Universidad de las Ciencias Informáticas

Facultad 15

“Desarrollo del sistema generador de la ayuda para el IDE Caxtor”

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

Autor: Leosdeny Megret Tito

Tutores: Ing. Dioleisys Fontela González
Ing. Michael Hernández Martínez

Ciudad de La Habana, 21 de junio de 2010
Año 52 de la Revolución.

"Si el presente es de lucha, el futuro es nuestro." **Che**

Declaración de Autoría

Declaro ser el único autor del presente trabajo y autorizo 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 junio del año 2010**.**

Leosdeny Megret Tito

_____

Firma del Autor

Ing. Michael Hernández Martínez

_____

Firma del Tutor

Ing. Dioleisys Fontela González

_____

Firma del Tutor

Datos de Contacto

Tutor Ing. Dioleisys Fontela González
Email: dfontela@uci.cu
Años de graduado: 0
Años de experiencia en el tema: 1

Tutor Ing. Michael Hernández Martínez
Email: mhmartinez@uci.cu
Años de graduado: 0
Años de experiencia en el tema: 1

Agradecimientos

A la Revolución, a Fidel y a Raúl por darme la posibilidad de alcanzar mis sueños.

A mis amigos por estar siempre en los buenos y malos momentos.

A mis tutores, en especial a Dioleisys por su paciencia, su apoyo y dedicación.

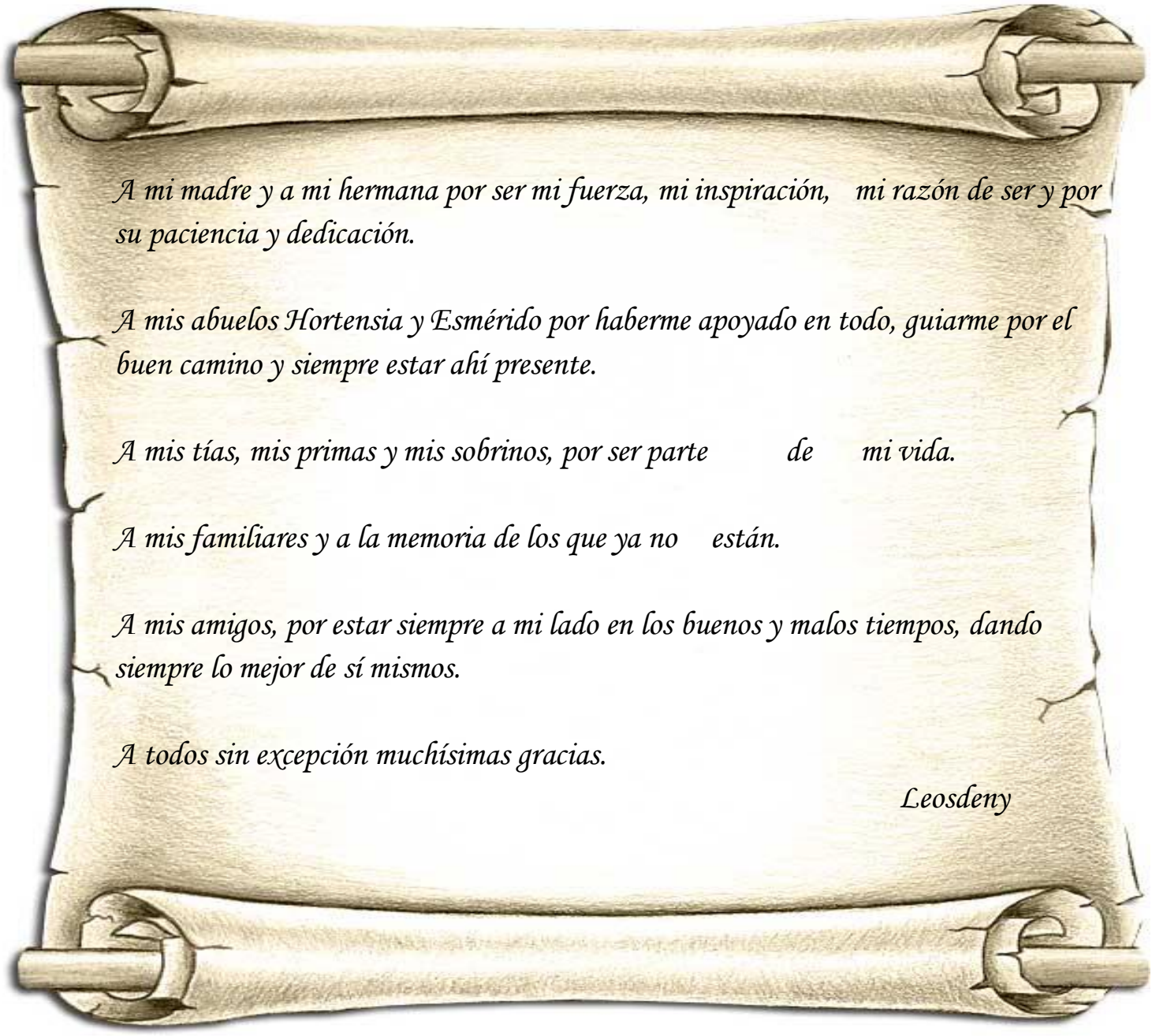
A mis compañeros de proyecto, que nos han brindado su ayuda cuando la hemos necesitado.

A todos los que han contribuido de una forma u otra para el desarrollo de este trabajo.

A todas las personas que me han apoyado y se han preocupado durante el desarrollo de mi carrera.

Siéntanse parte de este logro.

Dedicatoria



A mi madre y a mi hermana por ser mi fuerza, mi inspiración, mi razón de ser y por su paciencia y dedicación.

A mis abuelos Hortensia y Esmérido por haberme apoyado en todo, guiarme por el buen camino y siempre estar ahí presente.

A mis tías, mis primas y mis sobrinos, por ser parte de mi vida.

A mis familiares y a la memoria de los que ya no están.

A mis amigos, por estar siempre a mi lado en los buenos y malos tiempos, dando siempre lo mejor de sí mismos.

A todos sin excepción muchísimas gracias.

Leosdeny

Resumen

Para el diseño de interfaces que utilizan el framework ExtJS se está desarrollando el IDE Caxtor, con el fin de agilizar y estandarizar el proceso. A pesar de la fortaleza de sus funcionalidades este no cuenta con una guía de apoyo para los usuarios que interactúan con él. El presente trabajo que lleva por título "Desarrollo del sistema generador de la ayuda para el IDE Caxtor" pretende desarrollar una aplicación Web que permita construir y generar la ayuda del IDE; es decir, brindar la información referente al funcionamiento de cada una de las partes que conforman el sistema para facilitar y mejorar el trabajo de los usuarios con el mismo. Este documento recoge el resultado de todo el trabajo realizado para la elaboración del sistema generador de ayuda, incluyendo el estudio del estado del arte de aplicaciones con similares objetivos existentes en el mundo, el estudio y definición de las características del sistema, el diseño y la implementación del producto. El sistema propuesto estará disponible vía web, para su desarrollo se utilizó el framework ExtJS para la capa de presentación y PHP para llevar a cabo las funcionalidades del lado del servidor, próximamente se integrará a los mecanismos de seguridad que se implementarán para Caxtor. Se utilizaron en su gran mayoría herramientas de software libre teniendo en cuenta las políticas de la universidad y el país para la soberanía tecnológica.

Palabras claves: IDE Caxtor, Sistema, Generador, Ayuda, Construir, Aplicación Web.

Índice

Índice de Ilustraciones

Índice de tablas

Introducción.

A nivel mundial existe un gran avance en la informática y las comunicaciones y se desarrollan productos para todos los sectores de la economía. Cuba no ha quedado exenta de estos avances tecnológicos, durante los últimos 20 años diversas instituciones cubanas han desarrollado sistemas encaminados a lograr niveles de informatización para distintos sectores. Con la implementación en el 2003, de los programas de la revolución, que incluyen como prioridad la informatización de los servicios, se inicia en Cuba un avance vertiginoso hacia la informatización de la sociedad, orientado en primer lugar a la superación y desarrollo profesional, que a su vez se ha extendido a la automatización de los servicios médicos, la investigación, la información científico-técnica y el apoyo en la toma de decisiones.

Hoy en día la alternativa para el desarrollo de aplicaciones es la web con arquitectura cliente/servidor, ya que esta permite tener una aplicación desplegada en un servidor y sin más requerimientos que un navegador web y una conexión a la red se puede acceder a la misma, facilitando así el trabajo a distancia. Por las ventajas que este tipo de aplicaciones brinda, en la Universidad de las Ciencias Informáticas (UCI) se han desarrollado diversas aplicaciones que manejan esta alternativa, utilizando para el diseño de las interfaces y los componentes visuales del lado del cliente el framework ExtJS, ya que éste es uno de los más avanzados para el desarrollo rápido de aplicaciones enriquecidas de internet, totalmente novedoso y con una arquitectura flexible. Sin embargo, todo el proceso de creación de interfaces se hace muy engorroso debido a que no existe una herramienta para el diseño gráfico que proporcione un estándar para el entendimiento de todos los involucrados en el desarrollo. En base a la necesidad de crear un nuevo mecanismo que sea estándar en su utilización, se le ha dado la tarea a desarrolladores de la Unidad de Compatibilización Integración y Desarrollo (UCID) y el Centro de Gestión de Tecnologías y Datos (DATEC), desarrollar un Entorno de Desarrollo Integrado (IDE en inglés) que facilite el trabajo de diseño de interfaces para aplicaciones web y que permita agilizar y estandarizar este proceso.

El desarrollo actual de aplicaciones profesionales requiere que sean acompañadas por sistemas de ayuda que brinden facilidad y claridad de uso por parte del usuario para lograr un mayor entendimiento y manipulación de las mismas. Actualmente, no se cuenta con un sistema web que haciendo uso del framework ExtJS, permita construir y generar la ayuda de Caxtor, porque a pesar de ser una herramienta nueva, y con muy poco tiempo de desarrollo, ya se han liberado versiones estables y actualmente está siendo utilizado; pero éste no posee una guía de apoyo para que los usuarios interactúen con él, lo que trae como consecuencia una mala manipulación del sistema.

Luego de analizar lo anteriormente planteado y dada la situación actual, se plantea el siguiente **problema a resolver:**

No existencia de una herramienta web que permita construir y generar el sistema de ayuda para el IDE Caxtor.

Este problema se enmarca en el **objeto de estudio**: Los sistemas generadores de ayuda y el **campo de acción** lo constituye los sistemas generadores de ayuda para la Web.

El **objetivo general** de la investigación: Desarrollar una aplicación que permita construir y generar el sistema de ayuda para el IDE Caxtor.

Como **Idea a defender** se ha establecido: Con la utilización de un sistema informático que posibilite la construcción del sistema de ayuda y soporte para el IDE Caxtor, se logrará tener una mayor eficiencia, flexibilidad, organización y control a la hora de utilizar el mismo.

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

- Realizar un estudio sobre las diferentes herramientas que permiten la creación de sistemas de ayuda para la web.
- Diseñar la aplicación.
- Implementar la aplicación.
- Evaluar los resultados mediante pruebas exploratorias.

Para poder alcanzar los objetivos específicos trazados se desarrollarán las **tareas** siguientes:

- Análisis de las herramientas existentes que permitan la creación de sistemas de ayuda para la web.
- Diseño de la aplicación.
- Obtención del modelo de datos.
- Obtención del diagrama de componentes.
- Implementación de la aplicación.
- Validación del sistema a nivel de desarrollador.

Para darle cumplimiento a las tareas se guiará la investigación dentro del marco de los **métodos científicos:** "**Teóricos y Empíricos**". Dentro de los **métodos teóricos** se utilizará el método: "**Analítico-Sintético**", con éste se podrá analizar cada uno de los elementos de manera independiente, posibilitando una mayor capacidad de comprensión y de síntesis sobre los aspectos más importantes y para establecer

relaciones entre ellos. Por su parte la utilización del método "**Análisis Histórico-Lógico**" brindará la posibilidad de analizar toda la evolución del problema que se estará estudiando, así como analizar la trayectoria de las tecnologías y herramientas utilizadas en el proceso de generación y construcción de sistemas de ayuda en la web. También se utilizará el método "**Inductivo-Deductivo**" para a través de un razonamiento llegar a un grupo de conocimientos particulares y generales.
Uno de los **métodos empíricos** que será referenciado es la "**Observación**" este método permite adquirir información necesaria y puede utilizarse en cualquiera de las fases de la investigación, además ofrece un gran acercamiento a la realidad y permite ver la posible solución del problema desde diferentes ángulos.

Estructura del documento

El presente documento está estructurado de la siguiente manera: Resumen, introducción, 4 capítulos de contenido, conclusiones, recomendaciones, referencias bibliográficas, bibliografía y los anexos, además de un glosario de términos.

En el **Capítulo 1: Fundamentación Teórica**, Se abordan las tendencias, técnicas, tecnologías, metodologías y software usados en el mundo, para el desarrollo de las aplicaciones web, así como los sistemas de ayuda y soporte en aplicaciones web.

En el **Capítulo 2: Características de Sistema:** Se definen los objetivos estratégicos y se hace una descripción general de la propuesta de sistema y de cómo debe funcionar el negocio a través de procesos.

En el **Capítulo 3: Diseño del Sistema:** Se presentan los diagramas de clases correspondientes al diseño, y la realización de los Casos de Uso a través de los diagramas de interacción. Este capítulo está enfocado hacia como construir técnicamente el software.

En el **Capítulo 4: Implementación del Sistema:** Se describen todos los elementos relacionados con la implementación y prueba de la aplicación, algunos de los artefactos encontrados son los diagramas de componentes, despliegue y las características generales de la implementación.

Capítulo 1: Fundamentación Teórica

Introducción

Para comprender correctamente en qué consiste la presente investigación, se hace necesario dominar los aspectos más significativos relacionados con el objeto de estudio de la misma: Los sistemas generadores de ayuda. Con el fin de hacer posible lo anterior ha sido redactado este capítulo, en el cual se exponen los conceptos y elementos que constituyen la base teórica para la realización del presente trabajo de diploma y lógicamente para dar solución al problema a resolver planteado. Conocer el significado de términos clave en el tema que se analiza permitirá tener una noción acertada de todas las ideas que se plasman en el cuerpo del documento. Además, en este capítulo queda expuesto el marco teórico de la investigación que ha surgido como resultado de anteriores investigaciones y que a su vez permite orientar el trabajo de una manera coherente. Se analizan también otras aplicaciones existentes que centran sus funcionalidades en la gestión y construcción de sistemas de ayuda.

1.1 Los sistemas de ayuda en línea

Existen en el mundo disímiles sistemas de ayuda en línea que brindan información sobre diversas aplicaciones. Todos tienen como objetivo principal ofrecer la información necesaria para que los usuarios web estén en capacidad de utilizar en forma eficiente las potencialidades que le ofrece dicha aplicación para facilitar sus labores diarias. Dentro de estos sistemas podemos mencionar algunos:

- ✓ **Ayuda en línea de SAP:** SAP es un ERP desarrollado en España que tiene como objetivo acelerar la innovación en las empresas e industrias de todo el mundo y contribuir al desarrollo económico a gran escala. Este posee un Portal de Ayuda que proporciona la documentación basada en la web para todas las soluciones de SAP y permite buscar en la biblioteca en línea la información que desee cuando se necesite.
- ✓ **Ayuda en línea de Microsoft MSDN Library**: MSDN es un recurso imprescindible para los programadores que trabajan con herramientas, productos y tecnologías de Microsoft. Contiene una gran cantidad de información técnica de programación, incluidos código de ejemplo, documentación, artículos técnicos y guías de referencia. Proporciona información que va desde los conceptos básicos hasta instrucciones detalladas acerca de cómo crear aplicaciones Web. Está dirigida a programadores, arquitectos de sistemas, creadores de informes y especialistas en

implementaciones informáticas.

En Cuba también se han desarrollado sistemas de ayuda en línea para satisfacer las necesidades de los usuarios. Entre ellos se pueden destacar sitios como el de la Red Nacional de Salud (Infomed) y la Empresa de Telecomunicaciones de Cuba (ETECSA).

- ✓ **Ayuda en línea de Infomed:** es el Portal de Salud Cubano y la red de personas e instituciones que comparten el propósito de facilitar el acceso a la información de salud en el país. Para facilitar la navegación por el sitio se incluyó una ayuda al usuario, la cual permite conocer la información que podrá consultar en la Biblioteca Virtual de Infomed, así como las demás funcionalidades de navegación por la web, cómo realizar una búsqueda, dónde informarse sobre los principales eventos que se desarrollan en el sector de la salud en el país, cómo se distribuye la información en el sitio entre otros temas de interés.
- ✓ **Ayuda en línea de ETECSA:** es una organización de capital mixto y tiene como objeto social prestar los servicios públicos de telecomunicaciones, mediante la operación, instalación, explotación, comercialización y mantenimiento de redes públicas de telecomunicaciones en todo el territorio de la república de Cuba. En el sitio web de ETECSA se incluye una ayuda donde se presenta lo referente a los servicios que ofrece (servicios de acceso a internet, de telefonía virtual, servicio telefónico básico, nacional e internacional entre otros). Además, posee un buscador para facilitar la búsqueda a los usuarios y un mapa del sitio donde se visualiza toda la información que contiene la ayuda.

1.2 Sistemas informáticos existentes vinculados al objeto de estudio

En la actualidad existen múltiples soluciones informáticas para la generación de sistemas de ayuda, los cuales se mencionan a continuación:

HelpMaker

Es una aplicación para crear archivos de ayuda que acompañen a los programas. Permite la creación de archivos de ayuda enteros; en diferentes formatos, tales como: WinHelp, RTF (texto enriquecido) y HTML-Help. El programa es completísimo y rápido, en nada está creada la estructura de la ayuda, pudiendo cambiarla, ampliarla, editarla, añadir vínculos (incluso de una zona a otra de la misma ayuda), etc. **(Softonic 2000)**

HelpNDoc

Es una más que interesante utilidad gratuita para crear todo tipo de ayudas. Se trata de un editor visual muy fácil de utilizar, y que resulta ideal para crear ayudas ya sea para incluirla en las aplicaciones, como para insertarla en una página web. Crea una estructura en árbol con la que es muy fácil mantener una ordenación clara y práctica. Además, es realmente fácil crear cada categoría, incluso podrás asignarles palabras clave para que aparezcan cuando el usuario realice una búsqueda. **(Softonic 2000)**

HelpSmith

Es una herramienta visual de creación de sistemas de ayuda para la creación fácil de archivos de ayuda HTML, Ayuda Web y Documentación de Manuales Impresos usando el mismo documento de origen. El entorno independiente y muy fácil de usar tiene un potente editor de texto unicode al estilo de MS Word para la edición de temas de ayuda con estilos dinámicos y corrector ortográfico en tiempo real, un enfoque avanzado para trabajar con archivos gráficos y de video, y muchas otras características únicas. HelpSmith, la herramienta de creación de sistemas de ayuda de Divcom Software, le brinda una forma fácil de crear documentación técnica de principio a fin y también le permite importar temas de ayuda de Microsoft Word (.RTF), .HTML, y archivos de texto puro que pueden escribirse en múltiples codificaciones de caracteres ANSI. Puede usar HelpSmith para brindar archivos de ayuda para su software en formatos populares de ayuda incluyendo archivos de Ayuda CHM HTML dependientes del contexto, un formato de ayuda estándar para la ayuda de programas en Windows. HelpSmith tiene soporte completo para todas las características de la ayuda CHM, de modo que puede construir una tabla de contenidos en estructura arborescente, índice de palabras clave, también puede editar y crear pantallas de ayuda personalizadas en ayuda HTML. **(Softonic 2000)**

RoboHelp

Es una poderosa herramienta para crear sistemas de ayuda y documentación profesionales para aplicaciones de escritorio y basadas en la web, tales como aplicaciones .NET y aplicaciones dinámicas de Internet. Podrá crear sistemas de ayuda que incluyan elementos tales como temas de ayuda, tablas de materias, índices, glosarios y ayuda contextual entre otras características. Asimismo, podrá generar sistemas de ayuda en cualquier formato popular de ayuda en línea, además de documentación impresa. RoboHelp posee un entorno de autoría flexible.

HTML Help Workshop

Permite crear ficheros de ayuda de Windows (HLP) y páginas web que utilicen controles de navegación. HTML Help Workshop es un programa para crear estos ficheros y distribuirlos con las aplicaciones. Incluye un administrador de proyectos, un compilador de ayuda y un editor de imágenes. Este programa ofrece algunas ventajas sobre el estándar HTML, incluyendo la habilidad de implementar una tabla de elementos combinada y un índice, así como el uso de palabras clave para capacidades avanzadas de hiperenlazado.

A pesar de las funcionalidades que brindan los sistemas anteriormente descritos, son aplicaciones de escritorio, privativas y solo funcionan en sistema operativo Windows. El fin principal del presente sistema difiere un tanto de los existentes al ser implementado como una aplicación web con soporte para múltiples usuarios, sin necesidad de pagar licencia alguna e independientemente del sistema operativo el sistema debe brindar los mismos servicios, sin lugar a dudas estos elementos constituyen las ventajas del nuevo sistema a desarrollar con respecto a los sistemas existentes.

El sistema a desarrollar permitirá gestionar toda la información referente al IDE Caxtor y contará con la flexibilidad adecuada para ser aplicable a otras aplicaciones.

1.3 Tendencias y tecnologías actuales

El entorno donde se utilizará la presente propuesta de solución es un factor importante a tener en cuenta a la hora de seleccionar las mejores opciones para su desarrollo y puesta en funcionamiento. A continuación se muestra el estudio realizado para seleccionar las tecnologías, metodología y herramientas que serán utilizadas.

1.3.1 Aplicaciones Web

El uso de los sistemas desktop o de escritorio trajo consigo que la información y la lógica se encontraran esparcidas en cada ordenador que las utilizara. Se originara duplicidad de datos si no existía unificación. Actualizar o corregir un sistema resulta realmente complejo pues la modificación implicaba a cada ordenador. Estas situaciones propiciaron una nueva forma de concebir los sistemas.

En 1989 se dan los primeros pasos en la Web, lo cual se logra trabajando en la creación de una pequeña plataforma para la lectura de documentos científicos en el Centro Europeo de Investigación Nuclear

(CERN). Los principios tecnológicos de esta plataforma incluyeron la creación de tres componentes: HTML (HyperText Markup Language), HTTP (HyperText Transfer Protocol) y URI (Universal Resource Indentifier). El HTML era un lenguaje para escribir los documentos, y con el que se podían establecer los enlaces a otros documentos. HTTP, en cambio, era un protocolo para la transmisión de los documentos en formato HTML entre el servidor y el cliente. Y los URI eran los identificadores de los documentos que servían para establecer los enlaces.

Las aplicaciones Web tienen como arquitectura general la de un sistema Cliente - Servidor, donde tanto el cliente (el navegador) como el servidor (el servidor Web), y el protocolo mediante el cual se comunican son estándar, y no han de ser creados por el desarrollador. **(Arsys 2010)**

La parte del cliente de las aplicaciones Web está formada por el código HTML que forma la página Web, con opción a código ejecutable mediante los lenguajes script de los navegadores (JavaScript, VBScript, PerlScript) o mediante pequeños programas (applets) en Java. **(Méndez 2007)**

La parte del servidor está formada por un programa o script que es ejecutado por el servidor Web, y cuya salida se envía al navegador del cliente. El auge de las aplicaciones Web se ve favorecido por las innegables ventajas que brindan, pueden citarse algunas como:

- **Multiplataforma:** Las aplicaciones pueden ser utilizadas a través de variadas plataformas, tanto de hardware como de software apoyado en gran medida por el uso de estándares.
- **Actualización instantánea:** Dada la razón que los usuarios de la aplicación hacen uso de un sólo programa que interactúa directamente con el servidor, siempre utilizarán la versión más actualizada del sistema.
- **Acceso no fijo:** El usuario puede acceder a la aplicación usando cualquier equipo provisto de red y navegador, estando ubicado en cualquier lugar y suponiendo siempre que cuenta con los permisos de acceso necesarios.
- **Fácil integración:** Dado su basamento en protocolos estándares, la información manejada por el sistema puede ser accedida con mayor facilidad por otros sistemas.

Por lo antes expuesto, se considera que el sistema generador de ayuda tendrá un mejor funcionamiento y brindará servicios optimizados al ser implementado como una aplicación Web.

1.3.2 ExtJS

Alrededor del año 2006, Jack Slocum comenzó a desarrollar una serie de utilidades extendidas para la librería Yahoo User Interface (YUI). Rápidamente ganaron en popularidad dentro de la comunidad YUI que organizaron una librería independiente llamada YUI-Ext. En poco tiempo el nombre de la librería fue cambiada a Ext, ya que comenzó a extender otras librerías independientemente a YUI. Fue así que en abril de 2007 se libera ExtJS 1.0. Actualmente es la librería más madura en cuanto a desarrollo de interfaces de usuario en JavaScript se refiere. **(Zammetti 2009)**

El ExtJS es un conjunto de código que nos proporciona no solo estándares de JavaScript como la tecnología Ajax, utilidades del DOM y animaciones, sino también nos proporciona una amplia gama de componentes con diversas funcionalidades **(Alameda 2007)**, además de que cuenta con una excelente documentación.

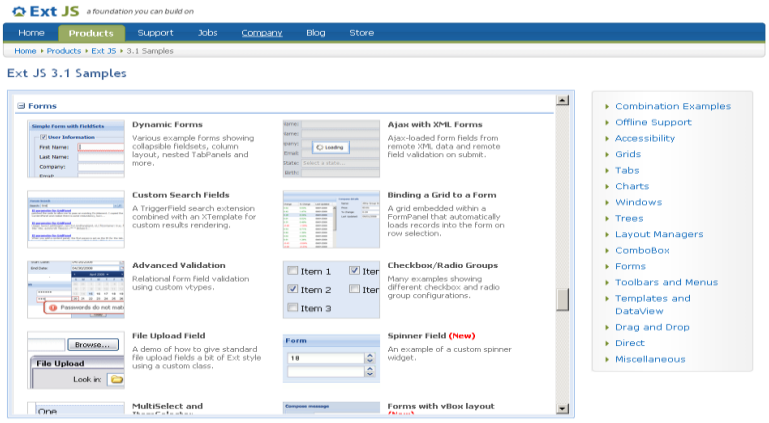


Figura 1 Framework ExtJS

ExtJS es un potente framework para el desarrollo con AJAX. Incluye las diferentes bondades:

- Modelo de Componentes.
- Modelo de Contenedores.
- Capas.
- Grid.
- Vistas de Datos.

Se puede afirmar que el uso del framework ExtJS en el desarrollo del sistema generador de ayuda,

permitirá que se logre simular una aplicación de escritorio que combine la interactividad necesitada, con los efectos visuales adicionales.

1.4 Lenguajes de programación del lado del cliente

Un lenguaje del lado cliente es totalmente independiente del servidor, y en el caso de una aplicación web esto permite que la página pueda ser albergada en cualquier sitio. El código, tanto del hipertexto como de los scripts, es accesible a cualquiera y esto puede afectar la seguridad. A continuación se describen algunos de estos lenguajes.

1.4.1 HTML

"*Es un lenguaje sencillo, permite definir documentos de hipertexto a base de ciertas etiquetas que marcan partes del documento dándoles una estructura o jerarquía. Presenta el texto de una manera estructurada y agradable, con enlaces (hyperlinks) que conducen a otros documentos o fuentes de información relacionadas, y con inserciones multimedia (gráficos, sonido, video). El lugar donde se encuentra esta información puede ser el mismo documento o cualquier otro lugar de Internet*." **(XPPS 2009)**

En un documento HTML se pueden ver tres partes bien diferenciadas:

- Una cabecera del tipo de documento, en ella se especifica el tipo de documento HTML.
- Una cabecera de documento, donde se especifica información del documento.
- El cuerpo del documento, donde se coloca toda la información que contiene la página Web.

1.4.2 CSS

El impulso de los lenguajes de hojas de estilos se produce a partir del auge y el crecimiento del uso de Internet y el incremento del lenguaje HTML para la creación de documentos electrónicos. La competencia entre los navegadores y la falta de un estándar para la definición de los estilos afectaban la creación de documentos con la misma apariencia.

En 1995, el W3C decidió apostar por el desarrollo y estandarización de CSS (del inglés, Cascading Style Sheets) y lo añadió a su grupo de trabajo de HTML. A finales de 1996, el W3C publicó la primera recomendación oficial, conocida como "CSS nivel 1".

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos

y su presentación y es imprescindible para crear páginas web complejas. **(Eguíluz 2008)**
El lenguaje CSS se utiliza para definir el aspecto o propiedades de cada elemento del contenido en una página web, tales como el color, tamaño, tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, etc.

1.4.3 JavaScript

Los lenguajes Script han surgido para extender las capacidades del Lenguaje HTML. JavaScript es un lenguaje interpretado que nos permite interactuar con el navegador de manera dinámica y eficaz, proporcionando a las páginas web dinamismo y vida **(WebEstilo 2009)** .Fue creado por Brendan Eich en la empresa Netscape Communications. El navegador tiene la responsabilidad de interpretar las sentencias. Se considera un lenguaje orientado a objetos y a eventos del usuario. Con su uso se pueden desarrollar scripts que den respuesta a dichos eventos.

Al ejecutarse en el lado cliente, evita la sobrecarga del servidor por peticiones que pueden tener respuestas locales. El sistema generador de ayuda se verá beneficiado por la característica mencionada anteriormente, siendo de vital importancia para su correcto funcionamiento. No se debe olvidar la vinculación existente entre el lenguaje JavaScript y el conjunto de tecnologías AJAX por lo que prescindir de su utilización en la realización de este trabajo no es una opción.

1.5 Lenguajes de programación del lado del servidor

La Programación del lado del servidor es una tecnología que consiste en el procesamiento de una petición de un usuario mediante el reconocimiento, ejecución e interpretación de un script en el servidor para generar páginas HTML dinámicamente como respuesta, las cuales son comprensibles para el cliente. Un lenguaje que se ejecuta en el lado del servidor es independiente del cliente por lo que es mucho menos rígido respecto al cambio de un navegador a otro o respecto a las versiones del mismo.

1.5.1 PHP

PHP es un robusto lenguaje de programación del lado del servidor. Fue diseñado por Rasmus Lerdorf en 1994 como un CGI escrito en C que permitía la interpretación de un número limitado de comandos. Dada la aceptación del primer PHP y de manera adicional, su creador diseñó un sistema para procesar formularios al que le atribuyó el nombre de FI (Form Interpreter) y el conjunto de estas dos herramientas,

sería la primera versión compacta del lenguaje: PHP/FI. **(Álvarez 2009)**

Dado que es un lenguaje abierto dando la posibilidad de modificar el código fuente y añadir nuevas funcionalidades ha tenido una rápida evolución y desarrollo. Actualmente se encuentra en su versión PHP5.3. PHP también ofrece la integración con varias bibliotecas externas, que dan al desarrollador la posibilidad de realizar cualquier tarea, desde generar documentos en pdf (Portable Document Format) hasta analizar código XML **(Van Der Henst 2009)**. Resulta indiscutible que PHP es la solución por excelencia a la implementación del sistema generador de ayuda. El hecho de ser libre y poder utilizarlo sin necesidad de disponer de una licencia aumenta sus ventajas. PHP posee soporte para la gran mayoría de las plataformas existentes en la actualidad. Es eficiente para el desarrollo de aplicaciones Web y logra integrarse con un gran número de servidores, incluso propietarios. Su sintaxis clara y bien definida es lo que permite que sea sencillo de aprender y utilizar.

1.6 Servidor Web Apache

El servidor Apache es un servidor HTTP de código abierto que se desarrolló dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation. Apache presenta entre otras características, mensajes de errores altamente configurables, bases de datos de autenticación y negociado de contenido, implementa el protocolo HTTP/1.1 y la noción de sitio virtual, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración. Apache tiene amplia aceptación en la red desde 1996, es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en el 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en su cuota de mercado en los últimos años. (Estadísticas históricas y de uso diario proporcionadas por Netcraft).

La mayoría de las vulnerabilidades de la seguridad descubiertas y resueltas tan sólo pueden ser aprovechadas por usuarios locales y no remotamente. **(Apache 2009)**

Ventajas:

- Modular
- Código abierto
- Multi-plataforma
- Extensible
- Popular(fácil de conseguir ayuda/soporte)
- Gratuito

1.7 SQLite

SQLite es una pequeña librería programada en lenguaje C que implementa un completo motor de base de datos multiplataforma que no precisa configuración. Se distribuye bajo licencia de dominio público. Es muy rápido y la ventaja fundamental es que permite utilizar un amplio subconjunto del lenguaje estándar SQL. SQLite destaca también por su versatilidad. El motor de PHP 5 incluye soporte interno para SQLite.

Combina el motor y la interfaz de la base de datos en una única biblioteca, y almacena los datos en un único archivo de texto plano. Esto hace que cada usuario pueda crear tantas bases de datos como desee sin la necesidad de la intervención de un administrador de bases de datos que gestione los espacios de trabajo, usuarios y permisos de acceso. El hecho de almacenar toda la base de datos en un único archivo, facilita la portabilidad de los datos, y solamente tiene la restricción del espacio de disco asignado al usuario en el servidor.

Su potencia se basa fundamentalmente en la simplicidad, lo que hace que no sea una buena solución en entornos de tráfico muy elevado y/o alto acceso concurrente a datos. SQLite encapsula toda la base de datos en un único fichero. **(ITE 2010)**

En su versión 3, SQLite soporta bases de datos de hasta 2 Terabytes de tamaño, y también permite la inclusión de campos tipo Blob.

Formas de uso:

- Como gestor de base de datos local en una PC. De esta forma podemos gestionar bases de datos con SQLite igual que si estuviéramos trabajando con un sistema gestor de base de datos como MySQL sin necesidad de instalar nada, ya que SQLite se compone de un único archivo ejecutable.
- Como una extensión más de PHP, utilizando las funcionalidades de SQLite configuradas, o bien como módulo de PHP, o como librería; sin necesidad de tener instalado o conectar con un servidor de base de datos. Ofrece una rápida interfaz de base de datos almacenada en archivo de texto plano. **(ITE 2010)**

SQLite como extensión de PHP.

Una de las opciones de utilización de SQLite es como extensión de base de datos para PHP.

Esta opción, ofrece una rápida interfaz de base de datos, al igual que ofrecen otras bases de datos como MySQL, pero con la ventaja de no tener la necesidad de tener instalado o conectar con un servidor de base de datos. SQLite tiene prácticamente las mismas funcionalidades y rapidez que el resto de gestores

de base de datos, y los datos se almacenan en un archivo de texto plano.
SQLite dispone de una completa interfaz orientada a objetos, con distintas funciones que nos facilitan la manipulación de datos. Funciones similares a las que podemos manejar con MySQL. **(ITE 2010)**

1.8 Metodología y herramientas

A continuación, se describen las principales características de la propuesta de metodología y herramientas para llevar a cabo el desarrollo de la presente investigación.

1.8.1 Metodología de desarrollo de software

El proceso de desarrollo de software no es una tarea sencilla. Muchos proyectos de los que se desarrollan abaten en problemas pues se salen del presupuesto, tienen importantes retrasos, o simplemente no cumplen con las expectativas del cliente. Como solución a esta problemática surge la Metodología, la cual se puede definir como un conjunto de actividades encaminadas a lograr un fin específico. Las metodologías imponen un trabajo disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente.
Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de los productos software. Es como un libro de recetas de cocina, en el que se van indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. Además, detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla. **(IAGP 2006)**

En la actualidad existen un gran número de propuestas que definen este marco de trabajo, se habla de metodologías tradicionales o robustas como Rational Unified Process (RUP), Microsoft Solutions Framework (MSF) y Métrica 3.0 o metodologías ágiles como Extreme Programming (XP), Scrum, Cristal Methods y Feature Driven Development, pero las experiencias indican que no existe una receta mágica y no se trata simplemente de seleccionar una de las propuestas y seguirla. Se hace necesario entender los procesos de desarrollo, tener pleno dominio de factores como el tiempo con el que se cuenta, presupuesto y sobre todo cual metodología se ajusta más al entorno y a lo que se desea desarrollar.

A partir de que el desarrollo del IDE Caxtor se lleva a cabo en el centro DATEC, en el mismo se emplea un

modelo de desarrollo basado en líneas de productos de software. La misma está basada en la adaptación de metodologías ágiles para el desarrollo de software en equipo como son SCRUM y Open UP, las cuales definen las fases, actividades y artefactos que se deben generar durante el ciclo de vida del software. Open UP se describe como: "*un proceso unificado que aplica acercamientos iterativos e incrementales dentro de un ciclo de vida estructurado. Open UP abraza una filosofía pragmática, ágil que se centra en la naturaleza colaborativa del desarrollo de software*" **(Eclipse 2007).**

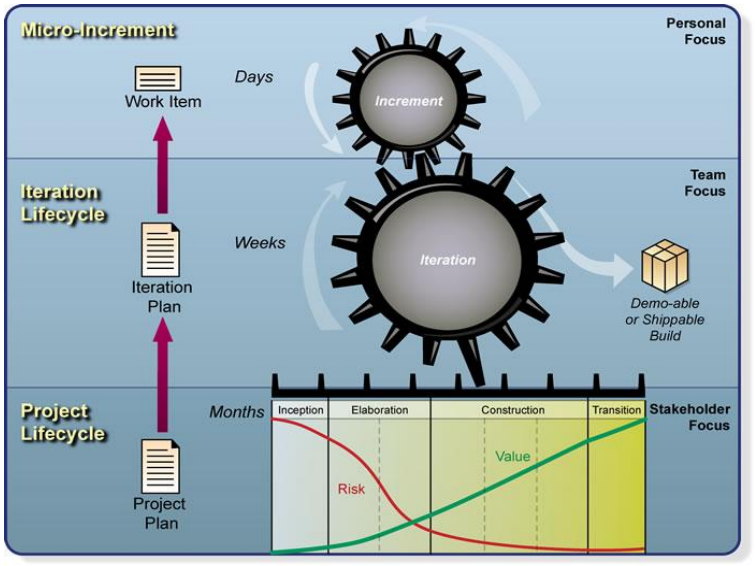


Figura 2 Metodología de Desarrollo de Software

Además Open UP se basa en los siguientes principios:

- Colaboración para alinear intereses y compartir puntos de vista.
- Balancear las prioridades con el objetivo de maximizar el valor del producto para el cliente.
- Centrarse en la arquitectura desde el principio para minimizar los riesgos y organizar el desarrollo.
- Retroalimentación y mejora continua.

SCRUM más que una metodología de desarrollo software, es una forma de auto-gestión de los equipos de desarrollo de software. Cada equipo de desarrollo decide cómo hacer sus tareas y cuánto van a tardar en ello, además ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro.

Básicamente es un framework iterativo, incremental para desarrollar cualquier producto o manejar cualquier trabajo. Permite que los equipos entreguen un conjunto de funcionalidades del sistema en cada

iteración, proporcionando la agilidad necesaria para responder rápidamente a los cambios de requisitos. Desafía constantemente a sus usuarios a centrarse en la mejora, y sus Sprints proporcionan la estabilidad para tratar las necesidades evolutivas que ocurren en cualquier proyecto. **(Scrum 2009)**

Flujo de actividades de la metodología

Particularmente la metodología de la línea define el flujo de trabajo de la siguiente manera: El jefe de la línea se ocupa de mantener el expediente de proyecto en orden y que todos los artefactos estén actualizados. El grupo de análisis principalmente se encarga de definir los requisitos del sistema y los casos de usos del mismo, una vez que están listos cada uno de estos artefactos el grupo de arquitectura se ocupará de establecer la arquitectura del sistema y realizar el diseño del mismo, con estos artefactos el grupo de desarrollo puede comenzar a codificar y es de vital importancia que cada clase que se codifique sea documentada pues así cualquier programador puede seguir el desarrollo de la aplicación con solo mirar el código pues ambos trabajan sobre un mismo estándar y con cada una de las clases documentadas. Una vez que se termina una funcionalidad, se realizan cada una de las pruebas correspondientes de conjunto con el analista.

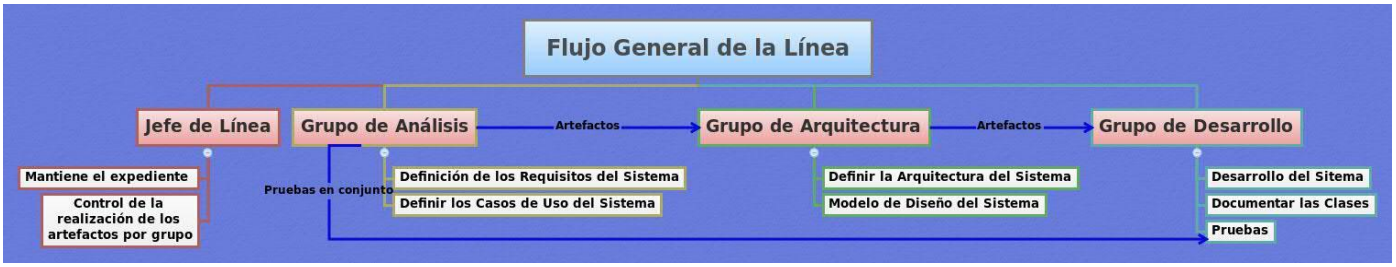


Figura 3 Flujo general de actividades

1.8.2 Herramienta para el modelado

Se decidió utilizar Visual Paradigm (VP) en su versión 6.1 por ser una herramienta robusta, usable y portable. Además de que utiliza UML como lenguaje de modelado. Es colaborativa, soporta múltiples usuarios trabajando sobre el mismo proyecto, permite control de versiones y realizar ingeniería tanto directa como inversa en diferentes lenguajes, entre ellos Java.

Además, se puede representar todos los tipos de diagramas UML para las distintas fases del proceso de desarrollo, como la captura de requisitos, análisis, diseño e implementación. Esta herramienta facilitará la

comunicación, ya que utiliza un lenguaje estándar común a todo el equipo de desarrollo. Presenta la posibilidad de la interoperabilidad con otras aplicaciones como es el Rational Rose.

1.8.3 Entorno de Desarrollo Integrado (IDE)

Un entorno de desarrollo integrado (IDE) es un ambiente de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica de usuario (GUI).

Los IDE proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, C#, Delphi, Visual Basic. En algunos lenguajes, un IDE puede funcionar como un sistema en tiempo de ejecución, donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto, como es el caso de Smalltalk u Objective-C.

Para llevar a cabo la implementación de la aplicación, se realizará con el uso del IDE Eclipse, ya que el mismo es el que se utiliza para el desarrollo del IDE Caxtor y es el adoptado para el desarrollo de aplicaciones por el grupo de arquitectura del centro DATEC.

Dentro de las facilidades que ofrece el IDE Eclipse está la posibilidad de ser extendido a muchos lenguajes y plataformas, tal es el caso del Spket que es un plugin para Eclipse y Aptana que provee un conjunto de utilidades para la edición de JavaScript, sobre todo para la edición de clases que extienden el framework JavaScript ExtJS o que usan la librería. Por esas razones es la herramienta de programación seleccionada para implementar el generador de ayuda propuesto.

1.9 Conclusiones parciales

Con el estudio de los sistemas generadores de ayuda y las búsquedas realizadas con el objetivo de encontrar sistemas similares, se determinaron las características propias de la aplicación a desarrollar. Esto proporciona la medida de las ventajas que se darán a los usuarios que interactúen con el sistema a implementar. También se realizó un estudio de las tecnologías que actualmente son utilizadas para la implementación de sistemas semejantes al propuesto, seleccionando las que serán utilizadas a lo largo del desarrollo del sistema, fundamentándose las elecciones del lenguaje de programación, la metodología de desarrollo y las herramientas a utilizar. Seleccionando como Metodología de Desarrollo la unión de SCRUM y OpenUp, como Lenguaje de Programación a PHP y como Servidor Web Apache.

Capítulo 2: Características del Sistema

Introducción

En este capítulo se realiza una descripción del objeto de estudio. Se utiliza el modelo de negocio para describir los procesos del negocio, se definen las funcionalidades del software a través de los requisitos funcionales y no funcionales; los actores y el diagrama de casos de usos del sistema, finalmente, se realiza la descripción de los casos de usos.

2.1 Modelo de Dominio

Como se ha explicado, no existe una aplicación web que brinde la posibilidad de crear una ayuda que sirva de guía a los desarrolladores que utilicen el IDE Caxtor, por tanto, se propone la realización de un Modelo de Dominio donde se mostrarán los principales conceptos a utilizar en el desarrollo de la aplicación. Esto ayuda a los usuarios, clientes, desarrolladores e interesados en general, a utilizar un vocabulario común para poder entender el contexto en que se ubica el sistema, para capturar correctamente los requisitos y poder construir una aplicación robusta que brinde los beneficios que de ella se esperan.

La figura muestra los principales conceptos y sus relaciones, presentes en el modelo de dominio.

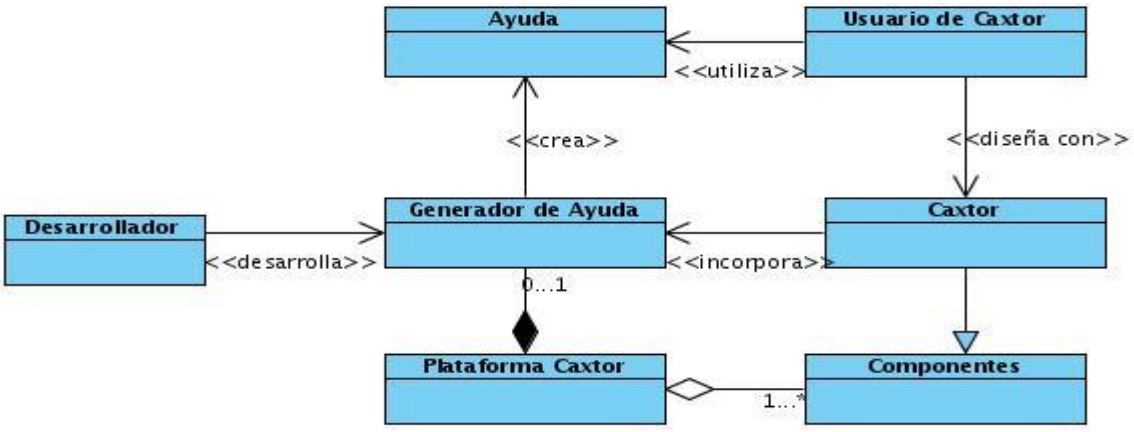


Figura 4 Modelo de Dominio

2.1.1 Conceptos del Modelo de Dominio

Descripción de los conceptos representados en el Modelo del Dominio, los cuales simbolizan objetos del mundo real que se encuentran dentro del dominio del negocio, con la necesidad de comprender las relaciones que se establecen entre las diferentes clases.

Usuario de Caxtor: Este objeto representa el rol que juegan los diferentes usuarios que utilizan a Caxtor para el diseño de las interfaces.
Caxtor: Representa el IDE como tal.
Ayuda: Es el producto creado con el generador.
Componentes: Referentes a los componentes de la Plataforma Caxtor.
Plataforma Caxtor: Plataforma basada en ExtJS 3.0, aunque intenta mantener una separación elemental de este origen. Incluye toda la definición de componentes (visuales y de representación de datos), acciones, y otros tipos de patrones que se encuentran con frecuencia en la capa de presentación de una aplicación Web.
Generador de ayuda: Aplicación a desarrollar con el fin de construir la ayuda y generarla.
Desarrollador: Rol que juega la persona que desarrollará el generador de ayuda.

Una vez definidos los conceptos que forman parte del modelo de dominio, el siguiente paso es la formulación de una propuesta de sistema para guiar el proceso de elaboración de la solución.

2.2 Propuesta del Sistema

El presente trabajo llevará a cabo la implementación de un sistema generador de ayuda web, que provea las funcionalidades que dan respuestas a las necesidades de los usuarios. Dicha aplicación debe permitir la gestión (crear, modificar y eliminar), así como exportar e importar una ayuda previamente creada. Dicho sistema será implementado como un componente de la plataforma Caxtor, haciendo uso del framework ExtJS para la capa de presentación, PHP para la capa intermedia y las principales funcionalidades, así como SQLite para el almacenamiento de los datos.

2.3 Requerimientos del Sistema

Una vez que se realiza el modelado del negocio se puede comenzar a ejecutar el proceso de captura de

requisitos del sistema. Los requisitos son la condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente software para satisfacer un contrato, estándar, u otro documento impuesto formalmente. A continuación se exponen los siguientes requisitos que debe cumplir la aplicación a desarrollar. La identificación de estos se ha realizado a partir de las necesidades reales planteadas por los desarrolladores de Caxtor.

2.3.1 Requerimientos Funcionales

Los requerimientos funcionales no alteran la funcionalidad del producto, lo cual quiere decir que estos se mantienen invariables sin importarle con que propiedades o cualidades se relacionen.

Después de un análisis de las principales necesidades de los clientes y las características que el sistema debe cumplir se definió un conjunto de requisitos funcionales que se muestran a continuación.

RF1. Gestionar Proyecto de Ayuda.

RF1.1 Crear Proyecto.
RF1.2 Modificar proyecto.
RF1.3 Eliminar Proyecto.
RF1.4 Crear Tema.
RF1.5 Modificar Tema.
RF1.6 Eliminar Tema.
RF1.7 Crear Tópico.
RF1.8 Modificar Tópico.
RF1.9 Eliminar Tópico.

RF2. Gestionar Contenido de Ayuda.

RF2.1 Insertar Texto.
RF2.2 Modificar Texto.
RF2.3 Eliminar Texto.
RF2.4 Insertar Imagen.
RF2.5 Eliminar Imagen.
RF2.6 Insertar Tabla.

RF2.7 Modificar Tabla.

RF2.8 Insertar Animación Flash.

RF3. Vista Previa.

RF4. Importar Proyecto.

RF5. Generar Ayuda.

RF5.1 Publicar Ayuda.

RF6. Salir de la Aplicación.

2.3.2 Requerimientos no Funcionales

Los requerimientos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con toda la funcionalidad requerida, las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

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

Software:

- El sistema se implementará con tecnología PHP 5.3.
- Se recurrirá a la tecnología Apache versión 2.2 o superior para el servidor Web.
- Para el correcto funcionamiento, en la PC servidor, configurar el php.ini como a continuación se describe:
    1. Asegurar que el límite de la memoria (memory_limit = 8M), tenga un tamaño suficiente.
    2. Asegurar que el tamaño máximo permitido para los archivos subidos (upload_max_filesize = 2M), tenga un tamaño suficiente.
    3. Asegurar que el tamaño máximo de los datos que PHP aceptará por POST (post_max_size = 8M), tenga un tamaño suficiente.
    4. Habilitar la extensión php_zip.

- En las computadoras de los clientes debe estar instalado un navegador Web que sea capaz de interpretar JavaScript y CSS.

Hardware:

Para las computadoras del cliente:

- Se requiere tengan tarjeta de red.
- Se requiere tengan al menos 256 MB de memoria RAM.
- Procesador 2.8 GHz como mínimo.

Para los servidores:

- Se requiere tarjeta de red.
- Se requiere tenga al menos 512 MB de RAM.
- Se requiere al menos 10 GB de disco duro.
- Procesador Pentium IV 3.0 GHz o superior.

Apariencia o interfaz externa:

- La aplicación propuesta será usada por personas que tengan conocimientos medios de informática, por lo que la interfaz debe ser amigable y fácil de usar, de manera que no sea una dificultad para los usuarios el trabajo con la misma en especial para aquellos que poseen pocas habilidades en el manejo de aplicaciones Web.
- El diseño responderá a la ejecución de acciones de una manera rápida, minimizando los pasos a dar en cada proceso.
- Todos los textos y mensajes de la aplicación aparecerán en idioma español.

Usabilidad:

- La ayuda debe ser de fácil manejo para los usuarios que tengan niveles básicos sobre la computación o hayan realizado algún trabajo previo con sistemas similares.
- Debe facilitar, principalmente, el ser manejado por usuarios que estén vinculados a los procesos que se llevan a cabo en el desarrollo de Caxtor.

Rendimiento:

- Para un funcionamiento óptimo de la aplicación se seguirán las diferentes técnicas de elaboración de la Web. La eficiencia del producto estará determinada en gran medida por el aprovechamiento de los recursos que se disponen en el modelo cliente/servidor, y la velocidad de las consultas de la base de datos. La aplicación propuesta debe ser rápida y el tiempo de respuesta debe ser el

mínimo posible, adecuado a la rapidez con que el cliente requiere la respuesta a su petición.

Seguridad:

- Protección contra acciones no autorizadas o que puedan afectar la integridad de la información.
- Mostrar opción de advertencia antes de borrar cualquier elemento o información que pueda existir.

Soporte:

- El sistema debe dar la posibilidad de incorporarle nuevas funcionalidades en caso de ser necesarias.
- Las pruebas realizadas al sistema deben permitir evaluar sus ventajas y funcionalidades, además de detectar los errores que presenta.

Portabilidad:

- La aplicación debe funcionar en varias plataformas, siendo posible su acceso a través de un navegador Web.

Disponibilidad:

- La aplicación deberá estar disponible en todo momento para aquellas personas con acceso a la información y los mecanismos utilizados para lograr la seguridad no deben ser un obstáculo a los usuarios para obtener los datos deseados en un momento dado.
- La ayuda deberá estar disponible las 24 horas del día para todos los usuarios con derechos a utilizarla.

A manera de resumen se puede expresar que los requisitos especificados representan la base para el Modelo de Casos de Uso del sistema. En el próximo epígrafe se enumeran los actores del sistema y una descripción de los casos de uso identificados.

2.4 Modelo del Sistema

Una vez que se identificaron y se clasificaron los requerimientos del sistema es necesario identificar los actores y casos de usos. Los casos de usos son priorizados y detallados, realizando la descripción de cada uno de ellos.

2.4.1 Definición de los actores del sistema

En el sistema se identificó 1 actor, el cual está relacionado con los diferentes casos de usos beneficiándose de ellos.

2.4.2 Determinación y Justificación de los Actores del Sistema

Tabla 1 Actor del sistema y justificación

Actor	Justificación
Usuario.	Representa a una persona que utilizará el producto para crear la ayuda y generarla.

2.4.3 Definición de los Casos de Uso del Sistema

Una vez descrito los actores se realiza el proceso de identificación de los casos de uso apoyados en los requisitos del sistema, los casos de uso identificados fueron:

Casos de uso base.

- Gestionar Proyecto de Ayuda.
- Gestionar Contenido de Ayuda.
- Generar Ayuda.

Casos de uso extendidos.

- Importar Proyecto de Ayuda.

2.4.4 Diagrama de Casos de Uso del Sistema

Para la realización del diseño se deben conocer las características principales del sistema especificando las diferentes acciones que el sistema debe permitir y los actores que intervienen en las mismas. Las relaciones que se establecen entre estos dos elementos se reflejan en el siguiente diagrama de casos de uso del sistema. Este representa gráficamente a los procesos y su interacción con los actores.

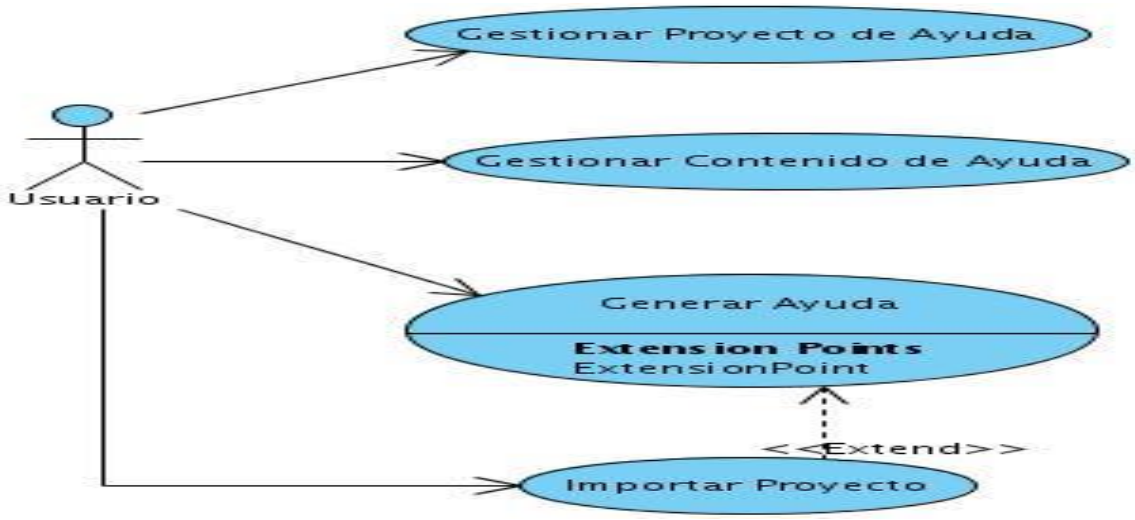


Figura 5 Diagrama de Casos de Uso del Sistema

2.4.5 Descripción Detallada de los Casos de Uso del Sistema

Tabla 2 Descripción del Caso de uso Generar Ayuda

CUS 3	Generar Ayuda.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el usuario decide generar un proyecto de ayuda creado.
Responsabilidades	
Referencias.	R5
Precondiciones	Debe haber al menos un proyecto de ayuda con su contenido. Seleccionar el proyecto a generar.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El caso de uso comienza cuando un usuario decide generar el proyecto de ayuda, para lo cual debe seleccionar el proyecto a generar y luego pulsa el botón Generar o mediante el atajo Ctrol+Shift+G.	

2- El usuario selecciona el proyecto a generar.	2.1 El sistema habilita el botón Generar.
3.- El usuario pulsa el botón Generar.	3-1.- El sistema muestra un mensaje al usuario, indicando que el proyecto de ayuda se está generando. 3.2- Una vez terminado este proceso, la aplicación muestra la ayuda generada en un compactado de extensión **.zip,** lista para ser descargada.
4.- El usuario decide descargar el proyecto de ayuda generado.	4.1 El proyecto se descarga en el directorio de descargas del navegador.
Prioridad	Crítico
Poscondiciones	Queda generado y descargado el proyecto de ayuda.

Las descripciones detalladas del resto de los casos de uso se muestran en el ANEXO I

2.5 Patrones de Casos de Uso

La experiencia en la utilización de casos de uso ha evolucionado en un conjunto de patrones que permiten con más precisión reflejar los requisitos reales, haciendo más fácil el trabajo con los sistemas, y mucho más simple su mantenimiento. Dado un contexto y un problema a resolver, estas técnicas han mostrado ser la solución adoptada en la comunidad del desarrollo de software. Se presentan a modo de herramientas que permiten resolver los problemas que se les planteen a los desarrolladores de una forma ágil y sistemática. Estos patrones se enfocan hacia el diseño y las técnicas utilizadas en modelos de alta calidad, y no en cómo modelar usos específicos.

Utilizando estos patrones, arquitectos, analistas, ingenieros, y gerentes pueden lograr mejores resultados de forma más rápida. **(EVA 2009)**

2.5.1 Patrón de Casos de Uso Utilizado

Es una relación de un caso de uso de extensión a un caso de uso base, que especifica cómo el comportamiento definido por el caso de uso de extensión puede insertarse dentro del comportamiento definido por el caso de uso base. La relación contiene una condición y referencia una secuencia de puntos de extensión en el caso de uso base. Una vez que una instancia del caso de uso ejecuta un comportamiento referenciado por el primer punto de extensión de la relación, la condición es evaluada. Si se cumple, la secuencia de la instancia se extiende para incluir la secuencia del caso de uso extensión. Las diferentes partes del caso de uso extensión son insertadas en los lugares definidos por la secuencia de puntos de extensión de la relación: una parte en cada punto de extensión referenciado.

Una relación de este tipo (extend) se emplea para mostrar alguna de las siguientes situaciones:

- Funciones opcionales
- Funciones complementarias que pueden ejecutarse en base a la selección del actor.

2.6 Conclusiones parciales

En este capítulo se estudiaron los procesos del negocio referente al IDE Caxtor, lo cual permitió comprender las necesidades existentes e identificar las actividades a automatizar.

Estipular las características del sistema a través de requerimientos expresados por el cliente, sirvió de punto de partida para determinar los casos de uso, estos se representaron a través de un Diagrama de Casos de Uso del Sistema, describiéndose paso a paso todas las acciones de los actores del sistema con los casos de uso que interactúan, los cuales guiarán a partir de ahora el proceso de desarrollo del sistema a implementar.

Capítulo 3: Diseño del Sistema

Introducción

El análisis y diseño constituyen partes fundamentales dentro del proceso de desarrollo de software. El objetivo del análisis es identificar un esbozo preliminar del comportamiento requerido a partir de los elementos de modelación del sistema. El diseño por su parte transforma este esbozo preliminar y algunas veces idealizado en un conjunto de elementos del modelo que serán posteriormente implementados. Las clases del análisis son con frecuencia lo suficientemente fluidas, cambiables y evolucionan satisfactoriamente antes de fraguarse en actividades del diseño. En el presente capítulo se plantea el diseño del sistema, utilizando para su modelación el UML y quedando plasmados los diagramas de clases del diseño y de interacción como artefactos fundamentales en esta fase. Los mismos han sido separados por casos de uso para facilitar su comprensión.

3.1 Arquitectura

La arquitectura del software proporciona una visión global del sistema a construir. Describe la estructura y la organización de los componentes del software, sus propiedades y las conexiones entre ellos. Los componentes del software incluyen módulos de programas y varias representaciones de datos que son manipulados por el programa. Además, el diseño de datos es una parte integral para la derivación de la arquitectura del software. La arquitectura marca decisiones de diseño tempranas y proporciona el mecanismo para evaluar los beneficios de las estructuras de sistema. **(Pressman, 2005)**

La arquitectura no es un software operacional propiamente dicho, sino que es la representación que provee al ingeniero de software analizar la efectividad del diseño para la consecución de los requisitos fijados, además permite considerar las diferentes variantes arquitectónicas y ayuda a disminuir los posibles riesgos que pudiera correr el desarrollo de software.

3.1.1 Arquitectura n-Capas

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

los problemas de escalabilidad, disponibilidad, seguridad e integración que pueden presentar las aplicaciones compactas, para ello se ha generalizado la división de las aplicaciones en capas. El modelo n-tier (n-capas) de informática distribuida ha emergido como la arquitectura predominante para la construcción de aplicaciones multiplataforma en la mayor parte de las empresas pertenecientes a Fortune 1000. En esta arquitectura se agrega la capa que surge de separación definitiva de la regla de negocio de la de Acceso a Datos. Esto trae como ventaja que se aísla definitivamente la lógica de negocios de todo lo que tenga que ver con el origen de datos, de este modo ante cualquier cambio eventual, solo se deberá tocar un módulo específico, así como al momento de plantear la escalabilidad del sistema no se afrontarán grandes modificaciones.

3.1.1.1 Arquitectura en tres capas

La arquitectura que se utilizó para la realización de la aplicación fue **Modelo en capas.** El patrón de arquitectura en capas genera una organización con cierta jerarquía permitiendo que cada capa provea a la superior de los servicios que requiere. Aunque pueden implementarse sistemas con dos, tres y n capas el ejemplo más representativo lo constituye el sistema con arquitectura de tres capas.

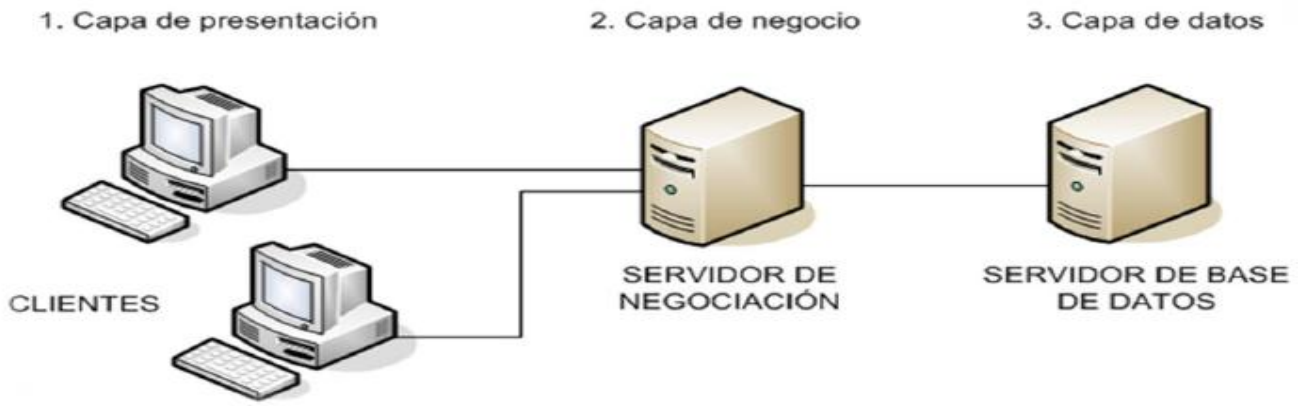


Figura 6 Representación del funcionamiento de un sistema con arquitectura de tres capas

Las tres capas planteadas son:

- Capa de Presentación.

Está habilitada para el manejo de las interfaces e interacción con los usuarios. Es la cara del sistema y debe ser funcional para lograr una buena comunicación entre el sistema y los agentes externos.

- Capa de Negocio (Capa Lógica).

La forman los elementos que dentro del software se encargan de la automatización de los procesos de negocio realizado por los usuarios. Se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. Contiene las reglas que deben cumplirse.

- La capa de Datos.

Reúne las capacidades que se ocupan del manejo de los datos. Puede estar formada por uno o más gestores de bases de datos. Recibe las órdenes de almacenamiento o recuperación de la información desde la capa de negocio.

Ventajas de una arquitectura de tres capas

Las llamadas de la interfaz del usuario, en la estación de trabajo, al servidor de capa intermedia, son más flexibles que en el diseño de dos capas, ya que la estación sólo necesita transferir parámetros a la capa intermedia. Con la arquitectura de tres capas, la interfaz del cliente no es requerida para comprender o comunicarse con el receptor de los datos. Por lo tanto, esa estructura de los datos puede ser modificada sin cambiar la interfaz del usuario en la computadora. El código de la capa intermedia puede ser reutilizado por múltiples aplicaciones si está diseñado en formato modular. Esto puede reducir los esfuerzos de desarrollo y mantenimiento, así como los costos de migración. La separación de roles en tres capas, hace más fácil reemplazar o modificar una capa sin afectar a los módulos restantes. Separando la aplicación de la base de datos, se hace más fácil utilizar nuevas tecnologías de agrupamiento y balance de cargas. Separando la interfaz de usuario de la aplicación, se libera de gran procesamiento a la estación de trabajo y permite que las actualizaciones de la aplicación sean centralizadas en el servidor de aplicaciones.

Con el uso de este patrón arquitectónico se logra en cierta medida darle organización al software y ocultar las tecnologías usadas. En sistemas de gran tamaño las capas son casi siempre entidades muy complejas.

3.2 Modelo de Diseño

Para poder realizar un sistema que soporte todos los requerimientos especificados, tanto funcionales

como no funcionales, se necesita modelarlo, lo cual se hace a través del diseño. El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar.

Propósitos del flujo de trabajo diseño:

- Adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia además de las tecnologías de interfaz de usuario.
- Crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.
- Descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo.
- Capturar las interfaces entre los subsistemas antes en el ciclo de vida del software, lo cual es muy útil cuando se crean interfaces como elementos de sincronización entre diferentes equipos de desarrollo.

3.2.1 Diagrama de Clases del Diseño

Los diagramas de clases son diagramas de estructura estática que muestran las clases del sistema y las relaciones entre ellas. Los diagramas de clase son el pilar básico del modelado con UML, siendo utilizados tanto para mostrar lo que el sistema puede hacer, como para mostrar cómo puede ser construido. Cuando se crea un diagrama de clases, se está modelando una parte de los elementos y relaciones que configuran la vista de diseño del sistema.

El diseño del sistema a construir es un diseño basado en la Web puesto que la solución informática a implementar es una aplicación Web. Sin embargo, en la práctica existen problemas cuando se trata de utilizar el diagrama de clases tradicional para modelar aplicaciones Web. Si se mirara una clase como una página Web en el modelo, habría una confusión pues no se sabría distinguir cuáles atributos, operaciones y relaciones están activas en el servidor y cuáles están activas cuando el usuario está interactuando con la página en el navegador cliente. Por este motivo, no se puede mapear una página Web como una clase UML pues no se alcanzaría una correcta comprensión del sistema.

Actualmente el UML brinda solución para modelar aplicaciones Web, para esto tiene varios mecanismos de extensión los cuales están compuestos por estereotipos, valores etiquetados y restricciones. En la presente investigación se construye una propuesta para el diagrama de clases del diseño basándose en la implementación de la arquitectura en 3 capas.

Se presenta a continuación el diseño de la aplicación.

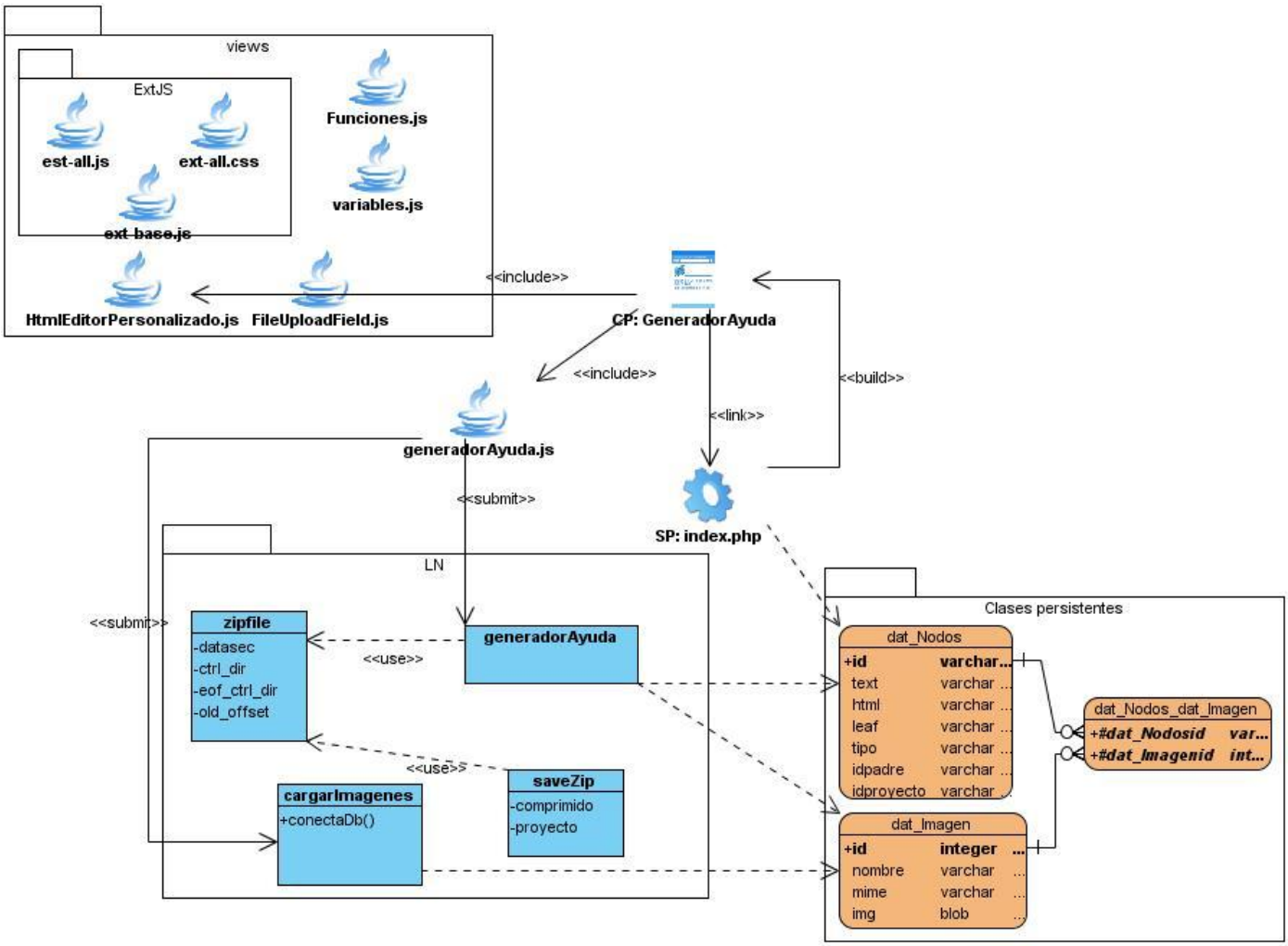


Figura 7 Diagrama de Clases de Diseño Web General

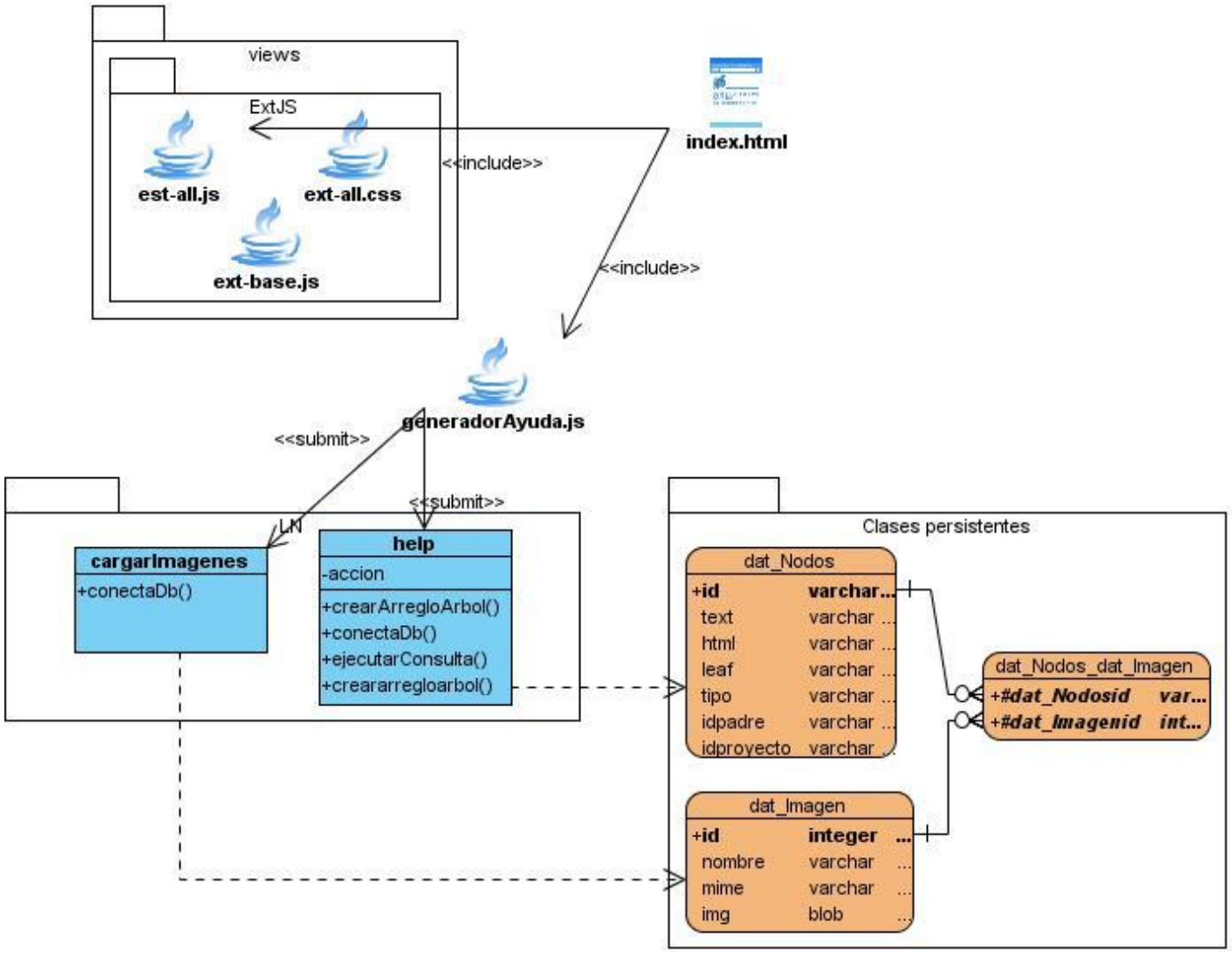


Figura 8 Diagrama de clases del diseño web del componente help

3.2.1.1 Descripción de las clases del diseño ANEXO II

3.2.2 Diagramas de interacción

Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema, lo que conlleva a modelar instancias concretas de clases interfaces, componentes y nodos, junto con los mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento. En el contexto de las clases describen la forma en que grupos de objetos colaboran para proveer un comportamiento.

En los diagramas de interacción se muestra un patrón de interacción entre objetos. Existen dos tipos de diagramas de interacción, ambos basados en la misma información, pero cada uno enfatizando un aspecto particular: Diagramas de Secuencia y Diagramas de Colaboración. En este trabajo se utilizará para la modelación del diseño de la herramienta los diagramas de secuencia, debido al nivel de detalle que describen, proporcionando un mayor flujo de información para el desarrollo de la herramienta.

3.2.2.1 Diagramas de Secuencia

El diagrama de secuencia es uno de los diagramas más efectivos para modelar interacción entre objetos en un sistema. Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso del sistema, este además contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes pasados entre los objetos. A continuación las figuras de los distintos diagramas de secuencia de la herramienta “Generador de ayuda”.
Se representa a continuación el diagrama de secuencia para el caso de uso “Generar Ayuda”.

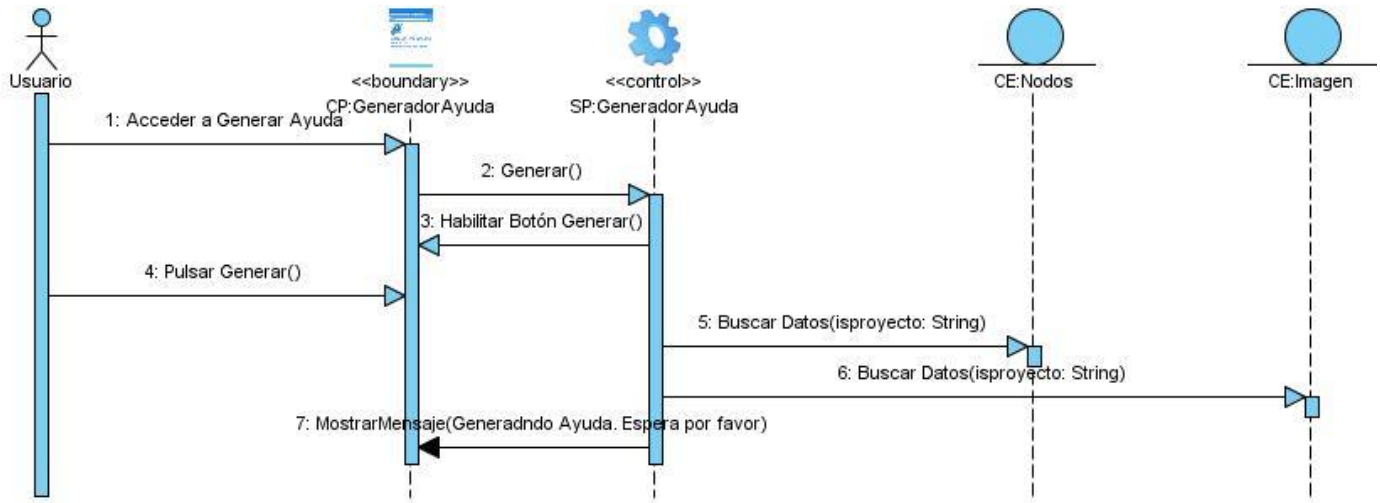


Figura 9 Diagrama de Secuencia CU: Generar Ayuda

El resto de los diagramas de secuencia se muestran en el ANEXO III

3.3 Patrones de diseño

Un patrón de diseño es: **(Valencia 2005)**

- una solución estándar para un problema común de programación.
- una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios.
- un proyecto o estructura de implementación que logra una finalidad determinada.
- un lenguaje de programación de alto nivel.
- una manera más práctica de describir ciertos aspectos de la organización de un programa.
- conexiones entre componentes de programas.
- la forma de un diagrama de objeto o de un modelo de objeto.

Los patrones de diseño pueden incrementar o disminuir la capacidad de comprensión de un diseño o de una implementación, disminuirla al añadir accesos indirectos o aumentar la cantidad de código, disminuirla al regular la modularidad, separar mejor los conceptos y simplificar la descripción. **(Valencia 2005)**
A continuación se describen dentro de las clasificaciones de los patrones de diseño que existen, cuáles patrones se utilizarán en esta investigación.

Patrones de asignación de responsabilidades

Patrones GRASP

Los patrones GRASP son parejas de problema solución con un nombre, que codifican buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades. Describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. **(VISCONTI 2005)**
Durante el diseño se aplicarán estos patrones porque los mismos propician la modularidad y consistencia del sistema así como la mayor independencia entre clases. Se describen a continuación los que serán utilizados en esta investigación:
Patrón creador (Creator, GRASP de Craig Larman): Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. Lo que define este patrón es que una instancia de un objeto la tiene que crear el objeto que tiene la información para ello. ¿Qué significa esto?, pues que si un objeto A utiliza específicamente otro B, o si B forma parte

de A, o si A almacena o contiene B, o si simplemente A tiene la información necesaria para crear B, entonces A es el perfecto creador de B. **(VISCONTI 2005)**

Patrón Alta Cohesión: Mantiene la complejidad dentro de límites manejables, es decir, asigna una responsabilidad de modo que la cohesión siga siendo alta. La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. **(VISCONTI)**

Beneficios:

1. Mejoran la claridad y facilidad con que se entiende el diseño.
2. Se simplifica el mantenimiento y las mejoras de funcionalidad.
3. A menudo, se genera un bajo acoplamiento.
4. Soporta mayor capacidad de reutilización.

Bajo Acoplamiento: Es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Acoplamiento bajo significa que una clase no depende de muchas clases. **(VISCONTI 2005)**

Beneficios:

1. No se afectan por cambios de otros componentes.
2. Fáciles de entender por separado.
3. Fáciles de reutilizar.

Experto: Su objetivo es asignar una responsabilidad al experto en información, la clase que cuenta con la información necesaria para cumplir determinada responsabilidad. Distribuye el comportamiento entre las clases que cuentan con la información requerida. **(VISCONTI 2005)**

3.4 Diseño de la Base de Datos

El diseño de la base de datos es un paso crucial en el desarrollo del software que dará solución al problema a resolver de la presente investigación. Una correcta selección de las tablas que formarán parte de la misma, así como de la información que contendrá cada una y sus relaciones, y teniendo en cuenta

las prácticas adecuadas de normalización, proporcionarán un rendimiento del sistema adecuado a las exigencias del cliente. Todo esto es relevante además en la minimización de redundancia de la información y en la erradicación de la excesiva e innecesaria existencia de campos nulos en la base de datos.

3.4.1 Diagrama de Clases Persistentes

Luego de analizar las funcionalidades que deberá llevar a cabo el sistema propuesto, se procedió a identificar las clases persistentes, que son aquellas que contienen la información valiosa y que necesariamente debe ser almacenada en una base de datos. Estas clases tienen su origen las clases clasificadas como entidades. El Diagrama de Clases Persistentes modela las relaciones existentes entre estas clases.

Figura 10 Diagrama de Clases Persistentes

3.4.2 Modelo Entidad-Relación

En el modelo Entidad-Relación se representan gráficamente las clases persistentes como tablas de la base de datos, con sus propiedades y relaciones. Durante la creación de este modelo, son realizadas las transformaciones necesarias para ajustar el diseño a la forma normal deseada: se crean las llaves foráneas teniendo en cuenta las distintas relaciones existentes y surgen las tablas como producto de las relaciones de “muchos a muchos” existentes. Es posible afirmar que la base de datos del Generador de ayuda, cuyo modelo Entidad-Relación se muestra en la Figura 11, se encuentra en la 3ra forma normal porque para todas sus tablas se cumple que:

- ✓ No existen atributos multivaluados.
- ✓ Los atributos no llaves dependen funcional y completamente de la llave primaria.
- ✓ No existen las dependencias transitivas de atributos no llaves respecto a la llave primaria.

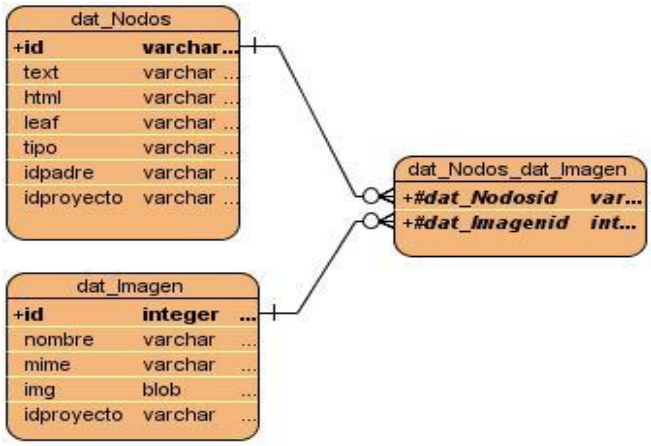


Figura 11 Diagrama Entidad Relación

3.5 Descripción de las tablas de la Base de Datos

Tabla 3 Descripción de la tabla de la BD: dat_Nodos

Nombre: dat_Nodos.		
Descripción: Tabla referente a los proyectos, temas y tópicos de la aplicación, la cual contiene los datos generales y específicos de los mismos, id, texto, contenido, tipo, entre otros.		
Atributo	**Tipo**	**Descripción**
• Id • Text • HTML • Leaf • Tipo • Idpadre • Idproyecto	• Varchar • Varchar • Varchar • Boolean • Varchar • Varchar • Varchar	• Identificador del nodo. • Texto del nodo. • Contenido HTML del nodo. • Saber si el nodo es hoja. • Saber si el nodo es: Proyecto, Tema ó Tópico. • Identificador del padre del nodo. • Identificador del Proyecto.

Tabla 4 Descripción de la tabla de la BD: dat_Imagen

Nombre: dat_Imagen.		
Descripción: Tabla referente a las imágenes y animaciones flash de la aplicación, la cual contiene los datos generales y específicos de las mismas, id, nombre, formato, código, así como el id del proyecto asociado.		
Atributo	**Tipo**	**Descripción**
• Id • Nombre • Mime • Img • Idproyecto	• Integer • Varchar • Varchar • Blob • Varchar	• Identificador de la imagen. • Nombre de la imagen. • Formato de la imagen. • Código de la imagen. • Identificador del Proyecto al que pertenece la imagen.

3.6 Conclusiones parciales

En este capítulo se ha desarrollado uno de los flujos fundamentales en el ciclo de vida del software: el flujo de trabajo Diseño. A partir de los casos de usos del sistema se modeló el diseño con los artefactos principales a generar en el mismo, como son los diagramas de clases del diseño, diagramas de interacción, el diagrama de clases persistentes, el modelo de datos, además de la descripción de las tablas de la BD etc., los artefactos mencionados anteriormente constituyen el punto de entrada para lograr una eficiente implementación y mayor tiempo de vida útil del sistema.

Capítulo 4: Implementación del Sistema

Introducción

Este capítulo está enfocado en el flujo de trabajo de implementación para dar solución a los requisitos especificados. Vale destacar que este flujo de trabajo está fuertemente regido por el flujo de Análisis y Diseño. En este capítulo se expondrán los artefactos diagrama de despliegue y diagrama de componentes, este último muestra la distribución física de los componentes para la implementación, además se evidencia la presencia de código relevante en la construcción de la herramienta, así como la validación de la misma.

4.1 Modelo de despliegue

El modelo de despliegue de la aplicación está compuesto por dos nodos que representan a una computadora cliente y el otro un servidor, ambas están interconectadas utilizando el protocolo HTTP.

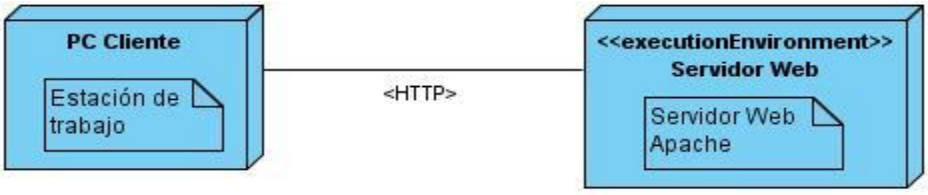


Figura 12 Diagrama de despliegue

4.2 Diagrama de Componentes

Los diagramas de componentes representan cómo un sistema es dividido en componentes y muestra las dependencias entre estos. Los componentes físicos incluyen archivos, cabeceras, librerías compartidas, módulos, ejecutables, o paquetes y pueden ser usados para modelar y documentar cualquier arquitectura de sistema. Debido a que estos son muy parecidos a los diagramas de casos de usos son utilizados para modelar la vista estática de un sistema, mostrando la organización y las dependencias entre un conjunto de componentes. Uno de los usos principales es que puede servir para ver qué componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.

A continuación se representa el diagrama de componentes utilizado en la presente investigación, compuesto por diferentes archivos de extensión .js, que responden al funcionamiento del sistema al lado del cliente con el uso del framework ExtJS. Además de la representación física de cada una de las clases de extensión .php, para llevar a cabo cada una de las acciones en el servidor.

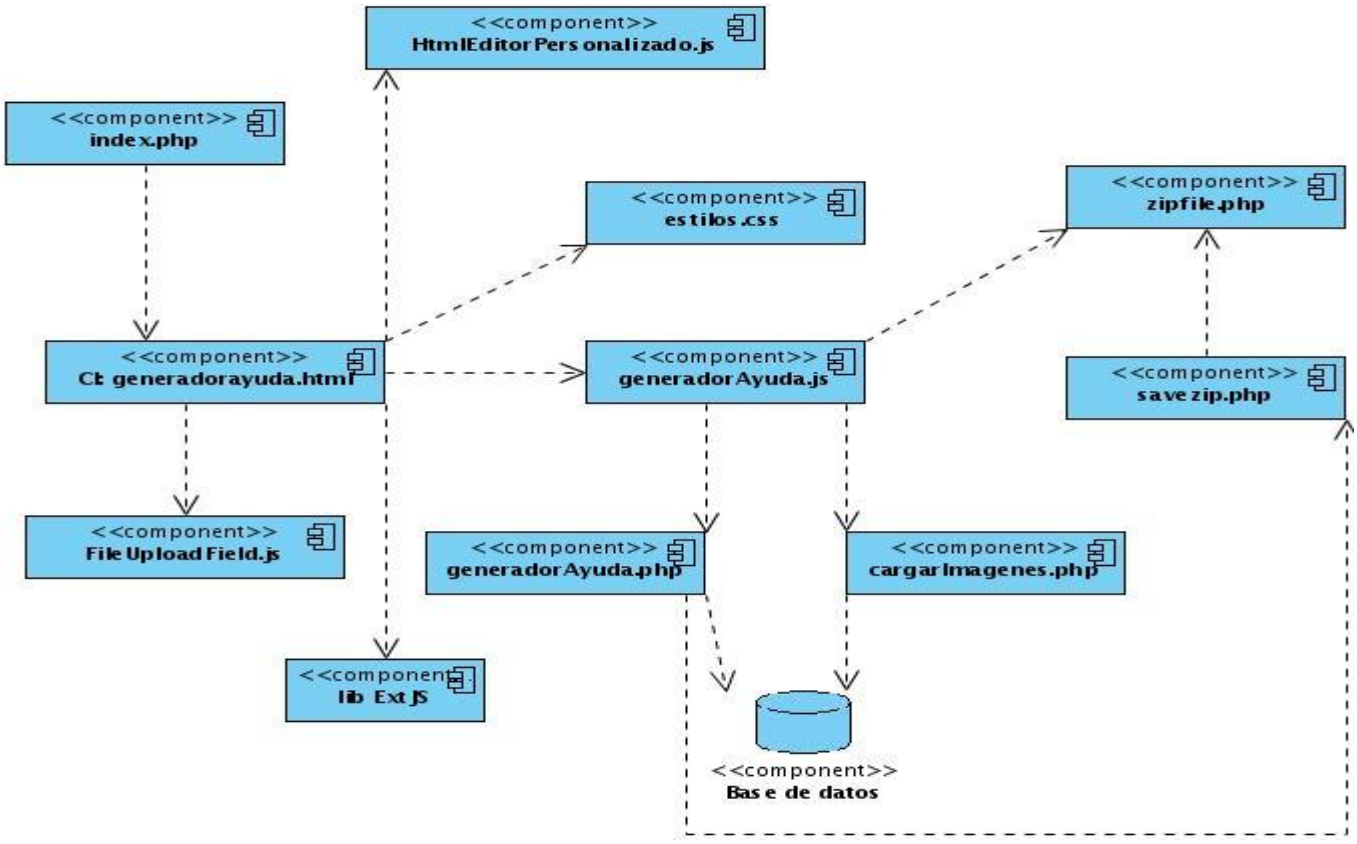


Figura 13 Diagrama de Componentes

4.3 Estándares de codificación.

La combinación de técnicas de codificación sólidas y las buenas prácticas de programación con el objetivo de lograr un código robusto, es de vital importancia para la calidad del software. La aplicación continua de un estándar de codificación puede contribuir a obtener un sistema de software fácil de comprender y de mantener.

Algunas reglas generales para codificar o definir un estándar de codificación:

- No usar métodos y variables multipropósito.

- Usar las sentencias SWITCH-CASE en lugar de las sentencias IF-ELSE repetitivas.
- Escribir todos los nombres de las estructuras en el mismo idioma.
- Usar nombres objetivos para las variables y métodos, evitando la ocurrencia de ambigüedad en el código y contribuyendo a la comprensión del mismo.
- Evitar contener más de una clase por archivo.
- Usar la misma sangría en todo el código

Estándar de codificación del sistema generador de ayuda:

Nomenclatura

- Los nombres de cada uno de los elementos del programa deben ser significativos; su nombre debe explicar en lo posible el uso del elemento.
- La mayoría de los elementos se deben nombrar usando sustantivos.
- La forma de construir los nombres será colocando primero el verbo o el sustantivo, seguido de cada uno de sus complementos con la primera letra en mayúscula.

Ejemplos:

```
function importarProyecto()
function eliminarDirectorio($dirname)
function generarZip($id)
function basedatosBackup($bd)
```

Figura 14 Estándar de codificación

4.4 Código fuente

A continuación se muestra el código fuente de dos de las principales funciones del sistema “importarProyecto ()”, esta función es la encargada de importar un proyecto de ayuda previamente creado, básicamente lo que hace la función es comprobar que exista el archivo especificado en la dirección, de existir se crea una instancia de la clase ZipArchive, luego se abre el archivo especificado, se extrae en el directorio tmp el archivo help.zip, quedando el directorio HELP y se cierra el directorio, mediante las funciones importarDatosSqlite() e importarImagenesSqlite() se hace una copia del contenido y las imágenes del fichero .sqlite que se encuentra en el directorio HELP producto de extraer el fichero help.zip y se pasan a la base de datos permanente de la aplicación. Finalmente, se borran el directorio HELP y el archivo help.zip que se encuentran en el directorio tmp. A continuación se describe el algoritmo de la

función “eliminarDirectorio ($dirname)”, encargada de borrar los directorios temporales creados en el directorio tmp: Dada la ruta de un directorio, primeramente comprueba que hay un directorio en esa ruta, de haberlo, abre el directorio, se hace una lectura al directorio y mientras ese directorio tenga archivos los elimina, finalmente cierra el directorio lo borra y retorna verdadero. En caso de no haber un directorio la función retorna falso.

```
function importarProyecto() {
    if (file_exists ( '../tmp/help.zip' )) { // comprueba que exista el archivo help.zip en el directorio tmp
        $zip = new ZipArchive ( );
        $zip->open ( '../tmp/help.zip' );
        $zip->extractTo ( '../tmp/' );
        $zip->close ();
        importarDatosSqlite ();
        importarImagenesSqlite ();
        eliminarDirectorio ( '..\tmp\HELP' );
        unlink ( '../tmp/help.zip' ); // borrar el archivo especificado
    }
}
```

Figura 15 Código fuente función ¨importarProyecto () ¨

```
function eliminarDirectorio($dirname) {
    if (is_dir ( $dirname ))
        $dir_handle = opendir ( $dirname );
    if (! $dir_handle)
        return false;
    while ( $file = readdir ( $dir_handle ) ) {
        if ($file != "." && $file != "..") {
            if (! is_dir ( $dirname . "/" . $file ))
                unlink ( $dirname . "/" . $file );
            else
                eliminarDirectorio ( $dirname . '/' . $file );
        }
    }
    closedir ( $dir_handle );
    rmdir ( $dirname );
    return true;
}
```

Figura 16 Código fuente función ¨eliminarDirectorio () ¨

4.5 Validación a nivel de desarrollador

Un instrumento adecuado para determinar la calidad de un producto software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos.

Una forma de verificar la consistencia de los datos de la aplicación es introduciendo datos erróneos o dejando campos en blanco que son de carácter obligatorio.

Ejemplo:

Para crear un nuevo proyecto, es necesario introducir un nombre en el campo de texto, de lo contrario ocurriría lo siguiente:

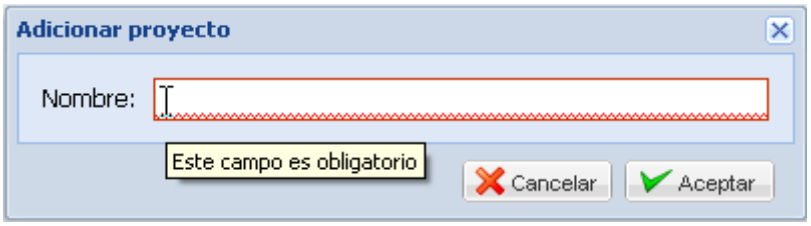


Figura 17 Campo de texto obligatorio

El sistema generador de ayuda para el IDE Caxtor se validó a nivel de cliente utilizando el lenguaje JavaScript, que se incluye en los formularios que necesiten validar los datos antes de enviarlos al servidor, en la misma se crearon expresiones regulares para verificar que los valores de entrada sean los correctos. En caso de entrar valores incorrectos, indicarle al usuario resaltando el campo de entrada y mediante mensajes.

Ejemplo:

Figura 18 Extensión incorrecta

Hay validaciones para diferentes tipos de datos por ejemplo para números enteros, decimales, letras, cadenas de texto, etc.

Existen otras validaciones que inhabilitan ciertos botones y menús, en dependencia de lo que queramos hacer.

Ejemplo:

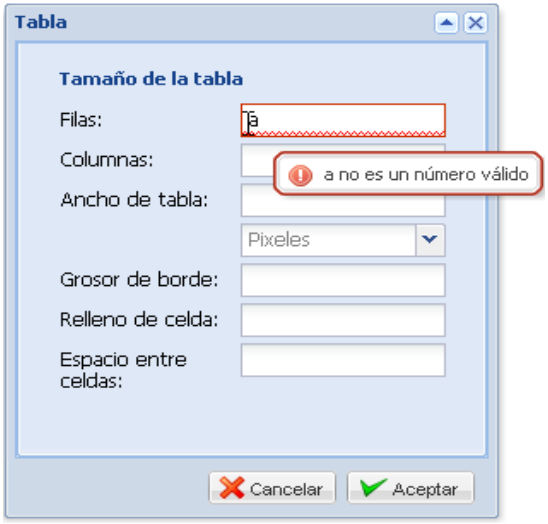


Figura 19 Números enteros

Para poder generar un proyecto de ayuda, este debe estar seleccionado.

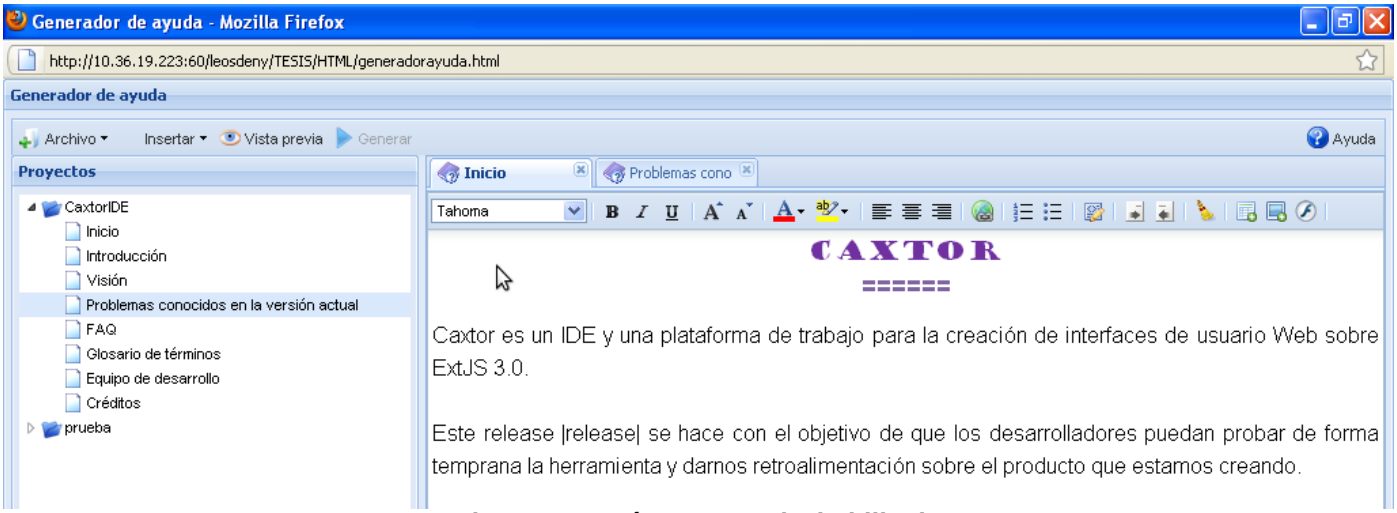


Figura 20 Botón generar deshabilitado

4.6 Conclusiones parciales

La arquitectura y los artefactos generados como resultado del flujo de trabajo Diseño, fueron la base para la realización del Sistema Generador de Ayuda para el IDE Caxtor, con el fin de responder a las principales funcionalidades encaminadas a la gestión de toda la información referente al mismo, y brindar de este modo una guía de apoyo a los usuarios que interactúen con el IDE.

Conclusiones generales

El sistema generador de ayuda para el IDE Caxtor fue desarrollado con la finalidad de llevar a cabo el proceso de gestión de la información referente a esta poderosa herramienta; pero además cuenta con la flexibilidad adecuada para ser aplicada a otras aplicaciones basadas en la web. Luego de la realización, en su totalidad, es posible afirmar que se han cumplido todos los objetivos propuestos a lo largo de este trabajo de diploma. Durante la presente investigación se ha arribado a las siguientes conclusiones:

- ✓ Para la elaboración de la presente investigación se efectuó un análisis de sistemas existentes para la creación de ayuda, esto ayudó en gran medida y sirvió como punto de partida para la realización del nuevo sistema.

- ✓ Las herramientas, metodología de desarrollo y lenguajes utilizados, brindaron el soporte necesario para lograr un producto con los requerimientos deseados, además de proporcionarle al mismo una calidad y rendimiento acordes a las exigencias planteadas por el cliente.

- ✓ Para el diseño y construcción del sistema se tuvo en cuenta el patrón arquitectónico Modelo en Capas, combinado con patrones de diseño. Esto facilitó una mayor reutilización del código y organización de las funcionalidades del sistema. De esta forma se garantiza mayor tiempo de vida útil del sistema.

- ✓ Para la construcción del sistema se emplearon tecnologías y herramientas en su mayoría distribuidas bajo licencias de software libre, logrando la libertad tecnológica del producto y cumpliendo con los lineamientos de producción de software de la Universidad de Las Ciencias Informáticas y el país, garantizando así la obtención de un producto software multiplataforma.

Como se puede observar, como resultado de este trabajo de diploma se dotará a los desarrolladores de aplicaciones web que utilizan el framework ExtJS para el diseño de interfaces, y que utilicen el IDE Caxtor con este propósito, de una ayuda creada con la herramienta producto de la presente investigación, la cual será de gran utilidad para los usuarios de Caxtor.

Recomendaciones

1. Añadir otras funcionalidades contempladas en el desarrollo de la investigación, con el objetivo de incrementar las facilidades que brinda la herramienta, ejemplo: Opciones de impresión, generar ayuda en otro formato, ejemplo: PDF.

2. Utilizar el sistema para crear ayuda de otras aplicaciones basadas en la web, teniendo en cuenta que fue implementado de forma genérica.

3. Utilizar el presente trabajo como bibliografía para posibles investigaciones referentes al tema desarrollado en el mismo.

Referencias bibliográficas

1. **ALAMEDA, E. 2007.** *Practical Rails Project.* New York : Edtion ed. USA: Apress, 2007.
2. **Álvarez, M. A. 2009.** Breve Historia de PHP. [En línea] 2009. [Citado el: 07 de Febrero de 2010.] http://www.desarrolloweb.com/articulos/436.php.
3. Arsys Programación Web. [En línea] [Citado el: 06 de Febrero de 2010.] http://www.arsys.es/soporte/programacion/windows.htm.
4. **2007.** *ECLIPSE, C.A.O. Eclipse Process Framework OpenUP. In.* 2007.
5. **Eguíluz Pérez, Javier. 2008.** *Introduccion a CSS.* 2008.
6. **EVA. 2009.** EVA UCI. [En línea] UCI, 19 de Noviembre de 2009. [Citado el: 02 de Junio de 2010.] http://eva.uci.cu/file.php/102/Curso_2009-2010/Semana_9/CTP_1/Materiales_Basicos/CTP1_Guia_Estudiante.doc.
7. **2010.** IAGP. Apuntes de Ingeniería del software. *Sistemas Informáticos. Universidad de Murcia. Metodologías de desarrollo de software.* [En línea] 2010. [Citado el: 06 de Febrero de 2010.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.
8. ITE, Instituto de Tecnologias Educativas. [En línea] [Citado el: 31 de Marzo de 2010.] http://usuarios.pntic.mec.es/sqlite.php.
9. **LAGOS TORRES, M. 2002.** Introducción al diseño con patrones. [En línea] 2002. [Citado el: 14 de Marzo de 2010.] http://www.elrincondelprogramador.com/default.asp?id=29&pag=articulos/leer.asp.
10. **Méndez, Néstor. 2007.** *Conceptos de Arquitectura Cliente - Servidor.* 2007.
11. **Pressman, R.S. 2005.** *Ingeniería del Software. Un enfoque práctico.* s.l. : V ed. Vol. I, 2005.
12. **2009.** *SCRUM, A. SCRUM. In.* 2009.
13. Softonic. [En línea] [Citado el: 04 de Febrero de 2010.] http://softonic.com.
14. The Apache Sotfware Foundation. Apache. [En línea] [Citado el: 07 de Febrero de 2010.] http://apache.org.
15. **Van Der Henst, Christian y Heredia Santos, Herminio. 2009.** ¿Qué es el PHP? [En línea] 2009. [Citado el: 07 de Febrero de 2010.] http://www.maestrosdelweb.com/editorial/phpintro.
16. **VISCONTI, M. A. A., HERNÁN. 2006.** Fundamentos de Ingeniería de Software, Patrones de Diseño. [En línea] 2006. [Citado el: 14 de Marzo de 2010.] http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf.
17. w3c. es Guía Breve de Tecnologías XML. [En línea] [Citado el: 07 de Febrero de 2010.]

http://www.w3c.es/Divulgacion/Guiasbreves/TecnologiasXML.

18. WebEstilo. Utilidad, programación y mucho más. [En línea] [Citado el: 06 de Febrero de 2010.] http://www.webestilo.com/JavaScript.
19. **2009.** XPPS. El lenguaje HTML. [En línea] 2009. [Citado el: 07 de Febrero de 2010.] http://www.xpps.net/contenido_xppsnet/areatec/HMTL.pdf.
20. **ZAMMETTI, F.W. 2009.** *Practical Ext Js Projects with Gears.* New York : edited by APRESS. Edtion ed., 2009.

Bibliografía

1. ALAMEDA, E. Practical Rails Project. Edition ed. New York , USA: Apress, 2007.
2. ECLIPSE, C.A.O. Eclipse Process Framework OpenUP. In., 2007, vol. 2010.
3. SCRUM, A. SCRUM. In., 2009, vol. 2010.
4. Softonic. [En línea] [Citado el: 04 de Febrero de 2010.] [Disponible en: http://helpndoc.softonic.com].
5. Free Download Manager. [En línea] [Citado el: 04 de Febrero de 2010.] [Disponible en: http://www.freedownloadmanager.org/es/downloads/HelpSmith_58762_p].
6. Instituto Nacional de Estadística e Informática. [En línea] [Citado el: 06 de Febrero de 2010.] [Disponible en: http://www1.inei.gob.pe/biblioineipub/bancopub/inf/Lib5038/defi.HTM].
7. Arsys Programación Web. [En línea] [Citado el: 06 de Febrero de 2010.] [Disponible en: http://www.arsys.es/soporte/programacion/windows.htm].
8. Méndez, Néstor. Conceptos de Arquitectura Cliente - Servidor. 2007.
9. Barrios, Marcio. AJAX ¿El futuro? [En línea] 30 de Julio de 2005. [Citado el: 07 de Febrero de 2010.] [Disponible en: http://www.marciobarrios.com/ajax].
10. Balú XMLHttpRequest& Ajax: Ventajas y Desventajas. [En línea] 2006. [Citado el: 07 de Febrero de 2010.] [Disponible en: http://www.baluart.net/articulo.php?id_art=55].
11. XPPS. El lenguaje HTML. [En línea] 2009. [Citado el: 07 de Febrero de 2010.] [Disponible en: http://www.xpps.net/contenido_xppsnet/areatec/HMTL.pdf].
12. Eguíluz Pérez, Javier. Introducción a CSS. 2008.
13. w3c. es Guía Breve de Tecnologías XML. [En línea] [Citado el: 07 de Febrero de 2010.] [Disponible en: http://www.w3c.es/Divulgacion/Guiasbreves/TecnologiasXML].
14. WebEstilo. Utilidad, programación y mucho más. [En línea] [Citado el: 06 de Febrero de 2010.] [Disponible en: http://www.webestilo.com/JavaScript].
15. Álvarez, M. A. Breve Historia de PHP. [En línea] 2009. [Citado el: 07 de Febrero de 2010.] [Disponible en: http://www.desarrolloweb.com/articulos/436.php].
16. Van Der Henst, Christian y Heredia Santos, Herminio. ¿Qué es el PHP? [En línea] 2009. [Citado el: 07 de Febrero de 2010.] [Disponible en: http://www.maestrosdelweb.com/editorial/phpintro].

17. The Apache Sotfware Foundation. Apache. [En línea] [Citado el: 07 de Febrero de 2010.] [Disponible en: http://apache.org].
18. IAGP. Apuntes de Ingeniería del software. Sistemas Informáticos. Universidad de Murcia. Metodologías de desarrollo de software. [En línea] 2010. [Citado el: 06 de Febrero de 2010.] [Disponible en: http://www.um.es/docencia/barzana/IAGP/Iagp2.html].
19. Mendoza Sánchez, M. A. Metodologías De Desarrollo De Software. Informatizate.net. [En línea] 2006. [Citado el: 06 de Febrero de 2010.] [Disponible en: http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html].
20. Iva, Jacobson, Booch, Grady y Rumbaugh, J. El Proceso Unificado de Desarrollo de software. Madrid: Addison-Wesley. 2000.
21. Hernán, S. M. Tesis de Grado en Ingeniería en Informática. Ciudad de La Habana: Instituto Superior Politécnico José Antonio Echeverría: s.n., 2004.
22. Fowler, Martin. ProgramacionExtrema.org. La Nueva Metodología. [En línea] 2007. [Citado el: 06 de Febrero de 2010.] [Disponible en: http://www.programacionextrema.org/articulos/newMethodology.es.html].
23. creangel.com. Lenguaje Unificado de Modelamiento. [En línea] [Citado el: 07 de Febrero de 2010.] [Disponible en: http://creangel.com/uml/intro.php].
24. creangel.com. ¿Qué es UML? [En línea] [Citado el: 07 de Febrero de 2010.] [Disponible en: http://www.creangel.com/uml/intro.php].
25. NETBEANS.ORG. NetBeans - Development Simplified. [En línea] [Citado el: 08 de Febrero de 2010.] [Disponible en: http://www.netbeans.org/features].
26. CERDA, F. Características de NetBeans. [En línea] [Citado el: 07 de Febrero de 2010.] [Disponible en: http://www.bloog.cl/talks/netbeans65es_cl.pdf].
27. techtear.com. Zend Studio for Eclipse, desarrollo profesional en PHP. [En línea] [Citado el: 08 de Febrero de 2010.] [Disponible en: http://www.techtear.com/2008/01/22/zend-studio-for-eclipse-desarrollo-profesional-en-php].
28. LAGOS TORRES, M. Introducción al diseño con patrones. 2002, Disponible en: http://www.elrincondelprogramador.com/default.asp?id=29&pag=articulos/leer.asp.

29. Materiales Básicos. EVA. [En línea] [Citado el: 08 de Febrero de 2010.] [Disponible en: http://eva.uci.cu/file.php/259/CURSO_2008-2009/Materiales_Basicos/Semana_3/Conf/Conferencia_2_de_Arquitectura_2009ok.doc].
30. Ext JS. [En línea] 2009. [Citado el: 11 de Febrero de 2010.] [Disponible en: http://www.extjs.com].
31. VALENCIA, U. P. D. Rational Unified Process, 2005.
32. VISCONTI, M. A. A., HERNÁN. Fundamentos de Ingeniería de Software, Patrones de Diseño. Disponible en: http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf
33. ZAMMETTI, F.W. Practical Ext Js Projects with Gears. edited by APRESS. Edition ed. New York 2009.
34. Larman, Craig. UML y Patrones. UML y Patrones. s.l.: Prentice Hall, 2006.
35. ITE, Instituto de Tecnologias Educativas. [En línea] [Citado el: 08 de Febrero de 2010.] [Disponible en: http://usuarios.pntic.mec.es/sqlite.php.
36. EVA, Entorno Virtual de Aprendizaje. FT Análisis y Diseño. Modelo de Diseño. [En línea] 2009. http://eva.uci.cu/mod/resource/view.php?id=14069.

Anexos

Anexo I "Descripción Detallada de los Casos de Uso del Sistema**"**

Tabla 5 Descripción del Caso de uso Gestionar Proyecto de Ayuda

CUS 1	Gestionar Proyecto de Ayuda.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el usuario decide crear un proyecto de ayuda.
Referencias.	R1, R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R1.9
Precondiciones	El usuario debe estar dentro de la aplicación.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El caso de uso comienza cuando un usuario solicita crear un nuevo proyecto de ayuda, para ello, accediendo mediante el menú Archivo, la opción Nuevo->Proyecto, ó mediante el atajo Ctrol+Shift+N.	1.1- El sistema solicita el nombre del proyecto a crear.
2- El usuario introduce los datos requeridos (Nombre).	
3- El usuario pulsa el botón aceptar o presiona la tecla Enter.	3.1- La herramienta verifica que los datos se hayan completado. 3.2- La aplicación comprueba si el nombre asignado al proyecto coincide con otro ya existente. 3.3- Guarda los datos.

	3.4- Visualiza en el área de Proyectos, donde definir la estructura del proyecto, brindando en la misma cuatro opciones, que se mostraran a través de un clic derecho sobre el proyecto al cual se desee crear un tema, ó crear un tópico, cambiar su nombre o eliminarlo. a) Crear tema, ver **Sección 1.** b) Crear tópico, ver **Sección 2.** c) Cambiar nombre, ver **Sección 3.** d) Eliminar, ver **Sección 4.**
Sección "Crear tema"	
1- El usuario decide crear un tema, lo mismo puede ser a un proyecto o a un tema en cuestión accediendo mediante un clic derecho, y selecciona la opción "Crear tema".	1.1- El sistema solicita el nombre del tema a crear.
2- El usuario introduce los datos requeridos (Nombre).	
3- El usuario pulsa el botón aceptar o presiona la tecla Enter.	3.1- La aplicación verifica que los datos se hayan completado. 3.2- La aplicación comprueba si el nombre asignado al tema coincide con otro ya existente. 3.3- Guarda los datos. 3.4- Visualiza en el área de Proyectos, un nivel debajo del proyecto o tema, el tema creado.
Sección "Crear tópico"	
1- El usuario decide crear un tópico, lo mismo puede ser a un proyecto o a un tema accediendo mediante un clic derecho, y selecciona la opción "Crear tópico".	1.1- El sistema solicita el nombre del tópico a crear.

2- El usuario introduce los datos requeridos (Nombre).	
3- El usuario pulsa el botón aceptar o presiona la tecla Enter.	3.1- La aplicación verifica que los datos se hayan completado. 3.2- La aplicación comprueba si el nombre asignado al tópico coincide con otro ya existente. 3.3- Guarda los datos. 3.4- Visualiza en el área de Proyectos, un nivel debajo del proyecto o tema, el tópico creado. 3.5- La aplicación abre un editor, en el área de edición (derecha), listo para comenzar a editar la ayuda.
Sección "Cambiar nombre"	
1- El Autor decide cambiar el nombre a un proyecto, tema o tópico, para lo cual, lo selecciona a través de un clic derecho, y escoge la opción "Cambiar nombre".	1.1- La aplicación habilita el campo nombre del proyecto, tema o tópico para que el autor pueda modificarlo.
2- El usuario introduce el nombre que desee.	
3- El usuario da clic en cualquier área de la ventana, ó presiona Enter.	3.1- La aplicación actualiza el archivo donde se almacena la estructura del proyecto.
	3.2- La aplicación cambia el nombre del proyecto, tema o tópico y lo muestra.
Sección "Eliminar"	
1.- El usuario decide eliminar un proyecto, tema o tópico para lo cual accede mediante un clic derecho y selecciona la opción "Eliminar", ó mediante la tecla Delete.	1.1- La aplicación muestra un mensaje de confirmación, preguntándole al usuario si realmente desea eliminar el elemento.
2.- El usuario presiona el botón Sí.	2.1- La aplicación elimina el elemento y todo lo

	que este contiene. 2.2- La aplicación actualiza los datos. 2.3- Muestra los elementos que aún permanecen.
Flujos Alternos	
	3.2. a - En caso de que coincida, la aplicación muestra un mensaje de error y vuelve al paso 2 del flujo normal de eventos.
Flujos Alternos-Sección "Crear tema"	
Acción del Actor	**Respuesta del Sistema**
1.- El usuario no crea ningún tema.	1.1 - La aplicación conserva la estructura inicial del proyecto, sin temas.
Flujos Alternos-Sección "Crear tópico"	
Acción del Actor	**Respuesta del Sistema**
1. - El usuario no crea ningún tópico.	1.1 -La aplicación conserva la estructura inicial del proyecto, sin tópicos.
Flujos Alternos-Sección "Cambiar nombre"	
Acción del Actor	**Respuesta del Sistema**
1.- El Autor no introduce ningún nombre.	1.1- La aplicación conserva el nombre que tenía dicho proyecto, tema o archivo.
Flujos Alternos-Sección "Eliminar"	
Acción del Actor	**Respuesta del Sistema**
1.- El usuario presiona el botón No.	1.1- La aplicación conserva la estructura.
Prioridad	Crítico

Poscondiciones	Queda estructurado el proyecto de ayuda.

Tabla 6 Descripción del Caso de uso Gestionar Contenido de Ayuda

CUS 2	Gestionar Contenido de Ayuda.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el usuario decide editar la ayuda.
Referencias.	R2, R2.1, R2.2, R2.3, R2.4, R2.5, R2.6, R2.7, R2.8
Precondiciones	Debe haber al menos un tópico en el administrador de proyectos.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El caso de uso comienza cuando un usuario decide insertar contenido en la ayuda.	
2- El usuario desea realizar una de las siguientes opciones: • Insertar textos. • Insertar imagen. • Insertar tabla. • Insertar animación flash.	2.1 El sistema ejecuta alguna de las siguientes acciones: a) Si se decide “Insertar textos” ir a la sección Insertar textos. b) Si se decide “Insertar imagen” ir a la sección Insertar imagen. c) Si se decide “Insertar tabla” ir a la sección Insertar tabla. d) Si se decide “Insertar animación flash” ir a la sección Insertar animación flash.

Sección “Insertar textos”	
1- El usuario decide Insertar textos.	1.1- El sistema muestra un editor al dar doble clic sobre el tópico a editar.
2- El usuario introduce los textos.	
3- El usuario salva el contenido mediante el menú Archivo la opción guardar o guardar todo.	3.1- La aplicación verifica que los datos se hayan completado. 3.2- La aplicación comprueba si el nombre asignado al tema coincide con otro ya existente. 3.3- Guarda los datos. 3.4- Visualiza en el área de Proyectos, un nivel debajo del proyecto o tema, el tema creado.
Sección “Insertar imagen”	
1- El usuario decide insertar una imagen, lo mismo puede ser mediante el menú Insertar, la opción Insertar imagen o mediante el atajo Ctrol+Shift+I.	1.1- El sistema solicita el nombre y selección de la imagen a insertar, además de la opción de alto y ancho de la imagen.
2- El usuario introduce los datos requeridos (Nombre y Selección de la imagen).	
3- El usuario pulsa el botón aceptar.	3.1- La aplicación verifica que los datos se hayan completado. 3.2- La aplicación comprueba si el nombre asignado a la imagen coincide con otro ya existente. 3.3- Guarda los datos. 3.4- Visualiza en el editor la imagen insertada.
4.- El usuario pulsa el botón cancelar.	4.1- El editor conserva el estado inicial sin imagen.
Sección “Insertar tabla”	

1- El usuario decide insertar una tabla, lo mismo puede ser mediante el menú Insertar, Insertar tabla o mediante el atajo Ctrol+Shift+T.	1.1- El sistema muestra una ventana con un formulario solicitando las propiedades de la tabla.
2- El usuario introduce los datos requeridos.	
3- El usuario pulsa el botón aceptar.	3.1- La aplicación verifica que los datos se hayan completado. 3.2- Guarda los datos. 3.3- Visualiza en el editor la tabla insertada.
4.- El usuario pulsa el botón cancelar.	4.1- El editor conserva el estado inicial sin tabla.
Sección "Insertar animación flash"	
1- El usuario decide insertar una animación flash, lo mismo puede ser mediante el menú Insertar, Insertar animación flash o mediante el atajo Ctrol+Shift+F.	1.1- El sistema solicita el nombre y selección de la animación a insertar, además de la opción de alto y ancho de la animación.
2- El usuario introduce los datos requeridos (Nombre y Selección de la animación).	
3- El usuario pulsa el botón aceptar.	3.1- La aplicación verifica que los datos se hayan completado. 3.2- La aplicación comprueba si el nombre asignado a la animación coincide con otro ya existente. 3.3- Guarda los datos. 3.4- Visualiza en el editor la animación insertada.
4.- El usuario pulsa el botón cancelar.	4.1- El editor conserva el estado inicial sin animación.
Flujos Alternos	
	3.2. a- En caso de que coincida, la aplicación muestra un mensaje de error y vuelve al paso 2 del flujo normal de eventos.

Flujos Alternos- Sección "Insertar textos"	
Acción del Actor	**Respuesta del Sistema**
1.- El usuario no guarda los cambios y decide cerrar el editor.	1.1.- La aplicación muestra un mensaje de confirmación, advirtiendo que los cambios no han sido guardados, si el usuario desea guardarlos.
2.- El usuario presiona el botón Sí.	2.1- Se cierra el mensaje de confirmación, se guardan los cambios y se cierra el editor.
3.- El usuario presiona el botón No.	3.1- Se cierra el mensaje de confirmación, no se guardan los cambios y se cierra el editor.
Flujos Alternos-Sección "Insertar imagen"	
Acción del Actor	**Respuesta del Sistema**
	1.1- Si el nombre asignado a la imagen coincide con otro ya existente, el sistema muestra una interfaz con la imagen existente del mismo nombre.
2.- Si el usuario presiona el botón Sí.	2.1- El sistema sobrescribe la imagen y en lugar de esta coloca la actual.
3.- Si el usuario presiona el botón No.	3.1- Se cierra la interfaz y la imagen existente queda como estaba.
Prioridad	Crítico
Poscondiciones	Queda la ayuda con el contenido requerido.

Tabla 7 Descripción del Caso de uso Importar Proyecto

CUS 4	Importar Proyecto.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el usuario decide importar un proyecto de ayuda existente.
Referencias.	R5
Precondiciones	El proyecto a importar debe ser el mismo **.zip**, generado por el sistema.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El caso de uso comienza cuando un usuario decide importar un proyecto de ayuda, para lo cual accede al menú Archivo opción Importar proyecto o mediante el atajo Ctrol+Shift+O.	1.1- El sistema muestra una ventana con un campo de entrada para seleccionar el proyecto a importar.
2- El usuario pulsa el botón buscar y selecciona el proyecto a importar.	
3.- El usuario pulsa el botón aceptar.	3-1.- El sistema muestra un mensaje, indicando al usuario que el proyecto de ayuda se está importando. 3.2- Una vez terminado este proceso, la aplicación muestra en el área de proyectos el proyecto importado.
Flujos Alternos	
1.- El usuario pulsa el botón cancelar.	1.1 Se cierra la ventana, el área de proyectos conserva el estado inicial.
Prioridad	Crítico

Poscondiciones	Queda importado el proyecto de ayuda.

Anexo II "Descripción de las clases del diseño**"**

Tabla 8 Descripción de la clase "GeneradorAyuda"

Nombre: GeneradorAyuda	
Tipo de clase: controladora.	
Responsabilidades: Esta es la clase que se encarga de todo el proceso de gestión de datos, la estructura de los proyectos, la creación y eliminación de directorios, así como la transferencia de archivos.	
Nombre:	**crearDirectorio($source)**
Descripción:	Crea un directorio dado una ruta.
Nombre:	**conectaDb($sqlite)**
Descripción:	Realiza una conexión a la base de datos SQLite.
Nombre:	**ejecutarConsulta($bd, $consulta)**
Descripción:	Dada la conexión y la consulta, ejecuta una consulta a la BD.
Nombre:	**guadarNodosArbol($bd)**
Descripción:	Guarda los datos de un nodo en la BD.
Nombre:	**crearArregloHijos($id, $bd)**
Descripción:	Dado el id de un proyecto, crea el arreglo de hijos.
Nombre:	**eliminarNodo($idnodo, $bd)**
Descripción:	Dado el id de un nodo, y la conexión, llama a la función eliminarNodosTodos ($bd, $idnodo).
Nombre:	**eliminarNodoBd($bd, $idnodo)**
Descripción:	Dado el id de un nodo y la conexión, lo elimina de la BD.
Nombre:	**eliminarImagenesAsociadas($idnodo, $bd)**
Descripción:	Dado el id de un nodo y la conexión, elimina la imagen asociada a ese nodo.
Nombre:	**eliminarImagenPorId($idimagen, $bd)**
Descripción:	Dado el id de una imagen la elimina de la BD.
Nombre:	**eliminaImagenPorNodo($idnodo, $bd)**
Descripción:	Elimina la imagen de un nodo dado el id.
Nombre:	**buscarCantidadUsosImagen($idimagen, $bd)**
Descripción:	Devuelve la cantidad de usos de una imagen dado su id.
Nombre:	**buscarIdProyectoNodo($idnodo, $bd)**

Descripción:	Devuelve el id de un proyecto dado el id de un nodo.
Nombre:	**buscarHtmlNodo($idnodo, $bd)**
Descripción:	Dado el id de un nodo y la conexión, retorna el contenido HTML de dicho nodo.
Nombre:	**cargarContenidoHtml($idnodo, $bd)**
Descripción:	Dado el id de un nodo y la conexión, le envía a la interfaz el contenido HTML de dicho nodo.
Nombre:	**salvarContenidoHtml($idnodo, $html, $bd)**
Descripción:	Dado el id, el contenido HTML y la conexión, actualiza el contenido de dicho nodo.
Nombre:	**buscarHijos($idpadre, $bd)**
Descripción:	Dado el id de un nodo padre y la conexión, devuelve el arreglo de hijos de dicho nodo.
Nombre:	**buscarImagenesPorNodo($idnodo, $bd)**
Descripción:	Dado el id de un nodo y la conexión, devuelve las imágenes de dicho nodo.
Nombre:	**eliminarImagenesSobrantes($idnodo, $html, $bd)**
Descripción:	Dado el id de un nodo, el contenido HTML y la conexión, elimina las imágenes que no están siendo utilizadas.
Nombre:	**cantidadUsosPorNodo($id, $idnodo, $bd)**
Descripción:	Dado un id, el id del nodo y la conexión, devuelve la cantidad de veces que una imagen con id igual al dado esta siendo utilizada en el nodo.
Nombre:	**comprobarEliminarImagen($id, $idnodo, $bd)**
Descripción:	Dado un id, el id del nodo y la conexión, elimina la imagen con id igual al dado y no este siendo utilizada.
Nombre:	**importarProyecto()**
Descripción:	Si existe un fichero .zip, extrae del archivo el fichero de la BD, coge los datos y las imágenes y los copia en el fichero de la BD real, y luego elimina el directorio temporal.
Nombre:	**importarDatosSqlite()**
Descripción:	Coge los datos de la BD que se crea temporal y los pasa para la BD real.
Nombre:	**importarImagenesSqlite()**
Descripción:	Coge las imágenes de la BD que se crea temporal y los pasa para la BD real.
Nombre:	**guardarImagenEnBdatos($bd)**
Descripción:	Dada la conexión, guarda la imagen en la BD.
Nombre:	**comprobarSiExiste($bd)**
Descripción:	Pasa una imagen subida al directorio temporal, quien al cerrarse el navegador será eliminada.
Nombre:	**generarZip($id)**
Descripción:	Crea un directorio con la ruta especificada y luego genera un .zip con ese directorio.
Nombre:	**basedatosBackup($bd)**
Descripción:	Devuelve un arreglo con todos los datos de la BD.
Nombre:	**eliminarValoresBd($id, $bd)**
Descripción:	Elimina todos los valores de la BD, donde el id del proyecto sea distinto del id dado.
Nombre:	**restaurarSqlite($datos, $bd)**
Descripción:	Restaura la BD con los nuevos datos, al generar un proyecto.

Nombre:	**incrementarUsoImagen($idimagen, $idnodo, $bd)**
Descripción:	Incrementa el uso de una imagen al ser utilizada.
Nombre:	**eliminarArchivosTemporales()**
Descripción:	Elimina un directorio y todo lo que este contiene y lo vuelve a crear pero vacio.

Tabla 9 Descripción de la clase "Help"

Nombre: Help	
Tipo de clase: acceso a datos.	
Responsabilidades: Esta es la clase que se encarga de crear el árbol de índices de la ayuda y realizar la búsqueda global sobre la base de datos.	
Nombre:	**crearArregloArbol($id)**
Descripción:	Dado el id de un proyecto de ayuda, devuelve un arreglo con la estructura arbórea de dicho proyecto.
Nombre:	**conectaDb()**
Descripción:	Realiza una conexión a una base de datos SQLite.
Nombre:	**ejecutarConsulta($bd, $consulta)**
Descripción:	Dada la conexión y la consulta, ejecuta una consulta a la BD.

Anexo III "Diagramas de secuencia**"**

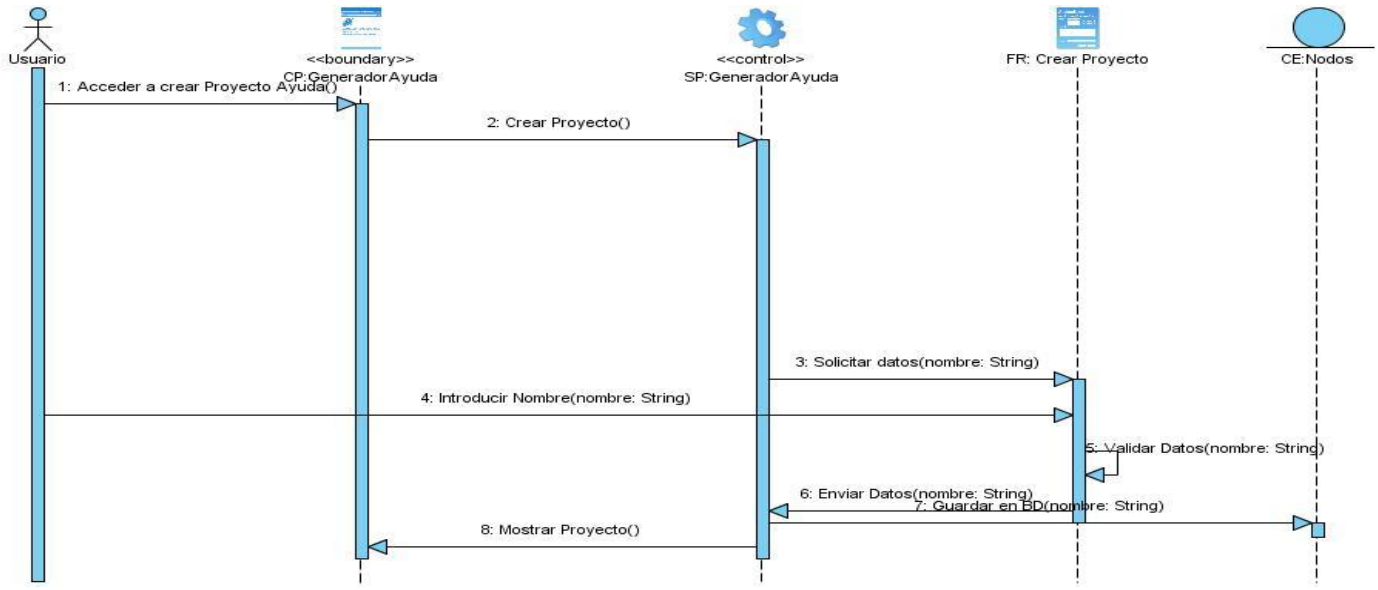


Figura 21 Diagrama de Secuencia CU: Gestionar Proyecto de Ayuda Escenario "Crear Proyecto"

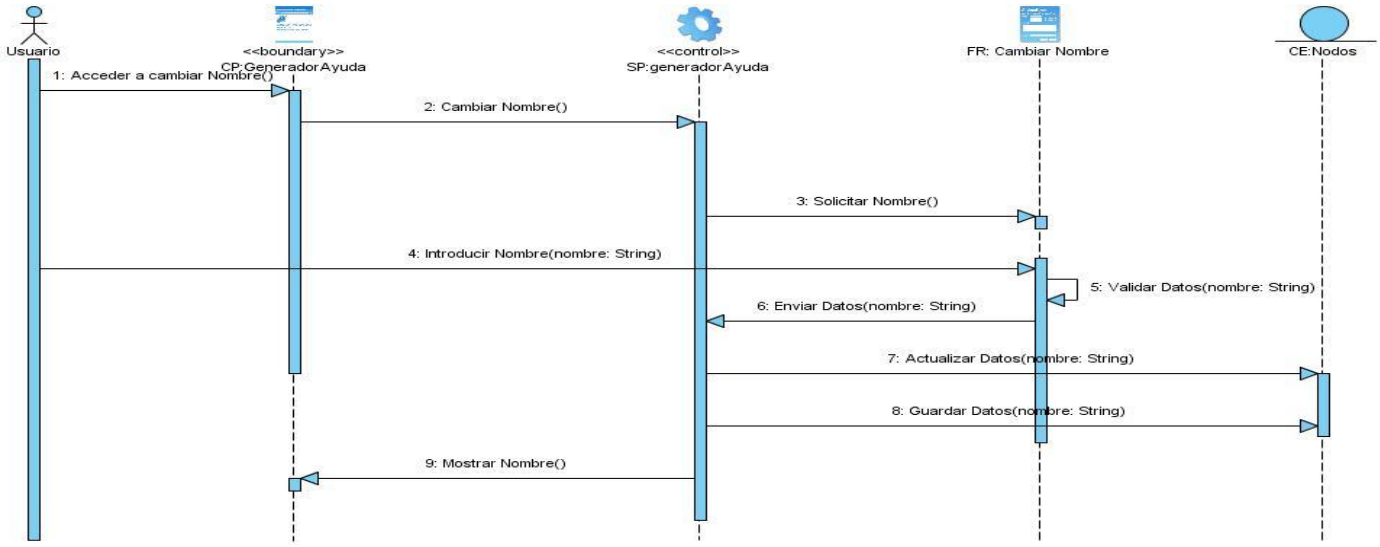


Figura 22 Diagrama de Secuencia CU: Gestionar Proyecto de Ayuda Escenario "Cambiar Nombre"

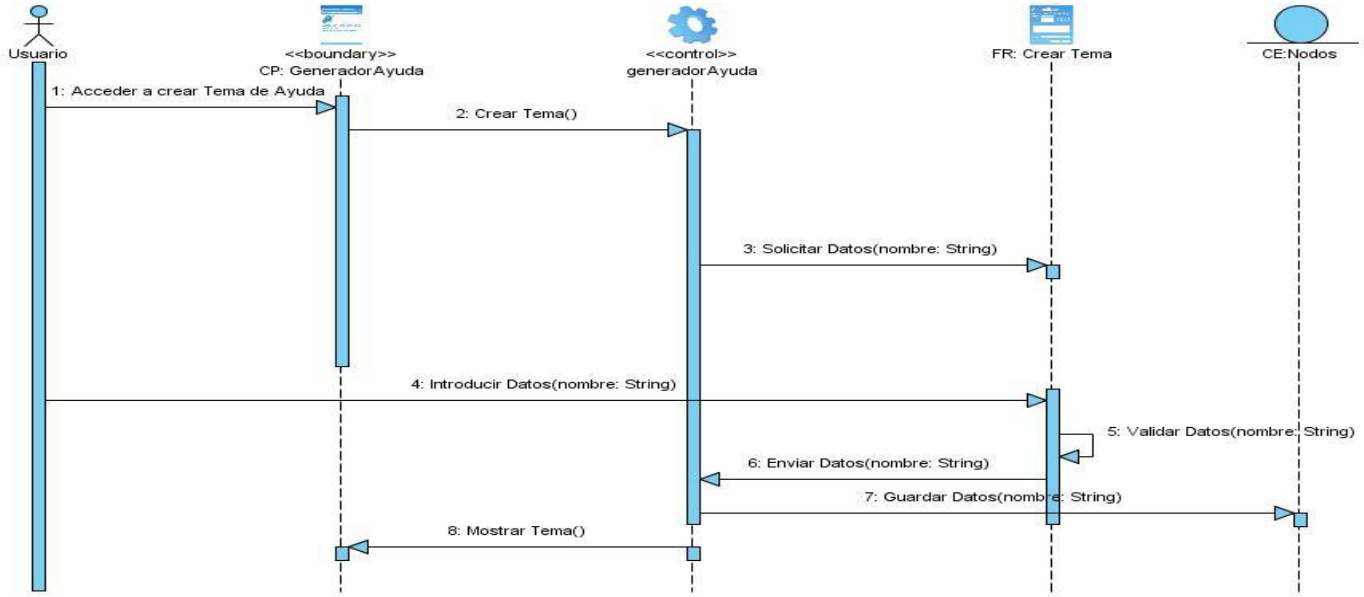


Figura 23 Diagrama de Secuencia CU: Gestionar Proyecto de Ayuda Escenario "Crear Tema"

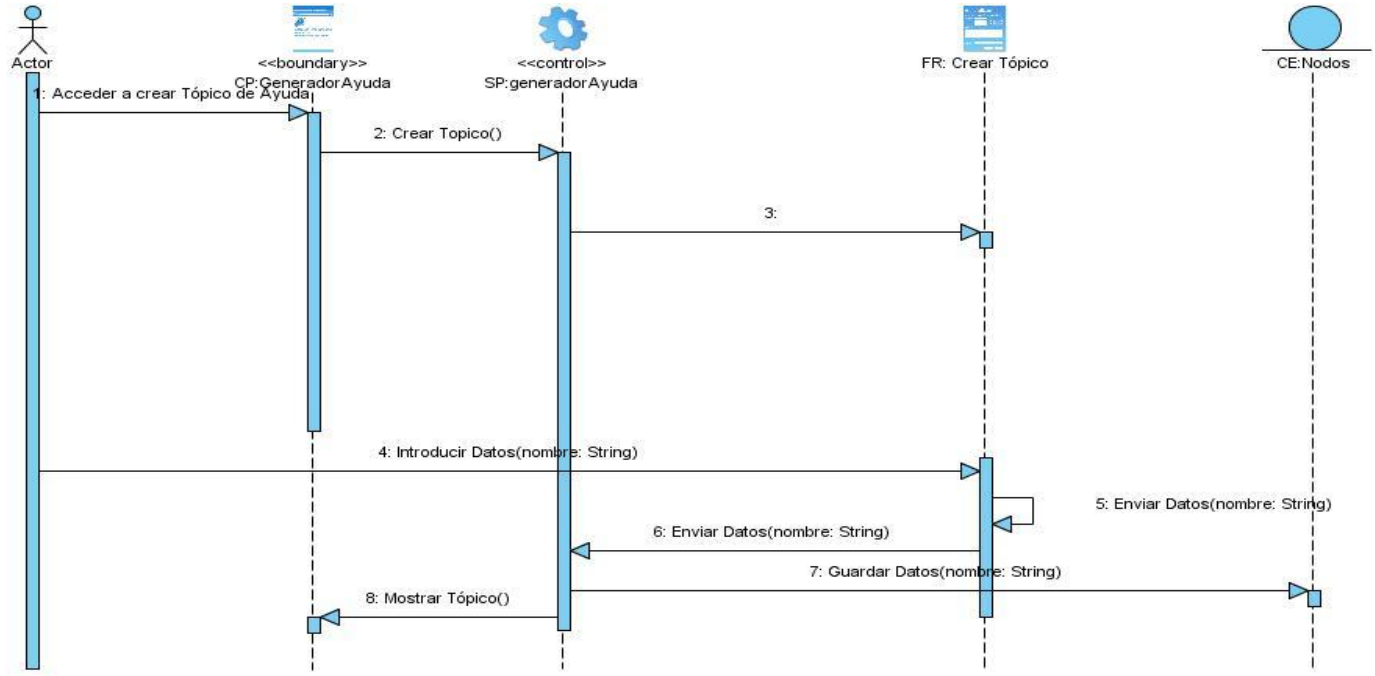


Figura 24 Diagrama de Secuencia CU: Gestionar Proyecto de Ayuda Escenario "Crear Tópico"

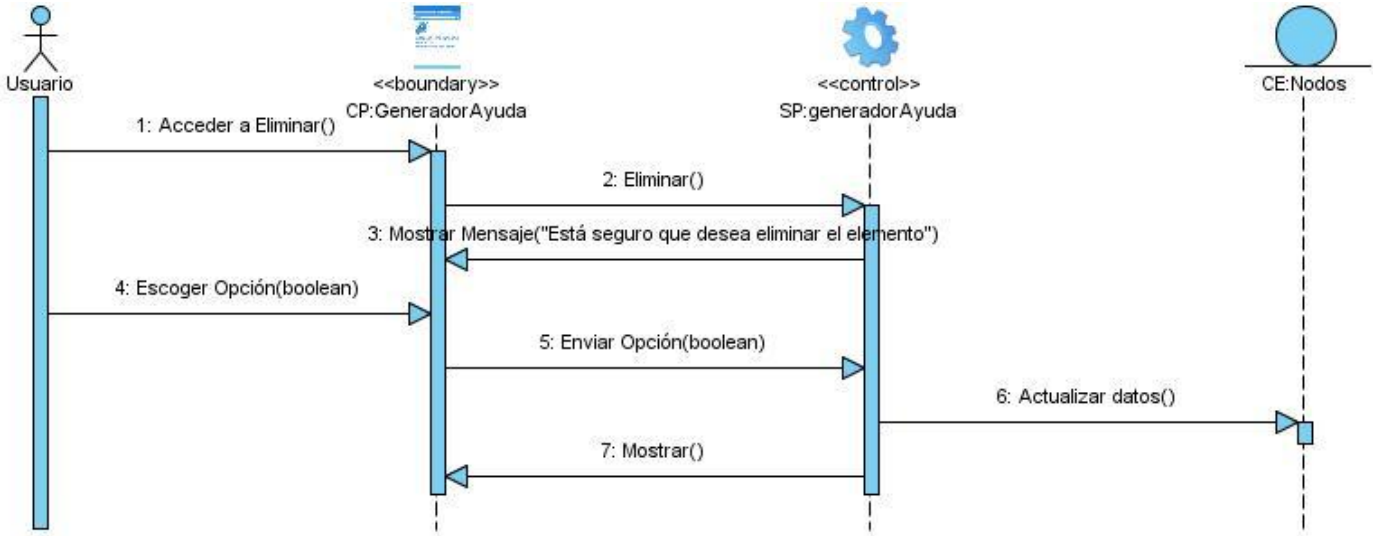


Figura 25 Diagrama de Secuencia CU: Gestionar Proyecto de Ayuda Escenario "Eliminar"

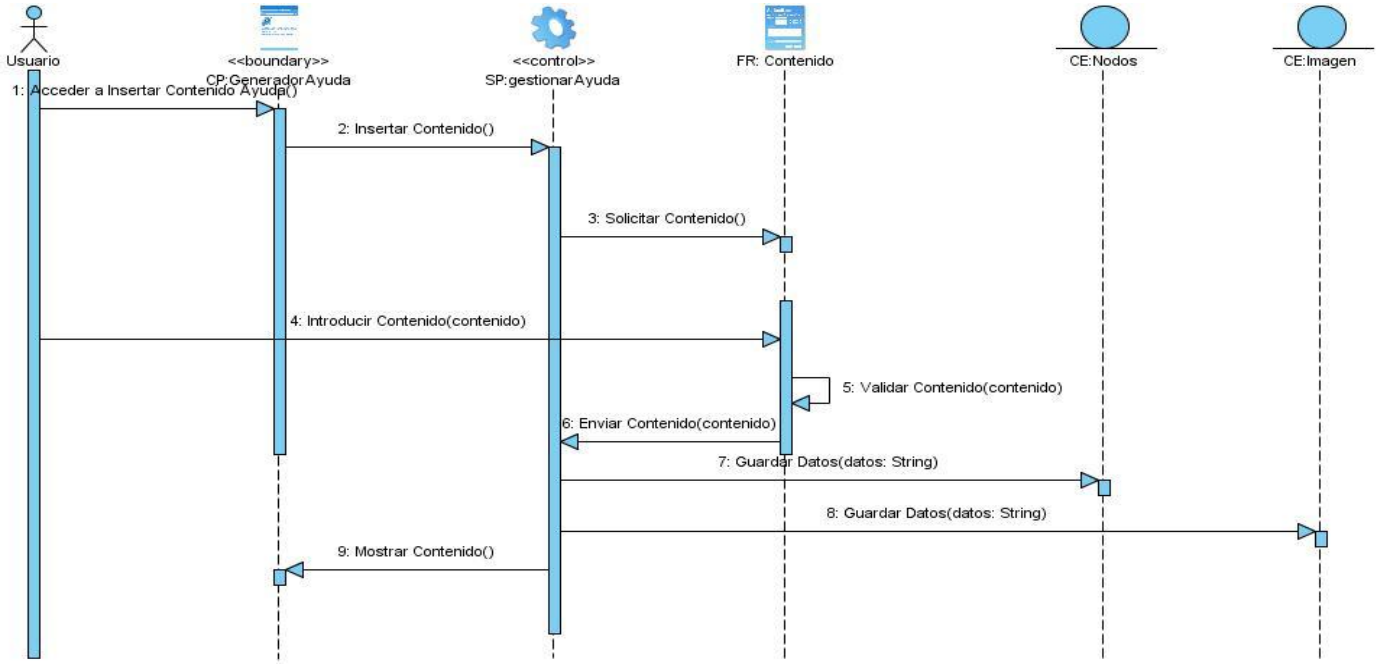


Figura 26 Diagrama de Secuencia CU: Gestionar Contenido de Ayuda Escenario "Insertar Contenido"

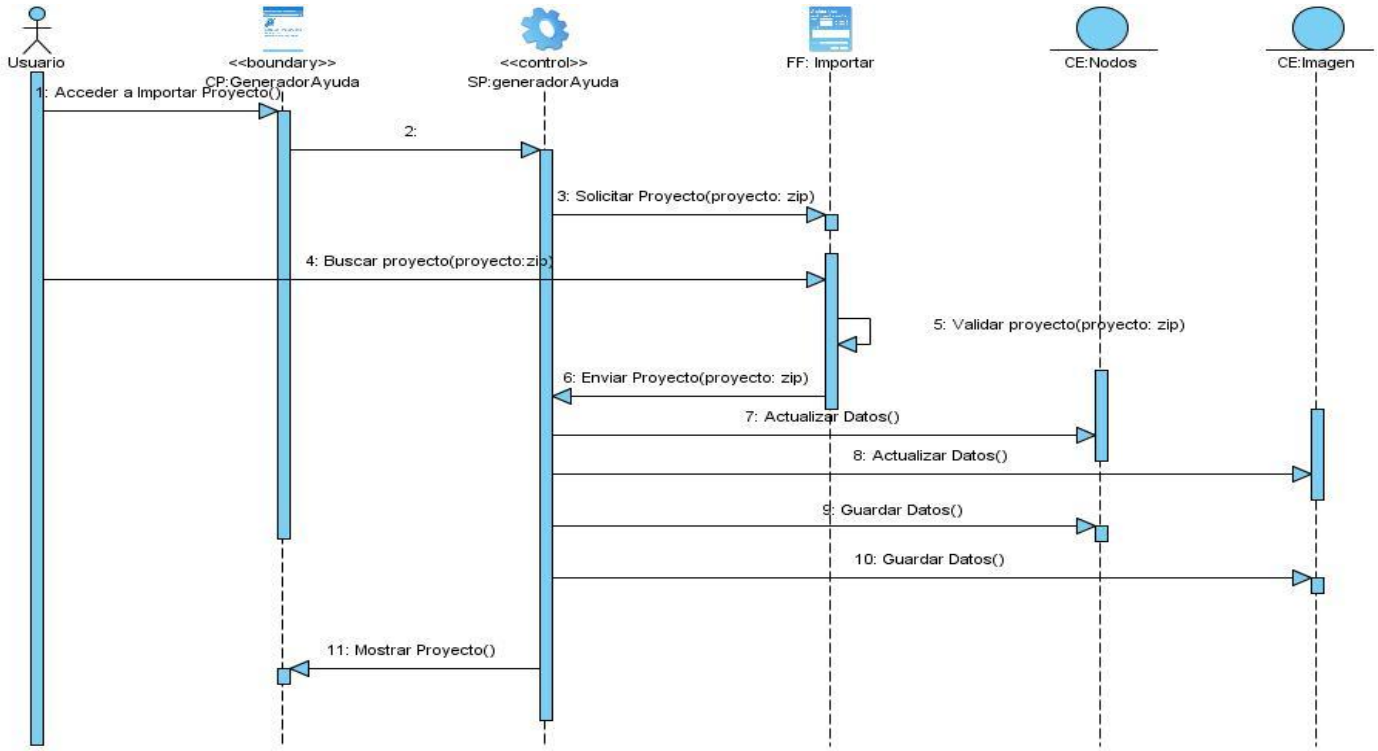


Figura 27 Diagrama de Secuencia CU: Importar Proyecto

Glosario de términos

API: Interfaz de programación de aplicaciones o API (del inglés application programming interface) es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

CGI: Common Gateway Interface, una tecnología que se usa en los servidores web.

DATEC: Centro de tecnologías y análisis de datos.

DOM: Document Object Model (Modelo en Objetos para la representación de Documentos), provee un conjunto estándar de objetos para representar documentos HTML y XML, permitiendo su combinación y ofreciendo una interfaz estándar para el acceso y manipulación.

ERP: (Enterprise Resource Planning en inglés), Planificación de Recursos Empresariales.

Framework: Es una estructura de soporte definida, representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

GPL: Es una licencia pública general de GNU o más conocida por su nombre en inglés General Public License.

IDE: Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.

UCI: Universidad de las Ciencias Informáticas.

UCID: Unidad de Compatibilización Integración y Desarrollo.

World Wide Web: En informática World Wide Web (o la "Web") o Red Global Mundial es un sistema de documentos de hipertexto accesibles a través de Internet.