



Subsistema de recomendación de información para el buscador cubano Orión

Trabajo de diploma para optar por el título de ingeniero en
ciencias informáticas

Autor: Leydis Arzoris Mata Arias

Tutores: Ing. Martha Luisa Gala Rodríguez

Ing. Eric Bárbaro Utrera Sust

La Habana, junio 2017



"Aprende como si fueras a vivir toda la vida y vive como si fueras a morir mañana".

Charles Chaplin

Declaración de Autoría

Declaro por este medio que yo Leydis Arzoris Mata Arias, con CI: 92042228254 soy el autor principal del trabajo titulado “Subsistema de recomendación de información para el buscador cubano Orión” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

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

Firma del Autor
Leydis Arzoris Mata Arias

Firma del Tutor
Ing. Eric Bárbaro Utrera Sust

Firma del Tutor
Ing. Martha Luisa Gala Rodríguez

AGRADECIMIENTOS

En cierto momento de nuestra vida universitaria pudimos haber llegado a mirar la tesis como algo lejano, algo que se demoraría un buen tiempo en llegar y que todavía había esperanza para meditar y prepararse. Pero lo cierto es que no importa cuánto nos preparemos siempre nos toma por sorpresa.

Al referirnos a la tesis, hay que conocer que nos marca considerablemente y después del arduo trabajo que esto precisa, nos damos cuenta de que llega la retribución y el placer de saber que concluimos lo complicado con éxito. Sé que en estos casos las palabras sobran, sin embargo, a las personas que de alguna manera contribuyeron a que este sueño fuera posible, les hago llegar mi más sincero agradecimiento.

A mis angelitos: mis padres Angela y Angel por apoyarme en todo momento, por los valores que me han inculcado, y por sus incontables sacrificios para darme la oportunidad de tener una excelente educación en el transcurso de mi vida. Sobre todo, por ser un excelente ejemplo de vida a seguir.

A Ale: no fue sencillo culminar con éxito este proyecto sin embargo, estuviste a mi lado en los momentos más tormentosos. Me ayudaste hasta donde te era posible, incluso más que eso.

A mis tutores Martha Luisa y Eric Bárbaro. Por haber compartido conmigo sus conocimientos. Les agradezco su confianza, apoyo y dedicación.

A mis compañeros de aula por todos los momentos que pasamos juntos y por todas las veces que aguantamos regaños como grupo. En especial a mis amigos Osmel, Ortelio y a Feitó y Leonar, que aunque no siempre fueron el mejor ejemplo a seguir, nunca olvidare nuestras risas.

A mis amigo del alma Yasser y Carla. Gracias por haberme apoyado en las buenas y en las malas, sobre todo por su paciencia y amor incondicional.

A todos los freakys, doteros, woweros y etcs que han hecho de mi etapa universitaria un trayecto de vivencias que nunca olvidaré.

A todos los presentes: Muchas gracias.

RESUMEN

Actualmente internet supone una de las principales fuentes de información que existe. Por este motivo es que en los últimos tiempos se han desarrollado alternativas para la recuperación de esta información como es el caso del buscador cubano Orión. En la presente investigación se desarrolla un Subsistema de recomendación de información con el objetivo de mejorar la calidad en los resultados mostrados a los usuarios por este buscador, teniendo en cuenta sus preferencias y necesidades individuales. Se realizó un análisis de Sistemas de Recomendación de Información usados a nivel mundial. La metodología que sirvió de guía en el proceso de desarrollo fue AUP-UCI. Como principales tecnologías empleadas se encuentran Solr como mecanismo de indexación, Symfony como marco de trabajo PHP, PHP como lenguaje de programación, NGINX como servidor de aplicación y VisualParadigm como herramienta para el modelado.

Palabras clave: buscador, sistemas de recomendación de información, sistemas de recuperación, *web*.

ÍNDICE DE CONTENIDO

CAPÍTULO 1. Fundamentación teórica asociada a los sistemas de recomendación de información y aplicaciones web	7
1.1 Fundamentos teóricos asociados al dominio del problema	7
1.1.1 Recomendación de información	7
1.1.2 Sistemas de recomendación de información	7
1.1.3 Diferencias entre motores de búsqueda y sistemas de recomendación	9
1.1.4 Tipos de sistemas de recomendación	9
1.1.5 Algoritmos utilizados en la recomendación de información de sistemas híbridos	11
1.1.6 Grado de personalización	11
1.2 Estudio de herramientas existentes para la recomendación de información	12
1.3 Propuesta de solución	15
1.4 Herramientas, lenguajes de programación y tecnologías	18
1.4.1 Lenguaje de programación	18
1.4.2 <i>Frameworks</i> de desarrollo	18
1.4.3 Metodología de desarrollo	19
1.5 Conclusiones parciales	23
CAPÍTULO 2. Diseño del Subsistema de recomendación de información	24
Introducción	24
2.1 Modelo de dominio	24
2.1.1 Descripción de clases del Modelo de Dominio	24
2.1.2 Diagrama de clases de Modelo de Dominio	25
2.2 Especificación de los Requisitos del <i>Software</i>	25
2.2.1 Requisitos funcionales	26
2.3 Historias de Usuario	27
2.4 Estilo arquitectónico	31

2.5 Patrones de diseño _____	31
2.6 Diagrama de despliegue _____	35
CAPÍTULO 3. Implementación y validación del Subsistema de recomendación de información _____	37
Introducción _____	37
3.1 Modelo de componentes del Subsistema de recomendación de información _____	37
3.2 Diagrama de componentes _____	37
3.3 Validación del Subsistema de recomendación de información _____	38
3.3.1 Pruebas funcionales _____	39
3.3.2 Pruebas de Integración _____	42
3.3.3 Pruebas de carga y estrés _____	42
3.3.4 Pruebas de seguridad _____	44
3.4 Validación de la Hipótesis de la investigación _____	45
3.5 Conclusiones parciales _____	47
CONCLUSIONES _____	49
RECOMENDACIONES _____	50
REFERENCIAS _____	51

ÍNDICE DE TABLAS

Tabla 1. Operacionalización de las variables de la Hipótesis (Fuente: Elaboración propia).....	4
Tabla 2. Comparación entre SR (Fuente: Elaboración propia).....	14
Tabla 3. Algoritmos para el Filtrado colaborativo (Fuente: Pepa Sofía, 2014).....	15
Tabla 4. Descripción de las clases del Modelo de Dominio (Fuente: Elaboración propia).....	24
Tabla 5. Requisitos funcionales (Fuente: Elaboración propia)	26
Tabla 6. Tabla de Historia de usuario #1 (Fuente: Elaboración propia).....	27
Tabla 7. Tabla de Historia de usuario #2 (Fuente: Elaboración propia).....	28
Tabla 8. Tabla de Historia de usuario #3 (Fuente: Elaboración propia).....	28
Tabla 9. Tabla de Historia de usuario #4 (Fuente: Elaboración propia).....	28
Tabla 10. Tabla de Historia de usuario #5 (Fuente: Elaboración propia).....	29
Tabla 11. Tabla de Historia de usuario #6 (Fuente: Elaboración propia)	29
Tabla 12. Tabla de Historia de usuario #7 (Fuente: Elaboración propia).....	29
Tabla 13. Tabla de Historia de usuario #8 (Fuente: Elaboración propia).....	30
Tabla 14. Tabla de Historia de usuario #9 (Fuente: Elaboración propia).....	30
Tabla 15. Tabla de Historia de usuario #10 (Fuente: Elaboración propia).....	30
Tabla 16. Relación de asignación de responsabilidades (Fuente: Elaboración propia).....	32
Tabla 17. Cantidad de no conformidades por cada iteración de las pruebas funcionales (Fuente: Elaboración propia)	39
Tabla 18. Caso de prueba Generar recomendación (Fuente: Elaboración propia)	40
Tabla 19. Caso de prueba Similitud entre perfiles (Fuente: Elaboración propia).....	41
Tabla 20. Caso de prueba Similitud entre perfiles (Fuente: Elaboración propia).....	41
Tabla 21. Vulnerabilidades del entorno (Fuente: Elaboración propia).....	44
Tabla 22. Vulnerabilidades del sistema (Fuente: Elaboración propia).....	44
Tabla 23. Resultados de la evaluación del indicador Precisión (Fuente: Elaboración propia).....	45
Tabla 24. Cuadro lógico de IADOV (Fuente: Elaboración propia).....	46
Tabla 25. Índices de satisfacción (Fuente: Elaboración propia).....	47
Tabla 26. Caso de prueba Eliminar Recomendación (Fuente: Elaboración propia)	57
Tabla 27. Caso de prueba Seleccionar Categoría (Fuente: Elaboración propia)	57
Tabla 28. Caso de prueba Desmarcar Categoría (Fuente: Elaboración propia).....	57

Tabla 29. Caso de prueba Autenticar usuario (Fuente: Elaboración propia)	58
Tabla 30. Caso de prueba Listar por categoría (Fuente: Elaboración propia)	58
Tabla 31. Caso de prueba Mostrar por fecha (Fuente: Elaboración propia).....	59
Tabla 32. Caso de prueba Mostrar por orden alfabético (Fuente: Elaboración propia)	59

ÍNDICE DE FIGURAS

Figura 1. Comportamiento del tráfico de internet según Internet Live Stats.....	1
Figura 2. Esquema del proceso de generación de una recomendación	8
Figura 3. Diagrama de clases del Modelo de Dominio (Fuente: Elaboración propia).....	25
Figura 4. Arquitectura MVC (Fuente: Elaboración propia)	31
Figura 5. Uso del patrón Método de Fábrica en la clase Recomendacion.php (Fuente: Elaboración propia)	34
Figura 6. Uso del patrón Decorador en la vista Recomendacion/Index.html.twig (Fuente: Elaboración propia)	34
Figura 7. Diagrama de despliegue (Fuente: Elaboración propia).....	35
Figura 8. Diagrama de componentes (Fuente: Elaboración propia).....	38
Figura 9. Comportamiento de las no conformidades de las pruebas funcionales ejecutadas (Fuente: Elaboración propia).....	39
Figura 10. Pruebas de rendimiento a las funcionalidades filtrar por categoría, seleccionar nueva categoría y desmarcar categoría (Fuente: Elaboración propia)	43
Figura 11. Pruebas de rendimiento a la funcionalidad generar recomendación (Fuente: Elaboración propia)	43
Figura 12. Prototipo funcional #1 (Fuente: Elaboración propia).....	55
Figura 13. Prototipo funcional #2 (Fuente: Elaboración propia).....	55
Figura 14. Prototipo funcional #3 (Fuente: Elaboración propia).....	56

INTRODUCCIÓN

En la actualidad, es incuestionable el importante papel que supone internet como poderoso instrumento de información y comunicación para los seres humanos. Prueba de ello es que cada segundo, aproximadamente 6.000 *tweets* son twiteados, se buscan más de 40.000 consultas en Google y más de 2 millones de correos electrónicos son enviados; como se muestra en la figura 1 de acuerdo con (*Internet Live Stats*, 2016). Es conocido además que, de los 7.395 millones de habitantes del planeta, 3.419 millones tienen acceso a internet (con un incremento del 10% en un año) (*We Are Social*, 2016).

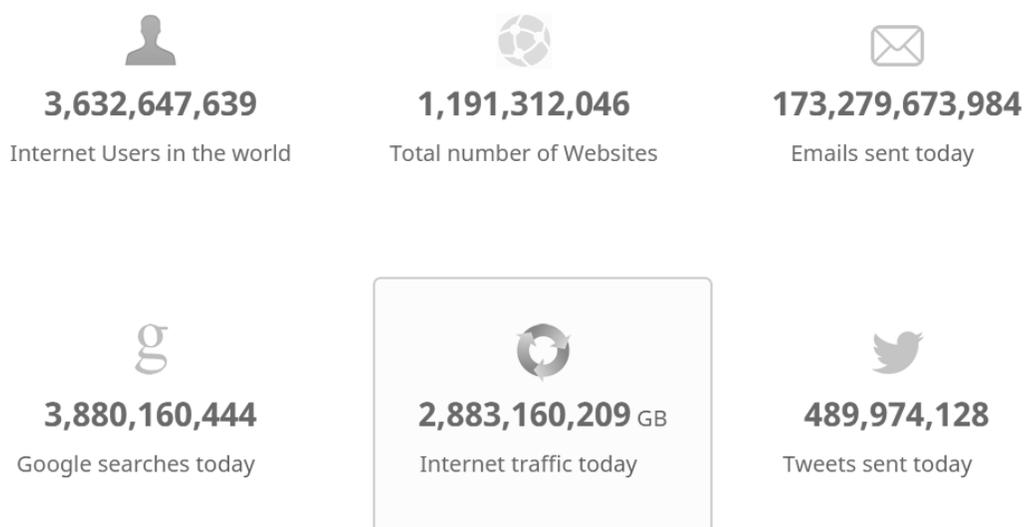


Figura 1. Comportamiento del tráfico de internet según Internet Live Stats (Fuente: Internet Live Stats, 2016)

Ante semejante volumen de información y usuarios, uno de los retos que tiene la informática en la actualidad es precisamente el poder representar, almacenar y organizar la mayor cantidad de información digital posible, ya que a menudo es preciso tomar decisiones sin tener la experiencia suficiente. Esta falta de experiencia o desconocimiento sobre las alternativas, hacen que cada día cobre más importancia el hecho de tener métodos automáticos o semiautomáticos de filtrado y/o selección de información para la toma de decisiones. Es por esto que surgen los Sistemas de Recuperación de Información (SRI).

Los SRI son herramientas que se encargan de encontrar y mostrar documentos relevantes que satisfagan la necesidad de información de un usuario, expresada en un determinado lenguaje de consulta (Giordanino, 2010). Esto consiste principalmente en la determinación de qué documentos de una colección contienen las

palabras claves de una pregunta realizada por el usuario para satisfacer su necesidad de información. La herramienta, luego de evaluar los documentos, muestra al usuario la respuestas a su petición. Estos documentos de salida aparecerán ordenados según su relevancia, decrecientemente.

Como se refiere anteriormente, la necesidad de información de los usuarios del sistema debe ser planteada al SRI en forma de *query*; o lo que es lo mismo consulta. Sin embargo, esta forma de expresar las necesidades puede suponer un importante factor que atente contra los resultados en las búsquedas. Si no se domina el lenguaje de consulta de estos sistemas, se tendrán pocas posibilidades de encontrar la información que se precisa debido a la compatibilidad que deben poseer tanto la expresión de las necesidades de información, el lenguaje y los documentos de la colección (Giordanino, 2010). Para contribuir a la solución de estos problemas y debido a que en muchos casos las personas no poseen un conocimiento basto del tema o simplemente tienen una idea vaga de cómo buscar lo que desean, los investigadores del campo de la recuperación de información, han desarrollado métodos, técnicas y herramientas para mejorar los resultados de las búsquedas realizadas por los usuarios. Una de las soluciones propuestas son los Sistemas de Recomendación (SR).

Los SR son herramientas de *software* inteligentes que estimulan los intereses o preferencias de los consumidores individuales. Tienen el potencial para apoyar y mejorar la calidad de las decisiones de los consumidores a la hora de buscar y seleccionar una determinada información tras realizar un “estudio” de su perfil¹, sus gustos e incluso de la forma en la que el usuario navega por internet. Se puede afirmar entonces que un SR no es sino un tipo específico de filtro de información cuyo objetivo es mostrar *items*; libros, artículos, películas, imágenes, *web-sites*, entre otros de interés para el usuario (Seguido, 2009).

En el ámbito nacional, la Universidad de Ciencias Informáticas (UCI), formadora de profesionales y productora de aplicaciones y servicios informáticos, en el año 2010 se dio a la tarea de desarrollar un motor de búsqueda que indexara la *web* cubana. Este motor de búsqueda llamado Orión, es un producto de *software* perteneciente al Centro de Ideoinformática (CIDI) de la Facultad 1.

El estudio realizado a los componentes de Orión permitió detectar problemas de precisión, los cuales se muestran a continuación.

- 1- Evaluación de los documentos con respecto a una consulta para establecer la relevancia de cada uno:

¹ Conjunto de datos que se refieren al usuario de un servicio informático.

El sistema solo comprueba la ocurrencia de una palabra entre la consulta escrita por el usuario y los documentos indexados en Solr para calcular la relevancia de la información. Debido a esto y que tampoco tiene en cuenta el factor humano para satisfacer las necesidades individuales de los usuarios, se incrementa la posibilidad de que los resultados mostrados no posea la relevancia necesaria.

2- Presentación de la necesidad de información:

Es muy frecuente que los usuarios al interactuar con Orión, no hagan un planteamiento correcto de lo que están buscando, debido a que en muchos casos no poseen una idea clara del tema en cuestión. Es por ello que puede pasar que no consulten los documentos que mejor se adaptan a sus necesidades.

3- Presentación de los documentos en el buscador:

Orión muestra contenido diverso que no se brinda a los usuarios atendiendo a sus preferencias individuales. A menudo es necesario seleccionar una entre varias alternativas sin tener un conocimiento exacto de cada una de ellas. En este sentido, Orión no hace sugerencia de búsquedas, es decir, no es capaz de asistir a los usuarios atendiendo a sus preferencias. Por este motivo el usuario es quien debe tomar la iniciativa, lo que fomenta la posibilidad de que puedan quedar datos importantes fuera de los resultados mostrados.

Por tanto, se puede decir que existen insuficiencias para potenciar las preferencias individuales de los usuarios, pues no se identifican sus necesidades y preferencias relacionadas con la información disponible en Orión.

Teniendo en cuenta la problemática anterior, se plantea como **problema de la investigación**: ¿Cómo mejorar la calidad² en los resultados brindados al usuario en el buscador Orión? Para la realización de la investigación se definen como **objeto de estudio** los sistemas de recomendación de información; enfocada en los sistemas de recomendación de información para buscadores como **campo de acción**.

Para dar solución al problema planteado, se define como **objetivo general**: Desarrollar un Subsistema de recomendación de información utilizando técnicas de recomendación de información para mejorar la calidad de los resultados brindados al usuario en el buscador Orión.

Para lograr el cumplimiento del objetivo general, se desglosan los siguientes **objetivos específicos**:

1. Caracterizar el estado del arte en relación a los sistemas de recomendación de información.

² Capacidad que posee un objeto para satisfacer necesidades implícitas o explícitas.

2. Definir las tecnologías, lenguajes, herramientas y la metodología de desarrollo para la implementación del sistema de recomendación de información para el buscador Orión.
3. Diseñar el Subsistema de recomendación de información para el buscador Orión.
4. Implementar el Subsistema de recomendación de información para el buscador Orión.
5. Validar el Subsistema de recomendación de información para el buscador Orión.

Para la presente investigación se formula la siguiente **hipótesis**: El desarrollo de un Subsistema de recomendación de información utilizando técnicas de recomendación de información mejorará la calidad de los resultados brindados a los usuarios en el buscador Orión.

Se define como **variable independiente**: El Subsistema de recomendación de información utilizando técnicas de recomendación de información.

Como **variable dependiente** se especifica: Calidad de los resultados brindados al usuario en el buscador Orión.

Tabla 1. Operacionalización de las variables de la Hipótesis (Fuente: Elaboración propia)

Variables	Descripción	Dimensiones	Indicadores	Unidad de Medida
Variable independiente	Subsistema de recomendación de información.	Recomendaciones	Cantidad de recomendaciones relevantes	Unidad
Variable dependiente	Calidad de los resultados brindados al usuario en el buscador Orión.	Calidad de las recomendaciones	Precisión	Por ciento

Para dar cumplimiento a los objetivos específicos se definieron las siguientes **tareas de investigación**:

1. Realización de un estudio del estado del arte de los sistemas de recomendación de información.
2. Selección de las tecnologías, lenguajes, herramientas y metodología que se necesitan para implementar el Subsistema de recomendación de información.

3. Elaboración de los artefactos de ingeniería de *software* requeridos por la metodología de desarrollo seleccionada.
4. Implementación del Subsistema de recomendación de información.
5. Realización y documentación de las pruebas realizadas para validar el Subsistema de recomendación de información desarrollado.

En el desarrollo de la investigación se utilizaron los siguientes métodos de investigación:

Métodos teóricos

- **Histórico-Lógico:** Para constatar teóricamente cómo han evolucionado en el tiempo los SR y las herramientas y tecnologías utilizadas en su desarrollo.
- **Analítico-Sintético:** Se utiliza para el análisis de los principales elementos referentes a las teorías, documentos y literatura relacionada con los SRI y los SR.
- **Modelación:** Permite la reproducción simplificada de la realidad del objeto que se está estudiando (en este caso los SR) para descubrir nuevas relaciones y cualidades del mismo.

Métodos empíricos:

- **Entrevista:** Permite a través de una serie de preguntas al jefe del proyecto Orión y demás trabajadores del departamento conocer las características y deficiencias que presenta el buscador en cuestión para determinar los cambios que deben realizarse que posibiliten mejorar resultados brindados a los usuarios.
- **Encuesta:** Permite conocer a través de un formulario el nivel de satisfacción de los usuarios potenciales con el Subsistema a implementar para la posterior validación de su calidad.

Estructura del documento

Capítulo 1. Fundamentación teórica asociada a los sistemas de recomendación de información y aplicaciones web: Aborda los conceptos que apoyan a la investigación. Se estudiarán además los principales sistemas de recomendación, fundamentalmente para buscadores, así como sus clasificaciones con el fin de brindar una solución a la problemática planteada. Finalmente se expondrán las distintas tecnologías a emplear en el desarrollo del Subsistema de recomendación de información, así como la metodología de desarrollo de *software* a utilizar.

Capítulo 2. Diseño del Subsistema de recomendación de información: Se expondrán las características del Subsistema de recomendación de información, sus requisitos funcionales y no funcionales, patrones de diseño y arquitectura utilizados, así como artefactos que requiera la metodología de desarrollo de *software* utilizada para brindar una mejor comprensión de lo que se desea implementar.

Capítulo 3. Implementación y validación del Subsistema de recomendación de información: Se definen una serie de estándares de codificación que debe seguir la implementación del Subsistema de recomendación de información. También se presentan los diseños de casos de pruebas para su validación, así como los resultados de las mismas, permitiendo así la evaluación de la calidad de la propuesta de solución.

CAPÍTULO 1. Fundamentación teórica asociada a los sistemas de recomendación de información y aplicaciones web

Introducción

En este capítulo se exponen los principales elementos teóricos que respaldan la presente investigación. Para ello se realiza un exhaustivo estudio acerca de los elementos y áreas del conocimiento que abarcan el objeto de estudio y el campo de acción. Además, se exponen las herramientas, tecnologías y la metodología de desarrollo de *software* a emplear.

1.1 Fundamentos teóricos asociados al dominio del problema

El término “información” se refiere a la comunicación o adquisición de conocimientos que permiten ampliar o precisar los conocimientos que se poseen. Si algo es cierto, es que en todas las épocas la información ha ocupado un lugar predominante hasta llegar a ser vital para cualquier actividad del hombre (Real Academia Española).

1.1.1 Recomendación de información

Con la información el hombre es capaz de intercambiar experiencias o hacer propuestas a otros a partir de lo que ya conoce. Estas propuestas a menudo son asociadas al término “recomendar”, el cual además de ser el acto de comunicar o sugerir a otras personas, constituye autoridad, representación o calidad por la que algo se hace más apreciable y digno de respeto (Real Academia Española). Entonces la recomendación de información no es más que atribuir cierta importancia a los datos tras ser recomendados.

1.1.2 Sistemas de recomendación de información

Con el paso de los años, el uso y la generación de información han crecido a gran ritmo, debido al aumento significativo de las instituciones dedicadas a la investigación y al desarrollo de nuevos horizontes tecnológicos. Es por ello que muchos estudios proponen automatizar la gestión de la información en las organizaciones e instituciones que antes se realizaba de forma manual. Tal es el caso del estudio que proponen (López y Zorita, 2008).

Debido al creciente avance tecnológico y la informatización de la información, a principios de la última década del siglo pasado los sistemas informáticos de recomendación comenzaron a ser objeto de investigación, haciéndose hasta la actualidad aplicaciones de uso cotidiano de cualquier consumidor de

tecnologías de la información. Estos van desde sistemas sofisticados y predictivos, hasta otros menos rebuscados que solo hacen una lista fija de sugerencias. (Delegorreta, 2015).

Los sistemas de recomendación, son herramientas y técnicas de *software* que proporcionan sugerencias de aquellas publicaciones en la *web* que son más probable de interés para un usuario. Como se presenta en la figura 2, las sugerencias que realizan estos sistemas están dirigidas a apoyar a sus usuarios en la toma de decisiones, tales como qué artículos comprar, qué música escuchar, o qué noticias leer. Además, son valiosos medios para que los usuarios en línea puedan hacer frente a la sobrecarga de información y ayudarles a tomar mejores decisiones (Ricci Francesco et al, 2015; de Campos Luis et al., 2011).

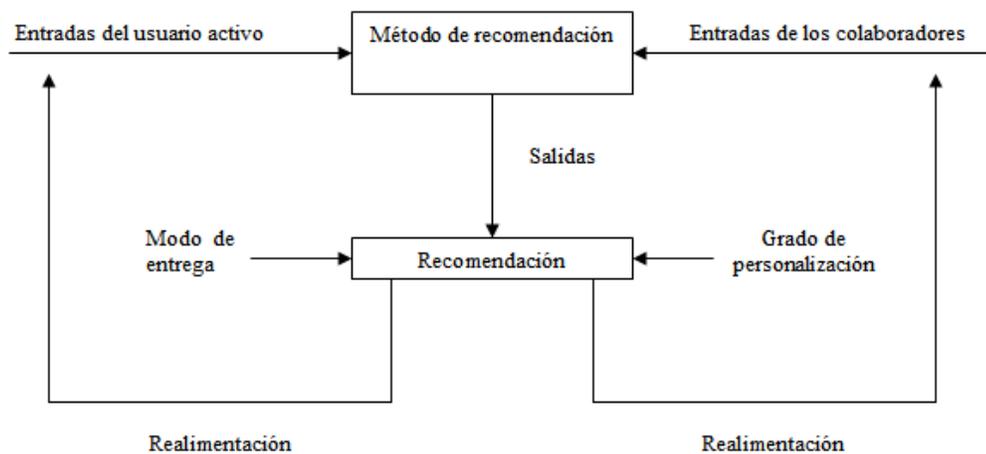


Figura 2. Esquema del proceso de generación de una recomendación (Fuente: Delegorreta, 2015)

Los sistemas de recomendación son capaces de recomendar información a usuarios individuales sobre la base de búsquedas y comportamientos de otros usuarios. Se distinguen por dos criterios fundamentales: Por un lado se encuentran aquellos que producen recomendaciones individualizadas en forma de “salidas” y, por otro, aquellos que tienen el efecto de guiar al usuario de forma personalizada para los intereses de éste, dentro de un dominio con grandes cantidades de *items*³ posibles a elegir. Cual sea el caso, se puede decir que los sistemas de recomendación funcionan como filtros que eliminan información inadecuada o no deseada de un flujo total de forma automática o semiautomática para ser presentada posteriormente a un usuario (Seguido, 2009).

³ cada una de las partes individuales que conforman un conjunto.

1.1.3 Diferencias entre motores de búsqueda y sistemas de recomendación

No todo hecho de proporcionar información a un usuario es necesariamente recomendación de información. Un aspecto relacionado con la recomendación es la “recuperación de información” o “búsqueda de información” que es llevada a cabo por los conocidos motores de búsqueda. Los sistemas de recomendación van un poco más allá. Según (Seguido, 2009), aunque ambos conceptos se asemejan, se distinguen por ciertas singularidades como:

- Frecuencia de uso: Los sistemas de búsqueda son enfocados por uso concreto y puntual del usuario mientras que los sistemas de recomendación están diseñados para un uso prolongado y de forma repetitiva.
- Representación de las necesidades de información: En sistemas de búsqueda la información requerida se expresa en forma de pregunta (*query*) mientras que en los sistemas de recomendación la información es descrita en los perfiles del usuario.
- Objetivo: Los sistemas de búsqueda seleccionan los *items* de la base de datos que coinciden con la pregunta (*query*) mientras que los sistemas de recomendación eliminan la información irrelevante de flujos de entrada de información o reúnen información relevante de diferentes repositorios de acuerdo al perfil del usuario.
- Base de datos: Los sistemas de búsqueda trabajan con bases de datos relativamente estáticas mientras que los sistemas de recomendación trabajan con información dinámica.
- Tipo de usuarios: En los sistemas de búsqueda no es necesario tener información sobre los usuarios que lo utilizan mientras que en los sistemas de recomendación se necesita saber o tener información sobre los usuarios.

1.1.4 Tipos de sistemas de recomendación

Los sistemas de recomendación se pueden clasificar atendiendo al tipo de información que utilizan en la realización de recomendaciones. Tradicionalmente son clasificados como: Basados en contenido, Filtrado colaborativo e híbridos (Jones, 2013; Silvi Luciano, 2016; Sosa Vázquez, 2013; Koren Yehuda y Bell Robert, 2016).

Filtrado colaborativo: el modelo es construido a partir del comportamiento de uno o más usuarios con rasgos similares, es decir, crear un modelo de comportamiento anterior del usuario para emitir

recomendaciones. En esencia, las recomendaciones son una colaboración automática de múltiples usuarios y se filtran en aquellos que exhiben similares preferencias o comportamientos.

Existe una serie de problemas que suelen afrontar los sistemas que utilizan Filtrado Colaborativo, destacándose los siguientes:

- **Arranque en frío (*cold start*):** Al no haber un historial anterior con el cual comparar, ante la aparición de un usuario o recurso nuevo, se dificulta la tarea de encontrar resultados relevantes.
- **Escalabilidad:** Cuando el sistema alcanza grandes dimensiones, parte de las metodologías y algoritmos utilizados con frecuencia en esta clase de sistemas se vuelven lentos y difíciles de aplicar. En estos casos es necesario adaptar las técnicas para lograr la rapidez de respuesta necesaria.
- **Dispersidad:** Los *items* en el sistema generalmente superan a la cantidad de usuarios, provocando que incluso los elementos más populares sólo hayan sido calificados por un pequeño grupo de usuarios, lo que puede dificultar la tarea de encontrar semejanzas entre usuarios. Este problema puede ser evitado utilizando Filtrado Colaborativo basado en recursos, donde, en lugar de buscar similitudes entre usuarios, se lo hace entre recursos, puesto que hay más cantidad de calificaciones por recursos que por usuario.

Basado en contenidos: este filtrado construye una recomendación sobre la base del comportamiento de un usuario. Este enfoque puede utilizar información de navegación histórica, como los *blogs* que el usuario lee, páginas *web* que visita y sus características y utiliza este historial para identificar y recomendar contenidos similares. Este contenido se puede definir manualmente o extraer automáticamente basándose en métodos de similitud.

La principal desventaja en esta clase de implementaciones es que, debido a su naturaleza, no se incorpora la posibilidad de sugerir elementos que, a pesar de no concordar con las preferencias específicas del usuario, pueden interesarle.

Híbridos: los enfoques híbridos son aquellos que combinan el filtrado colaborativo y basado en contenido aumentando así la eficiencia (y complejidad) de los sistemas de recomendación. La incorporación de los resultados del filtrado colaborativo y basado en el contenido crea el potencial para una recomendación más precisa. El enfoque híbrido también podría utilizarse para abordar el filtrado colaborativo que comienza con datos escasos, conocidos como arranque en frío, permitiendo que los resultados se ponderen inicialmente hacia el filtrado basado en contenido y, a continuación, cambien el peso hacia el filtrado colaborativo cuando el conjunto de datos de usuario disponible madure.

En el caso de sistemas híbridos comerciales, se añaden además heurísticas propias de cada proveedor, tanto por razones logísticas como de marketing (por ejemplo, promocionar un determinado producto). Esto posibilita, por ejemplo, integrar un nuevo producto al sistema al ser sugerido y calificado por la mayor cantidad de usuarios posible.

1.1.5 Algoritmos utilizados en la recomendación de información de sistemas híbridos

Existen muchos enfoques algorítmicos para los sistemas de recomendación. Puesto que las soluciones híbridas permiten disminuir el problema del arranque en frío propio del filtrado colaborativo, o la recomendación de elementos sin descripción, propio de los algoritmos basados en contenido, se escoge este tipo de sistema para la presente investigación. A continuación, se explican los principales algoritmos para este tipo de sistemas (Pepa Sofía, 2014):

Algoritmos para sistemas híbridos

- Monolítico (favoreciendo ciertas características): en este caso no se realiza ninguna combinación de resultados o algoritmos, sino que la combinación realizada es virtual. En otras palabras, se realiza una fase de pre-procesamiento para combinar el contenido de los recursos y sus correspondientes *ratings* (mediante pesos), y a partir de estos nuevos datos se ejecuta un único recomendador para generar el ranquin de recomendación.
- Paralelismo de varios sistemas: se realiza una recomendación por algoritmo y se combinan los resultados obtenidos en cada uno de ellos, posibilitando establecer diferentes pesos como en el caso anterior y dando lugar a un único ranquin de *items* para cada usuario objetivo.
- Implementación segmentada: un sistema de recomendación pre-procesa el *input* del siguiente algoritmo a ejecutar. Por ejemplo, el primer algoritmo excluye los *items* con puntuación nula, y el siguiente realiza el ranquin descartando los *items* excluidos por el primero.

1.1.6 Grado de personalización

Según su grado de personalización, también se pueden clasificar los Sistemas de Recomendación (SR) (Herrera Enrique et al., 2004):

- Cuando los SR proporcionan las mismas recomendaciones a todos los usuarios, son clasificados en este ámbito como no personalizados. Dichas recomendaciones estarán basadas en selecciones manuales, resúmenes estadísticos u otras técnicas similares.

- Los SR que tienen en cuenta la información actual del usuario que es objeto de las recomendaciones, proporcionan personalización efímera, puesto que las recomendaciones son repuestas al comportamiento y acciones del usuario en su sesión actual de navegación.
- Los SR que ofrecen el mayor grado de personalización son los que utilizan personalización persistente, ofreciendo recomendaciones diferentes para distintos usuarios, incluso cuando estén buscando el mismo *item*. Estos sistemas están basados en el perfil de los usuarios, por lo que hacen uso de métodos de filtrado colaborativo, filtrado basado en contenidos o correlaciones entre *items*.

1.2 Estudio de herramientas existentes para la recomendación de información

Son diversas las aplicaciones *web* que permiten la recomendación de información a usuarios atendiendo a diferentes criterios. A continuación, se analizan algunos de los SR más reconocidos como YouTube, Google +1 y el sistema de recomendación de Netflix.

Sistema de recomendación de YouTube

YouTube es un servicio gratuito de acceso compartido a vídeos en internet. Se ha convertido en el más grande, importante y principal canal de comunicación y promoción de videos, y en uno de los iconos de referencia de la *web* social. Este portal fue fundado en San Bruno, California por tres jóvenes Chad Hurley, Steve Chen y Jawed Karim, en febrero de 2005. Youtube nació de una necesidad de sus fundadores: intercambiar con facilidad entre amigos videos grabados durante un cumpleaños, y de compartirlo con el mayor número de personas posible.

Entre las funciones que realiza el sistema de recomendación de Youtube se puede encontrar (Md, 2013):

- Tiene en cuenta las preferencias de los usuarios para hacer la recomendación.
- Emplea la recomendación personalizada.
- Establece relaciones de gustos similares entre los usuarios.
- Además de la personalización se utiliza la regionalización (recomendar criterios basados en el área geográfica).
- Posee una vista donde se muestran y gestionan las recomendaciones.

Sistema de recomendación de Netflix

Netflix, es un servicio mensual de televisión online, actualmente cuenta con más de 60 millones de usuarios. Se trata de una plataforma de televisión online donde el usuario o cliente puede elegir qué ver y cuándo hacerlo.

El algoritmo que implementa Netflix comprende los siguientes elementos (Clarín, 2016) y (Netflix 2012):

- El sistema es capaz de detectar dónde vive el usuario, y de dónde extraer los datos, si del mundo o de su región.
- Establece diferencias entre catálogos.
- Tiene en cuenta la valoración que los usuarios hayan hecho de los contenidos consumidos.
- Busca similitudes entre los contenidos.
- Analiza las preferencias y gustos de los usuarios.
- Presenta recomendación personalizada.

Sistema de recomendación de Google+1

Google Inc. es una empresa multinacional estadounidense especializada en productos y servicios relacionados con Internet, *software*, dispositivos electrónicos y otras tecnologías (Google Inc., 2013). Recientemente está introduciendo una nueva característica en los resultados de búsqueda del buscador de Google: +1. Este +1 es un botón que el usuario puede pulsar en caso de que el resultado de búsqueda le sea útil o le parezca que puede ser de interés o utilidad para otros usuarios.

Esta característica comprende (GIZMODO, 2011):

- Para acceder a esta característica el usuario debe estar registrado.
- Presenta recomendación personalizada.
- Cuenta con un icono para dar un sello de aprobación a la publicación.
- Vista con las publicaciones a las que tus amigos han pulsado +1.
- También puedes añadir anuncios de +1, que junto con los datos de tus + 1s de resultados de búsqueda habituales, permitirán a Google aprender más sobre ti y orientar mejor sus anuncios.
- Utiliza el filtrado colaborativo para generar recomendaciones personalizadas para los usuarios de Google.
- Combina las recomendaciones de diferentes algoritmos usando un modelo lineal

Orión

El Motor de Búsqueda Orión es un sistema de recuperación de información que es capaz de realizar búsquedas sobre contenidos publicados en la *Web* cubana mediante una interfaz amigable e intuitiva. Este sistema solo brinda las funcionalidades de búsqueda general y avanzada de documentos publicados en la *Web* a través de los mecanismos de rastreo, indexación y de interfaz. Las búsquedas realizadas por Orión aun no satisfacen la necesidad existente en nuestro país con respecto al acceso a la información nacional. Además, la precisión de los resultados a partir de un criterio de búsqueda introducido por el usuario no es satisfactoria.

En la tabla 2 que se muestra a continuación se comparan las características de Orión así como las de los sistemas homólogos estudiados en la presente investigación.

Tabla 2. Comparación entre SR (Fuente: Elaboración propia)

Sistemas	Recomendación personalizada	Vista con recomendaciones	Preferencias de usuarios	Aprobación de los resultados	Regionalización	Algoritmo Híbrido
YouTube	X	X	X	X	X	X
Nettflix	X	X	X	X	X	X
Google +1	X	X	X	X	X	X
Orión	-	-	-	-	-	-

La combinación del crecimiento desmesurado de la información disponible y la falta de experiencia en sectores determinados por parte del usuario, hacen evidente la incitación para la búsqueda de sistemas de recomendación capaces de ayudar a los usuarios en la toma de decisiones mediante la aportación de sugerencias o indicaciones. Sin embargo, como se pudo apreciar en la tabla 2, el buscador cubano no posee un sistema de recomendación que guíe al usuario hacia sus intereses. Además, como estas herramientas homólogas son privativas se dificulta la disponibilidad de las mismas dado el limitado acceso a internet que tiene Cuba. Por tanto, se evidencia la necesidad de contar con un Subsistema de recomendación de información que muestre a los usuarios resultados más útiles o satisfactorios para que estos a su vez puedan hacer una búsqueda más concreta o efectiva.

1.3 Propuesta de solución

Por lo antes expuesto, se determina crear un Subsistema de recomendación de información para brindar una mayor precisión en las búsquedas realizadas por los usuarios del buscador cubano Orión, potenciando así su fácil acceso a la *Web* cubana y contribuyendo al uso de las nuevas tecnologías de la información y las comunicaciones a favor del desarrollo de la economía nacional, la sociedad y el servicio al ciudadano en Cuba.

Para desarrollar la propuesta de solución se tendrán en cuenta dos tipos de sistemas, "Basado en contenido" y "filtrado colaborativo". A continuación, se muestran más detalladamente estos algoritmos.

Filtrado colaborativo: esta familia de sistemas de recomendación se caracteriza porque se utiliza información de unos usuarios para producir recomendaciones a otros. Así pues, en estos algoritmos los usuarios se benefician de la experiencia de otros usuarios. Existe una amplia diversidad de algoritmos que implementan el filtrado colaborativo. Entre las dos estrategias de filtrado colaborativo más extendidas y actualmente exitosas son, los algoritmos de vecinos próximos y la factorización de matrices (Pepa Sofía, 2014; Galán Nieto, 2007).

Tabla 3. Algoritmos para el Filtrado colaborativo (Fuente: Pepa Sofía, 2014)

Algoritmo	Fórmula
Vecinos próximos (kNN)	
Basado en usuario (User kNN)	$f(u, i) = \sum_{\substack{v \in N_k(u) \\ r(v, i) \neq \emptyset}} sim(u, v) * r(v, i)$
Basado en <i>item</i> (Item kNN)	$f(u, i) = \sum_{\substack{j \in N_k(i) \\ r(u, j) \neq \emptyset}} sim(i, j) * r(u, j)$
<i>sim(u, v)</i>	
Jaccard	$sim(u, v) = \frac{ u \cap v }{ u \cup v } = \frac{ u \cap v }{ u + v - u \cap v }$

Pearson	$sim(u, v) = \frac{\sum_{i:r(u,i) \neq \emptyset} (r(u, i) - \bar{r}_u)(r(v, i) - \bar{r}_v)}{r(v, i) \neq \emptyset}$ $\sqrt{\sum_{i:r(u,i) \neq \emptyset} (r(u, i) - \bar{r}_u)^2} \sqrt{\sum_{i:r(v,i) \neq \emptyset} (r(v, i) - \bar{r}_v)^2}$
Coseno	$sim(u, v) = \frac{\sum_{i:r(u,i) \neq \emptyset} r(u, i) * r(v, i)}{\sqrt{\sum_{i:r(u,i) \neq \emptyset} r(u, i)^2} \sqrt{\sum_{i:r(v,i) \neq \emptyset} r(v, i)^2}}$
Factorización de matrices	
<i>pLSA (probabilistic Latent Semantic Analysis)</i>	$f(u, i) = p(i u; \theta) = \sum_f p(i f) * p(f u)$
<i>SVD (Singular Value Decomposition)</i>	$r = W * S * F^T$ $f(u, i) = \bar{r}_u + W \sqrt{S^T}(u) * \sqrt{S} F^T(i)$
<i>SVDN (SVD no-empty entries)</i>	$r = U_F * I_F$ $f(u, i) = U_F(u) * I_F(i)$
<i>HSVD (SVD with Hypergraph transformation)</i>	$X_m = (R_m > 0)$ $\bar{X}_m = D_e^{\frac{1}{2}} * X_m * D_v^{\frac{1}{2}}$ $SVD(\bar{X}_m) = W * S * F^T$ $\hat{R}_{te} = \theta^T F^T$ $f(u, i) = \hat{R}_{te}(u, i)$
<i>ASVD (Assymmetric SVD)</i>	<p>Se trata de la versión asimétrica de SVD, que solo usa la tercera matriz de la descomposición para realizar las recomendaciones. En otras palabras, el algoritmo es idéntico excepto que en lugar de usar X_m se usa R_m.</p>

Basado en contenido: este grupo de algoritmos se basa en la utilización de la descripción de cada *ítem* para recomendar, sin utilizar información de otros usuarios para generar la recomendación al usuario objetivo (Pepa Sofía, 2014). A continuación, se muestran dos de los algoritmos más utilizados para implementar los sistemas Basados en Contenido.

- **Rocchio:** se basa en el cálculo de centroides para cada usuario, de forma que se obtenga un vector “representante” para cada uno. En este caso el vector representaría las preferencias de cada usuario. La fórmula para el cálculo de los centroides es la siguiente:

$$u[f] = \frac{1}{|u|} \sum_{i:r(u,i) \neq \emptyset} tfidf(f,i) * r(u,i), \text{ donde } u = \{r(u,i) \neq \emptyset | i \in \mathcal{I}\}$$

Una vez se dispone de los centroides, el cálculo de la similitud de los usuarios con cada uno de los *ítems* se puede realizar mediante Jaccard, Pearson o Coseno. En este caso aplicando coseno quedaría de la siguiente forma:

$$sim(i,j) = \frac{\sum_f u[f] * tfidf(f,i)}{\sqrt{\sum_f u[f]^2} \sqrt{\sum_f tfidf(f,i)^2}}$$

- **Item kNN:** la estructura de este algoritmo es idéntica a la de FC del mismo nombre, pero se diferencian en la forma de calcular la similitud entre los *ítems*. Mientras el de FC utiliza los ratings de otros usuarios, éste utiliza la descripción de los *ítems*.

$$sim(i,j) = \frac{\sum_f tfidf(f,i) * tfidf(f,j)}{\sqrt{\sum_f tfidf(f,i)^2} \sqrt{\sum_f tfidf(f,j)^2}}$$

Para la implementación del algoritmo del método Basado en contenido no es necesario hallar un centroide, pues se conocen las preferencias de los usuarios a través del estudio de su navegación y se posee el *ítem* categoría que está en correspondencia con las preferencias del usuario. Por lo tanto, se escoge para la presente investigación el algoritmo kNN. El método seleccionado para implementar el Filtrado Colaborativo es Basado en usuario (*User kNN*). Para determinar la similitud entre los usuarios se implementará la similitud por coseno debido a que las variables del algoritmo son las que más se ajuntan a los datos que se conocen de los usuarios. Este método medirá el ángulo entre los vectores (*ratings*) de cada par de usuarios, de forma que serán más similares aquellos cuyos vectores tengan la misma orientación.

Se concluye entonces que la solución tendrá las características de un sistema de recomendación híbrido, utilizándose los sistemas Basado en contenido y Filtrado colaborativo. Su grado de personalización será

persistente ya que será capaz de ofrecer recomendaciones diferentes para distintos usuarios, incluso cuando estén buscando el mismo *item*. Se implementará además el paralelismo de estos dos sistemas anteriormente mencionados: se realiza una recomendación por cada uno de los algoritmos que corresponden a los sistemas Basado en contenido y Filtrado colaborativo respectivamente y posteriormente se combinarán los resultados obtenidos en cada uno de ellos.

El algoritmo Basado en contenido se utilizará para determinar los documentos que más puedan interesarle a un usuario a través del estudio de su propio perfil (páginas que visita frecuentemente) y el Filtrado colaborativo para detectar usuarios con preferencias similares a este y así recomendarle documentos consultados por otros usuarios que también puedan interesarle.

1.4 Herramientas, lenguajes de programación y tecnologías

Luego del estudio de herramientas homólogas, para desarrollar la propuesta de solución, se hace necesario estudiar varias tecnologías, lenguajes y herramientas disponibles que permitan llevar a cabo el objetivo general de la investigación.

1.4.1 Lenguaje de programación

PHP: “*Hypertext Pre-Processor*” es un lenguaje de programación dirigido especialmente a la creación de páginas *web* dinámicas, es decir que su información esté en una base de datos (BD). Su código solo es ejecutado del lado del servidor, pero es capaz de incrustarse dentro del código HTML. Cabe destacar que este lenguaje, es de código abierto, gratuito y multiplataforma y presenta buena integración con la mayoría de conectores a base de datos. MySQL, PostgreSQL, Oracle, etc. Además PHP tiene mucha documentación debido a su gran popularidad y una gran comunidad internacional (*Web Oficial php.net*) que colabora en el mejoramiento del código fuente, el desarrollo y actualización de librerías para el lenguaje y la traducción de la documentación del proyecto (Fumás, 2014).

Por todo lo anterior y dada la gran variedad de módulos que presenta PHP, teniendo en cuenta que es libre, que es un lenguaje completamente expandible, se decide utilizarlo como lenguaje del lado del servidor.

1.4.2 Frameworks de desarrollo

Bootstrap: para el trabajo en la *Web* se utilizará HTML, CSS y JavaScript, todos estos integrados en el marco de trabajo Bootstrap. Este es un *framework* o conjunto de herramientas de Código abierto para diseño

de sitios y aplicaciones *web*. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML (Cochran, 2012).

Symfony2: es un *framework* diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones *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. Symfony2 ha sido ideado para exprimir al límite todas las nuevas características de PHP 5.3 y por eso es uno de los *frameworks* PHP con mejor rendimiento (Eguiluz Javier, 2014).

Symfony2 es capaz de automatizar la mayoría de los elementos comunes en los proyectos, por ejemplo: tiene un sistema de cache eficiente y la autenticación y la gestión de credenciales simplifican la creación de secciones restringidas y la gestión de la seguridad de usuario entre otros. Además, este *framework* sigue la mayoría de las mejores prácticas y patrones de diseño para la *web*, es fácil de instalar y configurar en la mayoría de las plataformas y es adaptable a las políticas y arquitecturas propias de cada empresa (Pérez Manuel, 2009).

Además de lo anterior, Symfony es el marco de trabajo en el que fue y es desarrollado lo referente al motor de búsqueda Orión, por lo cual es recomendable continuar la misma línea.

1.4.3 Metodología de desarrollo

Las metodologías de desarrollo de *software* son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos *software* (Gutierrez Demián, 2011).

Es preciso incorporar una metodología de desarrollo de *software* ya que tiene entre sus objetivos aumentar la calidad del *software* que se produce, aplicando buenas prácticas que se centran en el desarrollo de productos y servicios de calidad.

Existen varias clasificaciones para estas metodologías dentro de las cuales se destacan (Figuroa, Solís et al, 2008):

- Metodologías tradicionales: también conocidas como metodologías formales están enfocadas en una documentación, planificación y procesos (plantillas, técnicas de administración, revisiones, etc.). Esto trae consigo que la utilización de este tipo de metodología no sea una buena solución para proyectos que se desarrollen sobre entornos volátiles, ya que realizar un cambio en el proyecto

implica elevados costos. En esta clasificación se destacan RUP (*Rational Unified Process*) y MSF (*Microsoft Solution Framework*).

- Metodologías ágiles: nacen para dar respuesta a los problemas derivados de la tradicional. Se basan principalmente en los siguientes aspectos:
 - 1- Asume que los individuos y las interacciones entre ellos son más importantes que las herramientas y procesos empleados.
 - 2- Crear un producto de *software* que funcione es más importante que escribir una documentación exhaustiva.
 - 3- Debe existir una estrecha colaboración entre el cliente y los desarrolladores del producto.
 - 4- La capacidad de respuesta ante un cambio es más importante que el seguimiento estricto de un plan.

Por lo antes mencionado, para el desarrollo de este trabajo de diploma se hace uso de la metodología de desarrollo ágil AUP-UCI. Esta se basa en una variación de la metodología “Proceso Unificado Ágil” (AUP por sus siglas en inglés) en unión con las mejores prácticas propuestas por el Modelo para Procesos de Desarrollo de Software CMMI-DEV v 1.3. Esta variación de la metodología cuenta con 3 fases, las cuales se explican de forma breve a continuación (Rodríguez, 2015):

1. Inicio: durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
2. Ejecución: en esta fase se ejecutan las actividades requeridas para desarrollar el *software*, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
3. Cierre: en esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

1.4.4 Herramientas

VisualParadigm

Es una herramienta multiplataforma, que soporta el ciclo completo de desarrollo de *software*: análisis, diseño, implementación y pruebas. Facilita la construcción de aplicaciones informáticas con un menor coste que destacan por su alta calidad y contribuye a mejorar la experiencia de usuario mediante el diseño de un gran número de artefactos de ingeniería de *software*. Permite la generación de bases de datos, conversión de diagramas entidad-relación a tablas de base de datos, mapeos de objetos y relaciones, ingeniería directa e inversa, la gestión de requisitos de *software* y la modelación de procesos del negocio (VisualParadigm, 2014).

Servidor *web* NGINX

Los servidores *web*, son programas que se instalan en ordenadores que cuentan con conexión a Internet, especialmente diseñados para transferir datos de hipertexto, es decir, páginas *web* con todos sus elementos (textos, *widgets*, *baners* entre otros) mediante el protocolo http. Estos, se encuentran a la espera de que algún navegador le haga una petición (como acceder a una página *web*) y responde a la petición, enviando código HTML mediante una transferencia de datos en red.

Para la presente investigación se analizaron los servidores *web* **NGINX** y **Apache**, ya que son los dos servidores *web* de código abierto más comunes en el mundo.

NGINX: ha crecido en popularidad desde su lanzamiento debido a su uso de recursos ligeros y su capacidad de escalar fácilmente en hardware mínimo. Actualmente este servidor es conocido por su alto rendimiento, estabilidad, conjunto de características ricas, configuración sencilla y bajo consumo de recursos. (Ellingwood Justin, 2015;).

A diferencia de los servidores tradicionales, NGINX no se basa en los hilos para manejar las solicitudes. En su lugar, utiliza una arquitectura (asincrónica) basada en eventos la cual le proporciona mayor escalabilidad. Esta arquitectura utiliza cantidades pequeñas, pero más importantes, predecibles de memoria bajo carga. Incluso si no espera manejar miles de solicitudes simultáneas, todavía puede beneficiarse del alto rendimiento y de la pequeña huella de memoria de NGINX (NGINX, 2016).

Apache 2: desarrollado bajo la licencia ASF por *The Apache Software Foundation*, es considerado una de las mejores y más aceptadas creaciones del mundo del *software* libre (Netcraft, 2015). Es un servidor *web* HTTP de código abierto, multiplataforma, gratuito, muy robusto y que destaca por su seguridad y rendimiento (Fumás Cases, 2014).

Además de su capacidad para manejar contenido estático, Apache también puede procesar contenido dinámico incrustando un procesador del lenguaje en cuestión en cada una de sus instancias de trabajo. Esto le permite ejecutar contenido dinámico dentro del propio servidor *web* sin tener que depender de componentes externos. Estos procesadores dinámicos pueden habilitarse mediante el uso de módulos dinámicamente cargables (Ellingwood Justin, 2015).

Por todo lo antes mencionado se puede apreciar que, aunque NGINX sea un servidor más ligero y Apache robusto, ambos son potentes, flexibles y capaces. Sin embargo, el proyecto real de Orión se está corriendo actualmente sobre NGINX por lo cual es recomendable utilizar este para lograr así compatibilidad con el sistema.

Gestor de Base de Datos

Una base de datos es una colección de información que se organiza para que pueda ser fácilmente accesible, administrado y actualizado (Rouse Margaret, 2014). Para la presente investigación se ha escogido como gestor de base de datos (BD) PostgreSQL 9.6 debido a las múltiples ventajas que ofrece. Esta versión permitirá a los usuarios escalar y escalar cargas de trabajo de base de datos de alto rendimiento. Entre las principales mejoras en PostgreSQL 9.6 incluyen (PostgreSQL, 2016):

- Ejecución paralela de escaneos secuenciales, uniones y agregados.
- Evite escanear páginas innecesariamente durante las operaciones de congelación por vacío.
- La replicación síncrona ahora permite que múltiples servidores en espera aumenten la fiabilidad.
- La búsqueda de texto completo ahora puede buscar frases (palabras adyacentes múltiples).
- Ahora admite combinaciones remotas, ordena, UPDATE y DELETE.
- Mejoras de rendimiento sustanciales, especialmente en el área de escalabilidad en servidores multi-CPU-socket.

Entorno Integrado de Desarrollo (IDE)

NetBeans IDE es un entorno de desarrollo - una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación. Es una herramienta de código abierto con una comunidad en constante crecimiento (NetBeans.org). Es un producto libre y gratuito sin restricciones de uso. Se encuentran disponibles versiones de NetBeans para los sistemas operativos Windows, Linux, Mac, OS y Solaris. La base en la que se sustenta su elección es que permite desarrollar aplicaciones utilizando el marco de trabajo Symfony y ejecutar los comandos de este directamente desde la interfaz del IDE (*Wellcome to NetBeans*, 2011). Además este ID

permite el auto-completamiento de código, señalamiento de errores con soluciones fáciles y posibilita una fácil navegación para proyectos y código.

Jmeter

Este es un proyecto de Apache que puede ser empleado como herramienta de prueba de carga para analizar y medir el desempeño de una variedad de servicios, con énfasis en aplicaciones *web*. JMeter puede ser usado como una herramienta de pruebas unitarias para conexiones de bases de datos con JDBC, FTP, LDAP, Servicios *web*, JMS, HTTP y conexiones TCP genéricas (*The Apache Software Foundation-JMeter*, 2015).

Acunetix

Acunetix puede utilizarse para realizar escaneos de vulnerabilidades en aplicaciones *web* y para introducir pruebas de acceso frente a los problemas identificados. La herramienta provee sugerencias para mitigar las vulnerabilidades identificadas y puede utilizarse para incrementar la seguridad de servidores *web* o de las aplicaciones que se analizan (Acunetix, 2015). Es por ello que se utilizará esta herramienta para comprobar el sistema en busca de múltiples vulnerabilidades que un atacante podría aprovechar para obtener acceso a los sistemas y datos.

1.5 Conclusiones parciales

En este capítulo se trataron los elementos teóricos que dan sustento a la propuesta de solución del problema planteado pudiéndose arribar a las siguientes conclusiones:

- El estudio de los conceptos asociados a la recomendación de información permitió alcanzar mejor entendimiento del tema en que se desenvuelve la investigación que se realiza.
- El análisis de las funcionalidades que ofrecen algunos de los sistemas homólogos más utilizados del mundo permitió identificar los requisitos necesarios para desarrollar el Subsistema de recomendación de información para Orión.
- La selección de las herramientas, metodología y tecnologías basadas en *software* libre, garantiza la libertad de modificación de la propuesta de solución.

CAPÍTULO 2. Diseño del Subsistema de recomendación de información

Introducción

En este capítulo se abordarán los principales aspectos en relación con el diseño del sistema a desarrollar teniendo en cuenta buenas prácticas y principios que contribuyan a agilizar posteriormente su implementación. Para la representación de las clases conceptuales significativas del problema a resolver se encuentra el diagrama del modelo del dominio. También se evidencian los casos de uso del sistema, los requisitos funcionales y artefactos relacionados a estos y los requisitos no funcionales.

2.1 Modelo de dominio

Un Modelo del Dominio es una representación de las clases conceptuales del mundo real, no de componentes de software. No se trata de un conjunto de diagramas que describen clases de software, u objetos de software con responsabilidades, sino que modela clases conceptuales significativas en un determinado problema (Larman, 2004).

Para mayor comprensión de la presente investigación se hace necesario la descripción del proceso de recomendación de información mediante una serie de conceptos y entidades, así como sus relaciones evidenciadas en un Modelo de Dominio.

2.1.1 Descripción de clases del Modelo de Dominio

En la siguiente tabla, se muestra la descripción de los conceptos o clases del Modelo de Dominio del ámbito de problema.

Tabla 4. Descripción de las clases del Modelo de Dominio (Fuente: Elaboración propia)

Conceptos	Descripción
Usuario	Persona que realiza las búsquedas
Buscador	Herramienta de recuperación de información en la <i>Web</i>
Rastreador	Mecanismo que recopila los documentos en la <i>Web</i>
Servidor de índice	Mecanismo que indexa los documentos
Interfaz <i>web</i>	Vista mediante la cual se muestran los resultados al usuario
Intranet	Red informática donde están los documentos

Sitio <i>web</i>	Colección de páginas en internet relacionadas a un dominio de internet o subdominio
Documento	Recursos publicados en la <i>Web</i> (páginas <i>web</i> , videos, documentos y otros)

2.1.2 Diagrama de clases de Modelo de Dominio

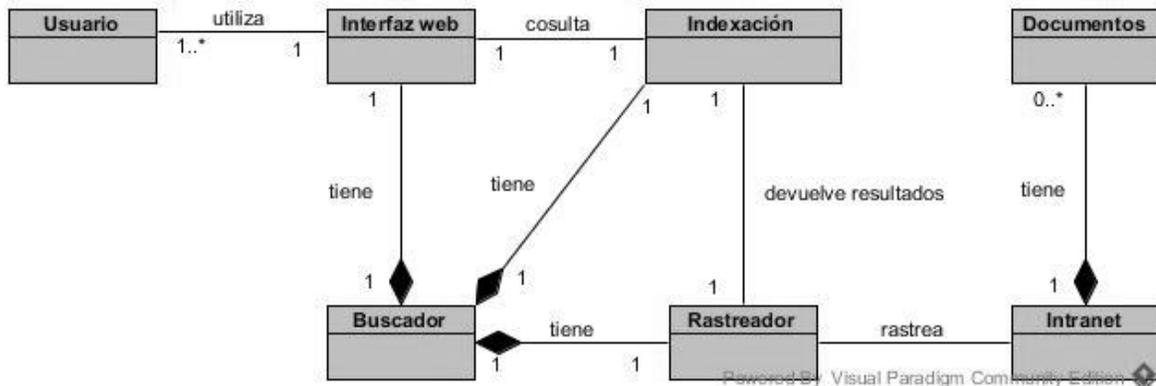


Figura 3. Diagrama de clases del Modelo de Dominio (Fuente: Elaboración propia)

Como se muestra en la figura 3, el Buscador contiene tres elementos fundamentales, una Interfaz *web*, un Servidor de índice y un Rastreador. Este último es el encargado de implementar el mecanismo de rastreo mediante el cual se escogen y almacenan en el Servidor de índice los documentos existentes en la intranet. Esta colección de documentos indexados, es la que toma en cuenta el buscador Orión a la hora de responder las peticiones de los usuarios expresadas en las consultas y será también la colección de documentos de entre las cuales se generarán las recomendaciones de la propuesta de solución.

2.2 Especificación de los Requisitos del Software

Según (Pressman Roger, 2010) los requisitos del *software* son aquellos que establecen las características fundamentales del sistema, así como sus restricciones tanto en el funcionamiento del sistema como en los procesos de desarrollo del *software*. Estos expresan las necesidades objetivas que presentan los usuarios ante un sistema que resuelve un problema en particular de un determinado dominio.

Gracias al estudio de sistemas homólogos y una vez realizadas las entrevistas a los diferentes trabajadores del proyecto Orión se pudieron identificar los requisitos funcionales y no funcionales del Subsistema de recomendación de información a desarrollar (ver tabla 4).

2.2.1 Requisitos funcionales

Tabla 5. Requisitos funcionales (Fuente: Elaboración propia)

No.	Requisito Funcional	Prioridad
RF 1.	Generar recomendación	Alta
RF 2.	Identificar similitud entre perfiles	Alta
RF 3.	Identificar similitud entre documentos	Alta
RF 4.	Eliminar recomendación	Alta
RF 5.	Seleccionar nueva categoría	Baja
RF 6.	Desmarcar categoría	Baja
RF 7.	Autenticar usuario	Media
RF 8.	Listar por categoría	Media
RF 9.	Mostrar recomendaciones por su fecha de recomendación	Media
RF 10.	Mostrar recomendaciones por orden alfabético	Baja

2.2.2 Requisitos no Funcionales

De software

RNF 1. Para el funcionamiento de la solución es necesario el sistema operativo Linux en cualquier distribución, o Windows XP o superior.

RNF 2. La solución debe estar optimizada para visualizarse en el navegador Mozilla Firefox 32 o superior.

De implementación

RNF 3. Como lenguaje de programación para la interfaz *web* se deberá utilizar PHP en su versión 5.5 o superior.

RNF 4. Para el desarrollo de la aplicación *web* se deberá utilizar Symfony 2.3 LTS o superior como marco de trabajo.

De diseño

RNF 5. En la página de resultados siempre estará visible el menú de opciones y el formulario de búsqueda.

De licencia

RNF 6. Se requiere el uso de herramientas y recursos de software libre para que la solución pueda ser usada, modificada y distribuida libremente.

De usabilidad

RNF 7. La interfaz debe ser sencilla y amigable de manera que potencie la comodidad del usuario para su trabajo.

De seguridad

RNF 8. El Subsistema de recomendación de información, así como la información se encontrarán protegidos contra accesos no autorizados utilizando mecanismos de autenticación propios del sistema.

De hardware

RNF 9. La solución requiere ser instalada en un entorno que posea como mínimo un procesador de 4 núcleos, 4 GB de RAM y conexión a la red.

2.3 Historias de Usuario

Tabla 6. Tabla de Historia de usuario #1 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 1	Nombre: Generar recomendación
Prioridad en negocio: Alta	
Descripción: Genera las recomendaciones para los usuarios teniendo en cuenta la similitud entre perfiles y la similitud ente documentos.	
Prototipo: No Aplica (N/A)	

Tabla 7. Tabla de Historia de usuario #2 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 2	Nombre: Identificar similitud entre perfiles
Prioridad en negocio: Alta	
Descripción: Establece comparaciones entre perfiles de usuarios para determinar su índice de similitud.	
Prototipo: N/A	

Tabla 8. Tabla de Historia de usuario #3 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 3	Nombre: Identificar similitud entre documentos
Prioridad en negocio: Alta	
Descripción: Establece comparaciones entre el contenido de los documentos del indexador para determinar su índice de similitud.	
Prototipo: N/A	

Tabla 9. Tabla de Historia de usuario #4 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 4	Nombre: Eliminar recomendación
Prioridad en negocio: Alta	
Descripción: Elimina una recomendación que no sea deseada. La funcionalidad comienza cuando el usuario elige el símbolo x (eliminar) que aparece a la derecha de la recomendación.	
Prototipo: Anexo 1 (Prototipo funcional #1)	

Tabla 10. Tabla de Historia de usuario #5 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 5	Nombre: Seleccionar nueva categoría
Prioridad en negocio: Alta	
Descripción: Incluye una nueva categoría al perfil de usuario. La funcionalidad comienza cuando el usuario registra una nueva categoría.	
Prototipo: Anexo 1 (Prototipo funcional #2)	

Tabla 11. Tabla de Historia de usuario #6 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 6	Nombre: Desmarcar categoría
Prioridad en negocio: Alta	
Descripción: Desmarca una categoría del perfil del usuario. La funcionalidad comienza cuando el usuario elige desmarcar del listado de categorías que se muestra.	
Prototipo: Anexo 1 (Prototipo funcional #2)	

Tabla 12. Tabla de Historia de usuario #7 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 7	Nombre: Autenticar usuario
Prioridad en negocio: media	
Descripción: Permite la autenticación de los usuarios que estén registrados. La funcionalidad comienza cuando el usuario introduce su usuario y contraseña. Los campos se muestran tras pulsar un botón que se encuentra en la página principal.	
Prototipo: Anexo 1 (Prototipo funcional #3)	

Tabla 13. Tabla de Historia de usuario #8 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 8	Nombre: Listar por categoría
Prioridad en negocio: Baja	
Descripción: Muestra las recomendaciones dada una categoría. La funcionalidad comienza cuando el usuario escoge de un "select" el criterio por el cual desea listar sus recomendaciones.	
Prototipo: Anexo 1 (Prototipo funcional #1)	

Tabla 14. Tabla de Historia de usuario #9 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 9	Nombre: Mostrar recomendaciones por su fecha de recomendación
Prioridad en negocio: Baja	
Descripción: Muestra las recomendaciones por su fecha de recomendación descendientemente.	
Prototipo: Anexo 1 (Prototipo funcional #1)	

Tabla 15. Tabla de Historia de usuario #10 (Fuente: Elaboración propia)

Historia de usuario	
Número: HU 10	Nombre: Mostar recomendaciones por orden alfabético
Prioridad en negocio: Baja	
Descripción: Muestra y ordena las recomendaciones según su orden alfabético de forma descendente de la "a" a la "z". La funcionalidad comienza cuando el usuario escoge de un menú desplegable el criterio por el cual desea listar sus recomendaciones.	
Prototipo: Anexo 1 (Prototipo funcional #1)	

2.4 Estilo arquitectónico

Para el desarrollo del Subsistema de recomendación de información se empleará como marco de trabajo *Symfony* que utiliza el patrón arquitectónico Modelo-Vista-Controlador (MVC). Este marco de trabajo sólo proporciona herramientas para la parte del Controlador y de la Vista. La parte del Modelo es responsabilidad del usuario (Potencier, 2011).

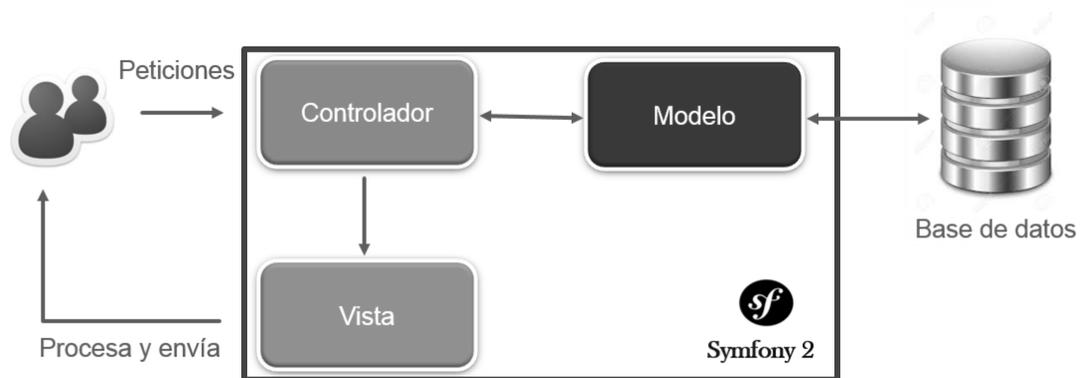


Figura 4. Arquitectura MVC (Fuente: Elaboración propia)

Como se observa en la figura 4, son recibidas y atendidas todas las peticiones a través del controlador. Cuando este recibe una petición del usuario, consulta los datos almacenados en la base de datos a través del modelo. Esta arquitectura está basada en el patrón arquitectónico MVC, donde la función del modelo la realizan las entidades usuario, recomendación, categoría y preferencia, las cuales son las encargadas de implementar la lógica del negocio. Por otra parte, el controlador es el intermediario entre el modelo de datos y las vistas. Responde a las peticiones que hace el propio Subsistema de recomendación de información para generar las recomendaciones, y se encarga de realizar las solicitudes de datos al modelo. Por último, la vista funciona como la capa de presentación de los datos de la propuesta de solución, es decir, presenta de manera gráfica todos los elementos que el usuario es capaz de visualizar dentro de la aplicación.

2.5 Patrones de diseño

Los patrones de diseño son soluciones para problemas típicos y recurrentes que es pueden encontrar a la hora de desarrollar una aplicación. Representan la descripción de un problema particular y recurrente, que

aparece en contextos específicos, y presenta un esquema genérico demostrado con éxito para su solución; este último se especifica mediante la descripción de los componentes que la constituyen, sus responsabilidades y desarrollos, así como también la forma como estos colaboran entre sí (Larman, 2004).

Patrones GRASP

En el diseño del Subsistema de recomendación de información de la presente investigación se tuvieron en cuenta los siguientes patrones GRASP (Patrones Generales de *Software* para Asignación de Responsabilidades) que describen los principios fundamentales de la asignación de responsabilidades a objetos:

Experto: este patrón plantea que se debe asignar una responsabilidad al experto en información, en otras palabras, a la clase que cuenta con los datos necesarios para cumplir la responsabilidad. De esta forma, se conserva el encapsulamiento de la información, puesto que los objetos ejecutan las tareas que le corresponden de acuerdo a la información que poseen, lo que da lugar a sistemas más robustos y fáciles de mantener (Larman, 2004).

Como se muestra en la siguiente tabla, dicho patrón se asigna en la propuesta de solución solamente a las clases que cuentan con la información necesaria para dar cumplimiento a las mismas, logrando así el encapsulamiento de la información ya que los objetos utilizan su propia información para llevar a cabo las tareas.

Tabla 16. Relación de asignación de responsabilidades (Fuente: Elaboración propia)

Clases modelo	Responsabilidad
Usuario.php	Se encarga de la autenticación de los usuarios.
Recomendacion.php	Se encarga de filtrar, listar y eliminar las recomendaciones.
Categoria.php	Mostrar todas las categorías y seleccionar las categorías por las cuales se emitirán las recomendaciones.
Preferencia.php	Posee las preferencias de los usuarios.

Creador: con la implementación de este patrón es posible identificar quién debe ser el responsable de la creación o instanciación de nuevos objetos o clases. Se utiliza para guiar la asignación de responsabilidades relacionadas con la creación de objetos, ya que esta última es una de las actividades más frecuentes en un sistema orientado a objetos.

Durante el desarrollo de la propuesta de solución se evidencia el uso de este patrón en las clases controladoras RecommendationController.php, DefaultController.php y UsuarioController.php que son las encargadas de crear instancias de otras clases (objetos).

Alta cohesión: en el diseño orientado a objetos, la cohesión es una medida de la fuerza con la que se relacionan y del grado de focalización de las responsabilidades de un elemento (clase o subsistema). Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas, que colaboran entre sí y con otros objetos para simplificar su trabajo. Una clase con alta cohesión es relativamente fácil de mantener, entender y reutilizar (Larman, 2004).

Symfony2 permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con alta cohesión ya que cada clase controladora maneja solamente las responsabilidades correspondientes a las entidades con las que se relaciona. En la propuesta de solución se evidencia el uso de este patrón, cuando por ejemplo RecommendationController.php al recibir un cambio en sus atributos requiere una reestructuración de los métodos para su correcto funcionamiento.

Bajo acoplamiento: este patrón expresa que entre las clases debe existir pocas ataduras, es decir, estas estarán lo menos relacionadas posible de forma tal que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases, incrementando la reutilización y disminuyendo la dependencia entre las clases.

Controlador: este patrón tiene como objetivo asignar la responsabilidad a una clase de recibir o manejar un mensaje de evento del sistema generado por un actor externo, por lo general a través de una interfaz gráfica de usuario a la que accede un usuario para realizar ciertas operaciones en el sistema (Larman, 2004).

La utilización de este patrón se evidencia a través de las clases controladoras RecommendationController.php y UsuarioController.php las cuales se encargan de atender y ofrecer respuesta a cada una de las peticiones realizadas por el usuario mediante la interfaz *web*.

Patrones GOF

Método de Fábrica: Symfony2 ofrece una forma confiable para la creación de objetos, así como la especificación de los parámetros necesarios para la creación de estos de una manera muy sencilla. El *EntityManager* (Administrador de entidades) implementado por Doctrine hace uso del Método de Fábrica para proporcionar una instancia de las clases repositorios asociadas a cada una de las entidades, que son las encargadas del proceso de búsqueda y filtrado relacionadas con la base de datos.

```
public function deleteAction($id)
{
    $em = $this->getDoctrine()->getManager();
    $recomendacion=$em->getRepository('AppBundle:Recomendacion')->find($id);
    $user=$this->getUser();
    $user->removeRecomendacion($recomendacion);
    $em->flush();
    return $this->redirectToRoute('recomend');
}
```

Figura 5. Uso del patrón Método de Fábrica en la clase Recomendacion.php (Fuente: Elaboración propia)

Decorador: en Symfony2 el motor de plantillas *Twig* implementa la herencia de plantillas, a través de la cual es posible utilizar el patrón decorador; permitiendo la declaración de manera general de la organización del sistema y definiendo los bloques que podrán ser redefinidos por sus descendientes. De esta forma puede ser definida la estructura del sistema en la plantilla *Symfony/app/Resources/views/index.html.twig*, dejando a las plantillas descendientes de esta la redefinición de los bloques que competen en cada nivel según la funcionalidad que se esté implementando en cada caso.

```
{% extends 'index.html.twig' %}

{% block contenido %}

{% endblock %}
{% block categorias %}
    {% for categoria in categ %}
        <li class="nav-item start active" style="" id="li{{ categoria.id }}">
            <a class="nav-link " href="{{ path('categoria_filtrar', { 'id': categoria.id }) }}">
                <span class="title">{{ categoria.nombre }}</span>
                <span class="selected"></span>
            </a>
        </li>
    {% endfor %}
{% endblock %}
```

Figura 6. Uso del patrón Decorador en la vista Recomendacion/Index.html.twig (Fuente: Elaboración propia)

2.6 Diagrama de despliegue

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Esto muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del *software* se relacionan en esos nodos (SparxSystems, 2014).

En la figura 7 se puede apreciar el nodo “Cliente”. Este nodo representa un dispositivo utilizado por el usuario desde el cual realiza la búsqueda de información a través del protocolo http haciendo uso de un navegador *web*. Como encargado de atender y brindar respuesta a cada una de las peticiones del cliente se encuentra el nodo “Servidor de Aplicación *web* NGINX” y el nodo “Base de Datos” representa la base de datos con la que contará el sistema.

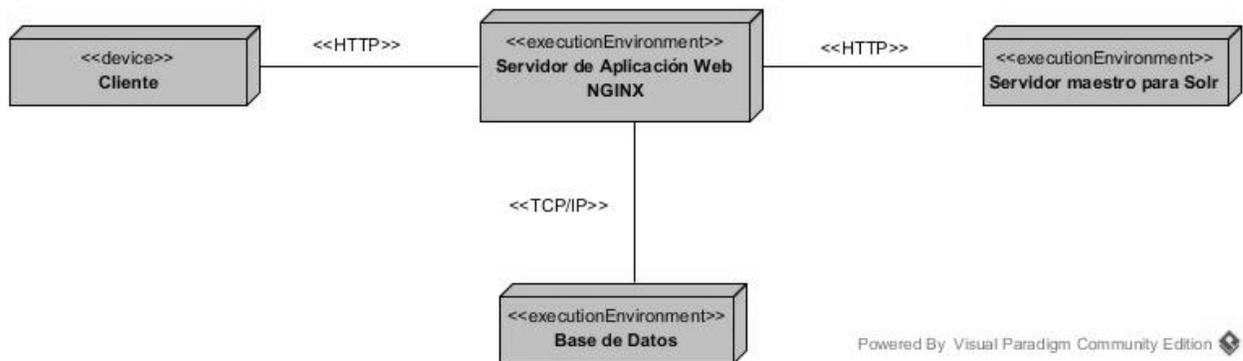


Figura 7. Diagrama de despliegue (Fuente: Elaboración propia)

2.7 Conclusiones parciales

En este capítulo se abordó lo referente al análisis y diseño del Subsistema de recomendación de información que se desea implementar arribándose a las siguientes conclusiones:

- El levantamiento de requisitos permitió una mayor comprensión y organización de los elementos del Subsistema de recomendación de información a desarrollar, así como las funcionalidades que deben implementarse.
- La representación de los artefactos generados propicia mayor entendimiento del proceso de recomendación de información.
- Con la definición de la arquitectura y los patrones de diseño a utilizar, se fomentan las buenas prácticas de programación durante la fase de implementación y se contribuye a disminuir el impacto de los futuros cambios que puedan ocurrir en el código fuente.

- Mediante el diagrama de despliegue se pudo identificar la disposición física que tendrán los artefactos del Subsistema de recomendación de información que se desea desarrollar.

CAPÍTULO 3. Implementación y validación del Subsistema de recomendación de información

Introducción

La implementación constituye una parte crucial durante el proceso de desarrollo de software. En esta etapa se define y organiza el código de la propuesta de solución, y de esta forma se materializan los artefactos de implementación, descripciones y arquitectura propuestos en la etapa de análisis y diseño para conformar el producto final requerido por el cliente. Aparejado a esta etapa descrita anteriormente, el software debe someterse a determinadas pruebas que garanticen la correspondencia entre los requisitos definidos previamente y el producto final. Esta fase es conocida como validación del sistema.

3.1 Modelo de componentes del Subsistema de recomendación de información

El modelo de componentes representa la forma en que es estructurado un sistema informático, atendiendo a las diferentes partes que lo componen y las relaciones de dependencia entre estas partes. Algunos ejemplos de componentes físicos lo constituyen los archivos, módulos, librerías, ejecutables, entre otros.

3.2 Diagrama de componentes

El diagrama de componentes representa la estructura general del sistema, permitiendo así la visualización del comportamiento de las funcionalidades que estos componentes proporcionan y utilizan a través de las interfaces como se muestra en la figura 8. A continuación, se describen los principales paquetes que incluye el diagrama de componentes correspondiente a la interfaz *web* del Subsistema de recomendación de información.

InterfaceBundle: Dentro de este se agrupan los componentes de la interfaz *web*, estableciendo una estructura organizativa acorde al patrón de arquitectura MVC.

Controller: Contiene la clase controladora encargada de procesar las peticiones de las páginas clientes y del usuario, así como devolver las respuestas con la información requerida.

Config: Contiene en su mayoría los archivos donde se definen las rutas de la aplicación.

Form: Alberga las clases para construir los formularios utilizados para el envío de información por parte de los usuarios.

Views: Agrupa las páginas referentes a las vistas de la aplicación, así como las plantillas bases.

Web: Dentro contiene diferentes componentes de apoyo a las vistas de la aplicación entre otros recursos. En este paquete se encuentran los estilos; archivos CSS y JavaScripts que son los archivos para aplicar dinamismo a partes del código HTML).

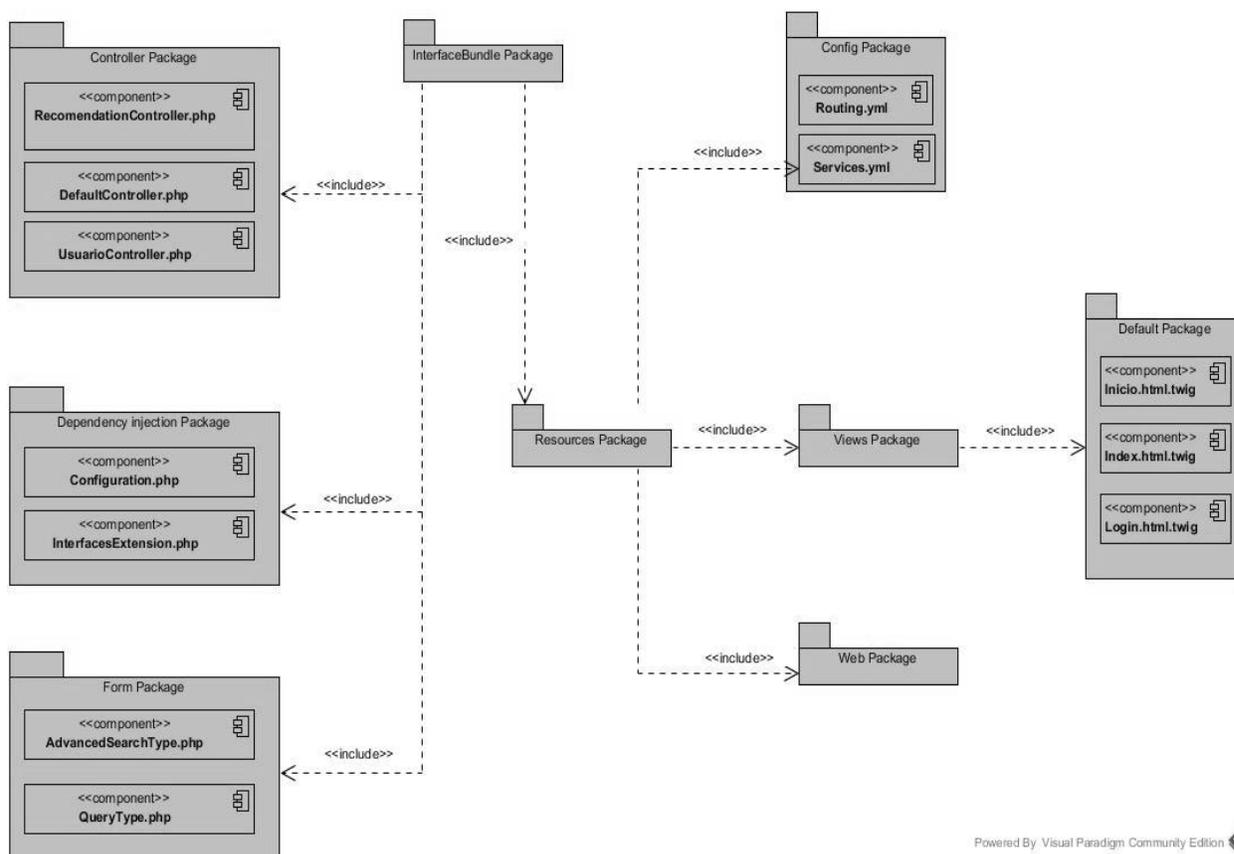


Figura 8. Diagrama de componentes (Fuente: Elaboración propia)

3.3 Validación del Subsistema de recomendación de información

En el siguiente epígrafe se definen y presentan los resultados de las pruebas aplicadas al producto de *software* implementado con el objetivo de detectar las no conformidades con respecto a las funcionalidades de la aplicación, el grado de usabilidad de las funcionalidades implementadas, así como la correcta integración entre los distintos componentes de la arquitectura del sistema.

3.3.1 Pruebas funcionales

Este tipo de pruebas se realiza para validar que las funcionalidades implementadas funcionen de acuerdo a las especificaciones de los requisitos definidos previamente. Para la ejecución de este tipo de pruebas, se emplean dos métodos fundamentales: el método de Caja Blanca y el método de Caja Negra. El primero está orientado a pruebas al código de las aplicaciones; mientras que el segundo, a través del análisis de los datos de entrada y los de salida, permite enfocar la atención en el funcionamiento de la interfaz. A continuación, se presentan los aspectos concernientes a las pruebas funcionales realizadas utilizando el método de Caja Negra, a partir de los casos de prueba diseñados.

Tabla 17. Cantidad de no conformidades por cada iteración de las pruebas funcionales (Fuente: Elaboración propia)

No conformidades	1ra iteración	2da iteración	3ra iteración
Detectadas	5	1	0
Resueltas	4	2	0
Pendientes	1	0	0

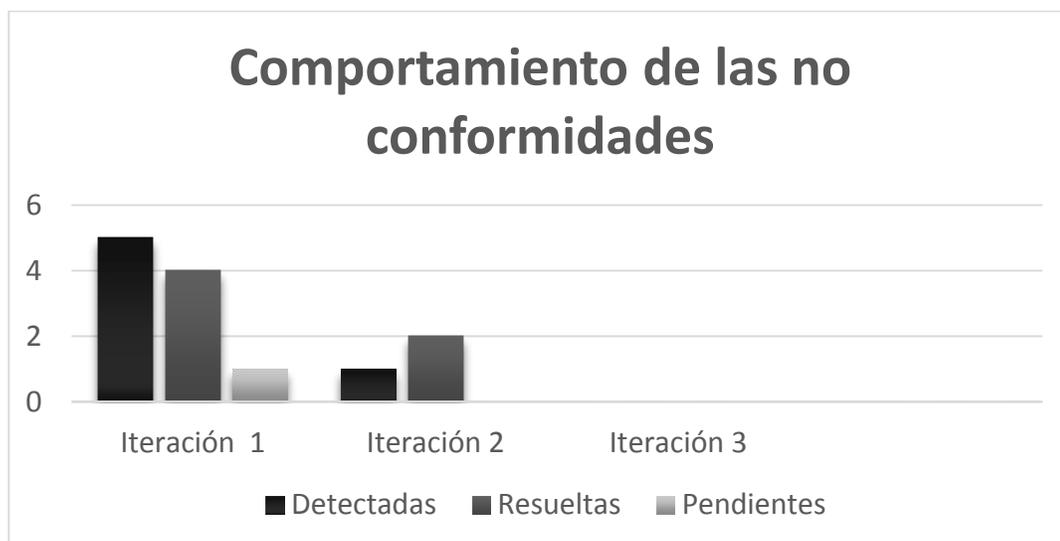


Figura 9. Comportamiento de las no conformidades de las pruebas funcionales ejecutadas (Fuente: Elaboración propia)

Resultado de las pruebas funcionales:

Como se muestra en la figura 9 para la validación de los requisitos funcionales se realizaron tres iteraciones.

Iteración 1: Fueron detectadas cinco no conformidades; dos de ortografía, dos de interfaz y una de funcionalidad. De estas fueron resueltas 4 quedando una no conformidad de interfaz pendiente.

Iteración 2: Fue detectada una no conformidad de interfaz. Se resolvieron en esta iteración un total de dos no conformidades.

Iteración 3: No fue detectada ninguna no conformidad, por lo cual se puede decir que al final de las iteraciones quedan resueltas todas las no conformidades.

Diseño de los casos de prueba

A continuación, se presentan las tablas correspondientes a los casos de prueba de la tercera iteración (ver anexo 2).

Tabla 18. Caso de prueba Generar recomendación (Fuente: Elaboración propia)

Caso de prueba Eliminar Recomendación	
Código de caso de prueba: 1	Nombre de historia de usuario: Generar recomendación
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad Generar Recomendación	
Entrada/Pasos de ejecución: Se realizan las siguientes acciones -Escoger un usuario. -Verificar sus preferencias. -Escoger una colección de documentos que se ajuste a sus preferencias. -Escoger una colección de documentos que no se ajuste a sus preferencias.	
Ejecutar la prueba	
Evaluación de la prueba: Satisfactoria. Las recomendaciones generadas se ajustan a las preferencias del usuario	

Tabla 19. Caso de prueba Similitud entre perfiles (Fuente: Elaboración propia)

Caso de prueba Eliminar Recomendación	
Código de caso de prueba: 2	Nombre de historia de usuario: Identificar similitud entre perfiles
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad Identificar similitud entre perfiles	
Entrada/Pasos de ejecución: Se realizan las siguientes acciones -Escoger un usuario. -Escoger una muestra de 5 usuarios al azar.	
Ejecutar la prueba	
Evaluación de la prueba: Satisfactoria. Los usuarios que el sistema determino como similares poseen preferencias similares al usuario en cuestión.	

Tabla 20. Caso de prueba Similitud entre perfiles (Fuente: Elaboración propia)

Caso de prueba Eliminar Recomendación	
Código de caso de prueba: 2	Nombre de historia de usuario: Similitud entre documentos
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad Identificar similitud entre documentos	
Entrada/Pasos de ejecución: Se realizan las siguientes acciones -Escoger un documento -Escoger una colección de documentos de la misma categoría que guarde relación con el documento en cuestión. -Escoger una colección de documentos de la misma categoría que no guarde relación con el documento en cuestión. -Escoger una colección de documentos con categorías diferentes que el documento en cuestión.	
Ejecutar la prueba	
Evaluación de la prueba: Satisfactoria. Los mayores porcentajes los presentan los documentos de la misma categoría que guardan relación con el documento en cuestión. Los documentos que no guardan relación con el documento en cuestión presentan un porcentaje muy bajo o casi nulo.	

3.3.2 Pruebas de Integración

Las pruebas de integración son definidas para verificar el correcto ensamblaje entre los distintos módulos que conforman un sistema informático. Las mismas validan que estos componentes realmente funcionan juntos, son llamados correctamente y transfieren los datos correctos en el tiempo preciso y por las vías de comunicación establecidas (Sommerville, 2005).

Posterior a las pruebas realizadas, una vez verificado que las funcionalidades implementadas se corresponden de acuerdo a los requisitos funcionales y no funcionales establecidos, se hace necesaria la realización de pruebas de integración, con la finalidad de validar la compatibilidad y el funcionamiento de las interfaces que comunican las diferentes partes que componen la solución informática.

Para la realización de las pruebas de integración se llevaron a cabo diferentes acciones, a continuación, se mencionan las fundamentales:

- Verificación de la conexión de Symfony y Solr para extracción de los datos de los documentos indexados.
- Comprobar el enlace entre Solr y la interfaz de resultados para verificar si se muestran todas las recomendaciones generadas para los usuarios.

La ejecución de las pruebas de integración permitió verificar el trabajo conjunto de los componentes del Subsistema de recomendación de información. Se pudo constatar que existe una correcta integración entre los componentes internos del Subsistema de recomendación de información, no encontrándose ninguna no conformidad.

3.3.3 Pruebas de carga y estrés

Las pruebas de carga consisten en probar el funcionamiento del *software* bajo condiciones extremas. Estudia la especificación del *software*, las funciones que debe realizar, las entradas y las salidas, analizando los valores límites (López, 2010). Por otro lado, las pruebas de estrés están diseñadas para enfrentar al programa a condiciones extremas. Consiste en ejecutar un sistema que demande recursos en cantidad, frecuencia o volúmenes extremos.

Es necesario ejecutar dichas pruebas en el Subsistema de recomendación de información pues resulta necesario comprobar su rendimiento soportando una cantidad máxima de usuarios que soliciten este recurso. Para la realización de las mismas se utiliza la herramienta JMeter en su versión 3.1.

Los resultados arrojados se muestran en las figuras 10 y 11:

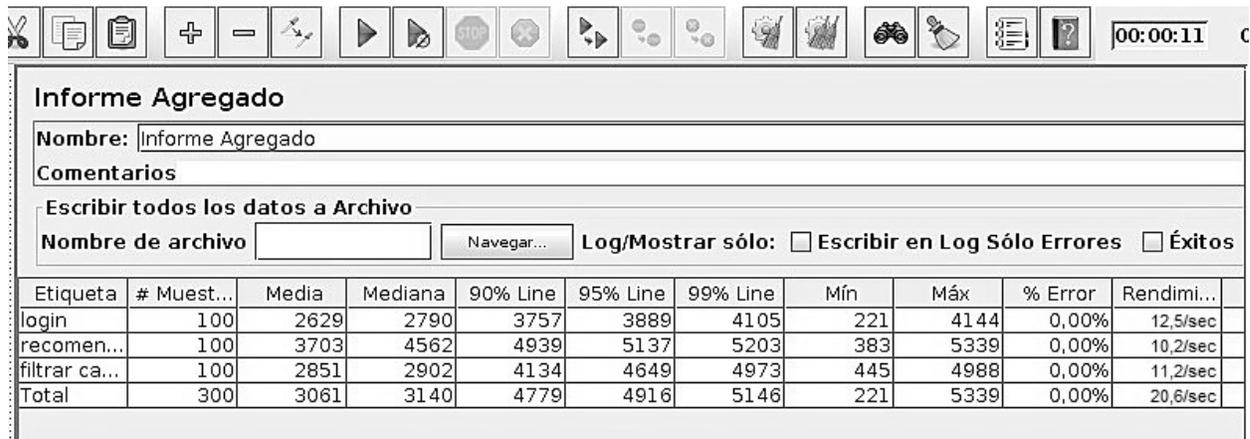


Figura 10. Pruebas de rendimiento a las funcionalidades filtrar por categoría, seleccionar nueva categoría y desmarcar categoría (Fuente: Elaboración propia)

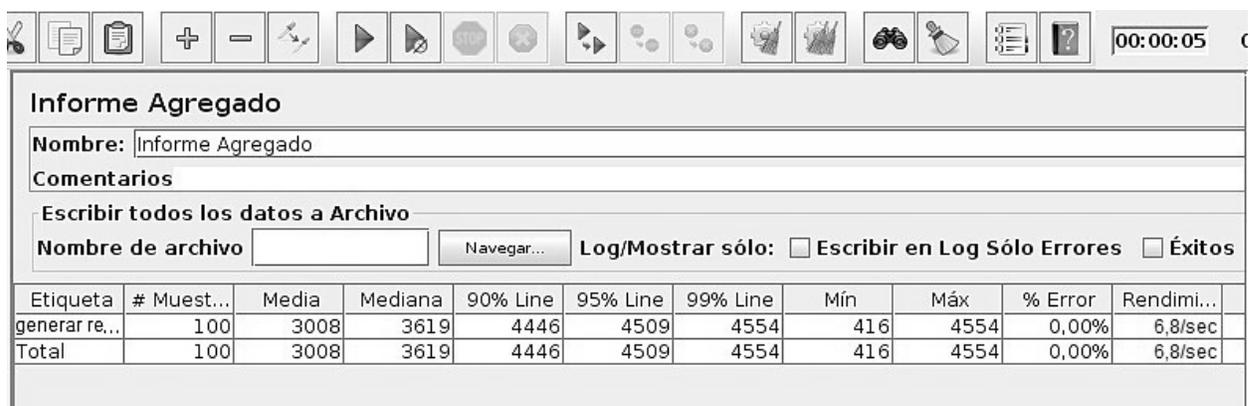


Figura 11. Pruebas de rendimiento a la funcionalidad generar recomendación (Fuente: Elaboración propia)

Resultado de las pruebas carga y estrés:

Como se aprecia en la figura 10 el tiempo promedio de las solicitudes para 100 hilos es de 3.061 segundos, realizándose un total de 300 solicitudes al servidor y atendándose en total 20.6 solicitudes por cada segundo. Para la funcionalidad generar recomendación mostrada en la figura 11, la media obtenida fue de 3.0 para un total de 100 solicitudes y atendándose un total de 6.8 solicitudes por cada segundo. De manera general estos resultados son favorables por lo que se puede decir que el sistema cumple con las condiciones de rendimiento necesarias.

3.3.4 Pruebas de seguridad

La seguridad informática comprende la puesta en práctica de un conjunto de medidas preventivas y reactivas en los sistemas informáticos y tecnológicos, que posibilitan la protección de la información, persiguiendo como objetivo principal la integridad, confidencialidad y disponibilidad de la misma (INTECO-CERT, 2014). Se realizan pruebas de seguridad al producto de *software* para de esta manera contribuir a la detección temprana de vulnerabilidades y la toma de medidas para la disminución de amenazas de ataque, y con ello proveer un sistema más seguro y confiable. El *software* utilizado para la realización de las pruebas es Acunetix. Las pruebas realizadas se muestran a continuación:

- Ataques de inyección.
- *Cross-Site Scripting* (XSS).
- Falsificación de petición (CSRF).
- Detección de ficheros y directorios.

Los resultados se muestran en las siguientes tablas:

Tabla 21. Vulnerabilidades del entorno (Fuente: Elaboración propia)

Tipo	Cantidad	Descripción	Recomendaciones
<i>Software</i>	1	La versión de PHP instalada contiene vulnerabilidades que han sido solucionadas en versiones más recientes.	Actualizar a la versión 5.6.4 o superior.

Tabla 22. Vulnerabilidades del sistema (Fuente: Elaboración propia)

Tipo	Cantidad	Descripción	Recomendaciones
Falso positivo	1	El sistema permite la inyección de código.	Esta vulnerabilidad no puede ser utilizada por el atacante para alterar el sistema ni obtener información por lo que se denomina falso positivo.

			Symfony no permite la inyección SQL.
--	--	--	--------------------------------------

Resultado de las pruebas de seguridad:

Los resultados obtenidos en las pruebas de seguridad fueron satisfactorios, llegando a la conclusión que el Subsistema de recomendación de información desarrollado es seguro y está en condiciones de ser usado por el cliente.

3.4 Validación de la Hipótesis de la investigación

Con el propósito de evaluar el indicador **Precisión** en el Subsistema de recomendación de información, correspondiente a la variable dependiente definida como parte de la hipótesis de investigación, se realizó un experimento a 3 usuarios. Este consistió en escoger 3 colecciones diferentes de 50 documentos cada una donde 12 respondían a necesidades y preferencias de estos usuarios los otros 38 no. Se realizaron 3 iteraciones arrojándose los siguientes resultados (ver tabla 22):

Tabla 23. Resultados de la evaluación del indicador Precisión (Fuente: Elaboración propia)

	Prueba 1	Prueba 2	Prueba 3
Cantidad de recomendaciones relevantes	9	11	10
Precisión	75%	91%	83%

Para la validación de la hipótesis se aplicó además la técnica IADOV, la cual permitió evaluar el nivel de satisfacción de potenciales usuarios con el modelo propuesto (Fernández de Castro, López Padrón, 2014) Esta consistió en realizar una encuesta a una muestra de 25 usuarios (ver Anexo 3) de entre los cuales 15 poseen experiencia con buscadores y conocimientos de informática y 10 son usuarios con poca experiencia en buscadores y bajo conocimiento de informática.

Tabla 24. Cuadro lógico de IADOV (Fuente: Elaboración propia)

	3. ¿Considera relevantes las recomendaciones obtenidas a partir de las búsquedas realizadas?								
	SI			NO			NO SÉ		
7. ¿Le satisfacen las recomendaciones dadas a partir de las salidas que provee y la manera en que muestra los resultados?	6. ¿Siente usted que este Subsistema será útil al buscador Orión?								
	SI	NO	NO SÉ	SI	NO	NO SÉ	SI	NO	NO SÉ
Me satisface mucho	1	2	6	2	2	6	6	6	6
No me satisface tanto	2	2	3	2	3	3	6	3	6
Me da lo mismo	3	3	3	3	3	3	3	3	3
Me insatisface más de lo que me satisface	6	3	6	3	4	4	3	4	4
No me satisface nada	6	6	6	6	4	4	6	4	5
No lo sé	2	3	6	3	3	3	6	3	4

En la tabla anterior, las filas subrayadas corresponden a las preguntas cerradas de la encuesta. El número resultante de la interrelación de las tres preguntas nos indica la posición de cada sujeto en la escala de satisfacción, siendo esta la que se presenta a continuación:

1. Clara satisfacción
2. Más satisfecho que insatisfecho
3. No definida
4. Más insatisfecho que satisfecho
5. Clara insatisfacción
6. Contradictoria

Para obtener el índice de satisfacción grupal (ISG) se trabaja con los diferentes niveles de satisfacción que se expresan en la escala numérica que oscila entre +1 y - 1 de la siguiente forma:

Tabla 25. Índices de satisfacción (Fuente: Elaboración propia)

+1	Máximo de satisfacción
0.5	Más satisfecho que insatisfecho
0	No definido y contradictorio
-0.5	Más insatisfecho que satisfecho
-1	Máxima insatisfacción

La satisfacción grupal se calcula por la siguiente fórmula:

$$ISG = \frac{A(+1) + B(0.5) + C(0) + D(-0.5) + E(-1)}{N}$$

En esta fórmula A, B, C, D, E, representan el número de sujetos con índice individual 1; 2; 3 ó 6; 4; 5 y donde N representa el número total de sujetos del grupo. Para un total de 25 usuarios el resultado es el siguiente:

$$ISG = \frac{15(+1) + 7(0.5) + 0(0) + 3(-0.5) + 0(-1)}{25}$$

$$ISG = 0.8$$

Conclusiones del método:

- Se obtuvieron criterios positivos expresados por unanimidad o mayoría.
- No existe contradicción en los resultados arrojados por los métodos aplicados.
- Se constata la capacidad del Subsistema de recomendación de información para mejorar la calidad en los resultados mostrados por el buscador Orión.

3.5 Conclusiones parciales

- Se pudo alcanzar mayor grado de entendimiento de la estructura del Subsistema de recomendación de información a través de la presentación y descripción del diagrama de componentes.
- Las pruebas realizadas al Subsistema de recomendación de información permitieron la detección y corrección de sus deficiencias para de esta manera ofrecer una aplicación con mayor calidad, usabilidad y seguridad.

- La aplicación del método experimental y la realización de la técnica de IADOV, aportaron elementos sustanciales que permitieron validar la hipótesis de la presente investigación y con ello la factibilidad del Subsistema implementado.

CONCLUSIONES

Una vez completada la presente investigación, se puede concluir que:

- A partir del estudio de sistemas homólogos se determinó que existen una serie de Sistemas de Recomendación de Información personalizados que contribuyen a la calidad en los resultados mostrados por los sistemas que los utilizan. Al no contar el buscador cubano Orión con ningún sistema de este tipo y de acuerdo a las necesidades existentes en cuanto a precisión de los resultados mostrados a los usuarios por el buscador, se define el Subsistema de recomendación de información como propuesta de solución.
- El uso de las tecnologías y herramientas seleccionadas además del enfoque ágil propuesto por la metodología AUP-UCI, permitió de manera sencilla realizar el diseño, la implementación y la validación de la solución construida.
- El estudio de los elementos que intervienen en el proceso de recomendación de información permitió la correcta modelación de los artefactos que contribuyeron al diseño de la propuesta de solución, posibilitando un mayor soporte a la implementación de los requisitos previamente definidos. Esto garantizó una correcta organización lógica del código fuente y la disminución del impacto ante futuras modificaciones en la aplicación.
- La implementación del Subsistema de recomendación de información utilizando las herramientas y tecnologías estudiadas y regida por la metodología AUP-UCI permitió solucionar los problemas existentes planteados en la problemática de la presente investigación.
- Las pruebas de software realizadas demostraron que la propuesta de solución es funcional, segura, con un rendimiento adecuado y que se integra sin dificultad al buscador Orión.

RECOMENDACIONES

Una vez concluida la investigación y el desarrollo de la propuesta de solución se recomienda:

- Implementar el filtrado colaborativo utilizando la inteligencia artificial para futuras versiones del Subsistema de recomendación de información.
- Integrar el Subsistema de recomendación de información en la plataforma de Red Cuba.

REFERENCIAS

- **ACUNETIX.** *Audit your website security with Acunetix Web Vulnerability Scanner.* [En línea]. Acunetix, 2015. [Citado el: 1 de noviembre de 2015.]. Disponible en: [<http://www.acunetix.com/>].
- **CLARIN.** *Cómo funciona el nuevo sistema de recomendación de Netflix.* [en línea]. 2016 [Citado el: 3 de noviembre 2016]. Disponible en la web: [http://next.clarin.com/funciona-nuevo-sistema-recomendaciones-Netflix_0_1532847068.html].
- **COCHRAN, DAVID.** *Twitter Bootstrap Web Development How-To.* Packt Publishing Ltd, 2012.
- **CONCEPTDEFINICION.DE.** *Definición de Youtube.* [en línea]. [Citado el: 5 noviembre 2016], Disponible en la web: [<http://conceptdefinicion.de/youtube>].
- **COSMIN, PUIUIONUT.** *Revista de Ciencias,* Universidad Pablo de Olavide, 2016.
- **DE CAMPOS, LUIS; FERNÁNDEZ LUNA, JUAN M; HUETE, JUAN F; RUEDA MORALES, MIGUEL A.** *Uso de conocimiento estructurado en un sistema de recomendación basado en contenido1.* Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada, 2011.
- **DELEGORRETA.** *Sistemas de recomendación de información, ejemplo sencillo.* [en línea]. 2015 [Citado el: 3 noviembre 2016]. Disponible en la web: [<https://dlegorreta.wordpress.com/2015/03/18/sobre-sistemas-de-recomendacion-ejemplo-sencillo>].
- **EGUILUZ, JAVIER.** *Desarrollo web ágil con Symfony.* 2014. 615 p.
- **ELLINGWOOD, JUSTIN.** *Apache vs Nginx: Practical Considerations.* [en línea]. 2015, [Citado el: 24 de noviembre de 2016]. Disponible en la web: [<https://www.digitalocean.com/community/tutorials/apache-vs-nginx-practical-considerations>].
- **FERNÁNDEZ DE CASTRO FABREI, ASTRID; LÓPEZ PADRÓN, ALEXANDER.** *Validación mediante criterio de usuarios del sistema de indicadores para prever, diseñar y medir el impacto en los proyectos de investigación del sector agropecuario.* Revista Ciencias Técnicas Agropecuarias, Universidad Agraria de La Habana, Centro de Mecanización Agropecuaria, San José de las Lajas, Mayabeque, Cuba, 2014.
- **FIGUEROA, ROBERTH G.; SOLÍS, CAMILO J.; CABRERA, ARMANDO A.** *Metodologías tradicionales vs. Metodologías ágiles.* Universidad Técnica Particular de Loja, Escuela de Ciencias en Computación. 2008. [En línea]. Disponible en la web: [<http://adonisnet.files.wordpress.com/2008/06/articulo-metodologia-de-sw-formato>].

- **FUMÁS CASES, EDUARD.** *¿Qué es PHP? ¿Para qué sirve?* [en línea]. 2014 [Citado el: 3 noviembre 2016]. Disponible en la web: [<http://www.ibrugor.com/blog/que-es-php-para-que-sirve/>].
- **FUMÁS CASES, EDUARD.** *Apache HTTP Server: ¿Qué es, como funciona y para qué sirve?* [en línea]. 2014, [Citado el: 28 de noviembre 2016], Disponible en la web: [<http://www.ibrugor.com/blog/apache-http-server-que-es-como-funciona-y-para-que-sirve/>].
- **GALÁN NIETO, SERGIO MANUEL.** *Filtrado Colaborativo y Sistemas de Recomendación* Universidad Calos III de Madrid, 2007.
- **GIORDANINO, EDUARDO PABLO.** *Sistemas de Recuperación de Información.* Informe inédito. Instituto de Formación Técnica superior.
- **GIZMODO.** *What Is Google +1?* [en línea]. 2011 [Citado el: 5 de noviembre de 2016], Disponible en la web: [<http://gizmodo.com/5787352/what-is-google-%252B1>].
- **GUTIERREZ DEMIÁN.** *Métodos de desarrollo de Software*, Universidad de los Andes, julio 2011.
- **HERRERA, ENRIQUE; PORCEL, CARLOS; HIDALGO, LORENZO.** *Sistemas de recomendaciones: Herramientas para el filtrado de información en Internet.* [en línea]. 2004 [Citado el: 16 de noviembre 2016]. Disponible en la web: [<https://www.upf.edu/hipertextnet/numero-2/recomendacion.html>].
- **INTERNET LIVE STATS.** *Internet live stats.* [en línea]. 2016. [Citado el: 3 de noviembre de 2016]. Disponible en la web: [<http://www.internetlivestats.com>].
- **JONES.** *Introduction to approaches and algorithms.* [en línea]. 2013. [Citado el: 3 de noviembre de 2016]. Disponible en la web: [<https://www.ibm.com/developerworks/library/os-recommender1>].
- **KOREN YEHUDA, BELL ROBERT.** *Advances in Collaborative Filtering.* Recommender systems handbook. Springer US, 2011, 145-186.
- **LARMAN, C.** *UML y Patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado.* Segunda. s.l.: Prentice Hall, 2004. pág. 520.
- **LÓPEZ, ALICIA Y ZORITA, LUIS.** *Las bibliotecas universitarias y la gestión de la información en el entorno digital: unas consideraciones para repositorios digitales.* [en línea]. 2008 [Citado el: 28 octubre 2016]. Disponible en la web: [<http://bid.ub.edu/20lopez2.htm>].
- **LÓPEZ, E., S.** *Ecología y libertad.* [en línea]. ATIX, Revista Digital, 2010. [Citado el: 14 de marzo de 2017.]. Disponible en: [<http://osl.ugr.es/descargas/atix16.pdf>].

- **MD.** *Analizamos el sistema de recomendaciones de YouTube, ¿es una herramienta demasiado simplona?* [en línea]. 2013 [Citado el: 5 noviembre 2016]. Disponible en la web: [<https://www.marketingdirecto.com/digital-general/digital/analizamos-el-sistema-de-recomendaciones-de-youtube-%C2%BFes-una-herramienta-demasiado-simplona>].
- **MIFSUF.** *Módulo iv: tic's y entornos virtuales.* Diplomado en educación a distancia y formación virtual. Universidad de San Francisco de Asís. Bolivia 2008
- **NETBEANS.** *Qué es NetBeans.* Sitio web NetBeans.org [en línea], disponible en la web: [https://netbeans.org/index_es.html].
- **NETCRAFT.** *Web server survey,* Sitio web de NetCraft. [en línea]. 2015. [Citado el: 12 de noviembre de 2016]. Disponible en la web: [<http://news.netcraft.com/>].
- **NETFLIX.** *Netflix Recommendations: Beyond the 5 stars (Part 1).* [en línea]. 2012 [Citado el: 2 de noviembre de 2016]. Disponible en la web: [<http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html>].
- **NGINX.** *Wellcome to NGINX Wiki's documentation!.* [en línea], 2016 [Citado el: 13 de noviembre de 2016]. Disponible en la web: [<https://www.nginx.com/resources/wiki/>].
- **PEPA, SOFÍA.** *Suit de algoritmos de recomendación en aplicaciones reales.* Trabajo de fin de grado. Universidad autónoma de Madrid. 2014.
- **PÉREZ MATA, MANUEL.** *¿Qué es Symfony?.* [en línea]. 2009 [Citado el: 2 noviembre de 2016]. disponible en la web: [<https://www.manelperez.com/linux/que-es-symfony>].
- **POSGRESSQL.** *PostgresSQL9.6 Release!.* [en línea]. 2016. [Citado el: 25 de noviembre de 2016]. Disponible en la web: [<https://www.postgresql.org/about/news/1703>].
- **POTENCIER, F.** *What is Symfony2?.* [en línea] SensioLabsNetwork, 2011. [Citado el: 2 de enero de 2016]. Disponible en: <http://fabien.potencier.org/article/49/what-is-symfony2>.
- **PRESSMAN, ROGER S.** *Ingeniería de Software, un enfoque práctico.* 2010
- **REAL ACADEMIA ESPAÑOLA.** *Diccionario de la lengua española.* [Citado el: 3 de noviembre de 2016]. Disponible en la web: [<http://dle.rae.es/?w=diccionario>].
- **RICCI, FRANCESCO; ROKACH, LIOR; SHAPIRA, BRACHA.** *Introduction to recommender systems handbook.* Springer US, 2011.
- **RODRÍGUEZ SÁNCHEZ, T.** *Metodología de desarrollo para la Actividad productiva de la UCI.* 3 junio 2015.

- **ROUSE MARGARET.** *Database*. [en línea]. 2014. [Citado el: 1 de diciembre de 2016]. Disponible en la web: [<http://searchsqlserver.techtarget.com/definition/database>].
- **SEGUIDO, MIGUEL.** *Sistemas de recomendación para webs de información sobre la salud*. Tesis de maestría en informática. Departamento de lenguajes y sistemas informáticos. Universidad politécnica de Catalunya. 2009.
- **SILVI LUCIANO.** *Aprendizaje activo para mejorar el Arranque en Frío de Sistemas de Recomendación*. Universidad Nacional de Córdoba. 2016.
- **SOMMERVILLE.** *Ingeniería de software*. Madrid: Pearson Educación S.A, 2005. 84-7829-074-5.
- **SOSA VAZQUE, SISLEY. FUENTES HERNÁNDEZ, YOENDRY.** *Propuesta de sistema de recomendación híbrido para sistema web de gestión y publicación de medias*. Universidad de las ciencias informáticas. 2013.
- **SPARXSYSTEMS.** *Diagrama de Despliegue UML 2*. [en línea]. Sparx Systems -Tutorial UML 2 - Diagrama de Despliegue, 2014. [Citado el: 12 de febrero de 2015.] Disponible en [http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html].
- **THE APACHE SOFTWARE FOUNDATION-TIKA.** *Apache Tika* [en línea]. Apache Solr, 2014. [Citado el: 13 de octubre de 2016.]. Disponible en: [<http://tika.apache.org/>].
- **VISUAL PARADIGM.** *Visual Paradigm for UML - Software design tools for agile software development*. [En línea] 2014. [Citado el: 4 de noviembre de 2016.]. Disponible en: [<http://www.visual-paradigm.com/product/vpuml/>].
- **WE ARE SOCIAL.** *Digital in 2016*, [en línea], 2016 [Citado el: 23 septiembre 2016] Disponible en la web: [<http://www.slideshare.net/wearesocialsg/digital-in-2016>].
- **WELCOME TO NETBEANS.** Sitio oficial del IDE Netbeans, 2011 [Citado el: 2 de diciembre de 2016]. Disponible en la web: [<http://netbeans.org>].

ANEXO #1

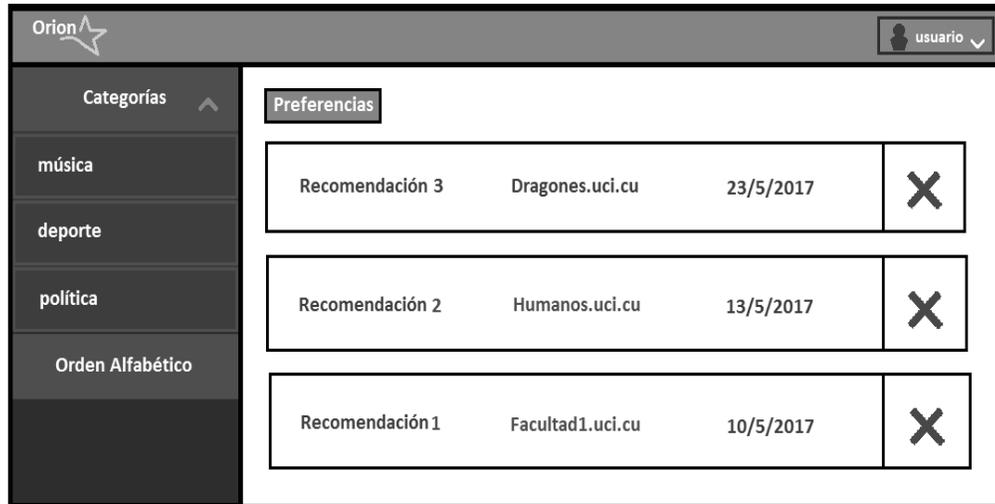


Figura 12. Prototipo funcional #1 (Fuente: Elaboración propia)

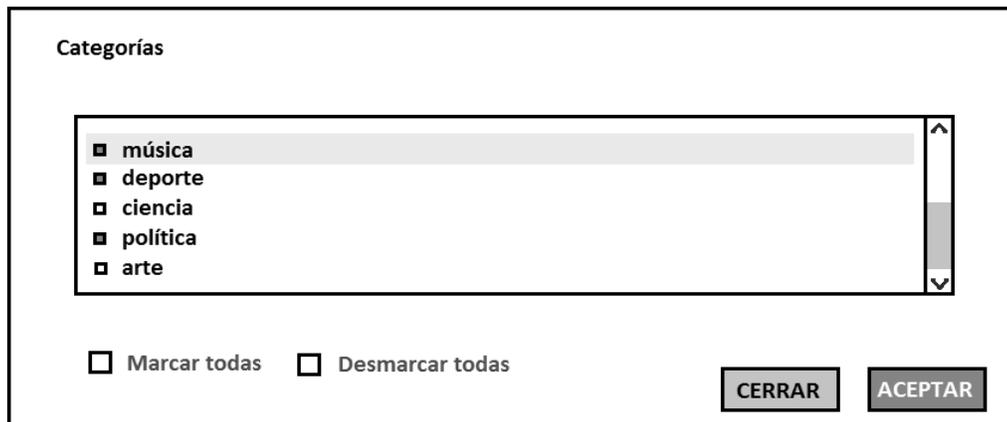


Figura 13. Prototipo funcional #2 (Fuente: Elaboración propia)

The image shows a functional prototype of a login page for 'Orion Buscador Cubano'. The page has a dark grey background. At the top center is the logo, which consists of the word 'orion' in a lowercase, sans-serif font, with 'buscador cubano' in a smaller font underneath. To the right of the text is a white outline of a five-pointed star. Below the logo, the text 'NOMBRE DE USUARIO' is centered above a white rectangular input field. Underneath that, the text 'CONTRASEÑA' is centered above another white rectangular input field. Below the password field, there is a small white square checkbox followed by the text 'RECORDAR'. At the bottom right of the form area, there is a white rectangular button with the text 'ENTRAR' centered on it.

Figura 14. Prototipo funcional #3 (Fuente: Elaboración propia)

ANEXO #2

Tabla 26. Caso de prueba Eliminar Recomendación (Fuente: Elaboración propia)

Caso de prueba Eliminar Recomendación	
Código de caso de prueba: 4	Nombre de historia de usuario: Eliminar recomendación
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad eliminar recomendaciones	
Entrada/Pasos de ejecución: Se realizan las siguientes acciones	
Se presiona el botón con el símbolo "x". Se confirma que se desea eliminar la recomendación.	
Evaluación de la prueba: Satisfactoria.	

Tabla 27. Caso de prueba Seleccionar Categoría (Fuente: Elaboración propia)

Caso de prueba Añadir Categoría	
Código de caso de prueba: 5	Nombre de historia de usuario: Seleccionar nueva categoría
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad Seleccionar nueva categoría	
Entrada/Pasos de ejecución: Se realizan las siguientes acciones	
Se presiona el botón que indica las categorías. Se marca la o las categorías que se deseen de la lista. Se presiona el botón aceptar.	
Evaluación de la prueba: Satisfactoria.	

Tabla 28. Caso de prueba Desmarcar Categoría (Fuente: Elaboración propia)

Caso de prueba Desmarcar Categoría	
Código de caso de prueba: 6	Nombre de historia de usuario: Desmarcar Categoría

Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias
Descripción de la prueba: Prueba a la funcionalidad Desmarcar categoría
Entrada/Pasos de ejecución: Se realizan las siguientes acciones
Se presiona el botón que indica las categorías. Se elige la o las categorías a desmarcar. Se presiona el botón aceptar.
Evaluación de la prueba: Satisfactoria.

Tabla 29. Caso de prueba Autenticar usuario (Fuente: Elaboración propia)

Caso de prueba Autenticar usuario	
Código de caso de prueba: 7	Nombre de historia de usuario: Autenticar usuario
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad Autenticar usuario	
Entrada1/Pasos de ejecución: Se introducen los siguientes datos para realizar la autenticación.	
Se introduce el nombre de usuario: brayan. Se introduce la contraseña: brayan123. Se presiona el botón aceptar.	
Evaluación de la prueba: Satisfactoria.	
Entrada2/Pasos de ejecución: Se introducen los siguientes datos para realizar la autenticación.	
Se introduce el nombre de usuario: brayan Se introduce la contraseña: incorrecta. Se presiona el botón aceptar.	
Evaluación de la prueba: Satisfactoria.	

Tabla 30. Caso de prueba Listar por categoría (Fuente: Elaboración propia)

Caso de prueba Listar por categoría	
Código de caso de prueba: 8	Nombre de historia de usuario: Listar por categoría
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad Listar por categoría	

Entrada/Pasos de ejecución: Se realizan las siguientes acciones
Desplegar el menú categoría. Seleccionar la categoría por que se desea filtrar. Se muestran las recomendaciones según el criterio seleccionado.
Evaluación de la prueba: Satisfactoria.

Tabla 31. Caso de prueba Mostrar por fecha (Fuente: Elaboración propia)

Caso de prueba Ordenar por fecha	
Código de caso de prueba: 9	Nombre de historia de usuario: Mostrar recomendaciones por su fecha de recomendación
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad “Mostrar recomendaciones por su fecha de recomendación”.	
Entrada/Pasos de ejecución: Se realizan las siguientes acciones.	
Se comprueba que las recomendaciones generadas estén ordenadas por la fecha en la que fueron emitidas.	
Evaluación de la prueba: Satisfactoria.	

Tabla 32. Caso de prueba Mostrar por orden alfabético (Fuente: Elaboración propia)

Caso de prueba Ordenar por orden alfabético	
Código de caso de prueba: 10	Nombre de historia de usuario: Mostrar por orden alfabético
Nombre de la persona que realiza la prueba: Leydis Arzoris Mata Arias	
Descripción de la prueba: Prueba a la funcionalidad “Mostrar recomendaciones por orden alfabético”.	
Entrada/Pasos de ejecución: Se realizan las siguientes acciones.	
Seleccionar del menú lateral el criterio Ordenar por orden alfabético. Se muestran las recomendaciones ordenadas por el criterio seleccionado.	
Evaluación de la prueba: Satisfactoria.	

ANEXO #3

Validación mediante la técnica IADOV (Encuesta)

Estimado usuario:

La siguiente encuesta de carácter anónimo nos permitirá familiarizarnos con su conocimiento y postura con respecto al Subsistema de recomendación de información para el buscador Orión. Le agradecemos que lea cuidadosamente cada pregunta antes de responder y marque con una X en las opciones que considere más acertadas. Su opinión será de gran importancia, pues permitirá evaluar la correcta implementación que supone el Subsistema de recomendación de información en cuestión.

Edad _____ Sexo _____ Nivel escolar _____

1. ¿Está familiarizado usted con la recomendación de información en buscadores?

SÍ _____ NO _____ NO SÉ _____

2. ¿Considera usted que las recomendaciones tributan a obtener resultados adecuada a sus intereses?

SÍ _____ NO _____ NO SÉ _____

3. ¿Considera relevantes las recomendaciones obtenidas a partir de las búsquedas realizadas?

SÍ _____ NO _____ NO SÉ _____

4. ¿Cree importante contar con la categorización de las recomendaciones obtenidas?

MUY NECESARIO _____ NECESARIO _____ POCO NECESARIO _____ INECESARIO _____

5. ¿Utilizaría la información recomendada como criterio de búsqueda?

SÍ _____ NO _____ NO SÉ _____

6. ¿Cree usted que este Subsistema será útil al buscador Orión?

SÍ _____ NO _____ NO SÉ _____

7. ¿Le satisfacen las recomendaciones dadas a partir de las salidas que provee y la manera en que muestra los resultados?

Me satisface mucho_____

No me satisface tanto_____

Me da lo mismo_____

Me insatisface más de lo que me satisface_____

No me satisface nada_____

No lo sé_____

Otros aspectos complementarios que se evalúan a través del cuestionario

A demás de las preguntas que permiten evaluar la satisfacción con el Subsistema de recomendación de información (preguntas 3, 6 y 7), también se pueden valorar aspectos complementarios asociados a la satisfacción. Dichos aspectos son: el grado de familiarización con los sistemas de recomendación de información (pregunta a). La importancia que atribuye el usuario a las recomendaciones en general (pregunta b). La relevancia que tiene para el sistema emitir recomendaciones teniendo en cuenta las categorías (pregunta 4).