Universidad de las Ciencias Informáticas

Facultad 4

Desarrollo del módulo General de la Colección El Navegante

Trabajo para optar por el título de Ingeniero en Ciencias Informáticas

Autores: Yadima Morales Hernández

Joel Tamayo Vega

Tutores: Ing. Mailyn Cabrera Torres

Ing. José Ernesto Lara Rodríguez

Asesor: Ing. Jorge Martínez Padrón

“Año 53 de la Revolución”

La Habana, junio 2011

Declaración de autoría

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

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2011.

_____ _____

Yadima Morales Hernández — Joel Tamayo Vega

Firma del Autor — Firma del Autor

_____ _____

Ing. Mailyn Cabrera Torres — Ing. José Ernesto Lara Rodríguez

Firma del Tutor — Firma del Tutor

Agradecimientos

De Yadima

A mis padres, por hacerme sentir tan orgullosa de lo unidos que hemos sido siempre. Sin ustedes no hubiera podido lograr todo lo que he sido hasta hoy. Son lo más importante para mí.

A mi novio, mi Cosi a ti te tengo que agradecer muchísimo por ser como eres, por tener paciencia en estos días tan difíciles para mí. Por siempre querer que yo sea mejor cada día, por dedicarme tanto amor, por estar siempre a mi lado y hacerme la mujer más feliz. Hoy no estaría escribiendo si no fuera por ti. Lo eres todo para mí. Te amo mi niño.

A Nenita, mi hermana por compartir alegrías, tristezas y siempre ser mi ejemplo a seguir.

A mi hermano por ese corazón tan grande y tan bueno que tiene.

A Disnaidis, siempre serás mi mejor amiga.

A Yansielito, Manolo y Joel (cuñi), mis hermanos de sentimientos.

A mi familia en Pinar que siempre me han apoyado en todo y han sido incondicionales conmigo.

A todos mis familiares por aportar un granito de arena a que yo cumpliera mis sueños.

A mi compañero de tesis, Joel, por ser optimista siempre y no dejar que las cosas me angustiaran tanto.

A todas las personas que en el transcurso de estos 5 cursos han hecho por mí y que saben que las estimo, un millón de gracias.

A la Revolución, fundamentalmente a Fidel por haber creado esta escuela permitiendo hacer realidad mis sueños.

Agradecimientos

De Joel

A mis padres que son las personas que me han dado todo lo que tengo y gracias a ellos hoy puedo cumplir este sueño y **A mi hermanita** que adoro y me ha apoyado siempre.

A toda mi familia que me ha visto crecer y me ha ayudado en todo y especialmente mi abuelita Hilda por todo y aunque no te encuentras hoy se que estás muy feliz por mí.

A mi novia y casi esposa Irena, Piti gracias por todo el amor que me has entregado, por todo el apoyo que me has dado desde que nos conocimos gracias a ti puedo hacer realidad este sueño, vivo muy orgulloso de ti y gracias por darme esa niña hermosa que es lo más bello que tengo en vida.

A mis amigos fuertes de la carrera, Mario, Carlos, Ancel, Luna, Antonio que me han apoyado a lo largo de la carrera, y a **A mis amigos de siempre** Sandor Vargas, José Eduardo, y Félix gracias.

A mi amigo Alejandro Santana que ha sido para mí el hermano que no tuve, gracias por todo el apoyo que me has dado desde que nos conocimos.

A todos mis compañeros de cuarto desde primer año y hasta quinto, a mi grupo 8101 y mi actual 4505 los que estuvieron conmigo en Venezuela, a todos los que siempre han estado cerca dando apoyo.

A mi compañera de Tesis Yadima por todo el apoyo que me ha dado durante todo el desarrollo de la tesis, y todos los sustos que pasó para hacer realidad este sueño.

A mis tutores Mailyn y Lara por todo el apoyo que me dieron durante el desarrollo de la tesis.

A los compañeros que me ayudaron en el desarrollo de la Tesis, mis muchachos de 4to año, Ernesto de la Soledad, Ernesto Yordi, Yolanda Mauri, y al Ing. Yaismel Miranda Poms, muchas gracias por todo.

A la familia de Manzanillo que me han apoyado y aguantado siempre y en especial al viejo Nené.

A todos los que de una forma u otra me han apoyado a lo largo de la carrera y la vida, muchas gracias.

Dedicatoria

De Yadima

A la memoria de **mis abuelos** que siempre tendrán un pedacito dentro de mí.

A mi mamá, por todo su sacrificio y dedicación, por escucharme siempre y saber comprenderme, gracias por ser la mejor madre del mundo y por toda tu confianza. Te adoro.

A mi papá, por estar siempre presente cuando lo necesito, y por poner tanta confianza en mis decisiones, gracias por estar tan orgulloso de mí. Te quiero mucho.

A mi novio por hacer realidad este sueño. Gracias por dejar acompañarte en estos cuatro años.

De Joel

A mis padres por todo el amor aprecio y cariño que me han dado, son mi orgullo.

A mi niña hermosa Isabellita.

Resumen

El desarrollo de software educativo en Cuba se ha incrementado en los últimos años debido a los beneficios que estos ofrecen en la educación. Varias son las colecciones de este tipo de software desarrolladas por el Ministerio de la Educación Cubana, siendo El Navegante, colección destinada a las escuelas secundarias, un ejemplo de ellas.

Como resultado de las relaciones existentes entre Cuba y Venezuela se firma en diciembre del 2010 un contrato entre ambos países para llevar a cabo el proyecto “Colecciones de Software Educativo Multisaber y El Navegante”. Uno de los objetivos de este nuevo compromiso con Venezuela era la entrega de los 10 productos o software que integran la Colección El Navegante, pero esta vez en una versión multiplataforma. Cada software está dividido en 6 módulos específicos que garantizan una mejor organización de los contenidos.

El presente trabajo tiene como objetivo desarrollar el módulo General de la Colección El Navegante, el cual agrupa los servicios comunes que pueden ser accedidos por el usuario desde los restantes módulos de la nueva versión del software. Para guiar el desarrollo del módulo se utilizó Rational Unified Process (RUP) como metodología de desarrollo de software, PHP y JavaScript como lenguajes de programación y Symphony como framework de desarrollo del lado del servidor. Se realizó el análisis, diseño, implementación y prueba; obteniéndose en cada caso los artefactos fundamentales que propone la metodología para cada etapa de trabajo.

Se obtuvo el módulo General, logrando con ello que a los productos puedan acceder los usuarios con permisos, permite manipular la mascota de la colección, ver las efemérides y noticias publicadas, ver la ayuda del software así como la información general de la Colección El Navegante.

Palabras clave: software educativo, Colección El Navegante, módulo General.

Índice

Introducción

Las Tecnologías de la Información y las Comunicaciones (TIC) han sido introducidas en el sistema educativo cubano como parte de las transformaciones que se llevan a cabo en nuestra sociedad. El Ministerio de Educación (MINED), en colaboración con otras instituciones ha desarrollado grupos docentes informáticos con el objetivo de elaborar software educativo que apoye el proceso de enseñanza-aprendizaje, encontrando un equilibrio entre los componentes instructivos y educativos.

En Cuba se han creado tres importantes colecciones de software educativos: Colección Multisaber para la enseñanza primaria, Colección El Navegante distribuida en la enseñanza secundaria y Colección Futuro para la educación superior. Estos productos han proporcionado nuevos conocimientos a nuestros niños y jóvenes mediante su interacción con las tecnologías.

Teniendo en cuenta la experiencia y el prestigio que ha alcanzado la educación cubana y en el marco de las relaciones existentes con la República Bolivariana de Venezuela, surge la idea de realizar varios software educativos destinados a fomentar el aprendizaje y creatividad de los jóvenes venezolanos. Esta tarea ha sido asignada a la Universidad de las Ciencias Informáticas (UCI).

El Ministerio del Poder Popular para la Educación de Venezuela se ha interesado en particular por dos colecciones cubanas de las antes mencionadas: Multisaber y El Navegante. Debido a que en ese país se promueve el uso prioritario del software libre en la administración pública nacional según el Decreto 3390 del 23 de diciembre del 2004, se ha orientado la tarea de migrar a software libre ambas colecciones ya que fueron realizadas con herramientas propietarias y solo se ejecutan en el sistema operativo Windows. El proyecto productivo Colecciones de Software Educativo Multisaber y El Navegante perteneciente al Centro de Tecnologías para la Formación (FORTES), de la Facultad 4, es el responsable de la migración de ambas colecciones a software libre y de la implementación de las nuevas necesidades identificadas por los usuarios venezolanos.

La Colección El Navegante está compuesta por 10 software educativos que integran algunos de los contenidos del nivel secundario, con un carácter curricular extensivo, lo que significa que el producto constituye un soporte informático para el proceso docente de este nivel. Esta colección en su primera versión carecía de varias funcionalidades, por lo que en enero del 2010 se comienza a trabajar sobre los

diferentes módulos de la misma: Contenido, Ejercicios, Mediateca, Juegos, Maestro y Resultados; sin embargo aún no satisface las necesidades del cliente, debido fundamentalmente a que no se ha logrado diferenciar el acceso de los usuarios al sistema, no brinda ayuda al usuario de cómo interactuar con el software, no se permite la impresión y búsqueda de contenidos. Por otra parte, no permite manipular la mascota mostrada en la colección, ni la consulta al usuario de los datos con los que fue creado su perfil y consultar la información referente a los créditos del producto, a sus patrocinadores y a las licencias bajo las que se libera el mismo. Todas estas funcionalidades forman parte de los servicios comunes que son agrupados en el módulo General de la Colección El Navegante.

A partir de la situación descrita surge como **problema a resolver** en esta investigación la siguiente interrogante: ¿Cómo lograr que el módulo General esté presente en las áreas de trabajo de los diferentes grupos de usuarios?

De la interrogante anterior se decide estudiar las aplicaciones educativas con tecnologías multimedia.

Se plantea como **objetivo general** de la investigación desarrollar el módulo General de la Colección El Navegante.

El **campo de acción** queda enmarcado en el módulo General de la Colección El Navegante.

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

- Realizar el estudio del estado del arte.
- Realizar el análisis del módulo General de la Colección El Navegante.
- Diseñar el módulo General de la Colección El Navegante.
- Implementar el módulo General de la Colección El Navegante.
- Probar el módulo General de la Colección El Navegante.

Las tareas que se proponen para dar solución a los objetivos anteriormente mencionados son:

- Elaboración del marco teórico de la investigación.
- Realización del Modelo del Sistema.
- Realización de los diagramas de clases del análisis del módulo General de la Colección El Navegante.
- Estudio y selección de los patrones de diseño.
- Realización de los diagramas de clases del diseño del módulo General de la Colección El Navegante.

- Realización de los diagramas de implementación del módulo General de la Colección El Navegante.
- Implementación de los componentes del módulo General de la Colección El Navegante.
- Realización de las pruebas al módulo General de la Colección El Navegante.
- Documentación de la investigación.

Se desea obtener el módulo General de la Colección El Navegante y que el mismo sea accesible desde los restantes módulos de la colección.

La presente investigación, para un mejor entendimiento, ha sido estructurada en cuatro capítulos, los cuales se mencionan a continuación.

Capítulo 1. "Fundamentación Teórica" se realiza un estudio de los principales elementos teóricos que constituyen la base de la investigación entre los cuales se encuentran las metodologías de desarrollo de software, las herramientas y lenguajes de programación utilizados en la propuesta de solución. Se justifican las opciones seleccionadas.

Capítulo 2. "Características del Sistema" se describe el Modelo de Dominio donde se capturan los objetos importantes del entorno donde será empleado el sistema. Se identifican los requerimientos funcionales y los no funcionales, se elabora el diagrama de casos de uso y las descripciones de cada uno de ellos.

Capítulo 3 "Construcción de la solución" se describe la solución que se propone a partir de los diagramas de clases del análisis asociado a las funcionalidades del sistema y se representan los diagramas de clases del diseño, que reflejan una vista interna del sistema.

Capítulo 4 "Implementación y Prueba" se realiza el diagrama de componentes y de despliegue, obteniéndose una descripción de la implementación del sistema. Se describen los casos de pruebas por los cuales los desarrolladores prueban el producto. Se selecciona el método de prueba y la técnica de prueba que se va a utilizar.

Capítulo 1 Fundamentación teórica

Al pasar de los años y con el avance de las tecnologías, ha evolucionado notablemente el proceso de enseñanza-aprendizaje, introduciendo en las aulas medios informáticos nombrados software educativo, los cuales han logrado elevar el porciento de entendimiento, rendimiento y aprendizaje de los estudiantes.

1.1 Conceptos relacionados al desarrollo de software educativo

Software educativo

El software educativo es un producto tecnológico diseñado para apoyar procesos educativos, dentro de los cuales se concibe como uno de los medios que utiliza quien enseña y quien aprende para alcanzar determinados propósitos. Además, este software es un medio de presentación y desarrollo de contenidos educativos, como lo puede ser un libro o un video, formato expresivo y secuencia narrativa. De esta manera, el software educativo puede ser visto como un producto y también como un medio. (1)

Estos programas pueden agrupar varias áreas curriculares como las matemáticas, ciencias naturales e idiomas. Se caracterizan por ser interactivos, a partir de que se le añaden programas de entretenimientos y recursos multimedia: videos, sonidos, fotografías, ejercicios, diaporamas y juegos instructivos, que apoyan la evaluación y el diagnóstico del estudiante. (1)

Existen aspectos comunes que distinguen al software educativo: (2)

- Son materiales desarrollados con finalidad didáctica.
- Utilizan la computadora como medio para realizar sus actividades.
- Son interactivos, ya que dan respuestas inmediatas a las acciones de los estudiantes.
- Permiten el intercambio de información entre la computadora y el estudiante.
- Se adaptan al ritmo de trabajo de cada estudiante, mostrándole ejercicios según las habilidades demostradas.
- Son fáciles de usar, no se requiere conocimientos avanzados en informática para su uso.

A pesar que los software comparten características esenciales, son diferentes en cuanto a su finalidad, es decir, algunos aparentan ser un laboratorio o bibliotecas, otros aparentan ser libros o juegos. Para darle orden a estos aspectos se han elaborado tipologías que permiten clasificarlos según el tipo de software: (3)

Software algorítmico: en este tipo de software el aprendizaje se da por medio de la transmisión del conocimiento.

- Sistema tutoriales: basado en el diálogo transmiten los conocimientos. Se tienen en cuenta las características de los estudiantes.
- Sistemas entrenadores: el objetivo es contribuir al desarrollo de una habilidad intelectual o manual, profundizando en la aplicación y retroalimentación.
- Libros electrónicos: el propósito es presentar información a los estudiantes a partir del uso de textos, gráficos, animaciones, videos, sonidos, creando motivación y facilitándole las acciones que realizan.

Software Heurístico: en este tipo de software el alumno interactúa con situaciones que permiten crear un ambiente de aprendizaje interactivo que le permita llegar a adquirir los conocimientos de manera funcional.

- Simuladores: apoya el proceso de enseñanza-aprendizaje, simulando la vida real.
- Juegos educativos: el propósito es crear situaciones entretenidas, sin dejar de simular la realidad.
- Sistemas de expertos: el objetivo es resolver problemas que normalmente realiza un experto. Dar respuestas y explicar sus razonamientos.
- Sistemas tutoriales de enseñanza: detectan errores y pueden explicar el por qué se producen. Favorecen la retroalimentación de los estudiantes.

El software educativo, aplicado a la realidad educativa, realiza funciones básicas de los medios didácticos y en algunos casos, según la forma de uso que determina el profesor puede proporcionar funcionalidades específicas.

Multimedia

"La multimedia es la tecnología que combina distintas medias: imagen (fotografía, ilustración, animación o video), sonido (voz, música o efectos sonoros) y el texto, bajo la gestión de uno o más programas informáticos (software)". (4)

Es utilizada en varias áreas: educación, arte, medicina, matemáticas e investigaciones científicas. Pueden incluir simulaciones, programas de aprendizaje, presentaciones, informaciones y juegos. Dentro de los materiales multimedia se encuentran los materiales con propósitos educativos los cuales son denominados software educativo con tecnologías multimedia. Estos materiales se han convertido en un factor importante dentro del sistema educacional y su uso se hace cada vez mayor en los niveles de enseñanza, ya que han sido diseñados como medios didácticos que facilitan el proceso de enseñanza-aprendizaje. Las principales funciones que pueden realizar los recursos educativos multimedia son las siguientes: informativa, instructiva o entrenadora, motivadora, evaluadora, entorno para la exploración y la experimentación, innovadora, apoyo a las orientación escolar y profesional, apoyo a la organización y gestión de centros. (5)

Informativa: a través de sus actividades, presentan contenidos que proporcionan información a los estudiantes.

Instructiva o entrenadora: los materiales didácticos multimedia, orientan y regulan el aprendizaje de los estudiantes. Además, mediante la estructuración de la información, condicionan los procesos de aprendizaje.

Motivadora: la interacción con la computadora puede resultar motivadora. Algunos programas incluyen elementos para captar la atención de los alumnos, mantener su interés y centrarlos hacia los aspectos más importantes.

Evaluadora: la posibilidad de retroalimentación inmediata a las respuestas y acciones de los alumnos, hace adecuados a los programas para evaluarles. Esta evaluación puede ser: implícita, el estudiante detecta sus errores, se evalúa a partir de las respuestas que le da el ordenador o explícita, el programa presenta informes valorando la actuación del alumno.

Metalingüística: al usar los recursos multimedia, los estudiantes también aprenden los lenguajes propios de la informática.

Innovadora: aunque no siempre sus planteamientos pedagógicos sean innovadores, los programas educativos pueden desempeñar esta función ya que utilizan una tecnología actual y, en general, suelen permitir muy diversas formas de uso. Esta versatilidad abre amplias posibilidades de experimentación didáctica e innovación educativa en el aula.

La multimedia como material educativo forma parte de las secuencias de aprendizaje y a la vez son objetos de aprendizaje. Hace posible que se integren los diversos medios de comunicación que tienen las personas para transmitir un mensaje, esto hace que el proceso de enseñanza-aprendizaje sea más atractivo e interactivo.

Hipertexto e Hipermedia

El hipertexto se caracteriza por tener los siguientes elementos: nodos y enlaces. Los nodos o secciones conforman las partes del hipertexto que contienen la información que es accedida por los usuarios. Los enlaces o hiperenlaces permiten navegar entre los distintos nodos de una manera no lineal. Si el usuario selecciona un hiperenlace del software, este muestra el documento enlazado a este vínculo. (6)

Por la gran cantidad de información organizada en distintas secciones, las cuales pueden estar relacionadas entre sí y ser lo que los usuarios necesitan, incluyen ayudas y documentación, diccionarios y enciclopedias electrónicas, las cuales pueden ser implementadas en ambientes cerrados o abiertos.

La hipermedia surge como resultado de la combinación de dos tecnologías, el hipertexto y la multimedia. El hipertexto es la organización de una determinada información en diferentes nodos, conectados entre sí a través de enlaces. La tecnología multimedia es la que permite integrar diferentes medios como sonido, imágenes y textos, en una misma presentación. La hipermedia, por tanto, es la tecnología que permite estructurar la información de una manera no lineal, a través de nodos interconectados por enlaces. La información presentada en estos nodos podrá integrar diferentes medios tales como: texto, sonido, gráficos y video. (7)

Una hipermedia es un sistema que provee al usuario una forma libre de acceder y explorar la información realizando saltos entre un documento y otro. (7)

Hiperentornos educativos

Los hiperentornos educativos o hiperentornos de aprendizajes surgen de integrar todas o algunas de las tipologías de software educativos: libros electrónicos, simuladores, juegos educativos y sistemas de expertos, todos estos basados en tecnologías hipermedia. Son técnicas que ayudan al sistema educacional a evaluar a los estudiantes mediante las tecnologías informáticas. (8)

1.2 Análisis de software similares

Colección Multisaber y El Navegante para las escuelas cubanas

La Colección Multisaber y El Navegante se encuentran actualmente en las escuelas cubanas de nivel primario y secundario respectivamente, están desarrolladas sobre el sistema operativo Windows XP. Fueron implementadas con herramientas propietarias como Macromedia Flash 5, Macromedia Director, Toolbook y Borland Delphi. Están constituidas por la concepción pedagógica que se dio a conocer como hiperentornos educativos en la que se integran los módulos: Temas, Ejercicios, Juegos, Biblioteca y un Registro o Traza. Contemplan una interfaz estandarizada, que proporcionan un ambiente de trabajo amigable e intuitivo con alto nivel de interactividad para acceder a la información existente en el software. Las colecciones poseen diversos servicios comunes como son la búsqueda y selección de información para realizar la impresión, además de integrarse con diferentes componentes que se identifican en la estructura didáctica de este tipo de programas:

- Libros electrónicos: los cuales constituyen la base de conocimientos.
- Entrenador: los cuales constituidos por cuestionarios interactivos incluyen varios ejercicios y preguntas que contemplan un sistema de retroalimentación.
- Componente lúdico: compuesto por varios juegos interactivos.
- Registro de resultados individuales.
- Sección para el maestro.

La Colección Multisaber posee una presentación genérica y una particular de cada producto. La idea general de la presentación de la colección muestra el título de la colección que metafóricamente expresa la idea del aprendizaje de múltiples contenidos que contribuyen a satisfacer la avidez de saberes que se manifiesta en los niños y niñas desde edades tempranas. (9)

La Colección El Navegante compuesta por 10 software educativos posee una presentación elaborada por el programa de diseño 3D Studio. Muestra la mascota representado por un robot llamado “navegante”, concebido mediante una moderna tecnología denominada Agentes de Microsoft. Entre las materias que aborda el software son Astronomía, Biología, Contabilidad, Economía, Medio Ambiente, Educación Especial, Electrónica, Física, Geografía, Historia y Matemática.

Ambas colecciones se estudiaron como referencia para desarrollar con tecnología libre un software con características similares y que además incluyera las nuevas funcionalidades identificadas por los usuarios venezolanos.

1.3 Colección El Navegante

La Colección El Navegante es considerada un hiperentorno de aprendizaje, debido a que combina distintas tipologías de software educativo como: juegos educativos, simuladores, libros electrónicos y sistemas entrenadores. Además, este software integra módulos propios de un hiperentorno como: Temas, Ejercicios, Juegos y un Registro o Traza, este último se ve reflejado en el módulo Maestro y Resultado respectivamente.

Crucigramas, Ludo, Sopa de letras y Descubre la imagen son algunos de los juegos presentes en la colección, todos estos juegos son educativos pues combinan contenidos de Matemáticas, Literatura y Español, Inglés, Educación Laboral y Dibujo Básico, Historia Antigua y Medieval. El Laboratorio Virtual es un módulo donde se simulan experimentos de la vida real como los experimentos de Biología y Física. El módulo Contenido se convierte en un libro electrónico ya que se presenta contenidos a los estudiantes a partir de textos, gráficos, videos y sonidos. Los ejercicios mostrados en la colección posibilitan el entrenamiento de los estudiantes.

Son 10 productos los que integran la Colección El Navegante, los cuales están estructurados didácticamente en 6 módulos básicos: Contenido, Mediateca, Ejercicios, Juegos, Resultados y Maestro.

La Colección El Navegante se caracteriza por: (10)

- Ha sido elaborado a partir de necesidades de la enseñanza y por encargo de las mismas, o sea, a partir del conocimiento de los problemas de cada una de las asignaturas y con la participación de los mejores especialistas en la materia.
- Pensados para el joven venezolano y adaptado a sus intereses y necesidades.
- Permiten el trabajo grupal, no solo para favorecer el empleo de los mismos por varios estudiantes sino para aprovechar el valor del aprendizaje colaborativo.
- Registro de la actuación de los estudiantes con el software para su empleo en el análisis de los resultados obtenidos para la toma de decisiones en la dirección del proceso pedagógico.
- Desarrollados por equipos multidisciplinarios formados por: guionistas, diseñadores gráficos, especialistas en audiovisuales, programadores, expertos en informática educativa, psicólogos.
- Incorporan recursos multimedia como imágenes fijas y en movimiento, sonidos y videos.
- Empleo de estrategias pedagógicas de análisis de respuestas y retroalimentaciones reflexivas o niveles de ayuda.
- Incorporación de servicios informáticos adicionales de búsqueda, exportación de información, impresión y música.

1.4 Metodología de desarrollo de software

En la creación de un software en equipos de trabajo, en un tiempo planificado y que cumpla los parámetros exigidos por el cliente, es necesario apoyarse en una metodología que planee el proceso de desarrollo utilizando procedimientos, técnicas y herramientas.

En la actualidad, las metodologías existentes se pueden clasificar en dos grupos. Las tradicionales o pesadas, estas se centran en el control del proceso, estableciendo una planificación de tareas y responsabilidades que se deben llevar a cabo, generan cantidades de documentos y artefactos. Son definidas por especificar las herramientas y notaciones que deben ser utilizadas en la implementación del producto. Entre estas metodologías pesadas se encuentra el Proceso Unificado de Desarrollo (RUP).

Las metodologías ágiles, constituyen otra clasificación, brindan mayor libertad al equipo de desarrollo, se basan en la interacción entre el desarrollador y el cliente, y el desarrollo incremental del software se pone de manifiesto mediante iteraciones cortas. En los proyectos en los cuales los requisitos son cambiados constantemente estas metodologías muestran una efectividad notable y reducen el tiempo de desarrollo manteniendo la calidad ante los cambios inesperados.

No existe una metodología universal que indique cómo crear todo tipo de software. Cuando un equipo decide construir un producto, se debe escoger la metodología a utilizar teniendo en cuenta las características, complejidad, envergadura del proyecto y el tipo de contrato establecido para el mismo.

Debido a que el sistema a desarrollar es extenso, se requiere realizar un levantamiento exhaustivo de requisitos y enfatizar el trabajo en la fase inicial del desarrollo con el objetivo de detectar los posibles errores antes que el producto gane madurez. Debido a esto se hace necesario realizar un desarrollo iterativo e incremental en el cual vayan adicionando funcionalidades a un proyecto inicial hasta lograr el producto final.

Teniendo en cuenta que el equipo de desarrollo trabaja distante del cliente y toda la interacción se hace a través de terceros, se necesita llevar un control de la documentación que ayude a los nuevos integrantes incorporarse al proyecto, además de servir para futuras versiones de la colección. Como no se cuenta con experiencias en proyectos similares, se necesita una metodología que reúna experiencias favorables de otros proyectos y pueda guiar al equipo de desarrollo utilizando las mejores prácticas probadas en la industria de la construcción de software.

Además, por la cantidad de miembros en el proyecto, se requiere realizar una correcta asignación de responsabilidades y una planificación exhaustiva, de manera que todos los miembros contribuyan al desarrollo del producto.

Todas las características anteriormente mencionadas colocan a RUP como la mejor opción entre las metodologías de desarrollo de software para desarrollar El Navegante.

El Proceso Unificado de Desarrollo

El Proceso Unificado de Desarrollo (RUP) es una metodología de desarrollo de software dirigida por casos de uso, centrado en la arquitectura, iterativo e incremental. Es dirigida por casos de uso porque estos no solo guían el proceso de desarrollo sino que proporcionan un hilo conductor, debido a que avanza a través de los distintos flujos de trabajos, se especifican, se diseñan y los casos de uso finales son la fuente de la cual los ingenieros de prueba construyen sus casos de pruebas. La arquitectura es el eje central para el desarrollo de un software, la arquitectura debe permitir que los casos de uso encajen en la misma, así como, que se implementen todos los casos de uso. Es iterativo e incremental ya que el proceso de desarrollo del software se puede dividir en varias iteraciones, obteniéndose en cada una de ellas un miniproyecto o partes más pequeñas del trabajo, y los incrementos del software se refieren al crecimiento del producto. Cada miniproyecto es una iteración que resulta en un incremento.

RUP define cuatro fases (Inicio, Elaboración, Construcción y Transición) y dentro de cada una de ellas el equipo de desarrollo pasa por todos los 9 Flujos de Trabajo (Modelación del negocio, Requerimientos, Análisis y diseño, Implementación, Prueba, Instalación, Administración del proyecto, Administración de configuración y cambios, Ambiente) que son transversales a las fases, inclusive en varias iteraciones. (11)

Una vez seleccionada la metodología que va a guiar el desarrollo del software, se hace necesario definir qué lenguaje utilizar para modelar el sistema. Para esta investigación se utiliza UML, como lenguaje de modelado debido a sus características.

1.5 Lenguaje de modelado UML

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. (12)

Se decide utilizar como lenguaje de modelado UML debido a las siguientes características:

UML no sólo se necesita para comunicar las ideas de los desarrolladores sino también sirve de apoyo en los procesos de análisis de un problema. Este lenguaje representa y modela la información con la que se trabaja en la fase del análisis y diseño. Es independiente del lenguaje de implementación, de tal manera

que los diseños realizados usando este lenguaje se pueden implementar en cualquier lenguaje que soporte las posibilidades de UML.

El Lenguaje Unificado de Modelado aporta las siguientes ventajas: (12)

- Mayor vigor en la especificación.
- Permite realizar una verificación y validación del modelo realizado.
- Permite generar código a partir de los modelos e inversa.

1.6 Herramientas de modelado de sistemas

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) son herramientas desarrolladas con el objetivo de aumentar la productividad en el desarrollo del software, logrando la reducción de tiempo. Utilizando estas herramientas se puede abstraer al código fuente, en un nivel donde la arquitectura y el diseño son más fáciles de entender y modificar.

Visual Paradigm 3.4

Visual Paradigm es una herramienta profesional que soporta el lenguaje de modelado UML y el ciclo de vida completo del desarrollo de software. Ayuda a modelar aplicaciones con una mayor calidad y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y documentación del proceso de desarrollo. Esta herramienta CASE proporciona abundantes tutoriales, demostraciones interactivas y proyectos UML. Una de las características más importantes del Visual Paradigm es que es multiplataforma. (13)

Se integra con NetBeans IDE y ofrece:

- Entorno de creación de diagramas para Lenguaje Unificado de Modelado 2.0 y 2.1.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo, que facilita la comunicación.
- Disponibilidad de integrarse en las principales herramientas de programación.
- Disponibilidad en múltiples plataformas.
- Modelado colaborativo con Control de Versiones y Subversión.

Se decidió utilizar como herramienta CASE para el modelado de la aplicación el Visual Paradigm, debido fundamentalmente a que es multiplataforma, y que ofrece funcionalidades como generación de código para varios lenguajes de programación, generación de bases de datos, lo que permite la generación automática de bases de datos a partir de un Modelo entidad-relación e interoperabilidad entre diagramas, la cual permite a partir de un diagrama obtener otro que guarde relación con el mismo. Todas estas funcionalidades favorecen el desarrollo de la aplicación en cuanto al tiempo y calidad de la misma.

1.7 Lenguaje de desarrollo

La selección del lenguaje de programación en un proyecto es uno de los aspectos más controversiales, debido a que los miembros del proyecto, en particular los desarrolladores, deben tener una familiaridad con el lenguaje seleccionado, conocer el IDE que lo soporta, la eficiencia, consumo de memoria y soporte del lenguaje para programar la aplicación que se desea. Los lenguajes de programación para las páginas web se dividen en dos grupos: lenguajes del lado del cliente y lenguajes del lado del servidor.

1.7.1 Lenguajes del lado del cliente

XHTML

XHTML (Lenguaje de Marcado de Hipertexto Extensible) es una versión más estricta de HTML, que nace precisamente con el objetivo de remplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML. (14)

Al estar orientado al uso de un etiquetado correcto, exige una serie de requisitos básicos a cumplir en lo que a código se refiere. Entre estos requisitos básicos se puede mencionar una estructuración coherente dentro del documento donde se incluirían elementos correctamente anidados, etiquetas en minúsculas, elementos cerrados correctamente y atributos de valores entrecomillados.

JavaScript

JavaScript es un lenguaje scripting[1], interpretado por la mayoría de los navegadores web, permitiendo el desarrollo de interfaces de usuarios mejoradas y páginas web dinámicas. No requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos. Como JavaScript es interpretado del lado del cliente se encarga de realizar acciones como: pedidos de datos, confirmaciones, mostrar mensajes, crear animaciones y comprobar campos.

CSS

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Es la mejor forma de separar los contenidos de la presentación y es imprescindible para crear páginas web complejas. Si el lenguaje HTML/XHTML se utiliza para marcar los contenidos, el lenguaje de estilo se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los textos, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista, entre otros aspectos de la apariencia de un sitio web. (15)

El uso correcto de XHTML proporcionará ventajas al módulo a implementar, debido a que aumentará la velocidad de navegación, ya que las páginas se construyen con mayor rapidez en los navegadores, además, brinda compatibilidad entre los navegadores que cumplen con los estándares web establecidos por la W3C, garantizando que en versiones posteriores funcione correctamente. Se decide utilizar JavaScript en el módulo debido a la gran cantidad de animaciones que contiene y además, brinda conexión asincrónica con el servidor web, aumentado la velocidad de respuesta del mismo.

1.7.2 Lenguajes del lado del servidor

PHP 5.2

PHP no es un lenguaje de marca como HTML, XML o XHTML, es un lenguaje de programación con variables, sentencias condicionales y funciones. La mayor parte de su sintaxis ha sido tomada de Java, C y Perl, con algunas características específicas. Se ejecuta en el servidor, por lo que permite acceder a los recursos que tenga el servidor, como las bases de datos y los resultados son enviados al navegador. Al

[1] Un lenguaje de script es un pequeño lenguaje de programación cuyo código se inserta dentro del documento HTML. Estos lenguajes permiten variar dinámicamente el contenido del documento, modificar el comportamiento normal del navegador, validar formularios, realizar pequeños trucos visuales.

ser en lenguaje ejecutado por el servidor no es necesario que el navegador lo soporte, es independiente de este, sin embargo para que sus páginas PHP funcionen, el servidor donde están alojadas debe soportar PHP.

La ventaja que tiene PHP sobre otros lenguajes de programación que se ejecutan en el servidor es que permite intercalar sentencias PHP en las páginas HTML. Otras de las ventajas se mencionan a continuación: (16)

- Fácil de aprender.
- Las aplicaciones desarrolladas con este lenguaje se caracterizan por tener un tiempo de ejecución corto.
- Es orientado a objeto.
- Capacidad de conexión con la mayoría de los manejadores de base de datos: MySQL, PostgreSQL, Oracle y MS SQL Server.
- Posee documentación en su página oficial, la cual incluye descripción y ejemplos de cada una de sus funciones.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Incluye gran cantidad de funciones.

Se decide utilizar este lenguaje de desarrollo debido a que admite la programación orientada a objeto, permitiendo programar el software organizado. Además, existe amplio conocimiento del lenguaje en el equipo de desarrollo y contiene mucha documentación en su página oficial, la cual está respaldada por la comunidad de desarrolladores.

1.8 Framework

Un framework o marco de trabajo es un software que se puede personalizar e intercambiar para el desarrollo de una aplicación. También, se puede considerar como otra aplicación incompleta y configurable a la que se le pueden añadir últimas piezas para construir una aplicación. Ayudan a desarrollar aplicaciones con mayor rapidez, pues poseen una estructura definida y una organización para el desarrollo y mantenimiento del software desarrollado. Permite la reutilización del código y facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. (17)

1.8.1 Framework del lado del cliente

jQuery 1.5

jQuery es un nuevo tipo de librería de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, permite manejar eventos, desarrollar animaciones e interactuar con la tecnología AJAX (Asynchronous JavaScript And XML). Está diseñado para cambiar la forma de escribir código JavaScript.

Algunas de las ventajas que proporciona su utilización: (18)

- Ahorro de líneas de código.
- Es soportada por cualquiera de los principales navegadores web como Internet Explorer, Firefox, Opera, Safari y Chrome.
- Proporciona un conjunto de funciones para animar el contenido de la página.
- Integra funcionalidades para trabajar con AJAX.
- Tiene gran aceptación por parte del grupo de desarrollo ya que es una librería basada en JavaScript, del cual se tiene un conocimiento amplio.
- Posee una amplia comunidad creadora de componentes o plugins, lo que proporciona una manera más fácil de encontrar soluciones ya creadas para desarrollar elementos de interfaz de usuario, galerías de imágenes y validadores.

La gran ventaja de la función de jQuery, es que la página se puede manipular en cuanto se ha cargado su código HTML, y por tanto, se ha construido el árbol DOM de la página, mientras que la función de JavaScript espera a que se carguen todos los elementos de la página, incluyendo las imágenes. De esta forma, las aplicaciones realizadas con jQuery pueden responder de forma mucho más rápida que las aplicaciones JavaScript tradicionales. (19)

Se selecciona el framework jQuery debido principalmente a que tiene gran aceptación por parte de los programadores, además, de ser un marco de trabajo bien documentado, estable y con un equipo de desarrollo a cargo de las mejoras y de sus actualizaciones. Otro de los aspectos que lo hacen favorable es la existencia de la comunidad de creadores de plugins o componentes, los cuales hacen más fácil encontrar soluciones ya creadas en este para implementar elementos como interfaces de usuarios y galerías. Es una librería muy utilizada por su sencillez, flexibilidad y rendimiento. Todas las características anteriormente

mencionadas colocan a jQuery como la mejor opción entre los framework del lado del cliente para desarrollar el módulo General de la Colección El Navegante.

1.8.2 Framework del lado del servidor

Symphony 1.4.3

Symphony es un framework basado en el lenguaje PHP que facilita el desarrollo de las aplicaciones web. Aumenta la productividad y ayuda a mejorar la calidad de las aplicaciones web aplicando todas las buenas prácticas y patrones de diseño que se han definido para la web. Cuenta además con una amplia documentación, y variada distribución de libros de forma gratuita y decenas de tutoriales.

Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. (20)

Se puede utilizar en plataformas Unix, Linux y Windows. Requiere de una instalación, configuración y líneas de comando, incorpora el patrón MVC, soporta AJAX, plantillas y un gran número de bases de datos.

Entre las características que determinaron su selección se encuentran: (21)

- Fácil de instalar y configurar en sistemas Windows, Mac y Linux.
- Funciona con todos los gestores de bases de datos comunes (MySQL, PostgreSQL, SQLite, Oracle, MS SQL Server).
- Compatible solamente con PHP 5, para asegurar el mayor rendimiento y acceso a las características más avanzadas de PHP.
- Flexible hasta cualquier límite y extensible mediante un completo mecanismo de plugins.
- Publicado bajo licencia Massachusetts Institute of Technology (MIT) de software libre y apoyado por una empresa comprometida con su desarrollo.

Se opta por la selección de Symphony, por ser un framework basado en el lenguaje de desarrollo PHP y por las características fundamentales de este marco de trabajo, ya que el mismo posee una amplia documentación. Posee integración con el sistema gestor de base de datos PostgreSQL. Mantiene su

seguridad debido a que es protegido de los ataques de tipo XSS (Cross Site Scripting) y CSRF (CROSS Site Request Forgery). Además, este framework del lado del servidor contiene una interfaz de línea de comando que con solo introducir una secuencia de comando se ejecutarán tareas que disminuirán el tiempo de desarrollo.

1.9 Patrón Modelo Vista Controlador

Según Grady Booch una arquitectura orientada a objetos bien estructurada está llena de patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles.

El patrón arquitectónico Modelo Vista Controlador (MVC) es utilizado en el desarrollo de aplicaciones web, debido a la posibilidad que brinda de organizar el código de una aplicación separando las interfaces de usuario, la lógica del negocio y los datos, donde cada capa realiza funciones específicas.

El modelo representa la información con la que trabaja la aplicación; su lógica de negocio. Es responsable de: (22)

- Acceder a la capa de almacenamiento de datos.
- Define las reglas de negocio (la funcionalidad del sistema).
- Lleva un registro de las vistas y controladores del sistema.

La vista transforma el modelo en una página web que permite al usuario interactuar con ella. Es responsable de: (22)

- Recibir datos del modelo.
- Tienen un registro de su controlador asociado.
- Pueden dar el servicio de actualización, para que sea invocado por el controlador o por el modelo.

El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista. Es responsable de: (22)

- Recibir los eventos de entrada (un clic, un cambio en un campo de texto, etcétera).
- Contener reglas de gestión de eventos.

Para desarrollar el módulo es necesario seguir un patrón de diseño como el MVC, debido principalmente a que Symphony, está basado en este patrón de diseño implementando las aplicaciones rápidas y sencillas. Además, desarrollar una aplicación siguiendo este patrón arquitectónico tiene muchas ventajas, el desarrollador no debe preocuparse de solicitar que las vistas se actualicen, ya que este proceso es realizado automáticamente por el modelo de la aplicación, si se desea hacer una modificación al modelo de dominio, como aumentar métodos o datos, solo debe modificarse el modelo y las interfaces del mismo con las vistas. Las modificaciones a las vistas no afectan a los otros módulos. El patrón MVC permite que una misma aplicación tenga diferentes vistas, además, permite que cada componente se especialice en su función, ofrece claridad en el diseño y facilita el mantenimiento.

1.10 Entorno de desarrollo integrado

Un entorno de desarrollo integrado (Integrated Development Environment, IDE), es un programa compuesto por herramientas que utilizan los programadores para desarrollar código. Puede ser utilizado con un único lenguaje de programación o con varios. Algunas de las herramientas que componen un IDE son: un editor de texto, un compilador, un intérprete, un sistema de ayuda para la construcción de interfaces gráficas de usuarios, y opcionalmente un sistema de control de versiones. (23)

NetBeans 6.9

NetBeans es un IDE escrito en el lenguaje de programación Java. El proyecto NetBeans consiste en un IDE de código abierto y una plataforma de aplicación, los cuales pueden ser usadas como un framework para compilar aplicaciones. Es una herramienta para escribir, compilar, depurar y ejecutar programas. Tiene soporte para varios lenguajes como PHP, JavaScript, HTML, CSS y su versión 6.8 tiene soporte para el framework Symfony. Funciona en los sistemas operativos: OpenSolaris, Linux, Windows y Mac OS. (24)

NetBeans IDE contiene las herramientas para que los desarrolladores de software puedan crear aplicaciones desktop, web, y aplicaciones móviles, con el lenguaje Java, así como también C/C++, PHP, JavaScript, Groovy, y Ruby.

Se decidió utilizar como entorno de desarrollo integrado el NetBeans, específicamente por las siguientes características: es compatible con PHP5 por lo que propone un esqueleto para organizar el código fuente,

el editor integra los lenguajes XHTML, CSS y JavaScript. Se integra con los sistemas de control de versiones tales como SVN, la cual es una condición necesaria para los proyectos. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

1.11 Sistema gestor de base de datos

Un gestor de base de datos o sistema de gestión de base de datos es un software que permite introducir, organizar y recuperar la información de las bases de datos. Ayuda a realizar acciones como: definición, mantenimiento de la integridad, control de la seguridad, privacidad y manipulación de los datos. (25)

PostgreSQL 8.4

PostgreSQL es un gestor de base de datos relacional, orientada a objeto muy conocido y usado en entornos de software libre, se distribuye bajo la licencia BSD (Berkeley System Distribution), lo que permite su uso, redistribución y modificación con la única restricción de mantener el derecho de autor a sus verdaderos autores. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de confiabilidad, integridad de datos y corrección. Funciona en los principales sistemas operativos, incluyendo Linux, Unix y Windows.

PostgreSQL se destaca por su lista de prestaciones que lo hacen capaz de competir con cualquier SGBD comercial: (26)

- La API de acceso al SGBD se encuentra disponible en C, C++, Java, Perl, PHP y Python.
- Cuenta con un conjunto de tipos de datos, permitiendo además su extensión mediante tipos y operadores definidos y programados por el usuario.
- Su administración se basa en usuarios y privilegios.
- Es altamente confiable en cuanto a estabilidad se refiere.

Se selecciona como sistema gestor de base de datos PostgreSQL, debido a que se distribuye bajo la licencia BSD, la cual tiene menos restricciones ya que el usuario tiene la libertad ilimitada del software, permite la redistribución, modificación e incluso redistribuirlo como no libre. PostgreSQL posee amplia documentación. Además, es compatible con el almacenamiento de objetos binarios, incluyendo imágenes, sonidos y videos. Tiene interfaces de programación de C/C++, Java, PHP y Ruby. Posee gran

escalabilidad, ya que puede ajustarse al rendimiento de CPU y a la cantidad de memoria que posee el sistema, haciéndole capaz de soportar cantidad de peticiones simultáneas. Además, tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.

1.12 Servidores web

Un servidor web es un programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten usando el protocolo HTTP o el HTTPS. Por medio de la especificación de la búsqueda el servidor web buscará una página específica o ejecutará un programa y enviará algún resultado sobre la búsqueda recibida. Entre los servidores web más utilizados se encuentra el Apache.

Apache 2.0

El servidor Apache es un software que está estructurado en módulos. La configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo. Los módulos del Apache se pueden clasificar en tres categorías: (27)

Módulos Base: módulo con las funciones básicas del Apache.

Módulos Multiproceso: son los responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos atender a las peticiones.

Módulos Adicionales: cualquier otro módulo que le añada una funcionalidad al servidor.

Las funcionalidades más importantes se encuentran en el módulo base, lo que hace necesario un módulo multiproceso para manejar las peticiones. Para cada uno de los sistemas operativos en los que se ejecuta Apache se han creado varios módulos multiproceso, optimizando el rendimiento y rapidez del código. El módulo adicional, como su nombre lo indica, es para las funcionalidades que se quieran agregar.

La gran popularidad de Apache se debe a múltiples características que brindan una serie de ventajas como: (28)

- Funciona en una multitud de sistemas operativos.
- Es una tecnología gratuita de código abierto.
- Es un servidor altamente configurable de diseño modular. Actualmente existen muchos módulos para Apache que son adaptables y permiten su instalación facilitando el trabajo con este servidor web.
- Trabaja con gran cantidad de lenguajes como Perl, PHP, Java, JavaScript; teniendo todo el soporte que se necesita para tener páginas dinámicas.
- Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.

Se selecciona como servidor web a Apache por ser multiplataforma, siendo así se puede adaptar a cualquier sitio web sin importar el sistema operativo por el cual se esté ejecutando. Es un servidor web flexible, rápido, eficiente y adaptado a los nuevos protocolos. Como consta de tres módulos de apoyo puede adaptarse a los diferentes entornos, esto le permite a los administradores elegir qué características van a ser incluidas en el servidor, seleccionando qué módulo se va a ejecutar. Es uno de los servidores web más seguros y más usados en la actualidad. Además, la integración de Apache con los lenguajes de programación PHP y JavaScript, con el sistema gestor de base de datos PostgreSQL y con el IDE NetBeans hace más fácil y rápida el manejo de datos y ofrecen las mismas ventajas y compatibilidad tanto en software libre como en algún otro sistema operativo.

Conclusiones

Para llevar a cabo el desarrollo del módulo General se estudiaron los conceptos relacionados con el proceso de desarrollo de software educativo, teniendo en cuenta las características de otras colecciones con características similares a la Colección El Navegante en su versión plataforma concluyendo que no satisfacen las necesidades existentes para este módulo, por lo que se hace necesario el desarrollo del mismo, así como se realizaron investigaciones sobre las tecnologías y herramientas a utilizar en el desarrollo del módulo. Como metodología para guiar el proceso de desarrollo se seleccionó RUP. Como lenguaje de modelado se seleccionó UML y Visual Paradigm como herramienta CASE. Se decidió utilizar PHP como lenguaje de programación, jQuery como framework del lado del cliente y con PHP como punto

de partida, se escoge Symphony como framework del lado del servidor pues es de fácil aprendizaje. Para la organización del módulo se decidió utilizar el patrón arquitectónico Modelo Vista Controlador. Por la gran integración que tiene con varias tecnologías, se adopta NetBeans, como servidor Web Apache es el mejor de los candidatos ya que es uno de los más utilizados por ser una herramienta totalmente libre. El Gestor de Bases de Datos que se seleccionó fue PostgreSQL, pues junto a Apache y PHP forma una potente plataforma totalmente libre, robusta, multiplataforma y compatible con muchas herramientas de desarrollo.

Capítulo 2 Características del Sistema

Para identificar las características de un sistema se realiza el levantamiento de requisitos identificándose las funcionalidades que debía tener el software. Estas funcionalidades deben especificarse antes de comenzar a construir el producto, para luego llevar a cabo el análisis y diseño correctamente, lo cual permitirá que el sistema se ajuste al contexto del trabajo del cliente y de los usuarios finales. Especificando estas funcionalidades desde el inicio se evitará cambios posteriores en el sistema y facilitará el entendimiento de los clientes de lo que el sistema debe hacer.

2.1 Modelo de dominio

Debido a que no se tienen bien definidos los procesos del negocio y no se identifican con claridad los actores ni sus responsabilidades, se decide realizar un modelo del dominio con el propósito de capturar los objetos o clases más importantes del sistema. Estos se describen mediante diagramas de UML, especialmente diagramas de clases, los cuales muestran a los clientes, usuarios y desarrolladores los conceptos del dominio y cómo se relacionan unos con otros mediante asociaciones.

Análisis de los conceptos del dominio

- Colección El Navegante: Colección de software educativo destinada a apoyar el proceso de enseñanza – aprendizaje de los jóvenes en las escuelas secundarias venezolanas.
- Módulo: conjunto de elementos fundamentales que componen el producto.
- Ejercicio: módulo que consta de un cuestionario interactivo dirigido a ejercitar, fijar y profundizar los contenidos estudiados en cada uno de los temas abordados, con un sistema de retroalimentación que facilita el logro de los objetivos.
- Juego: módulo que contiene los juegos que el estudiante debe realizar para profundizar algún contenido.
- Maestro: módulo que permite al profesor configurar algunas opciones del producto.
- Mediateca: módulo que contiene galerías de imágenes, fotos, y videos utilizados en cada software, ordenados y clasificados por temáticas.
- Resultado: módulo que permite interactuar con los archivos que almacenan toda la actividad del estudiante.

- Contenido: módulo que constituye la base de conocimientos. En él se proporciona información sobre el o los temas que se abordan en cada software.

Diagrama del modelo de dominio

En el diagrama del modelo de dominio del módulo General se muestra como están relacionados los anteriores conceptos:

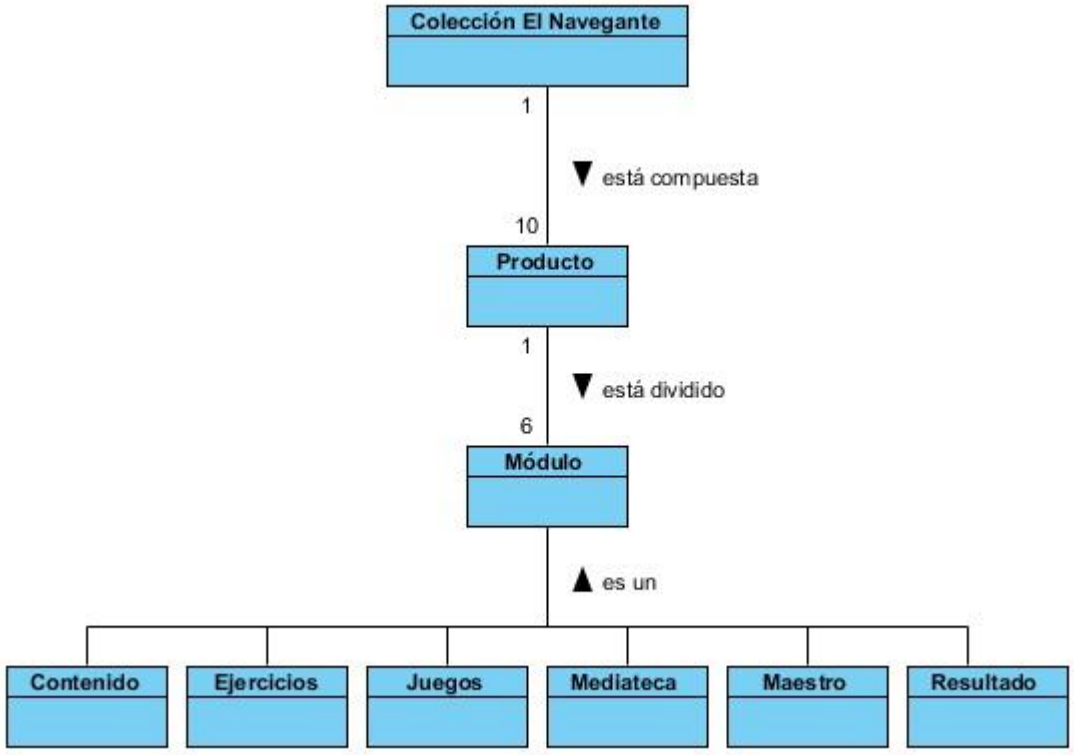


Figura 2.1 Modelo de Dominio

2.2 Descripción del sistema propuesto

Teniendo en cuenta el problema a resolver con esta investigación se propone desarrollar el módulo General de la Colección El Navegante, el cual agrupa los servicios comunes que podrán ser accedidos por

los usuarios desde cualquier interfaz de la aplicación. Estos servicios comunes se ven reflejados en el software con las siguientes funcionalidades: autenticación de usuarios, impresión, información del producto, perfil de usuario, búsqueda de contenidos, opciones de navegabilidad por las distintas páginas del producto, interacción con la mascota, noticias, efemérides, ayuda y consejos del software. Dichas funcionalidades proporcionarán una mejor interacción con el producto, garantizarán la comodidad, organización, uniformidad y seguridad de los usuarios.

2.3 Especificación de requisitos

Los requisitos expresan de manera clara lo que los usuarios quieren del producto, representan condiciones o capacidades que deben estar presentes en el software para cumplir con un acuerdo establecido con el cliente. Se establece que sean especificados por escrito, como parte de un documento formal, describiendo una característica del sistema a desarrollar que pueda ser probada o verificada una vez concluida la propuesta. Teniendo en cuenta sus características los requisitos se clasifican en: **funcionales** y **no funcionales**. (29)

Los **requisitos funcionales** son aquellas capacidades o condiciones que el sistema debe cumplir para satisfacer las necesidades funcionales del cliente. Son descripciones claras y precisas de las acciones que el sistema debe permitir al usuario. No especifican la forma en que el sistema resolverá las necesidades, imponen solamente lo que el sistema debe hacer y no explican nunca cómo lo hará. (29)

Los **requisitos no funcionales** son aquellas propiedades o cualidades que el producto debe tener una vez que esté finalizado. Se expresan como propiedades o características que hacen que el producto sea atractivo, usable, rápido o confiable. (29)

Las condiciones o capacidades que deben estar presentes en el módulo General de la Colección El Navegante para garantizar la satisfacción del cliente están reflejadas en los siguientes requisitos funcionales y no funcionales:

Requisitos funcionales

RF 1 Mostrar la imagen de presentación de la Colección El Navegante.

RF 2 Mostrar la imagen de presentación del producto ejecutado.

RF 3 Autenticar usuario.

RF 3.1 Seleccionar el tipo de usuario para inicio de sesión.

RF 3.2 Seleccionar la cantidad de estudiantes que iniciarán sesión en el software.

RF 3.3 Seleccionar el avatar.

RF 4 Generar escritorio de trabajo.

RF 4.1 Cargar perfil de usuario.

RF 4.2 Reproducir la locución de bienvenida al sistema.

RF 5 Mostrar la pantalla principal del producto.

RF 5.1 Regresar desde cualquier lugar del software a la pantalla principal del mismo.

RF 6 Mostrar la mascota del software.

RF 6.1 Minimizar y restaurar la mascota.

RF 6.2 Mostrar la ayuda de la mascota.

RF 7 Activar y desactivar el sonido de fondo del software.

RF 7.1 Reproducir de forma aleatoria el sonido de fondo del software.

RF 8 Mostrar la ayuda general y por pantalla del software.

RF 9 Realizar búsquedas en el software.

RF 9.1 Acceder al área donde se encontró el texto buscado.

RF 9.2 Navegar entre las páginas de los resultados de la búsqueda.

RF 10 Consultar información del producto.

RF 11 Imprimir contenido.

RF 12 Consultar efemérides.

RF 12.1 Ver efeméride.

RF 13 Consultar noticias.

RF 13.1 Ver noticia.

RF 14 Consultar Sabías que.

RF 15 Cerrar sesión.

RF 16 Registrar la traza del estudiante en el software.

RF 17 Registrar la traza de los profesores en el software.

RF 18 Consultar el perfil de usuarios.

18.1 Cambiar contraseña.

RF 19 Acceder a los módulos del sistema.

RF 20 Mostrar el nombre del software en cada una de las pantallas del mismo.

RF 21 Navegar por las pantallas del software.

Requisitos no funcionales

RNF 1 Requisitos de hardware

Uso local

- Requerimientos mínimos: 256 MB RAM, 9 GB de HDD, 500 Mhz.
- La resolución de pantalla a la que deben visualizarse los productos de la colección es 1024 x 600 píxeles.

Uso en red

- Velocidad de conexión mínima: 10 Mbit/s.
- Requerimientos mínimos: 128 MB RAM, 9 GB de HDD, 500 MHz.
- Cantidad máxima de máquinas conectadas: 20
- El sistema interactuará con una impresora que permita imprimir los diferentes contenidos como respuesta a las funcionalidades del sistema.

RNF 2 Requisitos de software

- Computadora personal con navegador Mozilla Firefox versión 3.5 o cualquier otra versión compatible con éste.
- Sistema operativo Canaima versión 2.1 y Ubuntu en su versión 10.0.4.
- Se podrá utilizar también en los sistemas operativos Linux, Mac y Windows.

RNF 3 Requisitos de restricciones en el diseño y la implementación

- Framework del lado del cliente: jQuery 1.4
- Framework del lado del servidor: Symphony 1.4.3
- Lenguaje de programación del lado del cliente: JavaScript
- Lenguaje de programación del lado del servidor: PHP 5.2
- Servidor web: Apache 2.x
- IDE de desarrollo: NetBeans 6.9
- Servidor de base de datos: PostgreSQL 8.4

- Patrón arquitectónico: Patrón Modelo Vista Controlador

RNF 4 Requisitos de apariencia e interfaz externa

- Las interfaces de usuario estarán acorde a las edades a las que van dirigidos cada uno de los 10 productos que integran la Colección El Navegante.
- Las interfaces se desarrollarán teniéndose en cuenta elementos de las aplicaciones con tecnología multimedia diseñadas para escritorio y las aplicaciones web.
- El diseño de la interfaz gráfica deberá garantizar la distinción visual entre los elementos del sistema.

RNF 5 Requisitos de usabilidad

- Facilidad de desarrollo con software libre debido a la amplia gama de herramientas disponibles para el desarrollo de aplicaciones Web.
- La utilización de la colección por parte de los estudiantes u otros usuarios sin avanzados conocimientos de computación, no requiere de previa preparación, debido al diseño sencillo y estándar que tienen los productos de la misma.
- Cada producto cuenta con una ayuda que se ubicará como un servicio en el mismo, cubriendo cada una de las pantallas del software y será escrita en lenguaje español lo más sencilla y asequible posible para facilitar su entendimiento por parte de los usuarios.

RNF 6 Requisitos de soporte

- Se realizará transferencia tecnológica de la colección a los clientes.
- Se impartirán clases a los profesores venezolanos para explicar el funcionamiento y utilidad del producto.

RNF 7 Requisitos Legales

- Deben respetarse las licencias y condiciones legales bajo las que fueron adquiridos cada uno de los recursos multimediales.
- Debe mostrarse de cada media la fuente de donde fue extraída.

RNF 8 Requisitos de rendimiento

Uso local

- Tiempo de respuesta de una página: no mayor a 30 segundos.

Uso en la red

- Velocidad de respuesta máxima: 5 minutos

2.4 Modelo de casos de uso del sistema

El modelo de casos de uso del sistema representa gráficamente la interacción de los casos de uso del sistema con los actores del sistema. Este modelo le permite a los desarrolladores y al cliente llegar a un acuerdo sobre los requisitos que debe presentar el producto, y describe el comportamiento del sistema para cada tipo de usuario.

Los casos de uso del sistema son un conjunto de secuencias de acciones que un sistema ejecuta y que produce un resultado observable para un actor, es decir, fragmentos de funcionalidad que el sistema ofrece a los actores que interactúan con el mismo. (29)

Diagrama de casos de uso del sistema

El diagrama de casos de uso del sistema que se muestra a continuación evidencia cómo interactúa el actor con los casos de uso que conforman la propuesta de sistema para el módulo General. Se evidencia en los casos de uso Generar escritorio de trabajo, Registrar traza del usuario y Manipular mascota el patrón de caso de uso Reusabilidad y Adición en el caso de uso Consultar ayuda.

Actor del sistema	Descripción
Usuario	Persona que interactúa con el software, puede ser Estudiante, Invitado o Maestro.

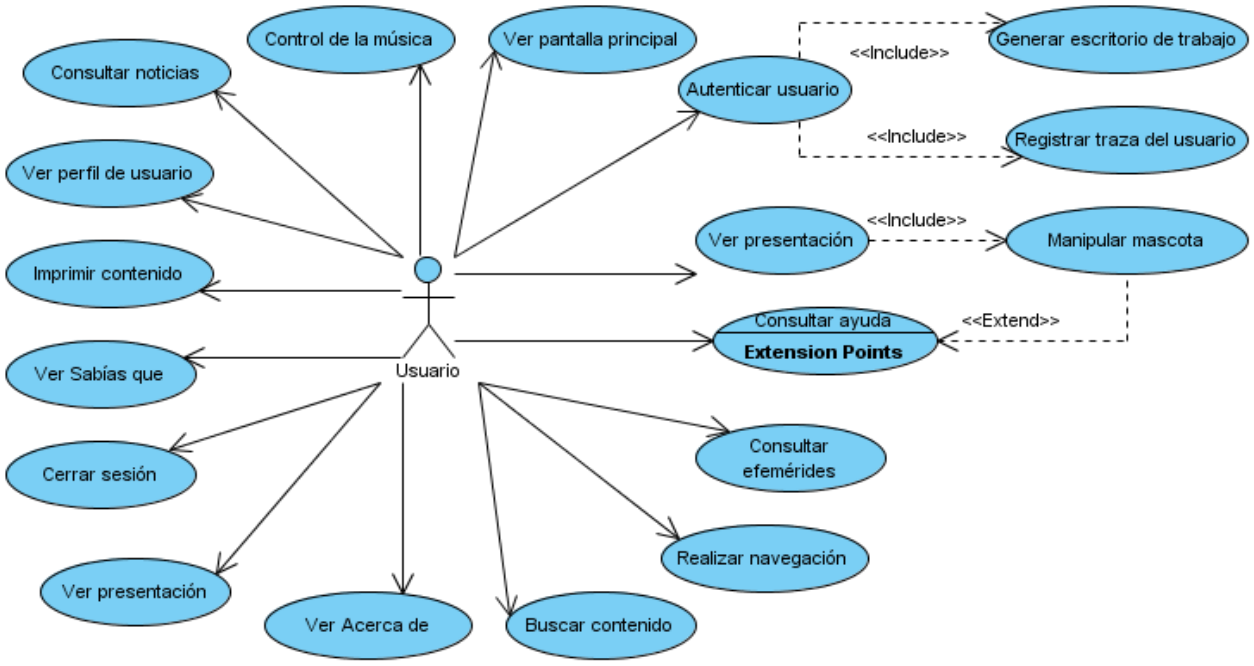


Figura 2.2 Diagrama de caso de uso del sistema

2.5 Descripción de casos de uso del sistema

Para entender la funcionalidad asociada a cada caso de uso no es suficiente con la representación gráfica del diagrama de casos de uso, sino que es necesario realizar las descripciones de estos casos de uso, con el objetivo de mostrarle al cliente un acercamiento de la propuesta de sistema y orientar a los desarrolladores durante la implementación. En las se detallan las acciones que tienen lugar durante la interacción actor-sistema.

A continuación se muestran algunas de las descripciones de los CU de la propuesta para el módulo General. Para consultar las restantes descripciones ver los Anexos sección Anexo 1 Descripciones textuales de los CU.

Caso de uso	Autenticar usuario
Actores	Usuario
Resumen	El caso de uso inicia cuando el actor selecciona algunos de los productos de la

	colección. El sistema brinda la posibilidad de seleccionar el tipo de usuario: Estudiante, Maestro e Invitado. El usuario selecciona el tipo de usuario, introduce los datos correspondientes para la autenticación, el sistema verifica la autenticidad de los datos y se inicia la sesión del usuario autenticado, finalizando así el caso de uso.
Precondiciones	Debe haber finalizado la presentación del sistema.
Referencias	RF 3, RF 3.1, RF 3.2 y RF 3.3
Flujo normal de eventos	
Acciones del actor	**Respuesta del sistema**
1. El caso de uso inicia cuando el actor selecciona algunos de los productos de la colección.	
	2. Muestra la pantalla de autenticación y brinda la posibilidad de seleccionar el tipo de usuario: ▪ Estudiante ▪ Invitado ▪ Maestro Y permite: ▪ Cerrar la vista ▪ Aceptar la operación
3. Selecciona el tipo de usuario “Estudiante”.	
	4. Muestra la interfaz para introducir los datos del estudiante: ▪ Cédula de identidad ▪ Contraseña Brinda la posibilidad de seleccionar la cantidad de

	estudiantes que iniciarán sesión simultáneamente en el software: ▪ Uno (se muestra predeterminado) ▪ Dos ▪ Tres ▪ Cuatro Y permite: ▪ Seleccionar un avatar ▪ Aceptar la operación ▪ Cancelar la operación ▪ Cerrar la vista
5. Introduce la cédula de identidad y la contraseña.	
6. Selecciona el avatar.	
7. Selecciona la opción de aceptar la operación.	
	8. Valida que los datos introducidos sean correctos.
	9. Inicia la sesión del usuario autenticado. Ver *CU Generar escritorio de trabajo.*
	10. Registra la traza del estudiante.
	11. El caso de uso termina.
Flujos Alternos	
***.a Selecciona la opción de cerrar o cancelar la vista actual en cualquier momento.**	
Acciones del actor	**Respuesta del sistema**
	*.a.1 Regresa a la vista anterior.
	*.a.2 El caso de uso termina.
3.a Selecciona como tipo de usuario "Maestro"	

Acciones del actor	Respuesta del sistema
	3.a. 1 Muestra la interfaz para introducir los datos del maestro: ▪ Cédula de identidad ▪ Contraseña Y permite ▪ Aceptar la operación ▪ Cancelar la operación ▪ Cerrar la vista
3.a. 2 Introduce la cédula de identidad y contraseña. Y selecciona la opción de aceptar la operación.	
	3.a. 3 Valida que los datos introducidos sean correctos.
	3.a. 4 Inicia la sesión del usuario autenticado. Ver *CU Generar escritorio de trabajo.*
	3.a. 5 Registra la traza del maestro.
	3.a. 6 El caso de uso termina.
3. b Selecciona como tipo de usuario "Invitado"	
Acciones del actor	**Respuesta del sistema**
	3. b.1 Inicia la sesión del usuario autenticado. *Ver CU Generar escritorio de trabajo*.
	3.b. 2 El caso de uso termina
6. a Selecciona más de un usuario	
Acciones del actor	**Respuesta del sistema**
	6.a.1 Para la cantidad de estudiantes seleccionados debe introducir: ▪ Cédula de identidad

	▪ Contraseña Y permite: ▪ Seleccionar un avatar ▪ Aceptar la operación ▪ Cancelar la operación ▪ Cerrar la vista
	6. a.2 Regresa al paso 5 del Flujo Básico
8. a La cédula introducida es incorrecta o no se encuentra registrado.	
Acciones del actor	**Respuesta del sistema**
	8. a.1 Muestra el mensaje de error “La cédula de identidad del usuario X no existe”.
	8. a.2 Regresa al paso 6 del Flujo Básico.
8. b La contraseña es incorrecta.	
Acciones del actor	**Respuesta del sistema**
	8. b.1 Muestra el mensaje de error “La contraseña del usuario X es incorrecta”.
	8. b.2 Regresa al paso 6 del Flujo Básico.
8. c No selecciona ningún avatar.	
Acciones del actor	**Respuesta del sistema**
	8. c.1 Asigna un avatar de los disponibles de forma predeterminada.
	8. c.2 Regresa al paso 6 del Flujo Básico.
8. d Existen campos vacíos.	
Acciones del actor	**Respuesta del sistema**
	8. d.1 Muestra el mensaje de información: “Existen campos vacíos”
	8. d.2 Regresa al paso 6 del Flujo Básico.

Tabla 2.1 Descripción textual del caso de uso Autenticar usuario.

Caso de uso	Consultar efemérides
Actores	Usuario
Resumen	El caso de uso se inicia cuando el actor selecciona la opción Efemérides o el botón Leer más. El sistema muestra las efemérides correspondientes a la fecha de inicio de sesión y permite especificar otra fecha para la consulta, finalizando así el caso de uso.
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado.
Referencias	RF 12 y RF 12.1

Flujo normal de eventos	
Acciones del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción Efemérides.	
	2. Muestra la lista de efemérides correspondientes a la fecha de inicio de sesión con los siguientes datos: ▪ Fecha de la efeméride ▪ Título de la efeméride Y permite ▪ Buscar efemérides especificando la fecha. ▪ Seleccionar una efeméride para ver su descripción. Ver *Sección 1 "Ver efemérides".*
3. Consulta las efemérides mostradas.	
	4. El caso de uso termina.

Flujos Alternos	
1. a Selecciona el botón Leer más.	
Acciones del actor	**Respuesta del sistema**
	1. a.1 Regresa al paso 2 del Flujo básico.
	1. a.2 El caso de uso termina.
3. a Da clic en la opción Fecha para ver las efemérides a partir de la fecha especificada.	

Acciones del actor	Respuesta del sistema
	3. a.1 Muestra el calendario para seleccionar la fecha que quiere ver las efemérides. Y permite seleccionar ▪ Mes ▪ Día
3. a.2 Selecciona el mes y el día de la fecha que quiere ver las efemérides.	
	3. a.3 Busca las efemérides correspondientes a la fecha seleccionada.
	3.a.4 Muestra el listado con las efemérides correspondientes a la fecha seleccionada.
	3. a.5 El caso de uso termina.
3. b Selecciona una fecha y no se encuentra efemérides.	
Acciones del actor	**Respuesta del sistema**
	3.b.1 Muestra el mensaje de información:"No existen efemérides para la fecha seleccionada".
	3.b.1 El caso de uso termina.
Sección 1 "Ver efeméride"	
Acciones del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción Efemérides.	
	2. Muestra la lista de efemérides correspondientes a la fecha de inicio de sesión con los siguientes datos: ▪ Fecha de la efeméride ▪ Título de la efeméride Y permite

	▪ Buscar efemérides especificando la fecha. ▪ Seleccionar una efeméride para ver su descripción.
3. Selecciona una efeméride.	
	4. Muestra la descripción de la efeméride seleccionada con los siguientes datos: ▪ Fecha de la efeméride ▪ Título de la efeméride ▪ Descripción de la efeméride ▪ Imagen si tiene Y permite ▪ Ir a la siguiente efeméride. ▪ Ir a la anterior efeméride. ▪ Ir a la última efeméride. ▪ Ir a la primera efeméride. ▪ Volver a la página de la lista de efemérides.
5. Consulta la descripción de la efeméride seleccionada.	
	6. El caso de uso termina.
Flujos Alternos	
5. a Selecciona la opción de ir a la siguiente efeméride.	
Acciones del actor	**Respuesta del sistema**
	5. a.1 Muestra la siguiente efeméride según el orden de la lista de efemérides.
	5. a.2 Regresa al paso 4 del flujo básico.
5. b Selecciona la opción de ir a la anterior efeméride.	
Acciones del actor	**Respuesta del sistema**
	5. b.1 Muestra la anterior efeméride según el orden de la

	lista de efemérides.
	5. b.2 Regresa al paso 4 del flujo básico.
5. c Selecciona la opción de ir a la última efeméride.	
Acciones del actor	**Respuesta del sistema**
	5. c.1 Muestra la última efeméride según el orden de la lista de efemérides.
	5. c.2 Regresa al paso 4 del flujo básico.
5. d Selecciona la opción de ir a la primera efeméride.	
Acciones del actor	**Respuesta del sistema**
	5. d.1 Muestra la primera efeméride según el orden de la lista de efemérides.
	5. d.2 Regresa al paso 4 del flujo básico.
5. e Selecciona la opción de ir volver a la lista de efemérides.	
Acciones del actor	**Respuesta del sistema**
	5. e.1 Muestra la lista de las efemérides correspondientes a la fecha de inicio de sesión.
	5. e.2 El caso de uso termina.

Tabla 2.3 Descripción textual del caso de uso Consultar efemérides.

Caso de uso	Consultar noticias
Actores	Usuario
Resumen	El caso de uso se inicia cuando el actor selecciona la opción Noticias o el botón Leer más. El sistema muestra un listado con las noticias publicadas, finalizando así el caso de uso.
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado.
Referencias	RF 13 y RF 13.1
Flujo normal de eventos	
Acciones del actor	**Respuesta del sistema**

1. El caso de uso se inicia cuando el actor selecciona la opción Noticias.	
	2. Muestra un listado de las noticias con los siguientes datos cada una: ▪ Título ▪ Fecha de publicación ▪ Síntesis Y permite ▪ Seleccionar una noticia para ver su información. Ver *Sección 1"Ver noticias".*
3. Consulta las noticias mostradas.	
	4. El caso de uso termina.
Flujos Alternos	
1. a Selecciona el botón Leer más.	
Acciones del actor	**Respuesta del sistema**
	1. a.1 Regresa al paso 2 del Flujo básico.
	1. a.2 El caso de uso termina.
Sección 1 "Ver noticia"	
Acciones del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción Noticias.	
	2. Muestra un listado de las noticias con los siguientes datos cada una: ▪ Título ▪ Fecha de publicación ▪ Síntesis

	Y permite ▪ Seleccionar una noticia para ver su información.
3. Selecciona una noticia.	
	4. Muestra la descripción de la noticia seleccionada con los siguientes datos: ▪ Título de la noticia. ▪ Descripción de la noticia. ▪ Fecha de publicación. ▪ Autor de la noticia. ▪ Imagen si tiene. Y permite: ▪ Ir a la siguiente noticia. ▪ Ir a la anterior noticia. ▪ Ir a la última noticia. ▪ Ir a la primera noticia. ▪ Volver a la página de la lista de noticias.
5. Consulta la información mostrada.	
	6. El caso de uso termina.
Flujos Alternos	
5. a Selecciona la opción de ir a la siguiente noticia.	
Acciones del actor	**Respuesta del sistema**
	5. a.1 Muestra la siguiente noticia según el orden de la lista de noticias.
	5. a.2 Regresa al paso 4 del flujo básico.
5. b Selecciona la opción de ir a la anterior noticia.	
Acciones del actor	**Respuesta del sistema**
	5. b.1 Muestra la noticia anterior según el orden de la lista de noticias.
	5. b.2 Regresa al paso 4 del flujo básico.

5. c Selecciona la opción de ir a la última noticia.	
Acciones del actor	**Respuesta del sistema**
	5. c.1 Muestra la última noticia según el orden de la lista de noticias.
	5. c.2 Regresa al paso 4 del flujo básico.
5. d Selecciona la opción de ir a la primera noticia.	
Acciones del actor	**Respuesta del sistema**
	5. d.1 Muestra la primera noticia según el orden de la lista de noticias.
	5. d.2 Regresa al paso 4 del flujo básico.
5. e Selecciona la opción de ir volver a la lista de noticias.	
Acciones del actor	**Respuesta del sistema**
	5. e.1 Muestra la lista de las noticias publicadas.
	5. e.2 El caso de uso termina.

Tabla 2.4 Descripción textual del caso de uso Consultar noticias.

Conclusiones

En este capítulo se realizó una propuesta de sistema para el módulo General de la colección El Navegante, la cual está constituida por diecisiete casos de uso. Con el objetivo de representar cómo será la interacción de los actores con el nuevo sistema se elaboró el diagrama de casos de uso y se describen estos. El trabajo realizado reúne las funcionalidades que los usuarios necesitan para acceder a los productos de la colección y servicios comunes presentes en esta. Además, constituye la propuesta de sistema que se utilizará como entrada fundamental para la construcción del software, de esta manera, se garantizará que se cumplan con las expectativas del cliente.

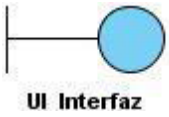
Capítulo 3 Construcción de la solución

Una vez identificados los requisitos, se analizan y se describen en lenguaje del desarrollador con el objetivo de conseguir una comprensión más precisa y a partir de ellos desarrollar una especificación de la aplicación, que sea fácil de mantener y que ayude a estructurar el sistema. El análisis de un sistema proporciona una visión abstracta del mismo y constituye el punto de partida para el diseño, donde se modela el sistema para que soporte todos los requisitos, adquiriendo una comprensión en profundidad de los aspectos relacionados con los requisitos no funcionales, lenguajes de programación, sistemas operativos y las entradas a las actividades de la implementación.

3.1 Modelo de análisis

En el modelo de análisis se analizan y se refinan los requisitos identificados, para conseguir una comprensión más precisa de ellos y una descripción de los mismos que ayude a estructurar el sistema. Es descrito para el desarrollador, para que pueda comprender cómo debería ser diseñado e implementado el sistema. Es un modelo abstracto, en el que no se especifica el lenguaje en el que se va a implementar.

En el Modelo de Análisis se identifican los siguientes estereotipos de clases:



Modelan la interacción entre el sistema y sus actores.

Coordinan la realización de uno o varios casos de uso controlando las actividades de los objetos que implementan la funcionalidad del caso de uso.

Modelan información que posee larga vida y que es a menudo persistente.

Diagrama de clases del análisis

El diagrama de clases del análisis constituye una vista estática de las clases que conforman el Modelo del Análisis y las asociaciones entre las mismas. Con la información recopilada, luego de identificar las clases que describen la realización de los casos de uso, los atributos y las relaciones entre ellos, se construye el diagrama de clases del análisis.

A continuación se muestran los diagramas de clases del análisis para algunos de los casos de uso del sistema, para consultar los restantes diagramas ver los Anexos sección Anexo 2 Diagramas de clases del análisis.

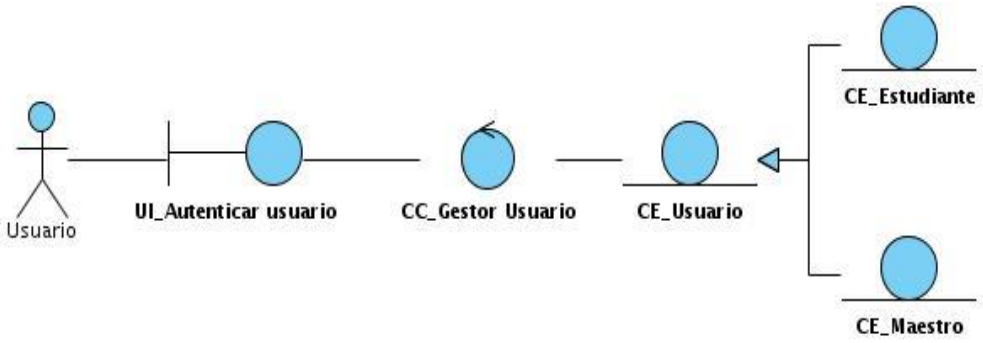


Figura 3.1 Diagrama de clases del análisis del CU Autenticar usuario.

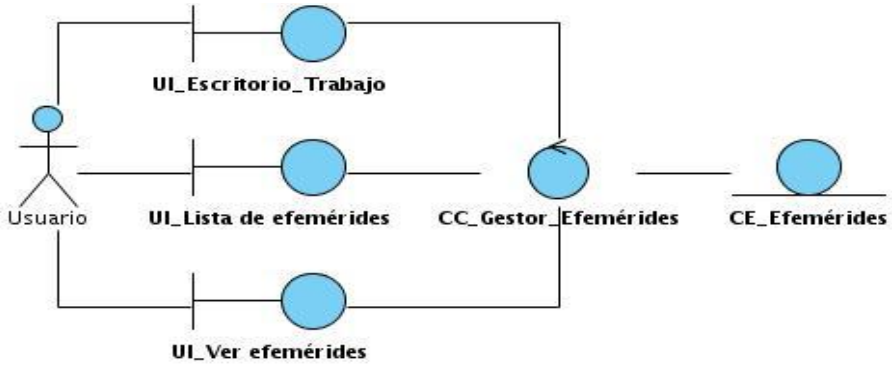


Figura 3.3 Diagrama de clases del análisis del CU Consultar efemérides.

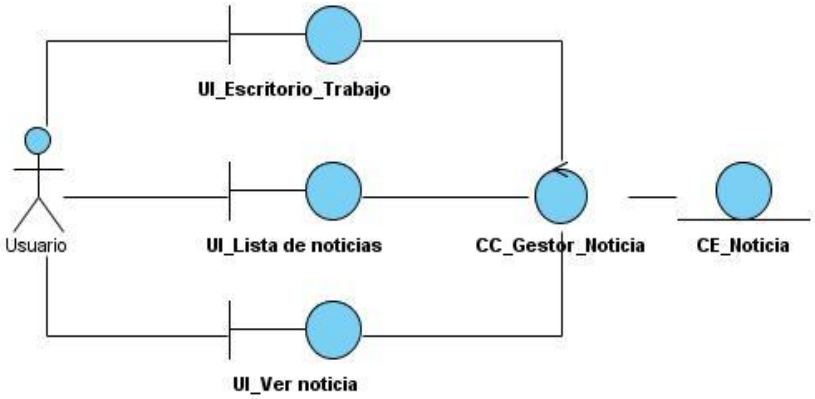


Figura 3.4 Diagrama de clases del análisis del CU Consultar noticias.

3.2 Modelo de Diseño

El modelo de diseño es un modelo de objeto que describe las realizaciones de los casos de uso y sirve como una abstracción del modelo de implementación y su código fuente. El diseño es un refinamiento del análisis que tiene en cuenta los requisitos no funcionales, es decir, cómo cumple el sistema sus objetivos. Es utilizado como una actividad esencial de entrada para la implementación.

Patrones de diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí, adaptada para resolver un problema de diseño general en un contexto particular. Son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software. Son utilizados principalmente en los modelos de clases y sus relaciones. (30)

Para el desarrollo del módulo General se utilizan los patrones de diseño de Symphony: Alta Cohesión y Controlador perteneciente al patrón GRASP y Decorator como patrón GoF. Del lado del cliente con JavaScript se utiliza el patrón de diseño Mediador.

Mediador

Define un objeto que encapsula cómo interactúan un conjunto de objetos. El mediador estimula la pérdida de acoplamiento ocultando las referencias explícitas entre los objetos, permitiendo variar su interacción de forma independiente. (31)

Se utiliza el patrón mediador debido a que en el desarrollo de aplicaciones es necesario asegurar que los objetos se puedan comunicar con otros sin tener que incluir referencias directas de código en sus clases, esto propicia el bajo acoplamiento[2]. La idea básica es, que cada objeto se comunique con el mediador base, el que conoce todos los objetos y cómo manipular su estado cuando un evento dado es reportado. De esta forma se evita que en cada uno de los objetos existan códigos específicos dentro de un método asociado con el evento. En su lugar se escribe un código genérico en la clase mediador, propiciando que si se utiliza un objeto mediador, se puede reemplazar el código especifico por el del padre.

[2] Es la idea de tener las clases lo menos relacionadas 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 clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

Diagramas de clases del diseño

Un diagrama de clases del diseño representa las clases del sistema con sus relaciones estructurales y de herencia. En las aplicaciones web, el diagrama de clases del diseño representa las colaboraciones que ocurren entre las páginas, donde cada página lógica puede ser representada como una clase.

En los diagramas de clases del diseño del módulo General se hacen uso de paquetes para distribuir y mostrar las responsabilidades de las clases según el patrón Modelo Vista Controlador. Los paquetes apps, html y model agrupan las clases controladoras, las interfaces de usuarios y las tablas de la base de datos respectivamente.

En el caso del subsistema Componentes de Symphony, contiene todos los componentes utilizados para la comunicación entre una capa y otra; componentes como: ficheros y clases internas. El subsistema Doctrine, es la representación de los componentes que utiliza el ORM Doctrine para la generación de la base de datos. El paquete js, modela la clase JavaScript relacionado con las vistas mostradas.

A continuación se muestran los diagramas de clases del diseño para algunos de los casos de uso del sistema, para consultar los restantes diagramas ver los Anexos sección Anexo 3 Diagramas de clases del diseño.

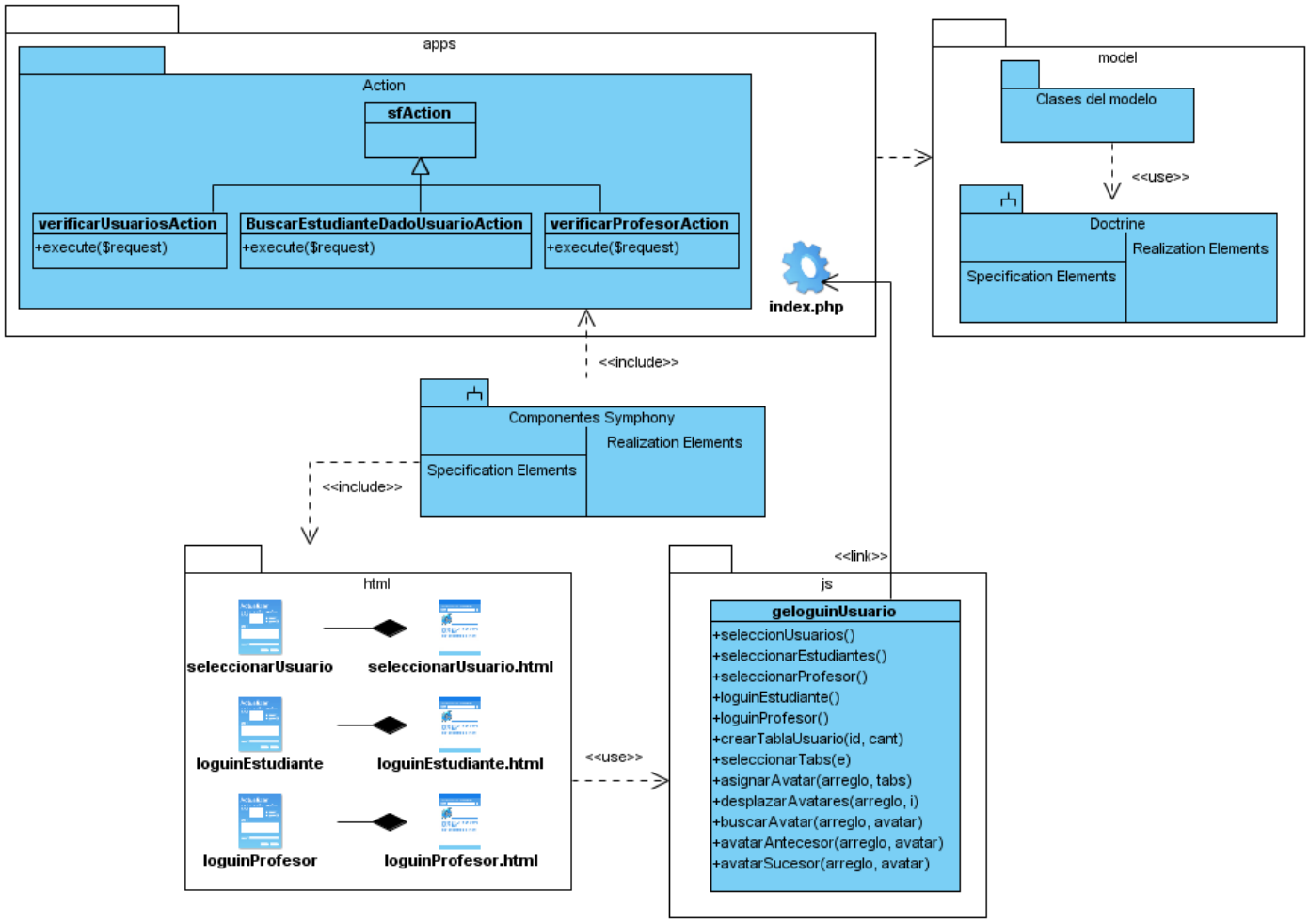


Figura 3.5 Diagrama de clases del diseño del CU Autenticar usuario.

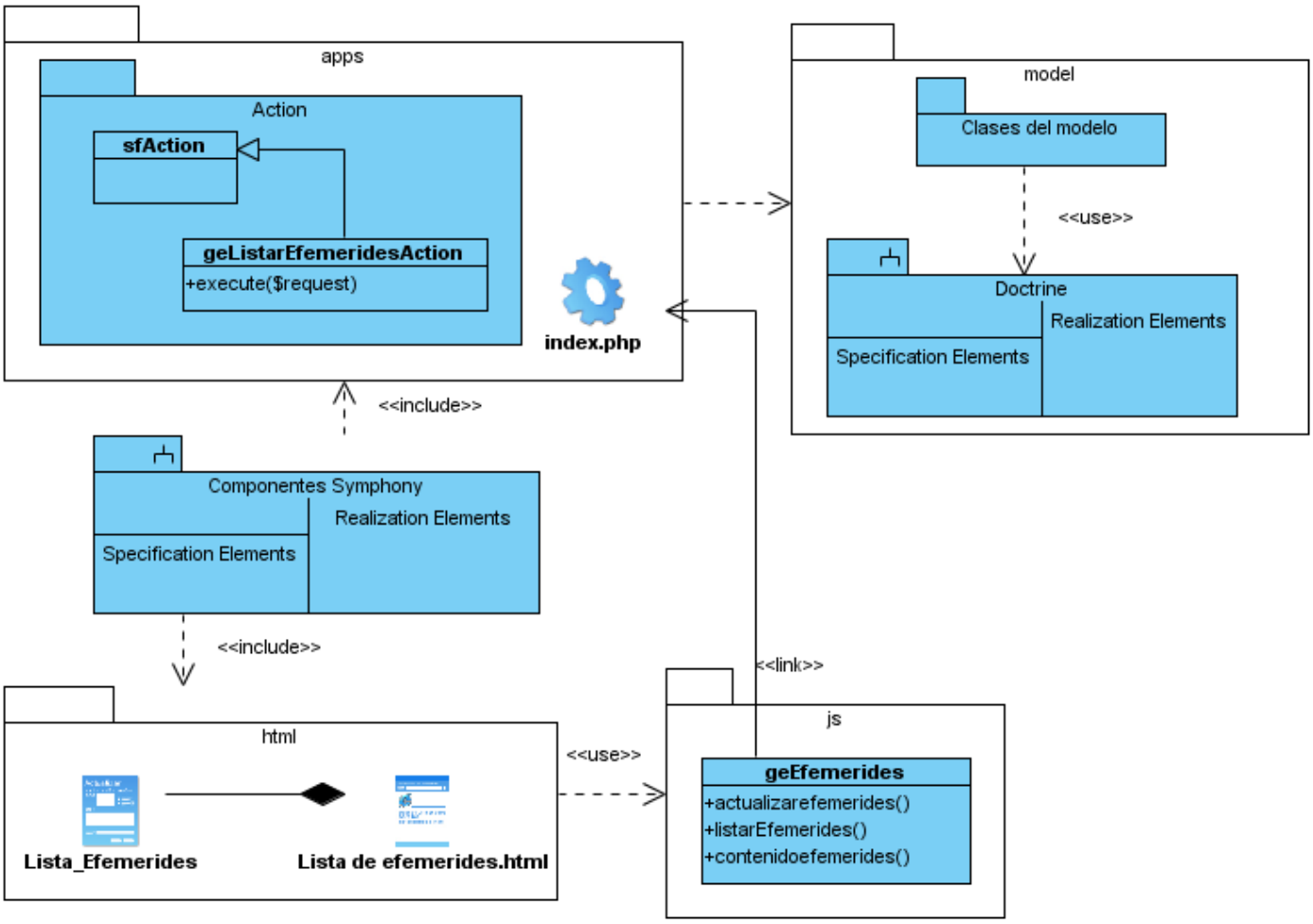


Figura 3.7 Diagrama de clases del diseño del CU Consultar efemérides.

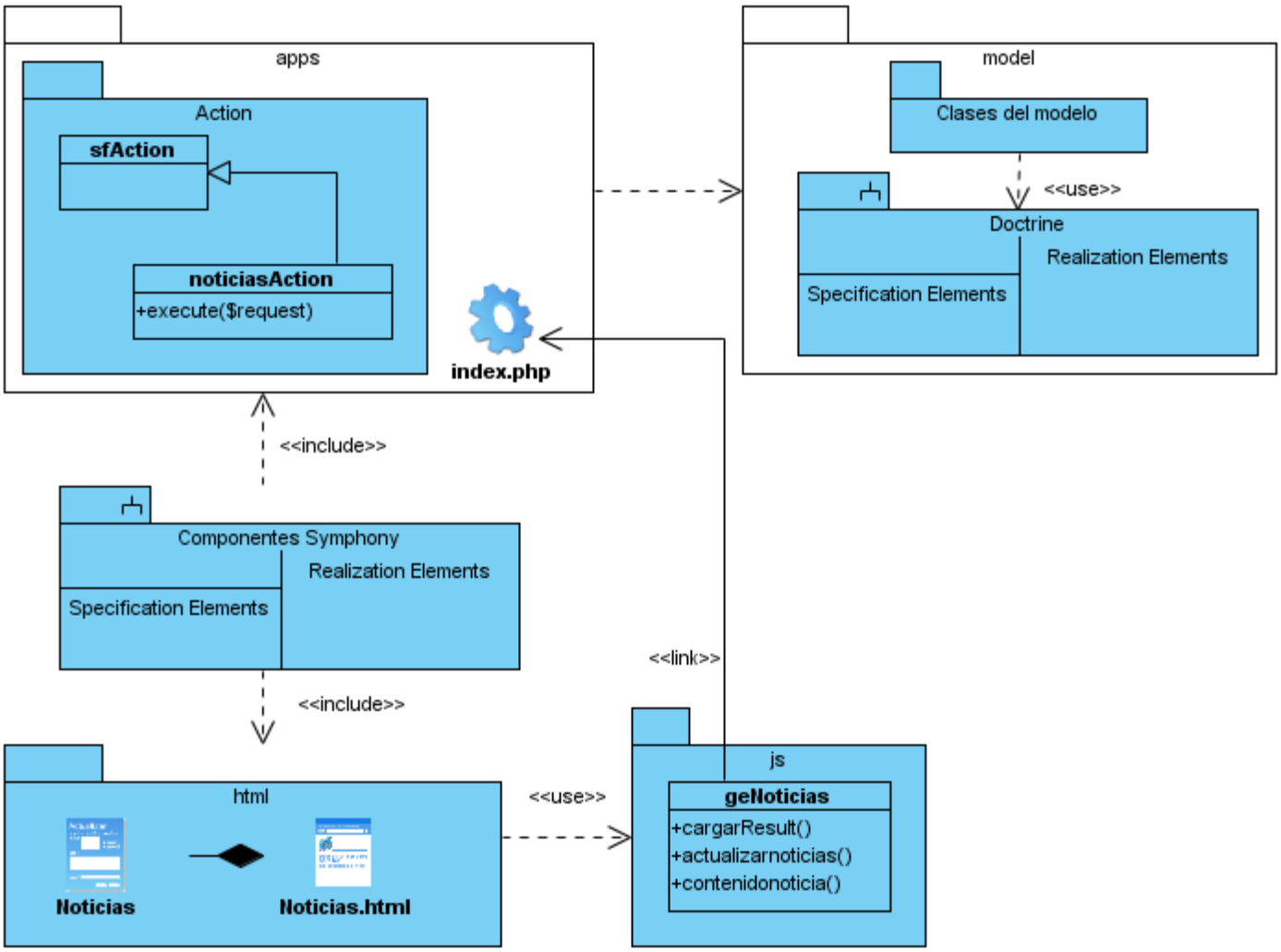


Figura 3.8 Diagrama de clases del diseño del CU Consultar noticias.

En el siguiente paquete se muestran todas las clases JavaScript que van a ser utilizadas en la implementación del módulo General, así como, su relación con la clase mediador y con las librerías a utilizar en la implementación.

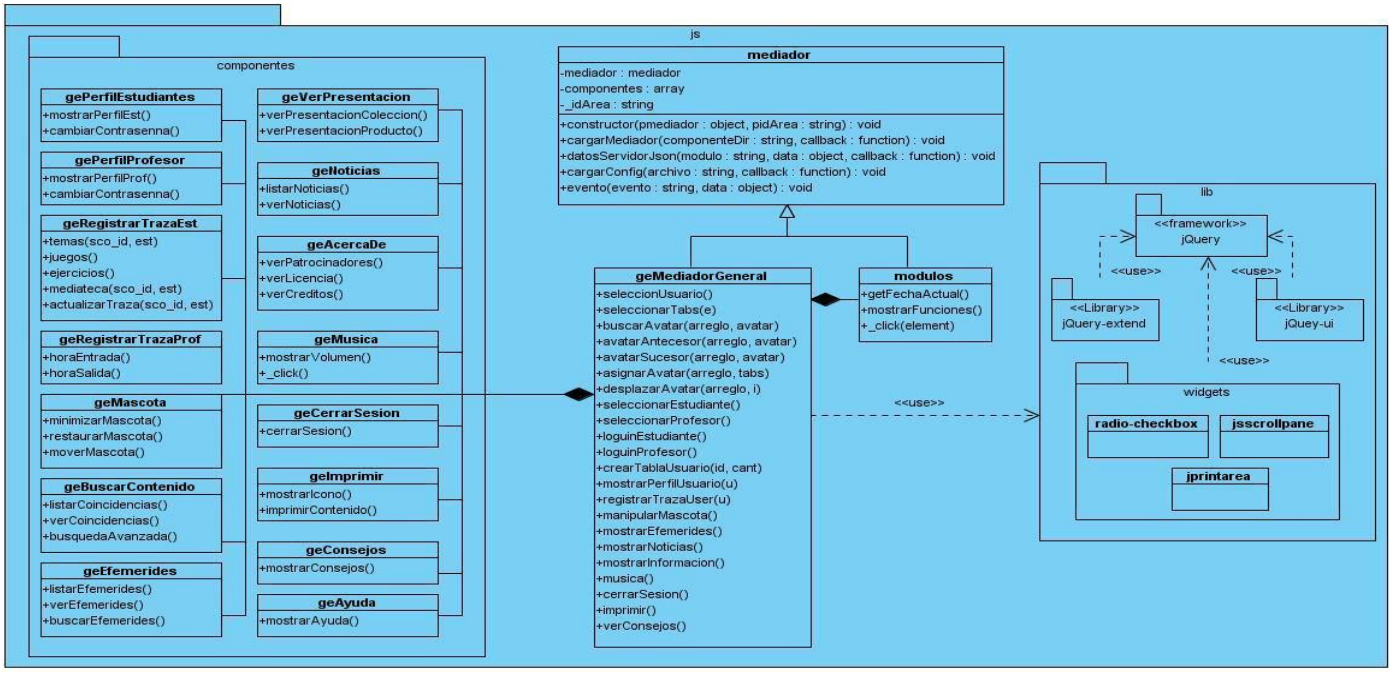


Figura 3.9 Subsistemas de clases JavaScript.

3.2 Diseño de la base de datos

El diseño de la base de datos juega un papel fundamental para cualquier aplicación que gestione información; se hace necesario que los datos se almacenen de forma coherente y organizada, para evitar que dicha información se pierda. El principal objetivo para realizar el diseño de la base de datos, es lograr la persistencia de los datos de los usuarios, así como la información de los recursos que se gestionan en el software.

Clases persistentes

Las clases persistentes son aquellas clases capaces de guardar su estado en un medio permanente; lo cual está dado por el almacenamiento físico y permanente de la información de la clase, para la copia de seguridad en caso del fracaso del sistema, o para el intercambio de información.

A continuación se muestra las clases persistentes del módulo:

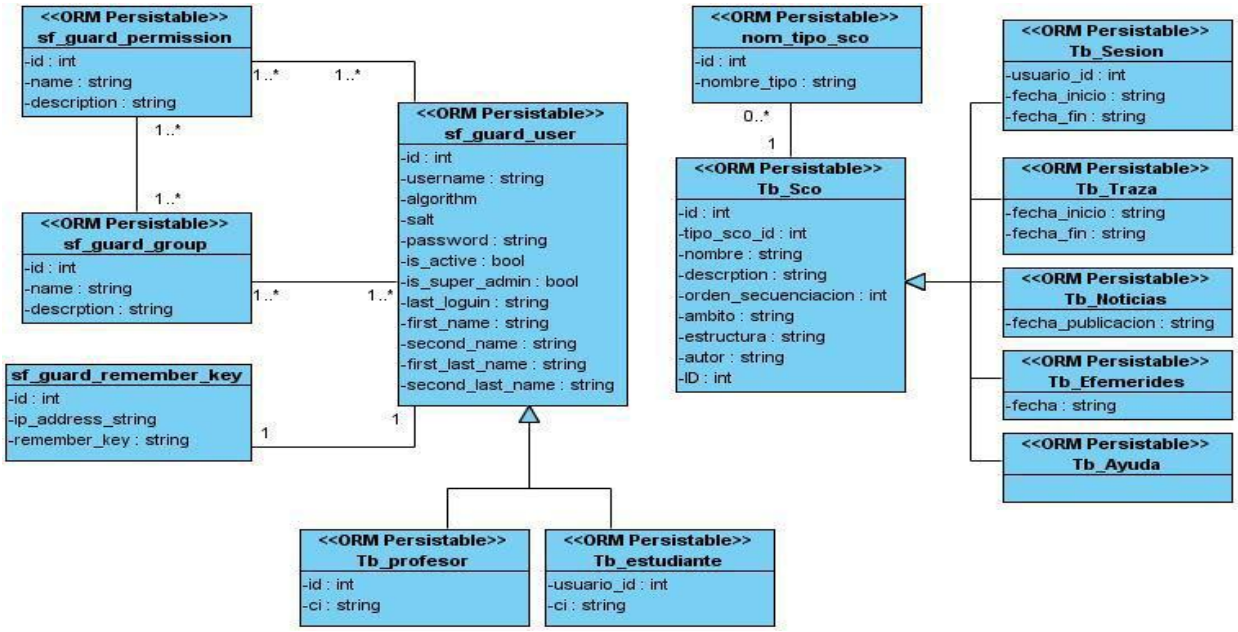
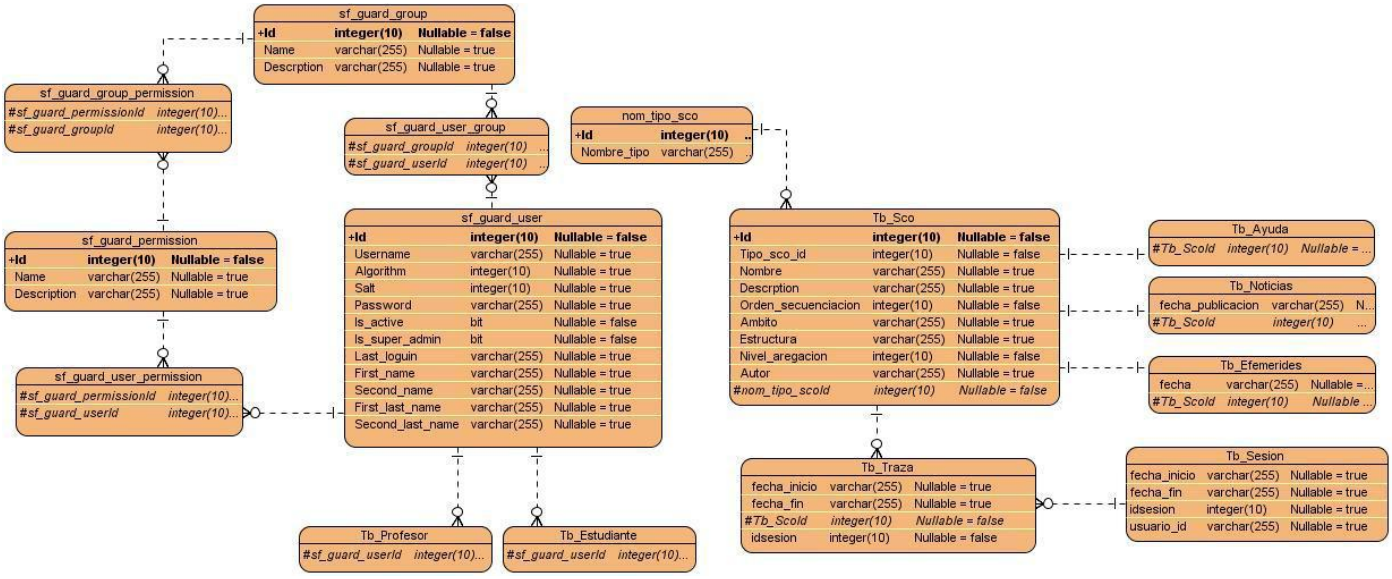


Figura 3.10 Diagrama de clases persistentes.

Modelos de datos

Un modelo de datos permite representar las estructuras presentes en la base de datos, su tipo y la forma en que se relacionan. De igual forma permite detallar las operaciones de manipulación de dichos datos, típicamente: de agregado, borrado, modificación y recuperación.

A continuación se muestra el modelo de datos correspondiente a las clases persistentes identificadas:



Conclusiones

En este capítulo se transformaron los requerimientos en un diseño de clases que estructura como debe quedar el sistema haciendo uso de los patrones definidos para el mismo, lo cual contribuye a que dicho diseño se corresponda con el entorno de implementación, diseñando sus funcionalidades y los patrones de diseño se utilizaron para diseñar correctamente y en menos tiempo el módulo.

Capítulo 4 Implementación y Prueba

Después de haber realizado el análisis y diseño, el equipo de desarrollo tiene una visión más clara de cómo estará estructurado el módulo General. Es entonces que el camino queda listo para empezar a implementar el sistema como un todo, garantizando que cada funcionalidad cumpla con el diseño establecido. Posteriormente, deben desarrollarse las pruebas para verificar que el producto desarrollado cumpla con las especificaciones para el mismo.

4.1 Diagrama de paquetes

El diagrama de paquetes ofrece un nivel de abstracción que permite dividir el sistema en agrupaciones de elementos denominados paquetes. Ayudando a distribuir entre los desarrolladores la implementación de cada uno de ellos. (32)

Se decidió dividir el módulo en tres paquetes, donde cada paquete representa una de las capas del patrón MVC.

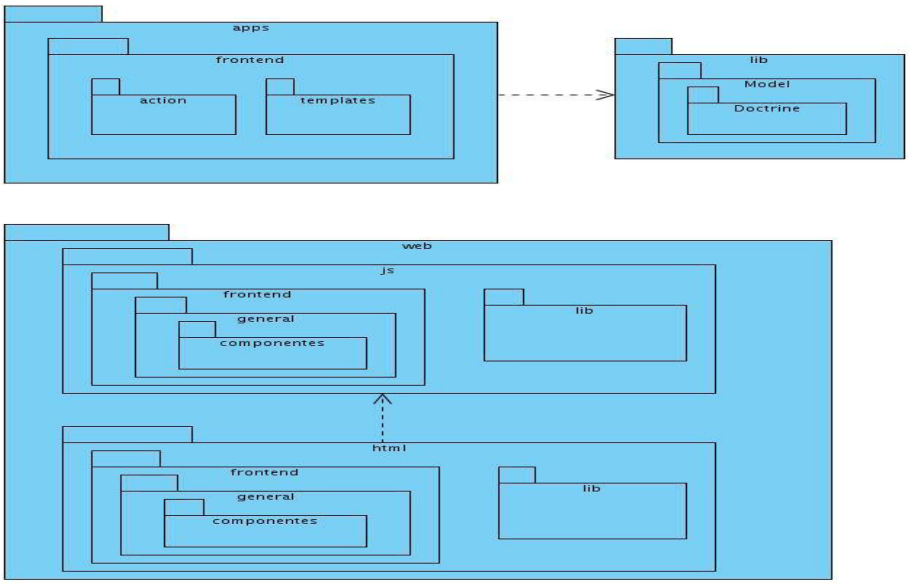


Figura 4.1 Diagrama de paquete del módulo General

4.2 Estándares de codificación

Los estándares de codificación permiten una mejor integración entre las líneas de código y establecen pautas que conlleven a lograr un código más legible y reutilizable, de tal forma que logren aumentar su mantenibilidad a lo largo del tiempo.

La mantenibilidad del código es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento. (33)

Los estándares para el desarrollo de la colección han sido definidos por la arquitectura del proyecto Colecciones de Software Multisaber y El Navegante, debido a esto se hacen mención de algunos y los restantes se pueden consultar en el Anexo 4 Estándares de codificación.

4.2.1 Para los lenguajes PHP y JavaScript se definieron los siguientes estándares.

Indentación y espacios en blanco

- Alineación

Se utiliza dos espacios como unidad de alineación en vez de utilizar la tecla TAB

- Longitud de la línea

Se evita utilizar líneas con más de 80 caracteres de longitud, estas no son bien manejadas por algunas herramientas y terminales.

- Líneas plegadas

Cuando una expresión no cabe en una línea simple debido a su extensión se divide en más de una línea, siguiendo las siguientes precisiones:

- Dividir después de una coma.
- Dividir después de un operador.
- Alinear la nueva línea al inicio de la expresión en el mismo nivel que la línea anterior.

4.3 Modelo de implementación

El Modelo de implementación es el artefacto más significativo del flujo de trabajo de Implementación, este tiene gran importancia porque mediante su uso los desarrolladores entienden claramente el funcionamiento del sistema antes de comenzar a escribir las líneas de código. Este modelo está conformado por los Diagrama de Componentes y de Despliegue. Su objetivo fundamental es desarrollar la arquitectura y el sistema como un todo, describir cómo las clases del Modelo de Diseño se implementan en términos de componentes tales como ejecutables, ficheros de código fuentes o tablas de una base de datos. (32)

Diagrama de componentes

Los Diagramas de Componentes muestran tanto los componentes software (código fuente, binario y ejecutable) como las relaciones lógicas entre ellos en un sistema. Representan todos los tipos de elementos del software implicados en la fabricación de aplicaciones informáticas.

El siguiente diagrama es la representación del módulo General dividido según la arquitectura del MVC. En el paquete apps se hace referencia a la capa Controlador que muestra la relación entre el paquete de componentes de las Action y la de las Success.

El paquete web, la capa Vista en el patrón MVC, se representan los paquetes de componentes de las páginas HTML y las clases JavaScript, por último en el paquete Modelo se representa los paquetes de componentes de las tablas que componen el módulo General.

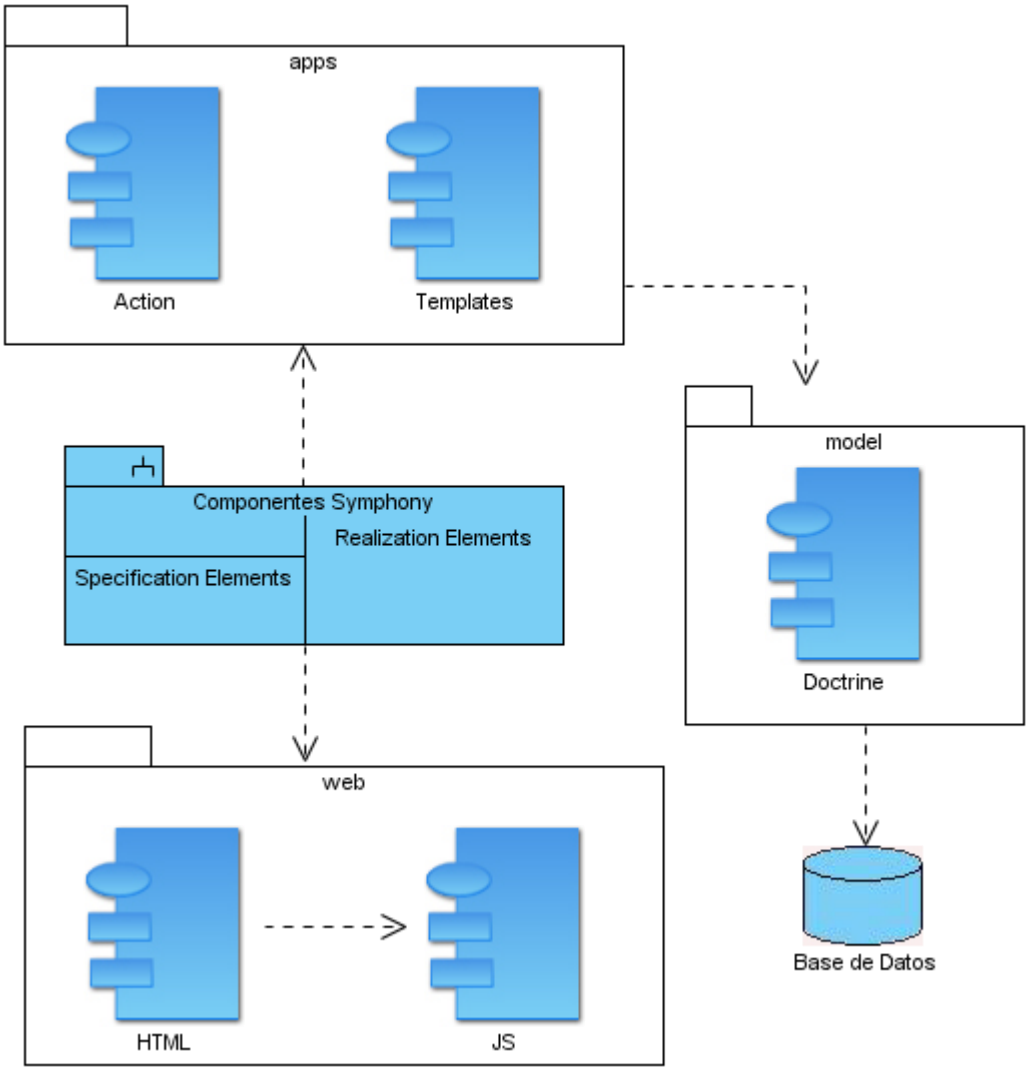


Figura 4.1 Diagrama de componentes del módulo General

Los siguientes diagramas representan los componentes de cada uno de los paquetes descritos anteriormente.

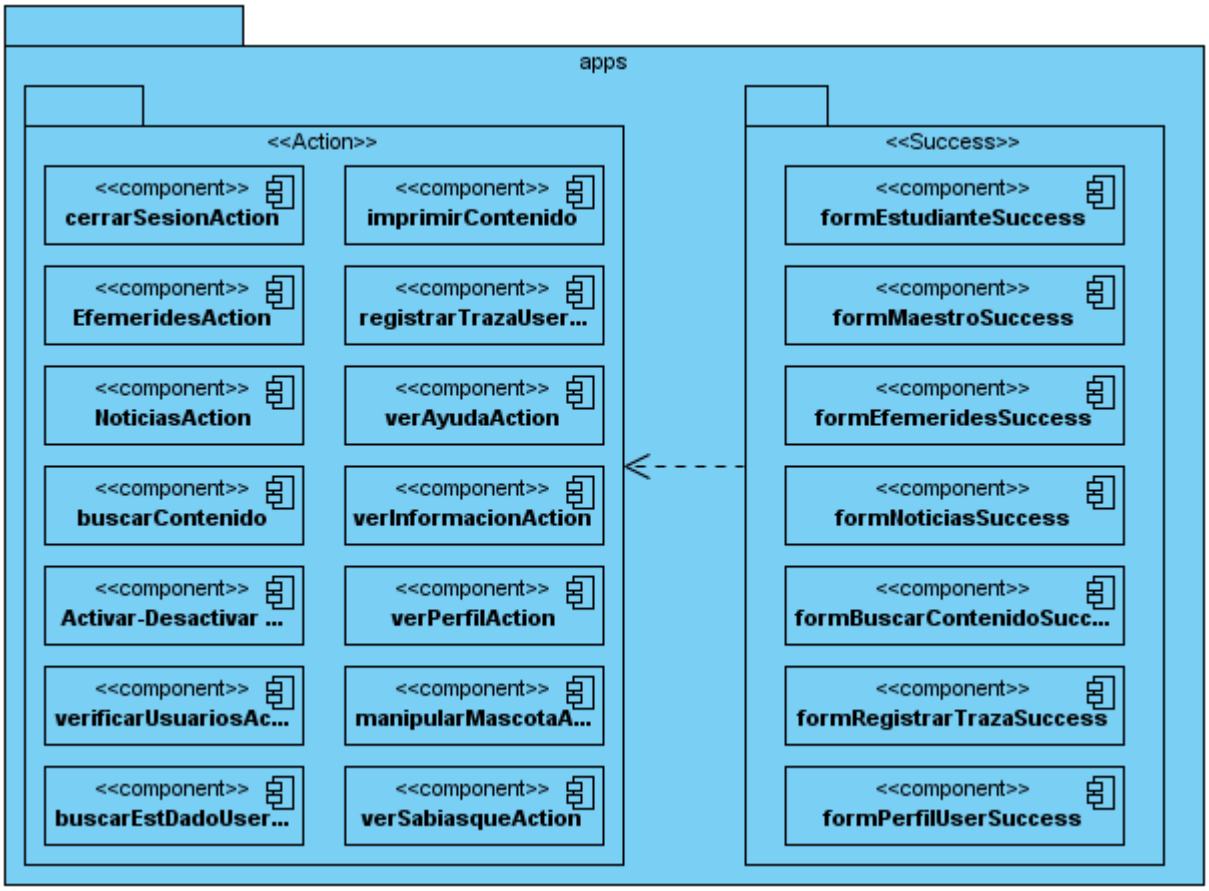


Figura 4.2 Diagrama de componentes Subsistema Controlador

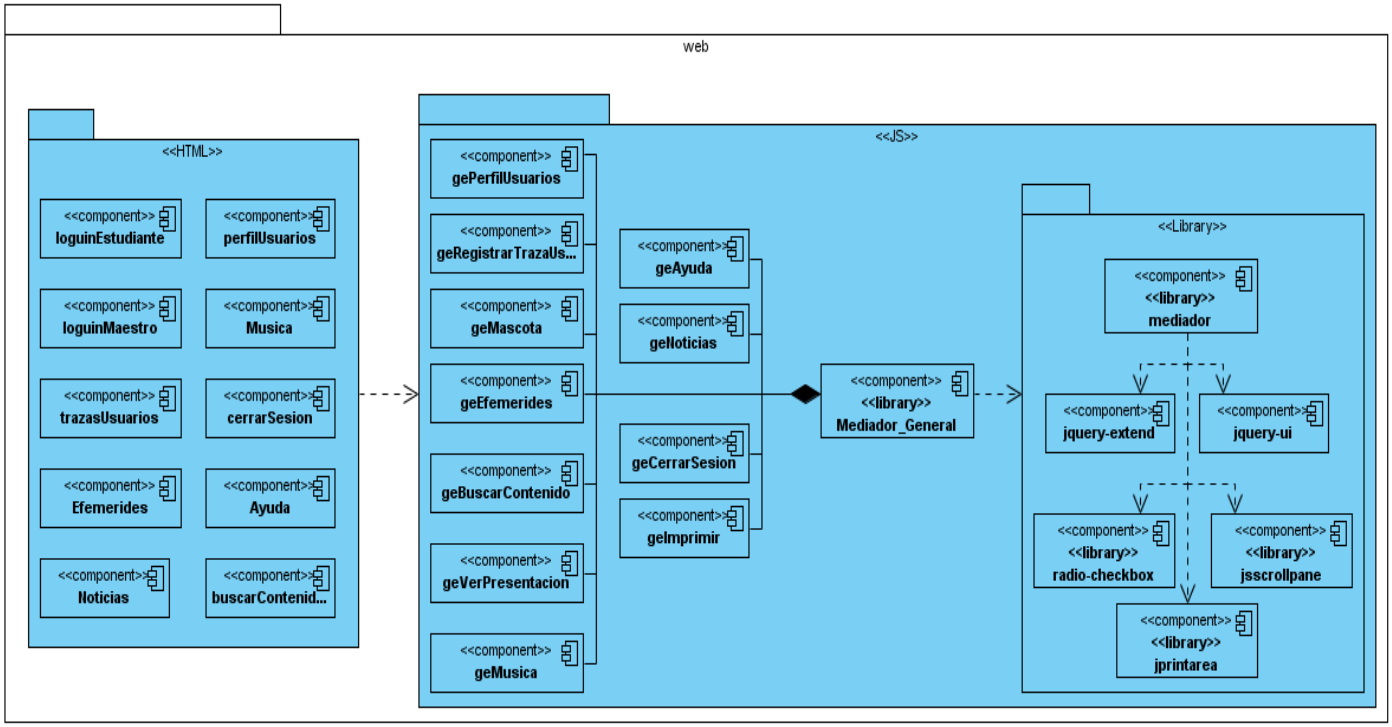


Figura 4.3 Diagrama de componentes Subsistema Vista

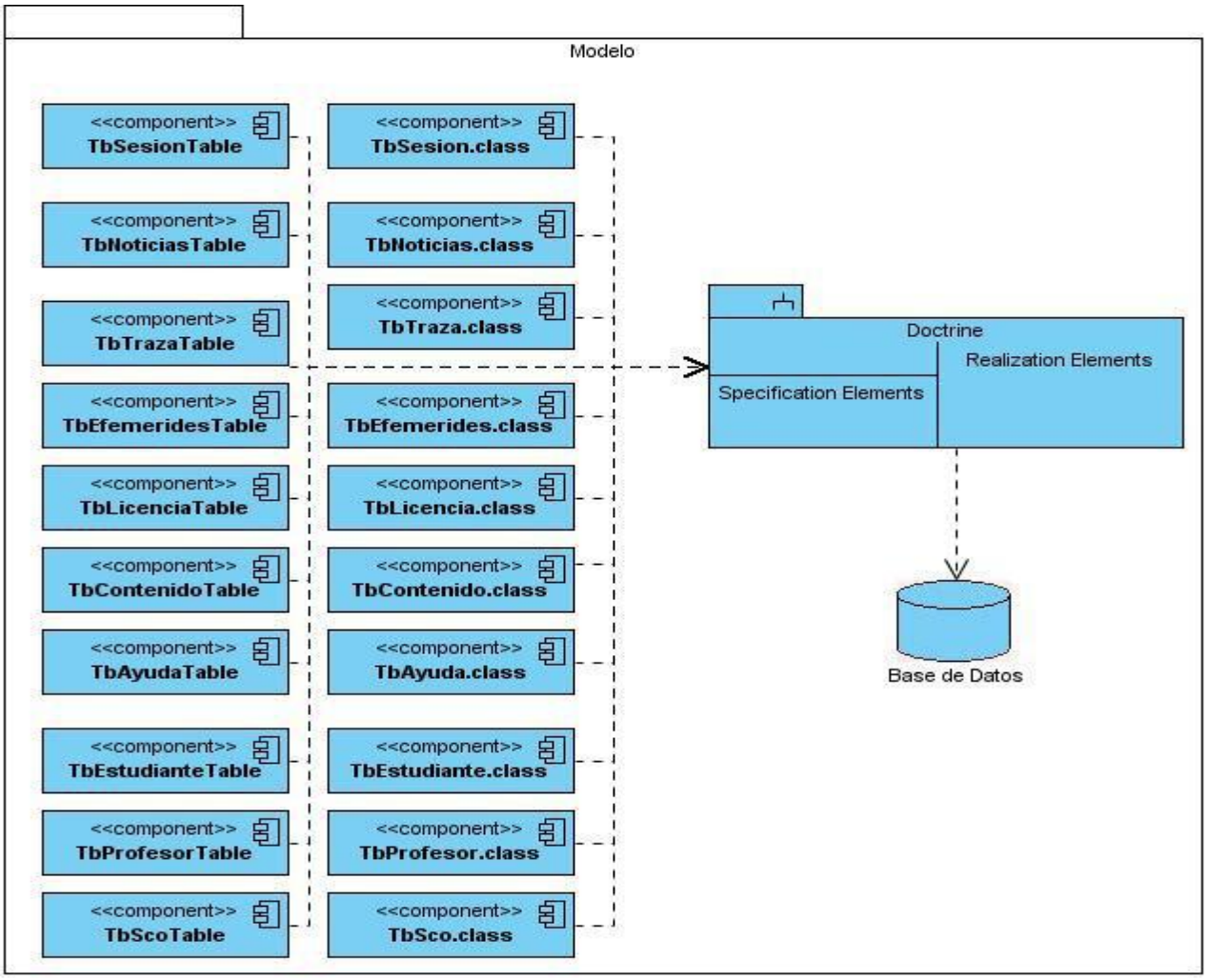


Figura 4.4 Diagrama de componentes Subsistema Modelo

Diagrama de despliegue

Un diagrama de despliegue muestra las relaciones físicas entre los componentes de hardware y de software, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes de software. Es una colección de nodos; donde cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo de hardware similar. (32)

A continuación se muestran las dos posibles formas de desplegar el software y cómo estarán distribuidos los nodos y dispositivos para la explotación del mismo.

Variante 1:

Se necesita de un ordenador que funcione como servidor, el cual tendrá instalado el servidor web Apache y el servidor de base de datos PostgreSQL donde se guardará la información generada por la interacción de los usuarios con los productos de El Navegante. Además, se necesita de una computadora que haga función de cliente.

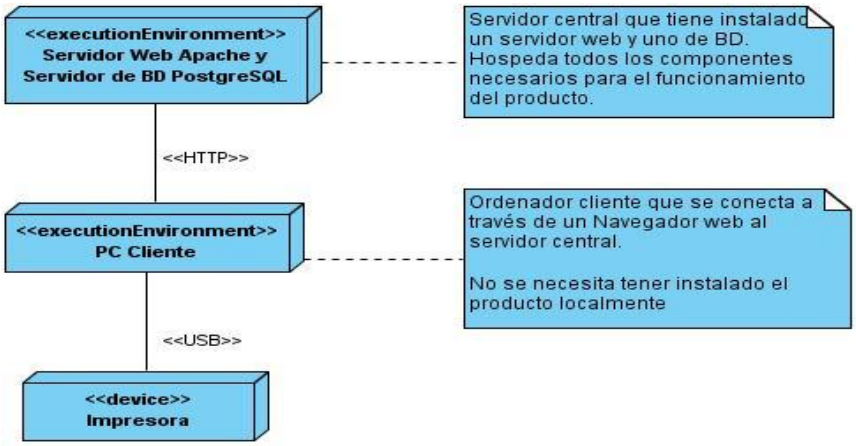
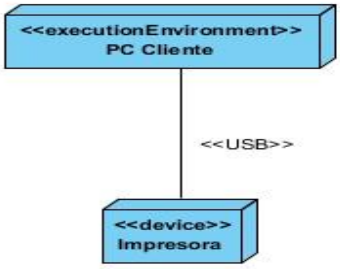


Figura 4.5 Diagrama de despliegue

Variante 2:

Instalar el servidor web Apache, servidor de base de datos PostgreSQL y la aplicación en cada una de las PC Cliente.



El sistema permite la impresión de contenidos del módulo Temas, de reportes de de las trazas de los estudiantes y de los términos del glosario, razón por la cual en ambas variantes se necesita de una impresora como dispositivo conectado a las PC cliente.

PC cliente: Ordenador Cliente que se conecta a través de un navegador web al servidor central donde reside la Colección. No se necesita tener instalado el producto localmente.

Descripción de elementos e interfaces de comunicación

<<HTTP>> Representa la conexión que se va a establecer entre una PC Cliente que se va a conectar con el servidor central donde reside el Servidor Web y BD, en otras palabras, significa la conexión entre el navegador de usuario y el servidor del sistema.

<<USB>> Conexión que existe entre la impresora y la PC Cliente del usuario. Se modela con USB, pero puede existir otro tipo, todo depende del tipo de impresora que esté usando el usuario y el tipo de conexión que utilice ésta.

4.4 Prueba

Las pruebas son un conjunto de actividades en las cuales un sistema es ejecutado bajo condiciones o requerimientos específicos, donde los resultados son observados y registrados para dar una evaluación de algún aspecto del sistema y determinar así la calidad del mismo. En ellas se describen cómo comprobar cada versión operacional del sistema, verificando que todos los requerimientos hayan sido implementados y determinando sus defectos. Es importante aclarar que las pruebas no pueden asegurar la ausencia de defectos; solo pueden demostrar que existen defectos en el software. (34)

Métodos de pruebas

Los métodos de prueba definen qué estrategia seguir en cuanto a la verificación y validación del sistema, ya que están diseñados con el propósito de descubrir fallos y no para demostrar que el software funciona. Existen dos métodos de pruebas: Pruebas de caja blanca, estas pruebas comprueban los caminos lógicos dentro del sistema y son derivadas a partir de las especificaciones internas del diseño o del código. Otro de los métodos son las Pruebas de caja negra las cuales pretenden demostrar que las entradas al software se aceptan de forma adecuada y se produce el resultado correcto. (35)

Las pruebas de caja negra son las que se llevan a cabo sobre la interfaz del software, donde los casos de prueba pretenden demostrar que las funciones del software son operativas, que las entradas se aceptan de la forma adecuada, que se produce el resultado correcto, y determinar errores en la interfaz y las funciones visibles al cliente. Dada la necesidad de que las pruebas abarquen los requerimientos, las funciones y las respuestas de la aplicación, se decide utilizar las pruebas de caja negra.

Un caso de prueba es un conjunto de entradas de pruebas, condiciones de ejecuciones y resultados esperados desarrollados para cumplir un objetivo en particular o una función esperada. No solo tienen que ser escritos para entradas válidas y esperadas, sino también condiciones no válidas e inesperadas. Para la realización de los casos de pruebas es necesario un número de datos que ayuden a la ejecución de los mismos y que permitan que el sistema se ejecute en todas sus variantes, existen varias técnicas que permiten el desarrollo de estos casos:

Técnica de la Partición de Equivalencia: esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. (34)

Técnica del Análisis de Valores Límites: esta técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables. (34)

Técnica de Grafos de Causa-Efecto: es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones. (34)

Considerando que el módulo no requiere de muchos datos de entrada la técnica que se propone para realizar los casos de pruebas es la variante de particiones equivalentes, esta técnica consta de 3 pasos fundamentales:

1. Para cada caso de uso, generar un sistema completo de escenarios.
2. Identificar los casos de pruebas.
3. Identificar valores de datos para las pruebas.

En el primer paso se identifican todas las combinaciones posibles de caminos de ejecución del caso de uso, cada combinación será un escenario de uso. En el segundo paso se identifican las variables que hacen que los escenarios se ejecuten, estas variables se representan en una matriz donde se representa el comportamiento esperado del sistema. Una vez identificados los casos de pruebas se deben validar para identificar posibles redundancias o ausencia de funcionalidades.

Resultado de las pruebas

Para evaluar la solución desarrollada se planificaron 4 iteraciones de pruebas en las cuales se probó la aplicación con un alto grado de detalle. Se abarcaron los métodos y técnicas de pruebas expuestas anteriormente en cada una de las iteraciones, arrojando resultados visibles, los cuales validan la calidad del producto construido. A continuación se muestran la cantidad de No Conformidades identificadas en las iteraciones realizadas:

Clasificaciones	1ra iteración	2da iteración	3ra iteración	4ta iteración	5ta iteración
Alta	6	4	4	1	0
Media	7	4	2	2	0
Baja	25	19	15	9	2

Se muestran de forma detallada un gráfico de los errores encontrados durante las cuatro iteraciones realizadas:

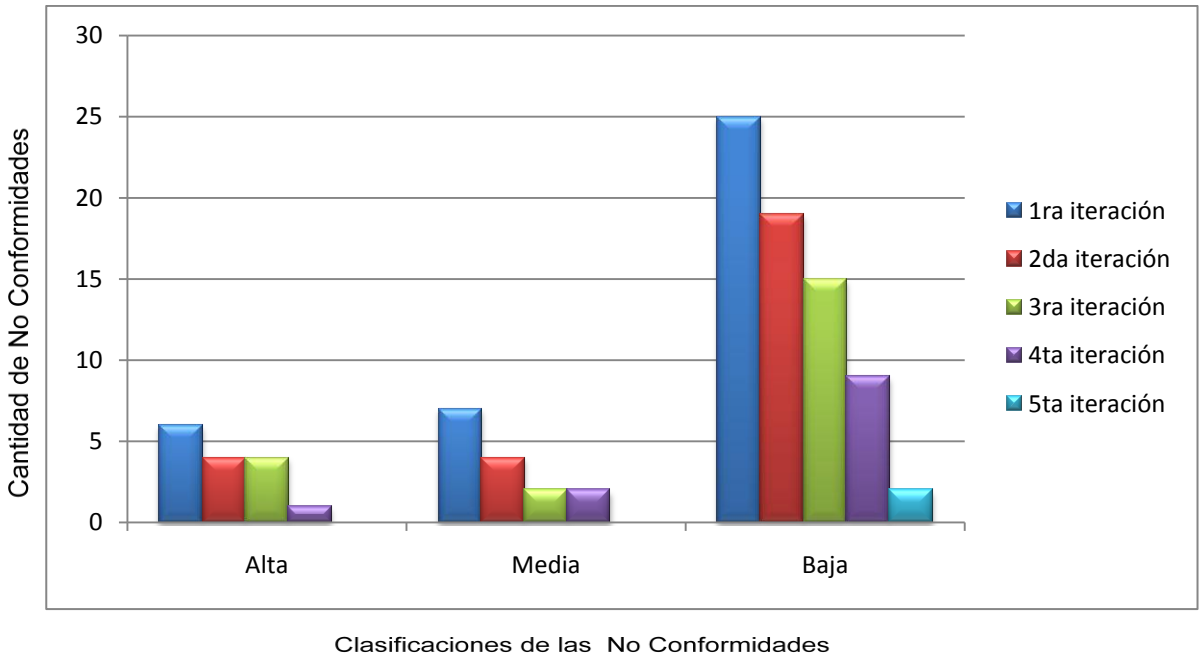


Figura 4.6. No Conformidades detectadas durante las Pruebas

Luego de concluida cada iteración de pruebas se analizaron, por parte del equipo de desarrollo, las No Conformidades encontradas para determinar cuáles realmente constituyeron defectos del sistema o presentaban alguna variación según la descripción de los casos de uso y necesitaban ser solucionadas. Las pruebas se realizaron de forma iterativa e incremental, comprobando en cada iteración que hubiesen sido corregidos los errores detectados en la iteración anterior, lo que contribuyó a mejorar la calidad y funcionalidad del software. Para consultar como fueron diseñados los casos de pruebas, ver el Anexo 5 CP CU Autenticar usuario.

Conclusiones

En este capítulo se representa el sistema a partir de sus componentes de implementación y del modelo físico en el que será implantado el mismo. Tras el análisis de los resultados expuestos en este capítulo, se demuestra la solidez de la propuesta, basada en el paso por 5 iteraciones de pruebas las cuales denotaron un número decreciente de No Conformidades encontradas, lo cual se corresponde con la calidad de la aplicación construida.

Conclusiones generales

- Se alcanzó un mejor entendimiento de las características del sistema, mediante la realización del modelo de dominio, la identificación de los requisitos funcionales y no funcionales así como la elaboración del diagrama de casos de uso del sistema y la descripción detallada de cada uno de estos.

- A través del análisis y diseño se obtuvieron los diagramas de clases que describen cómo implementar el sistema.

- La utilización de los frameworks jQuery y Symphony unido a los patrones de diseño permitieron el desarrollo rápido y con calidad del módulo.

- Se obtuvo el módulo General el cual permite la autenticación de los usuarios al sistema, ver las informaciones del producto, las noticas y efemérides publicadas en el software. Además, permite la impresión y búsqueda de contenido.

Recomendaciones

A partir del trabajo realizado se sugieren las siguientes recomendaciones al proyecto Colecciones de Software Educativo Multisaber y El Navegante:

- Realizar las pruebas de integridad para verificar la calidad del código.

- Publicar este trabajo a disposición de la comunidad universitaria como referencia para proyectos similares.

- Realizar encuestas entre los usuarios del proyecto y ajenos a él para determinar la popularidad e impacto de cada funcionalidad, con el objetivo de aumentar la utilidad de la aplicación.

Referencias bibliográficas

1. Coordinación General de Investigación y Desarrollo de Modelos Educativos. [En línea] [Citado el: 2 de noviembre de 2010.] http://investigacion.ilce.edu.mx/panel_control/doc/c36,evaluacsoft.pdf.

2. **Marquès, Pere.** El software educativo. [En línea] Universidad Autónoma de Barcelona. [Citado el: 2 de noviembre de 2010.] http://www.lmi.ub.es/te/any96/marques_software/.

3. **Galvis Panqueva, Álvaro.** SlideShare Inc. [En línea] [Citado el: 3 de noviembre de 2010.] http://www.slideshare.net/algalvis50/ingeniera-de-software-educativo-1992-parte-1-fundamentos.

4. **Matas i Dalmau, Toni.** PRODUCTOS MULTIMEDIA: DISEÑO Y ANÁLISIS CONCEPTUAL. [En línea] julio de 1996. [Citado el: 5 de noviembre de 2010.] http://www.bcnmultimedia.com/CAS_eines/articulos/ArticuloTMAnalisis.htm.

5. **Graells, Pere Marquès.** MULTIMEDIA EDUCATIVO: CLASIFICACIÓN, FUNCIONES, VENTAJAS, DISEÑO DE ACTIVIDADES. [En línea] 3 de agosto de 2010. [Citado el: 5 de noviembre de 2010.] http://peremarques.pangea.org/funcion.htm.

6. Conceptos y definiciones de hipertexto. [En línea] http://ldc.usb.ve/~abianc/hipertexto.html.

7. **María Jesús Lamarca Lapuente.** Hipermedia/Multimedia. *Hipertexto: El nuevo concepto de documento en la cultura de la imagen.* [En línea] [Citado el: 20 de noviembre de 2010.] http://www.hipertexto.info/documentos/hipermedia.htm. 7.

8. Pedagogía a tu alcance. [En línea] http://patalcance.rimed.cu/module/biblioteca/datos_esp.php?pag=glo&num=244&Tip=ssb&idMod=7.

9. Colección de Software Educativo Multisaber. *Características generales de la colección multisaber.* [En línea] [Citado el: 20 de enero de 2011.] http://blogs.rimed.cu/multisaber/.

10. **M. Sc. Orestes Coloma Rodríguez.** [En línea] [Citado el: 2 de diciembre de 2010.] http://www.forumcyt.cu/UserFiles/forum/Textos/1101687.pdf. 8.

11. **Ivar Jacobson, Grady Booch y Grady Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* s.l. : Pearson Educación, S.A, 2000. 9.

12. —. *El Lenguaje Unificado de Modelado.* s.l. : Addison Wesley, 2000.

13. Sitio de descargas de software. [En línea] [Citado el: 10 de diciembre de 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

Referencias bibliográficas

14. Guía Breve de XHTML. [En línea] [Citado el: 15 de febrero de 2011.] http://www.w3c.es/divulgacion/guiasbreves/XHTML.

15. **Javier Eguíluz Pérez.** Introducción a CSS. [En línea] [Citado el: 12 de enero de 2011.] http://librosweb.es/css/capitulo1.html.

16. **Damián Pérez Valdés .** Maestros del web. *Los diferentes lenguajes de programación para la web.* [En línea] [Citado el: 12 de enero de 2011.] http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/.

17. **Javier J. Gutiérrez.** [En línea] [Citado el: 14 de enero de 2011.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

18. no hay Limites. [En línea] [Citado el: 14 de enero de 2011.] http://www.nohaylimites.com/?p=141.

19. maestros del web. [En línea] [Citado el: 14 de enero de 2011.] http://www.maestrosdelweb.com/editorial/javascript-facil-y-rapido-con-jquery/.

20. librosweb. *Symfony en pocas palabras.* [En línea] [Citado el: 16 de enero de 2011.] http://www.librosweb.es/symfony/capitulo1/symfony_en_pocas_palabras.html.

21. symfony. *¿Qué es Symfony?* [En línea] [Citado el: 16 de enero de 2011.] http://www.symfony.es/que-es-symfony/.

22. **Ramiro Lago .** [En línea] Abril de 2007. [Citado el: 19 de enero de 2011.] http://www.proactiva-calidad.com/java/patrones/mvc.html.

23. [En línea] [Citado el: 11 de enero de 2011.] http://petra.euitio.uniovi.es/~i1667065/HD/documentos/Entornos%20de%20Desarrollo%20Integrado.pdf.

24. NetBeans. [En línea] Oracle Corporation and/or its affiliates, 2011. [Citado el: 19 de enero de 2011.] http://netbeans.org/community/releases/68/index_es.html.

25. Computer Audio Video Systems Integrator. *CAVSI.* [En línea] [Citado el: 17 de enero de 2011.] http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.

26. UOC OpenCourseWar. [En línea] 2008. [Citado el: 15 de diciembre de 2011.] http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02152.pdf.

27. desarrolloweb.com. [En línea] [Citado el: 13 de diciembre de 2010.] http://www.desarrolloweb.com/articulos/1112.php.

28. ciberaula. [En línea] 2010. http://linux.ciberaula.com/articulo/linux_apache_intro/.

29. IEEE Standard Glossary of Software Engineering Terminology. [En línea] [Citado el: 22 de febrero de 2011.] http://www.computing.dcu.ie/~nbrophy/ca222/week4_6/som1.pdf.

30. **Joaquin Gracia.** IngenieroSoftware. [En línea] mayo de 2005. [Citado el: 17 de marzo de 2011.] http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

31. **Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides.** *Design Patterns, Elements of Reusable Object-Oriented Software.*

32. Entorno Visual de Aprendizaje. [En línea] http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_8/Conferencia_6/Materiales_Basicos/Implementacion.pdf.

33. Connexions. [En línea] [Citado el: 7 de junio de 2011.] http://cnx.org/content/m17457/latest/.

34. Entorno Visual de Aprendizaje. [En línea] http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_9/Conferencia_7/Materiales_Basicos/Sobre_la_disciplina_de_Prueba.pdf.

35. Métodos de Prueba de Software. [En línea] [Citado el: 10 de mayo de 2011.] http://pruebasoftware.blogcindario.com/2005/10/00001-metodos-de-prueba-del-software.html..