Universidad de las Ciencias Informáticas Facultad 8

Sistema para la gestión de la información referente a los problemas tanto de software como de hardware que existen en las áreas productivas de la UCI

Autor: Raimundo Adrian Acosta Medina

Tutores: Ing. Arcel Labrada Batista

Ing. Eduardo Manuel Macías Sotolongo

Ciudad de La Habana

"Año del 50 aniversario del triunfo de la Revolución"

Declaración de autoría.

Declaro que soy el único autor del trabajo “Sistema para la gestión de la información referente a los problemas tanto de software como de hardware que existen en las áreas productivas de la UCI” y autorizo a la Facultad 8 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los 23 días del mes de Junio del año 2009.

Autor:

Raimundo Adrian Acosta Medina

Tutor:

Arcel Labrada Batista

Eduardo Manuel Macías Sotolongo

Agradecimientos.

Quisiera agradecer antes que todo a mis padres y a mi hermano, quienes son los principales responsables de la existencia de este trabajo, por estar siempre a mi lado.

Quisiera agradecer a todos aquellos que han brindado su ayuda desinteresada para la creación de este trabajo, en especial a mis compañeros de proyecto por siempre estar ahí en cualquier momento, a Leonardo y a Héctor por ayudarme desde un principio.

A mis amigos por ayudarme durante estos 5 años, este trabajo se lo debo a ustedes, muchas gracias.

Dedicatoria.

Quisiera dedicar este trabajo a mi abuelito por ser la luz que siempre me ha guiado y a mi abuelita, un besote para ellos.

Quisiera dedicarlo especialmente a mis padres que son el motor que mueve mi ser y a mi hermano por ser el paradigma de persona que siempre he tratado de ser.

Resumen.

A lo largo del documento, se exponen los elementos teóricos que soportan el trabajo. Ellos son, el estudio de sistemas existentes con el fin de gestionar información, así como algunas de las características más relevantes de herramientas, lenguajes y metodologías, entre las cuales se encuentran las empleadas en el desarrollo de la propuesta de solución. Se muestran las fases de exploración y planificación que corresponden a la metodología de desarrollo utilizada para llevar a cabo la implementación y desarrollo del sistema propuesto. Además se presentan los artefactos que se generan en el transcurso de dichas fases. También se detallan los artefactos del diseño, implementación y prueba del sistema, y se explica el diseño de la base de datos.

Índice

Índice de contenidos

Índice de tablas

Índice de figuras

Introducción

La Universidad de las Ciencias Informáticas (UCI) crece día a día exponencialmente, y junto con ella, aumenta la producción de software. Aparejado a este aumento, vienen surgiendo un grupo de problemas en las áreas productivas, tanto de software como de hardware, que sin lugar a dudas atrasan el desarrollo y la creación del producto, y en muchas ocasiones, la comunidad universitaria tiene soluciones que se desconocen debido a que no existe un sistema que permita a los usuarios de la UCI revelarlas.

Ejemplo de ello se pueden encontrar en algunos proyectos, que en ocasiones tienen que desarrollar un software que, sin saber, tiene con otros varias cosas en común. Sin embargo, se encuentran problemas a la hora del desarrollo, que pueden surgir debido a que no se usa el lenguaje de programación adecuado o las herramientas de hardware no son las indicadas, pues se tiene conocimiento de la existencia de proyectos con producciones similares, que pueden en algún momento haber presentado la misma problemática y ya tener la solución en sus manos. Surgiendo así la **situación problémica** de esta investigación.

Debido a que esta investigación tiene entre sus fines dar paso a la creación de una aplicación que se utilizará como soporte para la gestión y centralización de la información que se ha estado tratando anteriormente, quedó conformado como **problema** de la investigación, ¿Cómo gestionar y centralizar la información referente a los problemas tanto de software como de hardware, detectados en las áreas productivas en la UCI, así como las soluciones?

Como **objeto de estudio** de esta investigación, los sistemas dedicados a la gestión de información importante para el desarrollo de software.

Quedando definido como **campo de acción**, los sistemas dedicados a la gestión de información importante para el desarrollo de software en las áreas productivas de UCI.

Como **idea a defender** de esta investigación se plantea que:

Con el desarrollo de un sistema que gestione la información referente a problemas tanto de software como de hardware existentes en las áreas productivas de la UCI, así como sus soluciones, se contribuirá al desarrollo del software en la UCI.

El **objetivo general** de esta investigación es desarrollar un sistema para gestionar y centralizar los problemas tanto de software como de hardware, detectados en las áreas productivas en la UCI, así como sus soluciones, de forma eficiente.

Para el desarrollo del objetivo general de nuestra investigación, se han definido como **objetivos específicos**:

1. Realizar un estudio del estado de la Gestión de la Información a nivel internacional.
2. Realizar un análisis de las aplicaciones existentes que tienen como fin la gestión de información.
3. Definir las diferentes clasificaciones sobre las cuales se encuentran los principales problemas tanto de software como de hardware existentes en las áreas productivas de la UCI.
4. Determinar los elementos necesarios, que permitirán la creación del sistema para la gestión, clasificación y procesamiento de información referente a los problemas tanto de software como de hardware detectados en las áreas productivas en la UCI.
5. Determinar las normas que regularán la participación de los usuarios en el sistema para la gestión, clasificación y procesamiento de la información referente a los problemas tanto de software como de hardware detectados en las áreas productivas en la UCI.

Como **tareas a desarrollar** con el fin de darle solución a los objetivos de la investigación, quedaron conformadas las siguientes:

1. Investigación sobre el estado del arte:
    - Realizar revisión bibliográfica del tema.
    - Analizar la existencia de otras soluciones similares.
2. Elaborar el diseño teórico de la investigación:

3. Definir la situación problémica, el problema, el objetivo general y los objetivos específicos, el objeto de estudio, el campo de acción y la idea a defender.

    - Elaborar la propuesta de solución:

    - Identificar y definir los requerimientos de la aplicación, para así realizar las historias de usuario.

4. Realizar el análisis y el diseño del sistema a desarrollar.

5. Implementar la aplicación para la gestión de la información referente a los problemas tanto de software como de hardware, detectados en las áreas productivas en la UCI.

6. Analizar los resultados:

    - Realizar prueba de la aplicación.

Con el objetivo de lograr un mejor entendimiento del documento, el mismo fue estructurado en 3 capítulos:

Capítulo 1. Fundamentación Teórica: En este capítulo se exponen los elementos teóricos que soportan el trabajo realizado. Estos son: el estudio de sistemas existentes con el fin de gestionar información, así como algunas de las características más relevantes de herramientas, lenguajes y metodologías, entre las cuales se encuentran las empleadas en el desarrollo de la propuesta de solución, acompañadas de una breve justificación de su selección.

Capitulo 2. Exploración y Planificación: En este capítulo se muestran las fases de exploración y planificación que corresponden a la metodología de desarrollo utilizada para llevar a cabo la implementación y desarrollo del sistema propuesto. Además se presentan los artefactos que se generan en el transcurso de dichas fases.

Capitulo 3. Diseño, implementación y pruebas: En este capitulo se detallan los artefactos del diseño, implementación y prueba del sistema, así como se explica el diseño de la base de datos.

Capítulo 1: Fundamentación Teórica

1.1 Introducción.

En el presente capítulo, se ofrece un breve panorama de las tecnologías y las tendencias actuales que permitieron la selección de las herramientas, lenguajes de programación y metodologías, que fueron utilizados durante el desarrollo de la propuesta de solución para el problema planteado. También se presentan los principales conceptos y sistemas informatizados, así como los elementos teóricos que soportan el trabajo realizado.

1.2 Gestión de la Información.

Actualmente se maneja un gran volumen de información, lo cual implica que se deba emplear mucho tiempo en la organización de la información. Esto se debe, entre otras causas, a la liberación de los mecanismos regulatorios existentes en materia de publicaciones, que el surgimiento y desarrollo de Internet ha incrementado.

Hoy en día la información es un elemento fundamental para el desarrollo, por lo que el proceso de analizar y utilizar la información que se ha recabado y registrado para permitir a los administradores (de todos los niveles) tomar decisiones, que se denomina gestión de información, con el paso de los años ha pasado a ocupar, un mayor espacio a escala mundial en la economía de los países.

La gestión de la información implica:

- Determinar la información que se precisa.
- Recoger y analizar la información.
- Registrarla y recuperarla cuando sea necesaria.
- Utilizarla.
- Divulgarla.

1.3 Existencia de sistemas automatizados referentes al campo de acción.

En la actualidad existen disímiles aplicaciones web dedicadas la gestión de información importante, un ejemplo de ellas es el sitio todoexpertos.com, éste trata diferentes temáticas mediante las cuales clasifica la información a tratar en el mismo. El presente trabajo de diploma propone una solución similar a la antes mencionada, teniendo en cuenta las diferentes funcionalidades que brinda el mismo. A continuación se muestra parte de la interfaz del mismo.

todoexpertos.com

Figura 1: Sitio todoexpertos.com

En la UCI existen diferentes foros, donde se tratan temáticas muy diversas, en las cuales muchas veces se exponen diferentes problemáticas a la que muchos casos, se les dan respuestas y proposiciones de soluciones. Pero como hoy en día no existe una aplicación

donde se puedan almacenar estos problemas y sus soluciones, tampoco existen reportes de los mismos y no llega a conocimiento de la alta dirección del centro.

1.4 Tendencias y tecnologías actuales

1.4.1 Los Servicios Web.

Existen diversas definiciones para los servicios web; para lograr un mejor entendimiento, se pueden definir como, un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web. [1]

Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar. [1]

En este proceso de los servicios web intervienen una serie de tecnologías que hacen posible esta circulación de información. Por un lado, estaría SOAP (Protocolo Simple de Acceso a Objetos). Se trata de un protocolo basado en XML, que permite la interacción entre varios dispositivos y que tiene la capacidad de transmitir información compleja. Los datos pueden ser transmitidos a través de HTTP, SMTP, etc. SOAP especifica el formato de los mensajes. El mensaje SOAP está compuesto por un envolpe (sobre), cuya estructura está formada por los siguientes elementos: header (cabecera) y bady (cuerpo). [1]

Para optimizar el rendimiento de las aplicaciones basadas en Servicios Web, se han desarrollado tecnologías complementarias a SOAP, que agilizan el envío de los mensajes (MTOM) y los recursos que se transmiten en esos mensajes (SOAP-RRSHB). [1]

Por otro lado, WSDL (Lenguaje de Descripción de Servicios Web), permite que un servicio y un cliente establezcan un acuerdo en lo que se refiere a los detalles de transporte de mensajes y su contenido, a través de un documento procesable por dispositivos. WSDL representa una especie de contrato entre el proveedor y el que solicita. WSDL especifica la sintaxis y los mecanismos de intercambio de mensajes. [1]

Los servicios web juegan hoy en día un papel imprescindible en el desarrollo de las tecnologías, estas aplicaciones que permiten el flujo de información a través de la web, se puede decir que, a su vez han logrado un ascenso irreversible en el intercambio de datos alrededor del mundo.

1.4.2 Servidores Web.

Un servidor web es el ordenador encargado de manejar y almacenar los sitios web, este usa el protocolo de transferencia http para enviar páginas web al ordenador de un usuario cuando este las solicita.

Básicamente, un servidor web sirve contenido estático a un navegador, carga un archivo y lo provee a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante el protocolo de transferencia HTTP. Se pueden utilizar varias tecnologías en el servidor para aumentar su potencia más allá de su capacidad de entregar páginas HTML; éstas incluyen scripts CGI, seguridad SSL y páginas activas del servidor (ASP).[2]

1.4.2.1 Internet Information Server (IIS).

Internet Information Server, es el servidor de páginas web de Microsoft, una serie de servicios para los servidores que funcionan con Windows. IIS convierte a un ordenador en un servidor de Internet o Intranet. Es rápido y recomendable para la plataforma Windows 2000, dado por la integración que presenta con su Servicio de Directorios que permite el desarrollo de aplicaciones basadas en la Web fiables y escalables.

IIS 5.0 ofrece una administración muy sencilla que se realizará mediante el Administrador de servicios de Internet. Esta versión permite que el desarrollo de aplicaciones Web sea mucho más robusto y la creación de sitios Web sea más configurable y completa. Ofrece un entorno escalable basado en los componentes cliente/servidor que se pueden integrar dentro de las aplicaciones Web.

1.4.2.2 Servidor Web Apache.

Apache es el servidor web hecho por excelencia, está diseñado para ser un servidor web potente y flexible que pueda funcionar en la más amplia variedad de plataformas y entornos. Su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa.

Es importante tener en cuenta el por qué apache es un servidor muy potente y reconocido a nivel internacional, lo que viene dado por su capacidad de correr en múltiples Sistemas Operativos; es una tecnología gratuita de código fuente abierto; es un servidor altamente configurable de diseño modular; es capaz de interpretar varios lenguajes de programación entre los que se encuentran Perl y PHP; permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor y tiene una alta configurabilidad en la creación y gestión de logs, entre otras características que lo hacen prácticamente universal.

Los módulos de Apache pueden clasificarse en tres categorías específicas:

Módulos Base: Módulo con las funciones básicas del Apache.

Módulos Multiproceso (para manejar las peticiones)**:** Son los responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos a atender a las peticiones. Se han diseñado varios módulos multiprocesos para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código.

Módulos Adicionales: Cualquier otro módulo que le añada una funcionalidad al servidor. Simplemente hay que añadirle un módulo, de forma que no es necesario volver a instalar el software.

1.4.3 Lenguajes de programación para el desarrollo de Aplicaciones Web.

Un lenguaje de programación no es más que el medio que permite a las computadoras, interpretar las órdenes que se les dan, así como controlar su comportamiento. Este consiste en un grupo de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos respectivamente. A través de estas reglas el programador crea los programas o subprogramas. También dichos programas están formados por las instrucciones, conocidas como código fuente.

Los lenguajes de programación para el desarrollo de aplicaciones web, están divididos en dos grandes grupos, los Lenguajes del lado del servidor y los Lenguajes del lado del cliente.

1.4.3.1 Lenguajes del lado del servidor.

Los lenguajes de lado servidor son aquellos lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él. [3] Consiste en el procesamiento de una petición de un usuario mediante la interpretación de un script en el servidor web para generar páginas HTML dinámicamente como respuesta.

1.4.3.1.1 Python.

Python es un lenguaje de programación de muy alto nivel, por lo que tiene incluido tipos de datos al mismo nivel, con matrices flexibles y diccionarios. Es un lenguaje interpretado, lo que ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar.

Python permite dividir su programa en módulos reutilizables desde otros programas en Python. Viene con una gran colección de módulos estándar que se pueden utilizar como base para los programas (o como ejemplos para empezar a aprender Python). [4]

Debido a su sencillez, velocidad para crear los programas, cantidad de librerías que contiene, tipos de datos y funciones incorporadas en el mismo lenguaje, Python se ha hecho muy popular en estos últimos años.

1.4.3.1.2 Hypertext Preprocessor (PHP).

PHP es un lenguaje de script usado principalmente para códigos a ejecutar en servidores web, sobre todo Apache, es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. Puede verse como la alternativa open-source a ASP.

El PHP inició como una modificación a Perl escrita por Rasmus Lerdorf a finales de 1994. Su primer uso fue el de mantener un control sobre quien visitaba su currículum en su web. En los siguientes tres años, se fue convirtiendo en lo que se conoce como PHP/FI 2.0. Esta forma de programar llegó a muchos usuarios, pero el lenguaje no tomó el peso actual hasta que Zeev Surasky y Andi Gutmans le incluyeron nuevas características en 1997, que dio por resultado el PHP 3.0. [5]

Entre las características fundamentales de este lenguaje se encuentran que, es el soporte para gran cantidad de gestores de bases de datos, como mSQL, MySQL, InterBase, Informix, Oracle, PostgreSQL, y soporte para la mayoría de los servidores Web de hoy en día, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape y Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd entre otros muchos, que, para aquellos que soporten el estándar CGI, PHP puede usarse como procesador CGI. También la integración con las varias bibliotecas externas, que permiten al desarrollador hacer disímiles cosas, desde generar documentos en PDF hasta analizar código XML y brinda una solución simple y universal para las paginaciones dinámicas de la Web de fácil programación.

El hecho de ser un producto de código abierto, hace que PHP cuente con la ayuda de un gran grupo de programadores, lo que permite a su vez que los fallos de funcionamiento se encuentren y se reparen rápidamente, permitiendo además, que el código se ponga al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.

1.4.3.2 Lenguajes del lado cliente.

Los lenguajes de lado cliente son aquellos que pueden ser directamente "digeridos" por el navegador y no necesitan un pre tratamiento. Son totalmente independientes del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio sin necesidad de pagar más ya que, por regla general, los servidores que aceptan páginas con códigos del lado servidor son en su mayoría de pago o sus prestaciones son muy limitadas.[3] Poseer dicha característica, obliga a contar con él sin albergar dudas.

1.4.3.2.1 Hypertext Markup Language (HTML).

El formato HTML es el lenguaje estándar para la web, y fue definido por una organización internacional de estandarización (el Consorcio W3). El HTML es un formato universal flexible, rico y compacto. [6] HTML es un conjunto de símbolos o palabras que definen varios componentes de un documento Web; estos se pueden ver siempre dentro de las etiquetas "<", ">". Es un lenguaje simple de marcado utilizado para crear documentos de hipertexto para WWW. [7]

HTML, no solo permite establecer hiperenlaces entre diferentes documentos, sino que es un lenguaje de descripción independiente de la plataforma en que se utilice. Es decir, un documento HTML contiene toda la información necesaria sobre su estructura y con el usuario, es luego el buscador que se utilice el responsable de asegurar que el documento tenga un aspecto coherente, independiente del tipo de máquina desde donde se acceda al documento. [6]

HTML es un lenguaje que hace posible presentar información en Internet. Lo que se ve al visualizar una página en Internet no es más que la interpretación que hace el navegador del código HTML. Al mismo tiempo que si se quiere ver el código HTML basta con hacer clic en la opción "Ver" de la barra de menús y elegir "Código fuente" (en Internet Explorer).

Entre las ventajas de HTML se encuentra el hecho de ser muy fácil de aprender, lo que permite que cualquier persona, pueda enfrentarse a la tarea de crear una Web; permite utilizar estilos en formato CSS en las páginas para una mayor facilidad en su modificación y los contenidos son fáciles de actualizar. Además, su amplia difusión y el número de documentos estructurados

según sus normas son tan grandes que su consideración como lenguaje de definición de estructura se hace obligatoria.

1.4.3.2.2 JavaScript.

Javascript es un lenguaje de programación que permite a los desarrolladores crear acciones en sus páginas web. 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. Es un lenguaje con muchas posibilidades, utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes, orientados a objetos mucho más complejos. Con Javascript se puede crear diferentes efectos e interactuar con nuestros usuarios. [8]

Javascript tiene como principal característica ser un lenguaje independiente de la plataforma. Entre otra de sus características se tiene 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. También se puede mencionar que es un lenguaje basado en acciones que posee menos restricciones. Además, es un lenguaje que utiliza Windows y sistemas X-Windows, gran parte de la programación en este lenguaje está centrada en describir objetos, escribir funciones que respondan a movimientos del mouse, aperturas, utilización de teclas, cargas de páginas entre otros. [8]

Entre las funcionalidades de Javascript se puede destacar que, proporciona los medios para:

Controlar las ventanas del navegador y el contenido que muestran.

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.

Javascript es el lenguaje que por sus características y funcionalidades, facilitará la programación del lado del cliente.

1.4.3.2.3 Cascading style sheets (CSS).

El CSS (hojas de estilo) es un formato usado en las páginas web para separar el estilo (la forma en la que se ve una página web) de la estructura (o código), es una característica del HTML que da más control a los programadores, diseñadores y usuarios de la web sobre cómo se muestran las páginas web. Con CSS los diseñadores crean hojas de estilo que definen cómo se mostrarán diversos elementos, como encabezados, enlaces, texto, imágenes, etc. Estas hojas de estilo se pueden entonces aplicar a cualquier página o a todas las páginas en un sitio web particular, lo que hace la codificación mucho más fácil. El término cascading (en cascada) deriva del hecho de que múltiples hojas de estilo se pueden aplicar a una misma página web. El CSS fue desarrollado por el W3C; sin embargo, esta especificación continúa desarrollándose y no es apoyada completamente por todos los navegadores. [26]

Entre las ventajas de utilizar CSS vale mencionar que:

1. Se tiene el control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
2. Los Navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
3. Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario. Por ejemplo, para ser impresa, mostrada en un dispositivo móvil, o ser "leída" por un sintetizador de voz.
4. El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño. [27]

1.4.4 Integrated Development Environment ('IDE').

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI. Puede ser parte de aplicaciones existentes o bien ser aplicaciones por sí solas. Además puede servir para depurar y facilitar las diferentes tareas necesarias en el desarrollo de cualquier tipo de aplicación. Un mismo IDE pueda funcionar con diferentes lenguajes de programación, un ejemplo de ello es Eclipse.

1.4.4.1 Zend Studio.

Se trata de un programa de la casa Zend, uno de los mayores impulsores de PHP, orientada a desarrollar aplicaciones web, como no, en PHP. Zend Studio es un editor de texto para páginas PHP que proporciona un buen número de ayudas desde la creación y gestión de proyectos hasta la depuración del código. [9]

Zend Studio está escrito completo en Java, lo que ha permito a Zend lanzar con relativa rapidez y facilidad versiones del producto para Windows, Linux y MacOs, pero junto con esta ventaja, el hecho de estar escrito en Java hace que no funcione tan rápido como otras aplicaciones de uso diario.

Consta de dos partes en las que se dividen las funcionalidades, las de parte del cliente y las del servidor. Permite hacer depuraciones simples de scripts. Dispone de herramientas para gestionar los proyectos, lo cual es muy útil para mejorar la productividad en la programación.

1.4.4.2 EasyEclipse.

EasyEclipse es un editor (IDE) de alto desarrollo para PHP basado en Eclipse, es multiplataforma y código abierto que, y sin dudas ha logrado buenos resultados. Este proyecto empaqueta el entorno de desarrollo Eclipse junto con una cuidada selección de "plugins" código abierto para obtener un IDE final excepcionalmente bueno para el desarrollo de aplicaciones en PHP, Python, Ruby y Java.

El desarrollador final que utiliza EasyEclipse, sólo tiene que preocuparse del código de su aplicación y no de afinar su IDE, ya que cuenta con todos los plugins ya instalados y configurados. Además dispone de varias "distribuciones": para desarrollo Java de servidor, de aplicaciones Java de escritorio, para dispositivos móviles, para LAMP, para PHP, para Python y para Ruby. Provee un grupo de herramientas específicas y robustas tan pequeñas y simples como sea posible, siendo cada distribución empaquetada para un entorno específico con las funcionalidades justas para ese ambiente.

1.4.5 Sistemas Gestores de Base de Datos (SGBD)

Un Sistema Gestor de base de datos (SGBD) es un conjunto de programas que permiten crear y mantener una Base de datos, asegurando su integridad, confidencialidad y seguridad. [10] Es una aplicación que permite a los usuarios definir, crear y mantener la BD y proporciona un acceso controlado a la misma.

Todo SGBD debe proporcionar los siguientes servicios:

- Creación y definición de la base de datos.
- Manipulación de los datos.
- Acceso controlado a los datos mediante mecanismos de seguridad.
- Mantener integridad y consistencia de los datos.
- Acceso compartido a la base de datos.
- Mecanismos de copias de respaldo y recuperación. [11]

1.4.5.1 MySQL.

El software MySQL® proporciona un servidor de base de datos SQL (Structured Query Language) muy rápido, multi-hilo, multi usuario y robusto. El servidor MySQL está diseñado para entornos de producción críticos, con alta carga de trabajo así como para integrarse en

software para ser distribuido. MySQL es una marca registrada de MySQL AB, [12] que desarrolla MySQL como software libre en un esquema de licenciamiento dual.

MySQL es un sistema de gestión de bases de datos SQL Open Source, trabaja en entornos cliente/servidor o incrustados, una gran cantidad de software de contribuciones están disponibles para él, es muy rápido, fiable y fácil de usar. Además es muy utilizado en múltiples plataformas como: AIX/GNU-Linux/Windows 95/Windows 98/Windows NT/Windows 2000/Windows XP/ Windows Vista y otras versiones de Windows/Mac OS X/Solaris/SunOS-Apache-MySQL-PHP/Perl/Python; también es utilizado por herramientas de seguimiento de errores como Bugzilla y en aplicaciones Web como MediaWiki o Drupal.

1.4.5.2 Oracle.

Oracle es básicamente una herramienta cliente/servidor para la gestión de Bases de Datos. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales, por norma general. [13]

Es considerado como uno de los sistemas de bases de datos más completos, donde se destacan su soporte de transacciones, estabilidad, escalabilidad y soporte multiplataforma.

1.4.5.3 PostgreSQL.

Es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) que ha sido desarrollado de varias formas desde 1977. [14].

PostgreSQL proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales tales como DB2 u Oracle. La siguiente es una breve lista de algunas de esas características, a partir de PostgreSQL 7.1.x. [14]

DBMS Objeto-Relacional

PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transiciones, optimización de consultas, herencia, y arreglos. [14]

Altamente Extensible

PostgreSQL soporta operadores, funciones métodos de acceso y tipos de datos definidos por el usuario. [14]

Soporte SQL Comprensivo

PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92. [14]

Integridad Referencial

PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos. [14]

API Flexible

La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, y Pike. [14]

Lenguajes Procedurales

PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido. [14]

Cliente/Servidor

PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL. [14]

PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo, lo cual conlleva a contar.

1.4.6 Lenguajes de Modelado Visual

El Modelado Visual es el modelado de una aplicación usando notaciones gráficas. Un modelo es una simplificación de la realidad. El objetivo del modelado de un sistema es capturar las partes esenciales del sistema. Para facilitar este modelado, se realiza una abstracción y se plasma en una notación gráfica. Esto se conoce como modelado visual.

Entre los objetivos de este modelado visual se encuentra que sea independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML se puedan implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos).

El modelado visual del software ayuda a capturar las partes esenciales de un sistema:

Se utiliza para capturar los procesos de negocios desde la perspectiva del usuario.

El Modelado Visual se utiliza para analizar y diseñar una aplicación, distinguiendo entre los dominios del negocio y los dominios de la computadora.

Ayuda a reducir la complejidad.

El Modelado Visual se realiza de manera independiente al lenguaje de implementación.

Promueve la reutilización de componentes.

1.4.6.1 Lenguaje Unificado de Construcción de Modelos (UML).

El Lenguaje Unificado de Construcción de Modelos (UML), es ante todo un lenguaje. Un lenguaje proporciona un vocabulario y unas reglas para permitir una comunicación. En este caso, este lenguaje se centra en la representación gráfica de un sistema. Provee un sistema de arquitecturas trabajando con objetos, análisis y diseño, con una buena consistencia del lenguaje para especificar, visualizar, construir y documentar los artefactos de un sistema de software.

Los objetivos que se fijaron al desarrollar el UML fueron los siguientes:

Proporcionar a los usuarios un Lenguaje de Modelado Visual de tal forma que sea posible intercambiar información de los modelos.

Proporcionar mecanismos de extensibilidad y especialización para ampliar los conceptos básicos.

Ser independiente de un lenguaje en particular y del proceso de desarrollo.

Proporcionar bases formales para la comprensión del Lenguaje de Modelado.

Integración en una mejor práctica.

Sus funciones:

Visualizar: UML permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.

Especificar: UML permite especificar cuáles son las características de un sistema antes de su construcción.

Construir: A partir de los modelos especificados se pueden construir los sistemas diseñados.

Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

UML es además un método formal de modelado. Esto aporta las siguientes ventajas:

Mayor rigor en la especificación.

Permite realizar una verificación y validación del modelo realizado.

Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener la visión en el diseño, de más alto nivel, de la estructura de un proyecto.

1.4.7 Metodologías de Desarrollo de Software

Una metodología es un proceso que se encarga de elaborar estrategias para el desarrollo de un software. Fue definida como “el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software”, según Jacobson, Booch y Rumbaugh en su libro “El Proceso Unificado de Desarrollo de Software”. En la ingeniería de software de un proyecto, la metodología de desarrollo de software define quién debe hacer qué, cuándo y cómo debe hacerlo.

1.4.7.1 Metodologías Ágiles.

Las metodologías ágiles son sin duda uno de los temas recientes en ingeniería de software que están acaparando gran interés. Estas aportan nuevas técnicas y métodos de trabajo para el desarrollo de cada etapa de un software. Satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor, es sin lugar a dudas la prioridad de estas metodologías, además construir el proyecto en torno a individuos motivados y lograr que el personal del negocio y los desarrolladores trabajen juntos a lo largo del proyecto, así como capturar los cambios para que el cliente tenga una ventaja competitiva, entre otras.

Las metodologías de desarrollo de software ‘Livianas’ ayudan en el Análisis y Diseño Orientados a Objetos y son ideales en ambientes con herramientas basadas en Java y programación Web. Son llamadas ‘Livianas’ (LightWeight), debido a que no producen

demasiado overhead sobre las actividades de desarrollo, y no impiden el avance de los proyectos, ni obstruyen, poniéndose adelante.

1.4.7.1.1 Programación Extrema (XP).

Extreme Programing (XP) se basa en la idea de que existen cuatro variables que guían el desarrollo de sistemas: Costo, Tiempo, Calidad y Alcance. La manera de encarar los desarrollos avalados por este modelo de desarrollo es permitir a las fuerzas externas (gerencia, clientes) manejar hasta tres de estas variables, quedando el control de la restante en manos del equipo de desarrollo. Este modelo hace visibles de manera más o menos continua estas cuatro variables. Lo que dicta XP marca la diferencia que permite realizar hoy el software que cubra las necesidades de hoy, para que cuando mañana se conozcan mejor las necesidades futuras, se realicen en el momento en que se necesiten. Utiliza una aproximación minimalista y de mejora continua. [15]

XP promueve valores como: Comunicación, Simplicidad, Realimentación, Coraje. De estos cuatro valores recién mencionados, XP deriva una docena de Principios Básicos para guiar en su estilo de desarrollo de sistemas. La derivación de los principios esenciales es clara, desde las premisas anteriores: Realimentación rápida, Asumir la Simplicidad, El Cambio Incremental, Adherirse (Abrazar) al Cambio, Trabajo de Alta Calidad (desde 'trabajo excelente' hasta 'trabajo increíblemente sobresaliente'). Luego se puede mencionar algunos principios no tan evidentes inicialmente como: Enseñar a Aprender, La Inversión Inicial Debe Ser Pequeña, Jugar Para Ganar, Hacer Experimentos Concretos (reduce la discusión que lleva a divagar), Comunicación Honesta y Abierta, Trabajar a Favor de los Instintos de la Gente y no Contra Ellos, Fomentar la Aceptación de Responsabilidad, Adaptarse a las Condiciones Locales, Viajar Liviano, Mediciones Honestas (por ejemplo, de grado de avance). [15]

A continuación se mencionan las Doce Prácticas de XP que permiten realizar desarrollos de Alta Calidad, en Tiempo y Costo razonables: [15]

Jugar el Juego de la Planificación: Rápidamente determinar el alcance del próximo release, combinando las prioridades de negocios con los estimados técnicos. Cuando la realidad sobrepasa el Plan, adaptar el Plan. [15]

Hacer Pequeños Releases: Poner un sistema simple en producción rápidamente, entonces liberar nuevas versiones del mismo en un ciclo de desarrollo rápido, una por semana a una por mes. Cada ciclo no debería ser más largo. [15]

Hacer Historias y Usar Metáforas: Guiar todo el desarrollo del sistema a través de una Historia Compartida por el Equipo (o Metáfora) acerca de cómo trabaja (o debería trabajar) el Sistema. [15]

Diseñar Simple: El Sistema debería diseñarse de la manera más simple posible en cualquier momento dado. La complejidad extra es removida, tan pronto como es descubierta (ver Refactoring debajo). [15]

Probar - Testear: Los Desarrolladores continuamente escriben Testeos Unitarios, los cuales deben correr sin error para que el desarrollo pueda continuar. Cuando se detecta un error en una corrida, su reparación pasa a ser la máxima prioridad para el Programador y/o el Equipo. Los Clientes (ayudados por Desarrolladores) escriben Tests Funcionales para probar qué funcionalidades están terminadas de acuerdo a sus expectativas. [15]

Rearmar - Refactorizar: Los Desarrolladores reestructuran el sistema sin cambiar su comportamiento para remover duplicación de código, mejorar la comunicación, simplificar el código, o agregar flexibilidad. [15]

Programar por Pares: Todo el código desarrollado es escrito por dos desarrolladores sentados frente a una única estación de trabajo. [15]

Propiedad Colectiva: Cualquier integrante del Equipo puede cambiar cualquier código de cualquier parte del sistema en cualquier momento. [15]

Integrar Continuamente: El sistema se integra y se construye (por ejemplo, se compila), es decir, se unen sus partes, varias veces por día, hasta el extremo de integrar el sistema completo, cada vez que se termina una tarea. [15]

Semanas de 40 Horas: Trabajar no más de cuarenta horas por semana como una regla estándar. Nunca trabajar sobre-tiempo dos semanas seguidas; si esto es necesario, hay problemas más grandes que hay que descubrir. [15]

Cliente On-Site: Es condición esencial la inclusión de al menos un Cliente real, vivo, como parte del Equipo. Debe estar disponible Full-Time para responder preguntas e interactuar con el resto del Equipo. [15]

Usar Estándares de Codificación: Los Desarrolladores escribirán todo el código de acuerdo a reglas predeterminadas que enfatizarán la comunicación a través del código. Estos estándares serán simples de seguir. [15]

La programación extrema cuenta con características que la hacen muy funcional a la hora de la realización de proyectos que necesitan ser desarrollados con rapidez, por lo que es de vital importancia su uso para el desarrollo de la aplicación que se plantea en esta investigación.

1.4.7.1.2 Microsoft Solution Framework (MSF).

Microsoft Solutions Framework (MSF) es una flexible e interrelacionada serie de conceptos, modelos y prácticas de uso que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. MSF se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas. Originalmente creado en 1994 para conseguir resolver los problemas a los que se enfrentaban las empresas en sus respectivos proyectos, se ha convertido posteriormente en un modelo práctico que facilita el éxito de los proyectos tecnológico. [16]

MSF se compone de varios modelos encargados de planificar las diferentes partes implicadas en el desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el modelo de Aplicación. [16]

Entre las características de MSF se encuentran: es una tecnología agnóstica, adaptable, escalable y flexible. A continuación se expone una breve explicación de ellas:

1. **Adaptable:** es parecido a un compás, usado en cualquier parte como un mapa, del cual su uso es limitado a un específico lugar.
2. **Escalable:** puede organizar equipos tan pequeños entre 3 o 4 personas, así como también, proyectos que requieren 50 personas a más.
3. **Flexible:** es utilizada en el ambiente de desarrollo de cualquier cliente.
4. **Tecnología Agnóstica:** porque puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.

Microsoft Solutions Framework es sin dudas una metodología muy eficiente, pero cuenta con una desventaja, ya que como su nombre indica es un software propietario, por lo que Cuba no la utiliza para el desarrollo de sus software.

1.4.7.2 Metodologías Tradicionales.

Al hablar sobre las Metodologías Tradicionales, no se puede hacerlo sin mencionar la rigurosa disciplina de trabajo que impone sobre el proceso de desarrollo del software, haciendo énfasis en la previa y total planificación del proyecto a desarrollar. El ciclo de desarrollo del software no comienza hasta que todo el trabajo a realizar esté bien detallado. Estas metodologías se desarrollan con el objetivo de conseguir un software más eficiente y predecible, por lo que tienen gran peso a la hora de desarrollar un producto de gran envergadura. Entre ellas se destaca RUP como la metodología más conocida dentro de esta clasificación, y sin lugar a dudas es muy utilizada a nivel mundial al igual que en la UCI.

1.4.7.2.1 Rational Unified Process (RUP).

El Proceso Unificado de Rational (RUP) es un proceso de desarrollo de software que captura las mejores prácticas del conocimiento de líderes en ingeniería de software y proporciona a los equipos de desarrollo guías, estándares y recomendaciones para la construcción de software de alta calidad. [17]

Las mejores prácticas de desarrollo de software están documentadas como principios clave, en donde cada una de ellas corresponde a distintos aspectos del desarrollo de software que generalmente requieren habilidades específicas, esto se refleja en los roles y productos de trabajo de cada uno de ellos. [17]

El ciclo de vida de RUP, como se conoce al trazado de las actividades de desarrollo en el tiempo, está dividido en 4 fases: inicial, elaboración, construcción y transición, que corresponden a los 4 hitos principales de RUP: proyecto, arquitectura, versión β y release. [17]

Cada fase cambia el foco del equipo de trabajo para alcanzar cada uno de los hitos y es llevada a cabo en forma iterativa. Esto quiere decir que la fase se fragmenta en pequeños proyectos que recorren todas las disciplinas y producen un ejecutable –en el sentido de software. Dicho producto es la forma más efectiva de verificar el progreso del proyecto y de reducir los riesgos inherentes. [17]

El proceso de software propuesto por RUP tiene tres características esenciales: está dirigido por los Casos de Uso, está centrado en la arquitectura, y es iterativo e incremental. Estas características junto con las mencionadas anteriormente, hacen que RUP sea una metodología a usar por excelencia en proyectos de alta envergadura.

1.4.8 Herramientas CASE de modelado con UML.

Las herramientas CASE, están tomando cada vez mas relevancia en la planeación y ejecución de proyectos que involucren sistemas de información, pues suelen inducir a los usuarios a la correcta utilización de metodologías que le ayudan a llegar con facilidad a los productos de software construidos. [18] Estas no son más que diversas aplicaciones informáticas que están destinadas a aumentar la productividad en el desarrollo de software, reduciendo el coste de las mismas en términos de tiempo y de dinero.

Todas las herramientas CASE prestan un soporte a un lenguaje de modelado para acompañar la metodología y es lógico suponer, que un alto porcentaje de ellas soporta UML, teniendo en

cuenta la aceptación de este lenguaje y el valor conceptual y visual que proporciona, y su facilidad para extender el lenguaje para representar elementos particulares a determinado tipo de aplicaciones. [18]

1.4.8.1 Visual Paradigm.

Visual Paradigm para UML es una herramienta CASE profesional que utiliza UML como lenguaje de modelaje, que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Se integra con las herramientas Java Eclipse/IBM WebSphere, Jbuilder, NetBeans, Oracle, JDeveloper y BEA Weblogic. Está disponible en varias ediciones, cada una destinada a unas necesidades: Enterprise, Professional, Community, Standard, Modeler y Personal. Soporta un conjunto de lenguas, tanto en la generación de código y de ingeniería inversa en Java, C + +, CORBA IDL, PHP, XML Schema, Ada y Python. Además, apoya la generación de código C #, VB. NET, Objeto Definition Language (ODL), Flash ActionScript, Delphi, Perl, Objective-C, y Ruby.

1.4.8.2 Rational Rose Enterprise Edition.

Rational Rose Enterprise Edition es el producto más completo de la familia Rational Rose. Es la mejor elección para el ambiente de modelado que soporte la generación de código a partir de modelos en Ada, ANSI C++, C++, CORBA, Java™/J2EE™, Visual C++ y Visual Basic. [20] Cubre todo el ciclo de vida de un proyecto desde la concepción y formalización del modelo, pasando por la construcción de los componentes, transición a los usuarios, hasta la certificación de las distintas fases y entregables.

Permite el diseño detallado del software y la generación de código fuente de programas y bases de datos e ingeniería inversa (obtención del diseño a partir del código fuente), basado en modelos con soporte UML. También permite establecer una trazabilidad real entre el modelo (análisis y diseño) y el código ejecutable. Además permite que hayan varias personas trabajando a la vez en el proceso iterativo controlado, para ello posibilita que cada desarrollador

opere en un espacio de trabajo privado que contiene el modelo completo y tenga un control exclusivo sobre la propagación de los cambios en ese espacio de trabajo.

Esta herramienta facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información (vista de Casos de Uso, vista Lógica, vista de Componentes y vista de Despliegue), pero utilizan un lenguaje común para comprender y comunicar la estructura y la funcionalidad del sistema en construcción, y propone la utilización de cuatro tipos de modelos para realizar el diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas, creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software.

A continuación se exponen algunas características de Rational Rose Enterprise Edition:

Soporte para análisis de patrones ANSI C++, Rose J y Visual C++ basado en "Design Patterns: Elements of Reusable Object-Oriented Software"

Característica de control por separado de componentes modelo que permite una administración más granular y el uso de modelos

Soporte de ingeniería Forward y/o reversa para algunos de los conceptos más comunes de Java 1.5

La generación de código Ada, ANSI C ++, C++, CORBA, Java y Visual Basic, con capacidad de sincronización modelo- código configurables

Soporte Enterprise Java Beans™ 2.0

Capacidad de análisis de calidad de código

El Add-In para modelado Web provee visualización, modelado y las herramientas para desarrollar aplicaciones de Web

Modelado UML para trabajar en diseños de base de datos, con capacidad de representar la integración de los datos y los requerimientos de aplicación a través de diseños lógicos y físicos

Capacidad de crear definiciones de tipo de documento XML (DTD) para el uso en la aplicación

Integración con otras herramientas de desarrollo de Rational

Capacidad para integrarse con cualquier sistema de control de versiones SCC-compliant, incluyendo a Rational ClearCase

Publicación web y generación de informes para optimizar la comunicación dentro del equipo. [20]

1.4.9 La Arquitectura de Software (AS).

La Arquitectura de Software podría definirse de forma general como el diseño de más alto nivel de la estructura de un sistema, programa o aplicación y tiene la responsabilidad de definir los módulos principales, definir las responsabilidades que tendrá cada uno de estos módulos, definir la interacción que existirá entre dichos módulos, control y flujo de datos, secuenciación de la información, protocolos de interacción y comunicación, ubicación en el hardware. Sin embargo la definición oficial de Arquitectura del Software es la *IEEE Std 1471-2000* que reza así: “La Arquitectura del Software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán, y los principios que orientan su diseño y evolución”. [21]

La AS aporta una visión abstracta de alto nivel, posponiendo el detalle de cada uno de los módulos definidos a pasos posteriores del diseño. [21]

1.4.9.1 Arquitectura n-Capas.

Hoy en día los modelos de computación experimentan un cambio radical, los sistemas monolíticos basados en mainframe y los tradicionales sistemas cliente-servidor, han migrado hacia sistemas distribuidos multiplataforma altamente modulables, este proceso ha surgido con el objetivo de solventar los problemas de escalabilidad, disponibilidad, seguridad e integración

que pueden presentan las aplicaciones compactas, para ello se ha generalizado la división de las aplicaciones en capas.

El modelo n-tier (n-capas) de informática distribuida ha emergido como la arquitectura predominante para la construcción de aplicaciones multiplataforma en la mayor parte de las empresas pertenecientes a Fortune 1000.

En esta arquitectura se agrega la capa que surge de separación definitiva de la regla de negocio de la de Acceso a Datos. Esto trae como ventaja que se aísla definitivamente la lógica de negocios de todo lo que tenga que ver con el origen de datos, de este modo ante cualquier cambio eventual, solo se deberá tocar un módulo específico, así como al momento de plantear la escalabilidad del sistema no se afrontarán grandes modificaciones.

1.4.9.1.1 Arquitectura en tres capas.

La Arquitectura en tres capas es un estilo de programación, se basa en la división en el nivel de acceso a datos, nivel de lógica de negocio y nivel de presentación o aplicación.

Capas de de la Arquitectura:

Capa de presentación.- Esta capa es la que ve el usuario, presenta el sistema al usuario, le comunica la información y captura la información del usuario en mínimo de proceso. Esta capa se comunica únicamente con la capa de un negocio. También es conocida como interfaz gráfica y debe tener la característica de ser \"amigable\" para el usuario generalmente se presentan como formularios. [22]

Capa de negocio.- Aquí es donde, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos para almacenar o recuperar datos de él. [22]

Capa de datos.- Es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. [22]

También esta arquitectura brinda algunas ventajas:

Centralización de los aspectos de seguridad y transaccionalidad, que serían responsabilidad del modelo. [23]

No replicación de lógica de negocio en los clientes: esto permite que las modificaciones y mejoras sean automáticamente aprovechadas por el conjunto de los usuarios, reduciendo los costes de mantenimiento. [23]

Mayor sencillez de los clientes. [23]

1.4.9.2 Patrón de arquitectura Modelo Vista Controlador (MVC).

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software. Los patrones expresan el esquema fundamental de organización para sistemas de software. Proveen un conjunto de subsistemas predefinidos; especifican sus responsabilidades e incluyen reglas y guías para organizar las relaciones entre ellos; así como ayudan a especificar la estructura fundamental de una aplicación. [24] Entre las características que definen a este, se destaca el hecho de separar los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

MVC divide una aplicación interactiva en 3 áreas: procesamiento, salida y entrada. Para esto, utiliza las siguientes abstracciones:

Modelo (Model): Encapsula los datos y las funcionalidades. El modelo es independiente de cualquier representación de salida y/o comportamiento de entrada. [24]

Vista (View): Muestra la información al usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador. [24]

Controlador (Controller): Reciben las entradas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, etc. Los eventos son traducidos a solicitudes de servicio ("**service requests**") para el modelo o la vista. [24]

Entre las ventajas que brinda este modelo de arquitectura se encuentra que:

Hay una clara separación entre los componentes de un programa; lo cual permite implementarlos por separado.

Hay un API muy bien definido; cualquiera que use el API, podrá reemplazar el Modelo, la Vista o el Controlador, sin aparente dificultad.

La conexión entre el Modelo y sus Vistas es dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.

10.4.10 Sistemas de Gestión de Contenidos

Un Sistema de gestión de contenidos consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio. Permite la creación y administración de contenidos principalmente en páginas Web, permitiendo además manejar de manera independiente el contenido por una parte y el diseño por otra, facilitando el control de publicaciones en el sitio a varios editores. Un ejemplo clásico es el de editores que cargan el contenido al sistema y otro de nivel superior que permite que estos contenidos sean visibles a todo público.

10.4.10.2 Drupal

Es un sistema de administración de contenido para sitios Web. Permite publicar imágenes, artículos, u otros archivos, así como servicios añadidos entre los que se encuentran los foros, encuestas, votaciones y administración de usuarios y permisos.

Drupal es un sistema dinámico: en lugar de almacenar sus contenidos en archivos estáticos en el sistema de ficheros del servidor de forma fija, el contenido textual de las páginas y otras

configuraciones son almacenados en una base de datos y se editan utilizando un entorno Web incluido en el producto. [8]

Drupal se compone de una infraestructura base y un conjunto de módulos que ofrecen un amplio conjunto de funciones, incluyendo sistemas de comercio electrónico, galerías de fotos y administración de listas de correo electrónico. Es posible añadir módulos de terceros para modificar el comportamiento de Drupal u ofrecer nuevas funciones. [8]

Drupal encuentra entre sus usos más frecuentes, el desempeño en intranets de compañías, enseñanza en línea, comunidades de arte, administración de proyectos. También es considerado por muchos como ideal para soportar comunidades de usuarios, por lo que es reconocido a nivel mundial.

1.5 Conclusiones.

En este capítulo se realizó un estudio de las tecnologías y tendencias actuales en cuanto al desarrollo de aplicaciones web. Se analizaron las principales herramientas, lenguajes de programación y metodologías que han alcanzado un gran auge hoy en día. También se tuvo en cuenta el conocimiento previo sobre estas tecnologías.

Debido a que la universidad al igual que Cuba persigue el concepto de “independencia tecnológica”, las tecnologías que fueron seleccionadas se identifican por ser en su mayoría software libre y multiplataforma. De ahí que se decidió seleccionar para el desarrollo y construcción del sistema, como lenguaje del lado del servidor, PHP, ya que se tiene un poco más de experiencia en su uso comparado con el resto de los mencionados en el capítulo. Para su selección se tuvo en cuenta además el hecho de ser “open source” y que puede ser utilizado en cualquiera de los principales sistemas operativos. También posee una de las comunidades más grandes de desarrolladores en el mundo, incluyendo la UCI, así como un código simple que facilita su trabajo, y mucha documentación y una gran librería de funciones. Además PHP cuenta con la facilidad para funcionar con el servidor web Apache, ya que se integra como un módulo de este, justificando así la selección del mismo, ambos poseen gran compatibilidad y por su parte Apache es también gratuito, multiplataforma y presenta gran modularidad que lo hace personalizable a la vez que configurable y flexible. También fueron seleccionados como

lenguajes del lado del cliente, HTML y CSS, así como PostgreSQL fue el seleccionado como gestor de base de datos ya que cumple con las necesidades planteadas.

Para confeccionar la implementación de la aplicación, se tuvo en cuenta el IDE Zend Studio uno de los mayores impulsores de PHP, orientada a desarrollar aplicaciones web, como no, en PHP. Se seleccionó como metodología de desarrollo del software Extreme Programing (XP), debido que se adapta perfectamente a nuestras necesidades. UML se utiliza para el modelado del sistema, ya que constituye un estándar de facto a nivel internacional para especificar, visualizar, construir y documentar artefactos de un software. Además es el lenguaje para describir principalmente sistemas OO. También fue elegida Rational Rose, que es una herramienta CASE para el modelado con UML. Esta es la herramienta con la que se aprende a trabajar en la asignatura Ingeniera de Software, insertada en el proceso docente en la universidad. Debido a las posibilidades que brinda el sistema de gestión de contenidos Drupal, fue seleccionado como plataforma de publicación para la creación de la aplicación.

Capitulo 2: Exploración y Planificación.

2.1 Introducción.

En el presente capítulo se exponen las fases de exploración y planificación que corresponden a la metodología de desarrollo utilizada para llevar a cabo la implementación y desarrollo del sistema propuesto. Además se exhiben los artefactos que se generan en el transcurso de dichas faces.

2.2 Fase de exploración.

La metodología de desarrollo Extreme Programming comienza con la fase de exploración. Durante este período 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 sistema.

2.2.1 Historias de Usuario.

La historia de usuario es la forma de en que se especifican los requisitos del sistema. El cliente es el encargado de redactar estas aunque los desarrolladores pueden brindar su ayuda en la identificación de las mismas. Estas deben abarcar el contenido lo más concreto y sencillo posible. Durante este proceso se identifican 6 historias de usuario que se detallan a continuación.

<table>
<tr><td colspan="3">Historia de Usuario</td></tr>
<tr><td>Número: 1</td><td colspan="2">Usuario: Usuario</td></tr>
<tr><td colspan="3">Nombre historia: Autenticar usuario.</td></tr>
<tr><td colspan="2">Prioridad en negocio: Alta</td><td>Riesgo en desarrollo: Baja</td></tr>
</table>

Puntos estimados: 1	Iteración asignada: 1
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: El usuario accede a la aplicación y se autentica.	
Observaciones:	

Tabla 1: Autenticar usuario.

Historia de Usuario	
Número: 2	**Usuario:** Usuario
Nombre historia: Insertar pregunta.	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 2
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: El usuario, una vez dentro del sistema, puede insertar una pregunta. Esta inquietud es clasificada por temas. El usuario debe seleccionar la opción insertar pregunta. Luego escribe la pregunta, clasifica la pregunta y la envía.	
Observaciones:	

Tabla 2: Insertar pregunta.

Historia de Usuario	
Número: 3	**Usuario:** Usuario
Nombre historia: Ver respuesta sobre preguntas del usuario.	
Prioridad en negocio: Baja	Riesgo en desarrollo: Baja
Puntos estimados: 1	Iteración asignada: 3
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Una vez dentro del sistema el usuario debe seleccionar la opción ver mis respuestas, revisa si tiene alguna respuesta, y en caso de existir alguna respuesta, el usuario asigna una puntuación a la respuesta.	
Observaciones:	

Tabla 3: Ver respuesta sobre preguntas del usuario.

Historia de Usuario	
Número: 4	**Usuario:** Usuario
Nombre historia: Responder pregunta.	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 1	Iteración asignada: 3

Programador responsable: Raimundo Adrian Acosta Medina
Descripción: Una vez dentro del sistema el usuario debe seleccionar la opción buscar preguntas por tema. Luego debe seleccionar el tema, aparecen las preguntas y él debe seleccionar la pregunta a responder, responde y envía la respuesta. El sistema se encarga de enviar un correo de aviso al usuario que haya publicado dicha pregunta.
Observaciones:

Tabla 4: Responder pregunta.

Historia de Usuario	
Número: 5	**Usuario:** Usuario
Nombre historia: Navegación por categorías.	
Prioridad en negocio: Alta	Riesgo en desarrollo: Media
Puntos estimados: 1	Iteración asignada: 2
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Una vez dentro del sistema el usuario podrá seleccionar la categoría deseada y navegar por esta.	
Observaciones:	

Tabla 5: Navegación por categorías.

Historia de Usuario	
Número: 6	**Usuario:** Usuario
Nombre historia: Buscar usuarios.	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 1	Iteración asignada: 1
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Una vez dentro del sistema el usuario debe seleccionar la opción buscar usuario, la cual debe permitirle buscar información referente al perfil del usuario seleccionado.	
Observaciones:	

Tabla 6: Buscar usuarios.

Historia de Usuario	
Número: 7	**Usuario:** Usuario
Nombre historia: Buscar respuesta.	
Prioridad en negocio: Baja	Riesgo en desarrollo: Bajo
Puntos estimados: 1	Iteración asignada: 3

Programador responsable: Raimundo Adrian Acosta Medina
Descripción: Una vez dentro del sistema el usuario debe seleccionar la opción buscar pregunta, revisa si tiene alguna respuesta, y en caso de existir alguna respuesta, el usuario asigna una puntuación a la respuesta.
Observaciones:

Tabla 7: Buscar usuarios.

Historia de Usuario	
Número: 8	**Usuario:** Usuario
Nombre historia: Usuario experto	
Prioridad en negocio: Media	Riesgo en desarrollo: Bajo
Puntos estimados: 1	Iteración asignada: 1
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Una vez dentro del sistema y estando previamente logueado, al usuario debe brindársele la posibilidad de convertirse en usuario experto. Dichos usuarios son los que están autorizados a responder alguna pregunta.	
Observaciones:	

Tabla 8: Usuario Experto.

Historia de Usuario	
Número: 9	**Usuario:** Administrador
Nombre historia: Gestión de categorías.	
Prioridad en negocio: Media	Riesgo en desarrollo: Bajo
Puntos estimados: 1	Iteración asignada: 1
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: El administrador debe tener la posibilidad de insertar, editar o eliminar una categoría.	
Observaciones:	

Tabla 9: Gestión de categorías.

Historia de Usuario	
Número: 10	**Usuario:** Administrador
Nombre historia: Ranking de expertos.	
Prioridad en negocio: Baja	Riesgo en desarrollo: Bajo
Puntos estimados: 1	Iteración asignada: 3
Programador responsable: Raimundo Adrian Acosta Medina	

Descripción: El sistema de mostrar los expertos existentes ordenados según la puntuación que estos tengan asignada.
Observaciones:

Tabla 10: Ranking de expertos.

2.3 Fase de Planificación.

Durante esta fase se estima el esfuerzo que costará realizar la implementación de cada historia de usuario. El esfuerzo se representa usando como medida el punto. El punto se considera como una semana de ideal de trabajo. Esta estimación incluye todo el esfuerzo asociado a la implementación de la historia de usuario, por ejemplo, las pruebas unitarias, la integración, la refactorización del código y la preparación y ejecución de las pruebas de aceptación.

2.3.1 Estimación de esfuerzo por Historia de Usuario.

Con el objetivo de lograr un desarrollo eficiente del sistema propuesto, se realizó una estimación de esfuerzo para cada una de las historias de usuario. A continuación se muestra esta estimación.

Historias de Usuario	Puntos de Estimación.
Autenticar usuario.	1
Insertar pregunta.	1
Ver respuesta sobre preguntas del usuario.	1
Responder pregunta.	1

Navegación por categorías.	1
Buscar usuarios.	1
Buscar respuestas.	1
Usuario Experto.	1
Gestión de categorías.	1
Ranking de expertos.	1

Tabla 11: Estimación de esfuerzos por HU.

2.3.2 Plan de iteraciones.

Después ser descritas e identificadas las historias de usuarios, así como estimado el esfuerzo para la realización de cada una de ellas, se procede a la planificación de la etapa de implementación del sistema. Este plan especifica exactamente cuales son las historias de usuarios que serán implementadas para cada iteración y las posibles fechas de las liberaciones.

En base a lo antes mencionado, se decide realizar 3 iteraciones, las cuales están detalladas a continuación:

Iteración 1

Esta iteración tiene como objetivo la implementación de las historias de usuario 1, 6 y 8, las cuales hacen alusión a autenticación de los usuarios en el sistema, la búsqueda de otros

usuarios, así como a la posibilidad que tienen los mismos de convertirse en usuarios expertos. Estas historias de usuario han sido seleccionadas para ser implementadas en esta primera iteración después de llegar a un consenso con el usuario según la prioridad dada por el mismo.

Durante la presente iteración se desarrollo el modulo instalación, el mismo no se encuentra descrito en ninguna historia de usuario debido a que tiene como funcionalidad, la instalación de las tablas necesarias para el manejo de la información a tratar en el sistema. Es necesario aclarar que el mismo no tiene relevancia como para tratar como un modulo de implementación a tratar en las siguientes fases de desarrollo de la documentación en curso ya que es de muy fácil, rápida y sencilla implementación, además que no maneja información trascendente.

Iteración 2

Esta iteración tiene como objetivo la implementación de las historias de usuario 2, 9 y 5, las cuales hacen referencia a la posibilidad de insertar inquietudes, realizar la navegación por las diferentes categorías así como la gestión de las categorías por parte del administrador del sistema. Con esta iteración se tendrá la segunda versión del sistema propuesto, en la cual deben estar implementados los cambios decididos por el cliente junto con el grupo de desarrollo, planteados al finalizar la iteración 1.

Iteración 3

Esta iteración tiene como objetivo la implementación de las historias de usuario 3, 4, 7 y 10, las cuales hacen alusión a la inserción de respuestas y a la búsqueda de respuestas a preguntas puestas por el usuario así como cualquier otro, también permite ver el ranking de los expertos según la puntuación alcanzada por los mismos. Al finalizar la implementación de esta iteración el sistema será mostrado al cliente el cual propondrá cambios de ser necesario y en caso de ser aprobada, el sistema se pondrá a prueba por un tiempo para poder evaluar el servicio del mismo.

2.3.3 Plan de duración de las iteraciones.

Todo proyecto que tiene como guía el uso de la metodología XP, crea como parte del ciclo de vida el plan de duración de cada una de las iteraciones, en este caso se hace para el único equipo de desarrollo con el cual se cuenta. Este plan tiene como finalidad mostrar la duración de cada iteración, así como el orden en que serán implementadas las historias de usuario en cada una de las mismas.

No Iteración	Historias de Usuario	Duración total de la Iteración
Iteración 1	Autenticar usuario. Buscar usuario. Usuario experto	3 Semanas.
Iteración 2	Insertar pregunta. Gestión de categorías. Navegación por categorías.	3 Semanas.
Iteración 3	Ver respuesta sobre preguntas del usuario. Buscar repuesta. Responder pregunta. Ranking de expertos	4 Semanas.

Tabla 12: Plan de duración de iteraciones.

2.3.4 Plan de entregas.

A continuación se presenta el plan de entregas elaborado para la fase de implementación. El mismo fue elaborado teniendo en cuenta el número de iteraciones necesarias para el desarrollo del sistema y con el objetivo de facilitar la implementación de la aplicación.

Modulo	Historia de Usuario que le corresponde
User	1
Categoría	5, 9
Preguntas	2, 3, 4
Experto	8
Buscar	7, 6

Tabla 13: Módulos y HU abarcadas.

No. de iteración	Final de Iteración 1 3ra semana de abril	Final de Iteración 2 4ra semana de abril	Final de Iteración 2 3ra semana de mayo
user	0.2	Finalizado	Finalizado
categoría		1.0	Finalizado
preguntas		0.6	2.0
experto	0.4	Finalizado	Finalizado

buscar	0.6	Finalizado	Finalizado

Tabla 14: Cronograma de entrega.

2.4 Conclusiones

Durante el capítulo se abordó todo lo relacionado a las fases de exploración y planeación de la metodología XP, que fue la seleccionada para guiar el desarrollo de la aplicación.

Capitulo 3: Diseño, implementación y pruebas.

3.1 Introducción.

En el presente capítulo se abordará toda la temática referente al diseño, implementación y prueba del sistema. La metodología XP sugiere entre sus prácticas la necesidad de conseguir diseños simples y sencillos, por lo que estos aspectos deben tenerse en cuenta. Uno de los artefactos fundamentales es la creación de las tarjetas CRC (Clase-Responsabilidades-Colaboración) las cuales permiten brindar un mayor enfoque orientado a objetos. Por otro lado se describen las tareas planificadas para llevar a cabo el desarrollo de cada una de las historias de usuario detectadas. Además se realiza el diseño de la base de datos, que es de suma importancia producto de que dará soporte a la futura aplicación.

3.2 Diseño

A continuación se realiza una breve explicación del funcionamiento de Drupal como plataforma de publicación, con el objetivo de lograr una mejor comprensión del trabajo en curso.

Drupal provee al desarrollador de un potente sistema de seguridad basado en roles, el mismo Sistema de Gestión de Contenidos se encarga de la creación de usuarios y roles, así como del control de accesos a los diferentes módulos según los permisos definidos por el administrador.

El CMS Drupal contiene un tipo de contenido genérico llamado *Node* que puede ser extendido por cualquier desarrollador, este tipo de contenido tiene las propiedades básicas para cualquier publicación como son título, autor, fecha de creación y contenido, además Drupal proporciona los mecanismos para la creación, edición y publicación de este tipo de contenido. Cualquier desarrollador que desee una publicación personalizada sólo debe extender este tipo de contenido y de esta manera aprovechar sus propiedades.

3.2.1 Módulos de Drupal

Un módulo es la unión de varias funciones que se juntan en Drupal y ayudan a ofrecerle mayor funcionalidad a la Web. Un módulo para Drupal consta de uno o más ficheros, el fichero principal con extensión *.module* debe implementar una interfaz definida por el propio Drupal.

Existen dos tipos de módulos fundamentales: los módulos de contenido, que definen un nuevo tipo de contenido personalizado y la funcionalidad para su creación, edición y publicación y los módulos funcionales, que tienen disímiles propósitos dependiendo del objetivo con el que se desarrollen. La tarea de estas funciones es actuar como enganche, al ser llamadas por Drupal a la hora de construir una página Web y gestionar el contenido.

Drupal a su vez cuenta con un sistemas de bloques**,** este consiste en una serie de bloques (ya sea definidos por un módulo o en la misma interfaz de Drupal) que se activan o desactivan para ser mostrados en las áreas de menú de la plataforma.

El Sistema de Gestión de Contenidos Drupal como la mayoría de estos, cuenta con una estructura modular, que tiene como módulos básicos: *Theme*, *Module* e *Includes*. Casi todos los módulos tienen relación con algunos componentes, que constituyen capas intermedias; como pueden ser componentes de acceso a la base de datos y de lógica de negocio. En el módulo *Theme* se encuentran los mecanismos que soportan el sistema de plantillas, de modo que cuando se desee cambiar el diseño de la interfaz que presentará el sistema, se debe definir una nueva plantilla en este archivo, al mismo tiempo es posible crear una plantilla nueva y añadirla al módulo; en el módulo *Includes*, donde se encuentran ficheros de configuración y ficheros utilitarios, es este paquete donde se incluyen las API de acceso a datos; y por último el módulo *Modules*, provee a Drupal de sus funcionalidades, de forma tal que cuando se desee agregar un nuevo módulo, sólo debe copiarse dentro de esta carpeta y activarse a través de la interfaz de Drupal.

Es válido mencionar que Drupal contiene además de su sistema de módulos, una única página de servidor, la cual se basa en el sistema de clases y genera el contenido de la página final, teniendo en cuenta los argumentos con que se realiza la petición. Las páginas generadas pueden o no contener formularios, esto depende del módulo en cuestión y del propósito del mismo.

A continuación se modelará cada módulo por separado, debido a que son independientes uno del otro en cuanto a funcionalidad. Luego de haber mencionado los principales aspectos que

influyen en el diseño del diagrama de clases Web que representa el funcionamiento de Drupal, el mismo se muestra en la siguiente figura:

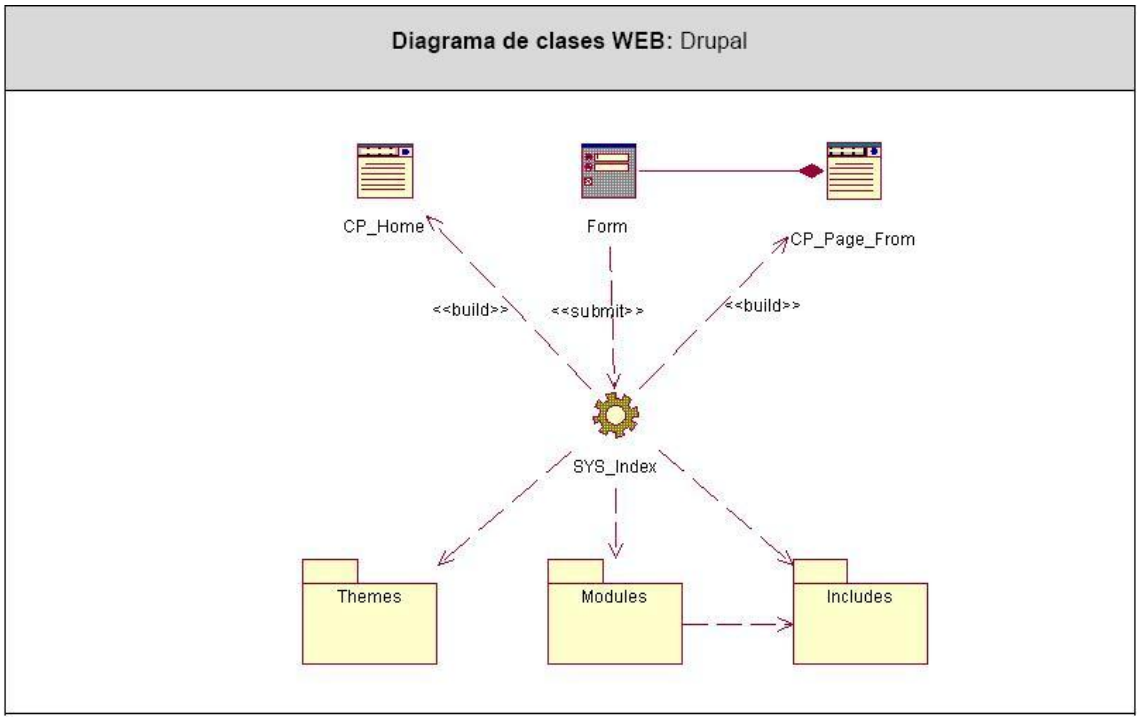

Figura 2: Diagrama de clases del diseño: Drupal.

Los módulos que dan soporte a las funcionalidades de Drupal, se encuentran dentro del módulo *Module,* y junto con estos los módulos desarrollados en este trabajo.

A continuación se define una tarjeta CRC por cada módulo desarrollado en este trabajo, con el objetivo de hacer descifrables funcionalidades encapsuladas en cada uno, así como para obtener un diseño simple y no incurrir en la implementación de características que no son necesarias. Véase que en este caso las tarjetas CRC cambian su estructura, en este caso se muestran con la siguiente distribución *(Módulo – Funcionalidades- Colaboraciones (Módulos))*.

Módulo Instalación

Funcionalidades	Colaboraciones(Módulos)
Instalación de las tablas en la bases de datos.	Categoría Preguntas Experto Buscar

Tabla 15: Tarjeta CRC Módulo Instalación.

Módulo Categoría	
Funcionalidades	**Colaboraciones(Módulos)**
Insertar categoría Modificar categoría Eliminar categoría Navegación por categorías	User Block Node Menu Preguntas Experto Buscar

Tabla 16: Tarjeta CRC Módulo Categoría.

Módulo Experto	
Funcionalidades	**Colaboraciones(Módulos)**
Convertir un usuario en experto.	User Block Node Menu Preguntas Categoría Buscar

Tabla 17: Tarjeta CRC Módulo Experto.

Módulo Preguntas	
Funcionalidades	**Colaboraciones(Módulos)**
Realizar pregunta Responder pregunta Posicionar el ranking de experto	User Block Node

Ver los diferentes expertos Gestionar el perfil personal de los usuarios	Menu Experto Categoría Buscar

Tabla 18: Tarjeta CRC Módulo Preguntas.

Módulo Buscar	
Funcionalidades	**Colaboraciones(Módulos)**
Buscar usuarios expertos. Buscar respuestas.	User Block Node Menu Preguntas Categoría

Tabla 19: Tarjeta CRC Módulo Buscar.

3.2.2 DISEÑO DE LA BASE DE DATOS

El diseño de la Base de Datos tiene como objetivo fundamental brindar la persistencia al modelo que se describe en el epígrafe anteriormente desarrollado, por esta razón es vital para el desarrollo del software en cuestión.

Debido a que las entidades son manejadas por el CMS Drupal (por lo que no han sido contempladas en el modelo de datos), el modelo de datos del problema en cuestión posee un nivel de complejidad bajo. A continuación se muestra el modelo de datos que se utilizó:

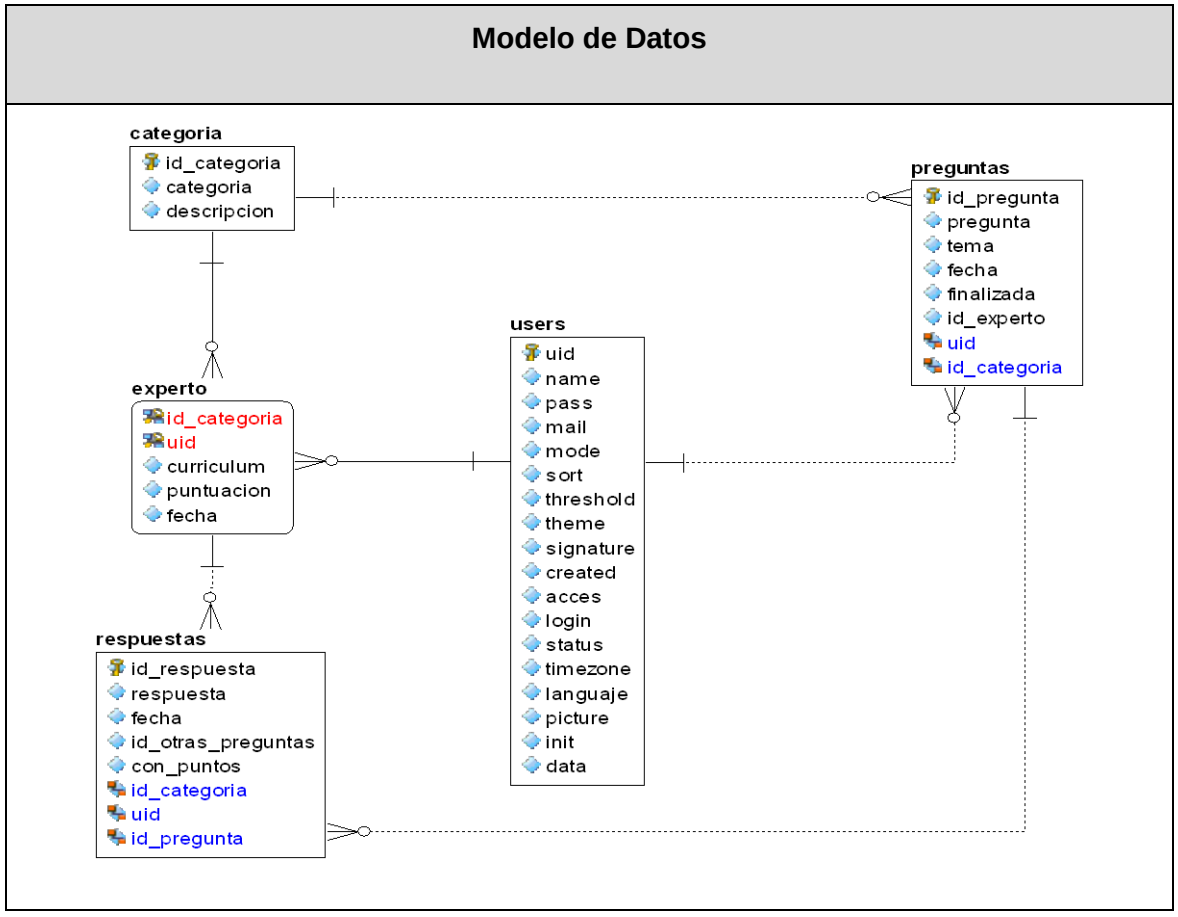

Figura 3: Modelo de Datos.

Implementación

A medida que van desarrollándose las iteraciones se realiza la implementación de las historias de usuario seleccionadas para ser realizadas en cada una de ellas. Al principio de estas se lleva a cabo una revisión del plan de iteraciones y se modifica en caso de ser necesario. Como parte de este plan, se descomponen las HU en tareas de desarrollo, asignando a un grupo de desarrollo (o una persona), responsable de su implementación. Estas tareas son para el uso

estricto de los programadores, pueden ser escritas en lenguaje técnico y no necesariamente entendible por el cliente.

Para llevar a cabo el desarrollo del software, durante la planificación, se desarrollaron tres iteraciones, obteniéndose como finalidad un producto con todas las restricciones y características deseadas para ser utilizado. A continuación se detalla cada una de las iteraciones.

3.3.1 Iteración 1

En esta iteración se implementaron las historias de usuario de mayor prioridad (1, 6 y 8), con el fin de obtener una versión del producto con algunas de las funcionalidades críticas para ser mostrado al cliente y tomar nuevas iniciativas de forma rápida.

Módulos	Historias de usuario	Tiempo de Implementación (puntos)	
		Estimación	Real
Experto.	Usuario experto.	1	0.2
Buscar.	Buscar usuario.	1	0.3
Users.	Autenticar usuario	1	0.1

Tabla 20: Módulos abordados en la primera iteración.

A continuación se muestran las tareas efectuadas para cada uno de los módulos implementados en esta iteración.

3.3.1.1 Módulo experto.

Tarea	
Número tarea: 1	**Número historia:** 8
Nombre tarea: Gestionar usuario experto.	
Tipo de tarea : Desarrollo	**Puntos estimados:** 0.50
Fecha inicio: 18/4/2009	**Fecha fin:** 19/42009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se creará la posibilidad de convertir un usuario logueado en experto. Se gestionará la verificación de los datos introducidos con los almacenados en la Base de Datos. Los datos serán almacenados en la Base de Datos de forma persistente.	

Tabla 21: Tarea 1 del modulo experto.

3.3.1.2 Módulo buscar.

Tarea	
Número tarea: 2	**Número historia:** 6
Nombre tarea: Buscar usuario.	
Tipo de tarea : Desarrollo	**Puntos estimados:** 1

Fecha inicio: 19/4/2009	**Fecha fin:** 20/4/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se creará la posibilidad de búsqueda de usuarios. Se gestionará la verificación de los datos introducidos con los almacenados en la Base de Datos.	

Tabla 22: Tarea 1 del modulo preguntas

3.3.2 Iteración 2

Durante el transcurso de la presente iteración se concluyó la implementación y configuración de las funcionalidades del módulo experto aunque se continúa en su perfeccionamiento y se le da continuación a la implementación del módulo preguntas, que tiene a cargo el desarrollo de las historias de usuario 2, 3, 4, 6 y 7, de las cuales de desarrollaron en esta iteración la 2, la 9 y la 5 pertenecientes al módulo categoría.

Módulos	**Historias de usuario**	**Tiempo de Implementación (puntos)**	
		Estimación	**Real**
Preguntas.	Insertar pregunta.	1	0.2
Categoría.	Navegación por categorías.	1	0.2
	Gestión de categorías.	1	0.6

Tabla 23: Módulos abordados en la segunda iteración.

3.3.2.1 Módulo preguntas

Tarea	
Número tarea: 3	**Número historia:** 2
Nombre tarea: Insertar pregunta.	
Tipo de tarea : Desarrollo	**Puntos estimados:** 1
Fecha inicio: 21/4/2009	**Fecha fin:** 22/4/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se insertará una pregunta. Se gestionará la verificación de los datos introducidos con los almacenados en la Base de Datos. Los datos serán almacenados en la Base de Datos de forma persistente.	

Tabla 24: Tarea 2 del modulo preguntas

3.3.2.2 Módulo categoría.

Tarea

Número tarea: 4	**Número historia:** 9
Nombre tarea: Crear categorías.	
Tipo de tarea : Desarrollo	**Puntos estimados:** 0.50
Fecha inicio: 22/4/2009	**Fecha fin:** 23/4/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se crearán las categorías de navegación sobre las cuales se plantearán las preguntas.	

Tabla 25: Tarea 1 del modulo categoría.

Tarea	
Número tarea: 5	**Número historia:** 5
Nombre tarea: Verificar la navegación por las categorías	
Tipo de tarea : Desarrollo	**Puntos estimados:** 0.50
Fecha inicio: 23/4/2009	**Fecha fin:** 23/4/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se navegará por las diferentes categorías creadas, verificando así los	

problemas de navegabilidad.

Tabla 26: Tarea 2 del modulo categoría.

Tarea	
Número tarea: 6	**Número historia:** 9
Nombre tarea: Eliminar categoría	
Tipo de tarea : Desarrollo	**Puntos estimados:** 0.50
Fecha inicio: 24/4/2009	**Fecha fin:** 24/4/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se eliminarán las categorías de navegación que el administrador entienda que ya no cumplen ninguna funcionalidad.	

Tabla 27: Tarea 3 del módulo categoría.

Tarea	
Número tarea: 7	**Número historia:** 9
Nombre tarea: Modificar categoría	

Tipo de tarea : Desarrollo	**Puntos estimados:** 0.50
Fecha inicio: 25/4/2009	**Fecha fin:** 26/4/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se modificarán las categorías de navegación a consideración del administrador.	

Tabla 28: Tarea 4 del modulo categoría.

3.3.3 Iteración 3

En esta iteración se implementaron las funcionalidades de baja prioridad. En este caso las iteraciones 3, 6 y 7, que tienen como fin la introducción y la búsqueda de respuestas. Al finalizar se cuenta con un producto listo para poner en funcionamiento.

Módulos	**Historias de usuario**	**Tiempo de Implementación (puntos)**	
		Estimación	**Real**
Preguntas.	Buscar respuesta.	1	1
Categoría.	Responder pregunta.	1	1
Buscar.	Ver respuesta sobre preguntas del usuario.	1	1
		1	1

Tabla 29: Módulos abordados en la tercera iteración.

3.3.3.1 Módulo preguntas

Tarea	
Número tarea: 8	**Número historia:** 3
Nombre tarea: Ver respuestas sobre preguntas de un usuario	
Tipo de tarea : Desarrollo	Puntos estimados: 1
Fecha inicio: 28/4/2009	**Fecha fin:** 4/5/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se creará la posibilidad al usuario de ver las respuestas a sus preguntas, existiendo con anterioridad una pregunta hecha por el usuario.	

Tabla 30: Tarea 4 del módulo preguntas

Tarea	
Número tarea: 9	**Número historia:** 7
Nombre tarea: Buscar una respuesta	

Tipo de tarea : Desarrollo	**Puntos estimados:** 1
Fecha inicio: 5/5/2009	**Fecha fin:** 11/5/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se creará la posibilidad al usuario de buscar una respuesta. Se gestionará la verificación de los datos introducidos con los almacenados en la Base de Datos.	

Tabla 31: Tarea 5 del módulo preguntas.

Tarea	
Número tarea: 10	**Número historia:** 4
Nombre tarea: Insertar respuesta.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 1
Fecha inicio: 12/5/2009	**Fecha fin:** 18/5/2009
Programador responsable: Raimundo Adrian Acosta Medina	
Descripción: Se insertará una respuesta asociada a una pregunta. Se gestionará la verificación de los datos introducidos con los almacenados en la Base de Datos. Los datos serán almacenados en la Base de Datos de forma persistente.	

Tabla 32: Tarea 6 del módulo preguntas.

3.3.4 Diagrama de Componentes

Un diagrama de componentes muestra las dependencias entre estos componentes y representa la separación del sistema de software en componentes físicos (por ejemplo archivos, módulos, paquetes, etc.). Se usa con el objetivo de modelar la vista estática de un sistema, además de mostrar la organización y las dependencias entre un conjunto de componentes. Entre sus principales usos se encuentra que puede servir para ver qué componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.

Con el propósito de brindar un mejor entendimiento del sistema se muestra el diagrama de componentes del sistema.

Diagrama de componentes del sistema

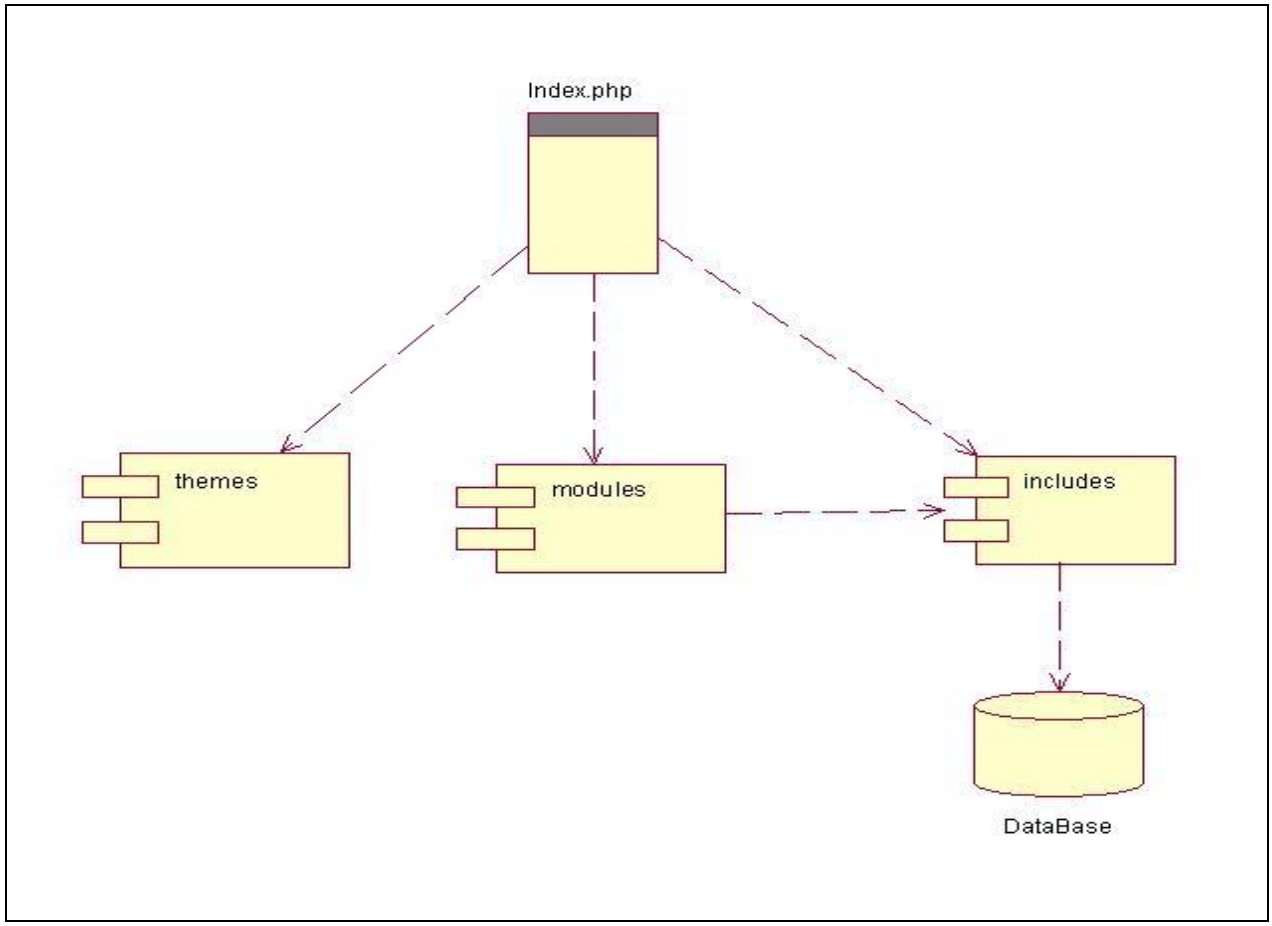

Figura 4: Diagrama de componentes del sistema.

3.3.5 Diagrama de Despliegue

El diagrama de despliegue permite apreciar de forma visual cómo se encuentran relacionados físicamente los componentes de la aplicación. En este caso la aplicación se encuentra hospedada en un servidor Web y la misma se comunica con un sistema de gestión de base de datos (MySQL).

Diagrama de Despliegue

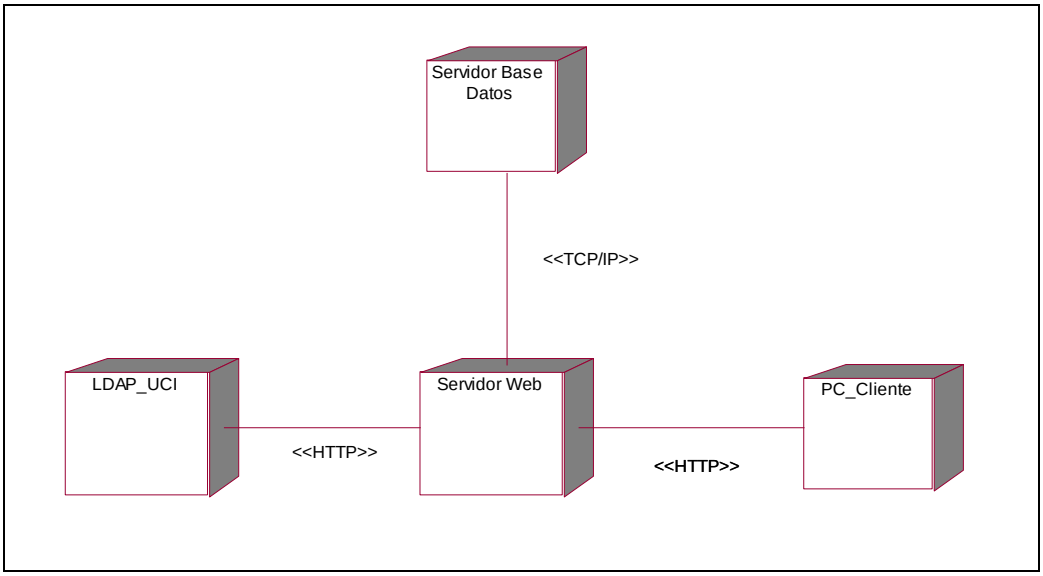

Figura 5: Diagrama de Despliegue.

Pruebas.

Dentro de los pilares fundamentales de la metodología XP, se encuentra la realización de pruebas al sistema. Esta filosofía ayuda a reducir el número de errores no detectados, así como el tiempo entre la introducción del error en el sistema y su detección.

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. [25]

Pruebas de aceptación.

Las pruebas de aceptación son pruebas de caja negra que se crean a partir de las historias de usuario. Durante el transcurso de las iteraciones, las HU seleccionadas serán traducidas a pruebas de aceptación. En ellas se especifican los escenarios para probar que una HU ha sido implementada correctamente, siempre desde la perspectiva del cliente. Una HU no se

considera completa hasta que no ha pasado por sus pruebas de aceptación, siempre teniendo en cuenta que una HU puede tener tantas pruebas de aceptación como necesite.

A continuación se detallan las diferentes pruebas de aceptación realizadas al sistema, para ello el cliente llena la siguiente planilla:

Prueba de aceptación
HU: Nombre de la historia de usuario que va a comprobar su funcionamiento.
Nombre: Nombre del caso de prueba.
Descripción: Descripción del propósito de la prueba.
Condiciones de ejecución: Precondiciones para que la prueba se pueda realizar.
Entrada/Pasos de ejecución: Pasos para probar la funcionalidad.
Resultado esperado: Resultado que se desea de la prueba.
Evaluación de la prueba: *Aceptada o Denegada.*

Tabla 33: Planilla prueba de aceptación.

Prueba de aceptación
HU: Autenticar usuario.

Nombre: Autenticar usuario.
Descripción: El usuario intenta autenticarse.
Condiciones de ejecución: Los datos introducidos por el usuario deben ser válidos.
Entrada/Pasos de ejecución: El usuario intenta autenticarse introduciendo los datos.
Resultado esperado: El usuario se encuentra navegando dentro del sistema.
Evaluación de la prueba: Aceptada.

Tabla 34: Autenticar usuario.

Prueba de aceptación
HU: Gestión de categorías.
Nombre: Adicionar categoría.
Descripción: El administrador del sistema puede adicionar una categoría.
Condiciones de ejecución: El usuario debe estar logueado en el sistema, siendo este, administrador.
Entrada/Pasos de ejecución: Se intenta adicionar una categoría al sistema.
Resultado esperado: Se adiciona la categoría y se brinda la posibilidad de adicionar otra.

Evaluación de la prueba: Aceptada.

Tabla 35: Adicionar categoría.

Prueba de aceptación
HU: Gestión de categorías.
Nombre: Modificar categoría.
Descripción: El administrador del sistema puede modificar una categoría.
Condiciones de ejecución: El usuario debe estar logueado en el sistema siendo administrador, debe existir al menos una categoría.
Entrada/Pasos de ejecución: Se intenta modificar una categoría al sistema.
Resultado esperado: Se modifica la categoría y se muestra el panel de categorías con los cambios realizados en la misma.
Evaluación de la prueba: Aceptada.

Tabla 36: Modificar categoría.

Prueba de aceptación
HU: Gestión de categorías.

Nombre: Eliminar categoría.
Descripción: El administrador del sistema puede eliminar una categoría.
Condiciones de ejecución: El usuario debe estar logueado en el sistema siendo administrador, debe existir al menos una categoría.
Entrada/Pasos de ejecución: Se intenta eliminar una categoría al sistema.
Resultado esperado: Se elimina la categoría y se muestra el panel de categorías sin la existencia de la misma.
Evaluación de la prueba: Aceptada.

Tabla 37: Eliminar categoría.

Prueba de aceptación
HU: Navegación por categorías.
Nombre: Navegar por categorías.
Descripción: Cualquier usuario del sistema o un espectador del mismo, entiéndase como espectador un usuario anónimo, puede seleccionar cualquier categoría y acceder a los datos de la misma.
Condiciones de ejecución: Debe existir al menos una categoría.
Entrada/Pasos de ejecución: Se intenta seleccionar una categoría para acceder a sus

datos.
Resultado esperado: Luego de seleccionada la categoría se muestran los datos de la misma.
Evaluación de la prueba: Aceptada.

Tabla 38: Navegar por categorías.

Prueba de aceptación
HU: Buscar usuarios.
Nombre: Buscar usuario.
Descripción: Se brinda la posibilidad de buscar los datos de un usuario existente en el sistema.
Condiciones de ejecución: Debe existir al menos un usuario, los datos introducidos para su búsqueda deben ser válidos.
Entrada/Pasos de ejecución: Se introducen los datos para la búsqueda, y se acepta la opción de buscar.
Resultado esperado: Se muestran los datos del usuario.
Evaluación de la prueba: Aceptada.

Tabla 39: Buscar usuario.

Prueba de aceptación
HU: Buscar respuesta.
Nombre: Buscar respuesta.
Descripción: Se brinda la posibilidad de buscar una respuesta existente en el sistema.
Condiciones de ejecución: Los datos introducidos para su búsqueda deben ser válidos.
Entrada/Pasos de ejecución: Se introducen los datos para la búsqueda, y se acepta la opción de buscar.
Resultado esperado: Se muestran las respuestas existentes que cumplen con el criterio de búsqueda.
Evaluación de la prueba: Aceptada.

Tabla 40: Buscar respuesta.

Prueba de aceptación
HU: Ver respuesta sobre preguntas del usuario.
Nombre: Ver respuestas realizadas a preguntas del usuario.
Descripción: Se brinda la posibilidad de ver las respuestas que han sido realizadas a preguntas hechas por el usuario.

Condiciones de ejecución: El usuario debe estar logueado en el sistema.
Entrada/Pasos de ejecución: El usuario intenta ver las respuestas que se le han realizado.
Resultado esperado: Se muestran las respuestas realizadas a las pregunta hechas por él.
Evaluación de la prueba: Aceptada.

Tabla 41: Ver respuestas realizadas a preguntas del usuario.

Prueba de aceptación
HU: Ver respuesta sobre preguntas del usuario.
Nombre: Dar puntuación a un experto.
Descripción: El usuario debe tener la posibilidad de otorgarle una puntuación a un experto según la respuesta dada por el mismo.
Condiciones de ejecución: El usuario debe estar logueado en el sistema, debe existir al menos una respuesta a una de sus preguntas.
Entrada/Pasos de ejecución: Luego de acceder a las respuestas existentes, el usuario debe otorgar una puntuación al experto que respondió.
Resultado esperado: Se le muestra al usuario que dio la puntuación y se le elimina la posibilidad de otorgar otra.

Evaluación de la prueba: Aceptada.

Tabla 42: Dar puntuación a un experto.

Prueba de aceptación
HU: Insertar pregunta.
Nombre: Insertar una pregunta
Descripción: El usuario debe tener la posibilidad de insertar una pregunta, siempre seleccionando una categoría.
Condiciones de ejecución: El usuario debe estar logueado en el sistema, debe existir al menos una categoría.
Entrada/Pasos de ejecución: El usuario intenta insertar una pregunta, para ello selecciona una categoría.
Resultado esperado: Se inserta la pregunta y se le brinda al usuario la posibilidad de insertar otra si así lo desea.
Evaluación de la prueba: Aceptada.

Tabla 43: Insertar una pregunta.

Prueba de aceptación

HU: Responder pregunta.
Nombre: Responder una pregunta
Descripción: El usuario debe tener la posibilidad de responder una pregunta.
Condiciones de ejecución: El usuario debe estar logueado en el sistema, debe existir al menos una pregunta, debe existir al menos una categoría, el usuario debe ser experto en la categoría correspondiente a la pregunta.
Entrada/Pasos de ejecución: El usuario intenta responder la pregunta.
Resultado esperado: Se inserta la respuesta, y se avisa al usuario que su pregunta ha sido respondida.
Evaluación de la prueba: Aceptada.

Tabla 44: Responder una pregunta.

Prueba de aceptación
HU: Usuario experto.
Nombre: Convertir un usuario en experto.
Descripción: El usuario debe tener la posibilidad de convertirse en experto en una categoría determinada.
Condiciones de ejecución: El usuario debe estar logueado en el sistema, debe existir al

menos una categoría.
Entrada/Pasos de ejecución: El usuario intenta convertirse en experto.
Resultado esperado: El usuario se convierte en experto.
Evaluación de la prueba: Aceptada.

Tabla 45: Convertir un usuario en experto.

Conclusiones.

En el presente capítulo se detallaron los artefactos pertenecientes al diseño y la implementación del sistema, así como las planillas de pruebas, pertenecientes el ciclo de vida del proyecto haciendo uso de la metodología XP.

Conclusiones generales:

En este trabajo se realizó un estudio de las tecnologías y tendencias actuales en cuanto al desarrollo de aplicaciones web. Se analizaron las principales herramientas, lenguajes de programación y metodologías que han alcanzado un gran auge hoy en día. También se detallaron los artefactos generados por XP a lo largo de la investigación, debido a que es la metodología que se utilizó para guiar el desarrollo del trabajo. El uso de esta metodología permitió la documentación desde un inicio del desarrollo, lo cual sirve de ayuda a los futuros desarrolladores, permitiendo una mejor comprensión del sistema.

Con este trabajo se presenta una aplicación web que gestiona la información referente a problemas tanto de software como de hardware existente en las áreas productivas de la UCI, así como sus soluciones, además se centraliza dicha gestión. La versión actual del sistema, constituye una base para futuras versiones de la misma.

Recomendaciones

Con el objetivo de lograr mejoras en el funcionamiento de la aplicación se recomienda:

- Publicar el portal en la universidad para que puedan utilizarlo los estudiantes y profesores en función de viabilizar los problemas existentes en la UCI.
- Profundizar en el estudio de aplicaciones web de gestión de información para introducir nuevas funcionalidades y mejorar el sistema propuesto.
- Medir el impacto de la aplicación en la universidad mediante encuestas y otras técnicas.

Glosario de términos:

API: Del inglés *Application Programming Interface* - Interfaz de Programación de Aplicaciones es el conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

CGI: Interfaz de entrada común (en inglés Common Gateway Interface) es una importante tecnología de la World Wide Web que permite a un cliente (explorador web) solicitar datos de un programa ejecutado en un servidor web. Especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor web y una aplicación externa.

CMS: *Content Management System* (Sistema de Gestión de Contenidos), son sistemas usados para la construcción de aplicaciones que gestionan contenido.

Open Sourse (Código Abierto): Es una tendencia internacional del desarrollo de software que profesa la distribución del código junto a las aplicaciones, se rigen por licencias tales como GNU/GPL.

Http: *HyperText Transfer Protocol* (Protocolo de transferencia de hipertexto). Es el protocolo usado para intercambiar archivos (texto, gráfica, imágenes, sonido, video y otros archivos multimedia) en la World Wide Web.

Microsoft: Compañía de software más grande del mundo. Fue fundada en 1975 por Paul Allen y Bill Gates. Aunque también se conoce por sus lenguajes de programación y aplicaciones para computadores personales, el éxito sobresaliente de Microsoft se debe a sus sistemas operativos DOS y Windows.

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

Release: Nueva versión de una aplicación informática.

SSL: Del inglés *Secure Sockets Layer* es un protocolo criptográfico que proporciona comunicaciones seguras en Internet.

UCI: Universidad de las Ciencias Informáticas.

Servicios web: Componente de software que puede auto describirse y provee cierta funcionalidad a otras aplicaciones, a través de una conexión de Internet. Esas aplicaciones, acceden los Web Services vía protocolos Web y formatos de datos estándares, tales como http y XML, sin tener en cuenta en absoluto cómo los Web Services están implementados.

Zend: Compañía líder de infraestructuras para web; está reconocida internacionalmente como la autoridad actualmente en PHP. Sus fundadores son los diseñadores del PHP v.4 en adelante, actualmente es una compañía líder dentro de la comunidad Open Source.

XML, siglas en inglés de *E**x**tensible **M**arkup **L**anguage* («lenguaje de marcas), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C).

Simple Mail Transfer Protocol (SMTP) Protocolo Simple de Transferencia de Correo, es un protocolo de la capa de aplicación. Protocolo de red basado en texto utilizado para el intercambio de mensajes de correo electrónico entre computadoras u otros dispositivos (teléfonos móviles, etc.).

WSDL son las siglas de *Web Services Description Language*, un formato XML que se utiliza para describir servicios Web (algunas personas lo leen como *wisdel*). La versión 1.0 fue la primera recomendación por parte del W3C y la versión 1.1 no alcanzó nunca tal estatus. La versión 2.0 se convirtió en la recomendación actual por parte de dicha entidad.

Active Server Pages (**ASP**) es una tecnología de Microsoft del tipo "lado del servidor" para páginas web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Services (IIS).

Perl es un lenguaje de programación diseñado por Larry Wall en 1987. Perl toma características del lenguaje C, del lenguaje interpretado shell (sh), AWK, sed, Lisp y, en un grado inferior, de muchos otros lenguajes de programación.

La **refactorización** (del inglés *Refactoring*) es una técnica de la ingeniería de software para reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo.

Referencias bibliográficas:

1. *W3C WORLD WIDE WEB*. (08 de 04 de 2008).Recuperado el 2 de Febrero del 2009, de W3C: http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb.

2. *masadelante.com*. (2008). Recuperado el 2 de Febrero de 2009, de masadelante.com: http://www.masadelante.com/faq-servidor.htm.

3. *desarrolloweb.com.* Recuperado el 5 de febrero del 2009, de http://www.desarrolloweb.com/articulos/239.php.

4. *pyspanishdoc.sourceforge.net*. Recuperado el 5 de febrero del 2009, de http://pyspanishdoc.sourceforge.net/tut/node3.html.

5. *maestrosdelWeb.com.* (23 de 05 de 2001). Recuperado el 5 de febrero del 2009, de http://www.maestrosdelWeb.com/editorial/phpintro.

6. *openformats.org.* Recuperado el 6 de febrero del 2009, de http://www.openformats.org/es61.

7. *fismat.umich.mx.* Recuperado el 6 de febrero del 2009, de http://www.fismat.umich.mx/~elizalde/tesis/node49.html.

8. *maestrosdelweb.com.* (3 de 07 de 2007) Recuperado el 5 de febrero del 2009, de http://www.maestrosdelweb.com/editorial/%C2%BFque-es-javascript.

9. *tufuncion.com.* Recuperado el 6 de febrero del 2009, de http://www.tufuncion.com/zend-studio.

10. *www.error500.net. (2004)* Recuperado el 8 de febrero del 2009, de http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base_de_datos_sgbd.php.

11. *slideshare.net.* Recuperado el 8 de febrero del 2009, de http://www.slideshare.net/alexmerono/sistemas-gestores-de-bases-de-datos.

12. *dev.mysql.com.* Recuperado el 8 de febrero del 2009, de http://dev.mysql.com/doc/refman/5.0/es/introduction.html.

13. *desarrolloweb.com*. Recuperado el 8 de febrero del 2009, de http://www.desarrolloweb.com/articulos/840.php.

14. *eaprende.com.* Recuperado el 8 de febrero del 2009, de http://www.eaprende.com/gestor-de-basededatos-mysql-postresql-sqlite.html

15. *brconsulting.info.* Recuperado el 8 de febrero del 2009, de http://brconsulting.info/portal/articulos/metodologias-de-desarrollo/extreme-programming---xp.html.

16. *microsoft.com.* Recuperado el 8 de febrero del 2009, de http://www.microsoft.com/spanish/MSDN/estudiantes/ingsoft/planificacion/msf.mspx.

17. *iteraprocess.com.* Recuperado el 8 de febrero del 2009, de http://www.iteraprocess.com/index.php?option=com_content&task=view&id=18&Itemid=42

18. *redalyc.uaemex.mx*. (2005) Recuperado el 9 de febrero del 2009, de http://redalyc.uaemex.mx/redalyc/pdf/215/21513706.pdf

19. *sparxsystems.com.ar*. Recuperado el 9 de febrero del 2009, de http://sparxsystems.com.ar/products/ea.

20. html*rational.com.ar.* Recuperado el 9 de febrero del 2009, de http://www.rational.com.ar/herramientas/roseenterprise.html

21. *desarrolloweb.com.* Recuperado el 9 de febrero del 2009, de http://www.desarrolloweb.com/articulos/1622.php

22. *slideshare.net.* Recuperado el 9 de febrero del 2009, de http://www.slideshare.net/Decimo/arquitectura-3-capas.

23. *oness.sourceforge.net.* Recuperado el 9 de febrero del 2009, de http://oness.sourceforge.net/proyecto/html/ch03s02.html

24. Drupal Web. Recuperado el 9 de febrero de 2009. http://www.drupalweb.com/

25. Allende, Roberto. *Desarrollo de Portales y Extranet con Plone.* 2006.

26. *masadelante.com.* Recuperado el 6 de abril del 2009. http://www.masadelante.com/faq-css.htm

27. *comunique.com.do*. Recuperado el 10 de abril del 2009. http://www.comunique.com.do/blogs/6/CSS-Definicion.html.

Bibliografía

1. *Phil Bartle.* Recuperado el 2 de febrero del 2009, de http://www.scn.org/mpfc/modules/mon-miss.htm.

2. *pyspanishdoc.sourceforge.net.* Recuperado el 5 de febrero del 2009, de http://pyspanishdoc.sourceforge.net/tut/node3.html

3. *buenmaster.com.* (11 de 12 de 2007). Recuperado el 6 de febrero del 2009, de http://buenmaster.com/?a=1203

4. *www.tufuncion.com.* (2008). Recuperado el 6 de febrero del 2009, de http://buenmaster.com/?a=1203.

5. *maestrosdelweb.com*. Recuperado el 6 de febrero del 2009, de http://www.maestrosdelweb.com/editorial/zendstudio/

6. *dev.mysql.com.* Recuperado el 8 de febrero del 2009, de http://dev.mysql.com/doc/refman/5.0/es/what-is.html

7. *utm.mx.* Recuperado el 8 de febrero del 2009, de http://www.utm.mx/~temas/temas-docs/nfnotas516.pdf

8. *El lenguaje Unificado de Modelado (UML).* Recuperado el 8 de febrero del 2009, de http://www.acta.es/articulos_mf/26067.pdf

9. *REVISTA Universidad EAFIT Vol. 41 No. 137 2005.* Recuperado el 9 de febrero del 2009, de http://redalyc.uaemex.mx/redalyc/pdf/215/21513706.pdf

10. Curbello, F. A. (Julio de 2007). Sistema de Gestión de Información de la Facultad 8. Módulo de Gestión de la Residencia estudiantil. *Sistema de Gestión de Información de la Facultad 8. Módulo de Gestión de la Residencia estudiantil.* Ciudad Habana, Cuba.

11. Allende, Roberto. (2006)*Desarrollo de Portales y Extranet con Plone.*