Universidad de las Ciencias Informáticas

Facultad 10

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

Título: Sistema para el control de la Base de Microbús de la Universidad de las Ciencias Informáticas.

Autores: Camilo Trujillo Pacheco
José Alejandro Roque Pérez

Tutor: Ing. Radel Calzada Pando

Ciudad de La Habana, Mayo de 2009

DECLARACIÓN DE AUTORÍA

Declaro ser el autor de la presente tesis y reconozco 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 _____.

Camilo Trujillo Pacheco

_____

Firma del Autor

José A. Roque Pérez

_____

Firma del Autor

Radel Calzada Pando

_____

Firma del Tutor

RESUMEN

El presente trabajo tiene como propósito diseñar e implementar un sistema que permita la administración del proceso de solicitud de transporte que de forma manual se desarrolla en la base de microbús de piquera de la Universidad de las Ciencias Informáticas, teniendo como resultado la centralización de la información relacionada a dicho proceso de solicitud. Su desarrollo está basado en tecnologías libres, multiplataforma y sobre una arquitectura en capas utilizando PHP 5 como lenguaje de programación; se implementa el patrón de arquitectura Modelo Vista Controlador (MVC) a través del CMS Drupal, MySQL como Sistema Gestor de Base de Datos y se hace uso de tecnologías AJAX (Asíncronos Java Script And XML) para realizar de forma más eficientes las peticiones al servidor. Valida la propuesta de solución a través del desarrollo de un conjunto de pruebas realizadas al producto final.

ÍNDICE DE CONTENIDO

INTRODUCCIÓN

Teniendo en cuenta que la Universidad de las Ciencias Informáticas desarrolla una serie de actividades fuera del centro que demandan el traslado de un grupo de personas, se hace necesario optimizar el uso de los medios de transporte para garantizar una mayor eficiencia en la prestación de estos servicios. Una vía importante para lograr dicha optimización es la sustitución del sistema manual de administración de microbús por uno automatizado.

El proceso que actualmente se lleva a cabo para la reservación de microbús es muy engorroso y no garantiza la calidad del servicio por lo que se hace necesario realizar una descripción detallada del mismo para su mejor entendimiento.

Previamente se realiza un proceso de reservación de micro. Para llevar a cabo este proceso en la universidad las personas envían al vicerrector correspondiente un correo solicitando el transporte con todas las características del viaje y éste le da la respuesta por correo si es aprobado o no, luego el vicerrector o su secretaria directamente le informan al chofer del carro asignado lo que debe hacer.

Dadas las ineficiencias que se presentan durante este trámite surge la necesidad de un sistema que permita automatizar todo ese proceso, lo que genera una situación problémica a la cual en nuestro trabajo de diploma nos proponemos enfrentar, dando respuesta al siguiente **problema científico:** ¿Cómo automatizar el proceso de reservación de Microbús en las áreas de la Vicerrectoría Económica, la Vicerrectoría Producción y la Oficina del Rector de la Universidad de las Ciencias Informáticas?

Teniendo como **objeto de estudio:** Desarrollo de sistemas informáticos.

Nuestro **campo de acción**: Desarrollo de sistemas informáticos para la gestión de reservaciones de transporte por microbús en la UCI.

Para darle solución al problema científico planteado anteriormente, se propone el siguiente **objetivo general:** Desarrollar un sistema para el control de la Base de Microbús de la Universidad de las Ciencias Informáticas.

Para el desarrollo de nuestra aplicación nos apoyamos en los siguientes **objetivos específicos**:

- Estudiar las herramientas y tecnologías a utilizar en el desarrollo del sistema.
- Diseñar e implementar un sistema informático para gestionar el proceso de reservación de Microbús.
- Realizar pruebas para verificar el correcto funcionamiento del sistema.

Para el desarrollo de la aplicación en correspondencia con los objetivos específicos se han propuesto las siguientes **tareas de investigación:**

- Investigación del proceso de reservación de Microbús en la Universidad de las Ciencias Informáticas.
- Estudio de las metodologías, tecnologías y herramientas de interés para desarrollar la aplicación, así como las tendencias en Cuba y en el mundo.
- Profundización sobre los estándares y equipamientos a utilizar para su mejor uso.

Aportes prácticos que se esperan del trabajo.

El sistema permitirá al Departamento de Transportación de la Universidad de las Ciencias Informáticas controlar el proceso de reservación de Microbús que se realizan en las diferentes áreas.
Este permitirá desarrollar diferentes funciones tales como reservar, modificar o cancelar reservaciones, además de mostrar listados de reservaciones existentes, ómnibus disponibles, así como realizar búsquedas avanzadas o simples y llevar un control de la gestión de usuarios y sus roles.

Métodos científicos de investigación.
Para el desarrollo de la investigación se utilizarán métodos teóricos y empíricos.

Métodos teóricos.
Análisis histórico-lógico: A través de este método se estudia la trayectoria real de los elementos que se utilizan en la implementación de sistemas Web, dígase origen y evolución de los diferentes lenguajes de

programación, frameworks, Entornos de Desarrollo Integrado (en lo adelante IDE) y Sistemas Gestores de Base de Datos.

Análisis y Síntesis: Para el desarrollo del sistema se realizó una investigación previa de los procesos que intervienen en el desarrollo de software y los principales elementos que integran las metodologías como son las fases, disciplinas, actividades, roles, artefactos y otros.

Técnica de investigación.

Entrevista: Se realizarán múltiples entrevistas a especialistas que laboran en las áreas del Departamento de Transportación y en las Vicerrectorías. Esto se logra a través del método Selección y Muestreo (Ver anexo 1).

CAPÍTULO I FUNDAMENTACIÓN TEÓRICA

Introducción

En este capítulo se abordan varios temas como el estado actual de los sistemas de reservación de ómnibus en el mundo, en nuestro país y en nuestra universidad, además se proponen las herramientas, metodología y recursos a utilizar en el desarrollo de la aplicación a defender en nuestro trabajo de diploma.

1.1 Estado del arte.

En el mundo de hoy el desarrollo de la web ha alcanzado un nivel muy alto, con el uso de la misma se brinda una serie de servicios para satisfacer las necesidades de los usuarios. Realizar solicitudes de transporte a tal medida que con un click el solicitante quede satisfecho es una de las motivaciones para prestar un servicio donde los usuarios puedan pedir un trasporte sin que tenga que salir de su casa, de su centro de trabajo o sencillamente de un lugar donde esté vacacionando. La aplicación internacional **www.trasladoscolombia.com** es un sistema que permite reservar ómnibus para viajes empresariales, de turismo, viajes fuera y dentro la ciudad, entre otros. Nuestro país cuenta con una serie de aplicaciones que no quedan atrás de las que en el mundo tienen mayor privilegio en lo que se refiere a la prestación de este importante servicio como **www.viazul.cu** y **www.transtur.cu,** que principalmente brindan servicios de viajes turísticos e interprovinciales, dando solución a una situación que con el paso de los años ha ido disminuyendo de forma gradual.

La Dirección de Transporte, escenario principal donde se gestionan las solicitudes de transporte dentro de la Universidad de las Ciencias Informáticas carece de un sistema que facilite la prestación de este servicio, generando así un problema al cual se le da solución con la realización del presente trabajo de diploma.

1.2 Herramientas a utilizar en el desarrollo de la aplicación.

1.2.1 CMS. Drupal.

Cuando se intenta desarrollar una aplicación web se necesita de un diseño e implementación de componentes de software que se usan regularmente. Muchas de las aplicaciones, necesitan autenticación

de usuarios, permisos, registros, diseño, publicación de noticias, etc. Estas funcionalidades llegan a convertirse en muchos casos, obstáculos para que la aplicación se realice en el tiempo requerido pues sus implementaciones son algo engorrosas. Pero actualmente se han creado alternativas para darle solución a estos problemas, los CMS reducen a gran escala el tiempo de creación de una aplicación web en dependencia del tipo de aplicación que se quiere realizar y las comodidades que brindan cada uno de estos.

CMS o Sistema de Gestión de Contenido, consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio. El sistema permite manejar de manera independiente el contenido y el diseño. Así, es posible manejar el contenido y darle en cualquier momento un diseño distinto al sitio sin tener que darle formato al contenido de nuevo, además de permitir la fácil y controlada publicación en el sitio de editores. (1)

Aspectos importantes de un CMS:

- ***Inclusión de nuevas funcionalidades en el web***. Esta operación puede implicar la revisión de multitud de páginas y la generación del código que aporta las funcionalidades. Con un CMS eso puede ser tan simple como incluir un módulo realizado por terceros, sin que eso suponga muchos cambios en la web. El sistema puede crecer y adaptarse a las necesidades futuras.
- ***Mantenimiento de gran cantidad de páginas***. En una web con muchas páginas hace falta un sistema para distribuir los trabajos de creación, edición y mantenimiento con permisos de acceso a las diferentes áreas. También se tienen que gestionar los metadatos de cada documento, las versiones, la publicación y caducidad de páginas y los enlaces rotos, entre otros aspectos.
- ***Reutilización de objetos o componentes***. Un CMS permite la recuperación y reutilización de páginas, documentos, y en general de cualquier objeto publicado o almacenado.
- ***Páginas interactivas***. Las páginas estáticas llegan al usuario exactamente como están almacenadas en el servidor web. En cambio, las páginas dinámicas no existen en el servidor tal como se reciben en los navegadores, sino que se generan según las peticiones de los usuarios. De esta manera cuando por ejemplo se utiliza un buscador, el sistema genera una página con los resultados que no existían antes de la petición. Para conseguir esta interacción, los CMS conectan con una base de datos que hace de repositorio central de todos los datos de la web.

- ***Cambios del aspecto de la web****.* Si no hay una buena separación entre contenido y presentación, un cambio de diseño puede comportar la revisión de muchas páginas para su adaptación. Los CMS facilitan los cambios con la utilización, por ejemplo, del estándar CSS (Cascading Style Sheets u hojas de estilo en cascada) con lo que se consigue la independencia de presentación y contenido.
- ***Consistencia de la web****.* La consistencia en una web no quiere decir que todas las páginas sean iguales, sino que hay un orden (visual) en vez de caos. Un usuario nota enseguida cuándo una página no es igual que el resto de las de la misma web por su aspecto, la disposición de los objetos o por los cambios en la forma de navegar. Estas diferencias provocan sensación de desorden y dan a entender que la web no la han diseñado profesionales. Los CMS pueden aplicar un mismo estilo en todas las páginas con el mencionado CSS, y aplicar una misma estructura mediante patrones de páginas.
- ***Control de acceso****.* Controlar el acceso a la web no consiste simplemente en permitir la entrada a esta, sino en gestionar los diferentes permisos a cada área de la web aplicados a grupos o individuos. (2)

CMS Drupal: Es un sistema de administración de información para páginas web desarrollado en PHP. Se distribuye bajo la licencia GNU GPL, y por tanto es software libre. Fue creado por el entonces estudiante universitario Dries Buytaert, no como un CMS, sino simplemente como un sitio con espacios colaborativos. El rápido crecimiento de su comunidad de desarrolladores y los valiosos aportes de esta a sus funcionalidades permitió que Buytaert lo lanzara con éxito al mercado de los CMS en el 2001. Los nodos de Drupal (que son su estructura básica) son altamente configurables y toda la estructura de datos es flexible, permitiendo establecer estructuras organizacionales y taxonómicas muy variadas: jerarquías simples, múltiples, etiquetamiento múltiple o vocabularios controlados.

Motivos para usar Drupal:

1. Cualquier diseño es posible con Drupal. Muchas veces la gente dice que un CMS pre configurado no tiene un diseño flexible y que puede limitar la creatividad de los diseñadores. Drupal te permite hacer cualquier cosa.

2. El concepto de usuario es intrínseco a Drupal. Esto quiere decir que si alguien se registra se registra para todas las aplicaciones que metas en Drupal. Desde foros, blogs, encuestas, UGCs, tienda, este control centralizado del usuario es clave para los proyectos de hoy en día.
3. Multi-idioma. Es normal tener un sitio en dos idiomas o más. Esta funcionalidad es natural en Drupal.
4. Abierto a PHP. Drupal por defecto te permite que en cualquier tipo de contenido puedas agregar tu PHP a medida. Este te permite retocar y personalizar la funcionalidad del back o del front.
5. Soporte de comunidad muy amplio. Quizás la principal ventaja de Drupal frente a Joomla es que Drupal tiene una comunidad muy buena. Drupal.org es el punto de encuentro. Cualquier duda está documentada y explicada.
6. Multimedia, Vídeo, Audio, UGCs. El soporte natural de Drupal está muy bien para este tipo de aplicaciones, pero si lo que te encuentras por defecto no te satisface, añade un módulo sin problemas.
7. Rápido y bajo consumo de servidor. El consumo de recursos del servidor que hace Drupal es muy bajo. Cualquier servidor (por barato que sea) podrá con Drupal y el servicio (con la cache activada) es muy rápido.
8. Control de administración. El control para los administradores es total. Respecto a tareas administrativas del sitio (informes de registros, informes de formularios, etc.) están por defecto disponibles y son muy fáciles de exportar como Excel por ejemplo puede ser "dame un informe de todos los usuarios registrados por provincias", igual esto en Drupal se hace muy sencillo. (3)

Hoy en día nuestro país se encuentra bajo un proceso de migración al software libre como alternativa para independizarnos tecnológicamente de los países más desarrollados en cuanto al tema de la informática. Drupal debido a que es una herramienta libre y de código abierto nos libera de muchas dependencias en lo que respecta a la creación de aplicaciones web. En nuestra universidad es uno de los CMS más usados por su gran facilidad de aprendizaje, además de tener una gran comunidad de desarrolladores y estar bajo la licencia GPL razones por la cual nos lleva a usar este CMS en la implementación de nuestro trabajo.

1.2.2 Servidor de Aplicaciones. Apache.

Un servidor de aplicaciones es un software que proporciona aplicaciones a los equipos o dispositivos cliente, por lo general a través de Internet y utilizando el protocolo http. Los servidores de aplicación se distinguen de los servidores web por el uso extensivo del contenido dinámico y por su frecuente integración con bases de datos.

Además, un **servidor de aplicaciones** es un producto basado en un componente que se encuentra en el plano medio de la arquitectura central de un servidor. Proporciona servicios de 'middleware', es decir, trabaja como un intermediario para la seguridad y el mantenimiento, además de proveer un fácil acceso a todos los datos.

Un servidor de aplicación maneja la mayoría de las transacciones relacionadas con la lógica y el acceso a los datos de la aplicación (esto se solía llamar 'centralización', hace algún tiempo). El término se utiliza para referirse a los servidores de aplicaciones basadas en la web, como el control de las plataformas de comercio electrónico integrado, sistemas de gestión de contenido de sitios web y asistentes o constructores de sitios de Internet. Por esta razón, algunos los llaman también 'servidor web'. (4)

El servidor Apache es un servidor web HTTP de código abierto para plataformas Unix, Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Su desarrollo comenzó en febrero de 1995, por Rob McCool. La primera versión apareció en enero de 1996, el Apache 1.0. Hacia el 2000, el servidor Web Apache era el más extendido en el mundo.

Es el servidor web hecho por excelencia, su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa.

Es un software que está estructurado en módulos, así los programadores asumen que el software puede ser ampliado por otros desarrolladores, los cuales pueden escribir pequeñas partes del código que se integrará en Apache de una manera fácil. Esto se lleva a cabo al haber creado un API modular y una serie de fases bien definidas por las que cada petición al servidor debe atravesar. Estas fases van desde la inicialización del servidor (cuando Apache lee los ficheros de configuración), hasta la traducción de una URL en un nombre de fichero del servidor, o registrar los resultados de la transacción.

La configuración de cada módulo de Apache se hace mediante la configuración de las directivas que están contenidas dentro del módulo. Los módulos se pueden clasificar en tres categorías:

- **Módulos Base:** Módulo con las funciones básicas del Apache.
- **Módulos Multiproceso:** son los responsables de la unión con los puertos de la máquina, acepando las peticiones y enviando a los hijos a atender a las peticiones.
- **Módulos Adicionales:** Cualquier otro módulo que le añada una funcionalidad al servidor.

Las funcionalidades más elementales se encuentran en el módulo base, siendo necesario un módulo multiproceso para manejar las peticiones. Se han diseñado varios módulos multiproceso para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código. El resto de funcionalidades del servidor se consiguen por medio de módulos adicionales que se pueden cargar para añadir un conjunto de utilidades al servidor sin volver a instalar el software. (5)

Debido a su gran cantidad de usuarios en el mundo entero en el tema de la web, en nuestra universidad también es uno de los servidores web más utilizados para el desarrollo de aplicaciones, por lo que se toma la decisión de usar este servidor para el desarrollo de la aplicación propuesta en nuestro trabajo de diploma.

1.2.3 MySQL.

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. Desde enero del 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009 desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSIC. MySQL es un gestor de base de datos sencillo de usar e increíblemente rápido. También es uno de los motores de base de datos más usados en Internet, la principal razón de esto es que es gratis para aplicaciones no comerciales.

Las características principales de MySQL son:

- **Es un gestor de base de datos**. Una base de datos es un conjunto de datos y un gestor de base de datos es una aplicación capaz de manejar este conjunto de datos de manera eficiente y cómoda.

- **Es una base de datos relacional**. Una base de datos relacional es un conjunto de datos que están almacenados en tablas entre las cuales se establecen unas relaciones para manejar los datos de una forma eficiente y segura. Para usar y gestionar una base de datos relacional se usa el lenguaje estándar de programación SQL.

- **Es Open Source**. El código fuente de MySQL se puede descargar y está accesible a cualquiera, por otra parte, usa la licencia GPL para aplicaciones no comerciales.

- **Es una base de datos muy rápida**, segura y fácil de usar. Gracias a la colaboración de muchos usuarios, la base de datos se ha ido mejorando optimizándose en velocidad. Por eso es una de las bases de datos más usadas en Internet.

- **Existe una gran cantidad de software que la usa**, es muy popular en el mundo entero debido a la adaptabilidad a las necesidades de los usuarios. (6)

MySQL es muy utilizado en aplicaciones web. Es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones. MySQL usa el GPL (GNU General Public License) para definir qué puede hacer y que no puede hacer con el software en diferentes situaciones. Por su gran usabilidad en el mundo entero así como en la UCI se decide utilizar este gestor en el desarrollo del presente trabajo.

1.2.4 HTML y CSS.

1.2.4.1 HTML

Es el acrónimo inglés de HyperText Markup Language, que se traduce al español como *Lenguaje de Etiquetas de Hipertexto*. Es el lenguaje más utilizado para la presentación de textos estructurados en formato hipertexto, estándar de las páginas web. HTML es utilizado por la práctica totalidad de navegadores web del mercado con el fin de presentar al visitante de una página web el contenido de la misma tal como el diseñador quiere que se muestre a su público. Gracias a Internet y a los navegadores como Internet Explorer, Opera, Firefox, Netscape o Safari, el HTML se ha convertido en uno de los formatos más usados y fáciles de aprender que existen para la elaboración de documentos para web.

En realidad HTML no es un lenguaje de programación si no que son sentencias-etiquetas de las cuales se indican que operaciones se van a realizar con el texto o con los atributos que se estén manejando con esa sentencia-etiqueta, la verdad estas secuencias son muy necesarias para el diseño de una página web ya que cada una de esas sentencias le indican a internet como está compuesta la estructura de cualquier Web.

El lenguaje HTML contiene dos partes:

- El contenido, que es el texto que se verá en la pantalla de un ordenador,
- Las etiquetas y atributos que estructuran el texto de la página web en encabezados, párrafos, listas, enlaces, etc. y normalmente no se muestra en pantalla. (7)

En la aplicación que se desarrolló se hizo uso del lenguaje HTML para crear la interfaz gráfica. Este lenguaje puede ser creado y editado con cualquier editor de textos básico, como puede ser Gedit en Linux, el Bloc de Notas de Windows, o cualquier otro editor que admita texto sin formato como GNU Emacs, Microsoft Wordpad, TextPad, Notepad++, entre otros.

1.2.4.2 CSS.

CSS (Cascading Style Sheets, u Hojas de Estilo en Cascada), es una tecnología que nos permite crear páginas web de una manera más exacta. Permite incluir márgenes, tipos de letra, fondos, colores. Además

ayuda a mejorar el posicionamiento Web**.** Adicionalmente se podrá cambiar rápidamente los estilos de ciertas palabras, modificando la importancia que les quieres otorgar ante los robots de búsqueda.

También, aunque en el orden de la página los elementos se muestren en un orden (Ej: cabecera, menú superior, menú izquierdo, contenido principal), a través de CSS podemos lograr que en el código (lo que ven los buscadores) aparezcan los elementos en el orden que deseemos (contenido principal, menú izquierdo, menú principal, cabecera). Esto es muy importante ya que ante los buscadores el contenido más importante es el que primero aparece en la página. (8)

Principales ventajas de las hojas de estilo:

- Control del diseño
- Redefinición de etiquetas
- Uso de etiquetas para su misión
- Personalización
- Maquetación y accesibilidad
- Respeto a los estándares
- Tamaño
- Usabilidad
- Visibilidad

El modo de funcionamiento de las CSS consiste en definir, mediante una sintaxis especial, la forma de presentación que le aplicaremos a:

- Un web entero, de modo que se puede definir la forma de todo el web de una sola vez.
- Un documento HTML o página, se puede definir la forma, en un pequeño trozo de código en la cabecera, a toda la página.
- Una porción del documento, aplicando estilos visibles en un trozo de la página.
- Una etiqueta en concreto, llegando incluso a poder definir varios estilos diferentes para una sola etiqueta. Esto es muy importante ya que ofrece potencia en nuestra programación. Podemos

definir, por ejemplo, varios tipos de párrafos: en rojo, en azul, con márgenes, sin márgenes y muchos más. (9)

La potencia de la tecnología salta a la vista. Pero no solo se queda aquí, ya que además esta sintaxis CSS permite aplicar al documento formato de modo mucho más exacto. Si antes el HTML se nos quedaba corto para maquetar las páginas y teníamos que utilizar trucos para conseguir nuestros efectos, ahora tenemos muchas más herramientas que nos permiten definir esta forma:

- Podemos definir la distancia entre líneas del documento.
- Se puede aplicar identado a las primeras líneas del párrafo.
- Podemos colocar elementos en la página con mayor precisión, y sin lugar a errores.
- Definir la visibilidad de los elementos, subrayados, tachados.

Para que el diseño de la aplicación fuese más eficiente y amigable se hizo uso de la hoja de estilo debido a que le ahorra trabajo al navegador, pues el mismo podrá indexar los documentos con mayor rapidez. Además, se logra aumentar la densidad de las palabras claves ya que se reduce la utilización de etiquetas.

1.2.5 JQuery.

JQuery es una librería javascript de código abierto que ayuda a desarrollar funcionalidades complejas tanto en javascript como en Ajax (Asynchronous Javascript + XML). Facilita la interacción de los diferentes códigos cliente y proporciona un desarrollo basado en objetos y modular para realizar aplicaciones más interactivas y con comportamientos más aparentes. Así que es una librería a utilizar tanto por aquellos que empiezan con javascript, ya que es más sencillo de utilizar para desarrollar Ajax, como por aquellos que tengan experiencia utilizando lenguajes de cliente, en este caso facilitará el desarrollo y acelerará los tiempos de implementación.

La librería JQuery en resumen aporta para la aplicación desarrollada las siguientes ventajas:

- Ahorra muchas líneas de código.
- Hace transparente el soporte de nuestra aplicación para los navegadores principales.
- Provee de un mecanismo para la captura de eventos.

- Provee un conjunto de funciones para animar el contenido de la página en forma muy sencilla.
- Integra funcionalidades para trabajar con AJAX.

A través de JQuery se puede acceder al código DOM (Document Object Model) de las páginas HTML o XHTML y manipular sus elementos, así como cambiar propiedades de las CSS, reaccionar ante eventos o facilitar el uso de efectos y transiciones, entre otras cosas. Además, es modular y se puede extender mediante la gran cantidad de plugins existentes que complementan y habilitan funcionalidades extra que van desde efectos drag y drop a multitud de presentaciones de datos en forma de carrusel, de acordeón, etc. (10)

1.2.6 JavaScript.

Javascript es un lenguaje de programación que permite a los desarrolladores crear acciones en sus páginas web. Es un lenguaje que puede ser utilizado por profesionales y para quienes se inician en el desarrollo y diseño de sitios web. No requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos.

Entre los diferentes servicios realizados con Javascript en Internet se encuentran:

- Correo
- Chat
- Buscadores de Información

También podemos encontrar o crear códigos para insertarlos en las páginas como:

- Reloj
- Contadores de visitas
- Fechas
- Calculadoras
- Validadores de formularios
- Detectores de navegadores e idiomas

Javascript es un lenguaje con muchas posibilidades, es utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes. Con el mismo podemos crear diferentes efectos e interactuar con nuestros usuarios.

JavaScript proporciona los medios para:

- Controlar las ventanas del navegador y el contenido que muestran.
- Programar páginas dinámicas simples sin tener que matar moscas a cañonazos de Java.
- Evitar depender del servidor Web para cálculos sencillos.
- Capturar los eventos generados por el usuario y responder a ellos sin salir a Internet.
- Simular el comportamiento de las macros CGI cuando no es posible usarlas.
- Comprobar los datos que el usuario introduce en un formulario antes de enviarlos.
- Comunicarse con el usuario mediante diversos métodos.

La característica que más simplifica la programación es que, aunque el lenguaje soporta cuatro tipos de datos, no es necesario declarar el tipo de las variables, argumentos de funciones ni valores de retorno de las funciones. El tipo de las variables cambia implícitamente cuando es necesario, lo que dificulta el desarrollo de programas complejos, pero ayuda a programar con rapidez macros sencillas. En esto, JavaScript se separa totalmente de lenguajes como C, C++ o Java.

Como síntesis se pude decir que Javascript es un lenguaje interpretado, basado en prototipos, por lo que en el desarrollo de esta aplicación su uso está enmarcado en la validación de formularios y otras funciones del sistema, asegurando así la entrada válida de datos en el sistema.

1.2.7 PHP.

PHP o Personal Home Page, es un lenguaje de programación interpretado del lado del servidor, diseñado originalmente para la creación de aplicaciones web dinámicas, aunque actualmente puede ser usado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas QT (Quásar Technologies) o GTK+ (The GIMP Toolkit) que no son más que un conjunto de bibliotecas multiplataforma para desarrollar interfaces

gráficas de usuario. Fue creado originalmente en el año 1994 por Rasmus Lerdorf como un CGI en C que permitía la interpretación de un número ilimitado de comandos.

PHP dispone de una gran cantidad de características por las cuales se ha convertido en el lenguaje de programación ideal para la creación de aplicaciones web dinámicas:

- La principal característica por la cual se utilizó php como lenguaje de programación es precisamente porque Drupal está totalmente implementado con dicho lenguaje.
- Se pueden hacer grandes cosas con pocas líneas de código. Lo que hace que merezca la pena utilizarlo.
- Viene acompañado por una excelente biblioteca de funciones que permite realizar cualquier labor (acceso a base de datos, encriptación, envió de correo, gestión de un comercio electrónico, XML, creación de PDF).
- Es utilizado con éxito en varios millones de sitios web.
- Es multiplataforma, funciona en todas las plataformas que soporten apache.
- Es software libre. Se puede obtener en la web y su código está disponible bajo la licencia GPL. Este es uno de los aspectos fundamentales para la elección de este lenguaje.
- Soporte para una gran cantidad de bases de datos como MySQL, PostgreSQL, Oracle, LDAP, etc.
- Soporte para acceso a servidores IMAP y FTP, envío de correo con SMTP, acceso a SNMP para gestión de redes y equipos, etc.
- Perceptiblemente más fácil de mantener al día que el código desarrollado en otros lenguajes.
- Soportado por una gran comunidad de desarrolladores, como producto de código abierto, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente.
- El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.
- Con PHP se puede procesar información en formularios, crear foros de discusión, manipulación de cookies y paginas dinámicas. (12)

En la universidad es muy utilizado este lenguaje por su facilidad de uso. Poco a poco este lenguaje se ha ido promocionando para el desarrollo de sitios web. Además un gran por ciento de la institución utiliza los

CMS, y casi todos están implementados y soportan este lenguaje, por otra parte hoy en día se ha hecho necesario en el país la migración al software libre. Las nuevas versiones ya soportan la programación orientada a objeto así como todos sus paradigmas, su código puede ser embebido dentro del código HTML pudiendo crear objetos dinámicos en tiempo de ejecución. Es por todo este conjunto de características PHP ofrece ser la mejor opción para crear un sistema potente y seguro, por lo que se decidió utilizarlo para la realización de la aplicación que defiende nuestro trabajo de diploma.

1.2.8 Herramientas CASE

Las herramientas Case son un conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo del sistema de información, completamente o en algunas fases. También pueden mejorar la productividad en el desarrollo de una aplicación de bases de datos. Y por productividad se entiende tanto la eficiencia en el desarrollo, como la efectividad del sistema desarrollado.
La eficiencia se refiere al costo, tanto en tiempo como en dinero, de desarrollar la aplicación. La efectividad se refiere al grado en que el sistema satisface las necesidades de los usuarios. Para obtener una buena productividad, subir el nivel de efectividad puede ser más importante que aumentar la eficiencia.

Las herramientas fueron diseñadas para:

- Soportar un entorno personal dedicado.
- Utilizar gráficos para especificar y documentar los sistemas.
- Unir todas las fases del ciclo del software.
- Utilizar la inteligencia artificial para realizar automáticamente muchas de las rutinas, tareas de desarrollo y mantenimiento del software.

Las herramientas Case se diferencian de otro software en el eventual desarrollo de elementos automatizados de modo personal por el usuario, en entorno gráficos y no gráficos, a su vez la herramientas Case están orientadas a los desarrolladores de software y no al cliente final. La tendencia de dichas herramientas es proporcionar un conjunto de ellas bien integrado y que ahorren trabajo, enlazando y automatizando todas las fases del ciclo de vida del software. (13)

1.2.8.1 Lenguaje Unificado de Modelado (UML)

En todas las disciplinas de la Ingeniería se hace evidente la importancia de los modelos ya que describen el aspecto y la conducta de "algo". Ese "algo" puede existir, estar en un estado de desarrollo o estar en un estado de planeación. Es en este momento cuando los diseñadores del modelo deben investigar los requerimientos del producto terminado y dichos requerimientos pueden incluir áreas tales como funcionalidad, performance y confiabilidad. Además, a menudo, el modelo es dividido en un número de vistas, cada una de las cuales describe un aspecto específico del producto o sistema en construcción.

El modelado sirve no solamente para los grandes sistemas, aun en aplicaciones de pequeño tamaño se obtienen beneficios de modelado, sin embargo es un hecho que entre más grande y más complejo es el sistema, más importante es el papel de que juega el modelado por una simple razón: "El hombre hace modelos de sistemas complejos porque no puede entenderlos en su totalidad".

Los principales beneficios de UML son:

- Mejores tiempos totales de desarrollo (de 50 % o más).
- Modelar sistemas (y no sólo de software) utilizando conceptos orientados a objetos.
- Establecer conceptos y artefactos ejecutables.
- Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- Mejor soporte a la planeación y al control de proyectos.
- Alta reutilización y minimización de costos.

UML es un lenguaje para hacer modelos y es independiente de los métodos de análisis y diseño. Existen diferencias importantes entre un método y un lenguaje de modelado. Un método es una manera explícita de estructurar el pensamiento y las acciones de cada individuo. Además, el método le dice al usuario qué hacer, cómo hacerlo, cuándo hacerlo y por qué hacerlo; mientras que el lenguaje de modelado carece de estas instrucciones. Los métodos contienen modelos y esos modelos son utilizados para describir algo y comunicar los resultados del uso del método. (14)

UML es ahora un estándar, su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que UML ha sido diseñado para modelar cualquier tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otra rama.

1.2.8.2 Visual Paradigm

Visual Paradigm para UML es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

Visual Paradigm permite escribir toda la especificación de un caso de uso sin necesidad de utilizar una herramienta externa como editor de texto. Es posible crear especificaciones de casos de uso utilizando plantillas que se encuentran definidas, o que pueden ser creadas por los usuarios. (15)

Con la utilización de esta herramienta se pudo realizar un profundo análisis del desarrollo de la aplicación, mediante su uso se construyeron diagramas que permiten un mejor entendimiento del proyecto así como generar la documentación que soporta el ciclo de vida del mismo.

1.2.9 Herramientas de Desarrollo para PHP. NetBeans

Los IDEs (Integrated Development Environment) son un conjunto de herramientas para el programador, que suelen incluir en una misma suite, un buen editor de código, administrador de proyectos y archivos, enlace transparente a compiladores y debuggers e integración con sistemas controladores de versiones o repositorios.

NetBeans es un IDE de código abierto ("open source") para desarrolladores. Posee todas las herramientas necesarias para crear aplicaciones web, de escritorio y para teléfonos móviles con los lenguajes de programación Java, C, C++ e incluso lenguajes dinámicos como: PHP, Javascript, Groovy, y Ruby. Es fácil de usar, instalar y puede ser ejecutado en múltiples plataformas como Windows, Linux, Mac OS X y Solaris.

NetBeans IDE 6.5 soporta todas las características estándar, como el autocompletado de código, los sucesos marca de resaltado de sintaxis, refactoring, plantillas de código, documentación de pop-up, navegación de código, las advertencias de editor y lista de tareas.

Entre sus principales características podemos mencionar:

- Brinda completamiento automático de código PHP, así como coloreado de código sintáctico y semántico.
- Permite depurar el código usando Xdebug.
- Genera fragmentos de código para bases de datos MySQL. (15)

Dadas las características, ventajas y soporte que brinda este potente entorno de desarrollo y sobre todo por ser su licencia “open source”, se eligió como IDE a utilizar para el desarrollo del sistema a implementar, además emplearemos algunos de los software más usados como ZendStudio y Dreamweaver que al mismo tiempo tienen similares características pero son software privativos, estando así en contra de la política seguida para el desarrollo de nuestra aplicación.

1.3 Metodología para el desarrollo de software.

1.3.1 Extreme Programming (XP).

La Programación Extrema surge ideada por Kent Beck, como proceso de creación de software diferente al convencional. En palabras de Beck: "XP es una metodología ligera, eficiente, con bajo riesgo, flexible, predecible y divertida para desarrollar software". Fue probado en distintas empresas como Bayerische Landesbank, Credit Swiss Life, DaimlerChrysler, First Union National Bank, Ford Motor Company, UBS. Acepta cambios de requerimientos aún tardíos en el ciclo de desarrollo, integra gerentes, clientes y desarrolladores en la búsqueda de calidad en el software. Mejora el proyecto en comunicación, simplicidad, realimentación y emprendimiento, mantiene el diseño simple y claro, ensaya el software desde el primer día, entrega temprano e implementa los cambios al ir siendo sugeridos.

La Programación Extrema introduce un cambio en la forma de programar, no ahorra en hardware, construye programas entendibles y extensibles. Automatiza las pruebas: escribe código de prueba antes, durante y después de la programación: antes, al definir la funcionalidad, durante, al descubrir errores,

después, en la integración. No tropieza dos veces con la misma pulga (bug). Los clientes ven el software tempranamente y es una revalorización del software con mayor énfasis en la calidad.

Metodológicamente cuestiona la forma actual de producir software, con realimentación tardía, prueba al final, descubrimiento de errores en producción. Reexamina las prácticas corrientes de desarrollo y es una de las varias "metodologías liviana" surgidas para bajar costos de desarrollo.

Cuándo usar XP.

- Cuando los clientes no tienen idea clara de los requerimientos y los van cambiando.
- Para proyectos de riesgo: fecha fija de entrega, algo nunca hecho por el grupo, algo nunca hecho por la comunidad de desarrolladores.
- Entre 2 y 10 programadores. No es apto para proyectos con mucho personal.
- Integra gerentes y clientes a la formulación de preguntas, la negociación de cronograma y alcances, la creación de las pruebas.
- Automatiza las pruebas; es posible en casi todos los dominios. Es lícito repensar el diseño para facilitar el ensayo.
- El objetivo es entregar el software tal cual se necesita y en el momento en que se necesita. Incidentalmente, los proyectos XP muestran mayor productividad.

Características Fundamentales.

- **Desarrollo iterativo e incremental**: pequeñas mejoras, unas tras otras.
- **Pruebas unitarias continuas**, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación. Se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándonos en algo hacia el futuro, podamos hacer pruebas de las fallas que pudieran ocurrir. Es como si nos adelantáramos a obtener los posibles errores.
- **Programación en parejas**: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera

es más importante que la posible pérdida de productividad inmediata (el código es revisado y discutido mientras se escribe).

- **Integración del equipo de programación con el cliente** o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- **Corrección de todos los errores** antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- **Refactorización del código**, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- **Propiedad del código compartida**: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.
- **Simplicidad en el código**: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.
- **Uso de Metáforas**: La comunicación fluida es uno de los valores más importantes de la Programación Extrema, el hecho de incorporar al equipo una persona que represente los intereses del negocio y otras prácticas son valiosas entre otras cosas porque potencian enormemente la comunicación. Para conseguir que la comunicación sea fluida es imprescindible, entre otras cosas, utilizar el vocabulario del negocio. También es fundamental huir de definiciones abstractas. Dicho de otro modo, la XP no pretende seguir la letra de la ley, sino su espíritu. Dentro de este enfoque es fundamental buscar continuamente metáforas que comuniquen intenciones y resulten descriptivas, enfatizando el qué por delante del cómo.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Cuanto más simple es el sistema, menos tendrá que comunicar sobre éste, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.

Objetivos de XP:

La satisfacción del cliente. Esta metodología trata de dar al cliente el software que él necesita y cuando lo necesita. Por tanto, debemos responder muy rápido a las necesidades del cliente, incluso cuando los cambios sean al final de ciclo de la programación.

Potenciar al máximo el trabajo en grupo. Tanto los jefes de proyecto, los clientes y desarrolladores, son parte del equipo y están involucrados en el desarrollo del software.

Bases de XP.

La programación extrema se basa en la simplicidad, la comunicación y el reciclado continuo de código, para algunos no es más que aplicar una pura lógica. Lo que buscan en definitiva es la reducción de costes.

Variables XP.

XP define cuatro variables para proyectos de software:

- Coste
- Tiempo
- Calidad
- Ámbito.

Características esenciales de XP.

- ***Roles.***

- **Programador*:*** Produce el código del sistema.
- **Cliente*:*** Escribe las historias de usuario y las pruebas funcionales para validar su implementación, así como asigna la prioridad de la historia de usuario y decide cuál se implementara en cada iteración.
- **Encargado de pruebas*:*** Ayuda al cliente a escribir las pruebas funcionales, ejecuta las pruebas y difunde resultados además es responsable de las herramientas de soporte para prueba.

- **Rastreador (Tracker):** también conocido como "Metric Man", observa sin molestar y mantiene los datos históricos.
- **Consultor:** Es un miembro externo del equipo con conocimiento especifico de algún tema necesario para el proyecto. Guía al equipo para resolver un problema específico.
- **Gestor (Big Boss):** Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación.

➢ ***Artefactos esenciales.***

- Historias de Usuario.
- Tareas de Ingeniería.
- Pruebas de Aceptación.

➢ ***Procesos.***

- Ciclo de desarrollo.
- Ciclo de vida (Fases).

Extreme Programming es un éxito porque hace hincapié en la satisfacción del cliente. Faculta a los desarrolladores con confianza responder a las cambiantes necesidades de los clientes, incluso a finales del ciclo de vida. Hace hincapié en el trabajo en equipo, los administradores, clientes y desarrolladores son socios iguales en un equipo de colaboración. Implementa un equipo simple, pero efectivo medio ambiente propicio para llegar a ser altamente productiva, el equipo se auto-organiza en torno al problema a resolver es la forma más eficiente posible. (16)

Mejora un proyecto de software en cinco aspectos esenciales: la comunicación, la sencillez, la retroalimentación, el respeto y el coraje. Programadores Extreme constantemente se comunican con sus clientes y colegas programadores, mantienen su diseño sencillo y limpio, reciben retroalimentación mediante pruebas de su software a partir del primer día. Cada pequeño éxito profundiza su respeto por las contribuciones únicas de cada uno y cada miembro del equipo. Con esta base Extreme programadores son capaces de responder con valentía a las cambiantes necesidades y la tecnología.

1.3.2 Rational Unified Process (RUP)

El Proceso Unificado es un proceso de construcción de software que plantea el desarrollo de un conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema. El Rational Unified Process o de manera simplificada, RUP; es un marco de trabajo genérico que puede especializarse en una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipo de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto.

Es el resultado de varios años de desarrollo y uso práctico en el que se han unificado técnicas de desarrollo, a través del UML, y trabajo de muchas metodologías utilizadas por los clientes. Define Quién debe hacer Qué, Cuándo y Cómo debe hacerlo. RUP define entre sus principales elementos:

Trabajadores ("quién") Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.

Actividades ("cómo") es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos. Artefactos ("qué") Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.

Flujo de actividades ("Cuándo") secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable. Está compuesto por cuatro fases y nueve flujos de trabajo, los cuales son:

Fases en el desarrollo de software.

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

Flujos de trabajo.

- **Modelamiento del negocio:** Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.

- **Requerimientos:** Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
- **Análisis y diseño:** Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.
- **Implementación:** Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
- **Prueba (Testeo):** Busca los defectos a los largo del ciclo de vida.
- **Instalación:** Produce release del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.
- **Administración del proyecto:** Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
- **Administración de configuración y cambios:** Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.
- **Ambiente:** Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

Características fundamentales de RUP.

Dirigido por casos de uso

Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo, ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso (cómo se llevan a cabo).

Centrado en la arquitectura

La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción.

Iterativo e Incremental

RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Es práctico dividir el trabajo en partes más pequeñas o mini proyectos. Cada mini proyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto.

Dadas las características de estas excelentes metodologías para el desarrollo de software y el uso frecuente de las mismas en proyectos de nuestra universidad, se ha decidido aplicar XP(Extreme Programming) a nuestro trabajo de diploma para obtener un producto sencillo y eficaz que materialice nuestro objetivo general y resuelva el problema que se plantea.

Conclusiones

En este capítulo se ha hecho un análisis de todos los lenguajes, metodologías y herramientas propuestas las cuales son de vital importancia para darle solución al problema que se plantea. El dominio de estas herramientas nos brinda la mejor forma de aplicarlas con el objetivo de desarrollar un sistema de máxima calidad que cumpla con los requisitos propuestos y de al cliente una versión del producto que satisfaga sus intereses.

CAPÍTULO II CARACTERÍSTICAS DEL SISTEMA

Introducción

El capítulo actual tiene como objetivo hacer una valoración de las características principales del sistema a desarrollar, para lo cual se tendrá en cuenta la situación problémica que dio origen al mismo. Se detallan las necesidades de los usuarios, describiéndose las funcionalidades que serán objeto de automatización. Finalmente se presenta una propuesta del software a implementar, especificando detalladamente los requerimientos funcionales y no funcionales.

2.1 Objeto de Automatización.

Durante el período de desarrollo existen varios procesos que deben ser automatizados, puesto que consumen una valiosa porción de tiempo a los desarrolladores. Muchos de estos procesos son de vital importancia para lograr un nivel de calidad óptimo en los productos finales.

Se desea automatizar el control de las solicitudes de reservación de transporte proveniente de las áreas de la Vicerrectoría Económica, la Vicerrectoría de Producción y la Oficina del Rector en la base de microbús de la UCI, centralizando la información relacionada a dichas solicitudes, lo que permitirá un fácil acceso y comodidad a los usuarios brindando así un mejor servicio.

2.2 Personas Relacionadas con el Sistema.

Se denomina persona relacionada con el sistema a aquella que interactúa e intercambia con el sistema y obtiene resultados de los procesos desarrollados en la aplicación. Además aquella que interactúa con la misma sin poder hacer uso de las secciones privilegiadas del sistema. En la siguiente tabla se detalla una breve descripción sobre las actividades que puede realizar una persona con la aplicación.

Personas relacionadas con el sistema	Justificación
Administradores	Son las personas facultadas para la gestión del sistema. Son los encargados de administrar las diferentes cuentas de los usuarios autenticados en la aplicación. Cuentan además con todos los privilegios.

Jefes de áreas	Son los encargados en las áreas de la Vicerrectoría Económica, la Vicerrectoría de Producción y la Oficina del Rector de gestionar las solicitudes de reservación de trasporte por microbús, así como los vehículos pertenecientes a dichas aéreas, actualizando e intercambiando información con el sistema.
Usuario autenticado	Es la persona que navega por el sistema con una serie de privilegios lo que le permite realizar un grupo de actividades como realizar, cancelar y modificar solicitudes de trasporte.

Tabla 2.1 Personas relacionadas con el sistema.

2.3 Requisitos Funcionales del Sistema.

Los requisitos funcionales son las principales funciones que debe cumplir el sistema para darle solución a la problemática planteada en el presente trabajo de diploma. A continuación se describen cada uno de ellos.

- Autenticar usuario: Interfaz mediante la cual el usuario se autentica en el sistema.
- Gestionar usuario: El administrador del sistema elimina o agrega un usuario, dándole al mismo un rol determinado o modificando los privilegios de este en la aplicación.
- Insertar vehículo: En esta interfaz el administrador del sistema podrá insertar en la base de datos un vehículo que se le da de alta por ingresar nuevo a la base de microbús.
- Eliminar vehículo: Se podrá eliminar vehículos de la base de datos por causar baja de la base de microbús, por rotura u otro problema que imposibilite su funcionamiento.
- Gestionar consumo de combustible: Interfaz que permite gestionar el consumo de combustible del vehículo en un periodo de tiempo.
- Listar solicitudes: Se mostrará una lista de las solicitudes realizadas. El usuario podrá realizar modificaciones en las solicitudes hechas por él así como cancelarlas.
- Realizar solicitud: Se muestra un formulario donde el usuario especifica los datos de la solicitud que desea realizar.
- Cancelar Solicitud: El usuario debe de especificar en un formulario la solicitud que desea cancelar. Una vez cancelada la solicitud es eliminada automáticamente de la base de datos.
- Realizar reportes: A través de esta interfaz el administrador del sistema podrá generar una serie de reportes que serán útiles para atender a las solicitudes que realizan los usuarios en la aplicación.

2.4 Requisitos no Funcionales del Sistema.

Los requerimientos no funcionales especifican criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los requisitos funcionales. Por tanto, se refieren a todos los requisitos que ni describen información a guardar, ni funciones a realizar. Estos requisitos responden a las características del sistema en cuanto a funcionalidad, usabilidad, confiabilidad, compatibilidad con hardware y software, especificaciones del producto, etc.

Interfaz de usuario: La aplicación presenta una interfaz sencilla de manejar y agradable que permite el acceso a cualquier parte de la misma.

Requerimientos de apariencia o interfaz externa

La aplicación propuesta será usada por personas que tengan o no conocimientos básicos de informática, por lo que la interfaz es amigable, sencilla y fácil de usar, de manera que no es una dificultad para los usuarios el trabajo con la misma. No tiene animaciones ni imágenes pesadas que obstaculizan la rapidez del sistema.

Requerimientos de usabilidad

A los administradores del sistema se les dará un adiestramiento básico para el uso de la aplicación. Estas personas tendrán un nivel de acceso amplio en la aplicación para poder darle respuesta a cada incidente ocurrido.

Requerimientos de rendimiento

Para un funcionamiento óptimo de la aplicación se seguirán las diferentes técnicas de elaboración de sitios Web, que faciliten el acceso rápido a sus páginas. La eficiencia del producto estará determinada en gran medida por el aprovechamiento de los recursos que se disponen en el modelo cliente/servidor y la velocidad de la consultas de la base de datos. La aplicación propuesta debe ser rápida y el tiempo de respuesta debe ser el mínimo posible, adecuado a la rapidez con que el cliente requiere la respuesta a su petición.

Requerimiento de portabilidad

Las herramientas utilizadas podrán ser usadas bajo cualquier sistema operativo Windows NT en adelante o cualquier distribución de Linux. El servidor Web y el servidor de Base de Datos pueden estar en la misma PC sin ocasionar problema alguno.

Requerimientos de seguridad

Confiabilidad: La información manejada por el sistema debe estar protegida de acceso no autorizado.

Integridad: La información manejada por el sistema debe ser objeto de cuidadosa protección contra la corrupción y estados de inconsistencia.

Disponibilidad: La aplicación deberá estar disponible en todo momento para aquellas personas con acceso a la información y los mecanismos utilizados para lograr la seguridad no deben ser un obstáculo a los usuarios para obtener los datos deseados en un momento dado.

Requerimientos de software

En las computadoras de los usuarios solo se requiere un navegador Web, bajo cualquier sistema operativo Windows NT en adelante o cualquier distribución de Linux. En el servidor de base de datos se requiere Windows NT en adelante o cualquier distribución de Linux. Para su implementación se utilizará cualquier herramienta de desarrollo y como gestor de base de datos MySQL.

Requerimientos de hardware

En el cliente se requiere una máquina con 128 MB de RAM como mínimo, el servidor Web junto con el servidor de base de datos debe tener 256 MB de RAM y 20 GB de disco duro mínimo, todas las maquinas implicadas en la funcionalidad de la aplicación deben estar conectadas a la red de al menos 100 Mbps de velocidad.

Restricción en el diseño y la implementación

Se debe realizar una aplicación que permite dar respuesta en el menor tiempo posible, garantizando la calidad del sistema apoyado completamente en el CMS Drupal. Para garantizar el desarrollo de la aplicación se utilizará la metodología XP. Para realizar los modelos se utilizó UML y como herramienta de apoyo a este lenguaje de modelación se utiliza Visual Paradigm.

Conclusiones

Al finalizar el presente capitulo se ha presentado la descripción del sistema y el conjunto de actividades que se realizan con el mismo, así como el grupo de funciones que se pueden realizar con dicho sistema, expresando los requerimientos funcionales y no funcionales, lo que nos permite comenzar con la propuesta de nuestra aplicación, velando por el cumplimiento de todas las pautas consideradas.

CAPÍTULO III EXPLORACIÓN Y PLANIFICACIÓN

Introducción

En el presente capítulo se abordan las fases de exploración y planificación pertenecientes a la metodología de desarrollo XP (Extreme Programming) utilizada para la implementación del sistema que se propone. Además se exponen los artefactos generados durante el transcurso de dichas fases.

3.1 Fase de Exploración.

La metodología de desarrollo XP comienza con la fase de exploración. Durante esta se realiza el proceso de identificación de las historias de usuario, así como la familiarización de los equipos de trabajo con las tecnologías y herramientas seleccionadas para la construcción del proyecto.

3.1.1 Historias de Usuarios.

Las historias de usuario son la forma en que se especifican en XP los requisitos funcionales del sistema. Estas se escriben desde la perspectiva del cliente aunque los desarrolladores pueden brindar también su ayuda en la identificación de las mismas. El contenido de estas debe ser concreto y sencillo. Durante la fase de exploración se identificaron quince historias de usuario, las cuales se detallan a continuación.

Historia de Usuario	
Número: 1	**Usuarios:** Estudiantes, Profesores y Trabajadores
Nombre Historia: Autenticar Usuario.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Bajo.
Puntos Estimados: 0.1	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El usuario se autentica en la aplicación.	
Observaciones:	

Tabla 3.1.1.1 Historia de Usuario Autenticar Usuario.

	Historia de Usuario
Número: 2	**Usuarios:** Administrador del Sistema.
Nombre Historia: Insertar Usuario.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Bajo.
Puntos Estimados: 0.1	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El administrador del sistema inserta usuarios nuevos en la base de datos añadiéndole roles y privilegios a dicho usuario en caso de que los posea.	
Observaciones:	

Tabla 3.1.1.2 Historia de Usuario Insertar Usuario.

	Historia de Usuario
Número: 3	**Usuarios:** Administrador del Sistema.
Nombre Historia: Eliminar Usuario.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Bajo.
Puntos Estimados: 0.1	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El administrador del sistema elimina usuarios de la aplicación quitándole los privilegios a dicho usuario en caso de que los posea.	
Observaciones:	

Tabla 3.1.1.3 Historia de Usuario Eliminar Usuario.

	Historia de Usuario
Número: 4	**Usuarios:** Administrador del Sistema.
Nombre Historia: Insertar Vehículo.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Medio.
Puntos Estimados: 0.5	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El administrador del sistema podrá insertar vehículos los cuales serán ubicados según el área a la que pertenece el vehículo. Permitirá gestionar los datos de un vehículo.	
Observaciones: Los vehículos están distribuidos por áreas.	

Tabla 3.1.1.4 Historia de Usuario Insertar Vehículo.

	Historia de Usuario
Número: 5	**Usuarios:** Estudiantes, Profesores y Trabajadores.
Nombre Historia: Buscar Vehículo.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Medio.
Puntos Estimados: 0.6	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Los usuarios podrán realizar la búsqueda de un vehículo para ver los datos y el estado del mismo.	
Observaciones:	

Tabla 3.1.1.5 Historia de Usuario Buscar Vehículo.

	Historia de Usuario
Número: 6	**Usuarios:** Administrador del Sistema.
Nombre Historia: Modificar Vehículo.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Medio.
Puntos Estimados: 0.5	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El administrador del sistema podrá modificar los datos de los vehículos quedando registrada las actualizaciones en la base de datos.	
Observaciones:	

Tabla 3.1.1.6 Historia de Usuario Modificar Vehículo.

	Historia de Usuario
Número: 7	**Usuarios:** Administrador del Sistema.
Nombre Historia: Eliminar Vehículo.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Medio.
Puntos Estimados: 0.5	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El administrador del sistema podrá eliminar un vehículo así como sus datos de la base de datos. El vehículo se elimina automáticamente de la lista del área a la que pertenece.	
Observaciones:	

Tabla 3.1.1.7 Historia de Usuario Eliminar Vehículo.

Historia de Usuario	
Número: 8	**Usuarios:** Usuarios con privilegios para realizar solicitudes.
Nombre Historia: Realizar Solicitud.	
Prioridad en Negocio: Alta.	**Riesgo en Desarrollo:** Alto.
Puntos Estimados: 1	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El usuario con privilegios para realizar solicitudes debe de plasmar los datos de la misma, así como dejar claro una justificación de por qué la solicitud.	
Observaciones: Las solicitudes deben de realizarse con 72 horas de antelación.	

Tabla 3.1.1.8 Historia de Usuario Realizar Solicitud.

Historia de Usuario	
Número: 9	**Usuarios:** Estudiantes, Profesores y Trabajadores.
Nombre Historia: Buscar Solicitud.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Bajo.
Puntos Estimados: 0.6	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Los usuarios podrán buscar solicitudes y en caso de que tenga privilegios podrá realizar cambios sobre alguna que haya hecho con anterioridad.	
Observaciones:	

Tabla 3.1.1.9 Historia de Usuario Buscar Solicitud.

	Historia de Usuario
Número: 10	**Usuarios:** Usuarios con privilegios para modificar solicitudes.
Nombre Historia: Modificar Solicitud.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Alto.
Puntos Estimados: 0.5	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El usuario podrá modificar los datos de las solicitudes realizadas por él. El usuario debe de realizar los cambios con 24 horas de antelación.	
Observaciones: En caso de que la solicitud se haya aceptado, el usuario debe de comunicar los cambios a los jefes de las áreas a la que el usuario realiza la solicitud.	

Tabla 3.1.1.10 Historia de Usuario Modificar Solicitud.

	Historia de Usuario
Número: 11	**Usuarios:** Jefes de áreas.
Nombre Historia: Aceptar Solicitud.	
Prioridad en Negocio: Alta.	**Riesgo en Desarrollo:** Bajo.
Puntos Estimados: 0.3	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El jefe de cada una de las áreas revisa las solicitudes pendientes y las acepta en caso de que existan todos los medios disponibles para darle respuesta a la solicitud.	
Observaciones:	

Tabla 3.1.1.11 Historia de Usuario Aceptar Solicitud.

	Historia de Usuario
Número: 12	**Usuarios:** Usuarios con privilegios para realizar solicitudes.
Nombre Historia: Cancelar Solicitud.	
Prioridad en Negocio: Media.	**Riesgo en Desarrollo:** Bajo.
Puntos Estimados: 0.5	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: El usuario podrá cancelar las solicitudes realizadas por él con 24 horas de antelación como mínimo.	
Observaciones: En caso de que la solicitud se haya aceptado el usuario debe de comunicar al jefe de área la cancelación de la solicitud realizada.	

Tabla 3.1.1.12 Historia de Usuario Cancelar Solicitud.

	Historia de Usuario
Número: 13	**Usuarios:** Jefes de áreas.
Nombre Historia: Generar Reporte.	
Prioridad en Negocio: Alta.	**Riesgo en Desarrollo:** Medio.
Puntos Estimados: 1	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Los jefes de áreas podrán realizar reportes en caso de que ocurra una incidencia durante la atención a una solicitud, en caso contrario el reporte se genera automáticamente.	
Observaciones:	

Tabla 3.1.1.13 Historia de Usuario Generar Reporte.

	Historia de Usuario
Número: 14	**Usuarios:** Jefes de áreas.
Nombre Historia: Modificar Reporte.	
Prioridad en Negocio: Medio.	**Riesgo en Desarrollo:** Bajo.
Puntos Estimados: 0.3	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Los jefes de áreas podrán modificar los reportes realizados aunque estos hayan sido generados automáticamente.	
Observaciones:	

Tabla 3.1.1.14 Historia de Usuario Modificar Reporte.

	Historia de Usuario
Número: 15	**Usuarios:** Jefes de áreas.
Nombre Historia: Eliminar Reporte.	
Prioridad en Negocio: Medio.	**Riesgo en Desarrollo:** Bajo.
Puntos Estimados: 0.4	**Iteración asignada: 1**
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Los jefes de áreas podrán eliminar los reportes realizados.	
Observaciones:	

Tabla 3.1.1.15 Historia de Usuario Eliminar Reporte.

3.2 Planificación.

Durante la fase de planificación se realiza una estimación del esfuerzo que costará implementar cada historia de usuario. Este se expresa utilizando como medida el punto. Un punto se considera como una semana ideal de trabajo donde los miembros de los equipos de desarrollo trabajan el tiempo planeado sin

ningún tipo de interrupción. Esta estimación incluye todo el esfuerzo asociado a la implementación de la historia de usuario.

3.2.1 Estimación de Esfuerzo por Historias de Usuario.

Para el desarrollo de la aplicación propuesta en este trabajo se realizó una estimación del esfuerzo para cada una de las historias de usuario identificadas, permitiendo tener una medida real de la velocidad de progreso del proyecto y brindando una guía razonable a la cual ajustarse, llegándose así a los resultados que se muestran en la siguiente tabla.

Historias de usuario	Puntos estimados
Autenticar Usuario	0.1
Insertar Usuario	0.1
Eliminar Usuario	0.1
Insertar Vehículo	0.5
Buscar Vehículo	0.6
Modificar Vehículo	0.5
Eliminar Vehículo	0.5
Realizar Solicitud	1
Buscar Solicitud	0.6
Modificar Solicitud	0.5
Aceptar Solicitud	0.3
Cancelar Solicitud	0.5
Generar Reporte	1
Modificar Reporte	0.3
Eliminar Reporte	0.4

Tabla 3.2.1.1 Estimación de esfuerzo por historia de usuario.

3.3 Plan de Liberaciones.

El plan de liberaciones (entregas) tiene como objetivo definir el número de *releases* 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. Para realizar el plan de entregas, el sistema propuesto se dividió en módulos que contienen las historias de usuario relacionadas lógicamente de acuerdo a su propósito. Cada una de las entregas se correspondió con una iteración de desarrollo.

Módulo	Historias de Usuario
Gestión de Usuario	Autenticar Usuario
	Insertar Usuario
	Eliminar Usuario
Gestión de Vehículo	Insertar Vehículo
	Buscar Vehículo
	Modificar Vehículo
	Eliminar Vehículo
Gestión de Solicitud	Realizar Solicitud
	Buscar Solicitud
	Modificar Solicitud
	Aceptar Solicitud
	Cancelar Solicitud
Gestión de Reporte	Crear Reporte
	Modificar Reporte
	Eliminar Reporte

Tabla 3.3.1 Historia de Usuarios por Módulos.

3.4 Plan de Iteraciones.

Una vez identificadas las historias de usuario del sistema y estimado el esfuerzo dedicado a la realización de cada una de estas se procede a la planificación de la etapa de implementación del proyecto. De acuerdo a lo mencionado anteriormente se decidió realizar dicha planificación en cuatro iteraciones, detalladas a continuación.

3.4.1 Iteración 1.

En esta iteración se realizarán las historias de usuarios elegidas para este primer módulo: Gestión de Usuario, las mismas son consideradas de gran importancia para el sistema.

3.4.2 Iteración 2.

La actual iteración se centra en darle solución a las historias de usuario identificadas en el segundo módulo: Gestión de Vehículo, la realización de las mismas va dando una idea de cómo quedará la aplicación, aunque todavía estará en sus inicios.

3.4.3 Iteración 3.

En esta tercera iteración se implementarán las historias de usuarios correspondientes al modulo Gestión de Solicitud las cuales responden a las funcionalidades mas criticas del sistema.

3.4.4 Iteración 4.

La implementación de las historias de usuarios pertenecientes al módulo Gestión de Reportes en esta iteración proporcionará una idea completa de la aplicación, la cual al finalizar la implementación quedará terminado el sistema.

3.5 Plan de Duración de las Iteraciones.

Como parte del ciclo de vida de un proyecto utilizando la metodología XP se crea el plan de duración de cada una de las iteraciones, según los equipos de desarrollo con que se cuente, en este caso se hace para el único equipo de desarrollo que se tiene. Este plan se encarga de mostrar las historias de usuario que serán implementadas en cada una de las iteraciones, así como la duración estimada de cada una y el orden en que se implementarán.

Iteración	Orden de la Historia de Usuario	Duración de las iteraciones
Iteración 1	1.1 Autenticar Usuario 1.2 Insertar Usuario 1.3 Eliminar Usuario	2.5 Semanas
Iteración 2	2.1 Insertar Vehículo 2.2 Buscar Vehículo 2.3 Modificar Vehículo 2.4 Eliminar Vehículo	4.5 Semanas
Iteración 2	3.1 Realizar Solicitud 3.2 Buscar Solicitud 3.3 Modificar Solicitud 3.4 Aceptar Solicitud 3.5 Cancelar Solicitud	5 Semanas
Iteración 4	4.1 Crear Reporte 4.2 Modificar Reporte 4.3 Eliminar Reporte	3 Semanas

Tabla 3.5.1 Plan de duración de las iteraciones.

3.6 Plan de Entregas.

A continuación se presenta el plan de entregas ideado para la fase de implementación. Como producto del mismo se harán releases del sistema al finalizar cada iteración en la fecha aproximada que se indica en la siguiente tabla.

Iteración	Iteración 1	Iteración 2	Iteración 3	Iteración 4
Entrega	Final 1ra Iteración 4ta semana de febrero.	Final 2da Iteración 4ta semana de marzo.	Final 3ra Iteración 1ra semana de mayo	Final 4ta iteración 3ra semana de mayo

Tabla 3.6.1 Plan de Entregas.

Conclusiones

En el presente capítulo se abordó todo lo referente al desarrollo del sistema basado en la metodología XP, identificando quince historias de usuario a las que se realizaron una detallada descripción del negocio en la fase de exploración, y en la fase de planificación se generaron los artefactos de estimación de esfuerzo que costará implementar cada historia de usuario, el plan de duración de las iteraciones para hacer entrega del producto al final de las mismas en el tiempo requerido y el plan de entrega del sistema basado en las fechas de las iteraciones anteriores.

CAPÍTULO IV IMPLEMENTACIÓN Y PRUEBA

Introducción

La Metodología XP plantea que la implementación de un producto debe realizarse de forma iterativa, obteniendo al culminar cada iteración un producto funcional que debe ser probado y mostrado al cliente para incrementar la visión de los desarrolladores con la opinión de éste. Se detallan las cuatro iteraciones llevadas a cabo durante la etapa de construcción del sistema, exponiéndose las tareas generadas por cada historia de usuario, así como las pruebas de aceptación efectuadas sobre el sistema.

Durante el transcurso de las iteraciones se realiza la implementación de las historias de usuario seleccionadas para cada una de estas. Al inicio de las mismas, se lleva a cabo una revisión del plan de iteraciones y se modifica de ser necesario. Ajustándose a la planificación realizada. Se llevaron a cabo cuatro iteraciones de desarrollo sobre el sistema, obteniéndose al finalizar, un producto listo para su puesta en función.

4.1 Implementación

En esta fase se realiza una descripción detallada de las tareas a realizar en cada una de las historias de usuarios por cada iteración.

4.1.1 Historias de Usuario Implementadas en la Primera Iteración.

Durante esta iteración se abordaron las historias de usuario seleccionadas para el primer módulo, se construyó la base de la arquitectura del sistema con el fin de obtener un producto con las funcionalidades primarias para ser mostrado al cliente y obtener una rápida y amplia retroalimentación del mismo.

Historias de Usuarios	Estimado	Real
Autenticar Usuario	0.1	0.2
Insertar Usuario	0.1	0.2
Eliminar Usuario	0.1	0.2

Tabla 4.1.1 Historias de usuario implementadas en la primera iteración.

4.1.1.1 Tareas de la Historia de Usuario Autenticar Usuario.

	Tarea
Número Tarea: 1	**Número Historia:** 1
Nombre Tarea: Configuración de la interfaz para la autenticación de los usuarios.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.085
Fecha Inicio: 31 de Enero del 2010.	**Fecha Fin:** 2 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se realiza la configuración de la interfaz de autenticación para los usuarios que harán uso de la aplicación.	

Tabla 4.1.1.1 Tarea #1 de la historia de usuario Autenticar Usuario.

	Tarea
Número Tarea: 2	**Número Historia:** 1
Nombre Tarea: Verificar si el usuario es válido.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 3 de Febrero del 2010.	**Fecha Fin:** 5 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se comprueban los datos de los usuarios que se autentican en la aplicación verificando si los mismos son válidos o no.	

Tabla 4.1.1.2 Tarea #2 de la historia de usuario Autenticar Usuario.

	Tarea
Número Tarea: 3	**Número Historia:** 1
Nombre Tarea: Permisos y roles correspondientes.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 6 de Febrero del 2010.	**Fecha Fin:** 8 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Si el usuario es válido se le asignan los permisos y roles correspondientes para la navegación dentro de la aplicación.	

Tabla 4.1.1.3 Tarea #3 de la historia de usuario Autenticar Usuario.

4.1.1.2 Tareas de la Historia de Usuario Insertar Usuario.

	Tarea
Número Tarea: 1	**Número Historia:** 2
Nombre Tarea: Configuración de la interfaz para insertar un usuario.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 9 de Febrero del 2010.	**Fecha Fin:** 10 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se configura la interfaz para insertar un usuario en la aplicación.	

Tabla 4.1.1.4 Tarea #1 de la historia de usuario Insertar Usuario.

	Tarea
Número Tarea: 2	**Número Historia:** 2
Nombre Tarea: Verificar datos.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 11 de Febrero del 2010.	**Fecha Fin:** 12 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se verifican los datos de los usuarios que se desean insertar.	

Tabla 4.1.1.5 Tarea #2 de la historia de usuario Insertar Usuario.

	Tarea
Número Tarea: 3	**Número Historia:** 2
Nombre Tarea: Insertar datos de usuario.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 13 de Febrero del 2010.	**Fecha Fin:** 14 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Una vez verificados los datos del usuario se insertan y se actualiza la base de datos.	

Tabla 4.1.1.6 Tarea #3 de la historia de usuario Insertar Usuario.

	Tarea
Número Tarea: 4	**Número Historia:** 2
Nombre Tarea: Actualizar datos.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 15 de Febrero del 2010.	**Fecha Fin:** 16 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se selecciona el usuario de una lista, se le realizan los cambios de los datos y se actualiza la base de datos.	

Tabla 4.1.1.7 Tarea #4 de la historia de usuario Insertar Usuario.

4.1.1.3 Tareas de la Historia de Usuario Eliminar Usuario.

	Tarea
Número Tarea: 1	**Número Historia:** 3
Nombre Tarea: Configuración de la interfaz para eliminar un usuario.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 17 de Febrero del 2010.	**Fecha Fin:** 18 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se configura la interfaz para eliminar un usuario así como los privilegios que el mismo posea.	

Tabla 4.1.1.8 Tarea #1 de la historia de usuario Eliminar Usuario.

	Tarea
Número Tarea: 2	**Número Historia:** 3
Nombre Tarea: Mostrar lista de usuarios	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.1
Fecha Inicio: 19 de Febrero del 2010	**Fecha Fin:** 20 de Febrero 2010
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se muestra una lista de usuarios para seleccionar el que se desea eliminar.	

Tabla 4.1.1.9 Tarea #2 de la historia de usuario Eliminar Usuario.

	Tarea
Número Tarea: 3	**Número Historia:** 3
Nombre Tarea: Eliminar privilegios.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 21 de Febrero del 2010.	**Fecha Fin:** 22 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Una vez eliminado el usuario se eliminan los privilegios que este tenía de la base de datos.	

Tabla 4.1.1.10 Tarea #3 de la historia de usuario Eliminar Usuario.

4.1.2 Historias de Usuario Implementadas en la Segunda Iteración.

En la presente iteración se implementan las funcionalidades del segundo módulo de la aplicación, al finalizar el desarrollo de cada una de ellas en esta fase se tendrá una idea más amplia de cómo quedará el sistema.

Historias de Usuarios	Estimado	Real
Insertar Vehículo	0.5	0.5
Buscar Vehículo	0.6	0.7
Modificar Vehículo	0.5	0.6
Eliminar Vehículo	0.5	0.5

Tabla 4.1.2 Historias de usuario implementadas en la segunda iteración.

4.1.2.1 Tareas de la Historia de Usuario Insertar Vehículo.

	Tarea
Número Tarea: 1	**Número Historia:** 4
Nombre Tarea: Configuración de la interfaz para insertar vehículo.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 23 de Febrero del 2010.	**Fecha Fin:** 24 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se crea la interfaz para insertar los datos del vehículo.	

Tabla 4.1.2.1 Tarea #1 de la historia de usuario Insertar Vehículo.

	Tarea
Número Tarea: 2	**Número Historia:** 4
Nombre Tarea: Verificar existencia del vehículo.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 25 de Febrero del 2010.	**Fecha Fin:** 26 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se verifica en la base de datos si el vehículo ya existe o no.	

Tabla 4.1.2.2 Tarea #2 de la historia de usuario Insertar Vehículo.

	Tarea
Número Tarea: 2	**Número Historia:** 4
Nombre Tarea: Insertar datos del vehículo en la base de datos.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 27 de Febrero del 2010.	**Fecha Fin:** 28 de Febrero 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se insertan todos los datos de un vehículo en la base de datos y se actualiza la misma.	

Tabla 4.1.2.3 Tarea #3 de la historia de usuario Insertar Vehículo.

4.1.2.2 Tareas de la Historia de Usuario Buscar Vehículo.

	Tarea
Número Tarea: 1	**Número Historia:** 5
Nombre Tarea: Crear interfaz para buscar un vehículo	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.1
Fecha Inicio: 1 de Marzo del 2010	**Fecha Fin:** 2 de Marzo 2010
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se configura la interfaz para buscar los datos de un vehículo.	

Tabla 4.1.2.4 Tarea #1 de la historia de usuario Buscar Vehículo.

	Tarea
Número Tarea: 2	**Número Historia:** 5
Nombre Tarea: Insertar término de búsqueda.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 3 de Marzo del 2010.	**Fecha Fin:** 5 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se insertan los datos del vehículo que se desea buscar.	

Tabla 4.1.2.5 Tarea #2 de la historia de usuario Buscar Vehículo.

	Tarea
Número Tarea: 3	**Número Historia:** 5
Nombre Tarea: Mostrar datos del vehículo.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 6 de Marzo del 2010.	**Fecha Fin:** 7 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se muestran todos los datos del vehículo al cual se le realiza la búsqueda.	

Tabla 4.1.2.6 Tarea #3 de la historia de usuario Buscar Vehículo.

4.1.2.3 Tareas de la Historia de Usuario Modificar Vehículo.

	Tarea
Número Tarea: 1	**Número Historia:** 6
Nombre Tarea: Configurar interfaz para modificar un vehículo.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 8 de Marzo del 2010.	**Fecha Fin:** 9 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se configura la interfaz para modificar los datos de un vehículo determinado.	

Tabla 4.1.2.7 Tarea #1 de la historia de usuario Modificar Vehículo.

	Tarea
Número Tarea: 2	**Número Historia:** 6
Nombre Tarea: Lista de vehículos.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 10 de Marzo del 2010.	**Fecha Fin:** 11 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se muestra una lista de vehículos para seleccionar el que se desea modificar.	

Tabla 4.1.2.8 Tarea #2 de la historia de usuario Modificar Vehículo.

	Tarea
Número Tarea: 3	**Número Historia:** 6
Nombre Tarea: Insertar datos del vehículo.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 12 de Marzo del 2010.	**Fecha Fin:** 13 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se insertan los nuevos datos del vehículo seleccionado y se actualiza la base de datos.	

Tabla 4.1.2.9 Tarea #3 de la historia de usuario Modificar Vehículo.

4.1.2.4 Tareas de la Historia de Usuario Eliminar Vehículo.

	Tarea
Número Tarea: 1	**Número Historia:** 7
Nombre Tarea: Configurar interfaz para eliminar vehículo.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 14 de Marzo del 2010.	**Fecha Fin:** 15 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se configura la interfaz para eliminar un vehículo.	

Tabla 4.1.2.10 Tarea #1 de la historia de usuario Eliminar Vehículo.

	Tarea
Número Tarea: 2	**Número Historia:** 7
Nombre Tarea: Buscar vehículo.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 16 de Marzo del 2010.	**Fecha Fin:** 17 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se realiza una búsqueda del vehículo que se desea eliminar.	

Tabla 4.1.2.11 Tarea #2 de la historia de usuario Eliminar Vehículo.

	Tarea
Número Tarea: 2	**Número Historia:** 7
Nombre Tarea: Eliminar datos del vehículo	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.1
Fecha Inicio: 18 de Marzo del 2010	**Fecha Fin:** 19 de Marzo 2010
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se eliminan los datos del vehículo encontrado y se actualiza la base de datos.	

Tabla 4.1.2.12 Tarea #3 de la historia de usuario Eliminar Vehículo.

4.1.3 Historias de Usuario Implementadas en la Tercera Iteración.

Durante el transcurso de esta iteración se implementaron las historias de usuario referentes al tercer módulo. Al finalizar la misma, se consta de un producto casi terminado para su puesta en funcionamiento con la mayoría de sus funcionalidades más críticas ya implementadas.

Historias de Usuarios	Estimado	Real
Realizar Solicitud	1	1
Buscar Solicitud	0.6	0.7
Modificar Solicitud	0.5	0.6
Aceptar Solicitud	0.3	0.5
Cancelar Solicitud	0.5	0.5

Tabla 4.3.1 Historias de usuario implementadas en la tercera iteración.

4.1.3.1 Tareas de la Historia de Usuario Realizar Solicitud.

	Tarea
Número Tarea: 1	**Número Historia:** 8
Nombre Tarea: Configurar interfaz para realizar solicitud.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 21 de Marzo del 2010.	**Fecha Fin:** 22 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se configura la interfaz para realizar la solicitud de transporte.	

Tabla 4.1.3.1 Tarea #1 de la historia de usuario Realizar Solicitud.

	Tarea
Número Tarea: 2	**Número Historia:** 8
Nombre Tarea: Verificar solicitud.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 23 de Marzo del 2010.	**Fecha Fin:** 24 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se comprueba que la solicitud no se haya realizado aún.	

Tabla 4.1.3.2 Tarea #2 de la historia de usuario Realizar Solicitud.

	Tarea
Número Tarea: 3	**Número Historia:** 8
Nombre Tarea: Insertar datos de la solicitud.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 25 de Marzo del 2010.	**Fecha Fin:** 26 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se insertan todos los datos de la solicitud que se desea realizar en la base de datos y se actualiza la misma.	

Tabla 4.1.3.3 Tarea #3 de la historia de usuario Realizar Solicitud.

4.1.3.2 Tareas de la Historia de Usuario Buscar Solicitud.

	Tarea
Número Tarea: 1	**Número Historia:** 9
Nombre Tarea: Crear interfaz para buscar solicitud.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 27 de Marzo del 2010.	**Fecha Fin:** 28 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se desarrolla una interfaz para realizar la búsqueda de una solicitud deseada.	

Tabla 4.1.3.4 Tarea #1 de la historia de usuario Buscar Solicitud.

	Tarea
Número Tarea: 2	**Número Historia:** 9
Nombre Tarea: Insertar término de búsqueda.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 29 de Marzo del 2010.	**Fecha Fin:** 30 de Marzo 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se insertan los datos de la solicitud que se desea buscar.	

Tabla 4.1.3.5 Tarea #2 de la historia de usuario Buscar Solicitud.

	Tarea
Número Tarea: 3	**Número Historia:** 9
Nombre Tarea: Mostrar datos de la solicitud.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 31 de Marzo del 2010.	**Fecha Fin:** 1 de Abril 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Una vez insertado el/los termino(s) de búsqueda se muestran todos los datos de la solicitud que se desea buscar.	

Tabla 4.1.3.6 Tarea #3 de la historia de usuario Buscar Solicitud.

4.1.3.3 Tareas de la Historia de Usuario Modificar Solicitud.

	Tarea
Número Tarea: 1	**Número Historia:** 10
Nombre Tarea: Configurar interfaz para modificar solicitud.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 2 de Abril del 2010.	**Fecha Fin:** 3 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se configura la interfaz para modificar una solicitud.	

Tabla 4.1.3.7 Tarea #1 de la historia de usuario Modificar Solicitud.

	Tarea
Número Tarea: 2	**Número Historia:** 10
Nombre Tarea: Mostrar lista de solicitudes.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 4 de Abril del 2010.	**Fecha Fin:** 5 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se muestra una lista de las solicitudes realizadas para seleccionar la que se desea modificar.	

Tabla 4.1.3.8 Tarea #2 de la historia de usuario Modificar Solicitud.

	Tarea
Número Tarea: 3	**Número Historia:** 10
Nombre Tarea: Insertar nuevos datos.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 6 de Abril del 2010.	**Fecha Fin:** 7 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Una vez seleccionada la solicitud que se desea modificar se insertan los nuevos datos y se actualiza la base de datos.	

Tabla 4.1.3.9 Tarea #3 de la historia de usuario Modificar Solicitud.

4.1.3.4 Tareas de la Historia de Usuario Aceptar Solicitud.

	Tarea
Número Tarea: 1	**Número Historia:** 11
Nombre Tarea: Listar solicitudes realizadas.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 8 de Abril del 2010.	**Fecha Fin:** 9 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se muestra una lista de solicitudes realizadas para seleccionar la que se desea aceptar.	

Tabla 4.1.3.10 Tarea #1 de la historia de usuario Aceptar Solicitud.

	Tarea
Número Tarea: 2	**Número Historia:** 11
Nombre Tarea: Enviar notificación.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 10 de Abril del 2010.	**Fecha Fin:** 11 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Una vez aceptada la solicitud seleccionada se envía una notificación al solicitante.	

Tabla 4.1.3.11 Tarea #2 de la historia de usuario Aceptar Solicitud.

4.1.3.5 Tareas de la Historia de Usuario Cancelar Solicitud.

	Tarea
Número Tarea: 1	**Número Historia:** 12
Nombre Tarea: Listar solicitudes realizadas.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 12 de Abril del 2010.	**Fecha Fin:** 13 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se muestra una lista de solicitudes realizadas para seleccionar la que se desea cancelar.	

Tabla 4.1.3.12 Tarea #1 de la historia de usuario Cancelar Solicitud.

	Tarea
Número Tarea: 1	**Número Historia:** 12
Nombre Tarea: Cancelar solicitud.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 14 de Abril del 2010.	**Fecha Fin:** 15 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se cancela la solicitud seleccionada.	

Tabla 4.1.3.13 Tarea #2 de la historia de usuario Cancelar Solicitud.

	Tarea
Número Tarea: 3	**Número Historia:** 12
Nombre Tarea: Eliminar datos de la solicitud.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 16 de Abril del 2010.	**Fecha Fin:** 17 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se eliminan los datos de la solicitud cancelada y se actualiza la base de datos.	

Tabla 4.1.3.14 Tarea #3 de la historia de usuario Cancelar Solicitud.

4.1.4 Historias de Usuario Implementadas en la Cuarta Iteración.

Durante la presente iteración se implementaron las historias de usuario restantes pertenecientes al cuarto y último módulo, dichas historias involucran las funcionalidades concernientes a los reportes que brinda la aplicación. Al culminar esta última fase, se consta de un producto listo para su puesta en funcionamiento.

Historias de Usuarios	Estimado	Real
Crear Reporte	1	1
Modificar Reporte	0.3	0.3
Eliminar Reporte	0.4	0.4

Tabla 4.1.4 Historias de usuario implementadas en la tercera iteración.

4.1.4.1 Tareas de la Historia de Usuario Crear Reporte.

	Tarea
Número Tarea: 1	**Número Historia:** 13
Nombre Tarea: Configurar interfaz para crear reporte.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 18 de Abril del 2010.	**Fecha Fin:** 19 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se configura la interfaz que permitirá crear reportes en la aplicación.	

Tabla 4.1.4.1 Tarea #1 de la historia de usuario Crear Reporte.

	Tarea
Número Tarea: 2	**Número Historia:** 13
Nombre Tarea: Entrar datos del reporte en formulario.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 20 de Abril del 2010.	**Fecha Fin:** 21 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se introducen en un formulario los datos referentes a la creación de un reporte.	

Tabla 4.1.4.2 Tarea #2 de la historia de usuario Crear Reporte.

	Tarea
Número Tarea: 3	**Número Historia:** 13
Nombre Tarea: Entrar datos del reporte en base de datos.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 22 de Abril del 2010.	**Fecha Fin:** 23 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se introducen los datos referentes a la creación de un reporte en la base de datos y se actualiza la misma.	

Tabla 4.1.4.3 Tarea #3 de la historia de usuario Crear Reporte.

4.1.4.2 Tareas de la Historia de Usuario Modificar Reporte.

	Tarea
Número Tarea: 1	**Número Historia:** 14
Nombre Tarea: Configurar interfaz para modificar reporte.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 24 de Abril del 2010.	**Fecha Fin:** 25 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se crea la interfaz para realizar las modificaciones a un reporte.	

Tabla 4.1.4.4 Tarea #1 de la historia de usuario Modificar Reporte.

	Tarea
Número Tarea: 2	**Número Historia:** 14
Nombre Tarea: Lista de reportes.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 26 de Abril del 2010.	**Fecha Fin:** 27 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se muestra una lista de reportes para seleccionar el que se desea modificar.	

Tabla 4.1.4.5 Tarea #2 de la historia de usuario Modificar Reporte.

	Tarea
Número Tarea: 3	**Número Historia:** 14
Nombre Tarea: Modificar reporte.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 28 de Abril del 2010.	**Fecha Fin:** 29 de Abril del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se insertan los nuevos datos del reporte seleccionado, se guardan en la base de datos y se actualiza la misma.	

Tabla 4.1.4.6 Tarea #3 de la historia de usuario Modificar Reporte.

4.1.4.3 Tareas de la Historia de Usuario Eliminar Reporte.

	Tarea
Número Tarea: 1	**Número Historia:** 15
Nombre Tarea: Crear interfaz para eliminar reporte.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 30 de Abril del 2010.	**Fecha Fin:** 2 de Mayo del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se crea la interfaz para realizar la eliminación de un reporte deseado.	

Tabla 4.1.4.7 Tarea #1 de la historia de usuario Eliminar Reporte.

	Tarea
Número Tarea: 2	**Número Historia:** 15
Nombre Tarea: Listar reportes.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 3 de Mayo del 2010.	**Fecha Fin:** 4 de Mayo del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se muestra una lista de reportes para seleccionar el que se desea eliminar.	

Tabla 4.1.4.8 Tarea #2 de la historia de usuario Eliminar Reporte.

	Tarea
Número Tarea: 3	**Número Historia:** 15
Nombre Tarea: Eliminar datos del reporte.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 0.1
Fecha Inicio: 5 de Mayo del 2010.	**Fecha Fin:** 6 de Mayo del 2010.
Programador Responsable: José A. Roque Pérez – Camilo Trujillo Pacheco	
Descripción: Se eliminan todos los datos del reporte seleccionado y se actualiza la base de datos.	

Tabla 4.1.4.9 Tarea #3 de la historia de usuario Eliminar Reporte.

4.2 Prueba

Uno de los pilares fundamentales de la metodología XP es el proceso de pruebas, el cual anima a los desarrolladores a probar constantemente tanto como sea posible. Mediante esta filosofía se reduce el número de errores no detectados así como el tiempo entre la introducción de estos en el sistema y su detección. Todo esto contribuye a aumentar la calidad de los productos desarrollados, así como la seguridad de los programadores a la hora de proveer nuevos 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.

4.2.1 Pruebas de Aceptación.

Las pruebas de aceptación son pruebas que se crean a partir de las historias de usuario. Durante las iteraciones las historias de usuarios seleccionadas serán traducidas a pruebas de aceptación. En ellas se especifican, desde la perspectiva del cliente, los escenarios para probar que una historia de usuario ha sido implementada correctamente. Una historia de usuario puede tener todas las pruebas de aceptación que necesite para asegurar su correcto funcionamiento. El objetivo final de éstas es garantizar que los requerimientos han sido cumplidos y que el sistema es aceptable. Una historia de usuario no se considera completa hasta que no ha pasado por sus pruebas de aceptación.

4.2.1.1 Pruebas de aceptación para la iteración 1.

Caso de Prueba de Aceptación	
Código: HU1_P1	**Historia de usuario:** Autenticar usuario.
Nombre: Autenticar usuario en el sistema.	
Descripción: Probar que el usuario se autentique correctamente en la aplicación.	
Condiciones de ejecución: El sistema debe de ser ejecutado normalmente, el usuario y la contraseña deben de ser verificados en la base de datos de la UCI.	
Entrada/ Pasos de ejecución: El usuario se autentica introduciendo datos válidos.	

Resultado Esperado: El usuario es reconocido en el sistema y logueado correctamente.
Evaluación de la Prueba: Prueba satisfactoria.

Tabla 4.2.1.1 Prueba 1 al módulo Gestionar Usuario.

Caso de Prueba de Aceptación	
Código: HU1_P2	**Historia de usuario:** Autenticar usuario.
Nombre: Autenticar usuario en el sistema.	
Descripción: Probar que un usuario no se autentique si introduce datos no válidos en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado normalmente, el usuario y la contraseña deben de ser verificados en la base de datos de la UCI.	
Entrada/ Pasos de ejecución: El usuario intenta autenticarse introduciendo datos no válidos, datos incompletos o dejando algún campo vacío.	
Resultado Esperado: El usuario no es reconocido y no puede acceder correctamente en el sistema.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.2 Prueba 2 al módulo Gestionar Usuario.

Caso de Prueba de Aceptación	
Código: HU2_P1	**Historia de usuario:** Insertar Usuario.
Nombre: Insertar usuario en el sistema.	
Descripción: Probar que los datos de un usuario son añadidos correctamente en la base de datos del sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado con privilegios de administración, los datos del usuario deben de ser guardados en la base de datos.	
Entrada/ Pasos de ejecución: Se intenta insertar los datos de un usuario en la aplicación.	
Resultado Esperado: El sistema muestra un mensaje de error si los datos del usuario son incorrectos, están incompletos o si existe algún campo vacío. En caso contrario los datos del usuario son insertados correctamente en la base de datos del sistema.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.3 Prueba 3 al módulo Gestionar Usuario.

	Caso de Prueba de Aceptación
Código: HU2_P2	**Historia de usuario:** Insertar Usuario.
Nombre: Añadir privilegios a un usuario en el sistema.	
Descripción: Probar que se le añaden correctamente los privilegios a un usuario.	
Condiciones de ejecución: El sistema debe de ser ejecutado con privilegios de administración, los privilegios del usuario deben de ser añadidos en la base de datos.	
Entrada/ Pasos de ejecución: Se intenta adicionar privilegios a un usuario de la aplicación.	
Resultado Esperado: Los privilegios del usuario son adicionados correctamente en la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.4 Prueba 4 al módulo Gestionar Usuario.

	Caso de Prueba de Aceptación
Código: HU3_P1	**Historia de usuario:** Eliminar Usuario.
Nombre: Eliminar usuario del sistema.	
Descripción: Probar que se eliminen correctamente de la aplicación los datos de un usuario.	
Condiciones de ejecución: El sistema debe de ser ejecutado con privilegios de administración, los datos del usuario deben de ser eliminados de la base de datos.	
Entrada/ Pasos de ejecución: Se intenta eliminar los datos de un usuario de la aplicación.	
Resultado Esperado: Los datos del usuario son eliminados correctamente de la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.5 Prueba 5 al módulo Gestionar Usuario.

Caso de Prueba de Aceptación	
Código: HU3_P2	**Historia de usuario:** Eliminar Usuario.
Nombre: Eliminar privilegios de un usuario del sistema.	
Descripción: Probar que se eliminen correctamente los privilegios de un usuario del sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado con privilegios de administración, los privilegios del usuario deben de ser eliminados de la base de datos.	
Entrada/ Pasos de ejecución: Se intenta eliminar los privilegios de un usuario del sistema.	
Resultado Esperado: Los privilegios del usuario son eliminados correctamente de la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.6 Prueba 6 al módulo Gestionar Usuario.

4.2.1.2 Pruebas de aceptación para la iteración 2.

Caso de Prueba de Aceptación	
Código: HU4_P1	**Historia de usuario:** Insertar Vehículo.
Nombre: Insertar vehículo en el sistema.	
Descripción: Probar que se insertan correctamente los datos de un vehículo.	
Condiciones de ejecución: El sistema debe de ser ejecutado con privilegios de administración, los datos de un vehículo deben de ser adicionados de forma correcta a la base de datos.	
Entrada/ Pasos de ejecución: Se intenta adicionar los datos de un vehículo en el sistema.	
Resultado Esperado: El sistema muestra un mensaje de error si los datos del vehículo son incorrectos, están incompletos o si existe algún campo vacío. En caso contrario los datos de dicho vehículo son añadidos correctamente a la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.7 Prueba 1 al módulo Gestionar Vehículo.

Caso de Prueba de Aceptación	
Código: HU5_P1	**Historia de usuario:** Buscar Vehículo.
Nombre: Buscar vehículo en el sistema.	
Descripción: Probar que se pueden buscar vehículos en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado normalmente.	
Entrada/ Pasos de ejecución: Se intenta realizar una búsqueda en el sistema introduciendo datos correctos.	
Resultado Esperado: Se muestran todos los datos del vehículo al cual se le hace la búsqueda.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.8 Prueba 2 al módulo Gestionar Vehículo.

Caso de Prueba de Aceptación	
Código: HU6_P1	**Historia de usuario:** Modificar Vehículo.
Nombre: Modificar vehículo en el sistema.	
Descripción: Probar que se pueden modificar vehículos en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado con privilegios de administración, los datos de un vehículo deben de ser modificados correctamente.	
Entrada/ Pasos de ejecución: Se intenta modificar en el sistema los datos de un vehículo introduciendo nuevos datos.	
Resultado Esperado: Se cambian correctamente los datos del vehículo y se guardan los cambios en la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.9 Prueba 3 al módulo Gestionar Vehículo.

Caso de Prueba de Aceptación	
Código: HU7_P1	**Historia de usuario:** Eliminar Vehículo.
Nombre: Eliminar vehículo en el sistema.	

Descripción: Probar que se pueden eliminar vehículos en el sistema.
Condiciones de ejecución: El sistema debe de ser ejecutado con privilegios de administración, los datos de un vehículo deben de ser eliminados correctamente de la base de datos.
Entrada/ Pasos de ejecución: Se intenta eliminar en el sistema los datos de un vehículo.
Resultado Esperado: Se eliminan todos los datos de dicho vehículos de la base de datos.
Evaluación de la Prueba: Prueba satisfactoria.

Tabla 4.2.1.10 Prueba 4 al módulo Gestionar Vehículo.

4.2.1.3 Pruebas de aceptación para la iteración 3.

Caso de Prueba de Aceptación	
Código: HU8_P1	**Historia de usuario:** Realizar Solicitud.
Nombre: Eliminar vehículo en el sistema.	
Descripción: Probar que se pueden eliminar vehículos en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado por un usuario con privilegios para realizar una solicitud.	
Entrada/ Pasos de ejecución: Se intenta realizar una solicitud en el sistema introduciendo los datos correctamente.	
Resultado Esperado: Se introducen todos los datos de dicha solicitud de la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.11 Prueba 1 al módulo Gestionar Solicitudes.

Caso de Prueba de Aceptación	
Código: HU9_P1	**Historia de usuario:** Buscar Solicitud.
Nombre: Buscar solicitud en el sistema.	
Descripción: Probar que se pueden buscar solicitudes en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado normalmente.	
Entrada/ Pasos de ejecución: Se intenta realizar una búsqueda en el sistema introduciendo datos correctos.	

Resultado Esperado: Se muestran todos los datos de la solicitud a la cual se le realiza la búsqueda.
Evaluación de la Prueba: Prueba satisfactoria.

Tabla 4.2.1.12 Prueba 2 al módulo Gestionar Solicitudes.

Caso de Prueba de Aceptación	
Código: HU10_P1	**Historia de usuario:** Modificar Solicitud.
Nombre: Configurar interfaz para modificar vehículo en el sistema.	
Descripción: Probar que se pueden modificar vehículos en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado por usuarios con privilegios para modificar una solicitud.	
Entrada/ Pasos de ejecución: Se intenta realizar una modificación de una solicitud en el sistema introduciendo datos correctos.	
Resultado Esperado: Se introducen los nuevos datos y se guardan los cambios en la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.13 Prueba 3 al módulo Gestionar Solicitudes.

Caso de Prueba de Aceptación	
Código: HU11_P1	**Historia de usuario:** Aceptar Solicitud.
Nombre: Aceptar solicitudes en el sistema.	
Descripción: Probar que se pueden aceptar solicitudes realizadas en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado por usuarios con privilegios para realizar la aceptación de una solicitud.	
Entrada/ Pasos de ejecución: Se muestra una lista de solicitudes para seleccionar cual será aceptada.	
Resultado Esperado: Una vez aceptada la solicitud seleccionada se le envía un mensaje al solicitante informándole del estado de su solicitud.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.14 Prueba 4 al módulo Gestionar Solicitudes.

Caso de Prueba de Aceptación	
Código: HU12_P1	**Historia de usuario:** Cancelar Solicitud.
Nombre: Configurar interfaz para cancelar una solicitud.	
Descripción: Se configura la interfaz que permite cancelar una solicitud realizada en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado por usuarios con privilegios para realizar la cancelación de una solicitud.	
Entrada/ Pasos de ejecución: Se muestra una lista de solicitudes para seleccionar cual será cancelada.	
Resultado Esperado: Una vez cancelada la solicitud seleccionada se eliminan los datos de la misma de la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.15 Prueba 5 al módulo Gestionar Solicitudes.

4.2.1.4 Pruebas de aceptación para la iteración 4.

Caso de Prueba de Aceptación	
Código: HU13_P1	**Historia de usuario:** Crear Reporte.
Nombre: Configurar interfaz para crear un reporte.	
Descripción: Se configura la interfaz que permite crear un reporte en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado por usuarios con privilegios para crear un reporte.	
Entrada/ Pasos de ejecución: Se introducen datos correctos de un reporte.	
Resultado Esperado: Una vez introducidos los datos se crea el reporte en el sistema y se guardan los datos de dicho reporte en la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.16 Prueba 1 al módulo Gestionar Reporte.

Caso de Prueba de Aceptación	
Código: HU14_P1	**Historia de usuario:** Modificar Reporte.
Nombre: Configurar interfaz para modificar un reporte.	
Descripción: Se configura la interfaz que permite modificar un reporte en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado por usuarios con privilegios para modificar un reporte.	
Entrada/ Pasos de ejecución: Se busca el reporte al cual se le desea realzar cambios y se introducen los nuevos datos correctamente.	
Resultado Esperado: Una vez introducidos los nuevos datos en el sistema se modifica el reporte y se guardan los datos en la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.17 Prueba 2 al módulo Gestionar Reporte.

Caso de Prueba de Aceptación	
Código: HU15_P1	**Historia de usuario:** Eliminar Reporte.
Nombre: Configurar interfaz para eliminar un reporte.	
Descripción: Se configura la interfaz para eliminar un reporte en el sistema.	
Condiciones de ejecución: El sistema debe de ser ejecutado normalmente y por usuarios con privilegios para eliminar un reporte.	
Entrada/ Pasos de ejecución: Se busca el reporte al que se desea eliminar del sistema.	
Resultado Esperado: Una vez eliminado el reporte buscado se borran todos los datos correspondientes de la base de datos.	
Evaluación de la Prueba: Prueba satisfactoria.	

Tabla 4.2.1.18 Prueba 3 al módulo Gestionar Reporte.

4.3 Diagramas de Clases.

Para el diseño de las aplicaciones, la metodología XP no requiere la representación del sistema mediante diagramas de clases utilizando notación UML, en su lugar se usan otras técnicas como las tarjetas CRC.

No obstante el uso de estos puede aplicarse siempre y cuando mejore la comunicación, no sea un peso su mantenimiento, no sean extensos y se enfoquen en la información importante. Aunque el sistema no fue desarrollado orientado a objeto por lo que no se utilizaron clases en su desarrollo, ya que Drupal no es orientado a objetos, si se generaron diagramas de la base de datos que se muestran en el anexo II.

Conclusiones

En el presente capítulo se construyó el modelo necesario para llevar a cabo el proceso de implementación del sistema. También se diseñaron las clases persistentes que permiten hacer el diagrama de entidad-relación del sistema de gestión de bases de datos que se utilizará. Se hace alusión a las etapas de implementación y pruebas del software en desarrollo donde se exponen las tareas correspondientes para dar solución a las historias de usuarios así como las pruebas de aceptación que propician al cliente conformidad y seguridad ante el sistema. Con el fin de este capítulo se da por terminada la propuesta que trae el presente trabajo de diploma.

CONCLUSIONES GENERALES

En la actualidad se ha hecho muy popular el uso de herramientas informáticas para resolver problemáticas relacionadas con la gestión de la información por lo que encontramos en la informatización una de las soluciones viables y de alta calidad.

Con la realización del presente trabajo de diploma se ha logrado:

- Tener conocimiento sobre la situación actual y las tendencias de los sistemas de gestión de solicitudes de transporte en Cuba y el mundo, permitiendo así encontrar una solución al problema planteado.
- Demostrar la necesidad de crear un sistema que fuese capaz de gestionar las solicitudes de transporte realizadas a la base de microbús de la UCI, así como centralizar la información relacionada a dichas solicitudes.
- Implementar un sistema que resuelve las limitantes presentadas anteriormente y se pone en manos del Departamento de Transporte una herramienta que permite gestionar las solicitudes de transporte en la base de microbús de la UCI.
- Tener conocimiento sobre las herramientas informáticas puestas en práctica para desarrollar con la calidad requerida la solución propuesta.
- Realizar una serie de pruebas para comprobar la funcionalidad de la solución propuesta e implementada en este trabajo.

De esta forma se ha cumplido con los objetivos planteados en la presentación de este trabajo de diploma, pues se ha logrado de forma eficiente la implementación de un sistema que perfecciona la administración de la información referida a la gestión de solicitudes de transporte en la Universidad de las Ciencias Informáticas.

RECOMENDACIONES

Al mismo tiempo que se han cumplido los objetivos involucrados en el desarrollo del presente trabajo se realizan las siguientes recomendaciones:

- Realizar una investigación para determinar nuevas funcionalidades que se puedan agregar a la aplicación.
- Perfeccionar las funcionalidades ya implementadas con el fin de brindar un mejor servicio.
- Realizar mejoras a la interfaz visual del sistema.
- Extender el uso de la aplicación a otras entidades del país con el objetivo de reducir costos en cuanto a la prestación de este importante servicio así como mejorar la calidad del mismo en cada una de ellas.
- Se recomienda este trabajo para ser usado como material de estudio en la realización de alguna aplicación similar.

REFERENCIAS BIBLIOGRÁFICAS

1. **Miguel, Angel Alvarez.** desarrolloweb.com. [En línea] 11 de Noviembre de 2008. [Citado el: 5 de Marzo de 2010.] http://www.desarrolloweb.com/articulos/que-es-un-cms.html.

2. cms-hispano.org. [En línea] [Citado el: 5 de Marzo de 2010.] http://cms-hispano.org/index.php?s=content&p=necesidad.

3. **César, Martín.** alzado.org. [En línea] 27 de Julio de 2009. [Citado el: 5 de Marzo de 2010.] http://www.alzado.org/articulo.php?id_art=798.

4. editum.org. [En línea] 25 de Diciembre de 2007. [Citado el: 6 de Marzo de 2010.] http://www.editum.org/Que-Es-Un-Servidor-De-Aplicaciones-p-473.html.

5. ciberaula.com. [En línea] [Citado el: 6 de Marzo de 2010.] http://linux.ciberaula.com/articulo/linux_apache_intro/.

6. anadicsinaloa.com. [En línea] 18 de Agosto de 2009. [Citado el: 6 de Marzo de 2010.] http://www.anadicsinaloa.com/index.php?option=com_content&view=article&id=207:caracteristicas-mysql&catid=16:anadic-sinaloa&Itemid=33.

7. **Miguel, Angel Alvarez.** desarrolloweb.com. [En línea] [Citado el: 6 de Marzo de 2010.] http://www.desarrolloweb.com/articulos/que-es-html.html.

8. maestrosdelweb.com. [En línea] [Citado el: 6 de Marzo de 2010.] http://www.maestrosdelweb.com/editorial/introcss/.

9. **Javier, Garcia Gallego Felipe.** css.infames.org. [En línea] [Citado el: 7 de Marzo de 2010.] http://css.infames.org/ventajas.html.

10. **Miguel, Angel Alvarez.** desarrolloweb.com. [En línea] 25 de Marzo de 2009. [Citado el: 7 de Marzo de 2010.] http://www.desarrolloweb.com/articulos/introduccion-jquery.html.

11. tufuncion.com. [En línea] 6 de Marzo de 2007. [Citado el: 8 de Marzo de 2010.] http://www.tufuncion.com/ventajas-ajax.

12. linuxcentro.net. [En línea] 22 de Febrero de 2007. [Citado el: 8 de Marzo de 2010.] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP.

13. **Alberto, Román Zamitiz Carlos.** profesores.fi-b.unam.mx. [En línea] [Citado el: 8 de Marzo de 2010.] http://profesores.fi-b.unam.mx/carlos/aydoo/uml.html.

14. members.fortunecity.com. [En línea] [Citado el: 8 de Marzo de 2010.] http://members.fortunecity.com/software1/herramie.htm.

15. freedownloadmanager. [En línea] 5 de Marzo de 2007. [Citado el: 9 de Marzo de 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_14720_p/.

16. **Alvaro.** uptodown. [En línea] [Citado el: 9 de Marzo de 2010.] http://netbeans-ide.uptodown.com/.

17. **A., González Barbone Víctor.** iie.fing.edu.uy. [En línea] [Citado el: 9 de Marzo de 2010.] http://iie.fing.edu.uy/~nacho/blandos/seminario/XProg1.html.

BIBLIOGRAFÍA

1. **Miguel, Angel Alvarez.** desarrolloweb.com. [Online] Noviembre 11, 2008. [Cited: Marzo 5, 2010.] http://www.desarrolloweb.com/articulos/que-es-un-cms.html.

2. cms-hispano.org. [Online] [Cited: Marzo 5, 2010.] http://cms-hispano.org/index.php?s=content&p=necesidad.

3. **César, Martín.** alzado.org. [Online] Julio 27, 2009. [Cited: Marzo 5, 2010.] http://www.alzado.org/articulo.php?id_art=798.

4. editum.org. [Online] Diciembre 25 , 2007. [Cited: Marzo 6, 2010.] http://www.editum.org/Que-Es-Un-Servidor-De-Aplicaciones-p-473.html.

5. ciberaula.com. [Online] [Cited: Marzo 6, 2010.] http://linux.ciberaula.com/articulo/linux_apache_intro/.

6. anadicsinaloa.com. [Online] Agosto 18, 2009. [Cited: Marzo 6, 2010.] http://www.anadicsinaloa.com/index.php?option=com_content&view=article&id=207:caracteristicas-mysql&catid=16:anadic-sinaloa&Itemid=33.

7. **Miguel, Angel Alvarez.** desarrolloweb.com. [Online] [Cited: Marzo 6, 2010.] http://www.desarrolloweb.com/articulos/que-es-html.html.

8. maestrosdelweb.com. [Online] [Cited: Marzo 6, 2010.] http://www.maestrosdelweb.com/editorial/introcss/.

9. **Javier, Garcia Gallego Felipe.** css.infames.org. [Online] [Cited: Marzo 7, 2010.] http://css.infames.org/ventajas.html.

10. **Miguel, Angel Alvarez.** desarrolloweb.com. [Online] Marzo 25, 2009. [Cited: Marzo 7, 2010.] http://www.desarrolloweb.com/articulos/introduccion-jquery.html.

11. tufuncion.com. [Online] Marzo 6, 2007. [Cited: Marzo 8, 2010.] http://www.tufuncion.com/ventajas-ajax.

12. linuxcentro.net. [Online] Febrero 22, 2007. [Cited: Marzo 8, 2010.] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP.

13. **Alberto, Román Zamitiz Carlos.** profesores.fi-b.unam.mx. [Online] [Cited: Marzo 8, 2010.] http://profesores.fi-b.unam.mx/carlos/aydoo/uml.html.

14. members.fortunecity.com. [Online] [Cited: Marzo 8, 2010.] http://members.fortunecity.com/software1/herramie.htm.

15. freedownloadmanager. [Online] Marzo 5, 2007. [Cited: Marzo 9, 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_14720_p/.

16. **Alvaro.** uptodown. [Online] [Cited: Marzo 9, 2010.] http://netbeans-ide.uptodown.com/.

17. **A., González Barbone Víctor.** iie.fing.edu.uy. [Online] [Cited: Marzo 9, 2010.] http://iie.fing.edu.uy/~nacho/blandos/seminario/XProg1.html.

18. ajaxya.com. [Online] [Cited: Marzo 14, 2010.] http://www.ajaxya.com.ar.

19. librosweb.es. [Online] [Cited: Marzo 14, 2010.] http://librosweb.es/ajax/index.html.

20. extremeprogramming.org. [Online] Septiembre 28, 2009. [Cited: Marzo 14, 2010.] http://www.extremeprogramming.org.

21. [Online] [Cited: Marzo 14, 2010.] http://www.google.com.cu/url?sa=t&source=web&ct=res&cd=5&ved=0CC0QFjAE&url=http%3A%2F%2Fapuntes-utn.com.ar%2Fap.

22. slideshare.net. [Online] [Cited: Marzo 15, 2010.] http://www.slideshare.net/dersteppenwolf/la-ingeniera-de-software-y-rup.

23. masadelante.com. [Online] [Cited: Marzo 16, 2010.] http://www.masadelante.com/faqs/html.

24. lawebera.es. [Online] [Cited: Marzo 16, 2010.] http://www.lawebera.es/manuales/php/1.php.

25. luauf.com. [Online] [Cited: Marzo 17, 2010.] http://luauf.com/2008/05/13/entornos-de-desarrollo-integrado-para-java/.

26. **Joaquin, Gracia.** ingenierosoftware.com. [Online] Mayo 7, 2005. [Cited: Marzo 18, 2010.] http://www.ingenierosoftware.com/analisisydiseno/uml.php.