UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD 4

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

Gestión personalizada de softareas y su asignación en la plataforma educativa Zera

Autores

Yisel Arias Valdés
Abdelasís León Rodríguez

Tutor

Ing. Ernesto Vladimir Pereda Díaz

Cotutor

Ing. Irina Ivis Santiesteban Perez

La Habana, junio 2012

"Año 54 de la Revolución"

Declaración de autoría

Declaramos que somos los autores de este trabajo y autorizamos a la Facultad 4 de la Universidad de las Ciencias Informáticas, así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

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

Agradecimientos

De "Avner":

Agradezco a mi mamá por todo el amor, confianza, el apoyo que me ha brindado, y la fuerza que me ha trasmitido a lo largo de toda mi vida, en la cual no hubiese podido llegar a este momento. Agradezco a mi hermana que siempre me ha impulsado a dar lo mejor de mí y a mi padre que me ha mostrado todo lo que puedo lograr con esfuerzo y dedicación. A mis tutores por el apoyo que me han brindado en la realización de esta tesis.

De Yise:

A mi hermanita linda, por ser mi guía, ejemplo a seguir, amiga y madre. Por cada minuto de dedicación, preocupación, por toda su paciencia, esmero. Un besito grande para ti feita.

A mi mamita por su apoyo incondicional, por su experiencia, por saberme guiar, confianza, dedicación. Por brindarme esa sonrisa colmada de sabiduría. Por trasmitirme cada día un gesto de amor.

Muchas gracias papi por formarme, por inculcarme cada una de las cualidades que hoy tengo. Gracias por enseñarme que era este el mejor camino a seguir. Sin ustedes tres no habría sido posible realizar este sueño.

A mi familia, en especial a mi tía Olguita, a mi abuelita Estrella, a mi abuelito Chino. Gracias por todo su apoyo y confianza.

A Carlitos, mi novio, por ser mi compañero, mi sostén. Por soportarme estos últimos días, por apoyarme y darme fuerzas para seguir adelante. Gracias mi nene por quererme como soy.

A mis vecinos Osvaldo y Zoe, por su experiencia y preocupación, por cada consejo.

A mis tutores Ernesto e Irina, por apoyarnos y confiar en nosotros. A Ernesto por saberme levantar cuando pensaba que no iba a poder.

A mis amigos, los de la niñez y los de la juventud, en especial a la Mile, Gise y Javy. A todos los que de una forma u otra hicieron posible que hoy este sueño se hiciera realidad.

Dedicatoria

A mi mamá, gracias por todo yo se que este también es tu sueño…

Avner

A mi papá, que aunque lejos, hoy me escucha, que aunque distante hoy me ve, que aunque lejano hoy se que se sentiría muy orgulloso de mí. Te dedico este trabajo papito, siempre te voy a tener presente.

A mi hermanita y a mi mami por ser lo más importante para mí, por ser mi sentido de ser, mi guía y mi fuerza para seguir adelante, las quiero mucho.

Yise

Resumen

La plataforma Zera en un Sistema de Gestión de Aprendizaje que cuenta con un módulo llamado: “Docente”. Este dispone de una sección para la gestión y asignación de tareas, tales como: recorridos dirigidos, ejercicios, investigaciones y en las que se centran específicamente “softareas”. La softarea es una actividad de tipo compleja compuesta por las actividades anteriormente mencionadas.

El presente trabajo propone el desarrollo de funcionalidades que permitan al docente la gestión y asignación de softareas asociados a objetivos, competencias y habilidades que se pretende que el estudiante adquiera. Se le permitirá al estudiante una vez asignada una tarea que sea resuelta para finalmente obtener una evaluación. Para lograr el objetivo que se persigue, como parte de la investigación se realizó un estudio de las diferentes plataformas de aprendizajes. Para guiar el proceso de desarrollo de software se transitó por los flujos de trabajo que propone la metodología RUP (del ingés Rational Unified Process): modelamiento del negocio, requerimiento, análisis y diseño, implementación y pruebas, generando en cada caso los artefactos necesarios. Finalmente se obtuvo una solución que cumple con cada una de las propuestas definidas.

Palabras Clave: softarea, actividad, competencias, habilidades.

Índice

Introducción....1
Capítulo 1: Fundamentación Teórica....7
Introducción....7
1.1 Conceptos asociados al dominio del problema....7
1.2 Sistemas similares....9
1.2 Ambiente de desarrollo....13
1.2.1 Metodologías de desarrollo de software....13
1.2.2 UML 2.0....15
1.2.3 Herramienta CASE para el modelado UML....16
1.2.4 Servidor web....17
1.2.5 Lenguajes y tecnologías del lado del cliente....18
1.2.6 Lenguajes y tecnologías del lado del servidor....20
1.2.7 Framework de desarrollo....20
1.2.8 Sistema Gestor de Base de Datos....21
1.2.9 Entorno de desarrollo integrado....21
Conclusiones....22
Capítulo 2:Características del Sistema....24
Introducción....24
2.1 Modelo del dominio....24
2.1.1 Conceptos del dominio....24
2.1.2 Diagrama del modelo de dominio....27
2.2 Descripción de la solución propuesta....27
2.3 Requerimientos del software....28
2.3.1 Requerimientos Funcionales....28
2.3.2 Requerimientos no funcionales....31

2.5 Modelo de Casos de Uso del Sistema 32
2.5.1 Descripción de los actores del sistema 32
2.5.2 Patrones de Casos de uso 33
2.5.3 Diagrama de actores del sistema 34
2.5.4 Diagrama de casos de uso del sistema 35
2.5.4 Descripción de los casos de uso del sistema 35
Conclusiones 37
Capítulo 3: Análisis y diseño de las funcionalidadesa implementar 38
Introducción 38
3.1 Modelo del análisis 38
3.1.1 Diagramas de clases del análisis 39
3.2 Patronesarquitectónicos y de diseño 39
3.2.1 Patrón arquitectónico Modelo-Vista-Controlador 39
3.2.2 Patrones de Diseño en Symfony 40
3.3 Modelo de Diseño 42
3.3.1 Diagrama de clases del diseño 42
3.4 Diseño de la base de datos 43
Conclusiones 44
Capítulo 4: Implementación y Pruebas de las Funcionalidades 45
Introducción 45
4.1. Modelo de Implementación 45
4.1.1. Diagramas de componentes 45
4.2. Pruebas de software 48
4.2.1. Niveles de Prueba 48
4.2.2. Métodos de prueba 49
4.2.3. Diseño de Casos de Prueba 50

4.2.4. Resultados obtenidos 52

Conclusiones..... 53

Conclusiones..... 54
Recomendaciones 55
Referencias Bibliográficas 56
Bibliografía 60
Glosario de Términos..... 62

Introducción

La era digital exige cambios en el mundo educativo por lo que se comenzaron a aprovechar las posibilidades que proporcionan las Tecnologías de la Información y las Comunicaciones (TIC), impulsando dichos cambios hacia un nuevo paradigma educativo más personalizado y centrado en la actividad de los estudiantes. El desarrollo de las TIC dio origen a un entorno más flexible y dinámico: e-learning, facilitando la creación, adopción y distribución de contenidos, así como la adaptación del ritmo de aprendizaje y la disponibilidad de herramientas de aprendizaje.

En aras de llevar a cabo programas de formación basados en e-learning se hace uso de los Sistemas de Gestión de Aprendizaje (LMS, del inglés Learning Management System), también ampliamente conocidos como plataformas de aprendizaje. Un LMS es un sistema web que permite la gestión y administración de contenidos digitales, impartir y dar seguimiento a cursos, además de presentar herramientas agrupadas y optimizadas para fines docentes. Dichos sistemas facilitan la interacción y comunicación entre profesores, alumnos y contenidos educativos contribuyendo a la adquisición de habilidades necesarias para la formación. (1)

El software educativo se considera como el conjunto de recursos informáticos diseñados con la intención de ser utilizados en el contexto del proceso de enseñanza - aprendizaje. Se caracterizan por ser altamente interactivos, a partir del empleo de recursos multimedia. El software educativo puede tratar las diferentes materias de formas muy diversas (a partir de cuestionarios, facilitando una información estructurada a los alumnos, mediante la simulación de fenómenos) y ofrecer un entorno de trabajo más o menos sensible a las circunstancias de los alumnos y más o menos rico en posibilidades de interacción. (2)

Para apoyar el proceso de enseñanza-aprendizaje en los diferentes sistemas educacionales (primario, secundario, preuniversitario) en Cuba se comenzaron a desarrollar varias colecciones de software educativos denominados “Hiperentornos de Aprendizaje” propuestos y desarrollados por especialistas y pedagogos del Ministerio de Educación, entre los que se pueden citar la Colección Multisaber, Colección Futuro y El Navegante. Dichos sistemas no son más que “la combinación armoniosa de diferentes tipologías de software educativo sustentada en la tecnología hipermedia”. (3)

En este marco la Universidad de las Ciencias Informáticas creó la plataforma educativa Zera, dicha plataforma cuenta con un subsistema basado en los hiperentornos de aprendizaje, en la que se

integran módulos como: Contenido, Docente, Biblioteca, Prácticas, Tareas entre otros. El módulo "Docente" tiene como principal objetivo facilitar la gestión de evaluaciones tanto online, como presencial de los estudiantes, llevar un control de su actitud ante el estudio y de la asistencia a clases. Además, se le bridan a los profesores variada información para su auto-preparación, y la posibilidad de agrupar las actividades que se gestionan y asignan a los estudiantes, tales como:

- ✓ **Recorrido Dirigido:** constituyen rutas de aprendizaje creados por el profesor a partir del contenido oficial de la materia, ejercicios y recursos mostrados en la biblioteca. Permite estructurar el contenido seleccionado mediante el uso de plantillas previamente establecidas.
- ✓ **Recursos investigativos:** tareas docentes educativas en la cual el estudiante ha tenido que investigar, resolver un problema y subir un archivo a la plataforma con el resultado de este.
- ✓ **Reactivo o Asignación:** cuestionario orientado por el docente compuesto por uno o varios ejercicios agrupados.
- ✓ **Softarea:** Es una tarea docente evaluativa creada para el estudiante, donde interactúan diferentes tipos de actividades, recorridos dirigidos, recursos investigativos y ejercicios.

Dichas tareas incrementan la interacción entre docentes y estudiantes, permitiendo el seguimiento y evaluación de estos últimos.

La confección de las softareas estaba limitada a la selección de un recorrido dirigido, una evidencia y uno o varios ejercicios sin poder alterar su orden.

Sin embargo la gestión de las investigaciones, recursos y ejercicios no están asociados a objetivos, competencias y habilidades que se pretende que los estudiantes adquieran. Esto le dificulta al docente la gestión de las tareas a la hora de seleccionarlas actividades que la conforman, ya que tendría que llevar un control y tener conocimiento de a que va dirigida cada una de las actividades, así como tener presente los objetivos que persiguen. Podría hacerse engorroso cuando existan muchas, ya que tendría que analizar cada recurso en el momento de crear la tarea.

De esta forma la creación de las tareas no están asociadas a objetivos de aprendizaje y esto trae consigo que el docente no pueda consultar el estado en que se encuentran sus estudiantes en determinados objetivos y así conocer los problemas que presentan estos, a partir de los resultados obtenidos en la solución de las actividades.

El **problema de la investigación** quedaría resumido en la siguiente interrogante: ¿Cómo facilitar al docente la atención a las diferencias individuales de los estudiantes en la plataforma Zera?

El **campo de acción** en que se enmarca el trabajo es la creación de tareas docentes enfocadas a la atención diferenciada de los estudiantes en la plataforma educativa Zera.

El **objeto de estudio** de la investigación es la creación de tareas de aprendizaje en los Sistemas de Gestión de Aprendizaje.

El **objetivo general** del presente trabajo es desarrollar funcionalidades que permitan al docente la gestión y asignación de tareas atendiendo al cumplimiento de objetivos de aprendizaje.

Del objetivo general se derivan los siguientes **objetivos específicos:**

- ✓ Realizar el estudio del estado del arte acerca de las tendencias actuales y soluciones similares que realicen la gestión y asignación de tareas.
- ✓ Realizar el análisis y diseño de clases que gestionen las actividades para la plataforma Zera.
- ✓ Implementar las funcionalidades de acuerdo a la estructura de diseño definida.
- ✓ Aplicar métodos de prueba.

Para guiar la investigación se plantea la siguiente **idea a defender**: La gestión y asignación de tareas, teniendo en cuenta los objetivos de aprendizaje facilita la atención a las diferencias individuales de los estudiantes en la plataforma educativa Zera.

Para dar cumplimiento a los objetivos específicos se planifican las siguientes **tareas:**

- ✓ Estudio de soluciones similares y selección de metodologías y tecnologías de desarrollo a utilizar.
- ✓ Análisis y definición de lenguajes de programación y herramientas de desarrollo a utilizar.
- ✓ Investigación sobre marcos de trabajo (en lo adelante frameworks) más usados a nivel mundial, plugins y componentes.
- ✓ Análisis y definición de la metodología de desarrollo a utilizar y herramientas de modelado.
- ✓ Análisis de la propuesta de arquitectura desarrollada para la plataforma educativa Zera.
- ✓ Identificación y especificación de los casos de uso asociados a la gestión y asignación de tareas en la plataforma educativa Zera.
- ✓ Realización del modelo de casos de uso que da cumplimiento a los requisitos funcionales y no funcionales.

- ✓ Realización de los diagramas de clases de análisis y diseño para cada caso de uso del sistema.
- ✓ Diseño del modelo de datos correspondiente a la propuesta de desarrollo.
- ✓ Modelado de diagrama de componente del sistema.
- ✓ Implementación de las clases definidas en el modelo de clases del diseño.
- ✓ Integración de la solución obtenida al resto de los módulos de la plataforma educativa Zera.
- ✓ Diseño de los casos de prueba.
- ✓ Pruebas a nivel de sistema y de integración.

Métodos investigativos

Los métodos científicos de investigación a utilizar son:

Métodos teóricos:

- ✓ Histórico-Lógico: Para la realización del estudio del estado del arte se hace uso de este método en la investigación de estándares y métodos utilizados en plataformas o soluciones similares, para conocer acerca de las metodologías de desarrollo de software, lenguajes, frameworks y herramientas de desarrollo.
- ✓ Analítico-Sintético: Este método se utiliza en el análisis de la teoría y extracción de los conceptos más importantes que serán incluidos en el marco teórico. La obtención de los principales elementos vinculados con la plataforma se obtuvieron gracias al estudio y previo análisis de la información, de esta manera damos cumplimiento a las tareas de la investigación.
- ✓ Modelación: Este se utiliza principalmente en la realización de los diagramas de casos de uso, clases del diseño e interacción, aplicando patrones para el diseño del modelo de datos y la distribución física del sistema para dar cumplimiento a los requisitos funcionales y no funcionales asociados a la gestión y asignación de tareas personalizadas.

Métodos empíricos:

- ✓ Observación: Se hizo uso de este método para recopilar datos relevantes durante el desarrollo de la investigación. Se logra conocer la esencia del problema planteado, ya que la

propuesta de solución y otras soluciones existentes son analizadas desde diferentes ópticas, de esta manera se puede definir lo que está realizado y lo que falta por hacer.

- ✓ Análisis estático: Mediante este método se descubren errores realizando pruebas al software, mediante la descripción de casos de prueba se puede determinar la efectividad de la solución. Para determinar la mejor vía del desarrollo del producto se utilizó la recopilación obtenida.

Resultados esperados

Con el desarrollo del trabajo se pretende obtener un conjunto de funcionalidades que den solución a:

- ✓ Seleccionar contenidos, recursos, investigaciones y ejercicios de acuerdo a objetivos, competencias y habilidades específicas para la gestión y asignación de tareas.
- ✓ Mejorar la usabilidad para la gestión de softareas.
- ✓ Asignar softareas de acuerdo a objetivos docentes y diferencias individuales.
- ✓ Llevar un control del cumplimiento de los objetivos docentes de los estudiantes en la plataforma Zera.
- ✓ Llevar un control del cumplimiento de las actividades asignadas a los estudiantes en la plataforma Zera.
- ✓ Evaluar las softareas asignadas a los estudiantes.
- ✓ Compartir las softareas para otros profesores.
- ✓ Resolver la softarea asignada por el docente.

El presente documento consta de 4 capítulos:

Capítulo 1: Fundamentación Teórica

Se exponen los elementos teóricos que sustentan el problema científico y los objetivos del trabajo. Se realiza un estudio de las soluciones similares, metodologías y herramientas de desarrollo que se ajustan al desarrollo de la investigación, fundamentando la selección de cada una de ellas en base al estudio realizado.

Capítulo 2: Características del Sistema

Se realizará el modelo de dominio, y se definirán los requerimientos funcionales y no funcionales, además se definen las características del sistema a implementar, se identifican actores y casos de uso, se brinda una descripción de los mismos y finalmente se obtiene el diagrama de casos de uso del sistema.

Capítulo 3: Análisis y diseño de las funcionalidades a implementar

Se realiza el análisis y diseño del sistema, se obtienen los diagramas del análisis y del diseño, diagramas de clases con estereotipos web. Se documenta el modelo de datos y el modelo de despliegue. Además se realiza el estudio de patrones de arquitectura y de diseño.

Capítulo 4: Implementación y Pruebas de las funcionalidades

Se realiza la implementación, utilizando el lenguaje de programación y la metodología seleccionada. Se describe cómo está implementado el sistema. Además, se definen los métodos de pruebas a realizar y se documentan los casos de prueba que se le realizarán al software.

Capítulo 1: Fundamentación Teórica

Introducción

La realización del estudio y análisis de las diferentes plataformas de aprendizaje es de vital importancia en la presente investigación, para conocer cómo se realizan los procesos de gestión y asignación de tareas y cómo se les da seguimiento a los estudiantes en cuanto a sus evaluaciones. Como parte de la investigación para guiar el proceso de desarrollo de software se hace necesario el estudio de tecnologías y metodologías, así como las herramientas de desarrollo y modelado que permitan brindar una mejor solución a la propuesta.

1.1 Conceptos asociados al dominio del problema

E-learning

Según algunas bibliografías se define E-learning como:

- ✓ Una de las estrategias formativas que puede resolver muchos de los problemas educativos, que van desde el aislamiento geográfico del estudiante de los centros del saber hasta la necesidad de perfeccionamiento constante que nos introduce la sociedad del conocimiento. (4)
- ✓ La capacitación no presencial que, a través de plataformas tecnológicas, posibilita y flexibiliza el acceso y el tiempo en el proceso de enseñanza-aprendizaje, adecuándolos a las habilidades, necesidades y disponibilidades de cada discente, además de garantizar ambientes de aprendizaje colaborativos mediante el uso de herramientas de comunicación síncrona y asíncrona, potenciando en suma el proceso de gestión basado en competencias. (5)
- ✓ Modalidad de formación adaptada a las necesidades y posibilidades de las personas adultas, tanto ocupadas como desempleadas. (6)

A partir de las definiciones anteriores se puede decir que e-learning es una estrategia formativa que, mediante plataformas tecnológicas, posibilita y flexibiliza el acceso y el tiempo en el proceso de enseñanza-aprendizaje.

Software Educativo

Según algunas bibliografías se define software educativo como:

- ✓ Cualquier programa computacional cuyas características estructurales y funcionales sirvan de apoyo al proceso de enseñar, aprender y administrar. Es además un material de aprendizaje especialmente diseñado para ser utilizado con una computadora en los procesos de enseñar y aprender. (7)
- ✓ Son aplicaciones o programas computacionales que faciliten el proceso de enseñanza aprendizaje. (8)

A partir de las definiciones anteriores se puede decir que software educativo es aquel programa computacional destinado a la enseñanza y el aprendizaje autónomo, que además, permite el desarrollo de ciertas habilidades cognitivas.

Sistemas de Gestión de Aprendizaje (LMS, del inglés Learning Management System)

Según algunas bibliografías se define LMS como:

- ✓ Una plataforma que permite a equipos de formación, crear y administrar un curso a través de una página web. (9)
- ✓ Software que automatiza la administración de acciones de formación. (10)

A partir de estas definiciones se puede decir que un sistema de gestión de aprendizaje es un software que automatiza la administración de acciones de formación, principalmente en la educación a distancia.

Objetivo de aprendizaje

Según algunas bibliografías se define objetivo de aprendizaje como:

- ✓ Los objetivos de aprendizaje son conductas estudiantiles especificas, observables, de corto plazo, evaluables. Cimientos sobre los cuales se puede construir lecciones y valoraciones con las que pueda probar que se están cumpliendo las metas generales de su curso o lección. (11)
- ✓ Los objetivos de aprendizaje comunican lo que en el curso educativo se espera que el estudiante aprenda. Lo que el estudiante ha de ser capaz de demostrar al finalizar un período de aprendizaje. (12)

A partir de estas definiciones se puede decir que los objetivos de aprendizaje es lo que el docente se propone lograr en el estudiante durante el curso. Demostrará si el estudiante ha vencido los contenidos impartidos durante el período.

1.2 Sistemas similares

Dokeos

Dokeos es un sistema de aprendizaje virtual basado en la web, técnicamente conocido como un LMS o Sistema de Gestión de Contenido (CMS,del inglés Content Management System). Es intuitivo y fácil de usar para todo tipo de usuario (profesores, formadores, estudiantes), ofrece una amplia gama de herramientas y facilita la creación y organización de contenidos interactivos y ejercicios.

Al margen de su facilidad de uso, Dokeos es un software de código libre y gratuito. Ofrece un eficiente y amigable entorno virtual que integra herramientas colaborativas y de creación de contenidos y actividades. Además cuenta con otras más sofisticadas para dar seguimiento y generar informes sobre el desempeño de los alumnos en el curso. (13)

Entre las herramientas de interacción se cuenta con una encargada de la gestión y asignación de tareas. Cada tarea tiene un nombre, una descripción detallada de la orientación, de manera opcional una calificación, fecha límite y de cierre. Las tareas pueden ser editadas y eliminadas, brindando la opción de ponerlas visibles o no dependiendo de la estrategia didáctica que cada docente desee aplicar a sus estudiantes.
Presenta un itinerario de aprendizaje[1] dividido en dos pasos. El progreso de los estudiantes puede controlarse por los prerrequisitos, es decir tiene que completar ciertos pasos antes de continuar. El itinerario ofrece el potencial de crear un viaje a través de una base de conocimientos que incluye recursos, test, tareas, discusiones y evaluación.

Dokeos además cuenta con un baúl de tareas para facilitar el intercambio de archivos entre los participantes del curso. El creador del curso puede enviar archivos a uno o más estudiantes, estos al primero y también entre ellos. Los ficheros enviados pueden contener comentarios por parte del estudiante o del profesor.

Moodle

Moodle es un Sistema de Gestión de Aprendizaje que tiene como principal objetivo facilitar a los educadores las mejores herramientas para gestionar el aprendizaje.

[1]Secuencia instruccional estructurada dividida en capítulos.

Promueve una pedagogía constructivista social (colaboración, actividades, reflexión crítica, etc.). Su arquitectura y herramientas son apropiadas para clases, así como también para complementar el aprendizaje presencial. Cuenta con una interfaz atractivo, de tecnología sencilla, ligera eficiente y compatible. (14)

Moodle define una tarea como cualquier trabajo, labor o actividad que asigna el docente al estudiante. Los alumnos pueden subir sus tareas (en cualquier formato de archivo) al servidor, se registra la fecha en que se han subido. Para la creación de una tarea se debe elegir la opción que permite crear una actividad, la tarea tiene asociado título, descripción detallada de la actividad que se va a asignar, además incluye puntos que deben cubrir y objetivos. (15)

El sistema añade automáticamente la calificación máxima y la fecha límite de entrega de la misma, igualmente el profesor puede especificar la fecha final de entrega y la calificación máxima que se le podrá asignar.

Define tipos de tareas para indicar si se espera que los alumnos envíen un fichero.

Son posibles dos opciones:

- ✓ Actividad off-line: el trabajo es realizado fuera de la plataforma, permite a los estudiantes ver una descripción del mismo, pero no subir archivos. El docente puede calificar a todos sus alumnos y enviarles una notificación de sus notas.
- ✓ Subir un fichero: permite a los estudiantes subir un archivo de cualquier tipo, ya sea un documento realizado con un procesador de textos o una imagen, un sitio web comprimido o algo que les haya pedido el profesor que remitan según la orientación. El docente igualmente tendrá la posibilidad de calificar directamente los trabajos recibidos.

Se permite enviar tareas fuera de tiempo, pero el profesor puede ver claramente el tiempo de retraso. Las observaciones del profesor se adjuntan a la página de la tarea de cada estudiante y se le envía un mensaje de notificación.

Las tareas no pueden ser reenviadas después de haber sido calificadas, pero el docente tiene la posibilidad de activar la opción, y de esta manera se le permitirá al estudiante reenviar las tareas corregidas.

La desventaja que presenta Moodle es que muestra los mismos contenidos a todos los alumnos. Es decir, no tiene forma de guiar el proceso de aprendizaje de cada alumno, asignando a cada cual las tareas enfocadas en resolver los problemas individuales.

Claroline

Claroline es un Sistema de Gestión de Aprendizaje que permite a los profesores crear y administrar webs de cursos desde un navegador. Dispone de una administración muy sencilla, de un espacio de encuentro con sus estudiantes, interactuando con herramientas que le permitirán gestionar su curso. (16)

Claroline hace uso de la herramienta "Tarea" para que los profesores coloquen tareas a sus estudiantes y estos puedan enviar su trabajo a través de la plataforma.
Para la creación de una nueva tarea se define título, descripción y el tipo de entrega que requiere el profesor, estas pueden ser:

- ✓ Archivo elaborado en una herramienta independiente (procesador de texto, hoja electrónica, presentación con diapositivas, entre otras).
- ✓ Texto elaborado directamente en el editor de la plataforma, brindando la opción de anexar un archivo.

Otros aspectos que integran la tarea son la especificación de la entrega (individual o en grupo), fecha de inicio y fecha fin, habilitar permiso para entregar la tarea pasada la fecha de fin. Se podrá además definir si el trabajo entregado será visible para todos o solamente para el profesor y los autores.
Las tareas una vez creadas, pueden ser editadas, eliminadas o hacerlas invisibles para los estudiantes.

Claroline presenta la herramienta Usuarios, y dentro de ella dispone de una opción que permite dar seguimiento a las estadísticas sobre las actividades que ha realizado un usuario en particular en el curso. Dentro de los datos que brindan se encuentran; los resultados de los ejercicios realizados, las rutas de aprendizajes, los trabajos publicados y documentos.

Blackboard

Blackboard es una plataforma de e-learning flexible y sencilla, proporciona un sistema de administración de cursos, establece un portal personalizable, comunidades, así como una arquitectura que permite una fácil integración de múltiples sistemas administrativos. (17)

La página “Tareas” organiza actividades, mediante la definición de la prioridad y el seguimiento del estado de estas. El profesor puede crear, publicar, consultar, modificar, eliminar, ordenar y ver los detalles de determinada tarea en esta sección.

La información de las tareas se organiza por columnas que muestran el nombre, el estado y la fecha de vencimiento de la tarea. Además brinda la posibilidad de definir una propiedad (baja, normal o alta).

La herramienta Assignment en Blakboard es la encargada de diseñar actividades de aprendizaje, brindando la oportunidad de que los alumnos entreguen sus tareas de manera electrónica a través de la plataforma. Además permite generar un espacio en la que los alumnos obtienen instrucciones creadas por el profesor y donde encontrarán un vínculo a través del cual los estudiantes podrán entregar un archivo electrónico.

Las asignaciones disponen de un nombre, valor en puntos para la actividad, fecha de entrega, objetivo, instrucciones, y un material de apoyo si lo considera necesario el profesor.

Conclusiones del estudio de sistemas similares

Los Objetivos de Aprendizaje comunican lo que el curso, asignatura o unidad educacional espera que el estudiante aprenda. En otras palabras, lo que el estudiante debe ser capaz de demostrar al final de un período de aprendizaje. Del estudio realizado a los sistemas similares, se tomaron como referencia y se tuvieron en cuenta las características relacionadas con la problemática planteada. Algunas de estas, pueden aplicarse directamente a las softareas y se citan a continuación:

- ✓ Una vez creadas el autor pueda definir si serán compartidas con el resto de los profesores, es decir definirá si estarán publicadas.
- ✓ Generar una nota automática, brindándole al docente la opción de aceptarla o de asignar una él mismo.
- ✓ Enviar un mensaje de notificación cuando se le sea asignada una tarea al estudiante.

Algunas características ausentes en estas son:

- ✓ La posibilidad de filtrar el contenido de acuerdo a los objetivos de aprendizaje lo cual le facilitaría al usuario crear tareas más estructuradas y orientadas al cumplimiento de varios objetivos de aprendizaje.
- ✓ Consultar el estado de los estudiantes en el cumplimiento de dichos objetivos de aprendizaje o el estado en contenidos específicos de una materia.

- ✓ La creación de estructuras de tareas en las cuales el estudiante interactúe con varios tipos de actividades que tributen al cumplimiento de varios contenidos u objetivos de aprendizaje.

1.2 Ambiente de desarrollo

La selección de la metodología adecuada garantiza la creación de un software de calidad y el desarrollo de un producto en el tiempo planificado y con los costes previamente establecidos. Para dar inicio al desarrollo de la plataforma educativa Zera se realizó previamente un estudio rigoroso y detallado de tecnologías, herramientas de modelado, lenguajes de modelado, frameworks, servidores web, sistemas gestores de base de datos y lenguajes de desarrollo para escoger aquellas que se adaptaban mejor dependiendo de las características con las que contaba el proyecto. La solución propuesta será agregada al módulo Docente de dicha plataforma, por lo que se consideró trabajar sobre el ambiente de desarrollo existente.

1.2.1 Metodologías de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software. Durante la creación de la plataforma Zera se realizó un estudio de las diferentes metodologías de desarrollo (ágiles y tradicionales) y de acuerdo a sus características se concluyó guiar el proceso de desarrollo sobre las metodologías tradicionales.

Las metodologías tradicionales proponen mayor énfasis en la planificación y control de un proyecto, en la especificación precisa de requisitos y modelado. Además imponen una disciplina de trabajo sobre el proceso de desarrollo del software, con el fin de conseguir un software más eficiente, haciendo énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo de una aplicación.

Las metodologías tradicionales se centran en la documentación detallada, el control del proceso; mediante la definición de roles, actividades, artefactos, herramientas y notaciones para el modelado. Este tipo de metodología no es adaptable a los cambios, por lo que son métodos adecuados cuando se trabaja en un entorno, donde los requisitos no pueden variar. (18)

Proceso Unificado de Desarrollo

El Proceso Unificado de Desarrollo (RUP,del ingésRational Unified Process) es una metodología de desarrollo de software que está basado en componentes e interfaces bien definidas, y junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. (19)
Para guiar el proceso de desarrollo de software se transita por cinco flujos de trabajo de los que propone RUP (ver Figura 1), generando en cada uno los artefactos necesarios y realizando el rol que se precise para cada actividad a realizar. Las principales características que sirven de apoyo se mencionan a continuación:

Dirigido por casos de uso: Una vez que se haya modelado el negocio, se representa a través de requerimientos y estos son agrupados en casos de uso, reflejando la propuesta de solución.

Centrado en la arquitectura: La arquitectura define la visión común del sistema completo, describe los elementos del modelo que son más importantes para su construcción.

Iterativo e Incremental:Cada iteración influye en cada uno de los flujos de trabajo, aunque se desarrolla fundamentalmente en algunos más que otros.
Al no estar bien definidos los procesos que intervienen en el desarrollo de la propuesta de solución se transita por el flujo de trabajo de modelamiento del negocio, generando el modelo del dominio. Ello sirve como entrada al flujo de trabajo de requerimientos donde se identifican los requisitos funcionales y no funcionales, se agrupan en casos de uso y se generan los artefactos: diagrama de actores, diagrama de casos de uso del sistema. Pasando al flujo de trabajo Análisis y Diseño se traducen los requisitos a una especificación que describe cómo implementar el sistema, se obtiene una visión de lo que el sistema va a hacer, basándose en los requisitos funcionales. Por otro lado en el diseño se refina el análisis teniendo en cuenta los requisitos no funcionales, cómo cumple el sistema sus objetivos. Se generarán los artefactos: modelo del análisis, donde se identifican las clases del análisis y los diagramas de clases del análisis y el modelo de diseño, encargado de realizar los diagramas de clases con estereotipos web. Se genera el modelo de entidad relación de la base de datos y el diagrama de despliegue. Dichos artefactos constituyen la entrada al flujo de trabajo de implementación, el cual describe cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue. Finalmente en el flujo de trabajo de pruebas se evalúa la calidad de la propuesta a desarrollar apoyándonos en la descripción de casos de pruebas y métodos.

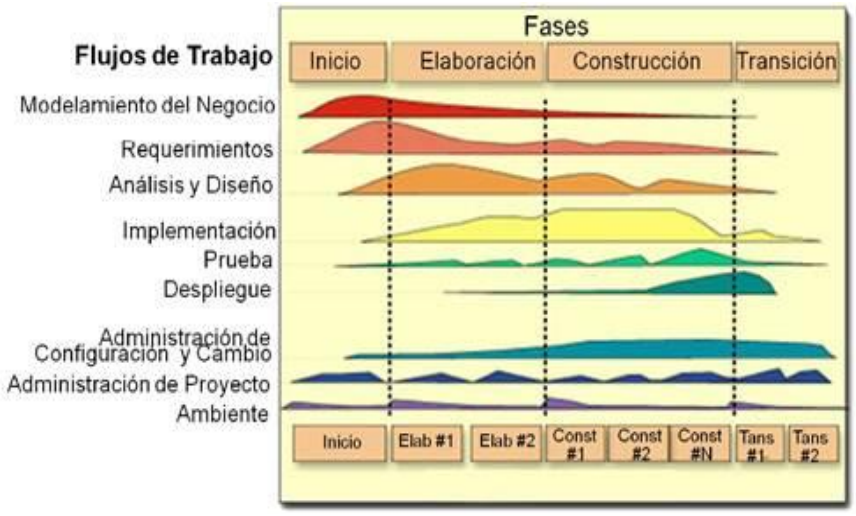


Figura 1 Proceso Unificado de Desarrollo (20)

1.2.2UML 2.0

Lenguaje Unificado de Modelado (UML, del inglés Unified Modeling Language) provee un sistema de arquitecturas trabajando con objetos, análisis y diseño, con una buena consistencia del lenguaje para especificar, visualizar, construir y documentar los artefactos de un sistema de software. Uno de los objetivos principales de la creación de UML es posibilitar el intercambio de modelos entre las distintas herramientasCASE (Ingeniería de Software Asistida por Computadora) orientadas a objetos del mercado. Para ello era necesario definir una notación y semántica común. (21)

Los objetivos de UML son muchos, pero se pueden sintetizar sus funciones: (21)

- ✓ Visualizar: permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- ✓ Especificar: permite especificar cuáles son las características de un sistema antes de su construcción.
- ✓ Construir: a partir de los modelos especificados se pueden construir los sistemas diseñados.
- ✓ Documentar: los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

Un modelo UML está compuesto por tres clases de bloques de construcción:

- ✓ Elementos: los elementos son abstracciones de entidades reales o ficticias (objetos, acciones, etc.)

- ✓ Relaciones: relacionan los elementos entre sí.
- ✓ Diagramas: son colecciones de elementos con sus relaciones.

1.2.3Herramienta CASE para el modelado UML

Se puede definir a la Ingeniería de Software Asistida por Computadora (CASE, del inglés Computer Aided Software Engineering) como el conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software. (22)

Estas herramientas pueden proveer muchos beneficios en todas las etapas del proceso de desarrollo de software, algunas de ellas son:

- ✓ Verificar el uso de todos los elementos en el sistema diseñado.
- ✓ Automatizar el dibujo de diagramas.
- ✓ Ayudar en la documentación del sistema.
- ✓ Ayudar en la creación de relaciones en la Base de Datos.
- ✓ Generar estructuras de código.

Visual Paradigm 8.0

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue (23).

El software de modelado UML ayuda a una rápida construcción de aplicaciones de calidad. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

Algunas características:

- ✓ Utiliza UML como lenguaje de modelado, facilitando la creación de los artefactos que propone la metodología seleccionada
- ✓ Brinda la posibilidad de sincronización del modelo de diseño y el código en todo el ciclo de desarrollo una vez que se integra con el entorno de desarrollo NetBeans.

- ✓ Presenta una interfaz de uso intuitiva y con muchas facilidades a la hora de modelar los diagramas.
- ✓ Disponibilidad en múltiples plataformas.
- ✓ Diseño centrado en casos de uso y enfocado al negocio, generando un software de mayor calidad.
- ✓ Permite importar y exportar ficheros XML

1.2.4Servidor web

Apache 2.2.6

Apache es un software libre desarrollado por la Fundación de Software Apache [2]. Tiene como objetivo suministrar páginas web a los clientes o navegadores que las solicitan. Realiza la transferencia de hipertexto mediante el Protocolo de Transferencia de Hipertexto (HTTP, del inglés HiperText Transfer Protocol), basado en el envío de mensajes mediante un conjunto de normas, a través de las cuales se remiten las peticiones de acceso a una web y la respuesta de esta. Utiliza la arquitectura cliente/servidor donde el equipo cliente hace una solicitud o petición al equipo servidor y éste la atiende. (24)

Entre sus características principales se encuentran:

- ✓ Implementa una estructura híbrida entre procesos e hilos, lo cual le permite atender un número creciente de peticiones con menos recursos del sistema. De esta forma puede mantener múltiples procesos cada uno con varios hilos sin perder la estabilidad. (25)
- ✓ Brinda soporte a PHP, permite personalizar la respuesta ante errores que pueden surgir en el servidor.
- ✓ Es extensible mediante módulos, flexible, rápido y eficiente.
- ✓ Es el servidor web más utilizado en la actualidad. (26)

Proporciona contenidos al cliente web o navegador como:

- ✓ Páginas estáticas: uso más generalizado que se hace de un servidor web. De esta forma se transfieren archivos HTML, imágenes, entre otras.

[2] http://www.apache.org

- ✓ Páginas dinámicas: muestra información en las páginas mediante contenido dinámico obtenidas a partir de consultas a bases de datos u otras fuentes de datos.

1.2.5 Lenguajes y tecnologías del lado del cliente

Un cliente Web es una aplicación de software que está formada por el código HTML[3] que forma la página web, con opción a código ejecutable mediante los lenguajes de scripting de los navegadores o mediante pequeños programas en Java. Los lenguajes del lado del cliente basan su procesamiento en el cliente web, es decir que se ejecutan en el navegador del usuario. Es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio.

Lenguaje de Marcado de Hipertexto Extensible

El lenguaje de Marcado de Hipertexto Extensible (XHTML, del inglés Extensible HyperText Markup Language) es una versión más estricta y limpia de HTML, que nace con el objetivo de reemplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML[4]. XHTML extiende HTML 4.0 combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos. (27)

XHTML no es solo un HTML con sintaxis XML. En realidad, incorpora una nueva filosofía de modelación de las páginas web, que busca la creación de una web semántica, permitiendo separar el contenido de la presentación. Proporciona una mejor velocidad de navegación ya que las páginas creadas y maquetadas correctamente se arman más rápido en los exploradores que si no lo están, ya sea porque hay errores de sintaxis o porque se usan elementos con un fin para el cual no fueron creados.

Hojas de Estilo en Cascada

Las hojas de estilo en cascada, (CSS,del inglés Cascading Style Sheets), es un lenguaje que describe la presentación de documentos estructurados en hojas de estilo para diferentes métodos de interpretación, es decir, describe cómo se va a mostrar un documento en pantalla. 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. Permite a los desarrolladores web

[3] siglas de HyperText Markup Language (lenguaje de marcado de hipertexto), hace referencia al lenguaje de marcado predominante para la elaboración de páginas web

[4] siglas de eXtensible Markup Language (lenguaje de marcas extensible), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium.

controlar el estilo y el formato de múltiples páginas web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento. (28) Se hará uso de CSS en su versión 2.1.

JavaScript

Java Script es un lenguaje del lado del cliente muy utilizado en el desarrollo web tanto para la realización de pequeñas tareas, como para la gestión de complejas aplicaciones. Es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador que lo soporte sin necesidad de procesos intermedios.

Las aplicaciones web cada vez son más complejas, ya que incorporan nuevos efectos visuales e interacciones dinámicas. Se hace imprescindible utilizar librerías Java Script que simplifiquen el desarrollo y permitan crear aplicaciones compatibles con todos los navegadores. (29)

La importancia de Java Script radica en la posibilidad de agregar respuestas inmediatas en las páginas HTML, es posible detectar y modificar eventos tales oprimir un botón o un vínculo, cargar páginas, validar formularios antes de enviarlos. Nos brinda efectos especiales sobre las páginas web, para crear contenidos dinámicos y elementos de las páginas que tengan movimiento, cambien de color o cualquier otro dinamismo. Además nos permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que podemos crear páginas interactivas. (30)

Ajax

Ajax es una técnica de desarrollo web para la creación de aplicaciones interactivas, es un acrónimo de Asynchronous JavaScript y XML. Dichas aplicaciones se ejecutan en el lado del cliente y mantienen comunicación asíncrona con el servidor, en segundo plano, sin que el usuario se percate de dichas comunicaciones a nivel de tiempos de respuestas, se considera un conjunto de tecnologías que se desarrollan por sí mismas y se unen en poderosas nuevas formas, incorporando: presentación basada en estándares usando XHTML y CSS, exhibición e interacción dinámicas usando DOM e intercambio y manipulación de datos usando XML. No necesitan refrescar la página completa para actualizar información solicitada, pueden simplemente actualizar parte de la página, dándole al usuario una respuesta a sus consultas por las páginas web. (31)

1.2.6 Lenguajes y tecnologías del lado del servidor

Los lenguajes del lado del servidor son reconocidos, ejecutados e interpretados por el propio servidor y se envían al cliente en un formato comprensible para él. Es independiente del cliente por lo que es mucho menos rígido respecto al cambio de un navegador a otro o respecto a las versiones del mismo.

PHP 5.3

Procesador de Hipertexto (PHP, del inglésHypertext Preprocessor) es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. Está muy orientado al desarrollo de aplicaciones web y permite insertar contenidos dinámicos en las páginas.

A continuación se exponen las principales características del lenguaje PHP: (32)

- ✓ Es multiplataforma, funciona tanto en sistemas Unix o Linux.
- ✓ Es un lenguaje basado en herramientas con licencia de software libre.
- ✓ Presenta sintaxis cómoda. Lo más destacado ocurre a nivel semántico, donde el tipado es muy poco estricto.
- ✓ Tiene soporte para la programación orientada a objetos.
- ✓ Cuenta con una extensa librería de funciones que facilitan el trabajo de los desarrolladores.
- ✓ Presenta soporte para un gran número de gestores de bases de datos: Adabas D, dBase, Empress, Ingress, InterBase, FrontBase, DB2, Informix, mSQL, MySQL, ODBC, Oracle, PostgreSQL, Sybase, etc.

1.2.7 Framework de desarrollo

Symfony 1.4.15

Symfony es un marco de trabajo diseñado para optimizar el desarrollo de las aplicaciones web mediante algunas de sus principales características:

- ✓ Extensible mediante plugins y la amplia variedad de estos disponibles.
- ✓ Es fácil de instalar y configurar en sistemas operativos GNU/Linux, Windows.

- ✓ Compatible solamente con PHP 5 desde hace años, para asegurar el mayor rendimiento y acceso a las características más avanzadas de PHP.
- ✓ Posee una extensa documentación, ya que cuenta con abundantes libros y tutoriales distribuidos de manera gratuita. (33)

1.2.8 Sistema Gestor de Base de Datos

Un sistema Gestor de Base de Datos es un conjunto de programas que permiten introducir y almacenar datos, ordenarlos y manipularlos, en fin realizar diversas operaciones con la información de una base de datos. El principal propósito de los SGBD es trabajar de forma sencilla, ordenada y clara con un conjunto de datos que posteriormente nos servirá de mucho apoyo para poder obtener una factible manipulación de los mismos. Poseen grandes ventajas como lo son, el control de la redundancia, restricción de los accesos no autorizados y además ofrecen recursos para definir y garantizar el cumplimiento de las restricciones de integridad.

PostgreSQL 9.0

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional bajo licencia BSD[5] (Distribución de Software de Berkeley) que dispone libremente con código fuente.

Se ha decidido seleccionar PostgreSQL porque sus características técnicas la hacen una de las bases de datos más potentes y robustas del mercado. Su desarrollo comenzó hace más de 15 años, y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo. PostgreSQL posee grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Estas razones antes mencionadas hacen que se seleccione a PostgreSQL para la solución propuesta.

1.2.9Entorno de desarrollo integrado

Un entorno de desarrollo integrado (IDE,del inglés Integrated Development Environment)se empaqueta como un programa de aplicación, es decir, consiste en un editor de código, un

[5] Licencia de software que permite quitar la libertad al software.

compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes.Proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, C#, Delphi, Visual Basic, etc. En algunos lenguajes, un IDE puede funcionar como un sistema en tiempo de ejecución, en donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto.

NetBeans 6.9

Decidimos utilizar esta versión por consumir menos recursos del sistema y poseer características como:

- ✓ Es una versión del IDE que facilita el desarrollo de sitios web PHP con una variedad de secuencias de comandos y lenguajes de marcado que se integra dinámicamente con HTML, JavaScript y CSS.
- ✓ Posee completamiento de código para los métodos mágicos que usa el ORM Doctrine.
- ✓ Se integra con Symfony.
- ✓ Se integra con Subversion, lo que le permite controlar la evolución de los cambios en los archivos.

Mediante NetBeans es posible diseñar aplicaciones con solo arrastrar y soltar objetos sobre la interfaz de un formulario. Permite elaborar potentes aplicaciones para la web. La programación mediante este entorno se realiza a través de componentes de software modulares, también llamados módulos. NetBeans pone a disposición de los usuarios decenas de módulos a través de su página web, que pueden ser integradas en él para conseguir mejores aplicaciones. (34)

Conclusiones

El estudio realizado a las soluciones similares permitió identificar y analizar algunas de las funcionalidades que se tendrán en cuenta para la posterior implementación de la propuesta de solución. Para el desarrollo de dichas funcionalidades y tomando como referencia las características mencionadas y explicadas en la selección de las herramientas y metodologías se determinó utilizar: como lenguaje de programación PHP 5.3, integrando los frameworks Symfony 1.4.15, como IDE Netbeans 6.9, como servidor Web Apache 2.2.6, gestor de base de datos PostgreSQL 9.0, para apoyar el proceso de desarrollo de software se decidió utilizar la

metodología RUP y el lenguaje de modelado UML 2.0, herramienta CASE Visual Paradigm 8.0 para la modelación de los diagramas necesarios.

Capítulo 2:Características del Sistema

Introducción

En el desarrollo de software es de vital importancia definir los procesos que intervienen en este, para lograr así un mejor entendimiento del sistema a desarrollar por parte de los clientes y desarrolladores, aquí surge el concepto de modelo de dominio, generándose en el flujo de trabajo Modelado del Negocio, cuando los conceptos no están bien claros y definidos. Posterior a este se realiza la captura de requisitos, perteneciente al flujo de Requerimientos, donde se identifican los requisitos funcionales y no funcionales, permitiendo tener con claridad las capacidades y cualidades que el producto debe tener. Se generan los casos de uso, actores, diagramas de casos de uso del sistema, constituyendo artefactos fundamentales en el desarrollo posterior.

2.1 Modelo del dominio

Después de haber realizado un estudio del proceso del negocio, se comprueba que no hay un concepto bien definido de los procesos existentes, por lo que se decide realizar un modelo de dominio, seleccionando y haciendo uso de los conceptos más importantes. El modelo dará la posibilidad de conocer los principales conceptos que se manejan en el dominio de los procesos actuales del negocio, donde tanto usuarios, clientes como desarrolladores manejarán un vocabulario común.

2.1.1 Conceptos del dominio

Usuario: Son las personas que generalmente interactúan con las tareas, puede comportarse como estudiante o docente.

Docente: Persona que puede impartir una o varias materias en uno o muchos grupos. Puede asignar actividades al grupo o individualmente, hace el seguimiento a las actividades de todos los estudiantes que estén a su cargo y evalúa a los estudiantes que le imparte el contenido.

Estudiante: Es la persona que recibe y resuelve las tareas asignadas por el docente, con el objetivo de incrementar sus conocimientos, y al mismo tiempo que el profesor lleve un seguimiento de su aprendizaje, para atender diferencias individuales.

Tareas: El rol docente tiene un espacio para la gestión de tareas. Podrán ser asignadas a los estudiantes de acuerdo a sus diferencias individuales, además serán evaluadas.

Asignación: la asignación es creada por el docente para el estudiante. Está dividido en 3 fases: la selección de la actividad a realizar, la selección de los estudiantes a los que le desea asignar dicha actividad y la presentación de los datos de la misma.

Softarea: es una tarea docente evaluativa creada para el estudiante, donde interactúan diferentes tipos de actividades, recorridos dirigidos, investigaciones y ejercicios.

Recorrido Dirigido: los recorridos dirigidos son rutas de aprendizaje creados por el profesor a partir del contenido oficial de la materia, los ejercicios y recursos mostrados en la biblioteca. Al recorrido se puede agregar, una página del contenido o un párrafo de la misma, permitiendo estructurar el contenido seleccionando mediante el uso de plantillas previamente establecidas.

Investigación: constituyen orientaciones que crea el docente para que los estudiantes desarrollen habilidades investigativas. Van a tener asociadas una evaluación y poseen una complejidad (difícil, media, fácil).

Ejercicio: actividad mental que consiste en la repetición de ciertas rutinas, con el fin de desarrollar una determinada habilidad. Permitirán comprobar el nivel de asimilación de un contenido por parte de un estudiante.

Recurso Didáctico: se entiende por recurso didáctico, la información que se presenta a partir de cualquiera de los siguientes formatos digitales o sus combinaciones: texto o hipertexto, video digital, sonido digital, animaciones e imagen. Elemento que permite enriquecer los contenidos de las diferentes materias.

Página: son los diferentes elementos por los que está compuesto un contenido. Contiene textos y puede incluir varios recursos.

Conocimientos específicos: son los objetivos para cumplir determinadas competencias.Se definen como el conjunto de comprensiones que deberá tener el estudiante para vencer un contenido determinado.

Índice: representa la estructura del contenido del curso a través del índice de contenido (capítulos, temas y subtemas).

Actividad: conjunto de acciones planificadas creadas por docentes y orientadas a estudiantes, de carácter individual o grupal, que tienen como finalidad alcanzar los objetivos o propósitos del docente. Las actividades constituyen recorridos dirigidos, ejercicios, investigaciones y softareas.

Evaluación: el docente podrá evaluar la softarea realizada por el estudiante, valorando su complejidad, fecha de cumplimiento y resolución de las diferentes actividades por la que esté compuesta.

2.1.2 Diagrama del modelo de dominio

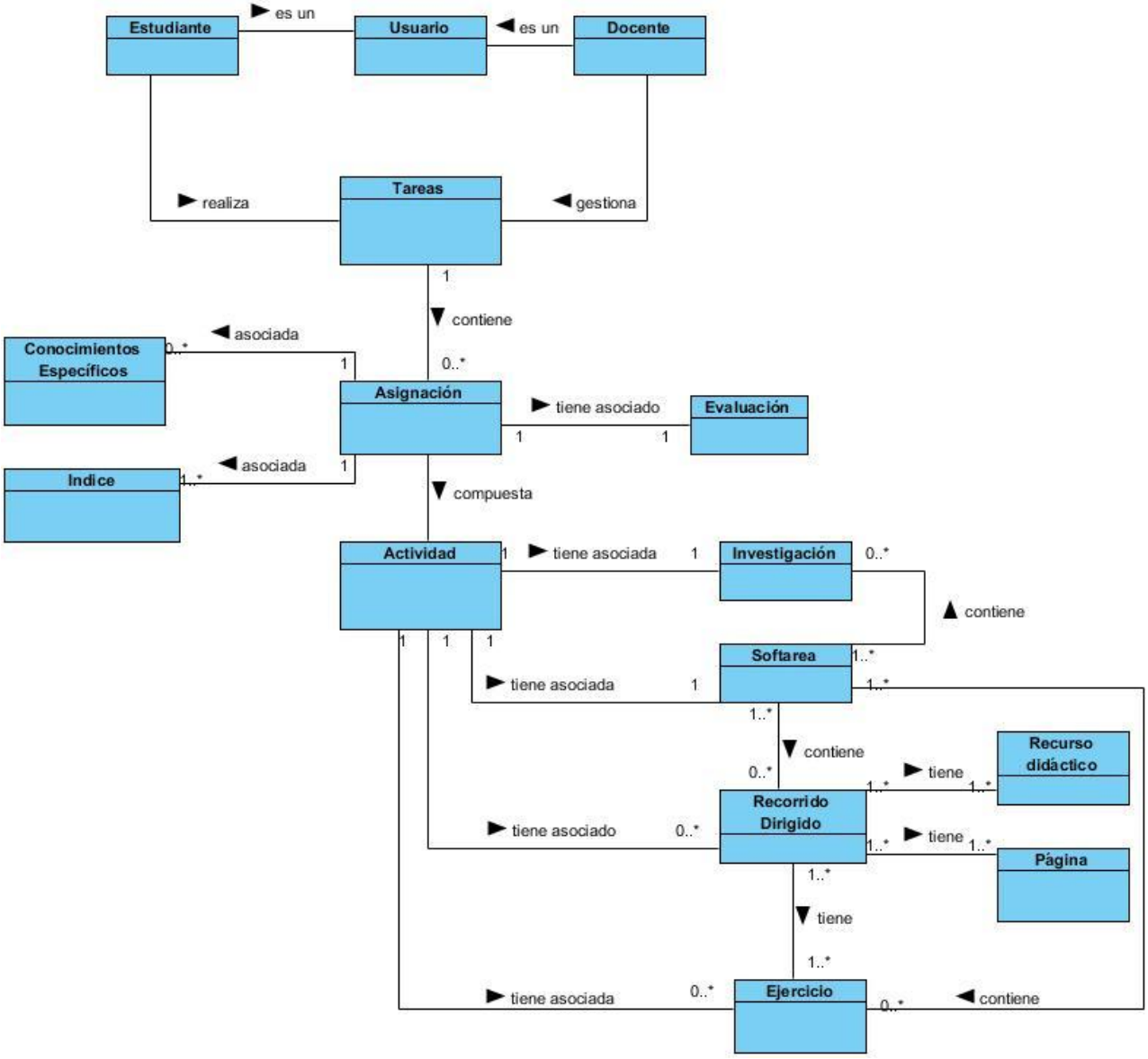


Figura 2 Modelo de Dominio

2.2Descripción de la solución propuesta

Al módulo Docente se integrarán nuevas funcionalidades, teniendo como objetivo principal el facilitarle al docente la personalización de tareas de acuerdo a las diferencias individuales de cada estudiante. Para ello se asociará a conocimientos específicos y al índice (capítulos, temas y subtemas) de acuerdo a las necesidades que tenga el profesor.

La propuesta proporcionará funcionalidades que permitirán de manera amigable la gestión y asignación de tareas de manera personalizada. Entre estas funcionalidades se encuentran el permitir consultar el listado de estudiantes, brindando la posibilidad de analizar sus evaluaciones, de manera general o más específica. Se podrá verificar los contenidos cumplidos y no cumplidos, además un por ciento de avance del estudiante y del grupo en general, todo ello asociado a los contenidos que sean previamente seleccionados. Después de haber realizado un análisis de los estudiantes se podrán consultar las softareas ya creadas o crear una nueva, y de acuerdo a las diferencias individuales que tenga cada estudiante estas serán asignadas, notificando al estudiante para su solución. Se permitirá evaluar las softareas manual y automáticamente, además compartirlas cuando un profesor así lo desee.

Esta constituye una breve descripción de la solución propuesta, la cual será desarrollada con el sentido de auxiliar al docente en el proceso de enseñanza-aprendizaje a partir del uso de los componentes de la plataforma educativa Zera.

2.3Requerimientos del software

Los requerimientos de software son las condiciones o capacidades que debe cumplir el sistema para satisfacer las necesidades del cliente.

2.3.1 Requerimientos Funcionales

Los requerimientos funcionales (RF) se encargan de definir lo que el software debe hacer. Define los alcances del sistema en cuanto a acciones que debe realizar.

RF1: **Consultar estudiante**

RF1.1 Filtrar estudiante según criterio especificado: el docente de acuerdo a sus necesidades selecciona el criterio de búsqueda por el que desea filtrar los estudiantes.

RF1.2 Mostrar listado de estudiantes por criterios de búsqueda: después de haber seleccionado el filtro por el que se desea realizar la búsqueda, se mostrará el listado con las coincidencias encontradas.

RF1.3 Mostrar evaluaciones asociadas a contenidos: el docente podrá ver de acuerdo a los contenidos previamente seleccionados la cantidad que tiene cumplidos y no cumplidos.

RF1.4 Mostrar por ciento de avance: Se mostrará el por ciento de avance para cada estudiante de acuerdo a los contenidos seleccionados, además un por ciento general del grupo.

RF1.5 Ver evaluaciones asociadas a contenidos: El docente podrá analizar las evaluaciones de los estudiantes asociadas a los contenidos que sean previamente seleccionados.

RF2: **Asignar softarea**

RF2.1 Asignar softarea al estudiante: el docente consulta las actividades y selecciona las que considera necesarias. Posteriormente consulta el listado de estudiantes para decidir a quién será asignada la softarea.

RF2.2 Asignar softarea creada al estudiante: el docente consulta el listado de softareas, selecciona la que considere para realizar la asignación. Consulta el listado de estudiantes y selecciona aquellos a los que va asignar la softarea creada.

RF2.3 Enviar notificación: Cuando le sea asignada la softarea al estudiante se le enviará una notificación.

RF3: **Gestionar actividades**

RF3.1 Adicionar actividades al panel: el docente podrá adicionar a un panel las actividades por las que estará compuesta la softarea (recorrido dirigido, ejercicio, investigación).

RF3.2 Eliminar actividaddel panel: el docente podrá eliminar una actividad contenida en el panel, que considere no necesaria para la creación de la softarea.

RF3.3: Ordenar elementos: los elementos agregados al panel podrán ser organizados de acuerdo a las preferencias del docente.

RF3.4: Ponderar actividades: El docente tendrá la posibilidad de asociar a cada elemento el valor que considere, teniendo en cuenta la complejidad que posea cada una.

RF4: Consultar por contenido

RF4.1 Mostrar estadísticas asociadas a contenidos: después de haber seleccionado los contenidos en el árbol, se mostrará para cada contenido la cantidad de estudiantes que han cumplido y no.

RF4.2 Mostrar por ciento de avance: Se mostrará para cada estudiante el por ciento de avance, además un por ciento para el grupo en general.

RF4.3 Ver evaluaciones asociadas a contenidos:El docente podrá analizar los estudiantes que han cumplido y los que no han cumplido, los que no y en que contenido.

RF5: Consultar softarea

RF5.1 Filtrar listado de softareas asociadas a contenidos: El docente selecciona los contenidos que considere necesarios para su consulta.

RF5.2 Mostrar listado de softareas asociadas acontenidos: Después de haber seleccionado los contenidos que considere el docente se muestran las softareas asociadas.

RF6: Gestionar softarea: este caso de uso ya se había realizado, pero fue modificado y se le agregó el RF6.1.

RF6.1 Compartir softarea: Permitir compartir las softareas creadas por un docente para los demás.

RF6.2 Crear softarea: El docente tiene la posibilidad de crear una nueva softarea para ser asignada en un momento determinado. Dicha tarea va a estar compuesta por un recorrido dirigido, ejercicios y evidencia.

RF6.3 Ver softarea: El profesor podrá ver los datos de la softarea cuando lo considere, así como darse cuenta si necesita cambiarle algo.

RF6.4 Modificar softarea: El docente podrá modificar los datos de la softarea una vez que sea consultada. La softarea será guardada con sus datos nuevos y publicada.

RF6.5 Eliminar softarea: El docente tendrá la posibilidad de ocultar la softarea que desee.

RF7: Evaluar softarea

RF7.1 Enviar notificación: Permite enviar una notificación al estudiante cuando se le evalúa la tarea.

RF7.2 Generar nota automática: Permitir una nota automática del sistema después de realizada la softarea orientada por el profesor.

RF7.3: Generar nota manual: permitir introducir una nota final después de realizada la softarea orientada por el profesor.

RF8: Resolver softarea. El estudiante tendrá un especio donde podrá ver la tareas que se les han sido asignadas por parte del profesor. Dicha tarea será evaluada.

2.3.2 Requerimientos no funcionales

Los requerimientos no funcionales (RNF) son propiedades o cualidades que el producto debe tener. Son las características del producto que se va a desarrollar.

RNF de Software:

- ✓ Los ordenadores deben tener instalado el navegador Mozilla Firefox 3.6 o superior, Internet Explorer 7 o superior, Chrome 10.0 o superior, Opera 10.6 o superior o Safari 5.0 o superior.

RNF de Hardware:

- ✓ Procesador Pentium 233 MHz (recomendado 500 MHz o mayor).
- ✓ 512 MB de RAM o superior.
- ✓ 1 GB de espacio en disco duro.
- ✓ Soporte de video que admita resolución de al menos 800x600 y 24 bits.
- ✓ Dispositivo de red de al menos 10 Mbits.

RNF de Diseño e Implementación:

- ✓ Lenguaje de programación: PHP 5.3
- ✓ El marco de trabajo base de desarrollo que se utilizará es: Symfony 1.4.15
- ✓ Como IDE se empleará NetBeans 7.1
- ✓ Como servidor Web se explotará Apache 2.2.6
- ✓ El SGBD deberá ser PostgreSQL 9.0

RNF de Apariencia o Interfaz Externa:

La interfaz debe ser amigable y sugerente para el usuario, la información será mostrada con claridad y organización. Diseño sencillo, con pocas imágenes y gráficos, para que la ejecución de las acciones se realice de manera rápida.

RNF de Seguridad y Confiabilidad:

La plataforma debe asegurar en todo momento, la seguridad de los contenidos, negando que estos sean copiados, garantizando que la información no sea modificada, sino es por el que está autorizado para ello. Tener en cuenta mensajes de eliminación sobre acciones que se realicen de

manera irreversible.

El sistema limitará el acceso no autorizado a los usuarios, creando niveles de acceso para los diferentes roles dentro de la plataforma a través de una clave. Cuando un usuario, sin importar su rol, permanece inactivo durante 20 minutos, se cerrará la sesión automáticamente.

RNF de Usabilidad:

Los usuarios deberán tener conocimientos básicos del manejo de las computadoras y los sistemas operativos. Además la aplicación deberá contar con una interfaz y navegación funcional, amena, asequible, para todo tipo de usuario. Se debe tener acceso al menú general desde cualquiera de las páginas. Los elementos gráficos como los íconos deberán contar con un tooltip o mensaje flotante que señalen el tipo de recurso al que se refiere. Se deben mostrar las rutas de acceso según la navegación que tenga el usuario.

RNF de Portabilidad:

- ✓ El sistema podrá utilizarse en cualquier sistema operativo.

RNF Legales, Derecho de Autor y otros:

- ✓ Una vez terminado el producto, el sistema debe ser sometido a una evaluación y certificación por parte del cliente del producto.

RNF de Disponibilidad:

- ✓ Los usuarios según los permisos definidos deben tener acceso en todo momento a la información que sea solicitada.

2.5Modelo de Casos de Uso del Sistema

2.5.1 Descripción de los actores del sistema

Actor	Descripción
Docente	El actor con el rol docente es el encargado gestionar las tareas docentes y asignarlas de acuerdo a contenidos comprendidos en el árbol (conocimientos específicos, capítulos, temas y subtemas). El profesor

	podrá consultar las notas de sus estudiantes asociadas a contenidos y asignar las tareas atendiendo lasdiferencias individuales.
Estudiante	El actor con el rol docente es el encargado de realizar las softareas asignadas por el docente y visualizadas en el módulo Tareas, para luego ser evaluadas según su desempeño.

Tabla 1 Descripción de los actores del Sistema

2.5.2 Patrones de Casos de uso

Un patrón se define como la pareja de problema-solución con un nombre, que codifica buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades.

Los patrones de casos de uso son comportamientos que deben estar presentes en el sistema, ayudan a describir qué es lo que el sistema debe hacer, el uso del sistema y cómo este interactúa con los usuarios. Estos patrones son utilizados generalmente como plantillas que describen cómo deberían ser estructurados y organizados los casos de uso. Son patrones que capturan mejores prácticas para modelar casos de uso. (21)

A continuación se explican los patrones presentes en el diagrama de casos de uso del sistema: (21)

Múltiples actores

Dentro de este patrón se encuentra el de roles comunes, en éste puede suceder que los dos actores jueguen el mismo rol sobre el caso de uso. 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.

CRUD (Creating, Reading, Updating, Deleting)

Completo: Este patrón se basa en la fusión de casos de uso simples para formar una unidad conceptual. 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. Este patrón se ve reflejado en el CU Gestionar softarea.

Patrón Concordancia

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

Adición: La subsecuencia común de casos de uso, extiende los casos de uso compartiendo la subsecuencia de acciones. Los otros casos de uso modelan el flujo que será expandido con la subsecuencia. Este patrón es preferible usarlo cuando otros casos de uso se encuentran propiamente completos, o sea, que no requieren de una subsecuencia común de acciones para modelar los usos completos del sistema. Un ejemplo lo constituye el CU Consultar estudiante y el CU Asignar softarea y CU Gestionar softarea.

Reuso: Consta de 3 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. Un ejemplo de su uso se evidencia en el CU Evaluar softarea y los dos casos de uso incluidos Consultar estudiante y Consultar softarea.

Patrón extensión concreta o inclusión

Este patrón está dividido en concreta extensión o concreta inclusión.

Concreta inclusión: Se incluye una relación del caso de uso base al caso de uso de inclusión. El último puede ser instalado en sí mismo. Un ejemplo lo muestra el CU Asignar softarea que incluye el CU consultar softarea.

2.5.3 Diagrama de actores del sistema

A continuación se muestra el diagrama de actores, en el que se representa una relación de generalización – especialización, donde al actor Usuario constituye la generalización de los actores Estudiante y Docente, los cuales son la especialización.

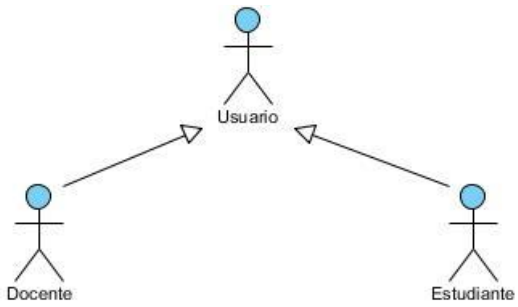


Figura 3 Diagrama de actores del sistema.

2.5.4 Diagrama de casos de uso del sistema

El diagrama de casos de uso del sistema muestra las relaciones que existen entre actores y casos de uso, así como sus relaciones. Se representan a continuación ocho casos de uso, basados en requisitos funcionales y dos actores, el docente podrá gestionar las softareas, consultarlas de acuerdo al árbol de índices o de conocimientos específicos. Para la creación de una asignación tendrá la posibilidad consultar los estudiantes y sus evaluaciones, así como la opción para consultar las softareas asociadas a sus objetivos trazados. Una vez resuelta la tarea por el estudiante el docente podrá evaluarla.

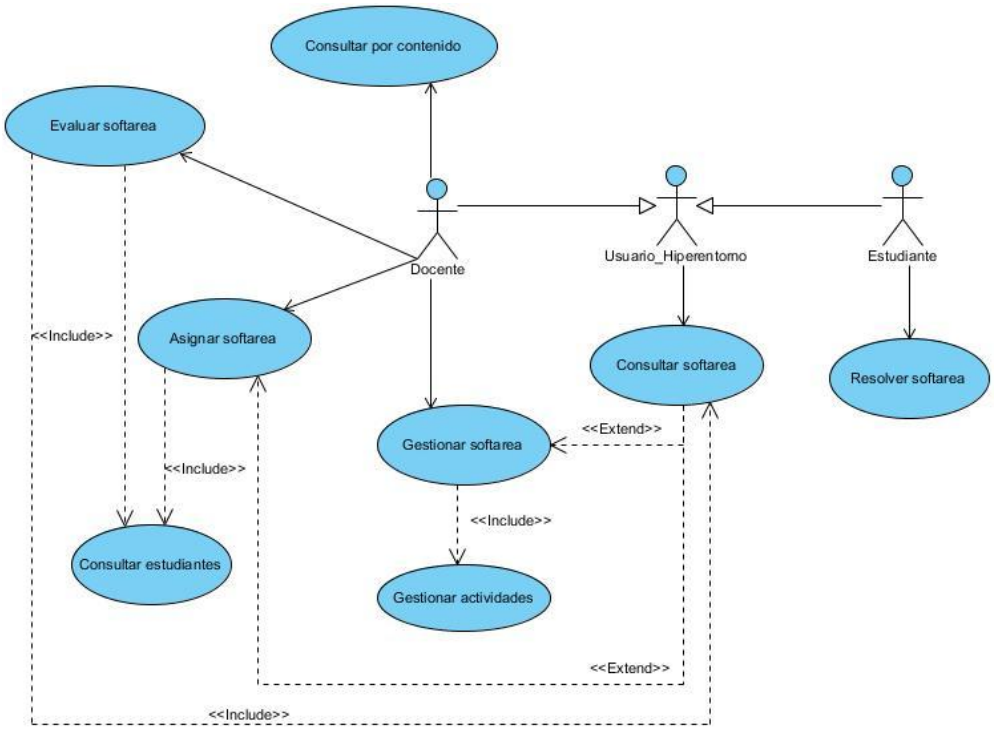


Figura 4 Diagrama de CUS

2.5.4 Descripción de los casos de uso del sistema

A continuación se presenta la descripción del caso de uso Asignar softarea.

CU Asignar softarea	
Objetivo	Asignar softarea al estudiante de acuerdo contenidos (conocimientos específicos e índices).
Actores	Docente
Resumen	El caso de uso inicia cuando el actor selecciona la opción "Asignar",

	posteriormente se le mostrarán las softareas creadas, permitiendo seleccionar una para su asignación. Brindará la posibilidad de seleccionar los estudiantes a la que será asignada, verificar sus notas y de acuerdo a los objetivos que persigue el profesor, la tarea será asignada. Termina el caso de uso.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	El docente ha sido autenticado. Se ha generado el escritorio de trabajo. Se ha accedido a Gestionar softarea/Asignar Se han seleccionado los estudiantes a los que va a ser asignada la softarea. Se han seleccionado la softarea a asignar.
Postcondiciones	Se ha asignado la softarea a los estudiantes seleccionados.

Flujo de eventos

Flujo básico Asignar Softarea

	Actor	**Sistema**
1.	Selecciona la softarea para realizar la asignación.	
2.		Muestra datos de la softarea. Ver Descripción del CU Consultar softarea. Además muestra una opción que permite asignar la softarea.
3.	Selecciona la opción que le permite asignar softarea.	
4.		Muestra listado de estudiantes, permitiendo seleccionarlos para la asignación de la softarea. Ver CU Consultar estudiante.
5.	Selecciona estudiantes a los que va a asignar la softarea.	

6.		Muestra datos de los estudiantes. Además una opción que permite asignar la softarea a los estudiantes que hayan sido seleccionados.
7.	Selecciona la opción Guardar softarea.	
8.		Permite insertar los datos: • Fecha de inicio. • Fecha fin. Brinda la opción de Asignar.
9.	Selecciona la opción que le permite Asignar.	
10.		Asigna la softarea a los estudiantes seleccionados. Envía notificación al estudiante. Termina el caso de uso.
Flujos alternos		
1 a. Selecciona Nueva Softarea		
	Actor	**Sistema**
1.		Ver CU Gestionar Softarea.

Tabla 2 Descripción del CU Asignar softarea

Conclusiones

Con la realización del presente capítulo se logró desarrollar un análisis de los principales conceptos involucrados en el negocio y se concretó la propuesta de solución del sistema. Los artefactos que se generaron en los flujos de trabajo Modelamiento del Negocio y Requerimientos servirán como punto de partida al flujo de Análisis y Diseño, teniendo un mejor dominio del negocio gracias al modelo de dominio generado. Los requisitos funcionales y no funcionales permitieron definir las cualidades y capacidades que el sistema debe cumplir. Se lograron obtener las relaciones entre casos de uso y actores y para un mejor entendimiento las descripciones de casos de uso.

Capítulo 3: Análisis y diseño de las funcionalidades a implementar

Introducción

La metodología RUP en su tercer flujo de trabajo Análisis y Diseño genera los artefactos: Modelo del Análisis, donde se identifican las clases del análisis y se realizan los diagramas de clases y el Modelo de Diseño, encargado de realizar los diagramas de clases con estereotipos web. Se define el patrón arquitectónico y de diseño a utilizar, además se genera el modelo de entidad relación de la base de datos y el diagrama de despliegue.

3.1 Modelo del análisis

En esta etapa se obtiene una visión de lo que el sistema debe hacer, el análisis de requisitos permite al desarrollador especificar la función y el rendimiento del software. Se pretenden traducir los requisitos a una especificación que describe cómo implementar el sistema.

A continuación se explican los estereotipos presentes en los diagramas de clases del análisis (DCA):

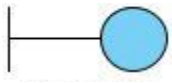
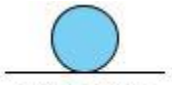
Estereotipo	Características
 Clase Interfaz	Modelan la interacción entre el sistema y sus actores.
 Clase Controladora	Coordinan la realización de uno o unos pocos casos de uso coordinando las actividades de los objetos que implementa la funcionalidad del caso de uso.
 Clase Entidad	Modelan información que posee larga vida y que es a menudo persistente.

Tabla 3 Estereotipos presentes en el diagrama de DCA

3.1.1 Diagramas de clases del análisis

Los diagramas de clases del análisis son la entrada para la etapa de diseño y representan la relación que existe entre clases y casos de uso.

A continuación se representa DCA del caso de uso Consultar Estudiante.

DCA CU Asignar softarea

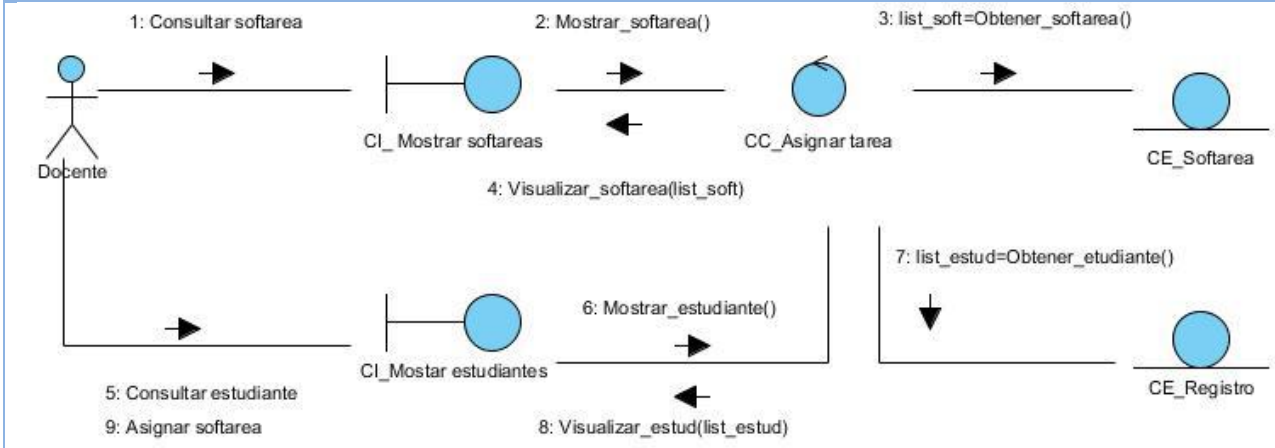


Tabla 4 DCA_CU Asignar softarea

3.2 Patrones arquitectónicos y de diseño

Según la escala a nivel de abstracción se encuentran los patrones arquitectónicos y los patrones de diseño. Los primeros expresan un esquema organizativo estructural fundamental para sistemas software y los segundos expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas software.

3.2.1 Patrón arquitectónico Modelo-Vista-Controlador

Symfony está basado en el patrón de arquitectura Modelo Vista Controlador (MVC), formada por tres niveles:

- ✓ El modelo representa la información con la que trabaja la aplicación, es decir, la lógica del negocio.
- ✓ La vista transforma el modelo en una página web, permitiendo al usuario interactuar con ella.

- ✓ El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

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. Ejemplo de ello, en caso de que se desee ejecutar una misma aplicación 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. (33)

A continuación se muestra una representación gráfica del MVC

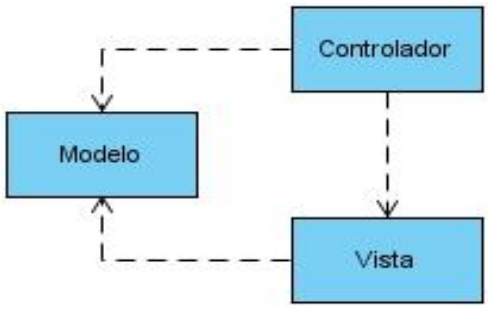


Figura 5 Estructura del patrón arquitectónico Modelo-Vista-Controlador (MVC)

3.2.2 Patrones de Diseño en Symfony

Patrones GOF (35)

Singleton (instancia única): se utiliza para garantizar que determinada actividad solo tenga una instancia, proporcionando un punto de acceso global a ella, lo cual permite y hace extensible su uso sin modificar su estructura.

Decorator (Decorador): se encarga de extender la funcionalidad de un objeto dinámicamente, 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. Se utiliza para manejar la modalidad en que se presentan las actividades, permitiendo no tener que crear sucesivas clases que hereden de la primera sino otras que se asocien a esta.

Creador: se usa para establecer el responsable de la creación de objetos en una clase determinada. Es utilizado en la mayoría de las acciones, en la creación de objetos entidades, de formularios y de clases de abstracción a la base de datos. En la clase Actions se encuentran las acciones definidas para el sistema y se ejecutan en cada una de ellas.SE utiliza para crear los objetos de las clases que representan las entidades.

Observer (Observador): define una dependencia de uno a mucho entre objetos, de forma que cuando uno cambia de estado se notifique y actualicen automáticamente todos los objetos que dependen de él. Se utiliza cuando asignamos una tarea o actividad, modificando el estado de esta, así como actualizando el estado de los usuarios a los que le son asignados.

Strategy (Estrategia): define un grupo de clases que representan un conjunto de posibles comportamientos. Dichos comportamientos pueden ser intercambiados en una aplicación, brindando la posibilidad de modificar la funcionalidad cuando se desee. Se utiliza para modificar el comportamiento de las funcionalidades de acuerdo a los usuarios que acceden.

Patrones GRASP (Patrones para Asignar Responsabilidades) (35)

Experto: Asigna responsabilidades al experto en información, es decir a la clase que cuenta con la información necesaria para cumplir la responsabilidad. Con el uso de este patrón se conserva el encapsulamiento, ya que los objetos se valen de información propia para hacer lo que se les pide. Esto soporta un bajo acoplamiento.
Creador: Asigna responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador para conectarlo con el objeto producido en cualquier evento.

Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada con otras clases. Acoplamiento bajo significa que una clase no depende de muchas clases. El grado de acoplamiento no puede considerarse aisladamente de otros principios como Experto y Alta Cohesión.

Alta Cohesión: La cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un alto trabajo.

Controlador: Un controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Un defecto frecuente al diseñar controladores consiste en

asignarles demasiada responsabilidad. Un controlador normalmente debería delegar a otros objetos el trabajo que ha de realizarse mientras coordina la actividad.

3.3 Modelo de Diseño

3.3.1 Diagrama de clases del diseño

Los diagramas de clases del diseño son una representación más concreta y detallada que los diagramas de clases del análisis, aunque también representan la parte estática del sistema conteniendo las clases y sus relaciones. Son empleados para representar las relaciones que se establecen entre las clases.

A continuación se presentan el diagrama de clases del diseño correspondiente al caso de uso Asignar Softarea.

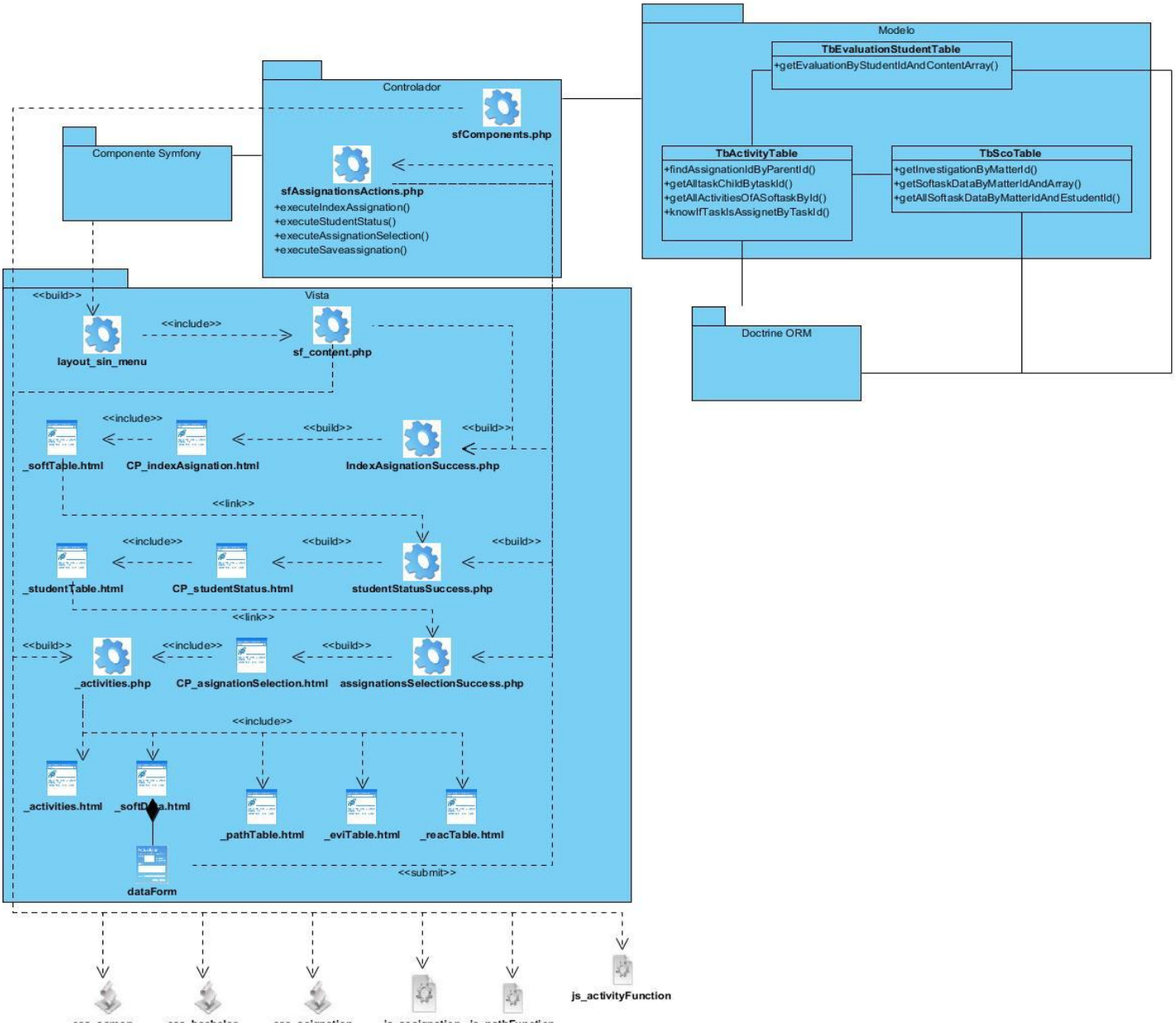


Figura 6Diagrama de clases del diseño CU Asignar softarea

3.4 Diseño de la base de datos

En esta sección se presenta el modelo de datos que estará compuesto por las entidades que pasarán a ser las tablas de la base de datos, para ser utilizadas por las funcionalidades a desarrollar. Para una mejor visualización el modelo se encuentra simplificado y solo se muestran las tablas que se van a utilizar.

Figura 7 Modelo de datos

Conclusiones

Para dar cumplimiento a los requisitos funcionales y no funcionales se generaron los diagramas de clases del análisis y del diseño, logrando una relación más concreta y detallada entre clases. Además los patrones de diseño y arquitectónicos utilizados en el desarrollo de las funcionalidades quedaron bien definidos. Se realizó el modelo de base de datos, evidenciando las relaciones entre las tablas que forman parte de la investigación y el diagrama de despliegue permitiendo obtener de manera detallada los nodos físicos y las asociaciones de comunicación que existen entre ellos.

Capítulo 4: Implementación y Pruebas de las Funcionalidades

Introducción

Los artefactos generados durante el flujo de trabajo análisis y diseño constituyen la entrada fundamental al flujo de trabajo de implementación, el cual describe cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue. Para poner a prueba los componentes desarrollados, se comienza a realizar la evaluación de la calidad del producto, para ellos es necesario que el producto de software funcione como está previsto y que la validación de los requisitos se aplique correctamente.

4.1. Modelo de Implementación

El modelo de implementación describe cómo los elementos del modelo de diseño, como las clases, se implementan en términos de componentes, ficheros de código fuente, ejecutables, etc. El modelo de implementación describe como se organizan los componentes de acuerdo a los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y como dependen unos de los otros. (35)

4.1.1. Diagramas de componentes

El diagrama de componentes muestra un conjunto de elementos del modelo tales como componentes, subsistemas de implementación y sus relaciones. Se utiliza para modelar la vista estática de un sistema, además muestra la organización y las dependencias lógicas entre un conjunto de componentes de software (componentes de código fuente, librerías, binarios o ejecutables).

Un componente es una parte física y reemplazable del sistema que cumple y proporciona la realización de un conjunto de interfaces.

Los subsistemas de implementación constituyen una colección de componentes y otros subsistemas de implementación usados para estructurar el modelo de implementación y dividirlos en pequeñas partes que pueden ser integradas y probadas de forma separada. (35)
Los diagramas de componentes realizados, muestran cómo están distribuidos según el patrón arquitectónico Modelo-Vista-Controlador que utiliza Symfony.
A continuación se presenta el diagrama de componentes del caso de uso Asignar softarea.

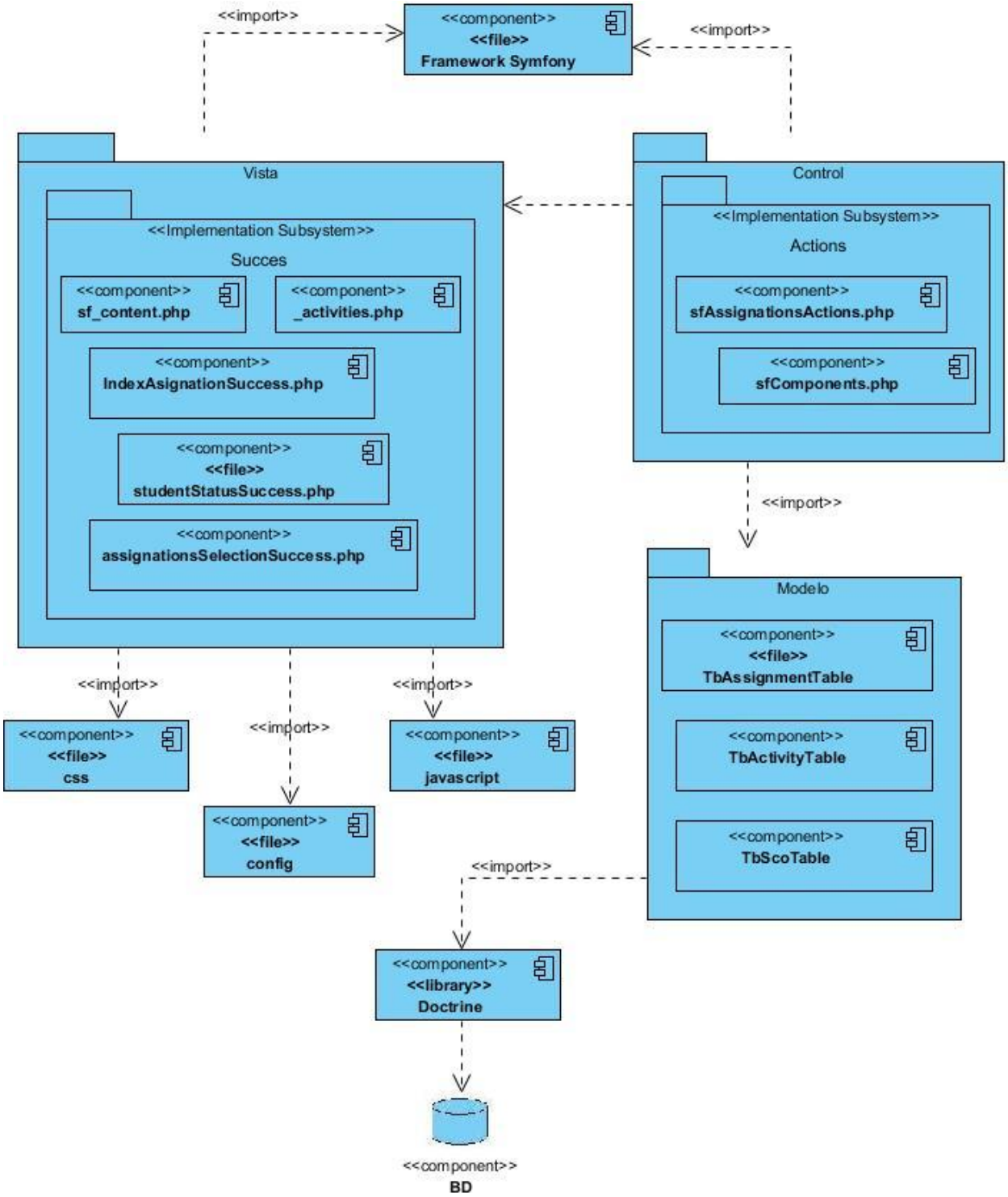


Figura 8 Diagrama de componentes CU Asignar softarea

4.2. Pruebas de software

La realización de las pruebas es una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos específicos. El objetivo de las pruebas al sistema es comprobar la integración del sistema, verificando el funcionamiento de las interfaces interactuando entre los distintos subsistemas que lo componen. Es de suma importancia validar que el software trabaje como fue previamente diseñado, probar los requisitos que debe cumplir el software y que hayan sido implementados correctamente.

4.2.1. Niveles de Prueba

Las pruebas pueden realizarse en diferentes escenarios o niveles de trabajo dependiendo del objetivo de los mismos. Toman un enfoque incremental, inicia con las pruebas de unidades individuales del programa, pasa a pruebas diseñadas para facilitar la integración de las unidades y culmina con las pruebas que se realizan sobre el sistema construido.

Pruebas unitarias

Las pruebas unitarias están enfocadas a los elementos testeables más pequeños del software, son una forma de probar el correcto funcionamiento de un módulo de código, sirviendo para asegurar que cada uno de los módulos 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 cinco ventajas básicas: fomentan el cambio, simplifica la fase de integración, documenta el código, separan la interfaz y la implementación y los errores están más acotados y son más fáciles de localizar. (36)

Pruebas de integración

Las pruebas de integración, también conocidas como pruebas integrales son aquellas que se realizan en el ámbito de desarrollo de software una vez que se han aprobado las pruebas unitarias. Únicamente se refieren a la prueba o pruebas de todos los elementos unitarios que componen un proceso, hecha en conjunto, de una sola vez. Consiste en realizar pruebas para verificar que un gran conjunto de partes del software funcionan juntos. Son las pruebas posteriores a las pruebas unitarias y preceden a las pruebas del sistema. (36)

Pruebas del sistema

Las pruebas del sistema se realizan cuando el software está funcionando como un todo. Toda 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. Es la actividad de prueba dirigida a verificar el programa final, después que todos los componentes de software y hardware han sido integrados. (36)

4.2.2. Métodos de prueba

RUP define dos métodos de pruebas: caja negra y caja blanca, a continuación se describen ambas, haciendo más énfasis en las pruebas de caja negra, donde basándonos en los casos de prueba se comprueba la validez de las funcionalidades de acuerdo a las acciones del usuario, entradas y respuestas del sistema:

Pruebas de caja blanca

La prueba de caja blanca, también denominada prueba de caja de cristal, es un método de diseño de casos de prueba que utiliza la estructura de control descrita como parte del diseño a nivel de componentes para obtener los casos de prueba. Mediante el uso de los métodos de caja blanca el ingeniero de software puede obtener casos de prueba que garanticen que se ejerciten todas las rutas independientes dentro de cada módulo al menos una vez, ejerciten las decisiones lógicas en sus vertientes verdadera y falsa; ejecuten todos los bucles en sus límites y dentro de sus límites operacionales, y ejerciten las estructuras internas de datos para asegurar su validez. (37)

Pruebas de caja negra

Las pruebas de caja negra, también denominadas, pruebas de comportamiento, se concentran en los requisitos funcionales del software. Permiten al ingeniero de software derivar un conjunto de condiciones de entrada que ejercitarán por completo todos los requisitos funcionales de un programa. Dicha prueba no es una alternativa a las técnicas de caja blanca, sino un enfoque complementario que intenta descubrir diferentes tipos de errores que se llevan a cabo sobre la interfaz del software. El objetivo es demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y se produce un resultado correcto, y que la integridad de la información externa se mantiene. (37)

Las pruebas de caja negra permiten encontrar:

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

4.2.3. Diseño de Casos de Prueba

A continuación se presenta el diseño de casos de prueba propuesto para el CU Asignar softarea:

Diseño de Casos de Prueba: CU Asignar softarea

Descripción General

El actor selecciona la opción "Asignar", posteriormente se le mostrarán las softareas creadas, permitiendo seleccionar una para su asignación. Brindará la posibilidad de seleccionar los estudiantes a la que será asignada, verificar sus notas y de acuerdo a los objetivos que persigue el profesor la tarea será asignada.

Condiciones de Ejecución

El docente ha sido autenticado.

Se ha generado el escritorio de trabajo.

Se ha accedido a Gestionar softarea/Asignar

Se han seleccionado los estudiantes a los que va a ser asignada la softarea.

Se han seleccionado la softarea a asignar.

Escenario	Descripción	Softarea	Estudiante	Fecha de inicio	Fecha fin	Respuesta del sistema	Flujo central

EC 1.1 "Selecciona softarea"	Selecciona la softarea para realizar la asignación.	V				Muestra datos de la softarea. Ver Diseño de caso de prueba del CU Consultar softarea. Además muestra una opción que permite asignar la softarea.	Módulo Docente/Administración del Aprendizaje/ Gestión de Tareas/ Softarea/Softarea
EC 1.2 "Selecciona Asignar"	Selecciona la opción que le permite asignar softarea.					Muestra listado de estudiantes, permitiendo seleccionarlos para la asignación de la softarea.	Módulo Docente/Administración del Aprendizaje/ Gestión de Tareas/ Softarea/Asignar
EC 1.3 "Selecciona estudiantes."	Selecciona estudiantes a los que va a asignar la softarea.		V			Muestra datos de los estudiantes. Ver diseño de caso de prueba del CU Consultar estudiante. Además una opción que permite asignar la softarea a los estudiantes que hayan sido seleccionados.	Módulo Docente/Administración del Aprendizaje/ Gestión de Tareas/ Softarea/Asignar/Ver estado de los estudiantes
EC 1.4 "Selecciona Asignar"	Selecciona la opción que le permite introducir los datos.			V	V	Permite insertar los datos: • Fecha de inicio. • Fecha fin. Brinda la opción de Asignar.	Módulo Docente/Administración del Aprendizaje/ Gestión de Tareas/ Softarea/Asignar/Ver estado de los estudiantes/Asignar
EC 1.5 "Selecciona Asignar"	Selecciona la opción que le permite asignar una softarea.					Asigna la softarea a los estudiantes seleccionados. Envía notificación a los estudiantes.	Módulo Docente/Administración del Aprendizaje/ Gestión de Tareas/ Softarea/Asignar/Ver estado de los estudiantes/Asignar

EC 1.6 "Selecciona Nueva softarea"	Selecciona la opción que le permite crear una nueva softarea.				Ver Diseño de caso de prueba CU Gestionar softarea.	Módulo Docente/Administración del Aprendizaje/ Gestión de Tareas/ Softarea/ Nueva softarea

Tabla 5 Diseño de caso de prueba CU Asignar softarea

4.2.4. Resultados obtenidos

Durante el desarrollo de las funcionalidades para la gestión y asignación de tareas en el módulo Docente se realizaron pruebas unitarias al código para comprobar su correcta implementación. Dichas pruebas fueron realizadas por el desarrollador, apoyándose en las ventajas de compilación que brinda el IDE Netbeans 7.1. Las pruebas no fueron planificadas, tampoco se registraron sus resultados ya que se realizaron conjuntamente con el desarrollo de la solución.

Haciendo uso del método de caja negra y apoyándonos en el diseño de CP se realizaron dos iteraciones de pruebas internas pertenecientes al nivel de sistema. Dichas pruebas fueron realizadas por el equipo de calidad interna del proyecto, en este caso los analista; con el objetivo de obtener un producto con la menor cantidad de errores posibles.

Se realizaron dos iteraciones de pruebas de integración para validar la solución y comprobar su funcionamiento íntegramente. Además se comprobó la relación de las funcionalidades implementadas con el resto de los módulos de la plataforma.

A continuación se evidencian los resultados asociadas a las diferentes pruebas aplicadas:

Resumen de pruebas realizadas

Casos de Prueba	No Conformidades				
	Alta	Media	Baja	No Procede	Resueltas
Asignar softarea		1	2		3
Gestionar softarea	2		1		3
Consultar softarea		2	1		3
Consultar estudiante		1		1	2

Gestionar actividades	1				1
Consultar por contenido		1	1		2
Resolver softarea		2			2
Evaluar softarea		2			2

Tabla 6 ruebas internas

Casos de Prueba	No Conformidades				
	Alta	Media	Baja	No Procede	Total
Asignar softarea		2			2
Gestionar softarea	2		1		3
Consultar softarea				1	1
Consultar estudiante			3		3
Gestionar actividades			1		1
Consultar por contenido	1		2		3
Resolver softarea	1			1	1
Evaluar softarea		1			1

Tabla 7Pruebas de integración.

Conclusiones

Durante el flujo de trabajo de implementación se obtuvo el modelo de implementación, el cual describió cómo los elementos del modelo del diseño se implementan en términos de componentes y como estos son agrupados en subsistemas de implementación para una mayor organización. En esta etapa se alcanzó una solución visible como resultado a las propuestas definidas al inicio de la investigación. Pasando al flujo de pruebas, se validaron las funcionalidades implementadas que cumplieran con los requisitos especificados. Se realizaron pruebas unitarias y de integración y como resultado se obtuvieron no conformidades, las que fueron corregidas parcialmente.

Conclusiones

Una vez terminado el trabajo se concluye que:

- ✓ El estudio de las diferentes plataformas de aprendizajes permitió identificar y analizar las funcionalidades de las mismas, siendo esto de apoyo para la posterior implementación de la propuesta de solución.
- ✓ Se desarrolló un análisis de los principales conceptos involucrados en el negocio y estos fueron el punto de partida para concretar las funcionalidades de la propuesta de solución del sistema.
- ✓ Se generaron los diagramas de clases del análisis y del diseño para dar cumplimiento a los requisitos funcionales y no funcionales, logrando una relación más concreta y detallada entre clases.
- ✓ Se implementaron las funcionalidades que permiten a los docentes personalizar las softareas, de acuerdo a las diferencias individuales que posea cada estudiante. Asociando estas a objetivos, competencias y habilidades que persiga el profesor con determinada tarea.
- ✓ Después de haber realizado las iteraciones planificadas al software, se comprobó que cumple con la especificación de requisitos funcionales y no funcionales establecida. Además se realizaron pruebas unitarias y de integración y como resultado se obtuvieron no conformidades, las que fueron corregidas parcialmente.

Recomendaciones

Luego de haber apreciado los resultados obtenidos y basándose en la experiencia adquirida durante la realización de la investigación se recomienda:

- ✓ Aplicar el estándar IMS-SS (Secuenciación Simple) a la gestión y resolución de softareas.
- ✓ Tener presente entre las características principales que caracterizan a la plataforma el proceso de consulta del estado de los estudiantes, al ser esta una característica ausente en los sistemas similares.

Referencias Bibliográficas

1. **Peñalvo García, Francisco José.** *Universidad de Salamanca. España.* 2006.

2. **Rojas, Nemecio Núñez.** *La Webquest, el aula virtual y el desarrollo de competencias para la investigación en los estudiantes del ciclo de educación – USAT.* Universidad de Málaga : Biblioteca virtual de Derecho, Economía y Ciencias Sociales, 2011.

3. **S, O. y Salazar.** *El software educativo en el proceso de enseñanza aprendizaje,.* Coloma R : s.n., 2010.

4. **Julio Cabero Almenara.** Bases pedagógicas del e-learning. s.l. : Revista de Universidad y Sociedad del Conocimiento, 2006. Vol. 3, 1.

5. **Guzmán, Clara López Peñalvo and García, Francisco J.** *Estándares y Especificaciones para los Entornos e-learning: Convergencia en Contenidos y Sistemas.* Universidad de Salamanca (España) : s.n., 2006.

6. **Marcelo García, C.** E-learning en la formación para el empleo: ¿qué opinan los usuarios? s.l. : Revista de Educación, 2008.

7. **Ilabaca, Jaime Sánchez.** *Construyendo y aprendiendo con el computador.* Universidad de Chile : s.n., 2006.

8. **Vidal Ledo, María and Gómez Martínez, Freddy.** *Software educativos.* 2010.

9. **EVO I.T.** *La agenda de un sistema de gestión del aprendizaje.* [Online] marzo 21, 2011. http://blog.evoit.com/tag/sistema-de-gestion-del-aprendizaje/.

10. **Cañellas, Alicia.** *CMS, LMS y LCMS. Definición y diferencias.* 2011.

11. **Mager, Robert Frank.** *Preparing Instructional Objectives. (2da edición).* belmont : s.n., 1984.

12. **Sánchez, R. Blanco.** Ediciones de la univercidad de Murcia (Editum). *Ediciones de la univercidad de Murcia (Editum).* [Online] 2007. http://revistas.um.es/eglobal/article/view/347.

13. **Pecquet , Emmanuel.** *Creando y publicando cursos virtuales con Dokeos 1.8.* 2007.

14. **Moodle.** Moodle. [Online] 2008. http://docs.moodle.org/all/es/Caracter%C3%ADsticas.

15. **Gomez, Jesús Martín.** *Moodle 1.5 -Manual de consulta.* 2006 .

16. **Argueta, Roberto.** *Claroline Manual del profesor.* 2006.

17. **Buenas Tareas.** Introducción a La Plataforma Blackboard. [Online] 2011. http://www.buenastareas.com/ensayos/Introduccion-a-La-Plataforma-Blackboard/2151319.html.

18. **Acuña, Karenny Brito.** *Selección de Metodologías de Desarrollo para Aplicaciones Web .* Universidad de Cienfuegos : s.n., 2009.

19. **Bonillo, Pedro.** *Metodología para la gerencia de los procesos del negocio sustentada en el uso de patrones.* Centro ISYS, Facultad de Ciencias, UCV, Caracas, Venezuela : s.n., 2006.

20. **Mitaritonna, Alejandro Daniel.** *CAPA-3: Una innovadora metodología para el desarrollo de software en ambientes de trabajo virtuales.* Universidad Tecnológica Nacional Facultad Regional Buenos Aires : s.n., 2010.

21. **Orallo, Enrique Hernández.** *El Lenguaje Unificado de Modelado(UML).* 2006.

22. **Instituto nacional de estadística e informática(INEI).** *Herramienta CASE.*

23. **Visual Paradigm International.** Visual Paradigm. *Visual Paradigm.* [Online] 2004. http://www.visual-paradigm.com.

24. **Ministerio de educación, cultura y deporte.** Observatorio Tecnologico. [Online] junio 2008.

http://recursostic.educacion.es/observatorio/web/es/software/servidores/580-elvira-mifsud.

25. **Apache Software Fundation.** Apache MPM worker. *Apache MPM worker.* [Online] 2011. http://httpd.apache.org/docs/2.0/mod/worker.html.

26. **Netcraft LTD.** Netcraft. *June 2012 Web Server Survey.* [Online] 6 2012. http://news.netcraft.com/archives/2012/06/06/june-2012-web-server-survey.html.

27. **Pérez, Javier Eguíluz.** *Introducción a XHTML.*

28. **Javier Eguíluz Pérez.** *Introducción al CSS.* 2009.

29. **Pérez, Javier Eguíluz.** *Introducción a JavaScript.*

30. **Parts, JavaScript: The Good.** *Douglas Crockford.* 2008.

31. **Javier Eguíluz Pérez.** *Introducción al Ajax.* 2009.

32. **autores, Colectivo de.** *Manual PHP.* 2012.

33. **Potencier, Fabien and Zaninotto, Francois.** *Symfony la guía definitiva.* 2008.

34. **Góngora, Alex.** NetBeans IDE. [Online] 2011. http://netbeans-ide.uptodown.com/..

35. **CRAIG, LARMAN.** *UML y patrones.* s.l. : Pearson, 2003.

36. **Rumbaugh James, Jacobson Ivar, Booch Grady.** *El lenguaje Unificado de Modelado. .* s.l. : Addison Wesley, 2010.

37. **Pressman, Roger S.** *Ingeniería del Software. Un enfoque práctico.* España : McGraw-Hil, 2002.

38. Enciclopedia Colaborativa Cubana. Hiperentornos de aprendizaje . [Online] 2011. http://www.ecured.cu/index.php/Hiperentorno_de_aprendizaje..

39. **Baltasar Fernández Manjón, José Luis Sierra Rodríguez, Iván Martínez Ortiz y Pablo Moreno Ge.** *Estandarización y Diseño Educativo.* 2009.

40. **Córdova, Carlos H. Alfaro- Jackeline M.** *Una Revisión de Metodologías de Software Educativo.* Universidad Inca Garcilaso de la Vega : s.n., 2007.

41. **Moodle.** [Online] 2011. http://moodle.org/about/.

42. **Canós, José, Letelier , Patricio and Penadés, Carmen.** *Métodologías Ágiles en el Desarrollo de Software.* DSIC -Universidad Politécnica de Valencia : s.n., 2006.

43. **Ramirez, Philippe Alejandro and Rueckert, Andreas.** ArgoUML: Manual de Usuario. [Online] [Cited: 12 20, 2011.] http://argouml-stats.tigris.org/nonav/daily-userdoc/es/printablehtml/manual/manual.html#d0e217..

44. W3C. [Online] [Cited: 1 4, 2012.] http://www.w3c.es/divulgacion/guiasbreves/XHTML#masinfo..

45. **ADR Infor S.L.** Soluciones integrales de e-learning para instituciones y empresas. [Online] [Cited: 1 4, 2011.]

46. **James Rumbaugh, Ivar Jacobson, Grady Booch.** *El Proceso Unificado de Desarrollo de Software. Manual de Referencia. s.l.* s.l. : Addison Wesley., 2010.

47. **Turner, Stephen.** *Analog.* 1996.

48. Definicion.org. [Online] www.definicion.org.

49. **Chase Ideas.** Apache.com. [Online] 2008. http://www.apache.com/.

50. **Pérez, Javier Eguíluz.** *Introducción a XHTML.*

51. **Alvarez, Miguel Angel.** *Taller CSS.* 2008.

Bibliografía

Acuña, Karenny Brito.*Selección de Metodologías de Desarrollo para Aplicaciones Web .* Universidad de Cienfuegos : s.n., 2009.

ADR Infor S.L. Soluciones integrales de e-learning para instituciones y empresas. [] [Citado: 1 4, 2011.]

Alvarez, Miguel Angel. *Taller CSS.* 2008.

Argueta, Roberto.*Claroline Manual del profesor.* 2006.

Baltasar Fernández Manjón, José Luis Sierra Rodríguez, Iván Martínez Ortiz y Pablo Moreno Ge.*Estandarización y Diseño Educativo.* 2009.

Buenas Tareas. Introducción a La Plataforma Blackboard. [] 2011. http://www.buenastareas.com/ensayos/Introduccion-a-La-Plataforma-Blackboard/2151319.html.

Canós, José, Letelier , Patricio and Penadés, Carmen.*Métodologías Ágiles en el Desarrollo de Software.* DSIC -Universidad Politécnica de Valencia : s.n., 2006.

Córdova, Carlos H. Alfaro- Jackeline M.*Una Revisión de Metodologías de Software Educativo.* Universidad Inca Garcilaso de la Vega : s.n., 2007.

Ecured. [] 2011. [Citado: 12 10, 2011.] http://www.ecured.cu/index.php/Herramienta_CASE.

Enciclopedia Colaborativa Cubana. Hiperentornos de aprendizaje . [] 2011. http://www.ecured.cu/index.php/Hiperentorno_de_aprendizaje.

James Rumbaugh, Ivar Jacobson, Grady Booch. El Proceso Unificado de Desarrollo de Software: s.n.,2010

Ilabaca, Jaime Sánchez.*Construyendo y aprendiendo con el computador.* Universidad de Chile : s.n., 2006.

Martin, Odell.*Métodos orientados a objetos.* Ingeneria de software 2. Ciudad de la habana : s.n., 2009.

Martínez, Alejandro Martínez y Raúl. Guía a Rational Unified Process. Escuela Politécnica Superior de Albacete – Universidad de Castilla la Mancha : s.n.,2006

Moodle. [] 2011. http://moodle.org/about/.

Orallo, Enrique Hernández.*El Lenguaje Unificado de Modelado(UML).* 2006.

Peñalvo García, Francisco José.*Universidad de Salamanca. España.* 2006.

Pecquet , Emmanuel. Manual del docente. [En linea] 2007. http://www.slideshare.net/adrysilvav/dokeos-manual-profesor-2454013.

Potencier, Fabien and Zaninotto, Francois.*Symfony la guía definitiva.* 2008.

Ramirez, Philippe Alejandro and Rueckert, Andreas. ArgoUML: Manual de Usuario. [En linea] [Citado: 12 20, 2011.] http://argouml-stats.tigris.org/nonav/daily-userdoc/es/printablehtml/manual/manual.html#d0e217..

W3C. [] [Citado: 1 4, 2012.] http://www.w3c.es/divulgacion/guiasbreves/XHTML#masinfo..

JavaScript.

Glosario de Términos

Plataforma Educativa

De acuerdo con un estudio realizado por Facundo Ortega, catedrático de la universidad de Córdoba, Argentina, una plataforma educativa es no es más que: "… una herramienta ya sea física o virtual que brinda la capacidad de interactuar con uno o varios usuarios con fines pedagógicos. Además, se considera un proceso que contribuye a la evolución de los procesos de aprendizaje y enseñanza, que complementa o presenta alternativas en los procesos de la educación tradicional. En la actualidad, la mayor parte de las plataformas educativas son programas computacionales (software) o equipos electrónicos (hardware)".

Hiperentornos de Aprendizaje

El concepto de hiperentornos de aprendizaje, definido por César Labañino Rizzo y Mario del Toro Rodríguez como "Sistema informático basado en tecnología hipermedia, en la que se mezclan disímiles dispositivos que representan las diversas tipologías de software educativo con el designio de atender las necesidades didácticas" (38), auspiciando esto, soluciones concretas a diversas situaciones de aprendizaje desde un mismo software, que van desde la introducción de nuevos contenidos, el desarrollo y consolidación de habilidades llegando, incluso, a proponer tareas para la casa.

Tareas Docentes

La tarea docente constituye la célula de la actividad conjunta profesor-estudiante y es la "Acción del profesor y los estudiantes dentro del proceso, que se realiza en ciertas circunstancias pedagógicas, con el fin de alcanzar un objetivo de carácter elemental de resolver el problema planteado a estudiar por el profesor", en los diferentes componentes en que se desarrolla el Proceso de Formación.

Objeto de aprendizaje

Están definidos por un recurso digital o no digital, reproducible o direccionado que sea usado para desarrollar actividades de aprendizaje. Ejemplo son páginas web, libros de texto, calculadoras, instrumentos, etc. El estándar LOM los define como: "Cualquier entidad que puede ser usada, reutilizada o referenciada durante un proceso de aprendizaje apoyado por la tecnología".

Actividades

Son el núcleo de los elementos de la estructura del modelo de flujo de aprendizaje de un diseño de aprendizaje. Estas forman el vínculo entre los roles, los objetos de aprendizaje y los servicios de un ambiente. Ellas especifican las condiciones de terminación y las acciones a realizar para su terminación.

Existen distintos tipos de actividades: (39)

> **Actividades de aprendizaje:** Consisten en una actividad-descripción. Esta le orienta al usuario la actividad que deberá desarrollar individualmente y está dirigida a alcanzar un objetivo de aprendizaje mediante la realización de la misma.
>
> **Actividades de apoyo:** Actividades que permiten a un rol proporcionar algún tipo de soporte a otro rol. Un ejemplo típico son las actividades de evaluación. Aquí, el profesor proporciona un soporte de evaluación a cada uno de los alumnos.

Diseño de aprendizaje

Es la descripción de un método que permite a los alumnos alcanzar ciertos objetivos de aprendizaje desarrollando ciertas actividades de aprendizaje en un orden dado en el contexto de un ambiente de aprendizaje. Está basado en los principios pedagógicos del diseñador en un dominio específico y en contextos variables.

Unidad de aprendizaje

Es el término abstracto para referirse a una pieza de educación o entrenamiento, como un curso, un módulo, una clase. Estas deben estar representadas por más que una colección ordenada de recursos; debe incluir una variedad de actividades proveídas por el "profesor". Un diseño de aprendizaje (LD, del inglésLearning Design) es una parte integral de una unidad de aprendizaje. En las palabras de R. Koper de la Open University of theNetherlands: “Una unidad de estudio es la menor unidad que proporciona eventos educativos a los estudiantes, satisfaciendo uno o más objetivos educativos interrelacionados”. (39)

Multihilos

En sistemas operativos, un hilo de ejecución, hebra o subproceso es la unidad de procesamiento más pequeña que puede ser planificada por un sistema operativo.