Universidad de las Ciencias Informáticas

Facultad 4

Herramienta de selección didáctica para guiar el aprendizaje interactivo en el módulo Ejercicios de la colección El Navegante

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

Autores:

Nadiela Milán Cristo

Ernesto Yordi Plasencia

Tutores:

Ing. Rosalba Carralero Medina

Ing. Hector Luis Reyes Pupo

Ing. Angel Alberto Vazquez Sánchez

Co-Tutor:

Ing. José Ernesto Lara Rodríguez

La Habana, Junio 2012

"Año del 54 Aniversario de la Revolución"

DECLARACIÓN DE AUTORÍA

Herramienta de selección didáctica para guiar el aprendizaje interactivo en el módulo Ejercicios de la colección El Navegante

Declaramos ser autores del presente trabajo de diploma y concedemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo, con carácter exclusivo.

Para que así conste firmamos el presente a los 13 días del mes de Junio del año 2012.

_____ _____

Firma de la tesista
Nadiela Milán Cristo

Firma del tesista
Ernesto Yordi Plasencia

_____ _____

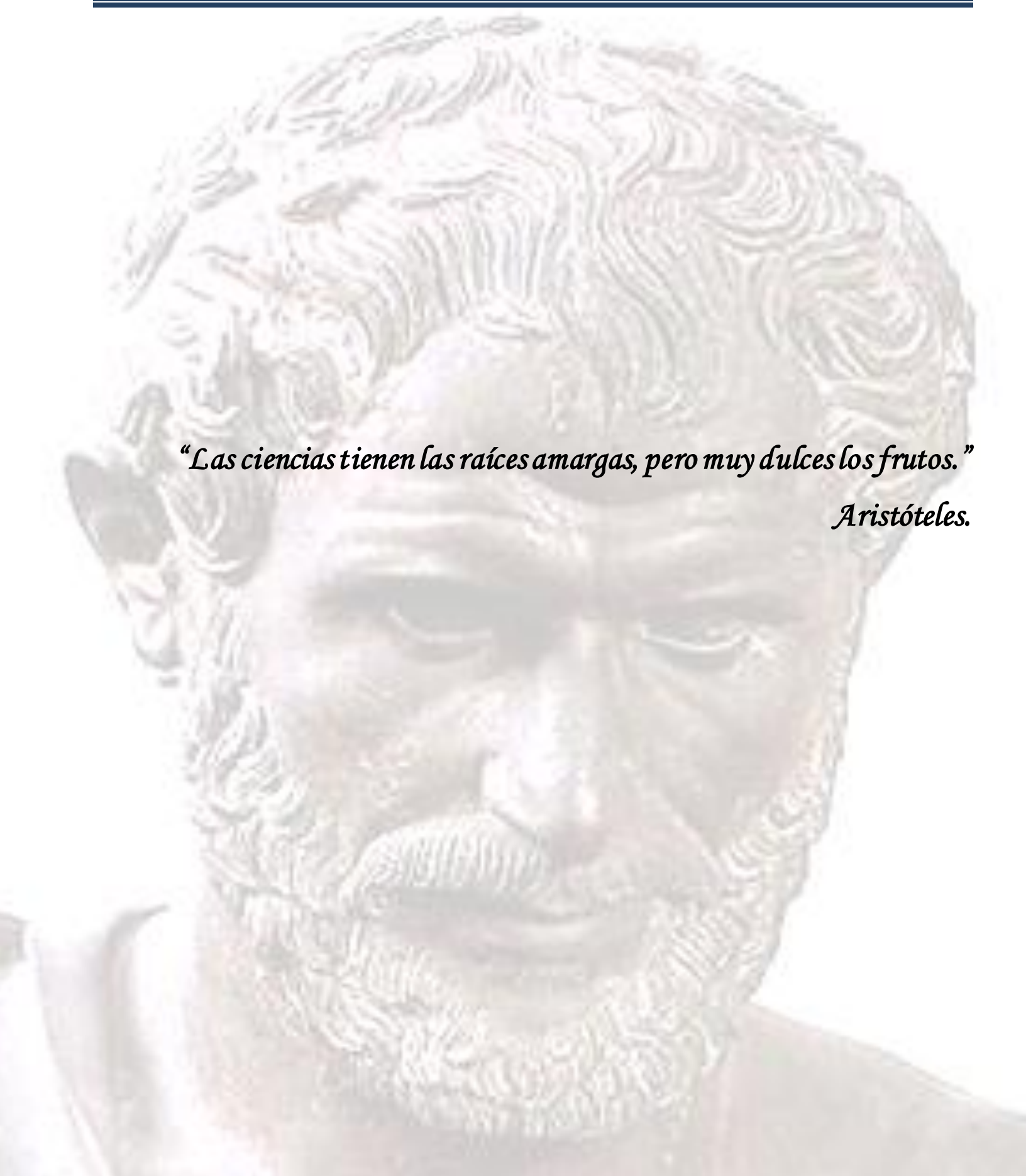
Firma del tutor
Ing. Rosalba Carralero Medina

Firma del tutor
Ing. Angel Alberto Vazquez Sánchez

_____ _____

Firma del tutor
Ing. Hector Luis Reyes Pupo

Firma del co-tutor
Ing. José Ernesto Lara Rodríguez



"Las ciencias tienen las raíces amargas, pero muy dulces los frutos."

Aristóteles.

De Nadiela

A mi abuela Obdulia Nieves Omaña Rubio, a mis padres Antonia Nalis Cristo y Noel Milán Casero y mi hermana Nadiezka Milán Cristo.

A mi novio Ernesto Yordi Plasencia.

De Ernesto

A mis padres Vailde Plasencia y Juan Yordi.

A mis cuatro abuelos.

A mis hermanos Daymara y Abraham.

A mis tíos Javier, Julio, Inesita, Lupe, José, Gloria y a mi primo Gustavo.

A mi prima Mirtha y a mi segunda madre Lula.

A mi novia Nadiela Milán Cristo.

Resumen

El desarrollo del software educativo en Cuba se ha incrementado debido a los beneficios que brindan para la educación. Diversas son las colecciones de este tipo de software desarrolladas por el Ministerio de la Educación, siendo El Navegante, colección destinada a las escuelas secundarias, un ejemplo de ellas. Los sistemas de recomendación son aplicaciones encargadas de realizar sugerencias a los usuarios basándose en la preferencia de estos por un grupo de elementos con características específicas. El presente trabajo tiene como objetivo desarrollar un recomendador de cuestionarios interactivos para los estudiantes que utilizan la colección El Navegante, mejorando así la consolidación del conocimiento. Para guiar el desarrollo se utilizó Rational Unified Process (RUP) como metodología de desarrollo de software, PHP y JavaScript como lenguajes de programación y Symfony como framework de desarrollo. Se realizó el análisis, diseño, implementación y prueba; obteniéndose en cada caso los artefactos fundamentales que propone la metodología para cada etapa de trabajo. Este sistema permite a los estudiantes atender aspectos en los que presentan mayores dificultades y así poder erradicar las mismas. Promover el uso adecuado y consciente de las Tecnologías de la Información y las Comunicaciones y disminuir el tiempo que emplean los docentes en la identificación de las dificultades de sus estudiantes, facilitando la atención diferenciada hacia cada uno de ellos.

Palabras clave: sistema de recomendación, software educativo, colección El Navegante.

Índice

ÍNDICE

Índice de figuras

Índice de tablas

Introducción

En la actualidad los sistemas educativos de todo el mundo se enfrentan al reto de utilizar las Tecnologías de la Información y las Comunicaciones (TIC), con el objetivo de proveer a los estudiantes herramientas y conocimientos necesarios en el siglo XXI. La posibilidad de interacción que ofrecen, pasando de una actitud pasiva por parte de los educandos a una actividad constante, la búsqueda y replanteamiento continuo de contenidos y procedimientos, el aumento de la implicación del alumnado en sus tareas y desarrollo de iniciativa de los mismos, son las mejoras que brindan las TIC al proceso de enseñanza-aprendizaje, obligando al escolar a tomar pequeñas decisiones, a escoger, seleccionar y a filtrar información. (1)

Un auténtico ejemplo del desarrollo de las TIC en la educación cubana es el desarrollo por el Ministerio de Educación de Cuba, de tres colecciones educativas para las enseñanzas primaria, secundaria básica y preuniversitario; colección Multisaber, colección El Navegante y colección Futuro respectivamente, utilizando para ello la tecnología multimedia. Dichas colecciones apoyan el proceso de aprendizaje de los niños y adolescentes, debido a que tributan a la formación de una cultura general integral con un enfoque curricular y multidisciplinario, por su relación con los contenidos de los programas de cada asignatura del currículo de estudio de los distintos niveles de enseñanza. (2)

La Universidad de las Ciencias Informáticas (UCI), institución surgida al calor de la batalla de ideas, entre sus misiones asume la de impulsar el desarrollo de la industria del software en Cuba. Con el objetivo de organizar la labor productiva en la misma fueron creados varios centros de desarrollo, entre los que se encuentra el Centro de Tecnologías para la Formación (FORTES). Este tiene como misión desarrollar tecnologías que permitan ofrecer servicios y productos, para brindar soluciones de formación aplicando las TIC, a todo tipo de instituciones con diferentes modelos de formación y condiciones tecnológicas.

FORTES cuenta con el proyecto Multisaber-Navegante, el cual consiste en el desarrollo de una nueva versión utilizando software libre, de catorce de las multimedia de la colección Multisaber y diez de la colección El Navegante, ya que las existentes están programadas con herramientas propietarias y utilizan recursos mediáticos no libres.

Se debe garantizar que todo el contenido incluido en los productos, dígase música, videos, imágenes, textos y diseños estén liberados bajo licencias que permitan su utilización, para ello se dispondrá de asesoría y revisión jurídica. La nueva implementación de los productos con tecnología multimedia se hace sobre plataforma web y se lleva a cabo su reconstrucción casi completa, solo aprovechándose una parte de los contenidos de los antiguos productos. Esto a su vez permite un proceso de estandarización en cuanto a diseño, contenidos y programación. (2)

La colección El Navegante es un conjunto de software que tributan a la formación de una cultura general integral en la educación secundaria. Los diez productos que la conforman abordan las asignaturas de Español, Matemática, Biología, Geografía, Física, Química, Educación Artística, Inglés, Educación Laboral e Historia. Está compuesta por siete módulos: Temas, Ejercicios, Juegos, Mediateca, Resultados, Maestro y General, los mismos permiten la interacción del usuario con el sistema, la adquisición de contenidos, así como la visualización de resultados.

El módulo Ejercicios es el encargado de controlar el aprendizaje a través de cuestionarios interactivos, los cuales pueden ser definidos por las catorce tipologías: selección simple, selección múltiple, completamiento por escritura, completamiento por desplazamiento, enlazar columnas, ordenar textos, ordenar pasos, selección de textos, verdadero o falso, rayo numérico, acentuar palabras, formar conjuntos, seleccionar palabras y dividir en sílabas. Los cuestionarios se evalúan teniendo en cuenta las categorías: correcta, parcialmente correcta o regular, incorrecta o mal.

La resolución de estos cuestionarios interactivos dentro de la colección El Navegante se realiza escogiéndolos a partir de la orientación del profesor o por voluntad propia del estudiante. El profesor que emplea la colección debe dedicar mucho tiempo y le resulta engorroso tener que identificar para cada estudiante los ejercicios a resolver, teniendo en cuenta las características y dificultades de cada uno de los alumnos para lograr una atención diferenciada y de esta manera la consolidación de los conocimientos y la corrección de los errores cometidos. Por otra parte si el estudiante desea realizar un autoestudio extra a las orientaciones realizadas por el profesor y no es consciente de sus dificultades, puede darse el caso de que invierta tiempo resolviendo cuestionarios que no lo ayuden a vencer las habilidades en las cuales presenta deficiencias.

Dada la situación anteriormente descrita, se plantea el siguiente **problema a resolver:** ¿Cómo favorecer el aprendizaje interactivo en los estudiantes de la enseñanza secundaria básica a través de la colección El Navegante?

Se define como **objeto de estudio:** Proceso de enseñanza-aprendizaje asistido por las Tecnologías de la Información y las Comunicaciones.

Enmarcándose en el **campo de acción:** Sistemas recomendadores de cuestionarios interactivos para la enseñanza-aprendizaje.

Teniendo como **objetivo general:** Desarrollar un sistema recomendador de cuestionarios interactivos para los estudiantes de la enseñanza secundaria básica a través de la colección El Navegante.

Como **objetivos específicos** se definen los siguientes:

1. Elaborar el estado del arte de los sistemas de recomendación de cuestionarios interactivos para la enseñanza-aprendizaje.
2. Diseñar el algoritmo a utilizar en el sistema de recomendación.
3. Realizar el análisis y diseño del sistema de recomendación.
4. Implementar el sistema recomendador e integrarlo con el módulo Ejercicios de la colección El Navegante.
5. Realizar pruebas internas para garantizar la funcionalidad del sistema.

Se define como **idea a defender:** Un sistema recomendador de cuestionarios interactivos permite la atención diferenciada de los estudiantes de la enseñanza secundaria básica a través de la colección El Navegante.

Con el desarrollo de esta herramienta se espera que los estudiantes y docentes cuenten con un mecanismo que les permita dirigir su atención hacia aquellos aspectos en los que presentan mayores dificultades y de esta manera erradicar las mismas. Siendo así, una herramienta pedagógica integral de

apoyo y soporte al proceso de enseñanza-aprendizaje. Además se promoverán las actitudes críticas, creativas e investigativas de los estudiantes y docentes, contribuyendo a la formación de estos en el uso adecuado y consciente de las TIC.

Para la realización de la presente investigación se utilizaron métodos teóricos, los cuales permitieron abordar la realidad, estudiar la naturaleza, la sociedad y el pensamiento del objeto, con el propósito de descubrir su esencia y sus relaciones, así como comprenderlo en su desarrollo, historia y lógica.

Los métodos teóricos empleados fueron:

1. Analítico-sintético: Se evidencia cuando se realiza un análisis de toda la teoría y documentación, que permiten la extracción de los elementos fundamentales relacionados con las herramientas recomendadoras que contribuyen al aprendizaje interactivo.

2. Histórico-lógico: Utilizado al tener en cuenta la caracterización de la evolución histórica de los sistemas de recomendación como herramientas bases para concebir el sistema actual.

3. Modelación: Se emplea como forma de representación de todos los datos esenciales relacionados con el campo de acción.

El uso de métodos de investigación empíricos permite describir y explicar las características fenomenológicas del objeto y representan un nivel de investigación cuyo contenido procede de la experiencia y es sometido a cierta elaboración racional.

El método empírico empleado fue:

1. Entrevista: Este permitió determinar los elementos fundamentales relacionados con la recomendación de ejercicios en la enseñanza secundaria.

Las **tareas de investigación** definidas para dar cumplimientos a los objetivos específicos fueron las siguientes:

1. Evaluar las tendencias actuales en el mundo de los sistemas de recomendación.

2. Realizar trabajo de grupo y entrevistas con especialistas que ayuden a definir los elementos a tener en cuenta para la recomendación de ejercicios en la enseñanza secundaria.

3. Analizar los componentes que conforman el módulo Ejercicios de la colección El Navegante en su versión multiplataforma.

4. Especificar conceptos que tributen a la comprensión de los elementos que conforman un recomendador de ejercicios.

5. Definir los requerimientos funcionales y no funcionales del sistema de recomendación.

6. Analizar la arquitectura definida por el proyecto Multisaber-Navegante para el desarrollo de la colección El Navegante en su versión multiplataforma.

7. Realizar el análisis y diseño del sistema de recomendación.

8. Implementar el sistema de recomendación.

9. Integrar el sistema de recomendación con el módulo Ejercicios de la colección El Navegante en su versión multiplataforma.

10. Realizar pruebas internas al sistema de recomendación para validar su funcionalidad.

El documento está organizado en cuatro capítulos, de la siguiente manera:

Capítulo 1: Fundamentación teórica. Se hace un análisis de los principales conceptos relacionados con el objeto de estudio. Además, se realiza un análisis del estado del arte de las herramientas recomendadoras orientadas al aprendizaje interactivo y de los sistemas de recomendación en sentido general. También de las distintas técnicas, tecnologías y metodologías a utilizar.

Capítulo 2: Concepción y características del sistema. Se describe la propuesta de solución del sistema y el Modelo de dominio donde se capturan los objetos importantes del entorno en el cual será empleado el sistema. Se identifican además los requerimientos funcionales y los no funcionales, se elabora el modelo de casos de uso y las descripciones de cada uno de ellos.

Capítulo 3: Análisis y diseño. Se describe la solución que se propone a partir de los diagramas de clases del análisis asociado a las funcionalidades del sistema y se representan los diagramas de clases del diseño que reflejan una vista interna del sistema.

Capítulo 4: Implementación y pruebas. Se realiza el diagrama de componentes y de despliegue, obteniéndose una descripción de la implementación del sistema. También se describen los casos de pruebas por los cuales los desarrolladores verifican el producto, además se selecciona el método de prueba y la técnica de prueba que se va a utilizar.

Capítulo 1 Fundamentación teórica

Introducción

A partir de las necesidades que produce el desarrollo del proceso de enseñanza-aprendizaje, el cual requiere de vías, métodos y estrategias acorde a las características propias de cada estudiante, tipo de contenido a impartir, motivaciones, escenario de la actividad docente, estilos de aprendizajes, niveles de desempeño y niveles de asimilación, entre otros factores, el desarrollo de materiales educativos requiere de una diversidad tal que ofrezcan variadas potencialidades para adecuarse a cada una de las características descritas anteriormente. En este capítulo se realiza un análisis de los principales conceptos relacionados con el objeto de estudio. Además de un análisis del estado del arte de las herramientas recomendadoras orientadas al aprendizaje interactivo y de los sistemas de recomendación en sentido general. También son descritas las herramientas, tecnologías y metodologías a utilizar en la solución propuesta.

Generalidades acerca del aprendizaje interactivo

El aprendizaje se puede definir de muchas maneras:

- Proceso de adquisición de conocimientos, habilidades, valores y actitudes, posibilitado mediante el estudio, la enseñanza o la experiencia. Este proceso puede ser analizado desde diversas perspectivas, por lo que existen distintas teorías del aprendizaje. (3)

- Proceso por medio del cual la persona se apropia del conocimiento, en sus distintas dimensiones: conceptos, procedimientos, actitudes y valores. (4)

- Proceso a través del cual se adquieren habilidades, destrezas, conocimientos como resultado de la experiencia, la instrucción o la observación. (5)

- Es una actividad dinámica, constructiva, acumulativa, intencional y sujeta al contexto, el estudiante es responsable de su proceso de aprendizaje y los materiales didácticos han de proporcionarle oportunidades de aprender y ser adecuados a sus posibilidades de aprendizaje. (6)

De forma general el aprendizaje no sería más que el proceso que permite obtener conocimientos acerca de algún tema, adquiridos a través del estudio, la enseñanza o de manera empírica para ser capaces de adaptarse a diferentes entornos, poder dar respuesta a transformaciones y las acciones que estas transformaciones propicien.

La interacción en el aprendizaje es importante para los estudiantes ya que diversos autores defienden que las interacciones constituyen un elemento que favorece el desarrollo cognitivo, la adquisición de conocimientos y habilidades y en general obtener buenos resultados escolares. (6)

El aprendizaje interactivo es la manera en que se adquiere el conocimiento a partir de la interacción con herramientas educativas o hiperentornos de enseñanza-aprendizaje, que presentan contenidos, juegos didácticos, medias y cuestionarios interactivos, para medir conocimientos a través de medios informáticos.

Los cuestionarios interactivos son un conjunto de preguntas redactadas de forma coherente, organizadas, secuenciadas y estructuradas de acuerdo con una determinada planificación.

La recomendación de cuestionarios interactivos en la enseñanza secundaria se realiza a partir de la orientación de ejercicios definidos por el profesor a los estudiantes. Esta recomendación se efectúa llevando a cabo un diagnóstico inicial. El mismo permite determinar el nivel de desempeño en el que se encuentra el estudiante.

Para medir los niveles de desempeño cognitivo en cada una de las asignaturas, se han considerado tres niveles de desempeño:

- Nivel 1: Mide la capacidad del alumno para realizar las operaciones de carácter instrumental, básicas de una asignatura dada. Para ello deberá reconocer, identificar, describir e interpretar los conceptos y propiedades esenciales en los que esa materia se sustenta. (7)
- Nivel 2: Evalúa la capacidad del alumno para establecer relaciones conceptuales. Además de establecer relaciones conceptuales, donde aparte de reconocer, identificar, describir e interpretar los conceptos, deberá aplicarlos a una situación planteada y reflexionar acerca de sus relaciones internas. (7)

- Nivel 3: Determina la capacidad del alumno para resolver problemas, para lo que deberá reconocer y contextualizar la situación problémica, identificar componentes e interrelaciones, establecer las estrategias de solución y fundamentar o justificar lo realizado. (7)

Luego de identificar en qué nivel de desempeño se encuentra el estudiante se procede a seleccionar los ejercicios para vencer las habilidades definidas en cada nivel de desempeño.

Las habilidades son formaciones psicológicas mediante las cuales el sujeto manifiesta en forma concreta la dinámica de la actividad con el objetivo de elaborar, transformar, crear objetos, resolver situaciones y problemas, actuar sobre sí mismo: autorregularse. Estos modos de actuación se caracterizan por ser útiles en diferentes contextos, ya sea aplicando conocimientos y acciones ya conocidas, experimentando, extrapolando o elaborando nuevas combinaciones sobre la base de viejos estereotipos y experiencias. (8)

Estas habilidades definidas en cada nivel de desempeño deben ser vencidas por los estudiantes a partir de un sistema de acciones variadas y desplegadas, elaboradas por el profesor con el propósito de vencer los objetivos propuestos.

Para la colección El Navegante se han definido varias tipologías de cuestionarios interactivos, las cuales se mencionan a continuación:

- Selección simple
- Selección múltiple
- Seleccionar textos
- Completar por escritura
- Completar por desplazamiento
- Ordenar pasos
- Ordenar textos
- Enlazar columna
- Rayo numérico
- Verdadero o falso
- Seleccionar palabras
- Formar conjunto

- Dividir en sílabas
- Acentuar palabras

Cada cuestionario de la colección tiene asociado una tipología y los que se aplican en el proceso de recomendación en la enseñanza secundaria tienen definido habilidades que debe vencer el estudiante para avanzar al próximo nivel de desempeño. De acuerdo a lo anterior se concluye que existe una vinculación entre las tipologías y las habilidades por lo cual se hace necesario determinar la relación existente entre ambas. Esta vinculación fue acordada a partir de entrevistas y grupos de trabajo, realizados con especialistas para establecer los elementos fundamentales relacionados con la recomendación de ejercicios en la enseñanza secundaria y es punto de partida para definir el diseño algorítmico del sistema recomendador.

A continuación se muestra la relación existente entre las tipologías definidas para los cuestionarios interactivos y las habilidades establecidas para cada nivel de desempeño:

Tipologías \ Habilidades	Observar	Analizar	Definir	Identificar	Describir	Interpretar	Reflexionar	Caracterizar	Comparar	Generalizar	Fundamentar	Clasificar
Selección Simple	x	x	x	x	x	x	x	x		x	x	x
Selección Múltiple	x	x	x	x	x	x	x	x		x	x	x
Seleccionar Textos	x	x	x	x	x	x	x	x		x	x	x
Completar por Escritura	x	x	x	x		x	x	x		x		x
Completar por Desplazamiento	x	x	x	x	x	x	x	x		x		x
Odenar Pasos	x	x		x		x			x			
Ordenar Textos	x	x		x		x			x			
Enlazar Columna	x	x	x	x		x	x	x	x	x		
Rayo Numérico	x	x		x		x						x
Verdadero o Falso	x	x	x	x		x	x	x		x		
Seleccionar Palabras	x	x	x	x		x		x				
Formar Conjunto	x	x	x	x		x	x	x		x		x
Dividir en Sílabas	x	x	x	x								x
Acentuar Palabras	x	x	x	x								x

Leyenda Nivel 1 Nivel 2 Nivel 3

Fig. 1 Relación entre tipologías y habilidades

Generalidades acerca de los sistemas de recomendación

Los sistemas recomendadores son herramientas de software desarrolladas para ayudar a los usuarios en el proceso de toma de decisiones con respecto a un tema específico. En la actualidad, el uso de estos sistemas se ha extendido a una gran variedad de aplicaciones.

El término "ítem", se utiliza para denominar el objeto que el sistema debe recomendar, por ejemplo, canciones, películas o noticias. Por lo general un sistema recomendador se especializa en un ítem específico, por lo cual elementos como el diseño, la interfaz gráfica y el algoritmo usado para generar las recomendaciones, son personalizados para obtener sugerencias efectivas para dicho ítem.

De manera general, se puede decir que los datos que utilizan los sistemas recomendadores pueden separarse en tres grupos:

Ítems: Como se mencionó anteriormente son los elementos que son recomendados. El valor de un ítem puede ser positivo si resulta útil para el usuario, así como puede ser negativo si el usuario considera que fue una mala decisión seleccionarlo. Los ítems tienen además, características que pueden indicar cuan necesario resultan para un usuario determinado.

Usuarios: Un usuario puede tener diversidad de metas o preferencias cuando accede a un software. Para poder ofrecer las recomendaciones adecuadas el sistema debe explorar toda la información disponible acerca de dicho usuario, ya sea la que inicialmente se haya solicitado o la que se va recopilando a lo largo de la interacción de la persona con el sistema.

Transacciones: Genéricamente se le denomina transacciones a la interacción de los usuarios con el sistema de recomendación. De esta manera se puede almacenar información acerca de las preferencias de los usuarios sobre los ítems. Una de las formas más frecuentes y populares de ver las transacciones es a través de los "ratings". Este término se refiere a la evaluación que realiza un individuo sobre un ítem que es sometido a su consideración, de esta forma se puede almacenar el criterio que tiene el usuario sobre un ítem determinado.

El problema de la recomendación

Considerando lo anterior, se puede introducir el problema de la recomendación: (9)

Sea *C* un conjunto de usuarios, y *S* el conjunto de todos los ítems posibles a ser recomendados, tales como películas, noticias, libros o ejercicios, pudiendo ser bien grande la cardinalidad de ambos conjuntos. Sea además *u* una función que mide la utilidad del ítem *s* al usuario *c*, o sea $u: C \times S \rightarrow R$, donde *R* es un orden total (enteros no negativos o números reales en un determinado rango). Con estos elementos, se desea, para cada usuario $c \in C$, aquel ítem $s \in S$ que maximice la utilidad al usuario. Más formalmente:

$$\forall c \in C, \quad s'_c = argmax_{s \in S} u(c, s)$$

Clasificación de los sistemas de recomendación

Los sistemas de recomendación pueden recibir diferentes clasificaciones. Una de las formas más utilizadas es separarlos atendiendo a la metodología que utilizan para la construcción del perfil de un usuario determinado. A continuación se expone esta forma de clasificar los sistemas recomendadores.

Recomendadores basados en contenido

El elemento clave de este tipo de recomendador es determinar cuán relacionado está un objeto con un determinado usuario, para lo cual utiliza la medida de similitud. De esta manera puede encontrar objetos que se adapten lo mejor posible a las preferencias de los usuarios. Para calcular esta relación el sistema se basa exclusivamente en la descripción del elemento y en la información recogida en el perfil del usuario que almacena los intereses de este.

Dicho perfil puede ser rellenado de manera explícita, preguntando directamente al usuario por sus preferencias o su criterio acerca de un elemento determinado. La información puede ser recogida además de manera indirecta, considerando elementos que revelen las preferencias del usuario, entre estos elementos se pueden encontrar: sitios visitados, tipos de archivos descargados, tiempo de permanencia en un sitio, categoría de noticias leídas y cualquier otra información que resulte útil.

Este tipo de sistemas puede ser ampliamente utilizado. Su uso puede extenderse a sitios noticiosos, librerías, descargas de música y video, restaurantes y ventas en general. Los elementos comunes para todos estos sistemas resultan: la necesidad de contar con un perfil de usuario que almacene sus preferencias, una descripción del elemento que posiblemente sea recomendado y un sistema que determine lo cerca que está dicho elemento del usuario, con el fin de poder recomendárselo.

Una ventaja importante de este tipo de sistema recomendador es que se pueden realizar recomendaciones sin la necesidad de contar con un historial previo, por lo cual se pueden efectuar predicciones independientemente del historial del usuario. Como principal desventaja se encuentra el hecho de que las recomendaciones en su mayoría tienen un alto grado de similitud, ya que se basan en los mismos datos.

Sistemas de soporte a la recomendación

Estos son sistemas que se dedican a brindar soporte a la actividad de compartir recomendaciones (10). Los usuarios están fundamentalmente agrupados en dos roles: el productor de recomendaciones, que se encarga, como su nombre lo indica, de construir las recomendaciones que posteriormente serán buscadas por un usuario con el rol de consumidor de recomendaciones.

Estos sistemas logran mejores resultados cuando hay suficientes usuarios dedicados al rol de productor de recomendaciones, o sea, a la recopilación de informaciones con el fin de ofrecer las mejores sugerencias, teniendo en cuenta, además, la retroalimentación que debe existir entre el productor y el consumidor. A través de esta comunicación se puede determinar el nivel de conformidad del usuario que recibe las recomendaciones y por lo tanto, la eficacia del sistema.

Sistemas basados en filtrado colaborativo

El filtrado colaborativo tiene como base el agrupamiento de personas que muestren intereses similares. Para realizar estas asociaciones el usuario debe expresar sus preferencias a través de la evaluación de elementos que se le presentan, sirviendo esto para crear un perfil aproximado de este (10). Una vez que se recopilan los datos sobre las preferencias del usuario y se conforma su perfil, el sistema busca entre los restantes usuarios los que posean características similares, formando así grupos de personas con intereses comunes. Estos grupos de usuarios pueden denominarse “usuarios más cercanos”. De esta

manera, para crear una recomendación efectiva, el sistema puede obtener la lista de los elementos mejor evaluados por el grupo de usuarios y que aún no han sido evaluados por el nuevo individuo. (10)

Una de las condiciones necesarias para que este tipo de recomendador funcione de manera efectiva es la necesidad de que la cantidad de personas que utilizan el sistema sea suficiente para que cualquier nuevo individuo pueda encontrar grupos de usuarios con características similares. Otros elementos importantes que debe poseer el sistema son: contar con una forma eficaz de representar los intereses de los usuarios, de modo que la comparación entre estos no resulte tan compleja y disponer de algoritmos que sean capaces de asociar de manera correcta a las personas dentro del sistema.

En este tipo de sistema coinciden en una misma persona los roles de constructor y consumidor de recomendaciones. Esto se debe a que la representación de un perfil será más exacta en tanto más recomendaciones o evaluaciones realice dicho usuario, por lo tanto se incrementa también la exactitud de las recomendaciones que el sistema puede brindar al usuario.

Sistemas de recomendación basados en datos demográficos

Este tipo de sistema se basa en el perfil demográfico del usuario. La idea es que se pueden construir diferentes recomendaciones para grupos demográficos variados. Muchos sitios web adoptan personalizaciones simples y efectivas basándose en esta información. Por ejemplo, algunos usuarios pueden ser redireccionados a un sitio web en particular atendiendo al país donde residen o al idioma que hablan. Además las recomendaciones que realiza el sistema pueden ser personalizadas de acuerdo con la edad del usuario (11).

Sistemas de recomendación híbridos

Como se ha podido observar, las técnicas antes descritas, presentan algunas desventajas cuando trabajan por separado. Algunas de las situaciones más preocupantes pueden ser la sobre-especialización en el caso de los sistemas basados en contenido y la necesidad de una gran cantidad de usuarios cuando se trata de un sistema basado en filtrado colaborativo.

Una opción que se ha encontrado para mitigar las desventajas que presentan por separado algunos de los sistemas anteriormente expuestos, es combinar la recomendación basada en contenido con el filtrado

colaborativo. De esta manera se pueden usar las ventajas que ofrece la primera para atenuar las desventajas de la segunda y viceversa.

Después de realizar un análisis de algunos de los tipos de sistemas de recomendación teniendo en cuenta las características, ventajas y desventajas de los mismos, se puede concluir que los elementos comunes para estos sistemas son: la necesidad de contar con un perfil de usuario que almacene sus preferencias, una descripción del elemento que posiblemente sea recomendado y un sistema que determine lo cerca que está dicho elemento del usuario, con el fin de poder recomendárselo.

Se decide utilizar un sistema de recomendación basado en contenido ya que su principal ventaja radica en realizar recomendaciones sin la necesidad de contar con un historial previo, por lo cual se pueden efectuar predicciones independientemente del historial del usuario. No se necesita de gran cantidad de personas que utilicen el sistema para determinar a qué grupo de usuarios con características similares debe pertenecer un nuevo individuo, cuestión en la que se basa el filtrado colaborativo. Además la recomendación no está definida para que un usuario realice recomendaciones a otro, ni teniendo en cuenta la edad del mismo, particularidades de los sistemas de soporte a la recomendación y los basados en datos demográficos, respectivamente.

Análisis de soluciones similares existentes

En función de facilitar la concepción del sistema de recomendación se realiza un estudio de soluciones similares existentes para determinar elementos que presentan dichas herramientas y que pueden ser útiles en el desarrollo de la solución propuesta. Los sistemas analizados son los siguientes:

Sistema inteligente para asistir la búsqueda personalizada de objetos de aprendizaje

Sistema recomendador de objetos de aprendizaje que ayuda a un usuario a encontrar los recursos educativos que le sean más apropiados de acuerdo a sus necesidades y preferencias. La búsqueda se realiza en diferentes repositorios de objetos de aprendizaje, donde cada objeto tiene metadatos descriptivos. Se utilizan estos metadatos para recuperar aquellos objetos que satisfagan no sólo el tema de la consulta, sino también el perfil de usuario, teniendo en cuenta sus características y preferencias. El sistema tiene una arquitectura multiagente que incluye varios tipos de agentes con diferentes funcionalidades. En particular, en este sistema se modela el Agente Recomendador (Agente-R), como

Agente BDI graduado, que se encarga de realizar una recuperación flexible y presentar una lista ordenada con los mejores recursos de acuerdo con el perfil de usuario. (12)

Next2Solve

Creado por Igor Navernioukk, estudiante de doctorado de la Universidad de Toronto, es un sistema de recomendación de ejercicios en jurados de la ACM, que se basa en el ACM Problem Grading de Urbaniak (sistema de recomendación que consiste en proporcionar un listado de problemas a resolver, el cual puede ser ordenado por diferentes criterios entre los que se pueden mencionar la cantidad de soluciones aceptadas para cada ejercicio, el porciento de soluciones correctas con respecto al total de soluciones y un score de simplicidad que se calcula tomando como base los dos anteriores criterios) enfocándose además en los usuarios ya que tiene en cuenta un identificador del mismo y en función de este muestra una sublista de la lista generada por la aplicación de Urbaniak, en la que sólo aparecen los problemas a resolver por el usuario entrado excluyéndose aquellos que él ya ha resuelto. (10)

Citesser

Es una biblioteca digital de literatura científica y motor de búsqueda que se centra principalmente en la literatura en ciencias de la computación de información electrónica. Tiene como objetivo mejorar la difusión de la literatura científica y proporcionar mejoras en la funcionalidad, facilidad de uso, integridad, eficiencia y puntualidad en el acceso a los conocimientos científicos y académicos. Sistema basado en contenido utilizando para ello la información de la palabra y el análisis de las citas comunes en los periódicos. (11)

Sistema de recomendación para jurados online de programación

Creado por Raciel Yera Toledo, estudiante de la Universidad de Ciencias Informática en Cuba, es un sistema de recomendación de ejercicios en jurados online. La filosofía de la recomendación está basada en el filtrado colaborativo aplicada a los jurados online de programación, que consiste en sugerir al usuario aquellos problemas que a pesar de haber sido resueltos por usuarios semejantes a él, aún no han sido resueltos por este. (10)

Tras analizar las anteriores aplicaciones se puede decir que ninguno de estos sistemas fueron concebidos para recomendar ejercicios con las particularidades que presentan los de la colección El Navegante, sin

embargo, su estudio fue provechoso dado que se identificaron características y funcionalidades comunes, por ejemplo, la creación del perfil del usuario y que pueden ser útiles para el desarrollo del presente recomendador.

Metodología de desarrollo

El desarrollo de software no es una tarea fácil. Desde el momento que se piensa en la realización de una aplicación, surge la necesidad por parte de los desarrolladores de coordinar las actividades que realizarán durante las múltiples fases del desarrollo. De esta manera, se hace necesario el empleo de un método común, un proceso que proporcione una guía para ordenar las actividades de un equipo, dirija las tareas de cada desarrollador por separado y del equipo como un todo, especifique los artefactos que deben desarrollarse y actividades que se realizan en el proyecto, además de ofrecer criterios para el control y la medición de los productos.

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software (13); están divididas en dos grupos: metodologías tradicionales o robustas y metodologías ágiles o ligeras. La diferencia que existe entre ambos grupos es que las metodologías tradicionales son procesos muchos más controlados, con políticas y normas tratando de buscar la calidad del software a través del orden y la documentación, sin embargo, las ágiles son procesos menos controlados y con pocos principios, que tratan de buscar la calidad del software a partir de la comunicación inmediata y directa entre las personas que intervienen en el proceso de desarrollo.

La herramienta a desarrollarse forma parte del proyecto Multisaber-Navegante; la metodología de desarrollo definida es Rational Unified Process (RUP). En aras de mantener la estructura definida en el momento de generar la documentación se decide utilizar la misma metodología, ya que el uso de otra podría traer algunas inconveniencias en el cronograma definido por el proyecto.

RUP es una metodología tradicional o robusta que está basada en componentes. El sistema de software en construcción está formado por componentes interconectados a través de interfaces bien definidas. Utiliza el Lenguaje Unificado de Modelado (UML) para preparar todos los esquemas en un sistema de software. Es iterativo e incremental, centrado en la arquitectura y guiado por casos de uso. Es un proceso

genérico, un marco de trabajo de proceso. Cada organización que lo utilice, lo especializará para ajustarlo a su situación, por ejemplo, dado su tipo de aplicación, plataforma entre otros criterios. (14)

Algunas ventajas de RUP:

- RUP es considerada universalmente una de las metodologías más difundidas y define claramente actividades realizadas por roles, generando a su vez artefactos que sustenten el proceso de construcción de software.
- Constituye una metodología adaptable al proyecto, utilizada para el análisis, implementación y documentación de sistemas a través del UML (Unified Modeling Language), que implementa el paradigma orientado a objetos.
- Esta metodología tiene como una de sus características principales el desarrollo iterativo e incremental que posee las ventajas siguientes:
    - Se reduce el coste del riesgo a los costes de un solo incremento ya que si los desarrolladores tienen que repetir una iteración, sólo pierden el esfuerzo empleado por la organización, no el valor del producto entero.
    - Las necesidades de los usuarios y las exigencias que no pueden definirse completamente al principio son refinados en iteraciones sucesivas, de manera que se hace más fácil adaptarse a los requisitos cambiantes.
    - Las iteraciones controladas aceleran el ritmo de desarrollo del producto dado el hecho de que los desarrolladores trabajan de forma más eficiente para obtener resultados claros a corto plazo, en lugar de tener un calendario que se prolonga una eternidad.
- En RUP no necesariamente se debe mantener un contacto frecuente con los clientes.
- RUP no recomienda que se lleven a cabo estrictamente todas las actividades y artefactos que se describen, sino por el contrario, se recomienda que en dependencia de las características del proyecto y de la organización se seleccionen los artefactos, actividades y roles que van a ser utilizados.

Herramientas y tecnologías a utilizar

Lenguajes de programación

Actualmente existen diversos lenguajes de programación que permiten el desarrollo de aplicaciones con tecnologías web. Estos han ido surgiendo por las tendencias y necesidades de las plataformas. Los mismos pueden ordenarse en dos grupos en correspondencia con dos estrategias complementarias: las funciones que se ejecutan del lado del cliente y las que se ejecutan del lado del servidor.

- Un lenguaje del lado cliente basa su procesamiento en el cliente web, es decir, que se ejecuta en el navegador del usuario.
- Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de la red al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución del lado del servidor. (15)

Lenguajes de programación y tecnologías del lado del cliente

XHTML (Lenguaje de Marcado de Hipertexto Extensible) es una versión más estricta y limpia de HTML, que nace precisamente con el objetivo de remplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML. XHTML extiende HTML 4.0 combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos.

XHTML, al estar orientado al uso de un etiquetado correcto, exige una serie de requisitos básicos a cumplir en lo que a código se refiere. Entre estos requisitos básicos se puede mencionar una estructuración coherente dentro del documento donde se incluirían elementos correctamente anidados, etiquetas en minúsculas, elementos cerrados correctamente y atributos de valores entre comillas. (16)

En el desarrollo de la solución, el uso correcto del lenguaje proporciona ventajas. Su sintaxis estricta permite un mejor rendimiento en el consumo del procesador, aumenta la velocidad de navegación ya que las páginas se construyen con mayor rapidez en los navegadores, además, todas se verán iguales en los

mismos; brindando compatibilidad entre los que cumplen con los estándares web y garantizando así que en versiones posteriores de estos funcionen correctamente.

JavaScript es un lenguaje de programación orientado a objetos. Los códigos JavaScript se insertan directamente en el mismo documento HTML. El manejo de objetos facilita la programación de páginas interactivas. Es dinámico, responde a eventos en tiempo real, permitiendo que se puedan cambiar totalmente el aspecto de la página al gusto del usuario. Es fácil de aprender, rápido y potente, una vez que se conocen las bases del lenguaje, no hay que esforzarse mucho para crear grandes aplicaciones. Es ideal para agregar funciones rápidas a una página web. Es un lenguaje de alto nivel que no realiza funciones directas al nivel de la máquina, sin embargo, es capaz de trabajar muchas funciones con exploradores web o con el propio sistema donde se ejecuta el explorador. Su usabilidad viene dada ya que es el lenguaje que más se emplea en la web. La reducción de la carga del servidor es una de las principales razones por la que los desarrolladores de aplicaciones web han adoptado JavaScript. Este se puede hacer cargo de funciones del lado del cliente de las cuales se encargaba el servidor, como las validaciones, mostrar mensajes, comprobar campos, entre otras.

CSS se le conoce como Cascading Style Sheets u hojas de estilo en cascada y se utilizan para darle forma y aplicar el diseño o presentación al contenido HTML y XML que forman una página o aplicación web, es decir, es el mecanismo que permite separar los contenidos definidos mediante XHTML y el aspecto que deben presentar esos contenidos. Una ventaja añadida de la separación de los contenidos y su presentación es que los documentos XHTML creados son más flexibles, ya que se adaptan mejor a las diferentes plataformas: pantallas de ordenador, pantallas de dispositivos móviles, impresoras y dispositivos utilizados por personas discapacitadas (17). Permite separar la capa de diseño del código HTML o XML, de modo que se puede modificar el diseño sin tener que transformar el código HTML o XML de manera constante (18). Otro propósito es que posibilita realizar cambios a múltiples elementos dentro del código al mismo tiempo si estos poseen igual identificador el cual puede ser un ID, clase o elemento (Tag) de HTML, agilizando así el proceso de cambios.

Después de varios años trabajando con la especificación CSS2 que definía varias funcionalidades, la web se prepara para un cambio en las hojas de estilo en cascada (19). La nueva versión CSS3 no es una especificación completa, sino que está dividida en diferentes módulos. Cada uno de ellos añade nuevas

capacidades o extiende aquellas ya existentes en especificaciones anteriores con el propósito de preservar compatibilidad hacia atrás. (20)

Se emplea CSS3 ya que posee varias mejoras en cuanto a interfaz gráfica, posicionamiento y tamaño de los objetos, usando condiciones de alineación para cada uno. Además que las nuevas funcionalidades permitirán el desarrollo de una aplicación que obedezca al fenómeno de la Web 2.0.

Lenguaje de programación del lado del servidor

PHP (acrónimo de Hipertext Preprocesor) es un lenguaje de programación script interpretado en el lado del servidor, utilizado para la generación de páginas web dinámicas. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas. PHP brinda un conjunto de ventajas que se muestran a continuación: (21)

- Muy fácil de aprender.
- Se caracteriza por ser un lenguaje muy rápido.
- Soporta en cierta medida la orientación a objeto. Clases y herencia.
- Es un lenguaje multiplataforma: Linux, Windows, entre otros.
- Capacidad de conexión con la mayoría de los manejadores de base de datos: MySQL, PostgreSQL, Oracle, entre otras.
- Capacidad de expandir su potencial utilizando módulos.
- Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Incluye gran cantidad de funciones.
- No requiere definición de tipos de variables ni manejo detallado del bajo nivel.

Uno de los problemas de las versiones anteriores de PHP era la clonación de objetos, que se realizaba al asignar un objeto a otra variable o al pasar un objeto por parámetro en una función. Para solventar este problema PHP 5 hace uso de los manejadores de objetos, que son una especie de punteros que apuntan hacia los espacios en memoria donde residen los objetos. Cuando se asigna un manejador de objetos o se pasa como parámetro en una función, se duplica el propio manejador de objeto y no el objeto en sí. Por

esta razón y las ventajas que brinda el lenguaje en general es que se hace uso del mismo para el desarrollo de la solución.

Sistema gestor de base de datos

Un **SGBD** (Sistema Gestor de Bases de Datos) o **DBMS** (Data Base Management System) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos. (22)

Un SGBD debe permitir:

- Definir una base de datos: especificar tipos, estructuras y restricciones de datos.
- Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.
- Manipular la base de datos: realizar consultas, actualizarla, generar informes.

Las características de un Sistema Gestor de Base de Datos son:

- **Abstracción de la información:** Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos, haciendo que estos sean transparentes al usuario.
- **Independencia:** La capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.
- **Redundancia mínima:** Un buen diseño de una base de datos logrará evitar la aparición de información repetida o redundante.
- **Consistencia:** Los datos repetidos se actualicen de forma simultánea.
- **Seguridad:** Se debe garantizar que la información se encuentre segura frente a usuarios malintencionados, que deseen manipular o destruir la información.
- **Integridad:** Protección de los datos ante fallos de hardware, datos introducidos por usuarios descuidados o cualquier otra circunstancia capaz de corromper la información almacenada.
- **Respaldo y recuperación:** Proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada, y de restaurar a partir de estas copias los datos que se hayan podido perder.
- **Control de la concurrencia:** Controlar el acceso concurrente a la información, que podría derivar en inconsistencias.

Existen muchos gestores de bases de datos, entre los que se pueden mencionar Oracle, PostgreSQL y MySQL. Estos últimos han alcanzado un nivel de madurez que garantiza su estabilidad y su continuidad a largo plazo. (23)

Se emplea PostgreSQL en su versión 8.4 porque es un sistema de gestión de bases de datos relacional, orientado a objetos. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación en cuanto a fiabilidad, corrección e integridad de datos. Se ejecuta en los principales sistemas operativos, incluyendo Linux, Unix y Windows. Tiene soporte completo para llaves foráneas, uniones, vistas, disparadores y procedimientos almacenados. Incluye la mayor parte de los tipos de datos: integer, boolean, char, varchar, date, interval y timestamp. También es compatible con el almacenamiento de objetos binarios grandes como imágenes, sonidos o vídeo. Tiene interfaces nativas de programación en C / C + +, Java, .Net, Perl, Python, Ruby, entre otros y una documentación amplia.
Ventajas de PostgreSQL: (24)

- Gran escalabilidad, se ajusta al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta y algunos casos llegando a soportar el triple de carga que soporta MySQL.
- Soporta transacciones y llaves foráneas con comprobaciones de integridad referencial.
- Tiene mejor soporte para triggers y almacenamiento de procedimientos en la propia base de datos, comparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle.
- Implementa subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz, y ofreciendo soluciones en campos en los que MySQL no podría.

En cuanto a las ventajas de la versión 8.4 se tienen: (25)

- Restauración de la base de datos usando procesos paralelos, acelerando la recuperación de un respaldo hasta en 8 veces respecto a la versión anterior.
- Privilegios por columna, para poder controlar el acceso a un nivel de detalle mayor.
- Configuración de idioma y ordenamiento por base de datos para que se pueda seleccionar la configuración más adecuada dependiendo del idioma que se requiera.
- Nuevas herramientas para monitorear las consultas, entregando mayor información a los administradores para saber lo que está sucediendo en la base de datos.

- Implementación de funcionalidades avanzadas que permiten realizar consultas complejas en una sola expresión en donde antes se requerían varias.
- Mejoras en procedimientos almacenados, por ejemplo usar valores por omisión en la declaración de parámetros o listas de argumentos de largo variable, al más puro estilo C++/Java.

Servidor web

Un servidor web es un programa que gestiona cualquier aplicación en el lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente, generando una respuesta en cualquier lenguaje o aplicación en el lado del cliente. El código recibido por el cliente suele ser compilado y ejecutado por un navegador web. (26)

Apache

El servidor HTTP Apache es un servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Apache es usado primariamente para enviar páginas web estáticas y dinámicas en la World Wide Web. La licencia de software bajo la cual el software de la fundación Apache es distribuido es una parte distintiva de la historia de Apache HTTP Server y de la comunidad de código abierto. La Licencia Apache permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original. (27)

Ventajas de Apache: (27)

- Modular: La arquitectura del servidor Apache consta de diversos módulos que aportan muchas de las funcionalidades que podrían considerarse básicas para un servidor web.
- Código abierto: La Licencia Apache permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original.
- Multiplataforma: Es usado en varias plataformas, tanto, libres como propietarias.
- Extensible: Se pueden añadir módulos y escribir sus propios códigos adaptándolos a las funciones del servidor Apache, de forma tal, que se ajusten a las necesidades de los desarrolladores.
- Popular: Es fácil conseguir ayuda y soporte. Apache tiene amplia aceptación en la red desde 1996 y es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en el 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo.

Se utiliza HTTP Apache 2.0 teniendo en cuenta las ventajas del servidor en general como las particulares de esta versión, entre las que se pueden citar: (27)

- Hebrado en Unix: En los sistemas Unix que soportan hebras POSIX, la nueva versión de Apache puede ejecutarse en modo híbrido multiproceso-multihebra. Esto mejora la escalabilidad para muchas aunque no para todas las configuraciones.
- Nuevo sistema de configuración y compilación: El sistema de configuración y compilación ha sido escrito de nuevo desde cero para basarlo en autoconf y libtool. Esto hace que el sistema de configuración de Apache se parezca ahora más al de otros proyectos de código abierto.
- Soporte multiprotocolo: La nueva versión tiene la infraestructura necesaria para servir distintos protocolos, por ejemplo, se ha escrito el módulo mod_echo.
- Nueva interfaz de programación (API) de Apache: La API para los módulos ha cambiado significativamente en la nueva versión. Muchos de los problemas de ordenamiento y prioridad de módulos de la versión 1.3 han desaparecido. Apache 2.0 hace automáticamente mucho de lo que es necesario, y el ordenamiento de módulos se hace ahora por hooks, lo que ofrece una mayor flexibilidad. También se han añadido nuevas llamadas que ofrecen capacidades adicionales sin tener que parchear el núcleo del servidor Apache.
- Filtros: Los módulos de Apache pueden ahora escribirse para que se comporten como filtros que actúan sobre el flujo de contenidos como salen del servidor o como son recibidos por el servidor. Esto permite, por ejemplo, que el resultado de un script CGI sea analizado por las directivas Server SideInclude usando el filtro INCLUDES del módulo mod_include.
- Mensajes de error en diferentes idiomas: Los mensajes de error que se envían a los navegadores están ahora disponibles en diferentes idiomas, usando documentos SSI.
- Configuración simplificada: Muchas directivas que podían inducir a confusión han sido simplificadas. Las directivas Port y BindAddress han desaparecido; para configurar la dirección IP en la que escucha el servidor ahora se usa únicamente la directiva Listen; la directiva ServerName especifica el nombre del servidor y el número del puerto solo para redireccionamiento y reconocimiento de host virtual.
- Actualización de la librería de expresiones regulares: Apache 2.0 incluye la librería de expresiones regulares compatibles con Perl (PCRE). Ahora, cuando se evalúan las expresiones tipo, se usa siempre la potente sintaxis de Perl 5.

Frameworks de desarrollo

Un marco de trabajo o framework puede ser considerado de varias formas. Algunas definiciones de framework se muestran a continuación:

- Conjunto de clases que cooperan para formar un diseño reutilizable en un dominio concreto de software. (28)
- Es una aplicación semicompleta, una estructura reutilizable común para compartir entre aplicaciones. Son incorporados por los desarrolladores en sus aplicaciones con el objetivo de cumplir sus necesidades concretas. (29)
- Es un conjunto de clases e interfaces que cooperan para resolver un problema específico en el ámbito del software. Entre sus características podemos encontrar que está compuesto por un conjunto de clases o componentes, cada uno de ellos provee o representa una abstracción de un concepto en particular. Define cómo estas abstracciones trabajan juntas para resolver un problema. Todos los objetos de los frameworks son reutilizables. (30)

Los frameworks simplifican el desarrollo de las aplicaciones mediante la automatización de las tareas comunes. Pueden incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Además de que proporcionan estructura al código fuente, forzando al programador a crear código más legible y más fácil de mantener.

Luego de analizar las definiciones anteriores se puede concluir que un framework es uno conjunto de clases que cooperan orientadas a un diseño reutilizable, formando una infraestructura que agilice y facilite el desarrollo de aplicaciones web.

Framework del lado del cliente

En la actualidad existe gran incremento en las tecnologías para el desarrollo de aplicaciones web dinámicas. Las aplicaciones asincrónicas son cada vez más usadas y más complejas ya que incorporan efectos visuales, animaciones, entre otros. El desarrollar estos efectos desde cero puede resultar complicado sobre todo si se tiene que presentar la solución con muy poco tiempo, en este tipo de situaciones el empleo de librerías como jQuery facilitan el desarrollo de la aplicación.

jQuery es una biblioteca de JavaScript rápida y concisa que simplifica el documento HTML, manejo de eventos, animación e interacciones AJAX para un rápido desarrollo web. Está diseñada para cambiar la manera en que se escribe el código JavaScript. (31)

Ventajas de jQuery:

- La curva de aprendizaje es menor que la de sus similares.
- Es ligero en comparación con otros marcos de JavaScript.
- Tiene una amplia gama de plugins disponibles para las necesidades específicas.
- Ahorra muchas líneas de código.
- Hace transparente el soporte de la aplicación para los navegadores principales.
- Provee de un mecanismo para la captura de eventos.
- Proporciona un conjunto de funciones para animar el contenido de la página en forma muy sencilla.
- Por defecto integra funcionalidades para trabajar con AJAX.
- Posee una comunidad de desarrollo, muchos plugins y buen soporte.

Se emplea jQuery en versión 1.5.1 por las características y ventajas que presenta esta biblioteca.

Framework del lado del servidor

Paralelo al avance de los frameworks del lado del cliente, han ido surgiendo framework del lado del servidor. Anteriormente se definió como lenguaje de programación del lado del servidor PHP 5 por las características expuestas, por lo que se decide utilizar un framework de desarrollo implementado en PHP. En este caso se optó por la selección de Symfony como framework de apoyo a la implementación en el lado del servidor.

Symfony es un framework PHP que facilita el desarrollo de las aplicaciones web. Se encarga de todos los aspectos comunes y aburridos de las aplicaciones web, dejando que el programador se dedique a aportar valor desarrollando las características únicas de cada proyecto.

Symfony es uno de los frameworks PHP más populares entre los usuarios y las empresas, ya que permite que los programadores sean mucho más productivos a la vez que crean código de más calidad y más fácil de mantener. Symfony es maduro, estable, profesional y está muy bien documentado. (32)

Symfony se diseñó para que se ajustara a los siguientes requisitos: (33)

- 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.
- Preparado para aplicaciones empresariales y es adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Las razones que motivaron a tomar la decisión de optar por Symfony en su versión 1.4.3 fueron, que es un completo framework diseñado para optimizar, gracias 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. Es estable para desarrollar aplicaciones a largo plazo. Es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Es un framework multiplataforma. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.

Lenguaje de Modelado

UML (Unified Modeling Language, traducido al idioma español como Lenguaje Unificado de Modelado) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimientos sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. (34)

UML incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos. (34)

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. Un sistema se modela como una colección de objetos discretos que interactúan para realizar un trabajo que finalmente beneficia a un usuario externo. (34)

Entorno Integrado de Desarrollo

Un **IDE** (Integrated Development Enviroment, traducido al español Ambiente Integrado de Desarrollo) es un entorno de programación, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica que además pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. (35)

Un IDE debe tener las siguientes características: (35)

- Multiplataforma.
- Soporte para diversos lenguajes de programación.
- Integración con sistemas de control de versiones.
- Reconocimiento de sintaxis.
- Extensiones y componentes para el IDE.
- Integración con framework populares.
- Depurador.
- Importar y exportar proyectos.
- Múltiples idiomas.
- Manual de usuarios y ayuda.

NetBeans

Es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java, aunque permite programar en distintos lenguajes. Ofrece un excelente entorno para programar en PHP. Tiene un excelente balance entre una interfaz con múltiples opciones y un aceptable completamiento de código. (36)

NetBeans en su versión 6.9 se caracteriza por: (36)

- Creación de proyectos PHP: NetBeans provee a los desarrolladores de una estructura para los proyectos que se puedan crear junto a este IDE. Propone un esqueleto para organizar el código fuente, el editor conjuntamente integra los lenguajes como HTML, JavaScript y CSS. Además NetBeans posee un sistema para examinar todo los directorios de cada proyecto, haciendo reconocimiento y carga de clases, métodos y objetos, para acelerar la programación.
- Integración con Symfony: Permite realizar aplicaciones utilizando frameworks, haciendo el trabajo más ágil.
- Editor de código fuente: Mejora en su editor, sobre todo en el editor de PHP, es mucho más rápido y a la vez robusto, contiene más ayuda en línea, reconocimiento de sintaxis y todo lo que provee la última versión de PHP, la 5.3.
- Depuración de PHP: NetBeans integra muy bien la utilización Xdebug, permitiendo inspeccionar y examinar cada variable local, establecer puntos de interrupción y evaluar el código en nuestra lógica.
- Integración con sistemas de control de versiones: Esta es una de las condiciones necesarias para los proyectos y es la posibilidad de contar con la integración de sistemas de control de versiones, tales como SVN, CVS, Mercurial y Git. Desde el editor es posible realizar la administración de estos sistemas versionados, sus commit, branch, importar, exportar, revert, clonar, entre otras funciones.

Herramientas CASE

Las herramientas CASE (Computer Aided Software Engineering, traducido al español como Ingeniería de Software Asistida por Computadora) son un conjunto de programas y ayudas que dan asistencia a los

analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software, es decir, son el conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases.

La principal ventaja de la utilización de una herramienta CASE, es la mejora de la calidad de los procesos realizados y el aumento de la productividad. Para conseguir estos dos objetivos es conveniente contar con una organización y una metodología de trabajo, además de la propia herramienta. (37)

Visual Paradigm

Es una herramienta CASE que utiliza como lenguaje de modelado UML y que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación (38). Se integra con varios IDE como Eclipse, .Net, NetBeans, entre otros. Provee un generador de mapeo de objetos relacionales para los lenguajes de programación Java, .Net y PHP. Se utiliza Visual Paradigm en su versión 3.5 dadas las características y ventajas que brinda.

Arquitectura de software

Patrón Modelo Vista Controlador

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el modelo es el sistema de gestión de base de datos y la lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista. La finalidad del modelo es mejorar la reusabilidad por medio del desacople entre la vista y el modelo. (39)
Los elementos del patrón son los siguientes: (39)

- El modelo, responsable de acceder a la capa de almacenamiento de datos, define las reglas de negocio, es decir, la funcionalidad del sistema y lleva un registro de las vistas y controladores del sistema.
- El controlador, responsable de recibir los eventos de entrada y contiene reglas de gestión de eventos.

- Las vistas, responsables de recibir datos del modelo y mostrarlos al usuario. Tienen un registro de su controlador asociado. Pueden dar el servicio de actualización, para que sea invocado por el controlador o por el modelo cuando es un modelo activo que informa de los cambios en los datos producidos por otros agentes.

Se hace uso del MVC por las ventajas y características que el mismo posee: (39)

- Es posible tener diferentes vistas para un mismo modelo.
- Es posible construir nuevas vistas sin necesidad de modificar el modelo inferior.
- Proporciona un mecanismo de configuración a componentes complejos mucho más tratable que el puramente basado en eventos (el modelo puede verse como una representación estructurada del estado de la interacción).

Conclusiones

Luego de realizar un estudio del arte de los sistemas de recomendación, teniendo en cuenta la forma en que procesan la información se concluye que de las clasificaciones se utilizará un sistema basado en contenidos. Además se analizaron los conceptos correspondientes con los cuestionarios interactivos estableciéndose la relación entre las tipologías definidas para estos y las habilidades especificadas en los niveles de desempeño. Finalmente se aprobaron la metodología de desarrollo, herramientas y tecnologías a utilizar, concluyendo que como metodología de desarrollo a utilizar será el Proceso Unificado de Desarrollo (RUP), como lenguajes de programación JavaScript de lado del cliente y PHP 5 del lado del servidor, como lenguaje de modelado UML, como frameworks jQuery en su versión 1.5.1 y Symfony en su versión 1.4.3, como servidor web Apache 2, como gestor de base de datos PostgreSQL en su versión 8.4, como entorno integrado de desarrollo NetBeans 6.9, como herramienta CASE Visual Paradigm en su versión 3.5 y como patrón de arquitectura Modelo Vista Controlador.

Capítulo 2 Concepción y características del sistema

Introducción

El modelado del negocio provee una vista estática de la estructura de la organización y una vista dinámica dentro de los procesos de la misma. Permite entender los problemas actuales de la compañía, asegurando que los clientes, usuarios, desarrolladores y otros involucrados tengan igual entendimiento de la empresa. En este capítulo se describe una propuesta de un sistema de selección didáctica basado en la interacción de los estudiantes con el módulo Ejercicios de la colección El Navegante. Además se representa el Modelo de dominio donde se capturan los objetos importantes del entorno donde será empleado el sistema. Se identifican los requerimientos funcionales y los no funcionales, se elabora el diagrama de casos de uso y las descripciones de cada uno de ellos.

Herramienta de selección didáctica

La herramienta que se propone se incluye entre las opciones que tiene el estudiante para la realización de los cuestionarios interactivos. Una vez que se accede al módulo Ejercicios de la colección se construye el perfil del usuario activo. Dicho perfil tiene en cuenta el nivel de vencimiento de cada una de las habilidades que debe desarrollar un estudiante de secundaria básica. De esta manera, si se selecciona la realización de cuestionarios con ayuda del sistema recomendador, se muestra un listado de los que más aportan en el desarrollo de las habilidades para cada tema marcado. Cada vez que el estudiante realiza un ejercicio, se registra la acción como traza en el sistema y por lo tanto se actualiza el perfil del usuario.

Con el fin de ofrecer recomendaciones acordes con las necesidades de cada estudiante, el sistema debe tener en cuenta varias características de los ejercicios y de los propios usuarios.

En el modelo se establecen las relaciones entre las tipologías de ejercicios y las habilidades que contribuyen a potenciar. Por esta razón se puede afirmar que cada ejercicio permite el desarrollo de habilidades que pueden haber sido vencidas o no por el usuario.

En el caso de los estudiantes, los datos son recolectados de las trazas que produce su interacción con el módulo Ejercicios. Teniendo en cuenta los cuestionarios que hayan sido resueltos de manera satisfactoria, se puede evaluar el nivel en que un estudiante ha vencido una habilidad.

Reglas de asociación

La inteligencia artificial y las estadísticas brindan un conjunto de técnicas para realizar una correcta minería de datos. Entre estas técnicas se encuentra la asociación. Esta consiste en la búsqueda de ítems que aparezcan en los mismos conjuntos de transacciones, para, a partir de ahí establecer reglas que indiquen dependencia entre algunos de los ítems. La forma más natural para representar estas asociaciones y dependencias son las reglas de asociación, que fueron introducidas por Awragal, Imielinski y Swami en (40) y (41).

Awragal et al (40) plantearon que una regla de asociación expresa la probabilidad de que la ocurrencia de un evento implique la ocurrencia de otro, planteando la siguiente formalización:

> Considérese un conjunto de ítems $I = \{i_1, i_2, i_3,..., i_{n-1}, i_n\}$, donde cada elemento *"i"* perteneciente a *I* puede asumir valores binarios 1 o 0 (verdadero o falso) que expresan respectivamente su presencia o ausencia en el conjunto. Además, sobre los elementos que componen *I*, se tiene un conjunto de transacciones $T = \{t1, t2, t3,..., tn-1, tn\}$, donde cada elemento *"ti"* perteneciente a *T* corresponde a un conjunto de ítems presentes en *I*, tal que $t \subseteq I$. Un ítem es considerado como una instancia de un atributo, o sea, su valor en un determinado registro (u objeto) que represente una transacción.

Además:

> Se tiene que si todos los elementos pertenecientes a un conjunto de ítems *A* en una transacción *"t"*, entonces el conjunto *A* es subconjunto de *"t"*, o sea, $A \subseteq t$. Una regla de asociación puede tener una representación equivalente a $A \rightarrow B$, es decir, la existencia de los ítems que pertenecen al conjunto *A* (antecedente), en una transacción, implican la existencia de los ítems que pertenecen al conjunto *B* (consecuente), donde $A \subseteq I$ y $B \subseteq I$. Es importante señalar que los ítems de *A* son distintos de los ítems de *B*, o sea, $A \cap B = \emptyset$.

La utilización de reglas de asociación puede contribuir a encontrar tendencias, las cuales posibilitan comprender y explorar patrones de comportamiento en los datos.

Algoritmo propuesto para la herramienta

Dadas las características anteriormente descritas, se emplea para construir la solución, un sistema basado en reglas de asociación. De esta manera, se pueden generar reglas del tipo $P \rightarrow Q$, donde, por ejemplo, *P* y *Q* representan las habilidades que debe vencer el estudiante. Así se puede interpretar que cuando un usuario ha vencido la habilidad *P*, está listo para realizar ejercicios que contribuyan a la habilidad *Q*. Por ejemplo: Si el porciento de vencimiento de la habilidad observar, analizar, identificar, describir e interpretar en el tema Introducción a la Educación Artística es mayor que el 80 % entonces se asignarán cuestionarios para vencer la habilidad reflexionar.

Modelo de dominio

El proceso de negocio del sistema de recomendación comprende pocos conceptos, por tanto, se decide utilizar un modelo de dominio. Este es una representación de los objetos que existen o los eventos que suceden en el entorno en que trabaja el sistema. Es un caso especial del modelo de negocio más completo que contribuye a una comprensión del problema que se supone que el sistema resuelve en relación a su contexto. (42)

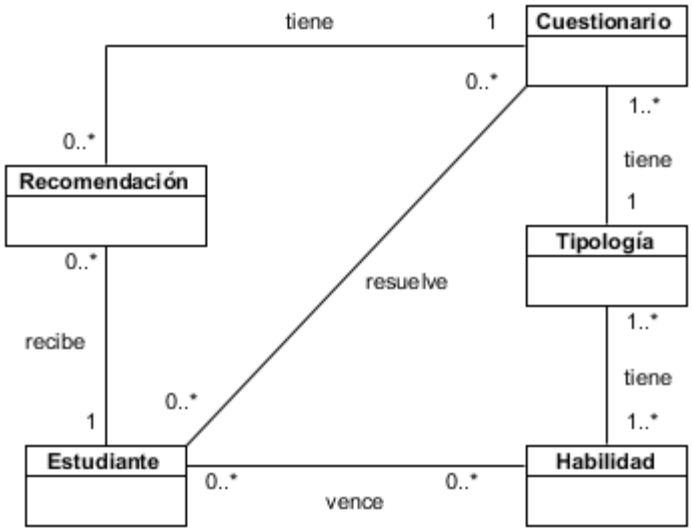


Fig. 1 Modelo de dominio

En la figura anterior se presenta el modelo de dominio de la aplicación. En este se tiene en cuenta las características del sistema y se pueden observar las relaciones entre los conceptos fundamentales relacionados con el sistema de recomendación. Consiste en la recomendación de cuestionarios que recibe un estudiante a partir de una habilidad. Esta debe ser vencida por el mismo y está relacionada con las tipologías definidas para los cuestionarios, con el fin de que el estudiante avance a un nivel de desempeño superior. Este modelo tiene el propósito de alcanzar un lenguaje común entre desarrolladores, clientes y usuarios finales.

Principales conceptos:

Estudiante: Persona que cursa estudios en las instituciones de nivel secundario.

Habilidad: Son formaciones psicológicas mediante las cuales el sujeto manifiesta en forma concreta la dinámica de la actividad con el objetivo de elaborar, transformar, crear objetos, resolver situaciones y problemas, actuar sobre sí mismo, autorregularse. Las habilidades se encuentran contenidas en las tipologías definidas en la colección.

Tipología: Cada uno de los catorce tipos de ejercicios que están definidos en la colección El Navegante en su versión multiplataforma.

Cuestionario: Controla el aprendizaje de los estudiantes de la enseñanza secundaria.

Recomendación: Es la sugerencia de cuestionarios que recibe el estudiante de acuerdo a su perfil.

Requisitos

Los requisitos tienen como principal objetivo guiar el desarrollo hacia un sistema correcto. Este propósito es conseguido por medio de los requisitos funcionales y los no funcionales que presente el software a desarrollarse. (43)

Requisitos funcionales

Los requisitos funcionales son las condiciones o capacidades que el sistema debe cumplir lo suficientemente bien para poder llegar a un acuerdo con los clientes. A continuación se muestran los requisitos funcionales del sistema de recomendación.

RF 1: Construir el perfil del estudiante.

RF 2: Actualizar el perfil del estudiante.

RF 3: Recomendar cuestionarios.

Requisitos no funcionales

Los requisitos no funcionales son las cualidades o propiedades del sistema, como restricciones del entorno o de la implementación, rendimiento, dependencias de la plataforma, soporte, entre otras.

Usabilidad

La utilización de la colección por parte de los estudiantes u otros usuarios sin avanzados conocimientos de computación, no requiere de previa preparación, debido al diseño sencillo y estándar que tienen los productos de la misma.

Fiabilidad

Aunque los productos de la colección serán implementados sobre plataforma web no forma parte del alcance del proyecto lograr su funcionamiento sobre la red, sino que funcionarán de forma local, deberán estar instalados todos los software que se necesiten para correr la aplicación, por tanto la aplicación no

depende de ningún elemento externo y la fiabilidad estará garantizada por el correcto funcionamiento de los software que se necesiten para ejecutar los productos de la colección.

Interfaces de usuario

Las interfaces de usuario estarán acorde a las edades a las que van dirigidos cada uno de los diez productos que integran la colección El Navegante.

Las interfaces se desarrollarán teniéndose en cuenta elementos de las aplicaciones con tecnología multimedia diseñadas para escritorio y las aplicaciones web, logrando equilibrar la eficiencia del software sobre la web con la interactividad y el diseño visual que exige la aplicación.

Restricciones de diseño

Los productos se implementarán como una aplicación web, garantizándose su funcionamiento solamente sobre el navegador Mozilla Firefox versión 3.5 o cualquier otra versión superior y sistema operativo Ubuntu en su versión 10.04 o superior, de forma local en cada máquina.

La facilidad de desarrollo con software libre es debido a la amplia gama de herramientas disponibles para el desarrollo de aplicaciones web. La resolución de pantalla a la que deben visualizarse los productos de la colección es 1024 x 768 píxeles. Los contenidos de la aplicación estarán soportados sobre una base de datos en PostgreSQL. El framework seleccionado para el desarrollo es Symfony 1.4.3, siendo la versión utilizada compatible solamente con PHP 5. El estilo arquitectónico a emplear será el Modelo-Vista-Controlador y para la implementación de las clases de la vista se usará el framework de JavaScript jQuery.

Estándares aplicables

Durante el proceso de desarrollo de software cada entregable es sometido a pruebas de liberación, en las que se evalúan las características de calidad definidas por la norma ISO/IEC 9126. Para el diseño de la base de datos se usará un convenio para el nombre de las tablas, así como para las vistas y funciones que sean necesarias implementar. La codificación estará basada en el estándar recomendado en el sitio oficial de PHP y disponible en http://pear.php.net/manual/en/standards.php. Podrán realizarse las adecuaciones que se crean necesarias con previo acuerdo entre las partes. Los contenidos serán

almacenados utilizando el sistema de metadatos definidos por el estándar LOM, pretendiéndose en desarrollos futuros incorporar soporte total al estándar SCORM.

Modelo de casos de uso

El modelo de casos de usos permite que los desarrolladores y clientes lleguen a un acuerdo acerca de los requisitos de la aplicación. Es un modelo del sistema que contiene actores, casos de uso y sus relaciones.

Actores del sistema

Un actor es alguien o algo, externo al sistema, que interactúa con este y se beneficia de esa interacción. Puede ser una persona, un dispositivo de hardware u otro sistema.

Tabla 1. Actores del sistema

Actores del sistema	Descripciones
Estudiante	Persona que interactúa con la aplicación con el objetivo de aprender y vencer los contenidos educativos expuestos.

Diagrama de casos de uso de sistema

Un caso de uso es un fragmento de la funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. De manera más clara, un caso de uso especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores.

El diagrama de casos de uso del sistema es el modelo que contiene actores, casos de uso y las relaciones entre estos.

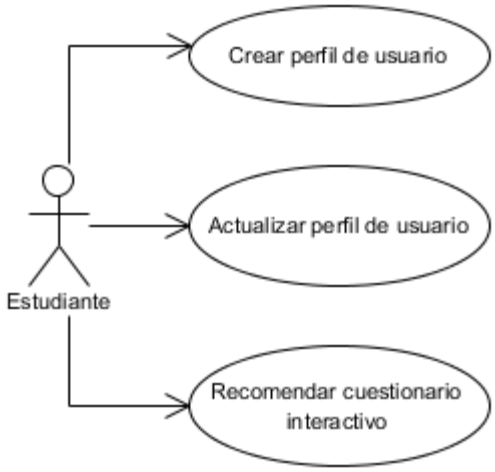


Fig. 2 Diagrama de casos de uso

Descripciones de casos de uso

A continuación se describe el caso de uso Recomendar cuestionario interactivo que servirá de guía para entender el funcionamiento del mismo, las restantes descripciones se pueden consultar en **¡Error! No se encuentra el origen de la referencia.**

Tabla 2. Descripción del CU Recomendar cuestionario interactivo

Caso de uso:	Recomendar cuestionario interactivo
Actores:	Estudiante (inicia)
Resumen:	El caso de uso se inicia cuando el estudiante selecciona la opción Recomendados del módulo Ejercicios. El sistema muestra el listado de cuestionarios recomendados y permite seleccionar, ordenar y realizar los cuestionarios recomendados finalizando así el caso de uso.
Precondiciones:	Debe haberse generado el escritorio de trabajo del estudiante autenticado. Se debe haber seleccionado uno o varios temas del módulo Ejercicios.
Referencias:	RF 3

Prioridad:	Normal
Flujo normal de los eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción Recomendados del módulo Ejercicios.	
	2. El sistema muestra el listado de los cuestionarios recomendados. Y permite: • Seleccionar los cuestionarios a resolver • Seleccionar la opción Configuración predeterminada
3. Selecciona los cuestionarios a resolver.	
	4. Muestra el listado de cuestionarios seleccionados y permite: • Comenzar • Ordenar el listado de cuestionarios
5. Selecciona la opción Comenzar. Resuelve los cuestionarios seleccionados.	
	6. Actualiza el perfil del estudiante. Ver *CU Actualizar perfil del estudiante.*
	7. El caso de uso termina.
Flujo alterno	

3. a Seleccionar la opción de Configuración predeterminada.	
Acción del actor	**Respuesta del sistema**
	3 a.1 El sistema muestra la pantalla principal del módulo Ejercicios. 3 a.2 El caso de uso termina.
5. a Ordenar el listado de cuestionarios.	
Acción del actor	**Respuesta del sistema**
	5 a.1 El sistema muestra el listado ordenado de cuestionarios.
5 a.2 Regresa al paso 3 del flujo básico.	
Poscondiciones:	Se recomiendan los cuestionarios interactivos.

Conclusiones

En este capítulo se describió de manera detallada la herramienta de selección didáctica llegando a la conclusión que dadas las características anteriormente especificadas, se empleará para construir la solución, un sistema basado en contenido utilizando reglas de asociación. Además se realizó el modelado del negocio, se identificaron los requerimientos funcionales y no funcionales con los que debe cumplir el sistema y se elaboraron el diagrama de casos de uso y las descripciones de cada uno de ellos.

Capítulo 3 Análisis y diseño del sistema

Introducción

El análisis tiene como propósito fundamental conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea más fácil de mantener y que ayude a estructurar el sistema completo, incluyendo la arquitectura (44). El diseño es más que analizar los requerimientos refinándolos y estructurándolos pues se preocupa en realidad de dar forma al sistema de manera que proporcione vida a todos los requisitos, incluidos todos los no funcionales que incorpora el mismo (45). De esta forma la disciplina de análisis y diseño específica y describe cómo se va a implementar el sistema, es decir, los diseñadores de software determinan la mejor solución técnica a partir de los requerimientos, de la arquitectura del sistema más adecuada y el diseño detallado necesario previo a las actividades de implementación. En este capítulo se describe la solución que se propone a partir de los diagramas de clases del análisis asociado a las funcionalidades del sistema y se representan los diagramas de clases del diseño, que reflejan una vista interna del sistema.

Modelo de análisis

El modelo de análisis es una estructura de clases y paquetes estereotipados que proporciona la idea de cómo llevar a cabo la funcionalidad del sistema, es decir, un modelo de objetos conceptuales, que ayuda a refinar los requisitos y permite razonar sobre los aspectos internos del sistema. Se centra en los requisitos funcionales por lo que su objetivo fundamental es ver qué hace el sistema. Es utilizado como una entrada en las actividades del diseño y la implementación. (46)

Diagrama de clase del análisis

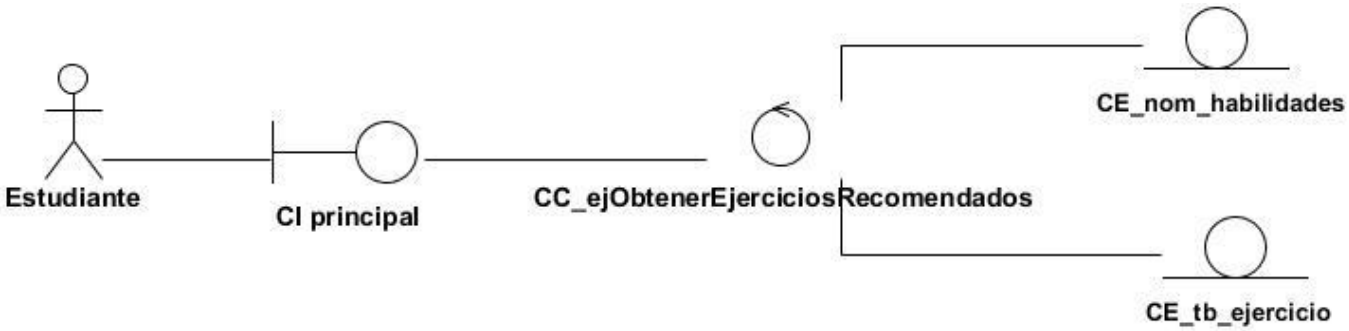


Fig. 3 DCA CU Recomendar cuestionario interactivo

El diagrama de clases de análisis del caso de uso Recomendar cuestionario interactivo ofrece una idea de cómo llevar a cabo la recomendación de cuestionarios como una funcionalidad del sistema. Cuenta con la clase interfaz principal que permite manejar la interacción del estudiante con la aplicación, es decir, maneja la comunicación entre el entorno y el interior del sistema. Esta clase será refinada en el diseño con el objetivo de facilitar la comunicación con otros sistemas y los mecanismos de interfaz de la implementación. Se relaciona con la controladora ejObtenerEjerciciosRecomendados, la cual coordina los eventos para la realización o especificación del caso de uso y controla las acciones sobre el mismo. Esta a su vez trata con las clases entidad tb_ejercicio y nom_habilidades que representan la información manejada en el caso de uso, así como el comportamiento asociado a eventos del mundo real, es decir, la información relacionada con los cuestionarios (enunciado, cantidad de intentos, etc.) y las habilidades (habilidad, tipo) que necesitan para el proceso de recomendación.

Diagramas de colaboración

Los diagramas de interacción explican gráficamente cómo los objetos a través de mensajes interactúan para realizar las tareas (47). Muestran las decisiones referentes a la asignación de responsabilidades entre los objetos, es decir, reflejan en los mensajes que son enviados a varias clases la descripción de comportamiento de los objetos del software.

UML define dos tipos de estos diagramas; ambos sirven para expresar interacciones semejantes o idénticas de mensaje:

- Diagramas de colaboración: Describen las interacciones entre los objetos en un formato de grafo o red.
- Diagramas de secuencia: Describen las interacciones en una especie de formato de cerca o muro.

A continuación se muestra el diagrama de colaboración del caso de uso Recomendar cuestionario interactivo que modela las interacciones entre las clases que intervienen en el proceso de recomendación proporcionando una visión de cómo se realiza el proceso en el sistema.

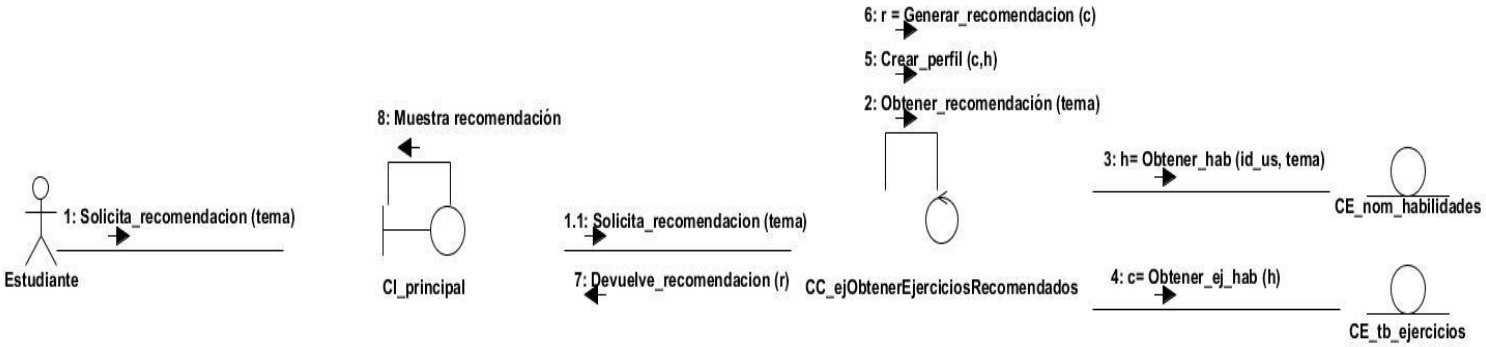


Fig. 4 DC CU Recomendar cuestionario interactivo

Modelo de diseño

El modelo de diseño es una representación no genérica, es decir, específica del plano de implementación. Se realiza teniendo en cuenta los requisitos tanto funcionales como no funcionales del sistema con el objetivo de adquirir una comprensión en profundidad de los aspectos relacionados con estos y los lenguajes de programación, componentes reutilizables, tecnologías de interfaz de usuario, entre otros (48). Es una disciplina clave dentro de la ingeniería de software a diferencia del análisis, que en muchos proyectos posee carácter opcional, siendo el centro de atención al final de la fase de elaboración e inicio de la fase de construcción.

De manera general, tiene como propósito transformar los requisitos tanto funcionales como no funcionales en un diseño de clases que describa las relaciones entre estas, en un lenguaje técnico lo más cercano posible al lenguaje de programación. Una entrada esencial en el diseño es el resultado del análisis, o sea, el modelo de análisis.

Aplicación de patrones de diseño

Los desarrolladores de software agrupan un conjunto tanto de principios generales como de soluciones basadas en aplicar ciertos estilos que les guían en la creación de software. Estos principios y estilos, si se codifican con un formato estructurado que describa el problema y la solución, y se les da un nombre, patrones de software (49).

Un patrón es una descripción de un problema y su solución, que recibe un nombre y que puede emplearse en otros contextos; en teoría, indica la manera de utilizarlo en diversas circunstancias (49), es decir, es un par problema-solución al que se le asigna un nombre y puede ser aplicado a varias situaciones, teniendo en cuenta las características del nuevo contexto en que se va a usar.

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí, adaptada para resolver un problema de diseño general en un contexto particular. (50)

Patrones GRASP

Los patrones de diseño de asignación de responsabilidades (GRASP) son los que ofrecen orientación de cómo asignar estas a los objetos ante determinada categoría de problemas, describiendo los principios fundamentales de la asignación. Los que se evidencian en el diseño de la herramienta son: Experto, Creador, Alta cohesión, Bajo acoplamiento y Controlador.

Experto

Asigna la responsabilidad al experto en la información, es decir, a la clase que cuenta con la información necesaria para cumplir con la responsabilidad (49). Es utilizado en la capa de abstracción del modelo ya que las clases generadas poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan.

Creador

Asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento promoviendo el bajo acoplamiento (49). Se evidencia, por ejemplo, en las acciones que realiza el controlador, las cuales crean objetos del modelo o los formularios que representan las entidades.

Alta cohesión

La información que almacena una clase debe ser coherente y en la medida de lo posible relacionada con ella (49). Es empleado en las acciones, que contienen varias funcionalidades relacionadas entre sí. Por ejemplo, la clase Actions define las acciones para las plantillas y colabora con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades de los mismos.

Bajo acoplamiento

Asigna responsabilidades para mantener el bajo acoplamiento. El acoplamiento es una medida de la fuerza con que está conectada una clase con otra. El propósito del patrón es tener las clases lo menos ligadas entre sí que se pueda y de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto (49). Este se evidencia en las clases pertenecientes a las acciones que heredan únicamente de sfAction para alcanzar un bajo acoplamiento de clases. Además, al no asociar las clases del modelo con las de la vista o el controlador, la dependencia entre las clases, en este caso, se mantiene baja.

Controlador

Mantiene el control actuando como intermediario entre una determinada interfaz y el algoritmo que la implementa. Además recibe los datos del usuario y los envía a las distintas clases según el método llamado y permite dividir los eventos del sistema en el mayor número de controladores para poder aumentar la cohesión y disminuir el acoplamiento (49). Dentro del sistema es utilizado en las clases de la capa del controlador del patrón MVC, como son: sfAction y las clases que heredan de la misma, ejObtenerEjerciciosRecomendados y ejCargarTemasAction.

Patrones Gof

Otros patrones de diseño de gran utilidad durante la fase de diseño orientado a objetos son los Gof, conocidos como patrones de la Pandilla de los Cuatro, dado que el libro que los explica fue escrito por cuatro autores. De estos se utilizan los patrones de diseño Iterator, Decorator y Mediador.

Iterador

Proporciona una forma de acceder a los elementos de diferentes colecciones de objetos sin necesidad de develar su representación interna (51). Este implementa una clase Iterator que define una interfaz para el acceso de los elementos de una lista. Se utiliza con el propósito de mantener la pista del elemento actual;

es decir, saber cuáles elementos ya han sido recorridos, en el caso que nos ocupa, estos son los cuestionarios.

Mediador

Define un objeto que encapsula cómo interactúan un conjunto de objetos. El mediador estimula la pérdida de acoplamiento ocultando las referencias explícitas entre los objetos, permitiendo variar su interacción de forma independiente (52). Se utiliza el patrón Mediador debido a que en el desarrollo de la herramienta es necesario asegurar que los objetos se puedan comunicar con otros sin tener que incluir referencias directas de código en sus clases, propiciando el bajo acoplamiento. La idea es tener clases lo menos relacionadas posibles, de modo que si se tiene que realizar modificaciones en alguna clase, exista menor repercusión en las demás, potenciando así la reutilización de código y disminuyendo la dependencia de clases.

Decorador

Patrón de tipo estructura, a nivel de objetos. Añade responsabilidades adicionales a un objeto dinámicamente (51). Se utiliza este patrón para la vista y el layout o plantilla global que decora el contenido de la misma.

Diagramas de clases de diseño

Un diagrama de clases del diseño representa las clases del sistema y sus relaciones. Este contiene una clase servidora que construye la clase cliente que está compuesta por un formulario que realiza envíos a la servidora; la cual accede a la controladora y esta a su vez a las entidades. Para una mejor organización del diseño, las clases son agrupadas en paquetes basándose en el patrón Modelo Vista Controlador. Los paquetes Controlador, Vista y Modelo agrupan las clases controladoras, las interfaces de usuarios y las tablas de la base de datos respectivamente. En el caso del subsistema Componentes de Symfony, contiene todos los componentes utilizados para la comunicación entre el paquete del control y la vista; como son los ficheros y clases internas. El subsistema Doctrine, es la representación de los componentes que utiliza el ORM Doctrine para la generación de la base de datos. El paquete Clases JavaScript (JS), modela la clase JavaScript relacionada con las vistas mostradas.

A continuación se muestran el diagrama de clases del diseño y el diagrama de paquetes.

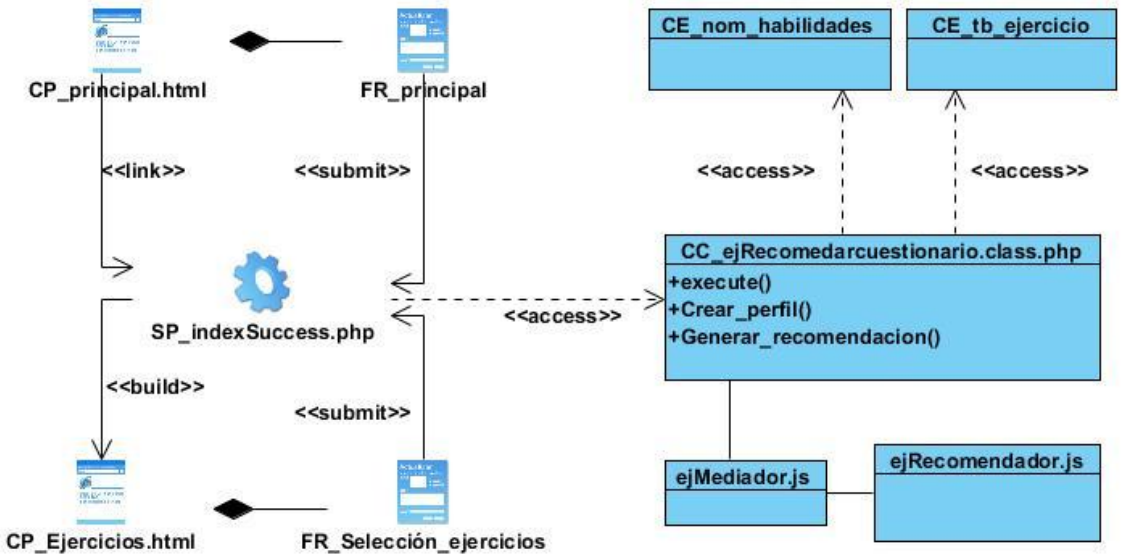


Fig. 5 DCD CU Recomendar cuestionario interactivo

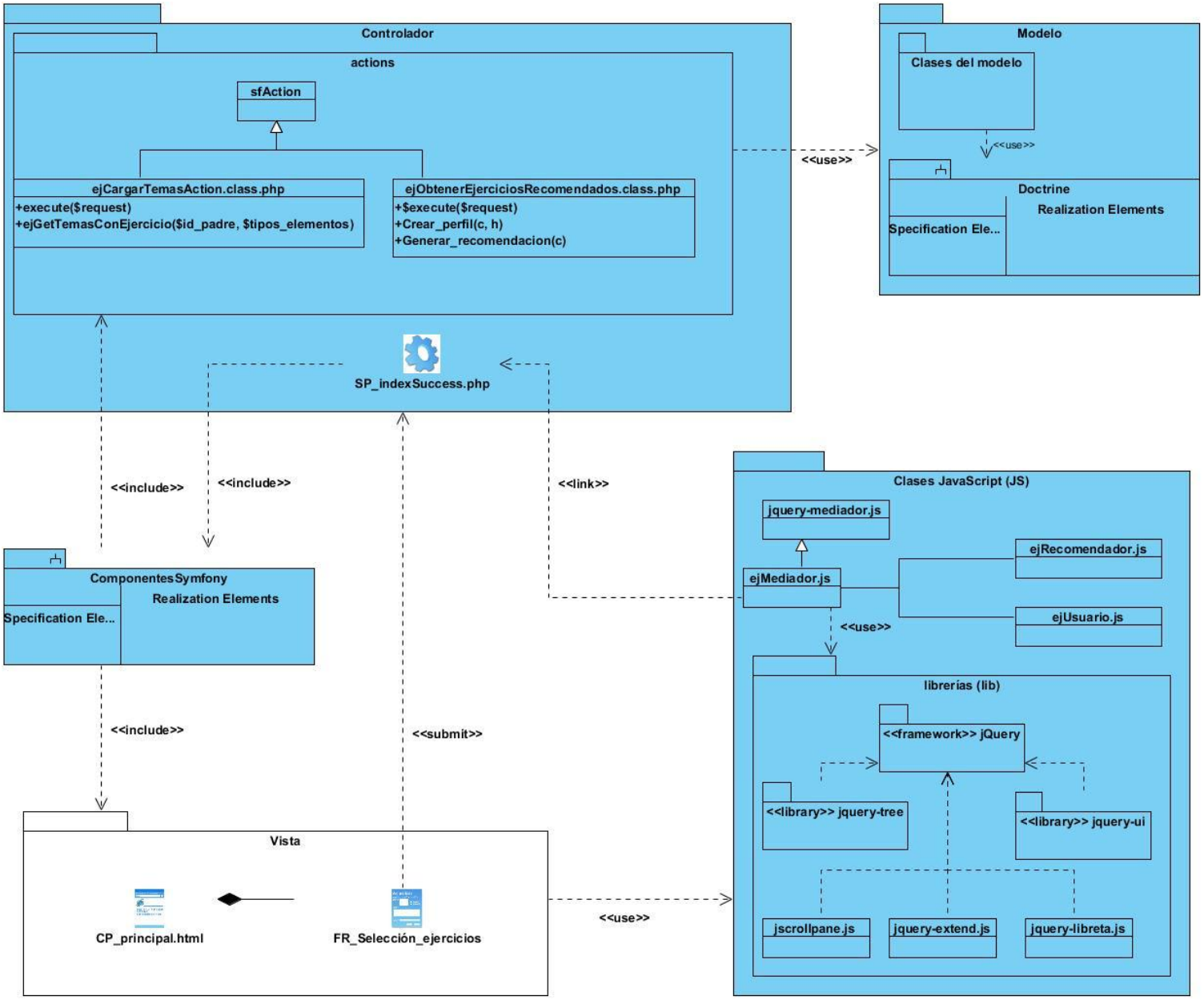


Fig. 6 DP CU Recomendar cuestionario interactivo.

Diagramas de secuencia

Los diagramas de secuencia muestran la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso. Estas interacciones son descritas en un formato de cerca

o muro. Permiten ver cómo se distribuyen las tareas entre los componentes e identificarse los modelos de interacción que dificultan la actualización de software.

A continuación se muestra el diagrama de secuencia para el caso de uso Recomendar cuestionario interactivo que contiene detalles de la implementación, incluyendo las interacciones entre los objetos y las clases que se usan para implementar la interfaz, además de mensajes intercambiados entre ellos en el proceso de recomendación:

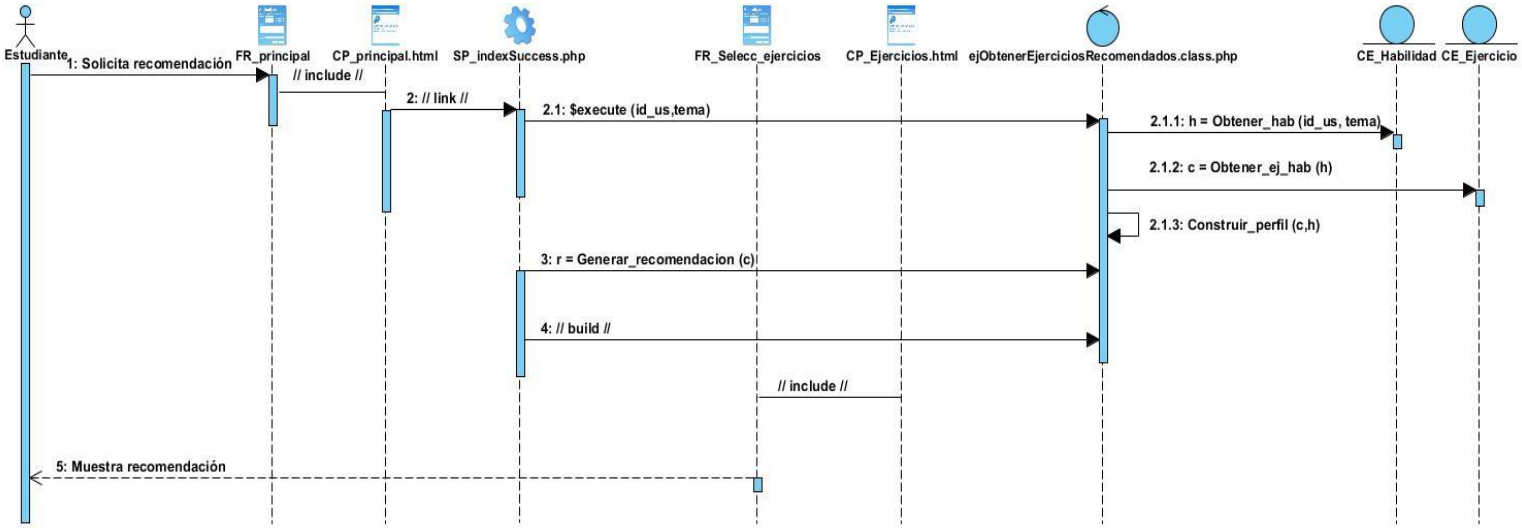


Fig. 7 DS CU Recomendar cuestionario interactivo

Modelo de datos

Un modelo es una representación de cualquier aspecto o tema extraído del mundo real. En un enfoque más relacionado al manejo de datos, se puede decir que un modelo de datos es un lenguaje utilizado para la descripción de una base de datos; que permite representar los elementos que intervienen en una realidad o en un problema dado y la forma en que se relacionan dichos elementos entre sí.

Los modelos de datos presentan dos sublenguajes:

- Lenguaje de Definición de Datos o DDL (Data Definition Language): Describe de una forma abstracta las estructuras de datos y las restricciones de integridad.

- Lenguaje de Manipulación de Datos o DML (Data Manipulation Language): Describe las operaciones de manipulación de los datos. Además la parte del DML enfocada a la recuperación de datos, se le suele conocer como Lenguaje de Consulta o QL (Query Language).

Se clasifican según su nivel de abstracción en tres modelos:

- Modelo de datos conceptuales: Describen las estructuras de datos y restricciones de integridad. Se utilizan durante la etapa de análisis de un problema dado y están orientados a representar los elementos que intervienen y sus relaciones.
- Modelo de datos lógicos: Se centran en las operaciones y se implementan en algún manejador de base de datos.
- Modelos de datos físicos: Son estructuras de datos a bajo nivel implementadas dentro del propio manejador.

A continuación se muestra el modelo de datos correspondiente a las clases persistentes identificadas en el sistema de recomendación. Proporciona la información de larga duración necesaria para realizar el proceso de recomendación de cuestionarios interactivos. Por ejemplo en la clase tb_sco se generalizan las características relacionadas con los objetos educativos que se manejan en el sistema de recomendación, como son los cuestionarios interactivos del módulo Ejercicios. En la clase tb_ejercicio se maneja la información relacionada con el cuestionario interactivo (enunciado, cantidad de intentos, etc.).

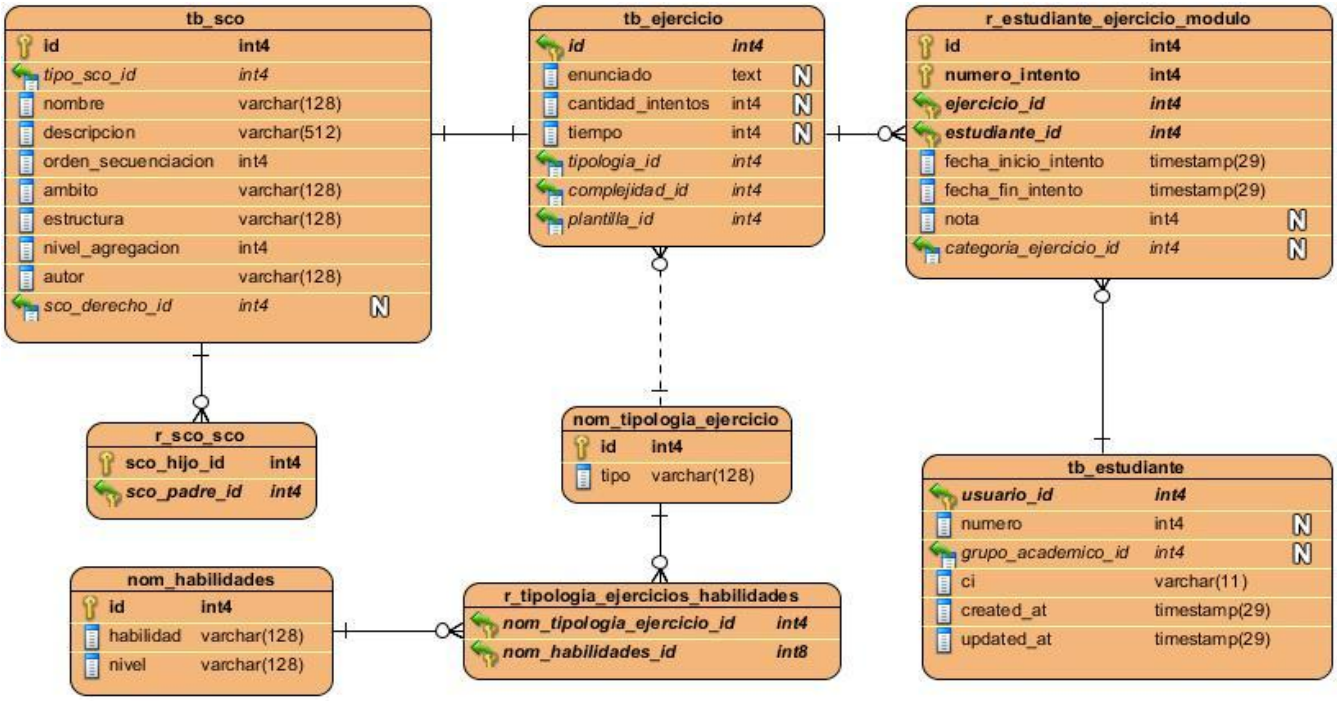


Fig. 8 Modelo de datos

Conclusiones

En este capítulo se realizó el análisis y diseño de los casos de uso del sistema donde se relacionan los requisitos funcionales y los no funcionales, llevándolos a una expresión de la programación a partir de la transformación de los requerimientos en un diagrama de clases del diseño. Se estructura cómo debe quedar el sistema haciendo uso de los patrones de diseño definidos para el mismo, contribuyendo a que dicho diseño se corresponda con el entorno de la implementación.

Capítulo 4 Implementación y pruebas

Introducción

Luego de realizarse el análisis y diseño del sistema de recomendación, el equipo de desarrollo cuenta con una vista más clara de cómo está estructurado el mismo. De esta manera queda el camino listo para comenzar a implementarlo como un todo, garantizando que cada funcionalidad cumpla con el diseño establecido. El propósito de la disciplina de implementación es definir la organización del código en términos de subsistemas de implementación organizados en capas. En esta, se toma como punto de partida el análisis y diseño para implementar el sistema en términos de componentes, entre los que están ficheros de código fuente, scripts, ficheros de códigos binarios y ejecutables. Posteriormente, deben desarrollarse las pruebas para verificar que el producto desarrollado cumpla con las especificaciones para el mismo.

Estándares de codificación

Los estándares de codificación son reglas que se utilizan para la escritura del código fuente. Permiten que los desarrolladores del proyecto puedan interpretar de manera eficiente la escritura del código; asegurando de que todos trabajen de forma coordinada y en un vocabulario común. Además aseguran la legibilidad del código y proporcionan una guía para el encargado del mantenimiento o actualización del sistema, con código claro y bien documentado.

Para el desarrollo del sistema de recomendación se utilizaron los estándares definidos por la arquitectura del proyecto Multisaber-Navegante. A continuación se hace mención de algunos y los restantes se pueden consultar en el Anexo 2 Estándares de codificación.

.

Indentación y espacios en blanco

Alineación y longitud de líneas

Indentar con dos espacios, sin tabulador, para que cualquier editor de texto reconozca correctamente la indentación. Utilizar líneas entre 75-80 caracteres de longitud, de esta manera se maneja mejor algunas herramientas y terminales.

Condicionales if, for, while, switch

Debe existir un espacio entre la palabra clave y el paréntesis de apertura. Esto mejora la legibilidad y disminuye la posibilidad de errores lógicos al agregar nuevas líneas de código.

Líneas plegadas

Cuando una expresión no cabe en una línea simple debido a su extensión se divide en más de una línea, siguiendo las siguientes precisiones:

- Dividir después de una coma.
- Dividir después de un operador.
- Alinear la nueva línea al inicio de la expresión en el mismo nivel que la línea anterior.

Modelo de implementación

Modelo de implementación es una descripción de cómo los elementos del modelo del diseño, como las clases, se implementan en términos de componentes como ficheros de código fuente, ejecutables, etc. (53), posibilitando que los desarrolladores entiendan claramente el funcionamiento del sistema antes de comenzar a escribir las líneas de código.

Diagrama de componentes

Los diagramas de componentes muestran tanto los componentes software (código fuente, binario y ejecutable) como las relaciones lógicas entre ellos en un sistema. Representan todos los tipos de elementos del software implicados en la fabricación de aplicaciones informáticas.

El siguiente diagrama es la representación del sistema de recomendación dividido según la arquitectura basada en el patrón MVC. En el paquete Controlador se muestran los componentes actions. En el paquete Vista se representan los paquetes de componentes de las páginas HTML y las clases JavaScript y en el paquete Modelo se representa los paquetes de componentes de las tablas que componen el sistema. Los diagramas de componentes de cada uno de los paquetes se pueden consultar en el Anexo 3 Diagramas de componentes.

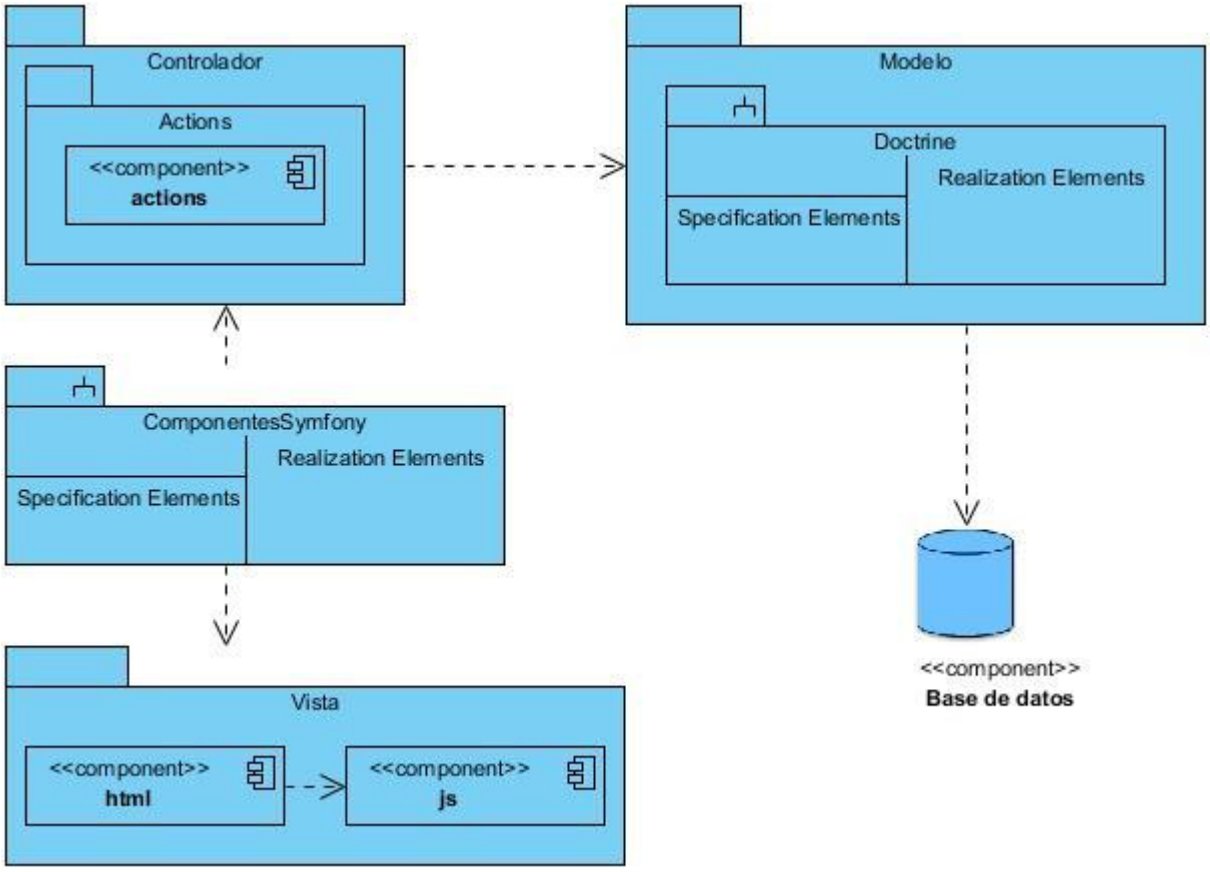


Fig. 9 Diagrama de componentes

Diagrama de despliegue

El diagrama de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Es una colección de nodos; donde cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo de hardware similar. (46)

A continuación se muestran las dos posibles formas de desplegar el software y cómo estarán distribuidos los nodos y dispositivos para la explotación del mismo.

Variante 1:

Se necesita de un ordenador que funcione como servidor de aplicaciones, el cual tendrá instalado el servidor web Apache, un servidor de base de datos PostgreSQL donde se guardará la información generada por la interacción de los usuarios con los productos de El Navegante. Además, se necesita de una computadora que haga función de cliente y un dispositivo de salida, en este caso una impresora, si se accede a alguna de las opciones que permitan este servicio.

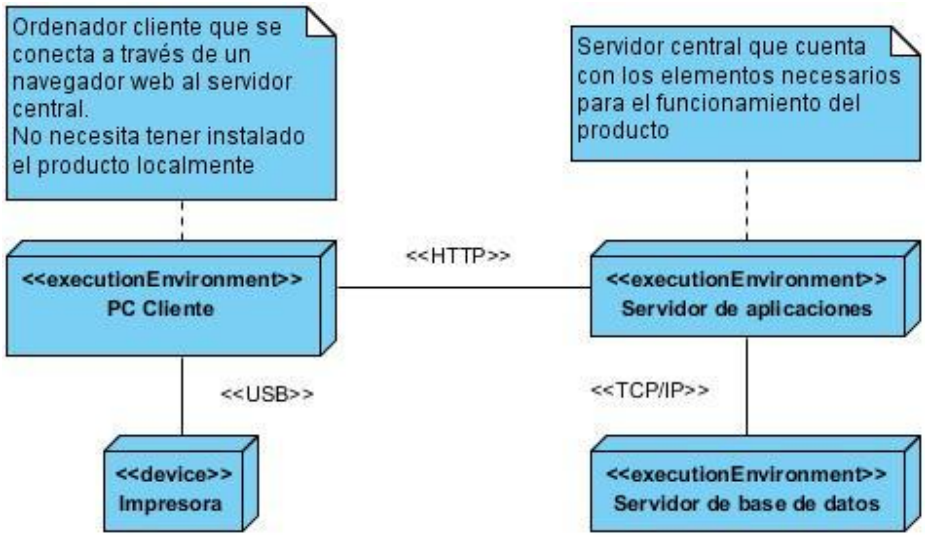


Fig. 10 Diagrama de despliegue variante 1

Variante 2:

Instalar el servidor web Apache, servidor de base de datos PostgreSQL y la aplicación en cada una de las computadoras que hacen la función de cliente. Además un dispositivo de salida, en este caso una impresora, si se accede a alguna de las opciones que permitan este servicio.

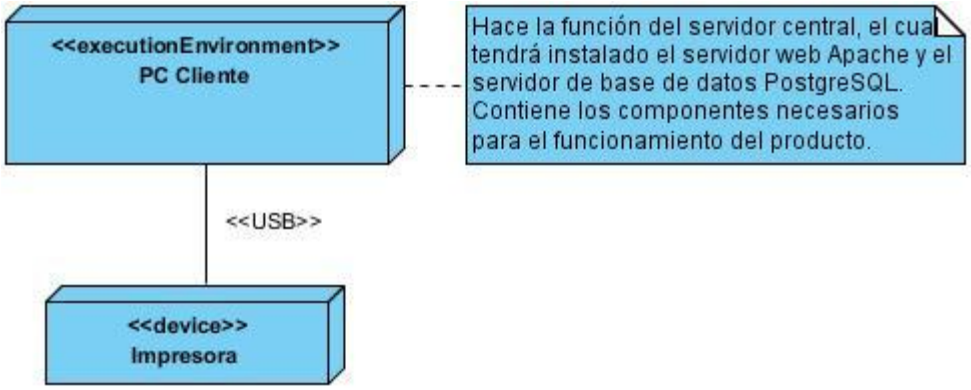


Fig. 11 Diagrama de despliegue variante 2

Pruebas

Las pruebas son procesos que se realizan con el fin de emitir una evaluación de un producto desde un punto de vista crítico, sometiéndolo a una serie de acciones que verifiquen si responde con los requerimientos definidos, determinando así la calidad del mismo. Los objetivos de las pruebas son:

- Planificación de pruebas necesarias para comprobar cada versión operacional del sistema (pruebas de integración y de sistema).
- Diseño e implementación de prueba, creando los casos de prueba para especificar qué probar y los procedimientos de prueba, estableciendo componentes de prueba para agilizar las pruebas.
- Realización de las pruebas y análisis de los resultados de las mismas con el propósito de determinar los defectos del sistema y poder corregirlos.

Para determinar si un sistema cumple con las expectativas del usuario final se define una etapa de pruebas, en las que se determinan los errores que presenta el software relacionados con los requisitos funcionales. Partiendo de lo planteado se decide realizar pruebas de caja negra para comprobar que el sistema de recomendación cumple con los requisitos especificados.

Pruebas de caja negra

Las pruebas de caja negra son las que le permiten al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra intenta encontrar errores de las siguientes categorías: funciones incorrectas o

ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación. (54)

Para la realización de las pruebas existen varias técnicas:

- Técnica de la partición de equivalencia: Divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- Técnica del análisis de valores límites: Prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- Técnica de grafos de causa-efecto: Permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

Considerando que el sistema de recomendación no requiere de muchos datos de entrada la técnica que se propone para realizar los casos de pruebas es la variante de particiones equivalentes, esta técnica consta de 2 pasos fundamentales:

1. Identificación de las clases de equivalencia, es decir, los conjunto de estados válidos o no válidos para condiciones de entrada.
2. Identificar los casos de pruebas.

En el primer paso, las clases de equivalencia son identificadas tomando cada condición de entrada (generalmente una oración o una frase en la especificación) y repartiéndola en dos o más grupos, cada combinación será un escenario del caso de uso. En el segundo paso se identifican las variables y las clases de equivalencia para la confección de los casos de prueba.

Los casos de prueba son un conjunto de entradas de pruebas, condiciones de ejecuciones y resultados esperados desarrollados para cumplir un objetivo en particular o una función esperada. Para la elaboración de los casos de pruebas es necesario un número de datos que ayuden a la ejecución de los mismos y que permitan que el sistema se ejecute en todas sus variantes (55). Los casos de prueba se pueden consultar en el Anexo 4 Diseño de casos de prueba.

Resultados de las pruebas

Para evaluar la solución desarrollada se planificaron 3 iteraciones de pruebas en las cuales se probó la aplicación con un alto grado de detalle. Se abarcaron los métodos y técnicas de pruebas expuestas

anteriormente en cada una de las iteraciones, arrojando resultados visibles, los cuales demuestran la calidad del producto construido. A continuación se muestra la cantidad de no conformidades identificadas en las iteraciones realizadas:

Tabla 3. Cantidad de no conformidades por iteración

Clasificación NC	1ra iteración	2da iteración	3ra iteración
Alta	2	1	0
Media	1	1	0
Baja	2	1	1

De manera más detallada se muestra en un gráfico cómo se comporta la cantidad de no conformidades encontradas en cada una de las iteraciones realizadas:

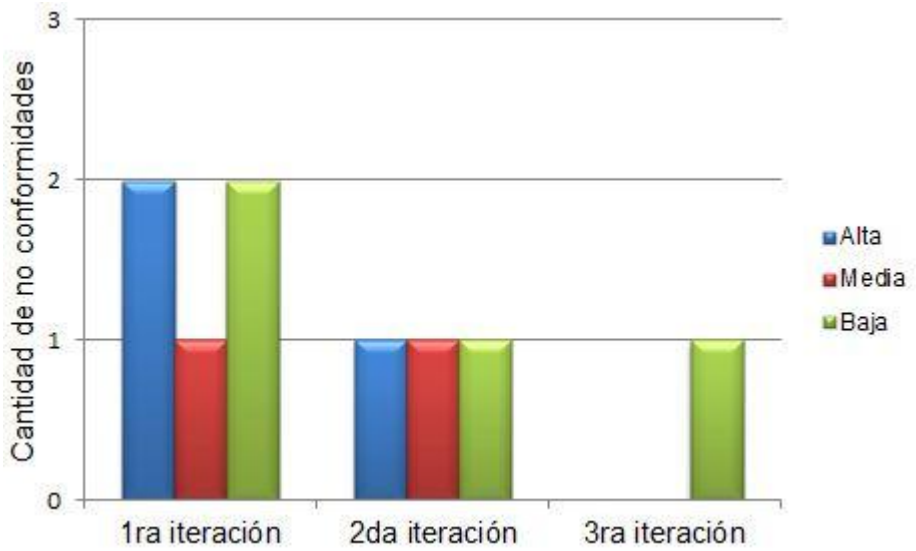


Fig. 12 Cantidad de no conformidades detectadas en cada iteración

Las no conformidades encontradas luego de concluida cada iteración de pruebas se analizaron por parte del equipo de desarrollo, determinando cuáles realmente constituyeron defectos del sistema o presentaban alguna variación según la descripción de los casos de uso y necesitaban ser modificadas. Las pruebas se realizaron de forma iterativa e incremental, comprobando en cada iteración que hubiesen

sido corregidos los errores detectados en la iteración anterior, lo que contribuyó a mejorar la calidad y funcionalidad del software.

Conclusiones

En este capítulo fue representado el sistema a partir de sus componentes de implementación. Se mostró la distribución física en el que será implantado el mismo para su explotación. Finalmente tras un análisis de los resultados obtenidos en las pruebas, se demostró la solidez de la solución, evidenciado en las 3 iteraciones de pruebas realizadas, las cuales denotaron un número decreciente de no conformidades encontradas.

Conclusiones

La concepción y desarrollo de un sistema de recomendación para el módulo Ejercicios de la colección El Navegante posibilitó arribar a las siguientes conclusiones:

- Los sistemas existentes no responden a las necesidades de los estudiantes de enseñanza secundaria.
- Mediante el análisis y diseño del sistema se hizo posible obtener la documentación necesaria para llevar a cabo las actividades de implementación.
- La realización de las pruebas y el análisis de los resultados de las mismas permitieron determinar y erradicar las deficiencias encontradas, garantizando la solidez del sistema.
- El desarrollo del recomendador de cuestionarios interactivos permite la atención diferenciada de los estudiantes de la enseñanza secundaria básica a través de la colección El Navegante.

Recomendaciones

A partir del trabajo realizado se recomienda al proyecto Multisaber-Navegante:

- Incorporar nuevas funcionalidades en el módulo Ejercicios que permitan mostrar al estudiante elementos relacionados con la recomendación.
- Permitir que el profesor desde el módulo Maestro pueda visualizar de cada uno de sus estudiantes elementos relacionados con la recomendación de cuestionarios interactivos.

Referencias bibliográficas

1. **Juan Bautista.** El blog de JUANBAUTISTA. *El blog de JUANBAUTISTA.* [En línea] 20 de Noviembre de 2007. [Citado el: 26 de Enero de 2012.] http://comunidadesvirtuales.obolog.com/tic-conceptualizacion-caracterizacion-tecnologias-informacion-40188.

2. **Araujo Pérez , Liana Isabel.** *Plan de Desarrollo de Software v2.0.* La Habana : s.n., 2007.

3. Definición.de. *Definición.de.* [En línea] [Citado el: 15 de Septiembre de 2011.] http://definicion.de/aprendizaje/.

4. **Rafael Ángel Pérez.** psicoPedagogia.com Psicologia de la educacion para padres y profesionales. *psicoPedagogia.com Psicologia de la educacion para padres y profesionales.* [En línea] [Citado el: 15 de Enero de 2011.] http://www.psicopedagogia.com/.

5. **Isabel García.** psicoPedagogia.com Psicologia de la educacion para padres y profesionales. *psicoPedagogia.com Psicologia de la educacion para padres y profesionales.* [En línea] [Citado el: 15 de Enero de 2012.] http://www.psicopedagogia.com/.

6. **Ramón, Jesús Murillo.** *Un entorno interactivo de aprendizaje con Cabri-actividades, aplicado a la enseñanza de la geometría en la E.S.O.* Barcelona : s.n., 2000.

7. **Pedagógicas, Colectivo de autores de la Dirección Nacional de Secundaria Básica y el Instituto Central de Ciencias.** *Modelo de la escuela Secundaria Básica.* s.l. : Molinos Trade S.A., 2007.

8. **Márquez, Dra Aleida.** *Habilidades.*

9. **Adomavicius, G y Tuzhilin, A.** *Recommendation Technologies: Survey of current methods and possible extensions.* Minessota : MISRC working paper 0329, 2003.

10. **Yera Toledo, Raciel.** *Concepción y desarrollo de un sistema de recomendación para jurados online de programación.* La Habana : s.n., 2010.

11. **Ricci, Francesco, y otros, y otros.** *Recommender Systems Handbook.* New York : Springer New York Dordrecht Heidelberg London, 2011.

12. **Gerling, Valeria Bibiana.** *Sistema Inteligente para Asistir la Búsqueda Personalizada de Objetos de Aprendizaje.* Facultad de Ciencias Exactas, Ingeniería y Agrimensura, Universidad Nacional de Rosario. 2009.

13. **UMU, Facultad de Derecho.** Informática Aplicada a la Gestión Pública. [En línea] [Citado el: 12 de Enero de 2012.] http://www.um.es/docencia/barzana/IAGP/IAGP2-Metodologias-de-desarrollo.html.

14. **Jacobson, Ivar, Rumbaugh, James y Booch, Grady.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. págs. 4-5.

15. **de la Torre, Anibal.** PHPNuke. *PHPNuke.* [En línea] 2006. [Citado el: 12 de Enero de 2012.] http://www.adelat.org/media/docum/nuke_publico/lenguajes_del_lado_servidor_o_cliente.html.

16. W3C World Wide Web Consortium. *W3C World Wide Web Consortium.* [En línea] 7 de Febrero de 2008. [Citado el: 12 de Enero de 2012.] http://www.w3c.es/divulgacion/guiasbreves/XHTML.

17. **Camy, Lázaro Issi.** *JavaScript.* Madrid : ANAYA MULTIMEDIA (GRUPO ANAYA, S.A.), 2002. pág. 69.

18. **ALANXR.** dgtallika. *dgtallika.* [En línea] 22 de Marzo de 2011. [Citado el: 12 de Enero de 2012.] http://www.dgtallika.com/2011/03/css-definicion-de-hoy/.

19. **Lemus, Juan Manuel.** maestros del web. *maestros del web.* [En línea] 18 de Diciembre de 2007. [Citado el: 2 de Febrero de 2012.] http://www.maestrosdelweb.com/editorial/css-3-mas-social-que-nunca/.

20. **Eguíluz Pérez, Javier.** *Introducción a CSS.* 2008.

21. **Valdés, Damián Pérez.** maestrosdelweb. [En línea] 2 de Noviembre de 2007. [Citado el: 12 de Enero de 2012.] http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/.

22. CAVSI Computer Audio Video Systems Integrator. *CAVSI Computer Audio Video Systems Integrator.* [En línea] [Citado el: 13 de Enero de 2012.] http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.

23. **Lavin Cuello, Alejandro y Reyes Pupo, Hector Luis.** *Desarrollo de un módulo para gestionar los cuestionarios interactivos en la Colección El Navegante en su versión multiplataforma.* La Habana : s.n., 2010.

24. **Aguilar, Vicente y Suau, Pablo.** bisente. *bisente.* [En línea] 18 de Agosto de 2000. [Citado el: 13 de Enero de 2012.] http://www.bisente.com/documentos/mysql-postgres.html.

25. SUMMARG. *SUMMARG.* [En línea] 4 de Julio de 2009. [Citado el: 6 de Febrero de 2012.] http://www.summarg.com/foro/threads/4838-Nuevo-PostgreSQL-8-4.

26. **disalzar.** Tecnoweb2.com. *Tecnoweb2.com.* [En línea] 5 de Septiembre de 2011. [Citado el: 15 de Enero de 2012.] http://tecnoweb2.com/que-es-servidor-web.

27. Apache. *Apache.* [En línea] [Citado el: 6 de Febrero de 2012.] http://httpd.apache.org/docs/2.0/es/new_features_2_0.html.

28. **CIVIL, DEPARTAMENTO DE INGENIERÍA.** *Examen Final de Programación Avanzada Tercer Curso.* Burgos : s.n., 2008.

29. Infociberland. *Infociberland.* [En línea] [Citado el: 12 de Enero de 2012.] http://infociberland.comxa.com/junit/.

30. **Sánchez, Zoraida Hidalgo.** *Diseño e implementacion de una BD de investigacion.* Barcelona : s.n., 2007. pág. 86.

31. jQuery write less, do more. *jQuery write less, do more.* [En línea] 2010. [Citado el: 15 de Enero de 2012.] http://jquery.com/.

32. **Zaninotto, François y Potencier, Fabien.** *Symfony 1.2, la guía definitiva.* s.l. : Apress, 2008.

33. **Potencier, Fabien y Zaninotto, François.** *Symfony 1.2, la guía definitiva.* 2008.

34. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Lenguaje Unificado de Modelado. Manual de Referencias.* California : Addison Wesley, 1998. pág. 3.

35. **Daniel M. Maldonado.** El CoDiGo K. *El CoDiGo K.* [En línea] 3 de Septiembre de 2007. [Citado el: 6 de Febrero de 2012.] http://elcodigok.blogspot.com/2007/09/que-son-los-ide-de-programacin.html.

36. **Maldonado, Daniel M.** El CoDiGo K. *El CoDiGo K.* [En línea] 30 de Septiembre de 2010. [Citado el: 6 de Febrero de 2012.] http://www.elcodigok.com.ar/2010/09/7-caracteristicas-de-netbeans-6-9-1-integrado-a-php/.

37. **López Pecho, Ramiro y Ballesteros, Julio César.** HERRAMIENTAS CASE. *HERRAMIENTAS CASE.* [En línea] 29 de Septiembre de 2008. [Citado el: 13 de Enero de 2012.] http://tpsis324.blogspot.com/.

38. Free Download Manager. *Free Download Manager.* [En línea] 5 de Marzo de 2007. [Citado el: 13 de Enero de 2012.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

39. ComuSoft. *ComuSoft.* [En línea] 13 de Noviembre de 2010. [Citado el: 13 de Enero de 2012.] http://www.comusoft.com/modelo-vista-controlador-definicion-y-caracteristicas.

40. **Agrawal, Rakesh, Imielinsk, Tomasz y Swami, Arun.** *Mining association rules between sets of items in large databases.* Washington D.C : Peter Buneman and Sushil Jajodia. págs. 207-216.

41. **Lucas, Joel Pinho.** *Métodos de clasificación basados en asociación aplicados a sistemas de recomendación.* Salamanca : s.n., 2010. págs. 30-32.

42. **Jacobson, Ivar, Rumbaugh, James y Booch, Grady.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. págs. 112-115.

43. —. *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. págs. 106-112.

44. —. *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. pág. 164.

45. —. *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. pág. 166.

46. —. *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. pág. 217.

47. **Larman, Graig.** *UML y patrones. Introducción al análisis y diseño orientado a objetos.* México : Prentice Hall Hispanoamericana, S.A. pág. 167.

48. **Jacobson, Ivar, Rumbaugh, James y Booch, Grady.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. págs. 205-208.

49. **Larman, Craig.** *UML y patrones. Introducción al análisis y diseño orientado a objetos.* México : Prentice Hall Hispanoamericana, S.A. pág. 189.

50. **Gamma, Erich, y otros, y otros.** *Design Patterns. Elements of Reusable Object-Oriented Software.* s.l. : Pearson Education, 2003. pág. 12.

51. —. *Design Patterns. Elements of Reusable Object-Oriented Software.* s.l. : Pearson Education, 2003. pág. 289.

52. —. *Design Patterns. Elements of Reusable Object-Oriented Software.* s.l. : Pearson Education, 2003. pág. 305.

53. **Jacobson, Ivar, Rumbaugh, James y Booch, Grady.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. pág. 257.

54. **Pressman, Roger S.** *Ingeniería de software, un enfoque práctico.* quinta. Madrid y Carchelejo : s.n., 2001. pág. 294.

55. **Murillo Díaz, Claudia y Martínez García, Jorge.** *Desarrollo de la versión 2.0 del módulo Resultados de la colección de software El Navegante en su versión multiplataforma.* La Habana : s.n., 2011.

Bibliografía

1. **Juan Bautista.** El blog de JUANBAUTISTA. *El blog de JUANBAUTISTA.* [En línea] 20 de Noviembre de 2007. [Citado el: 26 de Enero de 2012.] http://comunidadesvirtuales.obolog.com/tic-conceptualizacion-caracterizacion-tecnologias-informacion-40188.

2. **Araujo Pérez , Liana Isabel.** *Plan de Desarrollo de Software v2.0.* La Habana : s.n., 2007.

3. Definición.de. *Definición.de.* [En línea] [Citado el: 15 de Septiembre de 2011.] http://definicion.de/aprendizaje/.

4. **Rafael Ángel Pérez.** psicoPedagogia.com Psicologia de la educacion para padres y profesionales. *psicoPedagogia.com Psicologia de la educacion para padres y profesionales.* [En línea] [Citado el: 15 de Enero de 2011.] http://www.psicopedagogia.com/.

5. **Isabel García.** psicoPedagogia.com Psicologia de la educacion para padres y profesionales. *psicoPedagogia.com Psicologia de la educacion para padres y profesionales.* [En línea] [Citado el: 15 de Enero de 2012.] http://www.psicopedagogia.com/.

6. **Ramón, Jesús Murillo.** *Un entorno interactivo de aprendizaje con Cabri-actividades, aplicado a la enseñanza de la geometría en la E.S.O.* Barcelona : s.n., 2000.

7. **Pedagógicas, Colectivo de autores de la Dirección Nacional de Secundaria Básica y el Instituto Central de Ciencias.** *Modelo de la escuela Secundaria Básica.* s.l. : Molinos Trade S.A., 2007.

8. **Márquez, Dra Aleida.** *Habilidades.*

9. **Adomavicius, G y Tuzhilin, A.** *Recommendation Technologies: Survey of current methods and possible extensions.* Minessota : MISRC working paper 0329, 2003.

10. **Yera Toledo, Raciel.** *Concepción y desarrollo de un sistema de recomendación para jurados online de programación.* La Habana : s.n., 2010.

11. **Ricci, Francesco, y otros, y otros.** *Recommender Systems Handbook.* New York : Springer New York Dordrecht Heidelberg London, 2011.

12. **Gerling, Valeria Bibiana.** *Sistema Inteligente para Asistir la Búsqueda Personalizada de Objetos de Aprendizaje.* Facultad de Ciencias Exactas, Ingeniería y Agrimensura, Universidad Nacional de Rosario. 2009.

13. **UMU, Facultad de Derecho.** Informática Aplicada a la Gestión Pública. [En línea] [Citado el: 12 de Enero de 2012.] http://www.um.es/docencia/barzana/IAGP/IAGP2-Metodologias-de-desarrollo.html.

14. **Jacobson, Ivar, Rumbaugh, James y Booch, Grady.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. págs. 4-5.

15. **de la Torre, Anibal.** PHPNuke. *PHPNuke.* [En línea] 2006. [Citado el: 12 de Enero de 2012.] http://www.adelat.org/media/docum/nuke_publico/lenguajes_del_lado_servidor_o_cliente.html.

16. W3C World Wide Web Consortium. *W3C World Wide Web Consortium.* [En línea] 7 de Febrero de 2008. [Citado el: 12 de Enero de 2012.] http://www.w3c.es/divulgacion/guiasbreves/XHTML.

17. **Camy, Lázaro Issi.** *JavaScript.* Madrid : ANAYA MULTIMEDIA (GRUPO ANAYA, S.A.), 2002. pág. 69.

18. **ALANXR.** dgtallika. *dgtallika.* [En línea] 22 de Marzo de 2011. [Citado el: 12 de Enero de 2012.] http://www.dgtallika.com/2011/03/css-definicion-de-hoy/.

19. **Lemus, Juan Manuel.** maestros del web. *maestros del web.* [En línea] 18 de Diciembre de 2007. [Citado el: 2 de Febrero de 2012.] http://www.maestrosdelweb.com/editorial/css-3-mas-social-que-nunca/.

20. **Eguíluz Pérez, Javier.** *Introducción a CSS.* 2008.

21. **Valdés, Damián Pérez.** maestrosdelweb. [En línea] 2 de Noviembre de 2007. [Citado el: 12 de Enero de 2012.] http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/.

22. CAVSI Computer Audio Video Systems Integrator. *CAVSI Computer Audio Video Systems Integrator.* [En línea] [Citado el: 13 de Enero de 2012.] http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.

23. **Lavin Cuello, Alejandro y Reyes Pupo, Hector Luis.** *Desarrollo de un módulo para gestionar los cuestionarios interactivos en la Colección El Navegante en su versión multiplataforma.* La Habana : s.n., 2010.

24. **Aguilar, Vicente y Suau, Pablo.** bisente. *bisente.* [En línea] 18 de Agosto de 2000. [Citado el: 13 de Enero de 2012.] http://www.bisente.com/documentos/mysql-postgres.html.

25. SUMMARG. *SUMMARG.* [En línea] 4 de Julio de 2009. [Citado el: 6 de Febrero de 2012.] http://www.summarg.com/foro/threads/4838-Nuevo-PostgreSQL-8-4.

26. **disalzar.** Tecnoweb2.com. *Tecnoweb2.com.* [En línea] 5 de Septiembre de 2011. [Citado el: 15 de Enero de 2012.] http://tecnoweb2.com/que-es-servidor-web.

27. Apache. *Apache.* [En línea] [Citado el: 6 de Febrero de 2012.] http://httpd.apache.org/docs/2.0/es/new_features_2_0.html.

28. **CIVIL, DEPARTAMENTO DE INGENIERÍA.** *Examen Final de Programación Avanzada Tercer Curso.* Burgos : s.n., 2008.

29. Infociberland. *Infociberland.* [En línea] [Citado el: 12 de Enero de 2012.] http://infociberland.comxa.com/junit/.

30. **Sánchez, Zoraida Hidalgo.** *Diseño e implementacion de una BD de investigacion.* Barcelona : s.n., 2007. pág. 86.

31. jQuery write less, do more. *jQuery write less, do more.* [En línea] 2010. [Citado el: 15 de Enero de 2012.] http://jquery.com/.

32. **Zaninotto, François y Potencier, Fabien.** *Symfony 1.2, la guía definitiva.* s.l. : Apress, 2008.

33. **Potencier, Fabien y Zaninotto, François.** *Symfony 1.2, la guía definitiva.* 2008.

34. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Lenguaje Unificado de Modelado. Manual de Referencias.* California : Addison Wesley, 1998. pág. 3.

35. **Daniel M. Maldonado.** El CoDiGo K. *El CoDiGo K.* [En línea] 3 de Septiembre de 2007. [Citado el: 6 de Febrero de 2012.] http://elcodigok.blogspot.com/2007/09/que-son-los-ide-de-programacin.html.

36. **Maldonado, Daniel M.** El CoDiGo K. *El CoDiGo K.* [En línea] 30 de Septiembre de 2010. [Citado el: 6 de Febrero de 2012.] http://www.elcodigok.com.ar/2010/09/7-caracteristicas-de-netbeans-6-9-1-integrado-a-php/.

37. **López Pecho, Ramiro y Ballesteros, Julio César.** HERRAMIENTAS CASE. *HERRAMIENTAS CASE.* [En línea] 29 de Septiembre de 2008. [Citado el: 13 de Enero de 2012.] http://tpsis324.blogspot.com/.

38. Free Download Manager. *Free Download Manager.* [En línea] 5 de Marzo de 2007. [Citado el: 13 de Enero de 2012.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

39. ComuSoft. *ComuSoft.* [En línea] 13 de Noviembre de 2010. [Citado el: 13 de Enero de 2012.] http://www.comusoft.com/modelo-vista-controlador-definicion-y-caracteristicas.

40. **Agrawal, Rakesh, Imielinsk, Tomasz y Swami, Arun.** *Mining association rules between sets of items in large databases.* Washington D.C : Peter Buneman and Sushil Jajodia. págs. 207-216.

41. **Lucas, Joel Pinho.** *Métodos de clasificación basados en asociación aplicados a sistemas de recomendación.* Salamanca : s.n., 2010. págs. 30-32.

42. **Jacobson, Ivar, Rumbaugh, James y Booch, Grady.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. págs. 112-115.

43. —. *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. págs. 106-112.

44. —. *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. pág. 164.

45. —. *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. pág. 166.

46. —. *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. pág. 217.

47. **Larman, Graig.** *UML y patrones. Introducción al análisis y diseño orientado a objetos.* México : Prentice Hall Hispanoamericana, S.A. pág. 167.

48. **Jacobson, Ivar, Rumbaugh, James y Booch, Grady.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. págs. 205-208.

49. **Larman, Craig.** *UML y patrones. Introducción al análisis y diseño orientado a objetos.* México : Prentice Hall Hispanoamericana, S.A. pág. 189.

50. **Gamma, Erich, y otros, y otros.** *Design Patterns. Elements of Reusable Object-Oriented Software.* s.l. : Pearson Education, 2003. pág. 12.

51. —. *Design Patterns. Elements of Reusable Object-Oriented Software.* s.l. : Pearson Education, 2003. pág. 289.

52. —. *Design Patterns. Elements of Reusable Object-Oriented Software.* s.l. : Pearson Education, 2003. pág. 305.

53. **Jacobson, Ivar, Rumbaugh, James y Booch, Grady.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 1999. pág. 257.

54. **Pressman, Roger S.** *Ingeniería de software, un enfoque práctico.* quinta. Madrid y Carchelejo : s.n., 2001. pág. 294.

55. **Murillo Díaz, Claudia y Martínez García, Jorge.** *Desarrollo de la versión 2.0 del módulo Resultados de la colección de software El Navegante en su versión multiplataforma.* La Habana : s.n., 2011.

56. **Pedagógica, Colectivo de Autores de Formación.** *Cuarto Encuentro.* 2011.

57. **Herrera Rodríguez, José Ignacio y Román Cao, Eldis.** *Enseñar y aprender en la Sociedad del Conocimiento: el trabajo independiente y la labor del tutor una alternativa para su concreción.* Sancti Spiritus : s.n.

58. **Freeman, Eric y Robson, Elisabeth.** *Head First Design Patterns.* s.l. : O'Reilly Media, 2004. ISBN 978-0596007126.