Universidad de las Ciencias Informáticas

Facultad 8

Título: Desarrollo de un módulo para la gestión de contenidos en la versión multiplataforma de la colección El Navegante

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

Autor: Jorge Martínez Padrón

Tutor: Lic. Héctor Matías González

Co-Tutor: Ing. Mairelis Gari Maribona

A mi mamá, a mi papá, a mi familia.

Agradecimientos

Le agradezco a todas las personas que colaboraron con la realización de este trabajo: a mis tutores Héctor Matías y Mairelis Gari, al colectivo de profesores del proyecto Multisaber, en especial a la profesora Liana Isabel, a la gente de Alfaomega, sobre todo a Abel y a Geyser y a mis compañeros de aula y de trabajo.

A mi mamá y a mi papá, por el apoyo y el ejemplo.

A todos los profesores que de alguna manera han apoyado mi formación profesional y personal.

A la UCI, por haberme hecho sentir parte del momento histórico que vive mi país.

A los buenos amigos que he encontrado en esta universidad.

Y a Jita, claro.

Resumen

La versión multiplataforma de la colección El Navegante surge ante la necesidad de eliminar una serie de limitaciones que impiden la utilización de la colección El Navegante desarrollada por el MINED en el entorno venezolano. Uno de los módulos que componen la nueva versión es el módulo Temas; encargado de apoyar el proceso de creación, edición y visualización de la información referente a cada uno de los temas tratados en los productos que integran la colección.

El objetivo de este trabajo es desarrollar un módulo que garantice la gestión de contenidos en la versión multiplataforma de la colección El Navegante. Para lograrlo, primeramente se realizó un estudio sobre los sistemas de gestión de contenido y las principales tendencias en la gestión de contenidos para el aprendizaje. Seguidamente se transitó, siguiendo la metodología de desarrollo RUP, por los flujos de trabajo Modelamiento del negocio, Requisitos, Análisis y diseño e Implementación; obteniéndose en cada caso la mayoría de los artefactos propuestos. Por último se demostró la integración del módulo dentro de la línea base de la arquitectura.

Palabras claves: sistema, software educativo, colección El Navegante, multiplataforma.

Índice

Introducción

Un software educativo es un sistema informático destinado a apoyar el proceso de enseñanza-aprendizaje. Este tipo de sistemas ha demostrado ser tan efectivo, que un ordenador es un medio casi imprescindible en el proceso de enseñanza en la actualidad. Cuba, que siempre se ha preocupado por el nivel y la calidad de su sistema de educación, ha desarrollado varios software educativos destinados a diversos niveles de escolaridad. Entre los productos nacionales más sobresalientes se encuentran las colecciones Multisaber, destinada a los alumnos de la enseñanza primaria, El Navegante, para la enseñanza secundaria y Futuro para la enseñanza preuniversitaria.

En el marco del Convenio Integral de Cooperación Cuba-Venezuela, se firmó un acuerdo para la adecuación de los diez productos que componen El Navegante con la intención de utilizarlos en los Liceos Bolivarianos. Cuando se analizó la colección se detectaron una serie de limitaciones que se contraponían a las necesidades del cliente; entre ellas pueden citarse la falta de un mecanismo para crear y actualizar el contenido, la utilización de herramientas propietarias y la imposibilidad de ejecutar el software en un sistema operativo diferente de Windows. Ante la naturaleza y magnitud de los cambios a realizar se decidió desarrollar una versión de El Navegante para cumplir con los requerimientos del cliente. Desde el punto de vista estructural, la versión multiplataforma está compuesta por siete módulos: Temas, Ejercicios, Resultados, Mediateca, Juegos, Profesor y General. Específicamente el módulo Temas es el encargado de apoyar el proceso de creación, edición y visualización de la información referente a cada uno de los temas tratados en un producto, siendo representada a través de una combinación de texto, imágenes, sonidos y videos.

Teniendo en cuenta lo anteriormente planteado, el problema de investigación queda definido de la siguiente manera: ¿Cómo garantizar la gestión de contenidos en la versión multiplataforma de la colección El Navegante para que cumpla con sus requerimientos asociados?

El trabajo propone como objetivo general desarrollar un módulo para garantizar la gestión de contenidos en la versión multiplataforma de la colección El Navegante.

Por tanto, el objeto de estudio de la presente investigación será los sistemas informáticos para la gestión de contenidos.

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

- Realizar el estado del arte del tema tratado.
- Definir los requisitos del módulo.
- Efectuar el análisis y diseño del módulo.
- Implementar el módulo.
- Validar la integración de la solución.

El campo de acción que abarca la presente investigación es el proceso de desarrollo de software para la gestión de contenidos de la versión multiplataforma de la colección El Navegante.

Al finalizar esta investigación se espera que la versión multiplataforma de la colección El Navegante cuente con un módulo que garantice la gestión de contenidos.

Para satisfacer los objetivos específicos se planifican las siguientes tareas de investigación:

- Comparar los procesos de gestión de contenidos de sistemas informáticos similares.
- Analizar la versión anterior de la colección El Navegante para identificar los requisitos generales.
- Consultar a los encargados de definir las características de la versión multiplataforma de la colección El Navegante.
- Investigar la tecnología a utilizar.
- Identificar la estructura de la base de datos.
- Identificar las clases necesarias para la implementación del módulo.
- Implementar las clases necesarias para el funcionamiento del módulo.
- Comprobar el funcionamiento del módulo dentro de la línea base del producto general.

Los métodos científicos que serán utilizados en la investigación, serán los siguientes:

Métodos Teóricos

Análisis Histórico-Lógico: Este método se utiliza para analizar la historia de la gestión de contenido, los principales conceptos que giran alrededor de la misma y los sistemas que ya existen relacionados con este proceso.

Inductivo-Deductivo: Se utiliza este método para estudiar en detalle la tecnología a utilizar.

Modelación: Una vez realizada la investigación, y analizada la información, este método es muy útil para realizar los modelos correspondientes al ciclo de vida del software, por lo que brinda facilidades a la hora de cumplir con las tareas de análisis y diseño de los procesos que intervendrán en la aplicación, así como para implementar el sistema.

Método Empírico

Observación: Este método tiene relación con la mayoría de las tareas propuestas. Se puede observar cómo funciona el proceso de gestión de contenidos y los principales problemas cuando se realiza el mismo.

Estructura capitular

Capítulo I: Este capítulo recoge la fundamentación teórica de la investigación y muestra el estado del arte del tema tratado a nivel nacional e internacional. Se argumenta además la selección de las herramientas y tecnologías a utilizar para cumplir con el objetivo de esta investigación

Capítulo II: Se identifican los principales conceptos presentes en el negocio de la versión multiplataforma de la colección El Navegante y se identifican los requisitos asociados al módulo Temas. Además, se definen y describen los casos de uso del sistema.

Capítulo III: En este capítulo se realiza el modelado del análisis y el diseño, identificando y modelando elementos necesarios para el desarrollo del módulo.

Capítulo IV: Se documenta el proceso de implementación del sistema a través del modelo de implementación. Además se muestra cómo se integra el módulo Temas dentro de la línea base de la arquitectura.

Capítulo 1. Fundamentación teórica

1.1 Introducción

En el presente capítulo se analiza el proceso de gestión de contenidos en sistemas especializados en esta actividad y las principales tendencias en cuanto a la gestión de contenidos para el aprendizaje. Vale señalar que esta investigación no se realiza con el objetivo de seleccionar una herramienta existente para satisfacer las necesidades del cliente, sino para tomar ideas, adoptar buenas prácticas y conocer los principales conceptos sobre este tema; esto se debe fundamentalmente a que el resultado del presente trabajo formará parte de una aplicación realizada a la medida con características muy especificas.

Se argumentan además las decisiones tomadas por el equipo de arquitectura del proyecto al cual pertenece el sistema a desarrollar, con respecto a las herramientas y tecnologías que serán utilizadas para cumplir con el objetivo general de esta investigación.

1.2 Sistemas de gestión de contenidos

“Un Sistema de gestión de contenidos (Content Management System, en inglés, abreviado CMS) permite la creación y administración de contenidos (o información), principalmente en páginas web.” (1)

Cuando se trata el tema de la gestión de contenidos no pueden dejarse de analizar los CMS. Desde hace algunos años representan la solución más sencilla y económica para desarrollar un sitio web totalmente funcional y muy atractivo. En general, proporcionan un sistema estructural preestablecido apoyado por una sección de administración, donde después de algunas configuraciones, quedan satisfechos la mayoría de los requerimientos de los clientes. Según el *Open Source CMS award* patrocinado por la editorial *Packt Publishing* entre los CMS más utilizados se encuentran WordPress, Drupal y Joomla!.

1.2.1 WordPress

WordPress es un CMS desarrollado con PHP, diseñado para utilizar el gestor de base de datos MySQL y liberado bajo licencia GPL. Por su facilidad de uso, su licencia y su gran comunidad de desarrolladores y diseñadores logró convertirse en el CMS libre más utilizado en el año 2009 (Según la editorial *Packt*

Publishing). Para la gestión de contenidos establece tres estructuras organizativas fundamentales, la categoría, la etiqueta y el artículo. Dentro de los artículos se encuentra toda la información a publicar y estos pueden ser agrupados por categorías y por etiquetas. Las categorías y las etiquetas se encuentran al mismo nivel, es decir, no se pueden etiquetar las categorías ni categorizar las etiquetas. La idea es facilitar la búsqueda de un artículo en específico agrupándolo en categorías y etiquetándolo.

1.2.2 Drupal

Drupal está desarrollado con PHP bajo licencia GPL. Para la interacción con la base de datos incorpora una capa de abstracción que permite utilizar MySQL o PostgreSQL indistintamente. Es un excelente CMS para desarrollar aplicaciones web con una gran cantidad de plugins y temas visuales. Todo el contenido en Drupal se maneja como un nodo, lo que permite trabajar con una serie de datos comunes entre ellos. Un nodo puede ser toda una página y una página puede estar formada por varios nodos. El contenido está organizado a través de una taxonomía muy flexible que permite relaciones jerárquicas o lineales.

1.2.3 Joomla!

Este CMS está desarrollado con PHP para el gestor de base de datos MySQL bajo licencia GPL. Por sus características de usabilidad, de flexibilidad, de rendimiento y su excelente panel de administración fue seleccionado para desarrollar la versión multiplataforma de la colección Multisaber. El contenido está organizado en una estructura jerárquica bastante rígida que se divide en secciones, categorías y artículos. Todos los artículos pertenecen a una sola categoría y una categoría pertenece a una sección. Los artículos representan las páginas y pueden contener imágenes, videos, etcétera.

1.3 Gestión de contenidos para el aprendizaje.

En el entorno de la gestión de contenidos para el aprendizaje *online* existen dos conceptos fundamentales: los LMS (*Learning Management System*, en español Sistemas de Gestión de Aprendizaje) y los LCMS (*Learning Content Management System*, en español Sistemas de Gestión de Contenidos de Aprendizaje). Los LMS son plataformas que se encargan fundamentalmente de administrar, distribuir y controlar el proceso de enseñanza en línea. Aunque por lo general permiten la creación de objetos de aprendizaje simples, se especializan en tareas como la gestión y registro de cursos y alumnos, el control y

seguimiento de los mismos y la generación de reportes; importando objetos de aprendizajes construidos en otras herramientas para elaborar el contenido que presentan. Los LCMS son sistemas que permiten la creación, administración y visualización de objetos de aprendizaje y cursos. Siguiendo la filosofía de los CMS, estos sistemas gestionan el contenido que utilizan y algunos incluso pueden exportarlo.

Para lograr la interoperabilidad de los objetos de aprendizaje se utiliza usualmente el estándar SCORM (*Sharable Content Object Reference Model*, en español Modelo de Referencia para Objetos de Contenidos Intercambiables).

1.3.1 El estándar SCORM

SCORM surge como parte de una iniciativa de la organización ADL (*Advanced Distributed Learning*) de reunir en una solución las propuestas de otras organizaciones sobre la estandarización de los objetos de aprendizaje.

El modelo SCORM es un conjunto de especificaciones y estándares elaborados por distintos organismos que se postula como el modelo común para los objetos de aprendizaje. (12)

Utilizando el estándar SCORM se propician cuatro características fundamentales:

- Durabilidad: Es la característica destinada a impedir la caducidad tecnológica de los contenidos y de los estándares.
- Interoperabilidad: Ofrece la capacidad de que una plataforma pueda exhibir contenidos independientemente de quién y cómo fueron creados y de producir contenidos independientemente de la plataforma en la cual serán incorporados.
- Accesibilidad: Apunta a que los contenidos necesarios estén a nuestro alcance en todo momento y puedan ser accedidos desde cualquier lugar a través de los dispositivos disponibles. Esto se logra almacenando también metadatos de estos contenidos para tenerlos correctamente categorizados.
- Reusabilidad: Se enfoca en disminuir los tiempos de producción y aumentar la calidad de los contenidos. En lugar de comenzar de cero, reutilizar lo que ya existe y, si es necesario, mejorarlo.(2)

SCORM define dos tipos básicos de objetos que pueden formar parte de un contenido: *assets* (activos, en español) y SCO (*Shareable Content Object*, en español Objeto de Contenido Compartible). Los *assets* son los objetos elementales que pueden aparecer en el contenido (textos, imágenes, páginas web, documentos, multimedia, etcétera.). Los SCO están compuestos por los mismos materiales que los *assets* y, a diferencia de ellos, tienen la capacidad de comunicarse con la plataforma LMS. Para describir un curso, por ejemplo, los *assets* y los SCO son empaquetados en una estructura que los agrupa y organiza. Además, el curso incluye información sobre sí mismo que define su comportamiento en el LMS.

Para especificar los datos que se recogen por cada objeto de aprendizaje, SCORM propone seguir las especificaciones del estándar LOM (Learning Object Metadata, en español Metadatos de Objetos de Aprendizaje).

El estándar LOM permite describir las características de cualquier objeto de aprendizaje, digital o no, mediante una serie de metadatos agrupados en nueve categorías: General, Ciclo de vida, Meta-metadatos, Técnica, Uso educativo, Derechos, Relación, Anotación y Clasificación (3)

Estos metadatos garantizan la accesibilidad de los objetos de aprendizaje, posibilitando que luego puedan ser buscados y filtrados con facilidad tanto por el desarrollador como por el usuario final.

En la UCI se está utilizando el estándar SCORM para describir parte del contenido que aparece en el EVA (Entorno Virtual de Aprendizaje). También, como parte del proyecto productivo en desarrollo Alfaomega, se espera obtener una plataforma al estilo LMS compatible con SCORM y una herramienta de autoría de objetos de aprendizaje.

1.4 Metodología de desarrollo de software.

1.4.1 Rational Unified Process (RUP)

Esta metodología de desarrollo está clasificada dentro de las tradicionales o pesadas, pero gracias a su flexibilidad puede ser adaptada para realizar una planeación y documentación mínima. Es, sin dudas, la metodología más utilizada para realizar el análisis, diseño, implementación y documentación de sistemas basados en programación orientada a objetos.

RUP divide en 4 fases el desarrollo del software, estas son: inicio, la que define el modelo del negocio, el alcance y los límites del proyecto; elaboración, en la cual se define, valida y cimenta la arquitectura; construcción, donde se desarrolla el producto y transición, la encargada de poner el producto en manos de los usuarios. (4)

Por cada una de estas fases se transita a través de nueve flujos de trabajo, para los que RUP propone una serie de artefactos, actividades y roles que varían de un flujo de trabajo a otro. Su ciclo de vida está caracterizado por ser: (4)

- **Dirigido por casos de uso:** Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí, los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso.

- **Centrado en la arquitectura:** La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. La arquitectura se representa a través de vistas en las que se incluyen los diagramas de UML.

- **Iterativo e Incremental:** RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque hace hincapié en algunos más que en otros de acuerdo a la fase. (4)

Otras de las características que hacen idónea esta metodología para guiar el desarrollo del sistema a realizar son:

- Unifica los mejores elementos de metodologías anteriores.
- Preparado para desarrollar grandes y complejos proyectos.
- Orientado a Objetos.
- Utiliza el UML como lenguaje de representación visual.

- No necesariamente el cliente debe tomar un papel activo en el desarrollo del producto.
- Asociado al desarrollo se genera una gran cantidad de documentación que pueden ayudar a entender y mantener el sistema.

1.5 Lenguajes de programación y de modelado

1.5.1 PHP

PHP es el lenguaje libre más utilizado para desarrollar aplicaciones web. Permite la programación orientada a objetos y brinda la posibilidad de utilizar los gestores de base de datos más difundidos (MySQL, Oracle, PostgreSQL, MS SQL Server y otros). Gracias a su activa y numerosa comunidad de desarrolladores es posible encontrar una gran cantidad de ejemplos y documentación en la web.

Entre sus principales ventajas se encuentran:

- Fácil de aprender.
- Tiene muy buen rendimiento.
- Es un lenguaje multiplataforma.
- Liberado bajo licencia GNU.

1.5.2 JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Técnicamente, es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con este lenguaje se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. JavaScript supera a otros lenguajes del lado del cliente en cuanto a portabilidad, rendimiento, facilidad de uso y documentación.

1.5.3 UML

UML (Unified Modeling Language, en español Lenguaje Unificado de Modelado) es el lenguaje de modelado de software más difundido y utilizado en la actualidad. El objetivo de este lenguaje es visualizar, especificar, construir y documentar los elementos que conforman un sistema. UML permite especificar la distribución, el comportamiento, la arquitectura y la estructura de datos de una aplicación; así como los procesos del negocio referentes a la misma.

Está específicamente diseñado para modelar sistemas que utilizarán en su desarrollo el paradigma de programación orientado a objetos y no se encuentra ceñido a ninguna metodología de desarrollo en especial.

1.6 Herramientas de desarrollo y de modelado

1.6.1 NetBeans

NetBeans es una herramienta para programadores que les permite escribir, compilar, corregir errores y ejecutar programas. Está escrito en Java, pero puede servir de soporte a cualquier otro lenguaje de programación. Existe también un número enorme de módulos para extender el NetBeans IDE. El NetBeans IDE es un producto libre y gratuito sin restricciones de utilización. (5)

NetBeans brinda un excelente soporte para PHP y JavaScript que incluye resaltado de sintaxis, autocompletado, ayuda, control de versiones, gestión de ficheros, pre-visualización, refactorizado, debugger, etcétera. Su última versión estable, la 6.8, esta incluye entre sus funcionalidades el soporte para PHP 5.3 y la integración con el framework Symfony.

1.6.2 Visual Paradigm for UML

Visual Paradigm for UML es una herramienta CASE potente y fácil de utilizar, que permite el modelado visual UML. Esta herramienta no está diseñada para apoyar una metodología de desarrollo en específico, pero gracias a su flexibilidad se integra perfectamente con RUP.

Entre sus características más importantes sobresalen:

- Soporte para varias versiones de UML.

- Brinda una versión gratuita y otra comercial.
- Permite la ingeniería inversa (código a modelo, código a diagrama).
- Permite la generación de código (modelo a código, diagrama a código).
- Permite exportar el diseño de la base de datos para varios gestores.

1.7 Framework

1.7.1 Symfony

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web con PHP. Para empezar, 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. (6)

Entre las características que determinaron su selección se encuentran:

- Compatible con la mayoría de los gestores de bases de datos, como son: MySQL, PostgreSQL, Oracle, entre otros.
- Permite la integración de nuevas funcionalidades a través de plugins.
- Gracias a su estabilidad permite desarrollar aplicaciones a largo plazo.
- Utiliza prácticas y patrones de diseño de utilidad comprobada en el desarrollo.
- Encapsula operaciones complejas en instrucciones sencillas para la línea de comandos.

1.7.2 JQuery

JQuey es una librería JavaScript rápida y concisa, que simplifica las operaciones sobre el DOM, el manejo de eventos, animaciones e interacciones AJAX para el desarrollo en la WEB. (7).

JQuery encabeza la lista de las librerías JavaScript más utilizadas gracias a su sencillez, flexibilidad y rendimiento. Se caracteriza por tener una amplia aceptación por parte de los programadores y un grado

de penetración en el mercado muy amplio, es un producto bien documentado y con una comunidad de desarrolladores numerosa y activa.

Algunas de las ventajas que proporciona su utilización son:

- Optimiza la manipulación de los elementos del DOM.
- Compatibilidad entre los navegadores más utilizados.
- Provee un mecanismo para la captura y manipulación de eventos.
- Simplifica la creación de efectos gráficos y animaciones sobre el contenido de la página.
- Centraliza y facilita el trabajo con AJAX.

1.8 Gestor de Base de Datos

1.8.1 PostgreSQL

PostgreSQL es un sistema de gestión de base de datos relacional, orientada a objetos y libre que lleva más de una década de desarrollo y cuenta con una arquitectura probada; ganándose en este tiempo una sólida reputación de confiabilidad e integridad de datos. Funciona en los principales sistemas operativos, incluyendo Linux, UNIX y Windows. También es compatible con el almacenamiento de objetos binarios, incluyendo imágenes, sonidos o vídeos.

PostgreSQL es considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas características que tradicionalmente sólo se podían ver en productos comerciales de alto calibre. Es más eficiente cuando el tamaño de la base de datos es grande, y siendo la eficiencia un requisito fundamental en el desarrollo de aplicaciones web, es fácil entender su selección como gestor de base de datos a utilizar en la aplicación a desarrollar.

1.9 Conclusiones

En este capítulo se analizó el proceso de gestión de contenidos en los CMS y algunas características del proceso de gestión de contenidos para la educación, llegando a las siguientes conclusiones: Sobre los CMS, por las facilidades de uso y de organización que proporciona, decidió utilizarse la filosofía de mantener las acciones relacionadas con la creación, mantenimiento y organización del contenido

separadas de las que se encargan de la visualización del mismo. En cuanto a la gestión de contenidos educativos, se decidió utilizar parte del modelo SCORM en lo referente a la organización de los elementos educativos (*asset* y SCO) y la descripción de los mismos mediante el estándar LOM; ya que representa una excelente solución para estructurar el contenido y permitirá la posible utilización del modelo SCORM en futuras versiones del producto.

El equipo de dirección del proyecto al que pertenece esta solución seleccionó un grupo de herramientas y tecnologías para su realización. Con el objetivo de apoyar esta selección, en este capítulo se exponen las principales características por las que se eligió a RUP como metodología de desarrollo, a JavaScript y a PHP como lenguajes de programación, a UML como lenguaje de modelado, a NetBeans como IDE de desarrollo, a Visual Paradigm for UML como herramienta CASE, a Symfony y a JQuery como framework de desarrollo y a PostgreSQL como gestor de base de datos.

Capítulo 2. Características del Sistema

2.1 Introducción

En el presente capítulo se describen, a través de un modelo de dominio, los principales conceptos que aparecen en el negocio del sistema a desarrollar. Se exponen las características y propiedades que debe tener la aplicación, definiéndose los requisitos funcionales y no funcionales. Basado en estos requerimientos se explica brevemente la propuesta de sistema. También se analizan los actores involucrados, el diagrama de casos de uso del sistema y la descripción de cada uno de estos.

2.2 Modelo de Dominio

Los principales procesos que ocurren en el negocio del sistema a desarrollar no se encuentran bien definidos. Por esta razón, se decidió realizar un modelo de dominio con el objetivo de representar y entender los principales conceptos relacionados con el ambiente donde será desplegado el sistema y con la colección El Navegante desarrollada por el MINED.

Un modelo del dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabaja el sistema. (4)

2.2.1 Glosario de términos del dominio

Estudiante: Persona entre 12 y 15 años de edad que cursa estudios en los liceos bolivarianos.

colección El Navegante: colección de software educativo destinada a apoyar el proceso de enseñanza-aprendizaje de los niños en las escuelas secundarias venezolanas.

Liceo: Institución educativa centrada en la formación de estudiantes de entre 12 y 15 años de edad, homólogo venezolano de la escuela secundaria cubana.

Profesor: Persona responsable de educar, guiar y supervisar a los estudiantes en el proceso de enseñanza - aprendizaje.

Media: Imágenes, sonidos o videos que apoyan el proceso de enseñanza.

Video: Media que brinda información audiovisual sobre un contenido.

Imagen: Media que brinda información gráfica sobre un contenido.

Sonido: Media que brinda información sonora sobre un contenido.

Producto Multimedia: Producto que combina diversos tipos de medias.

Módulo: Se le denomina al conjunto de elementos fundamentales que componen el producto.

Temas: Módulo que agrupa el contenido educativo de un producto.

Juegos: Módulo que contiene juegos que ejercitan el contenido aprendido por el estudiante.

Ejercicio: Módulo que permite ejercitar y comprobar las habilidades adquiridas por los estudiantes.

Mediateca: Módulo que agrupa las medias presentes en el producto.

Resultado: Módulo que permite consultar las trazas y evaluaciones de los estudiantes que utilizan el producto.

Maestro: Módulo que le permite al profesor configurar algunas opciones del producto en cuestión.

Capítulo: La más general de las estructuras en las que se divide el contenido de un producto.

Tema: Estructura que agrupa el contenido dentro de un capítulo.

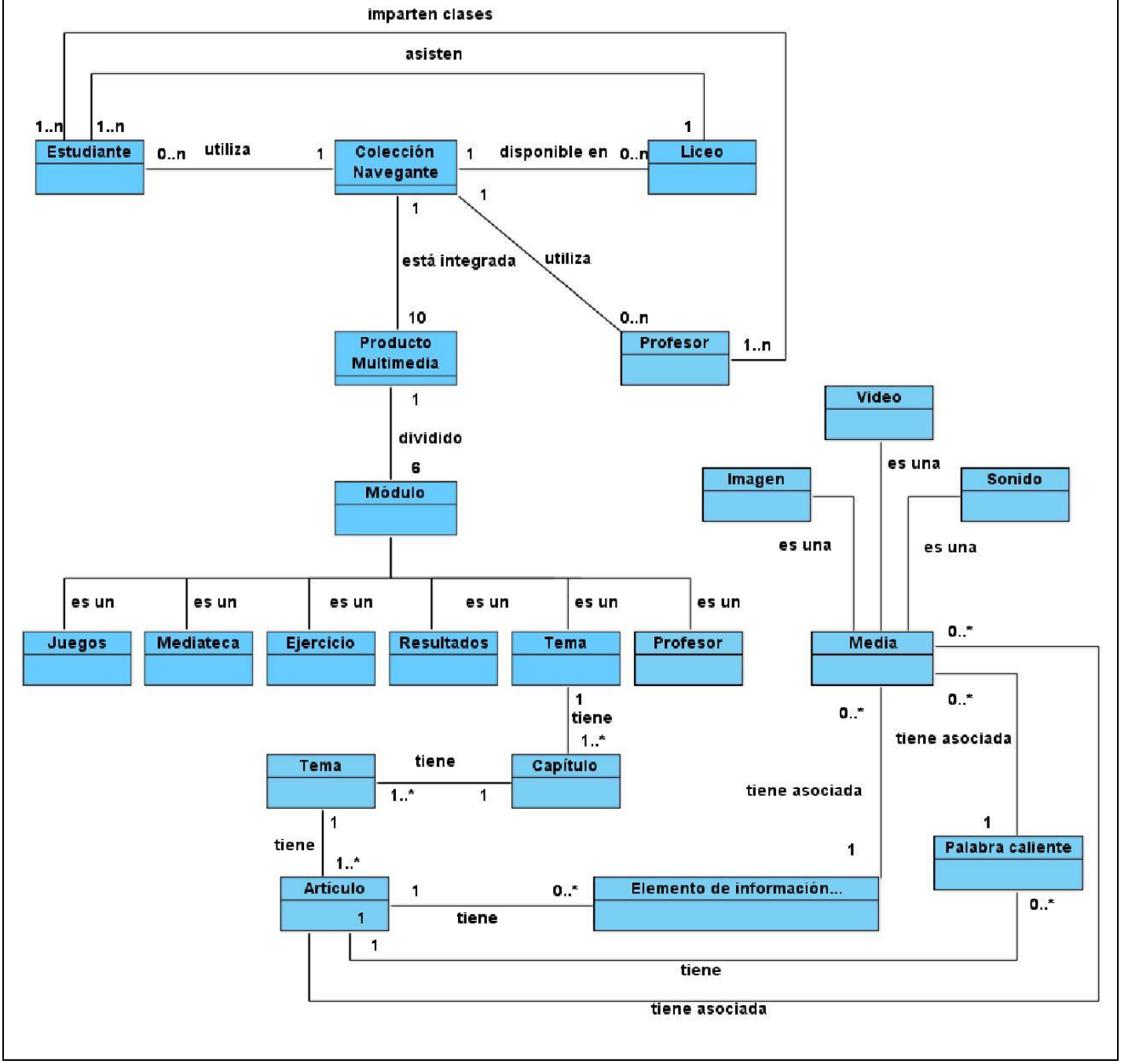
Artículo: Texto que expone un tema específico; representa la unidad más importante del contenido.

Palabra caliente: Palabra o fragmento de texto interactivo que ofrece información sobre sí mismo.

Elemento de información adicional: Información extra sobre un artículo, pueden ser curiosidades, datos de interés, entre otros.

2.2.2 Diagrama del modelo de dominio

Modelo de dominio



2.3 Especificación de los requerimientos del software.

2.3.1 Requerimientos funcionales

RF 1 Mostrar índice.

RF 2 Gestionar un contenedor.

- RF 2.1 Crear nuevo contenedor.
- RF 2.2 Modificar datos de contenedor.
- RF 2.3 Ver datos de un contenedor.
- RF 2.3 Eliminar contenedor.

RF 3 Gestionar un artículo.

- RF 3.1 Crear un nuevo artículo.
- RF 3.2 Modificar un artículo.
- RF 3.3 Ver datos de un artículo.
- RF 3.4 Eliminar un artículo.

RF 4 Administrar palabra caliente.

- RF 4.1 Crear una palabra caliente asociada a un artículo.
- RF 4.2 Eliminar una palabra caliente.

RF 5 Gestionar elemento de información adicional de un artículo.

- RF 5.1 Crear un elemento de información adicional.
- RF 5.2 Modificar datos de un elemento de información adicional.
- RF 5.3 Ver datos de un elemento de información adicional.

RF 5.4 Eliminar un elemento de información adicional.

RF 6 Administrar relación media – elemento del contenido.

RF 6.1 Crear asociación media – elemento del contenido.

RF 6.2 Eliminar asociación media – elemento del contenido.

RF 7 Consultar un artículo.

RF 7.1 Consultar medias asociadas al artículo.

RF 8 Consultar una palabra caliente.

RF 9 Consultar un elemento de información adicional.

RF 9.1 Consultar medias asociadas al elemento de información adicional.

2.3.2 Requerimientos no funcionales

Requisitos de hardware

- Procesador Pentium 233 MHz (recomendado 500 MHz o mayor).
- 64 MB de RAM (recomendado 128 MB de RAM o mayor).
- 1 GB de espacio en disco duro.
- Lector de CD-ROM.
- Dispositivos de audio.
- Soporte de video que admita resolución de al menos 1024x600px y 24 bits.
- Dispositivo de red de al menos 10 Mbits de velocidad de transmisión.

Requisitos de apariencia e interfaz externa

- El sistema proporcionará claridad y correcta organización de la información, permitiendo la interpretación correcta e inequívoca de ésta.
- El diseño de la interfaz gráfica deberá garantizar la distinción visual entre los elementos del sistema.

Requisitos legales

- Cada una de las medias (imágenes, videos, sonidos) que se utilicen en el producto deben tener el permiso legal de sus autores y su aprobación para hacer uso de ellas.

Requisitos de portabilidad

- El sistema podrá ser utilizado bajo cualquier Sistema Operativo.

Requisitos de software

- Computadora Personal con navegador Mozilla Firefox 3.x, Internet Explorer 8 o Google Chrome 4.x.

Restricciones en el diseño y la implementación

- JQuery 1.4
- PHP 5.2
- Symfony 1.4
- Apache 2.x
- PostgreSQL 8.4

2.4 Propuesta de sistema

El módulo “Temas” estará dividido en dos áreas fundamentales: presentación y administración de contenidos. El área de presentación debe permitir acceder y navegar a través de los elementos que integran un producto de la colección El Navegante de manera sencilla e intuitiva. La estructura estática de capítulos, temas y artículos será sustituida por una más genérica en la que un producto estará constituido por contenedores y estos a su vez por otros contenedores o artículos. Un artículo podrá contener medias asociadas, palabras calientes y/o elementos de información adicional. Tanto las palabras calientes como los elementos de información adicional podrán contener medias.

El área de administración de temas apoyará el proceso de montaje y actualización del contenido de los productos que integran la versión multiplataforma de la colección El Navegante. En esta área, el contenido

de un producto será representado mediante un índice jerárquico, lo que facilitará la comprensión y usabilidad de la herramienta. Mediante un menú contextual, podrán ser seleccionadas acciones específicas sobre cada elemento del índice.

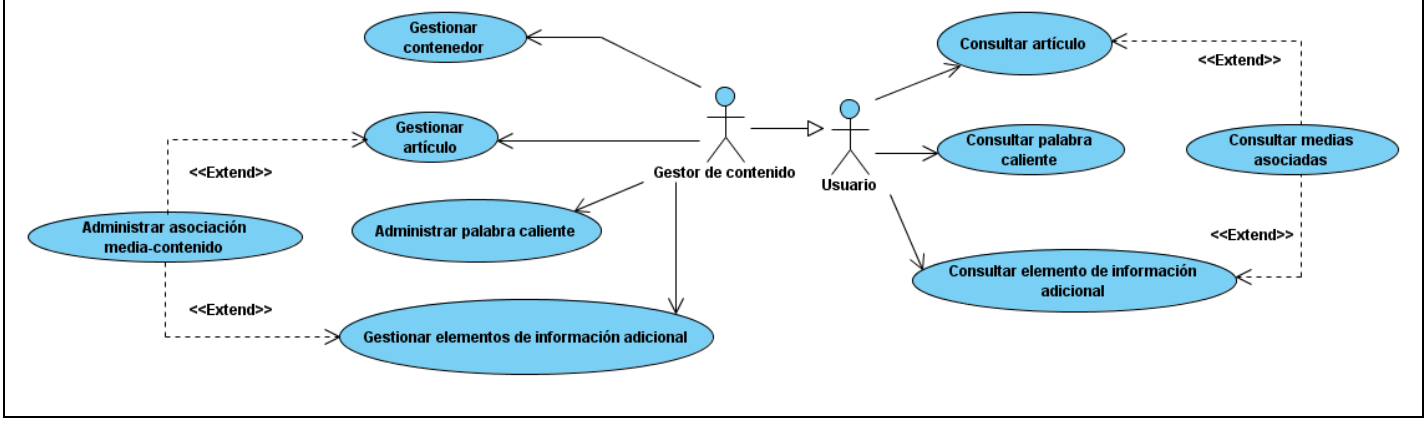
2.5 Definición de los actores

Actores	Justificación
Usuario	Representa a los estudiantes, profesores o invitados. Tiene permisos para consultar los elementos de un producto.
Gestor de contenido	Es un Usuario con permisos adicionales para crear, modificar, ver o eliminar el contenido de los productos.

Tabla 1: Actores del sistema

2.6 Modelo de casos de uso del sistema

Diagrama de casos de uso del sistema



2.7 Descripción de los casos de uso del sistema

<table>
<tr><th colspan="3">Caso de uso</th></tr>
<tr><td>CU-2</td><td colspan="2">Gestionar artículo</td></tr>
<tr><td>Propósito</td><td colspan="2">Incluir, modificar, ver o eliminar un artículo.</td></tr>
<tr><td colspan="3">Actores: Gestor de contenido (inicia)</td></tr>
<tr><td colspan="3">Resumen: El caso de uso se inicia cuando el actor selecciona la opción que le permite realizar una acción sobre un artículo. El actor puede incluir, modificar, ver o eliminar un artículo. En caso de que seleccione la opción de incluir un artículo, el sistema dará la posibilidad de insertar los datos necesarios y opcionalmente, de realizar una asociación entre una media y dicho artículo. Si el actor elige la opción de modificar un artículo, el sistema mostrará los datos que pueden ser editables dentro del mismo, así como asociar y disociar medias a dicho artículo; una vez realizados los cambios, guardará las modificaciones. Si selecciona la opción de ver un artículo, el sistema mostrará los datos del mismo. En caso de seleccionar la opción eliminar, el sistema eliminará el artículo seleccionado; terminado así el caso de uso.</td></tr>
<tr><td>Referencias</td><td colspan="2">RF 3</td></tr>
<tr><th colspan="3">Flujo Básico</th></tr>
<tr><th>Acción del actor</th><th colspan="2">Respuesta del sistema</th></tr>
<tr><td>1. El caso de uso se inicia cuando el actor selecciona la opción de realizar una acción sobre un artículo.</td><td colspan="2"></td></tr>
<tr><td></td><td colspan="2">2. Brinda la posibilidad de realizar las acciones:
• Incluir un nuevo artículo.
• Modificar los datos del artículo. Ver Sección 1:”Modificar datos del artículo”.</td></tr>
</table>

	• Ver datos del artículo. Ver Sección 2: "Ver artículo". • Eliminar artículo. Ver Sección 3: "Eliminar artículo".
3. Selecciona la opción de incluir un nuevo artículo.	
	4. Brinda la posibilidad de introducir los datos de un artículo. • Nombre • Descripción • Ámbito • Estructura • Nivel Agregación • Autor • Texto Opcionalmente permite seleccionar: • Medias asociadas. Ver *CU Administrar relación media-contenido*. Además permite: • Guardar los datos. • Cancelar la operación en cualquier momento.
5. Introduce los datos del artículo.	
6. Selecciona la opción de guardar los datos.	

	7. Valida los datos.
	8. Crea un nuevo artículo.
	9. El caso de uso termina.
Flujo alternativo ***. a El actor selecciona la opción de Cancelar**	
Acción del actor	**Respuesta del sistema**
	* .a.1 Elimina los datos creados.
	*. a.2 El caso de uso termina.
Flujo alternativo **7. a Existen datos incompletos.**	
Acción del actor	**Respuesta del sistema**
	7. a.1 Muestra un indicador sobre los campos vacíos: “Dato obligatorio”.
	7. a.2 Regresa al paso 5.
Flujo alternativo **7. b Existen datos incorrectos.**	
Acción del actor	**Respuesta del sistema**
	7. b.1 Muestra un indicador sobre los campos incorrectos: “Dato incorrecto”.
	7. b.2 Regresa al paso 5.

Sección 1: "Modificar datos del artículo"	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción de modificar los datos del artículo.	
	2. Muestra los datos del artículo y brinda la posibilidad de cambiar sus valores. Permite además: • Guardar los datos. • Cancelar la operación en cualquier momento.
3. Modifica los datos que necesite y selecciona la opción de guardar los datos.	
	4. Validar datos.
	5. Actualiza los datos del producto.
	6. El caso de uso termina.
Flujo alternativo ***. a El actor selecciona la opción de Cancelar.**	
Acción del actor	**Respuesta del sistema**
	*.a.1 Descarta los cambios realizados.
	*. a.2 El caso de uso termina.
Flujo alternativo **4. a Existen datos incompletos.**	

Acción del actor	Respuesta del sistema
	4. a.1 Muestra un indicador sobre los campos vacíos: “Dato obligatorio”.
	4. a.2 Regresa al paso 3.
Flujo alternativo **4. b Existen datos incorrectos.**	
Acción del actor	**Respuesta del sistema**
	4. b.1 Muestra un indicador sobre los campos incorrectos: “Dato incorrecto”.
	4. b.2 Regresa al paso 3.
Sección 2: “Eliminar artículo”	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción de eliminar un artículo.	
	2. Muestra el nombre del artículo y el mensaje de confirmación “Se eliminará el elemento seleccionado. ¿Desea continuar?”
3. Selecciona la opción Aceptar.	
	4. Elimina el artículo.
	5. El caso de uso termina.

Flujo alternativo 3. a El actor selecciona la opción Cancelar.	
Acción del actor	**Respuesta del sistema**
	3. a.1 Regresa a la vista anterior.
	3. a.2 El caso de uso termina.
Sección 3: "Ver artículo"	
1. Selecciona la opción de ver los datos del artículo.	
	2. Muestra los datos del artículo. Y permite: • Salir de la vista actual.
3. Consulta los datos del artículo.	
4. Selecciona la opción que le permite salir de la vista actual.	
	5. Muestra la pantalla anterior.
	6. El caso de uso termina.

Tabla 2: CU Gestionar artículo

Caso de uso	
CU-3	Administrar palabra caliente
Propósito	Incluir o eliminar una palabra caliente.

Actores: Gestor de contenido (inicia)	
Resumen: El caso de uso se inicia cuando el actor selecciona la opción que le permite realizar una acción sobre una palabra o frase caliente. El actor puede incluir o eliminar una palabra caliente. En caso de que seleccione la opción de incluir una nueva palabra caliente, el sistema establecerá una relación entre dicha palabra y el elemento del glosario de términos que le corresponda, convirtiéndola así en palabra caliente. Si el actor elige la opción de eliminar una palabra caliente, el sistema eliminará la relación existente entre la palabra caliente y el elemento del glosario de términos correspondiente, terminando así el caso de uso.	
Referencias	RF 6
Flujo Básico	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción de realizar una acción sobre una palabra o frase caliente.	
	2. Brinda la posibilidad de realizar las acciones: • Incluir una palabra caliente. • Eliminar una palabra o frase caliente. Ver Sección 1: “Eliminar una palabra caliente”.
3. Selecciona en el texto del artículo la palabra o frase a convertir en palabra caliente.	
4. Selecciona la opción de Incluir palabra caliente.	
	5. Compara la palabra o frase seleccionada con los

	nombres de los elementos del glosario de términos.
	6. Muestra la palabra o frase encontrada y permite seleccionar la opción: • Guardar (*guarda la relación de la palabra con el elemento del glosario de términos*) • Cancelar
7. Selecciona la opción Guardar.	
	8. Guarda el elemento seleccionado como palabra caliente y su relación con el elemento del glosario de términos.
	9. Resalta en el texto del artículo la palabra caliente.
	10. El caso de uso termina.
Flujo alternativo **5. a El sistema no encontró elementos coincidentes.**	
Acción del actor	**Respuesta del sistema**
	5 .a.1 Muestra el mensaje de información: “Esta palabra o frase no existe en el Glosario de Términos”.
	5. a.2 El caso de uso termina.
Flujo alternativo **7. a El actor selecciona la opción de Cancelar.**	
Acción del actor	**Respuesta del sistema**

	7. a.1 Muestra un mensaje de información: “La acción ha sido cancelada”.
	7. a.2 La palabra o frase mostrada no se guarda como caliente.
	7. a.3 El caso de uso termina.
Sección 1: “Eliminar una palabra caliente”.	
Acción del actor	**Respuesta del sistema**
1. Selecciona en el texto del artículo la palabra caliente a eliminar.	
	2. Permite: • Eliminar la palabra caliente. • Cancelar la operación en cualquier momento.
3. Selecciona la opción de eliminar la palabra caliente.	
	4. Muestra el mensaje de confirmación: “La palabra o frase seleccionada dejará de ser caliente ¿Desea continuar?”
5. Selecciona Aceptar.	
	6. Elimina la relación entre la palabra o frase seleccionada y el elemento del glosario de términos correspondiente.
	7. Regresa a la vista anterior actualizando los datos.
	8. El caso de uso termina.

Flujo alternativo *. a El actor selecciona la opción de Cancelar.	
Acción del actor	**Respuesta del sistema**
	3. a.1 Regresa a la vista anterior.
	3. a.2 El caso de uso termina.

Tabla 3: CU Administrar palabra caliente

Caso de uso	
CU-6	Consultar artículo
Propósito	Mostrar la información referente a un artículo.
Actores: Usuario (inicia), Gestor de Contenido.	
Resumen: El caso de uso se inicia cuando el actor accede al módulo Temas de un producto. El sistema muestra el índice con los contenedores que aparecen en un producto y sus artículos asociados. El actor selecciona el artículo que quiere consultar y el sistema le muestra el artículo con sus medias asociadas (en caso de tenerlas); terminando así el caso de uso.	
Referencias	RF 1, RF 3
Flujo Básico	
Acción del actor	**Respuesta del sistema**
1. El caso de uso inicia cuando el actor accede al módulo Temas.	
	2. Muestra un índice con los contenedores y sus artículos asociados y permite seleccionar un artículo.

3. Selecciona un artículo.	
	4. Muestra el texto del artículo y sus medias asociadas en caso de tenerlas. Ver *CU Consultar medias asociadas.* Permite: • Salir de la vista actual.
5. Consulta el artículo.	
6. Selecciona la opción que le permite salir de la vista actual.	
	7. Regresa a la pantalla anterior.
	8. El caso de uso termina.

Tabla 4: CU Consultar artículo

Caso de uso	
CU-7	Consultar palabra caliente
Propósito	Consultar una palabra caliente.
Actores: Usuario (inicia), Gestor de Contenido.	
Resumen: El caso de uso se inicia cuando el actor selecciona una palabra caliente. El sistema muestra el elemento del glosario de términos asociado a la palabra caliente seleccionada y el caso de uso termina.	
Referencias	RF 9
Flujo Básico	
Acción del actor	**Respuesta del sistema**

1. El caso de uso inicia cuando el actor selecciona la opción de consultar una palabra caliente incluida en el texto de un artículo.	
	2. Muestra el contenido del elemento del glosario de términos asociado a dicha palabra caliente en una ventana emergente. Y permite: • Cerrar la ventana emergente.
3. Consulta la información de la palabra caliente.	
4. Selecciona la opción que le permite cerrar la ventana emergente.	
	5. El caso de uso termina.

Tabla 5: CU Consultar palabra caliente

El resto de las descripciones de los casos de uso del sistema se encuentran en el ¡Error! No se encuentra el origen de la referencia..

2.8 Conclusiones

En este capítulo se definieron los principales conceptos presentes en el negocio del sistema a desarrollar. Se obtuvieron los requisitos funcionales y no funcionales de la aplicación y tomándolos como base, se planteó la propuesta de sistema. Además, se identificaron y describieron los casos de uso y los actores presentes en el sistema. Los artefactos generados sirven como punto de partida para el desarrollo del análisis y el diseño.

Capítulo 3. Análisis y diseño

3.1 Introducción

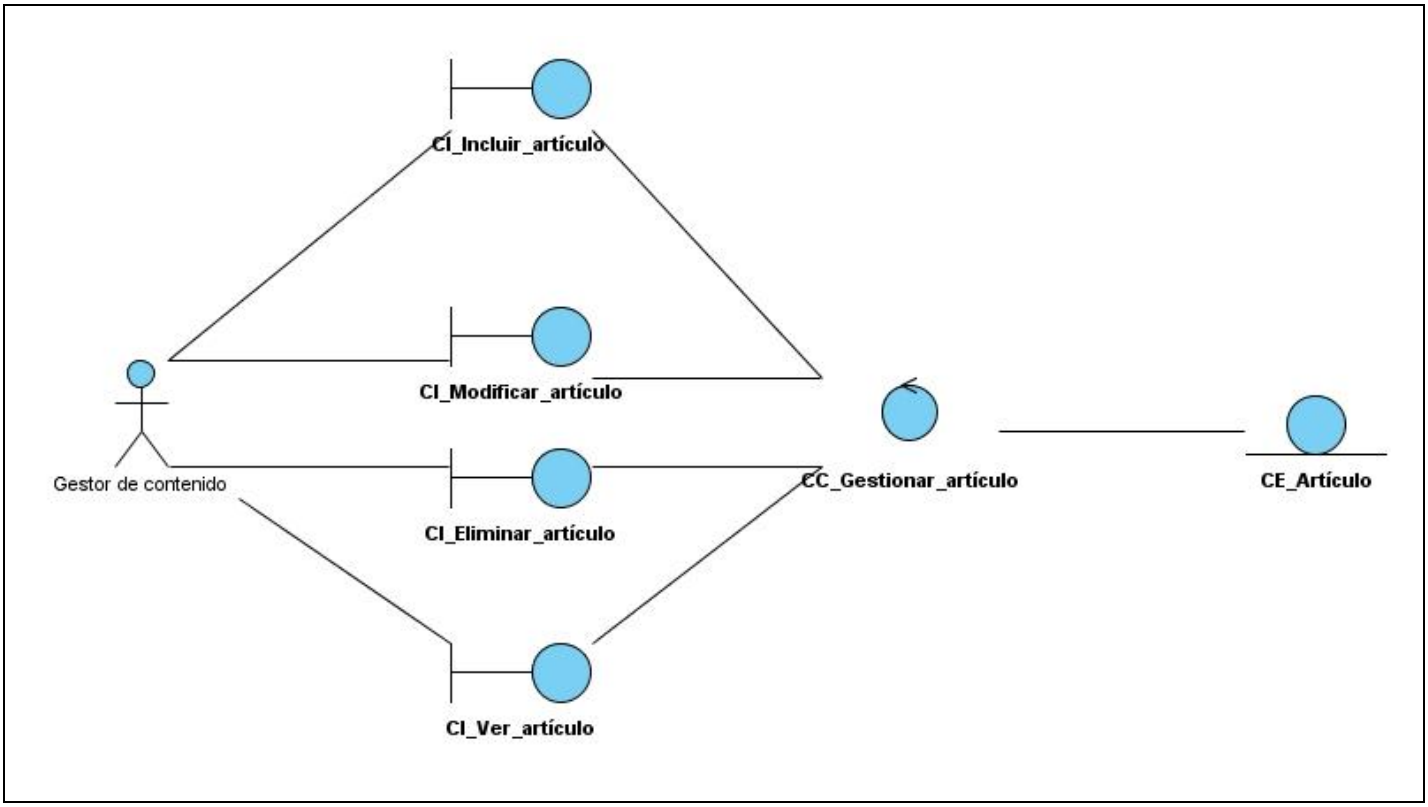
En este capítulo se realiza el análisis y diseño del sistema a desarrollar. Para ello se modelan los diagramas de clases del análisis y del diseño para cada caso de uso. Se diseña también la estructura de la base de datos a utilizar, tomando como base las clases persistentes. Para garantizar una adecuada comprensión del diagrama de clases del diseño, se explica el patrón arquitectónico y los patrones de diseño utilizados.

3.2 Modelo de análisis

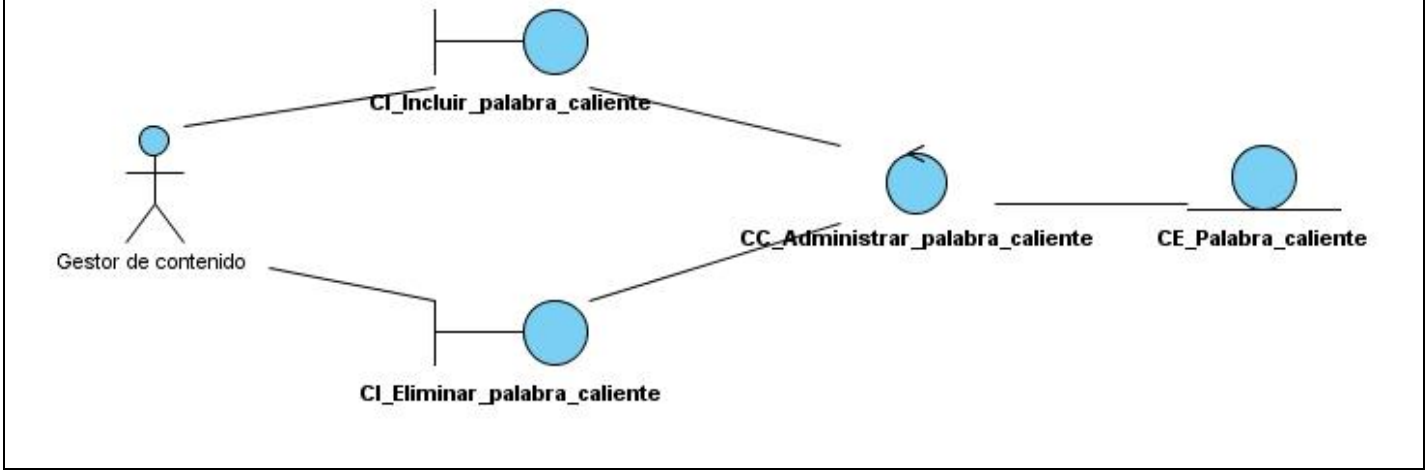
El Modelo de análisis es utilizado principalmente para obtener una primera visión del sistema y comprender en detalle los requisitos funcionales. Este modelo sirve como una primera aproximación al diseño, es un modelo abstracto, en el que no se especifica el lenguaje en el que se va a implementar.

A continuación se muestran los diagramas de clase del análisis (DCA) para los casos de uso del sistema más representativos. El resto se encuentra recogido en el ¡Error! No se encuentra el origen de la referencia..

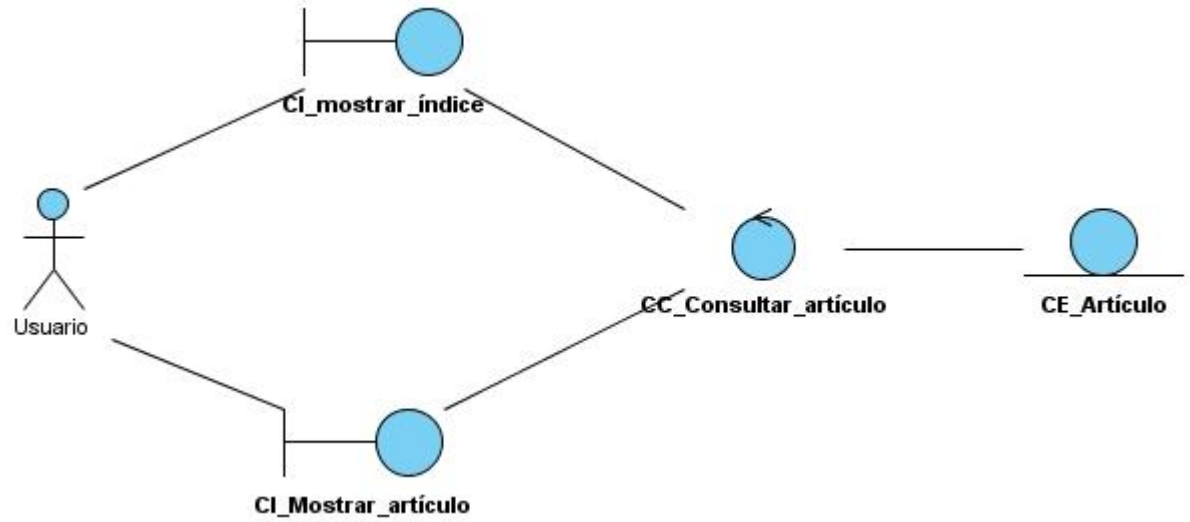
DCA Gestionar artículo



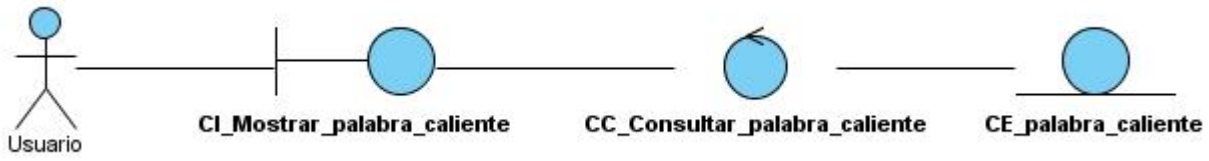
DCA Administrar palabra caliente



DCA Consultar artículo



DCA Consultar palabra caliente



3.3 El patrón arquitectónico Modelo Vista Controlador en Symfony

El framework Symfony está basado en el patrón arquitectónico Modelo Vista Controlador (MVC). Este patrón separa en tres niveles las funcionalidades de una aplicación con el objetivo de aumentar la usabilidad de las mismas. Estos niveles son:

- **Modelo:** Representa los datos y reglas del negocio.
- **Vista:** Permite al usuario interactuar con los objetos del modelo.
- **Controlador:** Se encarga de atender las peticiones de los usuarios y realizar los cambios necesarios tanto en el Modelo como en la Vista.

Algunas de las ventajas que proporciona la utilización de este patrón son: (11)

- Sencillez para crear distintas representaciones de los mismos datos.
- Reutilización de los componentes.
- Simplicidad en el mantenimiento de los sistemas.
- Los desarrollos suelen ser más escalables.

En Symfony la estructura de la base de datos es mapeada a clases del sistema mediante un ORM (Object-Relational Mapping). Utilizando el patrón de diseño Active Record el desarrollador puede interactuar con el contenido de la base de datos a través de las instancias de estas clases. Por cada tabla de la base de datos Symfony genera dos clases en el sistema. La primera, con el mismo nombre de la tabla, donde se implementan las funcionalidades que definen las reglas del negocio para cada entidad; y la segunda, con el sufijo *Table*, donde se implementan las funcionalidades que permiten interactuar con la tabla que representan.

La vista está representada por ficheros escritos en PHP que se encargan de construir la página HTML con la que interactúa el usuario. Estos ficheros se identifican por su nombre, constituido por el nombre de la acción que los utiliza y el sufijo *Success*.

El nivel controlador está constituido por clases que manejan las peticiones de los usuarios, la seguridad de la aplicación y la interacción entre los otros dos niveles. Las funcionalidades que implementa el desarrollador en este nivel son tratadas como acciones; cada acción es, básicamente, uno o varios métodos que satisfacen una petición del usuario. Por cuestiones de organización, en el sistema a desarrollar se decidió implementar una clase por cada acción; heredando cada una de la clase *sfActions.* El nombre de estas clases está constituido por el nombre de la acción y el sufijo *Action*.

3.4 Patrones de diseño utilizados

Una de las ventajas de desarrollar con Symfony es que utiliza en su implementación varios patrones de diseño. Entre los más representativos pueden encontrarse:

Controlador frontal: Define un único componente como responsable del procesamiento de las solicitudes en una aplicación. Por lo tanto, cuando cambia la estructura de las solicitudes o la manera de procesarlas sólo una pequeña parte de la aplicación necesita ser actualizada.

Intercepting filter: Propone crear una estructura de filtros que intercepten las peticiones y respuestas. Permite el pre-procesamiento y pos-procesamiento de las mismas.

Comando: Encapsula una petición en un objeto. Permite procesar de la misma manera diferentes solicitudes, registrarlas y realizar operaciones sobre ellas como deshacer.

Active record: Plantea que el objeto contenga los datos presentes en una fila de la tabla que represente y encapsule la lógica necesaria para interactuar con la base de datos. De esta forma el acceso a datos se realiza de manera más sencilla y uniforme.

Todas las peticiones realizadas a una aplicación web desarrollada con Symfony son procesadas a través de un controlador frontal. Antes de llegar a una acción, las peticiones pasan por una serie de filtros donde pueden ser modificadas en dependencia, por ejemplo, de los permisos del usuario que las realizó. Cada

petición es encapsulada como un objeto *sfWebRequest* y pasada como parámetro a la acción que le dará respuesta. Desde la acción, gracias a que el ORM implementa el patrón Active Record, puede interactuarse con la base de datos sin escribir sentencias SQL.

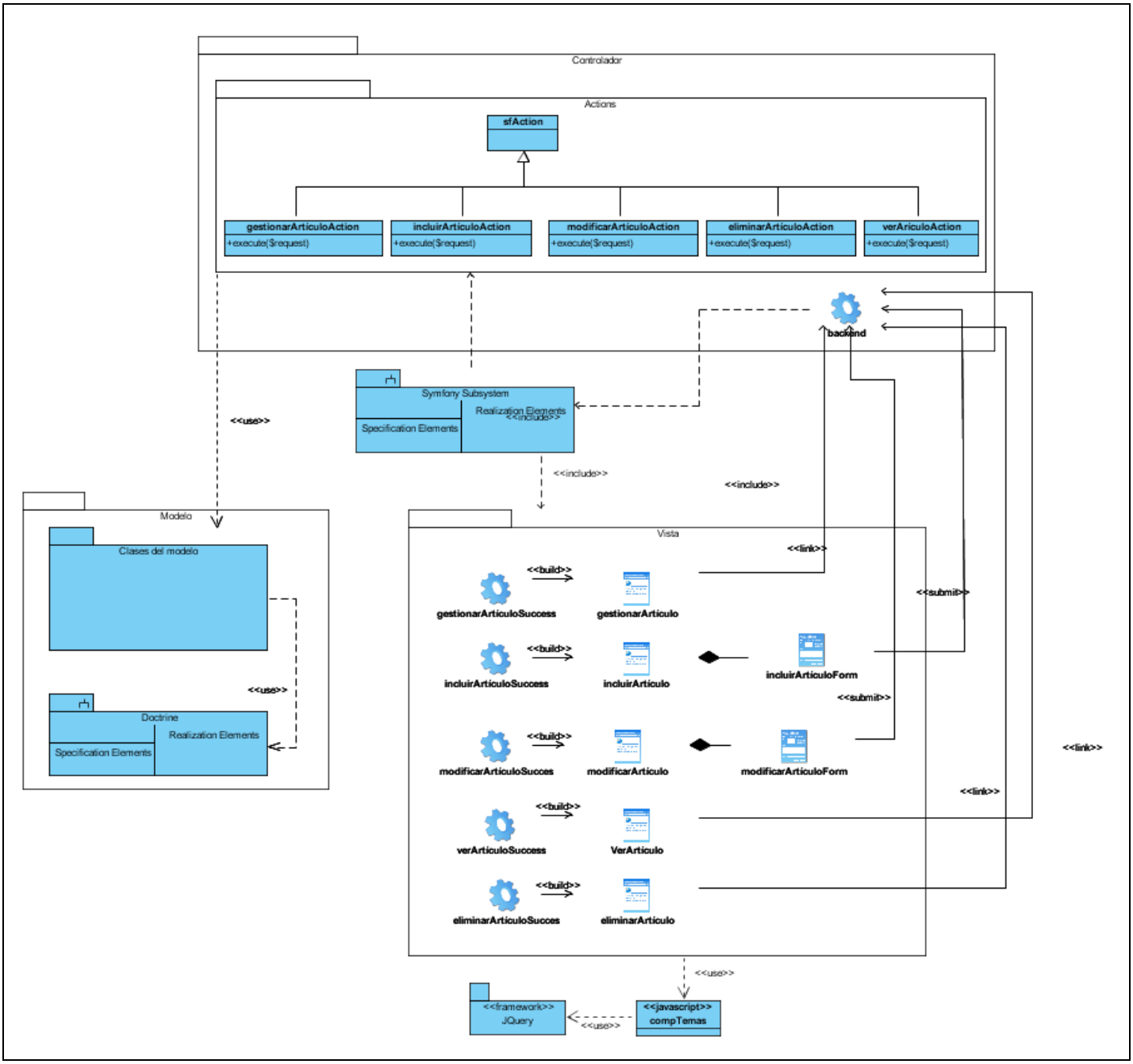
En el lado del cliente, con JavaScript, se utilizó el patrón de diseño **Mediador** con la intención de propiciar el bajo acoplamiento. Si muchos objetos interactúan con otros objetos, puede crearse una estructura muy compleja y prácticamente imposible de mantener; por eso se hace necesaria la utilización de un patrón que simplifique la interacción entre ellos.

3.5 Modelo de diseño

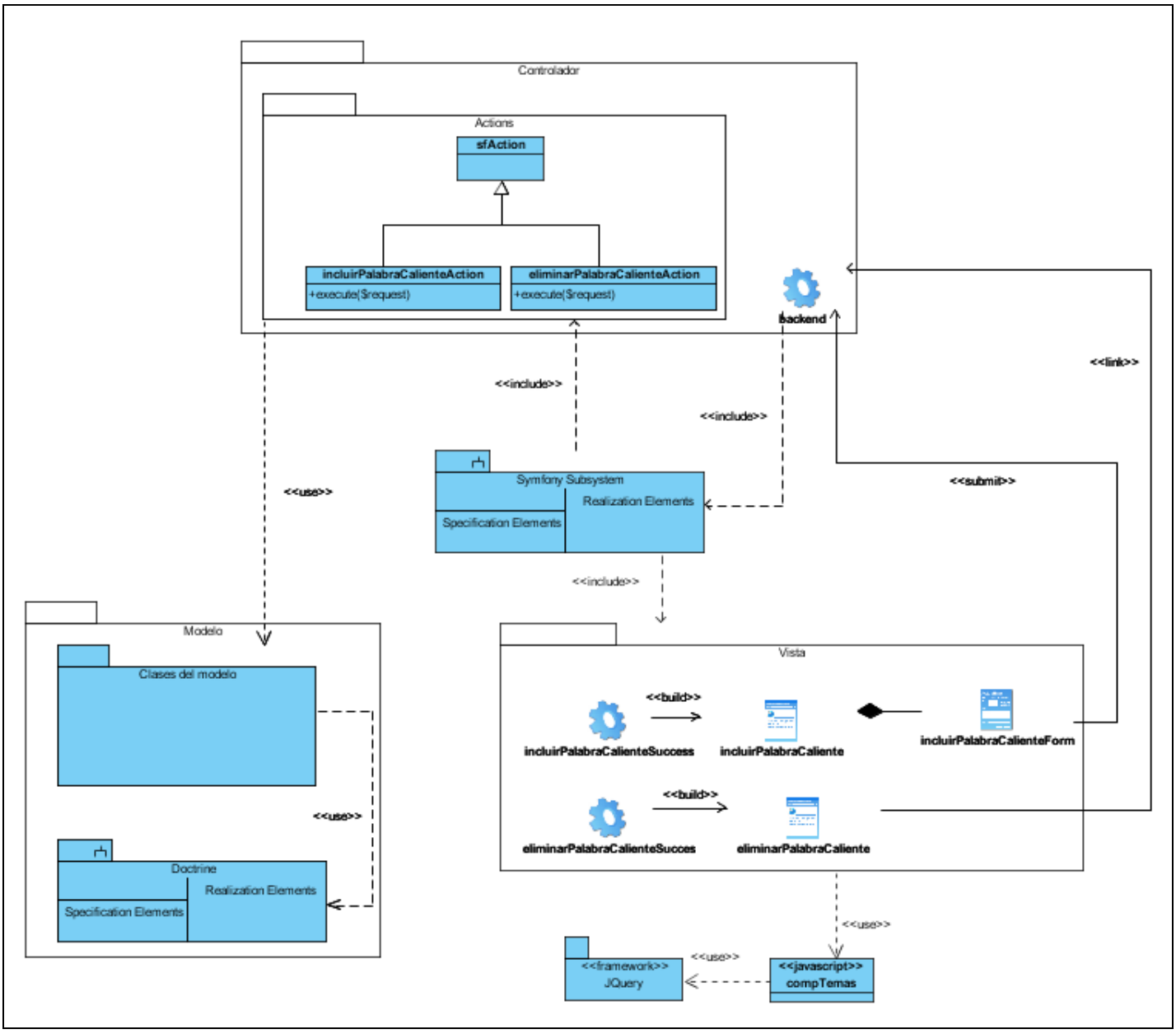
El Modelo de Diseño describe la realización de los casos de uso y constituye una abstracción del modelo de implementación y del código fuente. Constituye una entrada esencial a las actividades de implementación.

A continuación aparecen los diagramas de clase del diseño (DCD) para los casos de uso del sistema más representativos, el resto puede encontrarse en el ¡Error! No se encuentra el origen de la referencia.. En estos diagramas puede identificarse claramente el patrón arquitectónico MVC y algunos de los patrones de diseño utilizados.

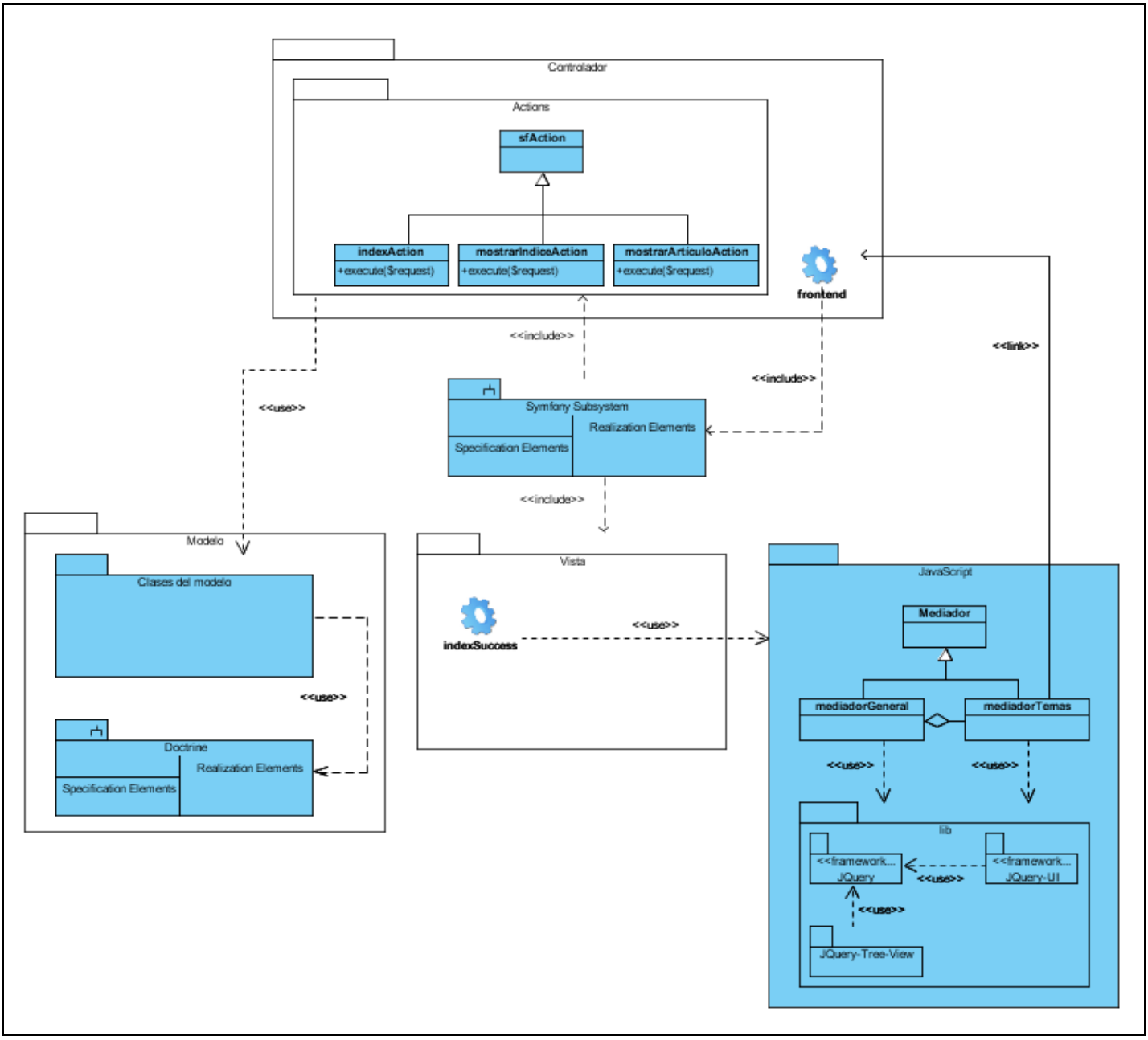
DCD Gestionar artículo



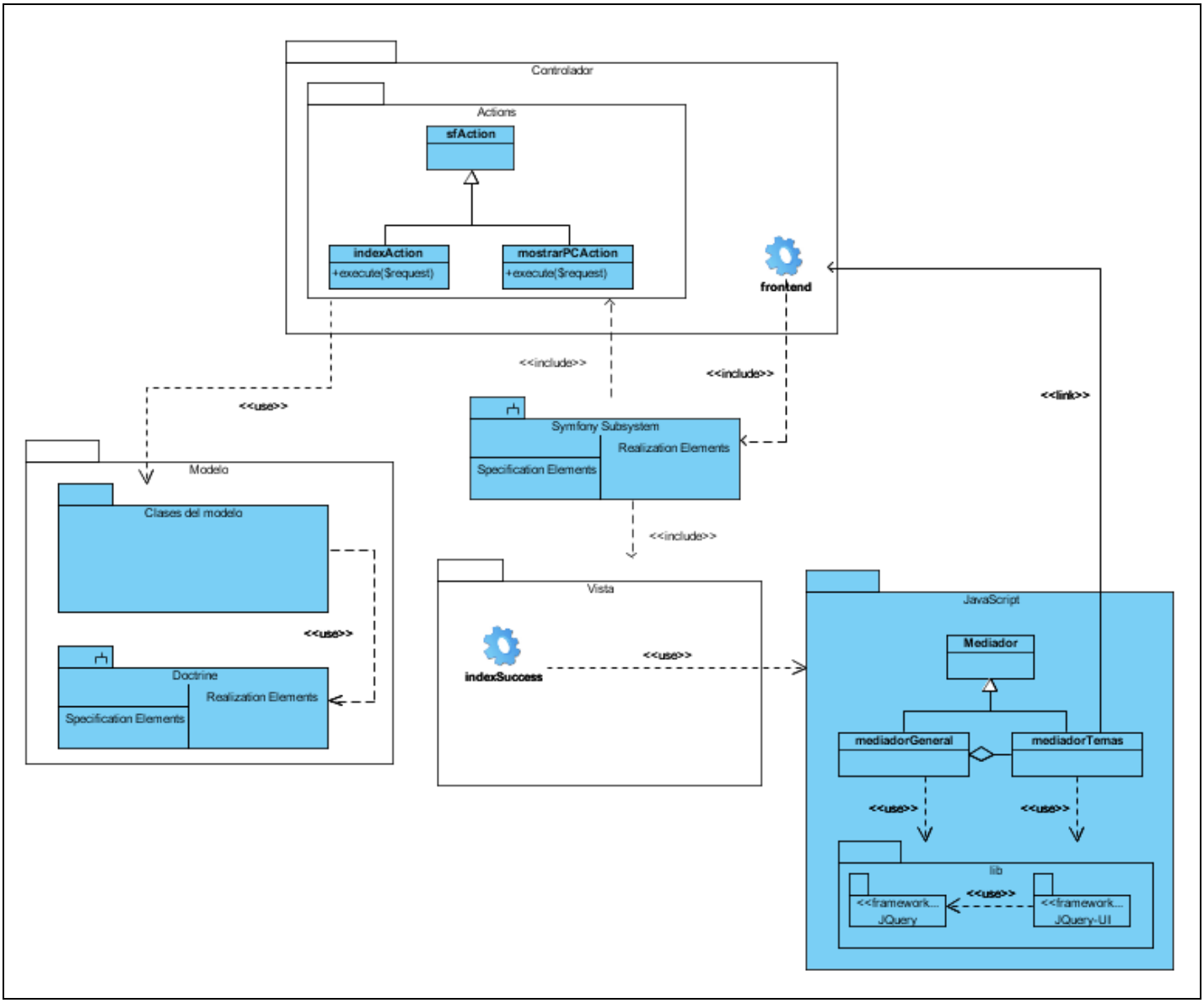
DCD Administrar palabra caliente



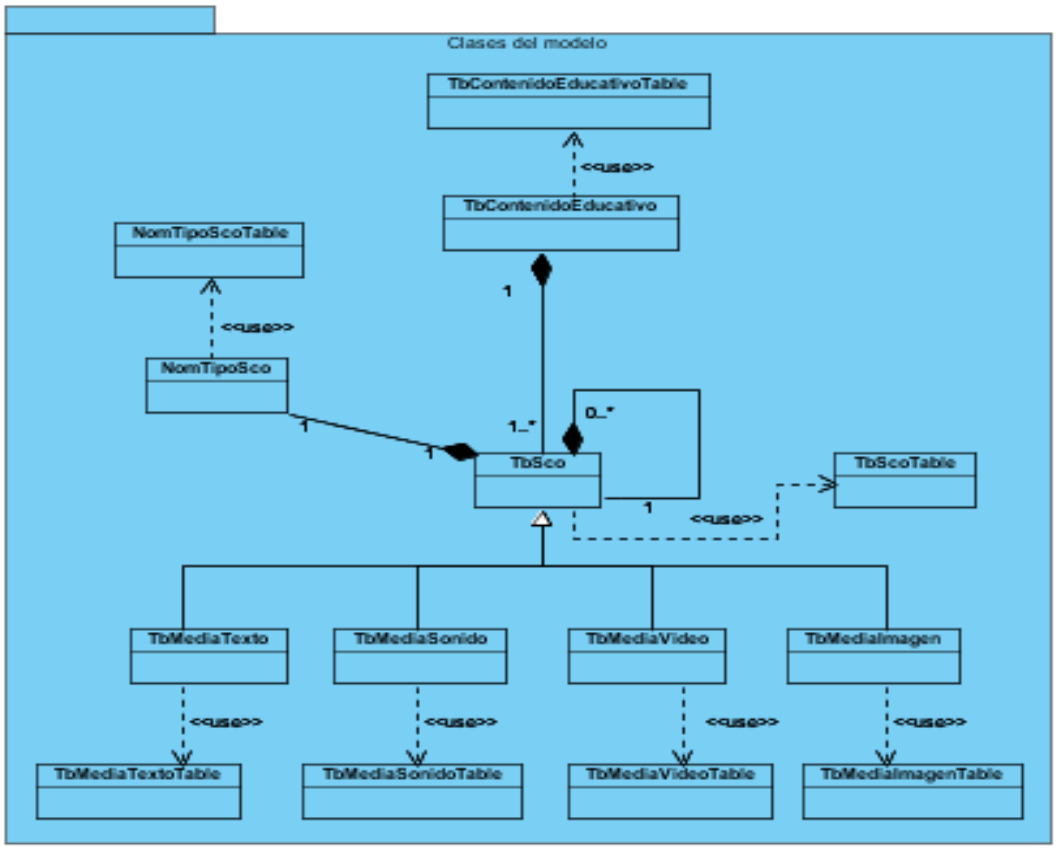
DCD Consultar artículo



DCD Consultar palabra caliente



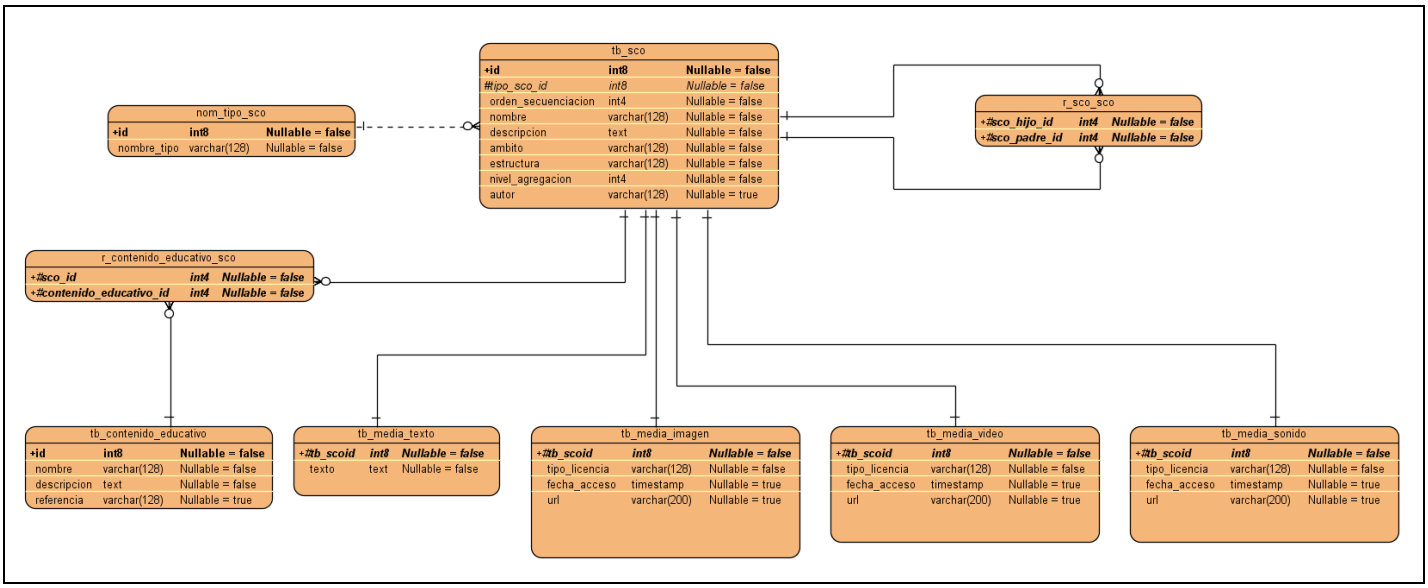
Clases del modelo



3.6 Diseño de la base de datos

A continuación aparece el modelo entidad relación referente a la base de datos del sistema a desarrollar y la descripción de las tablas que la conforman.

3.6.1 Modelo entidad relación



3.6.2 Descripción de las tablas

tb_sco		
Descripción: En esta tabla se almacenan los datos generales de los objetos de aprendizaje.		
Atributo	**Tipo**	**Descripción**
id	integer	Número único que identifica un objeto de aprendizaje.
tipo_sco_id	integer	Identifica el tipo del objeto de aprendizaje.
orden_secuenciacion	integer	Indica el orden que ocupa el objeto de aprendizaje dentro de un contenido.
nombre	varchar	Nombre del objeto de aprendizaje.
descripcion	text	Descripción del objeto de aprendizaje.
ambito	varchar	Descripción del ambiente donde se desarrolla el objeto de aprendizaje.
estructura	varchar	Descripción de la manera en que está estructurado el objeto de aprendizaje.
nivel_agregacion	integer	Indica el nivel de profundidad que ocupa el objeto de aprendizaje con respecto al producto.
autor	varchar	Nombre del autor del objeto de aprendizaje.

Tabla 6: Descripción de la tabla tb_sco

r_sco_sco		
Descripción: En esta tabla se recogen las relaciones de pertenencia entre objetos de aprendizaje.		
Atributo	**Tipo**	**Descripción**
sco_hijo_id	integer	Identificador del objeto de aprendizaje hijo.
sco_padre_id	integer	Identificador del objeto de aprendizaje padre.

Tabla 7: Descripción de la tabla r_sco_sco

nom_tipo_sco		
Descripción: En esta tabla se recogen los tipos de objetos de aprendizaje.		
Atributo	**Tipo**	**Descripción**
id	integer	Número único que identifica el tipo de un objeto de aprendizaje.
nombre_tipo	varchar	Nombre del tipo de objeto de aprendizaje.

Tabla 8: Descripción de la tabla nom_tipo_sco

tb_contenido_educativo		
Descripción: En esta tabla se almacenan los datos de los contenidos educativos.		
Atributo	**Tipo**	**Descripción**
id	integer	Número único que identifica un contenido educativo.
nombre	varchar	Identifica el tipo del contenido educativo.
descripcion	text	Descripción del contenido educativo.
referencia	varchar	Descripción de las referencias citadas en el contenido educativo.

Tabla 9: Descripción de la tabla tb_contenido_educativo

r_contenido_educativo_sco		
Descripción: En esta tabla se recogen las relaciones entre un contenido educativo y sus objetos de aprendizaje.		
Atributo	**Tipo**	**Descripción**
sco_id	integer	Identificador del objeto de aprendizaje.
contenido_educativo_id	integer	Identificador del contenido educativo.

Tabla 10: Descripción de la tabla r_contenido_educativo_sco

tb_media_texto

Descripción: En esta tabla se almacenan los datos de los textos.		
Atributo	**Tipo**	**Descripción**
tb_scoid	integer	Identificador de sus datos como objeto de aprendizaje.
texto	text	Texto del objeto de aprendizaje.

Tabla 11: Descripción de la tabla tb_media_texto

tb_media_imagen		
Descripción: En esta tabla se almacenan los datos de las imágenes.		
Atributo	**Tipo**	**Descripción**
tb_scoid	integer	Identificador de sus datos como objeto de aprendizaje.
tipo_licencia	varchar	Nombre de la licencia bajo la cual se encuentra liberada la imagen.
fecha_acceso	timestamp	Fecha del último acceso a la imagen.
url	varchar	Dirección donde se encuentra la imagen en el servidor.

Tabla 12: Descripción de la tabla tb_media_imagen

tb_media_video		
Descripción: En esta tabla se almacenan los datos de los videos.		
Atributo	**Tipo**	**Descripción**
tb_scoid	integer	Identificador de sus datos como objeto de aprendizaje.
tipo_licencia	varchar	Nombre de la licencia bajo la cual se encuentra liberado el video.
fecha_acceso	timestamp	Fecha del último acceso al video.
url	varchar	Dirección donde se encuentra el video en el servidor.

Tabla 13: Descripción de la tabla tb_media_video

tb_media_sonido
Descripción: En esta tabla se almacenan los datos de los sonidos.

Atributo	Tipo	Descripción
tb_scoid	integer	Identificador de sus datos como objeto de aprendizaje.
tipo_licencia	varchar	Nombre de la licencia bajo la cual se encuentra liberado el sonido.
fecha_acceso	timestamp	Fecha del último acceso al video.
url	varchar	Dirección donde se encuentra el sonido en el servidor.

Tabla 14: Descripción de la tabla tb_media_sonido

3.7 Conclusiones

En este capítulo se modelaron los diagramas de clases del análisis y del diseño de la aplicación con el objetivo de transformar los requerimientos en elementos del sistema. Se analizó la implementación del patrón arquitectónico MVC que realiza Symfony y los patrones de diseño más representativos utilizados. Además, se realizó el diseño de la base de datos y la descripción de las tablas presentes en la misma.

Capítulo 4. Implementación

4.1 Introducción

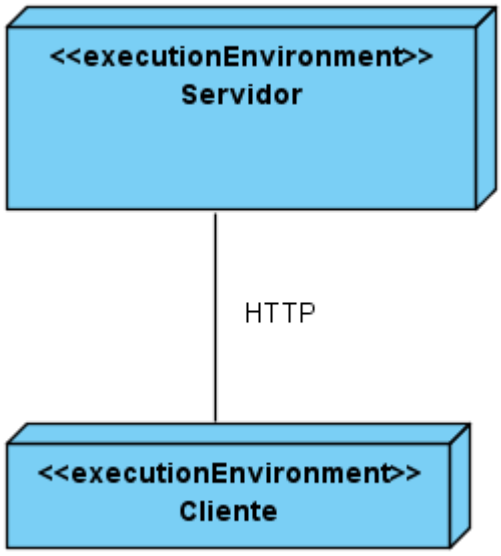
En este capítulo se documenta el proceso de implementación de los elementos identificados durante la realización del diseño. Para ello se modela el diagrama de despliegue, donde se muestra la distribución física de los elementos de hardware que conformarán el sistema y las relaciones entre ellos; y el diagrama de componentes, donde se representa la organización y las dependencias lógicas que existen entre los componentes del software. Además, se explica de forma general la integración del módulo *Temas* con la línea base de la arquitectura.

4.2 Modelo de implementación

4.2.1 Diagrama de despliegue

A continuación se presenta el diagrama de despliegue propuesto para el sistema:

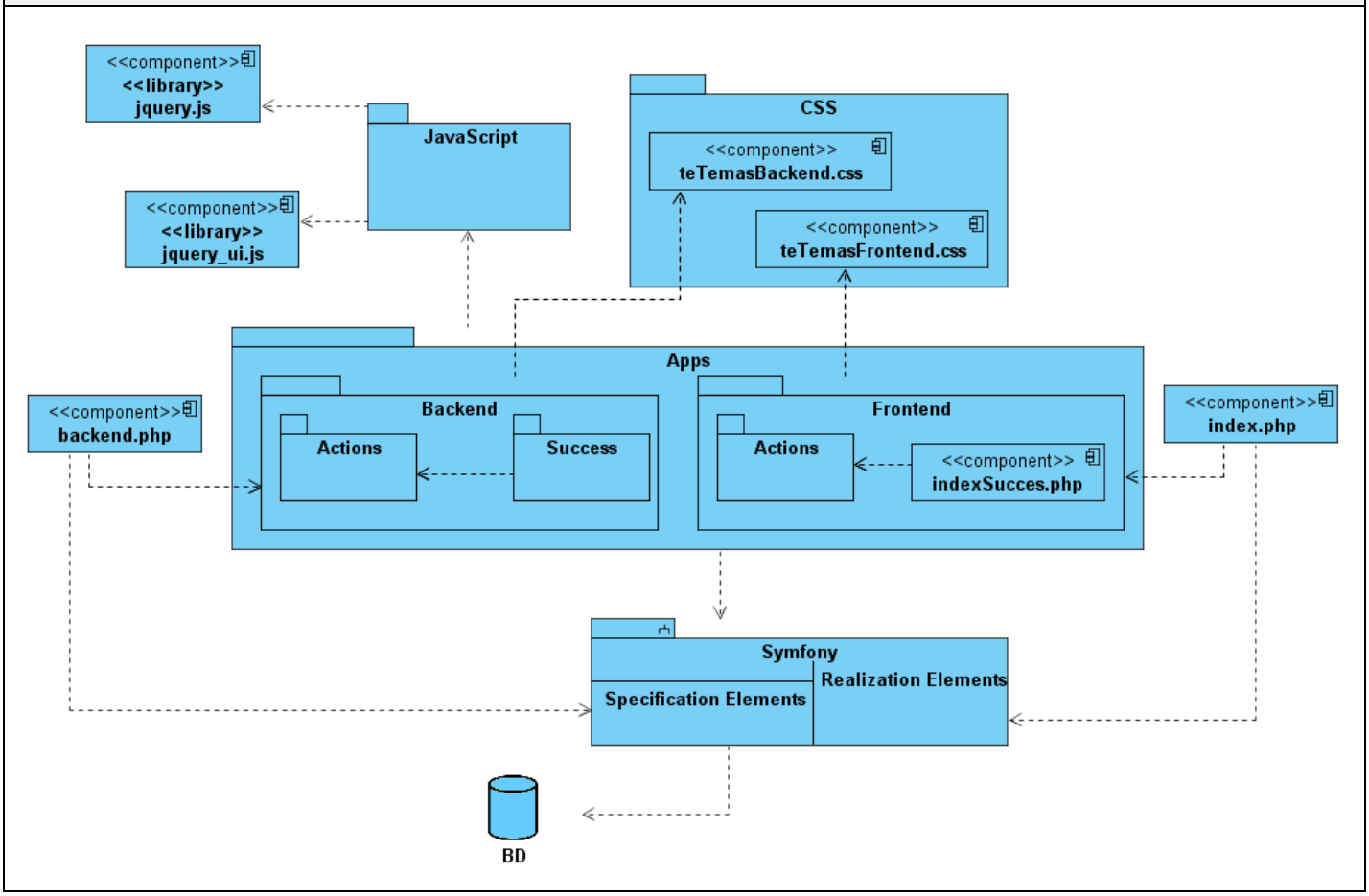
Diagrama de despliegue



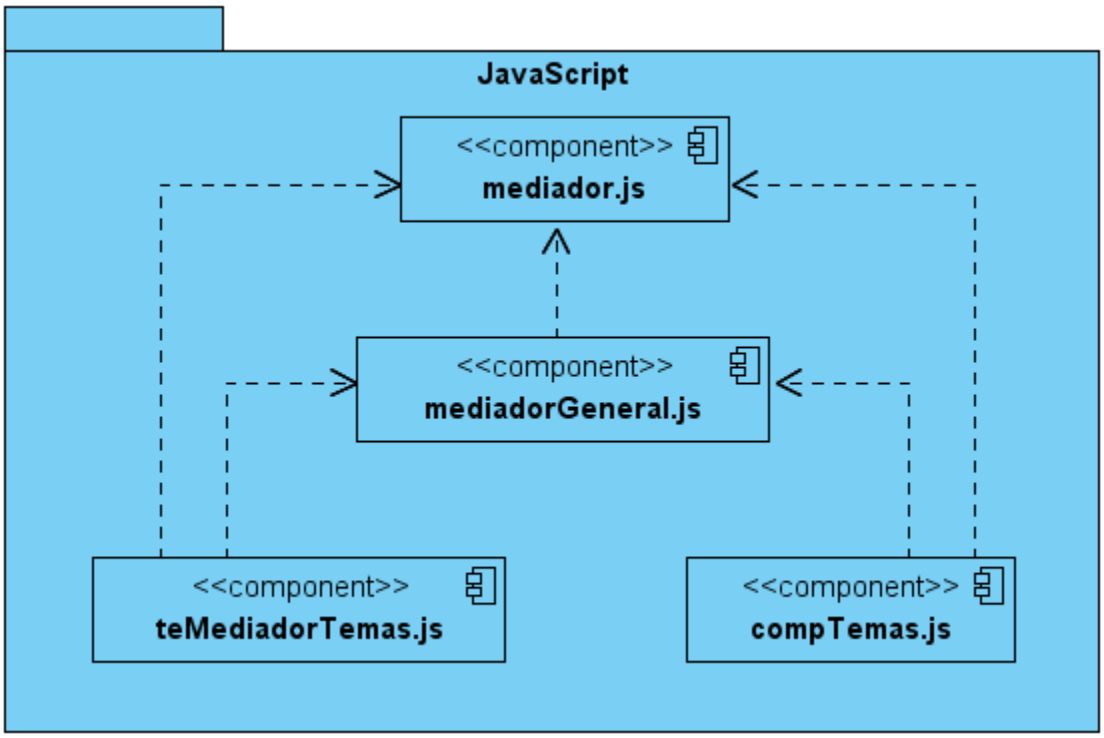
En el nodo *Servidor* se instala el servidor web Apache y el gestor de bases de datos PostgreSQL. El nodo *Cliente* permite a los usuarios interactuar con la aplicación y debe tener instalado alguno de los navegadores propuestos en los requisitos no funcionales de software.

4.2.2 Diagrama de componentes

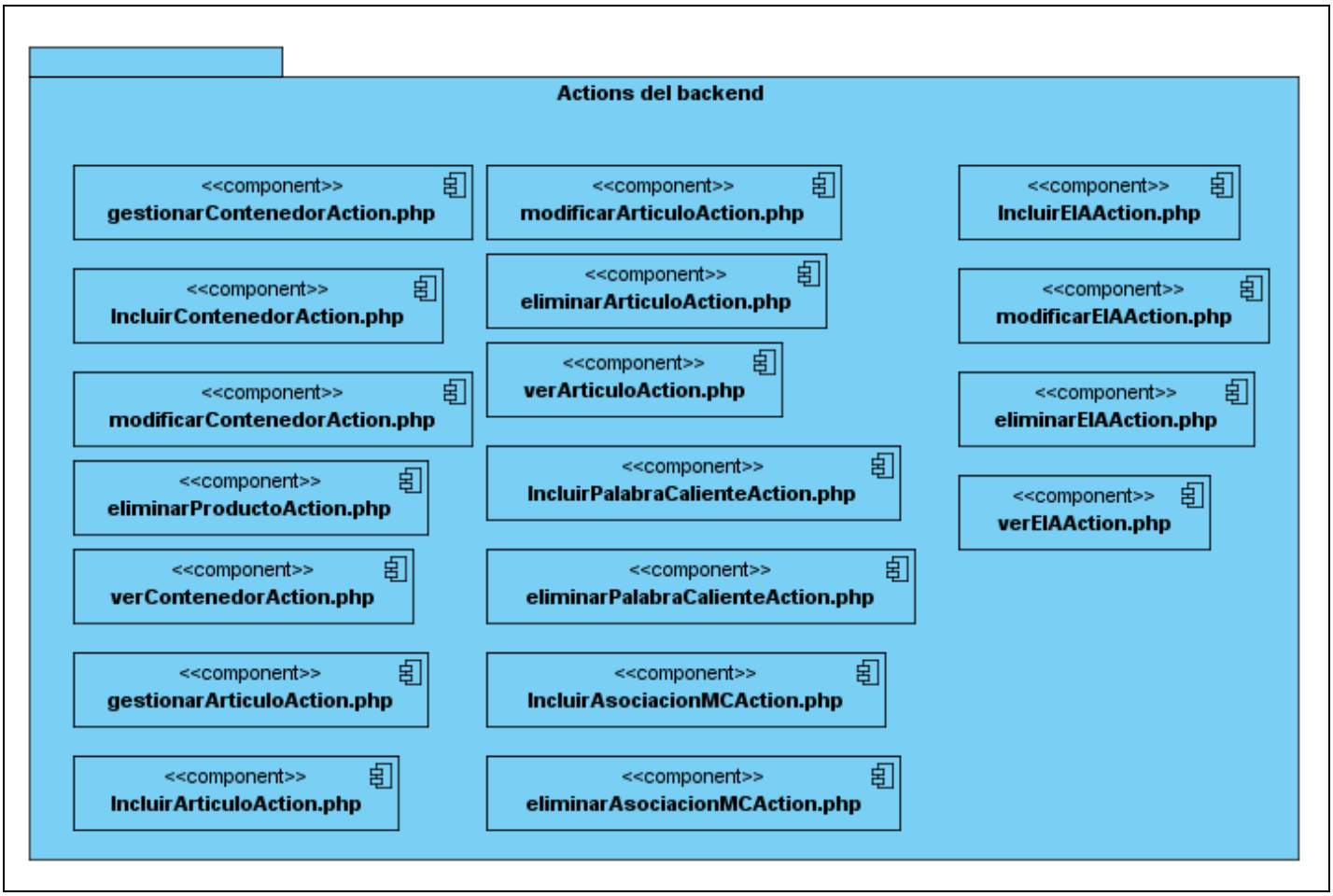
Diagrama de componentes



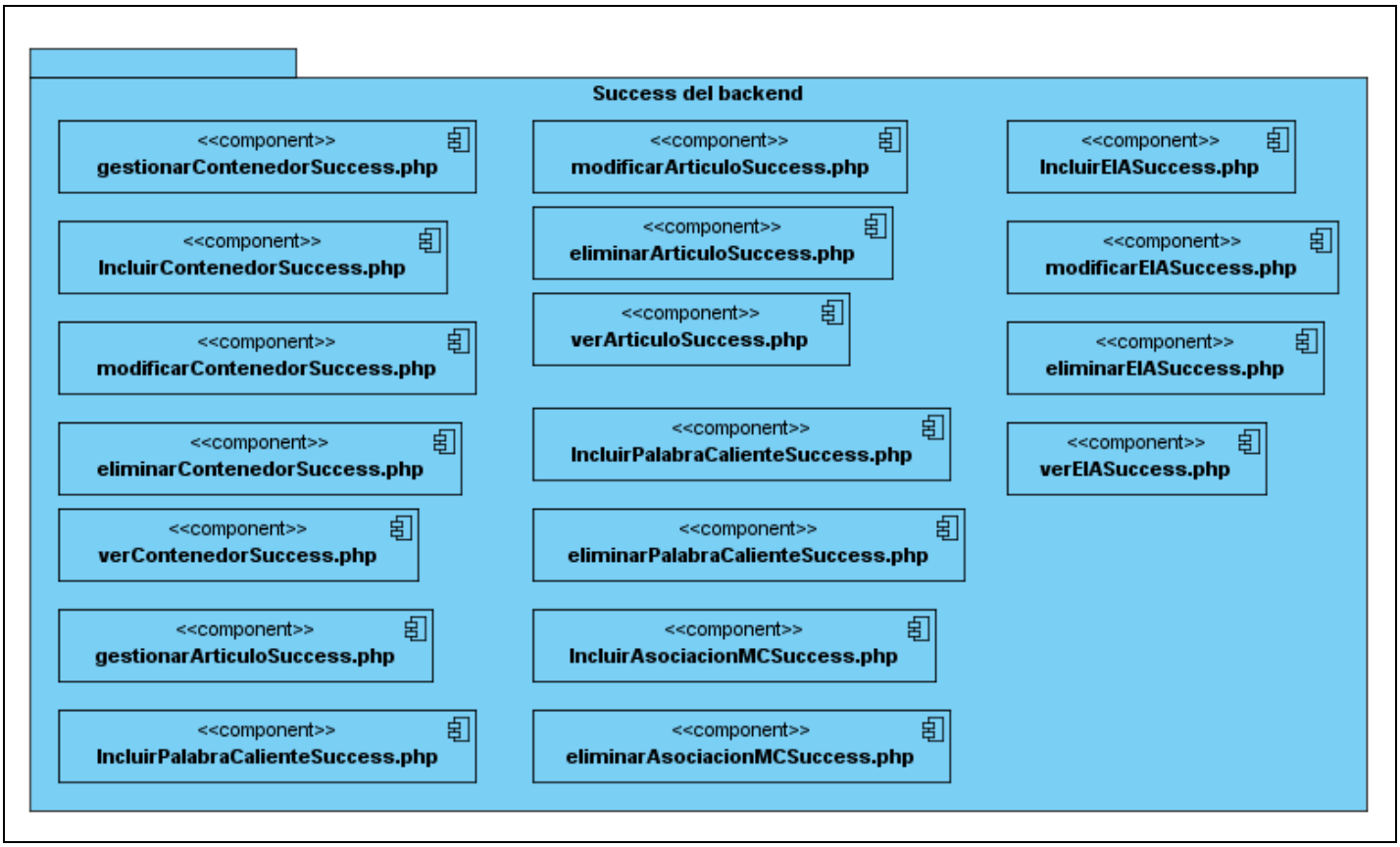
Paquete JavaScript



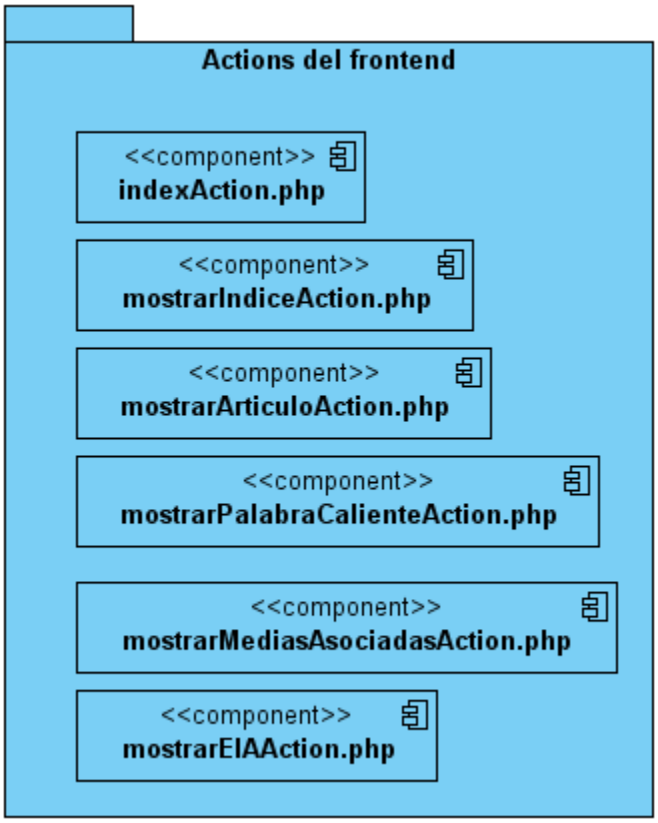
Paquete Actions del Backend



Paquete Success del Backend



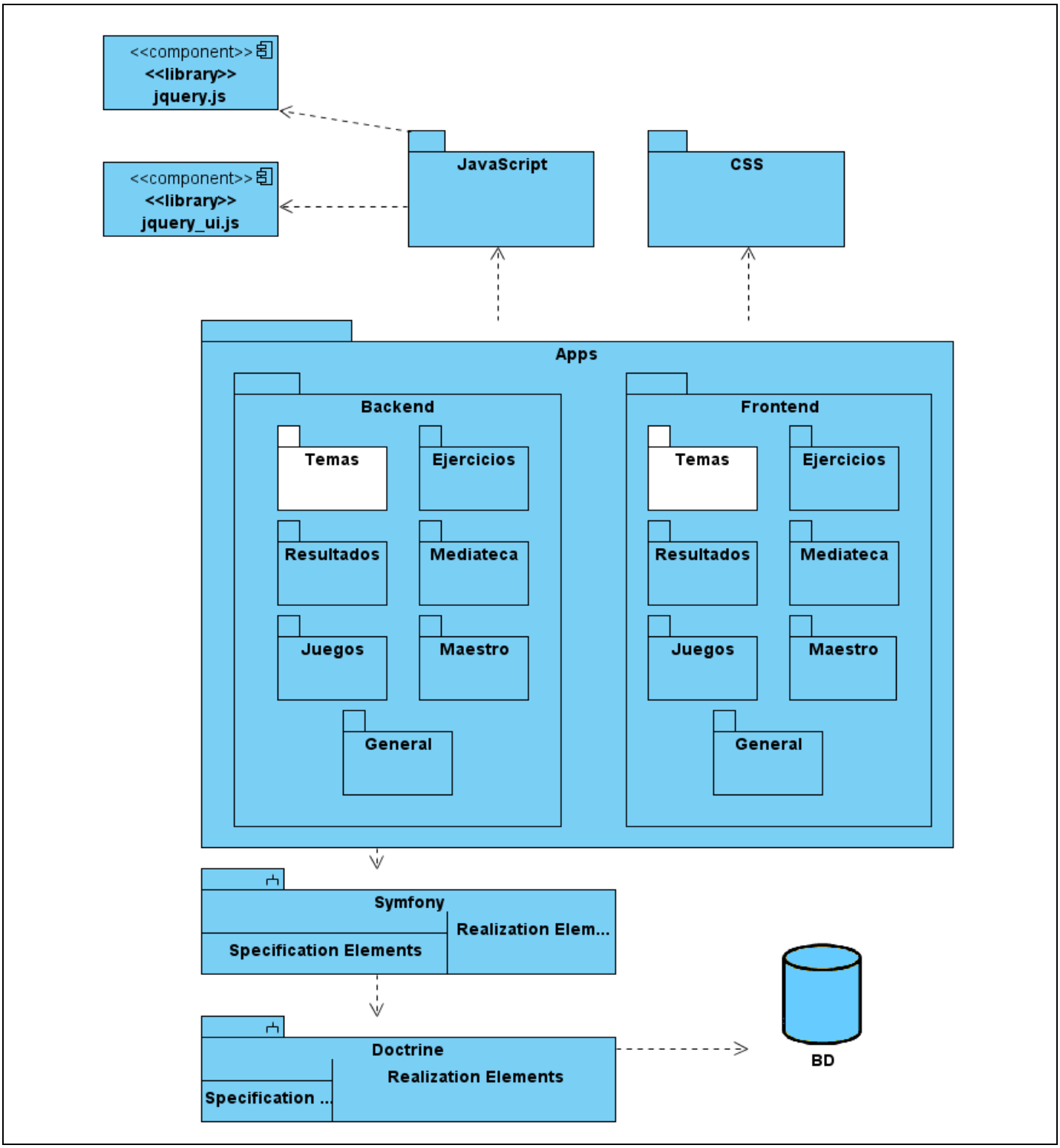
Paquete Actions del Frontend



4.3 Integración del módulo Temas

El proceso de desarrollo del módulo *Temas*, desde su concepción, estuvo muy relacionado con la arquitectura definida para todo el producto. Como se muestra en el siguiente diagrama de componentes, que forma parte de la vista de implementación de la arquitectura, el módulo *Temas* depende funcionalmente de varios elementos de la misma.

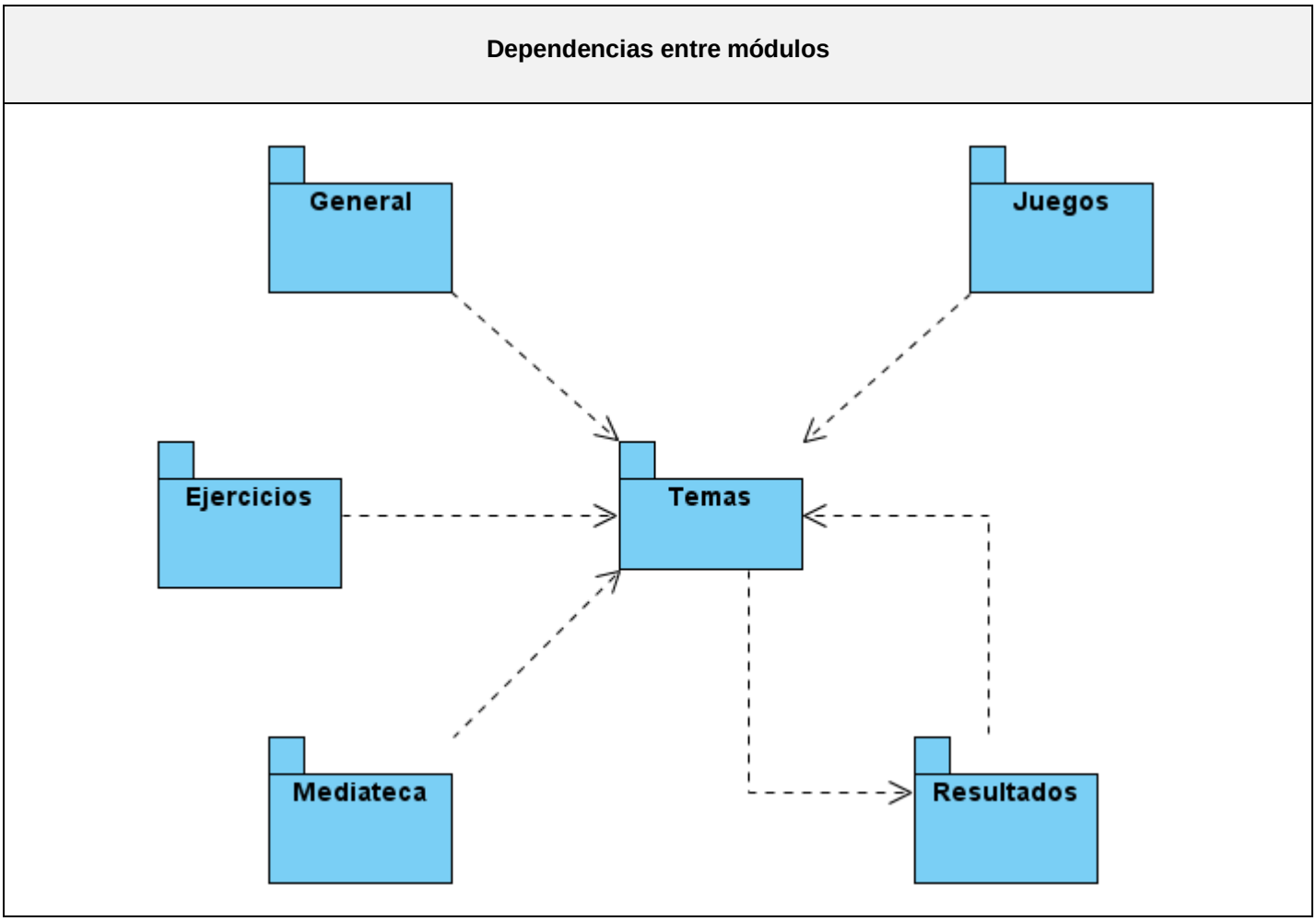
Vista de implementación de la arquitectura


<<component>>
<<library>>
jquery.js
<<component>>
<<library>>
jquery_ui.js
JavaScript
CSS
Apps
Backend
Temas
Ejercicios
Resultados
Mediateca
Juegos
Maestro
General
Frontend
Temas
Ejercicios
Resultados
Mediateca
Juegos
Maestro
General
Symfony
Realization Elem...
Specification Elements
Doctrine
Realization Elements
Specification ...
BD

Producto de estas dependencias el módulo nunca funcionó aislado, garantizándose así la integración del mismo en la línea base de la arquitectura a medida que iba siendo desarrollado.

Con el objetivo de verificar la integración con el resto de los módulos, fueron identificadas y probadas las interacciones entre estos y el módulo *Temas*. En el siguiente diagrama aparecen representadas las dependencias entre los módulos.

Dependencias entre módulos



Los módulos *Ejercicios*, *Mediateca*, *Juegos* y *Resultados* utilizan el módulo *Temas* ya que los elementos que se manejan en ellos están asociados directamente al contenido del producto. El módulo *General* depende del módulo *Temas* para el correcto funcionamiento del servicio *buscador*. A su vez, el módulo *Temas* utiliza al módulo *Resultados* para dejar trazas sobre el contenido consultado por los usuarios.

4.4 Conclusiones

En este capítulo, como parte del flujo de trabajo de implementación, se modelaron los diagramas de despliegue y de componentes. También se analizaron las dependencias del módulo *Temas* con respecto a la línea base de la arquitectura y al resto de los módulos; lo cual constituyó un aspecto fundamental en la correcta integración del mismo.

Conclusiones generales

El resultado fundamental de esta investigación es la obtención de un módulo que garantiza la gestión de contenidos en la versión multiplataforma de la colección El Navegante. Para lograrlo primeramente se analizaron los sistemas de gestión de contenidos y las principales tendencias en cuanto a la gestión de contenidos para el aprendizaje. Además, se analizaron las herramientas y tecnologías seleccionadas por el equipo de arquitectura del proyecto para el desarrollo de la solución. Siguiendo la metodología de desarrollo RUP se realizó un modelo del dominio donde se recogieron los principales conceptos relacionados con el ambiente donde será desplegado el sistema y con la colección El Navegante desarrollada por el MINED, se identificaron las características y propiedades del sistema a desarrollar y se modelaron y describieron los casos de uso del sistema. Por cada caso de uso se realizaron los diagramas de clases del análisis y diseño, lo que apoyó la implementación del módulo. Para comprobar la integración de la solución se identificaron las dependencias del módulo *Temas* con respecto a la línea base de la arquitectura y al resto de los módulos.

Recomendaciones

A modo de recomendación, se propone que se realicen las actividades sugeridas por la metodología RUP para el flujo de trabajo Pruebas con el objetivo de garantizar la calidad del módulo desarrollado. Además, se recomienda que en versiones futuras del producto sea posible exportar el contenido gestionado siguiendo completamente el estándar SCORM con la finalidad de poderlo reutilizar en otras herramientas.

Bibliografía

Trabajos citados

1. **Franco, Miguel.** Aula21. [En línea] [Citado el: 10 de Enero de 2010.] www.aula21.es/aula/spip.php?article6.

2. [En línea] [Citado el: 10 de Febrero de 2010.] http://www.e-abclearning.com/content/view/42/90/.

3. [En línea] [Citado el: 12 de Febrero de 2010.] www.elearningamericalatina.com/edicion/noviembre1_2004/it_1.php.

4. **Jacobson, Booch y Rumbaugh.** *Proceso Unificado de Desarrollo.*

5. [En línea] [Citado el: 15 de Febrero de 2010.] http://netbeans.org/index_es.html.

6. [En línea] [Citado el: 17 de Febrero de 2010.] http://www.librosweb.es/symfony/capitulo1/symfony_en_pocas_palabras.html.

7. [En línea] [Citado el: 17 de Febrero de 2010.] www.maestrosdelweb.com/editorial/javascript-facil-y-rapido-con-jquery.

8. [En línea] [Citado el: 17 de Febrero de 2010.] http://www.maestrosdelweb.com/editorial/dom.

10. [En línea] [Citado el: 10 de Marzo de 2010.] http://www.alegsa.com.ar/Dic/metadato.php.

11. Microsoft. [En línea] [Citado el: 10 de Marzo de 2010.] http://exequielc.wordpress.com/2007/08/20/arquitectura-modelovistacontrolador.

12. **Sicilia, Miguel Ángel.** Capítulo 16. - Estándares de e-learning. Buenas prácticas de e-learning. [En línea] [Citado el: 30 de Marzo de 2010.] www.buenaspracticas-elearning.com/capitulo-16-estandares-e-learning.html.

Trabajos consultados

1. **Franco, Miguel.** Aula21. [En línea] [Citado el: 10 de Enero de 2010.] www.aula21.es/aula/spip.php?article6.

2. [En línea] [Citado el: 10 de Febrero de 2010.] http://www.e-abclearning.com/content/view/42/90/.

3. [En línea] [Citado el: 12 de Febrero de 2010.] www.elearningamericalatina.com/edicion/noviembre1_2004/it_1.php.

4. **Jacobson, Booch y Rumbaugh.** *Proceso Unificado de Desarrollo.*

5. [En línea] [Citado el: 15 de Febrero de 2010.] http://netbeans.org/index_es.html.

6. [En línea] [Citado el: 17 de Febrero de 2010.] http://www.librosweb.es/symfony/capitulo1/symfony_en_pocas_palabras.html.

7. [En línea] [Citado el: 17 de Febrero de 2010.] www.maestrosdelweb.com/editorial/javascript-facil-y-rapido-con-jquery.

8. [En línea] [Citado el: 17 de Febrero de 2010.] http://www.maestrosdelweb.com/editorial/dom.

9. [En línea] [Citado el: 10 de Marzo de 2010.] http://www.alegsa.com.ar/Dic/metadato.php.

10. Microsoft. [En línea] [Citado el: 10 de Marzo de 2010.] http://exequielc.wordpress.com/2007/08/20/arquitectura-modelovistacontrolador.

11. [En línea] [Citado el: 17 de Febrero de 2010.] http://meld.medbiq.org/primers/SCORM_primer_cohen.htm.

12. [En línea] [Citado el: 17 de Febrero de 2010.] http://www.acm.org/crossroads/espanol/xrds7-4/frameworks.html.

13. [En línea] [Citado el: 15 de Febrero de 2010.] http://www.eqsoft.net/presentas/introduccion_a_postgresql.pdf.

14. [En línea] [Citado el: 20 de Marzo de 2010.] http://sistemasavanzadosderecuperaciondeinformacion.iespana.es/.

15. **LEARNING OBJECT METADATA (LOM**). [En línea] http://ares.cnice.mec.es/informes/16/contenido/21.htm.

16. [En línea] [Citado el: 25 de Febrero de 2010.] http://www.librosweb.es/symfony/capitulo2/el_patron_mvc.html.

17. [En línea] [Citado el: 5 de Febrero de 2010.] http://www.packtpub.com/open-source-cms-award-previous-winners.

18. **Jacobson, Booch y Rumbaugh**. Proceso Unificado de Desarrollo.

19. **Larman, Craig**. UML y Patrones.

20. **Sicilia, Miguel Ángel.** Capítulo 16. - Estándares de e-learning. Buenas prácticas de e-learning. [En línea] [Citado el: 30 de Marzo de 2010.] www.buenaspracticas-elearning.com/capitulo-16-estandares-e-learning.html.

21. [En línea] [Citado el: 6 de marzo de 2010.] http://www.biblioweb.dgsca.unam.mx/libros/repositorios/la_web.htm

Glosario de términos

Licencia GPL: La licencia GPL o General Public License, fue desarrollada por la FSF o Free Software Foundation. Los programas bajo esta licencia pueden ser instalados, modificados y usados en un ordenador o en varios, sin limitación. Además, puede distribuirse el programa GPL tal cual estaba o después de haberlo modificado.

Metodologías: Las metodologías de desarrollo de software son utilizadas para obtener un software de alta calidad, en el tiempo planificado y que satisfaga las necesidades del cliente. En la actualidad existen muchas metodologías de desarrollo, todas creadas para solucionar problemas tanto generales como específicos de algunos tipos de proyectos de desarrollo. Las mismas se han dividido en metodologías ágiles y tradicionales.

PHP: Hypertext Pre-processor, lenguaje de programación, diseñado originalmente para la creación de páginas web dinámicas. Es utilizado fundamentalmente como lenguaje interpretado del lado del servidor.

Open Source: El usuario no depende de una compañía específica para arreglar cosas que no funcionan, además no está forzado a pagar actualizaciones anuales para tener una versión que funcione.

CASE: Computer Aided Software Engineering / Ingeniería de Software Asistida por Ordenador.

BSD: Berkeley Software Distribution permite el uso del código fuente en software no libre.

Plugins: Aplicación que añade nuevas funcionalidades a un determinado sistema ya existente.

SCORM: Sharable Content Object Reference Model (Modelo de Referencia para Objetos de Contenido Distribuibles), es un conjunto de especificaciones y estándares elaborados por distintos organismos que se postula como el modelo común para los objetos de aprendizaje.

IDE: Integrated Development Environment (Entorno integrado de desarrollo). Aplicación compuesta por un conjunto de herramientas útiles para un programador.

Debugger (Depurador): Aplicación o herramienta que permite la ejecución controlada de un programa o un código.

Framework: Estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje de scripting entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

DOM: Document object model (Modelo de objetos del documento), el DOM proporciona una representación estructurada de un documento como una página web, así como un modo definido de acceder y manipular la estructura, el estilo y el contenido del documento. (8)

Metadatos: El término no tiene una definición única. Según la definición más difundida de metadatos es que son «datos sobre datos». También hay muchas declaraciones como, informaciones sobre datos, datos sobre informaciones e informaciones sobre informaciones. (9)

LOM: Learning Object Metadata (Metadatos de objetos de aprendizaje). Estándar que agrupa los metadatos en categorías de metadatos. Más concretamente, LOM distingue 9 categorías de metadatos diferentes.

ORM: Object-Relational Mapping (Mapeo objeto-relacional). Técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional. Desde la versión 1.4, Symfony trae por defecto la implementación de un ORM llamado Doctrine.

HTML: HyperText Markup Language (Lenguaje de marcado de hipertexto) Lenguaje de marcado utilizado para describir la estructura y el contenido de una página web.

AJAX: Asynchronous JavaScript And XML (JavaScript asincrónico y XML) Es una técnica de desarrollo web para crear aplicaciones interactivas o “ricas”. Se basa en realizar las peticiones al servidor de manera asincrónica y de esta forma realizar cambios sobre la página sin necesidad de recargarlas, lo que posibilita aumentar la interactividad, velocidad y usabilidad de las aplicaciones.