Universidad de las Ciencias Informáticas

Facultad 6

Título: “Aplicación para la generación de libros web”

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor(es): Marcos A. Reyes Medina

Yoandy Doble Herrera

Tutora: Ing. Claudia Nuñez Sanz

La Habana

22 de Junio del 2011

“Año 54 de la Revolución”

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

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

Marcos Antonio Reyes Medina

_____

Firma del Autor

Yoandy Doble Herrera

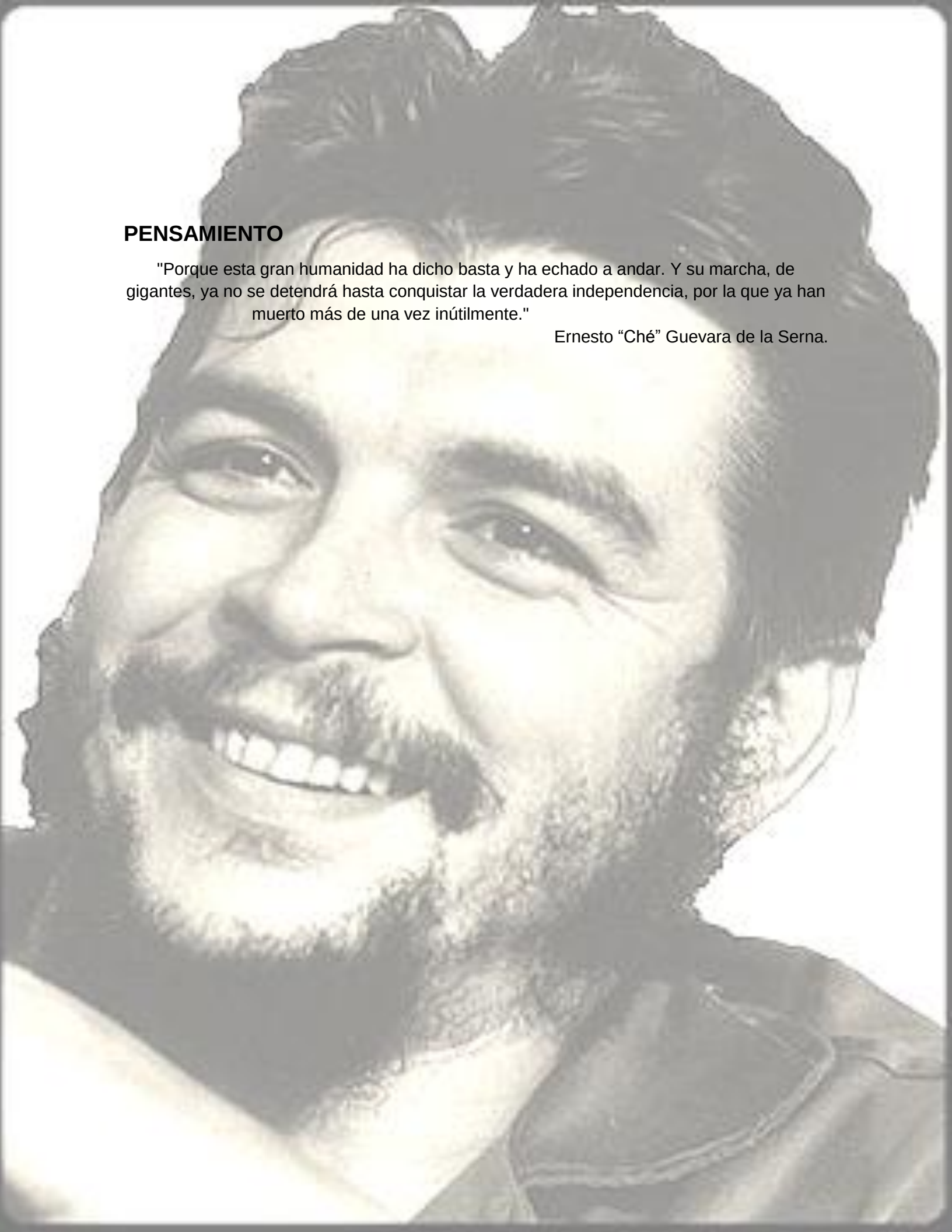
_____

Firma del Autor

Ing. Claudia Nuñez Sanz

_____

Firma del Tutor(a)



PENSAMIENTO

"Porque esta gran humanidad ha dicho basta y ha echado a andar. Y su marcha, de gigantes, ya no se detendrá hasta conquistar la verdadera independencia, por la que ya han muerto más de una vez inútilmente."

Ernesto “Ché” Guevara de la Serna.

Datos de Contacto

Autor (es):

Marcos Antonio Reyes Medina

Universidad de las Ciencias Informáticas

Ciudad de la Habana, Cuba

E-mail: mareyes@estudiantes.uci.cu

Yoandy Doble Herrera

Universidad de las Ciencias Informáticas

Ciudad de la Habana, Cuba

E-mail: ydoble@estudiantes.uci.cu

Tutor (a):

Ing. Claudia Nuñez Sanz

Universidad de las Ciencias Informáticas

Ciudad de la Habana, Cuba

E-mail: cnunez@uci.cu.

AGRADECIMIENTOS

“Los ideales que han iluminado mi camino, y una y otra vez me han infundido valor para enfrentarme a la vida con ánimo, han sido la bondad, la belleza y la verdad”.

Albert Einstein

La Revolución Cubana ha sido el motor impulsor de las ideas latinoamericanas, ha sido sin dudas la consagración y un gran paso en desarrollo político, económico y social de los oprimidos. Ha sido y será el despertar de la primavera; a la Revolución Cubana, sin ti no hubiese sido posible realizar mis sueños.

Al tesoro más valioso que he tenido desde pequeño, a mi guía, mi inspiración, a ese universo infinito que e inigualable que eres. Tus enseñanzas, tu amor, tu optimismo han hecho de mí un hombre, por tus lágrimas, por tu corazón. A ti mi madre querida.

Por tu espíritu de consagración, dignos ideales, respeto, amor. Por todas esas pequeñas y grandes cosas que me has enseñado a lo largo de la vida, por depositar tu infinita confianza en mí. A ti mi padre querido.

A toda mi familia, por tener siempre fe y confianza en mí, a mi hermana Rita María, mi hermano Rubén, mi sobrina Amanda, mis viejos Juana y Marcos Antonio, mis tíos y primos, por los vivos y por los que con amor se despidieron, los cuales viven dentro de mi corazón, a ti mi abuelita querida, mi segunda madre, a ti mi abuela María. Existen personas especiales, personas que por ley no estuvieron en el vientre de madre, pero son personas que se llevan dentro del corazón cada día que pasa, los amigos. Mencionarlos todos acabaría con la paciencia de muchos, a mi viejo amigo Yunier, José, Ramón, mi fiel amigo y compañero Yoandy, Víctor Manuel, Alexander, Julio César, Reynaldo, Adrián, Mandile y los demás compañeros que me han acompañado durante estos cinco años de estudio.

A una persona encantadora que me ha apoyado, me ha dado aliento a seguir adelante, a ti Aida. Agradezco a mi tutora Claudia Núñez Sanz, a todos los profesores que de una forma u otra han contribuido en mi enseñanza en estos cinco años de estudio. A esta gran universidad, a este gran sueño revolucionario a ti mi querida UCI. A todos agradezco infinitamente el apoyo, la ayuda y confianza depositada en mí.

Marcos

En este momento tan importante no puedo dejar de mencionar a las personas que han intervenido de una forma u otra en la realización de este sueño. A ellos va este agradecimiento:

A mi Papá por ser mi hermano, mi amigo, por enseñarme como dirigirme en la vida, por brindarme su mano siempre y mostrar confianza en los pasos que he dado.

A mi Mamá por ser mi guía, mi guardiana, por creer siempre en mí. Dicen que detrás de cada gran hombre ahí una gran mujer, en mi vida esa gran mujer eres tú.

A mi Tata gracias por ser mi hermana y mas que mi hermana mi mejor amiga.

A mis abuelos sé que no están físicamente pero donde se encuentren espero que puedan apreciar en que me he convertido y se sientan orgullosos.

A mi abuela Amelia por siempre preocuparse por mí, gracias mi viejita linda.

Este agradecimiento especial va dirigido a una persona que no se encuentra físicamente entre nosotros pero sé que siempre sus buenas ideas me acompañan. Esa persona no me dio a luz pero me formo como persona, me dio todo su amor y cariño para ti Abuela Chita mi agradecimiento. Te quiero mucho.

A mis primas Deolinda y Briyi, gracias por estar junto a mí estos años.

A mis tías y tíos por brindarme todo su amor y cariño.

A mis primos y primas que se que están muy orgullosos de mi.

A mi vecina Tete por ser parte de mi familia.

A mis amigos de siempre los de primer año: Julio, el Russo, Yosmany, Ismel y Yeney a todos gracias por estar ahí.

A mis amigos del futbol y mis hermanos Arcides, Marlon, Carlos y Yanier.

A Lorenzo y Enrique por ser como hermanos para mí.

A Lianet por ayudarme siempre en este año tan difícil.

A mi tutora por alentarme, orientarme y apoyarme en esta etapa tan difícil.

En fin a todas las personas que he conocido durante estos cinco años que desean lo mejor para mí, que me han ayudado y me han hecho salir adelante, muchísimas gracias.

Yoandy

DEDICATORIA

A mi abuela María por estar siempre en mi memoria:
A mi madre querida, a mi padre, hermanos y abuelos por el apoyo y confianza incondicional.

Marcos

A mis padres, las personas más especiales de mi vida, por ser la razón y el motivo para esforzarme al máximo en cada momento. Ellos han apoyado todas las decisiones importantes que he tomado en mi vida. A ustedes va dedicado este trabajo, para que se sientan orgullosos de su hijo. Los quiero mucho.

Yoandy

RESUMEN

Los libros constituyen una fuente de insaciable conocimiento y sabiduría por la inmensa información que atesoran. Ya sea en formato físico o digital su fin siempre será el mismo: el de brindar a todos las ideas contenidas en sus páginas. Entre los distintos tipos de libros se encuentran los libros electrónicos considerados una publicación en formato digital capaz de contener diferentes recursos. En el Centro de Tecnologías de Gestión de Datos (DATEC), se desarrolla el proyecto Libros Web, que tiene como objetivo principal la creación de libros web. Estos libros representan un tipo de libro electrónico en formato HTML, los cuáles contienen la ayuda de los sistemas que se desarrollan en el centro. La investigación surge producto de las necesidades del grupo de trabajo en materia de compatibilidad con las nuevas políticas de la Universidad. Se tiene, por tanto como finalidad desarrollar una aplicación informática sobre plataformas libres que facilite la edición y generación de los libros web. Para su elaboración se realizó un estudio del proceso de generación de libros electrónicos y de libros web tanto en Cuba como en el mundo. Para el desarrollo de la aplicación se utilizó la metodología ágil OpenUP, el lenguaje de programación PHP y el marco de trabajo Symfony. La aplicación fue probada utilizando las pruebas funcionales mediante los casos de pruebas basados en casos de usos y las pruebas unitarias para validar el completo funcionamiento de la aplicación.

Palabras claves: Aplicación Web, Libros Electrónicos, Libros Web, Software Libre.

PENSAMIENTO II

AGRADECIMIENTOS IV

DEDICATORIA VII

RESUMEN VIII

INTRODUCCIÓN 1

CAPÍTULO 1. FUNDAMENTO TEÓRICO Y TECNOLÓGICO DE LA APLICACIÓN 6

1.1. Aplicación Informática 6

1.1.1 Aplicación Web 6

1.2 Libro Electrónico 7

1.2.1 Libro Web 8

1.3 Herramientas para la generación de libros electrónicos y libros web 8

1.3.1 Herramientas para la generación de libros web en el mundo 10

1.3.2 Herramientas para la generación de libros web en la UCI 11

1.3.3 Comparación entre las herramientas generadoras libros electrónicos 11

1.4 Metodología de Desarrollo 12

1.4.1 Metodología Ágil 12

1.4.1.1 OpenUp 13

1.5 Tecnologías a utilizar 14

1.5.1 AJAX 14

1.5.2 XML 15

1.6. Lenguajes de programación a utilizar 15

1.6.1 JavaScript 15

1.6.2 PHP 5.2 16

1.7 Marco de Trabajo 17

1.7.1 Symfony 1.4.8 17

1.7.2 ExtJS 3.3.2 18
1.8 ORM 19
1.8.1 Propel 20
1.9 Gestor de bases de datos 20
1.10 Herramientas de desarrollo 21
1.11 Lenguaje de modelado 22
1.12 Herramienta CASE 22
1.12.1 Visual Paradigm 22
1.13 Conclusiones 23
CAPÍTULO 2. ANÁLISIS Y DISEÑO DE LA APLICACIÓN 24
2.1 Modelo de Dominio 24
2.1.1 Glosario de Términos 25
2.2 Requisitos 25
2.2.1 Requisitos funcionales 25
2.2.2 Requisitos no funcionales 26
2.3 Modelo del Sistema 27
2.3.1 Diagrama de Casos de Uso del Sistema (CUS) 27
2.3.2 Descripción de los actores del sistema 29
2.3.3 Descripción de los casos de usos del sistema. 29
2.4 Patrones 32
2.4.1 Patrones de Arquitectura 32
2.4.2 Patrones de Diseño 35
2.5 Estilos Arquitectónicos 37
2.6 Diseño del Sistema 38
2.6.1 Realizaciones de los casos de uso. 38

2.6.2 Diagrama de clases del diseño 39
2.7 Diagramas de interacción 40
2.7.1 Diagrama de Secuencia 40
2.8 Modelo de Datos 42
2.9 Conclusiones 43
CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA 44
3.1 Diagrama de despliegue. 44
3.2 Diagrama de Componentes 46
3.3 Mecanismos de Implementación 47
3.4 Pruebas 49
3.4.1 Tipos de pruebas 50
3.4.2 Pruebas de Caja Negra 51
3.4.3 Pruebas unitarias 56
3.5 Conclusiones 57
CONCLUSIONES 58
RECOMENDACIONES 59
REFERENCIA BIBLIOGRÁFICA 60
ANEXOS 65
GLOSARIO DE TÉRMINOS Y DEFINICIONES 68

INTRODUCCIÓN

Desde los orígenes, la humanidad ha tenido la necesidad de preservar y transmitir su cultura. La historia de la humanidad no sería relatada en la actualidad si no hubiese existido el impetuoso deseo de conservar los conocimientos en algo físico que pudiese hacer frente al tiempo. La pintura rupestre en cavernas, el papiro y las pieles de animales, fueron algunos de los intentos de los antiguos humanos en plasmar sus ideas y los primeros pasos en la invención del libro.

Los chinos fabricaron el papel a partir de los residuos de la seda, la paja de arroz y el cáñamo. Siglos después en Europa, Johannes Gutenberg de origen alemán, inventa la imprenta. Estos grandes avances dieron cabida al desarrollo y propagación de los libros en todo el mundo.

El constante desarrollo tecnológico propició el avance de muchos sectores, dentro de los cuales se provee el de las Tecnologías de la Información y las Comunicaciones (TICs). En la segunda mitad del siglo XX se hace inminente la sustitución del papel por algo intangible conocido como libro electrónico, debido a la necesidad de plasmar la información en soportes digitales. El primer paso fue el proyecto Gutenberg llevado a cabo en la Universidad de Illinois, liderado por Michael Hart, pero no fue hasta diez años más tarde cuando realmente salió al mercado el primer libro electrónico comercial, el Random House's Electronic Dictionary, editado por Random House's. Desde ese momento el concepto y la forma que se conocía como libro electrónico ha variado, adaptándose a diferentes formas y formatos. (1)

A partir de la aparición de los libros electrónicos y el desarrollo acelerado de Internet no tardó en aparecer el libro electrónico en formato web, conocido como libro web o web book por su significado en inglés. Su surgimiento estuvo condicionado en sus inicios por la necesidad de mostrar publicaciones que fuesen accesibles y compatibles con la mayoría de los navegadores, observándose la ventaja que estos poseían frente a otros formatos de libros electrónicos. En la actualidad este tipo de libro electrónico es considerado como uno de los más usados, gracias a su facilidad de uso y la similitud a una página web.

El desarrollo de la informática en Cuba ha alcanzado grandes logros, propiciado en gran medida por los esfuerzos del estado cubano en un intento por hacer frente a las nuevas tecnologías y el acelerado desarrollo de las TICs a nivel mundial. Entre los logros más

significativos se encuentra la creación de los Joven Club de Computación y Electrónica que constituyen un gran paso para el desarrollo del conocimiento y la preparación de una cultura informática en Cuba.

Muchos sectores empresariales se vieron beneficiados con el auge de la informática en el país, uno de estos fue la empresa papelera cubana que se encontraba en crisis por la falta de materias primas, sin embargo, en 1992 gracias a los esfuerzos del Estado en potenciar el desarrollo tecnológico, científico y técnico se logró publicar el primer libro electrónico en Cuba. (2)

Debido al ímpetu alcanzado por las TICs, en Cuba es creada la Universidad de las Ciencias Informáticas (UCI), en septiembre del 2002, por iniciativa del Comandante en Jefe Fidel Castro Ruz. En aras de convertirla en uno de los principales exponentes en el mercado del software y propiciar aun más el proceso de informatización de la sociedad cubana. La UCI tiene como una de sus principales misiones producir software, servicios informáticos y soluciones tecnológicas integrales. Partiendo de la utilización del modelo de formación estudio-trabajo y su modelo de producción desglosado en Centros de Desarrollo Productivos vinculados a las facultades por especialidades.

A la Dirección de Software Educativo (DSE) de la UCI se le otorga la tarea de crear un sistema capaz de generar libros electrónicos. Esta dirección poseía entre sus objetivos primordiales la creación de programas didácticos, creados para facilitar los procesos de enseñanza-aprendizaje y la realización de libros electrónicos en formato web. El proceso de obtención de los libros electrónicos se realizaba de manera manual propiciando como consecuencias errores sustanciales en los productos tanto de contenido como de estilos por lo que era necesario utilizar un grupo grande de personas entre estudiantes y profesores para su corrección. (2)

Debido a esto la dirección de la universidad orienta a la Facultad 3 el desarrollo de una aplicación de escritorio para la realización correcta de los libros electrónicos, obteniéndose la plataforma “Generación de Libros Electrónicos”. (2)

En el actual Centro de Tecnologías de Gestión de Datos (DATEC) que se especializa en proveer soluciones integrales así como servicios informáticos especializados en el análisis y almacenamiento de datos, se encuentra el proyecto Libros Web, el cual adoptó como herramienta para la confección de los manuales de ayuda de los proyectos liberados por DATEC la Plataforma de Generación de Libros Electrónicos.

Esta plataforma basa su arquitectura sobre el Sistema Operativo Windows y utiliza tecnologías no acordes con las líneas y políticas de la UCI tales como:

- Framework .Net en su versión 1.0.
- Entorno integrado de desarrollo CSharp 2003.
- Gestor de bases de datos SQLServer 2000.

Todas estas tecnologías enunciadas son obsoletas e incompatibles con el Sistema Operativo Linux que constituye el sistema operativo utilizado por DATEC.

Uno de los inconvenientes de esta plataforma de Generación de Libros Electrónicos es que posee dos módulos de gestión de contenidos para obtener los libros web, convirtiéndose en algo engorroso utilizar estas herramientas si no se conoce el proceso de generación de cada una.

Además, resulta inconveniente para los clientes que la plataforma posea solamente dos niveles de jerarquía (vertical y horizontal) y que no cuente con menús desplegables limitando la cantidad de epígrafes y capítulos por documento. Al intentar cargar las imágenes en la plataforma estas no se visualizan, originando una demora en la generación de los libros debido a que las imágenes deben ser insertadas de manera reiterada para que se puedan visualizar. Por otra parte, aunque el producto final de la plataforma es un libro electrónico en formato web compatible con la herramienta Adobe Dreamweaver, al ser generado es necesario ajustar las imágenes y la estructura del libro manualmente debido a que las direcciones de cada vínculo con el documento se crean con errores.

Estos problemas planteados repercuten de manera significativa en la obtención de los libros web debido a que no es posible utilizar la Plataforma en el Sistema Operativo Linux al no ser esta, multiplataforma. Además, los especialistas que se encargan de generar los libros web necesitan dedicar mucho esfuerzo y tiempo en corregir los errores obtenidos en la generación propiciando una demora en la entrega del producto y que este no cuente con la calidad esperada.

Teniendo en cuenta la **situación problémica** expuesta se propone como **problema de la investigación**:

¿Cómo generar libros web sobre plataformas libres para permitir la visualización de la información?

El **objeto de estudio** de la investigación se enmarca en el proceso de generación de libros electrónicos y el **campo de acción** se enfoca en el proceso de generación de libros web sobre plataformas libres.

Para dar solución al problema planteado, se propone como **objetivo general** desarrollar una aplicación para la generación de libros web sobre plataformas libres.

Objetivos Específicos:

- Analizar el estado del arte referente a la generación de libros electrónicos y libros web.
- Realizar el análisis y diseño de la aplicación para generar libros web de acuerdo con la metodología definida.
- Implementar una aplicación para generar libros web mediante plataformas libres y realizar las pruebas necesarias con el fin de validar los indicadores de calidad en la aplicación propuesta.

Tareas Investigativas:

- Análisis del estado del arte referente al proceso de generación de libros electrónicos y libros web tanto en el mundo como en Cuba.
- Selección de las tecnologías, metodologías, lenguajes y herramientas necesarias para el desarrollo de la aplicación para generar libros web.
- Selección de la herramienta para el componente motor generador de libros web.
- Aplicación de la Ingeniería de Requisitos con el fin de evidenciar el análisis de la aplicación generadora de libros web.
- Diseño de la aplicación.
- Implementación del módulo motor generador de libros web.
- Implementación de la aplicación web para generar libros web.
- Aplicación de las pruebas necesarias para validar el correcto funcionamiento de la aplicación.

El presente trabajo de investigación está estructurado en tres capítulos:

Capítulo 1: Fundamentación Teórica y Tecnológica del Sistema

En este capítulo se realiza un estudio, caracterización y análisis del tema de investigación, profundizando en las principales herramientas para la gestión de libros web en el mundo y en la UCI, se selecciona la metodología de desarrollo y la herramienta de modelado a

utilizar. Además, se especifica el lenguaje de programación, los patrones de software y las herramientas empleadas en el desarrollo de la aplicación.

Capítulo 2: Análisis y Diseño del Sistema

En este capítulo se definen los requerimientos a implementar en la aplicación, el diagrama de casos de usos y se realiza una breve descripción de estos. Además, se realiza el diagrama del modelo arquitectónico propuesto y la descripción de la arquitectura base. También se plasman los diagramas de clases del análisis, los diagramas del diseño y los diagramas de secuencia. Así como se describen las principales clases y los patrones de software empleados en el desarrollo de la aplicación

Capítulo 3: Implementación y Prueba

En este capítulo se describen las principales implementaciones y los diagramas de componentes asociados. Además, se evidencia el diagrama de despliegue, así como que se especifican las pruebas realizadas sobre la aplicación y sus resultados de acuerdo con los tipos de pruebas y las no conformidades identificadas después de la implementación.

CAPÍTULO 1. FUNDAMENTO TEÓRICO Y TECNOLÓGICO DE LA APLICACIÓN

En el presente capítulo se abordarán y analizarán temas relacionados con las diferentes herramientas existentes en el mundo y en la UCI para la generación de libros web, sus características más importantes y una breve comparación, mediante los aspectos más relevantes que permitan un análisis crítico de ellas. Este análisis tiene como objetivo definir las herramientas de acuerdo con las características más factibles a utilizar para desarrollar la aplicación web. Se brinda una descripción de la metodología a utilizar, el lenguaje de modelación, las tecnologías a utilizar y el marco de trabajo; todas estas temáticas son necesarias para hacer posible el desarrollo de este trabajo de investigación.

1.1. Aplicación Informática

Una aplicación consiste en el empleo o puesta en práctica de los procedimientos adecuados para conseguir un fin. Llevándolo a la informática se conoce como aplicación informática un programa o conjunto de programas informáticos que realizan un trabajo específico, diseñado para el beneficio del usuario final. (3)

Las aplicaciones informáticas en la actualidad se han convertido en un eslabón importante en el desarrollo de la industria del software que como bien decía Pressman deben "... *ser un producto que haga el modo de vivir más fácil al usuario*". (4)

Por lo tanto una aplicación informática es un producto informático donde se realizan tareas específicas brindando solución a algún problema, garantizando que el resultado final sea de total satisfacción para el usuario.

Otras aplicaciones aparecen integradas en forma de paquetes por ejemplo: procesador de textos, hojas de cálculo y presentaciones. Algunas son lanzadas bajo el concepto de suite, reuniéndose también en forma de paquetes de aplicaciones. Las aplicaciones informáticas se pueden clasificar en aplicaciones de escritorio y en aplicaciones web. El objetivo de la investigación se centra en las aplicaciones web.

1.1.1 Aplicación Web

Una aplicación web es una aplicación informática que los usuarios pueden utilizar accediendo a un servidor web a través de internet o de una intranet mediante un navegador.

En otras palabras, es una aplicación de software que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador. (5)

La popularidad creciente que han alcanzado las aplicaciones web se debe principalmente al desarrollo del internet y la facilidad con que los usuarios acceden a esta, además la mayoría de estas requiere menos tiempo en su desarrollo y construcción.

Entre las principales ventajas se encuentran:

- Las aplicaciones pueden ser utilizadas a través de múltiples plataformas, tanto de hardware como de software.
- Poseen actualización instantánea, debido a que todos los usuarios de la aplicación hacen uso de un solo programa que radica en el servidor, los usuarios siempre utilizarán la versión más actualizada del sistema.
- Fáciles de integrar con otros sistemas, debido a que se basa en protocolos estándares, la información manejada por el sistema puede ser accedida con mayor facilidad por otros sistemas.
- Posee acceso móvil, el usuario puede acceder a la aplicación con la única restricción de que cuente con un acceso a la red privada de la organización o a internet. Este acceso se puede realizar desde una computadora de escritorio en su casa, una computadora portátil o desde su oficina. (6)

1.2 Libro Electrónico

El libro electrónico es una publicación electrónica digital almacenada, que permite incorporar audio, videos y enlaces.

Con el objetivo de unificar el formato de publicación adoptado por las editoriales electrónicas se presentó la normativa Open eBook, integrada por compañías de software y hardware, editoriales, autores, asociaciones relacionadas y usuarios de libros electrónicos cuyo fin es establecer especificaciones comunes en los sistemas de los libros digitales. (7)

Desde su surgimiento hasta nuestros días el libro electrónico ha demostrado poseer grandes ventajas para el mundo tecnológico actual, algunas de estas ventajas son:

- Portabilidad y usabilidad.
- Reducción del coste de la publicación digital.
- Enriquecimiento del texto con enlaces, multimedia y audio.
- Accesibilidad en tiempo real.

Algunos de los formatos más usados en la actualidad son:

DjVu: Formato libre, se destaca en el almacenamiento de imágenes escaneadas. (8)

Doc(x): Formato creado por la compañía Microsoft perteneciente a su procesador de texto Microsoft Office.

EPUB: Formato libre desarrollado por el IDPF (International Digital Publishing Forum). (9)

HTML: Formato propio de la World Wide Web, o lenguaje de marcado de hipertexto. (10)

PDF: Adobe Portable Document. Formato creado por la compañía Adobe. (11)

1.2.1 Libro Web

El libro web es un tipo de libro electrónico que constituye una publicación electrónica en formato HTML o lenguaje de marcado de hipertexto, constando de varias páginas y vínculos entre ellas. Es posible mediante la utilización de estos libros insertar recursos como: imágenes, videos y audio.

Debido al avance de las TICs, los libros web también han experimentado cambios como: la forma de organizar su contenido, la inserción de audio y la inclusión de scripts.

1.3 Herramientas para la generación de libros electrónicos y libros web

A continuación se realiza el análisis de las herramientas para la generación de libros electrónicos y libros web existentes en el mundo y en Cuba brindando como resultado las características necesarias para seleccionar la herramienta idónea para utilizarse como guía en la obtención de la aplicación.

EPUB

Documento de publicación electrónica (electronic publication), por sus siglas en inglés que cuenta con licencia GPL definido por International Digital Publishing Forum (IDPF). EPUB consiste en un archivo ZIP que contiene tres ficheros XML basados en estándares de código abierto y permiten obtener un libro electrónico:

- Open Publication Structure (OPS): contiene las instrucciones de formato del contenido.
- Open Packaging Format (OPF): describe la estructura del documento EPUB en formato XML.
- Open Container Format (OCF): almacena todos los archivos que componen un EPUB y los comprime en formato ZIP. (9)

Entre las principales ventajas se encuentran**:**

- Permite representar en formato HTML el documento escogido, sin tener en cuenta las limitaciones de espacio y orientación de las diferentes aplicaciones que la

representan.

- Incorpora hojas de estilo (CSS) para un mejor diseño de las páginas HTML.
- Permite crear un eBook con herramientas como: el Bloc de Notas y el GEDIT, al poseer formato XHTML, formato utilizado para generar páginas web
- La mayoría de los navegadores lo pueden interpretar. (12)

Entre las principales desventajas se encuentran**:**

- Su formato de salida es EPUB, formato poco estándar semejante a un Zip, ocasionando que pocas aplicaciones lean este formato. (12)
- No es posible decidir el modo de consultar el contenido del libro electrónico al no poseer un índice con el que se pueda ir a las otras páginas y utilizar una navegación básica. (13)

InDesign

Software de la compañía Adobe que permite el diseño, edición y visualización previa de diseños de páginas en formato HTML para su impresión o distribución digital. Posee un control preciso sobre la tipografía e integra la interactividad mediante el vídeo y el audio para su reproducción en tabletas, teléfonos inteligentes y computadoras. (14)

Entre las principales ventajas se encuentran:

- Permite trabajar sobre varias hojas de edición a la vez sin que el rendimiento del programa aumente.
- Posibilita la creación de un índice sencillo por palabra clave o un índice por otro criterio más detallado de la información en el libro electrónico o en la revista. (14)

Entre las principales desventajas se encuentran:

- Es una herramienta con licencia privativa y de código no libre.
- Se especializa en el diseño de revistas en formato web.
- Solo permite administrar imágenes y la información en forma de párrafos.
- Es necesario poseer conocimientos previos sobre diseño y edición web. (14)

Reader Ebook Wizard

Es un programa que guía paso a paso la creación del propio libro electrónico, a partir de ficheros en formato txt o html, con soporte también para css, jpg, bmp y gif. Brinda la opción de personalizar totalmente el libro electrónico con detalles como título, nombre de autor, categoría temática, descripción e imagen de la portada. Una vez rellenos todos los pasos del asistente, Reader Ebook Wizard genera automáticamente el libro electrónico en formato LIT,

compatible con Microsoft Reader. Se ejecuta sobre el sistema operativo Windows en sus versiones 95, 98, 2000 y XP. (15)

Entre las principales ventajas se encuentran:

- Permite especificar el título, nombre del autor, categoría, descripción, imagen de la portada
- Permite crear libros electrónicos a partir de txt o html.
- Es posible añadirle imágenes en varios formatos. (15)

Entre las principales desventajas se encuentran:

- El libro electrónico generado no es compatible con las tildes y los caracteres extraños.
- Obtiene los libros electrónicos a partir de formato txt no admite formatos como: doc o pdf. (15)

1.3.1 Herramientas para la generación de libros web en el mundo

Python-Sphinx

Creada por Georg Brandl, el 21 de marzo de 2008. Es una herramienta que posee licencia BSD, lo que posibilita el uso del código fuente en software no libre y la protección del copyright en caso de posibles reclamaciones. Su principal utilización es la creación de documentación y manuales de usuario para proyectos en python. (16)

Entre las principales ventajas se encuentran:

- Genera la documentación en varios formatos como: html, texto plano y pdf.
- Permite una fácil definición de la estructura del documento, la estructura jerárquica que se puede especificar permite enlaces automáticos a los hermanos, padres e hijos o referencias a la raíz.
- Posibilita extensas referencias cruzadas, al permitir el uso de marcado semántico y vínculos automáticos de funciones, clases, citas y glosario de términos.
- Incorpora un buscador web en el mismo sitio que permite la búsqueda de palabras en el sitio web.
- Incorpora el índice general y el índice de los módulos. (16)

Entre las principales desventajas se encuentran:

- No permite importar documentos en formato doc o pdf.
- La edición y generación de la documentación o el manual de usuario se realiza mediante consola.

1.3.2 Herramientas para la generación de libros web en la UCI

La Plataforma para la Generación de Libros Electrónicos es una herramienta desarrollada por la Universidad de las Ciencias Informáticas (UCI), basada en tecnologías propietarias de Microsoft como: framework .NET y el gestor de bases de datos SQL Server 2000. Está compuesta por dos módulos el ePapyrus y el eScribe. El ePapyrus es un Gestor de Contenidos, el cual administra y gestiona la estructura y los contenidos de la información de los libros electrónicos mediante el Lenguaje Marcado Extendido, por sus siglas en inglés (XML) que brinda la posibilidad de gestionar los directorios de recursos tales como: documentos, imágenes y videos. El módulo eScribe es otro Gestor de Contenidos que importa el XML de salida del ePapyrus y una plantilla de diseño en Dreamweaver para obtener como producto final un libro electrónico en formato web.

Entre las principales ventajas se encuentran:

- Permite especificar el autor, las temáticas, los colaboradores.
- Posee una interfaz amigable y entendible.
- Genera un libro electrónico en formato web.
- Permite insertar un documento en formato doc.

Entre las principales desventajas se encuentran:

- Desarrollada usando arquitectura privativa (framework .NET, SQL Server 2000).
- Incompatibilidad con otras plataformas.
- En el módulo eScribe al intentar importar las imágenes, estas no se cargan la primera vez, es necesario realizar la acción de importar varias veces.
- Es necesario ajustar manualmente las imágenes y los vínculos en el sitio porque se genera con pérdida de direcciones.
- Solo permite importar documentos en formato doc.
- Navegación deficiente en el libro web generado.
- Posee solo dos niveles de jerarquía vertical y horizontal.

1.3.3 Comparación entre las herramientas generadoras libros electrónicos

La siguiente comparación tiene como objetivo conocer cuál es la herramienta más adecuada para utilizarse como guía en el desarrollo de la aplicación en la cual se basarán las principales funcionalidades que darán solución al trabajo.

Herramientas	Sistema Operativo	Licencia	Importar formatos(doc ó pdf)	Código Abierto	Insertar varios niveles jerárquicos	Formato de salida
EPUB	Windows, Linux	GPL	No	Si	No	EPUB
InDesign	Windows	Propietario	No	No	No	EPUB, HTML
Reader Ebook	Windows	Freeware	No	Si	No	EPUB
Plataforma Libros Electrónicos	Windows	Privativo	No	Si	No	HTML
Python - Sphinx	Windows, Linux	BSD	No	Si	Si	HTML, PDF, Texto plano

Tabla 1. Comparación entre las herramientas para la generación de libros web.

Las herramientas caracterizadas están especializadas en la generación de libros electrónicos. De acuerdo con ciertos criterios comparativos, tales como: sistema operativo, tipo de licencia, código abierto, importación o no de formatos doc o pdf, formato de salida y jerarquía; se ha obtenido que alguna de estas herramientas no poseen las cualidades para generar los libros web. Por tanto, se ha establecido como la herramienta a usar como guía para el desarrollo de la aplicación Python-Sphinx, debido a que posee una potente estructura para incorporar a los libros varios niveles de epígrafes y capítulos. Python-Sphinx es una herramienta desarrollada para el sistema operativo Linux con licencia BSD y de código abierto compatible con las políticas de DATEC. Además, se erradica la pérdida y demora al cargar las imágenes debido a que con el uso de XML se especifica exactamente la localización de dicho recurso.

1.4 Metodología de Desarrollo

Las metodologías son el conjunto de procedimientos, técnicas, herramientas y un soporte documental, que ayuda a los desarrolladores a realizar un nuevo software. Sin el apoyo de una metodología, nunca se llegaría a satisfacer las necesidades de los clientes para los cuales se trabaja.

Se entiende por metodología de desarrollo una colección de documentación formal referente a los procesos, las políticas y los procedimientos que intervienen en el desarrollo del software. Las metodologías de desarrollo se clasifican en robustas y ágiles. (17)

1.4.1 Metodología Ágil

Se consideran metodologías ágiles a aquellas que poseen la capacidad de proveer respuestas rápidas y una gran adaptabilidad a los cambios. Esto provoca menos complicaciones de acuerdo con que se provee la documentación necesaria para dar solución al problema. Además, trata de minimizar los riesgos hacia el final del ciclo de desarrollo.

Estas metodologías ágiles surgieron con el objetivo de simplificar esas metodologías pesadas que en muchas ocasiones no eran factibles para proyectos pequeños. (18)

Entre las metodologías ágiles se encuentran: Scrum, XP, Crystal, RUP Ágil y OpenUp.

A continuación se explican las características más importantes de la metodología OpenUp. Metodología esta seleccionada para el desarrollo de la aplicación.

1.4.1.1 OpenUp.

Esta metodología define las fases, actividades y artefactos que se generan durante el ciclo del software. La finalidad de esta metodología de desarrollo es garantizar la eficacia mediante el cumplimiento de los requisitos iniciales y minimizar las pérdidas de tiempo en el proceso de generación del software. (19)

Entre las ventajas de OpenUp se encuentran:

- El proceso de desarrollo del software es completo debido a que puede ser manifestado como todo el proceso para construir un sistema.
- Es extensible ya que los procesos se pueden agregar o adaptar según lo vayan requiriendo los sistemas.
- Es ligero y proporciona una comprensión detallada del proyecto, beneficiando a clientes y desarrolladores sobre productos a entregar y su formalidad.
- Se centra en una arquitectura temprana para reducir al mínimo los riesgos y organizar el desarrollo.
- Posee licencia libre.
- Maneja el ciclo de vida de desarrollo del software de manera apropiada al ofrecer una buena administración de las diferentes áreas del proyecto. (19)

OpenUP es un proceso iterativo cuyas iteraciones se distribuyen a través de cuatro fases: Concepción, Elaboración, Construcción y Transición. Está diseñado para soportar equipos pequeños de tres a seis integrantes, no dispersos y que trabajan en proyectos cuya duración está entre tres y seis meses.

Fases de la metodología OpenUP

- Concepción: Es la primera de las 4 fases en el proyecto del ciclo de vida, acerca del entendimiento del propósito y objetivos y obteniendo suficiente información para confirmar que el proyecto debe hacer. El objetivo de esta fase es capturar las necesidades de los stakeholder en los objetivos del ciclo de vida para el proyecto.
- Elaboración: Es la segunda de las 4 fases del ciclo de vida del OpenUP donde se

trata los riesgos significativos para la arquitectura. El propósito de esta fase es establecer la base la elaboración de la arquitectura del sistema.

- Construcción: Esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El propósito de esta fase es completar el desarrollo del sistema basado en la arquitectura definida.
- Transición: Es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, y evalúa la funcionalidad y performance del último entregable de la fase de construcción. Lenguajes de programación del lado del cliente y del lado del servidor. (19)

Se seleccionó como metodología de desarrollo OpenUp, por ser una metodología ágil que posee como una de sus principales ventajas la de poseer licencia libre, así como que se adapta perfectamente a proyectos de corta duración al estar diseñada para soportar equipos de trabajos de pocos integrantes. Además, está centrada en una arquitectura temprana para prevenir los riesgos en las posteriores fases que pudieran conllevar a un fallo del proyecto. Con esta metodología se evita también la confección de una extensa documentación, diagramas e iteraciones innecesarias.

1.5 Tecnologías a utilizar

1.5.1 AJAX

AJAX también conocido como XML y JavaScript Asincrónico traducido al español, es el compendio de varias tecnologías para obtener una mejor interactividad en las páginas web. Entre las tecnologías que intervienen se pueden mencionar: HTML, JavaScript, DHTML, XML, JSP y ASP.NET.

La principal importancia del uso de AJAX es que posibilita actualizar o recargar sectores de las páginas web con información obtenida del servidor sin tener que refrescar la página y en muchas ocasiones el proceso de recarga se muestra imperceptible al usuario. (20)

Entre las principales ventajas de AJAX se encuentran:

- Es soportada en la mayoría de los navegadores actuales.
- Permite la interactividad, se elimina la espera por el usuario de cierta información que debe proporcionar el servidor, se recargan esos sectores y el usuario no tiene que esperar por la respuesta del servidor.
- Acorta el tiempo de respuesta del servidor, no es necesario que se construya

totalmente la página a mostrar al usuario solamente los sectores necesarios y que se vayan a utilizar. (20)

1.5.2 XML

El lenguaje de marcado extensible por su acrónimo en inglés (XML), es un lenguaje de etiquetas desarrollado por la www.Consortium; según Howart Katz "*XML es un formato extremadamente versátil que ha sido usado para representar diferentes formas de datos, incluyendo páginas web, libros y transacciones financiera. Un documento XML es un lineamiento donde el orden y la jerarquía son las principales unidades estructurales*".

Otra definición según Jayavel "*XML es un estándar para la representación de datos en Internet, donde las etiquetas XML describen al dato en sí*".

A partir de ambas definiciones se tiene que el lenguaje de marcado extensible es un lenguaje de guía para otros lenguajes debido a que representa un estándar de como mediante el uso de las etiquetas propias de XML se puede estructurar y definir un documento de texto en Internet. (21)

Entre las principales ventajas de XML se encuentran:

- Es extensible. Es posible adicionarle nuevas etiquetas a medida que sea necesario.
- Es compatibilidad con otras aplicaciones. Si es necesario utilizarlo en otra aplicación el proceso de incorporarlo es sencillo.
- Permite el trabajo con varios conjuntos de caracteres entre ellos UTF-8.
- Es posible crear etiquetas propias y asignarles atributos. (21)

1.6. Lenguajes de programación a utilizar

1.6.1 JavaScript

JavaScript es un lenguaje de programación interpretado, creado para que se ejecute en el navegador de cada usuario. Es utilizado en la creación de páginas web dinámicas y en la manipulación y personalización de aplicaciones. En sus orígenes fue desarrollado por Netscape para su nuevo navegador Netscape 2.0, pero por sus grandes beneficios de permitir incrustar el código en una página HTML se propagó por todos los navegadores. La significación de que es un lenguaje interpretado es que no es necesario compilarlo para poder ejecutarlo. (22)

Entre las principales ventajas de JavaScript se encuentran:

- Reducción de la carga del servidor: para realizar funciones simples como

validaciones era necesario una respuesta del servidor y con el uso de JavaScript es posible desarrollar funciones del lado del cliente para realizar estar tareas.

- Gran usabilidad: al ser compatibles con la mayoría de los navegadores es muy utilizado para desarrollar funciones y ayudar en la visualización de las páginas dinámicas.
- Fácil de aprender, rápido y potente: no es necesario poseer grandes conocimientos de programación para comenzar a utilizar este lenguaje debido a que se puede visualizar directamente en el navegador. Además, integra las propiedades de los exploradores para crear funciones muy potentes.
- No es necesario un entorno de desarrollo para programarlo, con una hoja de texto se logra la implementación de una función en JavaScript
- Basado en programación orientada a objetos aunque no incorpora la creación de clases ni la herencia. (22)

1.6.2 PHP 5.2

PHP es una mezcla entre interpretación y compilación para intentar ofrecer a los desarrolladores un mejor rendimiento y flexibilidad. PHP compila para el código una serie de instrucciones siempre que estas son accedidas. La concepción de lenguaje interpretado es que las instrucciones son ejecutadas una por una hasta que termina el código.

Es usado principalmente en interpretación del lado del servidor para la generación de páginas web dinámicas. Las principales características de PHP son: rapidez, facilidad de aprendizaje, soporte multiplataforma tanto de diversos Sistemas Operativos como de servidores HTTP y de bases de datos. PHP se distribuye de forma gratuita bajo licencia GPL. (23)

Entre las principales ventajas de PHP 5.2 se encuentran:

- Puede interactuar con la mayoría de las bases de datos tales como MySQL, PostgreSQL y Oracle.
- PHP es de código abierto, le permite a los desarrolladores no depender de una compañía específica para arreglar funciones que no funcionen y no es necesario pagar dinero por actualizaciones.
- PHP es completamente expandible. Está compuesto de un sistema principal, un conjunto de módulos y una variedad de extensiones de código.
- PHP generalmente es utilizado como módulo de Apache, lo que lo hace

extremadamente veloz. (23)

1.7 Marco de Trabajo

Según Erich Gamma framework es "*Conjunto de clases cooperativas que construyen un diseño reutilizable para un tipo específico de software"*.

Otra definición realizada por Ralph E. Johnson y Brian Foote explica que framework son "*Aplicaciones semi-completas que pueden especializarse para producir software a la medida. El framework describe los objetos que componen el sistema, cómo estos interactúan y cuáles son sus responsabilidades*".

A partir de ambas definiciones se define como que framework o marco de trabajo es un conjunto de procesos, sistemas, conceptos y tecnologías, para resolver un problema mediante el desarrollo de un software, brindando la posibilidad de reutilización de componentes y especializado en facilitar el desarrollo del software. Además cuenta con una estructura bien definida garantizando la seguridad de las aplicaciones desarrolladas, así como bibliotecas de administración de recursos y de componentes reutilizables.

Entre las principales ventajas se encuentran:

- Minimiza el tiempo de desarrollo del software: Con el uso del framework se acorta el ciclo de desarrollo al encargarse de comprender la interacción con las aplicaciones.
- Reduce el riesgo en el desarrollo del software: Con el uso del framework los riesgos en los inicios de desarrollo del software se reduce debido a que proporciona una base confiable y probada por su potencia.
- Proporciona una arquitectura consistente entre aplicaciones: Con el uso del framework todas las aplicaciones generadas tendrán en común la arquitectura, posibilitando el fácil mantenimiento y soporte.
- Posibilita la reutilización de código existente y el desarrollo mediante los patrones arquitectónicos.
- Proporciona un alto nivel de abstracción: Esto se debe a que oculta la complejidad de las aplicaciones de desarrollo, debido a que automatiza las características estándares, ofrece un mecanismo de excepciones y solo las aplicaciones tienen una salida por la estructura web. (24)

1.7.1 Symfony 1.4.8

Symfony es un completo framework diseñado para optimizar el desarrollo de las aplicaciones web de acuerdo con algunas de sus principales características. Este marco de trabajo

separa la lógica de negocio, la lógica del servidor y la capa de presentación de la aplicación web. Presenta además varias herramientas y clases que permiten reducir el tiempo de desarrollo de las aplicaciones web complejas.

El framework Symfony fue desarrollado mediante el lenguaje de programación PHP 5 y ha sido probado en numerosos proyectos reales. Una de sus aplicaciones principales es su utilización en gran cantidad de sitios web de comercio electrónico. Es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server y es posible su ejecución tanto en el Sistema Operativo Windows como en Linux. (24)

Alguna de las características que presenta son:

- Código fácil de leer y permite un mantenimiento muy sencillo.
- Sigue las mejores prácticas de patrones de diseño para la web.
- Independiente del sistema gestor de bases de datos.
- Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- Fácil de instalar y configurar en la mayoría de plataformas.
- Posee una potente línea de comandos que facilitan generación de código, lo cual permite ahorrar tiempo de trabajo.
- Utiliza variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones. (24)

1.7.2 ExtJS 3.3.2

ExtJS en sus inicios era una de las tantas librerías desarrolladas para (YUI) abreviación de Interfaz de Usuario para Yahoo. Desarrollada por Jack Sloccum por su gran potencia y ventaja en el desarrollo de aplicaciones web en el 2007 sale como una Librería independiente o framework. Su principal utilidad es posibilitar construir aplicaciones web mediante lenguaje JavaScript con tecnologías tan reconocidas como Ajax y XML.

Entre las ventajas de ExtJS se encuentran:

- Integración con la mayoría de los navegadores: No es necesario atenerse al navegador que se utiliza para el desarrollo debido a que el código del framework ExtJs funciona en la mayoría de los navegadores: Firefox, Internet Explorer, Safari y Opera.
- Balanceada relación entre Cliente-Servidor: Con el uso de ExtJs se distribuye la carga de procesamiento entre el cliente y el servidor, permitiendo que el servidor pueda atender más clientes al mismo tiempo.

- Eficiencia de la red: Con el uso de ExtJs se disminuye el tráfico en la red debido a que se puede especificar cuáles datos se desean enviar al servidor o que datos necesita del servidor.
- Utilización de componentes predefinidos: Cuenta con una librería de componentes y api definidos para la creación de aplicaciones complejas.
- Posee comunicación asíncrona: Con el uso de ExtJs las aplicaciones pueden comunicarse con el servidor sin necesidad de estar sujetas a una acción del usuario, esto permite que el usuario no se aburra esperando por una información que la suministra el servidor. Con esto se actualizan los sectores necesarios. (25)

De acuerdo con las principales características del marco de trabajo Symfony se seleccionó este como framework a utilizarse en la programación del lado del servidor debido a que toma las mejores maneras del patrón arquitectónico modelo-vista-controlador. Además de minimizar sustancialmente el tiempo de desarrollo de las aplicaciones al comprender las interacciones con las aplicaciones y garantiza la seguridad al brindarle a la aplicación un solo punto de entrada y brinda una estructura bien organizada y probada. Para la programación en el lado del cliente se utilizará como framework ExtJS por poseer licencia libre, existir gran documentación tanto en inglés como en español sobre el desarrollo utilizando este marco de trabajo y contar con una comunidad en aumento. También una de sus ventajas primordiales es que integra un conjunto de componentes predefinidos para el desarrollo de aplicaciones.

1.8 ORM

ORM representa la abreviación de Mapeo de Objeto-Relacional (Object Relational-Mapping) es una técnica de programación para convertir datos entre un lenguaje de programación orientado a objetos y es utilizado en una base de datos relacional, mediante el uso de un motor de persistencia. Para acceder de forma efectiva a la base de datos desde un contexto orientado a objetos, es necesaria una interfaz que traduzca la lógica de objetos a la lógica relacional. Una de las interfaces más utilizada es ORM. Symfony utiliza como uno de sus ORM por defecto Propel y a su vez Propel usa Creole como una capa de abstracción. Entre los tipos de ORM que se encuentra Doctrine y Propel.

Importancia de los ORM

- Reutilización: Permite la reutilización llamando a los métodos de un objeto de datos desde distintas partes de la aplicación e incluso desde diferentes aplicaciones.

- Encapsulación: La capa ORM encapsula la lógica de los datos pudiendo hacer cambios que afectan a toda la aplicación únicamente modificando una función.
- Portabilidad: Utilizar una capa de abstracción nos permite cambiar en mitad de un proyecto de una base de datos MySQL a una Oracle sin ningún tipo de impedimento.
- Seguridad: Implementan mecanismos de seguridad que protegen nuestra aplicación de los ataques más comunes como inyecciones SQL.
- Mantenimiento del código: Posibilita modificar y mantener el código de una manera sencilla. (26)

1.8.1 Propel

Propel es un ORM cuya función es gestionar el acceso a la base de datos y gestionar el modelo del sistema. Implica que el acceso y la modificación de los datos almacenados en la base de datos se realicen mediante objetos, nunca de forma explícita, permitiendo un alto nivel de abstracción y fácil portabilidad. Propel tiene incluido tareas para generar automáticamente las sentencias SQL necesarias para crear las tablas de la base de datos. La librería Propel se encarga de esta generación automática, ya que crea el esqueleto o estructura básica de las clases y genera automáticamente el código necesario. La abstracción de la base de datos es completamente transparente para el programador, ya que se realiza de forma nativa mediante PDO (PHP Data Objects). Esto es muy importante ya que si se cambia el sistema gestor de bases de datos en cualquier momento, no es necesario reescribir todo el código, ya que tan solo es necesario modificar un parámetro en un archivo de configuración.

Ventajas de Propel frente a otros ORMs.

- Soporta la aplicación de varios comportamientos de los modelos usados, guardando las fechas y actualizándolas cada vez que se realice una consulta de actualización sobre un registro.
- Soporta accesorios de datos y migraciones, así como capturas, eventos, paginación y una interfaz de línea de comandos.
- Posee una extensa documentación tanto en línea como manuales guías.
- Utiliza métodos para setear y obtener valores. (26)

1.9 Gestor de bases de datos

PostgreSQL 8.4

Este es un Sistema de Gestión de Bases de Datos Relacionales Orientada a Objetos (ORDBMS), por sus siglas en inglés. Es una herramienta desarrollada por la Universidad de California Berkeley. Se considera por sus características como uno de los gestores de bases de datos de licencia libre y código abierto más avanzado de la actualidad por contar con más de diez años de evolución. PostgreSQL utiliza el modelo cliente-servidor así como el multiproceso en vez del multihilo, lo que garantiza la estabilidad del sistema. Con este multiproceso en caso de fallos no se afecta el sistema del todo ya que solo se afecta dicho proceso y no los demás. (27)

Entre las ventajas se encuentran:

- Es multiplataforma puede utilizarse en sistemas operativos como Linux y Windows.
- Documentación muy bien organizada, detallada, pública y libre.
- Posee comunidades de desarrollo muy activas, en Cuba existe una comunidad de PostgreSQL.
- Soporte nativo para lenguajes muy utilizados como PHP, C, C++, Python.
- Posee todas las características de una base de datos profesional tales como: procedimientos almacenados, funciones, triggers, relaciones, reglas y vistas. (27)

El sistema gestor de bases de datos seleccionado para el desarrollo de la aplicación es PostgreSQL 8.4 debido a que es considerado entre los ORDBMS de licencia libre como uno de los más completos por poseer todas las funcionalidades de un gestor de bases de datos privativo. Además, en la UCI existe una comunidad de PostgreSQL cubano en la que se trabaja en el desarrollo y explotación de este gestor. Entre otras de las ventajas determinantes es su documentación bien detallada y organizada.

1.10 Herramientas de desarrollo

NetBeans IDE 6.9

NetBeans IDE es una plataforma integrada de desarrollo bajo licencia GPL y de código abierto. Esta herramienta tiene la finalidad de permitirle a los desarrolladores escribir, compilar, depurar y ejecutar programas. En sus inicios fue realizado para el lenguaje de programación Java, pero por sus características se extendió a otros lenguajes como PHP, JavaScript y Python. Existe además un número importante de módulos para extender el NetBeans IDE. (28)

Entre las ventajas se encuentran:

- Cuenta con licencia GPL y es de código abierto.
- Interfaz sencilla, entendible, amigable; cualquier persona puede entender cada componente de la interfaz por ser esta muy explicativa.
- Posibilita el desarrollo de aplicaciones de todo tipo al contar con herramientas para la construcción de diferentes tipos de aplicaciones ya sean: aplicaciones de escritorio, web o móviles. (28)

1.11 Lenguaje de modelado

El Lenguaje de Modelado Unificado (UML), es un lenguaje semejante al de la vida real por ser expresivo, claro y uniforme. Utilizado en el diseño orientado a objetos al permitir la fuerte integración entre herramientas, procesos y dominios.

Según definición de Rumbaugh "*UML es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software*". (29)

1.12 Herramienta CASE

Las herramientas de Ingeniería de Software Asistida por Computadoras, conocida por sus siglas en inglés (CASE) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo el costo de la misma desde el punto de vista de tiempo y dinero. Con esta herramienta es posible la modelación y documentación de los artefactos obtenidos. Estas herramientas tienen como característica principal que agilizan los procesos de desarrollo del software al intervenir en todos los aspectos de su ciclo de vida, al tratar de automatizar los procesos de generación del producto a partir de los modelos realizados. (30)

1.12.1 Visual Paradigm

Visual Paradigm for UML (VP) está diseñado para facilitar a los que la utilizan el diseño de los diagramas y posibilitar generar documentación. VP es una potente herramienta CASE que posibilita generar el código para entornos integrados de desarrollo tales como: NetBeans, Eclipse, Oracle JDeveloper y JBuilder, así como que integra el ciclo completo de desarrollo del software desde el análisis, el diseño, la construcción, las pruebas y el despliegue. (31)

Entre las ventajas de Visual Paradigm se encuentran:

- Proporciona a los desarrolladores una plataforma que les permite diseñar un producto rápidamente y con la calidad requerida.

- Posibilita generar código y realizar ingeniería inversa para varios lenguajes de programación: NetBeans, C++, PHP, XML Schema, COBRA y ADA.
- Facilita la interoperabilidad con otras herramientas CASE.
- Se integra con múltiples herramientas de desarrollo, como Eclipse, Jbuilder, NetBeans IDE y Oracle Jdeveloper.
- Es posible la integración con el Microsoft Visio para importar imágenes de esa herramienta en la realización de un diagrama de despliegue. (31)

1.13 Conclusiones

En este capítulo se describieron y analizaron los principales conceptos necesarios para entender la importancia de este trabajo. Se realizó un estudio de varias de las herramientas existentes para la generación de libros electrónicos, definiéndose cuáles eran aceptables para utilizarlas en este trabajo. Se seleccionó como herramienta de referencia para la generación de los libros web Python-Sphinx, por ser una herramienta muy potente, estar basada sobre plataformas libres y poseer licencia GPL. La metodología de desarrollo a utilizar es OpenUP, diseñada para proyectos de corta duración y contar con licencia libre. El lenguaje de modelado será UML y como herramienta para la modelación se optó por Visual Paradigm.

CAPÍTULO 2. ANÁLISIS Y DISEÑO DE LA APLICACIÓN

Introducción

En el presente capítulo se realizará la descripción detallada del sistema a desarrollar, se identificarán los actores, los trabajadores y los casos de usos que permitirán una mejor comprensión y abstracción sobre las características del sistema. Además se especifican los requerimientos funcionales y no funcionales que debe tener el sistema, junto con su descripción textual; así como los distintos diagramas necesarios para entender los procesos del negocio que se efectúan en el centro DATEC para generar libros web.

2.1 Modelo de Dominio

El modelo de dominio representa la realidad de las interacciones entre los objetos del proyecto. Este es un diagrama que está compuesto por los objetos que se identifican como reales que se relacionan con el proyecto que se desea desarrollar y las relaciones existentes entre los mismos. La importancia del modelo de dominio es que ayuda a comprender mediante un diagrama de clases los conceptos más significativos sobre cómo es en realidad el sistema, sus relaciones y sus procesos. (32)

A continuación el modelo de dominio identificado para la aplicación que se desea obtener:

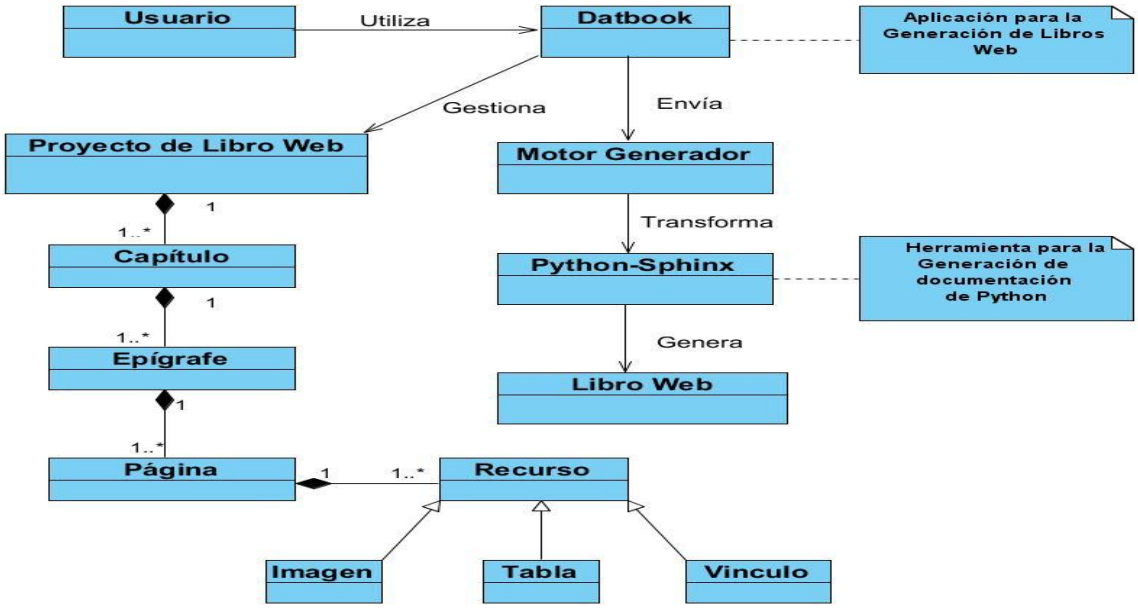


Figura 1. Modelo de dominio

2.1.1 Glosario de Términos

- **Clase Usuario:** Miembro del proyecto Libros Web encargado de confeccionar los libros a generar.
- **Clase Datbook:** Aplicación para la generación de libros web, en la cual se define el proyecto de libro web que será posteriormente generado.
- **Clase Proyecto Libro Web:** Prototipo del futuro Libro Web en él se van a definir los contenidos necesarios a mostrar.
- **Clase Motor Generador de Libros Web:** Es el encargado de obtener la información asociada al Libro Web y convertir la misma a un código entendible por la herramienta Python Sphinx.
- **Clase Libro Web:** Es el resultado final del proyecto libro web por parte de la aplicación el cuál se obtiene mediante la herramienta Python Sphinx.
- **Clase Capítulo:** Primer nivel del proyecto de libro web creado.
- **Clase Epígrafe:** Constituye el segundo nivel del proyecto de libro web creado.
- **Clase Página:** Constituye el último nivel del proyecto de libro web creado, en la cual se adicionará la información de los libros.
- **Clase Python-Sphinx:** Herramienta de generación de documentación de Python.

2.2 Requisitos

En el desarrollo de software se tiene presente la arquitectura, flujo imprescindible para llevar a feliz término el producto final. Esta se basa en los análisis detallados de los requerimientos funcionales que constituyen capacidades o condiciones que el sistema debe cumplir para lograr un objetivo determinado (33) y los requisitos no funcionales que representan las propiedades o cualidades que el sistema debe tener. (29)

De ahí la importancia de la identificación correcta de los requisitos a tener en cuenta para el éxito del sistema. Los requisitos se pueden clasificar en: requisitos funcionales y en requisitos no funcionales.

2.2.1 Requisitos funcionales

RF 1: Crear proyecto libro web: Crear un nuevo proyecto libro web es el primer paso para crear el sitio web con el contenido que se quiere llevar a formato HTML, en el mismo se

debe permitir al usuario introducir datos importantes del libro como: título, autor, versión del proyecto, descripción y colaboradores.

RF 2: Abrir un proyecto libro web existente: Consiste en poder cargar un proyecto existente desde la base de datos.

RF 3: Eliminar proyecto libro web existente: Consiste en eliminar un proyecto deseado que se encuentre visible en la aplicación.

RF 4: Gestionar los contenidos del proyecto libro web: Luego de creado el proyecto libro web se debe definir la estructura del contenido del mismo de acuerdo con los capítulos, epígrafes y páginas con que va a contar.

RF 5: Editar los contenidos de las páginas del libro web: Luego de creado el proyecto libro web y definida la estructura del contenido del mismo se gestionan los contenidos de cada página ya sea introduciéndolo desde el teclado o importándolo desde un documento DOC o PDF.

RF 6: Generar libro web: Una vez terminado el proceso de diseño del libro web se podrá generar el sitio web que no es más que la versión HTML del libro, con la estructura y diseño previamente desarrollados.

2.2.2 Requisitos no funcionales

Estos son los requisitos que repercuten en como el usuario va a interactuar con el sistema. Es la mejor vía para que dicha aplicación sea confiable, rápida, agradable y con una interfaz amigable para las personas a quien va dirigido el producto.

RNF 1: Requisitos de software: El sistema debe ser multiplataforma con enfoque en tecnologías basadas en software no propietario.

RNF 2: Requisitos de interfaz externa: La interfaz externa de la aplicación debe ser lo más amigable posible siempre teniendo presente el uso de los estándares de colores utilizados por el centro DATEC, intentando de dar la impresión a los clientes de ser fácil de utilizar.

RNF 3: Software requerido para desplegar y utilizar la aplicación

- **Paquetes:** apache2, php5 o superior, libapache2-mod-php5, php5-cli, python-sphinx 2.4 o superior, python-pygments, python-docutils 0.6.3, python-jinja2, popler-utils,

odt2text, wv, php5-gd, php5-cli, php5-pgsql, libxml2-utils, libzzip-0.13, zziplib-bin, libzzip-dev.

- **Servidor para instalar el Gestor de Base de Datos**

  SO: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior.

  PostgreSQL: versión 8.4.

  Administrador de PostgreSQL: PgAdmin III.

RNF 4: Hardware requerido para desplegar y utilizar la aplicación web

PC Cliente

Las PC clientes debe cumplir con los siguientes requisitos de hardware:

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 256MB.

PC Servidor

La PC servidor debe cumplir con los siguientes requisitos de hardware:

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 512MB.

RNF 5: Requisito de usabilidad: El sistema debe ser intuitivo y tener un alto nivel de usabilidad permitiendo que usuarios sin mucho conocimiento informático pueda utilizarlo sin problemas. Además el sistema debe utilizar en el diseño del mismo un lenguaje que resulte familiar y asequible al usuario que interactuará con él.

RNF 6: Tipo de usuario final: El usuario final de la aplicación debe ser una persona autorizada, que posea conocimientos básicos en el manejo de una computadora.

RNF 7: Finalidad: El objetivo que persigue la aplicación web es generar de forma sencilla libros web que serán utilizados para plasmar la información de los manuales de usuario de las productos liberados por DATEC.

2.3 Modelo del Sistema

2.3.1 Diagrama de Casos de Uso del Sistema (CUS)

Los diagramas de casos de uso se utilizan para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y entre otros sistemas. Además estos diagramas de casos de uso ilustran los requerimientos del sistema

al mostrar cómo reacciona a eventos que se producen en su ámbito o en él mismo y se especifican las relaciones entre los actores y los casos de uso. (29)

El diagrama de casos de uso siguiente se representan las interacciones que se establecen entre los actores del sistema los cuales son: el administrador, el usuario y el ldap y sus casos de usos asociados.

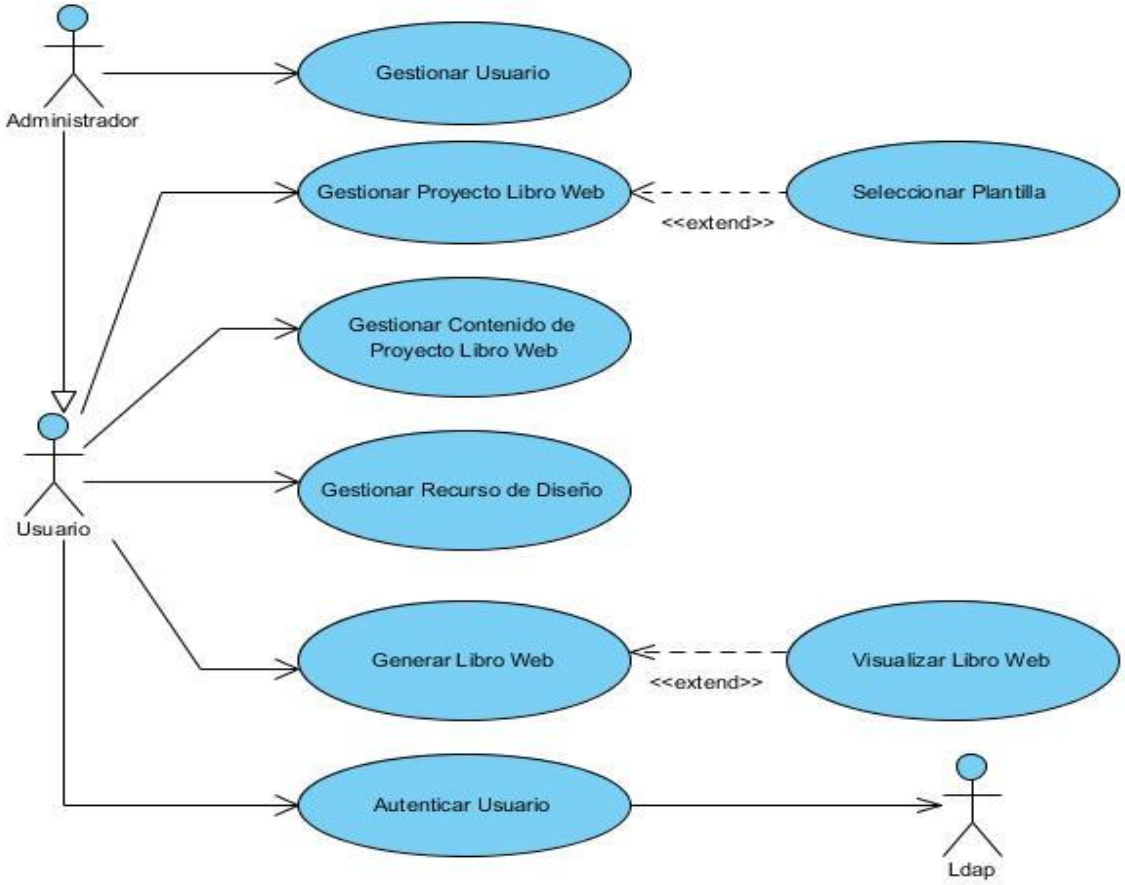


Figura 2. Diagrama de Casos de Uso del Sistema

- **Gestionar Usuario**: permite adicionar, modificar, eliminar, buscar, visualizar trazas y reasignar proyectos, lo cual posibilita gestionar los usuarios que pueden acceder a la aplicación.
- **Gestionar Proyecto:** permite adicionar, eliminar, actualizar, guardar y abrir un proyecto de libro web.
- **Seleccionar Plantilla:** permite seleccionar y visualizar la plantilla a aplicar a un proyecto.

- **Gestionar Contenido del Proyecto:** permite insertar, eliminar y actualizar los contenidos del proyecto del tipo de capítulo, epígrafe y página. Además, es posible editar el contenido de las páginas.
- **Gestionar Recurso de diseño:** permite insertar, eliminar y actualizar los recursos de las páginas del tipo: imagen, tabla y enlaces.
- **Autenticar Usuario:** permite la entrada al sistema ya sea como usuario normal o mediante Ldap.
- **Generar Libro Web:** permite obtener el proyecto de Libro web en un archivo Zip.
- **Visualizar Libro Web:** permite visualizar el estado del proyecto de Libro Web que se generará.

2.3.2 Descripción de los actores del sistema

Los actores del sistema constituyen agentes foráneos, es decir en un principio de desarrollo no se cuenta con ellos, son los que interactuarán con la aplicación solicitando y recibiendo información en aras de obtener un resultado.

Usuario	Actor humano. Encargado de utilizar la aplicación para generar libros web.
Administrador	Actor humano. Responsable de la gestión de los usuarios que van a poder acceder a la aplicación.
Ldap	Servidor ldap de la UCI que brinda la información del usuario que se autentica.

Tabla 2: Actores del Sistema

2.3.3 Descripción de los casos de usos del sistema.

Caso de Uso Generar Libro Web

Caso de Uso:	Generar Libro Web
Actores:	Usuario, Administrador
Resumen:	El caso de uso se inicializa al seleccionar la opción “Generar”, la cual muestra las características finales del libro web y obtiene el sitio web

	del libro web diseñado.
Precondiciones:	Se debe haber gestionado el contenido del libro web.
Referencias	

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. Se selecciona la opción “Generar”. 3. Se escoge la opción “Aceptar”, que obtiene el sitio web con la información gestionada por el usuario.	2. Muestra una interfaz con las características de la edición realizada por el usuario. 4. El sistema visualiza un mensaje satisfactorio.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
1. Se selecciona la opción “Cancelar”	2. Evita que se genere el proyecto y se cierra la interfaz.
Poscondiciones	Se muestra un empaquetado con el proyecto generado.

Tabla 3: Descripción del CU Generar Libro Web

Caso de Uso Gestionar Proyecto de Libro Web

Caso de Uso:	Gestionar Proyecto de Libro Web
Actores:	Usuario, Administrador
Resumen:	El caso de uso se inicia cuando el actor solicita a través del caso de uso base gestionar proyecto libro web. El caso de uso permite crear

	un nuevo proyecto, eliminar el proyecto, abrir un proyecto existente y guardar un proyecto en el que se esté trabajando.
Precondiciones:	Debe existir un proyecto en edición.
Referencias	

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. Se selecciona la opción “Archivo”. 3. Se selecciona la opción “Proyecto”. 4. Se procede a seleccionar opción **Abrir.** 6. Se procede a seleccionar la opción **Salvar**. 8. Se procede a seleccionar la opción **Renombrar.** 10. Se procede a seleccionar la opción **Eliminar.** 12. Se procede a seleccionar la opción **Crear.**	2. Se muestra un menú contextual con la opción “Proyecto”. 5. Se carga el proyecto seleccionado y se visualiza. 7. Se cierra la pantalla y se guarda el proyecto. 9. Dado los atributos insertados por el usuario se modifica el proyecto**.** 11. Se muestra un mensaje de confirmación y se elimina el proyecto**.** 13. Se procede a crear un nuevo proyecto dado los datos insertados y se visualiza.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
1. Se selecciona la opción “Cancelar”	2. Evita que se elimina el proyecto.
Poscondiciones	Se gestiona el proyecto de libro web.

Tabla 4: Descripción del CU Gestionar Proyecto de Libro Web

Interfaces de Usuario CU Gestionar Proyecto.

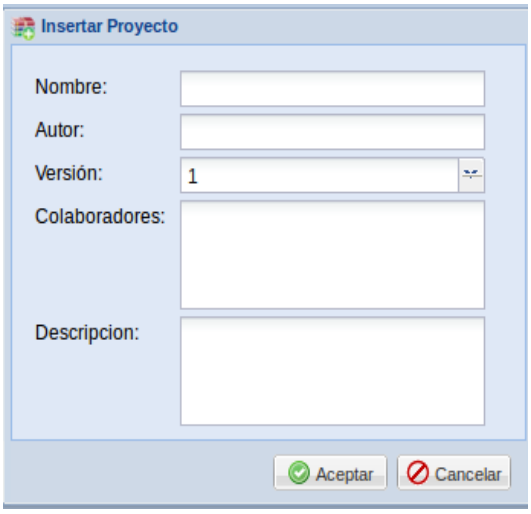


Figura 3: Interfaz crear proyecto

2.4 Patrones

Un patrón se define como una solución probada con éxito que aparece una y otra vez ante determinado tipo de problema en un contexto dado. Los patrones se definen por un nombre, un problema, una solución y las consecuencias de su aplicación. Este define una posible solución correcta para un problema de diseño dentro de un contexto dado, describiendo las cualidades invariantes de todas las soluciones.

Los patrones se categorizan según la escala o nivel de abstracción, sin embargo cada una de las categorías de patrones define un mismo nivel de abstracción o escala de aplicabilidad. (30)

2.4.1 Patrones de Arquitectura

Los patrones de arquitectura de software son patrones de diseño de software que constituyen una vía en la solución de problemas de arquitectura de software. Los mismos poseen un nivel de abstracción mucho mayor que los patrones de diseño. Además brindan una descripción de los elementos y el tipo de relación que tienen, así como las restricciones a para su uso. Los patrones de especifican describiendo los componentes, con sus responsabilidades, relaciones, y las formas en que colaboran. (34)

- **Arquitectura orientada a servicios:** La Arquitectura Orientada a Servicios (SOA) proporciona una metodología y un marco de trabajo para documentar las capacidades del negocio. Además representa un concepto de arquitectura de

software que define la utilización de servicios para dar soporte a los requisitos del negocio. (35)

- **Arquitectura dirigida por eventos:** es un patrón que promueve la producción, detección, consumo y reacción a eventos.
- **Arquitectura en Pipeline:** consiste en ir transformando un flujo de datos en un proceso comprendido por varias fases secuenciales, siendo la entrada de cada una la salida de la anterior. Esta arquitectura es muy común en el desarrollo de programas para el intérprete de comandos, ya que se pueden concatenar comandos fácilmente con tuberías. También es una arquitectura muy natural en el paradigma de programación funcional, ya que equivale a la composición de funciones matemáticas. (35)

A continuación se explican los conceptos y las características fundamentales por la cuales fueron seleccionadas como patrones arquitectónicos el modelo-vista-controlador y el patrón cliente-servidor.

2.4.1.1 Modelo Vista Controlador (MVC)

El patrón Modelo Vista Controlador separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos, principalmente separa la lógica del negocio de la lógica de presentación, ventaja esta que posibilita el mantenimiento de los sistemas, la seguridad y la simplificación en el desarrollo. (36)

Los componentes de este patrón son los siguientes:

- **Modelo:** Encapsula los datos y las funcionalidades. Es independiente de cualquier representación de salida y comportamiento de entrada.
- **Vista:** Muestra la información al usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador.
- **Controlador:** Reciben las entradas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón o pulsaciones de teclas. Los eventos son traducidos a solicitudes de servicio para el modelo o la vista.

Entre sus características se encuentran:

- Existe una clara separación entre los componentes de un programa, lo cual permite implementarlos por separado.
- Existe un API muy bien definido, cualquiera que use el API, podrá reemplazar el Modelo, la Vista o el Controlador, sin aparente dificultad.

- La conexión entre el Modelo y sus Vistas es dinámica, se produce en tiempo de ejecución, no en tiempo de compilación. (36)

Entre sus ventajas se encuentran:

- Posee soporte para múltiples vistas**,** debido a que la Vista se separa del Modelo y no hay ninguna dependencia directa entre ambos, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos al mismo tiempo.
- Proporciona un mecanismo de configuración a componentes complejos mucho más tratable que el puramente basado en eventos.
- Permite un mayor soporte a los cambios, debido a que los requisitos de interfaz tienden a cambiar más rápidamente que las reglas de negocio. Puesto que el modelo no depende de las vistas, la adición de nuevos tipos de vista al sistema generalmente no afectan al modelo. Por tanto, el ámbito del cambio se limita a la vista. (36)

2.4.1.2 Cliente-Servidor (C/S)

Es un modelo de aplicación distribuido en el que las tareas se reparten entre los servidores y los clientes. Un cliente realiza peticiones al software y el servidor da la respuesta. Este modelo se puede aplicar a programas que se ejecutan sobre una sola computadora. (37)

Entre sus características se encuentran:

- La relación que se establece entre el cliente y servidor es a través del intercambio de mensaje. Este intercambio mediante mensajes es un mecanismo para la petición y entrega de solicitudes.
- Las tareas realizadas por el cliente y el servidor poseen diferentes requerimientos en cuanto a velocidad de procesamiento y recursos de cómputo.
- El cliente o remitente de la solicitud es el que espera y recibe las respuestas del servidor. (38)

Entre sus ventajas se encuentran:

- Posee un fácil mantenimiento: al estar distribuidas las funciones y responsabilidades entre varios ordenadores independientes, es posible reemplazar, reparar, actualizar, un servidor, mientras que los clientes no se verán afectados por ese cambio o la afectación será mínimamente.
- Facilita la integración entre sistemas diferentes y comparte información, permitiendo por ejemplo que las máquinas ya existentes puedan ser utilizadas pero utilizando interfaces más amigables el usuario.

- Contribuye a proporcionar a los diferentes departamentos una organización al brindar soluciones locales y permitir la integración de la información. (38)

Los patrones arquitectónicos seleccionados para el desarrollo de la aplicación fueron el modelo-vista-controlador y el cliente-servidor debido a que el primero se encuentra presente en el framework Symfony. Además, entre sus ventajas distintivas es que separa la lógica del negocio de la lógica de presentación brindando una mayor seguridad a la aplicación debido a que el usuario cuenta con un solo punto de accedo al sistema. Con el patrón cliente-servidor es posible el acceso desde otra computadora a la aplicación o sea existe una separación entre el cliente y el servidor tipo lógica, donde el servidor no se ejecuta necesariamente sobre una sola máquina.

2.4.2 Patrones de Diseño

Un patrón de diseño constituye una solución estándar para un problema común de programación en el desarrollo del software. Además es una técnica muy eficaz para flexibilizar el código haciéndolo satisfacer ciertos criterios, así como permite una manera más práctica de describir ciertos aspectos de la organización de un programa. (39)

2.4.2.1 Patrones GRASP

Los patrones GRASP (Patrones de Software para Asignar Responsabilidades) son aquellos que indican y describen los principios de diseño de objetos para la asignación de responsabilidades. Son aquellos que indican la importancia de captar los principios de la responsabilidad y son importantes para el buen diseño del software.

Entre los patrones más significativos se encuentran:

- **Experto:** Debe considerarse como el principio fundamental a tener en cuenta siempre y cuando se esté asignando una responsabilidad a una clase. La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados. Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. Este patrón se encuentra presente en la capa modelo donde existen dos tipos de clases las de abstracción de datos y las de acceso a datos que trabajan directamente con la bases de datos. El framework Symfony usa el ORM propel para el mapeo de las tablas de la base de datos, creándose 4 clases. Las de acceso a datos que trabajan directamente con la base de datos y las de

abstracción de datos que son las que poseen los atributos necesarios para realizar dicha función. Por ejemplo la clase BaseUsuario no conoce los atributos para interactuar con la base de datos tan solo se debe implementar la responsabilidad en la que se le avise a la clase BaseUsuarioPeer que es la contenedora de los datos necesarios para ejecutar la acción.

- **Alta Cohesión:** Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable. La información que almacena una clase debe ser coherente y estar en la mayor medida relacionada con la clase. Estas clases mejoran la claridad y la facilidad con que se entiende el diseño. Este patrón se evidencia al incorporar el marco de trabajo Symfony el cual crea y trabaja con clases que poseen alta cohesión. Una clase encargada de la creación de los usuarios tiene la responsabilidad de definir las acciones correspondientes para adicionar usuarios y a su vez esta colabora con otras clases para realizar operaciones como: acceder a los usuarios e instanciar objetos.
- **Bajo Acoplamiento:** Debe haber pocas dependencias entre las clases. De manera que en caso de producirse una modificación en alguna de las clases, se tenga la mínima repercusión posible en el resto de las clases. Con un bajo acoplamiento las clases de acceso a datos son casi independientes de las de abstracción a datos, es decir que si se modificad las clases del modelo no se afectan las clases del controlador. Este patrón se evidencia en la capa modelo al existir una independencia entre las clases de acceso a datos y las de abstracción a datos. Esto posibilita una mayor reutilización. Es posible modificar las clases del modelo sin que se afecten las del controlador al ser acciones independientes.
- **Controlador:** Este patrón actúa como intermediario entre una determinada interfaz y el algoritmo que la implementa, recibiendo los datos del usuario y enviándolos a las distintas clases según el método llamado. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, para aumentar la reutilización de código. Con este patrón cada clase posee responsabilidades específicas de controlar el flujo de eventos del sistema. Symfony aplica este patrón por tanto tiene una estructura bien organizada desde el index.php del ambiente hasta las actions. Cada clase es esta capa tiene su responsabilidad y es única. En la

aplicación se utiliza este patrón ya que a las clases se le asignan responsabilidades específicas de controlar el flujo de eventos del sistema. (40)

2.4.2.2 Patrones GOF

Los patrones de diseño GOF (Banda de los Cuatro) dan una descripción de clases y objetos que se comunican entre sí, adaptada para resolver un problema general de diseño en un contexto particular. Estos patrones representan la base para solucionar los problemas en el desarrollo del software. (39)

Los Patrones GOF utilizados en la aplicación son:

- **Decorator:** Añade dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender la funcionalidad. Es muy importante debido a que está presente en el layout de la aplicación, el cuál contiene el código HTML y decora según sea la petición las demás plantillas. Este patrón está presente en el layout el cuál contiene el código HTML incluido y va decorando todas las plantillas a utilizar de acuerdo con la petición del usuario.
- **Front Controller:** Es uno de los patrones de diseño más importantes debido a que constituye el único punto de acceso a la aplicación. Es el encargado de recibir las peticiones y controlar y gestionar las peticiones web realizadas por los clientes. Este patrón se visualiza ya que es implementado por el framework Symfony y constituye el único punto de acceso a la aplicación. Además, recibe las peticiones y decide el flujo a seguir. También, carga la configuración de la aplicación y determina las acciones a ejecutarse. (40)

2.5 Estilos Arquitectónicos

Los estilos arquitectónicos definen las reglas generales de organización en términos de un patrón y las restricciones en la forma y la estructura de un grupo numeroso y variado de sistemas de software, es decir la descripción de los tipos de componentes y los patrones de interacción entre ellos. Un estilo provee una guía y análisis para crear una clase amplia de arquitecturas en un dominio específico donde los patrones se enfocan en solucionar los problemas más pequeños y más específicos del sistema. Con la utilización de los estilos se garantizan la existencia de restricciones que proporciona visibilidad a determinados aspectos de la arquitectura, evitando que sean ignorados o violados. (41)

Clasificación de los estilos arquitectónicos:

Estilos de Flujo de datos

- Tuberías y Filtros

Estilos centrados en datos

- Arquitecturas de Pizarra o repositorio

Estilos de Llamada y Retorno

- Modelo – Vista – Controlador (MVC)
- Arquitectura en Capas
- Arquitectura Orientada a Objetos
- Arquitectura basada en Componentes

Estilo de Código Móvil

- Arquitectura de Máquinas Virtuales

Estilos Peer–To–Peer

- Arquitectura basada en Eventos
- Arquitecturas Orientas a Servicios (SOA)

2.6 Diseño del Sistema.

El modelo de diseño constituye una abstracción al modelo de implementación y del código fuente, constituyendo la entrada principal la realización de la implementación del sistema que desea desarrollar. Consta de un conjunto de objetos que describen las realizaciones de los casos de uso y sus relaciones, incluyendo además los mensajes que pueden enviarse entre ellos. Se centra en los impactos que producen en el sistema los requisitos funcionales y no funcionales. Este modelo está compuesto por paquetes, subsistemas de diseño representados en una breve jerarquía, diagramas de clase del diseño y diagramas de interacción.

2.6.1 Realizaciones de los casos de uso.

Las realizaciones de los casos de uso son las colaboraciones en el modelo de diseño que describen como se realizara uno o varios casos de uso específicos y como se ejecutaran en términos de casos de uso del diseño, en el cual intervienen los diagrama de clases y los diagramas de interacción.

La realización del caso de uso no es más que concebir o llevar a cabo la demostración de cómo puede ser implementado un caso de uso determinado a través de la colaboración entre objetos.

Entre los pasos fundamentales para lograr un diagrama de colaboración se cuenta:

- Ir a través de los casos de uso del sistema simulando los mensajes enviados entre objetos y almacenando los resultados en diagramas de interacción.
- Agregar operaciones a los objetos que reciben los mensajes.

2.6.2 Diagrama de clases del diseño

Una clase del diseño en una abstracción de una clase o construcción similar en la implementación del sistema. El lenguaje que se utiliza en dichas clases es el mismo que se emplea para la implementación del sistema; se especifican los atributos y las operaciones; se pueden realizar interfaces si tienen sentido para la programación y los métodos tienen correspondencia con los métodos de la programación.

En resumen el modelo del diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en los requisitos funcionales, junto con otras restricciones relacionadas en el entorno de implementación, tienen un impacto en el sistema a considerar y sirve de abstracción a la implementación y al código fuente del sistema.

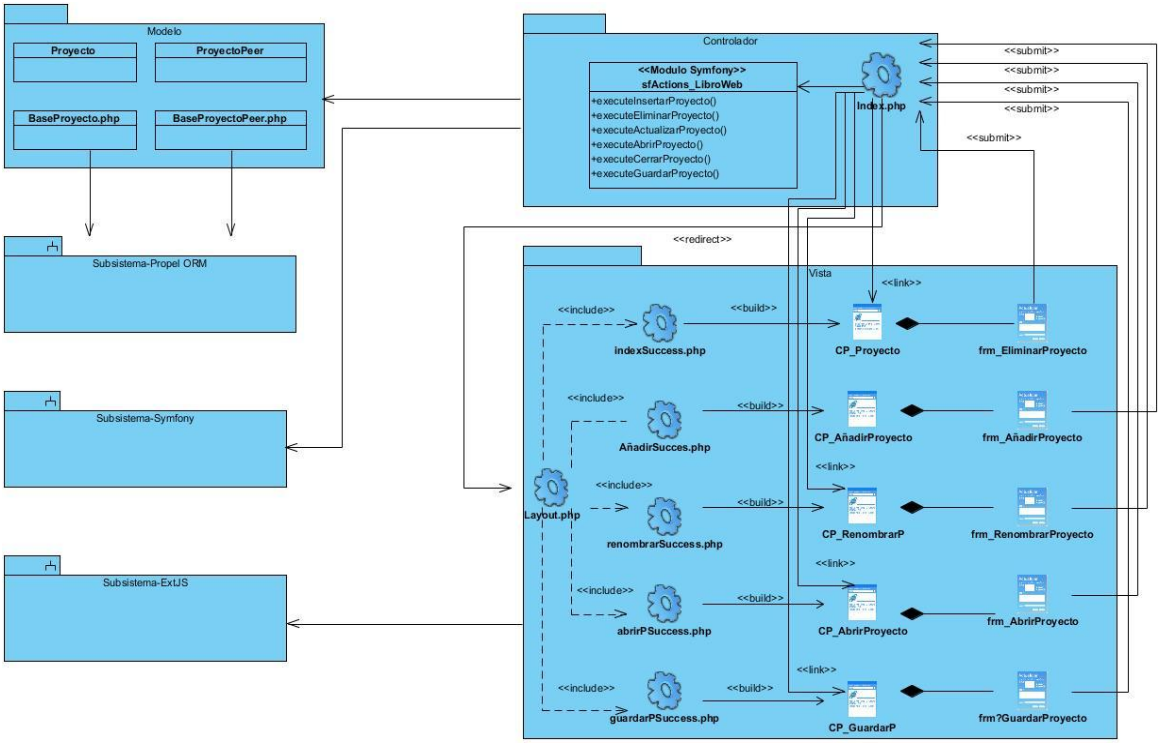


Figura 4. Diagrama Clases del diseño Gestionar Proyecto Libro Web

- **Controlador:** la capa controladora es la que recibe las peticiones de solicitud al modelo y realiza las acciones necesarias para obtener los datos del modelo y enviarlos a la vista que es la encargada de mostrar la información al usuario.
- **sfLibroWebActions:** esta clase contiene todas las acciones en dependencia del caso de uso que se trate, en este caso contiene todas las acciones referentes al Caso de Uso Gestionar Suscripción.
- **datbook.php:** esta clase es conocido como el controlador frontal y constituye el punto de entrada a la aplicación, es la que recibe las peticiones y decide el flujo que debe seguir para dar respuesta a la petición del usuario.
- **Modelo:** esta capa contiene todas las clases de la base de datos. El mapeo de la base de datos se realiza mediante Propel el cuál obtienen cuatro clases por cada tabla de la base de datos estas son: proyecto, proyectoPeer, baseProyecto y baseProyectoPeer.
- **proyecto, proyectoPeer:** representan las clases de acceso a datos y son responsables de interactuar con las clases de abstracción de datos, devuelven los objetos que necesitan los controladores en su forma original.
- **baseProyecto, baseProyectoPeer:** representan las clases de abstracción de datos. Estas clases se encargan de la abstracción de datos y de realizar las operaciones con la base de datos.
- **Vista:** en la capa de la vista se encuentran las interfaces de usuario de la aplicación, con las cuales los usuarios pueden interactuar.

2.7 Diagramas de interacción

Diagramas de Interacción Modelan el comportamiento dinámico del sistema; el flujo de control en una operación. Describe la interacción entre objetos; estos objetos interactúan a través de mensajes para cumplir ciertas tareas. Estas interacciones proveen un comportamiento y en varias ocasiones implementan un Caso de Uso. Los diagramas de interacción se pueden clasificar en diagramas de secuencia y en diagramas de colaboración.

Es decir que estos diagramas describen como un grupo de objetos colaboran entre sí para lograr un fin mediante el paso de mensajes entre ellos dentro del caso de uso.

2.7.1 Diagrama de Secuencia

Los diagramas de secuencia muestran las interacciones expresadas en función del tiempo donde se puede apreciar los objetos que intervienen y los mensajes que se establecen entre ellos. (42)

A continuación se expresan los diagramas de secuencias más relevantes:

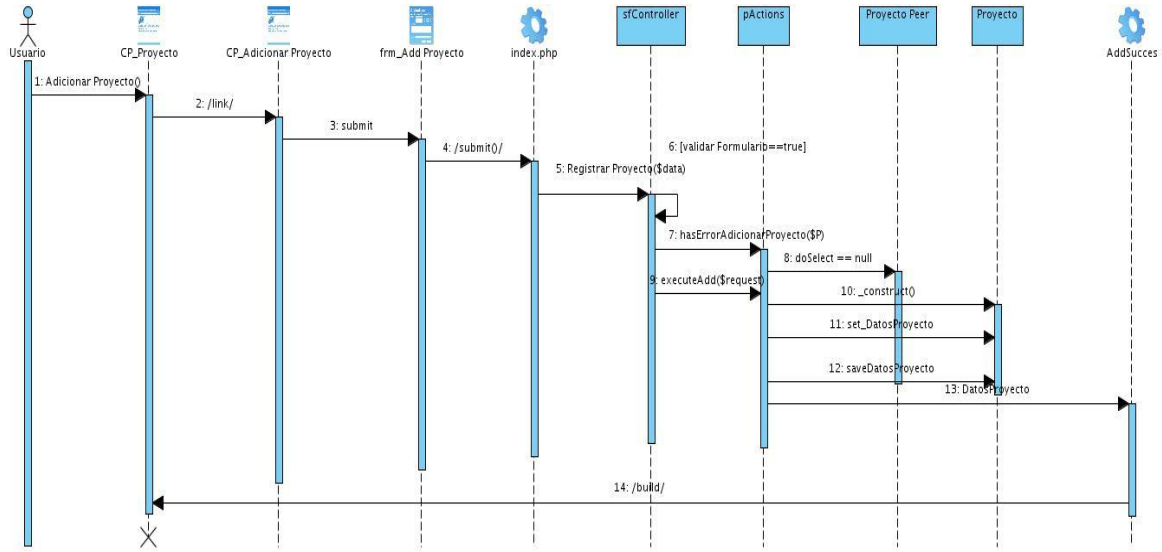


Figura 5. Diagrama Secuencia Crear Proyecto

El diagrama de secuencia visualiza los pasos para la creación de un proyecto de Libro web. El usuario introduce los datos necesarios para crear un proyecto, luego el sistema valida que los datos introducidos son correctos en caso afirmativo se muestra un mensaje de error. Si es posible insertar el proyecto se envía un mensaje de operación satisfactoria.

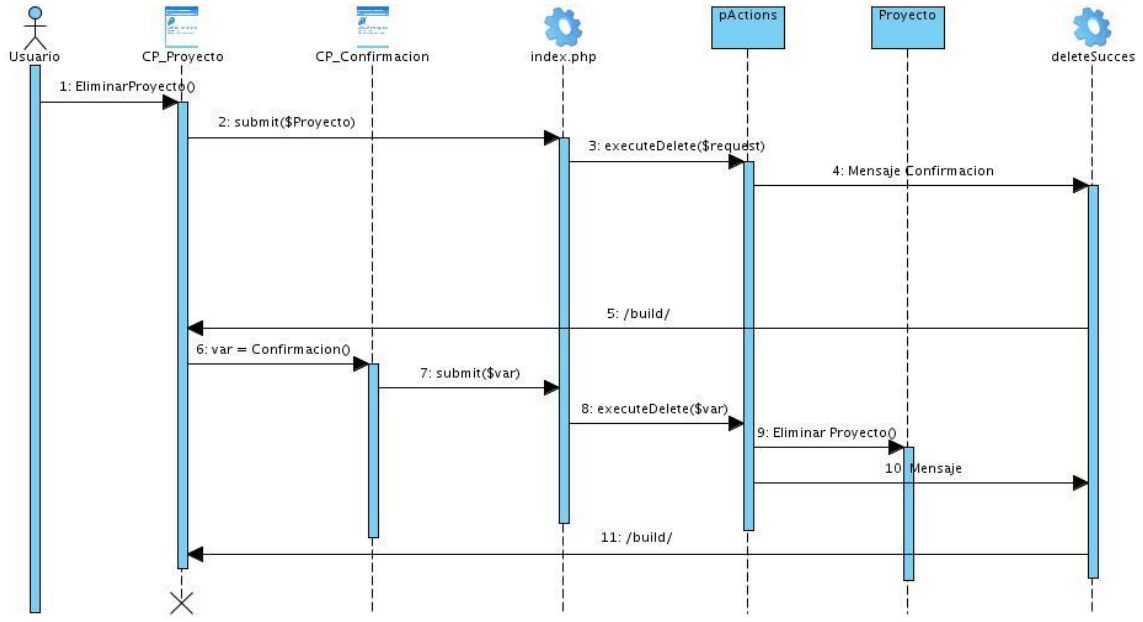


Figura 6. Diagrama Secuencia Eliminar Proyecto

El escenario eliminar proyecto el usuario selecciona el proyecto que desea eliminar, luego el sistema consulta la base de datos para ver si existe el proyecto en cuestión, en caso de no existir u ocurrir algún problema en la eliminación se muestra un mensaje al usuario indicándole el error. En caso de que exista en la base de datos se elimina el proyecto deseado.

2.8 Modelo de Datos

El modelo de datos permite describir la estructura de los datos que se encuentran en la base de datos y las relaciones que se establecen entre las tablas.

En la modelo de datos siguiente se aprecia las relaciones existentes entre las tablas de la base de datos asociada a la aplicación:

- **Usuario:** Permite almacenar la información de los usuarios que pueden utilizar la aplicación para la generación de libros web.
- **Proyecto:** Permite almacenar la información de los proyectos creados por los usuarios.
- **Trazas:** Permite almacenar la información de las trazas que se generan de acuerdo a las acciones que realiza un usuario en la aplicación.

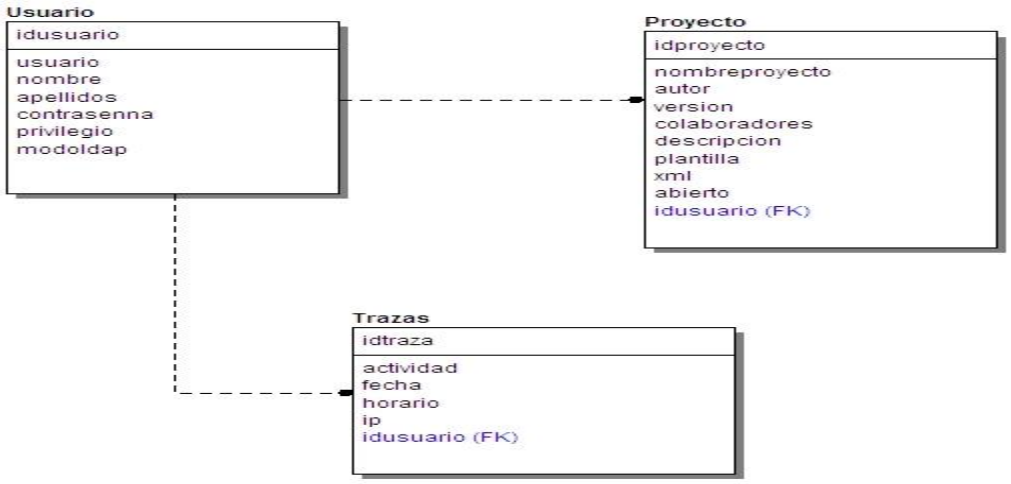


Figura 7. Modelo de datos

2.9 Conclusiones

En este capítulo se analizaron las características principales de la aplicación. Asociado a estas características se definió el modelo de dominio para comprender la estructura del sistema y se plasmaron los requisitos funcionales y no funcionales del sistema, los cuales serán funcionalidades del la aplicación. También se obtuvieron los actores participantes en el sistema y se definió en forma de descripción textual los casos de uso más importantes del sistema. Luego de definir el análisis y el diseño necesario como punto de partida en aras de un mejor entendimiento de las características de sistema mediante la descripción de los casos de usos y los diagramas pertinentes importantes para la próxima iteración. Se obtuvieron 24 requisitos funcionales agrupados en 7 casos de uso representados en el diagrama de casos de uso del sistema y 7 requisitos no funcionales. También se definieron los patrones de diseño y de arquitectura a utilizarse en el desarrollo del trabajo.

CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA

Introducción

Una vez terminado el análisis y el diseño del sistema ya se cuenta con una visión más general de la arquitectura del sistema que se quiere obtener. En este capítulo se define el diagrama de despliegue así como el diagrama de componentes. Uno de los artefactos más importantes que se generará en este capítulo es el modelo de implementación, el cual está conformado por el diagrame de despliegue y el diagrama de componentes. Además se abordan las tipos de pruebas realizadas a la aplicación con el fin de evidenciar su correcto funcionamiento.

3.1 Diagrama de despliegue.

La vista de despliegue describe los principales nodos físicos, ordenadores, así como los dispositivos que se necesitan para configurar la plataforma que pueda soportar la implementación del sistema. Para su modelación se cuenta con el diagrama de despliegue que es el encargado de contener la distribución de los componentes de software en los nodos físicos.

A continuación de muestra del diagrama de despliegue que se corresponde con el sistema que se desea implementar.

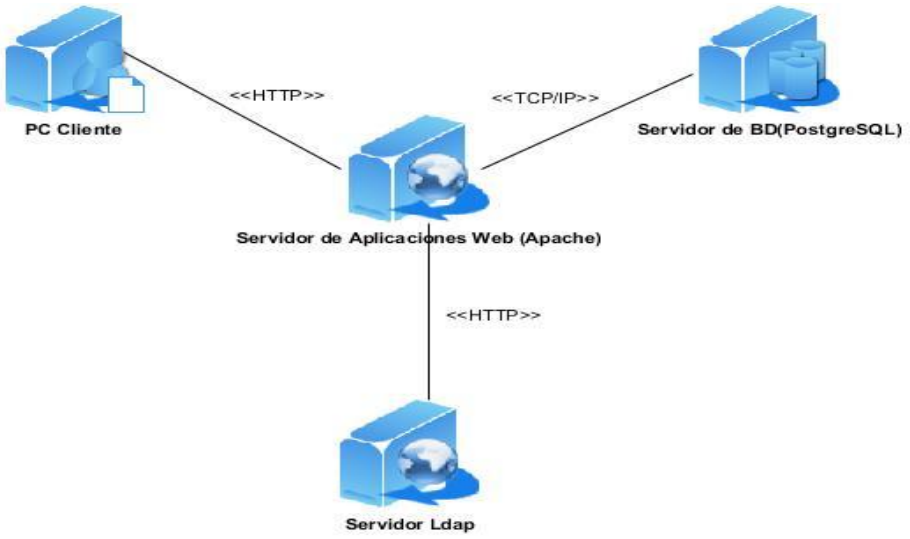


Figura 8. Diagrama de Despliegue

- **Estaciones de trabajo con la Aplicación Cliente**

  Está definida por las estaciones de trabajo que el usuario utilizará para acceder a la aplicación Web

- **Servidor de Aplicación**

  El servidor de aplicación es utilizado para la publicación de la aplicación. Es la herramienta principal para ejecutar la lógica de negocio en el lado del servidor y es el encargado de ejecutar el código de las páginas servidor. En el diagrama de despliegue se utiliza el servidor de aplicación Apache.

  Este servidor de aplicación es compatible con los sistemas operativos: Unix, GNU, Microsoft Windows y Linux. Además, es gratuito, de código abierto y entre sus características fundamentales se encuentra la flexibilidad que le otorga su estructura modular.

- **Protocolos de Comunicación**

  Un protocolo de comunicación es un conjunto de reglas establecidas entre dos dispositivos para permitir la comunicación entre ambos.

- **Conexión HTTP**

  Es el protocolo utilizado entre los browser de los clientes y el servidor Web. Este elemento de la arquitectura representa un tipo de comunicación no orientado a la conexión entre clientes y servidor.

- **TCP/IP**

  Es la base del Internet que sirve para enlazar computadoras. El protocolo TCP/IP es utilizado para establecer la conexión entre el servidor de aplicación y el servidor de base de datos.

- **Servidor Ldap**

  Se hace referencia al servidor que brinda los servicios de autenticación en la UCI. Se realiza una petición a este servidor y este devuelve los parámetros del usuario de la petición.

- **Servidor de Base de Datos**

Se hace referencia al gestor de bases de datos donde se encuentran los datos necesarios para el trabajo con el sistema. El servidor de base de datos elegido es PostgreSQL 8.4, que es un gestor de licencia no propietaria o sea libre y muy potente.

3.2 Diagrama de Componentes

Los diagramas de componentes muestran las dependencias lógicas ente componentes software. Estos diagramas son utilizados para describir la vista de implementación estática de un sistema determinado. Los mismos poseen un nivel mucho más alto de abstracción en correspondencia con el diagrama de clases y representan una o varias clases, interacciones o colaboraciones.

Los componentes físicos incluyen archivos, bibliotecas, ejecutables o paquetes. Estos son muy utilizados para modelar y documentar la arquitectura al permitir visualizar la organización y las dependencias entre un conjunto de componentes. Además los componentes son partes modulares del sistema que pueden desplegarse y reemplazarse. También encapsulan implementación y un conjunto de interfaces proporcionado la realización de los mismos.

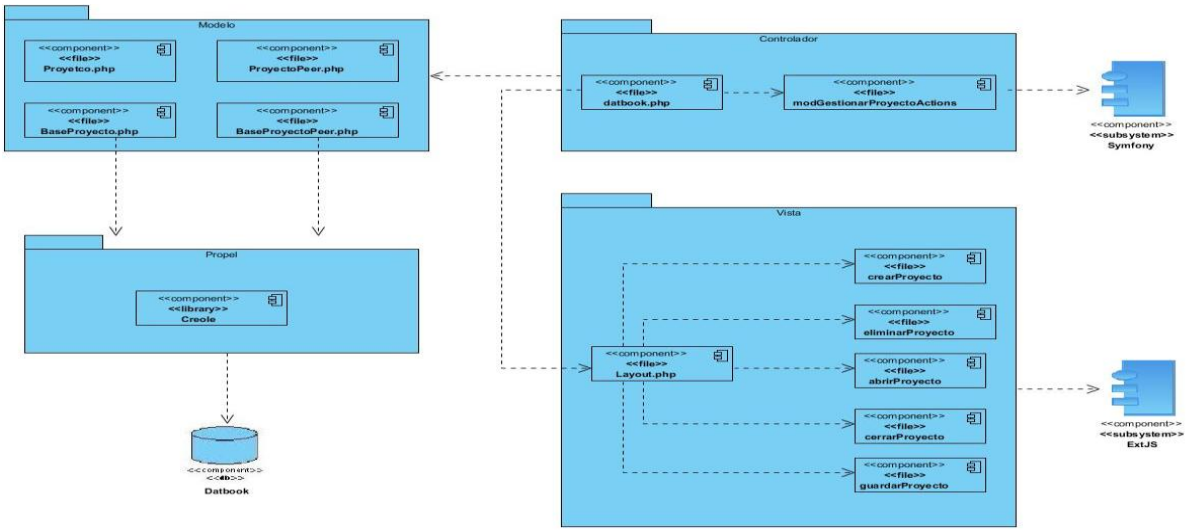


Figura 9. Diagrama de componente Gestionar Proyecto

- **Vista:** en la capa de la vista se cuenta con las interfaces de usuario asociadas al caso de uso Gestionar Proyecto.

- **Controlador:** contiene todas las acciones asociadas al caso de uso Gestionar Proyecto, para darle respuesta a las peticiones del usuario que llegan a través de datbook.php.
- **Modelo:** contiene todas las tablas de la base de datos asociadas al caso de uso Gestionar Proyecto.
- **ExtJS:** paquete que se utiliza en la capa Vista para el diseño de las interfaces.
- **Symfony:** paquete que se utiliza en la capa Controlador para la implementación de las acciones.
- **Propel:** paquete que se utiliza en la capa Modelo para convertir las tablas de la base de datos en clases.

3.3 Mecanismos de Implementación

La aplicación se ha desarrollado bajo la arquitectura del Modelo-Vista-Controlador, arquitectura tomada en cuanta por el marco de trabajo Symfony 1.4.8 utilizado para el desarrollo en el lado del servidor. Además el framework ExtJs 3.3.2 fue el seleccionado para el desarrollo en el lado del cliente así como las vistas de las interfaces. En la capa de modelo o acceso a datos se utilizó para el mapeo relacionar el ORM Propel y el flujo de eventos e intercambio de información entre el cliente y el servidor se estableció para el formato JSON. Se utilizó la tecnología XML para los proyectos creados por el usuario.

A continuación se plasman las implementaciones más importantes de la aplicación:

```
public function executeGenerarLibroWeb() {
    try {
        $file = new DOMDocument();
        $file->load("js/xml-tree.xml");
        $proyecto = $file->getElementsByTagName("proyecto")->item(0);
        $this->GenerarConfPY();
        $this->GenerarIndiceRST();
        $this->GenerarCapitulosRST();
        $this->GenerarEpigrafeRST();
        $this->GenerarPaginaRST();
        $this->CopiarImagenes();
        $entrada = "/var/www/Tesis/Datbook/web/uploads/generado/" . $proyecto->getAttribute("nombre") . "/s
        $salida = "/var/www/Tesis/Datbook/web/uploads/generado/" . $proyecto->getAttribute("nombre") . "/bu
        exec("sphinx-build -b html " . $entrada . " " . $salida);
        return $this->renderText(json_encode(array(
                    'success' => true,
                    'msg' => 'El proyecto se ha generado correctamente'
                )));
    } catch (Exception $ex) {
        return $this->renderText(json_encode(array(
                    'failure' => true,
                    'msg' => 'A ocurrido un error en la generaci&oacuten'
                )));
    }
```

Figura 10. Implementaciones Caso de Uso Generar Libro Web.

Se visualiza el código para generar un proyecto de libro web. Se crea la estructura del proyecto y después se ejecuta la herramienta Python-Sphinx para obtener el proyecto de libro web en formato HTML.

```
public function executeGuardarproyectoopen(sfWebRequest $request) {
    $btnguardar = json_decode($request->getParameter('guardarproyecto'));
    try {
        if ($btnguardar->idsave == yes) {
            $file = new DOMDocument();
            $file->load('js/xml-tree.xml');
            $xml = $file->getElementsByTagName('proyecto')->item(0);
            $id = $xml->getAttribute('id');
            $cr = new Criteria();
            $cr->add(ProyectoPeer::IDPROYECTO, $id);
            $proy = ProyectoPeer::doSelectOne($cr);
            $proy->setXml($file->saveXML());
            $proy->save();
            $result = array(
                'success' => true,
                'msg' => 'Guardado satisfactoriamente');
            return $this->renderText(json_encode($result));
        }
    } catch (Exception $exc) {
        return $this->renderText(json_encode(array(
                    'failure' => true,
                    'msg' => $exc->getMessage() . ' - ' . $exc->getLine()
                )));
    }
}
```

Figura 11. Implementación Caso de Uso Guardar Proyecto

Se visualiza el código para guardar un proyecto de libro web. El sistema busca el proyecto en la base de datos. En caso de no aparecer se muestra un mensaje de error y en caso de aparecer se guarda el proyecto.

3.4 Estrategias de codificación. Estándares y estilos a utilizar

Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Se hace necesario establecer un estándar de codificación para garantizar la coordinación entre los programadores.

Aunque la legibilidad y la estabilidad son el resultado de muchos factores, una faceta del desarrollo de software en la que todos los programadores influyen especialmente es en la técnica de codificación. El mejor método para asegurarse de que un equipo de programadores mantenga un código de calidad es establecer un estándar de codificación sobre el que se efectuarán luego revisiones del código de rutinas.

Generalmente no afectan a la funcionalidad de la aplicación, pero sí contribuyen a una mejor compresión del código fuente.

El uso de los estándares de codificación en la investigación, están presentes de la siguiente manera.

Notación Camello

La notación Camello consiste en escribir los identificadores con la primera letra de cada palabra en mayúsculas y el resto en minúscula. Se llama notación "Camello" porque los identificadores recuerdan las jorobas de un camello. Existen dos variantes:

- Notación camello mayúscula: En esta variante la primera letra también es mayúscula.
- Notación camello minúscula: La primera letra es minúscula.

Ejemplo de notación camello minúscula:

```
$datosProyecto = json_decode($request->getParameter('proyecto'));
$nombre = $datosProyecto->nombre;
$autor = $datosProyecto->autor;
$version = $datosProyecto->version;
$colaboradores = $datosProyecto->colaboradores;
$descripcion = $datosProyecto->descripcion;
```

Figura 12. Ejemplo notación camello

Notación Pascal

La Notación o convención Pascal, nos presenta una forma de definir nuestras variables al momento de programar, iniciando con la primer letra Mayúscula y la letra inicial de las palabras siguientes contenidas en la variable.

Es utilizada generalmente en los nombre de espacios los cuales se dividen por un "punto".

Ejemplo:

```
Ext.proy.AddProyecto=Ext.extend(Ext.Window,{

    constructor:function(){
```

Figura 13. Ejemplo notación pascal

3.5 Pruebas

Las pruebas son una actividad en la cual el sistema o los componentes son ejecutados bajo ciertas condiciones o requerimientos especificados donde los resultados son observados y registrados para realizar una evaluación de algún aspecto del sistema o del componente. Entre las buenas prácticas para realizar pruebas al sistema es que

dichas pruebas se realicen por un actor externo que no sea el desarrollador de la aplicación. (43)

Objetivos de las pruebas

Las pruebas buscan encontrar los posibles fallos en la implementación, en la usabilidad y en la calidad de un programa determinada para validar su correcto funcionamiento. Entre los objetivos de la realización de las pruebas se tiene:

- Detectar defectos en la aplicación.
- Verificar que todos los requisitos se han implementado satisfactoriamente.
- Identificar y asegurar que los defectos encontrados se han corregido antes de entregar el producto al usuario o cliente final.
- Diseñar los casos de prueba que permitan identificar diferentes clases de errores utilizando la menor cantidad de tiempo y esfuerzo. (43)

3.5.1 Tipos de pruebas

Las pruebas poseen una estructura en la que se define el objetivo de la prueba a realizar, la descripción detallada de la prueba y la técnica a utilizar. Entre las pruebas que se pueden encontrar se encuentran las siguientes:

- **Pruebas unitarias**: estas pruebas se encargan de probar una clase en específico, realizando pruebas a cada uno de los métodos presentes en ella. Es una forma de probar el correcto funcionamiento de un módulo de código de una clase. Además, permite validar que cada uno de los módulos funciona correctamente por separado
- **Pruebas funcionales**: estas pruebas van enfocadas a una funcionalidad completa. Las pruebas funcionales se realizan mediante el diseño de modelos de prueba que buscan evaluar cada una de las opciones con las que cuenta la aplicación.
- **Pruebas de regresión**: Las pruebas de regresión son una estrategia de prueba en la cual las pruebas que se han ejecutado anteriormente se vuelven a realizar en la nueva versión modificada, para asegurar la calidad después de añadir la nueva funcionalidad.
- **Pruebas de validación:** Proporciona una seguridad final de que el software satisface todos los requerimientos funcionales y de rendimiento. Además, valida los requerimientos establecidos comparándolos con el sistema que ha sido construido.

- **Pruebas de Caja Negra:** estas pruebas van dirigidas a la interfaz externa de una aplicación.
- **Pruebas de aceptación**: son un tipo de prueba funcionales, pero basadas en el cliente de la aplicación en aras de demostrar al cliente que la funcionalidad está terminada y funciona correctamente. (44)

Se seleccionaron como tipos de pruebas a realizar las pruebas de unitarias y funcionales. Las pruebas funcionales tienen como objetivo probar que la aplicación desarrollada cumple con las funciones específicas para los cuales han sido creados. Las pruebas unitarias tienen como objetivo asegurar la calidad del código entregado y que los módulos funcionen correctamente.

3.5.2 Pruebas de Caja Negra

La prueba de Caja Negra se centra principalmente en los requisitos funcionales del software o sea que pertenecen al tipo de pruebas funcionales. Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos.

Para preparar los casos de pruebas hacen falta un número de datos que ayuden a la ejecución de estos casos y que permitan que el sistema se ejecute en todas sus variantes, pueden ser datos válidos o inválidos para el programa según si lo que se desea es hallar un error o probar una funcionalidad. Los datos se escogen atendiendo a las especificaciones del problema, sin importar los detalles internos del programa, a fin de verificar que el programa funcione bien.

Casos de prueba

Los casos de prueba son un conjunto de condiciones mediante las cuales se comprueba si los requisitos funcionan correctamente. La descripciones del caso de prueba realizado contienen el nombre del caso de uso, la descripción del caso de uso, así como la respuesta del sistema y las variables asociadas a dicho caso de uso.

Caso de Prueba del Caso de Uso: Gestionar Proyecto

Descripción general: El caso de uso comienza al seleccionar la opción “Proyecto”, permite mostrar las opciones en un menú desplegable asociadas a la gestión de un proyecto de libro web.

Condiciones de ejecución: Que exista al menos un proyecto abierto en el parte de edición de proyectos.

Escenario: Adicionar Proyecto

Escenario	Descripción	Nombre del Proyecto	Autor del Proyecto	Versión	Colaborador(es)	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Adicionar proyecto correctamente.	Se adiciona un nuevo proyecto correctamente	V SIGICEM	V Marcos Antonio Reyes Medina	V 1,0	V Yoandy Doble	V Proyecto	Se muestra el mensaje " El proyecto ha sido creado correctamente".	1- El usuario selecciona en la interfaz principal el botón "Proyecto" desplegándose un menú con las opciones de un proyecto. 2-Se selecciona "Adicionar Proyecto¨. 3-Se visualiza una interfaz para adicionar un proyecto. 4- Se insertan los datos correctamente. 5- Se selecciona el botón Aceptar.
EC 1.2 Adicionar proyecto con campos vacíos.	Se adiciona un nuevo proyecto y se dejan campos que son obligatorios en blanco.	NA	NA	NA	NA	NA	Se muestra un mensaje:"Existen campos vacíos"	1- El usuario selecciona en la interfaz principal el botón "Proyecto" desplegándose un menú con las opciones de un proyecto. 2-Se selecciona "Adicionar Proyecto¨. 3-Se visualiza una interfaz para

								adicionar un proyecto. 4- Se dejan sin insertar algunos datos necesarios 5- Se selecciona el botón Aceptar.
EC 1.3 Adicionar proyecto con datos no válidos.	Se adiciona un nuevo proyecto con datos no válidos.	NA (&swws NA	NA (&swws NA	NA user123 NA	NA 33345 NA	NA erd34 NA	Se muestra un mensaje:"El nombre de proyecto o el autor poseen errores".	1- El usuario selecciona en la interfaz principal el botón "Proyecto" desplegándose un menú con las opciones de un proyecto. 2-Se selecciona "Adicionar Proyecto¨. 3-Se visualiza una interfaz para adicionar un proyecto. 4- Se insertan datos no válidos. 5- Se selecciona el botón Aceptar.
EC 1.4 Cancelar	Se selecciona la opción cancelar Adicionar proyecto.	NA NA	NA NA	NA NA	NA NA	NA NA	Se cancela la opción de Adicionar proyecto.	1- El usuario selecciona en la interfaz principal el botón "Proyecto" desplegándose un menú con las opciones de un proyecto. 2-Se

								selecciona "Adicionar Proyecto¨. 3-Se visualiza una interfaz para adicionar un proyecto. 4- Se selecciona el botón Cancelar.

3.5.3 Pruebas unitarias

Estas pruebas constituyen una forma de probar el correcto funcionamiento de un módulo de código determinado. Es decir que tiene como principal utilización la validación de que cada módulo funcione correctamente por separado. Como objetivo primordial para establecer este tipo de prueba es necesario conocer que permite simplificar la integración. Además documenta de una forma u otra el código al representar y acotar cual es la funcionalidad de dicho módulo y facilita la localización de los errores aunque por si solas no garantizan que el sistema funcione correctamente**.**

Se realizaron 7 pruebas unitarias a los métodos de las diferentes clases mediante el objeto Lime que viene por defecto en el framework Symfony para la realización de las pruebas de este tipo pudiéndose validar el correcto funcionamiento de los módulos.

```
1..5
# Realización de  la prueba para la clase addProyecto
ok 1 - Comparando con el id del proyecto
ok 2 - Comprarando con el nombre del proyecto
ok 3 - Comparando nombre de autor del proyecto
ok 4 - Comparando colaboradores
ok 5 - Comparando si está abierto el proyecto
# Looks like everything went fine.
root@lp06-308-23:/var/www/Datbook#
```

Figura 14. Prueba unitaria clase Usuario

```
1..3
# Comienza la prueba para la clase Contenido
ok 1 - Comparando nombre del contenido
ok 2 - Comprarando tipo contenido
ok 3 - Comparando title del contenido
# Looks like everything went fine.
root@lp06-308-23:/var/www/Datbook#
```

Figura 15. Prueba unitaria addContenido

Se realizó la liberación del sistema por el grupo de calidad del centro DATEC con el objetivo de lograr validar y alcanzar una mejor visibilidad en cuanto a errores del tipo de no correspondencia, funcionales y de ortografía. Esta liberación se basó en encontrar no conformidades entre la documentación presentada y la aplicación obtenida. Todos los errores encontrados en las no conformidades fueron erradicados correctamente. En la siguiente tabla se plasman los resultados obtenidos en las pruebas exploratorias realizadas por el grupo de calidad. Ver (**Anexo 8** 1. Acta de liberación de la aplicación).

	No Correspondencia	Ortografía	Funcionalidad
1ra Iteración	20	4	3
2da Iteración	15	0	1
3ra Iteración	0	0	0
Totales	35	4	4

Tabla 5. Resultados de la liberación

3.6 Conclusiones

En el presente capítulo se representaron los diagramas de componentes asociados a los casos de uso. Además, se ejecutaron los diferentes tipos de pruebas especificados para validar el correcto funcionamiento de la aplicación. Se realizaron pruebas de tipo caja negra y las pruebas unitarias. También se plasmaron los mecanismos de implementación acordes con el análisis y el diseño y el diagrama de despliegue del sistema. Además se realizaron los casos de prueba, basados en casos de uso que sirvieron como documentos de apoyo y guía al grupo de calidad que llevó a cabo el proceso de liberación. Originando como resultado 35 no conformidades del tipo de no correspondencia, 4 ortográficas y 4 de funcionalidades, en 3 iteraciones. Todas estas no conformidades fueron erradicadas en tiempo.

CONCLUSIONES

Una vez culminada la investigación es posible afirmar que se cumplieron todos los objetivos trazados para la misma:

- Se realizó un minucioso estudio sobre las herramientas generadoras de libros electrónicos y libros web existentes en Cuba como en el mundo, lo que posibilitó seleccionar la herramienta Python-Sphinx como la idónea a utilizarse en la realización de este trabajo.
- La metodología utilizada permitió la correcta obtención de los artefactos y diagramas generados en todas las fases, así como la realización del análisis y diseño de la aplicación.
- La Aplicación para la generación de libros web se implementó de acuerdo a los requisitos funcionales identificados erradicándose los problemas expuestos en la situación problémica.
- Se validó el correcto funcionamiento de la aplicación, a través de las pruebas de caja negra realizadas mediante los casos de prueba y las pruebas unitarias.

RECOMENDACIONES

Tomando como punto de partida los resultados obtenidos con la realización de este trabajo de diploma, se hacen las siguientes recomendaciones:

- Agregar nuevas funcionalidades que permitan simplificar aún más la edición de los libros web por parte de los usuarios.
- La utilización de la aplicación para la generación de los manuales de usuario de los proyectos liberados por DATEC.

REFERENCIA BIBLIOGRÁFICA

1. **Erosky, Erosky.** Erosky Consumer. [En línea] [Citado el: 27 de Octubre de 2011.] http://revista.consumer.es/web/es/20031101/internet/.

2. **Olivera Hernández, Rachel y Gómez Piñeiro, Karel Carmen.** Biblioteca UCI. [En línea] 2012. [Citado el: 26 de Octubre de 2011.] http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-detail.pl?biblionumber=8829.

3. **Pérez Salomón, Omar.** [En línea] 2008. [Citado el: 7 de Octubre de 2011.] http://www.ecured.cu/index.php/Historia_de_la_Informática_en_Cuba..

4. **Presman, Roger S.** *Ingeniería de Software. Un enfoque práctico.* 2005.

5. **Luján Mora, Sergio.** *Programación en Internet: Clientes Web.* s.l. : 1ª edición. Editorial Club Universitario., 2001.

6. **Solcre, Solcre.** [En línea] 2009. [Citado el: 8 de Octubre de 2011.] http://www.solcre.com/files/ventajas_de_las_aplicaciones_web.pdf..

7. **Conneally, Tim.** [En línea] 2008. [Citado el: 13 de Octubre de 2011.] http://www.betanews.com/article/Sonys-Reader-opens-support-for-more-publishers-formats/1216928081.

8. **Bottou, Léon, y otros.** Journal of Electronic Imaging. [En línea] 1998. [Citado el: 13 de Octubre de 2011.] http://leon.bottou.org/publications/pdf/jei-1998.pdf.

9. **IDPF.** [En línea] 11 de septiembre de 2007. Open Publication Structure (OPS) 2.0.

10. **Luján Mora, Sergio.** *Programación de aplicaciones web: historia, principios básicos y clientes web.* s.l. : 1ra Edición Editorial Club Universitario, 2002.

11. **Adobe.** Adobe. [En línea] [Citado el: 15 de Octubre de 2011.] www.adobe.es/products/acrobat/adobepdf.html.

12. **Dennis, Anita.** [En línea] 2008. [Citado el: 15 de Octubre de 2011.] http://www.creativepro.com/story/feature/8225.html.

13. **EcuRed.** EcuRed. [En línea] 2010. [Citado el: 28 de Noviembre de 2011.] http://www.ecured.cu/index.php/Libro_Electr%C3%B3nico#Formato_de_libro_electr.C3.B3nico_interactivo_EPUB.

14. **Briggs, Robin.** InDesign Magazine. [En línea] 2007. [Citado el: 6 de Noviembre de 2011.] http://www.indesignmag.com..

15. **Zibert, Carolina.** [En línea] 2006. [Citado el: 10 de Noviembre de 2011.] http://www.carolina.terna.net/ebooks/CrearEbooks.htm..

16. **Brandl, Georg.** [En línea] 2008. [Citado el: 15 de Octubre de 2011.] http://docs.python.org/documenting/index.html.

17. **Cockburn, Alistair.** *Agile Software Development.* 2006.

18. **Canós, José H, Letelier, Patricio y Penadés, Ma.** *Metodologías Ágiles en el Desarrollo de Software.* 2008.

19. **OpenUp, OpenUp.** [En línea] 2009. [Citado el: 18 de Octubre de 2011.] http://www.epf.eclipse.org/wikis/openup/.

20. **Project, Eclipse.** Página Oficial de Eclipse. [En línea] 2006. [Citado el: 19 de Octubre de 2011.] http://www.epf.eclipse.org/wikis/openupsp/openup_basic/deliveryprocesses/openup_basic_process_0uyGoMlgEdmt3adZL5Dmdw_desc.htm..

21. **Walsh, Norman.** [En línea] 2008. [Citado el: 20 de Noviembre de 2011.] http://www.xml.com..

22. **Oros Cabello, Juan Carlos.** *Diseño de páginas Web interactivas con Javascript y CSS. s.l.* s.l. : AlfaOmega, 2004.

23. **Rosalba.** [En línea] 2010. [Citado el: 20 de Octubre de 2011.] http://rosalba-rouse1.blogspot.com/..

24. **Galindo Haro, José María.** [En línea] 2007. [Citado el: 8 de Noviembre de 2011.] http://hdl.handle.net/10609/876..

25. **Sloccum, Jack.** Introduccion a ExtJs. [En línea] 2008. [Citado el: 10 de Noviembre de 2011.] http://www.google.com.cu/url?sa=t&rct=j&q=extjs%2Borigenes&source=web&cd=1&ved=0CB0QFjAA&url=http%3A%2F%2Fwww.ecured.cu%2Findex.php%3Ftitle%3DEspecial%3APdf.

26. **Doctrine, Doctrine.** [En línea] 2008. [Citado el: 18 de Noviembre de 2011.] http://www.doctrine-project.org/docs/orm/2.0/en/reference/introduction.html..

27. **PostgreSQL.** PostgreSQL. [En línea] 2012. [Citado el: 18 de Noviembre de 2011.] http://www.postgresql.org/..

28. **Netbeans, Netbeans.** [En línea] 2011. [Citado el: 20 de Noviembre de 2011.] http://netbeans.org/index_es.html.

29. **Rumbaugh, J, Jacobson, I y Booch, G.** *El Lenguaje Unificado de Modelado.* 2000.

30. **Pressman, Roger.** *Ingenieria de Software: Un enfoque práctico.* 2006.

31. **Paradigm, Visual.** Visual Paradigm. [En línea] 2011. [Citado el: 20 de Noviembre de 2011.] http://www.visual-paradigm.com..

32. **Laman, Craig.** [En línea] 2004. [Citado el: 20 de Enero de 2012.] http://es.scribd.com/doc/36837598/Applying-UML-and-Patterns-3rd-Ed-Craig-Larman-2004..

33. **Jacobson, Booch.** [En línea] 2004. [Citado el: 26 de Enero de 2012.]

34. **Patterns., Pattern-Oriented Software Architecture: A System of.** *Buschmann F, Meunier; R, Rohnert; .* s.l. : John Wiley & Sons, 1996.

35. Ivanek. [En línea] [Citado el: 10 de Noviembre de 2011.] http://ivanex.wikidot.com/patron-arquitectura.

36. **Burbeck, Steve.** *Application programming in Smalltalk-80 : How to use Model-View-Controller (MVC).* 1992. .

37. **Wiley, John.** *:Introduction to Client / Server Systems: A Practical Guide for Systems Professionals.*

38. **UCI.** E.V.A. [En línea] [Citado el: 15 de Febrero de 2012.] http://eva.uci.cu/ Coferencia#5 Modelo Cliente-Servidor. Teleinformática II.

39. **Gamma, Erich.** *Design Patterns: Elements of Reusable Object-Oriented Software.* 1995.

40. **Veloso Hernández, Pedro.** *Uso de patrones de arquitectura.* 1996. .

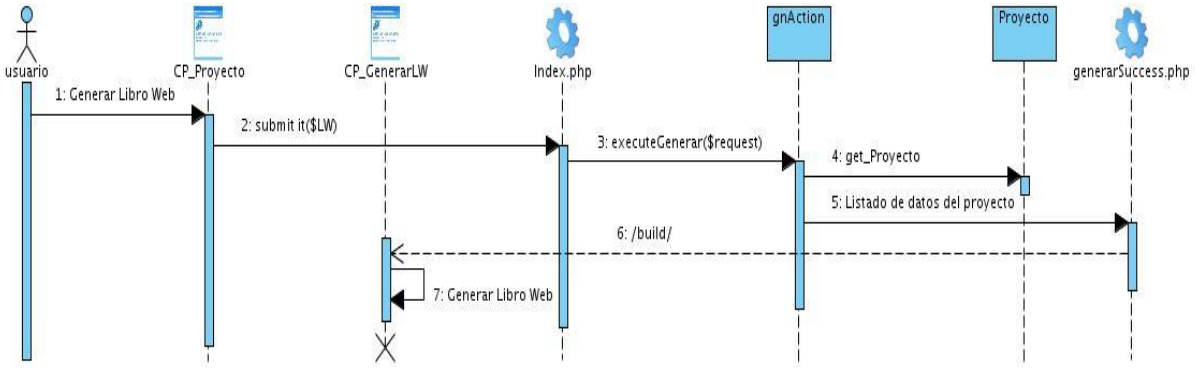
41. **Goguen, Joseph.** Oxford University Computer Laboratory : Programming Research Group. [En línea] 1992. The dry and the wet.

BIBLIOGRAFÍA

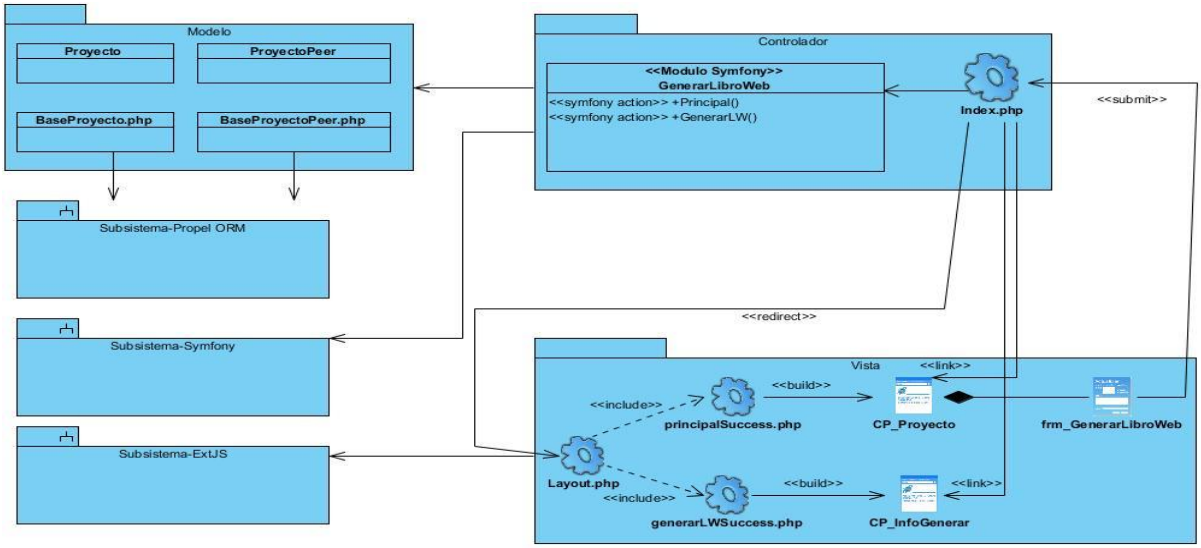
1. **Erosky, Erosky.** Erosky Consumer. [En línea] [Citado el: 27 de Octubre de 2011.] http://revista.consumer.es/web/es/20031101/internet/.
2. **Olivera Hernández, Rachel y Gómez Piñeiro, Karel Carmen.** Biblioteca UCI. [En línea] 2012. [Citado el: 26 de Octubre de 2011.] http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-detail.pl?biblionumber=8829.
3. **Pérez Salomón, Omar.** [En línea] 2008. [Citado el: 7 de Octubre de 2011.] http://www.ecured.cu/index.php/Historia_de_la_Informática_en_Cuba..
4. **Presman, Roger S.** *Ingeniería de Software. Un enfoque práctico.* 2005.
5. **Luján Mora, Sergio.** *Programación en Internet: Clientes Web.* s.l. : 1ª edición. Editorial Club Universitario., 2001.
6. **Solcre, Solcre.** [En línea] 2009. [Citado el: 8 de Octubre de 2011.] http://www.solcre.com/files/ventajas_de_las_aplicaciones_web.pdf..
7. **Conneally, Tim.** [En línea] 2008. [Citado el: 13 de Octubre de 2011.] http://www.betanews.com/article/Sonys-Reader-opens-support-for-more-publishers-formats/1216928081.
8. **Bottou, Léon, y otros.** Journal of Electronic Imaging. [En línea] 1998. [Citado el: 13 de Octubre de 2011.] http://leon.bottou.org/publications/pdf/jei-1998.pdf.
9. **IDPF.** [En línea] 11 de septiembre de 2007. Open Publication Structure (OPS) 2.0.
10. **Luján Mora, Sergio.** *Programación de aplicaciones web: historia, principios básicos y clientes web.* s.l. : 1ra Edición Editorial Club Universitario, 2002.
11. **Adobe.** Adobe. [En línea] [Citado el: 15 de Octubre de 2011.] www.adobe.es/products/acrobat/adobepdf.html.
12. **Dennis, Anita.** [En línea] 2008. [Citado el: 15 de Octubre de 2011.] http://www.creativepro.com/story/feature/8225.html.
13. **EcuRed.** EcuRed. [En línea] 2010. [Citado el: 28 de Noviembre de 2011.] http://www.ecured.cu/index.php/Libro_Electr%C3%B3nico#Formato_de_libro_electr.C3.B3nico_interactivo_EPUB.
14. **Briggs, Robin.** InDesign Magazine. [En línea] 2007. [Citado el: 6 de Noviembre de 2011.] http://www.indesignmag.com..
15. **Zibert, Carolina.** [En línea] 2006. [Citado el: 10 de Noviembre de 2011.] http://www.carolina.terna.net/ebooks/CrearEbooks.htm..
16. **Brandl, Georg.** [En línea] 2008. [Citado el: 15 de Octubre de 2011.] http://docs.python.org/documenting/index.html.
17. **Cockburn, Alistair.** *Agile Software Development.* 2006.
18. **Canós, José H, Letelier, Patricio y Penadés, Ma.** *Metodologías Ágiles en el Desarrollo de Software.* 2008.
19. **OpenUp, OpenUp.** [En línea] 2009. [Citado el: 18 de Octubre de 2011.] http://www.epf.eclipse.org/wikis/openup/.
20. **Project, Eclipse.** Página Oficial de Eclipse. [En línea] 2006. [Citado el: 19 de Octubre de 2011.] http://www.epf.eclipse.org/wikis/openupsp/openup_basic/deliveryprocesses/openup_basic_process_0uyGoMlgEdmt3adZL5Dmdw_desc.htm..
21. **Walsh, Norman.** [En línea] 2008. [Citado el: 20 de Noviembre de 2011.] http://www.xml.com..
22. **Oros Cabello, Juan Carlos.** *Diseño de páginas Web interactivas con Javascript y CSS. s.l.* s.l. : AlfaOmega, 2004.

23. **Rosalba.** [En línea] 2010. [Citado el: 20 de Octubre de 2011.] http://rosalba-rouse1.blogspot.com/..
24. **Galindo Haro, José María.** [En línea] 2007. [Citado el: 8 de Noviembre de 2011.] http://hdl.handle.net/10609/876..
25. **Sloccum, Jack.** Introduccion a ExtJs. [En línea] 2008. [Citado el: 10 de Noviembre de 2011.] http://www.google.com.cu/url?sa=t&rct=j&q=extjs%2Borigenes&source=web&cd=1&ved=0CB0QFjAA&url=http%3A%2F%2Fwww.ecured.cu%2Findex.php%3Ftitle%3DEspecial%3APdf.
26. **Doctrine, Doctrine.** [En línea] 2008. [Citado el: 18 de Noviembre de 2011.] http://www.doctrine-project.org/docs/orm/2.0/en/reference/introduction.html..
27. **PostgreSQL.** PostgreSQL. [En línea] 2012. [Citado el: 18 de Noviembre de 2011.] http://www.postgresql.org/..
28. **Netbeans, Netbeans.** [En línea] 2011. [Citado el: 20 de Noviembre de 2011.] http://netbeans.org/index_es.html.
29. **Rumbaugh, J, Jacobson, I y Booch, G.** *El Lenguaje Unificado de Modelado.* 2000.
30. **Pressman, Roger.** *Ingenieria de Software: Un enfoque práctico.* 2006.
31. **Paradigm, Visual.** Visual Paradigm. [En línea] 2011. [Citado el: 20 de Noviembre de 2011.] http://www.visual-paradigm.com..
32. **Laman, Craig.** [En línea] 2004. [Citado el: 20 de Enero de 2012.] http://es.scribd.com/doc/36837598/Applying-UML-and-Patterns-3rd-Ed-Craig-Larman-2004..
33. **Jacobson, Booch.** [En línea] 2004. [Citado el: 26 de Enero de 2012.]
34. **Patterns., Pattern-Oriented Software Architecture: A System of.** *Buschmann F, Meunier; R, Rohnert; .* s.l. : John Wiley & Sons, 1996.
35. Ivanek. [En línea] [Citado el: 10 de Noviembre de 2011.] http://ivanex.wikidot.com/patron-arquitectura.
36. **Burbeck, Steve.** *Application programming in Smalltalk-80 : How to use Model-View-Controller (MVC).* 1992. .
37. **Wiley, John.** *:Introduction to Client / Server Systems: A Practical Guide for Systems Professionals.*
38. **UCI.** E.V.A. [En línea] [Citado el: 15 de Febrero de 2012.] http://eva.uci.cu/ Coferencia#5 Modelo Cliente-Servidor. Teleinformática II.
39. **Gamma, Erich.** *Design Patterns: Elements of Reusable Object-Oriented Software.* 1995.
40. **Veloso Hernández, Pedro.** *Uso de patrones de arquitectura.* 1996. .
41. **Goguen, Joseph.** Oxford University Computer Laboratory : Programming Research Group. . [En línea] 1992. The dry and the wet.
42. **López Quesada, Juan Antonio**. Pruebas del software
43. **Fabien Potencier, François Zaninotto**. La guía definitiva de Symfony
44. **Fabien Potencier.** Tutorial de Jobbet y Chuletar
45. **Shea Frederick, Colin Ramsay, Steve Cutter Blades.** Learning Esencialts ExtJS
46. **Stoyan Stefanov.** Object Oriented JavaScript
47. EcuRed. http://www.ecured.cu/index.php/Modelo_de_dominio.
48. EcuRed. http://www.ecured.cu/index.php/Requisitos_de_Software

ANEXOS



Anexo 2. Diagrama de secuencia Generar Libro Web



Anexo 2. Diagrama Clases del diseño Generar Proyecto Libro Web

Requisitos	CU Gestionar Usuario	CU Gestionar Proyecto de Libro	CU Seleccionar Plantilla	CU Gestionar Contenido de Proyecto	CU Gestionar Recurso de Diseño	CU Generar Libro Web	CU Autenticar Usuario
RF1_Crear Nuevo Usuario	x						
RF2_Eliminar Usuario	x						
RF3_Actualizar Usuario	x						
RF4_Visualizar Trazas	x						
RF5_Reasignar Proyecto	x						
RF6_Buscar Usuario	x						
RF7_Crear Nuevo proyecto de Libro Web		x					
RF8_Eliminar proyecto de Libro Web		x					
RF9_Renombrar Proyecto de Libro Web		x					
RF10_Guardar Proyecto de Libro Web		x					
RF11_Abrir Proyecto de Libro Web		x					
RF12_Aplicar Plantilla			x				
RF13_Visualizar Plantilla			x				
RF14_Crear Nuevo Contenido(Capitulo, Epígrafe, Página)				x			
RF15_Eliminar Contenido(Capitulo, Epígrafe, Página)				x			
RF16_Actualizar Contenido(Capitulo, Epígrafe, Página)				x			
RF17_Buscar Contenido(Capitulo, Epígrafe, Página)				x			
RF18_Editar Contenido de Página				x			
RF19_Insertar Nuevo Recurso de Diseño (Imagen, Tabla,					x		
RF20_Eliminar Recurso de Diseño (Imagen, Tabla, Enlace)					x		
RF21_Actualizar Recurso de Diseño (Imagen, Tabla, Enlace)					x		
RF22_Buscar Recurso de Diseño (Imagen, Tabla, Enlace)					x		
RF23_Generar Libro Web						x	
RF24_Autenticar Usuario							x

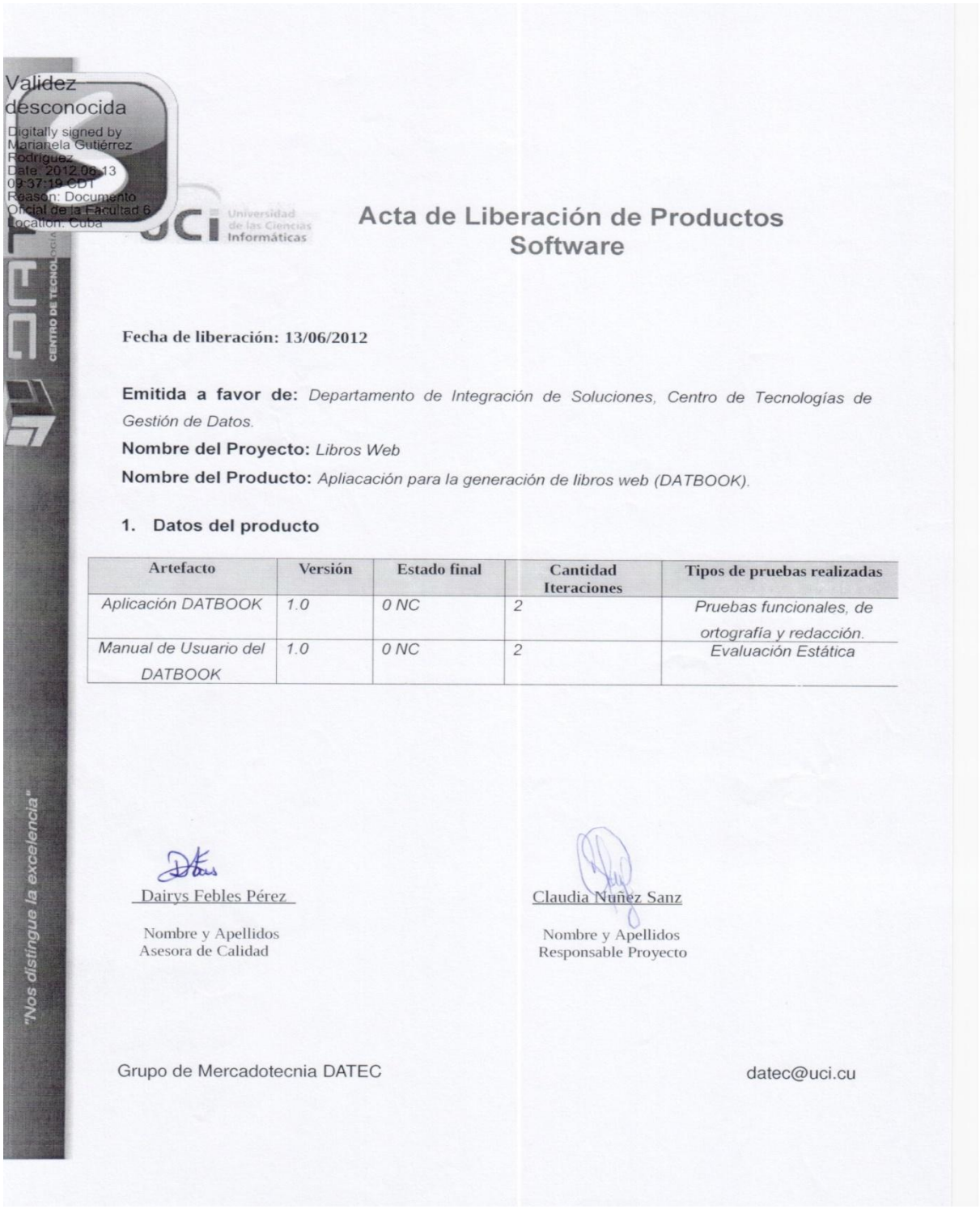
Anexo 3. Matriz de trazabilidad

```
1..3
# Comienza la prueba para la clase Contenido
ok 1 - Comparando nombre del contenido
ok 2 - Comprarando tipo contenido
ok 3 - Comparando title del contenido
# Looks like everything went fine.
root@lp06-308-23:/var/www/Datbook#
```

Anexo 6. Prueba Unitaria addContenido Capitulo

```
1..3
# Comienza la prueba para la clase Contenido
ok 1 - Comparando nombre del recurso
ok 2 - Comprarando tipo recurso
ok 3 - Comparando title del recurso
# Looks like everything went fine.
root@lp06-308-23:/var/www/Datbook#
```

Anexo 7. Prueba unitaria addRecurso



Validez desconocida
Digitally signed by Marianela Gutiérrez Rodríguez
Date: 2012.06.13 09:37:19 CDT
Reason: Documento Oficial de la Facultad 6
Location: Cuba

Universidad de las Ciencias Informáticas

CENTRO DE TECNOLOGÍAS

"Nos distingue la excelencia"

Acta de Liberación de Productos Software

Fecha de liberación: 13/06/2012

Emitida a favor de: *Departamento de Integración de Soluciones, Centro de Tecnologías de Gestión de Datos.*

Nombre del Proyecto: *Libros Web*

Nombre del Producto: *Apliacación para la generación de libros web (DATBOOK).*

1. Datos del producto

Artefacto	Versión	Estado final	Cantidad Iteraciones	Tipos de pruebas realizadas
Aplicación DATBOOK	*1.0*	*0 NC*	*2*	*Pruebas funcionales, de ortografía y redacción.*
Manual de Usuario del DATBOOK	*1.0*	*0 NC*	*2*	*Evaluación Estática*

Dairys Febles Pérez
Nombre y Apellidos
Asesora de Calidad

Claudia Nuñez Sanz
Nombre y Apellidos
Responsable Proyecto

Grupo de Mercadotecnia DATEC

datec@uci.cu

Anexo 8 1. Acta de liberación de la aplicación

GLOSARIO DE TÉRMINOS Y DEFINICIONES

DATEC: Centro de Tecnologías de Gestión de Datos.

Frameworks: Marco de trabajo.

OpenUp: Metodología ágil para el proceso del desarrollo del software.

UML: Unified Modeling Language (Lenguaje Unificado de Modelado).

AJAX: *Asynchronous JavaScript* + XML.

ExtJs: Extends JavaScript

XML: Extensible Markup Language (Lenguaje de Marcado Extensible).

CASE: Ingeniería de Software Asistida por Computación.

ORM: Object Relational Mapper, es una técnica de programación que permite generar clases a través de las tablas de la base de datos.

GOF: Gang of Four (Banda de los Cuatro), patrón de diseño.

GRASP: General Responsibility Assignment Software Patterns (Patrones de Software para Asignar Responsabilidades).

SOA: Arquitecturas Orientas a Servicios.

MVC: Modelo – Vista – Controlador, patrón arquitectónico.

GPL: Licencia Open Source (Licencia de Código Abierto).

LGPL: Licencia Pública General Reducida de GNU.

BSD: Berkley Software Distribution, Distribución de Software de Berkley.

Python-Spinx: Herramienta para la generación de documentación de python.

IDE: Entorno de desarrollo integrado**.**