Universidad de las Ciencias Informáticas

Arquitectura de Presentación de la Plataforma ZERA

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

Autores

Yanetsi Cáceres Díaz

Noslén Herrera Sánchez

Tutores

Ing.Jorge Antonio Díaz Gutiérrez

Ing.Jose Rodríguez Alarcon

La Habana, 2011

"Año 53 de la Revolución"

Declaración de Autoría

Declaramos que somos los únicos autores del trabajo “Arquitectura de presentación de la Plataforma ZERA” y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

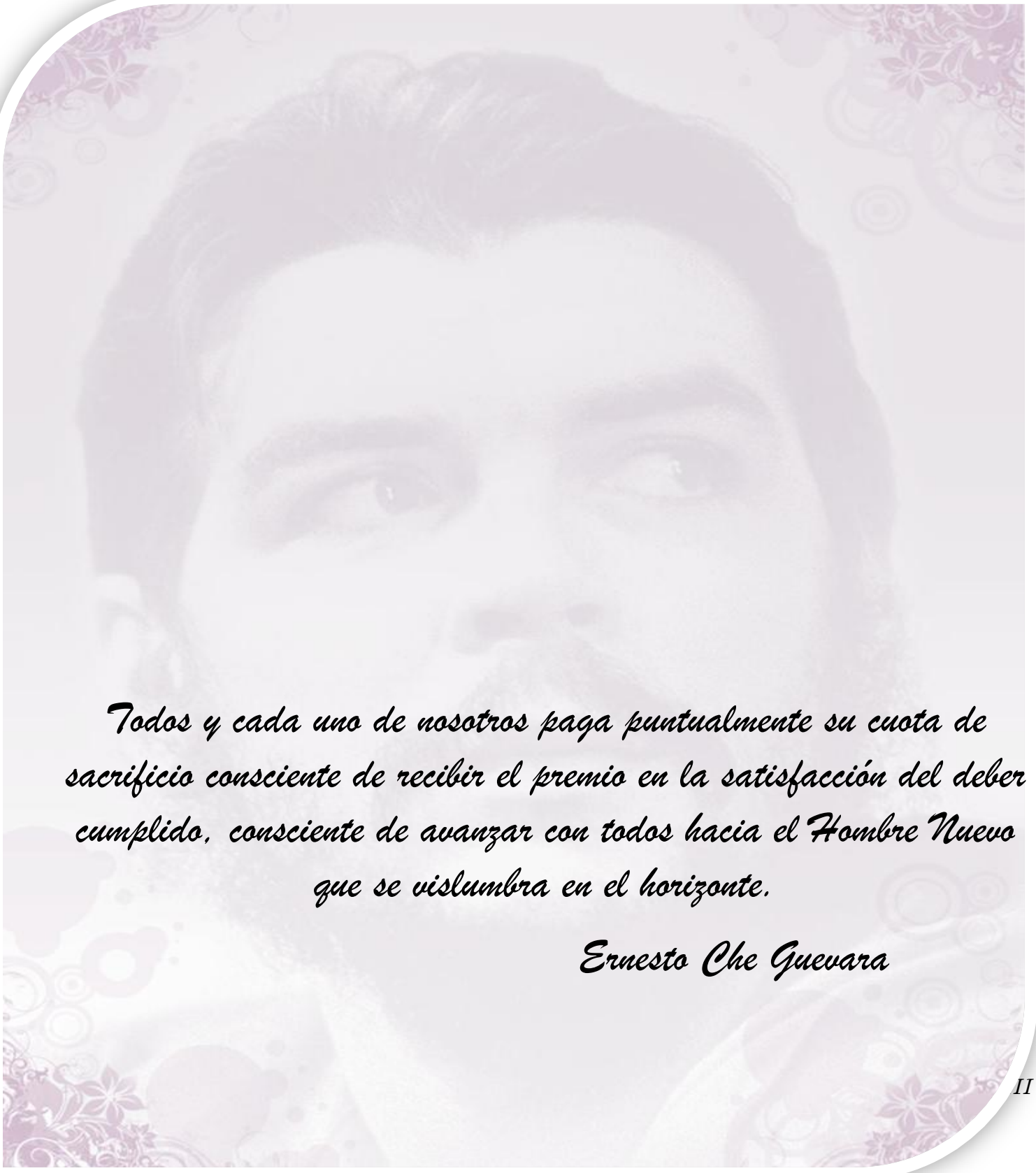
Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Autor: Yanetsi Cáceres Díaz

Autor: Noslén Herrera Sánchez

Tutor: Jorge A Díaz Gutiérrez

Tutor: José Rodríguez Alarcón



Todos y cada uno de nosotros paga puntualmente su cuota de sacrificio consciente de recibir el premio en la satisfacción del deber cumplido, consciente de avanzar con todos hacia el Hombre Nuevo que se vislumbra en el horizonte.

Ernesto Che Guevara

Agradecimientos

De Yanetsi:

A mi mamá Marilín por su amor, apoyo, sacrificio y ejemplo constante, a ti te debo más que mi vida, eres mi mayor tesoro y te agradeceré siempre por haberme guiado mis pasos hasta convertirme en la mujer que soy hoy.

A mi abuelito pupi por ser toda su vida un ejemplo para mí, por su enseñanza y guiarme por el mejor camino, dándome siempre las fuerzas paras cumplir mis metas.

A mi papá por sus consejos y experiencias que me hicieron aprender mucho para la vida.

A toda mi familia en especial a mi abuela Hilda, mi bisabuela mima, mis tíos, primos, por nunca dejar de confiar en mí y ayudarme a cumplir mi sueño.

A mi novio Nosly por su total dedicación, su amor y paciencia. Gracias por hacerme tan feliz, eso me daba las fuerzas necesarias para seguir adelante.

A mi amiga Jessica por siempre apoyarme, estar a mi lado en los momentos más difíciles y ser la hermana que no tengo.

A mis amigos de siempre Lisi, Miro, Freddy, Héctor, Anita, Zuly, Anett, Idalmita, Mily, Julio, Hismel y a todo el resto por haber vivido juntos tantos momentos alegres y difíciles para llegar hasta aquí y estar siempre ahí para mí.

A mi compañero de tesis y tutores Jorge y José, por su ayuda sin ellos no hubiese podido llegar a donde estoy hoy, por los consejos y sugerencias en todo momento, por nunca estar conforme con nada y mostrarme que puede ser mejor.

A La Revolución Cubana por hacer realidad el sueño de convertirme en una profesional y a La Universidad de las Ciencias Informáticas (UCI) por los momentos vividos.

Agradecimientos

De Noslén:

Primeramente agradezco a mis padres, por todo su amor, siéntanse ustedes autores inequívocos de este sueño hecho realidad.

A mi hermana y mi familia, por todo su apoyo.

A mi pareja y amiga Anita, por iluminar mi vida con su luz.

A mis tutores, por su abnegación y guiarme en el camino, por convertirse en amigos y hermanos.

A todos los amigos con los que compartí tantos momentos de alegría en estos años.

A la Revolución por darme la posibilidad de convertirme en un hombre de bien.

Dedicatorias

De Yanetsi:

La realización de este gran sueño se la dedico a las personas más especiales que tengo en mi vida:

Especialmente a mi mamá, por el amor incondicional que me ha brindado desde que vine al mundo, por ser una mujer de sostén infinito, sacrificio, de lucha constante pues no se ha dejado vencer a pesar de las dificultades, por impulsarme para que lograra enfrentarme a cada obstáculo que se presentara en mi camino, por haberme ayudado a construir mi futuro, y sobre todo por tener en ti una de las más grandes razones por las que continuar superándome cada día.

A mi abuelito pupi que aunque no tuve la dicha de compartir con él esta felicidad, la llevo en el corazón eternamente, mi tesis te la dedico papá.

A toda mi familia, todas las personas que quiero, en cada una de las líneas de este trabajo están presentes ustedes, les dedico este, mi más grande triunfo en los años de vida que tengo y la felicidad de este gran anhelado momento de graduarme.

A mi novio, alguien muy especial para mí, siempre ha estado a mi lado incondicionalmente en las buenas y malas, gracias por darme siempre lo mejor de ti.

De Noslén:

A mis padres por su apoyo y su amor. Todo lo que soy y seré se los debo a ellos. A mi familia, mi pareja y todas las personas que me han comprendido y brindado su ayuda desinteresada en el camino que he transitado para convertirme en un hombre de bien.

Resumen

Debido a la necesidad de aumentar la industria del software y mejorar la calidad de los productos en Cuba se creó la Universidad de Ciencias Informáticas, donde se lleva a cabo el desarrollo de la plataforma LMS (Sistemas de Administración de Aprendizaje, del inglés Learning management System) denomina ZERA. El presente trabajo surgió con el objetivo de lograr una optimización integral de los elementos presentes en la arquitectura de presentación de la plataforma, para que de este modo ZERA pueda proyectarse en los ordenadores de los usuarios con una mejor organización, calidad y rapidez. Además se realiza un administrador de temas que permitirá que se automatice el trabajo con los temas en la plataforma.

Para darle solución a lo anteriormente planteado, se realizan las investigaciones necesarias en cuanto al desarrollo de la arquitectura de presentación. Para guiar el proceso de desarrollo se usó la metodología de desarrollo RUP, escogiéndose herramientas de software libre. Se realiza un pequeño análisis, diseño e implementación para el administrador de temas dándoles respuesta a todos los requisitos funcionales y no funcionales definidos.

Finalmente se realizan pruebas para evaluar la calidad de la arquitectura desarrollada y el buen funcionamiento del administrador de temas obteniendo resultados satisfactorios.

Palabras Clave: arquitectura de presentación, optimización, calidad, rapidez, administrador de temas.

Índice de contenido

Índice de contenido

Índice de Tablas

INTRODUCCIÓN

El proceso de informatizar cada una de las esferas de la sociedad es una de las necesidades que se está tratando de resolver hoy en día en todo el mundo. Para hacer posible este sueño el país ha realizado grandes esfuerzos haciendo uso de las Tecnologías de la Información y las Comunicaciones (TICs). Estas repercuten significativamente en el crecimiento de productos de software que posibilitan la relación con el usuario y la satisfacción de sus necesidades formativas e informativas. Las TIC están siempre presentes, forman parte de la cultura tecnológica que nos rodea y con la que debemos convivir. Amplían las capacidades físicas, mentales y las posibilidades de desarrollo social.

Aunque generalmente son los países desarrollados los mayores productores y consumidores de sistemas informáticos en el mundo, Cuba en la oleada de este crecimiento se ha trazado estrategias para penetrar en el mercado mundial con productos de software que cubran las necesidades y expectativas de los usuarios. Unas de las esferas mas priorizadas hoy en día es la educación, con el uso de los nuevos medios informáticos se le facilita el trabajo para que los estudiantes adquieran un mayor conocimiento, teniendo en sus manos todo tipo de información y programas para el procesamiento de datos.

En este marco, surgen los Sistemas de Administración de Aprendizaje (del inglés Learning Management System, en lo adelante LMS) los cuales integran un gran número de funcionalidades encaminadas a facilitar la interacción entre los docentes y los estudiantes. Estos son plataformas para controlar, distribuir y administrar las actividades de formación no presencial o e-Learning de alguna institución como cursos en línea, teniendo la posibilidad de contar con personas capacitadas los cuales brindan contenidos, monitorean la participación de los estudiantes dentro del sistema, aportan herramientas para la gestión de contenidos académicos y permiten el seguimiento y la evaluación de cada uno de ellos.

Los LMS hacen que los estudiantes adquieran un conjunto de habilidades para el uso de herramientas interactivas como video-conferencias, chat, foros de discusión, correo electrónico, etc. Constituyen un espacio donde el usuario puede gestionar el aprendizaje por sí mismo, e interactuar con otros usuarios (alumnos, profesores, etc. Los aspectos gráficos, la sencillez, la reutilización de componentes de interfaz, el rendimiento, la claridad de las pantallas de entrada de datos y la experiencia de los usuarios con los sistemas, encabezan una lista muy amplia del movimiento conocido como la WEB 2.0. El usuario como centro de la información, los soportes para varias plataformas y la separación de los contenidos de la forma de presentarse, han dado lugar a que la construcción de estos sistemas se divida en 2 ramas: la gestión de la información y la arquitectura de la presentación.

Debido a la necesidad de desarrollar la industria del software y mejorar la calidad de los productos se creó la Universidad de Ciencias Informáticas (UCI), un centro de enseñanza superior destinado a convertirse en una potencia informática de desarrollo de software. En ella se lleva a cabo la realización de varios proyectos productivos de los cuales por su amplio desarrollo y aplicación se destaca el proyecto Alfaomega, donde se lleva a cabo el desarrollo de la plataforma LMS denominada ZERA.

ZERA integra los principales conceptos de los Hiperentornos, sus mejores prácticas, especificaciones y estándares educativos desarrollados y utilizados a nivel mundial en plataformas de aprendizaje colaborativo. Permite la visualización y la gestión del aprendizaje. Cuenta con un gran volumen de contenido y debido a la falta de coordinación y experiencia de sus desarrolladores la capa de presentación se encuentra desorganizada, existe redundancia en el uso de varios componentes como imágenes, estilos, código HTML y funciones Script, reduciendo considerablemente el rendimiento de la plataforma en el momento de visualizarse en los ordenadores de los clientes.

A partir de lo anteriormente planteado, se puede plantear como **problema a resolver** ¿Cómo lograr un estándar en la definición y organización de los estilos y elementos pertenecientes a la capa de presentación de la plataforma ZERA, para robustecer la arquitectura de presentación de toda la información contenida en la misma?

Este problema se enmarca en el **objeto de estudio:** La arquitectura de presentación de aplicaciones Web.

El **Objetivo General** de la investigación es lograr una optimización integral de los elementos presentes en la arquitectura de presentación de la plataforma ZERA.

Para dar cumplimiento al objetivo general planteado se desglosan los siguientes **objetivos específicos:**

- ✓ Analizar las diferentes arquitecturas de presentación de aplicaciones WEB actuales.
- ✓ Determinar las herramientas y tecnologías a utilizar.
- ✓ Determinar los estilos y patrones arquitectónicos que se emplearán durante el proceso de desarrollo.
- ✓ Desarrollar una arquitectura basada en temas.
- ✓ Construir los elementos físicos que componen la arquitectura.
- ✓ Especificar la arquitectura de presentación.
- ✓ Evaluar la arquitectura propuesta.

El **Campo de acción** de la investigación es: La arquitectura de presentación de la plataforma ZERA.

Para desarrollar la investigación se plantea, como **idea a defender**, que con el desarrollo de la arquitectura de presentación de la plataforma ZERA, se obtendrá una aplicación de alta calidad en cuanto organización, usabilidad y accesibilidad para los usuarios, además de quedar de una forma más organizada la capa de presentación, con una mejor estructuración de la información contenida en la misma.

Con el fin de dar cumplimiento a los objetivos de esta investigación se plantean las siguientes **tareas a realizar:**

- ✓ Realización de un levantamiento de sitios WEB con gran aceptación por parte de los usuarios por su calidad en la presentación de los contenidos.
- ✓ Realización de un estudio de soluciones similares de la arquitectura de presentación.
- ✓ Selección de estudios anteriores, las mejores prácticas y soluciones a utilizar como patrones.
- ✓ Realización de un análisis y de una evaluación de la situación actual de la arquitectura de presentación de la plataforma ZERA.
- ✓ Realización de un análisis de los diferentes subsistemas involucrados.
- ✓ Organización de todos los elementos actuales de presentación de la plataforma ZERA.
- ✓ Evaluación de la capacidad de asimilación de cambios de presentación en la plataforma ZERA.
- ✓ Ejecución del diseño de la nueva propuesta de arquitectura de presentación basada en temas.
- ✓ Construcción del Administrador de Temas de la Plataforma ZERA.
- ✓ Construcción de los elementos físicos que soportan la arquitectura de presentación.
- ✓ Confección del documento de tesis y el de arquitectura de presentación.

Como **Métodos de la Investigación Científica** para el desarrollo de la siguiente investigación se utilizaron:

Histórico-Lógico: Se realizó la investigación a partir del estudio de la problemática analizada, en lo que respecta a la arquitectura de presentación, teniendo conocimiento de su trayectoria y desarrollo en aplicaciones anteriormente realizadas por otros autores, se realizó un estudio de los conceptos y estándares a utilizar, lo que dio la posibilidad de seleccionar las herramientas y metodologías más apropiadas para el desarrollo de la aplicación.

Analítico-Sintético: Se encaminó la investigación a partir del estudio de conceptos y métodos, realizando una recopilación de información, la cual permitió extraer lo esencial de la bibliografía estudiada, lo referente a la arquitectura de presentación.

Inductivo-Deductivo: Este método facilita el análisis de los elementos generales a elementos más específicos, el cual permitió a través de un estudio general de diferentes propuestas de arquitecturas de presentación de plataformas anteriores, realizar comparaciones sobre las características de las mismas y de este modo determinar los elementos comunes que estarán presentes en el sistema.

Métodos Empíricos

Observación: Mediante este método se podría estar al tanto de la evolución del trabajo realizado, estando atentos de posibles atrasos o fallas en el cumplimiento de las tareas propuestas. Además este método permitirá recoger información de todos y cada uno de los indicadores de las variables que figuran en el problema.

Experimento: Después de haber estudiado lo referente a la problemática existente se realizarán una serie de pruebas con distintos frameworks para determinar cuál es el más apropiado para la realización de la arquitectura.

Entrevistas: Este método permitirá obtener información respecto al objeto de estudio, mediante personas especializadas en el tema, los cuales posibilitarán la determinación de los principales elementos para la automatización del sistema, así como la valoración teórica de la solución propuesta.

Estructura Capitular

El presente trabajo de diploma está estructurado en cuatro capítulos:

Capítulo 1: Fundamentación Teórica

En este capítulo se abarcan los elementos correspondientes al estudio de temas relacionados con la arquitectura de presentación como base teórica para fundamentar las decisiones tomadas, se analizan los principales conceptos, así como lo referente a estilos y patrones arquitectónicos, descripción de la arquitectura, herramientas, tecnologías asociadas, y la metodología de desarrollo a utilizar. Además de abordar un estudio de los temas relacionados con la evaluación de la arquitectura de presentación.

Capítulo 2: Propuesta de la arquitectura de presentación para la plataforma ZERA

En el desarrollo de este capítulo se analizan los principales problemas que afectan a la plataforma ZERA, realizándose un análisis de todos los elementos que la forman, ya sea estilos, códigos, imágenes, etc. Se exponen además las vías de solución práctica a las dificultades, para optimizar la Arquitectura de Presentación.

Capítulo 3: Descripción de la arquitectura realizada

Este capítulo incluye una descripción de los documentos y artefactos que componen la arquitectura de presentación para la plataforma ZERA, una descripción general de la propuesta para el sistema y la forma en que debe ser utilizada por cada uno de los usuarios.

Capítulo 4: Descripción y evaluación del administrador de temas y de la arquitectura propuesta.

En este capítulo se muestra una descripción de cada uno de los temas que se integraran en la plataforma, así como la evaluación de la arquitectura descrita en los capítulos anteriores, realizándose un pequeño ciclo de análisis, diseño, implementación y pruebas para un mejor entendimiento, logrando de este modo la validación de la arquitectura de presentación definida. Se analizan criterios hechos a miembros especializados del proyecto con el objetivo de evaluar y demostrar que es factible la arquitectura de presentación desarrollada para el producto.

Capítulo 1 Fundamentación Teórica

1.1-Introducción

El desarrollo de una buena arquitectura de presentación es uno de los aspectos más importantes en la realización de cualquier producto. Mediante esta los especialistas aportan sus ideas, para que los sistemas funcionen de una forma más flexible, organizada y con mejor calidad. En el presente capítulo se abordarán algunos de los aspectos más importantes que componen la arquitectura de presentación en un producto de software. Se muestran algunos conceptos que permiten tener un mejor conocimiento sobre estos procesos, las tendencias actuales de la Arquitectura de Presentación y sus principales elementos. Se realiza un estudio de propuestas existentes en todo el mundo, favoreciendo así el conocimiento para que el producto que se desea desarrollar tenga una mayor calidad y cumpla con las necesidades de los usuarios.

1.2- Conceptos asociados al problema en cuestión

1.2.1- Arquitectura de software

La Arquitectura de Software va a definir el sistema en términos de componentes e interrelaciones entre esos componentes. Va a mostrar la correspondencia entre los requisitos del sistema construido, además de reflejar los requisitos no funcionales, como son capacidad, rendimiento, consistencia, compatibilidad de componentes. (1)

Gracias a la Arquitectura de Software los programadores, diseñadores, ingenieros y analistas pueden trabajar bajo una línea común que les posibilite la compatibilidad necesaria para lograr el objetivo deseado.

1.2.2- Arquitectura Cliente-Servidor

Esta arquitectura consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta. Aunque esta idea se puede aplicar a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema. La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma.

Una disposición muy común son los sistemas multicapa en los que el servidor se descompone en diferentes programas que pueden ser ejecutados por diferentes computadoras aumentando así el grado de distribución del sistema. La arquitectura cliente-servidor sustituye a la arquitectura monolítica en la que no hay distribución, tanto a nivel físico como a nivel lógico. (2)

1.2.3- Arquitectura MVC (Modelo Vista Controlador)

Cada patrón describe un problema que ocurre una y otra vez en el entorno, para describir después el núcleo de la solución a ese problema, de tal manera que esa solución pueda ser usada más de un millón de veces sin hacerlo siquiera dos veces de la misma forma". El patrón es un esquema de solución que se aplica a un tipo de problema, esta aplicación del patrón no es mecánica, sino que requiere de adaptación y matices. (3)

El patrón Modelo Vista Controlador (MVC) es un patrón de arquitectura de software descrito por primera vez en 1979 por Trygve Reenskaug. Este patrón arquitectónico es muy usado en la ingeniería de software, gestiona la información y advierte a las otras capas de cambio en sus datos. El patrón MVC representa el dominio de datos. La vista es la que va a representar gráficamente el modelo para que el usuario pueda interactuar con él, va ser la interfaz de datos, el control recibe las peticiones de la vista y le responde actualizando el modelo de datos. Debido a que la vista observa los cambios en el modelo de datos esta actualiza sus componentes en función de estos.

Patrón Modelo Vista Controlador en Symfony

Symfony está basado en un patrón clásico del diseño web conocido como arquitectura Modelo Vista Controlador (en lo adelante MVC), que está formado por tres niveles (4):

Modelo: encapsula los datos y las funcionalidades, es independiente de cualquier representación de salida y/o comportamiento de entrada. Es responsable de acceder a la capa de almacenamiento, lleva un registro de las vistas y controladores del sistema.

Vista: muestra la información al usuario, pueden existir varias vistas del modelo y cada una tiene asociado un componente controlador. Estas tienen la responsabilidad de recibir los datos del modelo y mostrárselo al usuario.

Controlador: recibe las entradas tales como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, entre otros y los eventos son traducidos a solicitudes de servicio para el modelo o la vista.

El patrón seleccionado como guía para el diseño de la aplicación separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones.

Symfony aplica este patrón en su funcionamiento, este divide la aplicación en tres capas fundamentales:

- ✓ App: son almacenados los módulos creados, con las vistas (success) y sus controladoras (actions).
- ✓ Web: almacena los componentes referentes a las vistas, imágenes a utilizar, CCS, archivos JavaScript, etc.
- ✓ Lib: almacena lo relacionado a la capa del modelo, agrupado en la carpeta model las abstracciones generadas para el mapeo de la base de datos.

1.2.4- Arquitectura N Capas

La Arquitectura en N Capas es en realidad un estilo de programación donde el objetivo principal es separar los diferentes aspectos del desarrollo, tales como las cuestiones de presentación, lógica de negocio, mecanismos de almacenamientos, etc. El término capa es utilizado para diferenciar las distintas partes en que una aplicación se divide desde un punto de vista lógico. (5)

Arquitectura en 3 capas

Es el diseño más utilizado actualmente en el desarrollo de Aplicaciones Webs. Sus capas están constituidas por:

Capa de presentación: Es la que ve el usuario (también se la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio.

Capa de negocio: En esta capa 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 almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.

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.

Todas estas capas pueden residir en un único ordenador, si bien lo más usual es que haya una multitud de ordenadores en donde reside la capa de presentación (son los clientes de la arquitectura cliente/servidor). Las capas de negocio y de datos pueden residir en el mismo ordenador, y si el crecimiento de las necesidades lo aconseja se pueden separar en dos o más ordenadores.

Así, si el tamaño o complejidad de la base de datos aumenta, se puede separar en varios ordenadores los cuales recibirán las peticiones del ordenador en que resida la capa de negocio. Si por el contrario, fuese la complejidad en la capa de negocio lo que obligase a la separación, esta capa de negocio podría residir en uno o más ordenadores que realizarían solicitudes a una única base de datos.

En sistemas muy complejos se llega a tener una serie de ordenadores sobre los cuales corre la capa de negocio, y otra serie de ordenadores sobre los cuales corre la base de datos. (6)

1.2.5- Arquitectura de presentación en aplicaciones web

Tras el estudio de varios conceptos relacionados con la arquitectura de presentación no se obtuvo ninguno con la base y la calidad requerida ya que existe una gran variedad de criterios con respecto al tema, debido a esto se dará el siguiente concepto de arquitectura de presentación.

La arquitectura de presentación en una aplicación web constituye el conjunto de reglas, pautas y patrones de diseño que deben seguir los desarrolladores de las aplicaciones web. La arquitectura de presentación establece el camino a seguir para simplificar el trabajo y optimizar tanto el código como los estilos que componen la aplicación web.

En la construcción de la presentación de una aplicación web existen una serie de elementos regidos por la W3C[1]. Estos elementos son:

Usabilidad

La usabilidad es la efectividad, eficiencia y satisfacción con la que un producto permite alcanzar objetivos específicos en un contexto de uso determinado. Es la facilidad de uso, ya sea de una página web, una aplicación informática o cualquier otro sistema que interactué con un usuario. La usabilidad permite mayor rapidez en la realización de tareas y reduce las pérdidas de tiempo.

Accesibilidad

Hablar de Accesibilidad Web es hablar de un acceso universal a la Web, independientemente del tipo de hardware, software, infraestructura de red, idioma, cultura, localización geográfica y capacidades de los usuarios. Por ejemplo una web deja de ser accesible cuando para poder visualizarla correctamente necesitamos un plugin[2] especial u otra herramienta que el usuario no disponga en ese momento.

Entonces la idea es hacer la web accesible para todos los usuarios independientemente de las circunstancias y los dispositivos utilizados a la hora de acceder a la información. Una página accesible lo será tanto para una persona con discapacidad, como para cualquier otra persona que se encuentre bajo circunstancias externas que dificulten su acceso a la información.

[1] World Wide Web es una comunidad internacional donde las organizaciones y el público en general trabajan conjuntamente para desarrollar estándares Web.
[2] Es un módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

1.2.6- Arquitectura de presentación y diseño web

Los términos de Arquitectura de Presentación y Diseño Web, están estrechamente vinculados, ya que el segundo representa una nomenclatura que se le ha dado al proceso de construcción de los elementos de la arquitectura de presentación de las aplicaciones web.

El diseño web además, es la actividad que consiste en la planificación, diseño e implementación de sitios web. No es simplemente una aplicación del diseño convencional, ya que requiere tener en cuenta cuestiones tales como navegabilidad, interactividad, usabilidad, arquitectura de la información y la interacción de medios como el audio, texto, imagen y vídeo.

La unión de un buen diseño con una jerarquía bien elaborada de contenidos aumenta la eficiencia de la web como canal de comunicación e intercambio de datos, que brinda posibilidades como el contacto directo entre el productor y el consumidor de contenidos, característica destacable del medio.

1.3-Evolución de la arquitectura de presentación o diseño web

La Arquitectura de Presentación de las aplicaciones web fue mejorando y pasando por distintas generaciones, principalmente debido a mejoras en la tecnología, hardware y software. Actualmente existen 5 generaciones en la historia de la Arquitectura de Presentación y el diseño web.

Primera generación del diseño web (1992-1994)

Aunque el nacimiento de Internet data de 1989, la primera página en internet nace de la idea de interconectar información mediante enlaces o links basados en el sistema que Tim Berners-Lee en 1990 había creado de documentos enlazados y accesibles a través de Internet. Esta primera página fue insertada en internet el 6 de agosto de 1991, en ella se enviaba un pequeño resumen del proyecto World Wide Web al newsgroup alt.hypertext. No obstante se establece que las primeras maquetaciones estructurales de una página web y por tanto el nacimiento del diseño web están vinculados a los primeros navegadores gráficos, el primero fue World Wide Web creado para el sistema operativo NEXTSTEP, renombrado después Nexus en 1992 y Mosaic como primer navegador gráfico para Microsoft Windows en enero de 1993, El navegador web era capaz de mostrar tanto imágenes como textos, aunque tenía grandes problemas para diagramar la información dentro de la página web. El diseño web de esas páginas era lineal y enfocado a los científicos que eran los usuarios que compartían su información alrededor del mundo mediante estas páginas web. La tecnología de los navegadores web era muy limitada y no disponía de la capacidad de transmitir información gráfica para la comunicación visual. Esta etapa está muy vinculada al tipo de conexión vía modem, por lo tanto se limitaba el peso de la página. Las principales características de esta primera generación de diseño web eran las velocidades de transmisión de datos, ya que era conexión vía modem, lo que limitaba el peso de las páginas web. Otro detalle era el

uso de monitores monocromos. Respecto al diseño web en particular, la estructuración era bastante desordenada con imágenes dispuestas horizontalmente y líneas de texto separadoras.

Debido a este caos en el diseño web, un año más tarde se estableció un consorcio para establecer estándares y pautas para el futuro desarrollo de la WEB, el W3C. Se comenzaron entonces a desarrollar unos estándares de lenguaje HTML para una unificación del diseño web que trajo consigo la aparición de una serie de navegadores web con el constante desarrollo de nuevas funcionalidades y progresos en este ámbito. (7)

Segunda generación (1994-1996)

El diseño web de esta generación está basado en los conceptos de la primera salvo por que empiezan a utilizarse nuevos elementos en el diseño como fueron imágenes, el cambio de los fondos de las páginas, tablas para la maquetación de contenidos y distribución de información similares a la maquetación de las revistas entre otros. La aparición de estos elementos y muchos otros generó el desarrollo de unos estándares de lenguaje HTML para la unificación del diseño web, el denominado W3C. De forma paralela, gracias a la aparición tecnológica de los primeros monitores a color y las primeras tarjetas gráficas, en este periodo apareció la diferenciación en la adaptación de estándares de los dos principales navegadores; Internet Explorer y Netscape Navigator. (7)

Tercera Generación (1996-2000)

En la tercera generación, el diseño web sigue teniendo muchas restricciones con el uso del lenguaje para los dos navegadores web. El diseño web se orienta en esta generación a los diseñadores, los cuales tienen mucho más dinamismo al aparecer los componentes de Macromedia Flash, el cual revolucionaría la concepción de diseño web.

Es una era de enfocar las páginas web según el objetivo de las mismas, vender productos o servicios, comunidades, información, noticias. Para esta especialización del diseño web de acuerdo al objetivo de las páginas se necesita ayudar al usuario a encontrar la información, generando una navegabilidad estructurada e intuitiva. (7)

Cuarta Generación (2000-2005)

En la cuarta generación, el diseño web ya está enfocado totalmente a la multimedia, integrando en las páginas web los elementos multimedia de última generación. Con usuarios de todos los tipos, cualquiera tiene una página web hoy en día y la variedad de diseño es enorme debido a todas las posibilidades que ofrecen las últimas tecnologías para los programadores. A esto le podemos añadir que las últimas versiones de los navegadores soportan muchas más prestaciones y elementos en las páginas web. (7)

Quinta Generación (2005-Actualidad)

Generación que marca el nacimiento y desarrollo de las redes sociales, aplicaciones multimedia en los dispositivos móviles, iphone, televisión ip o televisión online, etc. Esta etapa del diseño web está más

enfocada a la gestión multimedia de todos los elementos para la interactividad del usuario con el medio web, a esta nueva tendencia se le ha denominado Web 2.0. (7)

1.3.1- Movimiento Web 2.0

El término Web 2.0 (2005–Actualidad) está comúnmente asociado con un fenómeno social, basado en la interacción que se logra a partir de diferentes aplicaciones en la web, que facilitan el compartir información, la interoperabilidad, el diseño centrado en el usuario y la colaboración en la World Wide Web. Ejemplos de la Web 2.0 son las comunidades web, los servicios web, las aplicaciones Web, los servicios de red social, los servicios de alojamiento de videos, las wikis, blogs. Un sitio Web 2.0 permite a sus usuarios interactuar con otros usuarios o cambiar contenido del sitio web, en contraste a sitios web no-interactivos donde los usuarios se limitan a la visualización pasiva de información que se les proporciona. (8)

Características

- ✓ Las páginas son dinámicas, integran recursos multimedia como videos, sonidos, que se pueden compartir.
- ✓ Los formatos utilizados para diseñarlas son java script, PHP, u otras similares, que permiten más funcionalidad.
- ✓ Emplean interfaces de fácil entendimiento para la interacción del usuario.
- ✓ La información se puede presentar en varias formas (escrita, audiovisual), y que esta se comparta entre los usuarios o entre estos y los dueños de las páginas.
- ✓ Permite que el usuario cree su propio contenido.
- ✓ La información se puede transmitir unidireccional o bidireccionalmente.

Diferencias con la Web 1.0

En la web 1.0 el usuario tenía acceso a la información solamente como receptor, no tenía la posibilidad de participar de los contenidos, las páginas eran estáticas, generalmente solo de texto y pocas imágenes, y el formato utilizado era el HTML. La interacción de los usuarios no era posible con esta forma de diseño de páginas, la información en la web era construida solo por los dueños de los sitios, no era posible sustentarla con opiniones, temas de interés, o recursos aportados por los usuarios.

1.4- Lenguajes, herramientas y metodologías

1.4.1- Lenguajes para la implementación de la propuesta

CSS 2: (del inglés Cascading Style Sheets, en lo adelante CSS) es un lenguaje creado para definir la presentación de un documento estructurado escrito en HTML o XML. Además se utiliza para dar estilos a documentos con estos lenguajes, separando el contenido de la presentación. Los estilos van a definir la

forma de mostrar los elementos HTML y XML además de permitir a los desarrolladores web controlar el estilo y el formato de múltiples páginas web al mismo tiempo. (9)

Las hojas de estilos van compuestas por una o más reglas, es decir declaraciones sobre el estilo de uno o más elementos. La regla contendrá dos partes un selector y la declaración, donde la declaración contendrá una propiedad y el valor que se le asigne. (10)

Características de CSS

- ✓ El estilo se puede guardar completamente por separado del contenido siendo posible, por ejemplo, almacenar todos los estilos de presentación para una web en un sólo archivo de CSS.
- ✓ Permite un mejor control en la presentación de un sitio web que los elementos de HTML, agilizando su actualización.
- ✓ Hay un aumento de la accesibilidad de los usuarios gracias a que pueden especificar su propia hoja de estilo, permitiéndoles modificar el formato de un sitio web según sus necesidades.
- ✓ El ahorro global en el ancho de banda es notable, ya que la hoja de estilo se almacena en cache después de la primera solicitud y se puede volver a usar para cada página del sitio, no se tiene que descargar con cada página web.
- ✓ Una página puede tener diferentes hojas de estilo para mostrarse en diferentes dispositivos, como pueden ser impresoras o móviles.

Niveles o Especificaciones CSS

A finales de 1996, el W3C publicó la primera recomendación oficial, conocida como "CSS nivel 1". El 12 de Mayo de 1998, el grupo de trabajo de CSS publica su segunda recomendación oficial, conocida como "CSS nivel 2". La versión de CSS que utilizan todos los navegadores de hoy en día es CSS 2.1, una revisión de CSS 2 que aún se está elaborando (la última actualización es del 23 de abril de 2009). Al mismo tiempo, la siguiente recomendación de CSS, conocida como "CSS nivel 3", continúa en desarrollo desde 1998 y hasta el momento sólo se han publicado borradores. (11)

El nivel CSS utilizado en la Arquitectura de Presentación de la Plataforma es el 2.1, siendo esta la última versión operativa ya que la tercera versión aún está en desarrollo.

XHTML 1. 0

XHTML, acrónimo en inglés de Extensible Hypertext Markup Language (lenguaje extensible de marcado de hipertexto), es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web. En su versión 1.0, XHTML es solamente la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML. Su objetivo es avanzar en el proyecto del World Wide Web Consortium de lograr una web semántica, donde la información, y la

forma de presentarla estén claramente separadas. La versión 1.1 es similar, pero parte a la especificación en módulos. En sucesivas versiones la W3C planea romper con los tags clásicos traídos de HTML.

Ventajas respecto a HTML

Las principales ventajas del XHTML sobre el HTML son (12):

- ✓ Se pueden incorporar elementos de distintos espacios de nombres XML (como MathML y Scalable Vector Graphics).
- ✓ Un navegador no necesita implementar heurísticas para detectar qué quiso poner el autor, por lo que el parser puede ser mucho más sencillo.
- ✓ Como es XML se pueden utilizar fácilmente herramientas creadas para procesamiento de documentos XML genéricos (editores, XSLT, etc.).

JavaScript 1.3

Es un lenguaje de programación que se utiliza para realizar acciones dentro del ámbito de una página Web. Con este lenguaje se pueden realizar tareas como la validación de los datos enviados por el usuario en un formulario, además de crear páginas Web dinámicas, en las que el usuario acceda a información personalizada, convirtiéndolas en páginas. Se trata de una programación del lado cliente, ya que su carga de procesamiento es soportada por el navegador, y puesto que es compatible con la mayoría de ellos. (13)

1.4.2- Herramientas para el desarrollo

1.4.2.1- Entornos de desarrollo

Entornos de Desarrollo

Un entorno de desarrollo integrado (del inglés Integrated Development Environment o IDE) es un programa compuesto por una serie de herramientas que utilizan los programadores para desarrollar código. Esta herramienta puede estar pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de estos. Las herramientas que normalmente componen un entorno de desarrollo integrado son las siguientes: un editor de texto, un compilador, un intérprete, unas herramientas para la automatización, un depurador, un sistema de ayuda para la construcción de interfaces gráficas de usuario y, opcionalmente, un sistema de control de versiones.

Netbeans 6.9

Es un proyecto muy exitoso de código abierto con una gran cantidad de usuarios, una comunidad en constante crecimiento en todo el mundo. Se fundó en junio 2000 siendo el patrocinador principal de muchos proyectos. Este tiene disponible dos productos siendo de gran importancia para muchos programadores: Netbeans Platform y Netbeans IDE.

Netbeans Platform es una base modular y extensible usada como estructura de integración para crear grandes aplicaciones de escritorio. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones. Es un producto gratuito para usos tanto comerciales como no comerciales y de código abierto.

Netbeans IDE es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Es un producto gratuito de código abierto y sin restricciones escrito en Java pero puede servir para cualquier otro lenguaje de programación. (14)

Zend Studio 8.0

Zend Studio es un completo entorno de desarrollo para lenguaje de programación PHP, escrito en Java y sirve de editor de texto, además proporciona una serie de ayudas que pasan desde la creación de gestión de proyectos hasta la depuración de código. Zend Studio ha lanzado con relativa facilidad y rapidez versiones del producto para Microsoft Windows, GNU/Linux y Mac Os X[3], aunque el desarrollo de las versiones de este último sistema se retrase un poco más. (15)

Zend Studio consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Este fue diseñado para usarse con el lenguaje PHP, sin embargo ofrece soporte básico para otros lenguajes Web como por ejemplo, HTML, XML, y Javascript.

Eclipse 3.6

La plataforma consiste en un entorno de desarrollo integrado, abierto y extensible. Eclipse sirve como IDE java y cuenta con numerosas herramientas de desarrollo de software. Además da soporte a otros lenguaje de programación, como son C/C++, Fortrn, PHP. Es una plataforma libre, robusta y de calidad para el desarrollo de aplicaciones. Su núcleo de ejecución se llama Equinox y es una implementación de la especificación OSCI (Open Services gatway initiative), la cual describe una arquitectura orientada a servicios para aplicaciones. (16)

Adobe DreamWeaver 8.0

Es una aplicación en forma de estudio destinado a la construcción y edición de sitios y aplicaciones web basados en estándares. Es de gran uso en el sector del diseño y la programación web gracias a su funcionalidad y a su integración con otras herramientas y programas, como Adobe Flash. La principal base que posee es su gran poder de ampliación y personalización. Muchas tareas se realizan con javascript permitiéndolo ser un programa fluido.

[3] Sistema Operativo desarrollado y comercializado por Apple Inc., basado en UNIX. Se construyó sobre las tecnologías desarrolladas en NeXT.

Adobe Dreamweaver 8 permite previsualizar las páginas web con la mayoría de los navegadores Web. Además, dispone de herramientas de administración de sitios dirigidas a principiantes, como reemplazar líneas de código y texto. (17)

Entorno de desarrollo para la implementación de la propuesta

Teniendo en cuenta el estudio realizado acerca de los entornos de desarrollo integrado se escogió NetBeans en su versión 6.9 como IDE a utilizar. Presenta las características esenciales de acuerdo a la solución que se desea implementar. Este IDE soporta los lenguajes de la web, tanto los del lado del cliente (HTML, CSS, JavaScript) como los del servidor (en este caso PHP) para los cuales brinda un gran completamiento de código, además de ser multiplataforma. NetBeans cuenta con un sistema de proyectos basado en control de versiones y refactorización sin necesidad de abrir una aplicación externa. Es altamente configurable y además una herramienta de código abierto, cumpliendo con las políticas de la universidad y del país al uso de software libre.

1.4.2.2- Frameworks

Frameworks

Los frameworks han adquirido gran popularidad en estos tiempos, teniendo reacciones en los programadores tanto positivas como negativas por sus características, estos sirven de gran ayuda en el desarrollo de software, proporcionando una estructura definida la cual ayuda a crear aplicaciones con mayor rapidez.

Los frameworks son desarrollados con el objetivo de brindarles a los programadores y diseñadores una mejor organización y estructura a sus proyectos. Estos ayudan a la hora de realizar el mantenimiento del sitio gracias a la organización durante el desarrollo de la aplicación, además de simplificar las tareas de los programadores gracias a sus propiedades.

En términos generales podemos definir al framework como una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado. Puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto. (18)

Ventajas del uso de Frameworks de Presentación

- ✓ Se ahorra tiempo en esos procesos que necesitamos repetir.
- ✓ No se empieza a programar de cero, a la hora de crear una nueva hoja de estilo.
- ✓ Se pueden utilizar tanto para páginas estáticas como para cualquier CMS[4] más complejo.

[4] Un sistema de gestión de contenidos es un programa que permite crear una estructura de soporte (framework) para la creación y administración de contenidos principalmente en páginas web por parte de los participantes.

- ✓ Se elimina la necesidad de utilizar tablas anidadas en HTML.
- ✓ Se reduce de gran manera los errores de codificación ya que existe una gran consistencia entre todos elemento.
- ✓ Tienen gran apoyo en estándares webs.

Framework CSS

Igual que sus parientes orientados a lenguajes de servidor o cliente, el objetivo de un framework CSS será ahorrar la realización de tareas básicas al trabajar con hojas de estilo. Normalmente los frameworks CSS se componen de uno o varios archivos con declaraciones predefinidas. Actualmente las páginas se maquetan con CSS, con HTML especificamos los contenidos y con CSS la forma o disposición con la que deben presentarse al usuario en los navegadores.

Los frameworks CSS disponen de una serie de clases u hojas de estilos ya creadas con las que ayudan a posicionar elementos en la página y crear estructuras de maquetación más o menos versátiles. De esta forma en el desarrollo de nuevas páginas o en el rediseño de páginas antiguas es posible tener la ayuda de estos frameworks para disponer de una rejilla donde posicionar los distintos componentes del diseño. Con estos adelantos se ahorra tiempo en tener que crear de nuevo decenas de clases que estamos aburridos de implementar para crear maquetaciones a 2,3 ó 4 columnas, con divisiones de cabeceras, cuerpos y pie. (19)

Ventajas de usar Framework CSS:

- ✓ Permite agilizar el desarrollo, sobre todo en sus momentos iniciales.
- ✓ Ahorra las habituales diferencias entre navegadores para conseguir que los layouts sean "cross-browser".
- ✓ Da la posibilidad sobre una base ordenada desarrollar un trabajo adicional.
- ✓ Si el framework CSS está bien documentado, agiliza el trabajo en un equipo relativamente grande.

Muchos son los frameworks CSS que han sido desarrollados, algunos de ellos son:

- ✓ Blueprint
- ✓ 960 Grid System
- ✓ Simple
- ✓ YAML

Blueprint 1.0.1

Blueprint es un framework CSS el cual tiene como objetivo reducir el tiempo de desarrollo de las páginas web, es un complemento para desarrolladores de webs que aumentará la productividad en las tareas de

diseño y maquetación de páginas. Se trata de una serie de librerías de Hojas de Estilo en Cascada que contienen código útil para maquetar una página web y aplicar otros tipos de estilos tipográficos o de impresión. Ofrece una estructura sólida en la que fundar el trabajo de diseño por medio de la típica rejilla, además de ofrecer una serie de clases muy útiles para estilizar componentes típicos como formularios, botones, pestañas, tipografías o para que las páginas web se puedan imprimir de manera óptima en papel. (20)

Ofrece además una serie de archivos con código CSS que podremos incluir en las páginas web para aplicar estilos de una manera rápida, y solucionar también muchos de los problemas de compatibilidad entre navegadores que pueden surgir durante la etapa de maquetación. Blueprint es uno de los framework más conocidos, debido a todas las posibilidades que le brindan a los programadores que quieran hacer uso de él, como son (20):

- ✓ Una rejilla, que permitirá crear cualquier estructura de página web. Básicamente se trata de un conjunto de clases CSS para posicionar cualquier elemento en un espacio de 950 píxeles de anchura, dividido en 24 columnas.
- ✓ Una definición de tipografía predeterminada, de manera precisa para todos los elementos de la página que pueden tener texto.
- ✓ Estilos para formularios, con los que mejorar las interfaces de usuario.
- ✓ Estilos para impresión, con las definiciones CSS más útiles para que las páginas se lean bien cuando se imprimen en papel.
- ✓ Estilos específicos para IE, con los que resolver algunos de los problemas más comunes del navegador Internet Explorer.

960 Grid System 1.0

El framework 960 Grid es uno de los más utilizados por los desarrolladores de sitios web donde cuyas páginas se construyen con una anchura de 960 píxeles, de ahí su nombre. Este ofrece además dos posibilidades de maquetación de páginas, con una sola rejilla. Las columnas que podremos colocar en la rejilla tendrán distintas anchuras, pero siempre el ancho total de la página será de 960 píxeles. Se escogió este número porque 960 es divisible por una buena cantidad de números lo que lo hace más versátil y además según estudios realizados al mismo se ha comprobado que es muy fácil de usar.

Este framework se creó por la necesidad de tener un estándar en el ancho de los sitios web, ya que actualmente la mayoría utilizan la resolución 1024 x 768 píxeles o mayores. Dada esta necesidad un grupo de personas decidió crear este sistema de maquetado basado en 960px de ancho, con configuraciones de 12 y 16 columnas para poder combinar entre sí, cada una de las columnas tiene un margen izquierdo y derecho de 10px, para crear 20px de separación entre columnas, de esta manera podemos crear layout de una forma simple y rápida. (21)

Simple 1.4

Es un sistema que implementa la infraestructura necesaria para trabajar con una rejilla, de modo que el desarrollador pueda maquetar una página web ajustando los contenidos del diseño a unas guías, que delimitarán los espacios disponibles para colocar los distintos elementos de la página. Con este framework CSS se obtiene un código de hojas de estilo en cascada que permitirán posicionar los elementos que componen el diseño de una manera rápida y sobre todo, como su nombre lo indica, simple. Está registrado bajo licencia Creative Commons Atribución-No Comercial-Licenciar Igual 2.0 Chile License. (22)

YAML 3.3

YAML sus iniciales son de Yet Another Multicolumn Layout, es un framework de desarrollo de HTML/CSS, para crear layouts flexibles y flotantes. Combina una serie de características bastantes interesantes que sin dudas harán más sencillo el trabajo a la hora de diseñar con CSS como son:

Un esquema definido de plantillas que incluye una cabecera, un pie de página y un área para una o más columnas muy propio de las nuevas tendencias de diseño.

Compatibilidad con la mayoría de los navegadores utilizando la plantilla por defecto proporcionada en la distribución. (23)

- ✓ Esquemas de diseños fijos o fluidos.
- ✓ Centrado en estándares web y accesibilidad.
- ✓ Parte central del framework reducida, con múltiples extensiones.
- ✓ Concepto de layout robusto y flexible (columnas y grids).
- ✓ Modelos de diseño para formularios, tipografía, microformats, etc.

Framework CSS para la implementación de la propuesta

El framework CSS escogido para el desarrollo de la Plataforma ZERA es Blueprint en su versión 1.0.1. La selección se justifica por sus características, que lo hacen superior al resto de los frameworks CSS estudiados anteriormente. Una de las más importantes es la presencia de un archivo CSS exclusivo para los usuarios que utilizan Internet Explorer. Esta característica (que los demás frameworks CSS estudiados no incluyen) posibilita que el diseño realizado por los desarrolladores usando BluePrint pueda ser compatible con este navegador. Otras características distintivas que permitieron su selección son:

- ✓ Su estructura de carpetas y archivos es más amplia, puesto que contiene iconos y archivos CSS para tipografía adicional y diseño para formularios.
- ✓ Además de las clases para crear espacios vacíos y restarle la sangría de derecha o izquierda, tiene también para aplicar bordes y agregar o mover X columnas a la capa.
- ✓ Posee un archivo CSS exclusivo para los usuarios que utilizan Internet Explorer.

1.4.2.3- Navegador principal de prueba

Un Explorador Web o Navegador es un programa que permite visualizar páginas web en la red además de acceder a otros recursos, documentos almacenados y guardar información. El mismo se comunica con el servidor a través del protocolo HTTP y le pide el archivo solicitado en código HTML, después lo interpreta y lo muestra en pantalla para el usuario.

Varios son los navegadores que existen en el mundo, adelante se analizan los utilizados como pruebas para el desarrollo debido a sus características y funciones explícitas. Estos son:

Internet Explorer 7.0

Internet Explorer (IE) viene integrado en el sistema operativo Windows de Microsoft. Ha sido el navegador más utilizado del mundo desde 1999, aunque desde 2002 ha ido perdiendo cuota de mercado a un ritmo lento pero constante debido a su importante competidor, Mozilla Firefox, que ha superado incluso a Internet Explorer en algunas ocasiones. Con el motor de renderizado (motor de navegación) Trident soporta HTML 4.01, CSS Level 1, XML 1.0 y DOM Level 1, con pequeñas lagunas de implementación, soporta también XSLT 1.0 y WD-xsl, y admite parcialmente CSS Level 2 y DOM Level 2 con importantes deficiencias de implementación, mientras que para MAC el motor de renderizado es Tasman. Hay varias versiones de Internet Explorer para los sistemas operativos UNIX y para Mac. La versión más reciente para Windows es Internet Explorer 8.0 que Microsoft describe como más rápido, más fácil y más seguro. (24)

Incompatibilidades de Internet Explorer con algunas propiedades CSS:

Tiene gran importancia el estudio de las principales incompatibilidades de este navegador con las propiedades CSS que se muestran a continuación, tomando como base, que a pesar de las dificultades que presenta el mismo, es el más usado en internet actualmente, y la Plataforma ZERA debe ser compatible con este navegador.

Propiedades CSS:

- ✓ Bordes redondeados (border-radius).
- ✓ Fallo de la transparencia png.
- ✓ Posicionamiento fijo de imagen de fondo (background-attachement: fixed).
- ✓ Ancho mínimo y máximo (min-width y max-width).
- ✓ Altura mínima y máxima (min-heigth y max-height).
- ✓ Sombra en los textos (text-shadow).

Otra incompatibilidad fundamental, que, aunque no es una propiedad CSS, constituye el fallo de la transparencia png para imágenes con este formato.

Mozilla Firefox 3.6.14

Mozilla Firefox es un navegador de software libre y código abierto, creado por la Corporación Mozilla, la Fundación Mozilla y numerosos voluntarios externos, con una gran aceptación por parte de los usuarios que lo definen como más seguro, rápido y de mejor rendimiento que Internet Explorer, destacando también por su sencillez y fácil manejo. Su motor de navegación Gecko para visualizar páginas web soporta varios estándares web incluyendo HTML, XML, XHTML, CSS 1, 2 y 3, SVG 1.1 (parcial), ECMAScript (JavaScript), DOM, DTD, MathML, XSLT, XPath, además de imágenes PNG con transparencia alfa. Es multiplataforma para varias versiones de Microsoft Windows, GNU/Linux, Mac OS X, y algunos sistemas basados en Explorer, e incluye el software de correo Thunderbird. Como características añadidas a las habituales de todos los navegadores, Mozilla Firefox ofrece también múltiples plugins, extensiones add-ons y la posibilidad de personalizar su apariencia, además Firefox ofrece herramientas muy útiles para los programadores web como la consola de errores, el inspector DOM o extensiones como Firebug, por estas razones y el hecho de ser de código abierto es el preferido por los programadores. (24)

Safari 5.0.3

Safari es el navegador creado por Apple Inc. el cual está integrado en el sistema operativo Mac OS X, en 2007 se creó una versión de Safari para el sistema operativo Microsoft Windows dando soporte tanto a Windows XP como a Windows Vista, y el teléfono inteligente iPhone también incorpora Safari a su sistema operativo. De Safari destacan la velocidad, el diseño, la seguridad y las prestaciones que ofrece, incluyendo los recursos para diseñadores y programadores, su motor de renderizado WebKit está basado en el motor KHTML, y debido a esto, el motor interno de Safari es software libre. La nueva versión recién estrenada es Safari 5.0.3 que ejecuta JavaScript casi ocho veces más rápido que IE 8 y más de cuatro veces más rápido que Firefox 3 gracias a su nuevo motor JavaScript Nitro. El nuevo navegador Safari 5.0.3 soporta además los innovadores estándares HTML 5 y CSS con unas avanzadas aplicaciones web multimedia, fuentes tipográficas y gráficos, y es el primer navegador que superó la prueba Acid3 que examina si los navegadores cumplen los estándares web CSS, JavaScript, XML, DOM, ECMAScript y SVG. (24)

Opera 10.63

Desarrollado por Opera Software company, Opera es además de un navegador una suite de Internet gratuita. Reconocido por su gran velocidad, seguridad y constante innovación, Opera es reconocido por su soporte de estándares a través de su motor de navegación Presto en especial CSS 2.1, además de HTML 4.01, XHTML 1.1, XHTML Basic, XHTML Mobile Profile, WML 2.0, XSLT, XPath, XSL-FO, ECMAScript 3 (JavaScript), DOM 2, Unicode, SVG 1.1 Basic, GIF89a, JPEG, HTTP 1.1, y completo soporte para PNG, incluyendo transparencia alfa, entre otros. Opera fue el primer navegador que implementó el sistema de

pestañas, y además de las características comunes de todos los navegadores, El navegador web Opera es multiplataforma para las versiones para Windows, GNU/Linux, Mac OS X, Solaris, QnX, OS/2, Symbian OS, FreeBSD y BeOS, entre otros, además de Opera Mini para móviles sencillos y Opera Mobile para teléfonos inteligentes y ordenadores de bolsillo, así como dispositivos de Java ME-enabled. Opera es el único navegador disponible para la nueva generación de videoconsolas Nintendo DS y Wii, también algunos decodificadores digitales de televisión usan Opera, y Adobe Systems integró la tecnología de Opera para usarla en Adobe Creative Suite. (24)

Google Chrome 12.0

Google Chrome es el navegador creado por Google en 2008 y se basa en el proyecto de software libre y código abierto Chromium, el motor de navegación de WebKit y su estructura de aplicaciones. Para conseguir su objetivo principal de facilitar un navegador con mayor velocidad, seguridad y estabilidad, Google Chrome combina tecnología sofisticada y un diseño minimalista, además de ofrecer una interfaz gráfica de usuario más sencilla y eficaz. Google Chrome está disponible para Microsoft Windows, para los usuarios de Windows Vista y Windows XP SP2, mientras que en junio de 2009 salió la versión de Google Chrome para Mac OS X (Leopard) y Linux destinada principalmente para los desarrolladores web, pues como advierten desde Google son aún muy poco estables e incluso recomiendan no descargar estas versiones pues cuenta con limitaciones importantes con un software incompleto e impredecible. A través del motor de renderizado Webkit Google Chrome soporta los estándares HTML, Javascript y CSS, cuyo proceso de instalación no requiere reiniciar el navegador para empezar a funcionar, además de hacerse como proceso independiente, al modo de las pestañas, de manera que si una de las ventanas falla no afecta al resto ni al navegador. Muchos aclaman su velocidad; implementación de distintas partes de HTML 5; poder reproducir vídeos a través de la etiqueta "vídeo" sin necesidad de utilizar Flash; o la aplicación Google Wave que va a revolucionar la forma de compartir información pues además de permitir a los usuarios charlar y compartir documentos, mapas y enlaces en tiempo real, también permite múltiples formas de interacción basados en la nube (Cloud computing). (24)

Navegador de Pruebas seleccionado

Mozilla Firefox en su versión 3.6.14 es el navegador escogido para el desarrollo y pruebas de la aplicación. La selección se justifica por sus funciones notablemente útiles, como el firebug y el web developer. Aunque el Google Chrome paulatinamente ha añadido nuevos complementos, aún para el desarrollo web no supera las cualidades y funcionalidades implementadas por Firefox.

No obstante del uso de este navegador, vale destacar, que en el desarrollo de la aplicación se realizarán pruebas de visualización y compatibilidad con otros navegadores: Internet Explorer 7.0, Safari 5.0.3, Opera 10.63 y Google Chrome 12.0.

1.4.3-Herramientas CASE

Visual Paradigm para UML

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. (25)

Características

- ✓ Soporte de UML como notación principal.
- ✓ Ingeniería de ida y vuelta.
- ✓ Ingeniería inversa-código a modelo, código a diagrama.
- ✓ Ingeniería inversa Java, C++, Esquemas XML, XML, NET.
- ✓ Generación de código.
- ✓ Generación de código para el desarrollo y despliegue de aplicaciones.
- ✓ Generación de bases de datos: Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- ✓ Distribución automática de diagramas: Reorganización de las figuras y conectores de los diagramas UML.

ArgoUML

ArgoUML es una herramienta utilizada en el modelado de sistemas, mediante la cual se realizan diseños en UML llevados a cabo en el análisis y pre-diseño de Sistemas de Software. Fue concebido como una herramienta para el análisis y el diseño de los sistemas de software orientados al objeto. En este sentido es similar a muchas de las herramientas comerciales CASE que se venden como herramientas para modelar sistemas de software. ArgoUML tiene un número de distinciones muy importantes de muchas de estas herramientas como son: (26)

- ✓ Las ayudas de ArgoUML abren extensiones UML, XMI, SVG, OCL y otros de los estándares.
- ✓ ArgoUML es 100% puro de Java. Esto permite que ArgoUML funcione en todas las plataformas para las cuales un puerto confiable de la plataforma J2EE esté disponible.
- ✓ ArgoUML es un proyecto abierto. La disponibilidad del código fuente asegura que diseñadores e investigadores de software tenga un marco probado donde puedan conducir el desarrollo y la evolución de las tecnologías.

Las herramientas de modelado de objetos son fundamentales para el análisis del sistema. En el proceso de desarrollo de software de la solución propuesta se escoge Visual Paradigm 6.4 ya que soporta la notación UML 2.1, ingeniería inversa, además brinda la posibilidad de generar código a partir de los diagramas y viceversa. Es multiplataforma, fácil de operar y posee una amplia documentación, lo que resulta provechoso para futuras modificaciones del subsistema.

1.4.4- Metodologías de desarrollo

Se puede definir una metodología de desarrollo de software como un conjunto de pasos y procedimientos que deben seguirse para desarrollar un software. El concepto metodología dentro de la ingeniería de software es uno de los más abarcadores y que más confusión provocan en las personas involucradas en procesos de desarrollo de software.

La constante innovación tecnológica hace que cada vez sea necesaria la aplicación de nuevas metodologías adaptadas a los nuevos tiempos, por lo que no es fácil decir una definición completa sin dejar algún detalle. Las metodologías imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente.

Metodologías Tradicionales

Las metodologías tradicionales son basadas en las metodologías existentes en otras áreas y adaptadas al desarrollo de software. Estas dividen el proceso de desarrollo en etapas de una forma secuencial, para de esta manera completar los objetivos deseados. Entre las metodologías tradicionales más conocidas esta RUP. Estos métodos ofrecen una documentación muy completa, exhaustiva y un plan de proyecto cuidadosamente definido.

Este tipo de metodología provee de un alto grado de ordenamiento, disciplina, son resistentes al cambio e importan más los procesos que las personas. Las metodologías tradicionales (formales) se focalizan en documentación, planificación y procesos, plantillas, técnicas de administración, revisiones, etc. (27)

Metodologías Ágiles

Las Metodologías ágiles se encargan de valorar al individuo y a las iteraciones del equipo mucho más que a las herramientas o los procesos utilizados. Además se hace mucho más importante crear un producto software que funcione que escribir tanta documentación. El cliente está en todo momento colaborando en el proyecto y es más importante la capacidad de respuesta ante un cambio realizado que el seguimiento estricto de un plan.

Metodologías actuales

RUP (Proceso Unificado de Desarrollo)

Es una metodología pensada para la Ingeniería de Software que va más allá del análisis y diseño orientado a objetos, para proporcionar un conjunto de técnicas que soportan el ciclo completo de

desarrollo de software. Además de que unifica completamente a un equipo de desarrollo y optimiza la productividad de cada uno de los miembros del mismo, ayuda a los líderes de proyecto a incrementar su experiencia en el desarrollo. RUP 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 un software de alta calidad.

Entre las principales características de RUP se encuentran:

- ✓ **Dirigido por Casos de Uso:** Los casos de usos describen las funcionalidades que presenta un sistema determinado, los mismos se obtienen a partir del modelado del negocio y la captura de requisitos. Estos guían el diseño, la implementación y las pruebas de todo proceso de desarrollo del software.
- ✓ **Centrado en la Arquitectura:** La arquitectura muestra la visión común del sistema completo donde tanto los desarrolladores como los usuarios están de acuerdo. La arquitectura ofrece una clara perspectiva del sistema, necesaria para controlar el desarrollo, describe los elementos del modelo que son más importantes, los cimientos del sistema, que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente.
- ✓ **Iterativo e Incremental:** RUP se divide en 4 fases y cada una de ellas se desarrolla en iteraciones, cada iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla unos más que otros, esto hace posible que se pueda dividir en pequeños proyectos perfeccionando su comprensión y desarrollo.

RUP posee un ciclo de vida que consta de 4 fases Concepción, Elaboración, Construcción y Transición. Contiene además 3 características fundamentales: Guiado por Casos de uso: sirven para describir el comportamiento del sistema y elaborar los casos de prueba con los que se comprueba que el sistema desarrollado “hace lo que el cliente quiere”. Centrado en la arquitectura: la arquitectura del sistema es la columna vertebral de todo el desarrollo del mismo. Cada iteración gira en torno a la misma, fortaleciendo y corrigiendo sus características. Iterativo e incremental: en cada ciclo de iteración se produce una nueva versión del software. Utiliza UML como lenguaje de modelado y cuenta con varias fases de trabajo en las cuales se desarrolla una serie de flujos fundamentales del desarrollo del proyecto. (28)

XP (Programación Extrema)

Es una metodología ligera o ágil para el desarrollo de software eficiente y altamente efectivo. Básicamente, la programación extrema, busca dos objetivos claramente: hacer un software con calidad y de la forma más rápida posible. XP centra su atención en la producción de software con una fuerte arquitectura, intentando sacar productos al mercado rápidamente, con gran calidad y motivando al equipo de trabajo para seguir mejorando esta tendencia. XP se centran en potenciar las relaciones

interpersonales como clave para el éxito en el desarrollo del software, promueve el trabajo en equipo y se basa en la retroalimentación entre el cliente y el equipo de desarrollo.

Esta metodología presenta muchos puntos comunes con el desarrollo incremental, comenzando por el hecho de que el software desarrollado con XP se realiza de forma incremental. Se divide en varias fases:

- ✓ **Probar:** se debe desarrollar la prueba desde el momento que se conocen las historias del usuario.
- ✓ **Escuchar:** tanto para diseñar, como para desarrollar pruebas, como para desarrollar, se necesita conocer exactamente lo que se quiere, para ello, se debe aprender a escuchar muy bien al cliente, al jefe de proyecto y a todos los implicados en el proyecto.
- ✓ **Diseñar:** el diseño también debe ser incremental y debe estar incluido en el software, lo cual quiere decir que la estructura de este debe ser clara. Se debe diseñar lo que las necesidades del problema requieren, no lo que se crea que debería ser el diseño.

Esta metodología suele estar especialmente orientada a proyectos cortos, aquellos en los cuales los equipos de desarrollo son pequeños, con plazos reducidos, requisitos volátiles, y basados en nuevas tecnologías, y en los cuales el cliente forma parte del equipo de desarrollo. (29)

SCRUM

Metodología ágil de desarrollo de software, que plantea que los ciclos de desarrollo de los proyectos sean lo más rápidos posible. Esta metodología se ajusta a los cambios que puedan presentarse y mantiene una estrecha relación con el cliente. Es un proceso en el que se aplican de manera sistemática un conjunto de mejores prácticas para trabajar colaborativamente, obteniendo el mejor resultado posible de un proyecto.

Con este se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. SCRUM está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales. Es muy fácil de aprender, ya que requiere de muy poco esfuerzo para utilizarlo, pero no se tiene una perspectiva global del proyecto más allá de una lista de tareas. Además, no genera toda la documentación que se obtiene con otras metodologías por lo que no es muy conveniente utilizarlo solo. Por lo que se utiliza para resolver situaciones en que no se está entregando al cliente el producto que realmente necesita.

Metodología de desarrollo a utilizar

Tras el estudio realizado se seleccionó RUP como metodología de desarrollo. Gracias a esta metodología se llevará a cabo un seguimiento en cada una de sus fases, se asegura un producto con un alto nivel de calidad el cual dará cumplimiento a los requerimientos de los usuarios finales. Tanto XP como SCRUM están destinadas a proyectos de corto plazo, todo lo contrario a RUP, destinado a proyectos y equipos

grandes sin la presencia permanente del cliente. Esta es una metodología robusta y bien definida que genera una gran cantidad de documentación detallada durante todo el proceso de desarrollo, además permite llevar a cabo un proceso de desarrollo práctico, brindando amplias guías, plantillas y ejemplos para todas las actividades.

1.4.5- Lenguaje Unificado de Modelado (UML)

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. (30)

UML es un lenguaje de modelado para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

1.4.6-Modelado de interfaces de usuarios

Durante el desarrollo de una aplicación o un software es necesario para un mejor entendimiento diseñar correctamente las interfaces que este posee. La calidad de estas puede ser uno de los motivos que conduzca al éxito de la aplicación. Para la realización de estos bocetos contamos con herramientas destinadas a dichas funciones, como son:

PencilProject

Es la alternativa libre (GPL v2) y multiplataforma para realizar esta tarea. Es una simple extensión de Firefox pero cuenta con una potencia y flexibilidad considerable. Permite crear proyectos con varias pantallas, en las cuales es posible añadir cualquier elemento (botones, tablas, cajas de activación, deslizadores). Es útil para aplicaciones de escritorio y cuenta además con multitud de elementos web. Una de sus características más importantes es la capacidad de alinear objetos para que el conjunto quede ordenado. Los proyectos se guardan en XML aunque es posible exportar los diseños a otros formatos como: PNG, PDF, ODT o HTML eligiendo entre varias plantillas disponibles. (31)

BalsamiqMockups

Es un programa de escritorio, propietario, programado en Flex y AIR. Su interfaz es bastante fácil de usar, con una colección de controles grande y puedes crear casi cualquier programa. Al ser creada en AIR, es

instalable en Windows, Linux y Mac OS X. Funciona como cualquier aplicación normal. Puedes arrastrar, soltar, guardar mockups en un archivo, exportar a PNG, copiar y pegar, deshacer, etc. También incluye ayudas de diseño similares al Flex Builder 3. El creador de Balsamiq Mockups fue desarrollador de Adobe Acrobat Connect (Antes Macromedia Breeze) (32)

Para el desarrollo de los bocetos se escogió la herramienta PencilProject 1.2 debido que es una herramienta libre mientras que Balsamiq Mockups no. PencilProject es muy fácil de usar y con excelentes características mencionadas anteriormente y gracias a eso será muy útil para todo el trabajo.

1.4.7-Conclusiones Parciales

Durante la realización de este capítulo se seleccionó al framework CSS Blueprint en su versión 1.0.1 que permitirá una maquetación más fácil y organizada, además de ser compatible con los diferentes navegadores. Las características propias de la metodología RUP permitirán llevar un control total durante todo el proceso de desarrollo. La selección del IDE NetBeans 6.9 en conjunto con los lenguajes de programación web XHTML 1.0, CSS 2.1 y JavaScript 1.3 facilitará en gran medida la implementación de cada uno de los elementos pertenecientes a la arquitectura de presentación de la plataforma ZERA.

Capítulo 2 Propuesta de la Arquitectura de presentación para la Plataforma ZERA

2.1-Introducción

Durante el desarrollo de este capítulo se realiza un análisis sobre la situación actual de la plataforma ZERA y los principales problemas existentes en la implementación de los estilos de cada uno de los elementos pertenecientes a la misma. Se muestra la nueva propuesta de arquitectura de presentación realizada para la plataforma ZERA de una forma clara y segura para el entendimiento de todos aquellos que la utilicen. Se exponen además las vías para la solución práctica a las dificultades para optimizar la misma.

2.2-Situación Actual

El análisis de las dificultades actuales de la Arquitectura de Presentación de la Plataforma ZERA, se basan en varios estándares de Aplicaciones Web, así como diversos principios, como son la reutilización, flexibilidad y mantenimiento futuro de los estilos. A continuación se analizan algunos de los principales problemas que atentan contra el óptimo desempeño de la Plataforma.

Repetición innecesaria de Estilos

La repetición innecesaria de estilos es uno de los principales problemas que presenta la arquitectura de presentación de la plataforma ZERA. Existen varios elementos que tienen, en la declaración de las reglas que componen sus estilos funciones repetidas como son el mismo color de fondo (background-color[5]), el posicionamiento (float[6]) o la imagen de fondo (background-image[7]). Debido a la poca reutilización de las reglas que componen los estilos, se hace mayor la longitud de los ficheros de estilos (.css), afectando aspectos como la portabilidad, o la velocidad de visualización de los elementos, además del buen funcionamiento y rendimiento de la aplicación.(Ver Anexo 1)

Repetición Innecesaria de imágenes

Otro de los problemas que hace que aumente el peso y la complejidad estructural de la plataforma es la repetición innecesaria de imágenes, es decir existe un gran número de estas que son similares y no son reutilizadas. Aparecen repetidas dentro de la propiedad de imagen de fondo, (background-image) en clases diferentes. (Ver Anexo 2)

[5] Propiedad que define el color de fondo de los elementos teniendo menos prioridad que la imagen de fondo.
[6] Propiedad que desplaza al elemento de su posicionamiento normal y lo ubica lo mas la izquierda (float: right) en su línea correspondiente.
[7] Propiedad que define una imagen de fondo al elemento.

Estilos aplicados en forma directa

Existen una gran cantidad de elementos, cuyos estilos se encuentran de manera directa en el código. Esta práctica negativa atenta contra los estándares actuales de separación del contenido y sus estilos. Hace engorrosa la tarea en el mantenimiento de la plataforma, ya que si por algún motivo se decide cambiar algún estilo de algún elemento, entonces habría que buscarlo en toda la plataforma y realizarle directamente el cambio. (Ver Anexo 3)

Elementos sin estilos

Una de las dificultades fundamentales es la presencia dentro de la plataforma de diversos elementos a los cuales aún no se le han asociado estilos. Este aspecto provoca que los contenidos no se visualicen de manera correcta, dándole a sus estilos los valores por defecto del navegador. (Ver Anexo 4)

2.3-Organización actual de la Arquitectura de Presentación de la Plataforma ZERA

La Plataforma ZERA cuenta con dos subsistemas: Administración e Hiperentornos. Dentro de estos se encuentran las funcionalidades de la plataforma y están compuestos por diversos elementos estructurales, como son dashboards[8], formularios, listas, tablas entre otros. Cada elemento tiene un estilo propio y responde a un patrón de interfaz específico. (Ver Anexo 5)

2.4-Rendimiento-Tiempo de Carga de la Plataforma ZERA en el navegador

El rendimiento fue medido utilizando la extensión Firebug del navegador Firefox para desarrollo web. Mediante esta extensión se obtiene una información detallada de cada uno de los elementos que componen las peticiones al servidor, dónde se encuentran, qué tamaño tienen, cuánto tiempo demora el proceso de carga en el navegador hasta que el mismo renderiza completamente la página.

Analizando la página: resources.php del subsistema Administración se obtuvieron los siguientes resultados generales:

[8] Interfaz donde el usuario puede administrar el equipo y/o software.

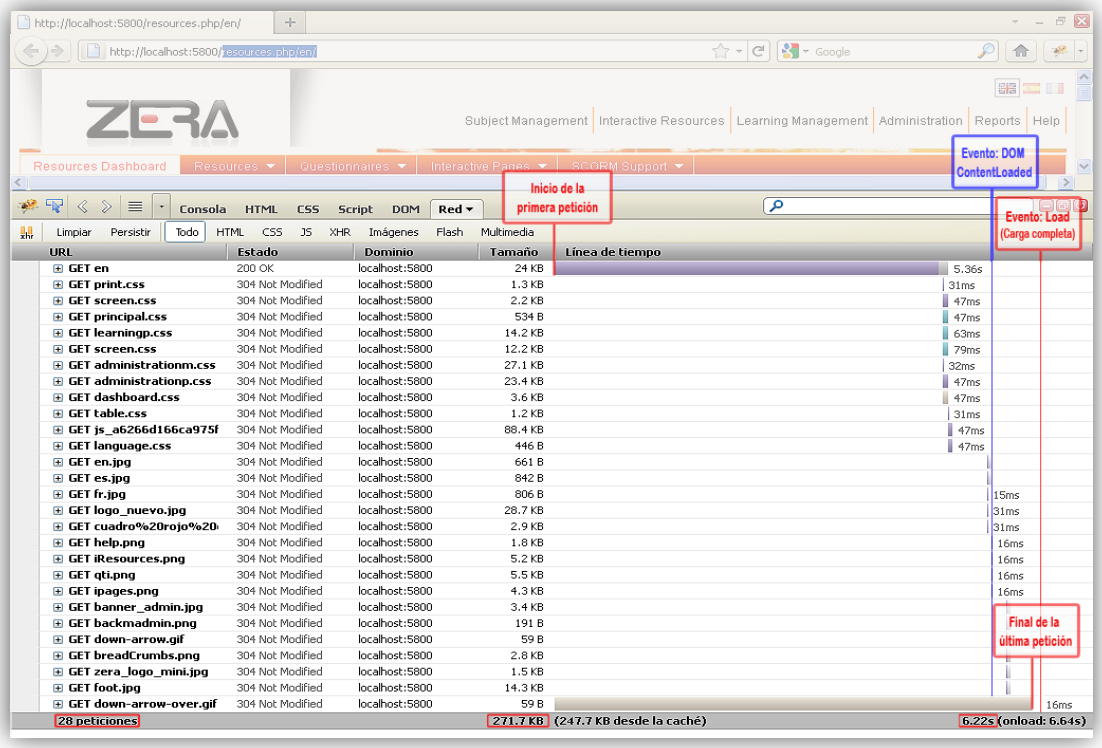


Imagen 1 *Vista del panel net de Firebug para la página resources.php*

En este panel se muestra la siguiente información:

- ✓ URL y modo del pedido: (GET).
- ✓ Estado y la descripción del pedido: (200 OK).
- ✓ Nombre de Dominio del cual proviene la respuesta del servidor: (localhost: 5800).
- ✓ Tamaño de la respuesta.
- ✓ Línea de Tiempo: Muestra el tiempo que demora cada petición al servidor.
- ✓ Evento ¨DOM Content Load¨: Cuando el navegador termina la lectura de todos los elementos estructurales de la página. En la página analizada (resources.php) fue de 6.22s.
- ✓ Evento ¨Load¨: Cuando el navegador carga completamente la página, en la página analizada (resources.php) fue de 6.64s.
- ✓ En total se realizaron 28 peticiones con un tamaño de 271.7 Kb.

Analizando la página: resources.php del subsistema Administración se obtuvieron los siguientes resultados correspondientes a los pedidos de las hojas de estilo (.css):

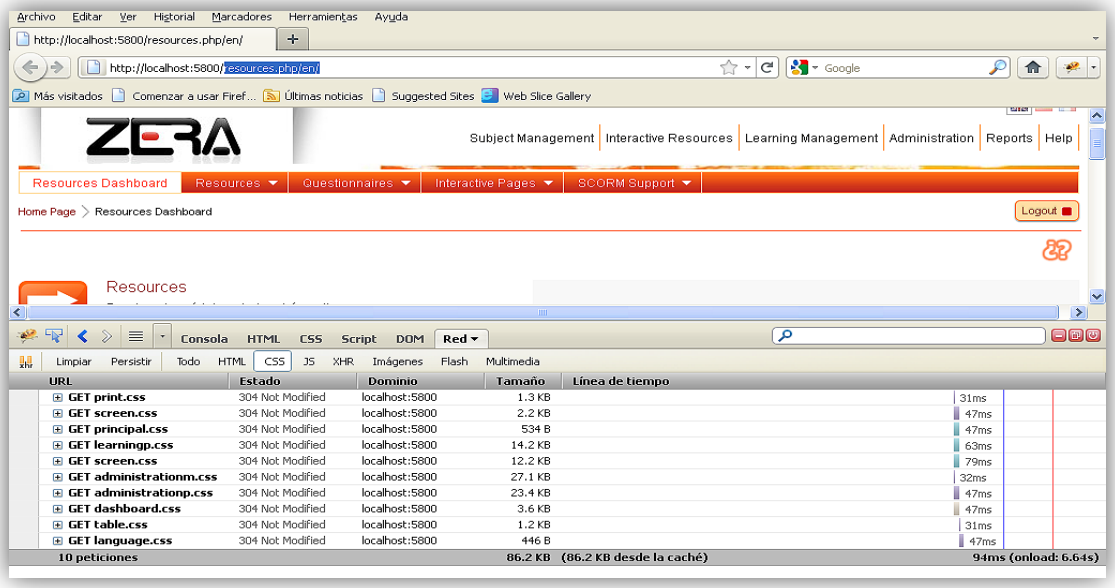


Imagen 2 *Vista del panel net de Firebug para la página resources.php correspondientes a las peticiones de hojas de estilos.*

En total se realizaron 10 peticiones al servidor con un tamaño de 86.2 Kb, solamente para cargar las hojas de estilo, esto infiere una dificultad de rendimiento, ya que para 4000 usuarios conectados de manera simultánea, serían 40 000 peticiones con un tamaño de 344 800 Kb ≈ 337 Mb.

En total se realizaron 16 peticiones al servidor con un tamaño de 73 Kb, para cargar las imágenes, para igual cantidad de usuarios conectados (4000) serían 64 000 peticiones con un peso de 292 000 Kb ≈ 285 Mb. (Ver Anexo 6)

2.5- Solución propuesta para la Arquitectura de Presentación de la Plataforma ZERA

Para eliminar las dificultades expuestas en este capítulo, se realizará una nueva reestructuración de los estilos y archivos que componen la Arquitectura de Presentación.

Patrón de Interfaz

Cada elemento estructural de los subsistemas de la Plataforma (Administración e Hiperentornos) contará con un documento que específica el patrón a seguir por los desarrolladores a la hora de utilizar algún elemento de esta forma, quedará documentado y estandarizado el proceder de los desarrolladores, simplificando el trabajo considerablemente y dando uniformidad a la Plataforma.

Maquetas

De forma paralela a los documentos de los Patrones de interfaz, cada subsistema (Administración e Hiperentornos) contará con una maqueta, dentro de la cual se encuentra un código de ejemplo de cada elemento, con sus estilos correspondientes. De esta forma se simplifica el trabajo de los desarrolladores,

ya que solamente se remitirán a los archivos que se encuentran dentro de las maquetas y copiarán los elementos que vayan a utilizar.

Reestructuración de Estilos e Imágenes

Todos los elementos que posean estilos redundantes se reagruparán en selectores de clases genéricas. De igual forma, todas las imágenes presentes en la plataforma que estén repetidas, serán eliminadas. El sistema de carpetas donde se encuentran tanto los estilos e imágenes, se reagruparán en dos directorios, una única carpeta (css) en la cual estarán los estilos del tema, y una carpeta (images) para las imágenes.

La nueva Arquitectura de Presentación de la Plataforma ZERA contará con maquetas y diversos documentos que establecen los patrones de interfaz de cada elemento: formularios, listas, dashboards, etc., de manera que los desarrolladores cuando necesiten utilizar o confeccionar algún elemento se remitan a dichos documentos, por lo que habrá en la plataforma uniformidad y estandarización.

2.6-Conclusiones Parciales

Durante el desarrollo de este capítulo se realizó un estudio de la situación actual de la plataforma y los elementos que la componen. Se evidenció que existen una serie de dificultades en cuanto a estructura, rendimiento y organización que orientan al equipo de desarrollo a realizar cambios en la arquitectura de presentación.

Capítulo 3 Descripción de la arquitectura realizada

3.1-Introducción

Después de presentar la propuesta de todo el proceso realizado para la elaboración de la Arquitectura de Presentación en la plataforma ZERA del proyecto Alfaomega, se hace necesario mostrar una buena descripción de cada uno de los elementos por el cual estará formada la misma, brindando una explicación detallada de lo que se realizó. En este capítulo se explica la solución que se quiere aplicar para el mejoramiento de la aplicación, ya sea cualitativamente como en su funcionamiento.

Se muestra una vista de cada uno de los patrones desarrollados para los elementos pertenecientes a los subsistemas de la plataforma, dando de cada uno de ellos una explicación general y así sea de entendimiento para todos. Se analizan criterios realizados a miembros con un conocimiento especializado del proyecto con el objetivo de evaluar y demostrar que es factible la arquitectura de presentación desarrollada para el producto.

3.2- Solución Aplicada a la plataforma ZERA

Durante el proceso de desarrollo de la Plataforma ZERA muchos elementos no han quedado de la forma más organizada, ni con una estructuración que pueda optimizar su buen funcionamiento. Para mejorar la situación que presenta actualmente, como se explica en el capítulo anterior, se ha creado una nueva solución la cual le permitirá a todas las personas involucradas con la plataforma tener una forma mucho más organizada, dinámica y entendible para trabajar garantizando que todos tengan un patrón por donde guiarse y seguir una misma línea de trabajo.

La solución que se ha creado para evitar todos estos problemas es la siguiente:

- ✓ Primeramente se arreglan cada unos de las páginas que no tengan estilos o que estos contengan algún error el cual provoque una vista desagradable de la información ante los ojos del usuario.
- ✓ Cada subsistema contendrá una carpeta en la cual van estar archivados sus estilos, imágenes, y código HTML, y otra donde se encuentren cada uno de los elementos comunes pertenecientes a todos los subsistemas. De este modo se perfeccionará el trabajo ya que todos los estilos que están repetidos dentro de la plataforma de forma innecesaria se eliminan y se crean una sola vez.
- ✓ Con vista a que el trabajo sea más organizado y que todas las personas trabajen de una misma manera se crean los patrones de interfaz, estos van a tener una descripción para aquellos usuarios que necesiten empezar desde el inicio la realización de un nuevo subsistema.

3.3- Patrones de Interfaz para cada subsistema de la Plataforma ZERA

3.3.1-Subsistema Administración

Este subsistema responde a las particularidades del negocio y se adapta en función de las necesidades de la institución o escuela. Sus principales procesos están relacionados con la gestión de usuarios, escuelas, réplicas, además de la gestión y configuración de los sistemas educativos. El objetivo principal es administrar y configurar los elementos que se gestionan en la plataforma así como garantizar la seguridad e integridad de la misma.

El subsistema de Administración va a estar compuesto por elementos tales como dashboard, listas, y formularios, los cuales serán los encargados de representar la información que en la plataforma se muestra.

A continuación se muestra un ejemplo del patrón de interfaz para los dashboard de Administración. Para consultar el resto de los patrones correspondientes a cada elemento y sus descripciones (Ver Anexo 7).

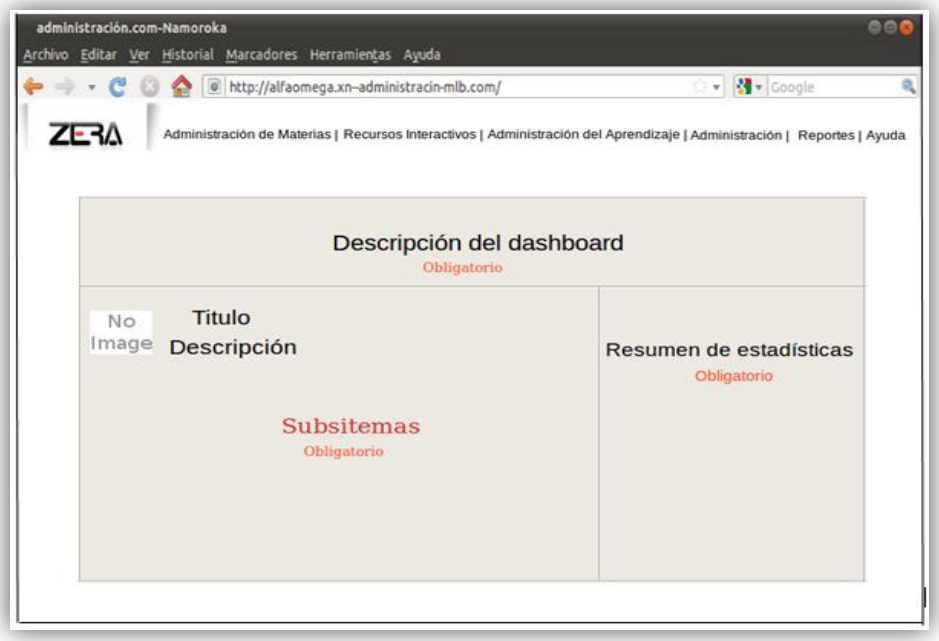


Imagen 3 *Patrón de Interfaz para los dashboard de Administración*

Como se puede observar en la imagen No.3 los dashboard de Administración van a estar divididos en tres secciones, primeramente tenemos una sección donde estará la descripción del dashboard, el cual va estar formado por un título y una simple explicación al usuario del contenido del mismo. Tienen además los subsistemas, como su nombre lo dice se tendrá acceso a cada uno de los subsistemas pertenecientes al módulo. Los subsistemas estarán compuestos por un título llevando el nombre que este representa, una

imagen que lo caracteriza y una descripción del contenido del mismo para aquellos usuarios que no tengan un conocimiento de lo que en él se aborda.

Por último tienen una sección donde se muestra un resumen estadístico de los elementos pertenecientes en la plataforma, este estará formado primeramente por un título general, el campo de información del cual se desea saber el dato y por último un número que representa el dato cuantitativo.

Ejemplo de los patrones en los dashboard

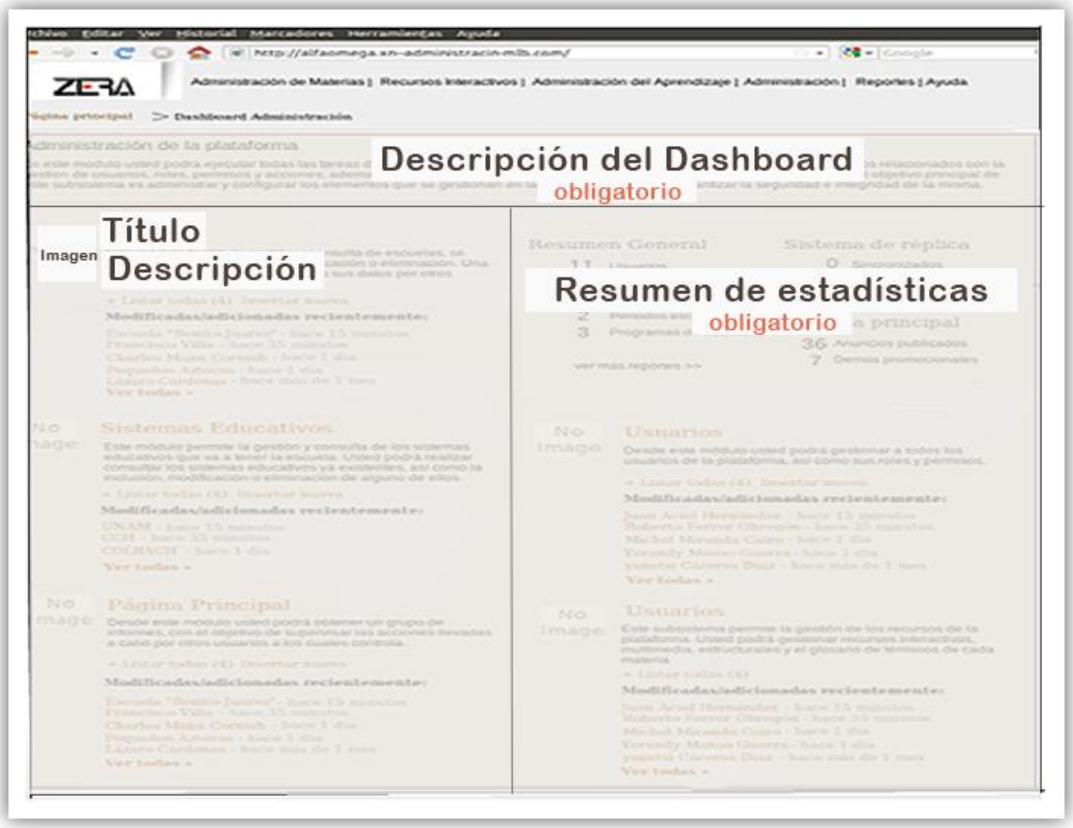


Imagen 4 *Ejemplo del patrón de Interfaz para los dashboard de Administración*

3.3.2- Subsistema Hiperentornos

Este subsistema es la base principal de la actividad académica, todo lo gestionado a través del resto de los subsistemas confluye en un Hiperentorno. Están compuestos por seis (6) módulos básicos y diversos servicios informáticos, los módulos son los siguientes; Contenidos, Prácticas, Portafolio, Biblioteca, Tareas, Docente, y para algunas materias, Simuladores, Graficadores o Laboratorios Virtuales. Al igual que las aulas virtuales, los Hiperentornos presentan servicios y funcionalidades que permiten el aprendizaje a distancia y satisface las necesidades de docentes y estudiantes de forma síncrona y asíncrona, facilitando la atención personalizada.

En los Hiperentornos estarán presentes los home page, los home secciones, las tablas y los tabs, siendo estos los elementos que predominan en el desarrollo del subsistema. A continuación se muestra un ejemplo del patrón de interfaz para los home page de Administración. Para consultar el resto de los patrones correspondientes a cada elemento y sus descripciones (Ver Anexo 8).

Patrón de interfaz para los home page en Hiperentornos

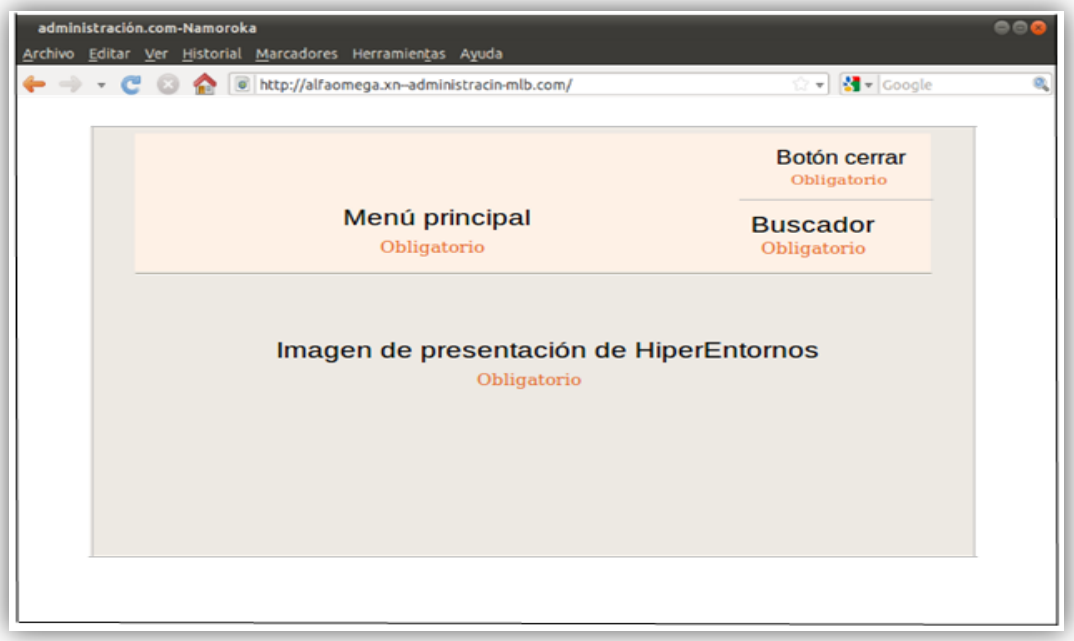


Imagen 5 *Patrón de interfaz para los home page de Hiperentornos.*

La página principal de Hiperentornos es una de las vistas más importantes ante el usuario porque ella es la que brinda la primera impresión de todo el contenido que contendrá este subsistema, debido a esto es necesario que se encuentre de una forma organizada y bien estructurada.

El patrón de interfaz que formará la misma va a estar compuesto primeramente por un botón cerrar en su esquina superior izquierda, el cual tendrá la misión de cerrar la página en que se encuentre el usuario, regresándolo nuevamente a la página principal.

Tendrá además un menú principal con el objetivo de permitirle al usuario desplazarse dentro de la plataforma en los módulos que este desee. En la esquina superior derecha contendrá un buscador con el objetivo que los usuarios puedan buscar de forma rápida cualquiera información que deseen. El resto de la página va estar representada por una imagen que es la que caracteriza el módulo en que se encuentren, como se puede observa en la imagen No 6.

Ejemplo de los patrones en los home page

Imagen 6 *Ejemplo del patrón de interfaz para los home page de Hiperentornos*

3.4- Estructura del código fuente para los subsistemas de la plataforma ZERA

Subsistema Administración

Para una mejor organización y estructuración de cada uno de los elementos pertenecientes a la plataforma se lleva cabo un orden de archivo en el que estarán almacenados de la mejor forma cada uno de los códigos que aquí se encuentran. El subsistema Administración contará con una carpeta donde en su interior tendrá otra además por cada elemento del mismo, en este caso los dashboard, las listas y los formularios. Cada una de ellas contendrá un documento pdf con los patrones creados como guía para la utilización de los programadores de la plataforma, una maqueta que sirviera de apoyo a la hora de querer realizar cualquiera de estos elementos, simplemente se utiliza y solo es necesario hacer los cambios a gusto del mismo. Esta carpeta contendrá un archivo HTML con la estructuración realizada, uno CSS con los estilos brindados y una carpeta con cada una de las imágenes utilizadas (Ver imagen No 7).

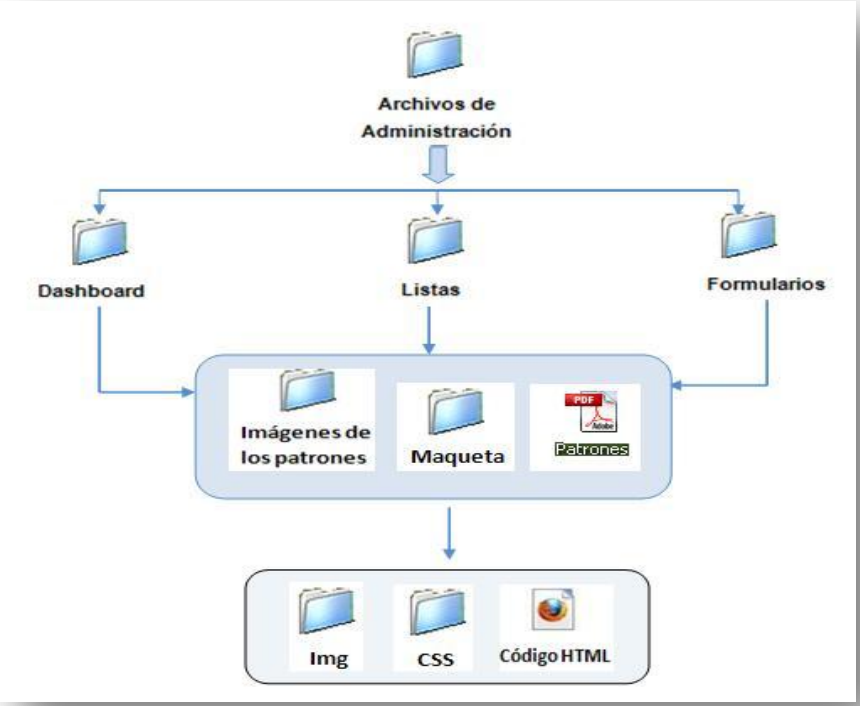


Imagen 7 Organización de los elementos pertenecientes al subsistema Administración.

Subsistema Hiperentornos

El subsistema Hiperentornos tendrá la misma estructuración que la explicada anteriormente en Administración es su caso contará con los home page, los home secciones, las tablas y los tabs. (Ver imagen No 8).

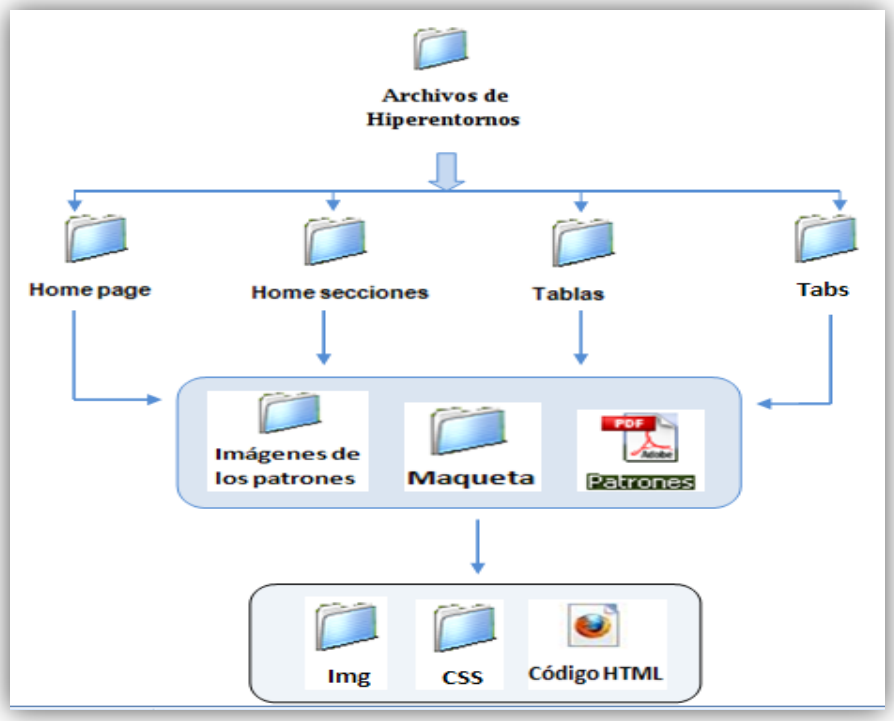


Imagen 8 *Organización de los elementos pertenecientes al subsistema Hiperentornos.*

3.5-Conclusiones Parciales

Durante el desarrollo de este capítulo se presentó una propuesta de solución a las dificultades existentes en la arquitectura de presentación de la plataforma ZERA donde se muestran los patrones de interfaz. Estos establecerán la guía a seguir por los desarrolladores a la hora de construir cada uno de los elementos pertenecientes a la aplicación. Las maquetas ofrecerán un código de ejemplo que facilitará en gran medida el trabajo del desarrollador.

Capítulo 4 Descripción y evaluación del administrador de temas y de la arquitectura propuesta

4.1-Introducción

En el presente capítulo se realiza una propuesta del sistema basado en los conceptos identificados en el modelo de dominio, se muestra además la concepción general al análisis y el diseño del sistema propuesto, se exponen las características de los patrones de interfaz utilizados y finalmente se documenta todo el proceso de implementación y prueba de los elementos identificados durante la realización del diseño, seleccionado el tipo de prueba que se va a utilizar y describiendo los mismos.

4.2-Descripción del sistema

Estudio de temas en los CMS Drupal y WordPress

En aplicaciones web actuales como es el caso de los CMS (Content Management System) como Drupal y WordPress, los temas se encuentran dentro del directorio themes, en la sección de Administración de Temas de cada CMS, se encuentra una imagen pequeña o screenshot de todos los temas que se encuentran activos dentro del directorio, así como el nombre y una descripción del mismo, para que el usuario identifique el tema y seleccione el de su preferencia. Las siguientes imágenes muestran cómo lo realizan Drupal en su más reciente versión 7.0 y WordPress en su versión 3.1.

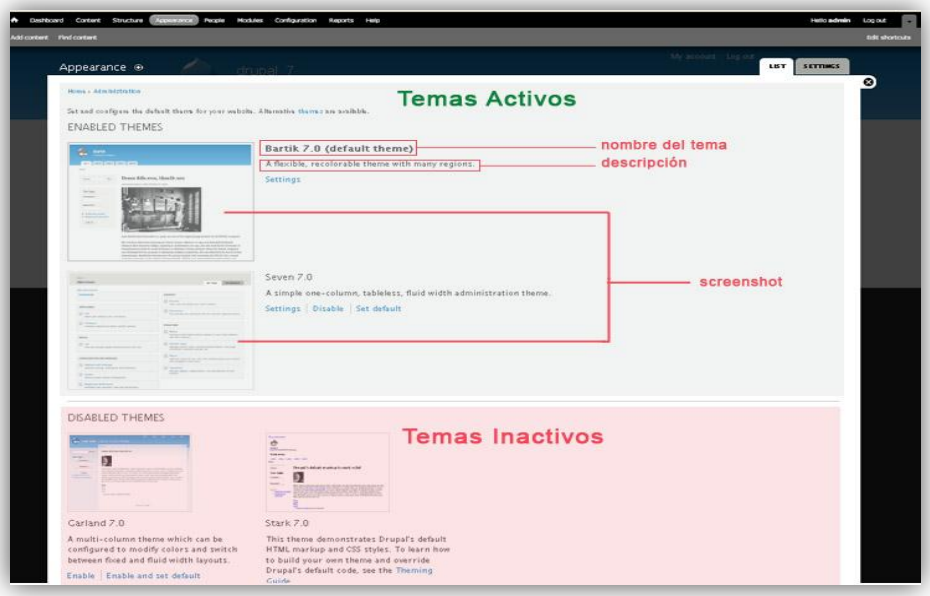


Imagen 9 *Muestra la estructura de la administración de temas en Drupal v.7.0.*

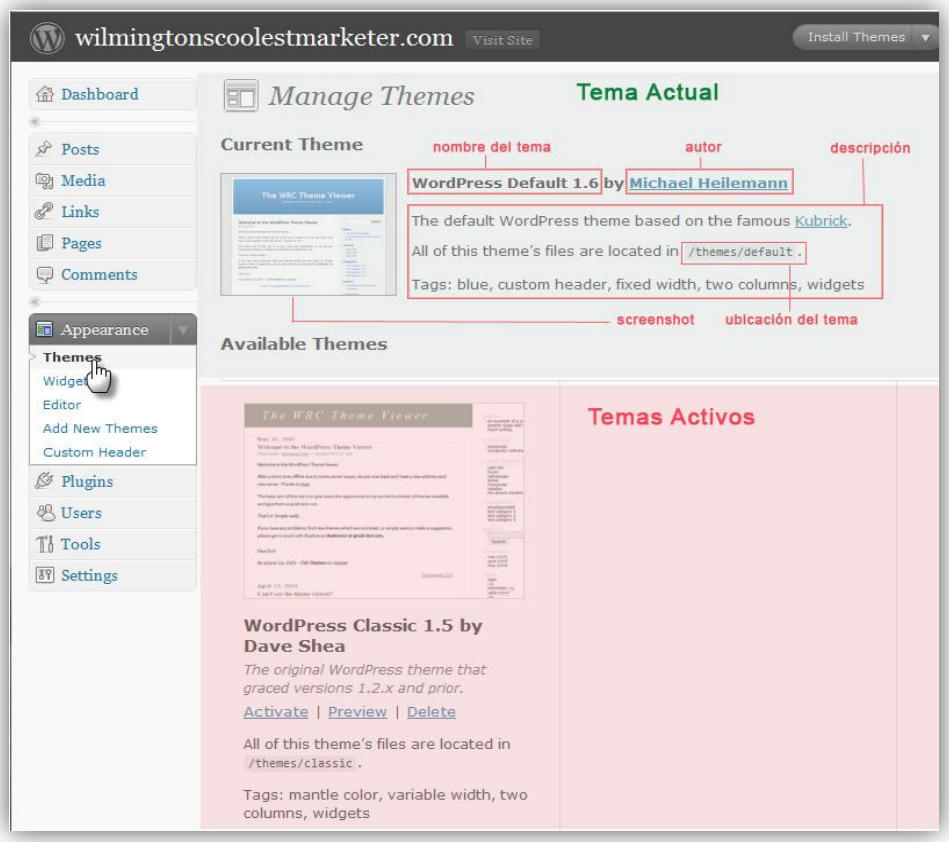


Imagen 10 *Muestra la estructura de la administración de temas en WordPress v3.1.*

Estructura de un Tema en la Plataforma ZERA

La construcción de un Tema se realiza a partir de la descarga de un **Tema Base** el cual se encuentra dentro de: *content.php/subject/themes.php* donde se muestran los temas activos y la opción para subir un nuevo tema a la Plataforma. Este tema base es modificado en dependencia de las especificaciones o necesidades del nuevo tema. El usuario, reemplazaría las imágenes y modificaría los archivos **.ini** (que tiene el nombre y la descripción) además de los estilos contenidos dentro de las hojas de estilo css: **principal** y **modules**.

Los temas en la Plataforma se encuentran alojados en el directorio: *web/uploads/themes/tema en cuestión*; los cuales tienen la siguiente estructura interna:

- ✓ Imagen screenshot.png (pequeña visualización del tema).
- ✓ Archivo description.ini (contiene dentro el nombre y la descripción del tema).
- ✓ Carpeta styles (contiene las hojas de estilo propios del tema: *principal.css* y *modules.css*).
    - ✓ Hoja de estilo principal.css (establece aspectos propios del tema como imágenes de fondo, tipografía y colores).

- ✓ Hoja de estilo modules.css (establece aspectos propios de los módulos de los temas).

- ✓ Carpeta images (contiene las imágenes propias del tema: icons, pages, scroll, common).
  - ✓ Carpeta icons (contiene las imágenes representativas de cada módulo del hiper-entorno).
  - ✓ Carpeta pages (contiene las imágenes representativas del paginado del tema).
  - ✓ Carpeta scroll (como su nombre lo indica contiene las imágenes del scroll del tema).
  - ✓ Carpeta common (contiene las imágenes representativas del tema).
- ✓ Archivo theme_content.html (contiene un listado de cada elemento que debe tener el tema para su correcta visualización, para que el usuario pueda chequear si le falta algún archivo a la hora de construir el tema a partir del base).

4.2.1- Modelo de dominio

El Modelo de Dominio es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Es decir, un diagrama con los objetos que existen relacionados con el proyecto que vamos a cometer y las relaciones que hay entre ellos. Representa clases conceptuales del dominio del problema. Se dice que es estática porque no representa la interacción en el tiempo de los objetos, sino que representa una visión parada de las clases y sus interacciones. Estos se describen mediante diagramas de UML, especialmente diagramas de clases, los cuales muestran a los clientes, usuarios y desarrolladores las clases del dominio y cómo se relacionan unas con otras mediante asociaciones.

Su principal objetivo es ayudar a los implicados en el desarrollo del producto a utilizar un vocabulario común que posibilite una mejor comprensión entre ellos, para poder entender y describir las clases más importantes dentro del contexto donde se ubica el sistema. Además permite y facilita el levantamiento de requisitos así como definir los procesos y roles más significativo.

4.2.2-¿Por qué hacer modelo de dominio?

Debido a que no se tiene definido de una forma clara los procesos del negocio se decide desarrollar un modelo de dominio con el propósito de ayudar a los usuarios, clientes, desarrolladores y demás interesados a utilizar un vocabulario común y de este modo entender el contexto en que se encuentra el sistema. No es factible realizar modelo de negocio con las condiciones existentes pues se es necesario un conjunto de datos manipulados por un conjunto de tareas en las que participan varios representantes del negocio de acuerdo a un flujo de trabajo, por lo que se hace necesario describir el problema de una forma distinta realizándose así el modelo de dominio también conocido como modelo conceptual.

Además el modelo de dominio puede ser tomado como el punto de partida para el diseño del sistema pues al realizar una programación orientada a objetos como es el caso, el mapa de conceptos de este

modelo constituye una primera versión del sistema imitando así en alguna medida la realidad y cuya función principal es ayudar a comprender el problema a tratar. Con este se logrará un glosario o diccionario de clases que sirva como común denominador a todos los componentes del sistema, incluyendo a las personas involucradas a lo largo del desarrollo.

4.2.3-Identificación de los conceptos del dominio

- ✓ **Administrador central:** Persona encargada de subir un nuevo tema a la plataforma.
- ✓ **Temas:** Contenidos específicos que se abordarán en la plataforma.
- ✓ **Hiperentornos de Aprendizaje:** Es la combinación de diferentes tipologías de software, sustentadas en una tecnología hipermedia.
- ✓ **Módulos:** Es un conjunto de elementos fundamentales que componen el producto.

4.2.4-Diagrama del modelo del dominio

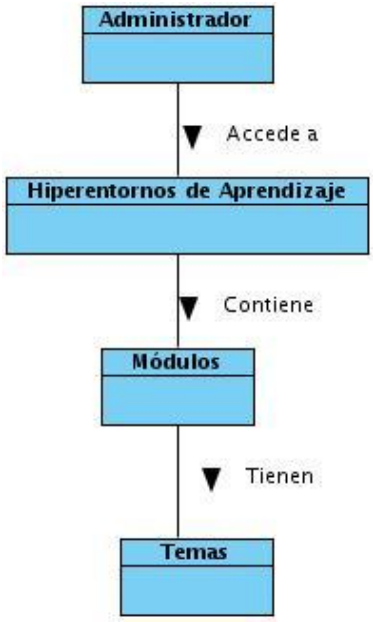


Imagen 11 *Modelo de dominio.*

4.3-Descripción del sistema propuesto

La plataforma ZERA está dividida por dos subsistemas Administración e Hiperentornos. Este último está integrado por temas, dada esta situación se propone construir un sistema de administración de temas que automatice el trabajo con los temas de la plataforma. El administrador de temas automatizará el trabajo una vez creado el tema por el usuario a partir de un tema base, subir el tema y listar todos los que se encuentren activos en la plataforma, mostrando de cada uno de ellos: el nombre, descripción, lugar en el

que se encuentran sus archivos y la fecha en la que fue subido a la plataforma. De esta manera, cuando se crea una materia, se le asocia a la misma un tema de los que se encuentran activos en la plataforma.

4.3.1-Especificación de los requerimientos del sistema

Requisitos Funcionales

Son capacidades o condiciones que el sistema debe cumplir. En la realización de los casos de uso del negocio, se obtienen las actividades que serán objeto de automatización. Estas actividades no son exactamente los requisitos funcionales, pero sí son el punto de partida para identificar qué debe hacer el sistema. Los requisitos funcionales se mantienen invariables sin importar con que propiedades o cualidades se relacionen.

RF 1-Subir nuevo tema

- ✓ **RF 1.1-Subir un nuevo tema a la plataforma:** El administrador central tiene la posibilidad de subir un nuevo tema a la plataforma.

RF 2- Listar temas

- ✓ **FR 2.1-Listar todos los temas existentes en la plataforma**: Se le brinda la posibilidad al usuario con el rol de administrador central de poder listar todos los temas existentes en la plataforma.

Requisitos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable, Los requisitos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con toda la funcionalidad requerida, las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

Existen múltiples categorías para clasificar a los requisitos no funcionales, siendo las siguientes representativas de un conjunto de aspectos que se deben tener en cuenta, aunque no limitan a la definición de otros.

RNF de Software:

- ✓ Las computadoras debe tener instalado el navegador Mozilla Firefox 3.0 o superior, Internet Explorer 7 o superior, Google Chrome 6.0 o superior, Opera 9 o superior, Safari 4.0 o superior.

RNF de Hardware:

- ✓ Procesador Pentium 233 MHz (recomendado 500 MHz o mayor).

- ✓ 512 MB de RAM o superior.
- ✓ 1 GB de espacio en disco duro.
- ✓ Soporte de video que admita resolución de al menos 800x600 y 24 bits.
- ✓ Dispositivo de red de al menos 10 Mbits.

RNF de Diseño e Implementación:

- ✓ Lenguaje de programación: PHP 5.3.
- ✓ El marco de trabajo base de desarrollo que se utilizará es: Symfony 1.4.11.
- ✓ Como IDE se empleará NetBeans 6.9.
- ✓ Como servidor Web se explotará Apache 2.
- ✓ El SGBD deberá ser PostgreSQL 8.4.

RNF de Apariencia o Interfaz Externa:

- ✓ El diseño de las interfaces debe ser sugerente al usuario.
- ✓ El sistema proporcionará claridad y correcta organización de la información, permitiendo la interpretación correcta e inequívoca de esta.
- ✓ Debe contener un diseño sencillo, con pocas imágenes y gráficos para acelerar la velocidad de respuesta del sistema.

RNF de Seguridad y Confiabilidad:

- ✓ Seguridad de acceso y administración de usuarios: otorgamiento de privilegios y roles, asignación de perfiles. Los niveles de acceso están determinados por los diferentes roles válidos dentro de la plataforma.
- ✓ Garantizar que la información sea editada únicamente por quien está autorizado y posea permisos para ello.
- ✓ Verificación sobre acciones irreversibles (mensajes de eliminaciones).
- ✓ Cuando un usuario, sin importar su rol, permanece inactivo durante 20 minutos, se cierra la sesión automáticamente.

RNF de Usabilidad:

- ✓ El sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente Web en sentido general.
- ✓ El sistema debe tener acceso al menú general desde cualquiera de sus páginas.
- ✓ Se deben mostrar las rutas de acceso según la navegación que tenga el usuario.
- ✓ Las rutas de acceso deben tener vínculos a las secciones que muestran.
- ✓ Los elementos gráficos como los iconos deberán contar con un mensaje flotante que señalen el tipo de recurso al que se refiere.

RNF de Portabilidad:

- ✓ El sistema podrá ser utilizado bajo cualquier Sistema Operativo.

RNF Funcionalidad:

- ✓ Reducir al mínimo el tiempo en que carga la plataforma.

RNF Rendimiento:

- ✓ Tiempos de respuestas rápidas al igual que la velocidad de procesamiento de la información.

RNF Legales, Derecho de Autor y otros:

- ✓ Una vez terminado el producto, el sistema debe ser sometido a una evaluación y certificación por parte del cliente del mismo.

RNF de Disponibilidad:

- ✓ Los usuarios del sistema deben tener acceso (según sus permisos) en todo momento a la información solicitada.

4.4- Definición de los actores y casos de uso del sistema

4.4.1-Actores del sistema

Un actor del sistema representa un tipo particular de usuario del sistema más que un usuario físico, ya que varios usuarios físicos pueden realizar el mismo papel en relación al negocio, o sea, ser instancias de un mismo actor. Cada trabajador del negocio (inclusive si fuera un sistema ya existente) que tiene actividades a automatizar es un candidato a actor del sistema.

Actor	Descripción
Administrador Central	Usuario que gestiona, personaliza y asigna el sistema educativo a las escuelas, gestionando también todos los procesos que conforman al mismo.

Tabla 1 *Descripción de los actores del sistema.*

4.4.2-Definición de los casos de uso del sistema

Casos de uso	Nombre
CU-1	**Subir nuevo tema**

CU-2	Listar temas

Tabla 2 Casos de Uso del sistema.

4.5-Diagrama de casos de uso del sistema

Un diagrama de casos de uso del sistema representa gráficamente a los procesos y su interacción con los actores.

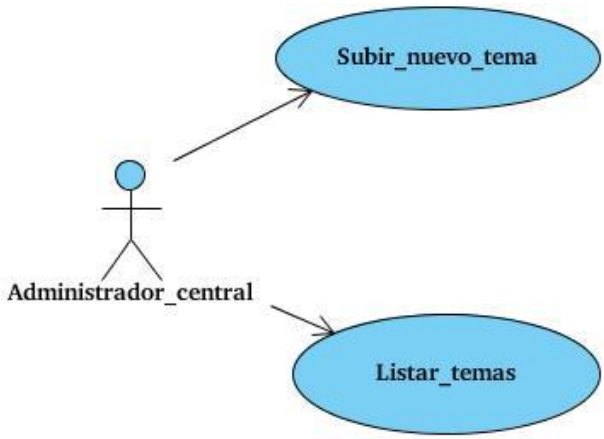


Imagen 12 Diagrama de casos de uso del sistema.

4.5.1-Descripción de los casos de uso del sistema

Para un buen entendimiento de las funcionalidades asociadas a cada caso de uso, no es suficiente con la representación gráfica representada anteriormente por lo que es necesario realizar la descripción de cada uno de ellos con el objetivo de mostrar un mejor acercamiento a lo que se desea realizar.

CU Subir nuevo tema

Caso de uso	Subir nuevo tema
Actores	Administrador central
Resumen	El caso de uso inicia cuando el actor selecciona un nuevo tema para subir a la plataforma. El administrador selecciona el tema, activa el botón para subir el tema, el sistema le da la posibilidad de examinar donde se encuentra el nuevo tema, selecciona el tema a insertar, activa el botón subir y la aplicación

	inmediatamente inserta este nuevo tema en el sistema, finalizando así el caso de uso.
Precondiciones	
Referencias	RF 1
Flujo normal de eventos	
Acciones del actor	**Respuesta del sistema**
1. El caso de uso inicia cuando el actor escoge la opción de subir un nuevo tema a la plataforma.	
	2. Muestra el administrador de plantilla dando la posibilidad de examinar y buscar el tema que desea insertar.
3. Selecciona el tema a insertar ejecutando el botón subir	
	4. El sistema muestra un nuevo tema a la plataforma.
	5. El caso de uso termina.
Flujos Alternos	
***.a Selecciona la opción de cerrar o cancelar la vista actual en cualquier momento.**	
Acciones del actor	**Respuesta del sistema**
	*.a.1 Regresa a la vista anterior.
	*.a.2 El caso de uso termina.

Tabla 3 *Descripción textual del caso de uso Subir nuevo tema.*

CU Listar temas

Caso de uso	Listar temas
Actores	Administrador central
Resumen	El caso de uso inicia cuando el actor presiona el botón de listar temas para ver los temas existentes en la plataforma. El sistema brinda la posibilidad de ver todos los temas que tiene la plataforma en ese momento activo, el actor presiona el botón que realiza esta función e inmediatamente estos se muestran.
Precondiciones	Debe existir algún tema activo en la plataforma
Referencias	RF 2

Flujo normal de eventos	
Acciones del actor	**Respuesta del sistema**
1. El caso de uso inicia cuando el actor presiona el botón listar temas para ver los temas activos en ese momento en la plataforma.	
	2. El sistema verifica que exista algún tema activo en la plataforma. 3. Muestra los temas existentes. 4. El caso de uso se termina.

Flujos Alternos	
***.a Selecciona la opción de cerrar o cancelar la vista actual en cualquier momento.**	
Acciones del actor	**Respuesta del sistema**
	*.a.1 Regresa a la vista anterior.

	*.a.2 El caso de uso termina.
2.a El sistema no tiene ningún tema activo	
Acciones del actor	**Respuesta del sistema**
	2. a.1-No muestra nada al usuario.

Tabla 4 *Descripción textual del caso de uso listar tema.*

4.6-Modelo de análisis

El objetivo de la realización del modelo de análisis es una mejor comprensión, un entendimiento más preciso de los requisitos y una descripción de los mismos donde sea fácil de mantener y que ayude a estructurar todo el sistema, incluyendo su arquitectura. Durante el análisis, se analizan los requisitos que fueron descritos en la captura de requisitos, refinándolos y estructurándolos. Aunque en el modelo del análisis hay un refinamiento de los requisitos, no se tiene en cuenta el lenguaje de programación a usar en la construcción, la plataforma en la que se ejecutará la aplicación, los componentes prefabricados o reusables de otras aplicaciones, entre otras características que afectan al sistema, ya que el objetivo del análisis es comprender perfectamente los requisitos del software y no precisar cómo se implementará la solución.

4.6.1-Diagrama de clases del análisis

El Diagrama de Clases del Análisis es un artefacto en el que se representan los conceptos en un dominio del problema. Constituye una vista estática de las clases que conforman el Modelo del Análisis y las asociaciones entre las mismas. Este va a representar el funcionamiento del mundo real, no de la implementación automatizada del mismo.

Estos diagramas representan las definiciones y relaciones entre las clases. Las clases del análisis se clasifican en Interfaz, de Control o Entidad.

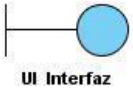
Imagen	Nombre	Descripción
	Interfaz	Modelan la interacción entre el sistema y sus actores.
	Control	Coordinan la realización de uno o unos pocos casos de uso coordinando las actividades de los objetos que implementan la funcionalidad del caso de uso.
	Entidad	Modelan información que posee larga vida y que es a menudo persistente.

Tabla 5 *Estereotipos de las clases del análisis.*

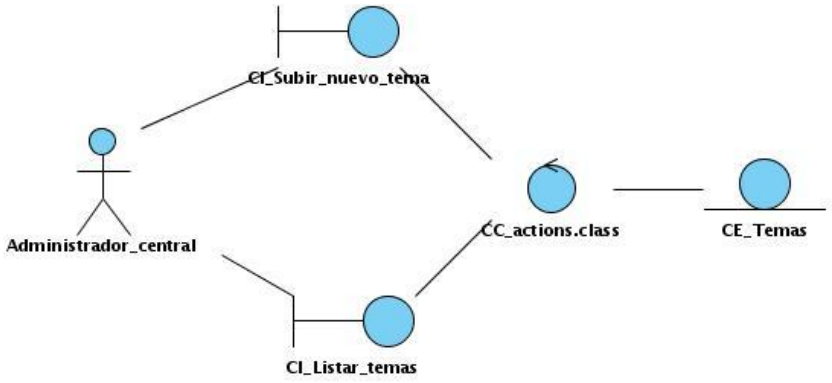


Imagen 13 *Diagrama general de casos de uso del sistema.*

Para observar los diagramas de clases del análisis de los casos de uso existentes independientemente (Ver Anexo 9)

4.6.2-Diagramas de Interacción

Los diagramas de interacción se aplican a varios tipos de diagramas que hacen hincapié en las interacciones entre objetos. Los diagramas de interacción tienen diferentes formas, basadas todas ellas en una misma información subyacente pero resaltando cada una un punto de vista de la misma: diagramas de secuencia, diagramas de colaboración.

Diagrama de colaboración

Un diagrama de colaboración muestra una interacción organizada en tomo a los objetos que efectúan operaciones. El uso de un diagrama de colaboración es mostrar la implementación de una operación. La colaboración muestra los parámetros y las variables locales de la operación, así como asociaciones más permanentes.

A continuación se muestran el diagramas de colaboración del caso de uso Subir nuevo tema, para consultar el resto de los CU asociado al problema (Ver Anexo 10).

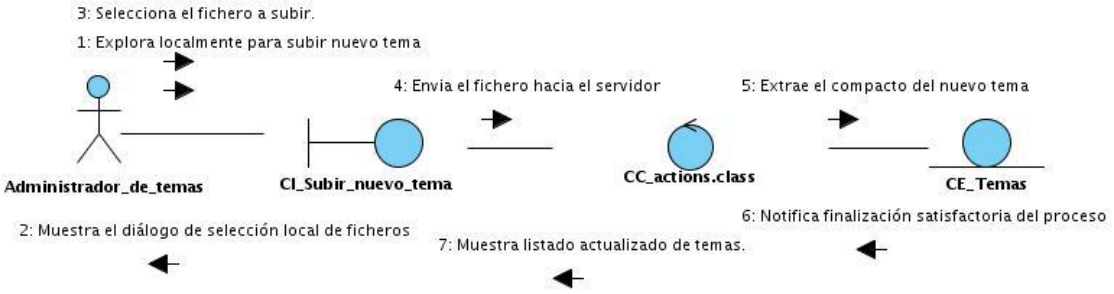


Imagen 14 *Diagrama de colaboración del CU Subir nuevo tema.*

4.7-Modelo de diseño

El Modelo de Diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además, el Modelo de Diseño sirve de abstracción de la implementación del sistema y es de ese modo, utilizado como una entrada fundamental de las actividades de implementación.

Consiste en el refinamiento de los Modelos de Análisis para crear especificaciones adicionales que enriquecen el modelo de análisis con detalles próximos a la implementación.

4.7.1-Patrones de diseño

Para el desarrollo del sistema serán utilizados los patrones de diseño de Symfony, siendo este el framework diseñado para optimizar el desarrollo de las aplicaciones web. Con el uso del framework Symfony brinda la posibilidad de utilizar el conjunto de patrones de diseño que este tiene incluido por defecto en su arquitectura.

Las principales ventajas de hacer uso del patrón son:

- ✓ La conexión entre el Modelo y sus Vistas es dinámica, se produce en tiempos de ejecución, no en tiempos de compilación.
- ✓ La separación del Modelo de la Vista, es decir, separar los datos de la representación visual de los mismos.

- ✓ Facilita agregar nuevos tipos de datos según sea requerido por la aplicación ya que son independientes del funcionamiento de las otras capas.
- ✓ En el estilo arquitectónico MVC se fomenta la escalabilidad del sistema, la seguridad y la reutilización de código.
- ✓ Facilita el mantenimiento en caso de errores.
- ✓ Ofrece maneras más sencillas para probar el correcto funcionamiento del sistema.

El patrón MVC brindado por Symfony en el Administrador de Temas está estructurado primeramente mediante la interfaz principal para el usuario: themes.php del subsistema de Administración. En dicha interfaz se realizan las llamadas a la clase controladora: action.class.php que contiene las funciones: executeThemes() y getDirectoryList() que se utilizan tanto para subir los temas como para listar cada uno de los datos de los temas que se encuentran activos en la plataforma. De esta forma se encuentra separada la vista al usuario de la lógica del negocio.

4.8-Diagrama de clases del diseño

Los diagramas de clases son los más utilizados en el modelo de sistemas orientados a objetos. Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema. Esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas. (33)

Un diagrama de clases del diseño es una representación más concreta que el diagrama de clases del análisis, representa la parte estática del sistema y además representa las clases y sus relaciones. Los estereotipos utilizados para representar las clases en los diagramas son:

Estereotipo	Descripción
 nombre_clase	Representa la página Web que tiene código que se ejecuta en el servidor. Este código interactúa con recursos en el servidor.
 nombre_clase	Representa una página Cliente es una página Web, con formato HTML. Mezcla de datos, presentación y lógica. Son interpretadas por el navegador.
 Formulario	Colección de elementos de entrada que son parte de una página cliente. Se relaciona diretamente con la etiqueta de igual nombre del HTML. Sus atributos son los elementos de entrada del formulario.

Tabla 6 *Estereotipos de las clases del diseño.*

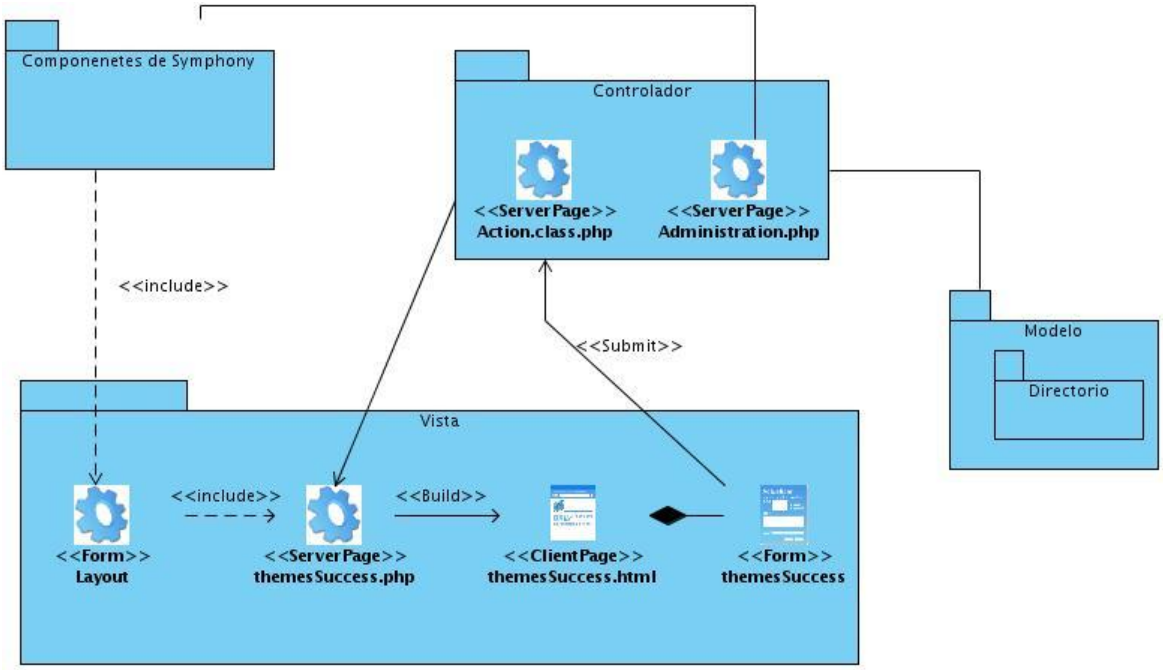


Imagen 15 *Diagrama de clase del diseño de los CU Subir nuevo tema y listar temas.*

4.9-Diseño de la base de datos

El diseño de la base datos es algo fundamental para el buen funcionamiento de cualquier software, este ayuda a comprender la estructura que posee el sistema y sus funcionalidades. El modelo de los datos describe la representación lógica y física de los datos persistentes en el sistema siendo este uno de sus principales objetivos

4.9.1-Clases persistentes

Las clases persistentes representan almacenamientos de datos que persistirán más allá de la ejecución el software. La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Las clases persistentes por lo general tienen como origen las clases clasificadas como entidad porque ellas modelan la información y el comportamiento asociado de algún fenómeno o concepto, como una persona, un objeto del mundo real o un suceso.

En este trabajo no es necesaria la realización de un diagrama de clases persistente debido que no tenemos la existencia de estas clases, es decir la persistencia del administrador de plantillas se encuentra en los temas, estos son los únicos que se mantendrán en el sistema todo el tiempo sin cambios.

4.9.2-Modelos de datos

El modelo de datos es un conjunto de conceptos para diseñar aplicaciones que permiten describir y manipular información que deseamos almacenar en la base de datos. Es por tanto una colección de conceptos bien definidos matemáticamente que ayudan a expresar las propiedades estáticas y dinámicas de una aplicación con un uso de datos intensivo. Son abstracciones que permiten la implementación de un sistema eficiente de base de datos; por lo general se refieren a algoritmos, y conceptos matemáticos. El resultado de un modelado de datos es una representación que tiene dos componentes: las propiedades estáticas se definen en un esquema y las propiedades dinámicas se definen como especificaciones de transacciones, consultas e informes.

En este trabajo no es necesaria la realización de un modelo de datos debido a que no utilizamos tablas de la base datos, el sistema de plantillas utiliza un directorio en el cual contendrá un sistema de carpetas que son las encargadas de almacenar toda la información referente a los temas, estos temas no son eliminados, ni modificados, por lo que no es necesario un sistema de tablas de la base datos con el objetivo de utilizar esa información posteriormente.

4.10-Implementación y prueba

El flujo de trabajo de implementación describe cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo éstos se organizan de acuerdo a los nodos específicos en el modelo de despliegue. En la implementación se comienza con el resultado del diseño y se implementa el sistema en términos de componentes, es decir: ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares. (34)

4.10.1-Diagrama de paquetes

Un diagrama de paquetes muestra cómo un sistema está dividido en agrupaciones lógicas mostrando las dependencias entre esas agrupaciones. El objetivo de estos diagramas es obtener una visión más clara del sistema de información orientado a objetos, organizándolo en subsistemas, agrupando los elementos del análisis, diseño construcción y detallando las relaciones de dependencia entre ellos.

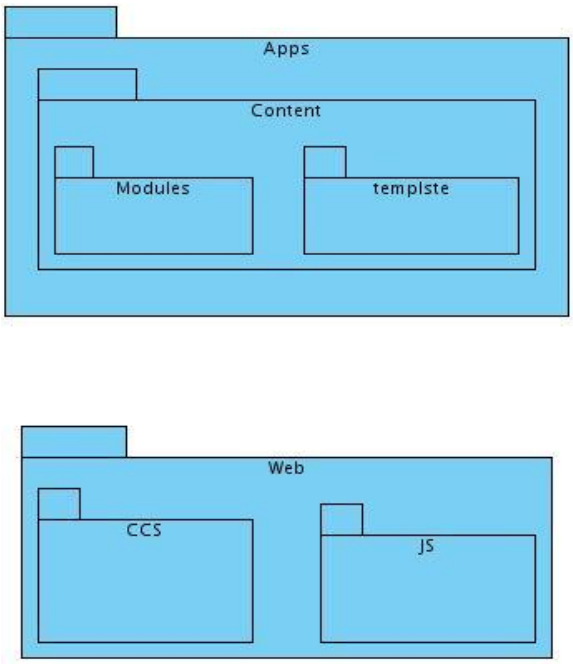


Imagen 16 *Diagrama de paquetes.*

4.10.2-Modelo de implementación

El modelo de implementación describe como los elementos del modelo de diseño, como las clases, se implementan en términos de componentes como ficheros de códigos fuentes, ejecutables, tablas, etc. El modelo de implementación describe como se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en los lenguajes de programación utilizados y de la forma en que dependen los componentes uno del otro.

4.10.2.1-Diagrama de componentes

Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas de Componentes pueden ser usados para modelar y documentar cualquier arquitectura de sistema. Debido a que estos son más parecidos a los diagramas de casos de usos estos son utilizados para modelar la vista estática y dinámica de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema. En él se situarán librerías, tablas, archivos, ejecutables y documentos que formen parte del sistema.

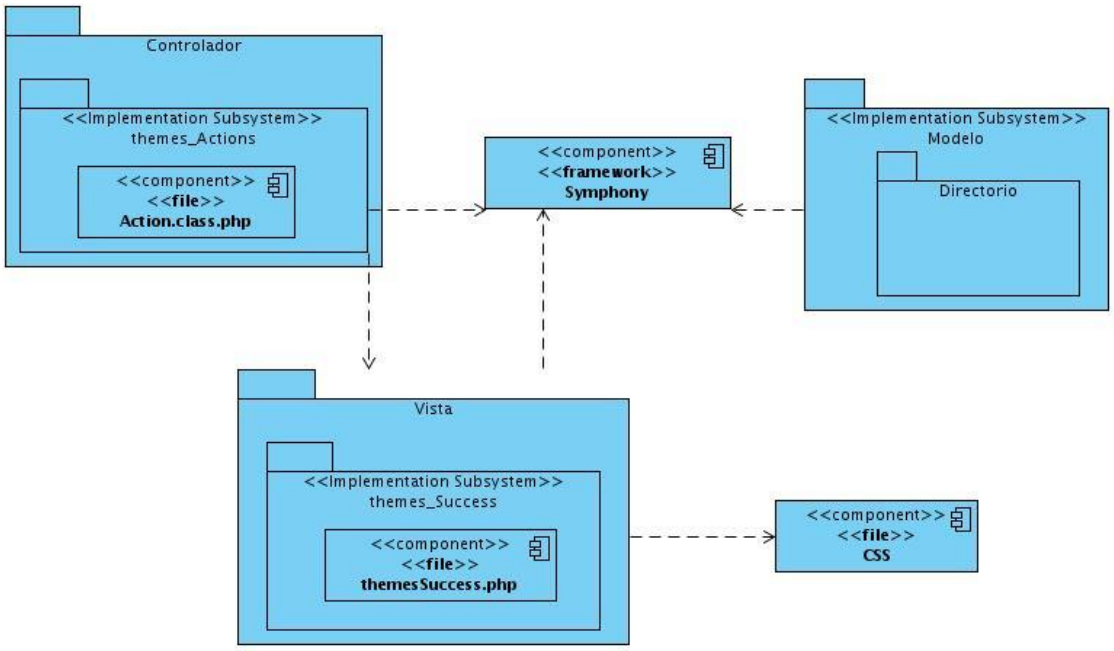


Imagen 17 *Diagrama de componentes.*

4.11-Diagrama de despliegue

Los Diagramas de Despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema entre los componentes de hardware y de software. Estos muestran la disposición física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación.

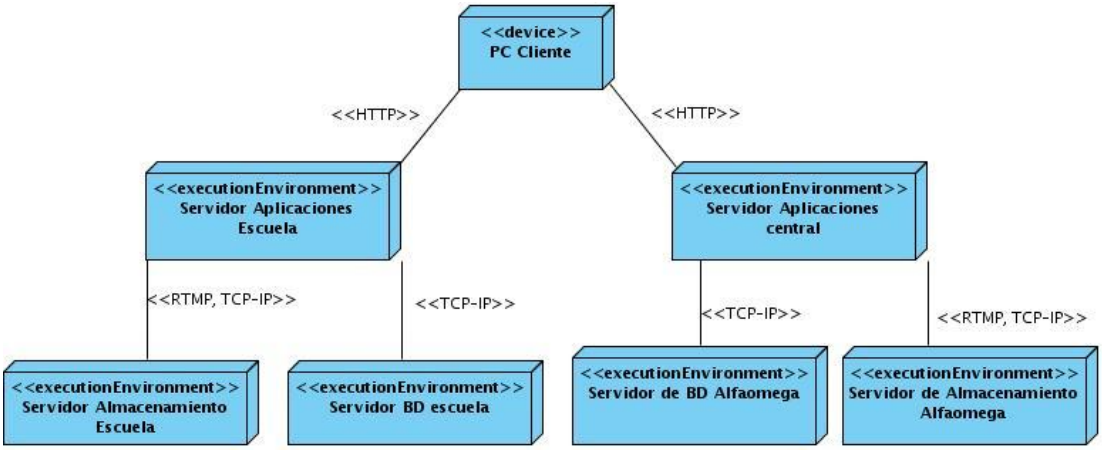


Imagen 18 Diagrama de despliegue

4.12-Pruebas

El flujo de trabajo prueba es uno de los más importantes en la construcción de un software. Durante este proceso se ejecuta el sistema a probar bajo condiciones específicas con el objetivo de relevar el incorrecto o incompleto funcionamiento de mismo, así como los errores de interfaz y rendimiento. Durante este proceso es posible detectar insatisfacción de los requerimientos planteados; todas las actividades y sus resultados son observados y registrado donde se realiza una evaluación del sistema o componente.

4.12.1-Mejoras en el Rendimiento de la Plataforma

Una de las máximas prioridades a la hora de optimizar el rendimiento de la Plataforma es la de reducir las peticiones que ésta hará al servidor una vez que el HTML se haya cargado en el navegador del cliente. Cada archivo de hoja de estilos (CSS), cada archivo Javascript, cada imagen, cada elemento externo en definitiva, se solicita de forma separada y aumenta por tanto la transferencia y el tiempo de carga, esto resulta obvio. Lo que no resulta tan evidente es que ese aumento no es proporcional a la suma del tamaño de los archivos externos, es decir, la carga de 10 elementos de 5 KB no es igual de rápida que la de un elemento de 50 KB. Además de sumar los tiempos de transferencia para cada archivo, deberemos sumar por un lado el tiempo que tarda en realizarse la propia conexión + petición + respuesta (las peticiones son secuenciales, para comenzar cada una, debe esperar que culmine la anterior, o sea, debe esperar por el tiempo que demora el establecimiento de la conexión con el servidor, el tiempo de la petición más el tiempo de retorno que demora la respuesta del servidor).

CSS Sprites

Es una de las técnicas más usadas actualmente en aplicaciones webs, empresas como Google y Yahoo! hacen uso de ella. Consiste en utilizar una sola imagen compuesta por varias y que, haciendo uso de las propiedades background-position de CSS se visualiza cada imagen de forma individual.

Imagen 19 *Uso de los CCS sprites en el Buscador de Google*

La utilización de la técnica de los CSS Sprites tiene principalmente dos ventajas: por un lado aumenta la velocidad en la descarga y renderizado de la Plataforma por parte de los navegadores y por otro lado, se reduce el consumo de recursos del servidor.

Aplicación de la Técnica de Sprites Css en la página resources.php:

Dicha página realizaba 4 peticiones por separado para las imágenes:

Imagen 20 *Figuras de ejemplo en la plataforma*

Con un peso de 1.5 Kb, 2.8 Kb, 1.2 Kb, y 1.2 Kb respectivamente. Aplicando el Sprite Css se unieron las imágenes anteriores en un único archivo:

 Realizando una petición al servidor con un peso de 4.9 Kb.

También se utilizó esta técnica para las imágenes **banner_admin.jpg** y **foot.jpg** las cuales se unieron en el archivo **large.png**.

Imagen 21 *Archivo large.png que agrupa las imágenes foot.jpg y banner_admin.jpg*

Analizando el estado inicial de la Plataforma descrito anteriormente, como resultado general se realizaron 4 peticiones menos para cargar las imágenes, siendo más rápida la carga de la página (de 6.64s a 5.9s). Además de la reducción del tiempo, se redujo el tamaño total de la descarga de las peticiones, de 73 kb a

66.8 KB. Tomando el caso de prueba de 4000 usuarios conectados de manera simultánea, se realizarían 16000 conexiones menos, con una reducción de 24 800 kb ≈ 24 mb. (Ver Anexo 11)

4.12.2-Pruebas de validación

El flujo de trabajo prueba es uno de los mas importante en la construcción de un software. Estas son un conjunto de actividades donde un sistema o componente es ejecutado bajo condiciones o requerimientos específicos. Entre sus principales objetivos esta detectar alguna insatisfacción de los requerimientos planteados; todas las actividades y sus resultados son observados y registrado realizándose una evaluación del sistema o componente. Es importante aclarar que las pruebas no pueden asegurar la ausencia de defectos; solo pueden demostrar que existen defectos en el software y que cada prototipo que se quiera entregar al final de una iteración debe ser probado y evaluado a diferentes niveles.

4.12.2.1-Métodos de prueba

Pruebas de caja negra: Estas pruebas que se llevan a cabo sobre la interfaz del software. O sea, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene.

En otras palabras, la prueba de la caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del software, o sea los casos de prueba pretenden:

- ✓ Demostrar las funciones del software son operativas.
- ✓ Que las entradas se aceptan de la forma adecuada y que se produce el resultado correcto.
- ✓ Determinar errores en la interfaz y las funciones visibles al cliente.

Pruebas de caja blanca: En esta prueba se comprueban los caminos lógicos del software proponiendo casos de prueba que se ejerciten conjuntos específicos de condiciones y/o bucles. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coinciden con el esperado o mencionado. Ellas requieren del conocimiento de la estructura interna del programa y son derivadas a partir de las especificaciones internas de diseño o el código.

Mediante los métodos de prueba de la caja blanca, el ingeniero de software puede obtener casos de prueba que garanticen que:

- ✓ Se ejerciten por lo menos una vez todos los caminos independientes para cada módulo.
- ✓ Se ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsa.
- ✓ Ejecuten todos los bucles en sus límites y con sus límites operacionales.
- ✓ Se ejerciten las estructuras internas de datos para asegurar su validez.

- ✓ Se obtengan los resultados esperados luego de ejecutar una funcionalidad dada.

Para la validación del sistema se realizaron las pruebas de caja negra entre estas los casos de prueba. Los casos de prueba son un conjunto de entradas de pruebas, condiciones de ejecuciones y resultados esperados desarrollados para cumplir un objetivo en particular o una función esperada. No solo tienen que ser escritos para entradas válidas y esperadas, sino también condiciones no válidas e inesperadas.

Caso de prueba para el CU Subir nuevo tema

Descripción general

El caso de uso inicia cuando el actor selecciona un nuevo tema para subir a la plataforma. El administrador selecciona el tema, activa el botón para subir el tema, el sistema le da la posibilidad de examinar donde se encuentra el nuevo tema, selecciona el tema a insertar, activa el botón subir y la aplicación inmediatamente inserta este nuevo tema en el sistema, finalizando así el caso de uso.

Condiciones de ejecución

Debe haberse generado el escritorio de trabajo.

SC1 Subir nuevo tema

ID del escenario	Escenario	Respuesta del sistema	Flujo central
EC1	EC 1.1 El actor escoge la opción de subir un nuevo tema a la plataforma.	Muestra el administrador de plantilla dando la posibilidad de examinar y buscar el tema que desea insertar.	Interactive Resources/Subjets/ Themes
EC 2	EC 2.1 Selecciona el tema a insertar ejecutando el botón subir	El sistema muestra un nuevo tema a la plataforma.	Interactive Resources/Subjets/ Themes

Tabla 7 *Tabla de secciones del caso de prueba Subir tema.*

Caso de prueba Listar temas (Ver Anexo 12)

No. NC	Requisitos Funcionales	No Conformidad	Etapa de la iteración	Descripción	Estado con respecto a la solución
1	CP_Subir_nuevo_tema	Se encontró una NC en el escenario EC 2.1 debido que se selecciona el tema a subir a la plataforma, lo carga pero no lo muestra.	1ra Etapa	Alta	Resuelta y aprobada
2	CP_Subir_nuevo_tema	Se encontró una NC en el escenario 2.1, muestra el tema en la plataforma y muestra el mensaje de que debe seleccionar un tema.	1ra Etapa	Media	Resuelta y aprobada
3	CP_Subir_nuevo_tema	Se encontróó una NC en el escenario 2.1, cuando el usuario selecciona y sube el tema, el mismo no se descompacta	2da Etapa	Alta	Resuelta y aprobada

		en el directorio.			

Tabla 8 *No conformidades.*

4.13- Conclusiones Parciales

El modelo de dominio realizado ayudó a comprender el entorno con que se relacionan las funcionalidades. Los requisitos funcionales y no funcionales obtenidos permitieron definir las capacidades y las cualidades que deben tener las funcionalidades. Los diagramas de componentes y despliegue permitieron conformar lo que se conoce como modelo de implementación, describiendo cómo los elementos del modelo del diseño se implementan en términos de componentes y además cómo se organizan. La realización de diferentes pruebas validó que las funcionalidades desarrolladas satisfacen los requisitos especificados y además arrojaron que aunque se detectaron varias no conformidades, las mismas ya están completamente arregladas y las funcionalidades cumplen con la calidad requerida.

Conclusiones Generales

A lo largo del desarrollo del presente trabajo de investigación se determinaron los estilos y patrones de interfaz que se utilizarán en el desarrollo de la Plataforma ZERA y que principalmente servirán de guía para la implementación de los nuevos elementos de la arquitectura de presentación.

El análisis de las herramientas, metodologías y lenguajes posibilitaron la realización de los modelos y diagramas necesarios para un buen entendimiento. Permitió identificar, diseñar e implementar cada una de las funcionalidades de la aplicación.

Se realizó además una propuesta de arquitectura basada en temas con todos sus elementos físicos incluidos garantizando un mayor nivel de configuración y personalización por parte del usuario.

Recomendaciones

Una vez concluido el desarrollo de este documento cumpliéndose con los objetivos propuestos se hacen las siguientes recomendaciones:

- ✓ Mantener el estudio abordado en la presente investigación en constante actualización en correspondencia a los nuevos avances tecnológicos referentes a aplicaciones web a nivel mundial actuales.
- ✓ Potenciar el uso de talleres o conferencias impartidas a los desarrolladores en cuanto al uso de los patrones de interfaz y maquetas para lograr un mayor adiestramiento en su utilización.
- ✓ Realizar una segunda iteración para perfeccionar el administrador de temas en la plataforma, enfocado principalmente a mejorar la previsualización de cómo quedarían los temas una vez que sean asociados a una materia.

Referencias Bibliográficas

1. **Ortega, Alvaro.** *Interfaces de Usuario Avanzadas.* Madrid : Universidad Autonóma de Madrid, 2008.

2. **EcuRed.** EcuRed. [En línea] 19 de diciembre de 2010. [Citado el: 13 de enero de 2011.] http:www.ecured.cu/index.php/Arquitectura_Cliente_Servidor.

3. **Lago, Ramiro.** *Patrones de diseño de software.* abril 2007.

4. **fabien Potencier, Francois zaninatto.** *Symfony La guía definitiva.* 2010.

5. **Fanjul, Alejandro.** *Arquitectura de tres capas.* 2010.

6. **Manzón, Lic. Ricardo.** *Programación III.* 2010.

7. Hooping.net. [En línea] Universidad del Saber , 2008. [Citado el: 16 de enero de 2011.] http://www.hooping.net/faq-historia-diseno-web.aspx.

8. **Liyan Suárez, Rafael Hernández, Alicia Gutiérrez,Sheira Rosas.** *Las herramientas 2.0 en la universidad.* 2011.

9. **Teisidor, Sergio Dote.** nestes 2.0. [En línea] 22 de julio de 2008. [Citado el: 13 de enero de 2011.] http://www.neleste.com/category/articulos.

10. **Masadelante.** Masadelante. [En línea] 2011. [Citado el: 17 de enero de 2011.] http://www.masadelante.com/faqs/css.

11. **Bos, Bert.** W3C. [En línea] 8 de diciembre de 2010. [Citado el: 18 de enero de 2011.] http://www.w3.org/Style/CSS/specs#css21.

12. **Perez, J.E.** Introducción a XHTML. [En línea] 2008. [Citado el: 18 de enero de 2011.] http://librosweb.com.

13. —. Introducción a JavaScript. [En línea] 2009. [Citado el: 18 de enero de 2011.] http://librosweb.es.

14. **NetBeans.** Netbeans. [En línea] 2011. [Citado el: 18 de enero de 2011.] http://netbeans.org/community/releases/68/index_es.html.

15. **tufunción.** tufunción. [En línea] 20 de noviembre de 2007. [Citado el: 18 de enero de 2011.] http://www.tufuncion.com/zend-studio.

16. **Joanquin Seoane, Jesús M Gregorio, Robles Atenas.** *Atenas.* 2007.

17. **Hooping.net.** Hooping.net. [En línea] 2008. [Citado el: 19 de enero de 2011.] http://www.hooping.net/glossary/adobe-dreamweaver-169.aspx..

18. **García, Alberto.** pixelover. [En línea] 4 de enero de 2008. [Citado el: 12 de enero de 2011.] http://www.pixelovers.com/mejores-frameworks-css-no-uso-51249.

19. **Alvarez, Miguel angel.** desarrolloweb.com. [En línea] 15 de junio de 2009. [Citado el: 15 de enero de 2011.] http://www.desarrolloweb.com/articulos/listado-frameworks-css.html .

20. **Alvarez, Miguel A.** desarrolloweb.com. [En línea] 15 de junio de 2009. [Citado el: 19 de enero de 2011.] http://www.desarrolloweb.com/articulos/listado-frameworks-css.html.

21. **Bernardino, Jepser.** jepserbernardino. [En línea] 2010. [Citado el: 19 de enero de 2011.] http://jepserbernardino.com/idea/960-grid-un-framework-para-..

22. desarrolloweb.com. [En línea] 22 de abril de 2009. [Citado el: 16 de diciembre de 2011.] http://www.desarrolloweb.com/de_interes/simple-framework-css-espanol-1705.html.

23. CREATIVASFERA. [En línea] 5 de mayo de 2011. [Citado el: 16 de diciembre de 2011.] http://www.creativasfera.com/2011/05/15-frameworks-para-desarrollo-web/.

24. masadelante.com. [En línea] [Citado el: 22 de enero de 2011.] http://www.masadelante.com/faqs/que-es-un-navegador.

25. **Manager, Free Download.** Free Download Manager. [En línea] 5 de marzo de 2007. [Citado el: 20 de enero de 2011.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.

26. **Caceres, Ana Mercedes.** *Herramienta case " ArgoUML" .* CUIDADELA DON BOSCO : ESCUELA DE COMPUTACION , 2007. BM040794 .

27. **Infante, Luis.** Business Intelligence Latin America. [En línea] 29 de mayo de 2009. [Citado el: 19 de enero de 2011.] http://www.bi-la.com/profiles/blogs/sobre-las-metodologias.

28. **Jacobson, Ivar, Booch, Grady, Rumbaugh, James.** *El proceso Unificado de Desarrollo de Software.* España : s.n., 1999.

29. **Cortizo Pérez, J.C, Expósito Gil, Ruiz Leyva M.** *eXtreme Programming.* s.l. : AINetSolutions Technical Report, 2003.

30. **Grady Booch, Jim Rumbaugh ,Ivar Jacobson.** *El Lenguaje Unificado de Modelado .* 2006.

31. **Saltares, david.** Sion Dream. [En línea] 2011. [Citado el: 16 de mayo de 2011.] http://siondream.com/blog/desarrollo-informatica/pencil-project-bocetos-de-interfaz/.

32. **Freddie.** CLcristalab. [En línea] 25 de septiembre de 2005. [Citado el: 16 de mayo de 2011.]

33. EcuRed. [En línea] 6 de abril de 2011. [Citado el: 2 de mayo de 2011.] http://www.ecured.cu/index.php/Flujo_de_Trabajo_An%C3%A1lisis_y_Dise%C3%B1o#Identificaci.C3.B3n_de_clases_del_dise.C3.B1o.

34. *Fase de Construcción. Flujo de Trabajo de Implementación. Modelo de implementación .* s.l. : Facultad Regional "Mártires de Artemisa", 2010.

35. **Charris, Eric.** monografías.com. [En línea] 8 de junio de 2010. [Citado el: 14 de enero de 2011.] http://www.monografias.com/trabajos57/mantenimiento-redes-lan/mantenimiento-redes-lan.shtml#xcapas.

36. **WebTaller.** WebTaller. [En línea] febrero de 2010. [Citado el: 14 de enero de 2011.] http://www.webtaller.com/maletin/articulos/tres-capas-presentacion-construccion-interfaces-web-accesibles.php.

37. **jaribia.** Blog de invistigaciones y desarrollo web. [En línea] 20 de agosto de 2009. [Citado el: 13 de enero de 2011.] http://jariviablog.wordpress.com/2009/08/20/que-es-y-para-que-sirve-un-framework-css.

38. *Modelo de Implementación:Diagrama de Componentes y Despliegue.* 2008.

39. **Manzano, Cristina.** *Plataforma Virtual.* Phoenix : s.n., 2010.

40. Centro de Formación Pedagógica. *Centro de Formación Pedagógica.* [En línea] 2007. http://www.cfp.us.es/web/contenido.asp?id=3417.

41. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* España : s.n., 1999.

42. **Millán, Cristy Johana León.** [En línea] 4 de Junio de 2008. [Citado el: 5 de Enero de 2011.] http://misespaciosenlau.blogspot.com/2008/06/hipertexto-el-hipertexto-es-un-sistema.html.

43. **B., Mónica María Agudelo.** AprendeenLínea. [En línea] Agosto de 2006. [Citado el: 5 de Enero de 2011.] http://aprendeenlinea.udea.edu.co/banco/html/plataformaseducativas/.

44. Publicalpha punto com. *Publicalpha punto com.* [En línea] 3 de December de 2008. [Citado el: 5 de Enero de 2011.] http://publicalpha.com/%C2%BFque-es-el-software-educativo/.

45. Dos Ideas. [En línea] [Citado el: 6 de Enero de 2011.] http://www.dosideas.com/wiki/Extreme_Programming.

46. eleZeta . [En línea] [Citado el: 6 de Enero de 2011.] http://elezeta.net/2004/08/27/extreme-programming-xp/.

47. **VALLE, JOSÉ GUILLERMO.** Monografia.com. *Definición arquitectura cliente servidor.* [En línea] 2005. [Citado el: 8 de Enero de 2011.] http://www.monografias.com/trabajos24/arquitectura-cliente-servidor/arquitectura-cliente-servidor.shtml.

48. webtaller.com. [En línea] 2007. [Citado el: 8 de Enero de 2011.] http://www.webtaller.com/construccion/lenguajes/html/lecciones/que-es-xhtml.php.

49. masadelante.com. *Definición de CSS - ¿Qué son las hojas de estilo o cascading style sheets?* [En línea] 2005. [Citado el: 9 de Enero de 2011.] http://www.masadelante.com/.

50. **Pérez, Javier Eguíluz.** *Introducción a Javascript.* 2008.

51. Programación en castellano. [En línea] 2008. [Citado el: 10 de Enero de 2011.] http://www.programacion.net/php.

52. maestros del web. [En línea] 2005. [Citado el: 10 de Enero de 2011.] http://www.maestrosdelweb.com/.

53. **Puebla, Ing. Yoan Arlet Carrascoso y Gómez, Ing. Enrique Chaviano.** GestioPolis. *Propuesta de arquitectura orientada a servicios para el módulo de inventario del ERP cubano .* [En línea] 2009. [Citado el: 11 de Enero de 2011.] http://www.gestiopolis.com/administracion-estrategia/erp-arquitectura-orientada-a-servicios.htm.

54. **Potencier, Fabien.** symfony.es. [En línea] 2010. [Citado el: 10 de Enero de 2011.] http://www.symfony.es/.

55. **Álvarez, Miguel Ángel.** DesarrolloWeb.com. [En línea] 2009. [Citado el: 10 de Enero de 2011.]

56. **Pecos, Daniel.** PostGreSQL vs. MySQL. [En línea] 2006. [Citado el: 11 de Enero de 2011.] http://danielpecos.com/docs/mysql_postgres/x15.html.

57. desarrolloweb.com. *Qué es Oracle.* [En línea] 2006. [Citado el: 12 de Enero de 2011.] http://www.desarrolloweb.com/articulos/840.php.

58. NetBeans. [En línea] [Citado el: 12 de Enero de 2011.] http://netbeans.org/community/releases/68/index_es.html.

59. buenmaster.com. [En línea] 2007. http://www.buenmaster.com/?a=1203.

60. tufunción. *Los mejores IDEs para Php.* [En línea] 2007. http://www.tufuncion.com/ide-php.

61. SES Subsecretaría de Educacion Superior. *SES Subsecretaría de Educacion Superior.* [En línea] [Citado el: 9 de 2 de 2011.] http://ses4.sep.gob.mx/wb/ses/ses_glosario?page=4.

62. **Moodle.** Moodle. *Moodle.* [En línea] [Citado el: 5 de Enero de 2011.] http://moodle.org/about/.

63. Claroline.net. [En línea] 2008. [Citado el: 5 de Enero de 2011.] http://www.claroline.net/.

64. **Sadhea.** Sadhea. *Sadhea.* [En línea] 2009. [Citado el: 5 de Enero de 2011.] http://www.sadhea.rimed.cu/.

65. deGUATE.com. *deGUATE.com.* [En línea] [Citado el: 15 de 2 de 2011.] http://www.deguate.com/infocentros/gerencia/glosario/h.htm.

66. UPtoDOWN. [En línea] [Citado el: 12 de 2 de 2011.] http://argouml.uptodown.com/.