Universidad de las Ciencias Informáticas

Centro de Informatización Universitaria

Facultad 1

Título: Aplicación del Modelo Vectorial a la Recuperación de Información en un Proveedor de Servicios OAI-PMH.

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

Autor:

Daimel Rolando Rosales Puig.

Tutores:

Ing. Maikel Manuel Fernández Fernández.

Ing. Yanet Del Carmen De Diego Ceruto.

La Habana, Junio 2013

DECLARACIÓN DE AUTORÍA

Declaro ser el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a que haga el uso que estime pertinente con esta investigación. Para que así conste firmo la presente a los_____ días del mes_____ del año_____.

Daimel Rolando Rosales Puig

_____

Firma del Autor

Maikel M Fernández Fernández

_____

Firma del Tutor

Yanet del Carmen De Diego Ceruto

_____

Firma del Tutor

A toda mi familia por el apoyo que me han dado y la preocupación que han demostrado, especialmente a mi abuela y mi mamá, que representan el motor impulsor para cualquier tarea que realizo. A Cony por apoyarme siempre y darme muchas fuerzas para terminar, por estar siempre pendiente de mí y hacer que la distancia que nos separa no sea un obstáculo para querernos mucho. A mis tutores Yanet y Maikel por todo su apoyo y dedicación ofrecida, muchas gracias. A todos mis amigos de la infancia a los nuevos y a los de siempre. A Tony y a Diosbel por ayudarme durante el proceso de elaboración de la investigación. A Alex y a Giselle que aunque son de los últimos en llegar son de los que van a quedar por siempre.

A mi abuela Elba: *por verme nacer, crecer, por educarme y cuidarme como toda una madre, por siempre darme todo su apoyo durante estos veinticuatro años de mi vida, por sus consejos, por siempre estar pendiente a cada paso que doy, por apoyarme y quererme mucho.*

A mi mamá: *por siempre guiarme por el camino correcto, por educarme correctamente desde que comencé a andar, por su preocupación y constancia hacia mí, por apoyarme en cualquier decisión que tome.*

A mi papá: *por sus consejos y ayuda durante estos años de la carrera.*

A Cony: *por hacerme creer una vez más en el amor, por quererme mucho, por hacerme hacer cosas que solo una persona enamorada es capaz de hacer, por ser toda mía para amarle noche y día.*

A mi tío Pedro Puig: *por su preocupación y apoyo durante estos años de la carrera, por sus consejos, por ser un paradigma para mí.*

A mis primos Yoan, Yoannis, Yasnier: *por estar juntos desde que nacimos, por ser un referente para mí.*

A mis hermanos Daymarelis, Jesús, Rafael, Raciel: *Aunque la vida no me dio la oportunidad de tener hermanos de sangre a ustedes los quiero como tal, por ayudarme siempre, escucharme, por estar en los momentos malos y buenos, por siempre tenernos presentes.*

Resumen.

El presente trabajo expone la implementación del modelo vectorial para la recuperación de información, tomando como contenido los documentos gestionados en el Repositorio Institucional de la Universidad de las Ciencias Informáticas. El trabajo persigue mejorar la recuperación de información en términos de precisión, motivado por algunas limitantes, que en este sentido presenta la herramienta DSpace, la cuál es la base del desarrollo del repositorio antes mencionado. En el trabajo se describen los diversos modelos clásicos para la recuperación de información, en especial el vectorial. Se detallan las etapas del algoritmo elaborado para la implementación del modelo, pasando por la selección de términos, la eliminación de palabras vacías, el proceso de *stemming* y la definición de la métrica de comparación entre consulta y documentos. Aunque el objetivo del trabajo no es el desarrollo de una aplicación de software, sí se construye una simple interfaz que permite visualizar las respuestas que produce el modelo. Para esto se emplean herramientas y lenguajes como: el IDE de desarrollo *NetBeans*, el lenguaje PHP y algunos artefactos para representar elementos de interés como el modelo de dominio y diagrama de clases. Finalmente se realizan pruebas de precisión sobre una muestra de 500 documentos y se constatan los resultados con los que arroja el motor de búsqueda de DSpace para la misma muestra. El resultado obtenido valida la premisa de una mejora de precisión mediante el empleo del modelo vectorial.

Palabras clave: recuperación de información, repositorio institucional, modelo vectorial, servicios OAI-PMH.

Índice de contenidos

Índice de figuras

Índice de tablas

Introducción.

Los avances tecnológicos de los últimos años como la creación de la *World Wide Web* en 1989 por Berners Lee han provocado un aumento de la información y además una mejora de su difusión. Cada vez hay más información disponible y mayores posibilidades para acceder a ella, lo cual ha incidido en el aumento de la relevancia de la Recuperación de Información (RI). La RI tiene como objetivo recuperar todos los documentos relevantes y al mismo tiempo obviar todos los documentos que son irrelevantes ante la formulación de una consulta por parte de un usuario expresada en lenguaje natural. La RI es una tarea importante ya que de nada sirve tener una colección estática sin que nunca muestre lo que tiene (1).

Existen varios modelos de recuperación de información que permiten realizar dicha tarea tales como el booleano, el booleano extendido, el de indexación latente semántica y el vectorial. Estos modelos tienen una gran diferencia con los algoritmos de búsqueda, ya que proporcionan un valor que dice qué tan relevante puede ser el documento para el usuario.

Al hablar de modelos de recuperación de información se refiere a la aplicación de teorías para generar procesos que den una jerarquización de los documentos y que permitan lograr un modelado de varios aspectos de un sistema de RI (2).

En la biblioteca de la Universidad de las Ciencias Informáticas se realiza la RI a través del servicio de Repositorio Institucional, en este repositorio se almacena todo tipo de investigación científica y académica realizada en la institución, dígase tesis, maestrías y doctorados, con el objetivo de servir de retroalimentación a futuras investigaciones. Actualmente el repositorio utiliza la herramienta *DSpace* para realizar la Recuperación de Información pero la misma presenta dificultades tales como la de realizar las búsquedas por similitud teniendo en cuenta la creación de índices, no se organiza la información teniendo en cuenta un ranking para devolvérsela al usuario, las búsquedas se hacen en profundidad y varios documentos responden a una misma consulta. No se utiliza un método o un modelo para realizar este proceso. En el Repositorio Institucional el intercambio de información se realiza utilizando el protocolo OAI-PMH.

Una vez identificados los problemas existentes actualmente en el Repositorio Institucional de la biblioteca se presenta el siguiente **problema de la investigación**: ¿Cómo mejorar la recuperación de información en un proveedor de servicios OAI-PMH?

Se define para la presente investigación como **objeto de estudio** los modelos de recuperación de

información.

Se tienen las siguientes **preguntas de investigación**:

- ¿Si se aplica el modelo vectorial se mejora la precisión en la recuperación de información?
- ¿Cuáles son las diferentes tecnologías y lenguajes que se pueden utilizar para elevar el nivel de la recuperación de información en el Repositorio Institucional de la Universidad de las Ciencias Informáticas?
- ¿Qué pasos seguir para implementar el modelo vectorial?
- ¿Cómo validar la implementación final del modelo aplicado?

Para dar solución al problema existente se ha tomado como **objetivo general**: Implementar el Modelo Vectorial mediante la selección de los metadatos título, resumen y palabras clave del estándar *Dublin-Core*, para mejorar la precisión en la recuperación de información en un proveedor de servicios OAI-PMH.

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

- Construir el marco teórico conceptual de la investigación.
- Crear una estructura de datos para aplicar el modelo vectorial para la recuperación de información.
- Implementar el modelo vectorial para la recuperación de información.
- Evaluar en términos de exhaustividad y precisión los resultados de la aplicación del modelo.

El **campo de acción** está enmarcado en el modelo vectorial para mejorar el proceso de recuperación de información en el repositorio institucional de la biblioteca.

Para dar respuesta a la interrogante presentada en esta investigación y con los objetivos trazados se plantean las siguientes **tareas de investigación**:

1. Valoración de diferentes modelos de recuperación de información.
2. Selección de las métricas a emplear para las comparaciones consulta-documentos.
3. Descripción de la arquitectura OAI-PMH.
4. Conversión de los documentos del repositorio en vectores.
5. Construcción de una estructura de datos para almacenar los documentos procesados.
6. Aplicación del modelo.

7. Validación de la aplicación sobre las colecciones Reuters y ACM.

La investigación estará sustentada en los siguientes métodos científicos:

- histórico-lógico: el empleo de este método posibilitará la comprensión lógica del objeto de estudio, haciéndose un análisis de sus antecedentes.
- analítico-sintético: permitirá la consulta de diversas fuentes bibliográficas y la obtención de los elementos más importantes que se relacionan con el objeto de estudio, este método será de gran utilidad en el estudio del estado del arte.
- encuesta: este método facilitará la comparación entre el modelo que se propone aplicar y el que está vigente actualmente en el repositorio para ver si se ha mejorado en la precisión de la recuperación de la información.

La presente investigación está dividida en 3 capítulos, distribuidos de la siguiente forma:

Capítulo 1. Diseño del marco teórico, estado del arte: se muestran los conceptos asociados al dominio del problema, además se fundamentan los lenguajes, las herramientas y tecnologías que se utilizarán para el desarrollo de la propuesta de solución.

Capítulo 2. Análisis e implementación: se describe la solución propuesta, se especifican los requerimientos funcionales en la aplicación del modelo, además se detalla cómo se realiza la implementación del algoritmo.

Capítulo 3. Validación y prueba: se valida la propuesta de solución y se muestran los principales beneficios de la aplicación del modelo aplicado.

Capítulo 1. Elementos teóricos sobre el modelo vectorial para la recuperación de información.

Al contextualizar el problema de la investigación se propone **Aplicar el Modelo Vectorial para la recuperación de información en un proveedor de servicios OAI-PMH** como parte de la renovación del sistema de información de la universidad y del repositorio institucional, con el objetivo de mejorar el proceso de búsqueda de información en el repositorio.

Este capítulo se inicia explicando en qué consiste la problemática de la recuperación de información, así como la arquitectura de un Sistema de Recuperación de Información. Se explican los diferentes modelos que existen para la recuperación de información y la clasificación de cada uno. Posteriormente se hace un estudio detallado del modelo vectorial indicando las características, el funcionamiento y otros aspectos del mismo. También se explica cómo se desarrolla el proceso de indexado para realizar la recuperación de información y las diferentes etapas que este posee. Se describe el entorno OAI-PMH enfatizando en el acceso abierto así como en el proveedor de datos y el de servicios del entorno OAI-PMH. Por último se describen dos sistemas que implementan el modelo vectorial para realizar la RI y se describe el entorno tecnológico a utilizar en el desarrollo de la investigación.

1.1 La problemática de la recuperación de información (RI).

El problema de la RI puede ser estudiado desde dos puntos de vista, el computacional y el humano. Desde el punto de vista computacional tiene que ver con la construcción de estructuras de datos y algoritmos eficientes que mejoren la calidad de las respuestas; en el caso del humano tiene que ver con el comportamiento y las necesidades del usuario (3)**.**

La problemática de la RI plantea que existe una colección de documentos que contienen información de interés sobre uno o varios temas, por otra parte se encuentran los usuarios que necesitan acceder a la información y acceden a la misma a través de una petición expresada en forma de consulta. Como respuesta a esta petición el sistema retorna referencias a los documentos relevantes, es decir a los documentos que satisfacen las necesidades del usuario. A continuación se muestra una imagen que ilustra lo expuesto anteriormente.

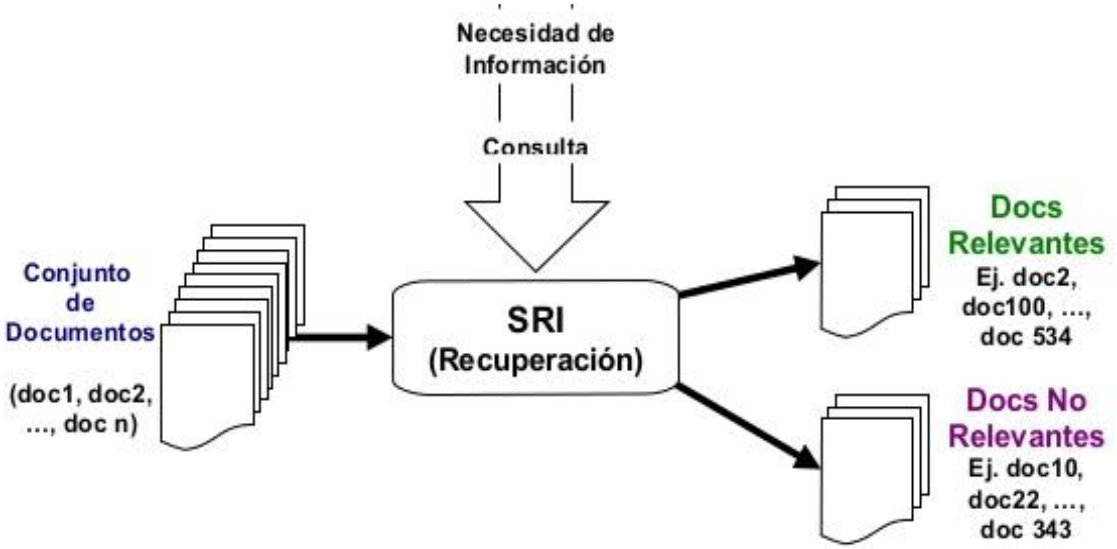

Figura 1. Problemática de la Recuperación de Información. (4)

La respuesta ideal de un Sistema de Recuperación de Información (SRI) es que muestre sólo los documentos que son verdaderamente relevantes a la consulta realizada por el usuario, pero en la práctica esto aún no es alcanzable ya que existe el problema de compatibilizar la expresión de la necesidad de información y el lenguaje de los documentos, además hay una carga de subjetividad subyacente y depende de los usuarios (4). Entonces el SRI recupera la mayor cantidad de documentos relevantes, minimizando la mayor cantidad de documentos no relevantes en la respuesta. Cuando se habla de recuperar los documentos relevantes hay que hablar de cuán preciso es el conjunto solución, es decir, mientras más documentos relevantes contenga la respuesta para una consulta dada, más preciso será el SRI.

El SRI para cumplir con sus objetivos debe realizar algunas tareas básicas las cuales se refieren a cuestiones computacionales. Primeramente se hace una representación lógica de los documentos de la colección y opcionalmente almacenamiento del original, algunos sistemas lo almacenan de manera completa. También se hace una representación de la necesidad de información del usuario en forma de consulta, se evalúan los documentos respecto a la consulta para establecer la relevancia de cada uno y son devueltos al usuario los documentos relevantes en forma de ranking como parte del conjunto solución.

En la siguiente figura se puede apreciar en mayor detalle la arquitectura de un SRI, el tratamiento que se le da a los documentos y la interacción con el usuario.

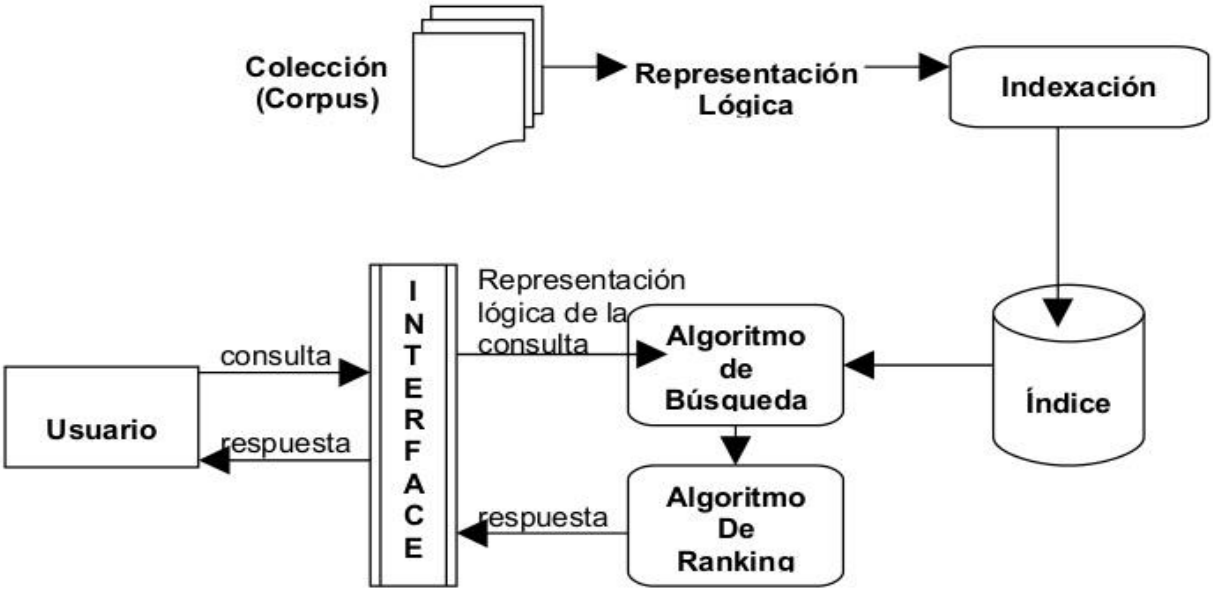

Figura 2 . Arquitectura de un SRI. (4)

Se parte de un conjunto de documentos de texto, los cuales están compuestos por sucesiones de palabras que forman estructuras gramaticales, estos documentos están escritos en lenguaje natural. Para realizar cualquier operación sobre la colección primero se debe hacer una representación lógica de todos sus documentos la cual puede consistir en un conjunto de términos, frases u otras unidades que permitan caracterizarlos. Luego de la representación lógica de los documentos se realiza un proceso de indexación en el cual se construyen estructuras de datos o índices que almacenen tal representación.

El algoritmo de búsqueda acepta como entrada la consulta hecha por el usuario y verifica en el índice qué documentos pueden satisfacerla. Luego un algoritmo de ranking determinará la relevancia de cada documento para posteriormente ser devuelta al usuario.

La interfaz es utilizada para que el usuario pueda expresar su consulta en un lenguaje natural y también para posteriormente mostrarle el conjunto solución devuelto por el sistema.

1.2 Modelos para la recuperación de información.

Los SRI toman un conjunto de documentos para procesar y responder a las consultas hechas por el usuario. Los documentos se pueden clasificar en estructurados y no estructurados. En los estructurados se pueden identificar elementos estructurales con una semántica bien definida, sin embargo los no estructurados corresponden a texto libre, sin formato. La diferencia de un SRI que procese documentos estructurados es que puede extraer información adicional al contenido textual, la cual utiliza en la etapa

de recuperación para facilitar y aumentar las prestaciones.

Existen una serie de modelos matemáticos teniendo en cuenta cada una de estas clasificaciones que se utilizan para la recuperación de información. En los estructurados se utilizan modelos basados en el álgebra de regiones y en el caso de los no estructurados se utilizan el modelo booleano, probabilístico y vectorial. A continuación se describen brevemente cada uno de ellos y posteriormente se profundizará más en el vectorial ya que es el que ocupa la presente investigación.

1.2.1 Modelos para documentos estructurados.

Un modelo de recuperación de documentos estructurados utiliza la estructura de los documentos con el objetivo de brindar servicios alternativos al usuario, por ejemplo uso de memoria visual, recuperación de elementos multimedia y mayor precisión en cuanto a la consulta. La estructura de los documentos a indexar está dada por marcas o etiquetas, las más utilizadas son el SGML (*Standard General Markup Language*), el HTML (*Hiper Text Markup Language*), el PDF (*Portable Document Format*) y el XML (*Extensible Markup Language*).

Según Baeza-Yates existen dos modelos en esta categoría, la de nodos proximales (5) y listas no superpuestas (6). Estos modelos consisten en almacenar las ocurrencias de los términos a indexar en estructuras de datos diferentes, según aparezcan en algún elemento de estructura o en otro como capítulos, secciones. En general las regiones de una misma estructura de datos no poseen superposición, pero regiones en diferentes estructuras sí se pueden superponer.

1.2.2 Modelos para documentos no estructurados.

Los modelos clásicos responden a consultas hechas por el usuario. En estos modelos se definen estructura de datos que representan el contenido de los documentos de una colección. En esta estructura se almacenan los términos relevantes a cada uno de los documentos de la colección. A continuación se explica brevemente cada uno de estos modelos clásicos.

1.2.2.1 Modelo Booleano.

En este modelo la representación de los documentos se hace sobre una matriz binaria documento-término. Los términos son extraídos manualmente o automáticamente de los documentos y estos representan el contenido de los mismos.

Las consultas se arman con términos vinculados por operadores lógicos AND, OR, NOT y los resultados son referencias a documentos donde cuya representación satisface las restricciones lógicas de la

expresión de búsqueda. En este modelo no se establece un ranking a la hora de devolverle los resultados al usuario y todos los documentos del conjunto solución tienen la misma relevancia.

1.2.2.2 Modelo Probabilístico.

Fue propuesto por Robertson y Spark-Jones (7) con el objetivo de representar el proceso de recuperación de información desde el punto de vista de las probabilidades. A partir de una expresión de consulta se divide una colección de N documentos en cuatro subconjuntos distintos: REL (conjunto de documentos relevantes), REC (conjunto de documentos recuperados), RR (conjunto de documentos relevantes recuperados) y NN el (conjunto de documentos no relevantes no recuperados).

El resultado ideal a una consulta se da cuando el conjunto REL es igual al REC. Como resulta difícil lograrlo en primera intención, el usuario genera una descripción probabilística del conjunto REL y a través de sucesivas interacciones con el SRI se trata de mejorar el rendimiento de la recuperación, dado que una recuperación no es inmediata, ya que involucra varias interacciones con el usuario.

1.2.2.3 Modelo Vectorial.

El modelo vectorial fue planteado y desarrollado por Gerard Salton (8) y originalmente se implementó en un SRI llamado SMART. Aunque el modelo posee más de treinta años, actualmente se sigue utilizando debido a su buen rendimiento en la recuperación de documentos. Este modelo es una mejora del modelo booleano, ya que reconoce que los pesos binarios son muy limitados y no proporcionan ningún valor intermedio.

El modelo de recuperación vectorial o de espacio vectorial propone un emparejamiento parcial a diferencia del modelo booleano, asignando pesos no binarios a los términos de las preguntas del usuario y de los documentos. Estos pesos de los términos se utilizan para calcular el grado de similitud entre cada documento de la colección y la consulta realizada por el usuario. Los documentos recuperados son ordenados en orden decreciente teniendo en cuenta el grado de similitud, solo se consideran los documentos que se relacionan parcialmente con la pregunta, así el conjunto de la respuesta es mucho más preciso que el conjunto recuperado por el modelo booleano.

La idea básica de este modelo consiste en la construcción de una matriz que contiene el vocabulario de la colección de referencia y los documentos existentes, donde las filas se corresponden a los documentos y las columnas a los términos contenidos en estos. En la intersección de un término y un documento se almacena un valor numérico de importancia que significa la frecuencia de aparición de ese término en el documento. Cada fila de la matriz representa un vector, la longitud de este está en dependencia de la cantidad de términos que contenga ese documento. En teoría, los documentos que

contengan términos similares estarán a muy poca distancia entre sí sobre tal espacio.

De igual forma se trata a la consulta, es un documento más y se la mapea sobre el espacio de documentos. Luego, a partir de una consulta dada es posible devolver una lista de documentos ordenados por distancia (comenzando por los más relevantes).

Para realizar esta operación se utilizan como por ejemplo la medida del coseno del ángulo entre los vectores, la distancia euclídea, el producto escalar de dos vectores, siendo la primera una de las más utilizadas. Esta función normaliza los vectores respecto a su longitud. Consiste en calcular el producto escalar entre el vector-consulta y el vector-documento y dividirlo por la raíz cuadrada del sumatorio de los componentes del vector consulta multiplicada por la raíz cuadrada del sumatorio de los componentes del vector-documento.

$$sem(p, di) = \frac{\sum_{j=1}^{n} wpj \times wij}{\sqrt{\sum_{j=1}^{n} wpj^2} \times \sqrt{\sum_{j=1}^{n} wij^2}} = \cos(a) \qquad [1]$$

De esta manera se calcula el valor de similitud. Si no hay coincidencia alguna entre los componentes de los vectores porque su producto escalar es cero, se dice que estos son ortogonales o lo que es igual a que su coincidencia es cero. La similitud máxima se alcanza cuando todos los componentes de los vectores son iguales, en este caso la función del coseno obtiene su máximo valor, que es uno.

1.2.2.4 Consideraciones sobre los modelos de Recuperación de Información.

El modelo booleano asigna pesos binarios a los términos en los documentos, lo que conlleva a que en el conjunto respuesta todos los documentos presenten la misma relevancia. Por su parte el modelo probabilístico presenta como desventaja que es iterativo, lo que trae como consecuencia que si el usuario en una primera búsqueda no encuentra la información que necesita debe interactuar varias veces con el sistema de recuperación de información para ir refinando los resultados. Sin embargo el modelo vectorial se destaca por el tratamiento que le da a los términos de los documentos, ya que propone asignar pesos no binarios que representan el nivel de importancia del término en el documento, con el objetivo de que la información recuperada desde la primera búsqueda realizada sea la más relevante a la consulta del usuario.

1.3 El Proceso de Indexado en la Recuperación de Información.

Para implementar un mecanismo de recuperación sobre una colección de documentos de texto es necesario obtener una representación de los mismos. Esta representación consiste en la implementación

de un conjunto de criterios mediante los cuales se obtienen los términos y las relaciones entre éstos. A la hora de implementar un Sistema de Recuperación de Información se comienza con el procesamiento de la colección. Esto se debe a que no todos los términos que componen un documento son representativos al contenido del mismo. Cuestiones como la posición, la cantidad de ocurrencias definen el grado de importancia de los términos. El resultado de este proceso es la representación de la colección que es computacionalmente adecuada para los procesos siguientes y se describe como indexación de la colección.

La indexación es un proceso que consiste en la identificación de los términos que representan al contenido de un documento y la traducción de estos a una forma que computacionalmente sea manejable (9).

En la RI la indexación incluye la creación de estructuras de datos para posteriormente almacenar los términos representativos para soportar la recuperación. El proceso de indexación puede realizarse desde dos enfoques, el basado en métodos no lingüísticos y métodos lingüísticos. En el primero se utilizan técnicas para análisis de frecuencias y cálculo de pesos de los términos; y en el segundo caso se utilizan técnicas derivadas del procesamiento del lenguaje natural.

1.3.1 Vista lógica de los documentos.

Es muy importante la manera en que se almacena la información. Esto ayuda a la eficiencia y precisión que existe en el proceso de la recuperación de información. La vista lógica se refiere a la manera en que se representa un documento dentro de sus índices. La mejor forma de representar un documento es por medio del conjunto de palabras del texto completo, pero este conjunto de palabras puede ser muy grande y por lo tanto lo más conveniente es reducirlo a las palabras claves del texto. Así se obtiene la primera forma de representación que es la de texto completo. Las vistas lógicas de los documentos pueden variar en dependencia de los tipos de operaciones que se realicen sobre el texto, entre las cuales destacan (3):

- La eliminación de palabras vacías que consiste en quitar de la lista de palabras aquellas que no dan ningún tipo de información, como son los artículos y los conectores.
- La lematización morfológica, llevando las palabras a su raíz, así una palabra que está en plural se lleva a singular y un verbo conjugado se reduce a su infinitivo.
- Agrupar las palabras representativas a sus sinónimos.
- Representación de frases que consiste en llevar las frases a su forma más simple.

La aplicación de una o varias de estas operaciones muestra diferentes vistas lógicas que representan las

colecciones de documentos, que varían desde una vista completa del texto a los descriptores generales de los documentos, permitiendo un rápido acceso a la información y logrando la representación del contenido del documento.

1.3.2 El Proceso de Indexación.

Las etapas de la indexación en esta investigación son las siguientes:

- Análisis lexicográfico, en el cual se extraen las palabras y se normalizan.
- Eliminación de palabras vacías o de alta frecuencia en la colección.
- Reducción de palabras morfológicamente parecidas a una forma base, con la finalidad de aumentar la eficiencia del SRI, este proceso se conoce como *stemming*.
- Se seleccionan los términos a indexar, consiste en extraer las palabras simples o compuestas que mejor representen el contenido de los documentos.
- Asignación de pesos a los términos que componen los índices de los documentos.

A continuación se explica cada una de estas etapas.

1.3.2.1 Análisis lexicográfico.

El objetivo de esta etapa es convertir un grupo de caracteres en palabras o *tokens*. Esto conlleva a detectar el comienzo y fin de las palabras bajo diversas circunstancias. El analizador debe ser capaz de tratar con símbolos no alfabéticos como dígitos y caracteres especiales que componen las palabras, los cuales generalmente se reemplazan por el carácter base relacionado. Además se debe expandir y normalizar siglas y tratar guiones como conectores de términos.

1.3.2.2 Eliminación de palabras vacías.

Los términos que aparecen en la mayoría de los documentos de la colección no son buenos para la recuperación de información ya que a la hora de realizar una consulta no aportan información al usuario. Estas palabras no son recomendables para seleccionar como término a indexar. Este conjunto de palabras están formadas por artículos, preposiciones, conjunciones y forman lo que se conoce como diccionario negativo o anti diccionario. Una ventaja de este procedimiento es que reduce el tamaño de la colección. La lista de palabras vacías que se utilizan en la aplicación del modelo son las propuestas por el SRI SMART.

1.3.2.3 *Stemming*.

Es una técnica de reducción que permite detectar variantes morfológicas de un mismo término (por ejemplo palabras como cómputo, computadoras, computable, computación son variantes del término computar) y reemplazarlas por el término raíz. El uso de *stemming* posibilita tener índices de menor tamaño y una mayor cantidad de respuestas a una consulta dada, debido a que al aplicarse lematización al *corpus* (colección de documentos) y a la consulta se recuperan documentos que tienen todas las variantes morfológicas de los términos contenidos en la consulta. La colección que se va a procesar en esta investigación se encuentra en el idioma español y el algoritmo que se utilizará para realizar el *stemming* es una variante del algoritmo de *Porter*[1] para las palabras en español, ya que este algoritmo fue desarrollado para los documentos en inglés (9).

1.3.2.4 Selección de los términos a indexar.

Aquí se analiza cada documento a indexar con el fin de extraer las palabras simples o compuestas, que representen de mejor forma el contenido. La importancia del término debe ser determinada en base a la frecuencia de aparición en el documento a procesar. Las palabras de alta frecuencia se eliminan con la asistencia de un diccionario de palabras vacías y las palabras de baja frecuencia también se eliminan pero de manera opcional.

1.3.2.5 Asignación de pesos.

Para realizar operaciones de cálculo de similitud entre documentos y consultas, resulta necesario poder determinar la ponderación de cada término dentro de estos, brindando a cada uno un peso o valor. El peso de cada término se basa en estudios realizados por *Luhn*[2] (10) y se encuentra relacionado con su frecuencia de aparición dentro de cada documento.

La tarea de ponderación o determinación de pesos puede ser considerada como de reducción de dimensionalidad. Esto se debe a que actúa como filtro para determinar cuáles son los términos que sobrepasen un umbral de aceptación (llamadas palabras raras) o que no lo alcancen (llamadas palabras comunes) y por lo tanto no se tendrán en cuenta para representar el contenido del documento. Inicialmente, el peso de un término en un documento se puede determinar por la aparición o no del término o se calcula a partir de obtener su frecuencia. Uno de los métodos que se utiliza para determinar esta frecuencia de aparición es el de Tf-IDf que es el que será aplicado.

[1] "El algoritmo de Porter es uno de los algoritmos de extracción de raíces de palabras en inglés más utilizado. Disponible en Internet a través de la dirección: http://www.tartarus.org/~martin/PorterStemmer".
[2] Luhn. Pionero en Ciencias de la Información. Justificó el uso de frecuencias como métrica a partir del siguiente concepto: "…un escritor repite normalmente ciertas palabras mientras avanza o varía sus argumentos...".

1.3.2.5.1 Método Tf-IDf.

Los pesos de los términos se utilizan para propiciar la jerarquización de los documentos, existen varios esquemas que proponen lograr esto, unos se basan en las frecuencias de los términos en los documentos de las colecciones y otros lo proponen a través de la normalización. Esta investigación se basa en el esquema Tf-IDF. Con este esquema se puede obtener el peso de un término dentro de un documento y de la colección, además proporciona también una medida para obtener el peso de cada término dentro de la consulta.

Para obtener el peso de los términos de un documento se utiliza la fórmula:

$$w_{i,j} = f_{i,j} \times idf_i \qquad [2]$$

Donde la expresión $w_{i,j}$ representa el peso del término *i* en el documento *j*, siendo $f_{i,j}$ la frecuencia normalizada del término *i* en el documento *j* y por último idf_i representa la frecuencia invertida del documento *i*. Las medidas anteriores se obtienen utilizando las fórmulas:

$$f_{i,j} = \frac{frec_{i,j}}{\max frec_j} \qquad [3]$$

$$idf_i = \log \frac{N}{n_i} \qquad [4]$$

En donde $frec_{i,j}$ es la frecuencia del término *i* en el documento *j* (el número de veces que aparece el término en un documento) y $\max frec_j$ representa la máxima frecuencia sobre todos los términos del documento *j*. *N* representa al número de documentos de la colección y n_i es la cantidad de documentos que contienen el término *i*.

Para obtener el peso de los términos de una consulta Salton y Buckley en el año 1987 sugirieron la fórmula:

$$w_{i,q} = \left[0.5 + \frac{0.5 frec_{i,q}}{\max frec_q}\right] \times \log \frac{N}{n_i} \qquad [5]$$

Donde $frec_{i,q}$ representa la frecuencia del término *i* en la consulta *q* y $\max frec_q$ es la máxima frecuencia sobre todos los términos de la consulta *q*. *N* y n_i fueron definidos anteriormente.

1.4 El entorno *OAI-PMH.*

OAI-PMH (en inglés, *Open Archives Initiative – Protocol for Metadata Harvesting*) es un protocolo para el intercambio de metadatos. Surge para normar la distribución de contenidos en el escenario del acceso abierto (OA, por sus siglas en inglés). Para establecer las características del protocolo y los requerimientos tecnológicos para alcanzar el acceso abierto existe la *Open Access Initiative* (OAI).

La OAI apoya la difusión de contenidos en internet, para mejorar el acceso a publicaciones electrónicas, además se encarga de normar aspectos legales en cuanto a las licencias de los documentos.

El paradigma de acceso abierto se enfrenta a un problema que está relacionado con la interoperabilidad que es necesario lograr para garantizar la distribución y el consumo de la información (11). OAI-PMH establece un conjunto de normas que rigen la comunicación independientemente de la tecnología que se emplee para la gestión de los documentos. La implantación del protocolo se divide en dos componentes: el proveedor de datos y el proveedor de servicios.

El protocolo OAI-PMH se caracteriza por ser sencillo. Se basa en peticiones HTTP mediante GET o POST y respuestas XML, además de abogar por el uso de *Dublin-Core* como estándar de metadatos para distribuir los documentos. Los proveedores de datos son los encargados de exponer los metadatos de sus contenidos. Los proveedores de servicios son aquellas herramientas desarrolladas para realizar peticiones a los proveedores de datos, indexar contenidos e interpretar resultados.

1.4.1 El proveedor de servicios *OAI-PMH*.

Muchas instituciones prefieren implementar su propio proveedor de servicios. Cuando se tiene implementado un proveedor de servicios se cuenta con una herramienta que permite la indexación de contenido y además que provee una interfaz para la recuperación de información (12).

El proveedor de servicios debe tener presente:

- La selección y validación de proveedores de datos.
- El control de flujo.
- La planificación de recolecciones.
- La recolección basada en Harvester.
- La detección de contenido duplicado.
- El análisis de la granularidad de las fechas.
- Interfaz de usuario para la interacción con el público.

1.4.2 El proveedor de datos *OAI-PMH*.

Un proveedor de datos está compuesto por:

- Intérprete para validar las peticiones (puede estar implementado en cualquier lenguaje de programación).
- Generador de errores con respuestas XML.
- Interfaz de acceso a datos para extraer los metadatos.
- Generador XML para emitir las respuestas.
- Servidor Web.
- Servidor de Base de datos.

Por el tema de la eficiencia y de hacer operativo un proveedor de datos, es necesario tener presente además de los estándares, el proceso de control de flujo. Un repositorio documental puede estar compuesto por miles de documentos, teniendo en cuenta que se usan los quince campos de *Dublin-Core* para describir cada documento, entonces no es razonable una respuesta que incluya más de 200 registros, porque el resultado XML sería muy grande y esto podría generar retardos, ineficiencia y en el peor de los casos interrupciones en el proceso de comunicación. El control de flujo viene a solucionar esta dificultad y se encarga de segmentar los resultados y dejar un indicador como referencia del último resultado obtenido. El control del flujo es un tema prácticamente indispensable.

1.5 Estándar *Dublin-Core.*

El estándar *Dublin-Core* es un conjunto básico de elementos de metadatos (indican estructura, organización y facilitan la interpretación); requerido para facilitar la recuperación de objetos como documentos. *Dublin-Core* se concentra en la descripción de las propiedades del objeto tales como el contenido intelectual reflejado por el título, autor o fuente y por la forma física que se refiere al formato del documento (22). Además contiene quince elementos de metadatos divididos en tres grupos:

- Contenido: título, tema y palabras clave, descripción, fuente, idioma, relación, cobertura.
- Propiedad intelectual: autor, editor, colaborador, derechos.
- Instanciación: fecha, tipo, formato, identificador.

Aunque estos elementos son opcionales esto no le quita mérito al estándar ya que es preferible tener descripciones simples utilizando solo algunos de los metadatos en lugar de ninguna. Para aplicar el modelo vectorial solo se tuvieron en cuenta los metadatos correspondientes al título, palabras clave y la descripción porque son los que están dirigidos a reflejar el contenido del documento, el resto de los

campos se refieren a propiedad intelectual y contexto de espacio y tiempo.

1.6 Aplicaciones que implementan la recuperación de información.

En Cuba aunque no se ha trabajado mucho en este tema si hay centros donde se implementan aplicaciones para realizar la recuperación de información, un ejemplo de ello es en la Universidad de las Ciencias Informáticas en la que el repositorio de la biblioteca está implementado bajo el modelo booleano para realizar la RI.

1.6.1 Sistema de Recuperación de Información SMART.

Basado en el modelo de espacio vectorial el Sistema de Recuperación Smart (13) ha sido y sigue siendo referencia para muchos investigadores de este tema. Fue desarrollado en los años sesenta en la Universidad de Cornell por un grupo de desarrolladores dirigidos por Gerard Salton. Este sistema está formado por un grupo de programas que componen un sistema completo de recuperación automática de documentos.

Permite la creación, mantenimiento y uso de colecciones de documentos, de tamaños pequeños a medios. Se desarrolló con el objetivo de ser una herramienta experimental para investigar métodos y técnicas de RI. Es una herramienta flexible, apta para la experimentación, presenta un entorno rápido, portable e interactivo aunque no posee una interfaz gráfica para el usuario.

Este SRI está compuesto por cuatro módulos básicos:

- Módulo de indexación: se encarga de convertir cualquier colección de documentos en su formato original a su representación en vectores.
- Módulo de recuperación: calcula la similitud, basada en la función del coseno, entre los documentos (ya indexados previamente) y una consulta, generando como resultado una lista ordenada decrecientemente por dicha similitud de todos los documentos de la colección que es mostrada al usuario.
- Módulo de realimentación de relevancia: a partir de los resultados de una consulta ya formulada y de los juicios de relevancia expresados por el usuario, genera una nueva consulta para recuperar más documentos relevantes.
- Módulo de evaluación: a partir de la lista ordenada y de los juicios de relevancia establecidos en las colecciones de prueba, genera las curvas de exhaustividad-precisión.

El sistema está desarrollado en el lenguaje C y su código se puede distribuir gratuitamente, debido a esto se pueden utilizar todas las rutinas desarrolladas para implantar estos cuatro módulos anteriores,

de tal forma que desde un programa externo se puede acceder a la información que almacena este SRI sin necesidad de utilizarlo como medio para ello.

1.6.2 Sistema de Recuperación de Información Karpanta.

Este sistema es un motor de recuperación de información completamente operacional, utiliza el modelo vectorial empleando el lenguaje natural para la entrada de consultas. También admite realimentación de consultas, aunque en la versión web no está implementada esta tarea (14). Este SRI fue desarrollado con el objetivo de diseñar una herramienta para la docencia y la investigación. El resultado fue lo suficientemente robusto como para ser utilizado en distintos entornos documentales.

En cuanto a la arquitectura Karpanta se apoya en dos módulos, uno de indización, que construye los vectores de documentos y consultas y otro de búsqueda que obtiene los documentos más similares a una consulta dada. Utiliza el Sistema Gestor de Bases de Datos (SGBD) *Microsoft Access*, por su facilidad de uso, transparencia del sistema y facilidades docentes (15).

En el módulo de indización se determina cuáles son los términos índices de la colección. Este módulo realiza el preprocesado del texto, que trae aparejado varias acciones (16). La primera consiste en un análisis léxico inicial con el objetivo de determinar el tratamiento que se realizará sobre números, guiones, signos de puntuación, tratamiento de mayúsculas, nombres propios, siglas. El tratamiento léxico en el sistema es bastante sencillo, solo se sustituyen las vocales acentuadas por las no acentuadas y se convierten todos los términos a mayúsculas. La segunda acción que realiza es la eliminación de palabras vacías con el objetivo de reducir el número de términos índices. Luego se aplica lematización (17) a las palabras, con el objetivo de buscar variaciones morfológicas a los términos para extraer la raíz común a ellos. El último de los pasos que se realiza en el preprocesado de los términos es la construcción o aplicación de técnicas que permitan expandir la consulta.

En el módulo de búsqueda el sistema posee una interfaz de usuario fácil e intuitivo. El sistema al recibir una consulta, esta primero pasa por el módulo de indización, que selecciona los términos índice y asigna los pesos adecuados a dichos términos. Karpanta acepta solo consultas en lenguaje natural, es decir el usuario plantea su necesidad informativa sin utilizar operadores booleanos. Los términos junto con el peso se almacenan entonces en otra tabla de la base de datos. Luego se realiza el cálculo de similitud entre la consulta y todos los documentos, para ello utilizan sentencias SQL. El sistema está diseñado para utilizar el producto escalar de los vectores que representan a los documentos y a la consulta, pues ellos permiten mostrar de forma ordenada los documentos de acuerdo al grado de similitud.

1.6.3 Consideraciones de las aplicaciones que implementan la RI.

Aunque el objetivo del SRI SMART fue ser una herramienta experimental, no presenta interfaz gráfica lo cual es una limitante para el usuario. Los cuatro módulos que posee son de gran importancia para el correcto funcionamiento del sistema pero el módulo de recuperación es el más relevante de todos ya que en este es donde se aplica el cálculo de la similitud por la función coseno, entre el vector consulta y cada uno de los vectores documentos de la colección, para así determinar los documentos relevantes a la consulta del usuario. Los documentos son devueltos en orden decreciente según la similitud con la consulta. Una ventaja que posee este sistema de recuperación de información es que como su código se puede distribuir gratuitamente se puede acceder desde un programa externo a la información de cada uno de los cuatro módulos que implementa.

El SRI Karpanta utiliza el lenguaje natural para la entrada de consultas. Aunque no se hizo con el objetivo de crear un motor de búsqueda operacional se puede utilizar en distintos entornos documentales. A diferencia del sistema SMART este SRI se apoya en dos módulos, uno de indización en el que se determinan los términos índices de la colección y otro módulo relacionado con la búsqueda de los documentos relevantes, en el que el usuario a través de una interfaz especifica el criterio de búsqueda.

Estos sistemas aunque son bastantes parecidos y persiguen el mismo objetivo que es el de realizar una recuperación de información exitosa, se puede concluir que el Karpanta presenta una mayor ventaja respecto al SMART, aunque los dos sirven de guía para implementar un Sistema de Recuperación de Información.

1.7 Entorno Tecnológico.

A continuación se describen cada una de las herramientas y tecnologías que se utilizarán en la aplicación del modelo.

1.7.1 Metodología de desarrollo.

Las metodologías de desarrollo de software se utilizan para guiar y organizar el proceso de desarrollo de un software. Actualmente existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Aunque para la realización de la aplicación del modelo vectorial no se utilizó una metodología de desarrollo de software si se tuvieron en cuenta algunos artefactos para un mejor entendimiento de la propuesta y para apoyar el desarrollo del modelo. Los diagramas realizados fueron el modelo de dominio para evidenciar la situación actual del repositorio y el diagrama

de clases de la solución propuesta. También se definieron los requisitos funcionales que debe cumplir el algoritmo implementado.

1.7.2 Lenguaje de programación PHP.

Se utiliza PHP pues es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas Web dinámicas, embebidas en páginas HTML y ejecutadas en el servidor. PHP 5.0 es un lenguaje de programación muy potente que, junto con HTML, permite crear sitios web dinámicos. PHP se instala en el servidor y funciona con versiones de Apache, Microsoft, Netscape Enterprise Server y otros. También permite la conexión a numerosas bases de datos, incluyendo MySQL, Oracle, ODBC. Puede ser ejecutado en la mayoría de los sistemas operativos: Windows, Mac OS, Linux, Unix.

A continuación se muestran otras características de este lenguaje:

- Es un lenguaje multiplataforma.
- El código fuente escrito en PHP es invisible al navegador web y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Integración con varias bibliotecas externas, permite desde generar documentos que se pueden leer con *Acrobat Reader* hasta analizar código XML.
- Capacidad de expandir su potencial utilizando módulos. Posee una amplia documentación en su sitio web oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.

1.7.3 Servidor web Apache.

Apache 2.2 es un servidor web de código abierto, multiplataforma y muy popular actualmente. Es usado para muchas tareas donde el contenido necesita ser puesto a disposición en una forma segura y confiable. Algunas de sus principales características son:

- Es un servidor web flexible, rápido y eficiente.
- Multiplataforma.
- Tecnología de código fuente abierto.
- Altamente configurable y de diseño modular.

- Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.

Apache es un servidor web potente por su modularidad, su robustez y estabilidad. Es el servidor web que se usará para el desarrollo de la solución que se propone en la presente investigación.

1.7.4 *NetBeans* para PHP.

NetBeans es un Entorno Integrado de Desarrollo (IDE), la versión a utilizar será la 7.2. Es una aplicación de código abierto diseñada para el desarrollo de aplicaciones fácilmente portables entre las distintas plataformas como son: la plataforma Java, así como JavaFX, PHP, *JavaScript* y Ajax, Ruby, y C ++. Es un producto libre y gratuito sin restricciones de uso. *NetBeans* IDE permite el desarrollo de aplicaciones web, el control de versiones, la colaboración entre varias personas, la creación de aplicaciones compatibles con teléfonos móviles y resaltados de sintaxis.

Características de *NetBeans*:

- Instalación y actualización más simple.
- Características visuales para el desarrollo web.
- Conjuntos de herramientas independientes de la plataforma y modulares.
- Flexibilidad entre plataformas.
- Soporte para PHP.

1.7.5 Sistema Gestor de Bases de Datos.

Un Sistema Gestor de Bases de Datos (SGBD) es un conjunto de programas no visibles al usuario final que se encargan de la privacidad, la integridad, la seguridad de los datos y la interacción con el sistema operativo. Proporciona una interfaz entre los datos, los programas que lo manejan y los usuarios finales (18).

MySQL

MySQL es un sistema manejador de bases de datos y multiusuario. Es muy rápido en la lectura lo que lo hace ideal en aplicaciones web con baja concurrencia en la modificación de datos, pero intensivo en la lectura.

Las características principales de MySQL son:

- Multiplataforma.

- Procedimientos almacenados.
- Vistas actualizables.
- Motores de almacenamiento independientes.
- Caché de consultas.
- Indexado y búsqueda de texto completo.
- Biblioteca para bases de datos incrustadas.
- APIs para acceder a bases de datos MySQL en los lenguajes: C, C++, Java, PHP, C#, Pascal, Delphi, Lisp, Perl, Python, Ruby, REAL basic.

MySQL es muy utilizado en aplicaciones web, como Drupal o phpBB, en plataformas (Linux, Windows, Apache, MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. Es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

1.7.6 *VisualParadigm*.

Visual Paradigm para *Unified Modeling Language* (UML) es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

Existen diversas herramientas CASE, una de las más usadas actualmente por las facilidades que ofrece es el *Visual Paradigm 8.0* para UML. A continuación se listan algunas de las principales características de esta herramienta:

- Las imágenes y reportes generados son de buena calidad.
- Varios idiomas.
- Genera la documentación del proyecto automáticamente en varios formatos como HTML y PDF.
- Fácil de instalar y actualizar.
- Permite modelar los procesos de negocio.

1.7.7 *OHS*

Open Harvester System (OHS por sus siglas en inglés) es un sistema de indexación de metadatos de código abierto desarrollado por *Public Knowledge Project* para ampliar y mejorar el acceso a la investigación. Permite crear un índice de búsqueda de los metadatos desde archivos compatibles con *Open Archives Initiative* (OAI), tales como sitios usando *Open Journal Systems* (OJS) u *Open Conference Systems* (OCS). Algunas características son que posee capacidad de recuperar metadatos OAI en una variedad de esquemas, tiene una interfaz de búsqueda flexible que permite la búsqueda simple y búsqueda avanzada mediante campos cruzados desde todos los archivos recuperados.

1.7.8 Matlab.

Es un software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio. Esta herramienta se utiliza para la solución numérica de problemas. Entre sus prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes. En esta investigación se utilizará la versión 7.6 con el objetivo de realizar las pruebas de validación de la aplicación a través de gráficas generadas por esta herramienta.

1.8 Conclusiones parciales.

La elaboración del estado del arte en el presente capítulo permitió obtener un mayor conocimiento sobre el tema en cuestión para posteriormente implementar el modelo vectorial. Después del estudio realizado se concluye que se necesita realizar un buen procesado de la colección de documentos aplicando correctamente el proceso de indexación para obtener mejores resultados en el proceso de búsqueda de la información. El estudio de homólogos realizado permitió comprender mejor cómo interactúan los componentes de un sistema de recuperación de información.

Capítulo 2. Análisis e implementación del sistema.

Introducción.

En este capítulo se explica la propuesta de solución al problema de la investigación, se identifican los requerimientos funcionales que debe cumplir el algoritmo implementado, se plantea el modelo de dominio para evidenciar como se realiza actualmente el proceso de recuperación de información en el repositorio de la biblioteca, además se construye el diagrama de clases de la solución propuesta, también se realiza una descripción del modelo propuesto así como los pseudocódigos del algoritmo a implementar.

2.1 Descripción de la propuesta de solución.

Inicialmente para comprobar los resultados arrojados por el modelo un usuario introduce una consulta formulada en lenguaje natural a través de una interfaz. Esta consulta se representa mediante un vector de *n* elementos donde *n* es el número de términos significativos de la consulta. A cada término en el vector lo acompaña un valor que significa el peso de dicho término en la consulta. Después de realizar el proceso de indexación a la colección en el que se eliminan las palabras vacías y se aplica el algoritmo de *stemming* se representa cada documento de la colección mediante un vector de *n* elementos, donde *n* es el número de términos indizables susceptibles de ocurrir o aparecer en cualquier elemento de la colección. Luego mediante el sistema de cálculo *Tf-IDF*, a cada término o elemento del vector de cada documento se le asigna un valor numérico o peso, que significa la importancia o valor informativo de ese término en ese documento. Luego se calcula la similitud entre la consulta y los documentos según la métrica del coseno, aplicándola entre el vector de la consulta y los vectores de los documentos. El resultado es un valor numérico que indica el grado de ajuste o semejanza entre la consulta y los documentos; de forma que, aquellos documentos que arrojen una cifra más alta, serán los que más se ajusten a la consulta formulada.

2.2 Modelo de dominio.

El modelo de dominio es un artefacto que se realiza durante el análisis de la solución, se construye siguiendo las reglas del Lenguaje Unificado de Modelado. Este modelo no contiene conceptos propios de un sistema de software sino que representa la realidad física de la situación a resolver. El modelo de dominio que se presenta a continuación ayudará a comprender de una mejor manera cómo se realiza actualmente la recuperación de información en el repositorio.

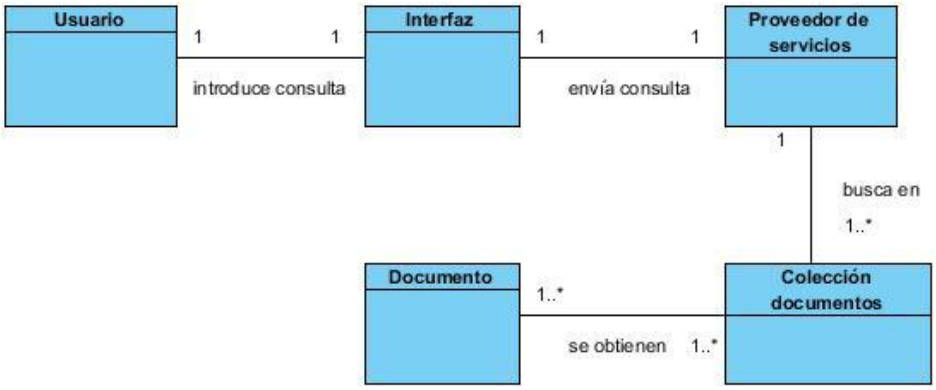

Figura 3. Modelo de dominio.

2.2.1 Descripción de las clases del dominio.

Usuario: representa al usuario que va a recuperar información en la aplicación.

Interfaz: representa la interfaz de la aplicación con la cual va a interactuar el usuario.

Proveedor de servicios: se realiza la recuperación de información utilizando la herramienta DSpace.

Colección documentos: representa la base de datos donde están almacenados los documentos recolectados previamente por el OHS (Open Harvester System).

Documentos relevantes: representa el conjunto de documentos relevantes que satisfacen la consulta del usuario y que le van a ser devueltos.

2.3 Requerimientos.

Los requisitos permiten identificar las facilidades que ha de proporcionar un sistema, describiendo las principales funcionalidades del mismo. A continuación se definen los requisitos funcionales para el modelo implementado.

2.3.1 Requerimientos funcionales de la aplicación.

No	Descripción	Prioridad
RF1	Indexar documentos del repositorio	Alta
RF2	Realizar análisis lexicográfico a la colección	Media
RF3	Eliminar palabras vacías	Media
RF4	Aplicar *stemming* para detectar variantes morfológicas	Alta

	de un mismo término	
RF5	Convertir documentos del repositorio en vectores	Alta
RF6	Crear estructura de datos para almacenar documentos procesados	Alta
RF7	Asignar pesos a los términos que componen los índices de los documentos	Alta
RF8	Aplicar métrica del coseno	Alta

2.4 Diagrama de clases.

Para la implementación del modelo vectorial se ha optado por modelar la solución siguiendo el paradigma de la Programación Orientada a Objetos (POO), con el objetivo de lograr una mejor distribución de las funcionalidades del sistema, a continuación se exponen el conjunto de clases pertenecientes a la solución.

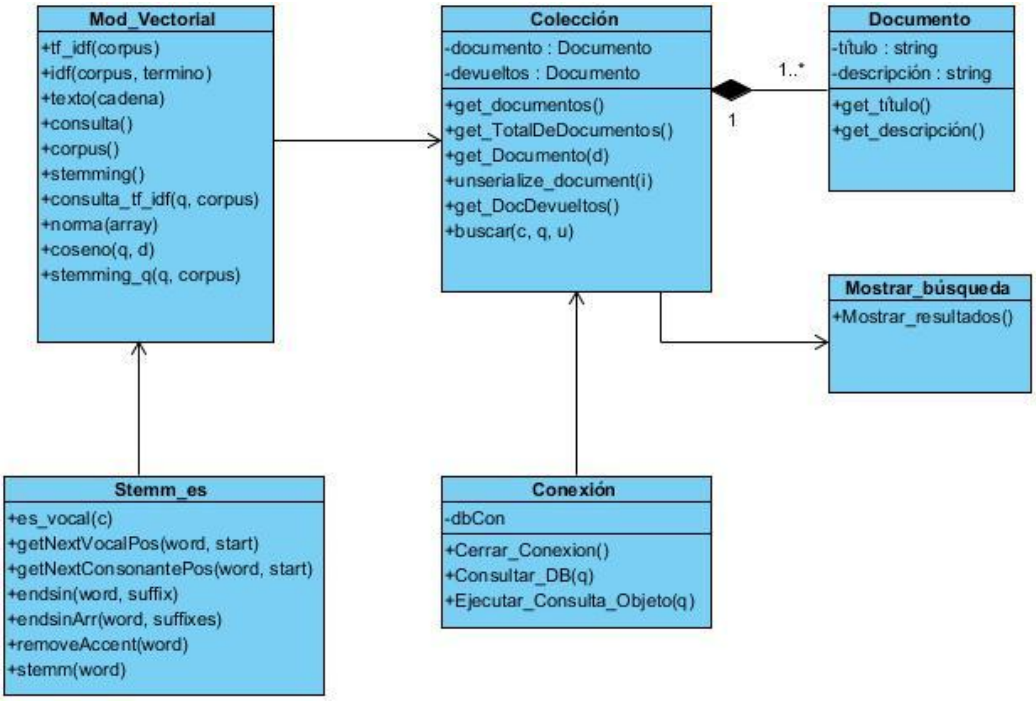

Figura 4. Diagrama de clases.

2.5 Descripción de la arquitectura.

La implementación del modelo se hizo aplicando el patrón arquitectónico Modelo Vista Controlador. Este patrón separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Este se ve frecuentemente en aplicaciones web, donde la vista es la página

HTML y el código que provee de datos dinámicos a la página. El modelo representa a toda la información con la que trabaja la aplicación, gestiona el comportamiento y los datos del dominio, responde a las peticiones de información sobre el estado que vienen de la vista y responde a instrucciones de cambio de estado provenientes del Controlador. La Vista gestiona la presentación de la información de la aplicación, todo lo relativo a la interfaz de usuario, los datos de que dispone para seguir interactuando con la aplicación. El Controlador responde a eventos invocados desde la Vista, llama a la lógica del negocio para procesar y producir una respuesta, interpreta las entradas del usuario, informando al Modelo y/o a la Vista de los cambios que supongan esas entradas. En la implementación del modelo vectorial este patrón arquitectónico se aplica de la siguiente manera: en la vista se tiene la interfaz principal con la que cuenta la aplicación a través de la cual el usuario introduce la consulta y posteriormente en esta misma vista le son devueltos los resultados, en la controladora se gestionan los eventos invocados desde la vista y también hace uso de las clases entidades definidas en la solución, por último a través del modelo donde está contenida la clase conexión se accede a la base de datos en la que se encuentra contenida la información con la que trabaja el sistema.

2.6 Implementación de la solución.

Para aplicar el modelo primeramente se realiza el procesado de la consulta, a la cual se le eliminan las palabras que no aportan significado a la búsqueda, también se reducen los términos a su raíz aplicando el algoritmo de *stemming* y se calculan los pesos de los términos. Posteriormente se realiza el procesado a la colección de documentos para aplicar el modelo, en la que se eliminan las palabras vacías, se reducen las palabras a su raíz morfológica, se calcula el peso de los términos en el documento y la colección aplicando el método de cálculo Tf-IDF. Finalmente, para obtener los documentos relevantes para devolverlos al usuario, se calcula la similitud aplicando la Función del Coseno entre el vector de la consulta y cada uno de los vectores de los documentos.

2.6.1 Procesamiento de la consulta.

A la consulta se le eliminan las palabras vacías, de hacer esto en la implementación se encarga el método **Texto(),** el cual recibe como parámetro la consulta, también a la consulta se le aplica el *stemming* y el cálculo a los pesos de los términos que la componen, todo esto se realiza del mismo modo que a la colección, este procesamiento se explica en el caso de la colección en el epígrafe 2.6.2.

A continuación se muestra el pseudocódigo del procesamiento que se realiza con la consulta para aplicar el modelo vectorial, específicamente el cálculo de los pesos a los términos. Este método acepta como entrada la consulta introducida y la colección de documentos y produce como salida los términos de la consulta con sus respectivos valores una vez efectuado el cálculo.

Cálculo Tf-IDF a la consulta.

INICIO

Entrada: q->consulta

c->colección documentos

Salida: r->términos de la consulta con su valor de Tf-IDF

consulta=**Texto(q**)

r=adicionar la cantidad de valores de la consulta

Para Cada término->valor $\in$ r

r[término]=(v/total de r)***IDF(c,término)**

escribir r

FIN

2.6.2 Procesamiento de la colección.

La colección es procesada mediante la eliminación de las palabras no significativas lo cual también contribuye a reducir el tamaño del *corpus*, además se eliminan caracteres como signos de puntuación, guiones, paréntesis, llaves, corchetes. Luego se aplica el algoritmo de *stemming* para que en el proceso de búsqueda se recuperen mayor cantidad de documentos que satisfagan el criterio de búsqueda.

2.6.2.1 Eliminación de palabras vacías y otros caracteres.

Primeramente se utilizaron como términos no significativos las palabras vacías definidas para implementar el sistema de recuperación de información *SMART.* En el modelo implementado la funcionalidad **Texto()** se encarga de realizar este proceso, en el cual se eliminan las palabras vacías y caracteres como signos de puntuación, llaves, paréntesis, signos de interrogación, exclamación.

2.6.2.2 Aplicación de *stemming* a la colección.

El método implementado utiliza como función auxiliar el método **Corpus()** que se encarga de seleccionar de cada documento recolectado en la base de datos cada término del mismo y la cantidad de

ocurrencias que posee. El método **stemm(palabra)** dentro de la clase stemm_es se encarga de realizarle el *stemming* a las palabras. Como salida se obtienen los vectores de cada documento en el que se almacena la raíz de cada término con sus valores de pesos correspondientes. Para cada documento de la colección se selecciona cada término, el cual es el parámetro de entrada al método **stemm** para determinarle su término base.

INICIO

Salida: resultado->forma base de las palabras con sus pesos.

posición=0

colección=**Corpus()**

Para Cada documento$\in$ colección

Para Cada término->valor $\in$ documento

palabra=**stemm(término)**

resultado[posición][palabra]=valor

aumentar posición en una unidad

escribir resultado

FIN

2.6.2.3 Cálculo de los pesos a los términos de la colección.

A continuación se describe como se realiza el proceso del cálculo de los pesos a los términos de la colección. Esta funcionalidad acepta como entrada la colección de documentos y produce como salida los vectores de cada documento, en dicho vector se almacena el término con su respectivo valor según el cálculo aplicado. Utiliza como método auxiliar el cálculo de la frecuencia inversa del término en la colección a través del método **IDF(colección, término)**, el cual devuelve la frecuencia inversa del término en la colección.

INICIO

Entrada: c->colección de documentos

Salida: corpus->términos con su peso según el cálculo del Tf-IDF

cantidad=asignar cantidad de términos de la colección

Desde inicio hasta cantidad con paso uno

Para Cada término->valor ∈ documento

corpus[i][término]=redondear (valor/total_términos_documento _en _cuestión)

corpus[i][término]=redondear(corpus[i][término]***IDF(c,término)**)

escribir corpus

FIN

2.6.3 Aplicación del modelo.

El algoritmo propuesto para aplicar el modelo vectorial parte, como se explicaba anteriormente, del análisis a la consulta formulada en lenguaje natural introducida por el usuario, a ésta se le realiza la eliminación de palabras vacías, caracteres como signos de puntuación, llaves, paréntesis (de esto se encarga en la implementación el método **Texto(cadena)**) y además se le aplica el proceso de *stemming* para reducir palabras parecidas morfológicamente a su forma base, por último se le aplica a los términos de la consulta la determinación de los pesos según el método Tf-IDF(en la implementación el método **IDF(colección,término)** se encarga de calcular la frecuencia inversa del término en la colección). También a la colección de documentos recolectados por el *OHS* se le realiza el mismo proceso. Por último cuando se ha procesado la consulta y la colección, se calcula la similitud según la métrica del coseno entre la consulta y cada uno de los documentos de la colección.

2.6.4 Cálculo de la similitud por la Función Coseno.

En este epígrafe se explica el cálculo de la similitud entre la consulta y cada uno de los documentos a través de la función coseno utilizando la fórmula [1]. Esta operación utiliza como método auxiliar la función **Norma(array),** la cual normaliza al vector consulta y el vector documento. Tiene como entrada la consulta y cada uno de los vectores de los documentos y devuelve como resultado un valor numérico que significa la similitud entre la consulta y el vector documento.

INICIO

Entrada: q-> consulta

d->documento

Salida: c->valor del coseno entre el vector consulta y el vector documento

c=0

Para Cada término->valor $\in$ q

Si término $\in$ d

c=c+(valor*d[término])

Fin Si

c=c/(norma(q)*norma(d))

escribir c

FIN

2.7 Conclusiones parciales.

En este capítulo se ha expuesto todo el análisis desarrollado para realizar la implementación del modelo propuesto, por lo que se concluye que la descripción de la propuesta de solución facilitó una mejor comprensión del funcionamiento del modelo de espacio vectorial. La elaboración del modelo de dominio permitió conocer en qué estado está la distribución física de la recuperación de información actualmente en el repositorio de la biblioteca. Con el levantamiento de requisitos funcionales se identificaron las funcionalidades del algoritmo a implementar y su prioridad a la hora de realizar la implementación. Con la implementación del modelo se pusieron en práctica las fórmulas y conceptos teóricos tratados en el primer capítulo.

Capítulo 3. Validación y prueba.

Este capítulo tiene como propósito comprobar si se han cumplido los objetivos trazados en la implementación del modelo vectorial realizando un grupo de pruebas y validaciones a la aplicación para luego compararlas con la recuperación de información existente actualmente en el repositorio de la biblioteca de la Universidad de las Ciencias Informáticas. Estas pruebas permiten medir el funcionamiento del sistema y dan una valoración para posteriormente llegar a conclusiones.

3.1 Medidas de evaluación.

En este sistema la evaluación de la recuperación de información está orientada primeramente a determinar cuán preciso es el conjunto solución de la respuesta a una consulta determinada (expresada en lenguaje natural) a partir de la relevancia de cada documento respecto a la consulta y en segundo lugar también está orientado a medir la efectividad del sistema, esta medida cuantifica la efectividad de recuperar documentos relevantes mientras no se recuperan documentos no relevantes.

La exhaustividad se define como la proporción de los documentos relevantes que han sido recuperados y permite evaluar la habilidad del sistema para encontrar todos los documentos relevantes de la colección. De donde se deriva la fórmula:

$$E = \frac{w}{w + x} \qquad [6]$$

Donde *E* equivale a la exhaustividad, *w* son todos los documentos relevantes recuperados y por último *x* se refiere a los documentos relevantes no recuperados.

La precisión es la proporción de los documentos recuperados que son relevantes y se define como:

$$P = \frac{w}{w + y} \qquad [7]$$

Donde *P* equivale a la precisión, *w* son todos los documentos relevantes recuperados y por último *y* representa a los documentos no relevantes recuperados.

Estas dos medidas están muy estrechamente relacionadas, ya que una alta exhaustividad se acompaña de una baja precisión y viceversa.

3.2 Colección de prueba.

Los datos con los cuales trabaja el sistema provienen del repositorio institucional de la Universidad de

las Ciencias Informáticas, son un total de 500 documentos experimentales previamente recolectados por el Open Harvester System que tratan sobre investigaciones académicas y científicas. Estos documentos están descritos teniendo en cuenta el estándar *Dublin-Core*, entre los metadatos que posee y se utilizan para la recuperación de información se encuentran el título, la descripción del documento y las palabras clave, identificados por las etiquetas <dc:title>, <dc:description>, <dc:subject>, como sugiere el estándar empleado.

3.3 Metodología para las pruebas.

Para la realización de los experimentos primeramente se realizaron 10 consultas y previamente se analizaron los resultados óptimos esperados. Estos resultados se comparan con los que arroja el sistema para varios valores del umbral de corte entre 0.1 y 0.07. Luego los resultados obtenidos se expresan en las gráficas de precisión y exhaustividad.

3.4 Experimentos y resultados.

	0.07	Recuperados	Relevantes	RR	NRR	precisión	exhaustividad
Técnicas de inteligencia artificial		13	15	11	2	0,84615385	0,733333333
software educativo		15	19	15	4	1	0,789473684
aplicación de las TIC en el sistema de salud de cuba		11	19	9	2	0,81818182	0,473684211
ingeniería de requisitos en proyectos de software		32	33	27	5	0,84375	0,818181818
juegos didácticos		4	5	3	1	0,75	0,6
Desarrollo de software multimedia		27	28	22	5	0,81481481	0,785714286
Sistemas de gestión de contenidos		7	7	5	2	0,71428571	0,714285714
arquitectura empresarial		24	25	12	3	0,5	0,48
planificación y gestión de riesgos		22	25	16	6	0,72727273	0,64
propuesta de metodología de desarrollo de software		15	16	5	3	0,33333333	0,3125
Promedio						0,73477923	0,634717305

Tabla 1. Consultas realizadas para un umbral del corte 0.07.

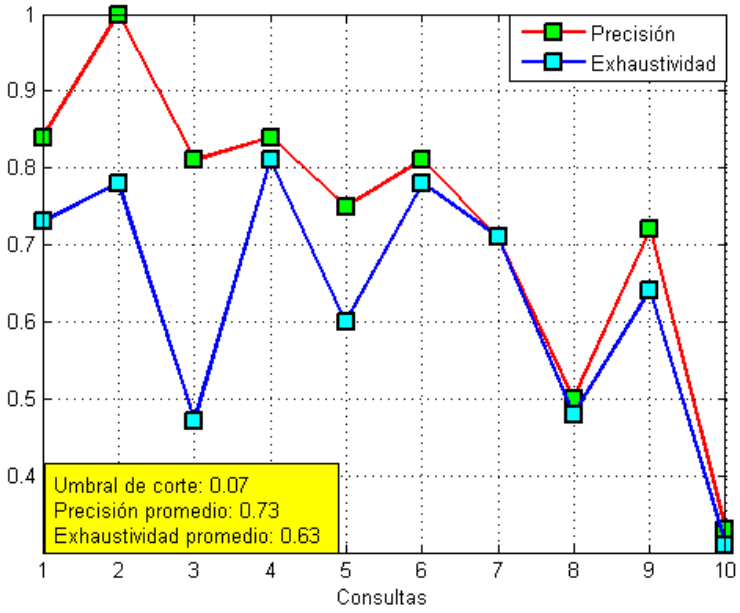

Figura 5. Comportamiento precisión-exhaustividad para umbral del corte 0.07.

Se ven valores similares para ambas variables, no se ve estabilidad en el sistema.

	0.08	Recuperados	Relevantes	RR	NRR	precisión	exhaustividad
Técnicas de inteligencia artificial		13	15	9	4	0,69230769	0,6
software educativo		12	19	11	1	0,91666667	0,578947368
aplicación de las TIC en el sistema de salud de cuba		23	19	16	7	0,69565217	0,842105263
ingeniería de requisitos en proyectos de software		32	33	23	9	0,71875	0,696969697
juegos didácticos		4	5	3	1	0,75	0,6
Desarrollo de software multimedia		17	28	12	5	0,70588235	0,428571429
Sistemas de gestión de contenidos		5	7	4	1	0,8	0,571428571
arquitectura empresarial		20	25	15	5	0,75	0,6
planificación y gestión de riesgos		21	25	15	6	0,71428571	0,6
propuesta de metodología de desarrollo de software		15	16	12	3	0,8	0,75
Promedio						0,75435446	0,626802233

Tabla 2. Consultas realizadas para un umbral del corte 0.08.

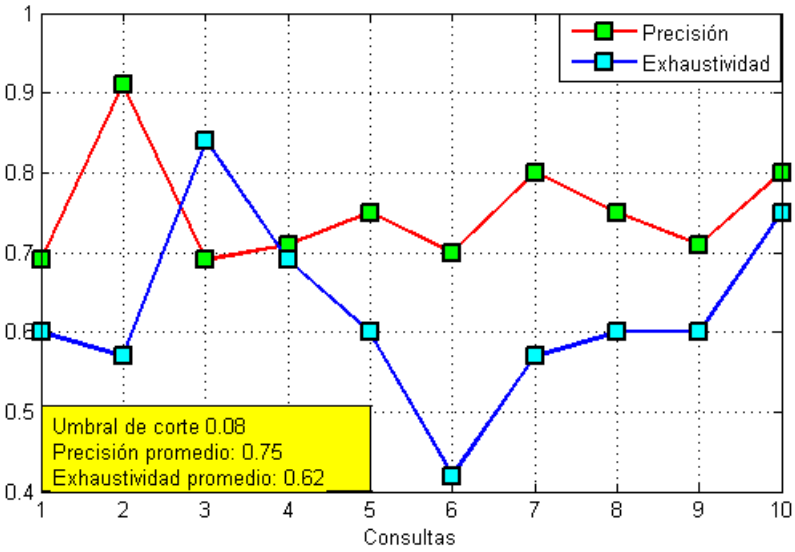

Figura 6. Comportamiento precisión-exhaustividad para umbral del corte 0.08.

Se van separando los valores de precisión y exhaustividad aunque persisten consultas donde continúa la exhaustividad superando la precisión.

	0.09	Recuperados	Relevantes	RR	NRR	precisión	exhaustividad
Técnicas de inteligencia artificial		10	15	8	2	0,8	0,533333333
software educativo		12	19	11	1	0,91666667	0,578947368
aplicación de las TIC en el sistema de salud de cuba		16	19	12	4	0,75	0,631578947
ingeniería de requisitos en proyectos de software		23	33	18	5	0,7826087	0,545454545
juegos didácticos		3	5	2	1	0,66666667	0,4
Desarrollo de software multimedia		13	28	9	4	0,69230769	0,321428571
Sistemas de gestión de contenidos		4	7	3	1	0,75	0,428571429
arquitectura empresarial		16	25	12	4	0,75	0,48
planificación y gestión de riesgos		19	25	16	3	0,84210526	0,64
propuesta de metodología de desarrollo de software		10	16	7	3	0,7	0,4375
Promedio						0,7650355	0,499681419

Tabla 3. Consultas realizadas para un umbral del corte 0.09.

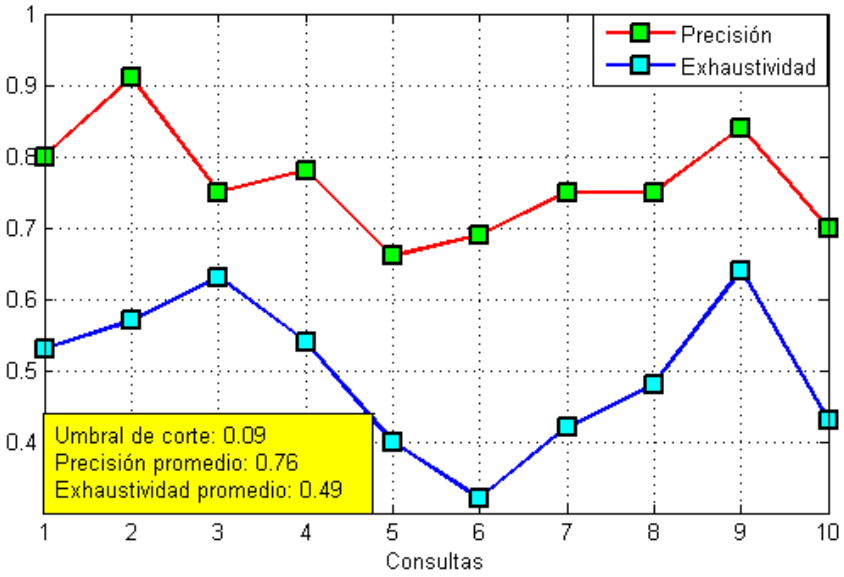

Figura 7. Comportamiento precisión-exhaustividad para umbral del corte 0.09.

Se comienza a apreciar la estabilidad del sistema con altos valores de precisión y disminuyendo la exhaustividad.

	0.1	Recuperados	Relevantes	RR	NRR	precisión	exhaustividad
Técnicas de inteligencia artificial		8	15	6	2	0,75	0,4
software educativo		12	19	11	1	0,91666667	0,578947368
aplicación de las TIC en el sistema de salud de cuba		11	19	9	2	0,81818182	0,473684211
ingeniería de requisitos en proyectos de software		20	33	16	4	0,8	0,484848485
juegos didácticos		2	5	2	0	1	0,4
Desarrollo de software multimedia		10	28	7	3	0,7	0,25
Sistemas de gestión de contenidos		3	7	3	0	1	0,428571429
arquitectura empresarial		15	25	12	3	0,8	0,48
planificación y gestión de riesgos		15	25	13	2	0,86666667	0,52
propuesta de metodología de desarrollo de software		8	16	5	3	0,625	0,3125
Promedio						0,82765152	0,432855149

Tabla 4. Consultas realizadas para un umbral del corte 0.1.

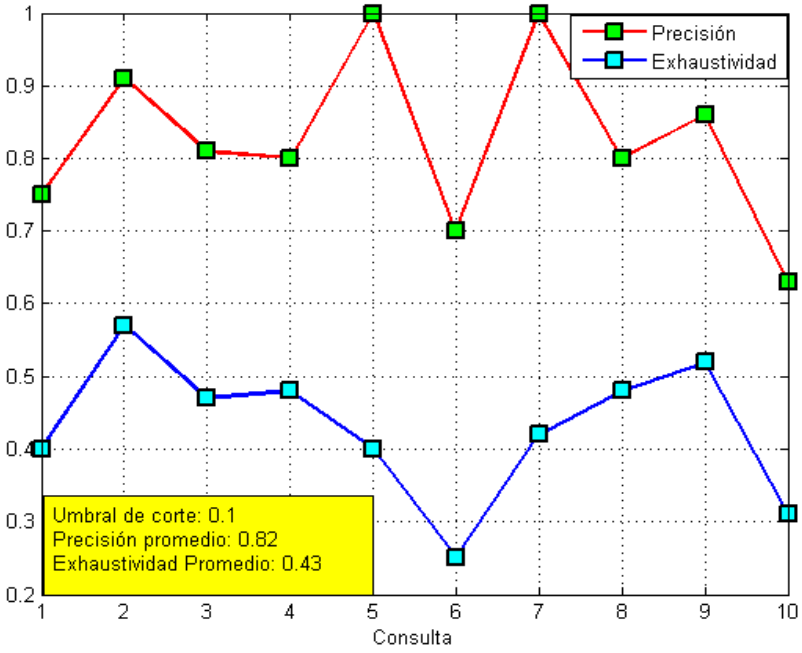

Figura 8. Comportamiento precisión-exhaustividad para umbral del corte 0.1.

Muy estable el sistema. Algunos de los valores de exhaustividad son pequeños, por esto se decide parar y tomar 0.1 como umbral de corte.

Los resultados anteriores muestran la eficiencia del modelo en términos de precisión. Esto sugiere que para el usuario final los resultados de la búsqueda son muy fiables.

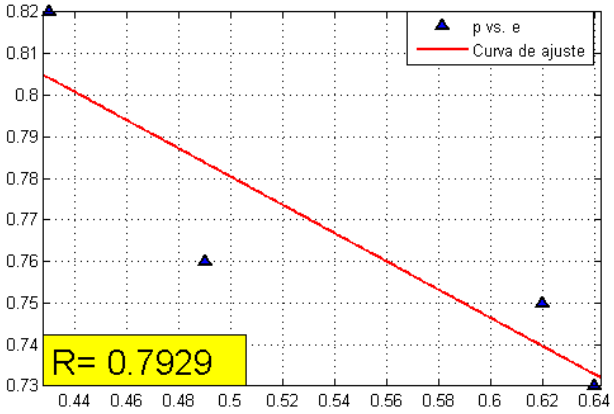

Figura 9. Curva de ajuste.

Esta gráfica representa el comportamiento del sistema teniendo en cuenta las variables precisión y exhaustividad. La variable R=0.79 representa la desviación cuadrática del sistema, es un valor bastante

pequeño y muestra un compromiso bastante óptimo entre ambas variables.

De aquí se deriva de forma analítica la relación entre ambas variables, independientemente del umbral del corte, mediante la fórmula:

$$p = -0.3398 * e + 0.9502$$

Los resultados anteriores son una muestra de que el modelo vectorial produce buenos resultados para la precisión sobre una colección de documentos cuyo contenido se representa por un vector de dimensiones pequeñas, ya que estará formado por los metadatos título, resumen y palabras clave. Los valores de precisión alcanzados para los diferentes umbrales de corte están dentro del rango reportado en la literatura (19) (20) (21).

No basta con conocer el buen funcionamiento del modelo, se necesita comparar con los resultados que provee el repositorio para las mismas consultas. Partiendo de que la colección del repositorio es mayor que 500 documentos, se debe primeramente normalizar este valor para que la comparación de los resultados sea sobre la misma base. Se instaló un DSpace y se le agregó la misma muestra sobre la que se realizaron los experimentos. De esta forma el motor de búsqueda se corresponde con el del repositorio institucional y la muestra es la misma sobre la que se realizaron los experimentos anteriores.

Se ejecutan las mismas consultas y se mide la precisión y la exhaustividad, estos resultados se comparan con los obtenidos previamente para umbral de corte 0.1. Los resultados se muestran en las figuras 10 y 11. Se obtiene una media de precisión mayor y una media de exhaustividad menor cuando se aplica el modelo vectorial. Este es un resultado esperado y correcto, partiendo de las fórmulas [7] y [6] es evidente que el aumento de una va en detrimento de la otra.

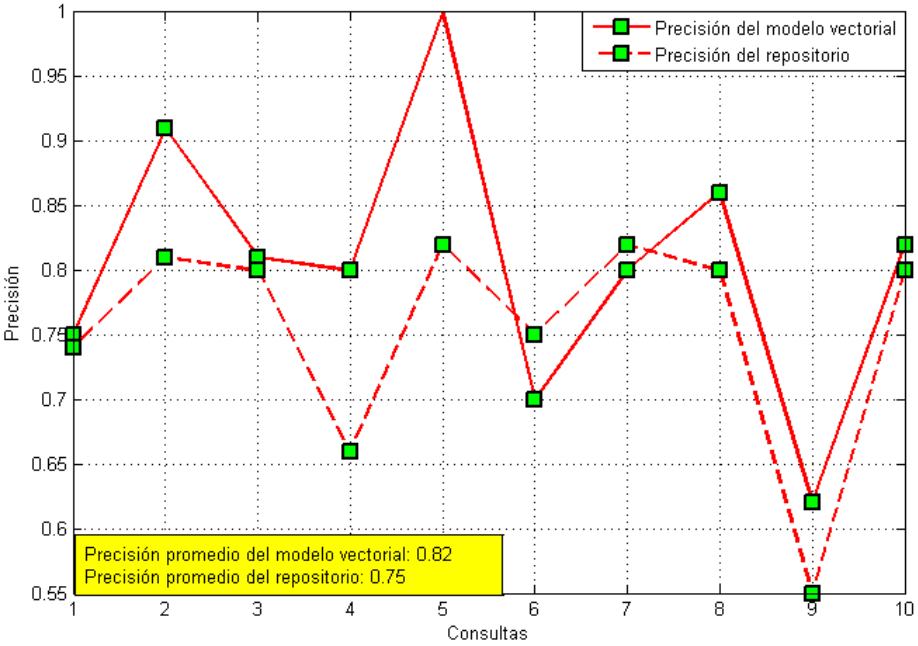

Figura 10. Comparación en términos de precisión entre el modelo vectorial y el repositorio institucional.

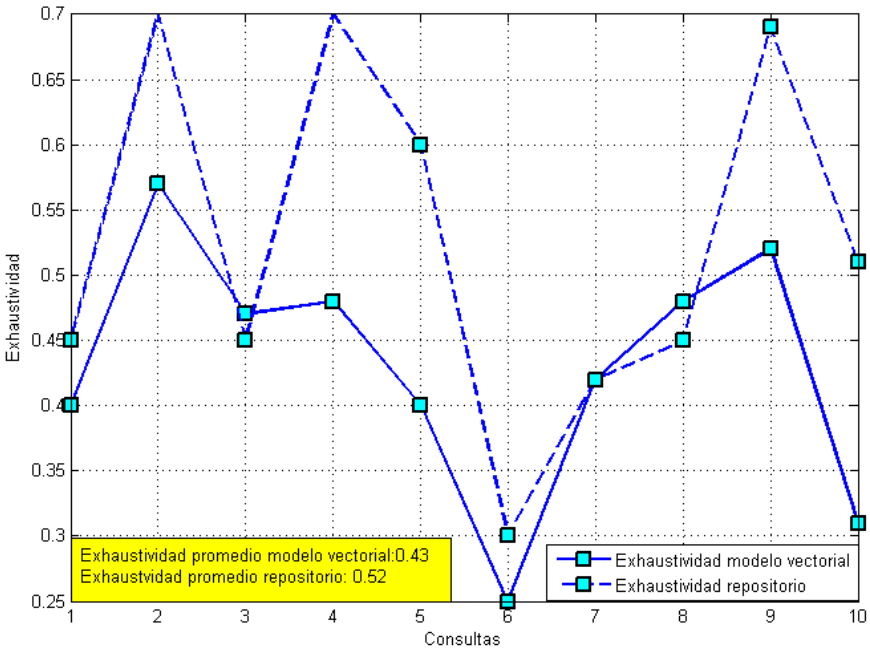

Figura 11. Comparación en términos de exhaustividad entre el modelo vectorial y el repositorio institucional.

3.5 Conclusiones parciales.

Al concluir las pruebas correspondientes a la solución se pudo verificar que el sistema cumple con las

funcionalidades previstas. Con el uso de las pruebas de precisión y exhaustividad en la aplicación se demostró el correcto desempeño de las funcionalidades implementadas como parte de la propuesta de solución, obteniéndose así un sistema capaz de mejorar la precisión en la recuperación de información en un proveedor de servicios OAI-PMH. Todo esto evidencia la fiabilidad y eficiencia del modelo implementado.

Conclusiones generales.

A partir del análisis de los procesos realizados para la recuperación de información en el sistema de información de la UCI, se demostró la necesidad e importancia de Implementar el Modelo Vectorial para mejorar la precisión en la recuperación de información en un proveedor de servicios OAI-PMH. Esta implementación permite realizar la gestión de información de los eventos que se realizan como parte de la renovación del sistema de información de la universidad y del repositorio institucional, con el objetivo de mejorar el proceso de búsqueda de información en el repositorio. De acuerdo con el objetivo general trazado en la investigación, se obtuvo la implementación del modelo propuesto, adaptándose a las necesidades actuales para la recuperación de información en el sistema de información de la UCI.

Con la realización de la presente investigación se dio cumplimiento a los objetivos propuestos, analizándose conceptos relacionados con el Modelo vectorial para mejorar la precisión en la recuperación de información en el repositorio de la universidad. Los métodos de validación definidos permitieron arribar a la conclusión de que se mejoró en la precisión de la recuperación de información. La solución implementada y probada permite mejorar la búsqueda de información en el repositorio institucional de la Universidad de las Ciencias Informáticas.

Recomendaciones.

Se recomienda fusionar el modelo implementado con otras técnicas que existen para mejorar la precisión en la Recuperación de Información tales como incorporarle un proceso de ranking a los documentos para darle mayor relevancia al devolvérselos al usuario y agrupar los documentos según el tipo de contenido, todo esto con el objetivo de que se realicen búsquedas exitosas.

Bibliografía referenciada.

1. **Grossman, D y Frieder, O.** *"Information Retrieval. Algorithms and Heuristics".* 1998.

2. **Sparck, Jones K y Willet, P. "***Readings in information retrieval".* San Francisco :s.n., 1997.

3. **Baeza-Yates, R y Ribeiro-Neto, B.** *"Modern InformationRetrieval".* 1999.

4. **H Tolosa, Gabriel y Fernando, Bordignon. "***Introducción a la Recuperación de Información. Conceptos, modelos y algoritmos básicos".* Argentina : s.n., 2007.

5. **Navarro, G y Baeza-Yates, R.** *"A language for queries on structure and contens of textual databases".* New York :s.n., 1995.

6. **Burkowski, F.** *"Retrieval activities in a databases consisting heterogeneous collections of structure texts".* New York :s.n., 1992.

7. **Robertson, S.E y Spark Jones, K.** *"Relevance Weighting of Search terms".* 1972.

8. **Salton, G.** *"The Smart Retrieval System - Experiments in Automatic Document Processing".* New Jersey :s.n., 1971.

9. **Porter, M.F.** *"An Algorithm for suffix stripping".* 1980.

10. **Luhn, H.P.** *"The automatic creation of literature abstracts".* 1952.

11. **Gómez Dueñas, L.** *"La iniciativa de archivos abiertos (OAI), un nuevo paradigma en la comunicación científica y el intercambio de información".* No.25, 2005, Vol. vol.4.

12. **Lagoze, C.** *"OAI-PMH Implementations Guidelines-Guidelines for Harvester Implementers". Open Archives.* [En línea] [Citado el: 26 de 3 de 2013.] http://www.openarchives.org/OAI/2.0/guidelines-harvester.htm.

13. **Buckley, C.** *"Implementation of smart information retrieval system".* Universidad de Cornell : s.n., 1985.

14. **Zazo Rodríguez, Ángel F, y otros.** *"El sistema de recuperación Karpanta: estudio de usuarios a través del archivo de registro".* Universidad de Salamanca (España) : s.n., 2004.

15. **Trigueros, J y Higuera, R.** *"Bases de datos relacionales versus bases de datos documentales".*

Boletín de la Asociación Andaluza de Bibliotecarios (España) : s.n., 1997.

16. **Fox, C.** *"Lexical Analysis and stoplist".* New Jersey :s.n., 1992.

17. **Hull, D.A.** *"Stemming Algorithms: A case study for detailed evaluation".*1996.

18. **Álvarez, Sara.** Desarrollo web. [En línea] 2007. [Citado el: 28 de Marzo de 2013.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.

19. **Zazo Rodríguez, Ángel F y Gómez Díaz, Raquel**. *"Recuperación de información utilizando el modelo vectorial".* Universidad de Salamanca(España). Ponencia en evento: CLEF. 2001.

20. **Maña López, Manuel J y Ureña López, L. Alfonso**. *"Tareas de análisis del contenido textual para la recuperación de información con realimentación".* Revista científica. 2012.

21. **MONSALVE García, Luz Stella.** "*Experimento de recuperación de información usando las Medidas de similitud coseno, jaccard y dice".* Revista Científica TECCIENCIA, 2013, vol. 6, no 12, p. 1-11.

22. **Universidad Carlos III de Madrid**, "*Metadatos. Recuperación y Acceso a la Información".* 2010.

Bibliografía consultada.

1. **Baeza−Yates, Ricardo, y Ribeiro−Neto, Berthier**. *Modern information retrieval*. Harlow [England], etc: Addison−Wesley, 1999.

2. ***Belkin, N.J.; Croft, W.B.*** *Retrieval techniques. Annual Review of Information Science and Technology, 22, p. 109−145. (1987)*

3. **Buckley, C.; Allan, J.; Salton, G.** *Automatic routing and ad−hoc retrieval using SMART: TREC 2*. // Donna Harman, editor, Proceedings of the Second Text Retrieval Conference TREC−2. NIST Special Publication , 500−215. (1994).

4. **Figuerola, C.G.; Gómez Díaz, R.; López de San Román, E.** *Stemming and n−grams in Spanish: an evaluation of their impact on information retrieval.* En: Journal of Information Science, 26 (6) 2000, pp. 461−467.

5. **Figuerola, C.G.; Alonso Berrocal, J.L.; Zazo Rodríguez, A.F.; Gómez Díaz, R.** *A simple approach to the Spanish−English Bilingual retrieval task. Cross Language Information Retrival and Evaluation*, pp. 224−229. Springer−Verlag: Berlin, N.Y., [etc.], 2001.

6. **Figuerola, C.G.; Alonso Berrocal, J.L.; Zazo Rodríguez, A.F.** *Diseño de un motor de recuperación de información para uso experimental y educativo.* V. 11, pp. 201−209, 2001.

7. **Christopher, Fox.** *Lexical Analysis and Stoplists. Information retrieval, data structures and algorithms.* New Jersey, London, etc.: Prentice−Hall, 1992.

8.**Gómez Díaz, Raquel.** *Estudio de la incidencia del conocimiento lingüístico en los sistemas de recuperación de información para el español.* Tesis doctoral. Salamanca: Ediciones Universidad de Salamanca, 2001.

9.**Harter, S.P.; Hert, C.A.** *Evaluation of information retrieval systems*. // Annual Review of Information Science and Technology, 32, p. 3−94. (1977).

10. **Salton, G.** *Automatic Information Organization and Retrieval*. McGraw−Hill, N.Y. (1968).

11. ***Salton, G.; McGill, M.J.*** *Introduction to Modern Information Retrieval. McGraw−Hill, New York. (1983)*

12. **Dr. Sánchez, Alfredo; Razo, Antonio.** *Estándares de metadatos y directrices de interoperabilidad*. Red Mexicana de Repositorios Institucionales. Diciembre 2012.