Universidad de las Ciencias Informáticas

Facultad 2

***Título:* Sistema Web Inteligente apoyado en Mapas Conceptuales para la asignatura de Álgebra Lineal**

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: *Yordi Chaveco Bustamante*

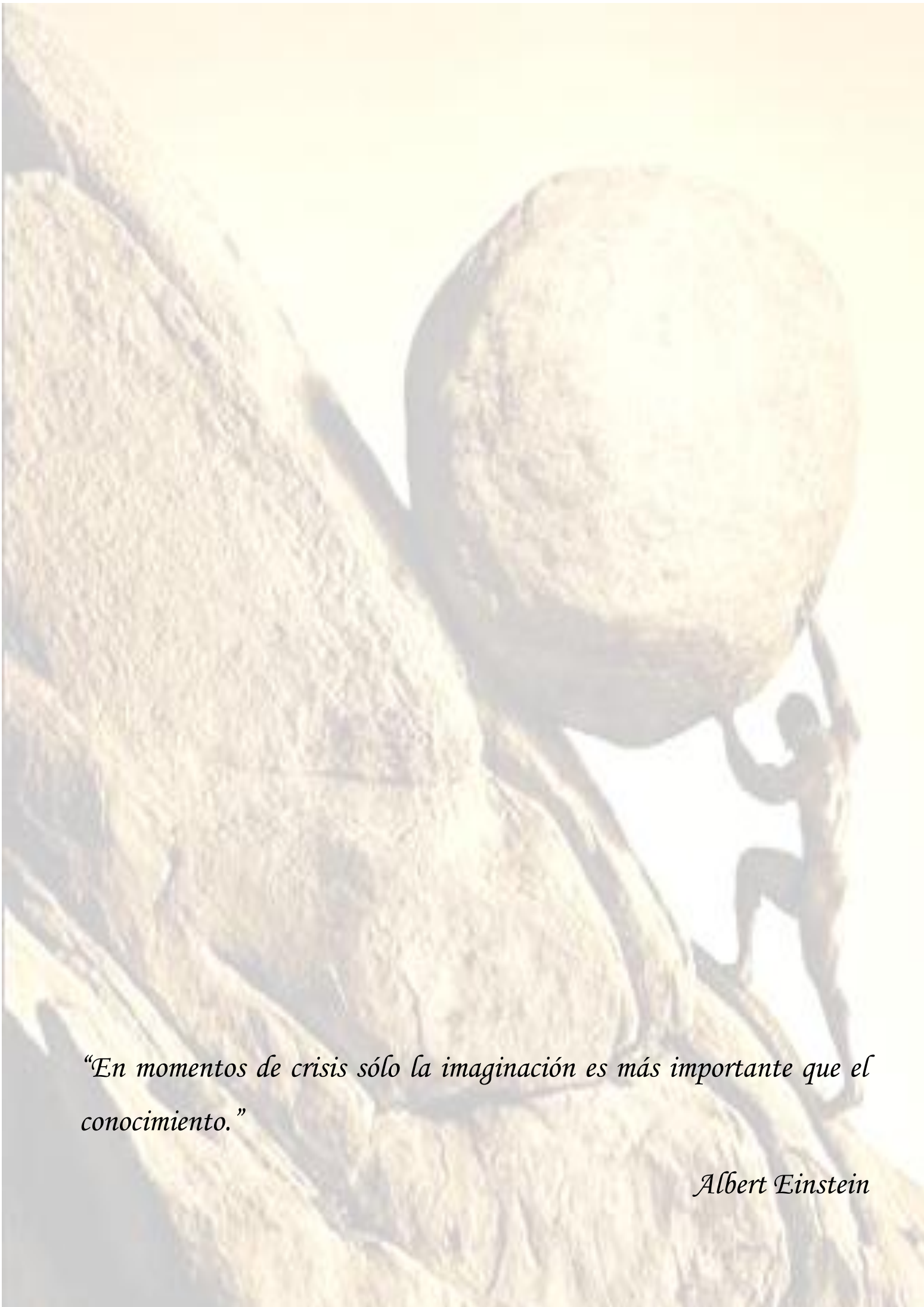
Efraín Rondón Martínez

Tutores: *Ing. Imirys Rovira Prieto*

Ing. Luis Ramón González Páez

La Habana

"Año del 54 Aniversario de la Revolución"



"En momentos de crisis sólo la imaginación es más importante que el conocimiento."

Albert Einstein

DECLARACIÓN DE AUTORÍA

Se declara ser autores de la presente tesis y se reconoce a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste se firma la presente declaración a los __días del mes de _____del año _____.

Autores:

Yordi Chaveco Bustamante. Efraín Rondón Martínez.

_____ _____

Tutores:

Ing. Imirys Rovira Prieto. Ing. Luis Ramón González Páez.

_____ _____

Resumen

Las Tecnologías de la Información y las Comunicaciones (TIC) ofrecen las condiciones necesarias para transformar el proceso de Enseñanza Aprendizaje, centrándolo en la auto preparación del estudiante, basado en los modelos de formación donde el estudiante es más responsable por su preparación, las mismas pueden brindar de este modo un apoyo necesario a las labores docentes.

Tradicionalmente se han utilizado métodos para apoyar el proceso de Enseñanza Aprendizaje, logrando con los mismos fomentar la adquisición de conocimientos en los estudiantes, entre estos métodos se encuentra la representación y organización del conocimiento en un determinado ámbito a través de Mapas Conceptuales (MC).

El desarrollo de las TIC ha introducido nuevas posibilidades en el aprovechamiento de los MC al vincularlos con Sistemas de Apoyo a la Docencia, los cuales pueden incorporar técnicas de Inteligencia Artificial (IA) para manipular conocimiento y tomar decisiones respecto a cuestiones docentes.

Teniendo en cuenta los aspectos mencionados anteriormente se propone la implementación de un Sistema de Apoyo a la Docencia para la asignatura Álgebra Lineal que incorpore MC adaptados al estado cognitivo particular de cada estudiante, utilizando la técnica de IA conocida como Razonamiento Basado en Casos (RBC) para obtener dicho carácter adaptado.

Palabras claves: Mapas Conceptuales, Inteligencia Artificial, Razonamiento Basado en Casos, Álgebra Lineal, Sistemas de Apoyo a la Docencia.

Índice de Contenidos

Índice de Contenidos

Índice de Contenidos

Índice de Tablas

Índice de Figuras

Introducción

Con el desarrollo de la humanidad el hombre ha tratado de encontrar nuevas vías que le permitan enseñar a otros diversas materias de una forma más práctica y sencilla, numerosos son los problemas que se han encontrado a la hora de transmitir el conocimiento a otras personas, uno de los principales ha sido la personalización de lo que se quiere enseñar para lograr mejores resultados, la atención personalizada es un problema pendiente aún para los pedagogos pues no siempre resulta sencillo determinar qué enseñar a quién.

Otra de las necesidades históricas del educador ha sido encontrar vías prácticas y sencillas para representar el conocimiento con el objetivo de transmitirlo de una forma más eficaz, ante este problema surgen los MC como una herramienta capaz de representar y organizar de forma precisa y sencilla el conocimiento.

Aún con la introducción de los MC el sistema de Enseñanza Aprendizaje no había encontrado la forma de enseñar de forma personalizada a los educandos, de ahí que fuera necesaria la búsqueda de otras alternativas para la enseñanza que garantizara de forma efectiva una personalización cognoscitiva en los estudiantes.

Con la introducción y desarrollo de los ordenadores se abrieron nuevas puertas para el perfeccionamiento de los procesos de Enseñanza Aprendizaje y surgen sistemas informáticos capaces de apoyar tanto al profesor como al estudiante en su desarrollo dentro de una determinada materia.

Teniendo en cuenta que los MC por sí solos no son capaces de suplir todas las deficiencias existentes, pues a pesar de ser instrumentos que combinan el rigor científico con la sencillez y flexibilidad produciendo siempre una gran acogida entre los educandos, no son capaces de personalizar el conocimiento de acuerdo a las necesidades se decide vincularlos con un algoritmo de IA para lograr el comportamiento de un sistema capaz de tomar decisiones respecto al contenido a mostrar y de mejorarse a medida que incremente sus datos. Finalizando el proceso con la representación de la información recaudada en un MC con el que el usuario podrá interactuar.

Bajo esta circunstancias, y enfocados en la realidad de nuestra universidad, se define como **situación problémica** que la Universidad de las Ciencias Informáticas (UCI) cuenta actualmente con pocos medios que permitan a los estudiantes, una vez impartidos todos los contenidos de una asignatura, medir sus conocimientos generales en la misma. Específicamente en la asignatura Álgebra Lineal los recursos existentes para este fin contienen muy pocos cuestionarios, los cuales sólo pueden ser introducidos por la asesora de la asignatura. Además contienen un carácter centralizado imposibilitando el trabajo diferenciado de un profesor con un grupo de estudiantes. Estos recursos a pesar de determinar las deficiencias de los estudiantes no son capaces de orientarlos y proponer soluciones a sus carencias cognoscitivas. Por otra parte, bajo el nuevo modelo basado en el

aprendizaje que se está aplicando, el profesor está delante del estudiante poco tiempo, razones estas por las cuales en numerosas ocasiones los estudiantes se encuentran desorientados al enfrentarse al plan de estudio de la asignatura.

Basándose en los diversos aspectos expuestos con anterioridad y bajo la situación problémica mencionada, se plantea como **problema a resolver**:

¿Cómo determinar y proponer solución a las deficiencias de los estudiantes de la UCI en la asignatura Álgebra Lineal?

Se define como **objeto de estudio**: Proceso de enseñanza de la asignatura Álgebra Lineal en la UCI, y como **campo de acción**: Herramientas computacionales en apoyo al proceso de enseñanza de la asignatura Álgebra Lineal en la UCI.

Como **objetivo general** se plantea: Desarrollar en la web un sistema inteligente apoyado en MC capaz de evaluar y orientar al estudiante de la UCI en la asignatura Álgebra Lineal.

Para lograr el cumplimiento de dicho objetivo se trazan como **objetivos específicos**:

- ✓ Realizar estudio del estado del arte.
- ✓ Realizar estudio de los Sistemas de Apoyo a la Docencia.
- ✓ Realizar un estudio de los MC.
- ✓ Realizar estudio de los temas de la asignatura Álgebra Lineal en la Universidad para conformar el MC de la misma.
- ✓ Realizar estudio de posibles técnicas de IA aplicables a la investigación para seleccionar la que mejor se ajusta a la misma.
- ✓ Proponer un algoritmo computacional donde se aplique la técnica seleccionada anteriormente.
- ✓ Analizar, diseñar e implementar un sitio web que gestione y visualice el algoritmo propuesto cumpliendo los requerimientos establecidos.
- ✓ Realizar pruebas para validar el correcto funcionamiento del software.

Se propone la siguiente **idea a defender**:

Con el Sistema de Apoyo a la Docencia de la asignatura Álgebra Lineal disponible en la web, los estudiantes de primer año en la UCI podrán determinar sus principales problemas en la asignatura y recibir una propuesta para erradicarlos.

Para cumplimentar el **objetivo general** propuesto se definen como **tareas de la investigación**:

- ✓ Estudio de sistemas para la enseñanza del Álgebra Lineal en la universidad, el país y el mundo.
- ✓ Estudio de sistemas que incorporen técnicas de IA.
- ✓ Estudio de los contenidos y conceptos necesarios para la investigación.
- ✓ Caracterización de las metodologías, herramientas y tecnologías a utilizar en el sistema.

- ✓ Estudio de las técnicas de IA candidatas a ser usadas.
- ✓ Elección de una técnica de IA para implementar el sistema.
- ✓ Levantamiento de requisitos funcionales y no funcionales.
- ✓ Descripción de los casos de uso del sistema.
- ✓ Definición de la arquitectura del sistema.
- ✓ Realización del Análisis, diseño e implementación del sistema.
- ✓ Realización de pruebas para validar el correcto funcionamiento del sistema.

Los **métodos científicos de investigación** a utilizar son los siguientes:

Métodos teóricos:

Histórico: Para el estudio del estado del arte de los sistemas de apoyo a la docencia existentes tanto en la universidad, el país y el mundo. También se aplica para la investigación del proceso de desarrollo de software así como para determinar las técnicas más usadas para el desarrollo de este tipo de sistemas informáticos.

Analítico – sintético: Este método es utilizado para analizar y comprender los procesos que se llevan a cabo en el estudio de la asignatura Álgebra Lineal por los estudiantes para poder fundamentar las formas de informatizar estos procesos.

Modelación: La modelación se utiliza para la creación de diagramas y modelos que plasmen el funcionamiento del sistema.

Métodos empíricos:

Entrevista: Se utiliza para conocer cuáles son las mayores dificultades que poseen los estudiantes a la hora de estudiar la asignatura Álgebra Lineal.

Aporte esperado del sistema:

El sistema a crear contribuirá a consolidar los conocimientos de los estudiantes en la asignatura, permitiéndoles una personalización del contenido a estudiar, logrando con esto que los estudiantes puedan determinar de una forma amena y entretenida cuáles son sus principales deficiencias en la asignatura, brindándoles a su vez la información necesaria para la erradicación de estas deficiencias

La estructura del presente documento está compuesta por:

Capítulo 1. Fundamentación Teórica:

Se hace referencia a las definiciones de los elementos esenciales en este trabajo, tales como los Sistemas de Apoyo a la Docencia, Mapas Conceptuales, entre otros. También se hace referencia a las

diferentes soluciones propuestas hasta el momento así como las diversas herramientas, técnicas y metodologías a usar.

Capítulo 2. Análisis y Diseño:

En este capítulo se hace referencia a los flujos de trabajo de análisis y diseño. Se definen los artefactos que se generan al realizarle la ingeniería de software al sistema aplicando la metodología de desarrollo definida y se define la propuesta de solución del sistema.

Capítulo 3. Implementación y Pruebas:

En este capítulo quedan plasmados los artefactos generados en los flujos de trabajo de implementación y pruebas, además se realiza una breve explicación de los aspectos fundamentales de ambos flujos.

Capítulo 1. Fundamentación Teórica

1.1 Introducción

En el presente capítulo se realiza una investigación acerca de los Sistemas de Apoyo a la Docencia sobre tecnologías web, existentes hasta el momento, exponiendo sus características fundamentales; también se muestran las principales herramientas de desarrollo, explicando sus ventajas y desventajas, así como las herramientas, lenguajes y frameworks escogidos para la realización del sitio web.

1.2 Sistemas web de apoyo a la docencia (SWAD)

Desde su creación por el inglés Tim Berners-Lee alrededor del año 1989 y su unión casi inmediata a la naciente tecnología: Internet, la web ha ido acaparando conocimientos, seguidores y nuevas tecnologías, haciéndose presente en la gran mayoría de los sectores de la sociedad. La educación se ha visto beneficiada en gran medida y son incontables los sistemas surgidos como apoyo a la docencia en todos los países del mundo. Por tanto, como lo indica su nombre, un SWAD es una aplicación informática sobre plataforma web que soluciona aspectos relacionados con un tema docente determinado.

Uno de los sistemas de este tipo más conocidos en Cuba es: *Modular Object-Oriented Dynamic Learning Environment, más conocido como Moodle por sus siglas en inglés, es un entorno virtual de aprendizaje diseñado para ayudar a educadores a crear cursos de calidad en Internet y orientado a dar soporte a un marco de educación social, se ha convertido en una de las plataformas de aprendizaje más extendidas y usadas, con una amplia comunidad de usuarios"*. (García 2008)

Para dar claridad a la idea se analizan definiciones de los entornos virtuales de aprendizaje:

- ✓ Los Entornos Virtuales de Aprendizaje (EVA) facilitan los recursos pedagógicos y los medios para la interacción/comunicación entre todos los miembros del curso, así como el seguimiento/evaluación de los estudiantes. (Belloch 2011)
- ✓ Los Entornos Virtuales de Aprendizaje (EVA) son en la actualidad el arquetipo tecnológico que da sustento funcional a las diversas iniciativas de teleformación. (Suarez 2004)

De forma general y respetando cada definición puede resumirse que los EVA son espacios obtenidos a partir del uso de tecnologías informáticas donde se llevan a cabo procesos de Enseñanza Aprendizaje.

La UCI cuenta con un EVA donde cada asignatura tiene su espacio, siempre bajo las restricciones propias del sitio. Algunas asignaturas y profesores en particular han creado sistemas propios para

apoyar la docencia. La asignatura Álgebra Lineal cuenta, como todas con su espacio dentro del EVA pero no con herramientas adicionales para mejorar el proceso de Enseñanza Aprendizaje.

1.3 Mapas Conceptuales

Los MC fueron creados por el profesor Joseph D. Novak en la Universidad de Cornell en los años 1960, el cual los definió como: *"técnica que representa, simultáneamente, una estrategia de aprendizaje, un método para captar lo más significativo de un tema y un recurso esquemático para representar un conjunto de significados conceptuales incluidos en una estructura de proposiciones."*

Básicamente se puede definir a un MC como la representación clara y precisa del conocimiento a fin de facilitarle al estudiante una mejor comprensión de la materia. Se utilizan para expresar relaciones entre ideas o conceptos así como para estructurar argumentos. Están compuestos por nodos, saetas y palabras de enlace. Los nodos representan los conceptos, mientras que las saetas unidas a las palabras de enlace representan la relación que existe entre dos o más aspectos del conocimiento.

Conceptos: "Pueden considerarse como aquellas palabras con las que se designa cierta imagen de un objeto o de un acontecimiento en nuestra mente. Algunos definen elementos concretos (mesa, computadora) y otros definen nociones abstractas o intangibles pero reales (nación, software). Los conceptos constituyen los nodos del MC" (Ojeda Cabrera 2007).

"Las palabras de enlace se escriben en la línea que une a dos nodos, pueden ser unidireccionales, bidireccionales o simplemente asociativas" (Dürsteler 2004).

"Los conceptos pueden representarse en forma jerárquica, con los conceptos más generales en la parte superior y los más particulares en la inferior" (Fernández Beltrán 2000).

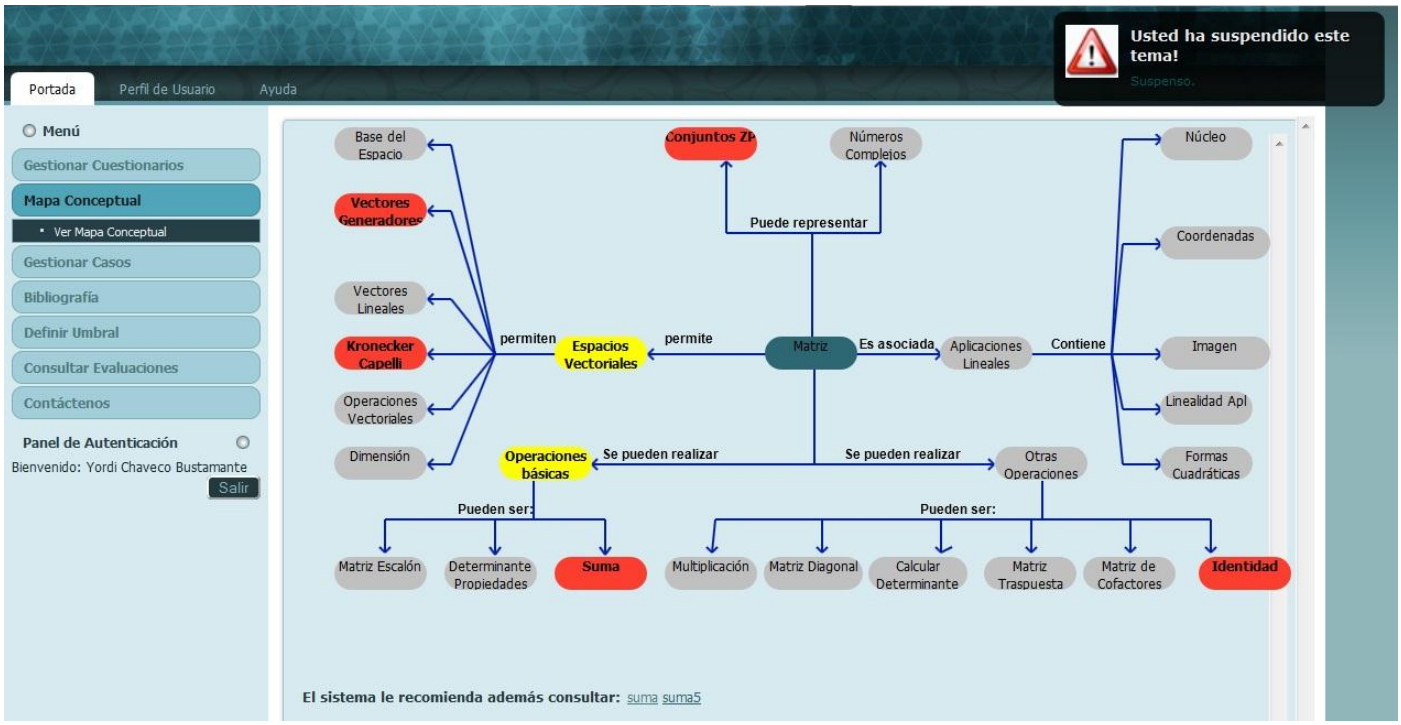


Figura 1. Representación de un Mapa Conceptual.

Numerosos han sido los desarrolladores que se han interesado por la realización de herramientas para la confección de MC, entre las principales herramientas existentes se pueden destacar algunas como:

- ✓ CmapTools
- ✓ Inspiration
- ✓ Smart Ideas
- ✓ Macosoft

Debido a que en este trabajo no se utilizará ninguna herramienta ya existente para la confección de MC, y teniendo en cuenta que el sistema a crear, partiendo de un cuestionario previo generará automáticamente el MC a utilizar, se ha decidido que la presente investigación se limite a mencionar las herramientas existentes para este fin, como se hizo anteriormente.

Luego de estudiar el programa de enseñanza de la asignatura Álgebra Lineal en la UCI se determinó realizar un MC con los temas básicos de la asignatura, los cuales son imprescindible para llegar a conocer otros de mayor complejidad, los mismos se enuncian a continuación:

- ✓ Suma de matrices
- ✓ Multiplicación de matrices

- ✓ Matriz de Cofactores
- ✓ Matriz Traspuesta
- ✓ Matriz Escalón
- ✓ Matriz Diagonal
- ✓ Matriz Identidad
- ✓ Números Complejos
- ✓ Conjuntos ZP
- ✓ Dimensión de Espacios Vectoriales
- ✓ Calcular Determinante
- ✓ Propiedades del determinante
- ✓ Núcleo
- ✓ Imagen
- ✓ Operaciones entre vectores
- ✓ Kronecker Capelli
- ✓ Sistema de vectores lineales
- ✓ Sistemas de vectores generadores
- ✓ Base de espacios vectoriales
- ✓ Coordenadas de un vector
- ✓ Linealidad de Aplicaciones
- ✓ Formas Cuadráticas

1.4 Técnicas de Inteligencia Artificial. Sistemas Basados en el Conocimiento (SBC)

La Inteligencia Artificial (IA): "Es la ciencia e ingeniería de hacer máquinas inteligentes, especialmente programas de cómputo inteligentes." (John McCarthy, 1956). A esta le conciernen dos ideas básicas: la primera es que esta involucra el estudio de los procesos del pensamiento de los humanos y la segunda que trata de representar estos procesos en una computadora. Conceptualizar estas ideas básicas condujo al desarrollo de los llamados SBC.

Los SBC son sistemas avanzados de representación y resolución de problemas complejos. Al utilizar un SBC la solución que se obtiene es similar a la solución obtenida por una persona experta en el campo del problema. Luego la información es almacenada sobre las características del estudiante para adecuar el proceso de Enseñanza Aprendizaje del mismo a la materia a enseñar.

Existen diversas formas de conocimiento y diferentes mecanismos de inferencias en los SBC, entre ellos los Sistemas Basados en Reglas (SBR), Sistemas Basados en Probabilidades (SBP), Sistemas Expertos Conexionistas (SEC) o redes expertas y los Sistemas Basados en Casos (SBC). El tipo de conocimiento determina qué método de solución de problemas es posible utilizar.

1.4.1 Sistemas Basados en Reglas

Un SBR es una forma de expresar el conocimiento atendiendo al siguiente patrón: situación - acción. Las reglas se rigen por este patrón debido a que se ejecutan únicamente en el caso de ocurrir un determinado suceso provocando una respuesta al mismo. Las reglas representan conocimiento informal o atajos, que permiten a los expertos encontrar rápidamente una solución a un problema sin tener que realizar un análisis detallado de situaciones particulares. Usualmente los sistemas basados en reglas están compuestos por:

- ✓ Reglas de producción: Una regla de producción no es más que una afirmación lógica compuesta por un conjunto de fórmulas denominadas premisas que tributan a una aserción llamada conclusión.
- ✓ Base de Conocimiento: Es donde se almacena toda la representación del conocimiento y contiene todas las reglas de producción. Sus dos principales componentes los constituyen la base de hechos y la base de reglas.
- ✓ Intérprete de reglas: Como su propio nombre lo indica es el elemento fundamental en un SBR y se encarga de analizar y procesar la información y enviar el resultado de la aplicación de la regla al lugar adecuado.

Es común en un SBR situaciones en las que la salida de una regla produzca la activación de otra regla y así sucesivamente, en este caso se estará en presencia de un encadenamiento de reglas, el cual puede ser, hacia adelante, hacia atrás o mixto.

- ✓ Encadenamiento hacia adelante: Este tipo de encadenamiento basa su funcionamiento en que el motor de inferencia parte de las premisas, e intenta llegar a deducir la conclusión. Es menos efectivo que el encadenamiento hacia atrás debido a que deberá ejecutar todas las reglas en búsqueda de hechos coincidentes.
- ✓ Encadenamiento hacia atrás: En este caso el sistema partirá de una conclusión dada e intentará encontrar un conjunto de premisas que constituyan un hecho y estén en correspondencia con el contexto de la situación analizada.
- ✓ Encadenamiento mixto: Teniendo en cuenta que tanto el encadenamiento hacia adelante como hacia atrás presentan ciertas limitaciones en la búsqueda lógica de una solución surge el encadenamiento mixto, el cual intenta aprovechar las ventajas de ambos encadenamientos, y disminuir al máximo posible sus deficiencias.

1.4.2 Sistemas Basados en Probabilidades

En los SBP la adquisición del conocimiento consiste en coleccionar muestras y realizar un procesamiento estadístico que produzca las probabilidades o frecuencias que forman la base de conocimiento. Estos utilizan estas probabilidades o frecuencias como forma de representar el problema. No son factibles para todo tipo de dominio, pues se dificulta construir las redes con ayuda de expertos humanos cuando existen carencias de conocimiento, además de que los métodos y modelos que utiliza están aún lejos de ofrecer explicaciones comprensibles.

1.4.3 Sistemas Expertos Conexionistas

En los SEC la adquisición del conocimiento incluye la selección de los ejemplos, el diseño de su topología y el entrenamiento de la red para hallar el conjunto de pesos. Facilitan el trabajo con información incompleta y brindan algoritmos poderosos de aprendizaje para crear la base de conocimiento; pero requieren de muchos ejemplos y son cajas negras que no explican cómo se alcanza la solución. Estas generalmente utilizan pesos como forma de representar el conocimiento y el cálculo de niveles de activación de las neuronas como método de solución de problemas.

1.4.4 Sistemas Basados en Casos

En los SBC la adquisición del conocimiento se reduce a la selección de un conjunto de ejemplos o casos resueltos y su organización en la base de casos. Infiere una solución mediante los casos que son relevantes al nuevo problema. Cada caso es la experiencia anterior almacenada. Su dificultad radica en la definición adecuada de la función de semejanza, al no existir una función de semejanza general apropiada para cualquier problema. Los SBC utilizan casos como forma de representar el conocimiento y el paradigma de Razonamiento Basado en Casos como método de solución de problemas.

Teniendo en cuenta que la base de conocimiento que se manipula en la propuesta de solución puede funcionar sobre la base de la concepción de ejemplos o casos resueltos, utilizando de esta forma la experiencia almacenada, resultaría conveniente la utilización de un Sistema Basado en Casos.

1.4.4.1 Razonamiento Basado en Casos

El Razonamiento Basado en Casos (RBC) (Kolodner 1993), es una perspectiva que aborda nuevos problemas tomando como referencias problemas similares que fueron resueltos en el pasado. Partiendo que problemas similares tienen soluciones similares, y la similitud juega un rol esencial (Rodríguez and García 2007). Sus componentes fundamentales son la base de casos, el módulo de recuperación de casos y el módulo de adaptación de soluciones.

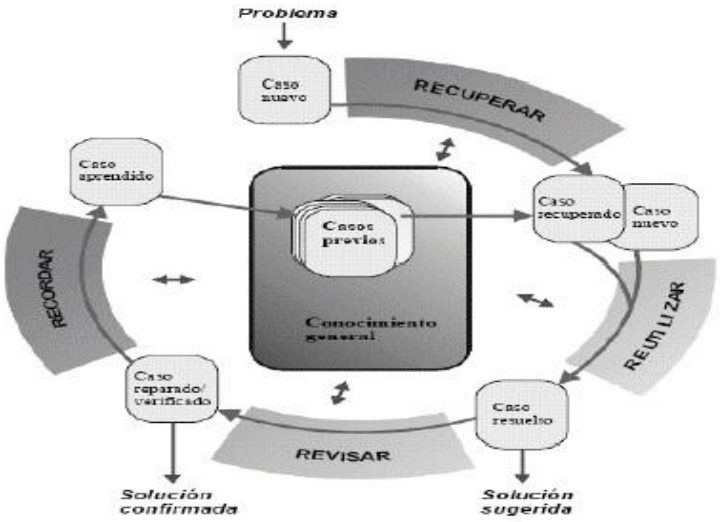


Figura 2. Ciclo de vida de un Sistema Basado en Casos.

1.4.4.2 Módulo de recuperación

En este módulo se recuperan de la base de casos los casos más semejantes al problema en cuestión. No existe una medida de semejanza única, general, para cualquier dominio, de ahí que la eficiencia del sistema depende de la función de semejanza que se defina. Constituye la primera etapa en el ciclo de vida de una SBC.

1.4.4.3 Módulo de adaptación

El módulo de adaptación es el que permite utilizar una solución ya existente una vez determinado cuáles son los casos más semejantes, aunque se debe aclarar que en ocasiones es necesario realizarle algunas modificaciones a las soluciones obtenidas. Básicamente su función es la de tomar el problema existente, buscar un problema similar, y aplicar la solución dada a ese problema al problema actual en cuestión.

1.5 Análisis de soluciones existentes

Con el avance de la informática y las técnicas de Inteligencia Artificial numerosos son los sistemas que se han desarrollado que aplican estas técnicas para perfeccionar el sistema de enseñanza aprendizaje, entre ellos se encuentran:

Herramienta de Auditoría HEDEA: Se desarrolló este software para el trabajo con un laboratorio virtual el cual brinda la posibilidad de crear un sistema partiendo de un temario de una asignatura, mediante este software el modelo del estudiante se realiza de forma automática y es transparente al usuario. El modelo del estudiante toma en cuenta los valores de experimentos previos lo cual permite darle mayor valor al historial o a su último resultado. (Romero 2009).

HESEI: es una herramienta de autor que le permite a los usuarios (no necesariamente expertos en el campo de la informática) la elaboración de Sistemas de Enseñanza de Auto Aprendizaje, cabe destacar que aunque estos usuarios no necesariamente deban poseer un gran dominio de la informática, si deben dominar la materia que imparten como docentes. Es importante saber que esta herramienta basa su funcionamiento en la elaboración de MC para la representación del conocimiento, pero presenta como inconveniente la imposibilidad de funcionar en múltiples plataformas, además es una aplicación de escritorio.

IRIS-D: La herramienta de autor IRIS-D es el resultado de la ampliación del sistema IRIS una vez integrado a éste el sistema de diagnóstico genérico: Detective (Ferrero 2001). IRIS-D es un entorno que ayuda a construir sistema, el objetivo principal es ayudar a los profesores que no necesariamente son expertos con la tecnología a construir sistemas adaptativos de Enseñanza-Aprendizaje. La función de IRIS-D consiste en adaptar, modificar y completar una arquitectura básica o genérica de un tutor para poder cumplir los requerimientos del profesor. La arquitectura de IRIS-D está integrada por 4 componentes fundamentales. (Ferrero 2001).

Plataforma SWAD, Universidad de Granada: Realiza a través de internet la mayoría de las tareas de gestión relacionadas con una asignatura y sus alumnos, a los cuales permite acceso a materiales e información y la posibilidad de autoevaluación a distancia, ampliando los medios de comunicación tanto entre alumnos como entre alumnos y profesores.

MOODLE: Visto en el epígrafe 1.2 del presente documento.

En la UCI se han desarrollado entre otros: Sistemas de apoyo a la docencia, sistemas que usan técnicas de IA, así como otros que usan MC, todos relacionados con la presente investigación, entre ellos se encuentran:

SEGEDIS: Es un Sistema Experto para el diagnóstico médico de las enfermedades genéticas con Dismorfias creado para el Centro Nacional de Genética Médica con el objetivo de proporcionar a los genetistas una herramienta en el apoyo a sus decisiones.

Análisis y diseño de una herramienta para elaborar Mapas Conceptuales Inteligentes: En la UCI se realizó el análisis y diseño de una herramienta computacional web para elaborar MCI, donde se integren los Sistemas Inteligentes utilizando los SBR y MC para facilitar a profesores el desarrollo de este tipo de Sistema de Enseñanza Aprendizaje en cualquier área del saber.

Sistema inteligente de soporte a la toma de decisiones: Se desarrolló un sistema que consiste en una aplicación Web que utiliza técnicas de inteligencia artificial y tecnologías libres para proponer a los jefes militares qué hacer en determinadas situaciones. Cuenta con una base de datos donde se aglomera de forma centralizada el conocimiento de muchos expertos militares. Se basa en la experiencia acumulada en la base de conocimientos para proponer las soluciones. Obtiene conclusiones y resuelve problemas de forma rápida. Permite diagnosticar patrones de comportamiento del enemigo. Facilita la integración con otros sistemas, así como la reutilización de sus componentes.

Entorno Virtual de Aprendizaje (EVA): Tratado en el epígrafe 1.2 del presente documento.

A pesar de existir Sistemas de Apoyo a la Docencia en la universidad, no existe uno capaz de vincular técnicas de IA, MC y los temas de la asignatura Álgebra Lineal.

1.6 Selección de las herramientas de desarrollo

Uno de los pasos más importantes en la confección de un software es la elección de las herramientas de desarrollo, pues las mismas determinarán la calidad, y el tiempo de desarrollo; además no todas las herramientas puedes ser usadas para todo tipo de software, incluso, existen gran cantidad de herramientas privativas que no permiten su uso libre. Como consecuencia se ha decidido realizar un análisis de las posibles herramientas a usar en el sistema.

1.6.1 Selección del tipo de software

En el campo de la informática se pueden destacar dos tipos de aplicaciones, las web, y las desktops. Las aplicaciones desktops tienen algunos inconvenientes, entre ellos el principal es la necesidad de instalar el software en cada ordenador donde se pretenda usar.

En cambio con el desarrollo de las redes, han aparecido las aplicaciones web, las mismas pueden ser usadas sin necesidad de instalar programa alguno, a excepción de un navegador web. Cuentan además con características que hacen más eficiente el consumo de recursos en el ordenador local pues el trabajo del ordenador se reduce sencillamente a enviar peticiones al servidor y a recibir respuestas desde el mismo, interpretándolas y mostrándolas al usuario en el navegador web. También se debe destacar que cualquier usuario con acceso a la red y con los debidos permisos será capaz de conectarse al sitio web y utilizar los beneficios del mismo, incluyendo que con el desarrollo de la telefonía móvil ya es posible acceder a este tipo de aplicaciones informáticas desde teléfonos celulares y otros dispositivos móviles, y no solo desde un ordenador como se ha hecho

tradicionalmente. Este tipo de aplicaciones al igual que las desktops permiten la asignación de permisos a determinados roles, lo cual es fundamental para mantener un sistema con la seguridad adecuada. Por los argumentos expuestos anteriormente se ha decidido la realización de una aplicación web.

1.6.2 Selección del lenguaje de programación del lado del servidor

Los lenguajes de programación web más utilizados en el servidor son Java y PHP. A continuación se describen las principales características de cada uno.

Java:

- ✓ Es un lenguaje de programación altamente potente para el trabajo web.
- ✓ Requiere de la instalación de una máquina virtual.
- ✓ Posee un código de fácil entendimiento.
- ✓ Permite la depuración de errores.
- ✓ Es capaz de soportar los nuevos paradigmas de la programación orientada a objetos, la recursividad, así como la herencia, polimorfismo y genericidad.

Lenguaje PHP (acrónimo de Hipertext PreProcesor):

- ✓ Es un lenguaje de fácil comprensión para los desarrolladores que desean aprenderlo.
- ✓ Cuenta con una comunidad de desarrolladores muy activa que constantemente ponen a disposición de todos nuevas funcionalidades de este lenguaje, y corrigen rápidamente cualquier error detectado.
- ✓ Posee gran volumen de información tanto en idioma español como en inglés.
- ✓ Es capaz de soportar los nuevos paradigmas de la programación orientada a objetos, la recursividad, así como la herencia y polimorfismo.
- ✓ Es multiplataforma pudiendo ejecutarse en cualquier sistema operativo.
- ✓ Permite la integración con todos los principales sistemas gestores de bases de datos.
- ✓ Permite el manejo de excepciones, la manipulación de cookies y la generación de páginas dinámicas.
- ✓ Es un lenguaje que se distribuye de forma libre bajo una licencia abierta.

Se escogió PHP para el desarrollo del sistema debido a las excelentes características mencionadas, es gratuito e independiente de la plataforma, rápido, con mucha documentación disponible y con una gran comunidad de desarrolladores. Soporta un gran número de bases de datos, entre ellas: MySQL, MSSQL, MSQL, Oracle, Informix, PostgreSQL y ODBC.

PHP es soportado por la mayoría de los servidores web existentes y no depende de la plataforma. Al ser libre permite su utilización en todo tipo de software sin restricciones como las que presentan los lenguajes privativos. Debido a que es un lenguaje muy sencillo de aprender el tiempo de aprendizaje del mismo es bastante corto, facilitando la preparación de nuevos desarrolladores en el caso de ser necesario. Además es importante señalar que el desarrollo de aplicaciones informáticas usando este lenguaje no requiere de potentes recursos de hardware.

Considerando que las técnicas de IA pueden ser implementadas prácticamente en cualquier lenguaje, y para lograr una mayor uniformidad en el sistema, se ha decidido utilizar también PHP como lenguaje de programación para la implementación de estas técnicas.

1.6.3 Selección del lenguaje de programación del lado del cliente

Los lenguajes HTML, CSS y JavaScript son los más usados del lado del cliente debido a la gran cantidad de funcionalidades que presentan. A continuación se realiza un breve resumen de los mismos.

HTML: Proveniente del inglés (HyperText Markup Language), en español: Lenguaje de Marcado de Hipertexto, es el lenguaje de marcado más usado en la construcción de páginas web por la comunidad de desarrolladores web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. Es compatible con un gran números de navegadores web tales como: Internet Explorer, Opera, Firefox, Netscape y Google Chrome. Además permite la incorporación de JavaScript, con lo que se puede modificar el comportamiento de una página web ante un evento determinado.

CSS: Proveniente del inglés (Cascading Style Sheets), en español: Hojas de Estilo en Cascada, es una tecnología desarrollada por el World Wide Web Consortium (W3C) que permite definir la presentación de un documento estructurado en HTML, XML o XHTML. El objetivo fundamental de este lenguaje es separar la estructura de un documento de su presentación. CSS beneficia la accesibilidad, principalmente por la separación de la estructura y la presentación de un documento. Permite un control preciso, fuera del marcado, del espaciado de caracteres, alineación de texto, posicionamiento de objetos en la página, salidas auditivas y habladas, características de fuentes, entre otros. Mediante la separación del estilo y el marcado, se puede simplificar y limpiar el HTML de sus documentos, haciendo al mismo tiempo más accesibles los documentos. CSS permite controlar

los tamaños de fuente, color y estilo, y proporciona un control más preciso sobre la presentación de contenido alternativo que HTML solo.

Las principales ventajas que ofrecen las hojas de estilo son las siguientes:

- ✓ Agiliza en gran medida la actualización de un sitio web debido a que proporciona un control centralizado de la presentación.
- ✓ Es posible personalizar a los usuarios las hojas de estilo a usar, facilitando así la accesibilidad al sitio. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
- ✓ Una página puede presentar diferentes hojas de estilo ante diferentes eventos o usuarios.
- ✓ El documento es más fácil de entender al no tener mezclado código HTML con otro tipo de código para estilizarlo, además reduce considerablemente el tamaño del software (siempre y cuando no se utilice estilo en línea).

JavaScript: Fue desarrollado por Netscape para aumentar las funcionalidades de HTML. Es un lenguaje de script interpretado, es decir, no requiere compilación, el navegador del usuario se encarga de interpretar las sentencias JavaScript contenidas en una página HTML y ejecutarlas adecuadamente. Está diseñado para responder a eventos, por ejemplo al mover el mouse o dar un clic se ejecuta una determinada acción como respuesta al evento.

- ✓ JavaScript es un lenguaje orientado a objetos, su modelo de objetos está reducido y simplificado, pero incluye los elementos necesarios para que los scripts puedan acceder a la información de una página y actuar sobre la interfaz del navegador.
- ✓ Es dinámico, responde a los eventos en tiempo. Con esto se puede cambiar totalmente el aspecto de una página al gusto del usuario, evitándose tener en el servidor una página para cada gusto.

1.6.4 Selección del Entorno de Desarrollo Integrado

Las siglas IDE provienen de “Integrated Development Environment”, y se refiere a un entorno de programación que ha sido empaquetado como un programa de aplicación. Consta de un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Puede admitir uno o muchos lenguajes de programación. Dentro de los principales IDEs para el desarrollo web se pueden destacar:

Eclipse:

- ✓ Es un entorno integrado de desarrollo libre, distribuido bajo la licencia pública de eclipse.

- ✓ Es independiente de la plataforma.
- ✓ Incluye las herramientas de desarrollo de Java, contiene un compilador de Java interno y un modelo completo de los archivos fuentes de Java.
- ✓ Permite la detección y depuración de errores.
- ✓ Consta con un conjunto de bibliotecas que facilitan el trabajo al desarrollador.
- ✓ Es multiplataforma.

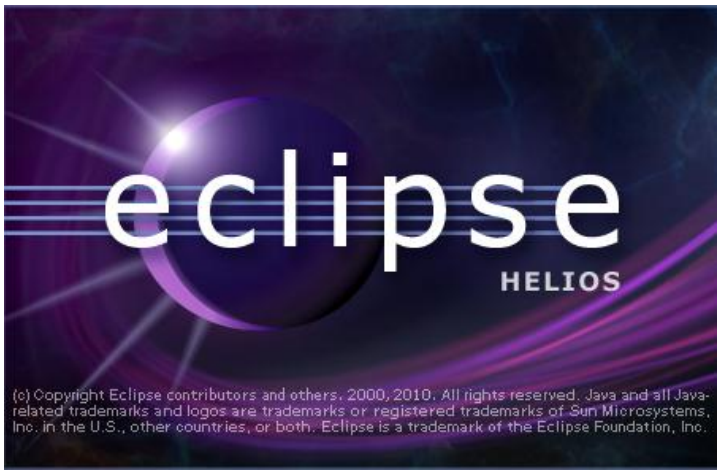


Figura 3. IDE Eclipse iniciando.

NetBeans:

- ✓ Es un entorno integrado de desarrollo libre y gratuito, distribuido bajo la general public licence (GPL).
- ✓ Está desarrollado para usarse generalmente con lenguaje Java, aunque permite su uso con otros lenguajes de programación como php.
- ✓ Cuenta con una amplia comunidad de desarrolladores que se encuentra en constante crecimiento.
- ✓ Permite crear aplicaciones web con PHP 5.
- ✓ Posee soporte para diversos frameworks, entre ellos Zend y Symfony.
- ✓ Es posible encontrarlo en múltiples idiomas y es independiente de la plataforma.
- ✓ Permite identificar errores y depurarlos.
- ✓ Permite el uso de los lenguajes de programación: JAVA, ANSI C, C++, JSP, PERL, y PHP.

Figura 4. IDE NetBeans iniciando.

Considerando que NetBeans ofrece una fácil integración con frameworks de desarrollo como Symfony y Zend, frameworks que aportan mucha seguridad al sistema y recursos al desarrollador. Y teniendo en cuenta además las diversas características expresadas de este entorno de desarrollo, se escoge es el IDE Netbeans en su versión 6.9 para la realización del sistema.

1.6.5 Sistema Gestor de Bases de Datos

Un sistema gestor de bases de datos es un programa que sirve de interfaz entra la base de datos, el usuario, y las aplicaciones que la utilizan. Tiene como objetivo fundamental manipular de forma ordenada, clara y precisa los datos permitiendo así una adecuada manipulación de los mismos, de ahí su gran importancia pues es la información el recurso más valioso en un sistema informático. Existen numerosos gestores de bases de datos, entre los principales se encuentran: MySQL y PostgreSQL de los cuales se hará una breve descripción a continuación.

MySQL:

- ✓ Es un programa de código abierto distribuido bajo la General Public Licence.
- ✓ Es multiplataforma y ampliamente usado por la comunidad de desarrolladores.
- ✓ Soporta gran cantidad de registros, teniéndose conocimiento de bases de datos de hasta 50 millones de registros totalmente funcionales.
- ✓ Ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor.
- ✓ Soporta procedimientos almacenados, triggers y vistas.
- ✓ Es muy sencillo de integrar con frameworks de desarrollo como Zend, Symfony o CodeIgniter.

PostgreSQL:

- ✓ Sistema de gestión de base de datos relacional, avanzado y orientado a objetos.

- ✓ Es una herramienta de software libre.
- ✓ Cuenta con mucha información tanto en formato duro como en la web.
- ✓ Es capaz de manejar complejas rutinas o reglas.
- ✓ Brinda soporte para subselects, triggers, vistas y procedimientos almacenados en el servidor.
- ✓ Admite una gran cantidad de usuarios de forma concurrente accediendo a la base de datos.

A pesar de ser MySQL un excelente gestor de bases de datos se opta por usar PostgreSQL para la realización de este software debido a sus excelentes características.

1.6.6 Servidor Web

Es un programa que implementa el Protocolo de Transferencia de Hipertexto (Hipertext Transfer Protocol, HTTP). Dicho programa se ejecuta constantemente en el ordenador en espera de peticiones de un cliente, en este caso un navegador web y que responde a estas peticiones adecuadamente, mostrando una página web en el navegador o un mensaje de error en el caso que exista.

Roxen:

Este servidor web fue implementado por un equipo de desarrollo sueco, se distribuye bajo la General Public Licence, es reconocido internacionalmente por permitir muchas funcionalidades. Fue desarrollado en el lenguaje "Pike".

Es multiplataforma, al poder ejecutarse sobre Windows, Linux, MAC OS/X y Solaris. Su excelente integración con bases de datos, le permite el acceso a PostgreSQL, Oracle, MySQL, entre otros. Cuenta con un alto grado de seguridad presentando un sistema de encriptación muy fuerte.

Apache:

Apache es una de las plataformas de servidores web mejor posicionadas y con más aceptación entre la comunidad de desarrolladores web, es un programa de software libre y multiplataforma que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Es una tecnología gratuita, altamente configurable y muy sencilla de utilizar e implementar, está compuesta por los siguientes módulos.

- ✓ Módulos Base: Son funciones básicas del Apache. (Van Schermbeek, 2008).
- ✓ Módulos Multiproceso: Responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos a atender las peticiones. (Van Schermbeek, 2008).
- ✓ Módulos Adicionales: Cualquier otro módulo que le añada una funcionalidad al servidor se puede añadir sin necesidad de volver a instalar el software. (Van Schermbeek, 2008).

Teniendo en cuenta que además de las características anteriores cuenta con bases de datos de autenticación, mensajes de error altamente configurables, negociado de contenido, reescritura de las URL, comprobación de la ortografía de las URL y gran cantidad de manuales on-line se ha decido usar este servidor web.

1.6.7 Metodología de desarrollo

Con el avance de la sociedad se hace más necesario el desarrollo de software que contribuya a facilitar el trabajo y la vida de las personas en las disímiles esferas sociales, no hay un área en la que el software no esté presente, incluso es un ente imprescindible en disímiles tareas, del cual en ocasiones depende la seguridad de muchas personas. Tener un software correctamente implementado y confiable es la meta superior tanto de clientes como de desarrolladores, no obstante a estos deseos, a lo largo de la historia han habido varios software que han fallado debido a una mala selección de la metodología a utilizar para desarrollarlo, y en algunas ocasiones por la ausencia total de metodologías, por la falsa creencia de muchos desarrolladores de que las mismas no son importantes y que lo único que hacen es retrasar el desarrollo del trabajo.

Las metodologías se basan en la planificación para lograr que el software sea eficiente y cumpla con los requerimientos pactados. No existe metodología universal para realizar con éxito el desarrollo de proyectos de software, las metodologías son flexibles y se adaptan a las necesidades, recursos y requerimientos de cada producto de software en particular.

1.6.7.1 Metodologías Robustas o Pesadas

Las metodologías robustas están enfocadas en el producto y en la forma en que este sea eficiente y de calidad. Las cuales establecen las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán durante el proceso de desarrollo. Aunque este término es flexible, porque las metodologías se pueden adaptar a las necesidades y requerimientos de cada producto de software en particular.

1.6.7.2 Rational Unified Process (RUP)

La metodología RUP, llamada así por sus siglas en inglés , es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML(Unified Modeling Language), el cual utiliza para preparar todos los esquemas de un sistema software, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP no es un sistema inflexible, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización.

El proceso unificado de desarrollo (RUP) es una metodología que va más allá del análisis y el diseño orientado a objetos, proporciona una familia de técnicas que soportan el ciclo completo de desarrollo

de software. El resultado es un proceso basado en componentes, dirigido por los casos de uso, iterativo e incremental y centrado en la arquitectura (Jacobson Ivar, Rumbaugh James, Booch Grady)

Beneficios que aporta RUP:

- ✓ Permite desarrollar aplicaciones sacando el máximo provecho de las nuevas tecnologías, mejorando la calidad, el rendimiento, la reutilización, la seguridad y el mantenimiento del software mediante una gestión sistemática de los riesgos.
- ✓ Permite que la producción de software cumpla con las necesidades de los usuarios, a través de la especificación de los requisitos.
- ✓ Enriquece la productividad en equipo y proporciona prácticas óptimas de software a todos sus miembros.
- ✓ Permite llevar a cabo el proceso de desarrollo práctico, brindando amplias guías, plantillas y ejemplos para todas las actividades críticas.
- ✓ Unifica todo el equipo de desarrollo de software y mejora la comunicación al brindar a cada miembro del mismo una base de conocimientos, un lenguaje de modelado y un punto de vista de cómo desarrollar software.
- ✓ Optimiza la productividad de cada miembro del equipo al poner al alcance la experiencia derivada de miles de proyectos y muchos líderes de la industria.

1.6.7.3 Metodologías Ágiles

Las metodologías ágiles ofrecen una solución para gran cantidad de proyectos pequeños. Estas se caracterizan por permitir hacer cambios de último momento y hacer entregas pequeñas de software, con ciclos rápidos y de forma cooperativa, lo que implica que el cliente y los desarrolladores trabajan juntos constantemente con una cercana comunicación.

1.6.7.4 Programación Extrema (XP)

Es la metodología más destacada de los procesos ágiles de desarrollo de software. Al igual que las otras metodologías ágiles, XP se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. XP es la primera metodología ágil y la que le dio conciencia al movimiento actual de dichas metodologías.

Objetivos de XP**:**

- ✓ La satisfacción del cliente es lo principal. Esta metodología trata de dar al cliente el software que él necesita y cuando lo necesita. Por tanto, se debe responder a las necesidades del cliente, incluso cuando los cambios sean al final de ciclo de la programación.

- ✓ Potencia al máximo el trabajo en grupo. Los jefes de proyecto, los clientes y desarrolladores, son parte del equipo y están involucrados en el desarrollo del software.

La metodología XP se basa en:

- ✓ Pruebas Unitarias: las pruebas realizadas a los principales procesos, teniendo en cuenta el funcionamiento, para en el futuro desarrollar pruebas de fallas que pudieran ocurrir, o sea obtener los posibles errores.
- ✓ Refabricación: se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- ✓ Programación en pares: una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Es como el chofer y el copiloto: mientras uno conduce, el otro consulta el mapa. Incrementa la productividad y la calidad del software desarrollado.
- ✓ Define cuatro variables para proyectos de software: coste, tiempo, calidad y ámbito.

Aunque XP es una metodología de software excelente, la metodología seleccionada es RUP debido a las características anteriormente mencionadas, además:

- ✓ Proporciona calidad, rendimiento, reutilización, seguridad y mantenimiento del software
- ✓ Es flexible, se puede ajustar a las necesidades del proyecto.
- ✓ Brinda facilidades relacionadas a la organización.

1.6.7.5 Jquery como framework para JavaScript

Jquery es un framework JavaScript que sirve para la programación avanzada de aplicaciones, el mismo aporta una serie de funciones para realizar tareas habituales, exonerando al desarrollador de realizar tareas básicas debido a que el framework las implementa automáticamente. Convierte la programación del lado del cliente en un problema más sencillo, en resumen, simplifica muchos procedimientos JavaScript que comúnmente se usan para programar la interacción de las páginas Web.

Por ejemplo, Jquery es de gran ayuda en la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de AJAX, etc. Al trabajar JavaScript con Jquery se obtiene una interfaz que con certeza funcionará correctamente en todos los navegadores. Simplemente con conocer las librerías del framework y programar utilizando las clases, sus propiedades y métodos para la creación de las aplicaciones se pueden obtener los resultados deseados.

Ventajas:

- ✓ Es un producto estable, bien documentado y con un gran equipo de desarrolladores a cargo de la mejora y actualización del framework.
- ✓ Es ligero y libre.
- ✓ Tiene una gran comunidad de creadores de plugins o componentes.
- ✓ Encadenamiento, una de las características más destacadas de Jquery es el encadenamiento, se refiere a que en una línea de código es posible cambiar un atributo a un elemento, ocultarlo y volver a mostrarlo con un efecto.
- ✓ Cuenta con una gran cantidad de selectores los que permiten tener acceso al DOM (Modelo de Objetos del Documento) de varias formas.
- ✓ Permite manipular los eventos (onclick, onmouseover, onmouseout, etc) de forma natural y permite trabajar con eventos como el funcional evento ready, dobleclick entre otros.
- ✓ De una manera sencilla, con muy pocas líneas de código es posible ejecutar acciones como post, get o un simple load gracias al AJAX con el que cuenta Jquery, eventos muy intuitivos como beforeSend, success, error, entre otros hacen muy fácil aplicar AJAX a una aplicación. (Alvarez, 2009)

1.6.7.6 Symfony como framework de desarrollo

Como todo framework de desarrollo, Symfony tiene como principal utilidad brindar a los desarrolladores la implementación de las funcionalidades más comunes y encapsular en funciones simples operaciones complejas. Symfony particularmente está diseñado para optimizar el desarrollo de aplicaciones Web. Se puede definir como un enorme conjunto de herramientas y utilidades que simplifican el trabajo de los desarrolladores web, para ello separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación.

Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios Web de comercio electrónico de primer nivel. Es compatible con la mayoría de los gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas Linux como Windows. A continuación se muestran algunas de sus ventajas. (Potencier, y otros, 2008)

Ventajas:

- ✓ Este Framework es fácil de instalar y configurar en la mayoría de las plataformas, su correcto funcionamiento se garantiza en los sistemas Windows y Linux.

- ✓ Independiente del sistema gestor de bases de datos.
- ✓ Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible y configurable como para adaptarse a los casos más complejos.
- ✓ Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador sólo debe configurar aquello que no es convencional.
- ✓ Sigue las mejores prácticas y patrones de diseño Web
- ✓ Posee buena estabilidad lo que permite el desarrollo de aplicaciones a largo plazo.
- ✓ Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- ✓ Fácil de extender, lo que permite su integración con librerías desarrolladas por terceras personas.

1.6.8 Lenguaje de Modelado

El lenguaje de modelado de objetos es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar un diseño de software orientado a objetos.

Estos muchas veces se utilizan extensivamente en combinación con una metodología de desarrollo de software para avanzar de una especificación inicial a un plan de implementación y posteriormente comunicar dicho plan a todo un equipo de desarrolladores. El uso de un lenguaje de modelado es más sencillo que la auténtica programación, pues existen menos medios para verificar efectivamente el funcionamiento adecuado del modelo.

1.6.8.1 Lenguaje Unificado de Modelado

El Lenguaje de Modelado Unificado (UML) es la sucesión de una serie de métodos de análisis y diseño orientados a objetos que aparecen a fines de los años 80 y principios de los 90. UML incrementa la capacidad de lo que se puede hacer con otros métodos de análisis y diseño orientados a objetos. Es uno de los lenguajes de modelado de sistemas de software más conocido y utilizado en la actualidad. Se utiliza para visualizar, especificar, construir y documentar los artefactos de un software.

Características del UML:

- ✓ Permite especificar todas las decisiones de análisis, diseño e implementación.
- ✓ Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- ✓ No es difícil de aprender ni de utilizar.

- ✓ Es libre.

1.6.9 Herramienta CASE

Visual Paradigm:

Se ha escogido la herramienta Visual Paradigm para UML, debido a que es una herramienta libre capaz de cubrir el ciclo de desarrollo en su totalidad: análisis y diseño orientado a objetos, construcción, pruebas y despliegue. Permite crear todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación, soporta el desarrollo de aplicaciones web y es multiplataforma.

Dicha herramienta ofrece un entorno para crear diagramas UML utilizando un lenguaje estándar para todo el equipo de desarrollo, lo que facilita la comunicación entre estos, por otra parte permite la ingeniería directa e inversa, y está disponible para múltiples plataformas, esta herramienta soporta aplicaciones web, permite generar imágenes y reportes de muy buena calidad, también admite generar código para java y exportarlo como HTML, es muy fácil de instalar y de usar.

1.7 Conclusiones

En este capítulo se profundizó en el estudio de las metodologías de desarrollo de software más usadas, se estudiaron las características fundamentales de los lenguajes de programación tanto del lado del cliente como del lado del servidor para el desarrollo de aplicaciones web, y se analizaron herramientas que dan soporte al proceso de desarrollo de software. Después de tener estos elementos se arribó a las siguientes conclusiones:

Se utilizará la metodología de desarrollo RUP por ser una metodología robusta que genera un gran volumen de información e implementa un conjunto de mejoras prácticas recomendadas para desarrollar software exitosamente. Se usará como herramienta CASE Visual Paradigm 6.4 para la especificación y el diseño de la aplicación. Los lenguajes de programación del lado del cliente que se usarán son JavaScript, CSS y HTML, y del lado del servidor PHP 5 y como servidor web se usará Apache 2.2.6 y como framework de desarrollo Symfony.

Capítulo 2. Análisis y Diseño

2.1 Introducción

En este capítulo se realiza el análisis y diseño del Sistema Web Inteligente apoyado en Mapas Conceptuales para la asignatura de Álgebra Lineal, generando fundamentalmente los diagramas de clases del análisis, los diagramas de colaboración y los diagramas de clases del diseño. Se elaborará un diagrama de clase de análisis por cada caso de uso y un diagrama de colaboración por cada sección de estos.

2.2 Propuesta de solución del sistema

El sistema permitirá a los profesores introducir cuestionarios, así como bibliografías por temas. Una vez que un estudiante accede al mismo puede realizar un cuestionario para determinar su nivel de conocimientos, una vez determinado se procede a aplicar el Razonamiento Basado en Casos para determinar casos semejantes al problema detectado, cuando el sistema ha encontrado o adaptado una solución se procede a generar un MC con el estado de cada tema. El mapa le permitirá al estudiante interactuar con sus nodos para conocer la solución que le propone el sistema.

2.2.1 Representación de la base de casos

Antes de explicar la representación de la base de casos a usar, es necesario definir la estructura que se ha utilizado para modelar un caso en el sistema.

Un caso estará compuesto por rasgos predictivos y rasgos objetivos:

- ✓ Los rasgos predictivos lo constituirán cada uno de los 22 temas básicos de la asignatura. Los mismos estarán determinados cuantitativamente por el nivel cognoscitivo detectado en sistema tras la realización del cuestionario. Los pesos definidos para cado uno de los rasgos pueden ser consultados en la sección de Anexos.
- ✓ Los rasgos objetivos lo constituirán las bibliografías que oriente el sistema como solución del caso y el nivel de éxito o fracaso que se determine.

Se utilizan muchas representaciones de casos diferentes:

- ✓ Memoria plana de pares atributo-valor: A menudo la estructura de caso simple es suficiente para resolver el problema. Además cuenta con gran facilidad de almacenamiento y recuperación en un SBC.

- ✓ Memoria Jerárquica: En casos más complejos es conveniente utilizar representaciones en forma de grafos: conjunto de nodos y arcos.

Teniendo en cuenta que el sistema almacenará casos sencillos, compuestos por informaciones básicas del estudiante, se ha decidido utilizar la representación plana, la cual tiene varias ventajas, entre ellas se puede destacar que el almacenamiento de casos se realiza de forma muy sencilla, además la representación plana necesita menos recursos para ejecutarse que una representación en forma de grafo.

La profesora Dra. Silvia Schiaffino (Schiaffino, 2012), de la facultad de ciencias exactas de la Universidad Nacional de Argentina en su investigación sobre inteligencia artificial 2012 se refirió a las siguientes formas para representar bases de casos.

- ✓ Texto libre: RBC textual
- ✓ Lista de preguntas y respuestas: RBC conversacional
- ✓ Representación tipo base de datos relacional: RBC estructural
- ✓ Videos
- ✓ Gráficos

Se ha decidido realizar la representación de la base de casos en una base de datos relacional, la misma funcionará de forma independiente a la base de datos del sistema. Las ventajas de la utilización de una base de datos para esta representación se muestran a continuación:

- ✓ Control sobre la redundancia de datos. No se almacenan varias copias de los mismos datos.
- ✓ Consistencia de datos. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente.
- ✓ Más información sobre la misma cantidad de datos. Al estar todos los datos integrados, se puede extraer información adicional sobre los mismos.
- ✓ Mejora en la integridad de datos. La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados.
- ✓ Mejora en la seguridad. Los sistemas gestores de bases de datos permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la base de datos.

- ✓ Mejora en la accesibilidad a los datos. Muchos SGBD proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos.
- ✓ Aumento de la concurrencia. En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información.
- ✓ Por otra parte al no existir la redundancia en los datos, desaparece el problema que se presentaba en el enfoque clásico, de que el cambio de un dato obligaba a actualizar una serie de ficheros. De esta forma se elimina también el inconveniente de las divergencias en los resultados debidas a actualizaciones no simultáneas en todos los ficheros.

La siguiente figura contiene una representación de la base de casos que se utilizará en el desarrollo del sistema.

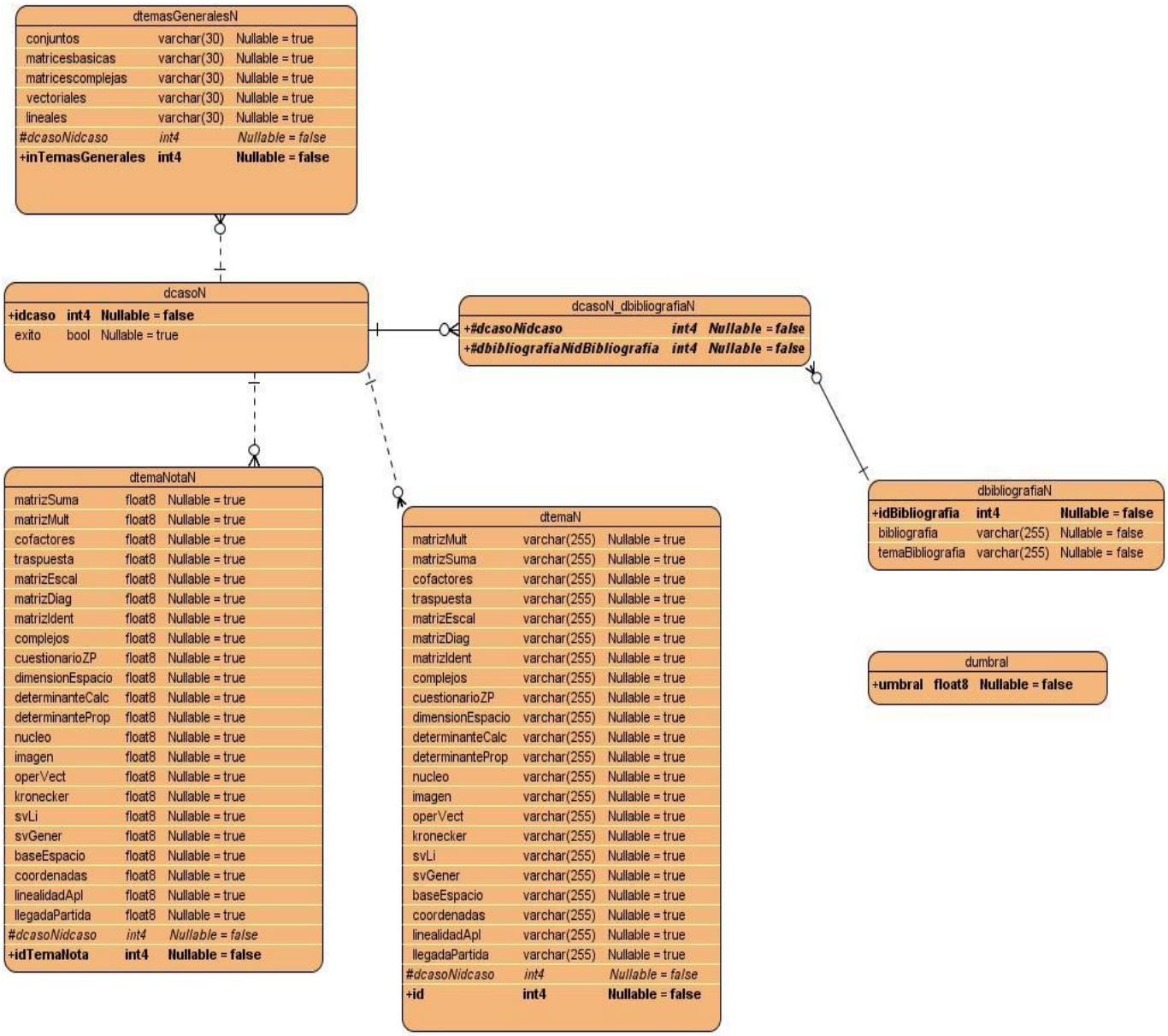


Figura 5. Representación de la base de casos del sistema.

2.2.2 Explicación detallada del algoritmo implementado

El razonamiento basado en casos comienza su ciclo en el sistema cuando un estudiante accede al sistema y realiza un cuestionario. A continuación el sistema procede a calificar el cuestionario obteniendo una nota por cada tema examinado. Con el conjunto de notas obtenido el sistema procede a determinar el nivel cognitivo del estudiante en los temas generales de la asignatura, clasificándolos de 3 formas posibles: avanzado, intermedio, o suspenso.

Si el estudiante queda aprobado en su totalidad no se almacena el caso, solamente se actualiza el perfil del usuario. En caso contrario se realiza el siguiente algoritmo.

Se accede a la base de casos para obtener los casos almacenados en el sistema, una vez obtenidos se procede a aplicar una función que se le ha denominado función de similitud; la misma es encargada de determinar cuán diferente es una nota de un tema del nuevo caso, de otra nota del mismo tema pero del caso comparado. Esta función se ha realizado sobre la concepción de que 2 notas idénticas deben devolver una similitud del 100%, y que notas como 0 y 5 deben devolver 0% de similitud, de esta forma mientras menor diferencia exista entre la nota comparada y la nota objetivo mayor será el valor obtenido, teniendo su cota mínima y máxima en 0 y 100 respectivamente.

Dicha función es la representada a continuación:

$$\mathrm{sim}(X,Y) = \ (5 - |X - Y|) * 20$$

Una vez obtenidos los valores para todos los rasgos predictivos se procede a aplicar la función de semejanza global, la misma será la encargada de determinar la semejanza entre el caso objetivo con cada uno de los casos comparados. Dicha función se representa a continuación.

$$\text{Semejanza } (C^R, C^N) = \frac{\sum_{i=1}^{n} \left(W_i * sim_i(C_i^R, C_i^N) \right)}{\sum_{i=1}^{n} W_i}$$

Luego de tener todos los valores de las semejanzas entre los casos se realiza un ordenamiento de los casos candidatos en sentido descendente, de esta forma se consigue tener los casos más semejantes en las primeras posiciones.

El siguiente paso a realizar consiste en escoger los casos que presentan un nivel de semejanza mayor que el umbral establecido, con los mismos se procede a realizar una combinación de sus soluciones para obtener la solución del nuevo caso. En caso de no existir ningún caso con el nivel de semejanza requerido se procede a determinar la solución aplicando un conjunto de condiciones de la forma if-then.

Una vez que se ha obtenido la solución del nuevo caso se procede a almacenarlo en la base de casos, con la condición que será un caso en espera de ser clasificado como éxito o fracaso. Como aún no se puede decidir si el caso almacenado será exitoso o no, no será tenido en cuenta en las soluciones de otros casos hasta que se obtenga su clasificación.

Cuando el estudiante accede nuevamente al sistema luego de haberse estudiado la bibliografía orientada como solución se le inhabilita la opción de realizar nuevos cuestionarios, y solamente se le

permite realizar el cuestionario pendiente (el que originó el caso). Si el resultado es satisfactorio el sistema clasifica el caso anterior como un éxito y pasa a formar parte de la base de casos. En caso contrario el sistema no puede clasificar directamente el caso como un fracaso debido a que un estudiante puede suspender por diversas razones independientemente de la bibliografía orientada por lo que el caso que se encontraba en espera de clasificación unido al caso nuevo son enviados a una tabla donde se encontrarán los casos en espera de revisión por un experto en la materia.

Este último caso creado se utiliza solamente para determinar si el caso principal es exitoso o no, por lo que al mismo no se le da una solución y solo se almacena en el sistema mientras dura el proceso de revisión.

La siguiente etapa en este sistema consiste en esperar que un profesor acceda al sistema, en ese caso se le listan los casos en espera de ser revisados de la siguiente forma:

- ✓ Estudiante que realizó el cuestionario.
- ✓ Primer Caso.
- ✓ Solución del primer Caso.
- ✓ Segundo Caso.

El profesor (experto en el tema), clasifica el caso como éxito o fracaso, y pasa a formar parte de la base de casos.

2.3 Modelo de Dominio

Considerando que en el entorno en el que se encuentra enmarcado el problema planteado no están definidos claramente los procesos del negocio se ha decidido la realización de un Modelo Conceptual o Modelo de Dominio, teniendo en cuenta conocimientos de expertos y análisis de soluciones similares.

El Modelo Conceptual es una representación de conceptos u objetos en el dominio del problema. La meta es lograr un conocimiento básico del vocabulario y de los conceptos que se incluyen en los requerimientos. (Larman, 2001).

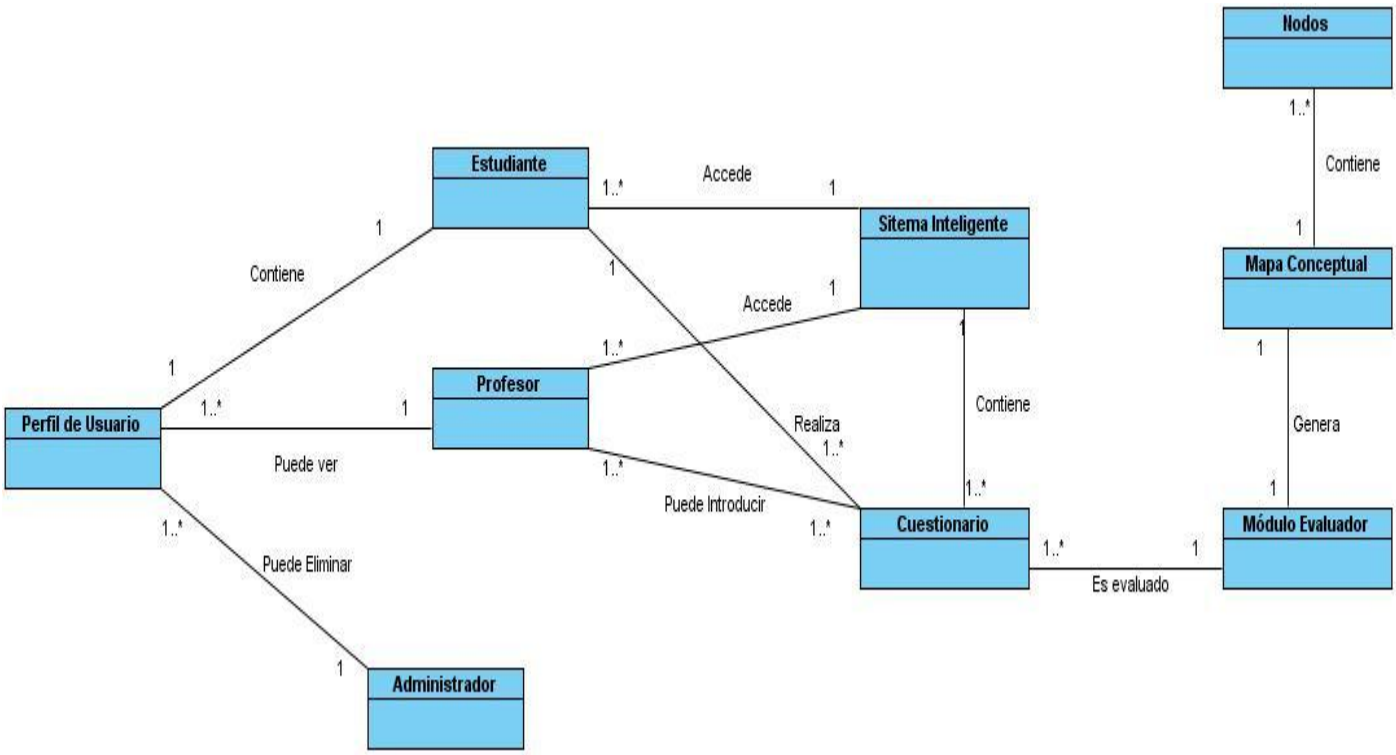


Figura 6. Modelo de Dominio del Sistema.

2.4 Requerimientos de la aplicación

Una vez que se ha analizado el dominio del problema es de vital importancia dejar plasmado qué debe hacer el sistema una vez creado. Para definir esto se han entrevistado a estudiantes y profesores vinculados directamente con esta asignatura con el objetivo de conocer cuáles deben ser los principales requerimientos que debe cumplir el sistema. Los requerimientos suelen dividirse en dos grupos: funcionales y no funcionales. A continuación se muestran los requerimientos identificados:

2.4.1 Requisitos funcionales

RF 1. Autenticar usuario: El sistema debe permitir autenticar los usuarios, para ello el usuario debe estar registrado en el dominio UCI**.**

RF 2. Eliminar Perfil de usuario: El sistema debe permitir al administrador eliminar un perfil de usuario, para ello se debe especificar el usuario del dominio UCI.

RF 3. Ver Perfil de Usuario: El sistema debe permitir a cualquier usuario ver su perfil.

RF 4. Guardar Perfil de Usuario: El sistema debe registrar el perfil de todos los usuarios que se autentiquen para ello se debe guardar el nombre completo, la facultad, el solapín, y la categoría

(estudiante, profesor o administrador), en el caso de los estudiantes deberá registrarse además el listado de notas obtenidas, el último mapa conceptual generado y una lista de bibliografías recomendadas por temas.

RF 5. Crear Cuestionario: El sistema debe brindar la posibilidad a los profesores de crear nuevos cuestionarios a través de la inserción de preguntas.

RF 6. Eliminar Cuestionario: El sistema debe brindar la posibilidad a un profesor de eliminar un cuestionario previamente creado por él.

RF 7. Consultar Cuestionario: El sistema debe brindar la posibilidad a los profesores de consultar los cuestionarios existentes.

RF 8. Eliminar Pregunta: El sistema debe permitir a los profesores eliminar una pregunta de algún cuestionario en cuestión.

RF 9. Calificar Cuestionario: El sistema debe ser capaz de permitir a los estudiantes realizar cuestionarios y obtener su calificación tanto por temas como general.

RF 10. Generar Mapa Conceptual: El sistema debe ser capaz de generar automáticamente a los estudiantes un mapa conceptual una vez realizado un cuestionario.

RF 11. Insertar Caso: El sistema debe ser capaz de insertar un nuevo caso a la base de casos.

RF 12. Eliminar Base de Casos: El sistema debe permitir al administrador eliminar la base de casos.

RF 13. Realizar Cuestionario: El sistema debe permitir a los estudiantes realizar cuestionarios.

RF 14. Revisar Casos: El sistema debe permitir a los profesores revisar los casos que se encuentran pendientes de revisión.

RF 15. Modificar Pregunta: El sistema debe permitir a los profesores modificar preguntas de los cuestionarios existentes.

RF 16. Introducir Bibliografía: El sistema debe permitir a los profesores introducir bibliografías, pues las mismas serán tomadas como soluciones de casos.

RF 17. Eliminar Bibliografía: El sistema debe permitir a los profesores eliminar una bibliografía existente.

RF 18. Consultar Evaluaciones: El sistema debe permitir a los profesores consultar las evaluaciones de los estudiantes.

RF 19. Establecer Umbral: El sistema debe permitir al administrador establecer el umbral del sistema.

RF 20. Establecer Administrador: El sistema debe permitir al administrador establecer un nuevo administrador.

2.4.2 Requisitos no funcionales

RNF 1. Accesibilidad: El sistema debe garantizar que el acceso a la información se realice de acuerdo al rol que desempeñan los usuarios.

RNF 2. Disponibilidad: El sistema debe estar disponible las 24 horas del día.

RNF 3. Rendimiento: La aplicación debe dar respuesta a las peticiones en un período corto de tiempo.

RNF 4. Portabilidad: El sistema debe ser multiplataforma.

RNF 5. Apariencia o Interfaz Externa: La interfaz del sistema debe ser sencilla e intuitiva, para que los usuarios puedan utilizar la aplicación con facilidad.

RNF 6. Software: La computadora cliente debe contar con navegador web Mozilla Firefox 4.0 o superior, Opera o Chrome. La computadora servidora debe contar con sistema operativo Linux o Windows y tener instalado el servidor Xampp 1.7.2 o superior. El servidor de base de datos debe contar con PostgreSQL en su versión 8.3 o superior y como administrador de base de datos PgAdmin 3.

RNF 7. Hardware:

Los requerimientos de hardware se ven más reflejados en este caso en la computadora servidora debido a que es la que mayor cantidad de operaciones debe realizar, por eso se definen los siguientes requerimientos mínimos de hardware.

- ✓ Microprocesador 1.5 GHz.
- ✓ 512 MB de memoria RAM.
- ✓ 10 GB de disco duro.

RNF 8. Licencia: El producto debe ser liberado bajo la licencia GNU/GPL.

RNF 9. Ayuda: El sistema debe contar con una ayuda que podrá ser accedida desde cualquier parte del mismo.

2.5 Diagrama de casos de uso del sistema

El diagrama de casos de uso del sistema permite que los clientes, usuarios y desarrolladores lleguen a un acuerdo sobre cómo deberá ser utilizado el sistema. En su mayoría, los sistemas informáticos

cuentan con varios tipos de usuarios, estos se representan en el modelo de casos de uso a través de los actores, los cuales se relacionan con los casos de uso (CU), que son segmentos de funcionalidades del sistema.

Un diagrama de casos de uso describe parte del Modelo de Casos de Uso y muestra un conjunto de casos de uso y actores con una asociación entre cada par actor/caso de uso que interactúan. (Jacobson, et al., 2000).

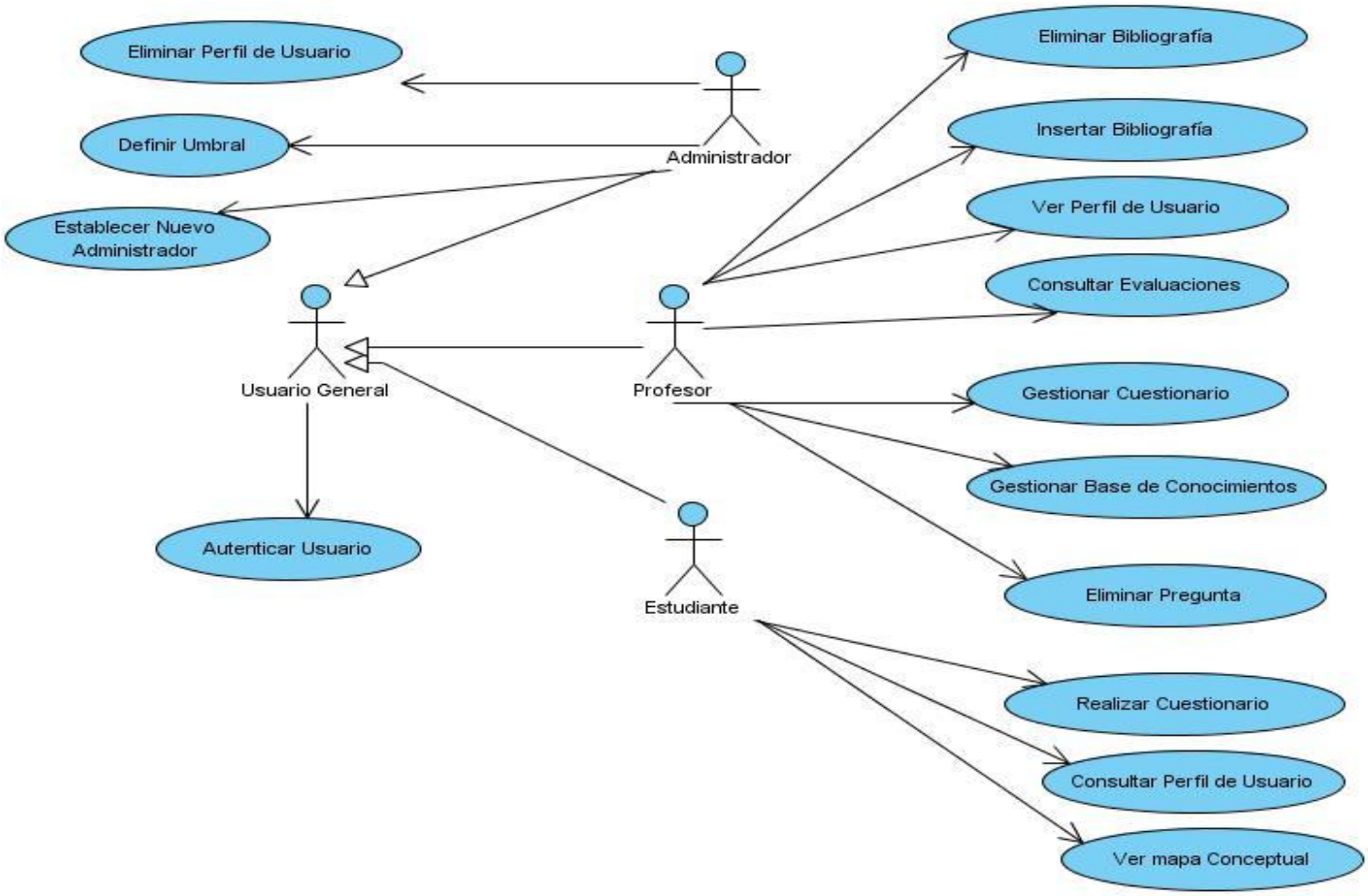


Figura 7. Modelo de casos de uso del sistema.

2.5.1 Actores del Sistema

Un actor es alguien o algo, externo al sistema, que de cierta forma interactúa con este. Los actores no son solamente roles que juegan personas, sino también organizaciones, software y máquinas. (Larman, 2001).

Para un mejor entendimiento del diagrama de casos de uso se describen a continuación los actores del sistema.

Tabla 1. Actores del Sistema.

Actor	**Descripción**
Administrador	Es el actor que posee los permisos necesarios para realizar actividades críticas en el sistema como: Establecer el umbral, eliminar la base de casos, eliminar un usuario o establecer el nuevo administrador.
Profesor	Es un actor muy importante e imprescindible para el correcto funcionamiento del sistema, Posee los permisos necesarios para gestionar cuestionarios, realizar cambios en las bibliografías, y revisar casos pendientes.
Estudiante	Es la razón de ser del sistema, sin él el sistema deja de tener sentido. Posee los permisos necesarios para realizar los cuestionarios y visualizar su mapa conceptual.

2.5.2 Descripciones de casos de uso del Sistema

Dentro de los principales elementos que contienen las descripciones textuales de los casos de uso se encuentran las Precondiciones y Postcondiciones del caso de uso, las referencias a los requisitos funcionales que satisface, así como la descripción detallada del flujo de eventos a ejecutar. A continuación, se presentan las descripciones textuales de dos de los casos de uso más significativos del sistema, ver tablas 2 y 3, el resto de las descripciones de casos de usos pueden ser consultadas en los anexos.

Tabla 2. Descripción textual: CU Realizar Cuestionario.

Caso de uso:	Realizar Cuestionario
Actor:	Estudiante
Resumen:	El caso de uso se inicia cuando el actor accede a la opción realizar cuestionario en la interfaz principal.
Precondiciones:	El actor debe estar autenticado en el sistema como estudiante y deben existir cuestionarios creados previamente.
Postcondiciones:	Se generó un mapa conceptual adaptado al estado cognitivo del estudiante, y se actualizó el perfil de usuario así como la base de casos.
Referencias:	RF-9, RF-10, RF-13
CU Asociados:	
Prioridad:	Crítico
Flujo Normal de los Eventos:	
Acción del Actor:	**Respuesta del sistema:**
1 El actor selecciona la opción Realizar Cuestionario en la interfaz principal.	2 El sistema muestra el mensaje:” Usted va a realizar un cuestionario, una vez que sean enviadas las preguntas no podrá modificarlas”.
3 El actor selecciona la opción Continuar.	4 El sistema muestra el cuestionario solicitado.

5 El actor realiza el cuestionario y presiona el botón Calificar.	6 El sistema califica el cuestionario. 7 El sistema actualiza el perfil de usuario. 8 El sistema procesa el nuevo caso y guarda el resultado. 9 El sistema genera y muestra el mapa conceptual correspondiente.
Flujo Alterno:	
*.1 El actor selecciona la opción Regresar.	*.2 El sistema regresa a la interfaz principal.
5.1 El actor deja preguntas sin responder.	5.2. El sistema muestra el mensaje de información: “No debe dejar preguntas sin responder”. 5.3. El sistema sombrea las preguntas que se han dejado de responder de color rojo.
Prototipo de Interfaz de Usuario	

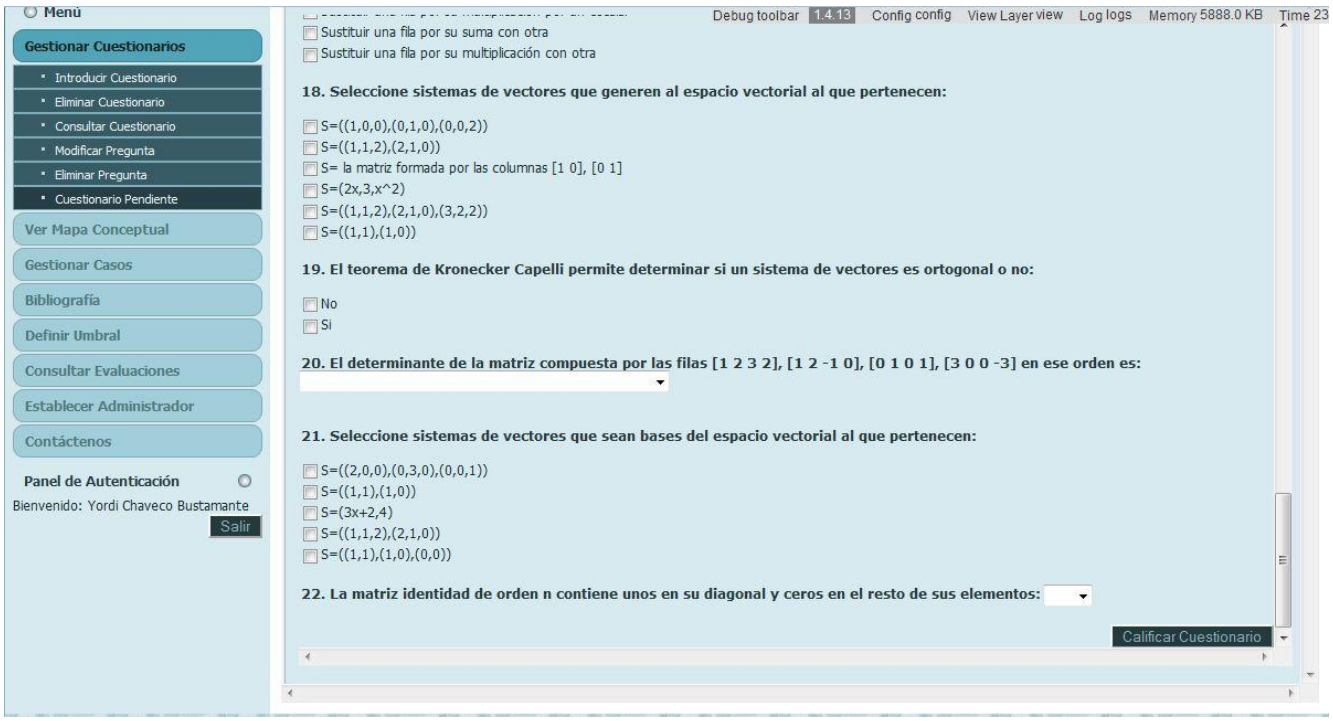
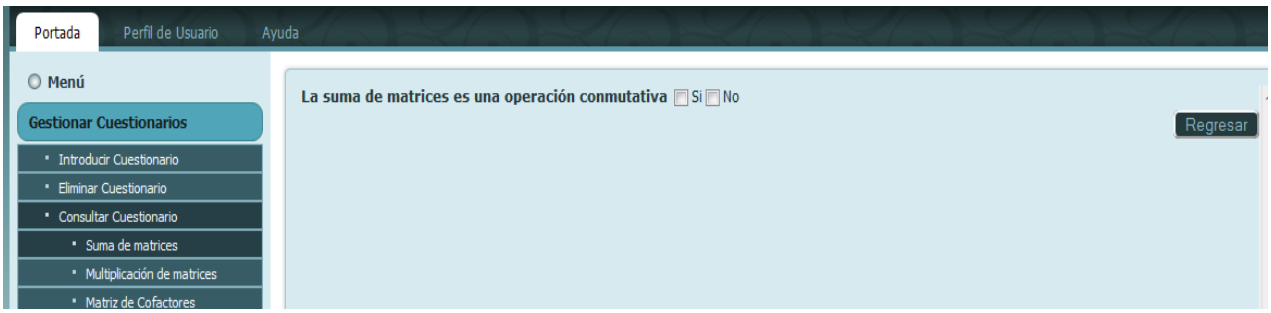


Tabla 3. Descripción textual: CU Gestionar Cuestionario.

Caso de uso:	Gestionar Cuestionario
Actor:	Profesor
Resumen:	El caso de uso se inicia cuando el actor accede a la opción que le permite gestionar cuestionarios.
Precondiciones:	El actor debe estar autenticado en el sistema como profesor.
Postcondiciones:	Se introdujo, consultó o eliminó un cuestionario.
Referencias:	RF-5, RF-6, RF-7
CU Asociados:	
Prioridad:	Crítico
Sección 1: Introducir Cuestionario	
Flujo Normal de los Eventos:	
Acción del Actor:	**Respuesta del sistema**
1 El actor selecciona la opción Introducir Cuestionario.	2 El sistema despliega un menú con los posibles cuestionarios a introducirse.
3 El actor selecciona el tema del Cuestionario.	4 El sistema muestra una interfaz con el campo correspondiente a una pregunta así como el campo correspondiente a una posible respuesta y a la forma de selección a usar.
5 El actor introduce el campo correspondiente a una pregunta así como el campo correspondiente a una posible respuesta. 6 El actor selecciona la forma de selección	8 El sistema crea un nuevo campo para introducir una nueva respuesta.

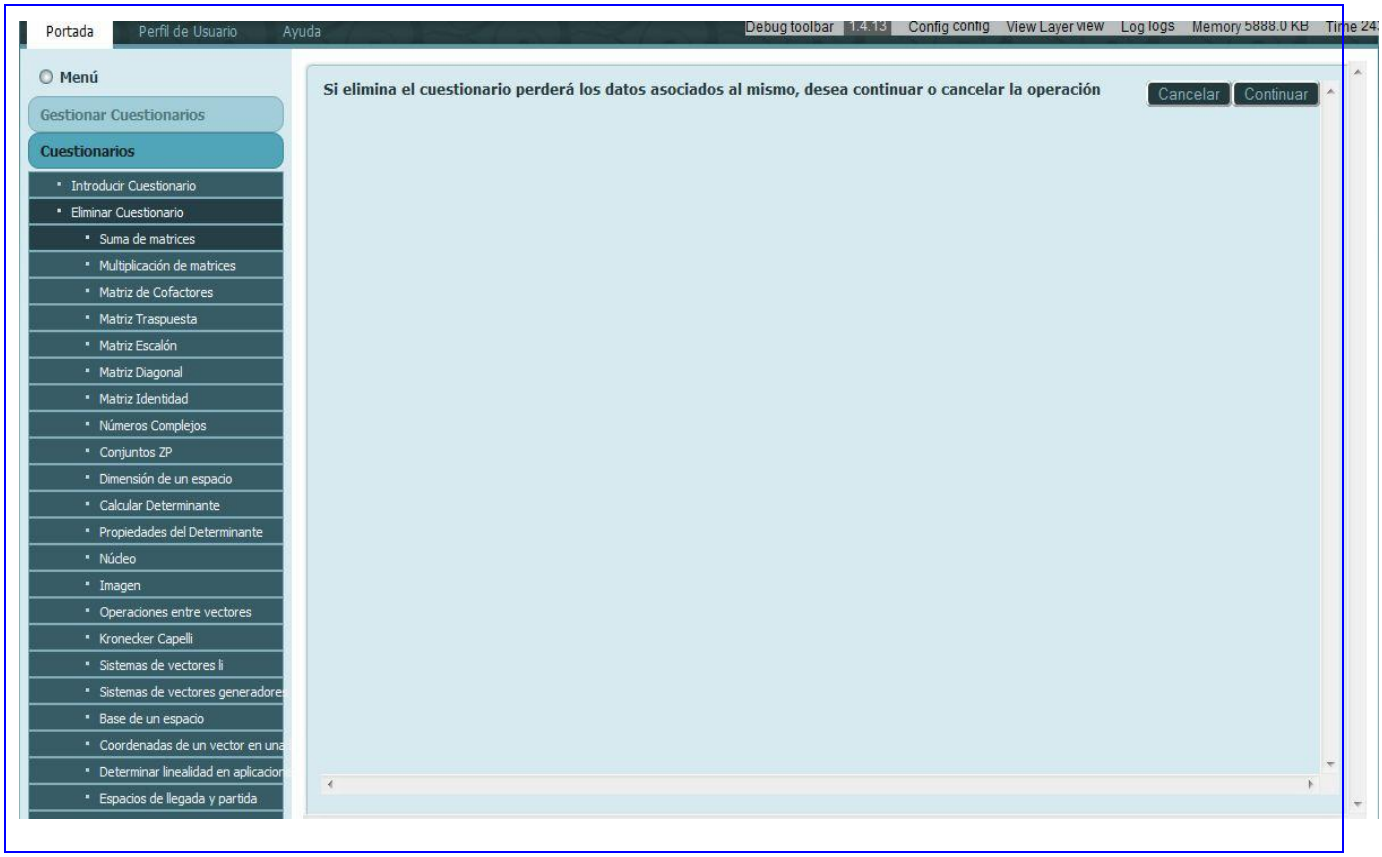
deseada. 7 El actor selecciona la opción Agregar Respuesta.	
9 El actor llena los campos correspondientes a las posibles respuestas y selecciona la opción Guardar Respuestas.	10 El sistema habilita la opción para seleccionar las respuestas correctas y la opción Guardar Pregunta.
11 El actor selecciona las respuestas correctas. 12 El actor presiona el botón Guardar Pregunta.	13 El sistema valida que no existan campos vacíos, guarda la pregunta, muestra el mensaje de información: “Ha sido creado un nuevo cuestionario sobre este tema y regresa al paso 4.
Flujo Alterno:	
12.1 El actor presiona el botón Guardar Pregunta pero el tema del cuestionario a crear ya existe.	13.1 El sistema muestra el mensaje: “La pregunta ha sido insertada correctamente”.
12.2 El actor presiona el botón Guardar Pregunta pero existen campos vacíos.	13.2 El sistema muestra el mensaje: “Existen campos vacíos, por favor, rectifíquelos”.
*.1 El actor presiona el botón Regresar.	*.2 El sistema regresa a la interfaz principal.
Prototipo de Interfaz de Usuario	

Sección 2: Consultar Cuestionario	**Respuesta del sistema**
1 El actor selecciona la opción Consultar Cuestionario.	2 El sistema despliega un menú con los posibles cuestionarios.
3 El actor selecciona el tema del cuestionario.	4 El sistema muestra una interfaz con el cuestionario seleccionado.
Flujo Alterno:	
3.1 El actor selecciona el tema del cuestionario pero no existe el cuestionario que se desea consultar.	4.1 El sistema muestra el mensaje de información: "No existe el cuestionario seleccionado"
Prototipo de Interfaz de Usuario	



Sección 3 Eliminar Cuestionario	**Respuesta del sistema**

1 El actor selecciona la opción Eliminar Cuestionario.	2 El sistema despliega un menú con los temas de los cuestionarios.
3 El actor selecciona el cuestionario.	4 El sistema muestra el mensaje: ¿Si elimina el cuestionario perderá los datos asociados al mismo, desea continuar o cancelar la operación?".
5 El actor selecciona la opción Continuar.	6 El sistema elimina el cuestionario, muestra el mensaje de información: "Cuestionario eliminado correctamente" y regresa a la interfaz principal.
Flujo Alterno:	
5.1 El actor selecciona la opción regresar.	6.1 El sistema regresa a la interfaz principal.
5.2 El cuestionario seleccionado no ha sido creado.	6.2 El sistema muestra el mensaje: "El cuestionario a eliminar no existe".
Prototipo de Interfaz de Usuario	



2.6 Modelo de análisis

La identificación de los requisitos funcionales del sistema, y la agrupación de las funcionalidades por casos de uso permiten obtener una visión más detallada del sistema que especifique qué debe hacer el mismo, esto puede ser representado a través del modelo de análisis.

El modelo de análisis constituye la primera aproximación al modelo de diseño, es el resultado del análisis de los casos de uso y está conformado por las clases del análisis (interfaz, control y entidad), las cuales encapsulan las diferentes funcionalidades que representan los casos de uso. (Jacobson, et al., 2000)

2.6.1 Realización de caso de uso del análisis

Una realización de caso de uso del análisis describe cómo se lleva a cabo y se ejecuta un caso de uso determinado en términos de clases del análisis y de sus objetos del análisis en interacción; proporciona por tanto una traza directa hacia un caso de uso concreto del modelo de casos de uso. (Jacobson, et al., 2000)

La realización de los casos de uso del sistema que se propone incluye diagramas de clases que muestran las clases del análisis y sus relaciones, y diagramas de interacción que explican gráficamente cómo los objetos interactúan a través de mensajes para realizar el flujo de eventos de cada caso de uso, se ha decidido mostrar en este capítulo los diagramas correspondientes a los casos de uso Gestionar Cuestionario y Realizar Cuestionario, el resto pueden ser consultados en la sección de anexos.

2.6.2 Diagramas de clases del análisis

Las figuras 8 y 9 muestran los diagramas de clases del análisis de los casos de uso Gestionar Cuestionario y Realizar Cuestionario respectivamente.

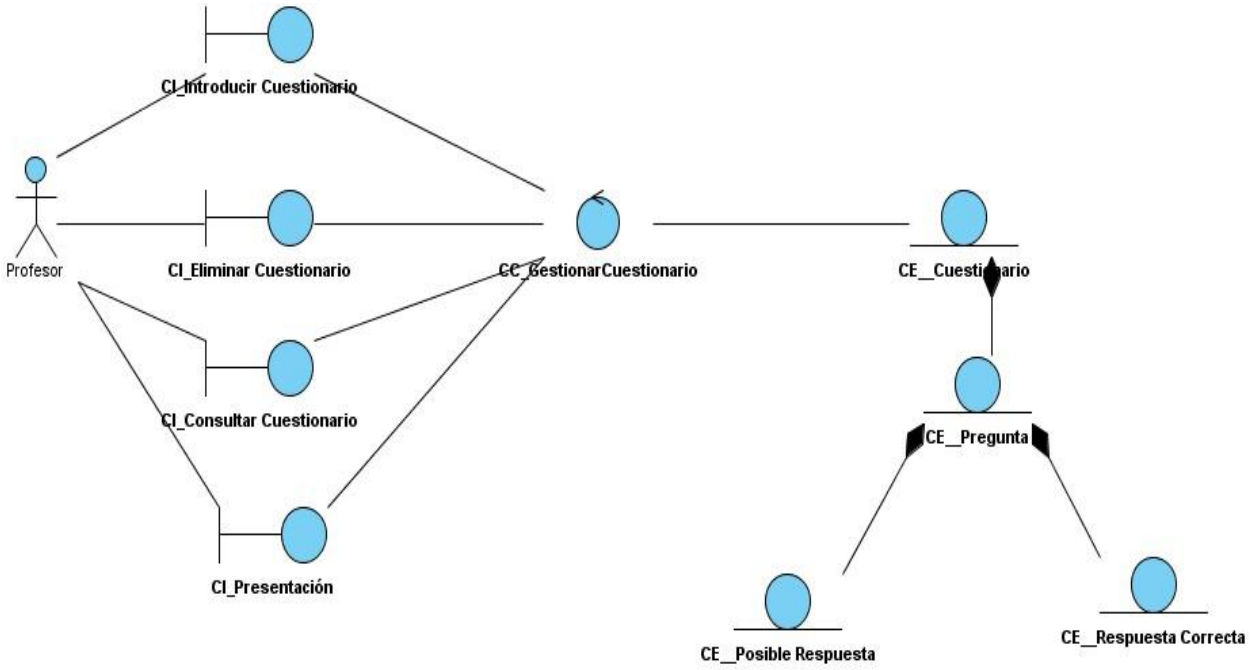


Figura 8. Diagrama de clases del análisis. CU Gestionar Cuestionario.

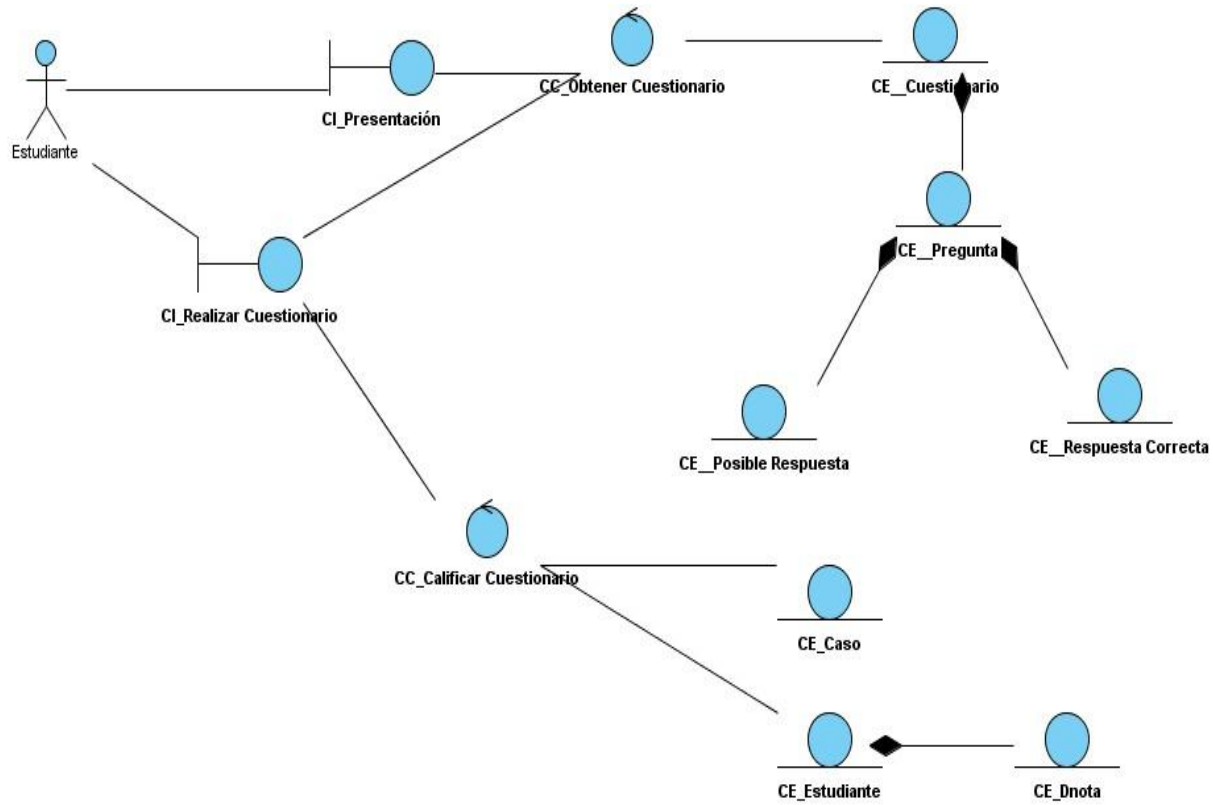


Figura 9. Diagrama de clases del análisis. CU Realizar Cuestionario.

2.6.3 Diagramas de interacción

En el análisis, las interacciones entre objetos se pueden representar a través de diagramas de secuencia o de colaboración. Para la solución que se propone, se emplean diagramas de colaboración, pues el objetivo fundamental es identificar requisitos y responsabilidades sobre los objetos, y no identificar secuencias de interacción detalladas y ordenadas cronológicamente, para lo cual serían más factibles los diagramas de secuencia.

Un diagrama de colaboración es similar a un diagrama de clases del análisis, pero contiene instancias y enlaces en lugar de clases y asociaciones. Muestra cómo interactúan los objetos, enumerando los mensajes que se envían unos a otros. (Jacobson, et al., 2000).

La figura 10 corresponde al diagrama de colaboración del CU Modificar Pregunta, el resto de los diagramas pueden ser consultados en los Anexos.

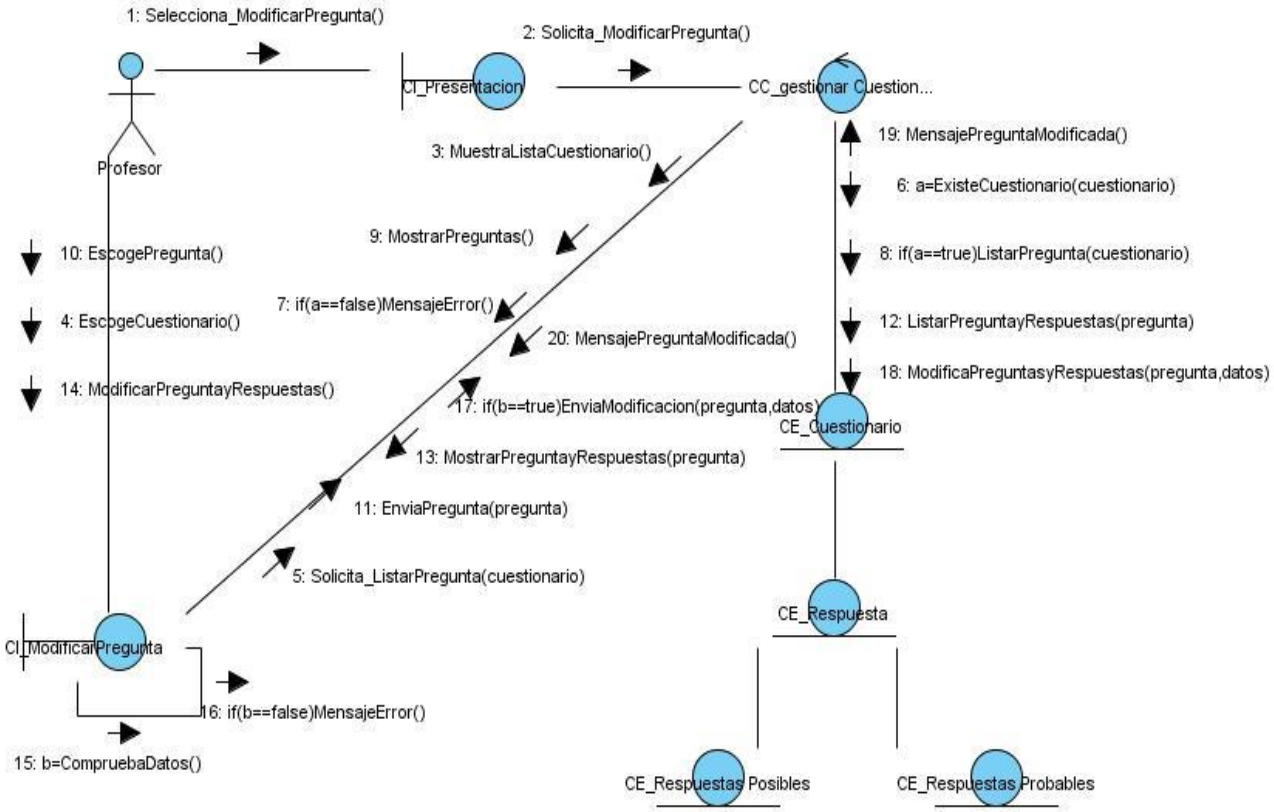


Figura 10. Diagrama de colaboración. CU Modificar Pregunta.

2.7 Descripción de la arquitectura

La IEEE define la arquitectura de software como: organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución. (Reynoso, 2005)

En la actualidad la arquitectura de software se ha convertido en uno de los temas fundamentales en el mundo de la ingeniería de software. En su libro dedicado a la arquitectura de software, Bass y sus colegas la identifican como un elemento de gran importancia durante la construcción del software, de la cual depende en gran medida el éxito del desarrollo. (Bass, et al., 1998).

2.7.1 Principales decisiones arquitectónicas

Durante la realización del capítulo uno se ha hecho referencia a una serie de elementos relacionados con la arquitectura de este software tales como la plataforma de desarrollo, los lenguajes a utilizar, entre otros. A continuación se especifica la principal definición arquitectónica utilizada.

El framework de desarrollo web Symfony está basado en un patrón clásico del diseño web conocido como arquitectura Modelo Vista Controlador (MVC), que está formado por tres niveles:

- ✓ El Modelo: el mismo representa la lógica de negocio de la aplicación, por tanto contiene la información con la que trabaja el sistema.
- ✓ La Vista: es la encargada de transformar el modelo en una página web para permitirle al usuario interactuar con él.
- ✓ El Controlador: es responsable por atender las interacciones del usuario y realizar los cambios necesarios en el modelo o la vista según sea necesario.

En las aplicaciones web es muy frecuente el uso del patrón modelo vista controlador, dicho patrón separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos, el modelo, la vista y el controlador. Esta distribución es muy importante para una aplicación web, ya que permite utilizarla en varios entornos modificando la vista y manteniendo el controlador y el modelo. Un ejemplo clásico de esto son las aplicaciones realizadas para ser visualizadas tanto en ordenadores como en dispositivos móviles.

El modelo tiene como objetivo principal encargarse de la abstracción relacionada con los datos, logrando gran independencia del gestor de base de datos utilizado. Finalmente el controlador tiene como tareas interconectar el modelo con la vista, controlando y especificando la interacción entre los mismos.

2.8 Modelo de diseño

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

2.8.1 Realización de caso de uso del diseño

Una realización de caso de uso del diseño es una colaboración en el modelo de diseño que describe cómo se realiza un caso de uso específico, y cómo se ejecuta, en términos de clases de diseño y sus objetos; proporciona por tanto una traza directa a una realización de caso de uso del análisis en el modelo de análisis. (Jacobson, et al., 2000)

Los diagramas de clases del diseño se realizaron basándose en la implementación que realiza Symfony de la arquitectura MVC, quedando estructurada de la siguiente manera:

- ✓ **Controlador frontal**: Ofrece un punto de entrada único para toda la aplicación, controla todas las peticiones de la misma, carga la configuración y determina la acción a ejecutarse. (Angel Dayán, 2009)
- ✓ **Acciones**: Incluyen el código específico del controlador de cada página y pueden incluir la lógica de la aplicación cuando esta no es muy compleja. Verifican la integridad de las peticiones y preparan los datos requeridos por la capa de presentación. Utilizan el modelo y definen variables para la vista. Las acciones son métodos con el nombre ***executeNombre*** de una clase llamada ***nombreActions*** que hereda de la clase ***sfActions.*** (Angel Dayán, 2009)
- ✓ **Vista**: Se separa en layout y plantilla. (Angel Dayán, 2009)
- ✓ **Layout**: Es global en toda la aplicación o al menos en un grupo de páginas y contiene el código HTML común para estas. (Angel Dayán, 2009)
- ✓ **Plantilla**: Presentan los datos de la acción que se está ejecutando y se encarga de visualizar las variables definidas en el controlador. (Angel Dayán, 2009)
- ✓ **Modelo**: Se divide en la capa de acceso a datos y en la capa de abstracción de la base de datos. De esta forma, las funciones que acceden a los datos no utilizan sentencias ni consultas que dependen de una base de datos, sino que utilizan otras funciones para realizar las consultas. (Angel Dayán, 2009)

La siguiente figura representa el flujo de trabajo del framework Symfony:

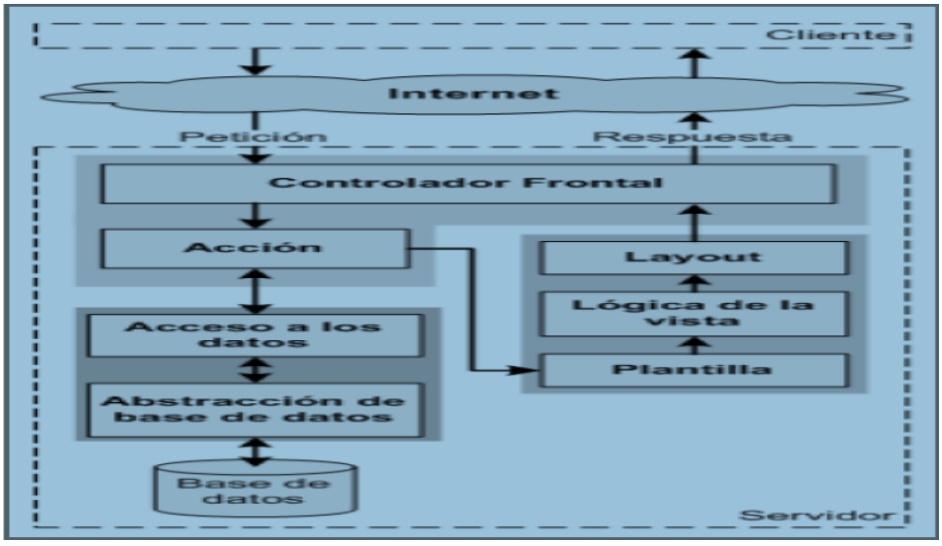


Figura 11. Flujo de trabajo del framework Symfony.

La realización de los casos de uso del diseño que se muestra a continuación contiene diagramas que muestran las clases del diseño participantes y sus relaciones, las figuras 12 y 13 muestran los diagramas de clases del diseño de los CU Gestionar Cuestionario y Realizar Cuestionario respectivamente, el resto pueden ser consultados en la sección de Anexos.

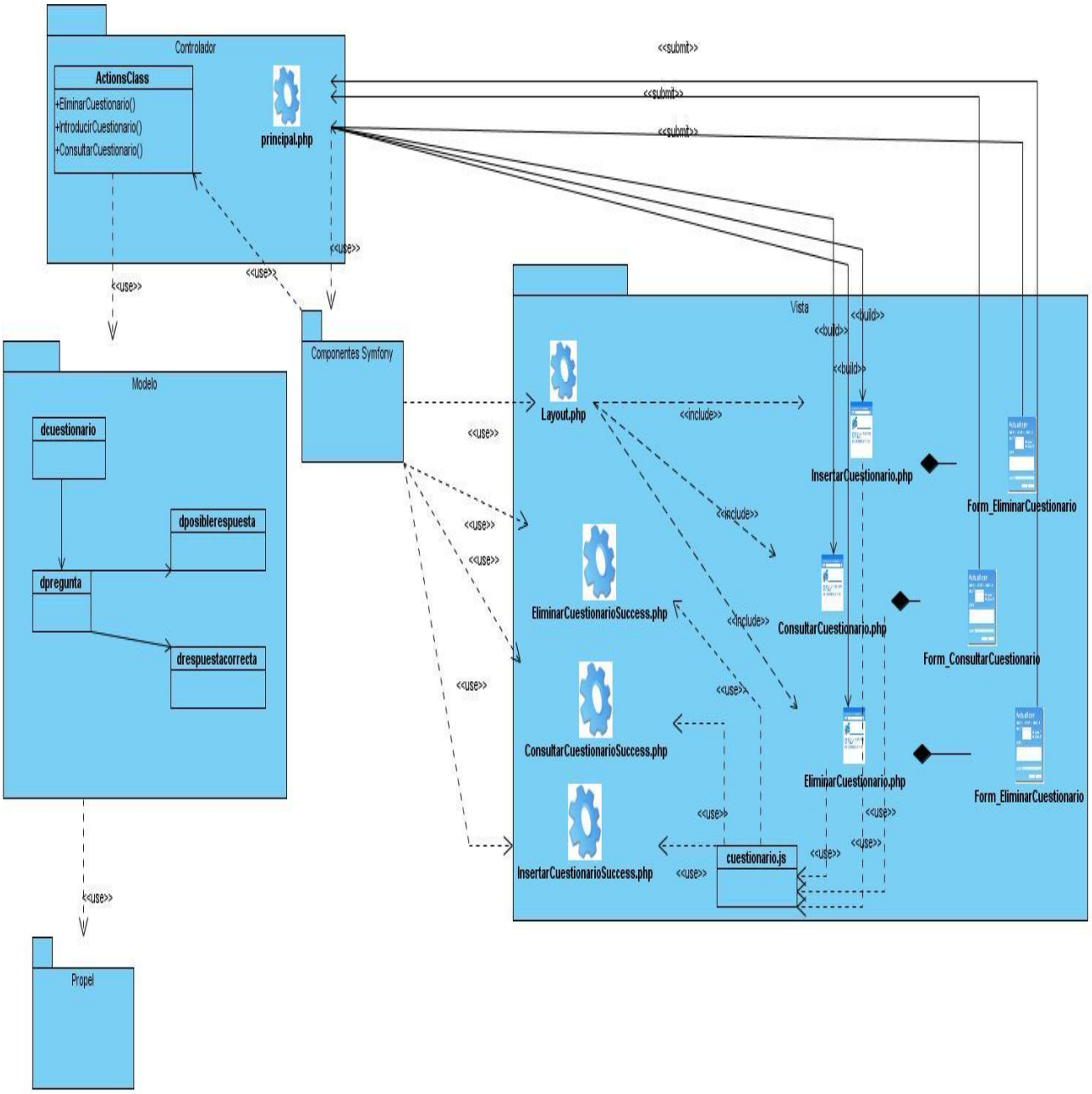


Figura 12. Diagrama de clases del diseño. CU Gestionar Cuestionario.

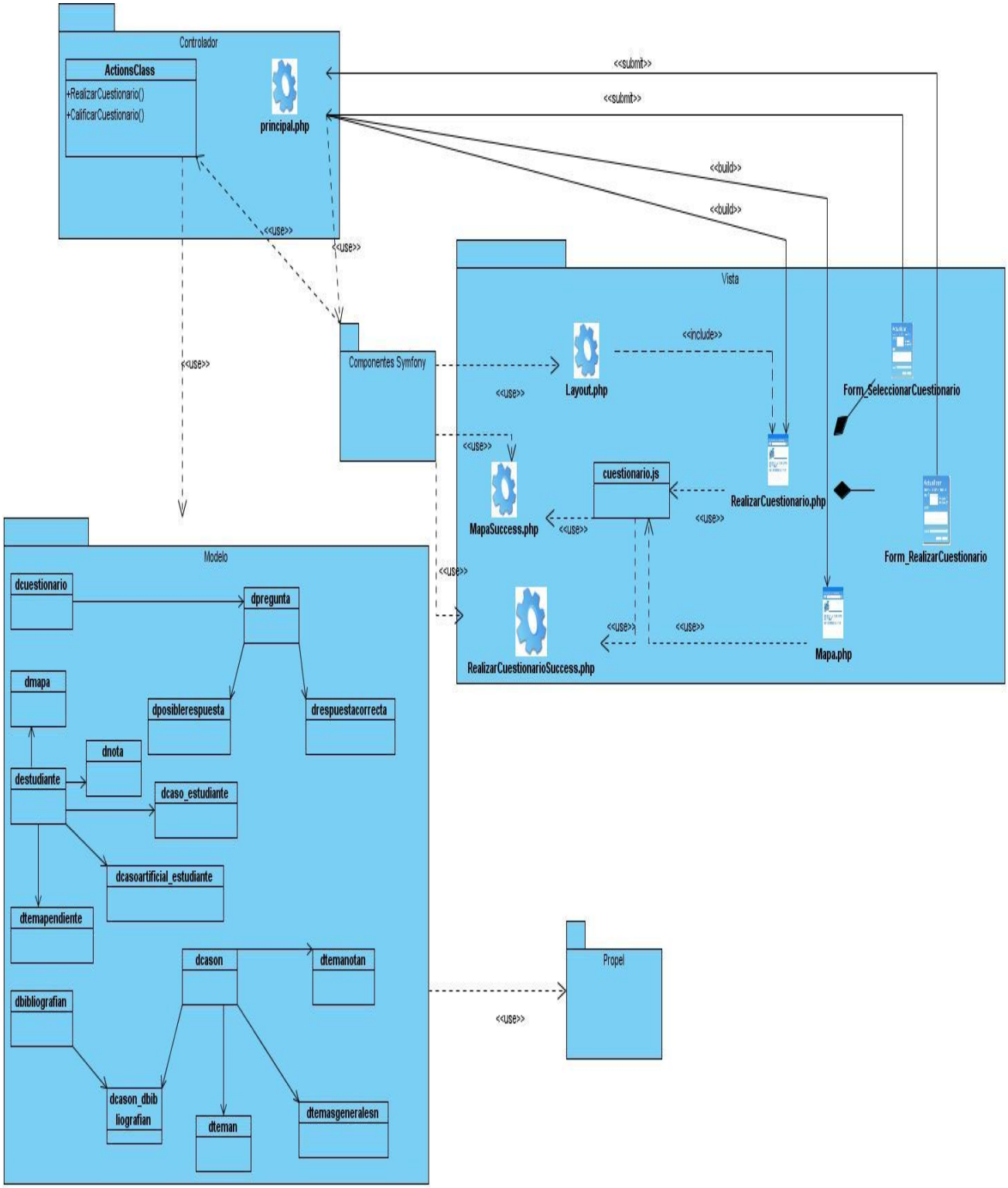


Figura 13. Diagrama de clases del diseño. CU Realizar Cuestionario.

Los diagramas de clases del diseño representados anteriormente contienen un paquete denominado Componentes Symfony, a continuación se hace representa con mayor nivel de detalles al contenido y funcionamiento del mismo.

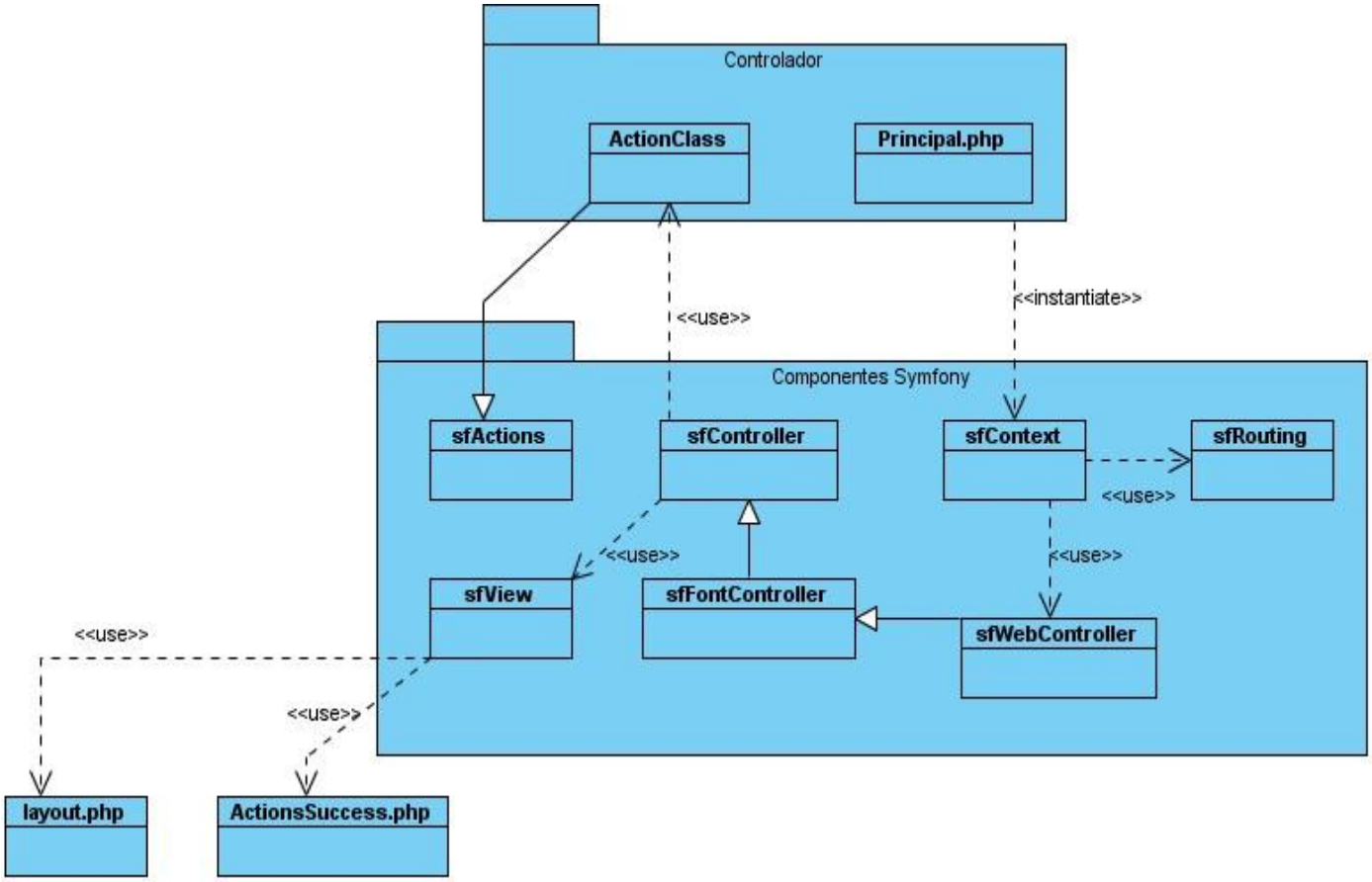


Figura 14. Estructura detallada del componente Symfony.

2.9 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.

Al aplicar un patrón, el código resultante no tiene por qué delatar el patrón o patrones que lo inspiró. No obstante, últimamente hay múltiples esfuerzos enfocados a la construcción de herramientas de desarrollo basados en los patrones y frecuentemente se incluye en los nombres de las clases el nombre del patrón en que se basan, facilitando así la comunicación entre desarrolladores.

Los patrones de diseño expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software. Estos describen un problema que ocurre constantemente en

el entorno y la solución básica de manera genérica para que dicha solución pueda ser utilizada en reiteradas ocasiones, evitando hacer lo mismo varias veces. Específicamente estos tipos de patrones constituyen una descripción de clases y objetos comunicándose entre sí, adaptados para resolver un problema de diseño general en un contexto particular.

Ventajas del uso de patrones:

- ✓ Proporcionan una estructura conocida por todos los programadores, de manera que la forma de trabajar no resulte distinta entre los mismos.
- ✓ Permiten tener una estructura de código común a todos los proyectos que implementen una funcionalidad genérica.
- ✓ Permiten ahorrar grandes cantidades de tiempo en la construcción del software.
- ✓ El software construido es más fácil de comprender, mantener y extender.
- ✓ Ofrecen una mejor imagen de profesionalidad y calidad.
- ✓ En la actualidad se ha extendido el uso de los patrones de diseño más allá del propio diseño del sistema, teniéndolos en cuenta desde el inicio de la concepción del software hasta la implementación.

Los Patrones Generales de Software para Asignar Responsabilidades, más conocidos como GRASP por sus siglas en inglés, representan los principios básicos para la asignación de responsabilidades a objetos expresados en forma de patrones. Aunque son considerados más bien como una serie de buenas prácticas de aplicación recomendable en el diseño de software. A continuación se hace referencia a los patrones GRASP utilizados en el software.

Experto: Se encarga de asignar una responsabilidad al experto en información, o sea, aquella clase que cuenta con la información necesaria para cumplir la responsabilidad.

Un modelo de clase puede definir docenas y hasta cientos de clases de software, y una aplicación tal vez requiera el cumplimiento de cientos o miles de responsabilidades. Si estas se asignan en forma adecuada, los sistemas tienden a ser más fáciles de entender, mantener y ampliar, y puede presentar la oportunidad de reutilizar los componentes en futuras aplicaciones.

El framework Symfony utiliza este patrón puesto que Propel es la librería que este usa para realizar su capa de abstracción en el modelo, encapsula toda la lógica de los datos y son generadas las clases con todas las funcionalidades comunes de las entidades.

Creador: Guía la asignación de responsabilidades relacionadas con la creación de objetos. La creación de objetos es una de las actividades más frecuentes en un sistema orientado a objetos. En consecuencia, conviene contar con un principio general para asignar las responsabilidades concernientes a ella. El diseño, bien asignado, puede soportar un bajo acoplamiento, una mayor claridad, el encapsulamiento y la reutilización. El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento.

En el framework usado la clase Actions define acciones dentro de las cuales se crean objetos de las clases PEER, que son las clases representativas de las entidades; evidenciando de este modo que la clase Actions es "creador" de dichas entidades.

Alta Cohesión: La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión garantiza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Un conjunto de clases con baja cohesión trae como consecuencia una serie de desventajas como por ejemplo:

- ✓ Difíciles de comprender.
- ✓ Difíciles de reutilizar.
- ✓ Difíciles de conservar.
- ✓ Las afectan constantemente los cambios.

Symfony permite asignar responsabilidades con una alta cohesión, por ejemplo la clase Actions tiene la responsabilidad de definir las acciones para las plantillas y colabora con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades, es decir, está formada por diferentes funcionalidades que se encuentran estrechamente relacionadas proporcionando que el software sea flexible frente a grandes cambios.

Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. El bajo acoplamiento existe cuando una clase no depende mucho de otras clases. Por eso este patrón se encarga de asignar una

responsabilidad de modo que el acoplamiento siga siendo bajo, permitiendo que las clases no puedan ser afectadas por ningún otro componente, se puedan entender mejor y sean fáciles de reutilizar.

Controlador: Un Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación.

Un evento del sistema es un evento de alto nivel generado por un actor externo; es un evento de entrada externa. Se asocia a operaciones del sistema: las que emite en respuesta a los eventos del sistema.

En el sistema a crear, todas las peticiones web son atendidas por un solo controlador frontal, denominado Principal.php, este el punto de entrada único de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento ubicado en el archivo apps/Principal/config/Routing.yml, para asociar el nombre de una acción y el nombre de un módulo con la URL entrada por el usuario.

2.9.1 Patrones GOF

Patrones Creacionales

Singleton (Instancia única): El patrón Singleton es uno de los más sencillos patrones de diseño, y es útil para limitar el máximo número de instancias de una clase en exactamente solo una. En este caso, si más de un objeto necesita utilizar una instancia de la clase Singleton, esos objetos comparten la misma instancia. En un uso más avanzado, este patrón puede ser utilizado también para administrar n instancias de una clase. (Pedro García, 2005)

El patrón Singleton también es muy útil cuando existen datos en una aplicación que van a ser compartidos por muchas clases y no se quiere cargar en memoria todas las veces que se va a utilizar porque ocupa mucho tiempo. (Pedro García, 2005)

Abstract Factory (Fábrica abstracta): Permite trabajar con objetos de distintas familias de manera que las familias no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando. Cuando el framework necesita por ejemplo crear un nuevo objeto para una petición, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea. (Angelfire, 2011).

2.9.2 Patrones Estructurales

Decorator (Envoltorio): Añade funcionalidad a una clase dinámicamente. El archivo layout.php, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el layout, o si se mira desde otro punto de vista, el layout decora la plantilla. (Angelfire, 2011).

Composite (Objeto compuesto): Permite tratar objetos compuestos como si se tratase de uno simple. Sirve para construir objetos complejos a partir de otros más simples y similares entre sí, gracias a la composición recursiva y a una estructura en forma de árbol .Esto simplifica el tratamiento de los objetos creados, ya que al poseer todos ellos una interfaz común, se tratan todos de la misma manera. (Angelfire, 2011).

2.10 Modelo de datos del sistema

El modelo de datos de un proyecto Symfony es generado automáticamente por el framework y está constituido por cuatro clases por cada tabla de la base de datos, estas son:

- ✓ **baseNombreTablaPeer**: Son clases que tienen métodos estáticos para trabajar con las tablas de la base de datos. Proporcionan los medios necesarios para obtener los registros de las tablas y sus métodos devuelven normalmente un objeto o una colección de objetos de la clase objeto relacionada. Representan la parte del modelo que se encarga del acceso a los datos.
- ✓ **baseNombreTabla**: Son clases objeto que representan un registro de la base de datos. Permiten acceder a las columnas de un registro y a los registros relacionados. Representan la parte del modelo que se encarga de la abstracción de los datos.
- ✓ **nombreTabla**, **nombreTablaPeer**: Sirven para extender las funcionalidades de sus clases bases correspondientes.

La figura que se muestra a continuación es una representación genérica del modelo generado por el framework Symfony.

Figura 15. Representación genérica del modelo.

Un modelo de datos es un lenguaje orientado a describir una base de datos. Típicamente un modelo de datos permite describir:

- ✓ Las estructuras de datos de la base, el tipo de los datos que hay en la base y la forma en que se relacionan.
- ✓ Las restricciones de integridad: Un conjunto de condiciones que deben cumplir los datos para reflejar correctamente la realidad deseada.
- ✓ Operaciones de manipulación de los datos: típicamente, operaciones de agregado, borrado, modificación y recuperación de los datos de la base.

Otro enfoque es pensar que un modelo de datos permite describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan esos elementos entre sí. No hay que perder de vista que una base de datos siempre está orientada a resolver un problema determinado, por lo que los dos enfoques propuestos son necesarios en cualquier desarrollo de software. A continuación se representa el modelo de datos del sistema.

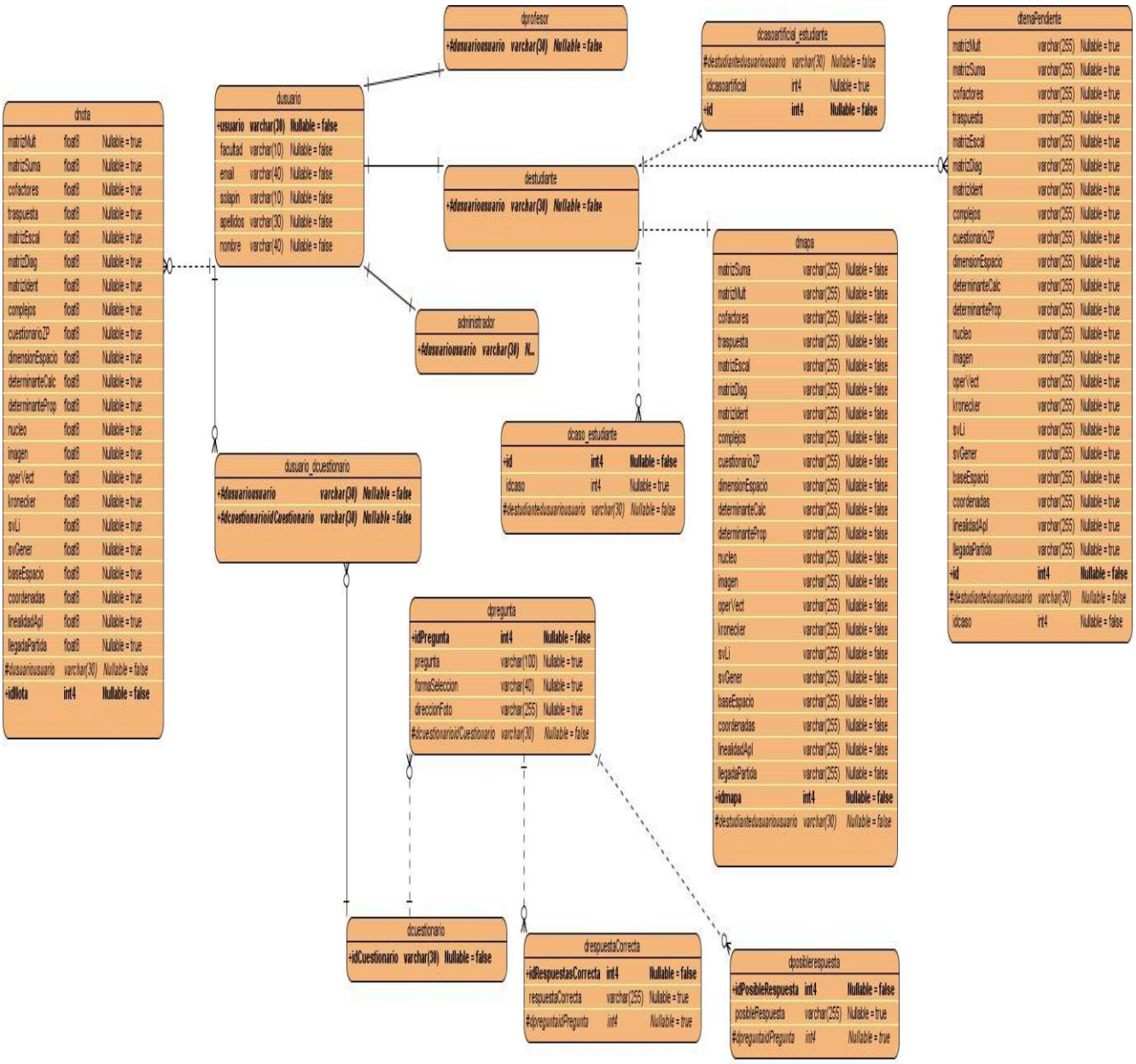


Figura 16. Modelo de datos del sistema.

La lógica de negocio de las aplicaciones web depende casi siempre de su modelo de datos. El componente que se encarga por defecto de gestionar el modelo en Symfony es una capa de tipo ORM (mapeo de objetos a bases de datos), que se ocupa de mapear la base de datos utilizada en clases. En las aplicaciones Symfony, el acceso y la modificación de los datos almacenados en la base de datos se realiza mediante objetos; de esta forma nunca se accede de forma explícita a la base de datos. Este comportamiento permite un alto nivel de abstracción y permite una fácil portabilidad. (Potencier, et al., 2008).

La principal ventaja que aporta el ORM es la reutilización, ya que brinda la posibilidad de llamar a los métodos de un objeto de datos desde varias partes de la aplicación y desde diferentes aplicaciones, ya que la distribución de archivos en Symfony lo permite. Esta capa se encarga de encapsular la lógica de los datos, en la versión utilizada se usa Propel como ORM, esta genera 4 clases por cada tabla de la base de datos para lograr utilizar objetos en vez de registros y clases en vez de tablas. Las clases de Propel se encargan de contener el código relacionado con el acceso a datos y la lógica de los mismos, así como su mapeo.

En la implementación del diseño propuesto se pretende agrupar la mayor parte de la lógica en la capa del modelo, aumentando considerablemente la reutilización de sus clases. También se pretende lograr que cada clase se encargue de realizar las funcionalidades de la tabla que le corresponde en la base de datos para lograr un bajo acoplamiento. Logrando con esto que las acciones se dediquen mayormente a enlazar al modelo con la vista, trayendo como consecuencia una alta cohesión.

2.11 Conclusiones

Durante la realización del presente capítulo se han abordado los aspectos fundamentales referentes al análisis y al diseño del sistema, han quedado representadas las realizaciones de casos de uso, tanto del análisis como del diseño, mediante el modelo de análisis y de diseño respectivamente. Se ha definido qué tipo de arquitectura se aplicará en la elaboración de la solución y ha quedado plasmado el modelo de datos que se usará para el almacenamiento de la información en la base de datos, además se detalló el trabajo realizado con los casos dejando claro el algoritmo seguido para implementar un Sistema Basado en Casos.

Capítulo 3. Implementación y Pruebas

3.1 Introducción

Los artefactos UML creados durante el análisis y diseño, como son: los diagramas de interacción y diagramas de clases del diseño del sistema, constituyen la entrada fundamental para las actividades de implementación y prueba. En la primera, se describe cómo los elementos del modelo de diseño se implementan en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares. Una vez realizadas las actividades de implementación, se requiere realizar pruebas para detectar la presencia de errores.

3.2 Modelo de implementación

El modelo de implementación describe de qué forma los elementos del modelo de diseño, como las clases, se implementan en términos de componentes y subsistemas de implementación. Describe además, la forma en que se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados. El modelo de implementación es la entrada principal de las etapas de prueba que siguen a la implementación. (Jacobson, y otros, 2000)

3.3 Diagrama de componentes

El diagrama de componentes se muestra como un grafo de componentes de software. Estos pueden ser de código fuente, de bibliotecas, entre otros, y unidos además por medio de relaciones de dependencia, como son compilación y ejecución, permitiendo que se muestren las interfaces que soportan. A continuación se muestra el diagrama de componentes de la aplicación:

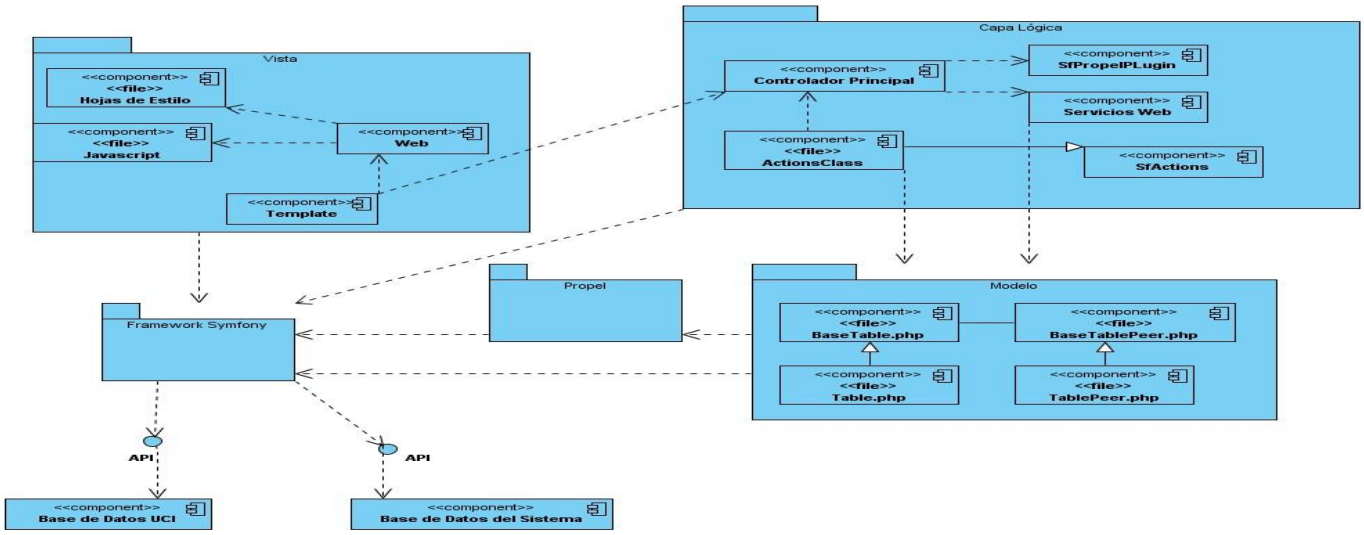


Figura 17. Diagrama de componentes del sistema.

3.4 Diagrama de despliegue

"Un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación. Un nodo es un objeto físico en tiempo de ejecución que representa un recurso computacional, generalmente con memoria y capacidad de procesamiento" (Jorge Andrés).

En el diagrama de despliegue se visualiza la relación física de los componentes de la aplicación, permite comprender la correspondencia entre la arquitectura software y la arquitectura hardware.

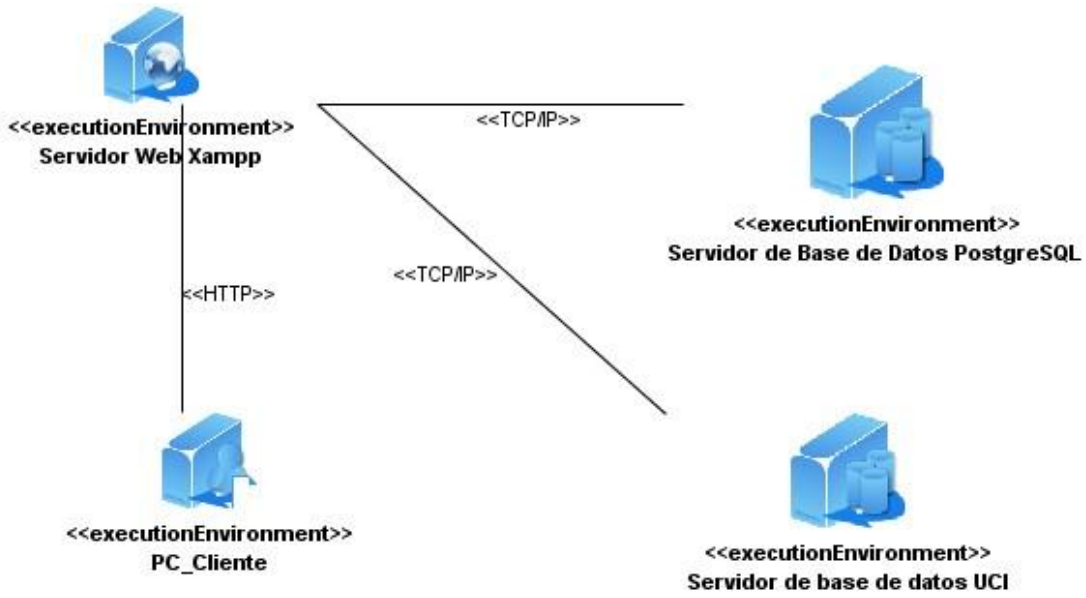


Figura 18. Diagrama de despliegue del sistema.

Para un mejor entendimiento del diagrama de despliegue a continuación se hace referencia a los principales elementos del mismo.

- ✓ PC_Cliente: Desde ella se podrán realizar todas las operaciones.
- ✓ Servidor Web Xampp: En él se encuentra instalado la aplicación y se realizan las operaciones del lado del servidor.
- ✓ Servidor de Base de Datos PostgreSQL: En él se encuentra instalada la Base de datos.
- ✓ Servidor de Base de Datos UCI: Servidor al cual se accede para obtener los datos de los usuarios en el dominio de la Universidad.
- ✓ <<TCP/IP>>: Conexión para intercambio de datos con las bases de datos PostgreSQL y UCI.
- ✓ <<HTTP>>: Conexión para intercambio de datos con el servidor.

3.5 Pruebas

3.5.1 Pruebas de validación del sistema

Una de las características típicas del desarrollo de software basado en el ciclo de vida es la realización de controles periódicos, normalmente coincidiendo con los hitos del proyecto o la terminación de documentos. Estos controles pretenden una evaluación de la calidad de los productos generados (especificación de requisitos, documentos de diseño) para poder detectar posibles defectos cuanto antes. Sin embargo, todo sistema o aplicación, independientemente de estas revisiones, debe ser probado mediante su ejecución controlada antes de ser entregado al cliente. Estas ejecuciones o ensayos de funcionamiento, posteriores a la terminación del código del software, se denominan habitualmente pruebas. (Pressman, 2005).

Las pruebas constituyen una de las principales etapas de un software, es mediante las pruebas que se puede determinar que el sistema responde correctamente a los requerimientos descritos durante la fase de inicio. De forma más específica los propósitos de las pruebas son:

- ✓ Realizar la validación del software desarrollado, entendiendo como validación el proceso que determina si el software satisface los requisitos.
- ✓ Realizar la verificación del software desarrollado, entendiendo como verificación el proceso que determina si los productos de una fase satisfacen las condiciones de la misma.

Es importante considerar que las pruebas de software no garantizan que un sistema esté libre de errores, sino que se detecten la mayor cantidad de defectos posibles para su debida corrección. (UCID. Proceso de Desarrollo y Gestión de Proyectos).

Hoy en día existen dos enfoques de pruebas muy extendidos y ampliamente practicados debido a su probada efectividad, los mismos son:

- ✓ Enfoque Estructural o Prueba de Caja Blanca.
- ✓ Enfoque Funcional o Prueba de Caja Negra.

3.5.2 Descripción de los métodos empleados para la realización de las pruebas

En este software se optó por el uso de las pruebas de caja negra, las mismas parten de la premisa de que lo más importante en un software es que se cumplan los requerimientos planteados, de esta forma provee un conjunto de datos de entrada por cada flujo de la aplicación e intenta descubrir los fallos del programa informático. Una ventaja muy importante de este método de prueba sobre las pruebas de caja blanca es que cualquier persona es capaz de probar un software, incluso cuando no es experta en el campo informático, no siendo así con las pruebas de caja blanca, ya que las mismas se realizan directamente al código.

La prueba de caja negra pretende encontrar errores de las siguientes categorías:

- ✓ Funciones incorrectas o ausentes.
- ✓ Errores de interfaz.
- ✓ Errores en estructuras de datos o en accesos a bases de datos externas.
- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y de terminación.

3.5.3 Diseño de casos de pruebas

Un caso de prueba es un conjunto de condiciones o variables bajo las cuáles se determina si un requisito es parcial o completamente satisfactorio. Su principal objetivo es comprobar que el sistema responde de la forma prevista ante la ejecución de determinados eventos del usuario. De esta forma si existe un error funcional el caso de prueba será capaz de detectarlo para su corrección.

Para la realización de los casos de prueba de este sistema se decidió crear un caso de prueba por cada caso de uso, garantizando que de esta forma queden probados todos los flujos del software.

En la etapa de prueba del software es importante la creación de una serie de casos de pruebas, los cuales permitirán a los probadores comprobar el correcto funcionamiento de cada funcionalidad de la aplicación.

Al generar casos de prueba, se deben incluir tanto datos de entrada válidos y esperados como no válidos e inesperados. Las pruebas deben centrarse en dos objetivos: Probar si el software no hace

lo que debe hacer y probar si el software hace lo que no debe hacer, es decir, si provoca efectos secundarios adversos.

Todos los casos generados en el sistema pueden ser consultados en la sección de Anexos.

Tabla 4. No conformidades detectadas.

No Conformidades	**Acciones Correctivas**
El sistema muestra una excepción al autenticar un usuario con más de 10 caracteres en sus apellidos.	Resuelta de manera inmediata después de finalizar la iteración de pruebas.
El sistema no responde al ser ejecutada la opción consultar mapa conceptual por el menú principal.	Resuelta de forma inmediata después de su detección.
Existen respuestas guardándose en la base de datos de forma indefinida.	Resuelta de forma inmediata después de su detección.
El sistema muestra un menú en la interfaz principal con la palabra personalizado mal escrita.	Resuelta inmediatamente después de detectado dicho error.
El umbral admite números mayores que 100.	Resuelta inmediatamente después de detectado dicho error.

3.6 Conclusiones

Durante la realización del presente capítulo fue desarrollado el diagrama de componentes, el cual contiene los componentes que fueron utilizados para el desarrollo del sistema siempre teniéndose en cuenta la arquitectura definida, además se desarrolló el diagrama de despliegue el cual visualiza la relación física de los componentes de la aplicación. También se le realizaron pruebas de caja negra al software, ya que la calidad del mismo está muy relacionada con las pruebas que se le realicen, de ahí la importancia que se le atribuye a todo el proceso de pruebas. De manera general se implementó el Sistema Web Inteligente apoyado en Mapas Conceptuales para la asignatura de Álgebra Lineal cumpliéndose de esta forma el objetivo general del presente trabajo de diploma.

Conclusiones generales

En este trabajo de diploma se realizó un análisis del marco teórico de la investigación, posibilitando el sustento científico del mismo. De igual forma se detallaron las herramientas, tecnologías y las metodologías utilizadas para desarrollar la implementación del sistema. La especificación de los requisitos funcionales, en conjunto con el análisis del diseño de clases y el modelo de datos del software posibilitaron la definición de las funcionalidades de la aplicación informática para cumplir los objetivos planteados. Se definió un algoritmo computacional donde se aplica el Razonamiento Basado en Casos; además se implementó el Sistema Web Inteligente apoyado en MC para la asignatura de Álgebra Lineal validado correctamente a través de pruebas de software de Caja Negra.

Bibliografía

- ✓ **Belloch, Consuelo, 2011**. *Entornos Virtuales de Aprendizaje.*
- ✓ **López García, Pablo, 2008**. *Moodle: Difusión y funcionalidades.*
- ✓ **Suárez Guerrero, Cristóbal, 2004**. *Los entornos virtuales de aprendizaje como instrumento de mediación.*
- ✓ **Lozano, Laura y Fernández, Javier.** *Razonamiento Basado en Casos: Una Visión General.*
- ✓ **Soria Francis, Sindy. 2010.** *Modelo para diseñar Mapas Conceptuales Inteligentes utilizando el Razonamiento Basado en Casos.* Ciudad Habana : s.n., 2010. Trabajo de diploma para optar por el título de ingeniero en Ciencias Informáticas.
- ✓ **Duque Tamayo, Edward Alexander. 2006.** *Diseño del prototipo de un sistema multiagente tutor virtual.* 2006.
- ✓ **Hernández Orozco, Yaneidis. 2011.** *Análisis y Diseño de una herramienta para elaborar Mapas Conceptuales Inteligentes.* Ciudad Habana : s.n., 2011. Trabajo de diploma para optar por el título de ingeniero en ciencias informáticas.
- ✓ **León Espinosa, Maikel y García Valdivia, Zenaida.** *La Inteligencia Artificial en la Informática Educativa.*
- ✓ **Martínez Sánchez, Natalia, y otros.** *El paradigma del razonamiento basado en casos en el ámbito de los sistemas tutoriales inteligentes.*
- ✓ **Moreno, L, y otros.** *Hacia un Sistema Inteligente basado en Mapas Conceptuales Evolucionados para la Automatización de un Aprendizaje Significativo. Aplicación a la Enseñanza Universitaria de la Jerarquía de Memoria.*
- ✓ **Msc Jiménez Castro, Maynor y Msc Morales Garay, Ismael.** *Propuesta educativa en la enseñanza de la matemática en Costa Rica a través de sistemas tutores inteligentes.*
- ✓ **Rosado, Luis y Ramón Herreros, Juan.** *Inteligencia artificial y educación: Mapas Conceptuales tradicionales e hipermediales en la enseñanza de la física.*
- ✓ **Soria Francis, Sindy. 2010.** *Modelo para diseñar Mapas Conceptuales Inteligentes utilizando el Razonamiento Basado en Casos.* Ciudad Habana : s.n., 2010. Trabajo de diploma para optar por el título de ingeniero en ciencias informáticas.

- ✓ **Introduction to OMG's Unified Modeling Language™ (UML®).** (*19 de Enero de 2012). Recuperado el 1 de Febrero de 2012, de Object Management Group, Inc.: http://www.omg.org/gettingstarted/what_is_uml.htm.*
- ✓ **Martínez, R.** *(2 de Octubre de 2010). PostgreSQL-es. Recuperado el 20 de Enero de 2012, de Portal en español sobre PostreSQL: http://www.postgresql.org.es/sobre_postgresql.*
- ✓ **pgAdmin.(s.f.).**Recuperado el 23 de Enero de 2012, de PostgreSQL Tools: *http://www.pgadmin.org/*.
- ✓ **NetBeans. (2012).** *Recuperado el 18 de Enero de 2012, de NetBeans IDE - The Smarter Way to Code: http://netbeans.org/features/index.html.*
- ✓ **Doctrine ORM for PHP***. (25 de Febrero de 2009). Recuperado el 17 de Enero de 2012, de Guide to Doctrine for PHP: http://phpdoc.org.*
- ✓ **Entorno Virtual de Aprendizaje.** *(n.d.). Recuperado el 2 de Febrero de 2012, de Introducción a* **JavaScript***: http://eva.uci.cu/mod/resource/view.php?id=4563*
- ✓ *Entorno Virtual de Aprendizaje. (n.d.). Recuperado el 2 de Febrero de 2012, de Introducción a* **XHTML***: http://eva.uci.cu/mod/resource/view.php?id=4563.*
- ✓ **Zaninotto, François y Potencier, Fabien.** *Las clases del modelo. [trad.] Miguel Sánchez. Symfony 1.2, la guía definitiva. s.l. : Apress, 19, pág. 427.*
- ✓ **Cabrera, Angela , y otros.** *Los Mapas Conceptuales: una poderosa herramienta para el aprendizaje significativo. Ojeda 2007. Habana : CITMA, 2007, Vol. 15. ISSN:1530-2880.*

Referencias bibliográficas

Referencias Bibliográficas

Andrés, Jorge. *Trabajo de grado como requisito parcial para optar al título de licenciado en informática. Escuela de Ciencias, Departamento de Matemáticas.*

Arruate. *Urretavizcaya M: Herramientas de Autor para Enseñanza y Diagnóstico: Iris-D. Revista Iberoamericana de Inteligencia Artificial. No.12 2001, pp. 13-28. Revista.*

Bass, et al. 1998. *Architecture: Description Really Matters. http://msdn.microsoft.com/en-us/library/bb421528.aspx. 1998.*

Ferrero, 2001. *Urretavizcaya M: Herramientas de Autor para Enseñanza y Diagnóstico: Iris-D. Iberoamericana de Inteligencia Artificial. No.12, 2001, pp. 13-28.*

García, Pedro. 2005. *Cerebro en la sombra. http://blog.osusnet.com/2009/05/. 2005.*

García, Rodríguez and. *http://www.doredin.mec.es/documentos/01220103010364.pdf.*

J, Kolodner. *Case-Based Reasoning. San Mateo. CA,Morgen Kaufmann. 1993.*

Jacobson Ivar, Rumbaugh James, Booch Grady. 2000. *El Proceso Unificado de Desarrollo de Software Edición 2000.*

Jacobson, et al, 2000. *El proceso unificado de desarrollo de software. 2000.*

Jacobson, y otros, 2000. *Linea Base de la Arquitectura. http://www.fi.unju.edu.ar. 2000.*

JohnMcCarthy. *http://portal.educ.ar/noticias/ciencia-y-tecnologia/murio-john-.*

otros, Potencier. *Symfony la guía definitiva. http://www.librosweb.es/symfony.*

Angelfire. 2011. *Patrones GoF. Geek the Planet. [En línea] 11 de Mayo de 2011. [Citado el: 05 de Mayo de 2012.] http://geektheplanet.net/5462/patrones-gof.xhtml.*

Schermbeek, Van. 2008. *http://www.elmundolinux.com/apachelinux.php. 2008.*

Schiaffino, Silvia. 2012. *Inteligencia Artificial Razonamiento Basado en Casos. Buenos Aires : ISISTAN - CONICET, 2012.*

Glosario de Términos

Estudiante: Persona que posee privilegios para realizar cuestionarios, ver su perfil de usuario, así como consultar su mapa conceptual.

Profesor: Persona que posee privilegios para confeccionar y ver cuestionarios, eliminar preguntas, y consultar su perfil de usuario.

Administrador: Persona que posee privilegios para eliminar los perfiles de usuario, ya sean profesores o estudiantes.

Sistema Inteligente: Se refiere al sistema al cual acceden los usuarios, el cual permite realizar las acciones correspondientes con los cuestionarios, genera el mapa conceptual inteligente utilizando técnicas de inteligencia artificial y registra las acciones de los usuarios en su perfil.

Cuestionario: Está conformado por un conjunto de preguntas que son insertadas por el usuario autenticado como profesor, dicho cuestionario una vez creado puede ser resuelto por cualquier estudiante que acceda al sistema.

Módulo Evaluador: Encargado de calificar el cuestionario resuelto por el estudiante, utiliza el razonamiento basado en casos para determinar casos similares.

Mapa Conceptual: Entidad que se genera una vez calificado un cuestionario con el propósito de representar el conocimiento de forma gráfica para lograr una mayor comprensión, cada estudiante posee un único mapa.

Nodos: Se le denomina nodo a cada uno de los elementos contenedores de contenido del mapa conceptual.