

Universidad de las Ciencias Informáticas
Facultad 4



Desarrollo de un componente que facilite la evaluación del aprendizaje en la Plataforma Educativa ZERA.

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

Autores

Marinés Alemán Llano

Yoandy Pérez Cáceres

Tutor

Ing. Nilber Barbán Góngora

Cotutores

Ing. Mairelis Gari Maribona

Ing. Yoennis Garrido Vargas

La Habana, junio de 2011
"Año 53 de la Revolución"

Declaración de autoría

Declaramos que somos los únicos autores del trabajo “Desarrollo de un componente que facilite la evaluación del aprendizaje en la Plataforma Educativa ZERA” y autorizamos a la Facultad 4 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Autor

Marinés Alemán Llano

Autor

Yoandy Pérez Cáceres

Tutor

Ing. Nilber Barbán Góngora

Cotutor

Ing. Mairelis Gari Maribona

Cotutor

Ing. Yoennis Garrido Vargas



De Marínés

A mis padres; Belkys y Héctor, por su infinito amor.

A mis segundos padres; Elia y Raúl, por acogerme como una hija.

De Yoandy

A mi madre, mi padre y mi hermano por su apoyo y amor en todo momento.

A mis tías, mis tíos y a los que ya no tengo a mi lado.

*“Creo en aquel que es capaz de alzar su puño y
su voz por defender las ideas que ama”*



Agradecimientos

De Marinés

A mi mamá Belkys; por darme cada día todo su amor sin medidas, por ser sencillamente la razón de mi existir, por ella es que estoy hoy aquí.

A mi mamá Elia; por criarme, por educarme, por estar pendiente de mi salud, de mi alimentación, por consentirme tanto. Hoy tu sueño se ha hecho realidad.

Al viejo; a papi por su apoyo durante todos estos difíciles años, Tato salió a buscar resultados y los consiguió.

A pipo Raúl; por sus consejos, sus enseñanzas, su cariño, por todo ello gracias, ya Cuty es una ingeniera.

A Yoandy; mi novio, mi compañero, mi hermano; por su amor, respeto, comprensión y paciencia durante estos largos años. No sé que hubiera sido de mí sin tu apoyo en todo este tiempo. Gracias Tusito, Te amo.

A Dios, por darme la oportunidad de vivir y por no abandonarme nunca.

De Yoandy

A mi mamá; por ser ella el motivo de todo mi empeño, por su amor, por ser ella mi ejemplo a seguir.

Mi papá; por haberme ayudado en todo momento, por todas sus enseñanzas y por todo su apoyo.

Mi hermano; sabes que te quiero aunque nos pasemos el día discutiendo.

A mi familia, a mis tías, tíos y primos; a todos ellos gracias por haberme apoyado siempre... son muchos, no los puedo mencionar a todos.

A mi compañera de tesis, a mi hermana, mi amiga, mi novia, a Marinés; por ser ella una de mis fuentes de energía y apoyo en todo momento. Gracias por aguantarme tanto en tantos años. Yo también te amo mi cielo, ¡más que tú!

A Tania; por ser ella mi segunda madre.

Mencionarlos a todos no basta, también sé que son muchos... a todos los que de una forma u otra estuvieron ahí cuando hacía falta, a ellos, muchas gracias.

RESUMEN.

En la actualidad muchos son los centros educativos que precisan mantener un proceso de enseñanza aprendizaje de elevada calidad, para garantizar el nivel científico de sus estudiantes. Esto contribuye a que se lleven a cabo procesos tan elementales como la presentación de los contenidos educativos de cada una de las materias que integran su plan de estudios, acompañados de un gran número de recursos interactivos, así como la realización de tareas que garanticen una ejercitación de los conocimientos y habilidades creadas; para posteriormente realizar una evaluación. La Plataforma Educativa ZERA, al igual que muchas otras, reconoce la importancia de la evaluación como elemento capaz de determinar el grado de instrucción de un individuo. El objetivo principal de este trabajo recae en el desarrollo de un componente que permita una valoración eficiente del aprendizaje. Durante el desarrollo de la presente investigación y con el firme propósito de cumplir con los objetivos trazados, se realizó el estudio de las plataformas educativas en el ámbito nacional e internacional. Se justificaron además las tecnologías, herramientas, lenguajes y metodologías que serán utilizadas para el desarrollo de un software de este tipo.

Palabras claves: aprendizaje, contenidos educativos, ejercicios, evaluación, plataformas educativas.

RESUMEN	5
INTRODUCCIÓN	10
CAPÍTULO I Fundamentación Teórica	14
1.1 Introducción.....	14
1.2 Conceptos asociados al dominio del problema.....	14
1.2.1 Software educativo.....	14
1.2.2 Plataforma educativa.....	14
1.2.3 Sistemas de Gestión de Aprendizaje (LMS).....	15
1.2.4 Sistemas de Gestión de Contenidos para el Aprendizaje (LCMS).....	15
1.2.5 Hiperentorno de Aprendizaje.....	15
1.2.6 Ejercicio.....	15
1.2.7 Tipologías de ejercicios.....	16
1.2.8 IMS QTI.....	16
1.3 Sistemas similares.....	16
1.3.1 Sistemas a nivel internacional.....	16
1.3.2 Sistemas a nivel nacional.....	17
1.4 Estudio de las metodologías y estándares para el desarrollo del software.....	18
1.4.1 Metodologías tradicionales.....	18
1.4.2 Metodologías ágiles.....	20
1.4.3 Estándares involucrados en el desarrollo del software.....	21
1.5 Entorno de desarrollo.....	21
1.5.1 El Lenguaje de Modelado Unificado (UML).....	22
1.5.2 Herramientas CASE.....	22

1.5.3	Lenguajes de desarrollo.	23
1.5.4	Framework.	25
1.5.5	Ambiente de Desarrollo Integrado (IDE).	27
1.5.6	Técnica para Mapear Objetos Relacionales (ORM).	28
1.5.7	Sistema Gestor de Base de Datos (SGBD).	30
1.5.8	Servidor web.	30
1.6	Conclusiones parciales.	31
CAPÍTULO II Características del Sistema.		32
2.1	Introducción.	32
2.2	Modelo de dominio.	32
2.2.1	Conceptos del dominio.	32
2.2.2	Diagrama del modelo de dominio.	35
2.3	Descripción del sistema propuesto.	36
2.4	Requisitos del software.	36
2.4.1	Requisitos funcionales (RF).	36
2.4.2	Requisitos no funcionales (RNF).	37
2.5	Modelo de casos de uso del sistema.	40
2.5.1	Descripción de los actores del sistema.	40
2.5.2	Consideraciones previas a los diagramas del análisis.	41
2.5.3	Patrones de casos de uso.	41
2.5.4	Diagrama de casos de uso del sistema.	44
2.5.5	Descripción de los casos de uso del sistema.	44
2.6	Conclusiones parciales.	47
CAPÍTULO III Análisis y Diseño del Sistema.		48

3.1	Introducción.....	48
3.2	Modelo de análisis.....	48
3.2.1	Diagramas de clases del análisis.....	49
3.3	Patrón arquitectónico Modelo – Vista – Controlador en symfony.....	50
3.4	Patrones de diseño utilizados.....	51
3.5	Modelo de diseño.....	51
3.5.1	Diagrama de clases del diseño.....	52
3.6	Diseño de la base de datos.....	53
3.6.1	Modelo entidad relación.....	53
3.7	Conclusiones parciales.....	53
CAPÍTULO IV Implementación y Prueba.....		54
4.1	Introducción.....	54
4.2	Módulo de Implementación.....	54
4.2.1	Diagrama de despliegue.....	54
4.2.2	Diagrama de componentes.....	55
4.3	Pruebas de software.....	56
4.3.1	Niveles de prueba.....	57
4.3.2	Métodos de prueba.....	57
4.4	Resultados obtenidos.....	73
4.5	Conclusiones parciales.....	74
CONCLUSIONES.....		75
RECOMENDACIONES.....		75
REFERENCIAS BIBLIOGRÁFICAS.....		76
BIBLIOGRAFÍA.....		79

Índice de tablas.

Tabla 1.1 Características de NetBeans 6.9.	28
Tabla 2.1: Descripción de los actores del sistema.....	41
Tabla 4.1 Resultados de las pruebas de liberación.	74

Índice de imágenes.

Figura 1.1 RUP en dos dimensiones.....	19
Figura 2.1. Representación del modelo de dominio.	35
Figura 2.2 Patrón Concordancia – Reusabilidad.	42
Figura 2.4 Patrón Concordancia – Especialización.	42
Figura 2.5 Patrón CRUD – Completo.	43
Figura 2.6 Patrón Múltiples actores – Roles comunes.	43
Figura 2.3 Diagrama de caso de uso del sistema.....	44
Figura 3.1 Estereotipo que representa la clase de interfaz.	48
Figura 3.2 Estereotipo que representa la clase control.	49
Figura 3.3 Estereotipo que representa la clase de entidad.	49
Figura 3.4 Estructura del patrón arquitectónico Modelo-Vista-Controlador (MVC).....	50
Figura 3.5 Modelo entidad – relación.	53
Figura 4.1 Diagrama de despliegue.	55
Figura 4.2 Diagrama de componentes.....	56

INTRODUCCIÓN.

La evolución de la tecnología ha tenido un gran impacto en todas las esferas de la sociedad y la rama de la educación no se ha visto exenta de esto. Una muestra fehaciente lo constituye el desarrollo del software educativo, utilizado como soporte a los procesos de enseñanza-aprendizaje. Con el vertiginoso avance de Internet y el auge de las tecnologías web se crearon nuevos servicios, entre ellos ocupa un lugar primordial la educación a distancia. Esta dio sus primeros pasos con la introducción de los conocidos “cursos virtuales” los cuales fueron enriqueciéndose cada vez más con disímiles recursos y actualmente se presentan como un elemento medular del “e-learning” o teleformación.

A raíz de toda esta revolución en materia de técnicas de enseñanza aparecen los Sistemas de Gestión de Aprendizaje Virtual, estos incluyen control de acceso, elaboración de contenido educativo, herramientas de comunicación, la administración de grupos de estudiantes entre otras funcionalidades. Dichos sistemas tienen como objetivo principal permitir la distribución de archivos de texto, audio y video, generar exámenes que serán desplegados en línea.

Dentro de los sistemas anteriormente mencionados, la evaluación constituye un elemento fundamental que es manejada, a través de los cuestionarios. Estos proveen a los docentes de una herramienta para evaluar el rendimiento y conocimiento de los estudiantes y a éstos últimos les permite evaluar su aprendizaje mediante la realización de diferentes tipos de ejercicios, los cuales les suministran retroalimentación constante e inmediata durante su realización.

La Universidad de las Ciencias Informáticas (UCI), cuya misión es producir software y servicios informáticos a partir de la vinculación estudio-trabajo como modelo de formación está enfocada al desarrollo de sistemas que apoyen el proceso de enseñanza-aprendizaje, entre dichos sistemas se encuentra la Plataforma ZERA. Un aspecto esencial dentro de dicha plataforma, lo constituye el hecho de garantizar una estabilidad asociada a realizar una periódica evaluación del estudiante para poder constatar el grado con que el mismo es capaz de apropiarse de los conocimientos que le son impartidos por el profesor. Por otra parte constituye una imperiosa necesidad ofrecer al estudiante un mayor número de recursos interactivos que propicien una motivación hacia el proceso de aprendizaje.

Un aspecto de máxima prioridad lo constituye el hecho de poder brindar una atención diferenciada de acuerdo a las características propias de cada estudiante. También constituye un elemento primordial poder brindar todo un seguimiento de las actividades que el mismo realiza. Además de colocar al alcance del estudiante un gran cúmulo de ejercicios interactivos con una variada cantidad de recursos asociados que desarrollen en él las más disímiles habilidades.

Sin embargo actualmente las plataformas educativas presentan limitaciones; la imposibilidad de poder asignar determinados grupos de ejercicios a estudiantes específicos de forma diferenciada según los intereses del profesor, también la poca variedad de interacciones o tipos de cuestionarios, no aprovechando al máximo las ventajas que estas ofrecen en materia de aprendizaje; así como el hecho de tener escasos recursos asociados a los ejercicios. Desde otro punto de vista la insuficiencia de los sistemas de estadísticas que permitan una rápida identificación de los contenidos con mayor nivel de dificultad para el estudiante.

Una vez analizadas las necesidades existentes se define como **problema a resolver** ¿Cómo facilitar la evaluación del aprendizaje en la Plataforma Educativa ZERA?

El **objetivo general** de este trabajo es el desarrollo de un componente que facilite la evaluación del aprendizaje en la Plataforma Educativa ZERA del Proyecto Alfaomega.

De una manera más detallada, los **objetivos específicos** son:

1. Realizar un análisis del estado del arte.
2. Investigar el estándar IMS-QTI en las plataformas educativas y su integración con otras especificaciones para la evaluación.
3. Aplicar mecanismos de diseño que garanticen el cumplimiento de las necesidades del componente propuesto.
4. Desarrollar un componente que facilite la evaluación del aprendizaje.
5. Validar la solución propuesta.

El **objeto de estudio** fijado son las plataformas digitales de contenidos educativos. Y el **campo de acción** en que se enmarca el trabajo recae en los componentes para la evaluación del aprendizaje en las Plataformas Educativas.

La **idea a defender** afirma que: Si se implementa un componente para la evaluación del aprendizaje en la Plataforma Educativa ZERA, se apoyará el proceso de evaluación, logrando una mayor flexibilidad y personalización del mismo.

Métodos de investigación utilizados:

Métodos Teóricos:

El método histórico-lógico se aplicó en la investigación para conocer la evolución y desarrollo de los sistemas de evaluación del aprendizaje en plataformas digitales de contenidos educativos a partir de 1990. El método analítico-sintético se utilizó para el análisis de la teoría y extracción de los principales conceptos a incluir en el marco teórico, además para el estudio y análisis de la información, lo que permitió obtener los elementos principales relacionados con la plataforma de gestión de aprendizaje, para dar cumplimiento a las tareas de investigación. Mediante la modelación se adquirieron los datos necesarios para efectuar la abstracción y de esta manera representar la realidad, lo que permitió el diseño del componente propuesto.

Métodos Empíricos:

Se utilizó la observación para recoger datos relevantes durante el desarrollo del proyecto. A través del análisis estático se realizaron exámenes al software con el objetivo de descubrir errores y un caso de estudio contribuyó a determinar la efectividad de las herramientas y el aprendizaje de las mismas. La replicación como parte del experimento se utilizó para determinar la mejor vía de desarrollo del producto, creando partes del mismo de varias formas, hasta obtener la mejor solución. Todo esto posibilitó llevar a cabo la implementación del componente.

Para cumplir con lo expuesto anteriormente se citan un conjunto de **tareas investigativas**:

1. Investigación de las tendencias tecnológicas actuales y estándares más adoptados en la construcción de plataformas educativas a nivel mundial.
2. Realización de un estudio acerca de las metodologías de desarrollo de software existentes y las herramientas de modelado.
3. Realización de un estudio acerca de las tecnologías y herramientas de desarrollo.
4. Incorporación de los requisitos funcionales y no funcionales identificados.

5. Diseño e implementación del componente propuesto.
6. Realización de las pruebas de software al componente de la Plataforma Educativa ZERA.
7. Integración de la solución planteada al sistema mientras este se desarrolla.

Estructura Capítular

Capítulo I: Fundamentación Teórica:

En este Capítulo se hará referencia a los elementos teóricos que soportan la investigación, se presentan los lenguajes de programación y modelado, así como tecnologías y metodologías que se ajustan al desarrollo del trabajo, fundamentando su selección en base al estudio realizado. Además se describen las soluciones similares existentes.

Capítulo II: Características del Sistema:

En este Capítulo se realizará una descripción de las características del sistema a implementar. Además se especifican los requisitos funcionales y no funcionales.

Capítulo III: Análisis y Diseño del Sistema:

Se realiza el diseño del sistema, así como los diagramas de clases web, que brindan una visión clara del producto.

Capítulo IV: Implementación y Prueba:

Se desarrolla la Implementación, utilizando el lenguaje de programación y la metodología seleccionada, documentándose en este el modelo de datos, el modelo de despliegue y las tareas de ingeniería. Además se definen los tipos de pruebas y los casos de prueba que se le realizarán al sistema.

CAPÍTULO I

Fundamentación Teórica.

1.1 Introducción.

En este capítulo se desarrollará la fundamentación teórica, la cual incluye el estado del arte del tema a investigar. Además se realizará un análisis preciso de otras soluciones existentes en el mundo hasta el momento y la descripción de las herramientas y tecnologías a utilizar.

1.2 Conceptos asociados al dominio del problema.

1.2.1 Software educativo.

Según Jaime Sánchez define el concepto genérico de Software Educativo como cualquier programa computacional cuyas características estructurales y funcionales sirvan de apoyo al proceso de enseñar, aprender y administrar.

Según Rodríguez Lamas (2000), es una aplicación informática, que soportada sobre una bien definida estrategia pedagógica, apoya directamente el proceso de enseñanza aprendizaje constituyendo un efectivo instrumento capaz de enriquecer vigorosamente el desarrollo educacional del hombre del próximo siglo.

Después de analizar varios conceptos se defiende el abordado por Sánchez, pues resulta más concreto y flexible.

1.2.2 Plataforma educativa.

Las plataformas son Entornos Virtuales de Educación y Aprendizaje (EVEA), que surgen y se desarrollan con la intención de dar soporte a la docencia, se apoyan en sistemas informáticos basados habitualmente en protocolos de la web con herramientas adaptadas a las necesidades educacionales requeridas. (1)

Las plataformas educativas permiten estimular la idea de cooperación y de interacción, como aspectos centrales del proceso de aprendizaje y enseñanza, mediante el uso de herramientas colaborativas que favorecen la adquisición de aprendizajes significativos. (2)

Después de analizados los conceptos anteriores se puede afirmar que plataformas educativas son un entorno virtual que acoge variadas herramientas para fines docentes. Su función es permitir la creación y gestión de cursos que permitan elevar el nivel de conocimiento de sus usuarios.

1.2.3 Sistemas de Gestión de Aprendizaje (LMS).

Los sistemas de gestión de aprendizaje (Learning Management Systems) son aplicaciones Web que proveen las funciones administrativas y de seguimiento necesarias para posibilitar y controlar el acceso a los contenidos, implementar recursos de comunicaciones y llevar a cabo el seguimiento de quienes utilizan la herramienta. En general, los LMS facilitan la interacción entre los docentes y los estudiantes, aportan herramientas para la gestión de contenidos académicos y permiten el seguimiento y la evaluación. (2)

1.2.4 Sistemas de Gestión de Contenidos para el Aprendizaje (LCMS).

Los sistemas de gestión de contenidos para el aprendizaje (Learning Content Management Systems) son LMS que permiten la gestión de contenidos. Entendiendo por Gestión de Contenidos el proceso que va de la creación de un Objeto de Aprendizaje (OA), que es la unidad mínima de contenido, hasta una publicación y seguimiento. (2)

1.2.5 Hiperentorno de Aprendizaje.

Sistema informático basado en tecnología hipermedia, en la que se mezclan disímiles dispositivos que representan las diversas tipologías de software educativo con el diseño de atender las necesidades didácticas.

1.2.6 Ejercicio.

Según la Real Academia Española un **ejercicio** no es más que:

Actividad destinada a adquirir, desarrollar o conservar una facultad o cualidad psíquica. (3)

1.2.7 Tipologías de ejercicios.

Es el formato en el que se puede presentar un determinado ejercicio. Las mismas pueden ser: selección simple, selección múltiple, seleccionar texto, completar por escritura, completar por desplazamiento, ordenar palabra, ordenar medias enlazar elementos de 2 conjuntos, entre otras.

1.2.8 IMS QTI.

La especificación de IMS sobre Question & Test Interoperability (QTI) describe un modelo para la representación de preguntas, exámenes y sus correspondientes resultados, permitiendo el intercambio de estos entre herramientas de autor, repositorios de objetos de aprendizaje y sistemas de gestión de contenidos educativos. La especificación se basa en un modelo de datos descrito en UML y con una traslación directa a XML para su intercambio. (4)

1.3 Sistemas similares.

1.3.1 Sistemas a nivel internacional.

La aplicación de las TIC en los procesos de enseñanza y aprendizaje, así como los cambios en las formas de aprendizaje se han visto reflejados en los Entornos Virtuales de Enseñanza y Aprendizaje (EVEA), los cuales cuentan con herramientas adaptadas a las necesidades de la institución para la que se implementan o adaptan. Entre las más usadas se encuentran:

Entorno de Aprendizaje dinámico Orientado a objetos y Modular (Moodle):

Es una aplicación web gratuita que los educadores pueden utilizar para crear sitios de aprendizaje efectivo en línea. Fue creado por Martin Dougiamas, administrador de herramienta para la creación de cursos en la web (WebCT) en la Universidad Tecnológica de Curtin. La gestión de la evaluación se realiza a través de un módulo llamado Cuestionario que presenta características tan deseables como que los profesores pueden definir una base de datos de preguntas que podrán ser reutilizadas en diferentes cuestionarios, además que los cuestionarios se califican automáticamente, y pueden ser recalificados si se modifican las preguntas entre otras. (5)

Claroline:

Es una plataforma de aprendizaje y trabajo virtual que tuvo sus inicios en el año 2001 en la Universidad Católica de Louvain en Bélgica (UCL), publicada bajo una licencia Open Source, posee dos herramientas

para el proceso de evaluación; un módulo llamado Ejercicios el cual consiste en un área en la que los estudiantes pueden enviar archivos que piensan que pueden ser útiles para otros miembros del curso o, en el caso de que el curso lo requiera, el envío de un trabajo determinado. Además de un módulo Cuestionario el cual consiste en un generador de pruebas, que pueden ser de varios tipos. Puede ser una herramienta muy interesante para generar pruebas de autoevaluación, para que su alumnado pueda comprobar sus conocimientos. (6)

Web Course Tool (WebCT):

Es una de las más prestigiosas plataformas educativas en la actualidad. Cuenta con un módulo especialmente diseñado para la autoevaluación. La evaluación automática puede evaluar múltiples opciones, respuestas concretas y respuestas cortas. Una sustancial desventaja en contra de la WebCT radica en que no puede calificar automáticamente preguntas de desarrollo. (7)

1.3.2 Sistemas a nivel nacional.

En Cuba, como en el resto de mundo, se han desarrollado plataformas para impartir cursos a distancia, como las que se presentan brevemente caracterizadas a continuación:

Sistema de Enseñanza Personalizada A Distancia (SEPAD):

Desarrollado en la Universidad Central de Las Villas (UCLV), es una plataforma que cuenta con varias interfaces que se mueven desde el ambiente clásico Web para los usuarios que tienen posibilidad de conexión en línea, una versión de clientes para poder acceder a los servicios de la plataforma a través de correo electrónico o una versión multimedia, capaz de ejecutarse sin necesidad de conexión alguna. Además, cuenta con un aula virtual donde se puede acceder a diferentes materiales, auto-evaluaciones, búsquedas, calificaciones, así como mensajería interna, foros de debate, anuncios y salas de chat. (8)

Dicha plataforma está compuesta además por una herramienta de publicación muy sencilla lo que trae aparejado que determinadas características de la didáctica se pierdan o que limiten la creatividad e innovación del profesor.

Mundicampus:

Desarrollado por la empresa española Mundicampus y el Centro de Estudios de Ingeniería de Sistemas (CEIS) del Instituto Superior Politécnico José Antonio Echeverría (CUJAE). Es una plataforma cómoda y flexible que permite la impartición de cursos a distancia en un entorno Web. (8)

Esta plataforma no presenta una interfaz muy agradable para el usuario, haciendo difícil la retroalimentación que es un aspecto vital en el proceso de enseñanza-aprendizaje. Al igual que SEPAD, también está dirigida fundamentalmente al ámbito nacional, sin acogerse a las normas que siguen otras plataformas.

AprenDIST:

Sistema desarrollado en el Instituto Superior Politécnico José Antonio Echeverría, es una plataforma digital interactiva para la educación a distancia que permite crear los más diversos cursos y cuenta con varias herramientas como chat, foros, correo electrónico, biblioteca. (9)

Esta plataforma fue diseñada con un marcado enfoque a la realidad cubana; lo que condujo a que se descuidaran las tecnologías que revolucionan el mercado mundial en esta área. Por ejemplo; no se abarcan los estándares internacionales IMS-QTI o SCORM, que rigen una parte importante del trabajo en los entornos virtuales, ya que permiten exportar o importar otros contenidos, ejercicios y demás recursos.

1.4 Estudio de las metodologías y estándares para el desarrollo del software.

Las metodologías o procesos de desarrollo de software ofrecen un método para la creación de aplicaciones flexibles y robustas de un modo organizado y disciplinado, facilitando su comprensión. Además abarcan procedimientos, técnicas, documentación y herramientas que tienen como base fundamental el fortalecimiento de los procesos de desarrollo de un software. Actualmente no existe una metodología universal que le haga frente con éxito a cualquier proyecto de desarrollo de software, ni que se adapte a todo tipo de sistema, debido a que las características y recursos de cada equipo de desarrollo no siempre son las mismas.

1.4.1 Metodologías tradicionales.

Las metodologías tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo del software, con el fin de conseguir un software más eficiente. Para ello, se hace énfasis en la planificación

total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del software. Se centran especialmente en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada. Además, las metodologías tradicionales no se adaptan adecuadamente a los cambios, por lo que no son métodos adecuados cuando se trabaja en un entorno, donde los requisitos no pueden predecirse o bien pueden variar.

1.4.1.1 Proceso Unificado de Desarrollo (RUP).

RUP es un proceso formal: Provee un acercamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga los requerimientos de los usuarios finales (respetando cronograma y presupuesto). Fue desarrollado por Rational Software, y está integrado con toda la suite Rational de herramientas. Puede ser adaptado y extendido para satisfacer las necesidades de la organización que lo adopte. Es guiado por Casos de Uso y centrado en la arquitectura, utilizando UML como lenguaje de modelado. Define cuatro fases esenciales (Inicio, Elaboración, Construcción y Transición) y nueve flujos de trabajos; seis de Ingeniería (Modelado del Negocio, Requerimientos, Análisis y Diseño, Implementación, Prueba y Despliegue) y tres de apoyo. (10)

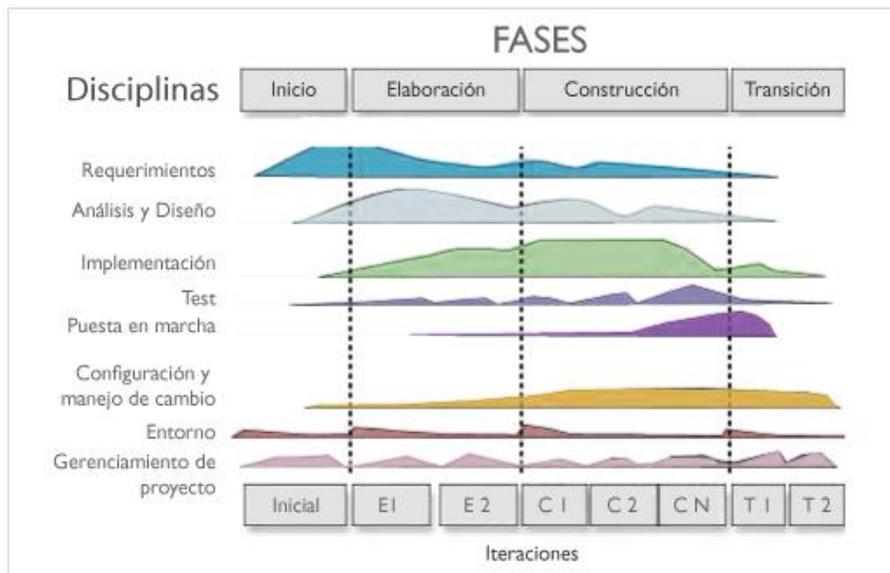


Figura 1.1 RUP en dos dimensiones.

Características de RUP:

- ✓ Establece un desarrollo iterativo.
- ✓ Permite la administración de requisitos.
- ✓ Hace uso de una arquitectura basada en componentes.
- ✓ Permite llevar un control de cambios.
- ✓ Permite realizar un modelado visual del software.
- ✓ Verifica la calidad del software.

1.4.2 Metodologías ágiles.

En febrero de 2001, tras una reunión celebrada en Utah-EEUU, nace el término “ágil” aplicado al desarrollo de software. En esta reunión participan un grupo de 17 expertos de la industria del software, incluyendo algunos de los creadores o impulsores de metodologías de software. Su objetivo fue esbozar los valores y principios que deberían permitir a los equipos desarrollar software rápidamente y respondiendo a los cambios que puedan surgir a lo largo del proyecto.

1.4.2.1 Extreme Programing (XP).

Programación Extrema o Extreme Programing (XP) por sus siglas en inglés, es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios.

Características de XP:

- ✓ **Pruebas Unitarias:** se basa en las pruebas realizadas a los principales procesos, de tal manera que el adelantarse en algo hacia el futuro, se puedan hacer pruebas de las fallas que pudieran ocurrir. Es como si pudieran obtenerse 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. (11)

Selección de la metodología para solucionar el problema

Por las características expuestas, se ha decidido que la metodología sea RUP, la cual basa su trabajo fundamentalmente en la documentación del software y expone un conjunto de actividades que están orientadas a visualizar, especificar, construir, documentar y comunicar los artefactos necesarios para el desarrollo de un software con calidad, presentando una exhaustiva definición de artefactos para ello. Además utiliza el lenguaje unificado de modelado (UML), cuya utilización de diagramas y gráficos brindan una mejor perspectiva de lo que se quiere.

1.4.3 Estándares involucrados en el desarrollo del software.

Como ya se ha mencionado previamente, la especificación IMS Question and Test Interoperability (IMS QTI) permite crear preguntas individuales y evaluaciones completas. El objetivo principal de esta especificación es permitir el intercambio de preguntas, evaluaciones y resultados entre distintas herramientas. Con este propósito IMS QTI plantea un modelo en el que se definen los componentes principales que intervienen en el proceso de evaluación y, adicionalmente a este modelo, se proporciona un formato de contenido para almacenar las preguntas de manera independientemente del sistema o herramienta de autoría utilizada para crearlas. (12)

El componente a implementar funcionará como un motor de ejecución proporcionando infraestructura y componentes varios para la creación de herramientas de evaluación en línea, en este caso compatible con el estándar IMS QTI aunque brindando un soporte parcial del mismo.

1.5 Entorno de desarrollo.

Para dar solución a la problemática, se centra la búsqueda en las tecnologías web, debido a las ventajas que proporciona en cuanto a eficiencia, flexibilidad y rapidez; evitando con esta selección las complicadas instalaciones y configuraciones a la hora de desplegar el producto y en los posteriores soportes que se puedan brindar al usuario.

1.5.1 El Lenguaje de Modelado Unificado (UML).

UML es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos. (13)

Los principales beneficios de UML son:

- ✓ Mejores tiempos totales de desarrollo (de 50 % o más).
- ✓ Modelar sistemas (y no sólo de software) utilizando conceptos orientados a objetos.
- ✓ Establecer conceptos y artefactos ejecutables.
- ✓ Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- ✓ Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- ✓ Mejor soporte a la planeación y al control de proyectos.
- ✓ Alta reutilización y minimización de costos.

1.5.2 Herramientas CASE.

En inglés Computer Aided Software Engineering (CASE) y en su traducción al español significa Ingeniería de Software Asistida por Computadora. Estas herramientas pueden ayudar en muchos de los aspectos del ciclo de vida de desarrollo del software, en tareas como la realización del diseño del proyecto, cálculo de costes, implementación de una parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores, entre otras. Las herramientas CASE de modelado con UML permiten analizar y diseñar orientado a objetos y abstraer el código fuente, a un nivel donde la arquitectura y el diseño se tornan más obvios y más fáciles de entender y modificar.

- ✓ **ArgoUML. Versión 0.32:** Es una aplicación para la realización de diagramas UML, escrita en Java y publicada bajo la Licencia BSD¹. Entre sus características más atractivas se encuentran que incluye soporte para los diagramas basados en UML 1.4, ingeniería inversa, además de disposición (layout) automática del diagrama de clases, así como la generación de ficheros PNG,

¹ Esta licencia viene a decir que puedes distribuir el código del programa o el ejecutable, indicando que está amparado por la licencia BSD y que en caso de crear una aplicación derivada de otra con esta licencia, **no se puede usar el nombre del autor original para promover tu creación.**

GIF, JPG, SVG, EPS desde diagramas, internacionalización y soporte para el lenguaje de generación de código: Java, PHP, Python, C++ y Csharp (C#).

- ✓ **Visual Paradigm UML. Versión 6.4:** Es una herramienta CASE creada para el ciclo vital del desarrollo de software, permitiendo la captura de requisitos, análisis, diseño e implementación. Considerada como una herramienta muy completa y fácil de usar, con soporte multiplataforma.

Es compatible con todas las fases y roles en un proceso de desarrollo de software tales como el ingeniero de software, analistas del sistema, arquitecto del sistema, entre otros. Visual Paradigm-UML 6.4 (VP-UML 6.4) es empleado para las aplicaciones grandes y complejas que presenten un enfoque orientado a objeto.

VP-UML 6.4 proporciona características tales como soporte para UML 2.1, generación del código, ingeniería inversa así como generación de bases de datos y diagramas de flujo de datos. Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la definición de base de datos a partir del esquema de clases. (14)

Visual Paradigm constituye un candidato más fuerte para la elección de una herramienta CASE, debido principalmente a que iguala a ArgoUML en cuanto a capacidades y es mucho más integrable con el resto de las herramientas y estándares escogidos.

1.5.3 Lenguajes de desarrollo.

Un lenguaje de programación es una serie de comandos que permiten codificar instrucciones de manera que sean entendidas y ejecutadas por una computadora. (15)

Con el auge de Internet y las nuevas tecnologías, surgen nuevos lenguajes de programación, los cuales mejoran la forma de interacción del usuario con las plataformas Web.

1.5.3.1 Entorno anfitrión.

- ✓ **Lenguaje Extensible de Marcado de Hipertexto (XHTML). Versión 1.1.** Según la W3C “XHTML (Lenguaje de Marcado de Hipertexto Extensible) es una versión más estricta y limpia de HTML, que nace precisamente con el objetivo de reemplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML, XHTML extiende HTML 4.0 combinando

la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos.

(...) XHTML surge como el lenguaje cuyo etiquetado, más estricto que HTML, va a permitir una correcta interpretación de la información independientemente del dispositivo desde el que se accede a ella. (...)”

- ✓ **Hojas de Estilo en Cascadas (CSS). Versión 2.** Según la W3C “CSS, es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.

CSS se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores Web controlar el estilo y el formato de múltiples páginas Web al mismo tiempo. (...)”

- ✓ **JavaScript.** Según Mozilla Developer Network (MDN), “JavaScript es un lenguaje de script multiplataforma. Es un lenguaje pequeño y ligero; no es útil como un lenguaje independiente, más bien está diseñado para una fácil incrustación en otros productos y aplicaciones, tales como los navegadores Web. (...)”

Adicionar que, permite el desarrollo de una interfaz de usuario mejorada, aumentando la experiencia del usuario en la Web.

1.5.3.2 Entorno servidor.

- ✓ **Python. Versión 2.6.6.** Es un lenguaje de programación creado en el año 1990 por Guido van Rossum, es un lenguaje eficiente que es considerado uno de los más ordenados para el desarrollo. Con Python se pueden crear cualquier tipo de aplicaciones, incluyendo a las aplicaciones Web. La mayor de sus desventajas es ser un lenguaje lento, aunque hay que destacar que posee una cantidad de librerías muy útiles.
- ✓ **Hypertext Preprocessor (PHP). Versión 5.3.3.** Entre las ventajas más destacadas de PHP se puede mencionar que es un lenguaje completamente libre por lo que se destaca por tener una

amplia comunidad, dándole un mayor soporte al mismo y mejorando el tiempo de recuperación ante errores encontrados. La documentación del lenguaje puede ser encontrada en varios idiomas por lo que mejora la agilidad con la cual un programador puede aprender/mejorar su desarrollo en PHP.

Facilita la realización de grandes cosas en pocas líneas de código y trabaja en combinación con otras tecnologías como **Perl**, **JavaScript**, **Phyton**, entre otras. Es muy legible y fácil de aprender, no soporta directamente punteros, de forma que no existen los problemas de depuración provocados por estos y es de fácil acceso para todos. (16)

En términos técnicos se debe mencionar que es un lenguaje seguro y muy eficiente puesto que no estresa la memoria del servidor al ser un lenguaje ligero en comparación con otros lenguajes como ASP.

Por las características antes mencionadas, se selecciona PHP como lenguaje de desarrollo. Agregar además que, la Universidad posee varios proyectos los cuales son desarrollados con PHP y que pueden ser consultados en busca de componentes re-utilizables. Los integrantes del equipo de desarrollo poseen conocimientos del mismo y no será necesario un tiempo de capacitación para la implementación.

1.5.4 Framework.

El concepto framework se emplea en muchos ámbitos del desarrollo de sistemas software, no solo en el desarrollo de aplicaciones Web. En general, el término framework, se refiere a una estructura de software compuesta de componentes, los cuales pueden ser personalizados, modificados o cambiados en dependencia de las necesidades de la aplicación a desarrollar. (17)

1.5.4.1 Framework para el entorno anfitrión.

Los framework de Javascript, son componentes compatibles con la mayoría de los navegadores Web modernos, mejorando el producto a desarrollar en gran medida puesto que con los mismos se facilita el uso de tecnologías asincrónicas (AJAX), manipulación del DOM, modificación de los estilos CSS aplicados y el intercambio de datos de una forma ligera usando JSON.

- ✓ **Prototype. Versión 1.7:** Es un framework desarrollado en JavaScript por Sam Stephenson para el desarrollo sencillo y dinámico de páginas Web. Este framework simplifica gran parte del trabajo cuando se pretende desarrollar páginas altamente interactivas.

- ✓ **jQuery. Versión 1.4.3:** Es un framework de JavaScript, creado inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web.

Entre las ventajas de su uso, resaltan características como el soporte a extensiones o plugins los cuales amplían las facilidades de uso y mejoran los tiempos de producción al poderse reutilizar componentes desarrollados por terceras personas.

Se selecciona jQuery como framework de Javascript por las ventajas antes mencionadas agregando, que, este framework posee una GPL² versión 2 y otra MIT³ lo cual facilita su uso en proyectos libres o privativos, según las necesidades del cliente.

Los framework para el desarrollo de CSS son herramientas útiles que mejoran el tiempo de desarrollo en la Capa Presentación de los Estilos, dando un soporte a los estándares de la W3C y mejorando la compatibilidad entre los navegadores más usados.

- ✓ **BluePrintCSS. Versión 1.0:** Es un framework para CSS, que permite reducir el tiempo de desarrollo del proyecto. Provee al desarrollador de una base sólida que incluye tipografía sensible, plugins útiles e incluso una hoja de estilo para la impresión. (18)

1.5.4.2 Framework para el entorno servidor.

Los framework para el desarrollo de aplicaciones basadas en el lenguaje seleccionado (PHP), son útiles puesto que en su mayoría están basados en el patrón Modelo-Vista-Controlador (MVC), el cual facilita la implementación y posterior soporte a secciones de la aplicación a desarrollar. Los framework para PHP, implementan mecanismos de autenticidad y control de acceso, facilitan la integración con Sistemas Gestores de Base de Datos (SGBD), brindan opciones para la internacionalización del producto implementado y usan configuraciones por defecto que son, en su mayoría, las más usadas.

² Licencia Pública General (GPL). El objetivo de esta licencia es **proteger el software libre de apropiaciones con restricción.**

³ Licencia asociada al prestigioso Instituto Tecnológico de Massachusetts (MIT). Esta licencia obliga a adjuntar la nota de la licencia en el software y en sus partes más importantes. A cambio **carece de restricciones**, por lo que puedes usar, copiar, modificar el código e incluso venderlo.

- ✓ **Zend Framework. Versión 1.11.3:** Es uno de los framework de PHP de amplio uso, entre sus mejores características se pueden mencionar la baja dependencia entre sus componentes. Esta arquitectura hace que sus componentes puedan ser usados de manera independiente, pero, como mejor configurado puede estar es, combinando sus variados componente, lo cual lo convierte en un framework potente y altamente extensible. Su implementación está basada en conceptos orientados a objetos, usando MVC y una capa de abstracción de base de datos de fácil uso.
- ✓ **Symfony. Versión 1.4.6:** Es un framework PHP que facilita el desarrollo de las aplicaciones web. Se encarga de todos los aspectos comunes y aburridos, dejando que el programador se dedique a aportar valor desarrollando las características únicas de cada proyecto. Symfony es, además, el framework más documentado del mundo, ya que cuenta con miles de páginas de documentación distribuidas en varios libros gratuitos y decenas de tutoriales. (19)

Es un framework fácil de instalar y configurar en sistemas operativos GNU/Linux, Windows y Mac. Su funcionamiento se basa en “convenir en vez de configurar” puesto que solo se deben modificar las configuraciones que no son convencionales. Una de sus grandes fortalezas es la amplia variedad de plugins que tiene a su disposición.

Por las características antes expuestas, se selecciona a Symfony como framework de desarrollo para PHP, teniendo presente también la amplia documentación generada en diversos idiomas y su alta seguridad y estabilidad. Symfony ha soportado grandes pruebas unitarias y funcionales lo cual lo convierte en un gran candidato, se pueden mencionar a sitios como Yahoo Ask con unos veinte millones de usuarios.

1.5.5 Ambiente de Desarrollo Integrado (IDE).

Los IDEs son programas con un conjunto de herramientas que brindan amplias comodidades en el desarrollo de aplicaciones. Las ventajas de usar IDEs en el desarrollo de la problemática planteada anteriormente, son: completamiento de código, disminuyendo el tiempo de desarrollo, depuración de código, resaltado de sintaxis, entre otras.

- ✓ **Eclipse. Versión 3.5.1:** Es un IDE de código abierto independiente de la plataforma, es una aplicación de cliente enriquecido, emplea plug-ins para proporcionar toda su funcionalidad, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las

necesite el usuario o no. Eclipse es, en el fondo, únicamente un almacén sobre la que se pueden montar herramientas de desarrollo. La arquitectura de plug-ins permite, además de integrar diversos lenguajes, introducir otras aplicaciones. (20)

- ✓ **NetBeans. Versión 6.9:** Es un IDE de código abierto, el cual soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Existe un número importante de módulos para extenderlo. Es un producto libre y gratuito sin restricciones de uso. Entre sus características se encuentra un sistema de proyectos basado en control de versiones y refactorización. Todas las funciones del IDE son provistas por plug-ins, al igual que Eclipse.

Desde julio de 2006, NetBeans IDE es licenciado bajo CDDL⁴(...). La versión 6.0 soporta las tecnologías Enterprise JavaBeans (EJB) 3.0, JAX-WS 2.1, Java Server Faces 1.2, Java Server Page 2.1, JSP Standard Tag Library (JSTL) 1.1, el editor es más rápido y más inteligente. (21)

Lenguajes Web: HTML, CSS, JavaScript	PHP
Reestructuración y búsqueda de usos para CSS y lenguajes parecidos a HTML	Anotaciones "Overrides/Implements" y "Is Overridden/Implemented"
Autocompletado y enlace para atributos id y class	Nuevo formateador con más reglas de formateo
Reestructuración de estilos en línea de CSS	Compatibilidad con PHP Zend Framework

Tabla 1.1 Características de NetBeans 6.9.

Ambos IDEs son herramientas poderosas que pueden ser usadas aunque, se destaca NetBeans por brindar soporte de forma nativa a framework como Symfony y jQuery anteriormente seleccionados para el desarrollo del componente, por lo que ha sido seleccionado como el IDE a usar.

1.5.6 Técnica para Mapear Objetos Relacionales (ORM).

Un ORM o (Object Relation Mapper) es una técnica de programación que permite convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base

⁴ Common Development and Distribution License es una licencia basada en Mozilla Public License (MPL)

de datos relacional, es decir, las tablas de la base de datos pasan a ser clases y los registros, objetos que se manejan con facilidad.

Utilizar un ORM tiene una serie de ventajas que facilitan enormemente tareas comunes y de mantenimiento:

1. **Reutilización:** La principal ventaja que aporta un ORM es la reutilización permitiendo llamar a los métodos de un objeto de datos desde distintas partes de la aplicación e incluso desde diferentes aplicaciones.
2. **Encapsulación:** La capa ORM encapsula la lógica de los datos pudiendo hacer cambios que afectan a toda la aplicación únicamente modificando una función.
3. **Portabilidad:** Utilizar una capa de abstracción permite cambiar en mitad de un proyecto de una base de datos MySQL a una Oracle sin ningún tipo de complicación.
4. **Seguridad:** Los ORM suelen implementar mecanismos de seguridad que protegen la aplicación de los ataques más comunes como inyecciones SQL.
5. **Mantenimiento del código:** Gracias a la correcta ordenación de la capa de datos, modificar y mantener el código es una tarea sencilla. (22)

Los ORM a analizar son:

- ✓ **Propel. Versión 1.5:** Uno de los más antiguos y respaldados con frameworks como symfony. Tiene muy buena documentación y sólo es compatible con PHP 5. Propel utiliza PDO (PHP Data Objects) como capa de abstracción de bases de datos. (23)
- ✓ **Doctrine. Versión 1.2.2:** Es un ORM para PHP 5.2.3 y posterior. Además de todas las ventajas que conlleva un ORM, uno de sus puntos fuertes es su lenguaje inspirado en el HQL de Hibernate, uno de los mejores ORM. También doctrine soporta migraciones, eventos, paginación e interfaces de líneas de comando (24)

Doctrine, a pesar de ser un ORM relativamente joven, es seleccionado por las características antes expuestas, tiene una documentación muy completa y su comunidad de usuarios es más activa por lo que se pueden solucionar las dudas con mayor agilidad.

1.5.7 Sistema Gestor de Base de Datos (SGBD).

Consiste en una colección de datos interrelacionados y una colección de programas para acceder a los datos. Proporciona un entorno conveniente y eficiente para los usuarios que lo usan para la recuperación y almacenamiento de la información. (25)

- ✓ **Oracle. Versión 11.2.0.1:** Es un SGBD relacional fabricado por Oracle Corporation. Se considera uno de los sistemas de bases de datos más completos, destacando su soporte de transacciones, estabilidad, escalabilidad y que es multiplataforma. Cuenta con administración de usuarios así como la administración de roles, además soporta disparadores y procedimientos almacenados.
- ✓ **PostgreSQL. Versión 8.4:** Es un SGBD Objeto-Relacional (ORDBMS), creado en el Departamento de Investigación de la Universidad de California en Berkeley y aunque su licencia es propiedad de dicha universidad, es libre para utilizar, copiar, modificar y distribuir, sin importar para los fines que se aplique, tanto comercial como académico.

También es compatible con el almacenamiento de objetos binarios, incluyendo imágenes, sonidos o vídeos. Tiene interfaces de programación nativo de C/C++, Java, NET, Perl, Python, Ruby, Tcl, ODBC, entre otros, y la documentación de carácter excepcional. Es altamente escalable, tanto en la enorme cantidad de datos que puede manejar como en el número de usuarios concurrentes que puede permitir.

Soporta casi todas las sintaxis SQL y puede almacenar grandes volúmenes de información. Cuenta con herramientas gráficas de administración y diseño de Bases de Datos tales como PgAdmin y phpPgAdmin, estas herramientas permiten que se haga sencilla la administración de las Bases de Datos. (26)

Después de haber realizado un análisis entre distintos Sistemas Gestores de Bases de Datos se ha decidido a favor de PostgreSQL principalmente por el elevado coste de las licencias de Personal Oracle, además de ser un SGBD muy eficiente cuando se almacenan un gran cúmulo de información y por su fácil integración con el framework Symfony.

1.5.8 Servidor web.

Un servidor web es un programa encargado de atender y responder diferentes tipos de peticiones realizadas por los navegadores web usando protocolos HTTP o HTTPS. Su funcionamiento consiste en

esperar peticiones de un cliente y posteriormente, devolver el recurso solicitado en caso que cumpla con las condiciones necesarias para acceder al mismo.

- ✓ **Cherokee. Version 1.0.0:** Una de las alternativas a Apache, es un servidor web *Open Source* que tiene unos 8 años de desarrollo el cual tiene una comunidad aún reducida, pero que está creciendo rápidamente. Entre sus ventajas está la existencia de una interfaz gráfica de administración que es especialmente fácil de utilizar y con la que se puede configurar y monitorizar el estado del servidor web. (27)
- ✓ **Apache. Versión 2.2.16:** Es el servidor Web de mayor uso a nivel internacional, es la mejor alternativa tanto en las opciones comerciales como libres. Este servidor Web es altamente configurable, flexible, rápido y eficiente. Presenta versiones tanto para Windows como para sistemas Linux, aportando características extendidas como personalización de las variables de entorno y soporte de reparación de errores.

Ambas alternativas son buenas, aunque se selecciona Apache por su configurabilidad y robustez, siendo además casi universal al estar en una multitud de Sistemas Operativos. Resaltar también que, Apache presenta un diseño modular excelente para ampliar las capacidades del servidor web, contando además, con una amplia variedad de módulos que se pueden instalar a medida que se necesiten.

1.6 Conclusiones parciales.

En este capítulo se realizó un resumen de los principales conceptos asociados al dominio del problema con el objetivo de lograr un mayor entendimiento del mismo y se explica de manera argumentada el objeto de estudio. Se realiza un análisis de algunas de las aplicaciones existentes en nuestro país y en el mundo que sirven de apoyo a los fines que se quieren lograr. También se ofrece una explicación de los lenguajes de programación a utilizar, el gestor de base de datos seleccionado por el cliente, así como la metodología que se considera óptima para la realización del producto.

CAPÍTULO II

Características del Sistema.

2.1 Introducción.

En este capítulo se realiza la descripción de la propuesta de solución para facilitar su comprensión. Puesto que fue definido que no es necesario un modelo completo del negocio se realiza un modelo del dominio que agrupa los principales conceptos con los que se trabajan. Además, se identifican los requisitos no funcionales y los casos de uso críticos que regirán el desarrollo del sistema. También se realizan los diagramas de casos de uso con la intención de reflejar las relaciones entre los actores del sistema y las secuencias de acciones en las que están involucrados.

2.2 Modelo de dominio.

El modelo de dominio ayuda a comprender los conceptos relacionados a los procesos que se desean desarrollar. Una vez analizado el problema e identificados los principales conceptos presentes en el mismo, se hace posible la elaboración del modelo de dominio, donde se presenta en un marco conceptual, la relación existente entre las definiciones.

2.2.1 Conceptos del dominio.

Para lograr una mayor comprensión del diagrama de modelo de dominio que se expondrá a continuación, es necesario realizar previamente la definición de los conceptos que involucra dicho modelo.

- ✓ **ZERA:** Plataforma educativa destinada a apoyar el proceso de enseñanza – aprendizaje.
- ✓ **Hiperentorno de aprendizaje:** Sistema informático basado en tecnología hipermedia, en el que se mezclan diversas tipologías de software educativo con el propósito de atender las necesidades didácticas.

CAPÍTULO II Características del Sistema

- ✓ **Estudiante:** Persona que cursa estudios, tiene el propósito de atender una materia, accede al hiperentorno en busca de conocimientos.
- ✓ **Docente (Profesor):** Persona encargada de educar a los estudiantes en el proceso de enseñanza – aprendizaje.
- ✓ **Tutor:** Persona responsable de guiar y supervisar las actividades de los estudiantes que tutorea.
- ✓ **Director:** Persona encargada de dirigir una escuela, ya sea porque es el dueño o porque se le asignó la tarea. Puede revisar y analizar la información a nivel estudiante, curso, docente y escuela.
- ✓ **Administrador local:** Realiza las tareas de administración en la plataforma instalada en la escuela.
- ✓ **Usuario de hiperentorno:** Es el usuario que accede al hiperentorno de aprendizaje, puede comportarse como un estudiante, docente, tutor, administrador local o director.
- ✓ **Ejercicio:** Actividad mental que consiste en la repetición de ciertas rutinas, con el fin de desarrollar una determinada habilidad. Los ejercicios permiten comprobar el nivel asimilación de un contenido por parte de un estudiante.
- ✓ **Tipología de ejercicio:** Es el formato en el que se puede presentar un determinado ejercicio.
- ✓ **Imagen de apoyo:** Representación gráfica que sirve de apoyo al ejercicio.
- ✓ **Asignación:** Proceso, mediante el cual se seleccionan los ejercicios y los estudiantes a los que estos estarán asociados para la evaluación.
- ✓ **Evaluación:** Proceso dinámico, continuo y sistemático, enfocado hacia los cambios de las conductas y rendimientos, mediante el cual se verifican los logros adquiridos en función de los objetivos propuestos.
- ✓ **Retroalimentación:** Mensajes de interacción para un ejercicio, con base en el número de intentos definidos para la realización de este.
- ✓ **Módulo:** Se denomina Módulo, al conjunto de elementos que agrupa las funcionalidades del producto.

CAPÍTULO II

Características del Sistema

- ✓ **Tareas:** Módulo que tiene como principal objetivo agrupar varios componentes como son: recorridos dirigidos, evidencias y/o ejercicios agrupados en una softarea.
- ✓ **Temas:** Módulo que tiene como principal objetivo permitir a los docentes, estudiantes y tutores la navegación por cada uno de los contenidos de las materias a través del índice del contenido (capítulos, temas y subtemas).
- ✓ **Docente:** Módulo que tiene como principal objetivo agrupar todas las acciones específicas que se le asignan al docente dentro del hiperentorno.
- ✓ **Prácticas:** Módulo que tiene como principal objetivo permitir a los estudiantes, la ejercitación y autoevaluación de los contenidos estudiados mediante la resolución de ejercicios.
- ✓ **Simuladores:** Módulo que tiene como principal objetivo la representación gráfica de los contenidos.
- ✓ **Índice:** Estructura de contenido, se compone de los capítulos, temas y subtemas.
- ✓ **Recorrido dirigido:** Es un resumen del contenido gestionado en clases apoyado por recursos multimedia, textos y referencias bibliográficas, el cual es asignado a un estudiante o grupos de éstos mostrado de una forma secuencial previamente definido por el docente.
- ✓ **Sección Ejercicios:** Sección donde el estudiante puede encontrar los ejercicios para su entrenamiento.
- ✓ **Sección Enfrenta Retos:** Sección que agrupa varias sub-secciones, con el objetivo de agrupar diferentes tipos de ejercicios.
- ✓ **Valora tus posibilidades:** Sección en la cual el usuario puede encontrar cuestionarios y pruebas de ingreso.
- ✓ **Aprende a resolver:** Sección en la cual el usuario puede encontrar guías de ejercicios resueltos y cuestionarios.
- ✓ **Investiga y aprende:** Sección en la cual el usuario puede encontrar temas que serán objeto de investigaciones.

2.2.2 Diagrama del modelo de dominio.

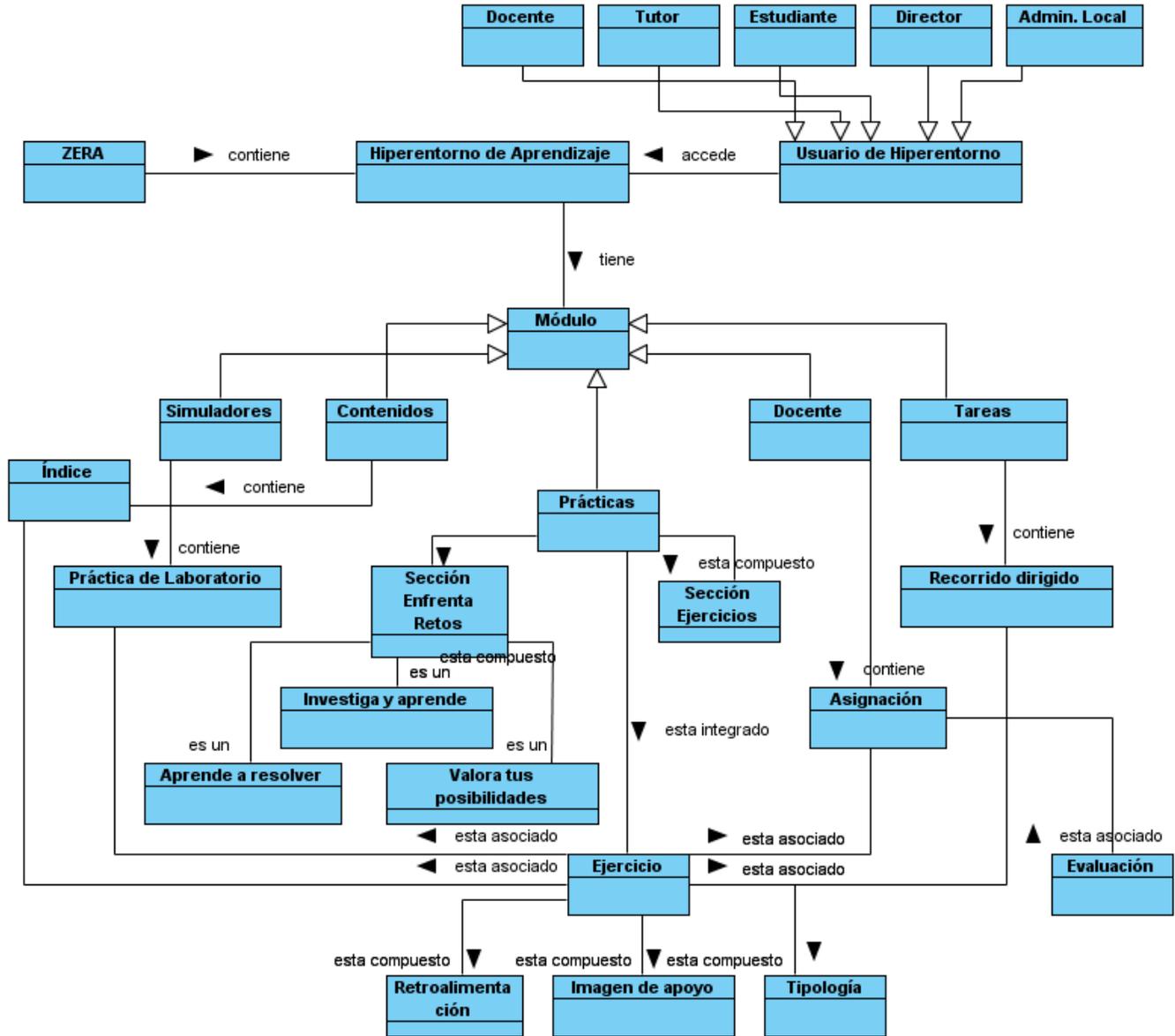


Figura 2.1. Representación del modelo de dominio.

2.3 Descripción del sistema propuesto.

El componente propuesto tiene como principal objetivo ofrecer al estudiante una gran variedad de ejercicios donde el mismo pueda ejercitar el contenido previamente estudiado. El componente estará dividido en dos secciones fundamentales: ejercicios y enfrenta retos. La sección de ejercicios debe permitir acceder los ejercicios a partir de la previa selección del capítulo, tema o subtema al cual estos están asociados. Dichos ejercicios podrán ser de diferentes tipologías, dentro de las que se encuentran selección simple, selección múltiple, seleccionar texto, completar por escritura, completar por desplazamiento, ordenar palabra, ordenar medias, enlazar elementos de dos conjuntos, clasificar palabra y clasificar medias. Además los ejercicios podrán tener asociados imágenes de apoyo y retroalimentación.

La sección de enfrenta retos estará dividida a su vez en tres secciones; estas serán valorar tus posibilidades, aprende a resolver e investiga y aprende. En cada una de ellas el estudiante podrá encontrar materiales para su autoevaluación, dentro de los que resaltan cuestionarios, pruebas de ingreso y guías de ejercicios resueltos.

2.4 Requisitos del software.

La identificación de los requisitos del software permite interpretar las condiciones o capacidades que debe cumplir el sistema para que se ajuste a las necesidades del cliente. Los requisitos de un software son clasificados en dos grupos, Requisitos Funcionales (RF) y Requisitos no Funcionales (RNF).

2.4.1 Requisitos funcionales (RF).

Los requisitos funcionales permiten identificar las condiciones que debe cumplir un sistema para satisfacer un contrato, estándar, especificación u otra documentación formalmente impuesta.

RF 1: Gestionar asignación.

RF 1.1 Insertar los datos de una asignación. Para ello se debe seleccionar los ejercicios que formarán parte de la asignación, seleccionar los estudiantes a los cuales les serán asignados los ejercicios e introducir un conjunto de datos informativos como son; nombre de la asignación, fechas de inicio y fin, así como una descripción de la misma. Se selecciona la opción incluir.

RF 1.2 Ver los datos de una asignación. Para ello debe existir al menos una asignación creada. Se selecciona la asignación de la cual se desea ver sus datos y estos se mostrarán.

RF 1.3 Modificar los datos de una asignación. Para ello debe existir al menos una asignación creada. Selecciona la asignación de la cual se desea modificar sus datos. Se selecciona la opción modificar.

RF 1.4 Eliminar los datos de un asignación. Para ello debe existir al menos una asignación creada. Selecciona la asignación de la cual se desea eliminar sus datos. Se selecciona la opción eliminar.

RF 2: Consultar asignación.

RF 2.1 Mostrar listado de asignaciones. Para ello debe existir al menos una asignación creada.

RF 2.2 Filtrar asignaciones según un criterio especificado. Para ello debe existir al menos una asignación creada. Se seleccionan los criterios por los que se desea filtrar. Se selecciona la opción mostrar.

RF 3: Consultar ejercicios realizados.

RF 3.1 Mostrar listado de ejercicios realizados. Debe existir al menos un ejercicio realizado.

RF 3.2 Filtrar ejercicios realizados según un criterio especificado. Debe existir al menos un ejercicio realizado. Se seleccionan los criterios por los que se desea filtrar. Se selecciona la opción mostrar.

RF 4: Realizar ejercicios de entrenamiento. Se selecciona el capítulo, tema o subtema al cual esté asociado el o los ejercicios que se deseen realizar. Se pueden filtrar estos ejercicios por la complejidad. Se selecciona la opción comenzar.

RF 5: Realizar ejercicios de evaluación. Debe existir al menos una asignación creada. Se selecciona la asignación que se desee realizar. Selecciona la opción comenzar.

RF 6: Mostrar efectividad. Se selecciona la realización de un ejercicio. Se selecciona la opción de terminar un ejercicio. Se selecciona la opción evaluar y terminar, y se muestra una ventana que contiene un análisis de los resultados obtenidos.

2.4.2 Requisitos no funcionales (RNF)

Los requisitos no funcionales permiten definir las propiedades que el producto debe tener. Estas características son las que hacen al producto más atractivo, usable, rápido y confiable.

Una vez que se conozca lo que el componente debe hacer será posible determinar cómo ha de comportarse, qué cualidades debe tener, cuán rápido y grande debe ser el mismo. Dichas propiedades

son importantes para los clientes y usuarios a la hora de valorar cuán aceptable está el producto con respecto a usabilidad, seguridad, y cumplimiento de las funcionalidades requeridas.

Seguridad

RNF 1: Base de datos de usuarios y calificaciones: ofuscada.

RNF 2: Base de datos de elementos multimedia y contenidos: encriptado.

RNF 3: Comunicación web de transferencia de datos: encriptados.

Restricciones de diseño e implementación

RNF 4: El lenguaje de programación a emplear debe ser PHP 5.3.3 o superior.

RNF 5: El marco de trabajo base de desarrollo que se utilizará es: Symfony 1.4.6.

RNF 6: Como IDE se empleará NetBeans 6.9.

RNF 7: Como servidor Web se explotará Apache 2.2.16.

RNF 8: El SGDB deberá ser PostgreSQL 8.4.

Apariencia e interfaz externa

RNF 9: Interfaz amigable para usuarios finales. Su funcionamiento deberá ser intuitivo y requerir de información mínima. Claridad y buena organización de la información, que permita la interpretación correcta e inequívoca de la misma. Ejecución de acciones de una manera rápida y minimizar los pasos a dar en cada proceso.

Interfaces de hardware

Los servidores deberán cubrir las siguientes características o contar con una variante equivalente a:

RNF 10: Servidores centrales. Servidores web con procesador: Intel Xeon de 4 núcleos. La cantidad de núcleos depende de la cantidad de usuarios conectados. (Ejemplo: 80,000 usuarios – 7 núcleos). RAM: 8 GB o más, con posibilidades de expansión en caso de ser necesario. Servidor de Base de Datos con Procesador: Intel Xeon de 4 núcleos. La cantidad de núcleos depende de la cantidad de usuarios conectados. (Ejemplo: 80,000 usuarios – 7 núcleos). RAM: 8 GB o más, con posibilidades de expansión

en caso de ser necesario. Servidor de Almacenamiento con 300 GB, inicialmente para el almacenamiento de todas las medias.

RNF 11: Servidores locales. En las escuelas en dependencia de la cantidad de usuarios pudiese elegirse usar un solo servidor para la web, Base de Datos y Almacenamiento, o de forma separada en dos: 1 para la web y otro para la BD.

Pueden ser servidores normales, o máquinas mejoradas en caso de escuelas pequeñas de menos de 200 usuarios:

- ✓ 1 Procesador CoreDuo.
- ✓ 2GB de RAM.
- ✓ 120 GB de HDD mínimo, aunque puede ser configurable de acuerdo a la cantidad de contenidos que se vayan contratando.
- ✓ Banda ancha.

Equipos de usuario final

Los usuarios finales deberán contar como mínimo con:

- ✓ Procesador Pentium II o superior.
- ✓ 1GB de RAM.
- ✓ 20GB de HDD.
- ✓ Bocinas.
- ✓ Si no cuentan con servidor local: conexión de banda ancha 256 kbps como mínimo.
- ✓ Si cuenta con servidor local: acceso a red interna.

Soporte

RNF 12: Ser generado en tecnología Web para ser accesible a través de Internet.

RNF 13: Correr sobre cualquier navegador, siendo como mínimo que sea compatible con:

- ✓ Explorer 7.0 y superior.
- ✓ Mozilla Firefox.
- ✓ Opera.
- ✓ Chrome.

- ✓ Safari.

Usabilidad

RNF 14: Los elementos gráficos como los iconos deberán contar con un tooltip o mensaje flotante que señalen el tipo de recurso al que se refiere.

Eficiencia

RNF 15: Cuando un usuario, sin importar su rol, permanece inactivo durante 20 minutos, se cierra la sesión automáticamente.

Legales, Derecho de Autor y otros

RNF 16: Una vez terminada la plataforma debe ser sometida a una evaluación y certificación por parte del cliente del producto.

2.5 Modelo de casos de uso del sistema.**2.5.1 Descripción de los actores del sistema.**

Actor	Descripción
Estudiante	Accede a los materiales que están desarrollados para los procesos de aprendizaje, así como a las situaciones y actividades pedagógicas que se han generado para evaluar su nivel de comprensión y su rendimiento académico.
Docente	Asigna actividades al grupo o individualmente, hace un seguimiento a las actividades de todos los estudiantes que estén a su cargo.
Tutor	Es la persona que representa legalmente a alguien, en este caso, es la persona que representa legalmente a un estudiante. Un tutor puede tutorar a varios estudiantes y debe tener permisos en el software de visualizar la actividad del estudiante que esté a su cargo y compararla a través de reportes gráficos con el promedio y actividades del resto del curso.
Director	Accede a los materiales que se han generado para evaluar el nivel de comprensión y el rendimiento académico del estudiante.
Administrador Local	Realiza las tareas de administración en la plataforma instalada en la escuela.

Usuario del Hiperentorno	Usuario que representa la generalización de los actores estudiante, docente, tutor, administrador local y director.
---------------------------------	---

Tabla 2.1: Descripción de los actores del sistema.

2.5.2 Consideraciones previas a los diagramas del análisis.

Existen tres tipos de relaciones que priman en los diagramas de casos de uso del análisis. Para una mejor comprensión de dichos diagramas a continuación se presenta una breve explicación de cada una de estas relaciones.

Relación de inclusión: Una relación *include* es una relación desde un caso de uso base a un caso de uso de inclusión, que especifica cómo el comportamiento definido para el caso de uso de inclusión se inserta explícitamente dentro del comportamiento definido para el caso de uso base.

Generalización/Especialización entre casos de uso: Un caso de uso generalización es una relación de un caso de uso hijo a un caso de uso padre que especifica cómo el hijo puede especializar todo el comportamiento y características descritas para el padre.

Generalización/Especialización entre actores: Una relación de generalización de una clase hija de actor a otra clase padre de actor indica que el hijo hereda el rol que la clase padre puede jugar respecto a un caso de uso.

2.5.3 Patrones de casos de uso.

La experiencia en la utilización de casos de uso ha evolucionado en un conjunto de patrones que permiten con más precisión reflejar los requisitos reales, haciendo más fácil el trabajo con los sistemas, y mucho más simple su mantenimiento. Dado un contexto y un problema a resolver, estas técnicas han mostrado ser la solución adoptada en la comunidad del desarrollo de software.

A continuación se explican los patrones presentes en el diagrama de casos de uso del sistema.

Concordancia

Extrae una subsecuencia de acciones que aparecen en diferentes lugares del flujo de casos de uso y es expresado por separado.

Reusabilidad: Consta de tres casos de uso. El primero llamado subsecuencia común, modela una secuencia de acciones que aparecerán en múltiples casos de uso en el modelo. Los otros casos de uso modelan el uso del sistema que comparte la subsecuencia común de acciones. De manera que deben existir al menos dos de ellos.

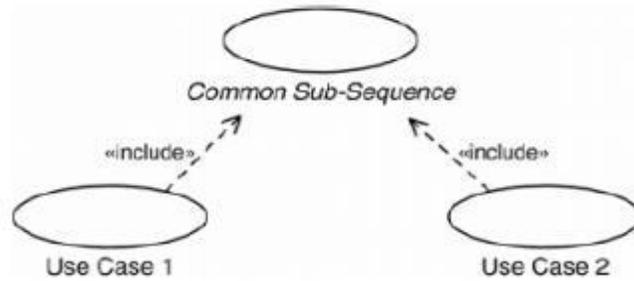


Figura 2.2 Patrón Concordancia – Reusabilidad.

Especialización: Otro patrón de concordancia que contiene casos de uso del mismo tipo. En este caso, estos son modelados como una especialización de casos de uso de tipo de uso común. Todas las acciones en estos casos de uso son heredadas por los casos de uso hijos, donde otras acciones serán adicionadas o acciones heredadas que serán especializadas. Este patrón es aplicable cuando la utilización de los casos de uso que han sido modelados son del mismo tipo, y este tipo debe hacerse visible en el modelo.

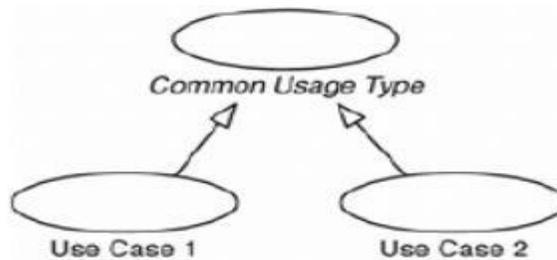


Figura 2.4 Patrón Concordancia – Especialización.

CRUD (Crear, Leer, Modificar, Eliminar)

Este patrón se basa en la fusión de casos de uso simples para formar una unidad conceptual.

Completo: Este patrón consta de un caso de uso, llamado Información CRUD o gestionar información, modela todas las operaciones que pueden ser realizadas sobre una parte de la información de un tipo específico, tales como creación, lectura, actualización y eliminación. Suele ser utilizado cuando todos los flujos contribuyen al mismo valor del negocio, y estos a su vez son cortos y simples.

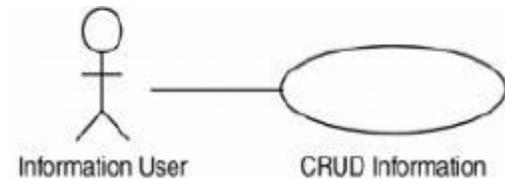


Figura 2.5 Patrón CRUD – Completo.

Múltiples actores

Roles comunes: Puede suceder que los dos actores jueguen el mismo rol sobre el CU. Este rol es representado por otro actor, heredado por los actores que comparten este rol. Es aplicable cuando, desde el punto de vista del caso de uso, solo exista una entidad externa interactuando con cada una de las instancias del caso de uso.

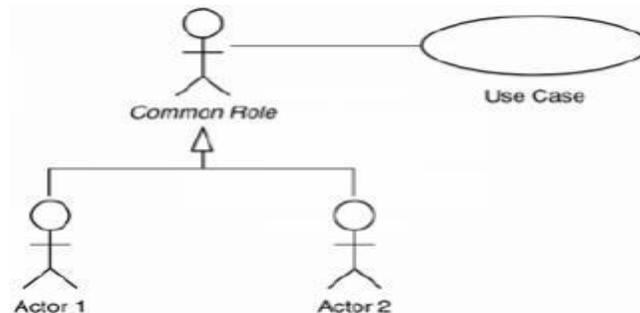


Figura 2.6 Patrón Múltiples actores – Roles comunes.

2.5.4 Diagrama de casos de uso del sistema.

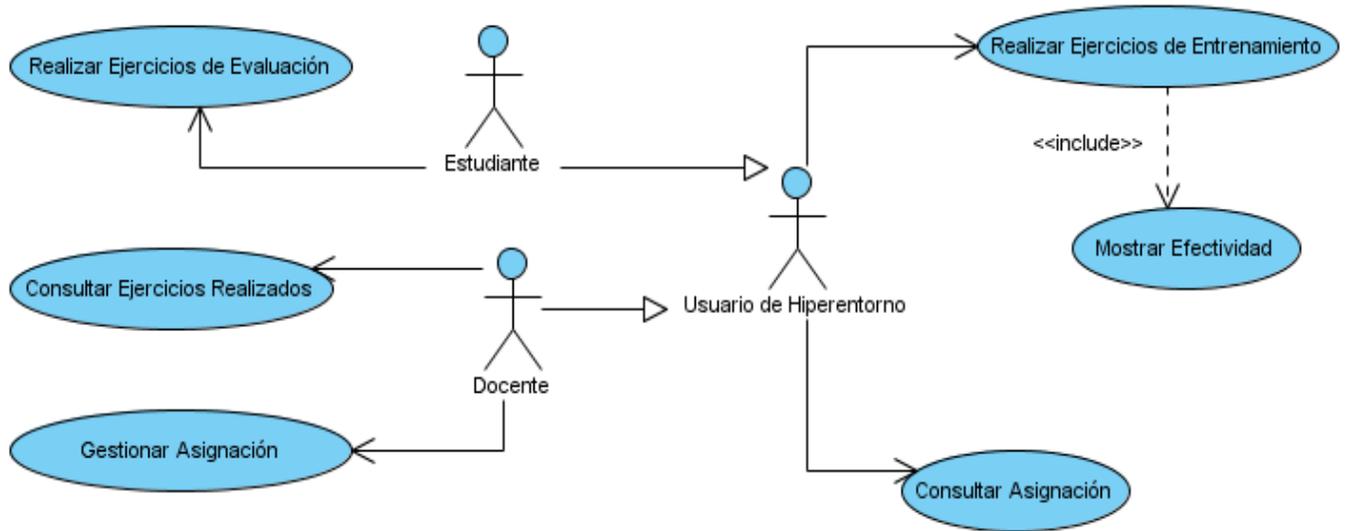


Figura 2.3 Diagrama de caso de uso del sistema.

2.5.5 Descripción de los casos de uso del sistema.

Descripción General del CU Consultar Ejercicios Realizados
<p>Objetivo</p> <p>Buscar y listar un resumen de los datos de los ejercicios realizados coincidentes con uno o varios criterios de búsqueda.</p>
<p>Actores</p> <p>Usuario de Hiperentorno (Inicia): Consultar un listado de ejercicios realizados.</p>
<p>Resumen</p> <p>El caso de uso se inicia cuando el actor accede a la opción que le permite realizar una acción sobre los ejercicios realizados, ya sea los ejercicios realizados de un estudiante en específico o de un grupo en general. Brinda la posibilidad de introducir los datos elementales de búsqueda y permite realizar una búsqueda a partir de los datos mostrados, listar todos los elementos y cancelar la operación en cualquier momento. Se muestra un resumen de los datos de ejercicios realizados. Se puede consultar el resumen de los datos de los elementos de la lista de coincidencias y así el caso de uso termina.</p>

Precondiciones

Debe haberse generado el escritorio de trabajo del usuario autenticado.

Poscondiciones

Se consultó un resumen con los datos de los ejercicios realizados coincidentes con uno o varios criterios de búsqueda.

Se consultó los ejercicios realizados.

Flujo de eventos

Flujo básico

Acciones del actor	Respuesta del sistema
1. El caso de uso se inicia cuando el actor accede a la opción de consultar ejercicios realizados registrado como profesor.	2. Brinda la posibilidad de filtrar la búsqueda por: <ul style="list-style-type: none"> • Grupo. • Complejidad. • Tipología. • Estudiante.
3. Selecciona el índice al cual se encuentran asociados los ejercicios.	
4. Introduce los datos que considere para realizar una búsqueda y selecciona el estudiante específico en el listado.	5. <i>Consulta los datos y muestra una lista de posibles coincidencias.</i>
	6. Muestra un resumen de los datos de los ejercicios realizados: <ul style="list-style-type: none"> • Nombre del ejercicio. • Entradas al ejercicio. • Evaluación obtenida en la realización del ejercicio. Y permite: <ul style="list-style-type: none"> • Ver la entrada a un elemento de la lista mostrada. • Realizar una nueva búsqueda.
7. Consulta el resumen de los datos de los elementos de la lista de coincidencias.	
8. Selecciona el número de la entrada que	

desea consultar y elige la opción Ver entrada.

9. Muestra una ventana con el ejercicio y la solución dada por el usuario. Y además permite:
 - Navegar por los intentos.
 - Ver la evaluación obtenida en cada intento.

2. Selecciona la opción de navegar por los intentos del ejercicio.

3. Muestra los ejercicios y las soluciones dadas en cada intento.
4. El caso de uso termina.

Flujo alternativo

3.a El actor no selecciona el índice asociado.

Acciones del actor

Respuesta del sistema

3.a.1 Muestra un mensaje de información.

3.a.2 Regresa al paso 3 del Flujo Básico.

4.a El actor no introdujo ningún dato de búsqueda.

Acciones del actor

Respuesta del sistema

4. a.1 Muestra un mensaje de información.

4.a.2 Regresa al paso 3 del Flujo Básico.

4.b El sistema no encuentra ninguna coincidencia.

Acciones del actor

Respuesta del sistema

4. b.1 Muestra un mensaje de información y permite realizar una nueva búsqueda.

4.b.2 Regresa al paso 3 del Flujo Básico.

8.a El actor selecciona la opción de Ver Entrada y solo hay una entrada.

Acciones del actor

Respuesta del sistema

8.a.1 Deshabilita el combo box correspondiente a las entradas.

8.a.2 Accede a la única entrada existente.

8.b El actor selecciona la opción de realizar una nueva búsqueda.

Acciones del actor

Respuesta del sistema

8.b.1 Regresa al Paso 2 del Flujo Básico.

10.a El actor selecciona la opción Siguiente.

Acciones del actor

Respuesta del sistema

10. a.1 Muestra la solución dada por el usuario

	en el intento siguiente.
	10. a.2 El caso de uso termina.
10.b El actor selecciona la opción Anterior.	
Acciones del actor	Respuesta del sistema
	10. b.1 Muestra la solución dada por el usuario en el intento anterior.
	10. b.2 El caso de uso termina.

El resto de las descripciones de los casos de uso del sistema se encuentran en el Anexo # 1.

2.6 Conclusiones parciales.

En este capítulo se reflejaron los principales conceptos del negocio, asociados al modelo de dominio. Quedaron además definidos los requisitos tanto funcionales como no funcionales que sustentan la propuesta de solución planteada. También se identificaron y describieron los casos de uso, así como los actores que representan las características y necesidades que requiere el componente.

CAPÍTULO III

Análisis y Diseño del Sistema.

3.1 Introducción.

En este capítulo se realiza el análisis y diseño que sustenta la propuesta de solución del sistema a desarrollar. Se modelan además los diagramas de clases del análisis y los del diseño. Se define el patrón arquitectónico, y los patrones de diseño que regirán el proceso. Para garantizar el almacenamiento de la información se modela la estructura de la base de datos.

3.2 Modelo de análisis.

El objetivo de este flujo de trabajo es traducir los requisitos a una especificación que describe cómo implementar el sistema. El análisis consiste en obtener una visión del sistema que se preocupa de ver qué hace, de modo que sólo se interesa por los requisitos funcionales.

A continuación se explican los estereotipos presentes en los diagramas de clases de análisis.

Clase de interfaz: Se utiliza para modificar la interacción entre el sistema y sus actores, lo que implica recibir y representar informaciones y peticiones de usuarios y de sistemas externos. Representa la abstracción de ventanas, formularios, paneles, interfaz de comunicación.

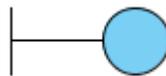


Figura 3.1 Estereotipo que representa la clase de interfaz.

Clase control: Representa coordinación secuencial, transacciones y control de otros objetos. Se usa para encapsular el control de un caso de uso en concreto. También se usa para representar derivaciones y cálculos complejos.



Figura 3.2 Estereotipo que representa la clase control.

Clase de entidad: Se usa para modelar información que posee larga vida, y que es a menudo persistente. Modela la información y comportamiento asociado de algún fenómeno o concepto (persona, objeto del mundo real o suceso). Se derivan de una clase de entidad del negocio o del dominio.

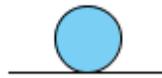
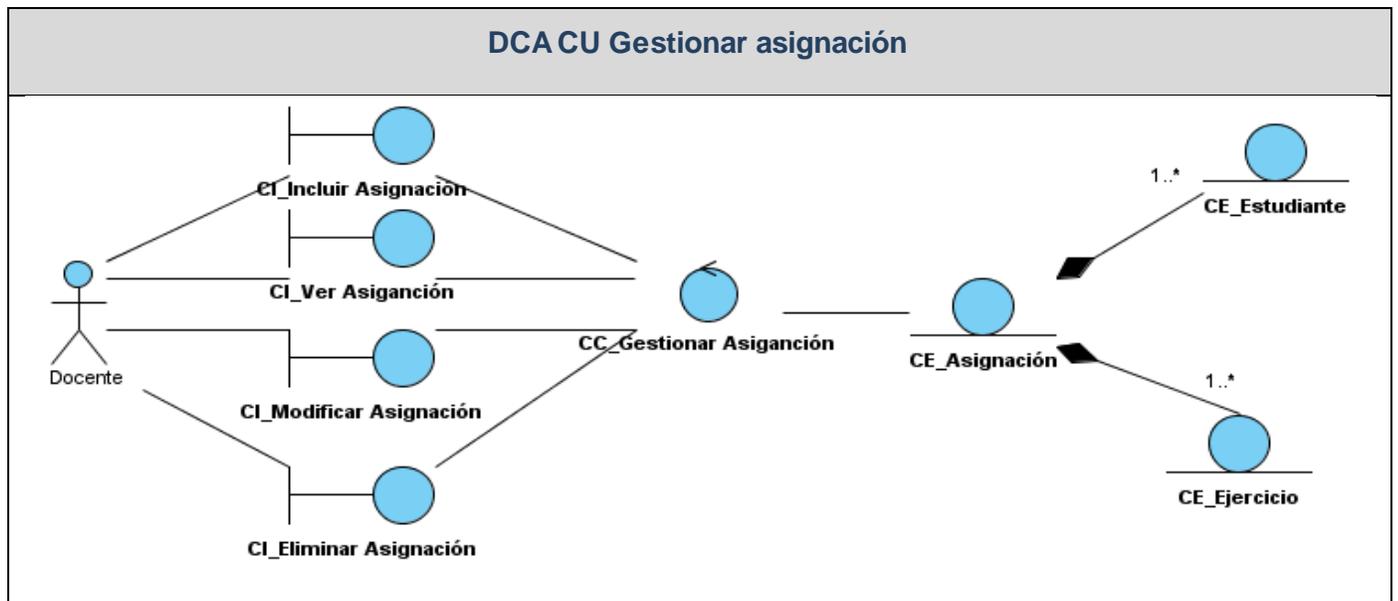


Figura 3.3 Estereotipo que representa la clase de entidad.

3.2.1 Diagramas de clases del análisis.

A continuación se muestran los diagramas de clase del análisis (DCA) para los casos de uso del sistema más representativos. El resto se encuentra recogido en el Anexo # 1.



3.3 Patrón arquitectónico Modelo – Vista – Controlador en symfony.

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

Modelo: Administra y maneja todo lo relacionado con los datos del sistema, da respuesta a peticiones de información sobre el estado de la aplicación (normalmente desde la vista), y responde con instrucciones de cambio de estado (usualmente desde el controlador) a la vista.

Vista: Gestiona lo relacionado con mostrar la información al usuario.

Controlador: El controlador interpreta los eventos que son lanzados por la entrada estándar del usuario (normalmente mouse y teclado), informando de los mismos al modelo y/o la vista para que se ejecuten los cambios apropiadamente.

A continuación se muestra una representación gráfica del MVC

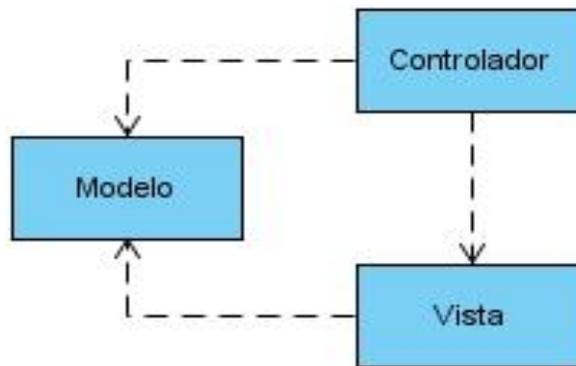


Figura 3.4 Estructura del patrón arquitectónico Modelo-Vista-Controlador (MVC).

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. Si por ejemplo una misma aplicación debe ejecutarse tanto en un navegador estándar como un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email, etc.). El modelo se encarga de la abstracción de la lógica

relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación. (23)

3.4 Patrones de diseño utilizados.

El framework Symfony utiliza en su implementación una serie de patrones de diseño, los cuales clasifican y describen formas de solucionar problemas frecuentes en el desarrollo.

Observer (Observador): Este patrón permite definir una dependencia entre objetos de forma que cuando un objeto cambia de estado, todos sus objetos dependientes son notificados y actualizados. Es utilizado para guardar las respuestas dadas por el usuario antes de que expire la sección.

Singleton: El contexto de este patrón es cuando se diseña una aplicación donde se tiene una clase, de la cual sólo puede existir una instancia. Es utilizado para tener una sola instancia de la modalidad en la que se pueden presentar los ejercicios.

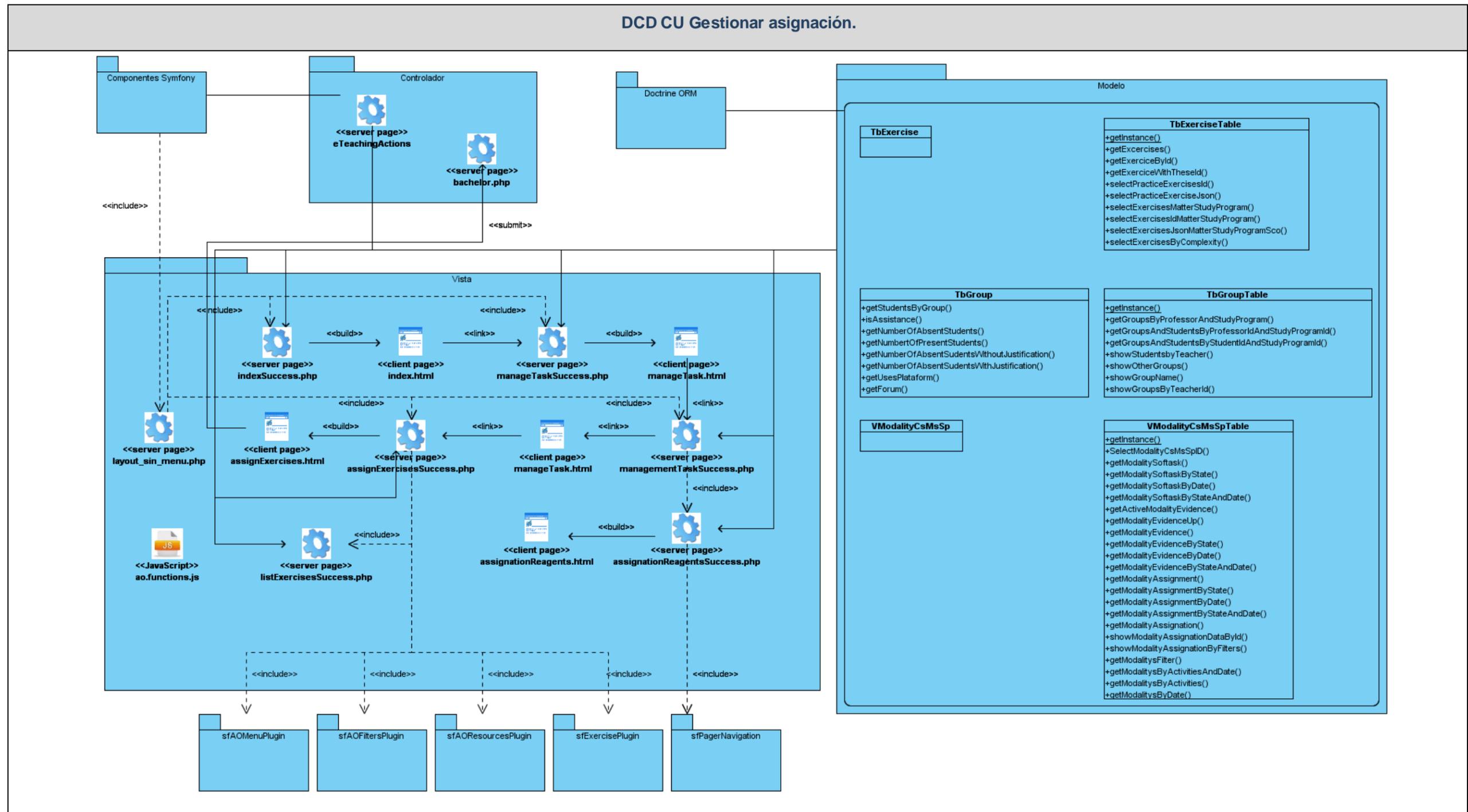
Strategy (Estrategia): El objetivo de este patrón es definir un grupo de clases que representan un conjunto de posibles comportamientos. Estos comportamientos pueden ser fácilmente intercambiados en una aplicación, modificando la funcionalidad en cualquier instante. Es utilizado para cargar el componente que se deriva de una tipología de ejercicios específica.

Decorator (Decorador): El objetivo de este patrón es añadir dinámicamente funcionalidad a un objeto. Es utilizado para manejar la modalidad en la que se pueden presentar los ejercicios, esto permite no tener que crear sucesivas clases que hereden de la primera incorporando la nueva funcionalidad, sino otras que la implementan y se asocian a la primera.

3.5 Modelo de diseño.

A continuación aparecen los diagramas de clase del diseño (DCD) para los casos de uso del sistema más representativos, el resto puede encontrarse en el Anexo # 2.

3.5.1 Diagrama de clases del diseño.



3.6 Diseño de la base de datos.

A continuación se presenta el modelo entidad relación referente a la base de datos. La descripción de las tablas que conforman dicho modelo se encuentran en el Anexo # 7.

3.6.1 Modelo entidad relación.

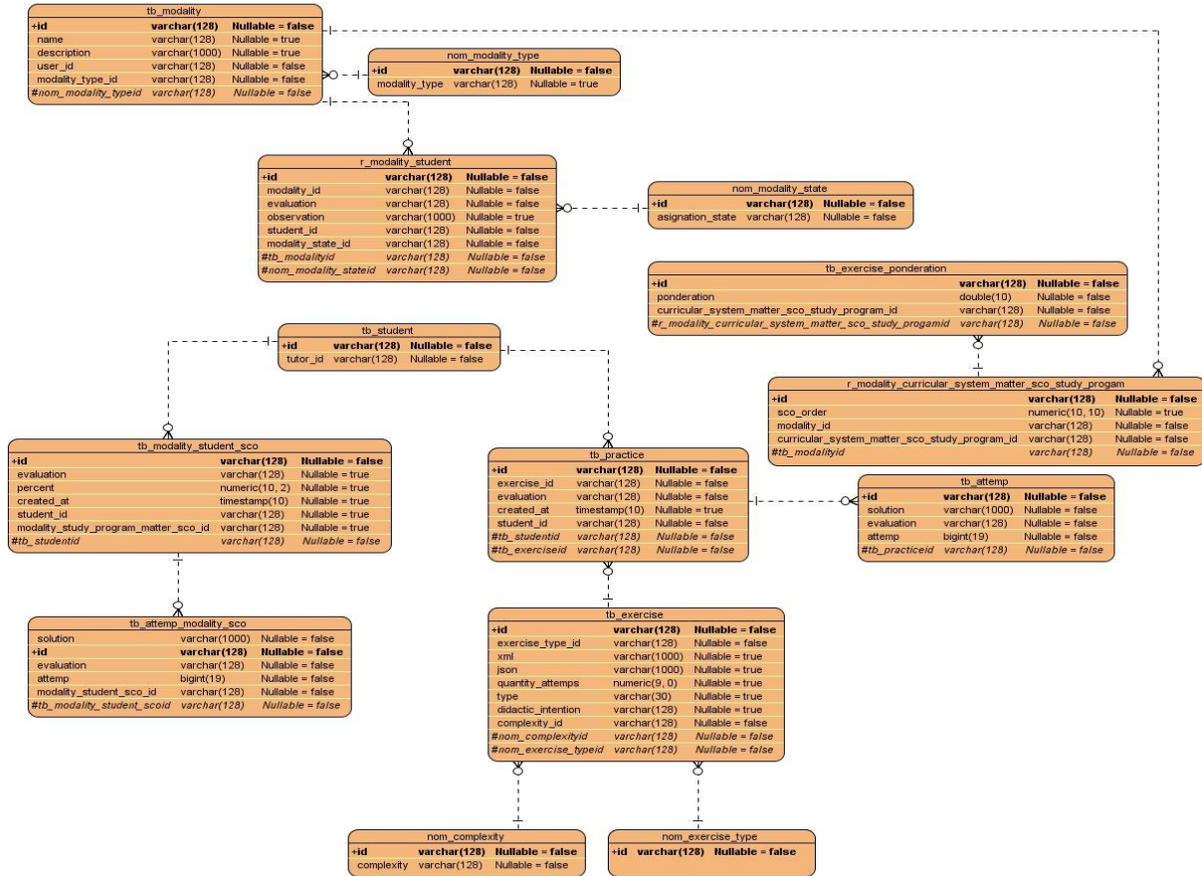


Figura 3.5 Modelo entidad – relación.

3.7 Conclusiones parciales.

En este capítulo se modelaron los diagramas de clases del análisis y del diseño enfocados al cumplimiento de los requerimientos funcionales del componente. Se valoró la aplicación del patrón arquitectónico, y de los patrones de diseño con vista a las características del componente. También se realizó el diseño de la base de datos.

CAPÍTULO IV

Implementación y Prueba.

4.1 Introducción.

En este capítulo se documenta el proceso de implementación de los elementos identificados durante la realización del diseño. Para ello se modela el diagrama de despliegue, el diagrama de componentes. Además se incluyen los resultados de las pruebas realizadas al componente.

4.2 Módulo de Implementación.

En la implementación se parte del resultado del diseño y se implementa el sistema en término de componentes, el propósito principal de la implementación es desarrollar la arquitectura y el sistema como un todo. El modelo de implementación representa la composición física de la implementación en término de subsistemas de implementación, y elementos de implementación (directorios y archivos incluyendo código fuente, datos y archivos ejecutables). Además, define las principales unidades de integración alrededor de las cuales se organizan los equipos, así como las unidades que se pueden versionar, desplegar y reemplazar separadamente.

4.2.1 Diagrama de despliegue.

El modelo de despliegue define la arquitectura física del sistema. Se usa para modelar de manera detallada los nodos físicos y las asociaciones de comunicación que existen entre ellos. Del mismo modo queda especificado qué hardware, sistemas operativos, software de interfaces y soporte conformarán el nuevo sistema.

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

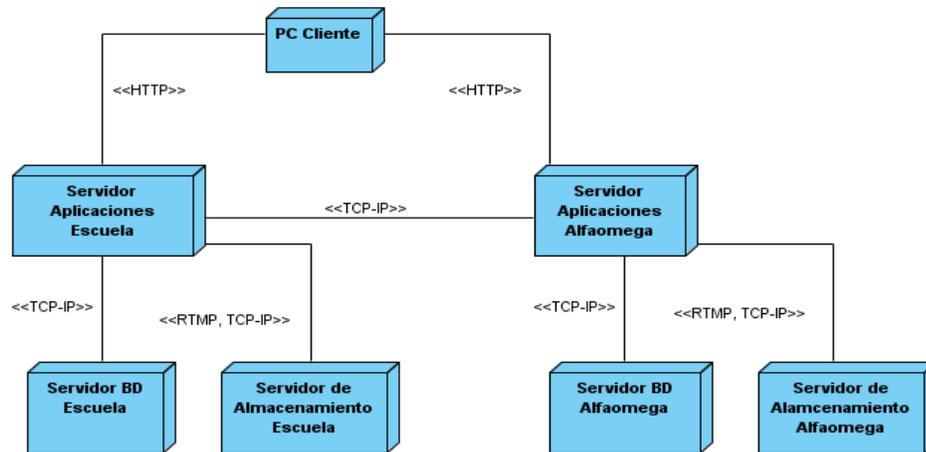


Figura 4.1 Diagrama de despliegue.

4.2.2 Diagrama de componentes.

Los diagramas de componentes son usados para estructurar el modelo de implementación en término de Subsistemas de implementación y mostrar las relaciones entre los elementos de implementación.

El uso más importante de estos diagramas es mostrar la estructura de alto nivel del modelo de implementación, especificando:

- ✓ Muestra las organizaciones y las dependencias entre tipos de componentes.
- ✓ Organizar los subsistemas de implementación en capas.

A continuación se presenta el diagrama de componentes propuesto para el CU Realizar ejercicios de entrenamiento, el resto se encuentra disponible en el Anexo # 5:

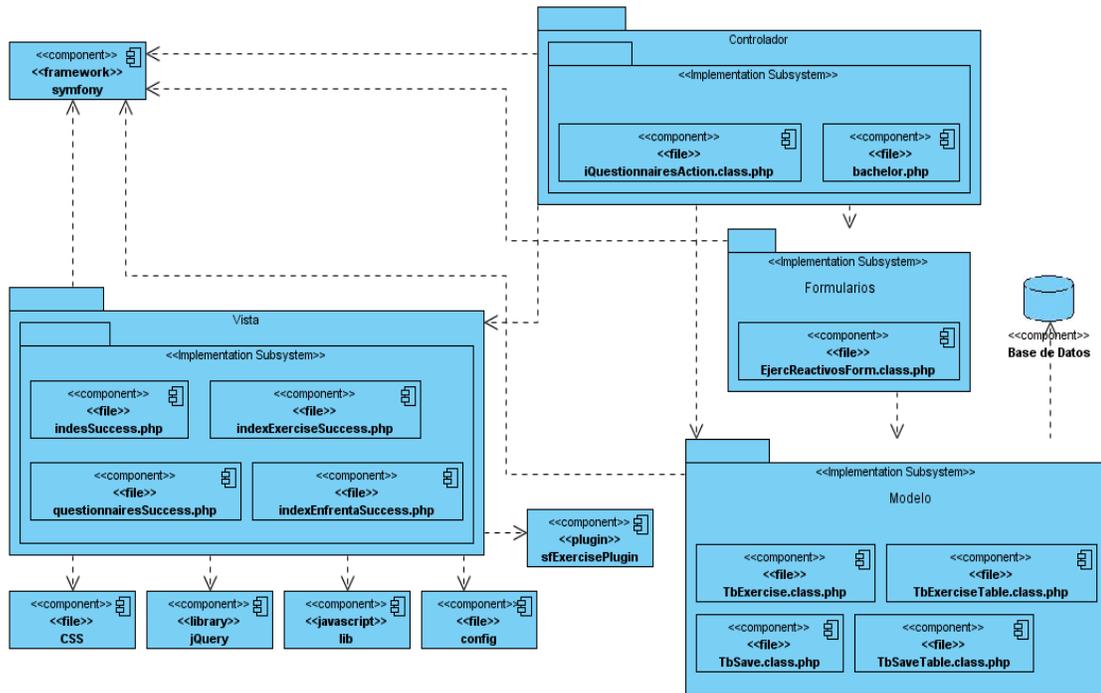


Figura 4.2 Diagrama de componentes.

4.3 Pruebas de software.

El desarrollo del software implica una serie de actividades de producción en las que las posibilidades de que aparezca la falibilidad humana son comunes. Debido a la imposibilidad humana de trabajar y comunicarse de forma perfecta, el desarrollo del software ha de ir acompañado de una actividad que garantice la calidad.

“Las pruebas son actividades en las cuales un sistema o componente es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente.” (28)

Las pruebas de software son realizadas con el objetivo de encontrar errores, verificando de esta manera la calidad del producto.

4.3.1 Niveles de prueba.

Las pruebas son aplicadas para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo. Se distinguen los siguientes niveles de prueba empleados para la identificación de los errores.

Pruebas Unitarias: Comienzan con la prueba de cada módulo. Una prueba unitaria es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de estos funcione correctamente por separado. El objetivo de las pruebas unitarias es aislar cada parte del programa y mostrar que las partes individuales son correctas. Proporcionan un contrato escrito que el fragmento de código debe satisfacer. Estas pruebas aisladas proporcionan cinco ventajas básicas; fomentan el cambio, simplifican la integración, documentan el código, separan la interfaz del código y hacen que los errores estén más acotados y sean fáciles de localizar. (13)

Prueba de integración: Es ejecutada para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso. Se prueba un paquete o un conjunto de paquetes del modelo de implementación. Estas pruebas descubren errores o incompletitud en las especificaciones de las interfaces de los paquetes. Esta prueba debe ser responsabilidad de desarrolladores y de independientes, sin solaparse las pruebas. (28)

Pruebas del Sistema: El software ensamblado totalmente con cualquier componente hardware que requiera, se prueba para comprobar que se cumplen los requisitos funcionales. Cualquier pieza de software completo, desarrollado o adquirido, puede verse como un sistema que debe probarse, ya sea para decidir acerca de su aceptación, para analizar defectos globales o para estudiar aspectos específicos de su comportamiento, tales como seguridad o rendimiento. Este tipo de pruebas estudia el producto completo. (13)

4.3.2 Métodos de prueba.

Las estrategias de pruebas del software se integran a las técnicas del diseño de casos de prueba, en una serie de pasos planificados que dan como resultado una correcta construcción del software, con tal grado de confianza que se detectarán la mayor parte de los errores existentes en él (nunca se puede decir que el producto está libre de errores).

Pruebas de caja blanca.

Las pruebas de caja blanca comprueban los caminos lógicos del software proponiendo casos de pruebas que se ejerciten conjuntos específicos de condiciones y/o bucles. Se puede examinar el estado de programa en varios puntos para determinar si el estado real coinciden con el esperado o mencionado. Requieren del conocimiento de la estructura interna del programa y son derivadas a partir de las especificaciones internas de diseño o el código.

Pruebas de caja negra.

También conocidas como Pruebas de Comportamiento, estas pruebas se basan en la especificación del programa o componente a ser probado para elaborar los casos de prueba. El componente se ve como una “Caja Negra” cuyo comportamiento sólo puede ser determinado estudiando sus entradas y las salidas obtenidas a partir de ellas. Para seleccionar el conjunto de entradas y salidas sobre las que trabajar, hay que tener en cuenta que en todo programa existe un conjunto de entradas que causan un comportamiento erróneo en nuestro sistema, y como consecuencia producen una serie de salidas que revelan la presencia de defectos. (13)

4.3.2.1 Diseño de casos de prueba.

Para confeccionar los casos de prueba de caja negra existen distintos mecanismos. Algunos de ellos son: Particiones de Equivalencia, Análisis de Valores Límites y Guiada por Casos de Prueba. A continuación se presenta el CP Realizar Ejercicios de Entrenamiento, el resto puede encontrarse en el Anexo # 6.

CP Realizar ejercicios de entrenamiento

Descripción general

El caso de uso inicia cuando el actor selecciona la opción que le permite realizar los ejercicios. El actor puede seleccionar los ejercicios a realizar para su auto-evaluación. Una vez realizados los ejercicios estos serán evaluados y finalizará el caso de uso.

Condiciones de ejecución

Debe haberse generado el escritorio de trabajo del usuario autenticado.

Debe haberse seleccionado el índice.

El actor debe tener asignado los permisos necesarios para realizar ejercicios.

SC Realizar Ejercicios

ID del escenario	Escenario	Secuencial	Al azar	Ejercicio inicial	Ejercicio final	Cantidad	Muy Fácil	Fácil	Mediana	Difícil	Muy Difícil	Resultados	Respuesta del sistema	Flujo central
EC 1	Selección a la opción de realizar Ejercicios												Brinda la posibilidad de seleccionar el tipo de ejercicio a realizar: <ul style="list-style-type: none"> Ejercicios. Enfrenta Retos. 	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios
EC 2	Selección a la opción de realizar Ejercicios. Selección a el índice asociado.												Brinda la posibilidad de seleccionar el tipo de Ejercicio a realizar, sean estos de tipo: <ul style="list-style-type: none"> Secuencial. Al Azar. También ofrece la posibilidad de: <ul style="list-style-type: none"> Filtrar los ejercicios seleccionados según la complejidad. Además brinda la opción de <ul style="list-style-type: none"> Continuar 	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selección un elemento en el índice/

CAPÍTULO IV
Implementación y prueba

															Y permite: <ul style="list-style-type: none"> • Seleccionar la opción "Comenzar". • Cancelar la operación en cualquier momento. 	
EC 3	Selección a realizar ejercicios de forma secuencial .	V													Brinda la posibilidad de introducir los siguientes datos: <ul style="list-style-type: none"> • Ejercicio Inicial. • Ejercicio Final. Y permite filtrar los ejercicios a realizar por: <ul style="list-style-type: none"> • Complejidad. • Seleccionar la opción "Resultados". 	Sección Mis Cursos/Selección un grupo/Pestaña Prácticas/Selección Ejercicios/Selección un elemento en el índice/Opción Secuencial
EC 4	Introduce los datos. Selección a la opción Comenzar .		V	V			V	V	V	V	V	V			Valida los datos. Muestra los ejercicios a realizar. Y permite: <ul style="list-style-type: none"> • Terminar Ejercicio. • Comprobar. • Ejercicio Anterior. • Ejercicio Siguiente. • Deshacer. 	Sección Mis Cursos/Selección un grupo/Pestaña Prácticas/Selección Ejercicios/Selección un elemento en el índice/Opción Secuencial/Introduce los datos/Botón Comenzar/
			/	V			/	V	V	N/A						
			V	/			V	V	V	N/A						
			V	V			/	V	V	N/A						
			V	V			V	/	V	N/A						
			V	V			V	V	/	N/A						
			N/A	N/A			N/A	N/A	N/A	N/A						

CAPÍTULO IV
Implementación y prueba

EC 5	Responde los ejercicios. Selecciona la opción de Comprobar.										Comprueba la respuesta del ejercicio. Guarda la evaluación obtenida por cada estudiante. Muestra la evaluación obtenida por cada estudiante. Muestra un mensaje. Si el ejercicio lo permite ofrece una salida temporal.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/Botón Comenzar/Resuelve ejercicios/Botón Comprobar/
EC 6	Confirma la salida temporal.										Muestra el contenido al cual se encuentran asociados los ejercicios. Guarda la selección de ejercicios.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/Botón Comenzar/Resuelve ejercicios/Botón Comprobar/Botón Aceptar/

CAPÍTULO IV
Implementación y prueba

EC 7	No selecciona el índice asociado.												Muestra un mensaje de información. Regresa a la vista anterior.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/
EC 8	Selección a la opción Al Azar.		V		V	V	V	V	V	V	V		Brinda la posibilidad de introducir la cantidad de ejercicios a realizar. Y permite filtrar los ejercicios a realizar por: <ul style="list-style-type: none"> • Complejidad • Seleccionar la opción "Resultados". 	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Al Azar/
						I	V	V	V	V	N/A			
						V	I	V	V	V	N/A			
						V	V	I	V	V	N/A			
						V	V	V	I	V	N/A			
						V	V	V	V	I	N/A			
						N/A	N/A	N/A	N/A	N/A	N/A			
EC 9	Introduce los datos.												Regresa al EC 4.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/
EC 10	Selección a la												Brinda la posibilidad de seleccionar el	Sección Mis Cursos/Sele

CAPÍTULO IV
Implementación y prueba

	opción Continuar.												ejercicio previamente guardado que desea continuar. Regresa al EC 4	ccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/Botón Continuar/
EC 11	Selección a la opción Resultados.											V	Muestra los ejercicios a realizar. Y permite: <ul style="list-style-type: none"> • Terminar Ejercicio. • Comprobar. • Ejercicio Anterior. • Ejercicio Siguiente. • Deshacer. • Respuesta 	Sección Mis Cursos/Selección un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/Opción Resultados/
EC 12	Selección a la opción Respuesta.												Muestra la respuesta del ejercicio. Muestra la evaluación obtenida.	Sección Mis Cursos/Selección un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en

CAPÍTULO IV
Implementación y prueba

													el índice/Opción Secuencial/Introduce los datos/Botón Comenzar/Resuelve ejercicios/Botón Respuesta/	
EC 13	Existen datos incorrectos.	V	N/A	I	V	N/A							Sobre cada campo incorrecto muestra el mensaje de error. Muestra un indicador sobre los campos incorrectos.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/
				V	I									
				N/A	N/A									
		/NA	V			I								
EC 14	Existen datos incompletos.	V	N/A	I	N/A	N/A							Muestra el mensaje de información Muestra un indicador sobre los campos incompletos.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/
				NA	I									
				NA	NA									
		N/A	V											

CAPÍTULO IV
Implementación y prueba

EC 15	La cantidad de ejercicios no coincide con la real.	V	N/A	/	N/A								Muestra el mensaje de información Muestra un indicador sobre los campos incorrectos. Regresa a EC 2.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/
		NA	V			/								
EC 16	Selección a la opción Terminar Ejercicio.												Muestra un mensaje de información asociado a la acción de finalizar el ejercicio.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Ejercicios/Selecciona un elemento en el índice/Opción Secuencial/Introduce los datos/Botón Comenzar/Resuelve ejercicios/Botón Terminar/
EC 17	Confirma la salida del Ejercicio.												Guarda la evaluación obtenida por el estudiante.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Se

CAPÍTULO IV
Implementación y prueba

		<ul style="list-style-type: none"> Investiga y Aprende. 	
EC 2	Selecciona la opción de realizar los ejercicios de la sección Valora tus posibilidades.	<p>Muestra los elementos disponibles que pueden ser:</p> <ul style="list-style-type: none"> Pruebas de ingreso a las universidades (por fecha de publicación). <p>Permite seleccionar la opción:</p> <ul style="list-style-type: none"> Atrás. 	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Enfrenta Retos/ Opción Valora tus posibilidades.
EC 3	Selecciona elemento a resolver.	<p>Muestra el fichero correspondiente en cada caso. Al poner el mouse encima de cada fichero se activa un tooltip con la siguiente información:</p> <ul style="list-style-type: none"> Título. Fecha de actualización. Tipo de fichero asociado. Tamaño. 	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Enfrenta Retos/ Opción Valora tus posibilidades/Selecciona el elemento a resolver/
EC 4	Selecciona la opción Aprende a Resolver.	<p>Muestra los elementos disponibles que pueden ser:</p> <ul style="list-style-type: none"> Guía de ejercicios resueltos. <p>Regresa al EC 3</p>	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Enfrenta Retos/ Opción Aprende a Resolver.
EC 5	Selecciona Investiga y Aprende.	<p>Muestra los elementos disponibles que son el tipo:</p> <ul style="list-style-type: none"> Investiga y Aprende. 	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Enfrenta Retos/ Opción Investiga y Aprende.
EC 6	Selecciona elemento a resolver.	<p>Muestra las orientaciones de los temas a realizar. Brinda la posibilidad de:</p> <ul style="list-style-type: none"> Buscar el archivo con la respuesta de la investigación. Guardar el archivo. 	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Enfrenta Retos/ Opción Investiga y Aprende/Selecciona elemento a resolver.
EC 7	Busca el archivo. Selecciona la opción Enviar Archivo.	<p>Valida los datos. Muestra un mensaje asociado a la validación de los datos del archivo.</p>	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Enfrenta Retos/ Opción Investiga y Aprende/Selecciona el archivo/Botón Subir/

CAPÍTULO IV
Implementación y prueba

EC 8	Existen datos incompletos.	Muestra el mensaje de error.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Enfrenta Retos/ Opción Investiga y Aprende/Selecciona el archivo/Botón Subir/
EC 9	Existen datos incorrectos.	Muestra el mensaje de error.	Sección Mis Cursos/Seleccionar un grupo/Pestaña Prácticas/Sección Enfrenta Retos/ Opción Investiga y Aprende/Selecciona el archivo/Botón Subir/

Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Secuencial	Campo de Selección.	No	
2	Al Azar	Campo de Selección.	No	
3	Ejercicio inicial	Campo de texto	No	Solo números
4	Ejercicio final	Campo de texto	No	Solo números
5	Cantidad	Campo de texto	No	Solo números
6	Muy Fácil	Campo de Selección.	Si	
7	Fácil	Campo de Selección.	Si	
8	Media	Campo de Selección.	Si	
9	Difícil	Campo de Selección.	Si	
10	Muy Difícil	Campo de Selección.	Si	
11	Resultados	Campo de Selección	Si	

4.4 Resultados obtenidos.

Durante el proceso de desarrollo del componente se llevaron a cabo pruebas unitarias para ir comprobando el funcionamiento del avance de la solución. Estas pruebas se enmarcan dentro del nivel de prueba de unidad, utilizando el método de caja blanca. Fueron realizadas por los desarrolladores aprovechando las ventajas de compilación paso a paso que brinda el IDE de desarrollo. Estas no se planificaron ni se registraron sus resultados, fueron haciéndose a medida que la solución se desarrollaba.

Para evaluar la solución desarrollada se concibieron varias iteraciones donde se estuvo probando el componente íntegramente. Se implicaron los métodos y niveles de prueba expuestos anteriormente en cada una de las iteraciones, en las cuales están implicadas prácticamente todas las personas del proyecto y el equipo de calidad, proyectando resultados visibles.

A continuación se especifican las principales pruebas realizadas a la aplicación, las tres primeras pertenecientes al nivel de prueba de sistema, en todos los casos haciendo uso del método de caja negra.

Pruebas Internas: Son realizadas por el equipo de calidad interna del proyecto con el fin de entregar un producto con la menor cantidad de errores posibles. Se centraron en el cumplimiento de las funcionalidades descritas en el listado de requerimientos y de casos de uso.

Pruebas Cruzadas: Pruebas realizadas al sistema por el resto de los equipos de desarrollo del proyecto. Se realizaron con el fin de encontrar la mayor cantidad de errores posibles en término de validaciones, pautas definidas por arquitectura de información, formato de los campos, entre otras.

Pruebas de Liberación: Pruebas realizadas por un tercero, en este caso Calisoft, institución encargada de validar que el software cuente con la calidad requerida para ser entregado a los clientes finales.

A continuación se muestran los siguientes resultados obtenidos en las pruebas de liberación.

Casos de prueba	No conformidades				
	Alta	Media	Baja	No proceden	Total
Gestionar asignación de ejercicios	-	-	-	-	0
Consultar asignación de ejercicios	-	-	-	-	0
Consultar ejercicios realizados	-	-	1	-	1
Realizar ejercicios de entrenamiento	-	-	1	2	3
Realizar ejercicios de evaluación	-	-	1	1	2
Mostrar efectividad	-	-	1	-	1

Tabla 4.1 Resultados de las pruebas de liberación.

4.5 Conclusiones parciales.

En este capítulo se presentaron los diagramas de implementación: el diagrama de despliegue que representa la distribución física por nodos de la aplicación y el diagrama de componentes que refleja la dependencia de los mismos con la base de datos. Se presentaron además los diseños de casos de prueba, así como los resultados obtenidos en las pruebas realizadas.

CONCLUSIONES.

Se puede concluir que la investigación cumplió los objetivos planteados al inicio de la misma. Además se puede decir que:

- ✓ Se obtuvo un componente completamente funcional para la evaluación del aprendizaje en la Plataforma Educativa ZERA.
- ✓ Se demostró lo beneficioso que resulta la utilización de una metodología tan sólida como RUP paralelamente a la utilización del framework Symfony, lo que propició la implementación efectiva del componente.
- ✓ Las herramientas seleccionadas soportaron el proceso de diseño e implementación de la solución.
- ✓ Las pruebas realizadas validan los artefactos sometidos a ellas y, por lo tanto, la calidad de dichos artefactos como parte de la solución.

RECOMENDACIONES.

A continuación se relacionan una serie de recomendaciones a tener en cuenta para futuros trabajos tomando como referencia el actual:

- ✓ Tener en cuenta este trabajo para proyectos futuros, siempre y cuando se cumpla con las normas de confidencialidad establecidas.
- ✓ Ampliar las funcionalidades del componente en versiones posteriores.
- ✓ Transmitir experiencias a otros proyectos que utilicen metodología, herramientas y tecnologías similares.
- ✓ Capacitar nuevos programadores para las próximas etapas de desarrollo de la plataforma, apoyándose en la experiencia de los actuales.

REFERENCIAS BIBLIOGRÁFICAS.

1. **Covadonga, A. A.** Entornos Formativos en el ciberespacio: las plataformas educativas. *Entornos Formativos en el ciberespacio: las plataformas educativas*. [En línea] 2004. [Citado el: 16 de octubre de 2010.] <http://web.ebscohost.com/ehost/detail?vid=1&hid=108&sid=f5e7f9e5-4daf-4953-958c-aaa579a65928%40sessionmgr112&bdata=JnNpdGU9ZW/hvc3QtbGl2ZQ%3d%3d#db=zbh&AN=25687867>
2. **Agudelo, M. M.** Plataformas educativas. *Plataformas educativas*. [En línea] 2006. [Citado el: 16 de octubre de 2010.] <http://aprendeonline.udea.edu.co/banco/html/plataformaseducativas/>.
3. Diccionario de la Real Academia Española Vigésima segunda edición. *Diccionario de la Real Academia Española Vigésima segunda edición*. [En línea] 2001. [Citado el: 17 de octubre de 2010.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=ejercicio.
4. **Mercenier, Philippe y Dekimpe, Philippe.** Claroline Lab. *Claroline Lab*. [En línea] Claroline. [Citado el: 14 de enero de 2011.] <http://lab.claroline.net/>.
5. Modular Object-Oriented Dynamic Learning Environment. *Modular Object-Oriented Dynamic Learning Environment*. [En línea] [Citado el: 17 de octubre de 2010.] <http://moodle.org>.
6. **Claroline.** Claroline. *Claroline*. [En línea] [Citado el: 18 de octubre de 2010.] <http://www.claroline.net/about-claroline.html>.
7. **WebCT.** WebCT. *WebCT*. [En línea] [Citado el: 18 de octubre de 2010.] <http://webct.uprm.edu/>.
8. **Campdesuñer, Dra. C. Ileana Moreno, Curbelo Cancio, MSc. Juan y Villar Vázquez, Ing. Gretchen.** *Aula Virtual para la enseñanza-aprendizaje de los*. Villa Clara : s.n.
9. **Peña, J. C. S.** *Plataforma de Teleformación aprenDIST*. 2005.
10. **Kruchten, Philippe.** *Rational Unified Process, An Introduction, Third Edition*. 2003.

Referencias bibliográficas

11. **Escribano, G. F.** Introducción a Extreme Programming. *Introducción a Extreme Programming*. [En línea] 2002. [Citado el: 23 de octubre de 2010.] <http://www.dsi.uclm.es/asignaturas/42551/trabajosAnteriores/Presentacion-XP.pdf>.
12. **Ministerio de Educación, España.** ITE. *ITE*. [En línea] Gobierno de España. [Citado el: 19 de septiembre de 2010.] <http://ares.cnice.mec.es/>.
13. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El lenguaje Unificado de Modelado. Manual de Referencia*. s.l. s.l. : Addison Wesley.
14. Sitio Web oficial del producto Visual-Paradigm. *Visual-Paradigm*. [En línea] [Citado el: 8 de diciembre de 2010.] <http://www.visual-paradigm.com/product/vpuml/>.
15. **Torrealday, Gustavo F.** Torrealday Infomática. *Torrealday Infomática*. [En línea] [Citado el: 10 de enero de 2011.] <http://www.torrealday.com.ar/articulos/articulo006.htm>.
16. Programación en castellano. *Programación en castellano*. [En línea] 1999-2008. [Citado el: 26 de octubre de 2010.] <http://www.programacion.net/php/>.
17. **Gutiérrez, Javier J.** [En línea] [Citado el: 17 de noviembre de 2010.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.
18. **BlueprintCSS, Sitio oficial de.** Sitio oficial de BlueprintCSS. *Sitio oficial de BlueprintCSS*. [En línea] 26 de octubre de 2009. [Citado el: 21 de enero de 2011.] <http://www.blueprintcss.org/>.
19. **Symfony.** Symfony.es. *Symfony.es*. [En línea] [Citado el: 2 de noviembre de 2010.] <http://www.symfony.es/que-es-symfony/>.
20. Eclipse. *Eclipse*. [En línea] Eclipse, 2008. [Citado el: 3 de diciembre de 2010.] <http://www.eclipse.org/>.
21. NetBeans. *NetBeans*. [En línea] NetBeans, 2008. [Citado el: 3 de diciembre de 2010.] <http://www.netbeans.org>.
22. **Pérez Mata, Manel.** TecnoRetales. *TecnoRetales*. [En línea] 3 de julio de 2009. [Citado el: 16 de enero de 2011.] <http://www.tecnoretalles.com/programacion/que-es-doctrine-orm/>.
23. **Potencier, F.** *Symfony la guía definitiva*. 2009.

Referencias bibliográficas

24. Sitio oficial de Doctrine. *Sitio oficial de Doctrine*. [En línea] [Citado el: 12 de diciembre de 2010.] <http://www.doctrine-project.org/>.
25. UFVdatabases. *UFVdatabases*. [En línea] 7 de febrero de 2009. [Citado el: 12 de diciembre de 2010.] <http://uvfdatabases.wordpress.com/2009/02/07/terminos-de-repaso->.
26. PostgreSQL. *PostgreSQL - Sitio Oficial*. [En línea] 2010. [Citado el: 20 de octubre de 2010.] <http://www.postgresql.org>.
27. **Admin, Somos Libres**. Somos Libres org. *Somos Libres org*. [En línea] 18 de Mayo de 2010. [Citado el: 15 de Mayo de 2011.] <http://www.somoslibres.org/modules.php?name=News&file=article&sid=3468>.
28. **II, Ingeniería de Software**. Entorno Virtual de Aprendizaje. *Entorno Virtual de Aprendizaje*. [En línea] EVA, 5 de Septiembre de 2010. [Citado el: 2 de Mayo de 2011.] http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_9/Conferencia_7/Materiales_Basicos/Sobre_la_disciplina_de_Prueba.pdf.
29. **RAE**. Diccionario de la Real Academia Española Vigésima segunda edición. *Diccionario de la Real Academia Española Vigésima segunda edición*. [En línea] 2001. [Citado el: 17 de octubre de 2010.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=ejercicio.

BIBLIOGRAFÍA.

1. *Conferencias de ISW. UCI, departamento de Ingeniería de Software.* 2010-2011.
2. **Collado, M.** (2003) "*Pruebas de software. Técnicas de prueba del software. Estrategias de prueba del software.*"
3. cyta.com. *Herramientas CASE. Ciencia y Técnica Administrativa.* [Online] [Citado el: 8 de noviembre de 2010.]
<http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c5/c5.htm>.
4. **Gamma, Erich, y otros.** 1994. *Design Patterns Elements of Reuseable Object-Oriented Software.* s.l: Addison Wesley, 1994.
5. **Gutmans, Andi.** 2005. *PHP 5 Power Programming.* [En línea] 2005.
<http://bibliodoc.uci.cu/pdf/reg04073.pdf>.
6. **James Rumbaugh, Ivar Jacobson, Grady Booch.** *El lenguaje Unificado de Modelado. Manual de Referencia.* s.l.: Addison Wesley.
7. **José Barrios, Emilio y Rojas Rojas, Johanna.** 2007. *INVESTIGACIÓN SOBRE ESTADO DEL ARTE EN DISEÑO Y APLICACIÓN DE PRUEBAS DE SOFTWARE.* [En línea] 2007. [Citado el: 18 de marzo de 2011.]
<http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/Pruebasdesoftware.html>.
8. **Kruchten, Philippe.** *Rational Unified Process, An Introduction,* Third Edition. 2003.
9. LaWebDelProgramador. [En línea] [Citado el: 21 de marzo de 2011.]
<http://www.lawebdelprogramador.com>.
10. **Larman, Craig.** 1999. *UML y Patrones.* México: Prentice Hall, 1999.
— 2003. *UML y Patrones,* Segunda Edición. s.l: PEARSON, 2003.
— 1999. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México: Prentice Hall, 1999. 970-17-0261-1.

11. Lenguajes de Programación,2009. *Lenguajes de Programación*. www.lenguajes-de-programacion.com. [Online] [Citado el: 23 de noviembre de 2010.] <http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml>.
12. **NetBeans** - *The Only IDE You Need*. www.netbeans.org. [Online] [Citado el: 5 de diciembre de 2010] <http://www.netbeans.org/features>.
13. **Molpeceres, Alberto**. *Procesos de desarrollo: RUP, XP y FDD*. Willi.net. [En línea] 2002. [Citado el: 19 de marzo de 2011.] <http://www.willydev.net/descargas/articulos/general/cualxpfdrrup.PDF>.
14. **Myers** (1979). "*Fase de Elaboración. FT Prueba (procedimientos genéricos y aplicación de algunos tipos de pruebas simples)*."
15. **Rational Unified Process**. *Ayuda del RUP*. Suite del Rational 2003.
16. **Sommerville, Ian**. 2005. *Ingeniería de Software - 7ma Edición*. Madrid: Addison Wesley, 2005. ISBN-84-7829-074-5. Sparx. [Online] [Citado el: 26 abril del 2011] <http://www.sparxsystems.com.ar/>.
17. **Visual Paradigm Organización**. 2009. *Sitio Web oficial Visual-Paradigm*. Sitio Web oficial Visual-Paradigm. [línea] 2009. [Citado el: 22 de febrero de 2011.] <http://www.visual-paradigm.com/product/vpum/>.
18. *Visual-Paradigm for UML*. **Visual-Paradigm**. [Online] [Citado el: 26 de febrero de 2011.] <http://www.visuparadigm.com/product/vpum/>.