Universidad de las Ciencias Informáticas

Facultad 2

Título:

“Sistema Tutorial Inteligente de apoyo al proceso de enseñanza-aprendizaje de la asignatura de Sistemas Operativos”

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

Autores:

- ✓ Luisa Idorka Morales Rodríguez
- ✓ Yarisleydi de Avila Fernández

Tutor:

- ✓ Ing. Daimara Martínez Borrell

Cotutor:

- ✓ Dra. Natalia Martínez Sánchez

La Habana, Junio de 2012

Año 54 de la Revolución

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

Para que así conste firmamos la presente a los __ del mes de Junio de 2012.

Luisa Idorka Morales Rodríguez	Yarisleydi de Avila Fernández
_____	_____
Autor	Autor
Ing. Daimara Martínez Borrell	Dra. Natalia Martínez Sánchez
_____	_____
Tutor	Cotutor

Dedicatoria

Yary:

Dedico mi tesis a mis padres por apoyarme siempre, por confiar en mí y darme fuerzas para salir adelante.

A mi hermanita Yanita por ser su ejemplo a seguir.

A mi abuelito pipa que le hubiera encantado verme graduar.

A mis dos abuelitas que me miman mucho y me complacen en todo lo que pueden.

A mi tía Marylin por ser como una madre para mí durante estos 5 años.

Luisa

A mi mamá por su sacrificio y dedicación.

A mi papá Carlo, por darme su cariño y apoyo.

A todos mis amigos y mi familia que siempre me han ayudado mucho.

Agradecimientos

Yary:

A mis padres mil gracias por apoyarme siempre, hoy soy lo que soy gracias a ellos. Por siempre confiar en mí y darme fuerzas cuando yo pensaba que no podía. Por todo el amor que me han dado y por todo lo que me han enseñando, por siempre estar allí cuando lo necesité, sobre todo a mi mamá por sus sabios consejos. Los amo.

A mi hermanita Yanita por ser su ejemplo a seguir y a mi hermana Liuba por siempre preocuparse por mí.

A mi abuelito pipa que le hubiera encantado verme graduar y a mis abuelitas por quererme tanto, me lo demuestran todos los días que hacen todo cuanto esté en sus manos para complacerme.

A mi tía Marylin por ser como una madre para mí durante estos 5 años por todo lo que ha hecho por mí muchas gracias, al igual que su esposo Pedry que mejor tío no ha podido ser.

A toda mi familia en general por siempre apoyarme, los quiero.

A mi compañera de tesis Luisa quien me ha demostrado ser una gran amiga, gracias por aguantarme todo este tiempo y por quererme como soy.

Luisa:

A mi mamá y mi papá Carlo, por ayudarme, confiar en mí y darme su apoyo siempre.

A mis amigos y mi tía Chela, los cuales contribuyeron a mi crecimiento profesional.

A mi compañera de tesis Yary que me apoyó siempre, y ha sido una fiel amiga desde que la conozco, gracias por ser parte de mi vida.

Agradecimientos comunes

A los profesores que nos ayudaron a perfeccionar el trabajo, en especial al primo Alain, que nuca dudó en apoyarnos y a nuestras tutoras Daimara y Natalia.

A nuestras compañeras de apartamento y a nuestros compañeros de aula durante estos 5 años, por tantos recuerdos bonitos, gracias por todo.

Resumen

El desarrollo de las Tecnologías de las Informáticas y las Comunicaciones (TICs), ha tenido gran influencia en las diversas esferas sociales, destacándose su uso en la educación, incorporando elementos novedosos que han revolucionado los antiguos métodos de enseñanza-aprendizaje.

Actualmente la Universidad de las Ciencias Informáticas (UCI), se encuentra inmersa en la búsqueda de nuevas formas que permitan facilitar el proceso de formación de los estudiantes, teniendo en cuenta el estado cognitivo de los mismos, para proporcionar la ayuda pedagógica adecuada.

El objetivo concreto del presente trabajo, lo comprende el desarrollo de un Sistema Tutorial Inteligente (STI) de apoyo al proceso de enseñanza-aprendizaje de la asignatura de Sistemas Operativos (SO) para la Facultad 2, que ayude a consolidar los contenidos del plan de estudio definido para la asignatura, utilizando para ello el Razonamiento Basado en Casos (RBC) como técnica de Inteligencia Artificial (IA).

Como resultado se obtiene un sistema web para ser utilizado por estudiantes y profesores.

Palabras claves: Sistema Tutorial Inteligente, proceso de enseñanza-aprendizaje, asignatura Sistemas Operativos, Razonamiento Basado en Casos.

Índice

Introducción 1

1. Capítulo 1. Fundamentación Teórica 5

1.1. Introducción 5

1.2. Sistemas Tutoriales Inteligentes 5

1.3. Sistemas basados en el conocimiento 8

1.4. Metodología de desarrollo de software 14

1.5. Lenguaje de Modelado 15

1.6. Plataforma de desarrollo 16

1.6.1. Lenguaje de Programación del lado del Servidor 16

1.6.2. Lenguaje de Programación del lado del Cliente 17

1.6.3. Framework Web 18

1.7. Herramientas de desarrollo 20

1.7.1. Herramientas CASE 21

1.7.2. Sistema Gestor de Base de Datos 23

1.8. Servidor Web 24

1.9. Conclusiones Parciales 25

2. Capítulo 2. Propuesta de solución. Exploración y Planificación 26

2.1. Introducción 26

2.2. Propuesta del sistema 26

2.3. Personas relacionadas con el sistema 32

2.4. Lista de Reserva del producto 33

2.5. Fase de exploración 35

2.6. Fase de planificación....36
2.8. Conclusiones Parciales....38
3. Capítulo 3. Diseño, Implementación y Prueba....39
3.1. Introducción....39
3.2. Patrón de la arquitectura....39
3.3. Tarjetas Clase-Responsabilidad-Colaborador....45
3.4. Modelo Físico de la Base de Datos....48
3.5. Despliegue del sistema....48
3.6. Tareas de Ingeniería....49
3.7. Pruebas....52
3.8. Validación....54
3.9. Conclusiones Parciales....57
Conclusiones....58
Recomendaciones....59
Referencias bibliográficas....60
Bibliografía....64
Glosario de Términos....69

Índice de Figuras

Figura 1: Mapa conceptual del modelo del STI-SO utilizando el RBC. 32

Figura 2: Capa Vista 40

Figura 3: Capa Controladora 41

Figura 4: Capa Modelo 42

Figura 5: Modelo físico de la Base de Datos. 48

Figura 6: Diagrama de Despliegue 49

Figura 7. Resultado de las pruebas de aceptación 54

Índice de Tablas

Tabla 1: Importancia de cada pregunta....28

Tabla 2: Personas relacionadas con el sistema....32

Tabla 3: HU Autenticar usuario.....36

Tabla 4: Plan de Iteraciones....37

Tabla 5: Plan de Entregas....38

Tabla 6: Tarjeta CRC clase Usuario....46

Tabla 7: Tarjeta CRC clase Cuestionario....46

Tabla 8: Tarjeta CRC clase Documento....46

Tabla 9: Tarjeta CRC clase Caso....47

Tabla 10: Tarjeta CRC clase Evaluacion....47

Tabla 11: Tarea de ingeniería 1. Comprobar existencia del usuario en LDAP y rol que ocupa en la base de datos.....52

Tabla 12: Caso de prueba de aceptación 1. Autenticar usuario....53

Tabla 13: Ejemplo de base de casos....55

Tabla 14: Descripción de rasgos objetivos....56

Tabla 15: Casos de Prueba....56

Tabla 16: Resultado de la validación....56

Introducción

En la actualidad existen sistemas que posibilitan el aprendizaje de forma autodidacta y la consolidación de una variedad de contenidos utilizando las facilidades que ofrecen las nuevas tecnologías aplicadas en el proceso docente, incrementando la utilización de las Tecnologías de la Información y las Comunicaciones (TICs) en el proceso de enseñanza-aprendizaje.

Los intentos de utilizar estas tecnologías para favorecer el aprendizaje, comenzaron a finales de la década del 60. A partir de ese momento, la presencia de ordenadores en los hogares y en las escuelas ha tenido un crecimiento exponencial. Con el tiempo, la evaluación de su utilización daba una garantía de mejores resultados en el proceso de enseñanza-aprendizaje, dejando a su paso, una visión más prudente y exigente de la forma apropiada en la que se debe utilizar. Las innovaciones tecnológicas y la creciente popularidad y disponibilidad de la Internet, fueron razones principales para el desarrollo de numerosas aplicaciones y proyectos de investigación del uso de los medios informáticos en el campo de la Tecnología Educativa (TE).

La enseñanza virtual (en el ambiente computacional) se emplea como una solución al crecimiento exponencial del conocimiento de la sociedad moderna, lo que conlleva a la búsqueda de nuevas soluciones pedagógicas y tecnológicas de enseñanza-aprendizaje. En este caso están los sistemas basados en la Inteligencia Artificial (IA), con el objetivo de construir modelos para facilitar y agilizar el desempeño educativo. Debido a esto, tanto profesores, como estudiantes, deben familiarizarse con dichas tecnologías, de forma tal que puedan aplicarlas para mejorar el proceso de enseñanza-aprendizaje.

En la Universidad de las Ciencias Informáticas (UCI) se desarrollan diferentes alternativas que adentran al estudiantado en el mundo de la Tecnología Educativa (TE). El modelo de integración formación-producción-investigación implementado, permite la incorporación progresiva a la enseñanza semipresencial, donde se hace uso del Sistema de Gestión de Cursos Moodle, el cual se utiliza a través del Entorno Virtual de Aprendizaje (EVA), donde estudiantes y profesores interactúan, propiciando así, un aprendizaje colaborativo además mantiene organizada toda la información referente a las diversas asignaturas del plan de Estudio de la Carrera. Sin embargo, el EVA no cuenta con una forma, con una guía inteligente que logre la personalización del proceso de enseñanza-aprendizaje.

Los estudiantes, debido al modelo que se implementa, no tienen el tiempo suficiente para consolidar el conocimiento de las diferentes asignaturas, ya que el profesor no está siempre disponible para que les aclare dudas. En la Facultad 2, en el presente curso se evidenció, que durante el desarrollo de las pruebas parciales de la asignatura de Sistemas Operativos (SO), la cual se imparte en el tercer año de la carrera, el por ciento de aprobados fue muy bajo. Durante la primera prueba parcial, aplicada a los cuatro grupos de la misma, para un total de 110 estudiantes, de ellos 104 presentados, resultaron aprobados sólo el 24,04%, mientras que en la segunda de 85 presentados fue de 20,00%. Demostrando que al subir el nivel de las mismas, los estudiantes no tenían las habilidades básicas ni el contenido bien consolidado para resolverlas. Trayendo la preocupación por parte del colectivo pedagógico de la asignatura, el lento avance de las habilidades cognitivas de los estudiantes.

Por lo anteriormente expuesto se plantea el siguiente **problema a resolver:** ¿Cómo desarrollar un modelo computacional que sirva de apoyo al proceso de enseñanza-aprendizaje en la asignatura de Sistemas Operativos?

Para darle solución al problema anterior se propone como **objeto de estudio:** Sistemas Tutoriales Inteligentes de apoyo al proceso de enseñanza-aprendizaje, siendo el **campo de acción:** Sistema Tutorial Inteligente de apoyo al proceso de enseñanza-aprendizaje de la asignatura de Sistemas Operativos.

Para lo cual se plantea el siguiente **objetivo general:** desarrollar un Sistema Tutorial Inteligente de apoyo al proceso de enseñanza-aprendizaje de la asignatura de Sistemas Operativos para la Facultad 2.

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

- ✓ Elaborar el marco teórico de la investigación.
- ✓ Formalizar un modelo computacional donde se apliquen técnicas de IA para el desarrollo de un Sistema Tutorial Inteligente (STI).
- ✓ Determinar la forma de representación del conocimiento que permita recuperar los modelos de estudiantes más similares para diagnosticar qué necesita el estudiante y cómo enseñar.
- ✓ Realizar la implementación computacional del modelo propuesto.
- ✓ Concebir una guía de orientación que facilite la creación del STI a los propios profesores.

Para dar cumplimiento a los objetivos se proponen las siguientes **tareas investigativas:**

- ✓ Estudio de características y funcionamiento de los STI.
- ✓ Análisis de diferentes tipos de sistemas basados en el conocimiento, en cuanto a la forma de representar el conocimiento, el método de solución del problema utilizado y la complejidad en el proceso de Ingeniería del conocimiento implícito en cada uno de los mismos para lograr una fundamentación de la selección del paradigma utilizado para el desarrollo del STI.
- ✓ Definición de la forma de representación del conocimiento atendiendo a la información necesaria para diseñar un STI.
- ✓ Estudio de las diferentes metodologías, lenguajes y herramientas de desarrollo.
- ✓ Determinación del método de solución de problema y modelación del estudiante a partir de la información almacenada en los casos.
- ✓ Realización del diseño del STI.
- ✓ Implementación computacional del modelo anterior en una herramienta de autor.

Al desarrollar estas tareas, se obtiene un STI de apoyo al proceso de enseñanza-aprendizaje de la asignatura SO.

Para el desarrollo de las tareas científicas se han combinado diferentes métodos y procedimientos teóricos y empíricos de la investigación científica en la búsqueda y procesamiento de la información. Los fundamentales son:

Métodos teóricos:

Analítico-Sintético: Este método permite realizar un análisis sobre las tendencias actuales entorno a la investigación de los STI, determinar cuáles son los principales métodos y algoritmos utilizados a nivel mundial y que pueden servir para la implementación del sistema en cuestión.

Métodos empíricos:

Observación: Es necesario entender bien la estructura y funcionamiento de un STI para lograr un mayor entendimiento del problema.

Entrevista a profesores del Departamento de SO: Se realizan entrevistas no formales a profesores del departamento de SO de la facultad 2, con el objetivo de obtener información necesaria con respecto a lo que debe hacer y debe tener el sistema en cuestión.

La tesis está conformada por tres capítulos estructurados de la siguiente forma:

Capítulo 1: Fundamentación teórica. En este capítulo se realiza un estudio de los STI, definiendo su estructura y funcionamiento. Se explican los diferentes sistemas basados en el conocimiento, así como la justificación de la selección. El estudio de las principales metodologías, lenguajes y herramientas, fundamentando en cada caso la elección.

Capítulo 2: Propuesta de solución. Exploración y Planificación. En este capítulo se describe el funcionamiento del sistema propuesto, definiendo las listas de reserva del producto, así como las fases de Exploración y Planificación siguiendo la metodología XP, confeccionando las Historias de Usuarios (HU) de cada iteración y los tiempos de entrega de cada una.

Capítulo 3: Diseño, Implementación y Prueba. En este capítulo se define la arquitectura del sistema, los patrones utilizados, así como las tareas de ingeniería generadas por cada HU y el diagrama de despliegue de la aplicación. Además de las pruebas realizadas y la validación de la propuesta de solución, con el objetivo de que el sistema cumpla con lo que desea el cliente.

Para finalizar se presentan las conclusiones, las recomendaciones, las referencias bibliográficas, la bibliografía, un glosario de términos y los anexos para un mejor entendimiento del trabajo.

1. Capítulo 1. Fundamentación Teórica

1.1. Introducción

En este capítulo se define qué es un STI, para qué se utiliza y cómo funciona. El análisis de los sistemas basados en el conocimiento, justificando el paradigma seleccionado, así como metodología, lenguaje y herramientas.

1.2. Sistemas Tutoriales Inteligentes

Los STI son sistemas que se adaptan al estado cognitivo de los estudiantes y que proporcionan la ayuda pedagógica adecuada para propiciar un buen aprendizaje.[1] Comenzaron a desarrollarse con la idea de impartir el conocimiento usando alguna forma de inteligencia para asistir y guiar al estudiante en su proceso de enseñanza, identificando la forma en que el mismo resuelve un problema a fin de poder brindarle ayudas cognitivas cuando lo requiera.[2] Por lo que se puede lograr un proceso de enseñanza-aprendizaje adaptado al estado cognitivo del estudiante sin la presencia de un tutor humano.

Estos sistemas son considerados “inteligentes”, pues realizan acciones pedagógicas sobre la forma de enseñanza y mantienen información sobre las necesidades del estudiante, ya que poseen la habilidad de determinar el qué, el cómo y a quién enseñar a través de sus módulos.

El **Módulo del Estudiante** define el conocimiento del estudiante. Tiene como objetivo realizar un diagnóstico del estado cognitivo del mismo, ya que deberá ser capaz de determinar su comprensión, identificando deficiencias y progreso a través del desarrollo de ciertos ejercicios que evalúan el contenido, recomendando así, la estrategia de estudio más conveniente de acuerdo a sus resultados, guardando de esta forma sus datos personales, siendo capaz de determinar la capacidad de adaptación a sus necesidades, es decir, se encarga de obtener una representación de las características de cada estudiante.

El **Módulo del Dominio** define el dominio del conocimiento. Tiene como objetivo proporcionar los conocimientos de forma adecuada para que el estudiante adquiera las habilidades y conceptos que se esperan, siendo capaz de generar preguntas, explicaciones, respuestas y tareas para el estudiante, resolver los problemas y corregir las soluciones presentadas.

El **Módulo Pedagógico** define el conocimiento pedagógico. Tiene como objetivo dirigir al estudiante en su proceso de aprendizaje y realizar ajustes a medida que progresa, definiendo y aplicando una estrategia pedagógica de enseñanza que contiene los objetivos que se persiguen así como los planes para alcanzarlos.

La **Interface** con el usuario permite la interacción del estudiante con el sistema. Debe ofrecer medios de comunicación para lograr una enseñanza adaptada, como por ejemplo imágenes o videos y contener toda la información necesaria para la realización de tareas que el sistema proponga.

A través de esta interacción entre los módulos, el sistema debe ser capaz de determinar lo que sabe el estudiante y cómo va en su progreso, por lo que para lograrlo, el sistema deberá identificar las fortalezas y debilidades de cada estudiante, con el fin de establecer un plan instruccional en dependencia de sus resultados.

Ejemplos de Sistemas Tutoriales Inteligentes

La mayoría de los STI desarrollados en el mundo, no presentan el nivel esperado de “inteligencia” ya que es muy difícil modelar el funcionamiento de la mente humana, por lo que aún no proveen un modo de aprendizaje adecuado de acuerdo a los conocimientos de cada estudiante y de su evolución.

Existen múltiples ejemplos de STI. En el mundo universitario, uno de los más conocidos es el sistema Andes, desarrollado por el equipo de Kurt Van Lehn de la Universidad de Pittsburg. Este sistema se encarga de guiar a los estudiantes mientras estos resuelven problemas y ejercicios. Si el estudiante pide ayuda en medio de un ejercicio, el sistema le da pistas para avanzar en la solución o le indica que ha fallado en algún paso anterior.[2]

El STI CircSim, fue desarrollado en conjunto por el Departamento de Ciencias de la Computación del Illinois Institute of Technology y el Departamento de Fisiología del Rush College of Medicine. Este tutor es el más avanzado en su tipo, y se utiliza en el Rush College of Medicine para complementar las clases teóricas sobre problemas cardiovasculares.[2]

AGT (Advancced Geometry Tutor), es un STI para el uso en clases de geometría avanzada, desarrollado en la University of Pittsburgh a través de la National Science Foundation through y el Center for Interdisciplinary Research on Constructive Learning Environments de la University of Pittsburgh y

Carnegie Mellon University. [2] Se basa en comparar dos estrategias de resolución de problemas, encadenamiento hacia adelante y encadenamiento hacia atrás para determinar el aprendizaje de los estudiantes sobre teorema de la geometría, con el fin de determinar qué estrategia acelera más el aprendizaje.

El Computer Tutoring Group (ICTG), que trabaja en el University's Computer Science and Software Engineering Department, en la University of Canterbury ha desarrollado una serie de STI entre ellos Sql-Tutor que es un sistema de enseñanza basado en el conocimiento que enseña Sql a los estudiantes. [2]

El STI “Mentor”, fue desarrollado para los estudiantes de la Universidad de La Salle, Bogotá, Colombia. Es una herramienta didáctica que permite personalizar el proceso de enseñanza-aprendizaje de la solución de problemas matemáticos en los que intervienen ecuaciones lineales. Además, es capaz de seguir el desempeño del estudiante en cualquier momento, de ajustarse a sus necesidades, y ejecutar las acciones pedagógicas pertinentes, influyendo así de manera significativa en la comprensión conceptual y el desarrollo de las habilidades cognitivas del estudiante para resolver problemas. [3]

El Ambiente Inteligente de enseñanza-aprendizaje asistido por computadora para la Programación Lógica (APA-Prolog) que utiliza mapas conceptuales para su solución y tiene asociado navegación dirigida por agentes pedagógicos inteligentes, fue desarrollado en el Centro Universitario de Sancti Spíritus (CUSS), con el apoyo de la Universidad Central de Las Villas (UCLV) y la Universidad de Granma y puesto a prueba con estudiantes de 4to año de la carrera de Ingeniería Informática del Centro Universitario de Sancti Spíritus.

Otros ejemplos de sistemas desarrollados por las universidades de Las Villas, de Sancti Spíritus y de Granma son las herramientas HESEI y MacBay.

HESEI es una herramienta computacional para elaborar sistemas de enseñanza-aprendizaje Inteligentes, la cual utiliza técnicas de IA y Mapas Conceptuales, con el objetivo de adaptar el sistema de enseñanza-aprendizaje, a través de una interfaz visual, a las características del alumno. Permite a expertos no especialistas en computación crear sistemas de enseñanza-aprendizaje inteligentes que aborden contenidos de cualquier especialidad y nivel de enseñanza. [4]

MacBay posibilita elaborar sistemas de enseñanza-aprendizaje inteligentes mediante la combinación de Redes Bayesianas y Mapas Conceptuales de cualquier asignatura y de cualquier nivel de enseñanza.

MacBay posibilita que los profesores que diseñen e implementen sistemas de enseñanza no necesariamente sean especialistas en Computación. [4]

Se han desarrollado STI en áreas como: Humedales (Laboratorio de Propagación Masiva de Plantas del Instituto de Biotecnología de las Plantas, de la UCLV), Teoría de Grafos, Análisis y Diseño de Sistemas y Estructura de Datos (Facultad de Matemática Física y Computación de la UCLV), Contabilidad y Finanzas (SUM de Santa Clara) y Carreteras (Facultad de Construcciones de la UCLV). Como filosofía de trabajo para el desarrollo de estas aplicaciones se siguen dos fases bien delimitadas: diseño del STI utilizando la guía de orientación y la herramienta computacional HESEI. [5]

Todos estos sistemas ayudan a mejorar el rendimiento, convirtiéndose en sistemas muy eficaces para el aprendizaje, pero ninguno de ellos contribuye a la mejoría del proceso de enseñanza-aprendizaje en la asignatura de SO, de ahí que se requiera desarrollar un STI que permita a los estudiantes consolidar sus conocimientos.

1.3. Sistemas basados en el conocimiento

Los sistemas basados en el conocimiento (SBC) constituyen técnicas de IA válidas para enfrentar la construcción de un STI, dada por sus aspectos afines. Pero no todos los paradigmas para crear SBC, facilitan la concepción de un STI, donde lo fundamental para su desarrollo es determinar cómo representar el conocimiento requerido para sus módulos y a partir de dicho conocimiento realizar un diagnóstico del estudiante para que el sistema se adapte a sus características. Sin embargo, similitudes de los STI y los SBC son factores a estudiar para concebir todos los módulos de los STI y un diagnóstico adecuado del qué y cómo enseñar dependiendo del estudiante.[5]

Una característica distintiva de estos sistemas es la separación del conocimiento (base del conocimiento) del método de solución del problema (máquina de inferencia).

En la base del conocimiento (BC) se almacena el conocimiento necesario para resolver los problemas del dominio de aplicación para el cual se desarrolla el SBC. El conocimiento que se almacena en la BC puede ser de diferentes tipos: conocimiento simbólico sobre cómo resolver los problemas del dominio el cual se puede representar mediante diversas formas de representación del conocimiento (reglas de producción, redes semánticas, etc.), probabilidades o frecuencias que modelan cómo se relacionan los valores de los diferentes rasgos que caracterizan el dominio, pesos de una red neuronal, casos o ejemplos de problemas

del dominio, etc. Estos diferentes tipos de conocimiento dan lugar a diferentes tipos de sistemas basados en el conocimiento, entre ellos: los Sistemas basados en reglas, los Sistemas basados en probabilidades, Sistemas expertos conexionistas o redes expertas y Sistemas basados en casos.

La máquina de inferencia (MI) es un procedimiento en el cual está implementado algún método que utiliza el conocimiento de la BC para resolver los problemas del dominio. El tipo de conocimiento determina que método de solución de problemas (MSP) será posible utilizar.

Sistemas basados en reglas

Un sistema basado en reglas (SBR) es un sistema basado en el conocimiento en el que se hace una representación mediante reglas de producción o reglas condicionales. Las reglas representan el conocimiento utilizando el formato SI-ENTONCES (IF-THEN).

SI (IF), es el antecedente, premisa, condición o situación.
ENTONCES (THEN), es el consecuente, conclusión, acción o respuesta. [6]

En este tipo de sistema, la BC contiene las variables y el conjunto de reglas que definen el problema, y la MI obtiene las conclusiones aplicando la lógica a estas reglas. La BC contiene la representación del conocimiento sobre el dominio de aplicación del sistema y se divide en la base de hechos y en la base de reglas. La base de hechos representa el conocimiento en las variables de entrada y salida del sistema y en la base de reglas se combinan variables de la base de hechos con el IF y el THEN junto con los operadores lógicos AND y OR. La parte de una regla que se encuentra entre IF y THEN es el antecedente y el resto el consecuente, en el antecedente es donde están las premisas que se deben cumplir para que la regla sea aplicable y en el consecuente el conjunto de acciones derivadas de la aplicación de la regla. La MI utiliza los datos y el conocimiento para obtener conclusiones o hechos. Por ejemplo, si la premisa de una regla es cierta, entonces la conclusión de la regla también debe ser cierta.

Para obtener estas conclusiones se utilizan diferentes tipos de reglas y estrategias de inferencia y control. Entre las reglas de inferencia se encuentran Modus Ponens y Modus Tollens y en cuanto a las estrategias de inferencia, el encadenamiento de reglas y el encadenamiento de reglas orientado a un objetivo, cuyas reglas son utilizadas por la MI para obtener conclusiones simples que son las que resultan de una regla simple y las compuestas que son las que resultan de más de una regla.

El MSP que emplea es forward Chaining: en el que los nuevos hechos son inferidos de hechos existentes y backward Chaining: por ejemplo, si hay una regla IF A THEN B, aplicar backward chaining significa usar el conjunto de reglas existentes para determinar que debe ser cierto para que B también lo sea. Esta técnica es empleada para probar una hipótesis.

Sistemas basados en probabilidades

Los sistemas basados en probabilidades (SBP) comenzaron a ser utilizados porque los SBR no tenían la capacidad de resolver situaciones que tuvieran incertidumbre, de ahí que se desarrollaran sistemas cuya BC contara con incertidumbre utilizando la probabilidad para medirla.

Los SBP, al igual que los SBR, cuentan con una BC pero con la diferencia que esta se forma por el espacio probabilístico que describe el problema. En este tipo de sistemas la MI está basado en probabilidades condicionales y se encarga de actualizar las probabilidades con base en los hechos que se observen del ambiente en que se desempeña y cuenta con el sistema de control de coherencia que logra la coherencia en el sistema especificando para ello, las restricciones que deben cumplir los datos introducidos. El MSP que se emplea es el teorema de Bayes para calcular probabilidades condicionales.

Estos sistemas memorizan fácilmente la información, pudiéndose almacenar y recuperar de la base de datos, pueden contar o calcular las frecuencias absolutas y relativas de cualquier subconjunto de variables a partir de la base de datos y pueden tomar o ayudar a las personas a tomar decisiones.

Sistemas expertos conexionistas o redes neuronales artificiales

Las redes neuronales artificiales (RNA) son modelos matemáticos inspirados en las neuronas biológicas del cerebro humano y programado en sistemas digitales, que tienen habilidades de aprendizaje automático, generalización y abstracción. Con estos modelos pueden resolverse una gran variedad de problemas de reconocimiento, aproximación, predicción, clasificación, optimización etc. [7] Los términos Sistemas Expertos (SE) y SBC, se utilizan como sinónimos.

Un SE es un conjunto de programas que, sobre una BC, posee información de uno o más expertos en un área específica. Un SE que utiliza redes neuronales (RN), es mejor conocido como sistema experto conexionista (SEC). El conexionismo se establece como un paradigma que permite modelar el

funcionamiento del sistema nervioso a partir de formas simplificadas del funcionamiento de las neuronas que lo constituyen y de las conexiones que entre ellas se establecen. [8]

La fusión de RNA y SE es uno de los campos abiertos más amplios de la investigación actual en IA. Su conjunción está soportada, de forma teórica, por la forma de procesar la información de los humanos. Una posible utilización compuesta de SEC es que: la red neuronal genera proposiciones y restricciones para los SE, deduciendo el conocimiento que no es capaz de explicitar el experto. Además pueden completar, gracias a la capacidad de las RN de restaurar patrones incompletos, los antecedentes o consecuentes incompletos de las reglas de producción de la BC del SE. Además, las RN pueden usarse para validar los SE, para actualizar parámetros estadísticos en módulos de SE, etc. Es posible desarrollar SEC a partir de la combinación de los SBR, propio de los SE clásicos, con las RNA. Esta combinación ofrece beneficios potenciales debido a que ambos modelos computacionales se complementan de manera mutua. El empleo de las RN en el desarrollo de SE ofrece entre sus ventajas principales el hecho de no necesitar un experto humano al cual extraerle el conocimiento. Esta es una tarea lenta, que consume mucho tiempo, se necesita que exista disposición por parte del experto y concordancia entre él y el ingeniero de conocimiento, posibilidades de que el experto pueda explicar sus métodos de solución de problemas, etc. Las RN son una alternativa pues ellas adquieren su conocimiento a partir de ejemplos. No obstante tienen también desventajas, entre las que se encuentran el hecho de necesitar una gran cantidad de ejemplos y no poder explicar cómo se alcanzan los resultados.[8]

En el desarrollo de una RN no hay que programar ni el conocimiento ni las reglas del procesamiento del conocimiento, ya que aprende mediante el ajuste de las conexiones ponderadas entre las neuronas de distintas capas de la red.

Sistemas basados en casos

Los sistemas basados en casos son los que más se asemejan al modo de pensar de los seres humanos. Diariamente, al enfrentarse a nuevas situaciones, lo primero que se hace es buscar en la memoria experiencias anteriores similares y a partir de allí, se establecen semejanzas y diferencias que se ajustan a las soluciones dadas anteriormente para obtener una nueva solución.

El razonamiento basado en casos (RBC) es una técnica de IA que intenta llegar a la solución de nuevos problemas de forma similar a como lo hacen los seres humanos utilizando la experiencia acumulada hasta

el momento en acontecimientos similares. [9] Entre los componentes fundamentales que contiene este tipo de sistema, se encuentran BC, el módulo de recuperación de casos y el módulo de adaptación de las soluciones. El MSP que se emplea es a través de la recuperación y la adaptación de los casos.

Los "casos" son problemas resueltos y almacenados en la BC. Cuando hay un nuevo problema que resolver, éste es descrito para el módulo de recuperación, el cual realiza una búsqueda en la BC y encuentra problemas o casos similares. Estos problemas o casos similares resueltos son recuperados (caso recuperado) y enviados al módulo de adaptación, donde son analizados para construir una solución para el nuevo problema y una vez hallada la solución se almacena junto con la descripción del problema en la BC, ya que constituye un nuevo caso.

El funcionamiento del RBC comprende cuatro actividades principales.

1. Recuperar los casos más parecidos.
2. Reutilizar el o los casos para tratar de resolver el nuevo problema.
3. Revisar y adaptar la solución propuesta, en caso de ser necesario.
4. Almacenar la nueva solución como parte de un nuevo caso.

Para desarrollar un sistema de este tipo, se tiene que tener en cuenta la forma de almacenar la experiencia, de modo que pueda ser recuperada correctamente y lograr que esta experiencia sea utilizada en un nuevo problema. También es muy importante definir bien la función de semejanza, encontrar una estructura apropiada para representar el contenido de un caso y decidir cómo la memoria de casos debe ser organizada para un almacenamiento, recuperación y rehúso efectivo.

Forma de almacenar los casos

Los casos pueden representar distintos tipos de conocimiento y pueden ser almacenados en diferentes formatos. Pueden representar personas, objetos, diagnósticos, planes, etc. Se puede elegir la implementación adecuada dependiendo del tipo de información a representar, teniendo en cuenta que la información almacenada en un caso, debe ser importante tanto para el propósito del sistema como para asegurar que siempre será elegido el caso más apropiado para solucionar un nuevo problema. No siempre se necesitan almacenar todos los casos existentes, en su lugar se sigue un criterio para decidir qué casos almacenar y cuales descartar.

Estructura para representar los casos

Existen principalmente dos estructuras de memoria: la plana y la jerárquica. En una estructura plana de la base, los casos se almacenan secuencialmente, en una lista simple, un arreglo o un fichero. Mientras que en una estructura jerárquica los casos se agrupan en categorías para reducir el número de casos a buscar en una consulta. Se almacenan en un árbol o en un grafo acíclico directamente. El grafo subdivide el espacio de casos de acuerdo con los atributos que les caracterizan. Las características comunes ocupan nodos del grafo de donde cuelgan los casos que las comparten.

Recuperación de casos

La tarea de recuperación comienza con una descripción del problema y finaliza cuando se encuentra un caso anterior, que es el que mejor ajusta. Los algoritmos de recuperación más investigados son: K-vecinos más cercanos, árboles de decisión y sus derivados. Estas técnicas necesitan de una medida de similitud para determinar la semejanza entre casos.

K-Vecinos más cercano: el caso recuperado es aquel en el que la suma de los pesos de las características que ajustan con las del caso nuevo es mayor que la de otros casos que ajustan. Esto quiere decir, que si los pesos de las características son iguales, un caso que ajusta con n características será recuperado antes que otro caso que empareja con k características, siendo k<n. Se puede asignar un peso mayor a las características consideradas más importantes.

Algoritmos inductivos: La inducción es una técnica desarrollada por los investigadores de aprendizaje de máquinas para extraer reglas o construir árboles de decisión a partir de datos pasados. En los sistemas RBC la base de casos es analizada por un algoritmo de inducción que determina que características son las que mejor diferencian los casos entre sí para producir un árbol de decisión que clasifica los casos. Este enfoque es muy útil cuando se requiere un único caso como solución y cuando la característica del caso depende de otras.

Fundamentos de la selección

A pesar que los SBR son los esquemas más comúnmente utilizados para la representación del conocimiento, no es conveniente aplicar este tipo de razonamiento pues la forma que se utiliza de descomponer el conocimiento del dominio y generalizarlo en reglas, resultaría un proceso engorroso que

no se ajusta al modelo que se quiere desarrollar. Un SBP no es favorable utilizarlo, pues este tipo de conocimiento es aplicado en sistemas que se centran en la utilización de la incertidumbre, lo cual no es el caso. Los SEC o RN tampoco se utilizan pues no es objetivo desarrollar modelos matemáticos.

Sin embargo, utilizar el **RBC**, es la estrategia más acertada, principalmente porque este tipo de razonamiento es el que más se asemeja al modo de pensar de los seres humanos, lo cual complementa el propósito del desarrollo del sistema, donde la base fundamental es ayudar al estudiante en su proceso de enseñanza-aprendizaje, por lo que a la hora de decidir cómo representar el conocimiento, no es muy difícil decidirse, pues la representación del conocimiento a través de los casos, mantiene la organización de la BC, de las estrategias de recuperación y de adaptación de los casos. Si se presenta una misma situación, no se tiene que generar a partir de cero, ya que se rehúsan las soluciones anteriores y como las respuestas no se dan a partir de cero sino de casos resueltos anteriormente, esto permite dar soluciones rápidamente.

1.4. Metodología de desarrollo de software

Para la selección de la metodología de desarrollo a utilizar, fueron consideradas dos de las más conocidas. Estas son el Proceso Unificado de Desarrollo (RUP) y la Programación Extrema (XP).

RUP está basado en componentes, lo que quiere decir que el sistema software en construcción está formado por componentes software interconectados a través de interfaces bien definidas y utiliza el Lenguaje Unificado de Modelado (UML) para preparar todos los esquemas de un sistema software, siendo una parte esencial del Proceso Unificado. Es iterativo e incremental, centrado en la arquitectura y guiado por casos de uso. Se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo consta de cuatro fases: inicio, elaboración, construcción y transición. [10] Durante las fases establecidas se desarrollan nueve flujos de trabajos, de ellos seis denominados de Ingeniería, que son modelamiento del negocio, requerimientos, análisis y diseño, implementación, prueba y despliegue. Los restantes tres flujos de trabajo son considerados de apoyo y son: la administración de proyectos, la gestión de configuración y cambios, y finalmente el ambiente.

XP es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado. [11] Es utilizada en proyectos con requisitos muy cambiantes ya que se basan en proyectos pequeños, caracterizados por su sencillez, tanto en el

aprendizaje como en su aplicación. Considerada ligera, flexible, predecible, de bajo riesgo, y no por ello menos científica. Entre otras ventajas pueden mencionarse los pocos requerimientos de documentación y planificación, así como la exigencia de tener siempre el cliente disponible para el desarrollo, implicando una mejor correspondencia entre el producto y la necesidad del negocio, promoviendo y propiciando un buen clima de trabajo en equipo. [12] Uno de los objetivos fundamentales de XP es la satisfacción del cliente, ya que la metodología trata de brindar el software que este necesite y cuando lo necesite. Por tanto, se debe responder muy rápido a sus necesidades, incluso cuando los cambios sean al final de ciclo de la programación.

Fundamentos de la selección

Se decide utilizar **XP**, ya que dadas las particularidades del grupo de desarrollo, resulta conveniente utilizar las técnicas esenciales de esta metodología, pues genera solo la documentación necesaria, la cual no es numerosa por ser un proyecto pequeño. Si durante el desarrollo del mismo, se detecta la necesidad de cambiar funcionalidades, se realiza un acuerdo con el cliente, se ajustan el plan de iteraciones y se toma una nueva dirección en el desarrollo, lo cual resulta conveniente pues si se realizan cambios al sistema luego de una posible propuesta para ser implementada en la Facultad, no sería necesario arreglar tanta documentación, mientras que RUP es un proceso pesado que se basa en la documentación.

1.5. Lenguaje de Modelado

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es una de las mejores herramientas para analizar y diseñar sistemas de software que ofrece un lenguaje común que todo desarrollador debe conocer. Es un conjunto de herramientas que permite modelar (analizar y diseñar) sistemas orientados a objetos. [13]

Fundamentos de la selección

Se escogió **UML** por ser el lenguaje de modelado más conocido y utilizado en la actualidad y representa el propósito general que pueden usar todos los modeladores ya que no tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática.

1.6. Plataforma de desarrollo

1.6.1. Lenguaje de Programación del lado del Servidor

PHP, acrónimo de "Hypertext Preprocessor" es un lenguaje interpretado por el servidor generando un HTML[1] con el resultado de sustituir las secuencias de instrucciones php por su salida. Ampliamente utilizado, de código abierto y de propósito general bajo licencia GPL. Es un lenguaje de scripting adecuado para el desarrollo web y embebido en páginas HTML. El principal objetivo del lenguaje es permitir que los desarrolladores web escriban páginas generadas dinámicamente de forma rápida, pero se puede hacer mucho más con PHP. [14]

Entre sus principales características cabe destacar su potencia, su alto rendimiento, su facilidad de aprendizaje y su escasez de consumo de recursos, permitiendo al desarrollador realizar varias funciones, desde generar documentos en pdf hasta analizar código XML[2]. Es multiplataforma y con capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destacando su conectividad con MySQL, lo cual permite la creación de aplicaciones web robustas. [14]

Java es un lenguaje de programación desarrollado por Sun Microsystems que ahora es una subsidiaria de Oracle Corporation. Su sintaxis toma mucho de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. [15]

Java tiene muchas ventajas, como por ejemplo es un lenguaje multiplataforma, es un software de distribución libre, no es necesario pagar una licencia para comenzar a desarrollarlo, así mismo es un lenguaje muy completo y poderoso, se pueden realizar muchas tareas con él, pues posee librerías y utilidades muy completas que facilitan la programación, aunque tiene la desventaja de ser un lenguaje de ejecución lenta, debido al uso de la máquina virtual de Java. [15]

[1] HTML ir al Glosario de Términos

[2] XML ir al Glosario de Términos

1.6.2. Lenguaje de Programación del lado del Cliente

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas, incorporando efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Los programas escritos con este lenguaje se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. Legalmente, JavaScript es una marca registrada de la empresa Sun Microsystems. [16]

Librerías de JavaScript

Con la potenciación de las aplicaciones cada vez tienen más código **Javascript** y son utilizadas más librerías para gestionar mejor el aspecto visual de la aplicación así como la carga de datos vía **Ajax**[3]. Por eso, librerías como **jQuery**, **Prototype o Dojo** han entrado en auge. Normalmente estas librerías tienen una versión de desarrollador y una versión comprimida para ponerla en producción y reducir la carga y transferencia del código Javascript. [17]

JQuery no es más que un conjunto de librerías Javascript que permite simplificar la manera de interactuar con los documentos HTML, permitiendo manejar eventos, desarrollar animaciones, y agregar interacción con la tecnología Ajax a páginas web. Está diseñada para cambiar la forma de escribir el código Javascript escribiendo menos y haciendo más, convirtiéndose en la librería JavaScript más utilizada actualmente. Es gratuita, de código abierto (bajo licencia MIT y GPL v2) e increíblemente ligera. Entre las ventajas de utilizar esta librería se encuentran reducción y reutilización de código que permite realizar scripts con dinamismos sencillos, código más conciso y de fácil lectura, fácil manejo de eventos y animaciones, permitiendo desarrollar componentes dinámicos con los que se enriquecen las aplicaciones. [18]

Prototype es una librería JavaScript similar a Jquery y muy bien desarrollada que facilita gran parte del trabajo asociado a crear páginas web altamente interactiva, pero su documentación es escasa. Es un

[3] Ajax ir al Glosario de Términos

simple archivo .js, que debe ser declarado en el documento principal. Su uso otorga una gran flexibilidad e inter-operabilidad en distintas circunstancias y navegadores que deseen emplear Ajax. [19]

Dojo es un proyecto Open Source bajo licencia AFL y BSD. Librería modular construida en JavaScript con el objetivo de resolver las incompatibilidades de los navegadores y hacer más fácil el desarrollo de aplicaciones complejas. Entre sus características se encuentran la manipulación del DOM[4], el manejo de Ajax y de eventos, uso gráficos, etc. [20]

1.6.3. Framework Web

Un Framework Web es un conjunto de componentes que forman un diseño reutilizable que facilita y agiliza el desarrollo de sistemas Web. Los objetivos principales que persigue un Framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. [21]

Symfony es un Framework completo diseñado para optimizar de acuerdo a sus características, el desarrollo de las aplicaciones Web. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación Web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación Web compleja. Emplea el tradicional patrón de diseño Modelo-Vista-Controlador (MVC) para separar las distintas partes que forman una aplicación web. [21]

Características de Symfony:

- ✓ Fácil de instalar y configurar en la mayoría de plataformas.
- ✓ Independiente del sistema gestor de bases de datos.
- ✓ Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- ✓ Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros. [21]

[4] DOM ir al Glosario de Términos

CodeIgniter es un entorno de desarrollo abierto que permite crear webs dinámicas con PHP. Su principal objetivo es ayudar a que los desarrolladores puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero. [22]

Características de CodeIgniter:

- ✓ Es capaz de trabajar la mayoría de los entornos o servidores y posee una gran capacidad de adaptación.
- ✓ Es compatible con la versión PHP 4, lo que hace que se pueda utilizar en cualquier servidor, incluso en algunos antiguos. Funciona correctamente con las últimas versiones de PHP, especialmente con la 5.
- ✓ Define una manera de trabajar específica.
- ✓ El núcleo de CodeIgniter es bastante ligero, lo que permite que el servidor no se sobrecargue interpretando o ejecutando grandes fragmentos de código. La mayoría de los módulos o clases que ofrece se pueden cargar de manera opcional, sólo cuando se van a utilizar realmente.
- ✓ La estructura de archivos es muy clara y ofrece gran flexibilidad en caso que necesite cambiarse por algún motivo. La división en models, views, controllers, libraries, helpers, config, etc. es muy práctica y comprensible.
- ✓ La documentación de CodeIgniter es fácil de seguir y de asimilar, pues está escrita en modo de tutorial. [22]

Fundamentos de la selección.

A partir de lo anterior se plantea que **PHP 5.3.8** es el lenguaje que más se adapta a las características que requiere el sistema, por su potencia, su alto rendimiento, su facilidad de aprendizaje y su escasez de consumo de recursos, es multiplataforma y con capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad destacando su conectividad con MySQL.

Se decide utilizar el **framework CodeIgniter 2.1.0**, ya que es un framework de apoyo desarrollado en PHP para la creación de aplicaciones web, contiene ayudas para la creación de aplicaciones PHP avanzadas que hacen que el proceso de desarrollo sea más rápido. Además define una arquitectura para programar de forma ordenada y contiene diversas herramientas que ayudan a hacer aplicaciones más versátiles y seguras. Está creado para que sea fácil de instalar en cualquier servidor y de empezar a utilizar.

Como lenguaje de programación en el cliente se emplea **HTML**, debido a que es el lenguaje de marcado predominante para la realización de páginas web, describiendo la estructura y el contenido en forma de texto, así como **CSS**[5] pues permite controlar los tamaños de fuente, color y estilo, y proporciona un control más preciso sobre la presentación de contenido alternativo que HTML por sí solo.

Para el desarrollo de la aplicación se hace necesario utilizar código **JavaScript** pues facilita el trabajo con formularios dinámicos, utilizando para ello una de sus librerías. En este caso se decide utilizar **jQuery**, ya que es una librería JavaScript muy rápida y muy ligera que simplifica el desarrollo de la parte del cliente de las aplicaciones web e incluye muchas utilidades para crear fácilmente las páginas web de las aplicaciones de forma dinámica, escribiendo menos y haciendo más.

1.7. Herramientas de desarrollo

IDE: Integrated Development Environment (entorno de desarrollo integrado), empaquetado como un programa de aplicación, o sea, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. [23]

NetBeans es un IDE que permite programar en distintos lenguajes, es ideal para trabajar con el lenguaje de desarrollo JAVA (y todos sus derivados), además ofrece un excelente entorno para programar en PHP. La IDE de NetBeans es perfecta y muy cómoda para los programadores. Tiene un excelente balance entre una interfaz con múltiples opciones y un aceptable completamiento de código. [23]

Características de NetBeans:

- ✓ Formado por un IDE de código abierto y una plataforma de aplicación que permite a los desarrolladores crear con rapidez aplicaciones web.
- ✓ Ofrece documentación y recursos de formación exhaustivos.
- ✓ Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, colaboración entre varias personas.
- ✓ Es un IDE multiplataforma.
- ✓ Está desarrollado para usarse generalmente con lenguaje Java, aunque permite su uso con otros lenguajes de programación como PHP.

[5] CSS ir al Glosario de Términos

- ✓ Permite crear aplicaciones Web con PHP 5. [24]

Eclipse es un IDE de código abierto y multiplataforma. Es una potente y completa plataforma de programación, desarrollo y compilación de elementos tan variados como sitios web, programas en C++ o aplicaciones Java así como también para PHP u otros, donde se encuentran todas las herramientas y funciones necesarias para trabajar, recogidas además en una atractiva interfaz que lo hace fácil y agradable de usar. [25]

Características de Eclipse:

- ✓ Su principal inconveniente es el consumo de recursos del sistema, que es común a otros IDE en mayor o menor medida.
- ✓ Es totalmente libre y no es necesario comprar ninguna licencia comercial, lo cual lo hace atractivo para muchos.
- ✓ Es multiplataforma.
- ✓ Es un potente editor de texto con la capacidad de resaltar sintaxis, así como también una compilación en tiempo real y soportes.[26]

Fundamentos de la selección

Se decide utilizar **NetBeans 7.1** pues ofrece opciones de desarrollo con plugins y herramientas incorporadas, así como completamiento de código, facilitándoles el trabajo a los programadores para la creación de aplicaciones web con rapidez, es una herramienta fácil de instalar y consume pocos recursos. El editor de código fuente que presenta es muy ágil y a la vez robusto, características que hacen que sea una excelente herramienta para el desarrollo de soluciones informáticas.

1.7.1. Herramientas CASE

Las herramientas CASE (Ingeniería de Software Asistida por Computación), son diversas aplicaciones informáticas destinadas a aumentar la productividad en el ciclo de vida de desarrollo del software. [27]

Rational Rose es la herramienta CASE desarrollada por los creadores de UML (Booch, Rumbaugh y Jacobson), que cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables. Propone la utilización de cuatro tipos de modelos para realizar el diseño del sistema,

utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite que haya varias personas trabajando a la vez en el proceso iterativo controlado, para ello posibilita que cada desarrollador opere en un espacio de trabajo privado que contiene el modelo completo y tenga un control exclusivo sobre la propagación de los cambios en ese espacio de trabajo. [28]

Características principales:

- ✓ Realiza Chequeo semántico de los modelos.
- ✓ Ingeniería "de ida y vuela": Rose permite generar código a partir de modelos y viceversa.
- ✓ Desarrollo multiusuario.
- ✓ Integración con modelado de datos.
- ✓ Generación de documentación.
- ✓ Tiene un lenguaje de script para poder ampliar su funcionalidad.
- ✓ Disponible en múltiples plataformas. [28]

Visual Paradigm es una herramienta CASE que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Constituye una herramienta de software libre de probada utilidad para el analista. [29]

Características principales:

- ✓ Entorno de creación de diagramas para UML
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Capacidades de ingeniería directa (versión profesional) e inversa.
- ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- ✓ Disponibilidad de múltiples versiones, para cada necesidad.
- ✓ Disponibilidad en múltiples plataformas. [29]

Fundamentación de la selección

Visual Paradigm 8.0 ha sido escogido como herramienta CASE debido a las características del software y el entorno en que se desarrolla el sistema, ya que genera código PHP que es el código que se utiliza, se añade la generación del modelo de datos para el gestor de base datos MySQL, funciona bien sobre

software libre, es una de las herramientas UML CASE más completa y fácil de usar, con soporte multiplataforma.

1.7.2. Sistema Gestor de Base de Datos

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. Desde enero del 2008 es una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril del 2009, desarrolla MySQL como software libre en un esquema de licenciamiento dual. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. [30]

Características de MySQL:

- ✓ Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
- ✓ Soporta gran cantidad de tipos de datos para las columnas.
- ✓ Gran portabilidad entre sistemas.
- ✓ Gestión de usuarios y passwords, manteniendo un nivel elevado de seguridad en los datos.
- ✓ Infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación.
- ✓ Gran rapidez y facilidad de uso
- ✓ Fácil instalación y configuración. [30]

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado. PostgreSQL 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. [31]

Características de PostgreSQL

- ✓ Funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.
- ✓ Disponible para Linux y UNIX en todas sus variantes y Windows 32/64bit.

- ✓ Por su arquitectura de diseño, funciona bien al aumentar el número de CPUs y la cantidad de RAM.
- ✓ Incluye características de la orientación a objetos, como la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional aunque no se considera un sistema de gestión de bases de datos puramente orientado a objetos.
- ✓ Soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos. [31]

Fundamentos de la selección

Se decide utilizar **MySQL 5.5.16** pues es un gestor de bases de datos muy utilizado en entornos en los cuales se emplea PHP, ya que PHP dispone de numerosas funciones que son integradas muy bien con MySQL y herramientas que permiten su uso en gran cantidad de lenguajes de programación. La gestión de usuarios y contraseñas que posee, mantiene una buena seguridad en los datos y consume pocos recursos, tanto de CPU como de memoria.

1.8. Servidor Web

Un servidor Web consta de un intérprete HTTP[6] el cual se mantiene a la espera de peticiones de clientes y le responde con el contenido según sea solicitado. El cliente, una vez recibido el código, lo interpreta y lo exhibe en pantalla. [32]

Apache es un servidor web de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP. Se distribuye bajo una licencia especial Apache Software License.[32]

Características de Apache:

- ✓ Es flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos.
- ✓ Es Multiplataforma.
- ✓ Es una tecnología gratuita de código fuente abierto.
- ✓ Es un servidor altamente configurable de diseño modular.
- ✓ Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.
- ✓ Gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP.[32]

[6] HTTP ir al Glosario de Términos

Fundamentos de la selección

Además de las características antes mencionadas, se escoge **Apache 2.2.21** como servidor Web, ya que es uno de los servidores web más utilizados, pues es una tecnología gratuita de código fuente abierto que ofrece instalaciones sencillas para sitios pequeños como es el caso y si se requiere se puede expandir. Es una solución sencilla, eficaz y rápida. Además, permite la creación de sitios web dinámicos utilizando PHP que es el lenguaje de programación seleccionado.

1.9. Conclusiones Parciales

En este capítulo se realizó un estudio de los STI, exponiendo en forma resumida sus principales características así como su estructura y funcionamiento, llegando a la conclusión que para el sistema propuesto se hará uso del RBC como técnica de IA para su implementación. Además, en el estudio de las posibles metodologías, lenguajes y herramientas para el desarrollo del mismo, se concluyó que se utilizará como metodología de desarrollo XP, UML como lenguaje de modelado, Visual Paradigm 8.0 como herramienta CASE, como lenguaje de programación del lado del cliente PHP 5.3.8 y del lado del servidor HTML, CSS y JavaScript, como framework de apoyo CodeIgniter 2.1.0, además de IDE NetBeans 7.1, MySQL 5.5.16 como gestor de base ded y Apache 2.2.21 como servidor web.

2. Capítulo 2. Propuesta de solución. Exploración y Planificación

2.1. Introducción

En este capítulo se describe la propuesta de solución del sistema, se definen las listas de reserva del producto, se detallan dos de las fases de la metodología seleccionada, las cuales son Exploración y Planificación, donde se confeccionan las Historias de Usuarios (HU) para cada iteración, definiendo los tiempos de entrega de cada una.

2.2. Propuesta del sistema

Durante el diseño de la propuesta de solución, fue necesario consultar expertos en el tema. Dichos expertos son considerados como los profesores de las asignatura de SO del Dpto. de Sistemas Digitales de la Facultad 2, con más de 3 años de experiencia impartiendo dicha asignatura.

Para el desarrollo del Sistema Tutorial Inteligente para Sistema Operativos (STI-SO), se utiliza el RBC, en el cual, los casos en la BC representan el estado del conocimiento y comportamiento del estudiante, así como el material didáctico más adecuado. Cada caso es un ejemplo de modelado de estudiante, el cual se divide en rasgos predictores y rasgo objetivo. Dado un nuevo usuario, se diagnostica utilizando el RBC, los materiales de estudio para el mismo, adaptados a sus conocimientos sobre el tema en cuestión.

Los **rasgos predictores** reflejan el estado cognitivo del usuario, donde cada rasgo tiene un valor asociado (1 en caso de respuesta correcta 0 en caso contrario) y contienen los datos de entrada, o sea la información a partir de la cual el sistema infiere el estado del mismo. La forma de valorizar los rasgos predictores es a través de cuestionarios, captando el estado cognitivo.

El **rasgo objetivo** es un rasgo multievaluado, los valores del mismo se corresponden con los materiales didácticos propuestos. El dominio del rasgo objetivo es el conjunto de materiales diseñados por los expertos que se adecuan a las características del contenido que se evalúa.

La **Base de Casos** debe facilitar el acceso y recuperación de los casos semejantes, por tanto es importante determinar cómo organizarla. En el modelo que se propone se almacenan los casos en una estructura lineal donde todos los casos son analizados para resolver el nuevo problema.

Definición de los rasgos predictores a través de cuestionarios

Para la obtención de los rasgos predictores a partir de cuestionarios se aplican 15 preguntas, cuyas respuestas son binarias (0/1). El rasgo objetivo contiene los materiales didácticos que se corresponden con el modelado del estudiante, que se obtiene por una escala valorativa según criterio de expertos.

Determinar la función de semejanza

Existen varios modos para medir el grado de parecido o semejanza entre dos objetos que puede ir desde expresiones analíticas de distancia y pseudo distancias hasta la descripción de un procedimiento para obtener el grado de parecido entre dichos objetos.
Cuando la medida numérica del grado de semejanza se calcula mediante una expresión analítica, a la misma se le llama función de semejanza.
No existe una medida de semejanza única, general, para cualquier dominio, de ahí que el buen funcionamiento del sistema radica en la función de semejanza que se defina.

La función de semejanza seleccionada para utilizar en el STI-SO es:

$$f(O_o, O_t) = \frac{\sum_{i=1}^{n} w_i \cdot \delta_i(x_i(O_0), x_i(O_t))}{\sum_{i=1}^{n} w_i}$$

donde O_0 es el caso a resolver y O_t es el caso existente en la BC, n es el número de rasgos predictores, w_i es la importancia asociada al rasgo i, pues todos los rasgos no tienen igual importancia a la hora de diagnosticar el estado cognitivo de un estudiante (modelo del estudiante) y δ_i es la función o criterio de comparación del rasgo i

La función de semejanza será aplicada a cada caso existente en la BC para ser comparada con el caso a resolver, donde la función de comparación por rasgos es la siguiente:

$$\delta_i(x_i(O_0), x_i(O_t)) = \begin{cases} 1 & si\ \delta_i(O_0) = \delta_i(O_t) \\ 0 & en\ otro\ caso \end{cases}$$

El resultado de comparar rasgo a rasgo es 1 si resulta igual y cero en otro caso.

Para el modelado del estudiante se tiene en cuenta la importancia de cada rasgo definida según criterio de expertos como se muestra en la siguiente tabla:

Tabla 1: Importancia de cada pregunta

Pregunta	Importancia de la pregunta
(1 a 3)	0.1
(4 a 6)	0.2
(7 a 9)	0.3
(10 a 12)	0.4
(13 a 15)	0.5

El procedimiento de recuperación consta de dos etapas: **acceso y recuperación**. En la etapa de acceso se seleccionan los casos más parecidos al nuevo caso para el proceso de recuperación.

Para ello se define un valor para el umbral de semejanza, que no es más que la cota inferior de los posibles valores de semejanza existente para cada uno de los casos recuperados con respecto al nuevo caso, logrando una mayor precisión en la obtención de los casos más semejantes. Cuyo valor es definido por criterio de expertos para garantizar que los casos sean lo más parecido posibles.

Algoritmo 1. Acceso a los casos más semejantes

Entrada: Problema a resolver: O_0, BC y el umbral de semejanza β_0 = 0.75.

Salida: Conjunto de casos potenciales para la recuperación dado O_0: $Pt(O_0)$.

A1: Para todos los casos de la BC.

A1$_1$ Calcular $\beta(O_0, O_t)$ utilizando la función de semejanza seleccionada.

A1$_2$: Seleccionar O_t cuyo valor de $\beta(O_0, O_t) \geq \beta_0$ y asignar a $Pt(O_0)$.

De esta forma en el conjunto quedan los casos más semejantes (k *-vecinos más cercanos*) para dar solución al caso O_t.

Determinación de la decisión

Para el STI-SO la fase de adaptación es nula, ya que el caso más semejante da directamente la solución al problema, pero en el posible caso que este resultado sea menor que el umbral definido, significa que no hay ningún caso parecido en la BC, por lo que el sistema guarda los datos de este caso para luego, según criterio de expertos, sea agregado como una nueva solución o eliminado completamente del sistema, informando al usuario una solución predeterminada que será orientada si el caso no es semejante a ningún otro.

Aprendizaje de la base de casos

Para el sistema propuesto, la base BC logra el aprendizaje con la incorporación de nuevos casos que se van resolviendo y almacenando para su posterior análisis por los expertos.

Diseño del STI-SO

Para el diseño de SITI-SO se siguió la guía de orientación para la Ingeniería del Conocimiento en el diseño de STI descrita en (Martínez, 2010).

El paso previo para el proceso de ingeniería del conocimiento de STI-SO, está determinado por tres etapas fundamentales, estrechamente relacionadas y con un orden de precedencia establecido, que facilitan definir los modelos de estudiantes y materiales didácticos incorporados.

Etapa I: Diagnóstico del contexto.

Objetivo: Justificar la necesidad de elaborar el STI-SO.

Aspectos a examinar:

1. Análisis de las necesidades educativas.
2. Disponibilidad de recursos tecnológicos.
3. Implicación de la introducción de un STI en el proceso de enseñanza-aprendizaje.
4. Estudio de los aspectos teóricos y metodológicos de la asignatura SO.

5. Determinación de los objetivos, contenidos, sistema de habilidades, entre otros, de la asignatura según plan de estudio de la carrera de Ingeniería en Ciencias Informáticas.

Etapa II: Definir la estructura del modelado del estudiante.

Objetivo: Decidir qué enseñar y cómo enseñar según la caracterización individual del estudiante

Aspectos a examinar:

1. Determinación de los aspectos cognitivos (preguntas del cuestionario). Organización de las preguntas según precedencia y nivel de complejidad (aumenta nivel de complejidad según el orden). Estos deben avalarse por expertos en el dominio del conocimiento que elaboran el STI (profesores de SO).
2. Determinación de otros aspectos a incluir (nombre y apellidos).
3. Determinación de los modelados de estudiantes a utilizar (con este paso se concretan ejemplos de casos).

Etapa III. Edición del modelado del estudiante.

Objetivo: Obtener un prototipo no computarizado del STI.

Aspectos a examinar:

1. Edición de rasgos cognitivos:

Para cada rasgo definido en la etapa anterior describir:

1.1. Nombre (pregunta 1..15)

1.2. Medidas cualitativas o cuantitativas a utilizar para evaluar cada rasgo (0/1)

1.3. Confección de los cuestionarios

1.3.1. Número de preguntas a realizar (15)

1.3.2. Orden en que se van a realizar las preguntas (teniendo una complejidad incremental)

1.3.3. Para cada pregunta definir el tipo de evaluación (1/0, bien o mal)

1.3.4. Para cada pregunta definir el formato (tipo) de la pregunta (seleccionar una respuesta, seleccionar varias respuestas y verdadero o falso.)

1.3.5. Elaborar las preguntas definidas atendiendo a las características decididas anteriormente.

2. Recuperación de materiales didácticos relacionados con el dominio del STI.

2.1. Definir el número de materiales didácticos a elaborar por cada modelo inicial del estudiante. (puede ser pdf, word, páginas de un libro)

2.2. Elaborar o recuperar materiales didácticos con las estrategias pedagógicas adecuadas para adaptarse a los modelos de estudiantes (se realiza en base al resultado de los cuestionarios evaluados).

3. Decidir los medios de enseñanza para mostrar la información a utilizar en cada material didáctico (documentos en Word, documentos en pdf, presentaciones en PowerPoint, videos).

En los anexos se puede encontrar un ejemplo de un cuestionario con los 3 tipos de preguntas, el cual está marcado con las respuestas correctas y un ejemplo hipotético para mostrar cómo funciona el modelado del estudiante. (VerAnexo1: Ejemplo de los tipos de preguntas de un cuestionario)

La siguiente figura muestra el mapa conceptual del modelo del STI-SO utilizando el RBC:

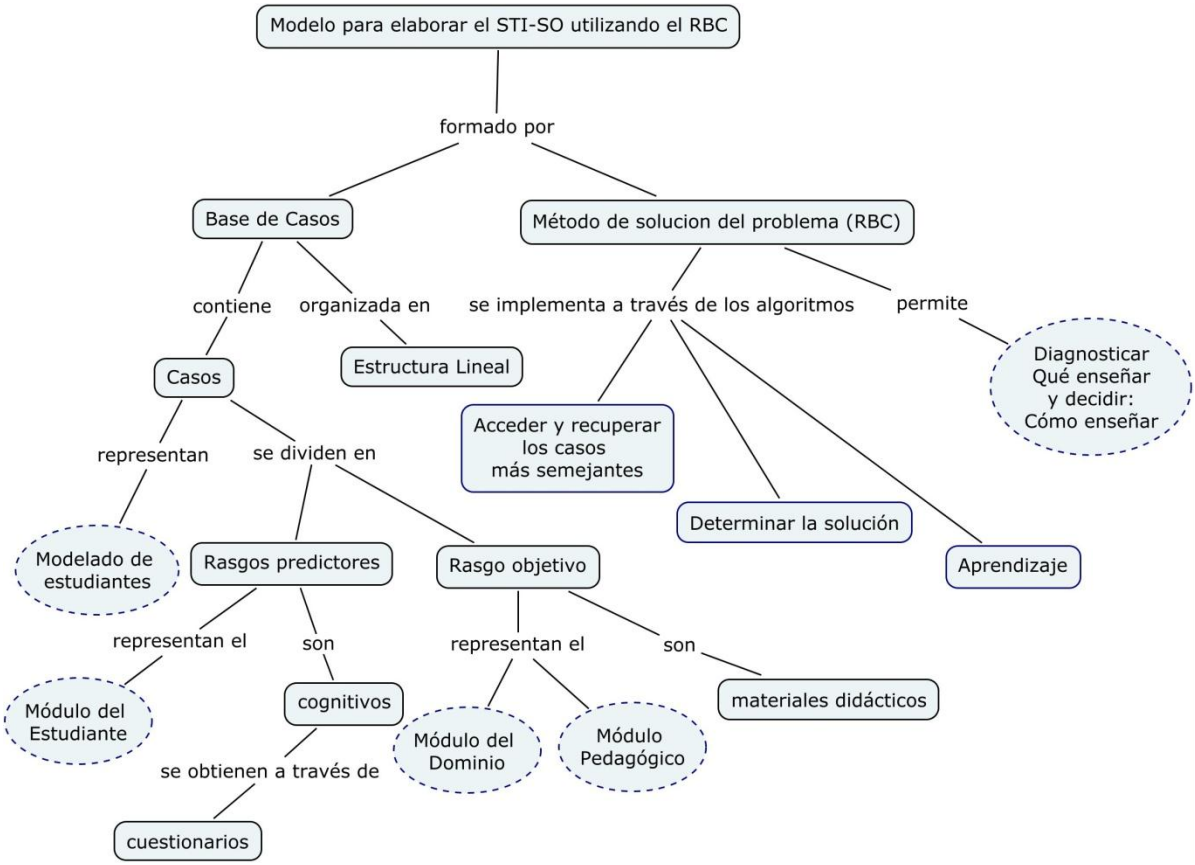

Figura 1: Mapa conceptual del modelo del STI-SO utilizando el RBC.

2.3. Personas relacionadas con el sistema

La siguiente tabla muestra la descripción de las personas relacionadas con el sistema que son todas aquellas que obtienen un resultado del mismo.

Tabla 2: Personas relacionadas con el sistema

Personas relacionadas con el sistema	Justificación
Usuario	Es cualquier persona que accede a la aplicación y se autentica y el sistema le da la posibilidad de realizar cuestionarios.

Profesor	Es aquel usuario al cual el Administrador otorga este rol, teniendo privilegios de gestionar documentos, gestionar cuestionarios y consultar los resultados de otros usuarios.
Administrador	Es la persona encargada de gestionar el sistema, administrando las cuentas de los usuarios registrados en la aplicación y gestionando el contenido publicado en el mismo.

2.4. Lista de Reserva del producto

Según define la metodología XP, la Lista de Reserva del producto (LRP) contiene los requisitos funcionales ordenados según la prioridad de implementación y los requisitos no funcionales del sistema.

Los requisitos funcionales, son las capacidades o condiciones que el sistema debe cumplir. El STI-SO debe permitir:

RF1. Autenticar usuario

RF2. Mostrar presentación

RF3. Gestionar usuario

RF4. Mostrar documentos

RF5. Gestionar documentos

RF6. Gestionar cuestionarios

RF7. Mostrar cuestionarios

RF8. Inferir la solución

RF9. Decidir si se almacena el nuevo caso

RF10. Consultar evaluación

Los requisitos no funcionales, son las propiedades o cualidades que el producto debe tener. Para un correcto funcionamiento del STI-SO, este debe tener los siguientes:

Software

Requerimientos mínimos para el cliente: debe tener instalado navegador Mozilla Firefox.

Requerimientos mínimos para el servidor: Se requiere la instalación del gestor de base de datos MySQL 5.0.45 , servidor web Apache 2.2.6, PHP 5.2.2, con la activación de los extensiones de php: ldap y soap.

Hardware

Los requerimientos mínimos para la ejecución de la aplicación se resumen en: Procesador a 2.0 GHz de velocidad de procesamiento o superior y 500 MB de RAM o superior y 5GB de espacio libre en disco duro.

Restricciones en el Diseño y la implementación

Se deben realizar validaciones de las entradas de los datos para que el usuario cometa el mínimo de errores, mostrar mensajes de envió y hacer uso del estándar de codificación que propone el framework Codeigniter.

Apariencia o interfaz externa

El logo debe estar ubicado en la parte superior izquierda del sitio, exponiendo el significado de sus siglas. El color de los textos debe contrastar con el del fondo y el tamaño de fuente debe ser lo suficientemente grande para que la información contenida en la misma sea fácil de leer.

Usabilidad

La interfaz de la aplicación debe ser sencilla, facilitando al usuario que encuentre fácil lo que desea y debe contar con la documentación necesaria para el propósito de misma.

Seguridad

La seguridad está dada por la **Confidencialidad** de los datos al autenticarse el usuario, pues la contraseña es de conocimiento personal y la **Integridad** ya que el sistema debe garantizar que la información se mantenga inalterable durante todo el proceso de uso del mismo.

2.5. Fase de exploración

El ciclo de vida ideal de un proyecto realizado con XP se inicia con la fase de Exploración, al final de la cual el equipo de desarrollo cuenta con suficiente material de trabajo traducido en HU, que se recopilan en esta etapa, como para producir una primera entrega. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

Historias de Usuario

Las HU son el primer paso de cualquier proyecto que siga la metodología XP. Tienen la misma finalidad que los casos de uso (CU) utilizados por la metodología RUP, pero con la diferencia que constan de 3 ó 4 líneas escritas por el cliente en un lenguaje no técnico sin muchos detalles, existiendo diferencias entre estas y la especificación de requisitos, como por ejemplo el nivel de detalles, ya que las HU solo contienen información sobre la estimación del riesgo y el tiempo que conllevará su implementación. El tiempo ideal para su desarrollo, es entre 1 y 3 semanas.

Clasificación de las Historias de Usuario

Las HU están compuestas por tablas divididas en las siguientes secciones:

Número: número de la HU.
Nombre: nombre que identifica la HU.
Usuario: nombre de usuarios involucrados en la HU.
Prioridad en el Negocio: se clasifican en **alta** (funcionalidades fundamentales en el desarrollo del sistema), **media** (funcionalidades a tener en cuenta pero que no tienen una afectación sobre el sistema) y **baja** (sirven de ayuda a los desarrolladores pero no tienen nada que ver con el sistema)

Nivel de Complejidad: también se clasifican en alta (un error en la implementación, lleva a la inoperatividad del código), media (un error en la implementación retrasa la entrega de la versión) y baja (el error se puede tratar sin que traiga problemas mayores)
Estimación: tiempo estimado que se demorará el desarrollo de la HU.
Iteración Asignada: número de la iteración en la que será realizada la HU.
Descripción: breve descripción de la HU.
Observaciones: a que requisito hace referencia la HU.

Durante este proceso se identifican 10 HU. A continuación una descripción de la HU Autenticar usuario, el resto de las descripciones se encuentran en los anexos. (Ver Anexo 2: Restantes Historias de Usuario)

Tabla 3: HU Autenticar usuario.

Historia de Usuario	
No: 1	**Nombre:** Autenticar usuario
Usuario: Usuario, Profesor, Administrador	
Prioridad en el Negocio: Alta	**Nivel de Complejidad:** Media
Estimación: 1 semana	**Iteración Asignada:** 1
Descripción: El usuario debe autenticarse para ser identificado. Si es el Administrador, puede gestionar usuario, gestionar documentos, gestionar cuestionario y consultar evaluaciones de otros usuarios. Si está registrado como Profesor, tendrá los mismos privilegios que el Administrador excepto gestionar usuario, de lo contario, solo podrá realizar cuestionarios.	
Observaciones: Se hace referencia al requisito R1.	

2.6. Fase de planificación

En esta fase el cliente establece la prioridad de cada HU, las cuales son organizadas en iteraciones, determinando un cronograma en conjunto con el cliente de la fecha de entrega de las mismas, cuya entrega debe obtenerse en no más de tres meses.

Plan de iteraciones

Todo proyecto que siga la metodología XP se divide en iteraciones, las cuales son relativamente cortas y no deben exceder las 4 semanas en su desarrollo. Las HU son divididas en tareas de entre 1 y 3 días que son asignadas a los programadores, creándose el plan de duración de cada una de las iteraciones, así como el orden en que serán implementadas dichas HU.

Tabla 4: Plan de Iteraciones

Iteraciones	Orden de las Historias de Usuario a implementar	Cantidad de tiempo de trabajo
1	Autenticar usuario Mostrar presentación Gestionar usuario	3 semanas
2	Mostrar documentos Gestionar documentos Gestionar cuestionario Mostrar cuestionarios	4 semanas
3	Inferir la solución	4 semanas
4	Decidir si se almacena el nuevo caso Consultar evaluación	2 semanas

2.7. Plan de Entregas

El plan de entregas establece qué HU serán agrupadas para conformar una entrega y el orden de las mismas. Se realiza en base a las estimaciones de tiempos de desarrollo realizadas por los programadores.

Tabla 5: Plan de Entregas

Sistema	Final de la Iteración 2 (4ta semana de abril del 2012)	Final de la Iteración 4 (1era semana de junio del 2012)
STI-SO	**Versión 1.0**	**Versión 1.0**

2.8. Conclusiones Parciales

En este capítulo se describió la propuesta de solución del sistema, identificando la lista de reserva del producto, las fases de Exploración y Planificación pertenecientes a la primera etapa del ciclo de vida de un proyecto siguiendo la metodología XP, detallando las HU identificadas, el plan de iteraciones y el plan de entrega.

3. Capítulo 3. Diseño, Implementación y Prueba

3.1. Introducción

En este capítulo se describe la fase de diseño, implementación y prueba desarrollada por la metodología XP, donde se detallan las tareas generadas por cada HU definida durante la fase de Planificación, se describen los patrones utilizados. Además son realizadas las pruebas, pues al finalizar cada iteración, el producto debe ser probado para ver si cumple con lo que realmente quiere el cliente y se realiza la validación del sistema propuesto.

3.2. Patrón de la arquitectura

Los patrones de la arquitectura de un sistema, definen la estructura fundamental con la que se va a desarrollar el software. Para el desarrollo del STI-SO el patrón que se utilizó es el Modelo-Vista-Controlador (MVC), ya que CodeIgniter se basa en este patrón.

El **MVC** es típicamente utilizado para la creación de aplicaciones web y no sólo CodeIgniter lo implementa, sino también otra serie de framework de desarrollo web, en PHP u otros lenguajes. Separa en varios grupos las complejidades de las distintas partes que componen una página web, como la vista y la lógica, así como el acceso a la base de datos. [22] Para un mejor entendimiento del funcionamiento del framework CodeIgniter se muestran a continuación la representación de las clases divididas por las capas de la arquitectura MVC.

Vistas

La vista codifica y mantiene la presentación final de la aplicación, se coloca todo el código HTML, CSS, Javascript, etc., que se tiene que generar para producir la página para que sea vista como la quiere el usuario. [22]

Para una mejor organización del sistema, los archivos vista son almacenados dentro de subcarpetas. Dentro de la carpeta views se encuentran todas las vistas, la cual a su vez, tiene dentro varias subcarpetas que contienen otras vistas.

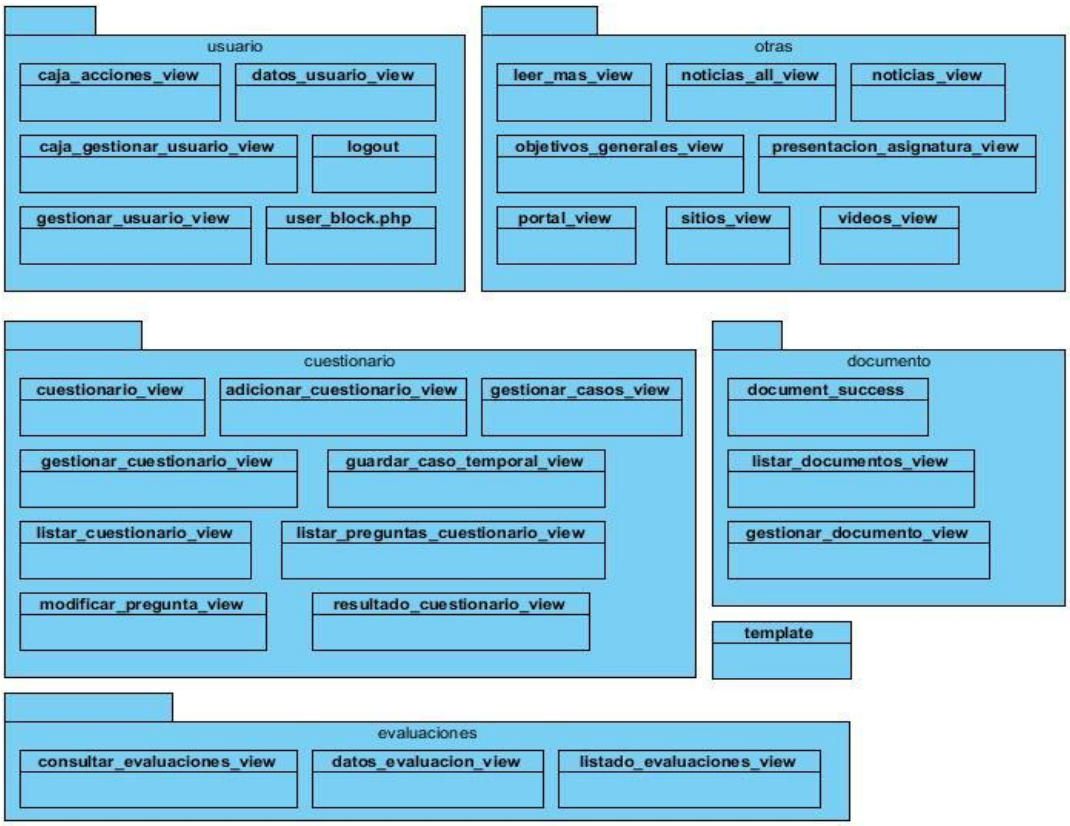

Figura 2: Capa Vista

template: es la plantilla que se utiliza para mostrar todo el contenido del sitio.

En la carpeta **cuestionario** se encuentran las siguientes vistas: adicionar_cuestionario_view, cuestionario_view, gestionar_casos_view, gestionar_cuestionario_view, guardar_caso_temporal_view, listar_cuestionario_view, listar_preguntas_cuestionario_view, modificar_pregunta_view, resultado_cuestionario_view, que permiten gestionar los cuestionarios y mostrar el listado de los cuestionarios.

En la carpeta **documento** se encuentran las siguientes vistas: gestionar_documento_view, listar_documentos_view, document_success que permiten gestionar los documentos y mostrar el listado de los documentos.

En la carpeta **usuario** se encuentran las siguientes vistas: caja_acciones_view, user_block, logout, caja_gestionar_usuario_view, datos_usuario_view, gestionar_usuario_view, que permiten gestionar los usuarios.

En la carpeta **evaluaciones** se encuentran las siguientes vistas: consultar_evaluaciones_view, datos_evaluacion_view, listado_evaluaciones_view, que permiten consultar las evaluaciones realizadas por los usuarios.

En la carpeta **otras** se encuentran otras vistas que son necesarias para mostrar informaciones sobre la asignatura, vínculos a otros sitios, noticias y videos que se encuentran en la presentación del sitio, las cuales son: leer_mas_view, noticias_all_view, noticias_view, objetivos_generales_view, portal_view, presentacion_asignatura_view, sitios_view, videos_view.

Controladoras

El controlador hace de enlace entre el modelo, la vista y cualquier otro recurso que se tenga que procesar en el servidor para generar la página web. En el controlador se guarda la lógica de la página y se realizan todas las acciones que sean necesarias para generarlas, ayudados del modelo o la vista. [22]

Figura 3: Capa Controladora

Cargar_vistas: clase encargada de cargar las vistas de la presentación del sitio.

Caso: clase encargada de controlar la gestión de los casos en el sistema.

Cuestionario: clase encargada de controlar la gestión de los cuestionarios.

Documento: clase encargada de controlar la gestión de los documentos.

Evaluacion: clase encargada de controlar el registro de las evaluaciones de los usuarios.

Portal: clase controladora que permite cargar la vista que muestra la interfaz principal.

Usuario: clase encargada de controlar la gestión de los usuarios.

Noticias: clase encargada de controlar la gestión de las noticias que aparecen en la presentación.

Modelo

Las clases modelos tienen el código referente al acceso a base de datos, donde se crean funciones para recibir, insertar, actualizar o eliminar información de las tablas. Al mantenerse todas las llamadas a la base de datos en un mismo código, desde otras partes del programa se pueden invocar las funciones que se necesiten del modelo y éste se encargará de procesarlas. [22]

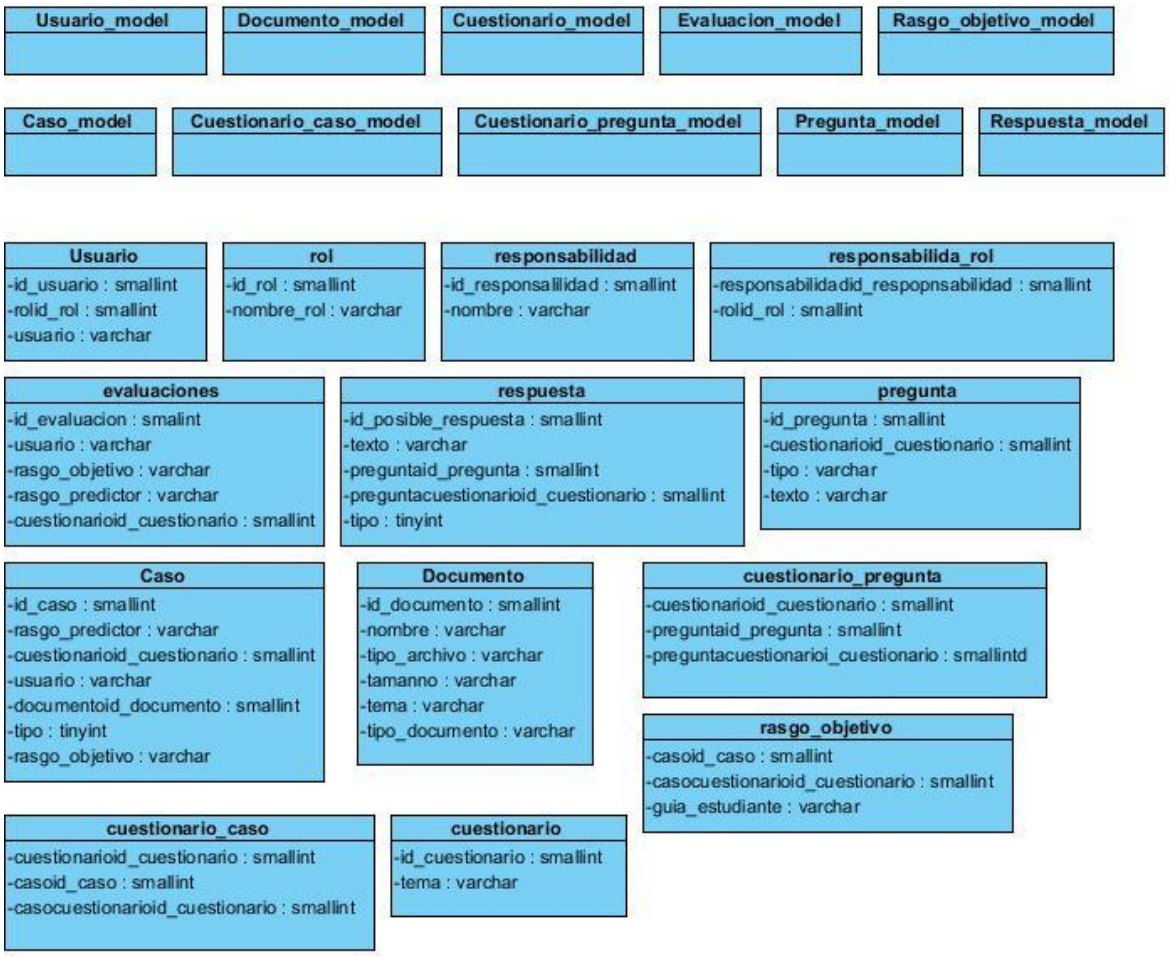

Figura 4: Capa Modelo

usuario: tabla encargada de registrar los usuarios.

rol: tabla encargada de registrar el rol que ocupan los usuarios.

responsabilidad: tabla encargada de registrar las responsabilidad que tiene los usuarios.

responsabilidad_rol: tabla encargada de registrar la relación entre el rol y la responsabilidad de los usuarios.

cuestionario: tabla encargada de registrar la información de los cuestionarios.

cuestionario_caso: tabla encargada de registrar la relación entre los casos y los cuestionarios.

cuestionario_pregunta: tabla encargada de registrar la relación entre los cuestionarios y las preguntas.

pregunta: tabla encargada de registrar las preguntas de los cuestionarios.

respuesta: tabla encargada de registrar las respuestas del cuestionario.

caso: tabla encargada de almacenar los casos.

documento: tabla encargada de registrar los documentos que se muestran en el listado de documentos.

evaluaciones: tabla encargada de registrar las evaluaciones realizadas.

rasgo_objetivo: tabla encargada de registrar los materiales propuestos para cada caso.

Usuario_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla usuario.

Documento_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla documento.

Cuestionario_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla cuestionario.

Evaluacion_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla evaluaciones.

Caso_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla casos.

Pregunta_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla pregunta.

Respuesta_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla respuesta.

Cuestionario_pregunta_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla cuestionario_pregunta.

Cuestionario_caso_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla cuestionario_caso.

Rasgo_objetivo_model: clase que tiene el código referente al acceso a la base de datos, donde se crean funciones para gestionar información de la tabla rasgo_objetivo.

Patrones de diseño

Patrones de diseño son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Su utilización permite ahorrar grandes cantidades de tiempo en la construcción de software, siendo más fácil de comprender, mantener y extender. [33]

Los patrones de **Asignación de Responsabilidades** (GRASP) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. [34] Dentro de los patrones GRASP utilizados para el desarrollo del STI-SO, se encuentran los siguientes:

El patrón **Experto** es el encargado de asignar una responsabilidad al experto en información, no es más que la clase que cuenta con la información necesaria para cumplir la responsabilidad. Esto se evidencia en las clases modelos y las clases controladoras. Por ejemplo, la clase controladora cuestionario, será la encargada de insertar, modificar y eliminar un cuestionario.

El patrón **Controlador** es el encargado de asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. Un ejemplo de esto se evidencia en la clase portal, ya que es la controladora que se encarga de mostrar la página de inicio del sistema.

El patrón **Bajo Acoplamiento** es el encargado de asignar una responsabilidad para mantener bajo acoplamiento con el objetivo de tener las clases lo menos ligadas entre sí, de forma que si se realiza modificación en alguna de ellas, tenga la mínima repercusión en el resto de clases. Esto se evidencia en todas las clases, ya que un cambio en las clases modelo no implica cambios en las controladoras ni en las vistas.

El patrón **Alta Cohesión** es el encargado de asignar una responsabilidad de modo que la cohesión siga siendo alta, con el objetivo de que cada una de las clases tenga responsabilidades específicas, por lo que cada clase tiene determinadas responsabilidades y colabora con otras para llevarlas a cabo, como por ejemplo las clases controladoras modifican los datos haciendo uso de las clases modelos, las cuales solo tienen la responsabilidad de hacer cambios en la base de datos.

3.3. Tarjetas Clase-Responsabilidad-Colaborador

La metodología XP sugiere diseños simples y sencillos para conseguir que sea fácil y entendible, logrando menos tiempo y esfuerzo a la hora de desarrollar. No requiere la presentación del sistema mediante diagramas de clases utilizando notación UML, en su lugar se usan otras técnicas como las tarjetas CRC (Clase-Responsabilidad-Colaborador), con el objetivo de organizar de forma simple, las clases más relevantes para las funcionalidades del sistema.

Las tarjetas CRC se dividen en tres secciones: **nombre de la clase**, **colaboradores** (otras clases con las que trabaja en conjunto para llevar a cabo sus funcionalidades) y las **responsabilidades** (lo que la clase sabe o hace).

Para el STI-SO, la mayoría de las clases son importantes para su correcto funcionamiento, por lo que a continuación se muestran las tarjetas CRC correspondientes a algunas de las clases controladoras.

Tabla 6: Tarjeta CRC clase Usuario

Clase: Usuario	
Descripción: clase que controla la gestión de los usuarios.	
Responsabilidad	**Colaborador**
Listar usuarios.	caja_acciones_view, caja_gestionar_usuario_view, datos_usuario_view, gestionar_usuario_view, Usuario_model
Insertar nuevo usuario.	
Modificar rol del usuario.	
Eliminar usuario.	

Tabla 7: Tarjeta CRC clase Cuestionario

Clase: Cuestionario	
Descripción: clase que controla la gestión de los cuestionarios.	
Responsabilidad	**Colaborador**
Listar los cuestionarios.	adicionar_cuestionario_view, cuestionario_view, gestionar_cuestionario_view, listar_cuestionario_view, listar_preguntas_cuestionario_view, modificar_pregunta_view, Cuestionario_model, Pregunta_model, Respuesta_model
Insertar cuestionario.	
Modificar cuestionario.	
Eliminar cuestionario.	

Tabla 8: Tarjeta CRC clase Documento

Clase: Documento	
Descripción: clase encargada de controlar la gestión de los documentos.	
Responsabilidad	**Colaborador**

Listar documentos.	gestionar_documento_view, listar_documentos_view Documento_model.
Insertar documentos.	
Eliminar documentos.	
Abrir y Guardar documentos.	

Tabla 9: Tarjeta CRC clase Caso

Clase: Caso	
Descripción: clase encargada de controlar la gestión de los casos en el sistema.	
Responsabilidad	**Colaborador**
Generar caso	Caso_model, Respuesta_model, Pregunta_model, Cuestionario_model, Cuestionario_pregunta_model, Usuario_model, Documento_model.
Guardar caso	
Eliminar caso	

Tabla 10: Tarjeta CRC clase Evaluacion

Clase: Evaluacion	
Descripción: clase encargada de controlar el registro de las evaluaciones de los usuarios.	
Responsabilidad	**Colaborador**
Guardar evaluaciones de los usuarios.	consultar_evaluaciones_view, datos_evaluacion_view, listado_evaluaciones_view, Respuesta_model, Pregunta_model, Cuestionario_pregunta_model, Caso_model, Evaluacion_model, Cuestionario_model, Usuario_model.

3.4. Modelo Físico de la Base de Datos

El diseño de la base de datos es fundamental para el desarrollo de cualquier aplicación. A continuación se muestra el modelo físico de la base de datos para el STI-SO.

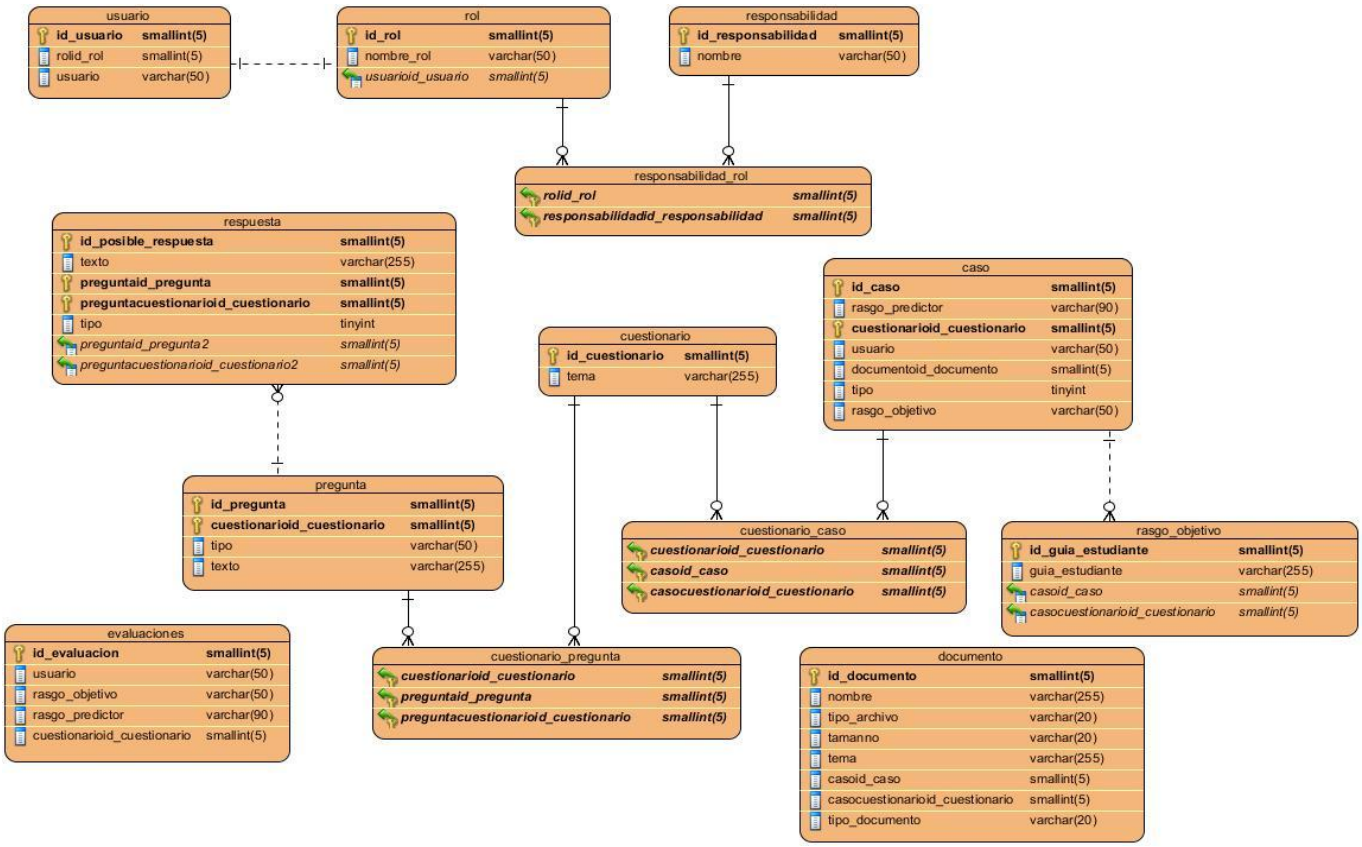

Figura 5: Modelo físico de la Base de Datos.

3.5. Despliegue del sistema

El diagrama de despliegue es un tipo de diagrama del UML que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones. Muestra la configuración de los nodos que participan en la ejecución y de los componentes que residen en ellos. Un nodo es un recurso de ejecución como un computador, dispositivo, conectados por asociaciones de comunicación tales como enlaces de red, conexiones HTTP, etc. [35]

A continuación el diagrama de despliegue para el STI-SO:

Figura 6: Diagrama de Despliegue

3.6. Tareas de Ingeniería

Durante la fase de implementación siguiendo la metodología XP, se realiza la implementación de las HU en sus respectivas iteraciones, dividiendo las mismas en tareas de ingeniería que serán implementadas por los desarrolladores, las cuales pueden ser escritas en lenguaje técnico y no necesariamente entendible por el cliente. A continuación se enumeran las tareas de ingeniería por cada HU en cada una de las iteraciones.

Iteración 1

El principal objetivo de esta iteración es desarrollar las HU 1. Autenticar usuario, 2. Mostrar presentación, y 3. Gestionar usuario.

Para la **HU 1. Autenticar usuario** se desarrollan las siguientes tareas:

Tarea Nº 1: Comprobar existencia del usuario en LDAP y rol que ocupa en la base de datos.
Tarea Nº 2: Mostrar sección correspondiente y guardar datos en la base de datos.
Tarea Nº 3: Salir del sistema en cualquier momento.

Para la **HU 2. Mostrar presentación** se desarrollan las siguientes tareas:

Tarea Nº 4: Mostrar objetivos generales, significado de STI-SO, presentación de la asignatura y vínculo a otros sitios.
Tarea Nº 5: Mostrar videos y noticias de la asignatura.

Para la **HU 3. Gestionar usuario** se desarrollan las siguientes tareas:

Tarea Nº 6: Mostrar un listado de todos los usuarios que ocupan el rol de Administrador y Profesor en la base de datos.
Tarea Nº 7: Modificar el rol que ocupa un usuario especifico en la base de datos.
Tarea Nº 8: Insertar un nuevo usuario en la base de datos con su rol correspondiente.
Tarea Nº 9: Eliminar un usuario especificado de la base de datos.

Iteración 2

El principal objetivo de esta iteración es desarrollar las HU 4. Mostrar documentos, 5. Gestionar documentos, 6. Gestionar cuestionarios y 7. Mostrar cuestionarios.

Para la **HU 4. Mostrar documentos** se desarrollan las siguientes tareas:

Tarea Nº 10: Mostrar lista de documentos.
Tarea Nº 11: Abrir y guardar documentos.

Para la **HU 5. Gestionar documentos** se desarrollan las siguientes tareas:

Tarea Nº 12: Insertar documentos.
Tarea Nº 13: Eliminar documentos.

Para la **HU 6. Gestionar cuestionarios** se desarrollan las siguientes tareas:

Tarea Nº 14: Insertar cuestionarios.
Tarea Nº 15: Modificar cuestionario.
Tarea Nº 16: Eliminar cuestionario.

Para la **HU 7. Mostrar cuestionarios** se desarrollan las siguientes tareas:

Tarea Nº 17: Mostrar lista de cuestionarios.
Tarea Nº 18: Abrir y responder cuestionarios.
Tarea Nº 19: Enviar respuestas del cuestionario.

Iteración 3

El principal objetivo de esta iteración es desarrollar la HU 8. Inferir la solución.

Para la **HU 8. Inferir la solución** se desarrollan las siguientes tareas:

Tarea Nº 20: Comparar rasgos predictores y asociar importancia a cada rasgo.
Tarea Nº 21: Guardar rasgo predictor con datos del usuario como caso temporal.
Tarea Nº 22: Aplicar función de semejanza al nuevo caso con cada caso de la base de casos.
Tarea Nº 23: Determinar caso más parecido y recuperar los datos de ese caso.
Tarea Nº 24: Adaptar la solución al nuevo caso.
Tarea Nº 25: Mostrar rasgo objetivo del nuevo caso.
Tarea Nº 26: Guardar resultados.

Iteración 4

El principal objetivo de esta iteración es desarrollar la HU 9. Decidir si se almacena el nuevo caso y 10. Consultar evaluación.

Para la **HU 9. Decidir si se almacena el nuevo caso** se desarrollan las siguientes tareas:

Tarea Nº 27: Mostrar listado de casos temporales.
Tarea Nº 28: Guardar caso temporal.
Tarea Nº 29: Eliminar caso temporal.

Para la **HU 10. Consultar evaluación** se desarrollan las siguientes tareas:

Tarea Nº 30: Buscar evaluaciones de un usuario.
Tarea Nº 31: Mostrar resultados de evaluación seleccionada.

Las tareas de la ingeniería serán representadas mediante tablas divididas en las siguientes secciones:

Número de Tarea: número de la tarea que debe ser consecutivo.
Número de HU: número de la HU a la que pertenece la tarea.
Nombre de la Tarea: nombre de la tarea.
Tipo de Tarea: las tareas pueden ser de: Desarrollo, Corrección, Mejora u otra.

Estimación: tiempo estimado que se le asignará a al desarrollo de la tarea.
Fecha Inicio: fecha en que inicia el desarrollo de la tarea.
Fecha Fin: fecha en que termina el desarrollo de la tarea.
Programador Responsable: nombre y apellidos del programador.
Descripción: breve descripción de la tarea.

A continuación se detalla la primera tarea de ingeniería para la iteración 1 y el resto se encuentra en los anexos. (Ver Anexo 3: Restantes tareas de ingeniería)

Tabla 11: Tarea de ingeniería 1. Comprobar existencia del usuario en LDAP y rol que ocupa en la base de datos.

Tarea de ingeniería	
Número de tarea: 1	**Número de HU**: 1
Nombre de la tarea: Comprobar existencia del usuario en LDAP y rol que ocupa en la base de datos.	
Tipo de tarea: Desarrollo	**Estimación**: 2 días
Fecha inicio: 5/3/2012	**Fecha fin:** 6/3/2012
Programador responsable: Luisa Idorka Morales Rodríguez y Yarisleydi de Ávila Fernández	
Descripción: El sistema comprueba la existencia del usuario en LDAP y el rol que ocupa en la base de datos para crear la sección correspondiente.	

Como resultado se obtiene el STI-SO cuya interfaz principal se muestra en los anexos. (Ver Anexo 4: Interfaz principal del sistema)

3.7. Pruebas

La metodología XP divide las pruebas en dos grupos: pruebas unitarias, desarrolladas por los programadores, con el objetivo de verificar el código y las pruebas de aceptación para evaluar si al final de cada iteración se obtuvo la funcionalidad requerida, comprobando que sea la esperada por el cliente.

Para el STI-SO las **pruebas unitarias** fueron desarrolladas constantemente una vez terminaba de implementar cada funcionalidad cumpliendo con los requisitos planteados.

Las **pruebas de aceptación** son creadas en base a las HU en cada ciclo de la iteración del desarrollo, donde una HU puede tener todas las pruebas de aceptación que se necesite para asegurar su correcto funcionamiento. El cliente debe especificar uno o diversos escenarios para comprobar que la HU ha sido correctamente implementada y en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una HU no se puede considerar terminada hasta que no pase correctamente todas las pruebas de aceptación.

Estas pruebas son representadas mediante tablas divididas por las siguientes secciones:

Código: código que identifica el caso de prueba.
Historia de Usuario: número de la HU que hace referencia el caso de prueba.
Nombre: nombre de la funcionalidad a la que se le hace la prueba.
Descripción: breve descripción del caso de prueba.
Condiciones de Ejecución: condiciones necesarias para ejecutar la prueba.
Entrada/ Pasos de ejecución: valores de entrada.
Resultado Esperado: salida de la ejecución.
Evaluación de la Prueba: evaluación de la prueba.

A continuación la prueba realizada a la HU Autenticar usuario, el resto se encuentra en los anexos. (Ver Anexo 5: Restantes pruebas de aceptación)

Tabla 12: Caso de prueba de aceptación 1. Autenticar usuario

Caso de prueba de aceptación
Código: HU1_P1 **Historia de Usuario: 1**
Nombre: Autenticar usuario
Descripción: Prueba para la funcionalidad Autenticar usuario
Condiciones de Ejecución: El usuario debe introducir usuario y contraseña
Entrada/ Pasos de ejecución: El usuario introduce usuario y contraseña y oprime el botón Entrar. El sistema comprueba su existencia en LDAP y el rol que ocupa en la base de datos para mostrar sección correspondiente al usuario autenticado.

Resultado Esperado: El usuario es identificado por el sistema y permite ver la interfaz correspondiente al rol que ocupa en la base de datos.
Evaluación de la Prueba: Prueba satisfactoria.

Resultado de las pruebas

Luego de realizarse las pruebas de aceptación por cada iteración asignada, se obtienen los siguientes resultados de no conformidades:

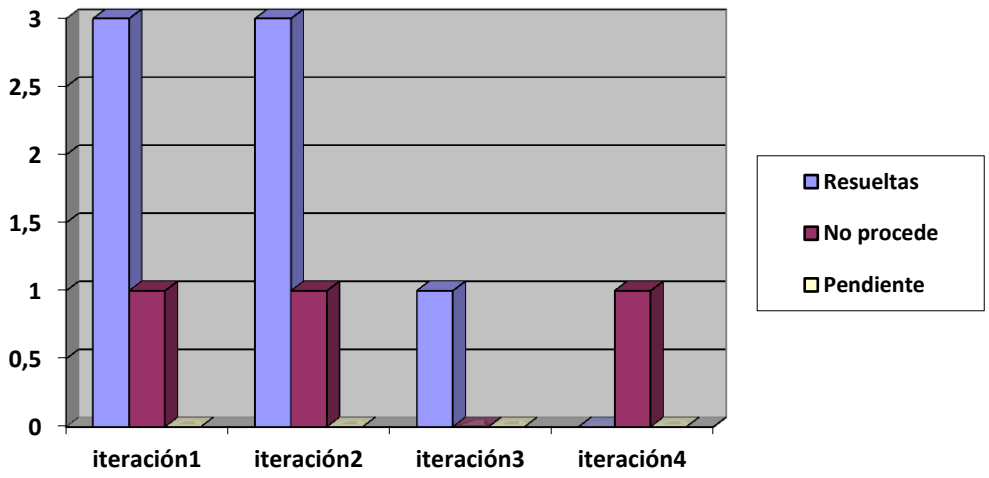

Figura 7. Resultado de las pruebas de aceptación

3.8. Validación

El sistema almacena en su base de casos un total de 26 casos que a continuación se muestran, compuesta por los rasgos predictores con su importancia asociada y la cantidad de veces que aparece el valor del rasgo (0/1).

Base de Casos		
Rasgos Predictores	Importancia	Valor/Cantidad
Pregunta 1	0.1	1/21 y 0/5
Pregunta 2	0.1	1/12 y 0/14
Pregunta 3	0.1	1/21 y 0/5
Pregunta 4	0.2	1/22 y 0/4

Pregunta 5	0.2	1/14 y 0/12
Pregunta 6	0.2	1/25 y 0/2
Pregunta 7	0.3	1/23 y 0/3
Pregunta 8	0.3	1/15 y 0/11
Pregunta 9	0.3	1/23 y 0/3
Pregunta 10	0.4	1/24 y 0/2
Pregunta 11	0.4	1/15 y 0/11
Pregunta 12	0.4	1/24 y 0/2
Pregunta 13	0.5	1/24 y 0/2
Pregunta 14	0.5	1/16 y 0/10
Pregunta 15	0.5	1/25 y 0/1

Tabla 13: Ejemplo de base de casos

Se propone un rasgo objetivo (Guía de estudio) por cada caso, los cuales se muestran a continuación:

Rasgos objetivos	
Guías de estudio	Si se equivoca en las siguientes preguntas
Documento 1	de la 1 a la 3
Documento 2	de la 4 a la 6
Documento 3	de la 7 a la 9
Documento 4	de la 10 a la 12
Documento 5	de la 13 a la 15
Documento 1-2	de la 1 a la 3 y de la 4 a la 6
Documento 1-3	de la 1 a la 3 y de la 7 a la 9
Documento 1-4	de la 1 a la 3 y de la 10 a la 12
Documento 1-5	de la 1 a la 3 y de la 13 a la 15
Documento 2-3	de la 4 a la 6 y de la 7 a la 9
Documento 2-4	de la 4 a la 6 y de la 10 a la 12
Documento 2-5	de la 4 a la 6 y de la 13 a la 15
Documento 3-4	de la 7 a la 9 y de la 10 a la 12
Documento 3-5	de la 4 a la 6 y de la 7 a la 9
Documento 4-5	de la 10 a la 12 y de la 13 a la 15
Documento 1-2-3	de la 1 a la 3, de la 4 a la 6 y de la 7 a la 9
Documento 1-3-4	de la 1 a la 3, de la 7 a la 9 y de la 10 a la 12
Documento 1-4-5	de la 1 a la 3, de la 10 a la 12 y de la 13 a la 15
Documento 1-2-5	de la 1 a la 3, de la 4 a la 6 y de la 13a la 15
Documento 1-2-3-4	de la 1 a la 3, de la 4 a la 6 , de la 7 a la 9 y de la 10 a la 12
Documento 1-3-4-5	de la 1 a la 3, de la 7 a la 9, de la 10 a la 12 y de la 13 a la 15
Documento 1-2-4-5	de la 1 a la 3, de la 4 a la 6, de la 10 a la 12 y de la 13 a la 15
Documento 1-2-3-5	de la 1 a la 3, de la 4 a la 6 y de la 7 a la 9
Documento 1-2-3-4-5	de la 1 a la 3, de la 4 a la 6, de la 7 a la 9, de la 10 a la 12 y de la 13 a la 15

Documento Especial	en ninguna o casi ninguna

Tabla 14: Descripción de rasgos objetivos

Para validar la propuesta de solución se tomó una serie de casos de prueba:

Casos de Prueba					
Rasgos Predictores	Caso 1	Caso 2	Caso 3	Caso 4	Caso 5
Pregunta 1	1	1	0	0	0
Pregunta 2	0	1	1	1	0
Pregunta 3	1	1	1	1	1
Pregunta 4	1	1	1	1	1
Pregunta 5	0	1	1	1	1
Pregunta 6	1	1	1	0	1
Pregunta 7	1	1	1	1	1
Pregunta 8	0	0	1	1	1
Pregunta 9	0	1	1	0	1
Pregunta 10	1	1	1	1	1
Pregunta 11	1	1	1	0	1
Pregunta 12	1	1	1	1	1
Pregunta 13	1	1	1	1	1
Pregunta 14	1	0	1	0	1
Pregunta 15	1	1	1	1	1

Tabla 15: Casos de Prueba

Dichos casos al probarlos en el sistema arrojaron los siguientes resultados:

Resultado de las validación		
Casos de Prueba	Resultado Esperado	Resultado alcanzado
Caso 1	Documento 1-2-3	Documento 1-2-3
Caso 2	Documento 3-5	Documento 3-5
Caso 3	Documento Especial	Documento Especial
Caso 4	Documento 1-2-3-4-5	Documento 4-5
Caso 5	Documento 1	Documento 1

Tabla 16: Resultado de la validación

Al ser la base de casos insuficiente, pues no cuenta con todas las combinaciones posibles por cada respuesta que se pueda dar a las preguntas, el resultado no siempre coincide con la opinión de los expertos como se puede ver en el resultado del caso 4, que aunque no fue el resultado esperado, se acercó bastante a la solución, ya que fue el que encontró más semejante.

3.9. Conclusiones Parciales

En este capítulo se abordó la etapa de diseño, implementación y prueba del STI-SO, describiendo la arquitectura MVC que utiliza el framework CodeIgniter, identificando los patrones de diseño, así como las tareas de ingeniería necesarias por cada HU, el diagrama de despliegue del mismo, las pruebas realizadas a la aplicación y la validación de la propuesta de solución.

Conclusiones

Durante la investigación se realizó un estudio que permitió tener conocimiento sobre el desarrollo de los STI, su estructura y funcionamiento, así como la forma de representar el conocimiento, escogiendo la más apropiada para el desarrollo del sistema propuesto. Se caracterizaron las principales metodologías, lenguajes y herramientas empleados para el desarrollo de la aplicación, fundamentando en cada caso la selección. Se propone una guía de orientación que facilita la creación del STI. Se identificaron las HU, determinando plan de iteraciones, fechas de entrega y tareas de ingeniería por cada una, posibilitando un trabajo ordenado y bien planificado para dar entrega del sistema en tiempo, llevándose a cabo satisfactoriamente cada una de las tareas que fueron trazadas para darle cumplimiento al problema a resolver y al objetivo general.

La etapa de prueba fue de gran importancia, pues se comprobó el adecuado funcionamiento del sistema, logrando la satisfacción del cliente.

Una vez finalizado el desarrollo del presente trabajo, se concluye con la creación de una aplicación web, capaz de brindar una estrategia de estudio dada por los resultados de la realización de cuestionarios que ayudan a consolidar el conocimiento, contando para ello con la bibliografía adecuada.

Teniendo en cuenta los resultados obtenidos, se concluye que los objetivos presentados al inicio de la investigación han sido cumplidos en su totalidad, garantizando de esta forma un STI de apoyo al proceso de enseñanza-aprendizaje de la asignatura SO para la Facultad 2.

Recomendaciones

Para futuras versiones del sistema se recomienda:

- ✓ Tener en cuenta los rasgos afectivos, motivacionales u otros para determinar la estrategia de estudio más conveniente.
- ✓ La base de casos debe ser capaz de determinar por sí sola la incorporación de nuevos casos sin consulta con el experto, logrando así el autoaprendizaje de la misma.
- ✓ Agregar otros tipos de preguntas al cuestionario para enriquecer la obtención del estado cognitivo de los usuarios.
- ✓ Extender el desarrollo del sistema para todos los temas definidos en el Plan de Estudios de la asignatura.
- ✓ Proponer soluciones similares para otras asignaturas que sirvan de apoyo al proceso de enseñanza-aprendizaje del estudiante en general.

Referencias bibliográficas

1. Franklin Parra, C.M.Q., ed. *Sistema Tutorial Inteligente*. ed. C.d.I.C.y.T. (CICYT). 2010, Escuela Superior Politécnica del Litoral.: Guayaquil, Ecuador.

2. Zulma Cataldi, F.J.l., *Sistemas Tutores Inteligentes Orientados a la Enseñanza para la Comprensión*, in *EDUTEC Revista Electrónica de Tecnología Educativa*. 28 marzo 2009: Facultad Regional Buenos Aires. Argentina.

3. Yharlan Alex Rojas Correa, T.A.M., *MENTOR: Sistema Tutorial Inteligente para el desarrollo de habilidades en la solución de problemas matemáticos*, in *Revista de Investigación Universidad La Salle*. julio-diciembre de 2007. p. pp. 235-246. .

4. Natalia Martínez Sánchez, Z.Z.G.V., Maikel León Espinosa,Denys de Medina Sotolongo,Mateo Lezcano Brito,Gheisa Ferreira Lorenzo,Lydia Rosa Ríos Rodríguez,Yolanda Soler Puig,José M. Linares Álvaro (2009) *VIR 167 Representación y organización del conocimiento en Software educativo*.

5. Natalia Martínez Sánchez, O.R.F.M., *Guía de orientación para la ingeniería del conocimiento en el diseño de sistemas de enseñanza-aprendizaje inteligentes utilizando el razonamiento basado en casos*, in *Revista Generación Digital*. Enero de 2011.

6. *Reglas de Producción*. 30 enero 2011 Available from: http://es.scribd.com/doc/90856631/REGLAS-DE-PRODUCCION.

7. Gil, M.d.P.G., *TEMARIO DE CURSO C291-69 TÓPICOS AVANZADOS: REDES NEURONALES ARTIFICIALES INAOE.* , in *Coordinación de computación, Lab. de "Machine Learning and PatternRecognition". Grupo de investigación en Redes Neuronales Artificiales.* Versión: 17-ene-2012.

8. *Sistemas expertos conexionistas*. jueves 4 de febrero de 2010; Available from: http://menteerrabunda.blogspot.com/2010/01/sistemas-expertos-conexionistas.html.

9. Arias S, F.J.J.B., Jovani A.; Ovalle C., Demetrio A. , *Revista Avances en Sistemas e Informática* in *Redalyc Sistema de Información Científica*. junio 2009, Red de Revistas Científicas de América

Latina, el Caribe, España y Portugal: Universidad Nacional de Colombia. Modelo de planificación instruccional en sistemas tutoriales inteligente. p. pp. 155-164. .

10. Ivar Jacobson, G.B., James Rumbaugh. *El proceso unificado de desarrollo.La guia completa del Proceso escrita por sus creadores*. 20 abril 2010; Available from: http://es.scribd.com/doc/30251931/El-Proceso-Unificado-de-Desarrollo-de-Soft-Jacobson. .

11. Laura Carabalí, J.G., Daniela Loboa. *Programación Extrema*. Jun 13, 2010; Available from: http://www.slideshare.net/urumisama/programacin-extrema-4488942.

12. *INGENIERÍA DEL SOFTWARE:METODOLOGÍAS Y CICLOS DE VIDA. Laboratorio Nacional de Calidad del Software.* primera versión editada en Marzo de 2009; Available from: http://es.scribd.com/doc/62931905/45/Extreme-Programming-XP#outer_page_39.

13. *Ingeniería del software UML - Definición, historia y especificaciones* 01/03/2009.; Available from: http://www.clubdesarrolladores.com/articulos/mostrar/68-uml-definicion-historia-y-especificaciones

14. Mehdi Achour, F.B., Antony Dovgal, Nuno Lopes, Hannes Magnusson, Georg Richter, Damien Seguy, Jakub Vrana. *PHP Manual* 2012; Available from: http://www.php.net/manual/en/intro-whatcando.php

15. *Java*. 12 Ago 2011 Available from: http://codigoprogramacion.com/java/47-introjava.html

16. Pérez, J.E. *Introducción a JavaScript*. 04-05-2009; Available from: http://www.librosweb.es/javascript/index.html

17. Rubira, J. *Google Libraries API, un repositorio de las librerías Javascript más utilizadas*. 14 de julio de 2011; Available from: http://www.genbetadev.com/desarrollo-web/google-libraries-api-un-repositorio-de-las-librerias-javascript-mas-utilizadas

18. *Jquery write less, do more*. 2012; Available from: http://jquery.com/.

19. *Tutorial JavaScript sin dolor usando Prototype (2da parte)*. 2012; Available from: http://www.tutorial-enlace.net/tutorial-JavaScript_sin_dolor_usando_Prototype_(2da_parte)-16619.html

20. Fuentes, A. *La web como plataforma con Dojo*. 11/10/09; Available from: http://www.slideshare.net/afuentes/la-web-como-plataforma-con-dojo-toolkit

21. FabienPotencier, F. *Symfony la guía definitiva*. 25-08-2009 Available from: http://www.librosweb.es/symfony.

22. Alvarez., M.A. *Manual de CodeIgniter (19 capítulos)*. 11 abril 2010 Available from: http://www.desarrolloweb.com/manuales/manual-codeigniter.html.

23. *IDE de Programación* 20 marzo 2012 Available from: http://www.ecured.cu/index.php/IDE_de_Programaci%C3%B3n#Ejemplos_de_IDE_de_programaci.C3.B3n

24. *NetBeans* 2012; Available from: http://netbeans.org/

25. *Eclipse, entorno de desarrollo integrado*. 20 de marzo de 2012; Available from: http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado.

26. wil63r. octubre 5, 2011; Available from: http://aplicaciones.org/eclipse-ide-de-desarrollo-open-source/

27. *Herramienta CASE* 20 de marzo de 2012; Available from: http://www.ecured.cu/index.php/Herramienta_CASE

28. *Rational Rose* 1 Dic 2011; Available from: http://es.scribd.com/doc/74347179/Rational-Rose

29. *Visual Paradigm*. 6 de febrero de 2012; Available from: http://www.ecured.cu/index.php/Visual_Paradigm

30. *Características MySQL* 18 de agosto de 2009; Available from: http://www.anadicsinaloa.com/index.php?option=com_content&view=article&id=207:caracteristicas-mysql&catid=16:anadic-sinaloa&Itemid=33

31. *PostgreSQL Portal en español sobre Postgre* 02/10/2010 Available from: http://www.postgresql.org.es/sobre_postgresql

32. *Apache*. 2012; Available from: http://www.ecured.cu/index.php/Servidor_HTTP_Apache

33. *El Club del Programador. Tutoriales, tips, herramientas y recursos para el desarrolldor. Introducción a los Patrones de Diseño* 29/10/2011 Available from: http://www.elclubdelprogramador.com/2011/10/29/patrones-de-diseno-introduccion-a-los-patrones-de-diseno/.

34. *Patrones de Asignación de Responsabilidades*. marzo 28, 2012; Available from: .http://www.ecured.cu/index.php/Patrones_de_Asignaci%C3%B3n_de_Responsabilidades.

35. *Lenguaje-Unificado-de-Modelado-UML* 29 Jul 2010 Available from: http://es.scribd.com/doc/35020783/Lenguaje-Unificado-de-Modelado-UML

Bibliografía

1. Franklin Parra, C.M.Q., ed. *Sistema Tutorial Inteligente*. ed. C.d.I.C.y.T. (CICYT). 2010, Escuela Superior Politécnica del Litoral.: Guayaquil, Ecuador.

2. Zulma Cataldi, F.J.l., *Sistemas Tutores Inteligentes Orientados a la Enseñanza para la Comprensión*, in *EDUTEC Revista Electrónica de Tecnología Educativa*. 28 marzo 2009: Facultad Regional Buenos Aires. Argentina.

3. Yharlan Alex Rojas Correa, T.A.M., *MENTOR: Sistema Tutorial Inteligente para el desarrollo de habilidades en la solución de problemas matemáticos*, in *Revista de Investigación Universidad La Salle*. julio-diciembre de 2007. p. pp. 235-246. .

4. Natalia Martínez Sánchez, Z.Z.G.V., Maikel León Espinosa,Denys de Medina Sotolongo,Mateo Lezcano Brito,Gheisa Ferreira Lorenzo,Lydia Rosa Ríos Rodríguez,Yolanda Soler Puig,José M. Linares Álvaro (2009) *VIR 167 Representación y organización del conocimiento en Software educativo*.

5. Natalia Martínez Sánchez, O.R.F.M., *Guía de orientación para la ingeniería del conocimiento en el diseño de sistemas de enseñanza-aprendizaje inteligentes utilizando el razonamiento basado en casos*, in *Revista Generación Digital*. Enero de 2011.

6. *Reglas de Producción*. 30 enero 2011 Available from: http://es.scribd.com/doc/90856631/REGLAS-DE-PRODUCCION.

7. Gil, M.d.P.G., *TEMARIO DE CURSO C291-69 TÓPICOS AVANZADOS: REDES NEURONALES ARTIFICIALES INAOE.* , in *Coordinación de computación, Lab. de "Machine Learning and PatternRecognition". Grupo de investigación en Redes Neuronales Artificiales.* Versión: 17-ene-2012.

8. *Sistemas expertos conexionistas*. jueves 4 de febrero de 2010; Available from: http://menteerrabunda.blogspot.com/2010/01/sistemas-expertos-conexionistas.html.

9. Arias S, F.J.J.B., Jovani A.; Ovalle C., Demetrio A. , *Revista Avances en Sistemas e Informática* in *Redalyc Sistema de Información Científica*. junio 2009, Red de Revistas Científicas de América

Latina, el Caribe, España y Portugal: Universidad Nacional de Colombia. Modelo de planificación instruccional en sistemas tutoriales inteligente. p. pp. 155-164. .

10. Ivar Jacobson, G.B., James Rumbaugh. *El proceso unificado de desarrollo.La guia completa del Proceso escrita por sus creadores*. 20 abril 2010; Available from: http://es.scribd.com/doc/30251931/El-Proceso-Unificado-de-Desarrollo-de-Soft-Jacobson. .

11. Laura Carabalí, J.G., Daniela Loboa. *Programación Extrema*. Jun 13, 2010; Available from: http://www.slideshare.net/urumisama/programacin-extrema-4488942.

12. *INGENIERÍA DEL SOFTWARE:METODOLOGÍAS Y CICLOS DE VIDA. Laboratorio Nacional de Calidad del Software.* primera versión editada en Marzo de 2009; Available from: http://es.scribd.com/doc/62931905/45/Extreme-Programming-XP#outer_page_39.

13. *Ingeniería del software UML - Definición, historia y especificaciones* 01/03/2009.; Available from: http://www.clubdesarrolladores.com/articulos/mostrar/68-uml-definicion-historia-y-especificaciones

14. Mehdi Achour, F.B., Antony Dovgal, Nuno Lopes, Hannes Magnusson, Georg Richter, Damien Seguy, Jakub Vrana. *PHP Manual* 2012; Available from: http://www.php.net/manual/en/intro-whatcando.php

15. *Java*. 12 Ago 2011 Available from: http://codigoprogramacion.com/java/47-introjava.html

16. Pérez, J.E. *Introducción a JavaScript*. 04-05-2009; Available from: http://www.librosweb.es/javascript/index.html

17. Rubira, J. *Google Libraries API, un repositorio de las librerías Javascript más utilizadas*. 14 de julio de 2011; Available from: http://www.genbetadev.com/desarrollo-web/google-libraries-api-un-repositorio-de-las-librerias-javascript-mas-utilizadas

18. *Jquery write less, do more*. 2012; Available from: http://jquery.com/.

19. *Tutorial JavaScript sin dolor usando Prototype (2da parte)*. 2012; Available from: http://www.tutorial-enlace.net/tutorial-JavaScript_sin_dolor_usando_Prototype_(2da_parte)-16619.html

20. Fuentes, A. *La web como plataforma con Dojo*. 11/10/09; Available from: http://www.slideshare.net/afuentes/la-web-como-plataforma-con-dojo-toolkit

21. FabienPotencier, F. *Symfony la guía definitiva*. 25-08-2009 Available from: http://www.librosweb.es/symfony.

22. Alvarez., M.A. *Manual de CodeIgniter (19 capítulos)*. 11 abril 2010 Available from: http://www.desarrolloweb.com/manuales/manual-codeigniter.html.

23. *IDE de Programación* 20 marzo 2012 Available from: http://www.ecured.cu/index.php/IDE_de_Programaci%C3%B3n#Ejemplos_de_IDE_de_programaci.C3.B3n

24. *NetBeans* 2012; Available from: http://netbeans.org/

25. *Eclipse, entorno de desarrollo integrado*. 20 de marzo de 2012; Available from: http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado.

26. wil63r. octubre 5, 2011; Available from: http://aplicaciones.org/eclipse-ide-de-desarrollo-open-source/

27. *Herramienta CASE* 20 de marzo de 2012; Available from: http://www.ecured.cu/index.php/Herramienta_CASE

28. *Rational Rose* 1 Dic 2011; Available from: http://es.scribd.com/doc/74347179/Rational-Rose

29. *Visual Paradigm*. 6 de febrero de 2012; Available from: http://www.ecured.cu/index.php/Visual_Paradigm

30. *Características MySQL* 18 de agosto de 2009; Available from: http://www.anadicsinaloa.com/index.php?option=com_content&view=article&id=207:caracteristicas-mysql&catid=16:anadic-sinaloa&Itemid=33

31. *PostgreSQL Portal en español sobre Postgre* 02/10/2010 Available from: http://www.postgresql.org.es/sobre_postgresql

32. *Apache*. 2012; Available from: http://www.ecured.cu/index.php/Servidor_HTTP_Apache

33. *El Club del Programador. Tutoriales, tips, herramientas y recursos para el desarrolldor. Introducción a los Patrones de Diseño* 29/10/2011 Available from: http://www.elclubdelprogramador.com/2011/10/29/patrones-de-diseno-introduccion-a-los-patrones-de-diseno/.

34. *Patrones de Asignación de Responsabilidades*. marzo 28, 2012; Available from: .http://www.ecured.cu/index.php/Patrones_de_Asignaci%C3%B3n_de_Responsabilidades.

35. *Lenguaje-Unificado-de-Modelado-UML* 29 Jul 2010 Available from: http://es.scribd.com/doc/35020783/Lenguaje-Unificado-de-Modelado-UML

36. M, M.V.C. (6 de junio del 2007) *SISTEMAS TUTORIALES INTELIGENTES*.

37. Francisco J, A.S., Jeovani A, Jiménez B, Demetrio A, Ovalle *Modelo de planificación instruccional en sistemas tutoriales inteligentes.* GIDIA: Grupo de Investigación y Desarrollo en Inteligencia Artificial Escuela de Ingeniería de Sistemas. Universidad Nacional de Colombia Sede Medellín, 30 mayo 2009.

38. Martínez Sánchez Natalia, F.L.G., García Lorenzo María M, García Valdivia Zenaida. , *El Razonamiento Basado en Casos en el ámbito de la Enseñanza/Aprendizaje.* , in *Revista de Informática Educativa y Medios Audiovisuales* 2008.

39. Natalia Martínez Sánchez, M.M.G.L., Zoila Zenaida García Valdivia, *Modelo para diseñar sistemas de enseñanza-aprendizaje inteligentes utilizando el razonamiento basado en casos. Model to design intelligent teaching-learning systems using the case-based reasoning*, in *Revista Avances en Sistemas e Informática*. Diciembre de 2009: Departamento de Ciencias de la Computación. Universidad Central de Las Villas, Cuba.

40. Elena., S.D.y.V., *Sistemas Tutores Inteligentes: profesores particulares en la red www.orsi.es* 5 de octubre de 2010, Equipo de investigación en e-learning de CEDETEL (CENTRO para el DESARROLLO de las TELECOMUNICACIONES de CASTILLA Y LEÓN)

41. Caviedes Pulido Diego, M.G.V.H., García Palencia Omar *Diseño de un sistema tutor inteligente basado en estilos cognitivos.* 2009.

42. *Razonamiento Basado en Casos*. 2012; Available from: http://www.ecured.cu/index.php/Razonamiento_Basado_en_Casos

43. María., F.F.J., *TESIS DE MÁSTER MÁSTER DE INVESTIGACIÓN EN INTELIGENCIA ARTIFICIAL GENERACIÓN DE SISTEMAS BASADOS EN REGLAS MEDIANTE PROGRAMACIÓN GENÉTICA.* Junio de 2008., Universidad Politécnica de Madrid Facultad de Informática.

44. A.López-Carmona., S.F.J.R.V.M., *Sistema basado en reglas difusas para el mapeo de ontologías*. febrero de 2010 Huelva, Departamento de Automática, Universidad de Alcalá, Madrid, España. XV Congreso Español Sobre Tecnologías y Lógica Fuzzy.

45. *Fases de la Programación Extrema.* . 2012; Available from: http://programacionextrema.tripod.com/fases.htm

46. Joskowicz, J. *Reglas y Prácticas en eXtremeProgramming*. febrero 10, 2008.

47. *Programación Extrema*. 13 Jun 2010; Available from: www.slideshare.net/urumisama/programacin-extrema-4488942

48. *EXTREME PROGRAMMING for xp* 28 Mar 2011 Available from: http://es.scribd.com/doc/51712448/1/EXTREME-PROGRAMMING.

49. Ros, I., *Moodle, la plataforma para la enseñanza y organización escolar.* e- Revista de Didáctica 2., 2008.

50. *Metodología XP*. 2012; Available from: http://synergyca.net/index.php?p=1_9_Metodologias.

51. *Ágil (IV): Extreme Programming (XP)*. Mundo Informático. Informática, seguridad y algo más… Publicado el 22 octubre, 2011por svoboda; Available from: http://infow.wordpress.com/Desarrollo

52. Perla Azucena, A.M. *Servidores Web*. 16 Abr 2012; Available from: http://www.monografias.com/trabajos75/servidores-web/servidores-web.shtml

Glosario de Términos

HTML: Acrónimo de *Hypertext Markup Language* (lenguaje de marcas hipertextuales) diseñado principalmente para mostrar información, animaciones en forma de hipertexto, etc. En la actualidad, es el lenguaje que utilizan todos los navegadores para mostrar la información final.

XML: En sus siglas en inglés *Extensible Markup Language* (lenguaje de marcas extensible), no es realmente un lenguaje en particular, sino un protocolo de comunicación entre aplicaciones Web.

Ajax: Técnica de desarrollo web para crear aplicaciones interactivas. Sus iniciales se corresponden con dos lenguajes de programación, JavaScript y XML que interactúan asincrónicamente (*Asynchronous JavaScript And XML*). No es un lenguaje de programación, sino la integración de varias tecnologías para acelerar la comunicación del lado del cliente con el servidor. Se ejecuta en el cliente y mantiene una comunicación asíncrona con el servidor, facilitando la actualización de parte de la información, sin necesidad de recargar nuevamente toda la página.

DOM: Document Object Model es esencialmente una interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, es un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos. Es una interfaz de programación de aplicaciones para acceder, añadir y cambiar dinámicamente contenido estructurado en documentos con lenguajes como JavaScript.

CCS: Las hojas de estilo en cascada (en inglés Cascading Style Sheets) se usan para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML).

HTTP: Es un protocolo que permite la transferencia de archivos (principalmente, en formato HTML) entre un navegador (el cliente) y un servidor web, localizado mediante una cadena de caracteres denominada dirección URL.

URL: localizador de recursos uniforme, más comúnmente denominado (sigla en inglés de *uniform resource locator*), es una secuencia de caracteres, de acuerdo a un formato modélico y estándar, que se usa para nombrar recursos en Internet para su localización o identificación, como por ejemplo documentos textuales, imágenes, vídeos, presentaciones, presentaciones digitales, etc.