí.

A los profesores que de una forma u otra han contribuido a mi formación profesional.

A mis amigos de la Universidad por compartir estos años juntos, Edilberto, Danir, Alberto, Pepo, Luis Ángel, a los de mi grupo.

A todas las personas que no menciono y que de una forma u otra me ayudaron a llegar donde estoy.

Resumen

En el mundo de hoy se percibe un avance vertiginoso de las tecnologías por lo que se imposibilita hablar de desarrollo, eficiencia y calidad sin disponer de estos medios para llevar a cabo cualquier actividad, el área deportiva no queda desapercibida de esta evolución, ya que para realizar prestaciones de implementos a los estudiantes se necesita de un servicio de gestión que garantice el control forma óptima, de todos los implementos deportivos. De ahí parte el tema de esta investigación “Sistema para la Gestión de préstamos de implementos deportivos en la Universidad de las Ciencias Informáticas”. Para el desarrollo de la aplicación se hizo uso del lenguaje de programación PHP5, el Framework CodeIgniter que utiliza la arquitectura Modelo-Vista-Controlador (MVC) y el ExtJS, el Sistema Gestor de Bases de Datos PostgreSQL y la metodología de Desarrollo XP. Además; para especificar, construir y documentar el sistema se hace uso del Lenguaje Unificado de Modelado (UML) y Visual Paradigm para el modelado. Esta aplicación facilitará la cátedra de cultura física de la UCI contar con un sistema para la gestión de préstamos de implementos deportivos y satisfacer las demandas de sus clientes, de forma rápida, sencilla, proporcionando una alta calidad en sus servicios.

Índice

Introducción

A partir del avance de las tecnologías, hoy en día es imposible hablar de desarrollo, eficiencia y calidad sin disponer de los medios tecnológicos necesarios para llevar a cabo cualquier tarea. La esfera deportiva no queda desapercibida de la evolución en los medios de comunicación y cómo estos contribuyen a la realización de cualquier tarea con rapidez y confiabilidad.

La extensión masiva del deporte a lo largo y ancho de nuestra Isla, para promover el desarrollo y el esparcimiento de la actividad física en niños y adultos, tiene como objetivos fundamentales el perfeccionamiento del sistema de participación deportiva así como la profundización en aspectos técnicos y organizativos.

En la Universidad de las Ciencias Informáticas (UCI) existe una dirección de deportes encargada de organizar y dirigir actividades deportivas para toda la comunidad universitaria. En la misma se cuenta con implementos deportivos variados, los que son registrados y controlados administrativamente, con la finalidad de garantizar la recreación sana, centrada en las actividades lúdicas y en ambientes naturales para vincular al joven universitario con el medio.

En dicha universidad se tiene como propósito perfeccionar el sistema de participación deportiva de la comunidad universitaria y profundizar en los aspectos técnicos y organizativos, estrechamente relacionados al logro de una calidad superior en los servicios que se le prestan a los estudiantes y al aprovechamiento extremo de cada una de las instalaciones deportivas. En la misma se brinda el servicio de préstamos de implementos deportivos, para lo cual se usa un registro en papel como medio de control de los recursos.

Los préstamos se basan en las políticas establecidas por el departamento de Cultura Física de la UCI, lo cual permite a la comunidad universitaria solicitar un implemento de uno a dos días. En los mismos no se incluyen aquellos implementos que son únicos o de difícil reposición, estos solo pueden ser utilizados en el gimnasio.

En observaciones realizadas a este proceso se evidencia un ascenso diario, lo que refleja la demanda que tiene la utilización de dichos medios por estudiantes y profesores. Esto provoca que sea engorroso el registro en papel,

puesto que no siempre se puede controlar con efectividad quienes poseen cada implemento, resulta difícil identificar la demora en la devolución de cada préstamo y no garantiza una vía para avisar, de forma anticipada, la fecha de devolución de los recursos.

Retomando las ideas anteriores se evidencia la carencia de la gestión de la información que garantice el registro y aviso anticipado para la devolución de los recursos en calidad de préstamos. Debido a esto se han generado pérdidas y afectación en la vida útil de los implementos deportivos, lo que evidencia las deficiencias en el control sistemático de estos recursos y además entorpece el desempeño de la comunidad universitaria en las actividades deportivas.

Por lo antes expuesto se presenta como **problema a resolver**: ¿Cómo facilitar la gestión de préstamos de implementos deportivos en el gimnasio de Cultura Física de la UCI? Teniendo como **objeto de estudio:** Los procesos para la gestión de préstamos, enmarcado en el **campo de acción**: Los procesos para la gestión de préstamos de implementos deportivos en la UCI.

Para dar solución al problema a resolver se define como **objetivo general:** Desarrollar un sistema de gestión que permita controlar el servicio de préstamos de implementos deportivos del gimnasio de Cultura Física de la UCI.

Para darle cumplimiento al objetivo general y darle solución a la situación problemática planteada, se proponen los siguientes **objetivos específicos:**

- ❖ Elaborar el marco teórico de la investigación.
- ❖ Efectuar el diseño del sistema de gestión para los préstamos de implementos deportivos.
- ❖ Realizar la implementación de dicho sistema.
- ❖ Comprobar el funcionamiento de la aplicación.

Tareas a cumplir:

- ❖ Seleccionar las tecnologías y herramientas a emplea en el desarrollo del software.
- ❖ Realizar un análisis de los sistemas informáticos existentes utilizados a nivel nacional e internacional destinados a la gestión de préstamos.
- ❖ Definir las principales funcionalidades a desarrollar y estimar el esfuerzo requerido para ello.
- ❖ Seleccionar patrones que posibiliten dar solución a los problemas de diseño.

- ❖ Implementar las funcionalidades propuestas.
- ❖ Comprobar el funcionamiento de la aplicación.

Para alcanzar los resultados propuestos en la investigación se utilizaron los siguientes métodos:

Del nivel empírico

Entrevistas: para obtener información sobre las necesidades del gimnasio de Cultura Física de la UCI, en cuanto a la gestión de préstamos de implementos deportivos, con el propósito de conocer las funcionalidades que debe tener el sistema a desarrollar. Este método se emplea al inicio y durante todo el transcurso del trabajo, pues a medida que se desarrolla el sistema se analiza si se cumplen las expectativas requeridas.

Del nivel teórico

Análisis y síntesis: para recopilar toda la información teórico conceptual y las herramientas relacionadas con la gestión de préstamos de implementos deportivos, con el propósito de extraer los elementos más importantes que se relacionan con el objeto de estudio.

Histórico-Lógico: permite estudiar el progreso de los portales Web dedicados a la gestión de préstamos en Cuba y el resto del mundo.

El contenido del presente trabajo de diploma está estructurado de la siguiente manera:

Capítulo 1: Fundamentos teóricos. Se hace referencia a los elementos teóricos que constituyen la base de la investigación realizada. Se exponen los resultados del estudio del estado del arte y se describen las soluciones similares existentes. Se presentan las herramientas, tecnologías y metodología a utilizar fundamentándose su selección teniendo como base el análisis exhaustivo desarrollado.

Capítulo 2: Características del sistema. Se presentan las características del sistema propuesto, describiendo cómo debe funcionar, sus potencialidades y características esenciales. Se hace alusión a los conceptos que rodean el negocio y se utiliza como apoyo auxiliar un modelo conceptual que los refleja

sintetizadamente. Se presentan las historias de usuario capturadas y la planificación realizada a partir de estas.

Capítulo 3: Diseño, implementación y pruebas del sistema. Se hace referencia a los elementos que conforman el diseño del sistema a construir. Se presenta el diagrama de clases persistentes, el modelo de datos, pruebas de caja negra y caja blanca así como los resultados de las pruebas de aceptación.

Capítulo1: Fundamentos teóricos

1.1 Introducción

El capítulo comprende un breve estudio de algunas de las aplicaciones relacionadas con el proceso para la gestión de préstamos tanto en el ámbito nacional como internacional, así como un estudio acerca de las tecnologías y software usados en la actualidad, donde se abordan aspectos como características, ventajas y desventajas, de este estudio se seleccionan las que se ajustan más al medio circundante y se conjugan mejor con las necesidades actuales de desarrollo.

1.2 Conceptos asociados al dominio del problema

Gestión

Del latín *gestío*, el concepto de gestión hace referencia a la acción y al efecto de gestionar o administrar. Es un proceso integral que se realiza por medio de las actividades que asume la organización para lograr los propósitos y metas previamente establecidos. Este proceso se da mediante la planeación, que a su vez le permitirá desarrollar la misión y visión institucional (1).

La gestión se interpreta como una cadena continua de acciones definidas en el Proceso Administrativo y representada en el ciclo PHVA (Planear, Hacer, Verificar y Actuar), actualmente conocido como PEEA, (Planear, Ejecutar, Estudiar y Actuar). Este ciclo busca hacer que se alcancen los objetivos y resultados esperados, optimizar los recursos disponibles y, mediante la autoevaluación y autocontrol, proponer acciones que permitan a la entidad el mejoramiento continuo (2).

Sistema de gestión

Un sistema de gestión es una estructura probada para la gestión y mejora continua de las políticas, los procedimientos y procesos de la organización.
Las mejores empresas funcionan como unidades completas con una visión compartida. Ello engloba la información compartida, evaluaciones comparativas, trabajo en equipo y un funcionamiento acorde con los más rigurosos principios de calidad y del medioambiente.

Un sistema de gestión ayuda a lograr los objetivos de la organización mediante una serie de estrategias, que incluyen la optimización de procesos, el enfoque centrado en la gestión y el pensamiento disciplinado.

Estos sistemas son de gran importancia ya que muchas empresas hoy en día se enfrentan a muchos retos, significativos, entre ellos:

- ✓ Rentabilidad
- ✓ Competitividad
- ✓ Globalización
- ✓ Velocidad de los cambios
- ✓ Capacidad de adaptación
- ✓ Crecimiento
- ✓ Tecnología

Equilibrar estos y otros requisitos empresariales puede constituir un proceso difícil y desalentador. Es aquí donde entran en juego los sistemas de gestión, al permitir aprovechar y desarrollar el potencial existente en la organización.

La implementación de un sistema de gestión eficaz puede ayudar a:

- ✓ Gestionar los riesgos sociales, medioambientales y financieros
- ✓ Mejorar la efectividad operativa
- ✓ Reducir costos
- ✓ Aumentar la satisfacción de clientes y partes interesadas
- ✓ Proteger la marca y la reputación
- ✓ Lograr mejoras continuas
- ✓ Potenciar la innovación
- ✓ Eliminar las barreras al comercio

El uso de un sistema de gestión probado le permite renovar constantemente su objetivo, sus estrategias, sus operaciones y niveles de servicio (3).

Sistema de gestión de préstamos

Para los estudios de este trabajo se hace necesario profundizar en diferentes términos que permiten establecer que es el sistema de gestión de préstamos.

Partiendo de los referentes anteriores el autor de este trabajo considera oportuno plantear y asumir desde la experiencia como estudiante e investigador que el Sistema de gestión de préstamos es el servicio de procesos

esenciales, relacionados entre sí, con nexos lógicos en sus acciones que proporcionan de manera eficiente la administración y agilización en la entrega de recursos materiales con la posibilidad de establecer un control de su entrega y devolución.

1.3 Soluciones similares

En múltiples revisiones bibliográficas y consultas realizadas en internet se aprecian trabajos internacionales referidos a la gestión de préstamos: el sistema de gestión de préstamos para computadoras portátiles y el sistema de gestión de bibliotecas, a diferencia del ámbito nacional, donde se desconoce de alguna herramienta con esta finalidad. A continuación se hace una breve descripción de los sistemas estudiados.

1.3.1 Sistema de gestión de préstamos para computadores portátiles

El Centro de gestión administrativa y fortalecimiento empresarial de Colombia ofrece a los aprendices, instructores y personal administrativo el préstamo de computadores portátiles; el objetivo de este servicio es facilitar la consulta a cada uno de ellos. Este servicio es administrado por una persona que es la encargada de hacer el préstamo y tener un control de cada uno de los computadores. Ante esta situación se vio la necesidad de crear un sistema de gestión de préstamos de portátiles el cual facilite este proceso y a la vez lo haga más rápido, tanto como para el administrador y los usuarios. El objetivo general es ofrecer, al administrador del servicio de préstamo de portátiles, un mejor manejo en cuanto a la información del usuario y el estado de los computadores (4).

1.3.2 Sistema integrado de gestión de bibliotecas

Koha-UNLP es un sistema integrado de gestión de bibliotecas utilizado actualmente en varias universidades de Argentina, el cual permite administrar los procesos bibliotecarios y gestionar los servicios a los usuarios. Este sistema está desarrollado sobre una plataforma de software libre, basado en la versión 2.0.0 del proyecto original KOHA y se encuentra totalmente en español.

Koha-UNLP incluye todas las funciones requeridas para la gestión una

biblioteca de cualquier tipo o tamaño ya que su adaptabilidad permite su uso en bibliotecas universitarias, públicas, escolares, populares o especializadas:

- ✓ Crear el catálogo de la biblioteca, ingresando los registros desde cero o bien importando registros bibliográficos de otros sistemas.
- ✓ Importar datos de usuarios de otros sistemas.
- ✓ Generar e imprimir etiquetas con código de barras.
- ✓ Administrar el registro de usuarios, generar e imprimir los carnets con códigos de barra.
- ✓ Gestionar el sistema de circulación: préstamos en las diferentes modalidades, renovaciones y devoluciones, imprimiendo los formularios correspondientes.
- ✓ Administrar esquema de sanciones según los usos de cada biblioteca.
- ✓ Autogenerar los números de inventario.
- ✓ Obtener estadísticas de préstamos, usuarios y material bibliográfico.
- ✓ Impresión de listados y reportes y la exportación de los datos a una planilla de cálculo.
- ✓ Enviar mail de aviso de préstamos vencidos, de reservas disponibles para retirar o reservas en lista de espera (5).

Luego de realizar una investigación sobre los diferentes sistemas utilizados para la gestión de préstamos, se decide desarrollar una aplicación que dé respuesta a la situación problémica planteada. Para ello es sugerente tomar como ejemplos algunas de las funcionalidades que brindan los sistemas analizados anteriormente.

Dentro de ellas se puede destacar la impresión de listados y reportes y la exportación de los datos a una planilla de cálculo, de modo que favorece a que el usuario que interactúa con la aplicación se mantenga al tanto de los préstamos de implementos. El envío de correo de aviso de préstamos vencidos, de reservas disponibles para retirar o reservas en lista de espera. La gestión del sistema de circulación: préstamos en las diferentes modalidades, renovaciones y devoluciones, imprimiendo los formularios correspondientes.

1.4 Tendencias y tecnologías actuales

1.4.1 Metodologías de desarrollo

Una metodología de desarrollo se refiere a un marco que se utiliza para estructurar, planificar y controlar el proceso. No existe una metodología universal, las características de cada proyecto exigen que el proceso sea flexible y configurable. La metodología en el transcurso del desarrollo de la creación de una aplicación informática define quién debe hacer qué, cuándo y cómo hacerlo (6).

En la actualidad existen una variada cantidad de metodologías de desarrollo de aplicaciones informáticas, clasificándose en dos grupos, tradicionales y ágiles. Las metodologías tradicionales se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán.

Dentro de las metodologías tradicionales se encuentra RUP[1] como la más conocida.

Las metodologías ágiles se centran principalmente en el factor humano o en la creación del sistema. Este tipo de metodologías le dan mayor importancia al individuo, a la colaboración con el cliente y al desarrollo incremental de la aplicación informática con iteraciones muy cortas. Está probada su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente el tiempo de desarrollo pero manteniendo la calidad requerida. Entre las más conocidas se encuentran XP[2], DSDM[3] y OpenUp.

A continuación se hace un análisis sobre las metodologías más usadas actualmente para escoger la que más se adecúe al proyecto:

[1]Rational Unified Process

[2]Extreme Programming

[3]Dynamic Systems Development Method

RUP

RUP, es una de las metodologías más generales y usadas que existen en la actualidad, pues se puede adaptar a cualquier proyecto. El proceso propuesto por RUP para la creación de aplicaciones informáticas posee tres características fundamentales:

- ✓ Dirigido por los casos de usos.
- ✓ Centrado en la arquitectura.
- ✓ Iterativo e incremental.

RUP consta de cuatro fases o etapas:

1. Comienzo o Inicio: Descripción del negocio y delimitación del proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.
2. Elaboración: Definición de la arquitectura del sistema y obtención de una aplicación ejecutable que responde a los casos de uso que la comprometen.
3. Construcción: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario.
4. Transición: Aplicación lista para su instalación en condiciones reales. Puede implicar reparación de errores (7).

Cada una de estas etapas se desarrolla sobre un ciclo de iteración que consiste en reproducir el ciclo de vida en cascada a menor escala. En estas iteraciones se agrupan todas las actividades de los flujos de trabajo, luego de finalizar cada iteración se obtendrá un incremento del producto.

La particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo de aplicaciones informáticas (7).

DSDM

Esta metodología se basa en programación rápida de aplicaciones (RAD). La estructura de DSDM está guiada por nueve principios:

1. El involucramiento del usuario es imperativo.
2. Los equipos de DSDM deben tener el poder de tomar decisiones.
3. El foco está puesto en la entrega frecuente de productos.
4. La conformidad con los propósitos del negocio es el criterio esencial para la aceptación de los entregables.
5. El desarrollo iterativo e incremental es necesario para converger hacia una correcta solución del negocio.
6. Todos los cambios durante el desarrollo son reversibles.
7. Los requerimientos están especificados a un alto nivel.
8. Las pruebas están integrada a través del ciclo de vida.
9. Un enfoque colaborativo y cooperativo entre todos los interesados es esencial (8).

Una de las características principales de esta metodología es que el desarrollo es iterativo e incremental y los clientes cooperan directamente con el equipo de trabajo.

DSDM propone cinco fases de estudio:

1. Estudio de factibilidad.
2. Estudio del negocio.
3. Modelado funcional.
4. Diseño y construcción.
5. Implantación.

Respecto a los roles que trabaja, especifica tres tipos:

1. Visionario: Asegura que se satisfagan las necesidades del negocio.
2. Usuario embajador: Brinda el conocimiento del negocio y define los requerimientos funcionales y no funcionales de la aplicación.
3. Coordinador técnico: Encargado de mantener la arquitectura, verificar la existencia de los componentes y el cumplimiento de los estándares técnicos (8).

DSMD ha tenido un refinamiento continuo y es ahora una de las metodologías más aceptadas en el mercado (8).

XP

La metodología XP permite establecer iteraciones cortas y apropiadas para un entorno caracterizado por requerimientos cambiantes, su objetivo principal es tener una nueva versión a cada instante, mostrarlo al cliente, ver lo que opina y seguir programando, tener una comunicación fluida con el cliente y el usuario final, por lo que define una manera de reunir a clientes y programadores en un equipo, firmemente integrado con condiciones de trabajo que promueven la comunicación y solución de un problema. Se ha clasificado como una metodología ágil, ya que plantea aumentar constantemente la velocidad del proyecto. XP funciona mejor para pequeños equipos, a diferencia de RUP, que es muy óptima para un equipo grande de desarrolladores, esto sin lugar a dudas lo pone en desventaja. La misma define cuatro fases fundamentales: exploración, planificación, implementación y pruebas. Sus principios son simplicidad, comunicación, retroalimentación (feedback), coraje y respeto (9).

OpenUp/Basic

OpenUP/Basic es un marco de trabajo de procesos de desarrollo de aplicaciones informáticas de código abierto. Es un proceso modelo y extensible, dirigido a gestión y desarrollo de proyectos de aplicaciones informáticas basados en desarrollo iterativo, ágil e incremental; y es aplicable a un conjunto amplio de plataformas y aplicaciones de desarrollo.

Este proceso de desarrollo unificado está basado en RUP, desarrollado por IBM y reconocido mundialmente como uno de los procesos de desarrollo de aplicaciones informáticas de mayor calidad.

Permite un abordaje ágil al proceso de desarrollo de aplicaciones informáticas, con sólo proveer un conjunto simplificado de contenidos, fundamentalmente relacionados con orientación, productos de trabajo, roles, y tareas.

OpenUP/Basic es un proceso interactivo de desarrollo de aplicaciones informáticas simplificado, completo y extensible. Es un proceso para pequeños equipos de desarrollo que valoran los beneficios de la colaboración y de los involucrados con el resultado del proyecto, por encima de formalidades innecesarias.

Está caracterizado por cuatro principios básicos interrelacionados:

- ✓ Colaboración para alinear los intereses y un entendimiento compartido.
- ✓ Balance para confrontar las prioridades, es decir, necesidades y costos técnicos para maximizar el valor para los stakeholders.
- ✓ Enfoque en articular la arquitectura para facilitar la colaboración técnica, reducir los riesgos y minimizar excesos y trabajo extra.
- ✓ Evolución continua para reducir riesgos, demostrar resultados y obtener retroalimentación de los clientes (10).

Procura un equilibrio entre las necesidades de los involucrados con los resultados del proyecto y los costos técnicos, con el fin de maximizar el valor de los involucrados y las guías del proceso de desarrollo. Desarrolla un ciclo de vida interactivo que mitiga el riesgo a tiempo y ofrece demostrar resultados en curso al cliente del proyecto (10).

Fundamentación de la metodología

Luego de un estudio realizado se decidió que la metodología XP es la óptima a utilizar, ya que es: 1) empleada para proyectos de corto plazo, 2) con un equipo de trabajo pequeño, 3) propone una realimentación continua entre el cliente y el equipo de desarrollo, 4) el modelo de 40 horas semanales que define para no trabajar horas extras y 5) la propiedad colectiva del código. Dada las condiciones, facilidades que brinda y la idea de desarrollo que se tiene de la solución, se acordó que sus características se asocian más al tipo de proyecto que se lleva a cabo durante la construcción de la solución informática. Hace énfasis que la comunicación y satisfacción del cliente es lo principal.

Además según Billy Reinoso en una conferencia sobre Metodologías ágiles de desarrollo de software, plantea que XP ha demostrado ser la metodología ágil que más estudios dispone y la de mayor número de artículos escritos (11).

1.4.2 Herramientas CASE

El desarrollo de sistemas que permitan incrementar la productividad y control de calidad en cualquier proceso de desarrollo de aplicaciones informáticas ha ido en aumento en los últimos años. La Ingeniería de Software Asistida por

Computadora (CASE por sus siglas en inglés) reemplaza al lápiz y al papel por la computadora para transformar la actividad de creación de aplicaciones informáticas en un proceso automatizado. Esta tecnología contribuye a elevar la productividad y la calidad en los sistemas (12). A continuación se hace un análisis sobre las herramientas CASE:

Rational Rose Enterprise

IBM Rational Rose Enterprise proporciona un lenguaje de modelado común que permite crear con mayor velocidad aplicaciones informáticas de calidad. Entre sus características principales se encuentra que es compatible con UML. Soporta patrones Analysis, ANSI C++, Rose J y Visual C++, Enterprise JavaBeans 2.0, permite la ingeniería directa e inversa para construcciones comunes en Java 1.5. Incluye un complemento de modelado web, que proporciona el modelado para el desarrollo de aplicaciones web. Esta herramienta resulta de gran utilidad para los desarrolladores de proyectos por lograr cubrir todo el ciclo de vida del proceso desde su fase inicial hasta que termina (13).

Rational Rose posibilita establecer una trazabilidad entre los modelos. Cada rol tiene su propia vista de arquitectura, utilizando un lenguaje común para comprender y comunicar la estructura y funcionalidad del sistema en construcción.

Sin embargo, cuando estas características la convierten en una herramienta muy eficaz, su particularidad de ser propietaria, la hace ser rechazada por muchas empresas y desarrolladores de software.

Visual Paradigm 8.0

La Herramienta CASE Visual Paradigm utiliza UML. Soporta el ciclo de vida completo del desarrollo de una aplicación informática, desde la fase de análisis hasta el despliegue del mismo, es una herramienta libre que permite el modelado de varios lenguajes de programación y otras tecnologías de forma fácil y asequible. Soporta todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Entre sus más recientes características se incluyen el modelado colaborativo con el

sistema de control de versiones CVS[4]y Subversion, así como la interoperabilidad con modelos UML2 (meta modelos UML 2.x para plataforma Eclipse) a través de XML de intercambio de metadatos (XMI o XML Metadata Interchange) (14).

Fundamentación de la herramienta seleccionada

Luego de un análisis de las herramientas existentes para modelar sistemas, se decidió trabajar con el Visual Paradigm 8.0 para UML por su robustez, usabilidad y portabilidad. Además por ser una herramienta que se conoce bastante en la universidad, se cuenta con la licencia para su utilización y soporta el ciclo completo del proceso de desarrollo de software. Utiliza UML como lenguaje de modelado, que tiene como propósito visualizar, especificar, construir y documentar proyectos de software.

1.4.3 Framework

Un framework es una estructura de soporte definida a partir de la cual un proyecto de software puede ser organizado y/o desarrollado. Típicamente, un framework incluye soporte de programas, bibliotecas y un lenguaje interpretado para ayudar a desarrollar y unir los diferentes componentes de un proyecto (15).

Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones.

Existen una variada cantidad de frameworks que son destinados solamente para la creación de aplicaciones web. Dentro de los más usados y conocidos se encuentran ASP.NET, Symfony, CodeIgniter, entre muchos otros. A continuación se describen algunos de ellos:

ASP.NET

ASP[5] es una tecnología de Microsoft del tipo lado del servidor para páginas web generadas dinámicamente. ASP.NET es la versión mejorada de ASP. Uno

[4]Concurrent Versions System

[5]Active Server Pages

de los objetivos principales de ASP.NET es aprovechar toda la capacidad del CLR[6], el CLR compila en algún punto todos los códigos de aplicaciones en códigos naturales de máquina. En ASP.NET el código se separa de la lógica de la aplicación, brinda la posibilidad de escoger el lenguaje que desee el programador, por defecto trae integrado C# y VB.NET.

Otra característica importante es que el framework .NET tiene integrado librerías de clases que son comunes en toda la plataforma .NET, esto permite crear aplicaciones con un considerable ahorro de líneas de código. ASP.NET por las múltiples funciones que tiene es más lento que la mayoría de los frameworks de desarrollo web, pero sin lugar a dudas el mayor inconveniente que presenta, es su condición de propietario (16).

Symfony

Symfony es un framework diseñado para optimizar el desarrollo de aplicaciones web, debido a todas las características que presenta. Se encuentra desarrollado completamente con PHP 5. Ha sido utilizado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony separa la lógica del negocio, la lógica del servidor y la presentación de la aplicación (17).

Reduce enormemente el tiempo de desarrollo de una aplicación web compleja. Automatiza las tareas más comunes y reutiliza el código cada vez que se crea una nueva aplicación web. Tiene compatibilidad con la mayoría de los gestores de base de datos. Es multiplataforma, sin embargo, su mapeo de objeto relacional para conectarse a las bases de datos es propenso a lentitud, pues conlleva a gastos innecesarios en la generación y configuración del código. El desarrollo de la autenticación en Symfony utiliza medidas de seguridad muy débiles, los roles son llamados credenciales. Esto no es una medida de seguridad por roles puesto que para saber a qué usuario se le han otorgado credenciales, antes se tiene que haber leído toda la lista de roles. Al estar desarrollado con PHP 5 se debe tener un avanzado conocimiento de este lenguaje (17).

[6]Common Languaje Runtime

CodeIgniter 2.1.0

CodeIgniter es un entorno de desarrollo abierto que permite crear webs dinámicas con PHP. Su principal objetivo es ayudar a que los desarrolladores, puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero.

También hay que destacar que CodeIgniter es más rápido que muchos otros entornos y que es distribuido bajo licencia de código abierto (18).

Características:

- ✓ Sistema basado en el Modelo-Vista-Regulador.
- ✓ Compatible con PHP.
- ✓ Características completamente equipadas de la base de datos con la ayuda para varias plataformas.
- ✓ Ayuda de base de datos.
- ✓ Validación de la forma y de los datos.
- ✓ Seguridad y filtración XSS.
- ✓ Gerencia de sesión.
- ✓ Clases de envíos de email. Soporte de agregados, HTML/Texto email, protocolos múltiples (sendmail, smtp[7], y mail) y más.
- ✓ Biblioteca de manipulación de imágenes. Ayudas GD, ImageMagick, y NetPBM.
- ✓ Clases que carga del archivo (18).

ExtJs Framework 2.2

Se seleccionó por el equipo de desarrollo el Framework ExtJs para el diseño de las interfaces. Este framework es una librería construida con Java Script, su potencia radica en la rica colección de componentes para el diseño de interfaces gráficas de usuario del lado del cliente haciendo uso extensivo de Ajax. Entre sus principales ventajas están:

- ✓ El diseño está completamente separado de la funcionalidad.
- ✓ Funciones comunes como validación, ventanas móviles (con minimizar y maximizar), grillas editables, son muy fáciles de implementar.

[7] Protocolo Simple de Transferencia de Correo

- ✓ Buena y amplia documentación, así como también su comunidad (19).

Fundamentación del framework seleccionado

Fue seleccionado el framework CodeIgniter, ya que el mismo está diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones Web. Vale destacar que CodeIgniter es compatible con la mayoría de los sistemas gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Además, implementa la mayoría de las mejores prácticas y patrones de diseño para la programación web. Al igual se seleccionó el framework ExtJs por todas las características antes mencionadas.

1.4.4 Sistema gestor de base de datos

El estudio de la bibliografía consultada muestra que un Sistema Gestor de Bases de Datos (SGBD) es un sistema de software que permite la definición de bases de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación (20).

En la bibliografía consultada existen diferentes SGBD, en lo adelante se describen MySQL y PostgreSQL

PostgreSQL

PostgreSQL es hoy en día el gestor de bases de datos objeto-relacional de código abierto más avanzado. Ofrece control de concurrencia multiversión, soportando casi todas las sintaxis SQL. Cuenta con amplios enlaces a lenguajes de programación importantes, como son C, C++, Java, Perl y Phyton incluyendo interfaz visual para ellos. Provee una arquitectura fiable y una integridad total de los datos (21).

Trabaja la concurrencia permitiendo el bloqueo de tabla y filas para controlar el acceso a los datos. Es un gestor de base de datos multiplataforma, permite la declaración de funciones propias, así como la definición de disparadores. Soporta el uso de índices, reglas y vistas. Incluye herencia entre tablas aunque no entre objetos, ya que no existen, por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales (21).

MySQL

MySQL es un sistema de gestión de bases de datos relacional. Su diseño multihilo le permite soportar una gran carga de información de forma muy eficiente. Está diseñado para entornos de producción críticos, con alta carga de trabajo así como para integrarse en aplicaciones para ser distribuido. MySQL es una marca registrada de MySQL AB, aunque tiene una doble licencia. Los usuarios pueden elegir entre usar el producto MySQL como un sistema Open Source bajo los términos de la licencia GNU (del inglés General Public License) o pueden adquirir una licencia comercial estándar de MySQL AB.

Se encuentra programado en C y en C++, probado con un amplio rango de compiladores diferentes, multiplataforma, proporciona sistemas de almacenamiento transaccionales y no transaccionales. El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor.

También es disponible como biblioteca y puede ser incrustado en aplicaciones autónomas. Dichas aplicaciones pueden usarse por sí mismas o en entornos donde no hay red disponible. Los clientes pueden conectar con el servidor MySQL usando sockets TCP/IP en cualquier plataforma (21).

Fundamentación del gestor de base de datos seleccionado

PostgreSQL se caracteriza por ser un sistema estable, de alto rendimiento, gran flexibilidad ya que funciona en la mayoría de los sistemas Unix, además tiene características que permiten extender fácilmente el sistema. PostgreSQL puede ser integrada al ambiente Windows permitiendo de esta manera a los desarrolladores, generar nuevas aplicaciones o mantener las ya existentes. Permite desarrollar o migrar aplicaciones desde Access, Visual Basic, Foxpro, Visual Foxpro, C/C++ Visual C/C++, Delphi, etc., para que utilicen a PostgreSQL como servidor de BD. Por lo expuesto PostgreSQL se convierte en una gran alternativa al momento de decidirse por un Sistema Gestor de BD.

1.4.5 Servidor Web

Un servidor web es un programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten usando el protocolo HTTP o el protocolo HTTPS (la versión cifrada y autenticada) (22).

Apache 2.2.17

Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal. Apache es una tecnología gratuita de código fuente abierto. El hecho de ser gratuito es importante pero no tanto como que se trate de código fuente abierto. Apache es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este. Otra cosa importante es que cualquiera que posea una experiencia en la programación de C o Perl puede escribir un módulo para realizar una función determinada. Apache trabaja con PHP y otros lenguajes de script. También trabaja con Java y páginas jsp. Teniendo todo el soporte que se necesita para tener páginas dinámicas. Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto (23).

Por lo anteriormente expuesto para el desarrollo del sistema se utilizará como servidor web Apache.

1.4.6 Entorno Integrado de Desarrollo

Un Entorno Integrado de Desarrollo (IDE) es un conjunto de aplicaciones informáticas que consisten en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Puede dedicarse exclusivamente a un lenguaje de programación o bien puede utilizarse en varios lenguajes.

Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto. Posee una funcionalidad muy grande, pero esa funcionalidad es muy genérica. Permite que nuevos componentes puedan utilizar distintos tipos de contenido, para realizar determinadas tareas con contenidos existentes. La plataforma Eclipse

permite descubrir, e invocar funcionalidad implementada en componentes llamados plug-ins.

Soporta las herramientas proporcionadas por diferentes fabricantes de aplicaciones informáticas independientes. Contiene funciones que permiten manipular diferentes contenidos como son HTML, Java, C, JSP, EJB, XML, y GIF. Facilita una integración transparente entre todas las herramientas y tipos de contenidos sin tener en cuenta al proveedor. Proporciona entornos de desarrollo gráfico (GUI por sus siglas en inglés) o no gráficos. Permite ejecutarse en una gran variedad de sistemas operativos, incluyendo Windows y Linux (24).

NetBeans 6.9

NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento. Sun Micro Systems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos. Hoy en día hay disponibles dos productos: el NetBeans IDE y NetBeans Platform. Es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas, está escrito en Java pero eso no quiere decir que no sirva para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender la plataforma. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

NetBeans Platform es una base modular y extensible usada como estructura de integración para crear grandes aplicaciones de escritorio. El código fuente del NetBeans está disponible para su reutilización de acuerdo con la Licencia de Desarrollo y Distribución Común v1.0 (CDDL por sus siglas en inglés) y la GPL v2 (25).

IntelliJ IDEA

IntelliJ IDEA es un IDE centrado en la productividad del código. El editor entiende profundamente el código, hace sugerencias muy adecuadas y ayuda a dar forma a su código. Soporta Java, Groovy, Scala, XML con asistencia de código, JUnit, CVS, Subversion. Trae integración Ant y Maven. Entre sus principales características se encuentra la asistencia inteligente de codificación,

la generación de código, el estilo de código, la documentación que posee, el análisis de código sobre la marcha lo cual usa un método intuitivo para ayudar a escribir el código. Posee una amplia gama de lenguajes de programación permitidos, los cuales son Java, JavaScript, Flex, HTML, XHTML, CSS, XML, XSL, Ruby, JRuby, Groovy y sintaxis SQL. Trae integrado varias tecnologías y framework que son JSP, JSF, EJB, AJAX, GWT, Struts, Struts 2, JBossSeam, Spring, Hibernate/JPA, Web Services, Rails, Grails, Java ME MIDP/CLDC (26).

Otra característica de importancia que se debe conocer, es que es una aplicación propietaria y por tanto se debe pagar por su licencia o uso.

Fundamentación del Entorno Integrado de Desarrollo

El entorno de desarrollo seleccionado es NetBeans, por todas las funcionalidades y la potencia que este posee a la hora de trabajar con el lenguaje de programación PHP y el framework CodeIgniter. Además, por ser una aplicación libre y multiplataforma.

1.4.7 Lenguaje del lado del servidor

Los lenguajes del lado del servidor son los encargados de realizar la conexión entre la base de datos a utilizar y el servidor, generan las páginas antes de enviarlas al cliente, en sentido general son los encargados de todos los aspectos relacionados con la funcionalidad del sistema.

PHP 5

PHP es un acrónimo recursivo que significa “Hypertext Pre-processor”. Es un lenguaje de programación de estilo clásico, con variables, sentencias condicionales, bucles y funciones. No es un lenguaje de marcas como podría ser HTML, XML o WML. Es un lenguaje interpretado, débilmente tipado y orientado a objeto. PHP se ejecuta en el servidor, permite acceder a los recursos del servidor, por ejemplo podría ser una base de datos. Las sentencias PHP son ejecutadas en el servidor y el resultado enviado al navegador. El resultado es normalmente una página HTML.

PHP es un intérprete que puede ser incluido en un servidor web como un módulo. Con él se pueden realizar accesos a ficheros y conexiones de red. Está diseñado para ser más seguro que cualquier otro lenguaje de

programación como C. Dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas (27).

Entre las principales características de PHP se destacan: su rapidez, facilidad de aprendizaje, soporte multiplataforma tanto de diversos sistemas operativos, como servidores http y de bases de datos y el hecho de que se distribuye de forma gratuita bajo una licencia abierta (27).

Perl

Es actualmente un lenguaje interpretado de propósito general aunque en sus inicios fue pensado por Larry Wall para hacer más sencillas las tareas relacionadas con la administración de sistemas Unix. Existen versiones para la mayoría de los sistemas operativos por lo que se ha convertido en un lenguaje multiplataforma ampliando un poco más su alcance. Una de sus limitaciones está dada por la herencia de algunas estructuras del intérprete de comandos Unix que convierte a su sintaxis, en muchas ocasiones, difícil de comprender.

Mediante una serie de módulos adicionales, tales como el DBD o el ODBC, Perl puede servir para acceder a bases de datos, desde BD gratuitas como MySQL hasta el Microsoft SQL server usando ODBC (28).

Fundamentación del lenguaje del lado del servidor

Después de analizar los distintos lenguajes de programación web se puede apreciar que todos son recomendables para desarrollar aplicaciones web con una alta calidad. Se escogió PHP para llevar a cabo el desarrollo de esta, principalmente por sus ventajas ya que es muy fácil de aprender, se caracteriza por ser muy rápido, es multiplataforma y tiene capacidad de conexión con la mayoría de los manejadores de base de datos. También se puede decir que tiene la posibilidad de expandir su potencial mediante módulos y además presenta una comunidad bastante activa.

1.4.8 Lenguaje del lado del cliente

CSS

CSS (Cascading Style Sheets) es una tecnología que permite crear páginas web de una manera más exacta. Son, además, un lenguaje formal que define cómo se va a mostrar un documento estructurado, escrito en HTML o XML y persigue separar la estructura de la presentación La gran ventaja que poseen las hojas de estilo en cascadas es que un cambio en la hoja de estilo modifica inmediatamente todas las páginas que estén relacionadas con esta sin necesidad de hacerlo directamente en cada una.

La sintaxis CSS permite aplicar al documento un formato de modo mucho más exacto. Si antes el HTML no era suficiente para maquetar las páginas y se tenían que utilizar trucos para conseguir los efectos, ahora con el uso de las CSS se tienen muchas más herramientas que permiten definir esta forma:

- ✓ Definir la distancia entre líneas del documento.
- ✓ Colocar elementos en la página con mayor precisión, y sin lugar a errores.
- ✓ Definir la visibilidad de los elementos, márgenes, subrayados y tachados (29).

Con el HTML tan sólo se podían definir atributos en las páginas con pixeles y porcentajes, ahora se pueden definir utilizando muchas más unidades como:

- ✓ Pixels (px) y porcentaje (%), como antes.
- ✓ Pulgadas (in).
- ✓ Puntos (pt).
- ✓ Centímetros (cm) (29).

JavaScript

Es un lenguaje de programación del lado del cliente, utilizado para crear pequeños programas encargados de realizar acciones dentro de una página web. El navegador del cliente, es el encargado de interpretar las instrucciones y ejecutarlas para realizar efectos e interactividades, de modo que el mayor recurso y tal vez el único, con que cuenta este lenguaje es el propio navegador.

Es bastante sencillo y pensado para hacer las cosas con rapidez, a veces con ligereza. Entre las acciones típicas que se pueden realizar en Javascript se destacan los efectos especiales sobre páginas web, la creación de contenidos dinámicos, elementos de la página que tengan movimiento, cambien de color o cualquier otro dinamismo. Permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que se pueden crear páginas interactivas con programas como calculadoras, agendas, o tablas de cálculo y validaciones de datos en los formularios (30).

Javascript es un lenguaje con muchas posibilidades, permite la programación de pequeñas secuencias de comandos, pero también de programas más grandes, orientados a objetos, con funciones y estructuras de datos complejos. Además, Javascript pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente. Permitiendo que este se convierta en el verdadero dueño y controlador, de cada cosa que ocurre en la página cuando la está visualizando el cliente (30).

Fundamentación del lenguaje del lado del cliente

El estudio realizado permitió seleccionar para la implementación de la propuesta como lenguaje del lado del cliente a JavaScript. Este lenguaje permite interactuar con el navegador de manera dinámica y eficaz, proporcionando a las páginas web dinamismo y vida. Los programas en JavaScript se pueden incluir en cualquier documento HTML, o todo aquel que termine traduciéndose en HTML en el navegador del cliente, ya sea PHP, ASP, JSP, SVG, y se encargan de realizar acciones en el cliente, como pueden ser pedir datos, confirmaciones, mostrar mensajes, crear animaciones, comprobar campos.

1.5 Conclusiones del capítulo

A partir del estudio realizado se pudo observar que no existe a nivel internacional o nacional un sistema para la gestión de préstamos de implementos deportivos, por tanto se decidió desarrollar una herramienta que dé solución al problema planteado. Se estudiaron las características de las

principales tecnologías existentes a nivel internacional para el desarrollo de los procesos de software. Se decidió utilizar en este trabajo XP como guía de desarrollo para obtener mejores resultados, pues es una de las metodología que más se usa en el mundo para la realización de proyectos pequeños. Para realizar el modelado del sistema se emplea la herramienta case Visual Paradigm, apoyándose en el lenguaje UML. El IDE seleccionado fue NetBeans junto con el lenguaje de programación PHP. Como framework de desarrollo de aplicaciones web, se seleccionó CodeIgniter y ExtJs y como gestor de base de datos PostgreSQL.

CAPÍTULO 2: Características del sistema

2.1 Introducción

El capítulo actual tiene como propósito exponer el contexto en que se desarrolla el problema que da origen al sistema. En términos de ingeniería se refiere al dominio alrededor del cual gira, y que facilita la comprensión de la situación en cuestión. También se hace referencia a las cuestiones relacionadas con las fases de Exploración y Planificación, en la que se detallan las historias de usuario para establecer luego el orden en que serán implementadas atendiendo a su prioridad, que constituyen los pasos iniciales de la metodología de desarrollo elegida para organizar el ciclo de vida de la herramienta.

2.2 Descripción del dominio

Para realizar la gestión de préstamos de implementos deportivos se hace necesaria una organización que implica el establecimiento de controles, de manera operativa que garantice la devolución oportuna de los mismos, según la herramienta que se diseña para esta gestión.

De acuerdo con lo anterior, la persona encargada de la gestión de préstamos de los implementos en el gimnasio de Cultura Física debe ser capaz de lograr que el servicio se obtenga con eficacia y que llegue oportunamente al cliente y a la comunidad universitaria, para lo cual debe administrar todos los recursos disponibles.

2.3 Modelo conceptual

Para complementar el entendimiento de la dinámica asociada se hace uso de un modelo conceptual, aunque la metodología Extreme Programming no propone artefactos en este sentido es flexible y puede contrastarse con algunos que simplifiquen y apoyen el proceso, que esclarezca los conceptos referentes al problema antes mencionado.

Un modelo conceptual captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabaja el sistema.

Los principales conceptos identificados en este dominio fueron:

Pañolero: persona encargada de realizar la gestión de los préstamos de implementos deportivos y generación de reportes.

Reportes: documento que en su estructura contiene aspectos relevantes del préstamo de implementos.

Administrador: usuario encargado de administrar la aplicación.

Préstamos: préstamos a corto plazo de los implementos.

Implementos: utensilio o herramienta solicitada**.**

Cliente: usuario UCI que solicita el préstamo del implemento deportivo.

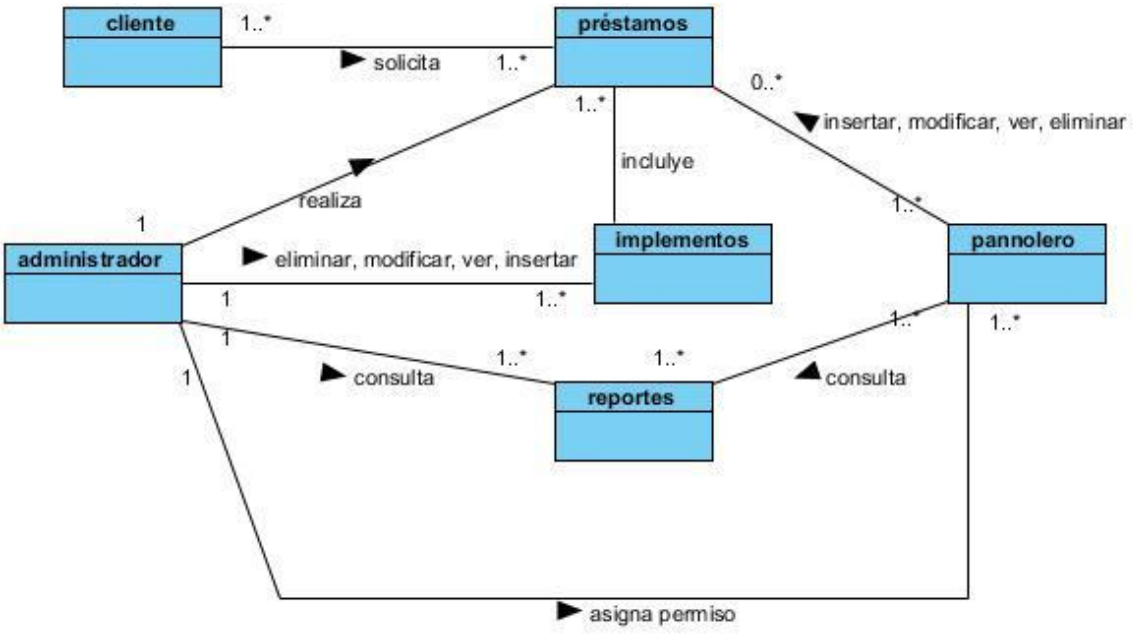

Figura # 1 Modelo conceptual.

2.4 Propuesta del sistema

Se implementará un sistema en el que los usuarios de la comunidad universitaria puedan solicitar un implemento deportivo desde la web o presentándose directamente en el gimnasio, este sistema debe ser capaz de

gestionar los préstamos de implementos deportivos de la UCI, en él los usuarios encargados de realizar el préstamo podrán consultar todo referente en cuanto al préstamo de implementos.

Esta aplicación propiciará tener un control de los implementos deportivos de forma digital, como también que dicha aplicación puede ser accedida desde cualquier lugar de la UCI.

2.5 Requerimientos funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Deben ser comprensibles por los clientes, usuarios y desarrolladores. Deben tener una sola interpretación y estar definidos en forma medible y verificable. Se mantienen invariables sin importar con qué propiedades o cualidades se relacionen. No alteran la funcionalidad del producto (31). A continuación se mencionan los requisitos funcionales del sistema en desarrollo:

RF1- Autenticar usuario

RF2- Gestionar usuario

- RF2.1-Insertar usuario
- RF2.2-Modificar usuario
- RF2.3-Eliminar usuario

RF3- Gestionar roles

- RF3.1-Insertar rol
- RF3.2-Modificar rol
- RF3.3-Eliminar rol

RF4- Gestionar tipo de implemento

- RF4.1-Insertar tipo de implemento
- RF4.2-Modificar tipo de implemento
- RF4.3-Eliminar tipo de implemento

RF5- Gestionar estado de implemento

- RF5.1-Insertar estado de implemento
- RF5.2-Modificar estado de implemento
- RF5.3-Eliminar estado de implemento

RF6- Gestionar implemento deportivo

RF6.1-Insertar implemento deportivo

RF6.2-Modificar implemento deportivo

RF6.3-Eliminar implemento deportivo

RF7-Generar reporte de implemento deportivo

RF8- Gestionar préstamo de implemento deportivo

RF8.1-Insertar préstamo de implemento deportivo

RF8.2-Modificar préstamo de implemento deportivo

RF8.3-Eliminar préstamo de implemento deportivo

RF9-Generar reporte de préstamo de implemento deportivo

RF10- Atender Solicitudes

RF11- Realizar solicitud

2.6 Flujos de trabajo para el desarrollo de la aplicación

2.6.1 Exploración

El ciclo de vida ideal de un proyecto realizado con XP se inicia con la fase de Exploración, al final de la cual el equipo de desarrollo cuenta con suficiente material de trabajo traducido en historias de usuario, que se recopilan en esta etapa, como para producir una primera entrega. Además se produce el contacto necesario con las herramientas y tecnologías que se emplearán para construir el sistema y algunas ideas experimentales en cuanto a su arquitectura son consideradas. Se pueden explorar soluciones puntuales también cuando no se tenga una concepción transparente de alguna funcionalidad.

Historias de Usuario

Una de las mejores prácticas adoptadas en el desarrollo de software es la administración de requerimientos. XP propone en este sentido hacer uso de las historias de usuario como técnica para especificar las funcionalidades que brindará el sistema y constituye una manera muy dinámica de realizar esta actividad. Como su nombre lo indica son especificadas por los propios usuarios y por tanto redactadas en su lenguaje; de manera sencilla y breve, evitando tecnicismos innecesarios que puedan crear confusión, aunque los

programadores pueden contribuir en la tarea. Además son la base para realizar las pruebas de aceptación, así como la estimación y planificación del proyecto. Una vez identificadas las historias de usuario necesarias para liberar una primera versión operativa los programadores proceden a descomponer cada una en tareas específicas, las denominadas tareas de programación que están escritas técnicamente, que darán solución a la historia correspondiente. A continuación se muestran las historias de usuario identificadas:

Tabla # 1: HU[8] Gestionar préstamos de implemento deportivo.

Historia de Usuario	
Número:1	**Nombre Historia de Usuario:** Gestionar préstamos
Usuario: Pañolero y Administrador	**Iteración Asignada**: 1
Prioridad en Negocio: Alta	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alto	
Descripción: Se ofrece la posibilidad de insertar, modificar y eliminar un préstamo de un implemento deportivo. El sistema debe informar que se realizó satisfactoriamente la operación escogida.	
Observaciones: Existe un administrador y usuarios genéricos.	

Tabla # 2: HU Generar reporte de préstamos de implemento deportivo.

Historia de Usuario	
Número:2	**Nombre Historia de Usuario:** Generar reportes de préstamos de implementos deportivos.
Usuario: Pañolero y Administrador	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 1
Riesgo en Desarrollo: Medio	

[8] Historia de usuarios

Descripción: Se ofrece la posibilidad de ver y generar los reportes de préstamos de implementos deportivos según criterios de búsquedas los mismos puedes ser: ✓ Los implementos que más se han prestado en la semana. ✓ Los usuarios que más han solicitado préstamos de implementos. ✓ Cantidad de implementos prestados. ✓ Cantidad de implementos disponibles.
Observaciones: Existe un administrador y usuarios genéricos

Tabla # 3: HU Chequear estado de implementos.

Historia de Usuario	
Número:3	**Nombre Historia de Usuario:** Chequear estado de Implementos.
Usuario: Administrador	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 1
Riesgo en Desarrollo: Medio	
Descripción: Se ofrece la posibilidad al administrador de ver el estado de los implementos. El estado es definido por el administrador.	
Observaciones: Existe un administrador y usuarios genéricos	

Tabla # 4: HU Gestionar implemento deportivo.

Historia de Usuario	
Número:4	**Nombre Historia de Usuario:** Gestionar implemento
Usuario: Administrador	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:**1
Riesgo en Desarrollo: Alto	
Descripción: Se ofrece la posibilidad de insertar, modificar y eliminar un implemento deportivo. El sistema debe informar que se realizó satisfactoriamente la operación escogida.	
Observaciones: Existe un administrador general del sitio.	

Tabla # 5: HU Gestionar usuarios.

Historia de Usuario	
Número:5	**Nombre Historia de Usuario:** Gestionar usuarios
Usuario: Administrador	**Iteración Asignada:**2
Prioridad en Negocio: Media	**Puntos Estimados:**1
Riesgo en Desarrollo: Medio	
Descripción: Se ofrece la posibilidad de agregar, ver y eliminar un usuario. El sistema debe informar que se realizó satisfactoriamente la operación escogida.	
Observaciones: Existe un administrador general del sitio.	

Tabla # 6: HU Autenticar usuario.

Historia de Usuario	
Número:6	**Nombre Historia de Usuario:** Autenticar usuario
Usuario: usuarios	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:**1
Riesgo en Desarrollo: Medio	
Descripción: El usuario entra sus datos (usuario y contraseña) en el formulario, el sistema verifica que los datos entrados estén correctos y que se trate de un usuario con privilegios, en caso correcto se le permite el acceso, en caso contrario se le informa un mensaje de aviso indicándole la negativa del acceso.	
Observaciones: Los usuarios deben tener el mínimo privilegio absoluto necesario para realizar las tareas asignadas.	

Tabla # 7 HU Realizar solicitud

Historia de Usuario	
Número:7	**Nombre Historia de Usuario:** Realizar Solicitud
Usuario: usuario	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:**1
Riesgo en Desarrollo: Medio	
Descripción: La HU se inicia cuando el usuario selecciona la opción realizar solicitud, mostrándose todo los implementos deportivos, una vez presentados los implementos en existencia selecciona el implemento de su interés, y seguidamente el sistema muestra un mensaje de confirmación.	
Observaciones: Cuando se realiza una solicitud esta obtiene el estado pendiente.	

Tabla # 8 HU Ver solicitud

Historia de Usuario	
Número:8	**Nombre Historia de Usuario:** Ver solicitud
Usuario: pañolero y Administrador	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:**1
Riesgo en Desarrollo: Medio	
Descripción: La HU se inicia cuando se selecciona la opción atender solicitudes del bloque operaciones. El sistema muestra una vista donde se listan todas las solicitudes realizadas y permite realizar la búsqueda de una o varias solicitudes en específico, mostrando un selector que permite filtrar por el criterio (fecha).	
Observaciones: En el caso de verificar la información que contiene el listado que se muestra inicialmente y luego de introducir los criterios de búsqueda, solamente se visualizan las solicitudes en estado pendiente y que cumplan con los criterios establecidos por el usuario. Luego de dar clic sobre la solicitud buscada, se muestra una vista con la plantilla de dicha solicitud.	

Tabla # 9 HU Atender solicitud

Historia de Usuario	
Número:9	**Nombre Historia de Usuario:** Atender Solicitud
Usuario: pañolero y Administrador	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:**1
Riesgo en Desarrollo: Medio	
Descripción: La HU se inicia cuando el usuario encargado de atender la solicitud selecciona esta opción, luego el usuario puede seleccionar una de las dos opciones que se proponen: ✓ Aceptar solicitud: Acepta la solicitud y se le envía un correo al usuario	

que solicitó el préstamo, confirmado que su solicitud ha sido confirmada. ✓ Rechazar solicitud: En caso de que no se pueda aprobar el préstamo de los implementos el sistema le enviara un mensaje en el que el que le explicará que no se puede prestar el implemento solicitado por algún motivo.
Observaciones:

Tabla # 10 HU Gestionar nomenclador de estado de implementos

Historia de Usuario	
Número:10	**Nombre Historia de Usuario:** Gestionar nomenclador estado de implementos
Usuario: Administrador	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:**1
Riesgo en Desarrollo: Medio	
Descripción: La HU se inicia cuando se selecciona la opción. Se muestra un listado de todos los tipos de estados existentes, si el usuario selecciona la opción: ✓ Añadir estado: el sistema muestra una vista con los campos que deben ser llenados, luego de introducir los datos, se muestra un mensaje de confirmación “El estado ha sido insertado satisfactoriamente” quedándose la vista activada brindando la posibilidad de crear otro estado. ✓ Modificar estado: luego de seleccionar el estado a modificar, el sistema muestra una vista con los datos del estado, permitiendo su modificación, después de modificar los datos deseados el sistema muestra un mensaje de confirmación “El estado ha sido modificado satisfactoriamente” y muestra el listado de los estados.	

✓ Eliminar estado: después de seleccionar el estado a eliminar, hacer clic en la opción eliminar y seguido de esto se muestra un mensaje de confirmación “El estado ha sido eliminado satisfactoriamente”.
Observaciones: Para modificar o eliminar un estado primero se debe seleccionar este en el listado de estados.

Tabla # 11 HU Gestionar nomenclador de roles

Historia de Usuario	
Número:10	**Nombre Historia de Usuario:** Gestionar nomenclador de rol
Usuario: administrador	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:**1
Riesgo en Desarrollo: Medio	
Descripción: La HU se inicia cuando se selecciona la opción. Se muestra un listado de todos los roles existentes, si el usuario selecciona la opción: ✓ Añadir rol: el sistema muestra una vista con los campos que deben ser llenados, luego de introducir los datos, se muestra un mensaje de confirmación “El rol ha sido insertado satisfactoriamente” quedándose la vista activada brindando la posibilidad de crear otro estado. ✓ Modificar rol: luego de seleccionar el estado a modificar, el sistema muestra una vista con los datos del rol, permitiendo su modificación, después de modificar los datos deseados el sistema muestra un mensaje de confirmación “El rol ha sido modificado satisfactoriamente” y muestra el listado de los estados. ✓ Eliminar rol: después de seleccionar el estado a eliminar, das clic en la opción eliminar y seguido de esto se muestra un mensaje de confirmación “El rol ha sido eliminado satisfactoriamente”.	
Observaciones: Para modificar o eliminar un rol primero se debe seleccionar este en el listado de roles.	

Tabla # 12 HU Generar reporte de implemento deportivo

Historia de Usuario	
Número:12	**Nombre Historia de Usuario:** Generar reportes de implementos deportivos.
Usuario: Pañolero y Administrador	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 1
Riesgo en Desarrollo: Medio	
Descripción: Se ofrece la posibilidad de ver y generar los reportes de los implementos deportivos que se encuentran en el gimnasio.	
Observaciones: Existe un administrador y usuarios genéricos	

Tabla # 13 HU Envío de correos

Historia de Usuario	
Número:13	**Nombre Historia de Usuario:** Envío de correos
Usuario: Pañolero y Administrador	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 1
Riesgo en Desarrollo: Medio	
Descripción: Se ofrece la posibilidad de enviar correos automáticamente a todos los usuarios que se le vence el préstamo del implemento.	
Observaciones: Existe un administrador y usuarios genéricos	

2.6.2 Planificación

A lo largo de esta fase los clientes establecen la prioridad de las historias de usuario de acuerdo a sus necesidades más inmediatas para luego asignarlas, por orden de relevancia, a las iteraciones planificadas. A partir de las historias se realiza la estimación del esfuerzo que se requiere por parte del equipo para su posterior implementación. El método escogido para realizar la estimación tiene como elemento el punto, el cual equivale a una semana perfecta de trabajo, esto se refiere a que solamente el equipo se dedica a labores relacionadas con la construcción del sistema sin la influencia o el retraso provocado por otros factores, lo que en la práctica es complejo lograr.

A modo de resumen se presenta la siguiente tabla que muestra las 2 iteraciones analizadas previamente con las historias de usuario que incluyen y su duración:

Tabla # 14 Plan de duración de las iteraciones

Iteraciones	**Historias de Usuario a implementar**	**Duración**
Iteración 1	Gestionar préstamo de implemento deportivo	**3 semanas**
	Gestionar implemento deportivo	
	Realizar solicitud	
	Atender solicitud	
	Gestionar nomenclador de roles	
	Gestionar nomenclador estado de implementos	
Iteración 2	Generar reportes de préstamos de implementos deportivos.	**3 semanas**
	Generar reportes de implementos deportivos	
	Chequear estado de implementos	
	Gestionar usuarios.	
	Autenticar usuario.	
	Envío de correos	

Estimación de esfuerzos por Historias de Usuario

Las estimaciones del esfuerzo para implementar las historias de usuario permiten tener una medida bastante real de la velocidad de progreso del proyecto y brindan una guía razonable a la cual ajustarse. Los resultados estimados se exhiben seguidamente:

Tabla # 15 Estimación por historias de usuario

No.	Historia de usuarios	Puntos de Estimación
1	Gestionar préstamos.	1
2	Generar Reportes de Préstamos de implementos.	1
3	Chequear estado de implementos.	1
4	Gestionar implemento deportivo.	1
5	Gestionar usuarios.	1
6	Autenticar usuario.	1
7	Gestionar nomenclador de estado de implementos	1
8	Atender Solicitud	1
9	Ver Solicitud	1
10	Realizar Solicitud	1
11	Gestionar nomenclador de roles	1
12	Generar reportes de implementos deportivos.	1
13	Envío de correos	1

2.6.3 Plan de Liberaciones

El plan de liberaciones (entregas) tiene como objetivo definir el número de versiones que se realizarán en el transcurso del proyecto y las iteraciones que se requieren para desarrollar cada una. El cliente se encarga de decidir cuáles historias de usuario comprende la primera entrega según sus prioridades para darle valor a su negocio y que por tanto justifique su ejecución y así sucesivamente para las demás.

Sin embargo, se precisa establecer el contenido de trabajo para todas y cada una de ellas y es aquí donde hace acto de presencia el plan de iteraciones, regulando la cantidad de historias de usuario a implementar dentro del rango establecido por la estimación efectuada.

2.7 Conclusiones

En este capítulo se efectuó un análisis exhaustivo de los conceptos relacionados con el negocio planteado empleándose para ello un modelo conceptual ilustrativo que sirvió de apoyo para una familiarización más adecuada. Se abordaron las peculiaridades de las fases de Exploración y Planificación y los artefactos que se generaron durante su desarrollo, entre ellos las historias de usuario y los planes de iteraciones. Ambas fases se repiten en cada iteración lo que posibilita realizar una estimación más exacta y real del esfuerzo necesario para cumplir con las historias de usuario negociadas y con las que el equipo de trabajo se ha comprometido.

CAPÍTULO 3: Diseño, Implementación y Pruebas

3.1 Introducción

En el presente capítulo se hace referencia a los elementos que conforman el diseño del sistema a construir. Algunas de las prácticas definidas en la metodología empleada proponen que se realice y mejore incrementalmente mediante pasos pequeños al igual que la arquitectura. Se describen las pruebas de aceptación a las que se sometió la aplicación para verificar su completitud en correspondencia a lo pactado con el cliente.

3.2 Diseño

XP establece prácticas especializadas que inciden directamente en la realización del diseño para lograr un sistema robusto y reutilizable tratando de mantener su simplicidad, es decir, crear un diseño evolutivo que se va mejorando incrementalmente y que permite hacer entregas pequeñas y frecuentes de valor para el cliente. A la hora de darle cumplimiento a la actividad de diseñar, XP no especifica ninguna técnica de modelado, puede utilizarse indistintamente sencillos esquemas en una pizarra, diagramas de clases utilizando UML o tarjetas CRC (Clase, Responsabilidad y Colaboración) siempre que sean útiles, tributen a la comprensión y no requieran mucho tiempo en su creación. Como base para el desarrollo de la aplicación propuesta se utilizó el framework CodeIgniter que implementa, como lo hacen muchos otros, el patrón arquitectónico MVC (Modelo Vista Controlador) que nos obliga a organizar el código de acuerdo a sus convenciones. Para entender todos estos aspectos funcionales con mayor claridad se describe su composición.

3.2.1 Tarjetas CRC

Las tarjetas CRC (Clase, Responsabilidad, Colaborador) son una herramienta de reflexión en el diseño de software orientado a objetos. Fueron propuestas por Ward Cunningham y Kent Beck. Se utilizan normalmente cuando primero

se determinan las clases que se necesitan y cómo van a interactuar y después se implementa la solución.

A continuación, las tarjetas CRC agrupadas por clase:

Tabla # 16 Tarjeta CRC de gestionar préstamo

Clase: gestprestamo	
Responsabilidades	**Clase relacionadas**
Insertar los préstamos de los implementos deportivos.	prestamo_model gestprestamo
Modificar los préstamos de los implementos deportivos.	
Eliminar un préstamo de implemento deportivo	
Mostrar los préstamos de implementos deportivos.	
Generar reportes de los préstamos de implementos deportivos.	

Tabla # 17 Tarjeta CRC de gestionar implemento deportivo

Clase: gestimplemento	
Responsabilidades	**Clase relacionadas**
Insertar los implementos deportivos.	gestimplemento
Modificar implementos deportivos.	
Eliminar implementos deportivos.	

Mostrar los implementos deportivos disponibles	implementodeportivo_model
Mostrar implementos deportivos	
Generar reportes de los implementos deportivos	

Tabla # 18 Tarjeta CRC de gestionar solicitud

Clase: gestsolicitud	
Responsabilidades	**Clase relacionadas**
Insertar solicitudes	gestsolicitud solicitud_model
Eliminar solicitudes	
Mostrar todas las solicitudes	

Tabla # 19 Tarjeta CRC de atender solicitudes

Clase: atendersolicitud	
Responsabilidades	**Clase relacionadas**
Eliminar solicitud	solicitud_model atendersolicitud
Aceptar solicitudes	
Mostrar todas las solicitudes	
Rechazar solicitudes	
Realizar préstamo	

Tabla # 20 Tarjeta CRC de nomenclador de roles

Clase: nomencladorrol	
Responsabilidades	**Clase relacionadas**
Insertar un rol.	nomencladorol_model nomenrol
Eliminar un rol.	
Mostrar todas los roles.	

Tabla # 21 Tarjeta CRC gestionar usuarios

Clase: nomencladorusuario	
Responsabilidades	**Clase relacionadas**
Insertar un usuario.	nomencladorusuario_model nomenusuario
Modificar un usuario.	
Eliminar un usuario	
Mostrar todos los usuarios.	

3.2.2 Patrones de diseño

Un patrón de diseño es una solución a un problema de diseño. La misma tiene la característica principal de ser capaz de resolver problemas similares. Debe ser reutilizable, o sea que es aplicable a diferentes problemas de diseño en distintos contextos. Existen varias clasificaciones de patrones, como por ejemplo los patrones GRASP[9] que se utilizan para la asignación de responsabilidades. Su uso es fundamental para un correcto diseño del software. Ayudan a la reutilización de diseño gráfico, identificando aspectos claves de la estructura de un diseño que puede ser aplicado en una gran

[9]Patrones generales de software para asignar responsabilidad

cantidad de situaciones. Lo que supone una gran ventaja ya que disminuye los esfuerzos de desarrollo y mantenimiento, mejora la seguridad informática, eficiencia y consistencia del diseño, y proporciona un inmenso ahorro en la inversión. También ayudan a tener un software más flexible, modular y extensible (32). A continuación se describen los patrones utilizados:

Patrón creador

El patrón creador nos ayuda a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases. La nueva instancia deberá ser creada por la clase que:

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

El uso de este patrón se evidencia en las clases controladoras como gestprestamo, permitiendo el acceso a la información almacenada en las clases entidades.

Patrón experto

Es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos; con él que se pretende designar una idea clara de la forma más eficiente posible; el patrón experto nos indica, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo, es decir , una clase contiene toda la información necesaria para realizar la labor que tiene encomendada (32).

El uso de este patrón se visualiza en la definición de las clases de acuerdo a las funcionalidades que deben realizar, como por ejemplo las clases controladoras y las del modelo. Ejemplo de ello en la clase gestimplemento, la cual gestiona las operaciones que conciernen a las funciones de adicionar, modificar, eliminar y visualizar los implementos deportivos.

Patrón bajo acoplamiento

El bajo acoplamiento fomenta el aumento de la reutilización y la eliminación de las redundancias, creando clases más independientes y con mayor resistencia al impacto de los cambios, que aumentan la productividad y la posibilidad de reutilización. Este patrón se basa en la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases (32).

Este patrón se pone de manifiesto en todas las clases modelos, donde estas clases hacen las funcionalidades que le corresponden y no dependen una de las otras, de manera tal que si se elimina una clase no afecta el funcionamiento de las otras clases

Patrón Alta cohesión

Como el patrón Bajo Acoplamiento, también Alta Cohesión es un principio que se debe tener presente en todas las decisiones de diseño, es la meta principal que ha de buscarse en todo momento.

La cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase, además de que una alta cohesión garantiza que clases con responsabilidades estrechamente relacionadas no realicen un trabajo enorme, y que cada elemento de nuestro diseño debe realizar una labor única dentro del sistema (32).

La herramienta destinada a la gestión de préstamos de implementos deportivos sigue los principios de alta cohesión ya que no se hace uso de las clases saturadas de métodos, y que las clases se encuentran agrupadas por funcionalidades, lo que las hace fácilmente reutilizable.

3.2.3 Descripción de la arquitectura

El sistema propuesto cuenta una arquitectura basada en uno de los estilos más utilizados para el diseño web, el patrón MVC. Este patrón le brinda al sistema

una clara definición de cómo es mostrada la información, el manejo de las acciones que el usuario desea hacer sobre él y como se realizan estas acciones modificando y validando la información.

La estructura del patrón MVC propuesto por el framework utilizado CodeIgniter, se evidencia de la siguiente forma:

La vista cuenta con dos elementos fundamentales: una plantilla que es el archivo que contiene el código HTML que no varía para las interfaces de usuario, un área de trabajo que contiene los formularios de entrada y las vistas de salida de datos. En el controlador se encuentran las clases controladoras que son las encargadas de manejar los eventos del sistema. Las librerías son clases en las cuales se desarrolla la lógica del negocio. La clase administradora se encarga de gestionar dentro de todas las modelos existentes, cual es la modelo que la librería está invocando. En la modelo se encuentra la capa de acceso a datos que contiene todas las funcionalidades que responden a la lógica del negocio.

De esta forma cuando el usuario interactúa con la aplicación lo hace mediante la vista, que envía las peticiones al controlador, este envía el flujo de datos hacia las librerías y esta hacia la acción correspondiente que sería la clase administradora la cual se encarga de llamar a la función implementada en la modelo, esta recepciona el resultado emitido por la modelo y se la envía a la librería, esta transforma la información y se la devuelve a la controladora la cual levanta una vista donde se muestran los resultados solicitados por el usuario.

3.2.4 Estrategia de integración

El sistema tiene la particularidad de consumir información gestionada por otros sistemas ya existentes en la universidad. Esta comunicación se realiza mediante el protocolo SOAP[10] para la comunicación con los servicios web, a continuación se especifica la funcionalidad de la cual se obtiene dicha información.

[10]Simple Object Access Protocol

LDAP[11]

- ✓ Comprobar usuario: este servicio realiza una búsqueda interna para comprobar si el usuario recibido por parámetro es válido en el directorio activo de la universidad.
- ✓ Obtener datos del usuario: este servicio permite obtener todos los datos de un usuario determinado, entre ellos: nombre, apellidos, solapín, facultad, número de teléfono, residencia, etc.

3.2.5 Modelo de datos

Para la persistencia de la información necesaria y para el correcto funcionamiento del sistema, se identificaron 7 tablas. Ver figura # 2

Figura # 2 Modelo de datos

[11]Protocolo Ligero de Acceso a Directorios

A continuación se explican las principales tablas:

implementos_deportivos: entidad que almacena los datos del implemento deportivo.

nom_estadoimplemento: entidad que almacena los estados de un implemento deportivo.

nom_tipoimplemento: entidad que almacena el tipo de implemento deportivo.

prestamo_implemento: entidad que almacena los datos del préstamo de los implementos.

rol: entidad que almacena los roles de usuarios.

solicitudes: entidad que almacena todas las solicitudes de los préstamos.

usuarios: entidad que almacena todos los usuarios del sistema.

3.3 Implementación del sistema

En esta fase se genera todo el código fuente necesario para satisfacer las historias de usuario definidas para la solución y se describen todas las tareas realizadas en cada iteración.

Tomando como referencia los aspectos antes tratados la aplicación que se pretende construir se desarrollará en 2 iteraciones, explicadas detalladamente a continuación:

Iteración 1

La iteración presente tiene como finalidad implementar las HU que se consideraron de prioridad alta, atendiendo a su relevancia e impacto para el negocio. Las mismas son: Realizar solicitud, Ver solicitud, Atender solicitud, Gestionar nomenclador estado de implementos, Gestionar nomenclador de rol, Gestionar implemento, Gestionar préstamos de implementos, estas dos últimas dándole respuestas a las funcionalidades de insertar, modificar, ver y eliminar un implemento así como insertar, modificar, ver y eliminar un préstamo de implemento.

Iteración 2

La actual iteración se centra en darle solución a las HU con un nivel medio de prioridad. Contempla la consulta y generación de reportes, el chequeo de los implementos, gestión de usuarios (adicionar, eliminar, modificar) y la autenticación de usuarios. Al finalizar esta iteración se obtiene un software completamente funcional.

XP es una de las metodologías que más enfatiza el uso de pruebas, proponiendo esta práctica como una de sus prioridades, fundamental para verificar la calidad, y aplicándola desde el comienzo hasta la conclusión del proyecto. A modo general dentro de cada iteración se ejecutan tanto las pruebas unitarias como de aceptación para tener la certeza de que se han implementado correctamente los requerimientos.

3.3.1 Tareas generales de la implementación (TI)

Para la implementación del sistema se llevan a cabo una serie de tareas que no se encuentran comprendidas en las HU y que se han definido como tareas generales a realizar. Las tareas de la implementación (tareas de ingeniería) son muy importantes para el programador porque guían el proceso de desarrollo del sistema.

Tabla # 22: TI Diseñar base de datos

Tarea de ingeniería	
Número de la tarea: 01	
Nombre de la tarea: Diseñar base de datos	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.1
Fecha de inicio: 20/02/2012	**Fecha de fin:** 12/4/2012

Programador responsable: Ariel Nuviola Rodríguez
Descripción: Utilizando PostgreSQL como gestor, se crea la base de datos en la cual se almacenarán los datos persistentes generados durante la utilización de la aplicación.

Tabla # 23: TI Generar las clases modelo

Tarea de ingeniería	
Número de la tarea: 02	
Nombre de la tarea: Generar las clases modelo	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.1
Fecha de inicio: 20/02/2012	**Fecha de fin:** 20/05/2012
Programador responsable: Ariel Nuviola Rodríguez	
Descripción: Se generan las clases modelo, especificándole el servidor de bases de datos, el nombre de la base de datos, el puerto a usar, el usuario y la contraseña para la conexión.	

Tabla # 24: TI Confeccionar las interfaces de usuario

Tarea de ingeniería	
Número de la tarea: 03	
Nombre de la tarea: Confeccionar las interfaces de usuario	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.1
Fecha de inicio: 14/04/2012	**Fecha de fin:** 25/4/2012

Programador responsable: Ariel Nuviola Rodríguez
Descripción: Se diseñan todas las interfaces de usuario que tendrá el sistema de acuerdo con las funcionales planteadas.

Tabla # 25: Implementar los métodos en las clases controladoras

Tarea de ingeniería	
Número de la tarea: 04	
Nombre de la tarea: Implementar los métodos en las clases controladoras	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.1
Fecha de inicio: 20/03/2012	**Fecha de fin:** 12/05/2012
Programador responsable: Ariel Nuviola Rodríguez	
Descripción: Se implementan en las clases controladoras los métodos con sus validaciones.	

3.3.2 Estándar de codificación

Identación, llaves de apertura, de cierre y tamaño de las líneas

Usar una identación sin tabulaciones, con un equivalente a cuatro espacios, el uso de las llaves “{}” será en una nueva línea. La longitud de las líneas de código es aproximadamente de 75-80 caracteres, para mantener la legibilidad del código.

Conversión de nomenclatura:

Variables: Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula.

Constantes: siempre debe ser todas en mayúsculas, con caracteres de subrayado "_" para separar palabras en caso de nombres compuestos.

Clases: siempre comienzan con mayúscula, en caso de nombre compuesto las palabras se separan con el carácter subrayado "_" y el resto en minúscula.

Funciones: Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Los parámetros son separados por espacio luego de la coma que los separa.

Ficheros: todo siempre en minúscula y en caso de nombres compuestos se usa el carácter subrayado "_".

1. Vistas: intuitivo y relacionado con el formulario y/o vista que representa.
2. Modelos: con el mismo nombre de la clase que representa que contiene en el nombre el sufijo _model o _base en caso de ser modelos base.
3. Librerías: con el mismo nombre de la clase que representa que contiene en el nombre el sufijo _lib.
4. Controladoras: con el mismo nombre de la clase que representa.
5. Manager: con el mismo nombre de la clase que representa que contiene en el nombre el sufijo _mng.

Estructuras de control

Se incluyen las palabras reservadas tales como: if, for, foreach, while, switch, entre las estructuras de control y entre los paréntesis debe de existir un espacio. Se recomienda utilizar siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales, esto aumenta la legibilidad y disminuye la probabilidad de errores lógicos. Si las condiciones son muy largas que sobrepasan el tamaño de la línea, estas se dividen en varias líneas. En el mejor de los casos cuando la condición es muy extensa, se puede dividir esta en variables y compararlas dentro de la estructura de control.

3.4 Pruebas

Las pruebas son un conjunto de actividades que se pueden planificar por adelantado y llevar a cabo sistemáticamente. Por esta razón se debe definir en el proceso de la ingeniería del software. Todo esto contribuye a elevar la

calidad de los productos desarrollados y a la seguridad de los programadores a la hora de introducir cambios o modificaciones.

La metodología XP divide las pruebas en dos grupos: pruebas unitarias, desarrolladas por los programadores, encargadas de verificar el código de forma automática y las pruebas de aceptación, destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente.

3.4.1 Pruebas de aceptación

Las pruebas de aceptación son pruebas de caja negra que se realizan a partir de las HU. Durante las iteraciones las historias de usuarios escogidas serán traducidas a prueba de aceptación. En ella se especifican, la perspectiva del cliente, y los escenarios para probar que la historia de usuario ha sido implementada correctamente. Una historia de usuario puede tener todas las pruebas de aceptación que desee para asegurar su funcionamiento. El objetivo específico de esta prueba es garantizar que los requerimientos han sido cumplidos y que el sistema ha sido aceptable (33).

Seguidamente se muestran las pruebas de aceptación para cada una de las tres iteraciones definidas.

Iteración 1

En dicha iteración no se detectó ninguna no conformidad. Luego se realizaron las pruebas de aceptación con el cliente y los resultados de las mismas fueron satisfactorios como muestran a continuación:

Tabla # 26 Prueba de aceptación historia de usuario 1 prueba 1

Caso de Prueba de aceptación	
Código: HU1_P1	**Historia de usuario:** 1
Nombre: Gestionar préstamos	
Descripción: Prueba para las funcionalidades de realizar, modificar y	

eliminar un préstamo en el sistema.
Condiciones de ejecución: El usuario debe estar autenticado y tener permisos para realizar estas operaciones.
Pasos de ejecución: Se intenta publicar, modificar o eliminar un préstamo por un usuario que contiene sus datos válidos, es decir, el administrador u otro rol con privilegios para hacerlo.
Resultados esperados: Los préstamos son realizados, modificados y eliminados sin generar error.
Evaluación de la prueba: Satisfactoria

Tabla # 27 Prueba de aceptación historia de usuario 4 prueba 1

Caso de Prueba de aceptación	
Código: HU4_P1	**Historia de usuario:** 4
Nombre: Gestionar implementos	
Descripción: Prueba para las funcionalidades de insertar, modificar y eliminar un implemento en el sistema.	
Condiciones de ejecución: El usuario debe estar autenticado y tener permisos para realizar estas operaciones.	
Pasos de ejecución: Se intenta insertar, modificar o eliminar un implemento por un usuario que contiene sus datos válidos, es decir, el administrador.	
Resultados esperados: Los implementos son insertados, modificados y eliminados sin generar error.	
Evaluación de la prueba: Satisfactoria	

Tabla # 28 Prueba de aceptación historia de usuario 7 prueba 1

Caso de Prueba de aceptación	
Código: HU7_P1	**Historia de usuario:** 7
Nombre: Realizar solicitud	
Descripción: Prueba para las funcionalidades de realizar una solicitud de un implemento deportivo.	
Condiciones de ejecución: El usuario debe estar autenticado y tener permisos para realizar estas operaciones.	
Pasos de ejecución: Se intenta realizar una solicitud por un usuario que contiene sus datos válidos.	
Resultados esperados: Las solicitudes son insertadas sin generar error.	
Evaluación de la prueba: Satisfactoria	

Tabla # 29 Prueba de aceptación historia de usuario 9 prueba 1

Caso de Prueba de aceptación	
Código: HU9_P1	**Historia de usuario:** 9
Nombre: Atender solicitud	
Descripción: Prueba para las funcionalidades de atender solicitudes.	
Condiciones de ejecución: El usuario debe estar autenticado y tener permisos para realizar estas operaciones. El administrador u otro rol con privilegios.	
Pasos de ejecución: Se intenta atender una solicitud por un usuario que contiene sus datos válidos.	
Resultados esperados: Las solicitudes son atendidas sin generar error.	
Evaluación de la prueba: Satisfactoria	

Tabla # 30 Prueba de aceptación historia de usuario 10 prueba 1

Caso de Prueba de aceptación	
Código: HU10_P1	**Historia de usuario: 10**
Nombre: Gestionar nomenclador estado de implementos	
Descripción: Prueba para las funcionalidades de insertar, modificar y eliminar estado de implemento.	
Condiciones de ejecución: El usuario debe estar autenticado y tener permisos para realizar estas operaciones.	
Pasos de ejecución: Se intenta insertar, modificar o eliminar un estado del implemento por un usuario que contiene sus datos válidos, es decir, el administrador.	
Resultados esperados: Las estados de los implementos son insertados, modificados y eliminados sin generar error.	
Evaluación de la prueba: Satisfactoria	

Tabla # 31 Prueba de aceptación historia de usuario 11 prueba 1

Caso de Prueba de aceptación	
Código: HU11_P1	**Historia de usuario: 11**
Nombre: Gestionar nomenclador de roles	
Descripción: Prueba para las funcionalidades de insertar, modificar y eliminar un rol.	
Condiciones de ejecución: El usuario debe estar autenticado y tener permisos para realizar estas operaciones.	
Pasos de ejecución: Se intenta insertar, modificar o eliminar un rol por un usuario que contiene sus datos válidos, es decir, el administrador.	

Resultados esperados: Los roles son insertados, modificados y eliminados sin generar error.
Evaluación de la prueba: Satisfactoria

Iteración 2

En dicha iteración no se detectaron no conformidades. Posteriormente se realizaron las pruebas de aceptación con el cliente y los resultados de las mismas fueron satisfactorios como muestran a continuación:

Tabla # 32 Prueba de aceptación historia de usuario 2 prueba 1

Caso de Prueba de aceptación	
Código: HU2_P1	**Historia de usuario:** 2
Nombre: Generar reportes de préstamos de implementos	
Descripción: Prueba para las funcionalidades de generar reportes de préstamos.	
Condiciones de ejecución: El usuario debe estar autenticado y tener permisos para realizar estas operaciones.	
Pasos de ejecución: Se intenta generar un reporte por un usuario que contiene sus datos válidos, es decir, el administrador u otro rol con privilegios.	
Resultados esperados: Los reportes son generados satisfactoriamente.	
Evaluación de la prueba: Satisfactoria	

Tabla # 33 Prueba de aceptación historia de usuario 5 prueba 1

Caso de Prueba de aceptación	
Código: HU5_P1	**Historia de usuario:** 5
Nombre: Gestionar usuarios	

Descripción: Prueba para las funcionalidades de insertar, modificar y eliminar un usuario del sistema.
Condiciones de ejecución: El usuario debe estar autenticado y tener permisos para realizar dicha operación.
Pasos de ejecución: Se intenta insertar, modificar o eliminar un usuario por un usuario que contiene sus datos válidos, es decir, el administrador.
Resultados esperados: Los usuarios son insertados, modificados y eliminados sin generar error.
Evaluación de la prueba: Satisfactoria

3.5 Conclusiones

En este capítulo se realizó la presentación y descripción de la arquitectura propuesta para la construcción del sistema. Se definió el diseño de la base de datos, la descripción de las tareas de ingeniería y los estándares de codificación que brindan una idea más clara de cómo desarrollar una aplicación confiable.

Se hizo una descripción de los pasos a seguir para la implementación en cada una de las iteraciones y finalmente quedaron documentadas las pruebas de aceptación que validan como finalizadas las historias de usuario propuestas por el cliente.

Conclusiones generales

A lo largo de la investigación se han ido describiendo todos los pasos seguidos para darle cumplimiento al objetivo de desarrollar una herramienta que permita la Gestión de los préstamos de implementos deportivos de la Universidad de las Ciencias Informáticas.

Una vez finalizado el trabajo se plantean las siguientes conclusiones:

1. El análisis realizado de los referentes teóricos, permitió corroborar la idea de favorecer el proceso de gestión de préstamos de implementos deportivos, así como la determinación del sistema operativo y sus características.
2. Los métodos aplicados, proporcionaron la determinación de las necesidades y potencialidades que tiene la para la cátedra de cultura física de la UCI para controlar los recursos y realizar prestaciones de servicio.
3. La herramienta diseñada como sistema de gestión de implementos deportivos se constituyen en una manera más efectiva para organizar, dirigir, controlar y satisfacer las demandas de sus clientes, de forma rápida, sencilla, proporcionando una alta calidad en sus servicios.
4. Con la puesta en práctica del sistema de gestión de implementos deportivos para la cátedra de cultura física, el gimnasio logra una mayor organización en el manejo de las informaciones y datos, se logra un aumento en la calidad de trabajo y se tiene un mejor control de los préstamos de implementos deportivos efectuados.

Recomendaciones

A partir de los resultados o beneficios que proporciona este trabajo de diploma, se proponen las siguientes recomendaciones.

- ✓ Continuar en la investigación del sistema con la finalidad de aportar nuevas funcionalidades.
- ✓ Desarrollar un manual de usuario para uso de los clientes, con el objetivo de lograr un mayor entendimiento del sistema y su funcionamiento.

Bibliografía

1. Definicion.de. [En línea] [Citado el: 5 de diciembre de 2011.] http://definicion.de/gestion/.

2. ADMINISTRACIÓN ESTRATÉGICA. [En línea] [Citado el: 4 de diciembre de 2011.] http://www.eumed.net/libros/2011c/993/administracion%20estrategica.html.

3. ¿Qué son los sistemas de gestión? [En línea] [Citado el: 4 de diciembre de 2011.] http://www.bsigroup.com.mx/es-mx/Auditoria-y-Certificacion/Sistemas-de-Gestion/De-un-vistazo/Que-son-los-sistemas-de-gestion/.

4. Sistema de Préstamos. [En línea] [Citado el: 12 de noviembre de 2011.] http://es.scribd.com/doc/13151181/Sistema-de-Prestamos.

5. Koha-UNLP. [En línea] [Citado el: 2 de noviembre de 2011.] http://koha.unlp.edu.ar/.

6. **Calleja, Manuel Arias.** *Estándares de configuración.* 2011.

7. El Proceso Unificado de Desarrollo. [En línea] [Citado el: 12 de diciembre de 2011.] http://yaqui.mxl.uabc.mx/~molguin/as/RUP.htm.

8. The Agile Project Delivery Framework. [En línea] [Citado el: 12 de diciembre de 2011.] http://www.dsdm.org.

9. **José H. Canós, Patricio Letelier, Mª Carmen Penadés.** Métodologías Ágiles en el Desarrollo de Software. [En línea] [Citado el: 12 de diciembre de 2011.] http://www.willydev.net/descargas/prev/TodoAgil.Pdf.

10. OpenUP. [En línea] [Citado el: 12 de diciembre de 2011.] http://ndpsoftware.com/OpenUpBasic/index.htm.

11. **Reinoso, Billy.** Métodos Agiles de desarrollo de Software. 2011.

12. Herramientas Case. [En línea] [Citado el: 12 de diciembre de 2011.] http://www.plusformacion.com/Recursos/r/Herramientas-Case.

13. IBM. [En línea] [Citado el: 12 de diciembre de 2011.] http://www-142.ibm.com/software/products/es/es/enterprise.

14. Visual paradimg UML. [En línea] [Citado el: 12 de diciembre de 2011.] http://www.visual-paradigm.com/product/vpuml.

15. CodeBox. [En línea] [Citado el: 21 de enero de 2012.] http://www.codebox.es/glosario.

16. Sitio Oficial de APS. [En línea] [Citado el: 12 de diciembre de 2011.] http://www.asp.net.

17. Symfony Web Framework. [En línea] [Citado el: 3 de enero de 2012.] http://www.symfony-project.org/..

18. CodeIgniter - Open source PHP web application framework. [En línea] [Citado el: 3 de noviembre de 2011.] http://www.codeigniter.com/.

19. ExtJS. [En línea] [Citado el: 12 de diciembre de 2011.] http://www.sencha.com/.

20. Ecured Sistema Gestor de Base de Datos. [En línea] [Citado el: 23 de diciembre de 2011.] http://www.ecured.cu/index.php/Sistema_Gestor_de_Base_de_Datos.

21. **Quiñones, Ernesto.** PostgreSQLPE. [En línea] [Citado el: 2 de diciembre de 2011.] http://www.postgresql.org.pe/articles/introduccion_a_postgresql.pdf.

22. Conceptos básicos del servidor web. [En línea] [Citado el: 3 de diciembre de 2011.] http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php..

23. Conceptos básicos del servidor web. [En línea] http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php.

24. Eclipse: El IDE de desarrollo Open Source. [En línea] [Citado el: 6 de diciembre de 2011.] http://aplicaciones.org/eclipse-ide-de-desarrollo-open-source/.

25. NetBeans. [En línea] [Citado el: 13 de diciembre de 2011.] http://netbeans.org.

26. JetBRAINS. [En línea] [Citado el: 12 de diciembre de 2011.] http://www.jetbrains.com/idea.

27. Un vistazo a PHP5. [En línea] [Citado el: 12 de diciembre de 2011.] http://libertonia.escomposlinux.org/story/2004/7/15/115328/134..

28. geneura. [En línea] [Citado el: 21 de diciembre de 2011.] http://geneura.ugr.es/~jmerelo/tutoperl/tutoperl1.html.

29. Características y ventajas de las CSS. [En línea] [Citado el: 5 de diciembre de 2011.] http://www.desarrolloweb.com/articulos/182.php.

30. ¿Qué es Javascript? [En línea] [Citado el: 3 de diciembre de 2011.] http://www.maestrosdelweb.com/editorial/%C2%BFque-es-javascript/..

31. Requisitos funcionales. [En línea] [Citado el: 12 de enero de 2012.] http://www.eumed.net/libros/2010b/698/Requisitos%20funcionales.htm.

32. **Saavedra, Jorge.** El mundo informático. . [En línea] 8 de mayo de 2007. [Citado el: 2 de abril de 2012.] http://jorgesaavedra.wordpress.com/category/patrones-grasp/..

33. La prueba de aceptación es la prueba más importante para los productos software. [En línea] [Citado el: 23 de abril de 2012.]
http://pruebasdesoftware.com/pruebadeaceptacion.htm.