



Universidad de las Ciencias Informáticas
Facultad 4

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA

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

Autores:

Anabel Santana Alfonso
Yandry Pozo Castillo

Tutores:

Ing. Nadiela Milán Cristo
Ing. Yandris Mata Cabrera

La Habana, Mayo 2013
“Año 55 de la Revolución”

Declaración de autoría

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) a que haga el uso que estimen pertinente con el mismo.

Para que así conste firmamos la presente a los ___ días del mes de _____ de 2013.

Anabel Santana Alfonso

Yandry Pozo Castillo

Firma del autor

Firma del autor

Ing. Nadiela Milán Cristo

Ing. Yandris Mata Cabrera

Firma de la tutora

Firma del tutor

*La ingeniería de software ha adoptado como su estatuto
"Cómo programar si usted no puede".*

Edger W. Dijkstra

"El conocimiento es la mejor inversión que se puede hacer."

Abraham Lincoln

De Anabel:

A mis padres por darme la vida, en especial a mi mamá, que me fue de ejemplo para lograr este sueño.

A mi tía por quererme como una hija y estar a mi lado en todo momento, por enseñarme desde pequeña a luchar por alcanzar mis metas. Mi triunfo es tu triunfo.

A mi tío, por su confianza y amor.

A mi hermana Dania por su ayuda incondicional y dedicación, por motivarme a seguir adelante.

A mi hermano Erick, por siempre porfiarme para ver cuánto sé. Por enseñarme a superarme.

A mis sobrinos Leonel y Daniel por hacer mi vida más feliz.

A mi novio Geonnelly por su amor, comprensión y cariño; sentimientos que día a día trato de retribuirle con la misma intensidad con la que él me los brinda a mí. Por ser mi mejor amigo y mi apoyo incondicional.

A mis suegros Nuvia y Eliseo, por acogerme en su casa como una hija más, por preocuparse y por cuidarme, por brindarme tanto amor y cariño. Gracias a los dos.

A mi abuelo Mariano por la inspiración de un nivel superior. Por lo orgulloso que estaría hoy, pero la vida no le dio la oportunidad de verme graduada, siempre lo recordaré y amaré.

Agradecimientos

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA

A mi familia, por siempre creer en mí.

A mis tutores por su apoyo y confianza.

A mi compañero de tesis que a pesar de los dolores de cabeza que me dio, resultó ser un gran amigo.

A todos mis amigos y compañeros. A todos, muchas gracias.

De Yandry:

A mis padres, las dos personas que más amo y que todo lo perdonan, y a mis hermanitos mayores.

A mis tutores Nadiela y Yandris, por todo sus consejos y ayuda.

A todos los integrantes del proyecto RHODA, especialmente a los profes que me ayudaron.

A todas mis amistades y compañeros de grupo que me soportaron, especialmente a Amado y Rafael.

A Natacha y Arianna por ayudarme a concretar varios de mis sueños.

A la iniciativa Xtreme de programación competitiva de la facultad 4 (antigua 8) y a la ACM-ICPC, gracias a todos los profes de la misma y a mis compañeros del equipo de programación Ajax (o Ajax) que tanto me enseñaron.

Agradecimientos

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA

Agradecer a todos los genios de este mundo que aportaron para crear lo que hoy llamamos “programación”.

De Yandry:

A mi madre Marta, mi padre Tony y mis hermanos Yannit y Yalfre.

De Anabel:

*A mis padres para que donde quiera que estén tengan otro motivo para
estar orgullosos de su hija.*

A mi tía por ser mi guía, mi fuerza y mi razón de ser.

*A mis abuelos por darme siempre sabios consejos y apoyarme en todo
momento.*

*A mi novio por ser parte de mis sueños y apoyarme siempre en todo lo que
me propongo hacer.*

*Al resto de la familia por su apoyo incondicional, por siempre estar
pendiente de mí, por su sacrificio y darme todo su apoyo. Todo lo que soy
es gracias a ustedes y para ustedes es todo lo que hago.*

*A todos los que de una forma u otra han contribuido con mi formación
como profesional y como persona.*

Resumen

Las Tecnologías de la Información y las Comunicaciones (TIC) desempeñan un papel esencial en el proceso de aprendizaje, principalmente en el e-learning traducido al español como aprendizaje electrónico, que incluye cualquier forma de enseñanza asistida por algún medio electrónico. En la Universidad de las Ciencias Informáticas (UCI) se utilizan sistemas e-learning, como apoyo al proceso docente, entre ellos se encuentra el proyecto Repositorio de Objetos de Aprendizaje (RHODA), plataforma que almacena objetos de aprendizaje (OA), de forma centralizada para su reutilización y brinda la posibilidad de crear, visualizar, almacenar y recuperar recursos educativos. El presente trabajo tiene como objetivo desarrollar un sistema de recomendación de OA para los usuarios que utilizan RHODA. Los sistemas de recomendación son aplicaciones encargadas de realizar sugerencias a los usuarios, basándose en las preferencias de estos o en su historial de acciones, para guiarlos en la toma de decisiones. Para guiar el desarrollo de esta aplicación se utilizó Rational Unified Process (RUP) como metodología de desarrollo de software, PHP, CSS y HTML como lenguajes de programación y Symfony como framework de desarrollo. Se realizó el análisis, diseño e implementación; obteniéndose en cada flujo los artefactos fundamentales que propone la metodología. Este sistema recomienda OA al usuario autenticado que pueden o no ser de su interés, teniendo en cuenta su perfil para un instante de tiempo dado, lo que disminuye la espera de los usuarios en buscar los OA en el repositorio.

Palabras claves: sistema de recomendación, perfil de usuario, objetos de aprendizaje.

Índice

INTRODUCCIÓN.....	1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.....	5
INTRODUCCIÓN	5
1.1 GENERALIDADES DE LOS SISTEMAS DE RECOMENDACIÓN.....	5
1.1.1 El problema de la recomendación	7
1.2 CLASIFICACIÓN DE LOS SISTEMAS DE RECOMENDACIÓN	7
1.2.1 Recomendadores basados en contenido	8
1.2.2 Sistemas de soporte a la recomendación.....	8
1.2.3 Sistemas basados en filtrado colaborativo	9
1.2.4 Sistemas de recomendación basados en datos demográficos	9
1.2.5 Sistemas de recomendación híbridos.....	9
1.3 ANÁLISIS DE SOLUCIONES SIMILARES EXISTENTES	10
1.3.1 Comparación entre estos sistemas	11
1.4 METODOLOGÍA DE DESARROLLO	12
1.5 HERRAMIENTAS Y TECNOLOGÍAS A UTILIZAR	13
1.5.1 Lenguajes de programación	13
1.5.2 Sistema gestor de base de datos	15
1.5.3 Servidor web	15
1.5.4 Framework de desarrollo	16
1.5.5 Entorno integrado de desarrollo.....	17
1.5.6 Herramienta CASE	17
1.5.7 Arquitectura de software	18
CONCLUSIONES	19
CAPÍTULO 2: DISEÑO DE LA SOLUCIÓN PROPUESTA.....	20
INTRODUCCIÓN	20
2.1 CARACTERÍSTICAS DEL SISTEMA RECOMENDADOR	20
2.2 FILTRADO COLABORATIVO Y EL CONTENIDO DEL OA.....	20
2.3 DEFINICIÓN DEL VECINDARIO	21
2.4 DEFINICIÓN DE LA CORRELACIÓN DEL VECINDARIO	21
2.5 DEFINICIÓN DE LAS RECOMENDACIONES.....	23
2.6 DEFINICIÓN DE LOS PESOS DE LA RECOMENDACIÓN	24
2.6.1 Definición de los pesos para la búsqueda ordenada	25
2.7 MODELO DE DOMINIO.....	25
2.8 REQUERIMIENTOS	27
2.8.1 Requisitos funcionales	27
2.8.2 Requisitos no funcionales	27
2.9 MODELO DE CASOS DE USO DEL SISTEMA	27
2.9.1 Actores del sistema	28
2.9.2 Diagrama de casos de uso del sistema.....	28

2.9.3 Descripciones de casos de uso	28
2.10 MODELO DE ANÁLISIS	33
2.10.1 Clases del análisis	33
2.10.2 Diagrama de clases del análisis	34
2.10.3 Diagramas de colaboración	35
2.11 MODELO DE DISEÑO	36
2.11.1 Aplicación de patrones de diseño	36
2.11.2 Arquitectura de la aplicación	38
2.11.3 Diagramas de clases de diseño	39
2.11.4 Diagramas de secuencia del diseño	41
2.11.5 Modelo de datos	42
CONCLUSIONES	44
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA	45
INTRODUCCIÓN	45
3.1 ESTÁNDARES DE CODIFICACIÓN	45
3.2 MODELO DE IMPLEMENTACIÓN	47
3.3 DIAGRAMA DE COMPONENTES	47
3.4 DIAGRAMA DE DESPLIEGUE	48
3.5 PRUEBAS DE SOFTWARE	49
3.6 DISEÑO DE CASOS DE PRUEBA	50
3.7 RESULTADOS DE LAS PRUEBAS	53
CONCLUSIONES	54
CONCLUSIONES GENERALES	55
RECOMENDACIONES	56
REFERENCIAS	57
ANEXOS	60
ANEXO 1 REGISTRO DE DEFECTOS Y DIFICULTADES	60
ANEXO 2 MODELO DE ENTREVISTA	61
ANEXO 3 ESTÁNDARES DE CODIFICACIÓN	62

Índice de figuras

Figura 1: Votación de los objetos de aprendizaje.	21
Figura 2: Definición de las recomendaciones.	24
Figura 3: Modelo de dominio.	26
Figura 4: Diagrama de casos de uso.	28
Figura 5: DCA CU Recomendar objetos de aprendizaje	34
Figura 6: DCA CU Buscar objetos de aprendizaje.....	35
Figura 7: DC CU Recomendar objetos de aprendizaje	36
Figura 8: DC CU Buscar objetos de aprendizaje.	36
Figura 9: DCD CU Recomendar objetos de aprendizaje.	40
Figura 10: DCD CU Buscar objetos de aprendizaje.	40
Figura 11: DSD CU Recomendar objetos de aprendizaje.	41
Figura 12: DSD CU Buscar objetos de aprendizaje.	42
Figura 13: Modelo de datos.	43
Figura 14: Diagrama de componentes.....	48
Figura 15: Diagrama de despliegue.	48
Figura 16: Cantidad de no conformidades detectadas en cada iteración.....	53

Índice de tablas

Tabla 1: Comparación entre los sistemas similares	11
Tabla 2: Consideraciones para calcular el coeficiente de Pearson.....	22
Tabla 3: Actores del sistema	28
Tabla 4: Descripción del CU Recomendar objetos de aprendizaje	29
Tabla 5: Descripción del CU Buscar objetos de aprendizaje	30
Tabla 6: Diseño del caso de prueba para el CU Recomendar objetos de aprendizaje.	50
Tabla 7: Diseño del caso de prueba para el CU Buscar objetos de aprendizaje.....	51
Tabla 8: Cantidad de no conformidades por iteración.....	53

Introducción

En la actualidad los sistemas educativos de todo el mundo se enfrentan al desafío de utilizar las Tecnologías de la Información y las Comunicaciones (TIC), para proveer a los alumnos con las herramientas y conocimientos necesarios que se requieren en el siglo XXI. Con el uso de las tecnologías educativas, se brindan herramientas que contienen las bases de la educación tradicional para así garantizar el aprendizaje, a través de la creación de contenidos y con la utilización de nuevos medios para transmitir el conocimiento. Estos contenidos son flexibles, fáciles de modificar y pueden hacer la experiencia del aprendizaje mucho más rica y dinámica mediante recursos multimedia. Estas nuevas tecnologías se ven reflejadas en sistemas e-learning, traducido al español como aprendizaje electrónico, que incluye cualquier forma de enseñanza asistida por algún medio electrónico. De esta forma se abren vías alternas para la interacción entre profesores y alumnos.

Un ejemplo de la aplicación de las TIC en la educación cubana se refleja en la Universidad de las Ciencias Informáticas (UCI), donde existe un Centro de Tecnologías para la Formación, al que pertenece el Departamento de Producción de Herramientas Educativas, donde se ha desarrollado un Repositorio de Objetos de Aprendizaje (RHODA). Este repositorio es un espacio creado para apoyar a la comunidad universitaria en la gestión de recursos didácticos, utilizando la tecnología de objetos de aprendizaje (OA), además de un lugar para el trabajo metodológico colaborativo, orientado a elevar la calidad de los recursos didácticos, brindando la posibilidad de crear, visualizar, almacenar y recuperar estos objetos de aprendizaje.

Un objeto de aprendizaje es un recurso con una intención formativa, compuesto de uno o varios elementos digitales, descritos con metadatos, que puede ser utilizado y reutilizado dentro de un entorno e-learning. (González, 2011)

Para la recuperación de los objetos de aprendizaje, en RHODA se ha desarrollado un módulo de búsqueda, el cual permite mostrar dichos objetos, a través de un filtrado por los metadatos asociados a ellos o por su contenido. Las recuperaciones se realizan sobre los objetos de aprendizaje almacenados en RHODA, o sobre colecciones de metadatos recolectados de otros sistemas e-learning. Estas búsquedas pueden devolver gran cantidad de resultados, los usuarios en general revisan sólo los primeros y estos pueden que no sean los más adecuados, ya que se realiza considerando las palabras claves de la temática de interés. Los usuarios poseen distintas preferencias, las que deberán ser tomadas

en cuenta a la hora de realizar las búsquedas.

Por otra parte el sistema actual no cuenta con una sección que le brinde al usuario de forma automática un listado con los objetos de aprendizaje que pudieran resultarle de interés de acuerdo a su perfil y al comportamiento de otros usuarios con un perfil similar.

Por todo lo antes planteado, surge como **problema de investigación**: ¿Cómo proponer a los usuarios del Repositorio de Objetos de Aprendizaje RHODA, los objetos de aprendizaje que más se adaptan a su perfil?

Se define como **objeto de estudio**: Proceso de recomendación de objetos de aprendizaje, utilizando las tecnologías de la información y las comunicaciones.

Se concibe como **campo de acción**: Los sistemas de recomendación de objetos de aprendizaje.

Teniendo como **objetivo general**: Desarrollar un sistema de recomendación para proponerle a los usuarios del Repositorio de Objetos de Aprendizaje RHODA, los objetos de aprendizaje que más se adapten a su perfil.

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

- Realizar la concepción del sistema de recomendación e identificar sus principales características.
- Desarrollar el sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.
- Validar funcionalmente el sistema.

Se define como **idea a defender**: El desarrollo de un sistema de recomendación de objetos de aprendizaje en RHODA, permitirá obtener un listado con los objetos de aprendizaje que más se le adapten.

Para la realización de esta investigación se utilizaron métodos teóricos, los que permitieron estudiar las características del objeto de investigación que no son observables directamente. Los métodos empleados fueron: el analítico-sintético, que realiza un análisis de toda la teoría y la documentación, permitiendo la extracción de los elementos fundamentales relacionados con los sistemas recomendadores; otro

método utilizado fue el histórico-lógico, empleado en la caracterización de la evolución histórica de los sistemas de recomendación, como herramientas bases para concebir el sistema actual. Además del inductivo-deductivo que posibilitó determinar las características de los sistemas de recomendación aplicables al objeto de estudio.

El uso de los métodos de investigación empíricos permiten describir el objeto de estudio, representando un nivel de la investigación cuyo contenido procede de la experiencia y es sometido a cierta elaboración racional. Se hace uso del método entrevista, que permite establecer la solución del problema en cuestión; definiéndose los elementos fundamentales relacionados con la recomendación de objetos de aprendizaje, basados en las preferencias y características de los usuarios.

Las **tareas de investigación** definidas para darle respuesta a los objetivos específicos son:

- Elaboración del marco teórico-conceptual referente a los sistemas de recomendación.
- Selección de las herramientas y tecnologías a utilizar.
- Identificación de los principales componentes y características del sistema de recomendación de objetos de aprendizaje.
- Definición de los requerimientos funcionales y no funcionales del sistema de recomendación.
- Análisis y diseño del sistema a desarrollar.
- Implementación del sistema de recomendación.
- Integración del sistema de recomendación con el módulo Búsqueda en el Repositorio de Objetos de Aprendizaje RHODA.
- Realización de pruebas internas para garantizar la funcionalidad del sistema.

El documento está organizado en tres capítulos:

Capítulo 1: Fundamentación teórica: Se hace referencia a los elementos teóricos que constituyen la base de la investigación realizada. Se describen los conceptos esenciales para comprender el problema, así como las tecnologías, herramientas y lenguajes de programación a utilizar durante el desarrollo de la solución.

Capítulo 2: Diseño de la solución propuesta: Se describen las características esenciales del sistema de recomendación para su concepción. Se identifican los requerimientos funcionales y no funcionales, los

casos de uso del sistema con sus respectivas descripciones textuales y su relación con los actores del sistema. Se representan los diagramas de clases del análisis, de interacción y de diseño web, además se tiene en cuenta la estructura de la base de datos con la que se trabaja en esta investigación.

Capítulo 3: Implementación y pruebas: En este capítulo se describe la implementación de las principales funcionalidades definidas para el sistema de recomendación. Se muestran además, los métodos de prueba aplicados al sistema y el diseño de casos de prueba correspondiente a cada caso de uso.

Capítulo 1: Fundamentación teórica

Introducción

En el ámbito educacional existe gran cantidad y diversidad de material que puede contribuir al proceso de enseñanza-aprendizaje. En particular, con el desarrollo de la Web y su utilización masiva, se tiene una amplia gama de posibilidades de acceso a material útil e interesante para ser empleado tanto por un alumno que desea aprender un tema, como por un docente que quiere preparar algún material didáctico. La Web se ha convertido en una herramienta fundamental para la recuperación de este tipo de material, por lo general a través de buscadores, pero no siempre el resultado es el esperado por el usuario. En los últimos años, los sistemas recomendadores han ido surgiendo para ayudar a resolver este tipo de problema, puesto que son capaces de seleccionar, de forma automática y personalizada, el material que mejor se adapte a las preferencias o necesidades de un usuario.

El objetivo de este capítulo es realizar un análisis de los principales conceptos relacionados con los sistemas de recomendación. Se define qué es un sistema de recomendación, así como las clasificaciones de los mismos según varios autores. Además se realiza un estudio del estado del arte, de las herramientas recomendadoras orientadas a los objetos de aprendizaje y de los sistemas de recomendación en sentido general. También se hace una breve descripción de las herramientas, tendencias y tecnologías a utilizar en el desarrollo de la propuesta.

1.1 Generalidades de los sistemas de recomendación

Los sistemas recomendadores son los encargados de sugerir a un determinado usuario cierta información. A continuación se presentan algunas definiciones que describen el funcionamiento de los mismos:

Los sistemas de recomendación utilizan la opinión de los miembros de una comunidad para ayudar a los individuos a identificar la información o los productos más relevantes o interesantes según sus necesidades actuales. (Konstan, 2004)

Los sistemas de recomendación constituyen una técnica de filtrado de información, la cual presenta distintos tipos de temas o ítems de información al usuario, basándose en la predicción del “ranking” o ponderación que este le daría a un ítems que el sistema aún no ha considerado; mediando,

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

automatizando o soportando de esta forma, el proceso de realizar recomendaciones. (Terveen y Hill, 2001)

Los sistemas de recomendación son herramientas cuyo objetivo es asistir a los usuarios en sus procesos de búsqueda de información, ayudando a filtrar los ítems de información recuperados, usando recomendaciones propuestas sobre esos ítems. Dichas recomendaciones se generan a partir de las opiniones proporcionadas por otros usuarios sobre ciertos ítems, tales como documentos, libros e informes en búsquedas previas o bien a partir de las preferencias del usuario objeto de la recomendación, dando lugar a los dos grandes grupos. (Yager, 2003)

Un sistema de recomendación es una tecnología de filtrado de información que permite resolver dos problemas específicos: (i) el problema de predicción, en el cual se predice si a un usuario en particular le gustará un ítem en particular; y (ii) el problema de recomendación, en el cual se determina un conjunto de n ítems a recomendar a un usuario a partir de sus preferencias personales. (Sarwar, 2001)

Este proceso el sistema debe tener la capacidad de hacer predicciones sobre las preferencias del usuario por cierto ítem. El sistema captura información de preferencia referente al usuario, ya sea de forma directa o a través de preguntas indirectas, creando así su perfil. Una vez obtenida dicha información el sistema se encuentra en total capacidad para recomendar un ítem, que él cree que es de interés para el usuario, basándose fundamentalmente en los datos almacenados en su perfil y el perfil de otros usuarios con características o preferencias similares.

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. (Plasencia y Milan, 2012)

De manera general, se puede decir que los datos que utilizan los sistemas recomendadores pueden separarse en ítems, usuarios y transacciones, los que se describen a continuación:

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

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

fue una mala decisión seleccionarlo. Los ítems tienen además, características que pueden indicar cuán 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.

1.1.1 El problema de la recomendación

El principio de funcionamiento de los sistemas de recomendación está resumido en el problema de la recomendación, el cual puede ser formulado de la siguiente forma: (Tuzhilin y Adomavicius, 2003)

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

$$\forall c \in \mathbf{C}, s'_c = \operatorname{argmax}_{s \in \mathbf{S}} u(c, s)$$

1.2 Clasificación de los sistemas de recomendación

Los sistemas de recomendación tienen diferentes clasificaciones. Separándolas atendiendo a la metodología que utilizan para la construcción del perfil de un usuario determinado, que se describen a continuación.

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

1.2.1 Recomendadores basados en contenido

Los recomendadores basados en el contenido son aquellos sistemas que recomiendan los ítems a los usuarios basándose exclusivamente en la descripción del ítem y de un perfil que contenga los intereses del propio usuario. (Billsus y Pazzani, 2007)

Este tipo de recomendador determina cuán relacionado está un objeto con un determinado usuario, para lo cual utiliza la medida de similitud. Esto le permite al sistema recomendar el objeto, partiendo de la descripción del mismo que mejor se adapte a las características de un usuario recogidas en su perfil. Dicho perfil puede ser creado de manera explícita, preguntando directamente al usuario por sus preferencias. 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 y cualquier otra información que resulte útil.

Una de las ventajas más importantes de este tipo de sistema recomendador es que no se basa en las acciones previas del usuario, siempre lo hará basándose en la descripción del objeto y las preferencias del usuario. La principal desventaja es que las recomendaciones serán muy parecidas, ya que tiene como base los mismos datos.

1.2.2 Sistemas de soporte a la recomendación

Estos son sistemas que se dedican a brindar soporte a la actividad de compartir recomendaciones. (Toledo, 2010)

Contiene dos roles fundamentales: el rol de productor de recomendaciones y el rol de consumidor de recomendaciones. El productor está representado generalmente por un pequeño grupo de personas que se encargan de introducir en el sistema, datos, preferencias, entre otros, el sistema se encarga de almacenarlos y luego presentarle los elementos al consumidor de la misma forma en que fueron introducidos. Estos sistemas son más efectivos cuando ocupan el rol de productor suficientes personas dedicadas a la recopilación de información con el fin de crear recomendaciones, para así determinar la eficiencia de las sugerencias, partiendo de la retroalimentación productor-consumidor.

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

1.2.3 Sistemas basados en filtrado colaborativo

Tiene como base identificar personas con similitudes en sus preferencias para así crear las recomendaciones. Con el filtrado colaborativo los usuarios expresan sus intereses a partir de algunos ítems que se les presentan, ayudando al sistema de esta manera a crear un perfil, el sistema compara el perfil de este con los restantes usuarios construyendo una lista de usuarios con similitudes, la cual se puede denominar como “usuarios más cercanos”. Combinando esta lista, se devuelve un listado de recomendaciones, conteniendo los ítems mejor evaluados por estos usuarios y que no han sido aún evaluados por el nuevo individuo. (Toledo, 2010)

Para que este sistema tenga éxito es necesario que sea utilizado por varios usuarios, para que cuando ingrese uno nuevo pueda encontrar grupos de personas con intereses comunes, esto sería una forma sencilla de obtener las preferencias, para que la comparación de los mismos sea más sencilla y así disponer de algoritmos capaces de relacionar a los usuarios.

1.2.4 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. (Ricci, 2011)

1.2.5 Sistemas de recomendación híbridos

Como se ha podido apreciar, las técnicas antes descritas presentan algunas desventajas, por ejemplo para la recomendación colaborativa influye la cantidad de usuarios que deben estar registrados en el sistema, para que el mismo funcione correctamente; la recomendación basada en contenidos le afectará la similitud que existirá en las recomendaciones. Para mitigar estas desventajas se puede combinar la recomendación basada en contenido con la recomendación colaborativa, creando un sistema de recomendación con características híbridas, aprovechando las ventajas de cada uno de estos tipos de sistemas de recomendación.

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

Después de haber realizado un análisis de los distintos tipos de sistemas de recomendación, teniendo en cuenta las características, ventajas y desventajas de los mismos, se ha llegado a la conclusión de que los elementos comunes para estos sistemas de recomendación son: contar con una descripción del objeto que puede ser recomendado, un perfil del usuario que contenga sus preferencias y un sistema que compare cada uno de los objetos a recomendar con el perfil del usuario, para determinar cuáles de ellos serán recomendados.

Se decide utilizar un sistema de recomendación híbrido ya que combina el basado en contenido con el filtrado colaborativo, teniendo la necesidad de que existan una gran cantidad de personas para que accedan al sistema y así incluir a algún nuevo usuario a los grupos de preferencias similares. También se basará en las acciones previas del usuario, fundamentándose en la descripción del objeto y las preferencias del usuario, peculiaridad en la que se basa el filtrado colaborativo y el de contenido.

1.3 Análisis de soluciones similares existentes

Para ayudar en la elaboración de esta investigación, se ha realizado un estudio de sistemas de recomendación similares existentes, para determinar las herramientas y tecnologías a utilizar en la solución propuesta. A continuación se muestran los sistemas analizados:

Sistema inteligente para la recomendación de objetos de aprendizaje

Se describe el desarrollo de un sistema recomendador de objetos de aprendizaje. Este sistema 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.

El sistema tiene una arquitectura multiagente que incluye varios tipos de agentes con diferentes funcionalidades. Se modela el Agente Recomendador (Agente-R), como un agente BDI graduado.

Este agente se encarga de realizar una recuperación flexible y presentar una lista ordenada con los mejores recursos de acuerdo con el perfil del usuario. (Casali, Gerling, Deco y Bender, 2010)

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

Diseño de un sistema de recomendación en repositorios de objetos de aprendizaje basado en la percepción del usuario: Caso RODAS

Es un sistema que se basa en el filtrado colaborativo, que utiliza una adaptación del algoritmo k-vecinos y con fundamento en la percepción de usabilidad y utilidad que el usuario tiene acerca de los objetos de aprendizaje que descargan del repositorio. También muestra la forma como el algoritmo k-vecinos se adaptó al concepto de percepción con la implementación de un sistema de votación de los objetos de aprendizaje por parte de los usuarios. (Hernández y Piñeres, 2011)

1.3.1 Comparación entre estos sistemas

Tabla 1: Comparación entre los sistemas similares

Nombre	Tipo de metodología que utiliza	Algoritmo que utiliza	Definición de la recomendación
Sistema inteligente para la recomendación de OA.	Se basa en el contenido del OA.	Modela el Agente Recomendador (Agente-R), como un agente BDI graduado.	Tiene un sistema de evaluación, que depende de la percepción del usuario.
Diseño de un sistema de recomendación en repositorios de objetos de aprendizaje basado en la percepción del usuario: Caso RODAS.	Se basa en el filtrado colaborativo.	Usa una adaptación del algoritmo k-vecinos.	Tiene un sistema de votación que depende de la percepción del usuario.

Estos sistemas tienen características propias y comparten similitudes con RHODA, las cuales pueden ser aprovechadas para la realización del sistema recomendador, teniendo en cuenta los algoritmos y el tipo de metodología que utilizan (basado en contenido y filtrado colaborativo) tomando de ellas las ventajas para el recomendador con características híbridas. De los sistemas analizados el que más semejanzas tiene con RHODA es “Diseño de un sistema de recomendación en repositorios de objetos de aprendizaje basado en la percepción del usuario: Caso RODAS”, sirviendo de base para la solución propuesta.

1.4 Metodología de desarrollo

La construcción de un sistema de software implica la toma de decisiones sobre la arquitectura del mismo. Estas decisiones pueden ser cruciales para el éxito o fracaso del sistema resultante, por lo que se requiere seleccionar un proceso de desarrollo de software con el fin de obtener la calidad deseada y que cumpla con los requerimientos establecidos. 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 en sí 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.

El sistema a desarrollarse forma parte del proyecto RHODA; la metodología de desarrollo definida en este proyecto 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, pues el uso de otra podría traer algunas inconveniencias en el cronograma definido por el proyecto.

RUP es una metodología tradicional y robusta que está basada en componentes. El sistema de software en construcción está formado por componentes interconectados a través de interfaces bien definidas. RUP utiliza el Lenguaje Unificado de Modelado (UML) para preparar todos los esquemas en un sistema de software. Esta metodología es iterativa e incremental, centrada en la arquitectura y guiada por casos de uso. Cada organización que la utilice, la especializará para ajustarla a su situación, por ejemplo, dado su tipo de aplicación, plataforma, entre otros criterios. (Jacobson, Rumbaugh y Booch, 1999)

Algunas ventajas de 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, 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:

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

- Se reduce el coste del riesgo, a los costes de un solo incremento ya que si los desarrolladores tienen que repetir una iteración, solo 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 RUP no necesariamente se debe mantener un contacto frecuente con los clientes.
 - No recomienda que se lleven a cabo estrictamente todas las actividades y artefactos que se describen, sino por lo contrario, se recomienda que en dependencia de las características del proyecto y de la organización se seleccionen los artefactos, actividades y roles utilizados.

La característica de ser iterativo e incremental permite también determinar si han aparecido nuevos requisitos o han cambiado los existentes, afectando a las iteraciones siguientes. Los casos de uso no sólo inician el proceso de desarrollo sino que proporcionan un hilo conductor, permitiendo establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo. Por su parte la arquitectura permite tener una visión común entre todos los involucrados (desarrolladores y usuarios) y una perspectiva clara del sistema completo, necesaria para controlar el desarrollo.

1.5 Herramientas y tecnologías a utilizar

1.5.1 Lenguajes de programación

Son considerados lenguajes artificiales que pueden ser usados para controlar el comportamiento de una máquina, específicamente, una computadora y que están compuestos por un conjunto de reglas sintácticas y semánticas que permiten expresar instrucciones que luego serán interpretadas. Se utilizan para este sistema, lenguajes y tecnologías del lado del cliente como son HTML y CSS, así como el lenguaje de programación del lado del servidor PHP, que se describe a continuación.

Lenguajes y tecnologías del lado del cliente

HTML 4.0

Es el lenguaje utilizado para la creación de páginas web, significa "Hyper Text Mark-Up Language", y en español, "Lenguaje para el Formato de Documentos de Hipertexto". Los documentos HTML no son documentos de texto normal, sino documentos de hipertexto ya que en el propio documento aparecen enlaces a otros documentos.

El propio World Wide Web Consortium (W3C) define el lenguaje HTML como "un lenguaje reconocido universalmente y que permite publicar información de forma global". Desde su creación, el lenguaje HTML ha pasado de ser un lenguaje utilizado exclusivamente para crear documentos electrónicos a ser un lenguaje que se utiliza en muchas aplicaciones electrónicas como buscadores, tiendas online y banca electrónica. Es un lenguaje utilizado únicamente para dar estructura a una página web. (Rodríguez y Valdés, 2010)

CSS 2.0

Cascading Style Sheets (CSS) es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación, y es imprescindible para crear páginas web complejas. Mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro de la página: párrafo, titular, texto destacado, tabla, lista de elementos, entre otras. Una vez creados los contenidos, se utilizan para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, entre otros elementos. (Pérez, 2008)

Lenguaje de programación del lado del servidor

PHP 5

Hypertext Pre-processor es un lenguaje de "código abierto" interpretado de alto nivel, especialmente pensado para aplicaciones web, el cual puede ser embebido en páginas HTML. Su sintaxis es similar a C/C++, Java y Perl. (Achour, 2007)

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

De manera general, las principales características que distinguen a PHP son: su rapidez, su facilidad de aprendizaje, su soporte multiplataforma, como de servidores HTTP y de bases de datos; y el hecho de que se distribuye de forma gratuita bajo una licencia abierta. Es un lenguaje orientado a objetos y posee variedad de funciones ya implementadas de utilidad para los desarrolladores. (Triana y Fernández, 2010)

1.5.2 Sistema gestor de base de datos

Un Sistema Gestor o Manejador de Bases de Datos (SGBD) es un conjunto de programas que tienen como propósito general, facilitar el proceso de definir, construir y manipular una base de datos para diversas aplicaciones.

El sistema a desarrollarse forma parte del proyecto RHODA; el sistema gestor de bases de datos definido en el proyecto es PostgreSQL. En aras de mantener la estructura definida en la arquitectura técnica del proyecto, se decide utilizar el mismo gestor de bases de datos.

PostgreSQL 9.0

Es un poderoso SGBD de código abierto. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de confiabilidad, integridad de datos y corrección. Funciona en los principales sistemas operativos, incluyendo Linux, UNIX y Windows. Tiene soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados (en varios idiomas). También es compatible con el almacenamiento de objetos binarios, incluyendo imágenes, sonidos o vídeos. Tiene interfaces de programación nativo de C / C + +, Java, NET, Perl, Python, Ruby, Tcl, entre otros. (PostgreSQL, 2010)

1.5.3 Servidor web

Un servidor es un tipo de software que realiza ciertas tareas en nombre de los usuarios. El término servidor, también se utiliza para referirse al ordenador físico en el cual funciona ese software, una máquina cuyo propósito es proveer datos de modo que otras máquinas puedan utilizar esos datos. Para la realización de este sistema se utiliza Apache.

Apache 2.2

Es un servidor web gratuito desarrollado por el Apache Server Project (Proyecto Servidor Apache), cuyo objetivo es la creación de un servidor web fiable, eficiente y fácilmente extensible con código fuente

abierto gratuito. Apache es uno de los servidores más usados en el mundo, multiplataforma y extensible. Las principales ventajas que tiene la utilización de este servidor web son: (Díaz y Vargas, 2002)

- Una talentosa comunidad de desarrolladores siguiendo un proceso abierto de desarrollo.
- Portabilidad: Apache trabaja sobre todas las versiones recientes de UNIX, Linux y Windows, entre otros.
- Es robusto y seguro.
- Posee una arquitectura modular.

1.5.4 Framework de desarrollo

Un framework simplifica el desarrollo de una aplicación, mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. (Potencier y Zaninotto, 2008) Para la realización de este sistema se utiliza Symfony.

Symfony 1.3.8

Es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, 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. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas, es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. (Potencier y Zaninotto, 2008)

Algunas de las características que posee: (Potencier y Zaninotto, 2008)

- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del SGBD.
- Sencillo de usar en la mayoría de los casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

- Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

1.5.5 Entorno integrado de desarrollo

Un Entorno de Desarrollo Integrado o IDE (por sus siglas en inglés), es una herramienta que permite a los desarrolladores de software escribir sus programas en uno o más lenguajes. Consiste básicamente en una plataforma en la que se integran un editor de código, un compilador, un depurador y una interfaz gráfica de usuario. Para la realización de este sistema se utiliza Netbeans.

Netbeans 6.8

Es una herramienta de código abierto diseñada para el desarrollo de aplicaciones fácilmente portables entre las distintas plataformas, haciendo uso de la tecnología Java.

NetBeans IDE dispone de soporte para crear interfaces gráficas de forma visual, así como desarrollo de aplicaciones web, control de versiones, colaboración entre varias personas, creación de aplicaciones compatibles con teléfonos móviles, sus funcionalidades son ampliables mediante la instalación de plugins. (Dorado, 2005)

Otra característica importante de NetBeans es que posee un buen completamiento de código para cada uno de los lenguajes de programación que soporta, como es el caso de PHP, para el cual proporciona completamiento de código, incluyendo los métodos mágicos Get y Set. En el caso de JavaScript, brinda soporte y autocompletado de código para sus principales librerías, entre las que se destacan: JQuery, Symfony, Dojo y Prototype.

1.5.6 Herramienta CASE

CASE es una sigla, que corresponde a las iniciales de: Computer Aided Software Engineering; y en su traducción al español significa Ingeniería de Software Asistida por Computación. Se puede definir a las

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

herramientas CASE como 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. (Rodríguez y Valdés, 2010) Para la realización de este sistema se utiliza Visual Paradigm.

Visual Paradigm 8.0

Herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Soporta todos los diagramas UML y el Diagrama Entidad-Relación, ayudando a una rápida construcción de aplicaciones con calidad y a un menor coste. Posee una interfaz amigable y se puede modelar en varios idiomas. Es una herramienta colaborativa, pues soporta múltiples usuarios trabajando sobre el mismo proyecto. La documentación del proyecto puede ser generada automáticamente en varios formatos (Web o .pdf), y permite el control de versiones.

Algunas de sus ventajas: (Triana y Fernández, 2010)

- Apoya todo lo básico en cuanto a artefactos generados en las etapas de definición de requerimientos y de especificación de componentes.
- Tiene disponibilidad en múltiples plataformas: Microsoft Windows, Linux, Mac OS X, Solaris.
- Brinda la posibilidad de cargar proyectos de aplicaciones como Visio y Rational Rose.
- Brinda la posibilidad de generar código a partir de los diagramas, para plataformas como .Net, Java y PHP, así como obtener diagramas a partir del código.
- Brinda la posibilidad de documentar todo el trabajo sin necesidad de utilizar herramientas externas.

1.5.7 Arquitectura de software

La Arquitectura de Software es la organización fundamental de un sistema representada en sus componentes, las relaciones entre ellos y el ambiente, así como los principios que orientan su diseño y evolución. Para la realización de este sistema se utiliza el Patrón Modelo Vista Controlador.

Patrón Modelo Vista Controlador

Capítulo 1: Fundamentación teórica

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

El patrón Modelo Vista Controlador (MVC) propone separar los módulos de entrada, procesamiento y salida de datos en una aplicación, donde cada uno va a tener sus funcionalidades específicas. Un modelo puede tener diversas vistas, cada una con su correspondiente controlador. (Rodríguez y Valdés, 2010)

El modelo es responsable de:

- Acceder a la capa de almacenamiento de datos.
- Define las reglas de negocio (la funcionalidad del sistema).
- Lleva un registro de las vistas y controladores del sistema.

Las vistas son responsables de:

- Recibir datos del modelo.
- Tienen un registro de su controlador asociado.
- Pueden dar el servicio de actualización, para que sea invocado por el controlador o por el modelo.

El controlador es responsable de:

- Recibir los eventos de entrada (un clic, un cambio en un campo de texto, entre otros).
- Contener reglas de gestión de eventos.

Conclusiones

Luego de realizar un análisis de las aplicaciones similares, ninguna de ellas da respuesta a todos los problemas identificados ni a la necesidad actual, por lo que se decide implementar un sistema de recomendación híbrido (filtrado colaborativo y basado en contenido) que dé solución a las mismas. Se resuelve utilizar como metodología de desarrollo RUP, como lenguajes de programación del lado del cliente HTML, JavaScript y CSS y del lado del servidor PHP, como sistema gestor de base de datos PostgreSQL, como servidor web Apache, como framework de desarrollo Symfony, como entorno integrado de desarrollo Neatbeans, como herramienta CASE Visual Parading y como arquitectura de software Patrón Modelo Vista Controlador.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

Capítulo 2: Diseño de la solución propuesta

Introducción

Existen varios recursos disponibles para organizar la información en Internet, y así ayudar a los usuarios a encontrar los recursos que necesitan o prefieren. Uno de ellos son los repositorios, que son una mezcla entre un buscador y un servidor de descargas, en el caso del sistema recomendador a implementar contendrá objetos de aprendizaje. Poseen la ventaja de que el contenido es producido con fines educativos y tienen una serie de herramientas que permiten acceder a los materiales digitales que son requeridos por los usuarios.

En este capítulo se describe una propuesta de un sistema de recomendación híbrido, teniendo en cuenta el algoritmo de vecindad próxima (k-nearest neighbors); y el contenido del OA. Además se representa el modelo de dominio, donde se capturan los objetos importantes del entorno que serán empleados en el sistema. Se identifican los requerimientos funcionales y los no funcionales, se elabora el diagrama de casos de uso con sus respectivas descripciones, los diagramas de clases del análisis, de colaboración y de secuencia de diseño, así como los patrones del diseño utilizados, se representa también la estructura de la base de datos que se utiliza en dicho sistema.

2.1 Características del sistema recomendador

El sistema que se propone le recomienda objetos de aprendizaje al usuario que puedan interesarle. Una vez que se accede a la aplicación se construye el perfil del usuario que está autenticado en ese momento. Dicho perfil tiene en cuenta el lenguaje preferido por el usuario y el tipo de recurso que desea. Si se selecciona una búsqueda general de los objetos de aprendizaje, a cada uno se le asigna un peso o valor calculado a partir del perfil del usuario y el listado de objetos de aprendizaje es organizado de mayor a menor por este valor. Cada vez que el usuario visualice, descargue o evalúe algún objeto de aprendizaje se registra la acción en el sistema, además de actualizarse el perfil del mismo.

2.2 Filtrado colaborativo y el contenido del OA

Para la implementación del sistema recomendador se optó por el recomendador híbrido que se basa en el algoritmo de vecindad próxima (k-nearest neighbors). (Koren y Bell, 2007) De esta manera se obtiene una

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

medida de preferencia más amplia gracias a otros usuarios con preferencias similares, que son hallados mediante una variación de dicho algoritmo.

Para lograr una mayor precisión del sistema recomendador se tiene en cuenta el contenido del OA, se inicia con la creación del perfil del usuario donde especifican sus preferencias sobre los ítems. Los datos asociados son: el tipo de recurso (ejercicio, descripciones, exámenes, animaciones, entre otros) del contenido del OA y el lenguaje (español, portugués, inglés, entre otros) en el que está escrito el contenido del OA.

El sistema le brinda la posibilidad al usuario de evaluar los OA sobre su funcionalidad al ser usados: se mide con una escala del 1 al 5, como se muestra en la Figura 1.



Figura 1: Votación de los objetos de aprendizaje.

2.3 Definición del vecindario

La definición del vecindario consiste en buscar todos los usuarios que votaron por los mismos objetos de aprendizaje que el usuario autenticado. El vecindario está constituido por todos los usuarios que tienen un nivel de similitud con el autenticado. (Chen y Cheng, 2009)

2.4 Definición de la correlación del vecindario

Se definen los pesos para cada usuario con base en el coeficiente de Correlación de Pearson. (Shakhnarovich, 2005) El coeficiente es un índice que mide el grado de covariación entre distintas variables relacionadas linealmente. Sus valores absolutos oscilan entre 0 y 1. (Rashid, Karypis y Riedl,

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

2006) Dicho coeficiente se usa para determinar el nivel de cercanía o relación entre los usuarios, con respecto de la percepción que éstos tienen de los OA.

En la ecuación (1) el peso $\omega_{\alpha,u}$ que se asigna al usuario u para predecir al usuario actual autenticado α viene dado por: $r_{\alpha,i}$ que corresponde con la votación del usuario α al elemento i .

$$\omega_{\alpha,u} = \frac{\sum_{i=1}^m (r_{\alpha,i} - \bar{r}_{\alpha}) * (r_{u,i} - \bar{r}_u)}{\sigma_{\alpha} \sigma_u} \quad (1)$$

Para calcular el coeficiente de correlación de Pearson, (Bregon, Arancha y Rodríguez, 2005) se tienen en cuenta las siguientes consideraciones (tabla 2):

Tabla 2: Consideraciones para calcular el coeficiente de Pearson

r_{α}	Usuario actual autenticado
$r_{\alpha,i}$	Voto del usuario actual por el objeto de aprendizaje i .
\bar{r}_{α}	Promedio de los votos del usuario actual.
$r_{u,i}$	Voto de un vecino u por el mismo objeto i .
r_u	Un vecino del usuario autenticado.
\bar{r}_u	Promedio de los votos del vecino u .
σ_{α}	Desviación estándar de α (usuario autenticado).
σ_u	Desviación estándar de u (vecino u del usuario autenticado).
m	Número de votos en común entre los usuarios.

La desviación estándar especifica el grado de dispersión de un grupo de datos de su media aritmética. En este caso, determina qué tanto se alejan los votos del usuario sobre los objetos del promedio de votos.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

La tendencia de votación del usuario autenticado se da por la desviación estándar σ_u y la tendencia de votación del vecino actual se define por la desviación estándar σ_v (Ni y Nguyen, 2009). Dado lo anterior, las respectivas fórmulas para el cálculo serían:

$$\sigma_u = \sqrt{\frac{\sum_{i=1}^m (r_{u,i} - \bar{r}_u)^2}{m}} \quad (2)$$

$$\sigma_v = \sqrt{\frac{\sum_{i=1}^m (r_{v,i} - \bar{r}_v)^2}{m}} \quad (3)$$

La fórmula anterior calcula la desviación estándar que es igual a la raíz cuadrada de la varianza, si el usuario tiene más de un voto. Si sólo tiene un voto, la desviación es igual al valor votado, y si no ha realizado voto alguno, la desviación estándar es igual al máximo que puede tener un voto.

Si la desviación estándar de los votos del vecino por la desviación estándar de los votos del usuario autenticado es diferente de cero, se calcula el peso, dividiendo la suma entre el producto de las desviaciones. De lo contrario, el peso del vecino es cero.

2.5 Definición de las recomendaciones

Consiste en encontrar los objetos que hayan sido de interés para los vecinos del usuario autenticado y por los cuales se haya realizado una valoración (votación); además, que el usuario autenticado no haya valorado aún. (Lemire y Maclachlan, 2005) Cabe destacar que si dos o más vecinos han valorado el mismo objeto, este se escoge solo una vez.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

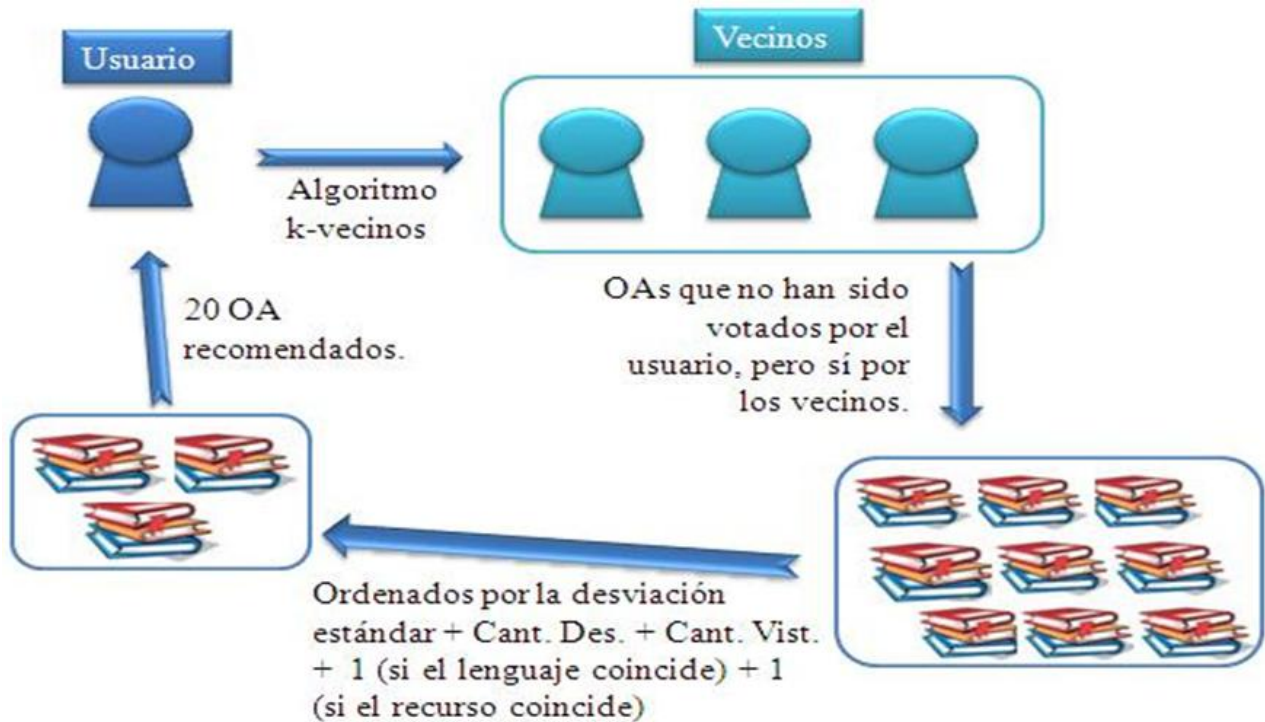


Figura 2: Definición de las recomendaciones.

2.6 Definición de los pesos de la recomendación

Las valoraciones realizadas a un objeto por los vecinos del usuario autenticado y la cantidad de vecinos que hayan valorado dicho objeto, van a definir el peso del objeto o qué tan recomendable es para el usuario autenticado. Estas valoraciones se asocian con su desviación estándar para no sesgarlas, según el nivel de optimismo de las valoraciones de los usuarios. (Koren y Bell, 2007)

$$p_{a,i} = r_a + \sigma_a \frac{\sum_{u=1}^n \frac{(r_{u,i} - \bar{r}_u)}{\sigma_a} \omega_{a,u}}{\sum_{u=1}^n \omega_{a,u}} + \text{SumLang} + \text{SumRes} + \text{CantDow} + \text{CantView}$$

Para calcular el peso de las recomendaciones $p_{a,i}$ se tienen en cuenta las siguientes consideraciones: r_a denota al usuario autenticado, σ_a representa la tendencia de votación del usuario autenticado, $r_{u,i}$ son los

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

votos de un vecino u , por el mismo objeto i , mientras que \bar{r}_u se refiere al promedio de los votos del vecino u , el peso asignado al usuario autenticado viene dado por $\omega_{a,u}$, SumLang puede tomar valores de 0 o 1 en dependencia de la coincidencia del lenguaje entre el objeto y el usuario autenticado; SumRes puede tomar valores de 0 o 1 en dependencia de la coincidencia del tipo de recurso entre el objeto y el usuario autenticado; CantDow almacena la cantidad de veces que el usuario ha descargado ese objeto de aprendizaje; y CantView almacena la cantidad de veces que el usuario ha visualizado ese objeto de aprendizaje.

$$\text{SumLang} = \begin{cases} 1 & \text{si el lenguaje del objeto coincide con el lenguaje preferido por el usuario} \\ 0 & \text{si el lenguaje del objeto no coincide con el lenguaje preferido por el usuario} \end{cases}$$
$$\text{SumRes} = \begin{cases} 1 & \text{si el tipo de recurso del objeto coincide con el lenguaje preferido por el usuario} \\ 0 & \text{si el tipo de recurso del objeto no coincide con el lenguaje preferido por el usuario} \end{cases}$$

Las recomendaciones se ordenan por los pesos y se seleccionan los 20 primeros para sugerirlos al usuario autenticado.

2.6.1 Definición de los pesos para la búsqueda ordenada

Luego de haber realizado la búsqueda por un filtro, se calcula el peso de los objetos de aprendizaje, ordenándolos de mayor a menor a través de la fórmula:

$$P = \text{SumLang} + \text{SumRes} + \text{CantDow} + \text{CantView}$$

La variable P hace alusión al peso de la búsqueda, el resto de las variables expuestas en la misma ya fueron descritas anteriormente.

2.7 Modelo de dominio

El proceso de negocio del sistema de recomendación comprende pocos conceptos, por lo que se decide utilizar un modelo de dominio. Este es una representación de los objetos que existen o los eventos que

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

sucedan 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. (Jacobson, Rumbaugh y Booch, 1999)

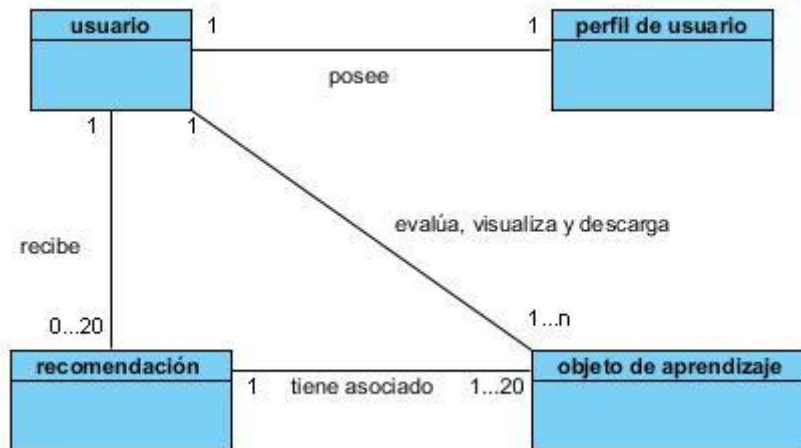


Figura 3: Modelo de dominio.

En la figura 3 se presenta el modelo de dominio del sistema de recomendación, en el que se tiene en cuenta las características del mismo y se puede observar la relación que existe entre los conceptos fundamentales y la aplicación a desarrollar, para alcanzar un lenguaje común y mejor entendimiento entre desarrolladores, clientes y usuarios finales. Consiste en las recomendaciones de hasta 20 objetos de aprendizaje que recibe un usuario, el cual puede evaluarlos, visualizarlos y descargarlos. El propio usuario posee un perfil académico, donde se almacenan todas sus características.

Principales conceptos:

Usuario: Persona que accede a la aplicación.

Recomendación: Es la proposición de objetos de aprendizaje que se le hace al usuario.

Objeto de aprendizaje: Recurso con una intención formativa, compuesto de uno o varios elementos digitales, descritos con metadatos, que puede ser utilizado y reutilizado dentro de un entorno e-learning.

Perfil académico: Almacena todas las preferencias y características del usuario.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

2.8 Requerimientos

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 tenga el software a desarrollarse. (Jacobson, Rumbaugh y Booch, 2000)

2.8.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir, para llegar a un acuerdo con el cliente. A continuación se muestran los requisitos funcionales de este sistema de recomendación.

RF1 Recomendar objetos de aprendizaje: Recomienda al usuario objetos de aprendizaje de acuerdo a su perfil.

RF2 Buscar objetos de aprendizaje: Busca los objetos de aprendizaje de acuerdo a un filtro definido por el usuario y el resultado puede ser ordenado, mostrando en las primeras posiciones los OA que más se adaptan al perfil del usuario.

2.8.2 Requisitos no funcionales

A continuación se mostrarán los requisitos no funcionales que son propiedades o cualidades que el sistema debe tener.

Disponibilidad: El sistema estará disponible las 24 horas del día.

Eficiencia: El sistema debe demorar, como promedio en una petición, de uno (1) hasta cinco (5) segundos, dependiendo del ancho de banda que tenga el local donde sea instalado el software.

Restricciones del diseño: El producto se implementará como una aplicación web.

Interfaz de usuario: El sistema tiene que ofrecer una interfaz fácil de operar. Igualmente tiene que mantener las pautas de diseño establecidas en el proyecto RHODA.

2.9 Modelo de casos de uso del sistema

Un diagrama de casos de uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema, en lo que se refiere a su interacción externa donde un

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

actor es como una persona, un sistema informatizado u organización que realiza algún tipo de interacción con el sistema; y un caso de uso es una descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica. (Grau, 2004)

2.9.1 Actores del sistema

Los actores de un sistema son agentes externos, es decir, aquellas personas o sistemas que interactúan con el propio sistema.

Tabla 3: Actores del sistema

Actor del sistema	Descripción
Usuario	Persona que tendrá acceso al sistema como usuario estándar, teniendo la posibilidad de recibir recomendaciones.

2.9.2 Diagrama de casos de uso del sistema

Especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores, así como los requisitos funcionales que este satisface.

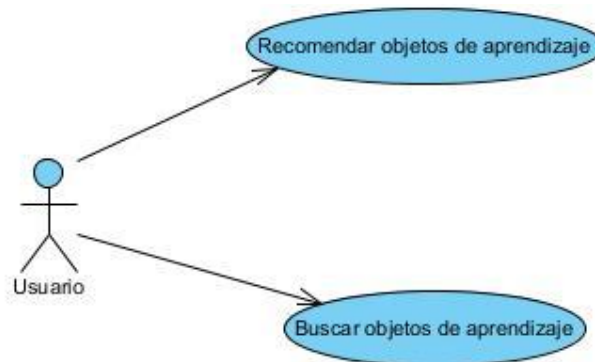


Figura 4: Diagrama de casos de uso.

2.9.3 Descripciones de casos de uso

La descripción de los casos de uso del sistema, detallan las acciones que tienen lugar durante la interacción actor-sistema, es decir, describe el flujo de actividades que realiza el actor al hacer uso del

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

sistema y las correspondientes respuestas del mismo. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre qué es lo que el sistema debe hacer (requisitos). A continuación se realizan las descripciones de los casos de uso “Recomendar objetos de aprendizaje” y “Buscar objetos de aprendizaje”.

Tabla 4: Descripción del CU Recomendar objetos de aprendizaje

Caso de uso:	Recomendar objetos de aprendizaje
Actores:	Usuario (Inicia)
Resumen:	El caso de uso inicia cuando el usuario se autentica en la aplicación. El sistema muestra una interfaz con los objetos de aprendizaje recomendados, finalizando así el caso de uso.
Precondiciones:	Debe haberse generado el escritorio de trabajo del usuario autenticado.
Poscondiciones:	Se recomendaron los objetos de aprendizaje.
Prioridad:	Normal
Referencia:	RF 1
Flujo normal de los eventos	
Acción del actor	Respuesta del sistema
1. Se autentica en la aplicación.	
	2. Muestra un listado con los objetos de aprendizaje recomendados que contiene el título del OA y permite: <ul style="list-style-type: none">• Ver más (en caso de que el listado exceda los 5 OA)
	3. Termina el caso de uso.
Flujos alternos	
3. Selecciona la opción Ver más.	
Acción del actor	Respuesta del sistema
1. Selecciona la opción Ver más.	

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.


	2. Muestra un listado de 20 objetos de aprendizaje que contiene el título del OA.
	3. Termina el caso de uso
Prototipo de interfaz	
	

Tabla 5: Descripción del CU Buscar objetos de aprendizaje

Caso de uso:	Buscar objetos de aprendizaje
Actores:	Usuario (Inicia)
Resumen:	El caso de uso inicia cuando el usuario desea realizar una búsqueda general de OA. El sistema realiza la búsqueda y muestra los resultados de la misma puede estar ordenado, mostrando en las primeras posiciones los OA que más se adaptan al perfil del usuario, finalizando así el caso de uso.
Precondiciones:	El usuario tiene que estar autenticado.
Poscondiciones:	Se buscaron los objetos de aprendizaje.
Prioridad:	Normal
Referencia:	RF 2
Flujo normal de los eventos	


Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

Acción del actor	Respuesta del sistema
1. Selecciona la opción Búsqueda general.	
	<p>2. Muestra la pantalla para realizar la búsqueda que contiene:</p> <ul style="list-style-type: none">• El tiempo de búsqueda.• El mensaje: Buscando en Item (Recursos) y Objetos de aprendizaje (Título, Descripción, Palabra clave).• El nombre del OA.• El autor.• La fecha de publicación.• La palabra clave. <p>3. Permite introducir una frase que verificará, esté incluida en el título del OA, en su descripción o en su contenido y realizar las acciones:</p> <ul style="list-style-type: none">• Buscar.• Ordenar el resultado de la búsqueda de acuerdo al perfil del usuario.• Descargar (Ver caso de uso: Descargar)• Incluir (Ver caso de uso: Incluir a Favoritos)
4. Introduce la frase del objeto de aprendizaje que desea buscar y selecciona la opción Buscar.	
	5. Muestra el listado de objetos de aprendizaje que coincidan con la frase introducida.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

	6. Termina el caso de uso.
Flujos alternos	
4. Selecciona la opción de ordenar el resultado de la búsqueda.	
Acción del actor	Respuesta del sistema
1. Selecciona la opción Ordenar el resultado de la búsqueda y luego la opción Buscar.	
	2. Calcula el peso a cada OA del resultado de la búsqueda y ordena este listado de mayor a menor. 3. Muestra el listado ordenado de objetos de aprendizaje que coinciden con la frase introducida.
	4. Termina el caso de uso.
Prototipo de interfaz	
 <p>The screenshot shows the RHODA search interface. At the top, there is a search bar containing the text 'objeto de aprendizaje' and a 'Buscar' button. Below the search bar, there is a checkbox labeled 'Ordenar resultado por mi perfil.' which is currently unchecked. The search results section displays 'Se encontraron 2 resultados (0.22 segundos).' and 'Buscado en Item(Recursos) y Objetos de Aprendizaje(Título, Descripción, Palabra clave).' There are two search results listed, each with a small icon, a title, author, publication date, and keywords. The first result is 'Objeto de Aprendizaje 1' by Yandris Mata Cabrera, published on 21/05/2013, with the keyword 'objeto'. The second result is 'Objeto de Aprendizaje 2' by Eduardo Padron Suarez, published on 20/03/2013, with the keyword 'objeto'. Each result has 'Descargar' and 'Incluir' links.</p>	

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

2.10 Modelo de análisis

El modelo de análisis es un modelo conceptual, debido a que está compuesto por clases y relaciones, que en su conjunto permiten interpretar la parte del sistema modelada. Es utilizado fundamentalmente por los desarrolladores para comprender cómo debería darse forma al sistema, es decir, cómo debería ser diseñado e implementado. Sirve como una primera aproximación del diseño y su objetivo es comprender perfectamente los requisitos del software y no precisar cómo se implementará la solución. (Jacobson, Rumbaugh y Booch, 2000)

2.10.1 Clases del análisis

Las clases del análisis se centran en los requisitos funcionales y son evidentes en el dominio del problema, porque representan conceptos y relaciones del dominio. RUP propone clasificar las clases en: interfaz, control y entidad.

- Las clases interfaz modelan la interacción entre el sistema y sus actores. Representan ventanas, formularios, comunicación con otros sistemas o dispositivos.
- Las clases entidad modelan la información que posee larga vida y que es a menudo persistente.
- Las clases control se encargan de coordinar la realización de uno o pocos casos de uso, coordinando las actividades de los objetos que implementan la funcionalidad del caso de uso.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

2.10.2 Diagrama de clases del análisis

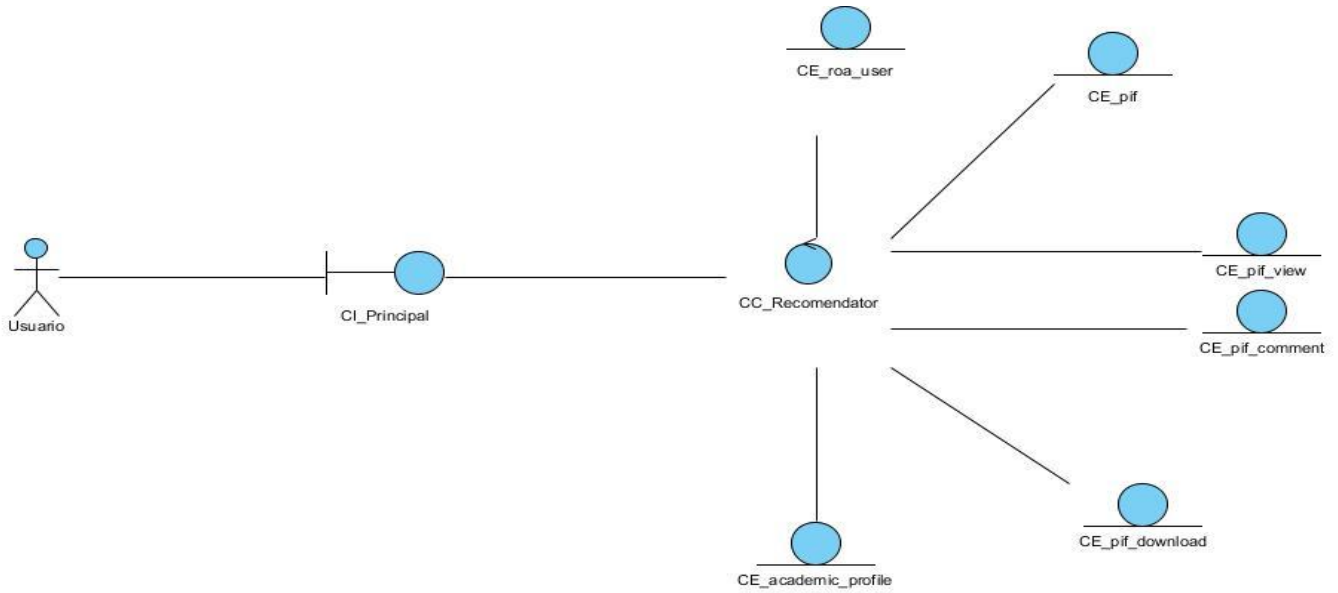
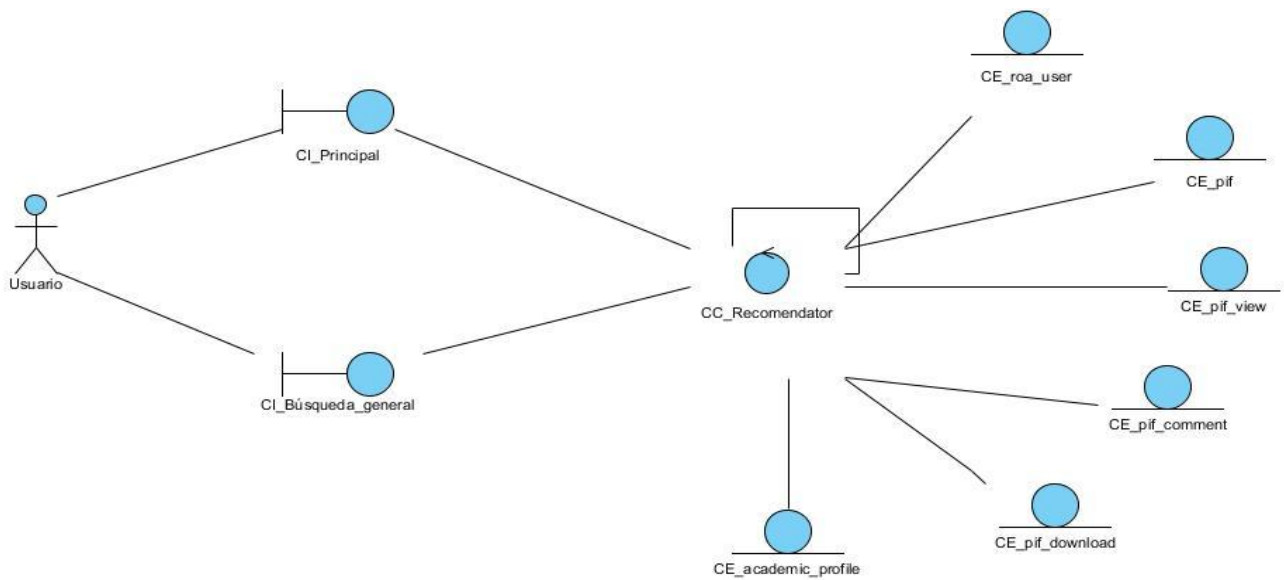


Figura 5: DCA CU Recomendar objetos de aprendizaje



Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

Figura 6: DCA CU Buscar objetos de aprendizaje.

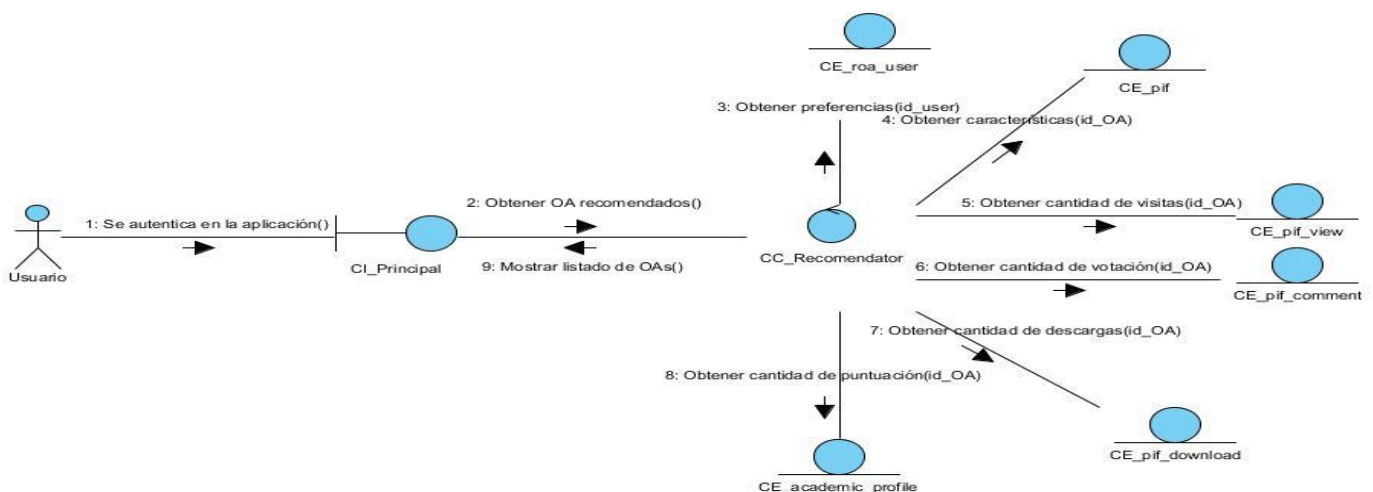
2.10.3 Diagramas de colaboración

Los diagramas de interacción explican gráficamente cómo los objetos, a través de mensajes interactúan para realizar las tareas. (Larman, 1999) 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:

- 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.

Los diagramas de colaboración muestran las relaciones entre los roles de los objetos. La secuencia de los mensajes y los flujos de ejecución concurrentes, deben determinarse explícitamente mediante números de secuencia. En un diagrama de colaboración los mensajes se muestran como flechas, ligadas a las líneas de la relación, que conectan a los roles. La secuencia de mensajes se indica con los números secuenciales que preceden a las descripciones del mensaje. Un diagrama de colaboración se utiliza para mostrar la implementación de una operación. (Eguíluz, 2012)



Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

Figura 7: DC CU Recomendar objetos de aprendizaje

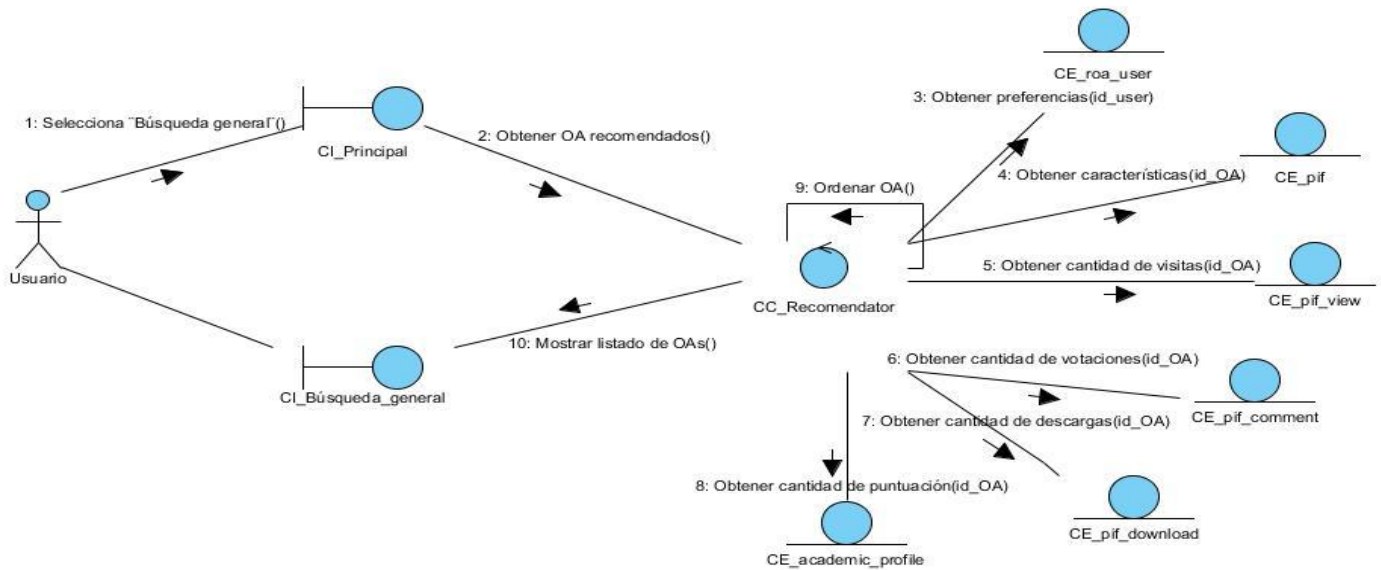


Figura 8: DC CU Buscar objetos de aprendizaje.

2.11 Modelo de diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso, centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación.

2.11.1 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, se les denomina: patrones de software. (Larman, 1999)

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,

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

(Larman, 1999) 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. (Gamma, 2003)

Se hace uso de los Patrones Generales de Software para Asignación de Responsabilidades (GRASP) por sus siglas en inglés, General Responsibility Assignment Software Patterns.

Patrones Graps:

Los patrones de diseño de asignación de responsabilidades 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 del sistema 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. (Larman, 1999) Se utiliza 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. (Larman, 1999) 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. (Larman, 1999) 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

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

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. (Larman, 1999) 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 posibles para poder aumentar la cohesión y disminuir el acoplamiento. (Larman, 1999)

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. De estos se utilizan los patrones de diseño Iterator y Decorator.

Iterator

Proporciona una forma de acceder a los elementos de diferentes colecciones de objetos sin necesidad de develar su representación interna. (Gamma, 2003) 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.

Decorator

Patrón de tipo estructura, a nivel de objetos. Añade responsabilidades adicionales a un objeto dinámicamente. (Gamma, 2003) Se utiliza este patrón para la vista y el layout o plantilla global que decora el contenido de la misma.

2.11.2 Arquitectura de la aplicación

Para realizar la solución propuesta se utiliza Symfony, que implementa el patrón MVC distribuyendo los componentes de la siguiente manera: en el modelo, la abstracción de la base de datos y el acceso a los

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

datos; en la vista, las plantillas, los layouts¹ y las vistas, y en el controlador, las acciones y el controlador frontal. El controlador frontal y el layout son comunes para todas las acciones de la aplicación. Se pueden tener varios controladores y varios layouts, pero se debe tener de manera obligatoria al menos uno. El controlador frontal es un componente que solo tiene código relativo al MVC, por lo que no es necesario crear uno, ya que Symfony lo genera de forma automática. (Eguíluz, 2012)

Symfony dentro de la arquitectura MVC implementa varias clases como son:

- `sfController`: es la clase del controlador. Se encarga de decodificar la petición y transferirla a la acción correspondiente.
- `sfRequest`: almacena todos los elementos que forman la petición (parámetros, cookies y cabeceras).
- `sfResponse`: contiene las cabeceras de la respuesta y los contenidos. El contenido de este objeto se transforma en la respuesta HTML que se envía al usuario.
- El singleton de contexto: almacena una referencia a todos los objetos que forman el núcleo de Symfony y puede ser accedido desde cualquier punto de la aplicación.

2.11.3 Diagramas de clases de diseño

Los diagramas de clases del diseño muestran el diseño del sistema desde un punto de vista estático, a través de una colección de elementos declarativos, como clases, colaboraciones y sus relaciones. Principalmente, esto incluye modelar el vocabulario del sistema, las colaboraciones o los esquemas.

¹Plantillas que definen los elementos comunes, como menús y cabecera.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

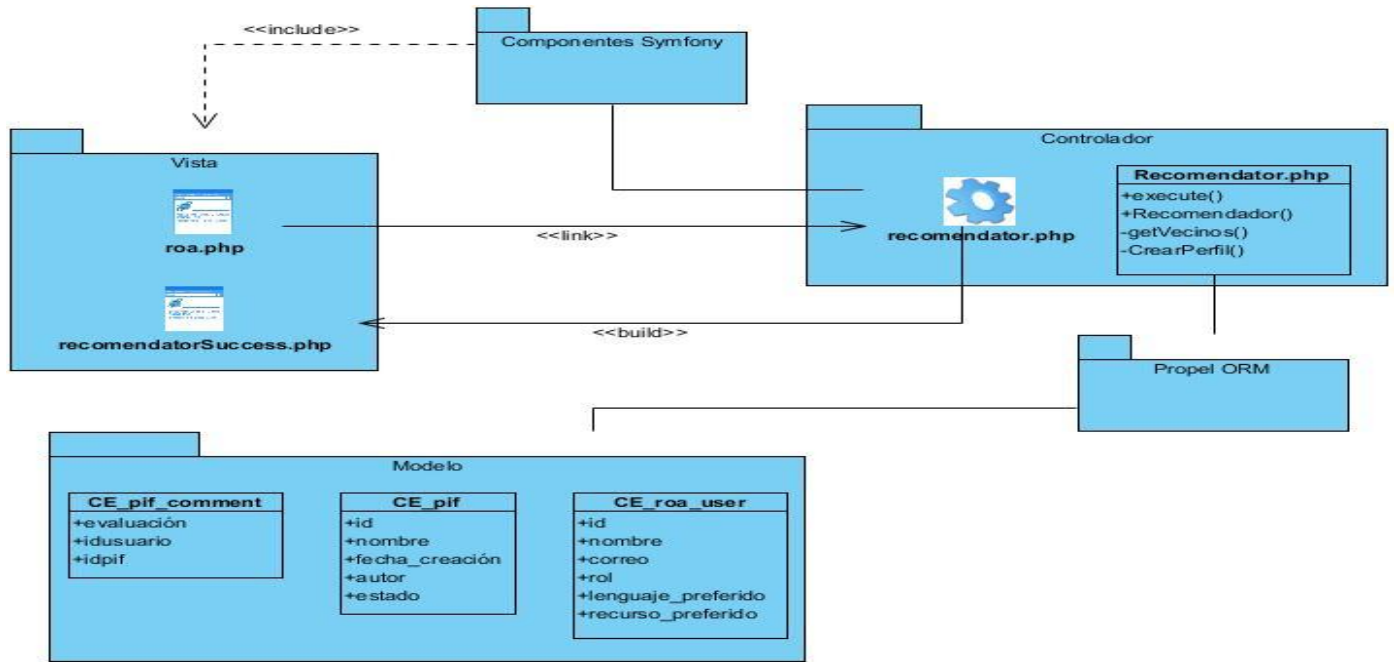


Figura 9: DCD CU Recomendar objetos de aprendizaje.

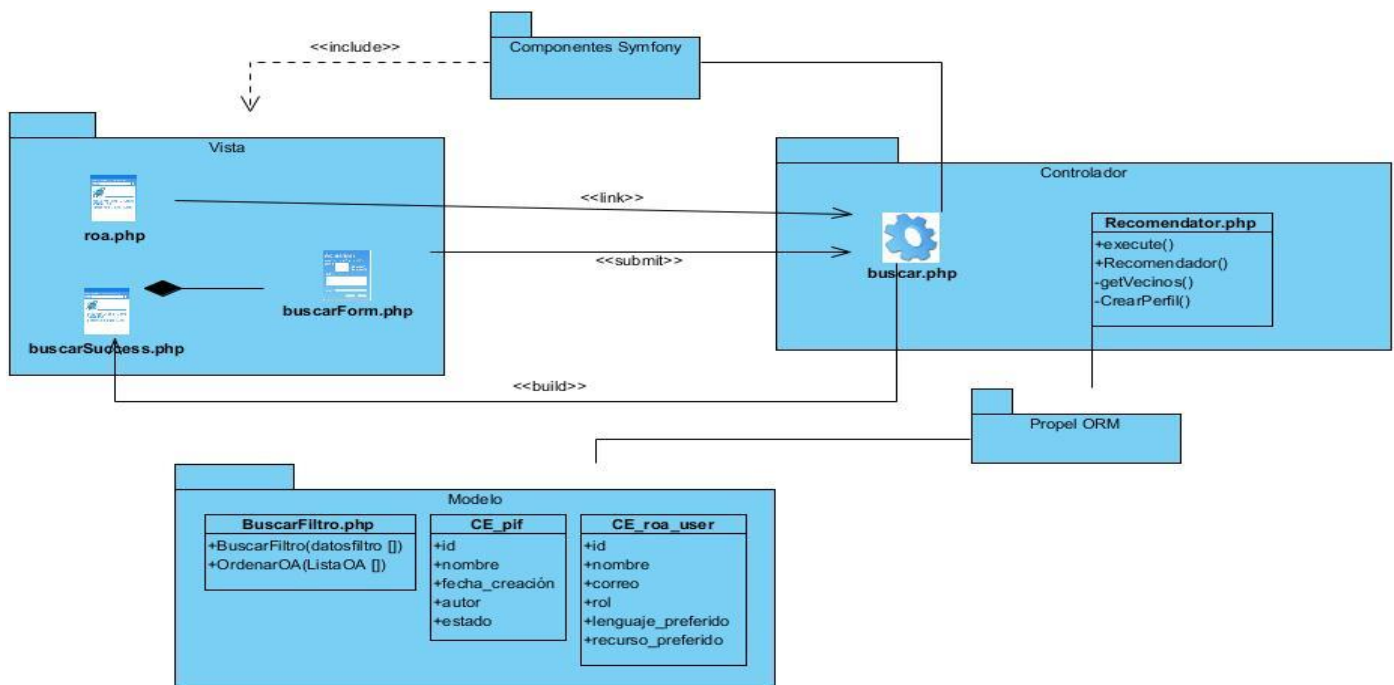


Figura 10: DCD CU Buscar objetos de aprendizaje.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

2.11.4 Diagramas de secuencia del diseño

Un diagrama de secuencia muestra una interacción ordenada según la secuencia de eventos. Muestra los objetos que participan en la interacción mediante sus líneas de vida y mediante los mensajes que intercambian, organizados en forma de una secuencia temporal. El eje vertical representa el tiempo, y en el eje horizontal se colocan los objetos y actores participantes en la interacción, sin un orden prefijado. Cada objeto o actor tiene una línea vertical, y los mensajes se representan mediante flechas entre los distintos objetos. El tiempo fluye de arriba abajo. Se pueden colocar etiquetas bien en el margen izquierdo o bien junto a las transiciones o activaciones a las que se refieren. (Segura, 2004)

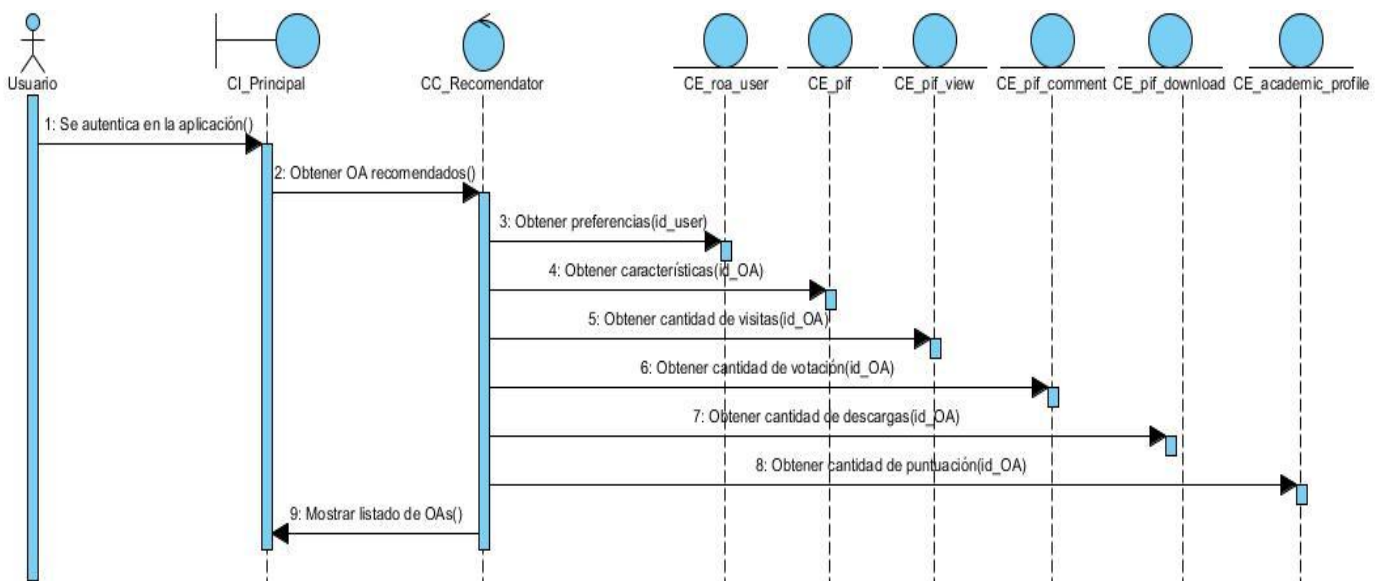


Figura 11: DSD CU Recomendar objetos de aprendizaje.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

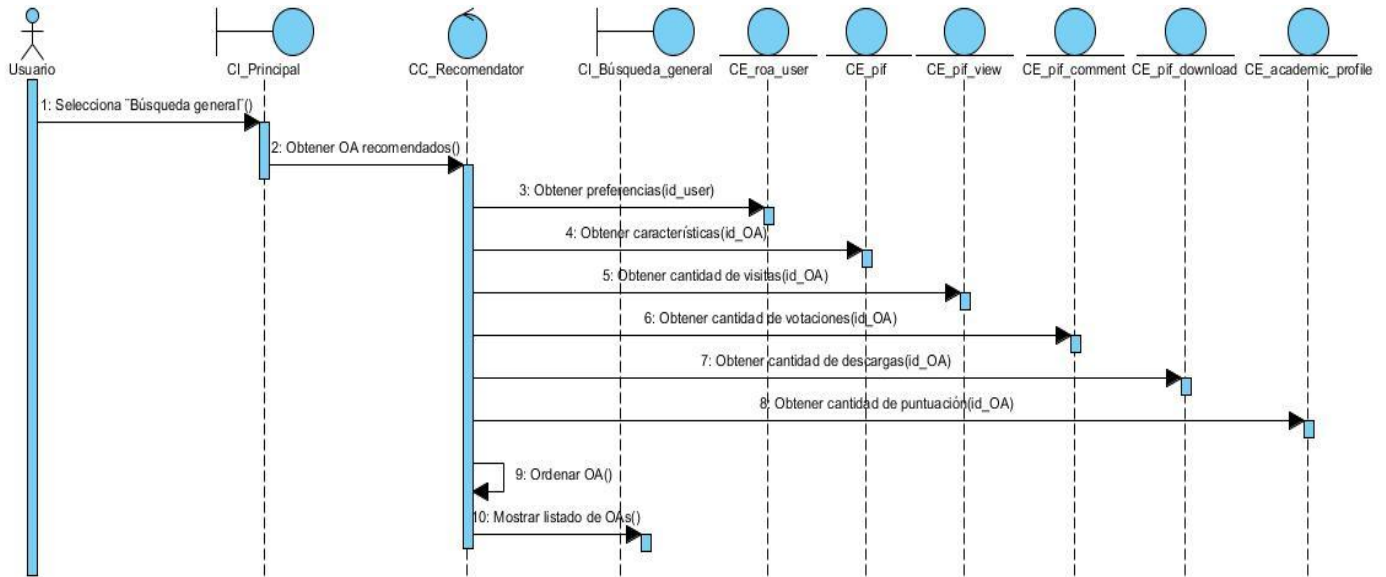


Figura 12: DSD CU Buscar objetos de aprendizaje.

2.11.5 Modelo de datos

Un modelo de datos permite describir la estructura de las bases de datos, teniendo en cuenta los tipos de datos y las relaciones que existen entre las tablas.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

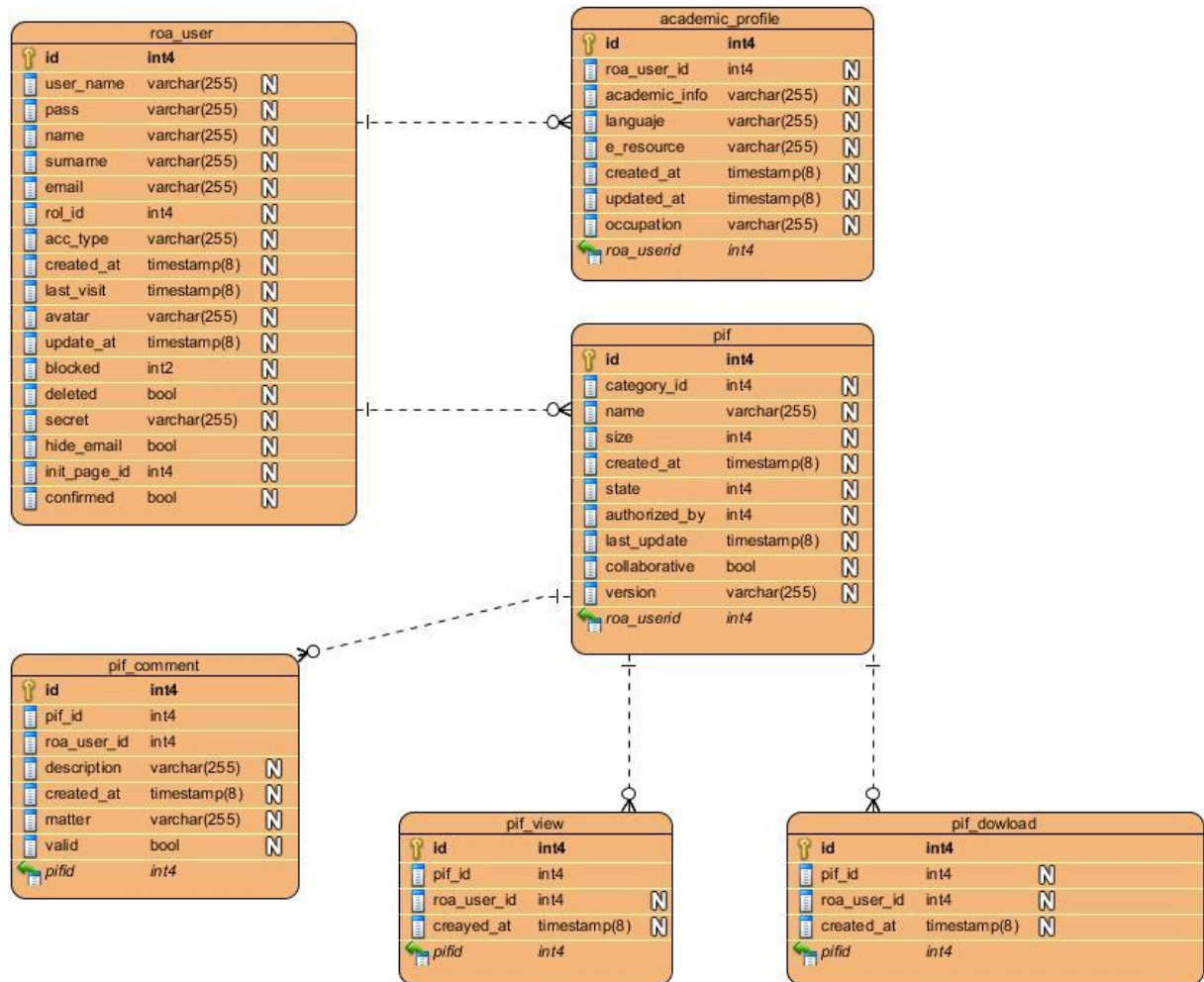


Figura 13: Modelo de datos.

2.11.5.1 Descripción de las tablas del modelo de datos

pif: Modela el paquete que contiene el objeto de aprendizaje según el estándar SCORM.

pif_comment: Clase para modelar los comentarios que se agregan a un paquete.

pif_views: Modela las visualizaciones de los OA por parte de los usuarios.

roa_user: Usada para almacenar los usuarios del sistema, con todas sus características y las acciones que este puede realizar en el repositorio.

academic_profile: Clase que modela el perfil académico de los usuarios.

Capítulo 2: Diseño de la solución propuesta

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

pif_download: Usada para almacenar las descargas de los OA realizadas por los usuarios del sistema.

Conclusiones

Se identificaron los requisitos funcionales y no funcionales que debe cumplir el sistema en cuestión, y con ello fueron expuestos los casos de uso a tratar durante el desarrollo del mismo, con su correspondiente descripción, lo cual provee una visión general de qué es lo que el sistema debe hacer. Además se transformaron los requerimientos en diagramas, que indican cómo debe ser implementado el sistema. También como parte de las actividades de análisis y diseño se identificó el patrón arquitectónico y los patrones de diseño que se deben emplear para el desarrollo de las funcionalidades identificadas. Además se realizó el modelo de base datos, donde quedaron plasmadas todas las relaciones de las tablas necesarias para recomendar objetos de aprendizaje a los usuarios.

Capítulo 3: Implementación y prueba

Introducción

La metodología RUP plantea que la implementación de un software debe realizarse de forma iterativa e incremental, permitiendo que durante todo el proceso de desarrollo se produzcan versiones superiores como resultado de cada ciclo. En este capítulo se detallan los ciclos llevados a cabo durante la fase de construcción del sistema, exponiéndose las actividades generadas por cada caso de uso, así como las pruebas efectuadas para verificar el buen funcionamiento del sistema.

3.1 Estándares de codificación

Son reglas que se utilizan para la escritura del código fuente. Permiten que otros desarrolladores puedan interpretar de manera eficiente la escritura del código; asegurando que todos trabajen de forma conjunta 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 RHODA. A continuación se hace mención de algunos y los restantes se pueden consultar en Anexo 3 Estándares de codificación.

Funciones y Métodos

Los nombres de funciones pueden contener únicamente caracteres alfanuméricos. Los guiones bajos (_) no están permitidos. Los números están permitidos en los nombres de función pero no se aconseja en la mayoría de los casos.

Los nombres de funciones deben empezar siempre con una letra minúscula. Cuando un nombre de función consiste en más de una palabra, la primera letra de cada nueva palabra debe estar en mayúsculas. Esto es llamado comúnmente como formato camelCase.

Por norma general, se recomienda la elocuencia. Los nombres de función deben ser lo suficientemente elocuentes como para describir su propósito y comportamiento.

Estos son ejemplos de nombres de funciones admisibles:

Capítulo 3: Implementación y pruebas

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

`filterInput()`

`getElementById()`

`widgetFactory()`

Para la programación orientada a objetos, los métodos de acceso para las instancias o variables estáticas deben ir antepuestos con un `get` o un `set`. Al implementar el patrón de diseño, tales como el patrón singleton o el patrón factory, el nombre del método debe contener en la práctica el nombre del patrón para describir su comportamiento de forma más completa.

Para el caso en que los métodos son declarados con el modificador `private` o `protected`, el primer carácter del nombre de la variable debe ser una barra baja (`_`). Este es el único uso admisible de una barra baja en un nombre de método. Los métodos declarados como públicos no deberían contener nunca una barra baja.

Las funciones de alcance global (también llamadas funciones flotantes) están permitidas pero desaconsejadas en la mayoría de los casos. Considere envolver esas funciones en una clase estática.

Identación

La indentación debe estar compuesta por 4 espacios. Las tabulaciones no están permitidas.

Tamaño máximo de línea

La longitud recomendable para una línea de código es de 80 caracteres. Esto significa que los desarrolladores deberían intentar mantener cada línea de su código por debajo de los 80 caracteres, siempre que sea posible. No obstante, líneas más largas pueden ser aceptables en algunas situaciones. El tamaño máximo de cualquier línea de código PHP es de 120 caracteres.

Final de línea

El final de línea sigue la convención de archivos de texto Unix. Las líneas deben acabar con un carácter linefeed (LF). Los caracteres Linefeed están representados con el número 10 ordinal, o el número 0x0A hexadecimal.

Nota: No pueden utilizar retornos de carro (carriage returns, CR), como en las fuentes de Apple (0x0D) o la combinación de retorno de carro-linefeed (CRLF) estándar para sistemas operativos Windows (0x0D, 0x0A).

3.2 Modelo de implementación

El 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, entre otros. (Jacobson, Rumbaugh y Booch, 2000)

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

3.3 Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las dependencias lógicas entre componentes de software, sean estos componentes fuentes, binarios o ejecutables. Prevalen en el campo de la arquitectura de software, pero pueden ser usados para modelar y documentar cualquier arquitectura del sistema, ayudando a describir la vista de implementación estática. A continuación se presenta el diagrama de componentes propuesto para este sistema:

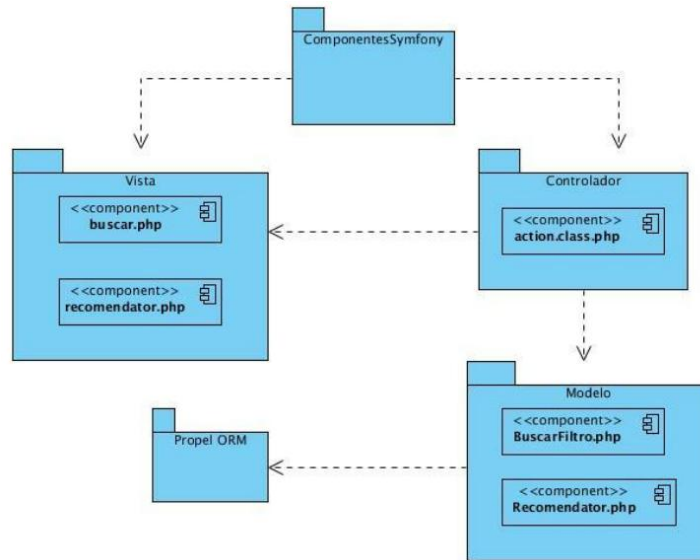


Figura 14: Diagrama de componentes.

3.4 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. (Jacobson, Rumbaugh y Booch, 2000)

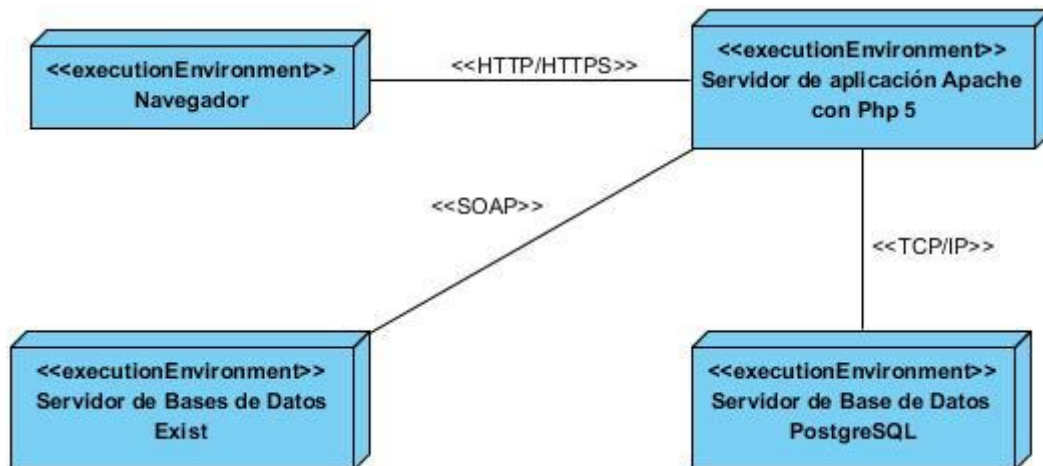


Figura 15: Diagrama de despliegue.

3.5 Pruebas de software

Para determinar el estado de la calidad de un producto software es necesario realizar el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del mismo o al sistema en su totalidad, con el objetivo de medir el grado en que el producto cumple con los requerimientos.

Métodos de prueba

RUP propone dos métodos de prueba: caja blanca y caja negra. A continuación se describe este último método que fue el utilizado en la comprobación de la solución.

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. (Pressman, 2005)

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.

Capítulo 3: Implementación y pruebas

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

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. (Murillo y Martínez 2011)

3.6 Diseño de Casos de Prueba

Descripción general del CU Recomendar objetos de aprendizaje:

El caso de uso inicia cuando el actor se autentica en la aplicación. El sistema muestra una interfaz con los objetos de aprendizaje recomendados, finalizando así el caso de uso.

Condiciones de ejecución:

El usuario tiene que haberse autenticado.

Secciones a probar en el caso de uso:

Tabla 6: Diseño del caso de prueba para el CU Recomendar objetos de aprendizaje.

SC 1 Recomendar objetos de aprendizaje			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Recomendar objetos de aprendizaje.	El usuario se autentica en la aplicación. El sistema muestra un listado con los objetos de aprendizaje recomendados.	Muestra un listado con los objetos de aprendizaje recomendados que contiene el título del OA y permite: <ul style="list-style-type: none">• Ver más (en caso de que el listado exceda los 5 OA).	Inicio
EC 1.2 Selecciona la opción Ver más.	El usuario se autentica en la aplicación. El sistema muestra un listado con los	Muestra un listado de 20 objetos de aprendizaje que contiene el título del OA.	Inicio

Capítulo 3: Implementación y pruebas

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

	objetos de aprendizaje recomendados. Selecciona la opción Ver más. El sistema muestra un listado de 20 objetos de aprendizaje.		
--	--	--	--

Descripción general para el CU Buscar objetos de aprendizaje:

El caso de uso inicia cuando el usuario selecciona la opción de “Búsqueda general”. El sistema ordena los objetos de aprendizaje de mayor a menor, finalizando así el caso de uso.

Condiciones de ejecución: El usuario tiene que haberse autenticado.

Secciones a probar en el caso de uso:

Tabla 7: Diseño del caso de prueba para el CU Buscar objetos de aprendizaje.

SC 1 Buscar objetos de aprendizaje			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Buscar objetos de aprendizaje.	Selecciona la opción de Búsqueda general. El sistema muestra la pantalla para realizar la búsqueda y permite introducir el título del objeto de aprendizaje que desea buscar.	Muestra la pantalla para realizar la búsqueda que contiene: <ul style="list-style-type: none"> • El tiempo de búsqueda. • El mensaje: Buscando en Ítem (Recursos) y Objetos de aprendizaje (Título, Descripción, Palabra clave). • El nombre del OA. • El autor. • La fecha de publicación. • La palabra clave. Permite introducir una frase que	Inicio/Búsqueda general

Capítulo 3: Implementación y pruebas

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

		<p>verificará, esté incluida en el título del OA, en su descripción o en su contenido y realizar las acciones:</p> <ul style="list-style-type: none"> • Buscar. • Ordenar el resultado de la búsqueda de acuerdo al perfil del usuario. • Descargar (Ver caso de uso: Descargar) • Incluir (Ver caso de uso: Incluir a Favoritos) 	
EC 1.2 Realizar la búsqueda de objetos de aprendizaje.	<p>Selecciona la opción de Búsqueda general. El sistema muestra la pantalla para realizar la búsqueda y permite introducir la frase del objeto de aprendizaje que desea buscar.</p> <p>Introduce la frase del objeto de aprendizaje que desea buscar y selecciona buscar.</p>	Muestra el listado de objetos de aprendizaje que coincidan con la frase introducida.	Inicio/Búsqueda general
EC 1.3 Selecciona Ordenar el resultado de la búsqueda.	<p>Selecciona la opción de Búsqueda general. El sistema muestra la pantalla para realizar la búsqueda y permite introducir la frase del objeto de aprendizaje que desea buscar.</p> <p>Selecciona ordenar el resultado</p>	<p>A cada OA del resultado de la búsqueda le calcula un peso y ordena este listado de mayor a menor de acuerdo a los pesos del OA.</p> <p>Muestra el listado ordenado de objetos de aprendizaje que</p>	Inicio/Búsqueda general

Capítulo 3: Implementación y pruebas

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

	de la búsqueda y luego buscar.	coinciden con la frase de búsqueda introducida.	
--	--------------------------------	---	--

3.7 Resultados de las pruebas

Se realizaron tres iteraciones donde se encontraron cuatro no conformidades entre la primera y la segunda iteración, las que fueron mitigadas en la tercera iteración para lograr la calidad y el correcto funcionamiento del sistema. 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 8: Cantidad de no conformidades por iteración

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

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:

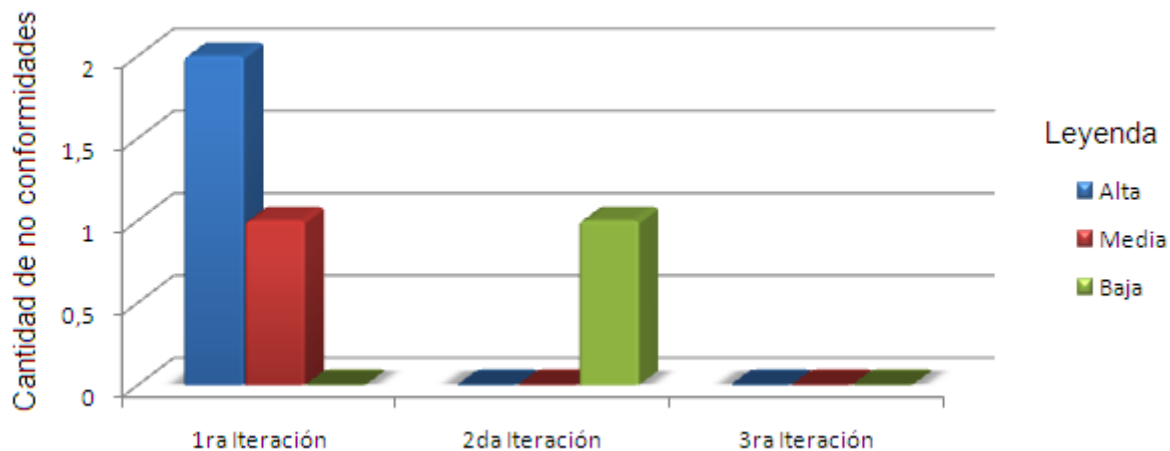


Figura 16: Cantidad de no conformidades detectadas en cada iteración.

Capítulo 3: Implementación y pruebas

Sistema de recomendación de objetos de aprendizaje en el Repositorio de Objetos de Aprendizaje RHODA.

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

El diagrama de despliegue permitió modelar la estructura física de los nodos que componen el sistema y la relación entre ellos, de igual forma el diagrama de componentes reflejó la relación entre cada uno de los elementos que intervienen en el sistema propuesto. Finalmente tras un análisis de los resultados obtenidos en las pruebas, se demostró la solidez de la solución, evidenciado en las tres iteraciones de pruebas realizadas.

Conclusiones generales

- Durante el transcurso de esta investigación se realizó un estudio que permitió tener un conocimiento de la situación actual y las tendencias de los sistemas de recomendación.
- Se demostró la necesidad de desarrollar un sistema recomendador que fuese capaz de sugerirle al usuario los OA que más se adaptan a su perfil en el repositorio.
- Con la utilización de la metodología RUP se generaron los artefactos necesarios para la implementación de la solución, la cual es validada con un conjunto de pruebas de caja negra las que en un final arrojaron cuatro no conformidades que fueron mitigadas completamente en la tercera iteración, por lo que se concluye que el sistema recomienda al usuario OA que más se adaptan a su perfil.

Recomendaciones

Como resultado del proceso de investigación y realización del sistema de recomendación, han surgido nuevas ideas que se podrían tener en cuenta por el equipo de desarrollo del proyecto RHODA, para un futuro perfeccionamiento del sistema, a continuación se listan las mismas:

- Extender el recomendador de objetos de aprendizaje, cuando se realicen búsquedas federadas.
- Incluir en las recomendaciones de objetos de aprendizaje, el perfil del usuario en otros entornos académicos.
- Investigar la posibilidad de crear una funcionalidad que determine un rango óptimo para hallar la cantidad de vecinos y de recomendaciones.

Referencias

1. **Hill, Loren Terveen y Will.** Beyond Recommender Systems: Helping People Help Each Other. 2001.
2. **Yager, Ronald R.** Fuzzy logic methods in recommender systems. Elsevier North-Holland : s.n., 2003.
3. **Konstan, Josep.** IntroductionTo Recommender Systems: Algorithms and Evaluations. 2004.
4. **Sarwar, Badrul.** Item-based collaborative filtering recommendation algorithms. 2001.
5. **Plasencia, Nadiela Milán y Ernesto.** Herramienta de selección didáctica para guiar el aprendizaje interactivo en el módulo Ejercicios de la colección El Navegante. 2012.
6. **Tuzhilin., G Adomavicius y A.** *Recommendation Technologies: Survey of current methods and possible extensions.* Minessota : MISRC working paper 0329, 2003.
7. **Billsus, Michael Pazzani y Daniel.** Content-Based Recommendation System. The Adaptive Web: Methods and Strategies of Web Personalization. Berlin : s.n., 2007.
8. **Toledo, Raciél Yera.** Concepción y desarrollo de un sistema de recomendación para jurados online de programación. La Habana : s.n., 2010.
9. **Ricci., Francesco.** *Recommender Systems Handbook.* New York : Springer New York Dordrecht Heidelberg London, 2011.
10. **Hernández., Manuel Fernando Caro Piñeres y Jaime.** Diseño de un sistema de recomendación en repositorios de objetos de aprendizaje basado en la percepción del usuario: Caso RODAS. Bogotá : s.n., 2011.
11. *Sistema inteligente para la recomendación de objetos de aprendizaje.* **Ana Casali, Valeria Gerling, Claudia Deco y Cristina Bender.** s.l. : Revista Generación Digital, 2011.
12. **Rodríguez, Ana María Álvarez Valdés y Osmany Montes de Oca.** Desarrollo del módulo Ejercicios de la Colección Multisaber en su versión multiplataforma. 2010.
13. **Achour, M.** *Información General, in PHP Manual.* 2007.
14. **Triana, Daylén Delgado Fernández y Yasniel Benavides.** Análisis, Diseño e Implementación del Módulo Resultados de la Multisaber, en su versión multiplataforma. 2010.

15. PostgreSQL. [Online] enero 17, 2010. <http://www.postgresql.org/about/>.
16. **José Márquez Díaz, L.S. y Félix Vargas.** Instalación y configuración de Apache, un servidor Web gratis. 2002. Vol. 23.
17. **Koren, R. Bell y Y.** *Scalable collaborative filtering with jointly derived neighborhood interpolation weights.* USA : Conf. on Data Mining, 2007.
18. **Chen, R. Cheng y L.** *Evaluating probability threshold k-Nearest-neighbor.* USA : IEEE TC on Data Engineering, 2009.
19. **Shakhnarovich, D.** *Nearest neighbor methods in learning and vision.* USA : MIT Press, 2005.
20. **Rashid A., Karypis G. y Riedl J.** *Influence in ratings-based recommender systems: an algorithm-independent approach.* s.l. : En Journal of the ACM., 2006.
21. **Bregon A., Arancha C. y Rodríguez J.** *Un sistema de razonamiento basado en casos para la clasificación de fallos en sistemas dinámicos.* España : Actas del III Taller Nacional de Minería de Datos y Aprendizaje., 2005.
22. **Nguyen, K. Ni y Q.** *An adaptable -nearest neighbors algorithm for MMSE image interpolation.* USA : IEEE transactions on image processing., 2009.
23. **Maclachlan, D. Lemire y A.** *One predictors for online rating based collaborative filtering.* Noruega : Secure Data Management - SDM05., 2005.
24. **Grau, Xavier Ferré.** Diagrama de Caso de Uso. *Clikear.com.* [Online] 2004. <http://www.clikear.com/manuales/uml/diagramascasouso.aspx>.
25. **Ivar Jacobson, James Rumbaugh y Grady Booch.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 2000.
26. **Eguíluz, Javier.** librosweb.es. [Online] [Citado el: 15 de Marzo de 2012.] . http://www.librosweb.es/symfony/capitulo2/el_patron_mvc.html .
27. **Segura, X. F.** Diagramas de Interaccion de Clikear.com. [Online] 2004 (Recuperado el 1 de marzo de 2011). <http://www.clikear.com/manuales/uml/diagramasinteraccion.aspx> .
28. **Gamma, otros y Erich.** *Design Patterns. Elements of Reusable Object-Oriented Software.* s.l. : Pearson Education, 2003.

29. **Larman, Graig.** *UML y patrones. Introducción al análisis y diseño orientado a objetos.* México: Prentice Hall Hispanoamericana, S.A. : s.n., 1999.
30. **Potencier, Fabien y Zaninotto, François.** *Symfony 1.2, la guía definitiva.* 2008.
31. **Pérez, Javier Eguíluz.** *Introducción a CCS.* 2008.
32. *NetBeans IDE 4.1. La alternativa a Eclipse.* **Dorado, Manuel Domínguez.** Madrid : Editorial Iberprensa, Noviembre, 2005.. , Vol. Todo Programación.
33. **Pressman, Roger S.** *Ingeniería de Software Un enfoque práctico.* s.l. : Sexta edición, 2005.
34. **Mestras, Juan Pavón.** *Estructura de las aplicaciones Orientadas a Objeto.* 2009.
35. **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.
36. **González, Leonardo Rodríguez.** *Análisis y Diseño de la versión 3.0 de RHODA. Tesis de Grado.* La Habana, UCI : s.n., 2011.

Anexos

Anexo 1 Registro de defectos y dificultades

No.	No conformidad	Aspecto de la NC	Etapas de detección	Significativa	No significativa
1	Le recomienda OA a los usuarios sin que estos se hayan autenticado en la aplicación.	Recomienda OA sin que los usuarios se hayan autenticado en la aplicación.	2da Iteración		X
2	El sistema les recomienda varios OAs a los usuarios sin que existan otros usuarios en la aplicación, para definirlos como vecinos.	Les recomienda cualquier OA a los usuarios.	1ra Iteración	NC de aplicación: validación	
3	El sistema no ordena la búsqueda de los OA de acuerdo al perfil del usuario.	Le muestra al usuario cualquier OA, pudiendo no ser de su interés.	1ra Iteración	NC de aplicación: validación	
4	El sistema les muestra más de 5 OA a los usuarios.	Les muestra demasiados OAs a los usuarios.	1ra Iteración	NC de aplicación: validación	

Anexo 2 Modelo de entrevista

Este modelo de entrevista refleja las preguntas realizadas a especialistas con el fin de entender cómo es el proceso de recomendaciones y búsquedas dentro del proyecto RHODA.

Preguntas realizadas a especialistas:

1. ¿Como el usuario accede a los objetos de aprendizaje publicados en el repositorio RHODA?
2. ¿Existe algún mecanismo en el cual los usuarios dejen registradas sus preferencias?
3. ¿Es almacenado el historial de las acciones de visualizar y descargar de los objetos de aprendizaje?
4. ¿Los autores pueden dejar sus opiniones y evaluaciones sobre otros objetos de aprendizaje?
5. ¿Como los usuarios realizan la búsqueda de objetos de aprendizaje?
6. ¿Bajo qué criterios la búsqueda es ordenada y mostrada?

Anexo 3 Estándares de codificación

Convenciones de Nombres

Clases

Los nombres de clases pueden contener sólo caracteres alfanuméricos. Los números están permitidos en los nombres de clase. Las barras bajas (_) están permitidas solo como separador de ruta.

Si el nombre de una clase está compuesto por más de una palabra, la primera letra de cada palabra debe aparecer en mayúsculas. Poner en mayúsculas las letras siguientes no está permitido, ejemplo: Zend_PDF no está permitido, mientras que Zend_Pdf es admisible.

Interfaces

En general, las clases abstractas siguen las mismas convenciones que las clases, con una regla adicional: los nombres de las interfaces opcionalmente pueden acabar con el término, Interface, pero la terminación no debe ser precedida por un guión bajo. Ejemplo, Zend_Controller_Plugin_Interface es considerado un nombre no válido, pero Zend_Controller_PluginInterface o Zend_Controller_Plugin_PluginInterface serían nombres válidos.

Nombres de Archivo

Para cualquier otro archivo, sólo caracteres alfanuméricos, barras bajas (_) y guiones (-) están permitidos. Los espacios en blanco están estrictamente prohibidos.

Cualquier archivo que contenga código PHP debe terminar con la extensión .php, con la excepción de los scripts de la vista.

Los nombres de archivo deben apuntar a nombres de clases como se describe arriba.

Variables

Los nombres de variables pueden contener caracteres alfanuméricos. Las barras bajas (_) no están permitidas. Los números están permitidos en los nombres de variable pero no se aconseja en la mayoría de los casos.

Para las variables de instancia que son declaradas con el modificador private o protected, el primer carácter de la variable debe ser una única barra baja (_). Este es el único caso admisible de una barra

baja en el nombre de una variable. Las variables declaradas como public no pueden empezar nunca por barra baja.

Al igual que los nombres de funciones, los nombres de variables deben empezar siempre con una letra en minúscula y seguir la convención camelCaps.

Por norma general, se recomienda la elocuencia. Las variables deberían ser siempre tan elocuentes como prácticas para describir los datos que el desarrollador pretende almacenar en ellas. Variables escuetas como \$i y \$n están desaconsejadas, salvo para el contexto de los bucles más pequeños. Si un bucle contiene más de 20 líneas de código, las variables de índice deberían tener nombres más descriptivos.

Constantes

Las constantes pueden contener tanto caracteres alfanuméricos como barras bajas (_). Los números están permitidos.

Todas las letras pertenecientes al nombre de una constante deben aparecer en mayúsculas.

Las palabras dentro del nombre de una constante deben separarse por barras bajas (_).

Las constantes deben ser definidas como miembros de clase con el modificador const. Definir constantes en el alcance global con la función **define** está permitido pero no recomendado.

Estilo de código

Demarcación de código PHP

El código PHP debe estar delimitado siempre por la forma completa de las etiquetas PHP estándar:

```
<?php
```

```
?>
```

Las etiquetas cortas (short tags) no se permiten nunca. Para archivos que contengan únicamente código PHP, la etiqueta de cierre debe omitirse siempre.

Concatenación de cadenas

Las cadenas deben ser concatenadas usando el operador punto ("."). Un espacio debe añadirse siempre antes y después del operador "." para mejorar la legibilidad:

```
$company = 'Zend' . ' . 'Technologies';
```

En estos casos, cada línea sucesiva debe llevar un margen de espacios en blanco de forma que el operador "." está alineado bajo el operador "=".

Clases

Declaración de clases

Las Clases deben ser nombradas de acuerdo a las convención de nombres. La llave "{" deberá escribirse siempre en la línea debajo del nombre de la clase ("one true brace"). Todo el código contenido en una clase debe ser separado con cuatro espacios. Únicamente una clase está permitida por archivo PHP.

A continuación se muestra un ejemplo de una declaración de clase que es permitida:

```
/**
 * Bloque de Documentación aquí
 */
class SampleClass
{
    // el contenido de la clase
    // debe separarse con cuatro espacios
}
```

Las clases que extiendan de otras clases o interfaces deberían declarar sus dependencias en la misma línea siempre que sea posible.

```
class SampleClass extends FooAbstract implements BarInterface
{
}
```

Si como resultado de esas declaraciones, la longitud de la línea excede la longitud del tamaño máximo de línea, se debe romper la línea antes de la palabra clave "extends" y / o "implements" e indentarlo con un nivel de indentación (cuatro espacios).

```
class SampleClass
    extends FooAbstract
    implements BarInterface
```

```
{  
}
```

Funciones y Métodos

Declaración de Funciones y Métodos

Los métodos dentro de clases deben declarar siempre su visibilidad usando un modificador `private`, `protected` o `public`. Como en las clases, la llave "{" debe ser escrita en la línea siguiente al nombre de la función ("one true brace" form). No está permitido un espacio entre el nombre de la función y el paréntesis de apertura para los argumentos. La llamada por referencia está estrictamente prohibida.

El valor de retorno no debe estar indicado entre paréntesis. Esto podría afectar a la legibilidad, además de romper el código si un método se modifica posteriormente para que devuelva por referencia.

Uso de Funciones y Métodos

Los argumentos de la función tendrían que estar separados por un único espacio posterior después del delimitador coma. A continuación se muestra un ejemplo de una invocación admisible de una función que recibe tres argumentos:

```
threeArguments(1, 2, 3);
```

La llamada por referencia está estrictamente prohibida. Vea la sección de declaraciones de funciones para el método correcto de pasar argumentos por referencia.

Sentencias de Control

If/Else/Elseif

Las sentencias de control basadas en las construcciones *if* y *elseif* deben tener un solo espacio en blanco antes del paréntesis de apertura del condicional y un solo espacio en blanco después del paréntesis de cierre. Dentro de las sentencias condicionales entre paréntesis, los operadores deben separarse con espacios, por legibilidad. Se aconseja el uso de paréntesis internos para mejorar la agrupación lógica en expresiones condicionales más largas. La llave de apertura "{" se escribe en la misma línea que la sentencia condicional. La llave de cierre "}" se escribe siempre en su propia línea. Cualquier contenido dentro de las llaves debe separarse con cuatro espacios en blanco.

```
if ($a != 2) {  
    $a = 2;  
}
```

Para las declaraciones "if" que incluyan "elseif" o "else", las convenciones de formato son similares a la construcción "if". Los ejemplos siguientes demuestran el formato correcto para declaraciones "if" con construcciones "else" y/o "elseif":

```
if ($a != 2) {  
    $a = 2;  
} else {  
    $a = 7;  
}
```

```
if ($a != 2) {  
    $a = 2;  
} elseif ($a == 3) {  
    $a = 4;  
} else {  
    $a = 7;  
}
```

PHP permite escribir sentencias sin llaves -{}- en algunas circunstancias. Este estándar de código no hace ninguna diferenciación- toda sentencia "if", "elseif" o "else" debe usar llaves.