Universidad de las Ciencias Informáticas

Facultad 1

Componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autor:

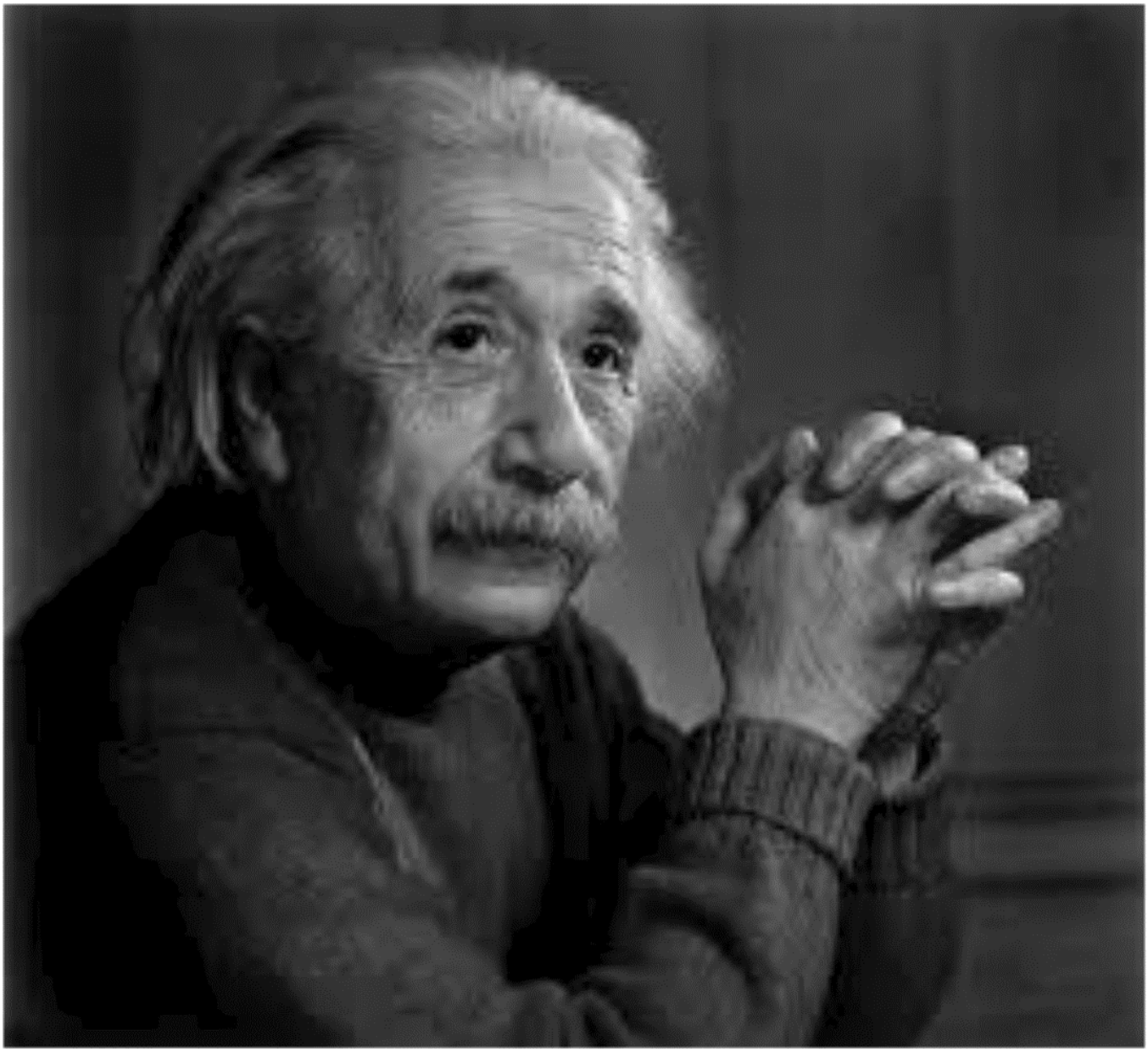
María de los Angeles Díaz Meriño

Tutores:

Ing. Yanedi Abreu Bartomeo

Ing. Walter Daniel Camejo López

Ing. Yuneisy Barrios Pérez



"Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: la voluntad"

Albert Einstein

DECLARACIÓN DE AUTORÍA

Declaro por este medio que yo María de los Angeles Díaz Meriño, con carnet de identidad 93121408239 soy la autora principal del trabajo de diploma titulado “Componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste se firma la presente declaración jurada de autoría en La Habana a los ____ días del mes de _____del año 2016.

_____
Autor
Maria A. Díaz Meriño

_____
Tutora
Ing. Walter D. Camejo López

_____
Tutora
Ing. Yanedi Abreu Bartomeo

_____
Tutor
Ing. Yuneisy Barrios Pérez

Agradecimientos

A Dios por permitirme vivir este momento de mi vida, y que mi madre sea testigo.

A mi madre por ser mi pilar, mi contención, mi guía, por alentarme, por comprenderme, por preocuparse, por demostrarme que si algo se quiere en la vida hay que luchar con todas las fuerzas, por prácticamente obligarme a optar por una carrera universitaria, gracias por ayudarme a convertirme en ingeniera informática y a ser la mujer que soy hoy, todos mis triunfos en la vida van dedicados a ti, Te Amo.

A mi padrastro por el aprecio, por la paciencia, el cariño, la preocupación, por tratarme como una hija, por las sonrisas, por la confianza, gracias Tata eres un padre para mí, te quiero mucho.

A mi padre por su orgullo, preocupación y su entera confianza.

A Jaqueline por siempre tenerme en cuenta, aconsejarme, comprenderme, por bendecirme cada vez que tiene oportunidad y brindarme su cariño, gracias mi madrina te quiero muchísimo.

A ti Adianez amiga mía por animarme y ayudarme a obtener una carrera universitaria, por quererme a pesar de todo, gracias, nunca se me olvida que contigo le empecé a tomar aprecio a las matemáticas.

A mi amiga Flor por ser la hermana pequeña que siempre quise tener, por los consejos, por las noches de chisme, por los regaños, por el ánimo, y por tratarme como familia, gracias titinga.

A mi amiga Zuinara la más venática del mundo, pero con un súper corazón, gracias por obsequiarme tu amistad y por preocuparte tanto por mí, te diría que eres una… pero no se puede.

A mis amigos Dayi y Pino por todos los buenos momentos que compartí con ustedes, por los días de fiestas, por el cariño, por escuchar todas mis historias

cada vez que pasaba un fin de semana, por aconsejarme, gracias Dayi por las noches teseo y tu Pino por hacer más amenas las tardes de PP.

A mis compañeros de aula y a todas las otras personas con las cuales compartí de una forma u otra en la universidad, gracias por todos los momentos que pasamos juntos y los bonitos recuerdos que tengo de ustedes.

A Reinier porque a pesar de que actualmente no exista relación entre nosotros, fue una persona que estuvo presente en todo el transcurso de mi carrera, gracias por el tiempo, la dedicación, y el ánimo.

A mis tutores por su ayuda, en especial a Walter por el apoyo y la confianza.

A todos les agradezco por contribuir a que la culminación de mis estudios universitarios fuera posible.

Dedicatoria

Dedicado al gran amor de mi vida, a mi guía, mi guerrera, a la luz de mis ojos, a ti madre, con toda la gratitud y el amor del mundo.

Resumen

En la actualidad Internet es uno de los principales medios de información que existe y cuenta con un gran volumen de contenido, y diariamente la cantidad de información crece exponencialmente. Cuba no está exenta de esta regla, por ello como alternativa cubana surge el buscador Orión con el objetivo de indexar la web nacional. Debido a que este sistema no satisface las necesidades de información de los usuarios al no determinar con certeza cuan relevante es un documento, es necesario mejorar los resultados de las búsquedas, para solucionar esta dificultad se propone el desarrollo de un componente para el cálculo de la relevancia de la información con anotación semántica con el objetivo de mejorar el proceso de recuperación de información del buscador. Para la implementación de la propuesta de solución, guiada por la metodología AUP-UCI, se utiliza Java como lenguaje de programación, Visual Paradigm como herramienta para el modelado, Eclipse como entorno de desarrollo integrado y como servidor web Apache Tomcat. El componente implementado posee un conjunto de características y funcionalidades que contribuyen a asignar un score de relevancia permitiendo obtener resultados más exactos.

Palabras clave: relevancia, anotación semántica, score, buscador.

ÍNDICE

Índice de Tablas

Índice de Figuras

Introducción

A partir de los avances científicos producidos en los ámbitos de la informática y las telecomunicaciones se desarrollan las Tecnologías de la Información y las Comunicaciones (TIC), permitiendo ampliar las posibilidades del progreso de la sociedad y la adquisición de conocimiento útil. El impacto causado por las TIC se extiende al conjunto de las sociedades del planeta, produciendo una innovación y cambio constante en todos los ámbitos sociales (Belloch, 2012).

Como parte de estas tecnologías se encuentra internet, siendo su elemento más representativo (Belloch, 2012); suponiendo una revolución sin precedentes en el mundo de la informática y de las comunicaciones. Una de las actividades en la que dedican tiempo los usuarios en internet es la búsqueda de información, pero la red es inmensa, conteniendo millones de páginas web, y diariamente la cantidad de información en el mundo crece exponencialmente, encontrándose dispersa y poco estructurada lo que hace muy engorroso el proceso de encontrar información útil (Rodríguez, 2007).

Para agilizar el proceso de búsqueda y darle solución a este tipo de dificultades son usados los Sistemas de Recuperación de Información (SRI), tales como los motores de búsqueda, los cuales se han convertido en herramientas indispensables para los usuarios, procesando la información de forma rápida y automática, permitiendo localizar cualquier contenido existente en la web, tales como textos, imágenes, videos, archivos de sonido, entre otros.

La web cubana se encuentra también en constante crecimiento; Cuba no está exenta de esta regla, por ello la inclusión de los sitios cubanos en los sistemas de búsquedas de Internet como Yahoo! y Google, desempeña un papel fundamental en la divulgación de la realidad del pueblo cubano en el mundo. Sin embargo, por restricciones impuestas por el embargo de los Estados Unidos, no se puede acceder a muchos de los servicios que estos SRI brindan.

Como alternativa cubana para la búsqueda en la web surge el buscador Orión, desarrollado en el marco de trabajo de la Universidad de las Ciencias Informáticas (UCI), la cual surge como parte de los programas de la Batalla de Ideas y constituye un frente de vanguardia en el desarrollo e investigación de nuevas tecnologías. Dentro de la infraestructura productiva que la conforma, el Centro de Ideoinformática (CIDI) es el encargado de la creación de este buscador, que permite la recuperación de información de los documentos publicados en la web nacional.

Actualmente el buscador Orión no satisface adecuadamente las necesidades de información de los usuarios, al no determinar con certeza cuan relevante es un documento, ya que existe el problema de compatibilizar la expresión de la necesidad de información y el lenguaje de los documentos.

Para poder llevar a cabo la tarea de determinar la relevancia de un documento el buscador Orión utiliza el modelo base de RI vectorial. Este es un modelo estadístico-matemático que no tiene en cuenta la estructura sintáctico-semántica del lenguaje natural. Por lo que para determinar cuán relevante es un documento (entiéndase por documento a cualquier recurso publicado en la Web) respecto al criterio de búsqueda introducido por el usuario, solo se tienen en cuenta hasta el momento, el peso del documento y su popularidad (Tolosa, 2008).

El peso del documento viene dado principalmente por la frecuencia de aparición de los términos en el documento. Actualmente los campos del documento que se utilizan para este cálculo son: título, contenido, URL y algunas palabras claves definidas. Sin embargo para el mismo no se tiene en cuenta el significado o el contexto en que se utilizan estos términos.

La popularidad del documento viene dado principalmente por la cantidad de sitios o enlaces entrantes que lo referencian, así como por la reputación y prestigio de estos sitios.

Acarreando como principales consecuencias que el acceso a la información no se realice de la mejor manera, los resultados más relevantes al usuario no siempre son los que se muestran como primeros. Por tanto el usuario pierde mucho tiempo en encontrar lo que desea, revisando los resultados mostrados, y en otras ocasiones no lo encuentra, ya sea porque desista de la búsqueda o porque el SRI no los considere relevantes y no los muestre.

Sobre la base de los elementos expuestos anteriormente se formula el siguiente **problema de investigación**: ¿Cómo contribuir al proceso de recuperación de información del buscador Orión, para obtener mejores resultados en las búsquedas?

Para la realización de la investigación se define como **objeto de estudio** el proceso de cálculo de la relevancia de la información con anotación semántica. Enmarcándose en el proceso de cálculo de la relevancia de la información del buscador Orión como **campo de acción**.

Para dar solución al problema planteado, se propone el siguiente **objetivo general**: Desarrollar un componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión basado en un modelo semántico para la obtención de mejores resultados en las búsquedas.

Del objetivo general enunciado se desglosan los siguientes **objetivos específicos**:

- ✓ Sistematizar los referentes teóricos relacionados con las tecnologías y funcionalidades de los SRI con anotaciones semánticas.
- ✓ Proponer el componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión.
- ✓ Implementar un componente que permita mejorar el cálculo de la relevancia de la información en el buscador Orión.
- ✓ Validar que el componente permita mejorar el cálculo de la relevancia de la información en el buscador Orión.

Teniendo en cuenta lo antes expuesto se plantea la siguiente **hipótesis:** El desarrollo de un componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión basado en un modelo semántico contribuirá a la obtención de mejores resultados en las búsquedas.

Se define como **variable independiente**: componente para el cálculo de la relevancia de la información con anotación semántica. El mismo consiste en una aplicación informática para otorgar una puntuación (o score) de relevancia, la cual se mide como la probabilidad de que un documento dado contenga las relaciones que existen en la mente del usuario en el momento de definición de la consulta. Como **variable dependiente** contribuir a la obtención de mejores resultados en las búsquedas.

Las **tareas de la investigación** trazadas son:

- ✓ Estudio bibliográfico de los principales elementos teóricos y conceptos que permitan analizar el estado actual del desarrollo de tecnologías y herramientas para el cálculo de la relevancia de la información con anotación semántica.
- ✓ Análisis de diferentes motores de búsquedas para la determinación de otros métodos de cálculo de la relevancia de la información con anotación semántica.

- ✓ Determinación de las herramientas y la metodología adecuada para el desarrollo de un componente para el cálculo de la relevancia de la información.
- ✓ Identificación de los requisitos funcionales y no funcionales.
- ✓ Implementación del componente para el cálculo de la relevancia de la información con anotación semántica.
- ✓ Validación de las funcionalidades del componente mediante las pruebas y la verificación del cumplimiento de las necesidades del usuario.

Durante el proceso de desarrollo del componente para el cálculo de la relevancia se realizarán investigaciones para profundizar en el objeto de estudio definido, dichas investigaciones serán guiadas por los siguientes **Métodos Científicos:**

Métodos Teóricos:

- ✓ Analítico Sintético: Utilizado en el estudio de las fuentes bibliográficas existentes referentes al proceso de cálculo de la relevancia de la información y la extracción de los elementos más importantes relacionados a este proceso.
- ✓ Histórico Lógico: Aplicado en el estudio de la evolución y desarrollo de las principales herramientas existentes en la actualidad para el cálculo de la relevancia de la información.

Métodos Empíricos:

- ✓ Modelación: Empleado para reflejar la estructura, relaciones internas y características de la solución través de diagramas.

Estructura del documento

En este apartado se describe brevemente el contenido de cada uno de los tres capítulos en los que se divide el presente documento.

Capítulo 1. Caracterización del entorno conceptual y tecnológico para mejorar las búsquedas en Orión: serán abordados los conceptos fundamentales que permitirán entender el proceso de cálculo de la relevancia de la información con anotación semántica. Así como la fundamentación de las herramientas y tecnologías que se emplearán en el desarrollo del trabajo de diploma.

Capítulo 2. Diseño y análisis del componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión: serán abordados los temas relacionados con el dominio y la caracterización del sistema a desarrollar, incluyendo los requerimientos planteados y formulándose una propuesta de la solución a implementar.

Capítulo 3. Pruebas funcionales del componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión: se exponen los resultados de la tesis. Serán abordados aspectos relacionados con la implementación del componente y las principales pruebas realizadas al módulo dirigidas a verificar su correcto funcionamiento.

CAPÍTULO 1. Caracterización del entorno conceptual y tecnológico para mejorar las búsquedas en Orión

1.1 Introducción

Con el objetivo de facilitar la comprensión de la presente investigación, en el presente capítulo se realiza un análisis de los sistemas homólogos existentes vinculados al campo de acción, así como una valoración de los conceptos fundamentales relacionados con la temática. Además, se analizan las metodologías, tecnologías y herramientas utilizadas durante el ciclo de desarrollo de la solución que se propone.

1.2 Conceptos asociados al dominio del problema

A continuación se exponen una serie de conceptos fundamentales asociados al dominio del problema para lograr una mejor comprensión de la investigación que se realiza.

1.2.1 Modelo vectorial

La idea básica de este modelo de recuperación vectorial reside en la construcción de una matriz (podría llamarse tabla) de términos o alfabeto inicial y documentos, donde las filas corresponden a estos últimos y las columnas corresponden a los términos incluidos en el alfabeto. El alfabeto se obtiene inicialmente extrayendo todos los términos de la colección de documentos y luego aplicando algoritmos de normalización, extracción de palabras poco relevantes y sin valor; así como eliminando las repeticiones de los mismos términos entre otras técnicas utilizadas. De esta forma, las columnas de esta matriz estarían representadas por una palabra o término determinado del alfabeto y las filas (que en términos algebraicos se denominan vectores) serían equivalentes a los documentos que se expresarían en función de la frecuencia de cada término.

Tanto los documentos como las consultas pueden tratarse matemáticamente como vectores en un espacio n dimensional, del cual viene dado el nombre de modelo vectorial. Este modelo permite además, una vez calculada la similitud entre cada documento de la colección y la consulta, ordenar todos los documentos de la colección en orden decreciente de su grado de similitud con la consulta, incorporando de este modo a los resultados aquellos documentos que satisfacen solo parcialmente los términos de la consulta (Comeche, 2014).

1.2.2 Recuperación de Información (RI)

En la bibliografía consultada referida directamente a la RI, las definiciones aportadas por los diferentes autores en (Baeza y Ribeiro, 1999; Korfhage, 1997; Salton 1983 y Croft 1987), ofrecen características similares. A partir de lo planteado, a los efectos de este trabajo se define recuperación de información como:

El área que trata con la representación, el almacenamiento, la organización, análisis y el acceso a ítems[1] de información. Es el conjunto de tareas mediante las cuales se intenta resolver el problema de encontrar y ordenar documentos relevantes que satisfagan la necesidad de información de un usuario, expresada en un determinado lenguaje de consulta.

1.2.3 Sistema de recuperación de información (SRI)

(Pinto, 2015) Estima que un sistema de recuperación de información permite realizar el proceso donde se accede a una información previamente almacenada y estructurada, mediante herramientas informáticas que permiten establecer ecuaciones de búsqueda específicas.

Están diseñados para el procesamiento de texto en lenguaje natural, raramente estructurado y, por lo general, de semántica[2] ambigua. El objetivo de un SRI es maximizar el número de documentos relevantes devueltos a la vez que se minimiza el número de documentos no relevantes devueltos (Tolosa, 2008).

1.2.4 Motor de búsqueda

Los buscadores o motores de búsqueda son Sistemas de Recuperación de Información que permiten obtener aquellos documentos de mayor relevancia, a partir de un criterio de búsqueda introducido por el usuario. Desde la perspectiva del usuario, los buscadores deben cumplir dos requisitos fundamentales: “un tiempo corto de respuesta y una gran colección de documentos web disponibles en su índice. La calidad de un buscador reside en lo abundante, relevante y actualizada que sea su colección” (Baeza y Ribeiro, 1999).

[1] Se designa como cada uno de los elementos que forman parte de un dato.
[2] Parte lingüística que estudia el significado de las expresiones lingüísticas.

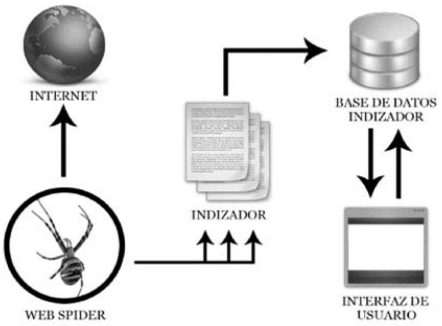


Figura 1: Arquitectura de un Buscador. (Fuente: Herrera 2006).

Teniendo en cuenta el objetivo general de la investigación, se hace necesario el estudio del concepto relevancia de la información, ontología, similitud semántica y anotación semántica.

1.2.5 Relevancia de la información

La recuperación de información intenta resolver el problema de encontrar documentos relevantes que satisfagan la necesidad de información de un usuario. Sin embargo, se ha planteado la dificultad para llevar a cabo esta tarea debido a la imposibilidad de expresar exactamente la relevancia de la información (Tolosa, 2008).

Para definir este concepto en la presente investigación se ha realizado un estudio mediante el cual se hace un análisis breve de las diferentes bibliografías consultadas referentes a la relevancia de la información.

En (Tolosa, 2008) se plantea la relevancia como similitud; tiene que ver con la concordancia entre el término de búsqueda escrito por el usuario en los buscadores y las palabras clave presentes en los contenidos y código de la página web.

Según (Wanson, 1986) se puede resumir la relevancia en dos tendencias: la objetiva y la subjetiva. La objetiva hace énfasis en los sistemas, definiendo cómo la materia de la información recuperada coincide con la de la pregunta. La subjetiva, es la que tiene en cuenta al usuario, el cual juzgará la relevancia del documento devuelto respecto a su necesidad de información.

Para (Schamberg, 1990) la relevancia se refiere a la utilidad, o potencial uso de los materiales recuperados, con relación a la satisfacción de los objetivos, el interés, el trabajo o los problemas intrínsecos del usuario.

Teniendo en cuenta lo antes expuesto se puede concluir que la relevancia surge a partir de la necesidad de información. Permite evaluar la respuesta de un SRI respecto de una consulta de un usuario, demostrando que el documento es relevante a la consulta si son similares ,donde la medida de similitud puede estar basada en diferentes criterios tales como coincidencias de términos, significado y frecuencia de aparición de términos y distribución del vocabulario.

1.2.6 Ontología

Una ontología es una especificación formal de conceptos y relaciones entre los mismos, de forma jerarquizada basada en el conocimiento, restringidos por axiomas[3]. Puede usarse para representar las entidades de un área en particular y sus relaciones. Las ontologías de dominio pueden contribuir a eliminar la confusión terminológica y conceptual generada por las lenguas de especialidad; pero su mayor ventaja, en relación con la terminología, reside en la posibilidad de realizar inferencias a partir del conocimiento explícito (Sánchez y Fernández, 2005).

Se basan en la lógica matemática para representar los conocimientos de un dominio. Emplea lógica de predicados de primer orden (lógica en la cual se utilizan variables, predicados y se permiten cuantificadores de variables. Una ontología proporciona los significados que describen explícitamente la conceptualización del conocimiento representado en una base de conocimientos de las máquinas puedan entender, es por esto que el vocabulario debe ser definido con gran precisión permitiendo diferenciar términos y referenciarlos de manera precisa.

Entre los lenguajes formales para representar las ontologías se destacan los siguientes:

RDF (*Resource Description Framework*): RDF es un método general para descomponer conocimiento en piezas pequeñas, con algunas reglas acerca de la semántica o significado de esas piezas. El punto es que es un método tan simple que pueda expresar cualquier hecho, y a la vez tan estructurado que aplicaciones de computadora puedan usar el conocimiento expresado para hacer cosas útiles. Es un Marco de descripción de recursos desarrollado por W3C (Web semántica) y basado en XML (*eXtensible Markup*

[3] son verdades incuestionables universalmente válidas y evidentes, que se utilizan a menudo como principios en la construcción de una teoría o como base para una argumentación.

Language)[4]. En este lenguaje las relaciones entre dos objetos se establecen mediante el nombre de la relación y dichos elementos, similar a las redes semánticas.

OWL: Lenguaje basado en XML y RDF, pertenece a la WC3 y es el estándar actual. Puede representar los elementos de lógica descriptiva. Además tiene mayor capacidad expresiva. Es el más usado en internet, estando sus elementos definidos con las fuentes de RDF. Tiene tres variantes según la complejidad que se necesite especificar (Sánchez y Fernández, 2005):

OWL Lite: suficiente para los usuarios que tan sólo piden posibilidades de clasificación en la jerarquía de conceptos (clases) de la ontología y restricciones simples. Por ejemplo, aunque OWL proporciona restricciones de cardinalidad, sólo permite valores 0 o 1. Por tanto tiene una complejidad formal inferior a OWL DL.

OWL DL (Description Logic): es el lenguaje indicado para los usuarios que requieren el máximo de expresividad pero exigiendo completitud computacional (se garantiza que todas las conclusiones son computables) y decibilidad (todos los cálculos acaban en un tiempo finito). Incluye todos los constructores de OWL, pero sólo se pueden usar con restricciones; por ejemplo: mientras una clase puede ser a la vez subclase de muchas clases, no puede ser una instancia de otra clase.

OWL Full: se dirige a aquellos usuarios que necesitan la máxima expresividad y la libertad sintáctica de RDF pero sin garantía computacionales. Permite, por ejemplo, aumentar el significado de vocabulario predefinido (en RDF o en OWL).

Componentes de una Ontología

Conceptos: son las ideas básicas que se intentan formalizar. Los conceptos pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento u otros.

Relaciones: representan la interacción y enlace entre los conceptos del dominio. Suelen formar la taxonomía del dominio. Por ejemplo: subclase-de, parte-de, parte-exhaustiva-de, conectado-a.

[4] es un meta-lenguaje que nos permite definir lenguajes de marcado adecuados a usos determinados.

Funciones: son un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología. Por ejemplo, pueden aparecer funciones como categorizar-clase, asignar-fecha.

Instancias: se utilizan para representar objetos determinados de un concepto.

Axiomas: son teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología.

1.2.7 Similitud Semántica

El análisis de las relaciones sintáctico-semánticas que realizaremos en esta investigación se sustenta en la noción de similitud semántica, esta se remonta a los aportes realizados por Russell (1937) con su teoría de clases y similitud. Hoy en día, esta noción se implementa como medida probabilística o como el grado de intercambiabilidad de una palabra por otra, en un contexto similar se comportarán de manera similar (Venegas, 2006).

La similitud semántica en el área de procesamiento de lenguajes naturales, es la medida de la interrelación existente entre dos palabras cualesquiera en un texto, y se realiza mediante la relación existente entre los conceptos de la red semántica[5] . Es un método para medir la semejanza semántica, o la distancia semántica entre dos conceptos según una ontología dada. En otros términos, la similitud semántica se utiliza para identificar conceptos que tienen características comunes. Este concepto se fundamenta en la idea que dos palabras o términos por el hecho de tener su existencia en un mismo documento poseen un contexto similar. Se puede afirmar, por tanto, que las palabras que comparten un contexto similar están generalmente relacionadas, y por consiguiente, se pueden seleccionar sus sentidos a partir de la distancia semántica (Rada, 1989).

La similitud semántica y la distancia semántica se definen a la inversa. Sea C1 y C2 dos conceptos que pertenecen a dos nodos diferentes n1 y n2 en una ontología dada, la distancia entre los nodos (n1 y n2) determina la similitud entre estos dos conceptos C1 y C2. Tanto n1 como n2 pueden considerarse como una ontología (también llamada nodos conceptuales) que contiene un conjunto de términos sinónimo.

Son disímiles los modelos que permiten calcular la similitud y utilizan diferentes representaciones para entender la semántica de la información que necesita manipular. Por lo general, toman un par de palabras

[5]Representación de conocimiento lingüístico en la que los conceptos y sus interrelaciones se representan mediante un grafo.

de entrada y devuelven un valor que indica el grado de similitud que existe entre ambos conceptos. A continuación se ejemplifican algunos de estos modelos.

- ✓ La distancia de Levensthein o distancia de edición (Levensthein, 1966) .La idea consiste en determinar el número mínimo de operaciones requeridas para transformar una cadena de caracteres en otra, estas operaciones son: inserción, eliminación o sustitución de un carácter.
  Por ejemplo la distancia entre los términos hotel y hostal es de dos porque necesitan al menos dos ediciones elementales para cambiar un término en el otro, inserción de *s* antes de *t* y sustitución de *e* por *a*.
- ✓ El algoritmo propuesto en (Resnik, 1993) propone que la similitud entre dos conceptos (se hace referencia al conjunto de términos que apuntan a una misma idea) y de una estructura taxonómica[6], puede ser obtenida mediante la ecuación :

$$sim\ (C_1, \quad C_2) = \max_{C \in (C_{1.,}C_2)} (-\log p(C))$$

Donde:

- ✓ $sim\ (C_1,\ C_2)$ representa el conjunto de conceptos de los cuales tanto C_1como C_2 descienden.
- ✓ $p(C)$ es la probabilidad de encontrar una instancia de tipo C.

- ✓ Una propuesta más para medir la similitud es representada por (Leacock, 1998) considera que la similitud entre $C_{1,}$ y $C_{1,}$, y puede ser obtenida por la siguiente ecuación:

$$sim\ (C_{1,}C_{1,}) = \log(\frac{len(C_{1,}C_2)}{2 * MAX})$$

Donde:

- ✓ $len(C_{1,}\ C_2\ )$cuantifica el número de saltos entre C_1 y C_2 , y
- ✓ MAX es el número de saltos entre el nodo raíz y los nodos hoja de la taxonomía.

[6] ordenada

1.2.8 Anotación Semántica

La anotación semántica se refiere al proceso mediante el cual se adicionan metadatos[7] semánticos a los recursos web, se puede considerar como una información sobre las entidades o conceptos de una ontología. Combina conceptos de metadatos y ontologías, es decir, los campos de los metadatos son asociados con términos en una ontología, los cuales son usados para describir estos campos (Sánchez y Fernández, 2005).

Las anotaciones semánticas se pueden clasificar en: **manuales**, **semi-automáticas** y **automáticas**. Las anotaciones manuales son realizadas generalmente por un anotador humano con la ayuda de un sistema de anotación de textos que vincula los términos de una ontología con los recursos web. Sus principales debilidades radican en que las anotaciones se ven afectadas por factores como: la experticia del anotador sobre el dominio de la información, la motivación personal y su entrenamiento en el proceso de anotación. La anotación semi-automática es realizada por herramientas que explotan técnicas de Procesamiento del Lenguaje Natural (PLN) para encontrar las referencias en el texto a los conceptos existentes en ontologías. Este tipo de herramientas necesitan como entrada un corpus de documentos previamente anotados para que sirvan para entrenar el sistema de anotación (Sánchez y Fernández, 2005). En el caso de la anotación automática, aún queda mucho por hacer, aunque se han obtenido algunos resultados interesantes en el cual se obtiene un sistema de anotación automática para fuentes de datos espaciales utilizando flujos de trabajo científicos.

1.3 Estudio de Sistemas Homólogos

Con la introducción de los conceptos de búsqueda semántica distintos métodos de recuperación de documentos se han desarrollado procurando mejorar los resultados de las búsquedas. Se estudiaron diferentes sistemas teniendo en cuenta cómo recuperan la información semánticamente para obtener resultados relevantes y qué nivel de madurez existe sobre esta problemática.

1.3.1 Google

Es el motor de búsqueda más utilizado del mundo y tiene acceso a miles de millones de sitios web de todo tipo. Su objetivo principal es el de la búsqueda de texto en páginas web, pero también se pueden buscar específicamente libros, blogs, imágenes, videos, documentos académicos, entre otras cosas (Ruiz, 2013).

[7] Son elementos HTML que muestran información sobre la propia página web que los contiene.

Para poder realizar estas búsquedas y posicionarse en el primer lugar de los buscadores utiliza diferentes algoritmos entre ellos:

Algoritmo Hummingbird

Este nuevo algoritmo se enfoca hacia la búsqueda semántica, Google Hummingbird se centra más en el significado de las frases, en entender lo que le preguntamos para poder ofrecernos unos resultados más acordes a lo que buscamos, y ya sea de paso, ofrecernos la publicidad más adecuada.

Las mejoras de Hummingbird son particularmente búsqueda contextual y conversacional, dos áreas que están fuertemente vinculadas a la semántica fundamental y las relaciones entre las palabras. En la búsqueda contextual, Google devuelve cada vez más resultados que coinciden con la intención detrás de la consulta. Los resultados ya no se limitan a las palabras mismas, sino que incluyen una interpretación de la intención de los términos de búsqueda. En la búsqueda conversacional pretende comprender mejor búsquedas en el marco de una conversación, frases largas y preguntas, haciendo que los resultados realmente respondan la intención de la búsqueda del usuario.

1.3.2 Powerset

Es un motor de búsqueda que en un principio tenía como objetivo el de procesar el lenguaje natural, entender la búsqueda completa y no enfocarse solo en palabras claves.

La idea de Powerset se basa en (Converse, 2008):

- ✓ Interpretar la web indexando los contenidos según su semántica.
- ✓ Recibir una consulta e interpretarla semánticamente.
- ✓ Buscar en el repositorio de contenidos y devolver los resultados que tengan la misma semántica.

A pesar de que se hace una búsqueda semántica, se hace especial hincapié en el almacenamiento de los documentos indexados con una estructura lingüística que permita comparar luego semánticamente con las búsquedas.

1.3.3 Kosmix

Es un buscador semántico cuyo objetivo es proporcionar información sobre un tema de búsqueda ofreciendo una colección de links a contenidos, descripciones, videos, enlaces y otros objetos de interés relacionados con la búsqueda (Baquia, 2016).

El enfoque del desarrollo de Kosmix fue justamente permitir a los usuarios explorar la web, ayudando al cliente a organizar mejor sus resultados. El motor primero explora y reúne el contenido de toda la web mediante rastreo creando un catálogo enciclopédico multimedia. Esta exploración va generando una clasificación completa de categorías temáticas, incluyendo conexiones entre todas ellas.

Una vez indexado el contenido de toda la web, al realizar una búsqueda se detecta la temática buscada y mediante un algoritmo propietario al que no se tiene acceso, se recuperan los contenidos de mayor *ranking* según los temas que abarquen a veces hasta sin importar las palabras claves de búsqueda. De esta forma es un buscador que no intenta conocer la consulta semánticamente sino que la categoriza y luego explora en la web que resultados encuentra sobre la temática (Baquia, 2016).

1.3.4 Hakia

Es un proyecto de la compañía Hakia, que se dedica exclusivamente a mejorar la experiencia de los usuarios al realizar búsquedas basándose en la búsqueda semántica, los resultados obtenidos en cualquiera de estas satisfacen tres criterios (Martinez, 2017):

- ✓ Proceden de sitios web creíbles.
- ✓ Representan la información disponible más reciente.
- ✓ Son absolutamente relevantes.

Las búsquedas semánticas que realiza Hakia emplean tres tecnologías:

La primera es OntoSem, que procesa y analiza textos en lenguaje natural. La segunda tecnología es QDEX (*Query indexing technique*), este sistema es una manera de analizar y almacenar el contenido de un artículo web interpretando y entendiendo semánticamente qué quiere decir el mismo. Por último, Hakia se basa en el algoritmo SemanticRank, que toma como entrada los párrafos relevantes que proceden de QDEX para una determinada búsqueda. Este algoritmo determina la relevancia de los resultados que se van a mostrar, y por tanto su orden, basándose en un análisis de las concordancias entre los conceptos relacionados con la búsqueda (no solamente las palabras clave) y las sentencias más relevantes de cada párrafo.

1.3.5 Orión

Orión se ha convertido hoy en la principal herramienta cubana que permite la recuperación de información existente en la intranet nacional. Este buscador usa a Solr como mecanismo de indexación, el cual

proporciona potentes funcionalidades de búsqueda y navegación por facetas, es decir, explorar la información desde diversas perspectivas.

Solr utiliza la librería Lucene como núcleo para el tratamiento del índice, utilizando así los mismos algoritmos implementados en esta librería. Lucene implementa una variante del modelo TFIDF (Term Frequency - Inverse Document Frequency) para determinar el cálculo del score.

La fórmula utilizada es la siguiente:

$$s(q,d) = \left(\sum_{t \in q} tf(t \in d) idf^2(t) t.getBoost() norm(t,d) \right) coord(q,d) queryNorm(q)$$

Donde*:*

q: es la consulta que se desea buscar.

d: documento al cual se le desea hacer el score.

t: es un token[8] o símbolo en q.

A continuación se explica que significa cada factor presente en la fórmula del score:

- ✓ **Tf :** *term frequency* en un documento, medida de cuán frecuente un término aparece en un documento.
- ✓ **idf:** *inverse document frequency,* medida de cuán frecuente un término aparece en todo el índice.
- ✓ **coord:** número de términos de la consulta que fueron encontrados en el documento.
- ✓ **lengthNorm:** medida de cuán importante es un término de acuerdo al número total de términos en el campo.
- ✓ **queryNorm:** factor de normalización para la comparación de las consultas
- ✓ **boost (index|query):** factor de aumento (o disminución) para la ocurrencia de un término en un campo determinado.

[8] componente léxico, cadena de caracteres que tiene un significado coherente en cierto lenguaje de programación.

- ✓ **t.getBoost():** Consulta en impulso de tiempo (boost) para el token t. Se puede dar mayor peso a ciertos tokens o campos mediante boost. Por ejemplo, name:(baseball^5 cubano) impulsaría en 5 al token baseball almacenado en el campo name.

A partir del análisis antes expuesto, se han tomado un grupo de funcionalidades que coinciden en los sistemas estudiados, las cuales son buscadores dirigidos a la búsqueda semántica; teniendo en cuenta sus principales características se conforma la tabla que se muestra a continuación en la que se recogen las principales opciones que se tienen en cuenta para la obtención de resultados relevantes en las búsquedas:

Tabla 1: Comparación entre buscadores en cuanto a la búsqueda semántica.

Funcionalidades	**Buscadores**				
	Google	Kosmix	Powerset	Hakia	Orión
Procesamiento del lenguaje natural	Sí	No	Sí	Sí	No
Interpretar una consulta semánticamente	No	No	Sí	Sí	No
Entender una consulta semánticamente	No	No	Sí	Sí	No
Determinar la relevancia de la información basados en la semántica.	No	No	Sí	Sí	No

Este estudio tuvo como objetivo buscar sistemas homólogos libres que pudieran ser reutilizados para la investigación, no encontrándose ninguno de código abierto, disponible, y sin dificultad de acceso que cumpliese con las necesidades de la propuesta solución. Sin embargo el estudio de estos sistemas permitió identificar un grupo de funcionalidades comunes para tenerlas en cuenta en el desarrollo del componente para el cálculo de la relevancia de la información con anotación semántica, teniendo como principales referentes los buscadores Hakia y Powerset.

1.4 Metodología de software

Una metodología es un conjunto integrado de técnicas y métodos que permite abordar de forma homogénea y abierta cada una de las actividades del ciclo de vida de un proyecto de desarrollo. Es un proceso de software detallado y completo, por lo que se puede decir que comprende los procesos a seguir

sistemáticamente para idear, implementar y mantener un producto software desde que surge la necesidad del producto hasta que cumplimos el objetivo por el cual fue creado (Letelier, 2003).

La metodología de desarrollo utilizada en la investigación es la AUP-UCI, esta metodología es una modificación realizada por la UCI del Proceso Unificado Ágil (AUP por sus siglas en inglés) en unión con el modelo CMMI-DEV v1.3.Se ajusta al ciclo de vida definido por la UCI para el desarrollo de aplicaciones y además es capaz de aumentar la calidad del software.
Consta de 3 fases ya que de las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes de AUP en una sola, llamada Ejecución y se agrega la fase de Cierre.

Algunas de sus principales características son:

- ✓ Se obtiene una comprensión común cliente-equipo de desarrollo del alcance del nuevo sistema y definir una o varias arquitecturas candidatas para el mismo.
- ✓ Permite que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura.
- ✓ Admite que el sistema desarrollado sea probado al completo en el ambiente de desarrollo.

1.5 Herramientas y tecnologías

En las siguientes secciones se exponen brevemente las herramientas y tecnologías empleadas para organizar, facilitar, agilizar y automatizar las tareas generadas durante el transcurso de la investigación, teniendo en cuenta cada una de sus características esenciales.

1.5.1 Modelado

El lenguaje de modelado usado en el diseño y construcción de la solución que se quiere llevar a cabo en la presente investigación es el Lenguaje de Modelado Unificado (UML). (Pessman, 2010) Plantea que es un lenguaje estándar para escribir diseños de software, puede usarse para visualizar, especificar, construir y documentar los artefactos de un sistema de software intensivo.

1.5.2 Herramienta CASE

Las herramientas de Ingeniería de Software Asistida por Computadoras, conocida por sus siglas en inglés como (CASE) de modelado con UML permiten representar el software mediante diagramas que se generan durante las diferentes etapas del proyecto. Entre las facilidades más apremiadas que proporcionan estas herramientas se encuentran: el diseño de proyectos, cálculo de costos, implementación automática de parte del código dado un diseño previo, compilación automática, documentación, detección de errores, entre otras.

Visual Paradigm

Se utiliza Visual Paradigm for UML Enterprise Edition v8.0 como herramienta CASE profesional, que permite realizar todo tipo de diagramas de clases, generar informes y códigos. Dicha herramienta ayuda a una rápida construcción de la aplicación con alta calidad y a un menor costo. Posibilita la elaboración de documentos, además soporta el ciclo de vida completo del desarrollo de software: análisis, diseño, construcción, prueba y despliegue, posibilitando que el modelo y el código permanezcan sincronizado durante todo el ciclo (Visual Paradigm, 2014).

1.5.3 Lenguaje de Programación

Un lenguaje de programación es aquel elemento dentro de la informática que nos permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes (Torres, 2012).

Java

Se empleará Java como lenguaje de programación ya que es de propósito general, además es concurrente, y orientado a objetos esto último debido a que la programación en Java se centra en la creación, manipulación y construcción de objetos. Ha sido desarrollado después de incluir los conceptos de diversos lenguajes, como C, C++, pero tiene un modelo de objetos simples y elimina herramientas de bajo nivel, que suelen inducir a muchos errores. Tiene una gran capacidad de red y proporciona un interfaz sencilla de fácil entendimiento para los usuarios y los desarrolladores considerándose como el lenguaje más simple en comparación a otros lenguajes de programación (Java, 2016). Además las principales herramientas con las que trabaja el buscador Orión utilizan ese lenguaje.

1.5.4 Entorno de desarrollo integrado

Un entorno de desarrollo integrado, es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, PHP, Python, Java, C#, Delphi, Visual Basic, entre otros. (fergarciac, 2013)

Existen diversos entornos integrados de desarrollo (IDE) tales como, NetBeans, MS Visual Studio, Visual C++, Eclipse, seleccionándose este último para el desarrollo de la solución propuesta.

Eclipse

Se empleará Eclipse Neon.2 en su versión Release 4.6.2 como entorno de desarrollo por ser una potente y completa plataforma de programación, desarrollo y compilación de elementos tan variados como aplicaciones Java. Diseñada para ser extendida de forma indefinida a través de *plugins*. Fue concebida desde sus orígenes para convertirse en una plataforma de integración de herramientas de desarrollo (Genbeta Dev, 2014).

Principales características:

- ✓ Perspectivas, editores y vistas: en Eclipse el concepto de trabajo está basado en las perspectivas, que no es otra cosa que una pre configuración de ventanas y editores, relacionada entre sí, y que permiten trabajar en un determinado entorno de trabajo de forma óptima.
- ✓ Gestión de proyectos: el desarrollo sobre Eclipse se basa en los proyectos, que son el conjunto de recursos relacionados entre sí, como puede ser el código fuente, documentación, ficheros configuración, árbol de directorios. El IDE nos proporcionará asistentes y ayudas para la creación de proyectos. Por ejemplo, cuando creamos uno, se abre la perspectiva adecuada al tipo de proyecto que estemos creando, con la colección de vistas, editores y ventanas pre configurada por defecto.
- ✓ Depurador de código: se incluye un potente depurador, de uso fácil e intuitivo, y que visualmente nos ayuda a mejorar nuestro código. Para ello sólo debemos ejecutar el programa en modo depuración (con un simple botón). De nuevo, tenemos una perspectiva específica para la depuración de código, la perspectiva depuración, donde se muestra de forma ordenada toda la información necesaria para realizar dicha tarea.

- ✓ Extensa colección de *plugins*: están disponibles en una gran cantidad, unos publicados por Eclipse, otros por terceros. Al haber sido un estándar de facto durante tanto tiempo (no el único estándar, pero sí uno de ellos), la colección disponible es muy grande. Los hay gratuitos, de pago, bajo distintas licencias, pero casi para cualquier cosa que nos imaginemos tenemos el *plugin* adecuado.

1.5.5 Servidor Web

Un servidor web o servidor HTTP es un programa informático que procesa una aplicación del lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente. Estos generan una respuesta en cualquier lenguaje entendible por el cliente. Para la transmisión de todos estos datos suele utilizarse generalmente el protocolo HTTP.

Tomcat

Es un servidor que funciona como un contenedor de *servlets*, desarrollado bajo el proyecto Jakarta en la *Apache Software Foundation*. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP) de Sun Microsystems (Foundation the Apache Software, 2017).

Incluye el compilador Jasper, que compila JSPs convirtiéndolas en *servlets*. El motor de *servlets* de Tomcat a menudo se presenta en combinación con el servidor web Apache. Puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era solo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y Tomcat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad. Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java.

Por las características anteriormente expuestas se selecciona Apache Tomcat en su versión 7 como servidor web para el correcto funcionamiento del servidor Sorl, pues como contenedor de *servlets* brindará estabilidad y robustez al sistema.

1.5.6 Servidor de indexación Sorl

Apache Solr es una plataforma de búsquedas basada en Apache Lucene, que funciona como un servidor de búsquedas. Sus principales características incluyen búsquedas de texto completo, resaltado de

resultados, clustering dinámico, y manejo de documentos ricos (como Word y PDF). Solr es escalable, permitiendo realizar búsquedas distribuidas y replicación de índices, y actualmente se está usando en muchos de los sitios más grandes de Internet (Foundation the Apache Software, 2017).

Solr está escrito en Java y se ejecuta como un servidor de búsqueda de texto completo independiente dentro de un contenedor de *servlets* como Tomcat. Presenta un esquema de datos configurables, utiliza varios caches para agilizar las búsquedas y realiza navegación de resultados por facetas, es decir, explora la información desde diferentes perspectivas. En la presente investigación se empleará en su versión 4.10.

1.5.7 Gestor de dependencias Maven

Una de las herramientas más útiles a la hora de utilizar librerías es Maven. Esta se utiliza en la gestión y construcción de software. Posee la capacidad de realizar ciertas tareas claramente definidas, como la compilación del código y su empaquetado. Es decir, hace posible la creación de software con dependencias incluidas dentro de la estructura del JAR. Es necesario definir todas las dependencias del proyecto (librerías externas utilizadas) en un fichero propio de todo proyecto Maven, el POM (*Project Object Model*). Este es un archivo en formato XML que contiene todo lo necesario para que a la hora de generar el fichero ejecutable de nuestra aplicación este contenga todo lo que necesita para su ejecución en su interior.

Sin embargo, la característica más importante de Maven es su capacidad de trabajar en red. Cuando definimos las dependencias, este sistema se encargará de ubicar las librerías que deseamos utilizar en Maven Central, el cual es un repositorio que contiene cientos de librerías constantemente actualizadas por sus creadores. Maven permite incluso buscar versiones más recientes o más antiguas de un código dado y agregarlas a nuestro proyecto. Todo se hará de forma automática sin que el usuario tenga que hacer nada más que definir las dependencias.

1.6 Conclusiones parciales

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

- ✓ El estudio de los diferentes sistemas asociados al cálculo de la relevancia de la información, permitió la selección de un modelo semántico para calcular la relevancia de los documentos y mejorar el resultado de las búsquedas del buscador Orión.
- ✓ Actualmente el sistema que utiliza el buscador Orión no satisface las necesidades de información de los usuarios en las búsquedas en la intranet nacional, por lo que demuestra la necesidad de desarrollar un componente para el cálculo de la relevancia de la información capaz de darle solución a esta dificultad.
- ✓ La investigación realizada a las tecnologías informáticas definió la base tecnológica que se utilizará en el desarrollo del componente, seleccionando a AUP-UCI como metodología para guiar los pasos del desarrollo, UML como lenguaje de representación visual y Visual Paradigm 8.0 como herramienta CASE para el modelado del sistema. Además se utilizará el lenguaje de programación Java, para el desarrollo del componente en la plataforma de desarrollo Eclipse.

CAPÍTULO 2. Diseño y análisis del componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión

2.1 Introducción

En este capítulo se abordarán aspectos fundamentales relacionados con el diseño del sistema a desarrollar. Se identifican los requerimientos funcionales y no funcionales tenidos en cuenta para la realización de la solución propuesta. Se realiza el modelado de casos de uso del sistema y las descripciones textuales de estos, así como el diagrama del modelo del dominio, mediante el cual se representan las clases conceptuales significativas del problema a resolver. Además se incluirán los patrones de diseño utilizados para lograr buenas prácticas de diseño y programación.

2.2 Propuesta de solución

Para darle respuesta al problema de la presente investigación se define como propuesta de solución desarrollar un componente para el cálculo de la relevancia de la información con anotación semántica. Por lo que en este estudio, se propone un modelo semántico, consiste en un algoritmo de rango de página basado en la información que podría extraerse de las consultas de los usuarios y la ontología de un documento determinado. La puntuación de relevancia se mide como la probabilidad de que un recurso dado (documento) contenga las relaciones que existían en la mente del usuario en el momento de la definición de la consulta. La idea es usar las relaciones existentes en la ontología, denominadas "enlaces virtuales" y aplicarlas a un conjunto de documentos para aumentar las probabilidades de encontrar las relaciones implícitas hechas por el usuario en el momento de la consulta (Rojas, 2012).

En el modelo semántico propuesto, se crea un sistema de clasificación basado en una estimación de la probabilidad de que palabras clave y / o conceptos dentro de un documento "A", estén vinculados entre sí de una manera similar al de la mente del usuario en el momento de la definición de la consulta.

Este algoritmo es independiente a:

- ✓ La forma de elaborar las ontologías para los documentos.
- ✓ La forma en que se separan los diferentes términos de la consulta y se asocien a conceptos en la ontología.

- ✓ Los principales procesos son los siguientes (Rojas, 2012):
- ✓ Inicialmente se obtienen los términos y / o conceptos de la consulta realizada por el usuario, los cuales serían los conceptos que se asocian en la ontología.
- ✓ Una vez asociados los términos, el algoritmo genera subgrafos por cada documento de los cuales se generan árboles de expansión.
- ✓ Calcula las probabilidades por cada árbol de expansión teniendo en cuenta las relaciones que existen entre los conceptos asociados.
- ✓ Calcula la puntuación de relevancia que se define como :

$$PR(Q, A) = \frac{\sum_i^n PA(i)}{C(i)} + L$$

$$PA = T_{12} \cap T_{23}$$

$$T = \delta/\mathrm{r}$$

En donde:

$PR(Q, \mathrm{A}, L)$: es la puntuación de relevancia del documento ‘’A’’, dada una consulta Q.

$PA(i)$: son los valores de probabilidad que tienen cada uno de los árboles de expansión que se derivan del subgrafo i que se obtiene del documento ‘’A’’.

$C(i)$: es el número total de árboles de expansión del subgrafo i.

L: es la longitud de los árboles de expansión[9], cantidad de conceptos (nodos del árbol) – 1.

T: es la probabilidad que tiene una arista (relación) en el árbol de expansión.

[9] definido como el mayor conjunto de aristas de un grafo ‘’G’’ que no contiene ciclos, o como el mínimo conjunto de aristas que conecta todos los vértices.

δ: es el peso que tiene una arista, cantidad de veces que aparece la relación en el documento.

r: cantidad de veces que aparece la relación en la consulta.

- ✓ Por último ordena los documentos según la puntuación de relevancia, en orden decreciente.

2.3 Modelo de dominio

El modelo de dominio también conocido como Modelo Conceptual es una representación visual de los conceptos u objetos del mundo real, significativos para un problema o área de interés (Ríos, 2007).

Teniendo en cuenta que la definición de procesos y roles del negocio se hace difícil encontrarlos, se hace necesario describir el contexto en el cual la aplicación va a ser desarrollada, mediante una serie de conceptos, entidades y sus relaciones, agrupándose en un modelo de dominio con el fin de contribuir a la comprensión del contexto del sistema.

2.3.1 Descripción de Clases del Modelo de Dominio

Para lograr un mejor entendimiento de las necesidades del usuario y los requisitos de software de la propuesta de solución se realizó un modelo conceptual, el cual constituye la herramienta fundamental para garantizar la comprensión y descripción de las clases o conceptos y sus relaciones más importantes dentro del contexto del problema. A continuación se presenta la descripción de los conceptos identificados en la presente investigación.

Tabla 2: Descripción de las clases del modelo del dominio.

Concepto	**Descripción**
Usuario	Persona que realiza las búsquedas.
Interfaz web	Constituye la vista mediante la cual se le muestran los resultados al usuario.
Buscador	Constituye una herramienta de recuperación de información en la Web.
Rastreador	Mecanismo que se encarga de recopilar los documentos en la Web.
Intranet	Red informática donde están los documentos.

2.3.2 Diagrama de Clases del Modelo del Dominio

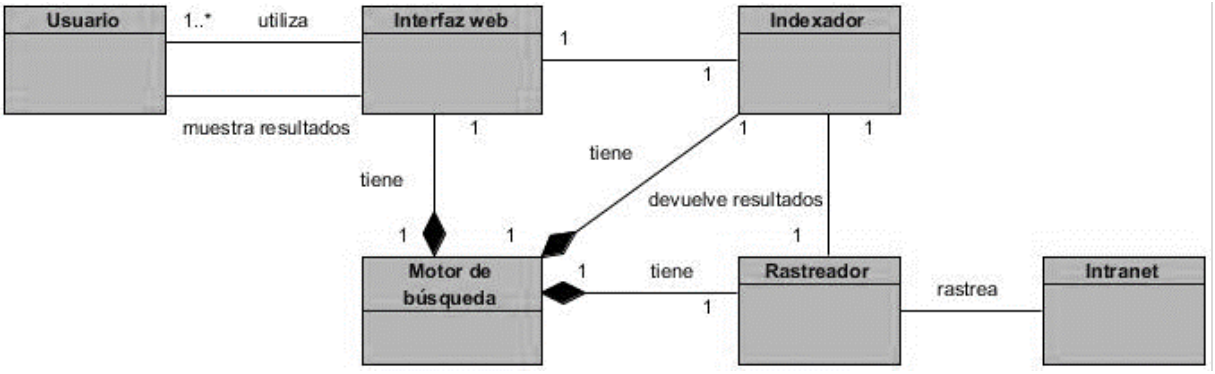


Figura 2: Diagrama del modelo del dominio

2.4 Especificación de los Requisitos del Software

Los requerimientos del sistema son características, necesidades o funcionalidades que el sistema deberá responder una vez terminada la solución propuesta. El propósito fundamental de la captura de los requisitos es guiar el desarrollo del sistema exitosamente con una especificación correcta y exhaustiva de los mismos.

2.4.1 Requisitos funcionales

Permiten establecer lo que el sistema debe hacer, sus características fundamentales, y las restricciones en el funcionamiento del sistema y los procesos de desarrollo del software (Sommerville, 2005). A continuación se presentan los requisitos funcionales que debe cumplir el sistema que se propone.

Tabla 3: Requisitos funcionales.

Código	Descripción (Requisitos Funcionales)	Prioridad
RF1	Obtener la consulta realizada por el usuario, luego de ser procesada.	Alta
RF2	Obtener los documentos, referidos a la consulta procesada.	Alta
RF3	Cargar datos de archivos RDF.	
RF4	Construir el subgrafo de cada documento basado en los conceptos y relaciones de la ontología.	Alta
RF5	Calcular la relevancia de la información de los documentos.	Alta

RF6	Ordenar los documentos por relevancia.	Alta

2.4.2 Requisitos no funcionales

Los requisitos no funcionales se refieren a las propiedades emergentes del sistema como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento (Sommerville, 2005). A continuación se presentan los requisitos no funcionales que debe cumplir el sistema que se propone.

Requerimientos de software

- ✓ RNF 1. Se requiere la instalación de la máquina virtual de Java en su versión 8.0.
- ✓ RNF 2. Se requiere la instalación del servidor web y de *servlets* Tomcat 7 para el correcto funcionamiento del servidor de Solr.

Requerimientos de hardware

- ✓ RNF 3. Para el servidor de índice se necesita como mínimo: 4 GB RAM, CPU de 4 núcleos y al menos 60 GB.

Requerimientos de diseño e implementación

- ✓ RNF 4. Como lenguaje de programación para el desarrollo del *plugin* se deberá utilizar Java.

Requerimientos de usabilidad

- ✓ RNF 5. Se requiere el uso de herramientas y recursos de *software* libre, las cuales se podrán usar, modificar y distribuir libremente.

Requerimientos de eficiencia

- ✓ RNF 6. El sistema debe ser capaz de responder como máximo en 1000ms.
- ✓ RNF 7. La precisión al devolver los resultados debe ser mayor que 0.8.

2.4.3 Historias de usuario (HU)

Las HU sirven para registrar los requerimientos de los clientes según el negocio y son utilizadas para poder realizar la estimación de cada una de las iteraciones durante la fase de planificación. Las HU son escritas

por el equipo de trabajo en conjunto con los clientes en base a lo que se estima que es necesario para el sistema (Balarezo, 2013).

Tabla 4: Historia de usuario #3.

Historia de usuario	
Número: HU_3	**Nombre Historia de Usuario:** Cargar datos de archivo RDF.
Prioridad en negocio : Alta	
Descripción: El sistema debe ser capaz de obtener el grafo de un documento, a partir de un archivo RDF que lo describa.	
Prototipo	

Tabla 5: Historia de usuario #4.

	Historia de usuario
Número: HU_4	**Nombre Historia de Usuario:** Construir el subgrafo de cada documento basado en los conceptos y relaciones de la ontología.
Prioridad en negocio : Alta	
Descripción: El sistema debe ser capaz de construir el subgrafo de cada documento basado en los conceptos y relaciones de la ontología, de forma tal que se usen las relaciones existentes en la ontología, denominadas enlaces virtuales y aplicarlas al conjunto de documentos, para aumentar las probabilidades de encontrar las relaciones implícitas hechas por el usuario en el momento de la consulta.	
Prototipo	

Tabla 6: Historia de usuario # 4

	Historia de usuario
Número: HU_5	**Nombre Historia de Usuario:** Calcular la relevancia de la información de los documentos.
Prioridad en negocio : Alta	
Descripción: El sistema debe ser capaz de calcular la relevancia de la información de los documentos.La puntuación (o score) de relevancia se mide como la probabilidad de que un documento dado contenga las relaciones que existen en la mente del usuario en el momento de definición de la consulta. Se crea un sistema de clasificación basado en una estimación de la probabilidad de que palabras clave y / o conceptos dentro de un documento "A", estén vinculados entre sí de una manera similar al de la mente del usuario en el momento de la definición de la consulta.	
Prototipo	

Tabla 7: Historia de usuario #5.

	Historia de usuario
Número: HU_6	**Nombre Historia de Usuario:** Mostrar los documentos ordenados.
Prioridad en negocio : Alta	
Descripción: El sistema debe ser capaz de mostrar lo documentos ordenados, teniendo en cuenta los resultados obtenidos al calcular la relevancia de la información de dichos documentos.	
Prototipo	

2.5 Estilo arquitectónico

Un estilo arquitectónico es una transformación impuesta al diseño de todo un sistema, teniendo como objetivo establecer una estructura para todos los componentes del mismo (Pressman, 2010).

Se propone la utilización, del estilo arquitectónico tubería y filtros, la cual consiste en transformar un flujo de datos en un proceso comprendido por varias fases secuenciales, los datos se transportan a través de las tuberías entre los filtros, transformando gradualmente las entradas en salidas (Reynoso, 2004). En la presente investigación se usa este estilo con el propósito de organizar la infraestructura de comunicación entre los sub-procesos que componen la etapa de la obtención de la consulta y el cálculo de la relevancia de dichos documentos.

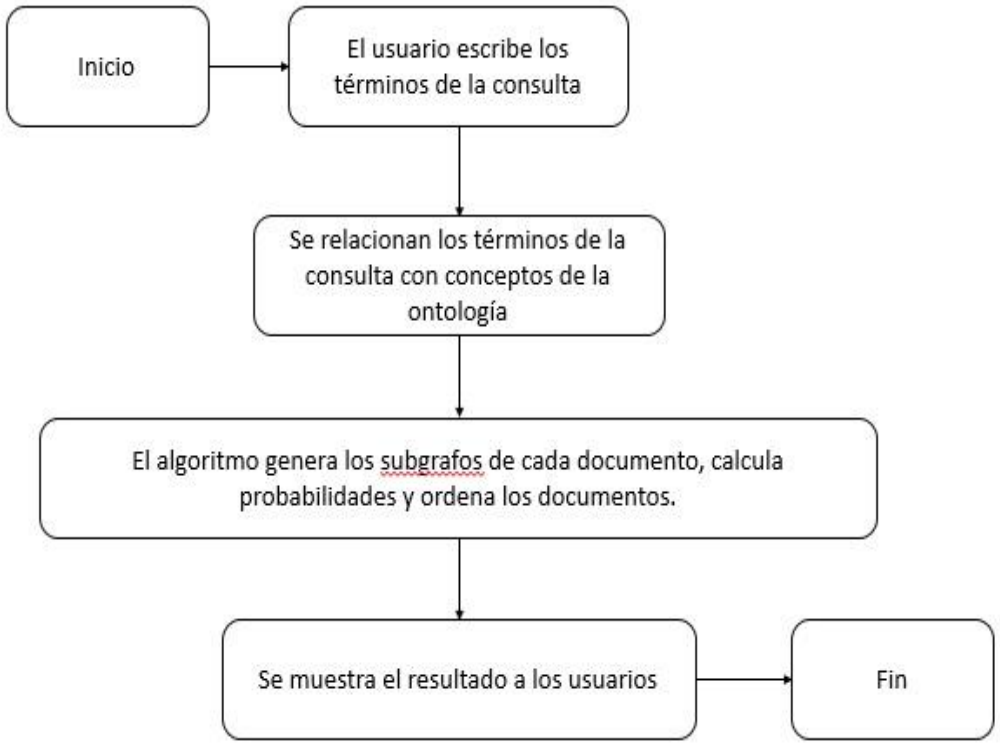


Figura 3: Diagrama de flujo de trabajo.

2.5.1 Arquitectura del sistema

La arquitectura de software es la organización fundamental de un sistema. Define los componentes que lo integran, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución (Reynoso, 2004). Para el desarrollo y organización estructural del componente se propone como arquitectura base, una arquitectura basada en componentes, ya que describe una aproximación de ingeniería de software al

diseño y desarrollo del sistema, además es un estilo de diseño para aplicaciones compuestas de componentes individuales.

A continuación se representa la arquitectura de la solución propuesta representada por dos componentes principales, el componente de indexación Sorl y el componente desarrollado en la presente investigación:

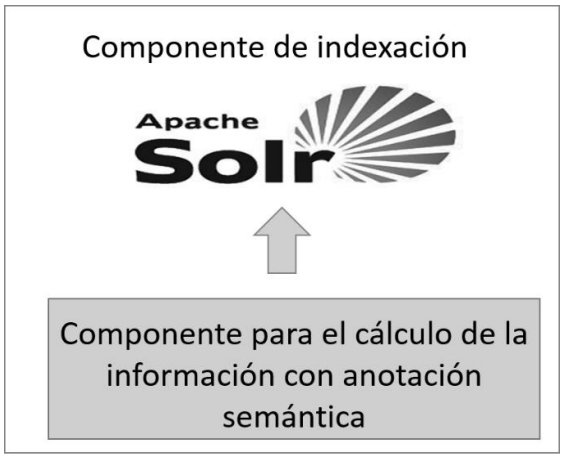


Figura 4: Arquitectura del sistema propuesto.

2.6 Patrones de diseño

Los patrones de diseño 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 (Larman, 2004). Los mismos identifican: clases, instancias, roles, colaboraciones y la distribución de responsabilidades.

En la presente investigación se utilizaron los patrones GRASP (Patrones Generales de Software para Asignación de Responsabilidades), los cuales describen los principios fundamentales de la asignación de responsabilidades a objetos.

Experto: Este patrón indica que la responsabilidad de la creación de un objeto o la implementación de un método debe recaer sobre la clase que conoce toda la información necesaria para realizarla (Larman, 2004). En este caso este patrón se evidencia en la clase RelationBasedRankComponent encargada de crear los subgrafos de los documentos.

Creador: Este patrón es empleado para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia del mismo solo pueda ser creada por la clase que contiene la información necesaria para ello (Larman, 2004). En este caso este patrón se evidencia en la clase OntologyGrap encargada de tener los conceptos y las relaciones de una ontología, que será utilizado posteriormente por RelationBasedRankComponent.

Alta cohesión: La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con funciones estrechamente relacionadas que no realicen un trabajo enorme (Larman, 2004). Está basado en la asignación de responsabilidades teniendo en cuenta que la cohesión permanezca alta. Este patrón se pone de manifiesto entre la relación que existe entre las clases RelationBasedRankComponent y IPageRankComponent.

Bajo acoplamiento: Plantea que debe existir una alta reutilización entre las funcionalidades de las clases con una mínima dependencia, contribuyendo al mantenimiento de las mismas. Este patrón es fundamental siempre que se desee realizar un diseño de clases independientes que puedan soportar los cambios de manera sencilla. El empleo de los patrones Experto y Creador favorecen al bajo acoplamiento entre las clases del sistema (Larman, 2004).Este patrón se evidencia en la clase IPageRankComponent encargada de mostrar los documentos ordenados.

2.7 Modelo de Diseño

El modelo de diseño se utiliza como medio de abstracción del modelo de implementación y el código fuente del software. Su objetivo fundamental es transmitir, a través de la representación mediante diagramas, una comprensión en profundidad de los aspectos relacionados con los requerimientos no funcionales y restricciones concernientes a los lenguajes de programación (Larman, 2004).

2.7.1 Diagrama de clases del diseño

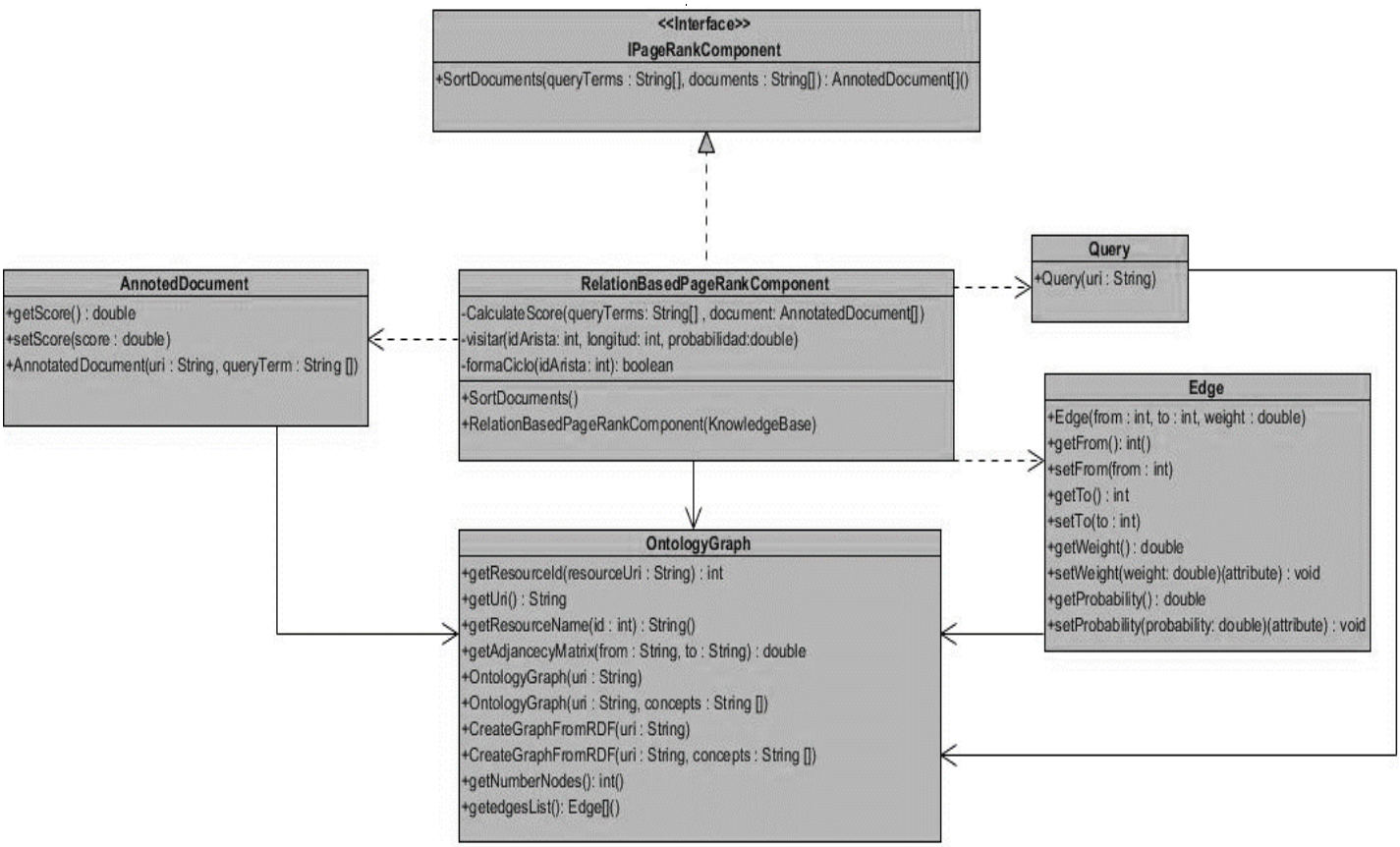


Figura 5: Diagrama de clases del diseño.

2.8 Modelo de despliegue

El diagrama de despliegue se utiliza para mostrar la estructura física del sistema, incluyendo las relaciones entre el hardware y el software que se despliega, estas relaciones son representadas por los protocolos de comunicación que se utilizan para acceder a cada uno (SparxSystems, 2014). En la siguiente figura puede visualizarse el diagrama de despliegue definido para la solución propuesta:

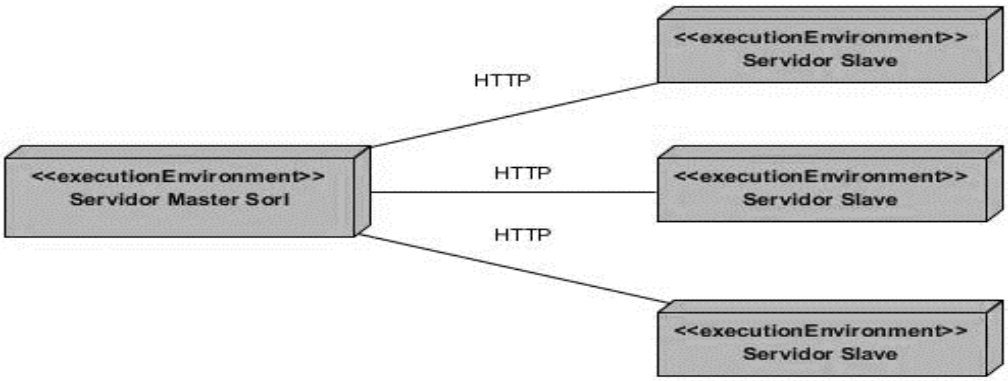


Figura 6: Diagrama de despliegue.

2.9 Conclusiones parciales

En este capítulo se abordaron una serie de aspectos correspondientes al análisis y diseño del componente para el cálculo de la relevancia con anotación semántica para el buscador Orión llegándose a la siguiente conclusión:

- ✓ La representación y descripción de los artefactos generados garantizaron un mejor entendimiento de los flujos de trabajos presentes en el proceso de cálculo de la relevancia de la información.
- ✓ La especificación de los requisitos funcionales y no funcionales del sistema, dieron paso a una mejor comprensión de los resultados que se pretenden obtener de una manera precisa y sirvieron de guía para la implementación del sistema.
- ✓ La definición de la arquitectura y los patrones de diseño a utilizar, permitieron establecer las bases para fomentar la reutilización y las buenas prácticas de programación durante la fase de implementación, así como disminuir el impacto de los cambios futuros en el código fuente.
- ✓ La elaboración del diagrama de despliegue permitió identificar la disposición física de los artefactos de la herramienta informática a desarrollarse.

CAPÍTULO 3. Pruebas funcionales del componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión.

3.1 Introducción

La implementación del sistema es una de las fases imprescindibles dentro del proceso de desarrollo de software. Esta fase comprende la materialización, en forma de código, de todos los artefactos, descripciones y arquitectura propuestos en la etapa de análisis y diseño; con el objetivo de conformar el producto final requerido por el cliente (Larman, 2004). Aparejado al proceso de implementación, el software que se construye debe ser sometido a determinadas pruebas que corroboren la correspondencia entre el producto y los requisitos definidos en las etapas anteriores. A esta etapa se le conoce como validación del sistema y en ella, pueden realizarse diferentes tipos de pruebas en función de los objetivos de las mismas.

3.2 Modelo de componentes

El modelo de componentes representa la forma en que es estructurado un sistema informático atendiendo a las diferentes partes que lo componen. Partiendo de este punto, (Sommerville, 2005) puntualiza que cada componente debe ser tratado como una unidad de composición independiente e indispensable dentro de un sistema, y que puede contraer relaciones de dependencia con otros componentes. Algunos ejemplos de componentes físicos lo constituyen los archivos, módulos, librerías, ejecutables, binarios, entre otros.

3.2.1 Diagrama de componentes

Un componente es una parte física de un sistema (modulo, base de datos, programa ejecutable). Se puede decir que un componente es la materialización de una o más clases (Hernández, 2013).

El diagrama de componentes establece la relación entre componentes de software (librerías, binarios, ejecutables y códigos fuentes), dependencias, comunicación y ubicación en el sistema. A continuación se presenta el diagrama de componentes correspondiente al componente para el cálculo de la relevancia de la información con anotación semántica para el buscador Orión:

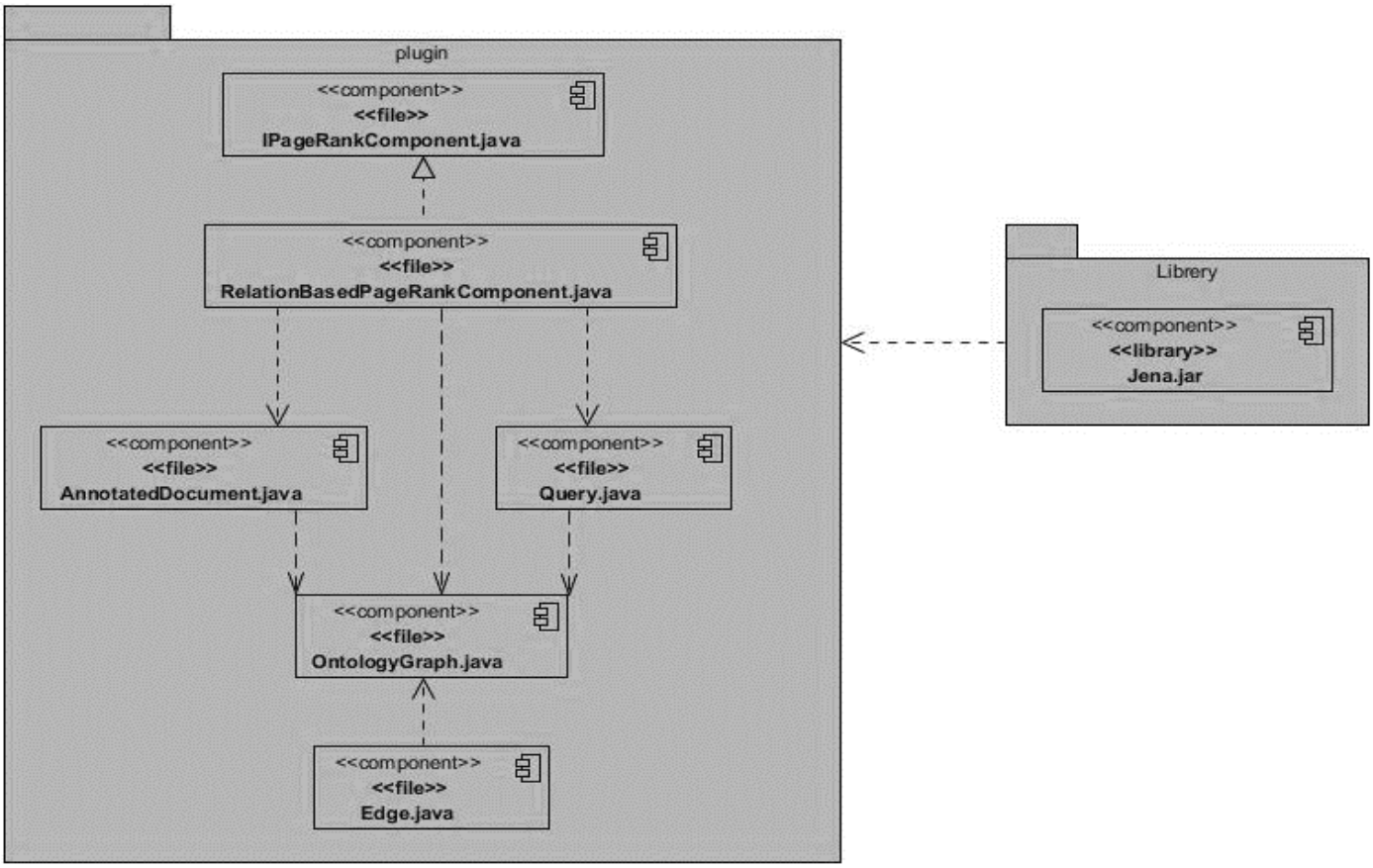


Figura 7: Diagrama de componente.

3.3 Estándares de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Resultan importantes en cualquier proyecto de desarrollo los mismos ayudan a asegurar que el código tenga una alta calidad, menos errores, pueda ser mantenido fácilmente y sea legible. El estándar de codificación debería establecer cómo operar con la base de código existente (Microsoft, 2017). A continuación, se explican y ejemplifican los estándares de codificación utilizados en la propuesta de solución**:**

3.3.1 Estructuras de control

Las estructuras de control incluyen if, for, while, swhitch, entre otros. Las mismas deben presentar un espacio entre la palabra de control (if, for) y el paréntesis que abre, para distinguirlos de las llamadas a las funciones. La llave de apertura ({) se situará en la misma línea que la definición de la estructura de control.

```
if (longitud == totalConceptosDocumento-1 ) {
    probabilidadTotalAcumulada *= probabilidad;
    totalArbolesExpansion++;
    return ;
}
```

3.3.2 Operadores

Todos los operadores binarios como: +,-, ! , =, =, ==, >, deben tener un espacio antes y después del operador, los operadores unarios como ++, no utilizan espacios entre el operador y la variable.

```
for (int idArista = 0; idArista < totalRelacionesDocumento; ++idArista)
    Edge arista = aristasDocumento.get(idArista);
    visitedEdge[arista.getFrom()][arista.getTo()] = true;
```

3.3.3 Nombres de clases y funciones

Los nombres de las clases y las funciones adoptarán la notación CamelCase y no se utilizará el guion bajo como delimitador entre palabras.

```
public class RelationBasedPageRankComponent implements IPageRankComponent {
```

3.3.4 Utilización del punto y coma (;)

En Java es siempre obligatorio el terminador de línea (;).

```
int from = aristasDocumento.get(i).getFrom();
int to = aristasDocumento.get(i).getTo();
```

3.4 Validación de la propuesta de solución

El aseguramiento de la calidad del software se ha convertido en una necesidad prioritaria y en una tarea vital en el desarrollo de cualquier sistema informático por la necesidad de garantizar que el producto cumpla con los requisitos especificados y que no presente errores. Las pruebas de software responden

fundamentalmente a dos interrogantes, ¿se ha obtenido un buen producto?, ¿se ha desarrollado de forma correcta? Este concepto da lugar al proceso de verificación y validación del software.(Zamora, 2011)

A continuación se detallan los tipos de pruebas de software aplicadas al componente implementado:

3.4.1 Pruebas de integración

Este tipo de prueba es una técnica que permiten probar en conjunto distintos de subsistemas funcionales o componentes del proyecto para verificar que interactúan de manera correcta y que se ajustan a los requisitos especificados (sean estos funcionales o no) (Echavarria, 2014).

Para poder llevar a cabo estas pruebas se creó un escenario especial, un entorno semántico en el que se utilizaron documentos de prueba, con ontologías de características especiales, y el usuario de la aplicación de prueba (no el usuario del buscador) fue el encargado de asociar los términos a conceptos de la ontología. Mediante la ejecución de las pruebas de integración se pudo verificar el correcto funcionamiento del buscador Orión con el componente incorporado.

Se realizaron 3 iteraciones:

Iteración 1:

- ✓ No se lograba realizar el subgrafo por cada documento obtenido referido a la consulta realizada.
- ✓ No lograba mostar la puntuación de relevancia de los documentos.
- ✓ No mostraba los documentos ordenados.

Iteración 2:

- ✓ Se asignaba la misma puntuación de relevancia a todos los documentos.
- ✓ No mostraba los documentos ordenados.

En las primeras iteraciones se detectaron errores que fueron corregidos. En la última iteración no se detectó ningún error, como se muestra en la siguiente tabla:

Tabla 8: Errores por cada iteración de las pruebas de integración.

Errores	Primera iteración	Segunda iteración	Tercera iteración
Detectados	3	2	0
Resueltos	2	2	0
Pendientes	1	0	0

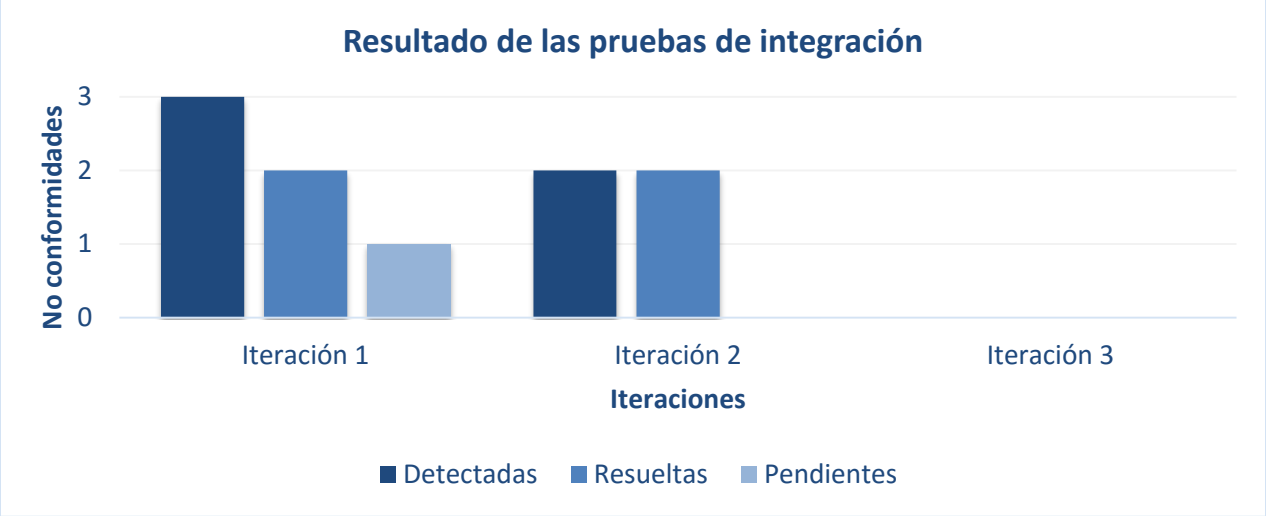


Figura 8: Resultados de las pruebas de integración.

3.4.2 Pruebas funcionales

Las pruebas funcionales son aquellas que se aplican a un software determinado, con el objetivo de validar que las funcionalidades implementadas funcionen de acuerdo a las especificaciones de los requisitos definidos con anterioridad. Se hacen mediante el diseño de modelos de prueba que buscan evaluar cada una de las opciones con las que cuenta el paquete informático. Dicho de otro modo son pruebas específicas, concretas y exhaustivas para probar y validar que el software hace lo que debe y sobre todo, lo que se ha especificado.

Casos de prueba

Las celdas de la tabla contienen V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante.

Caso de Prueba 1: SC RF4_Calcular la relevancia de la información de los documentos.

Tabla 9: Caso de prueba #1.

Escenario	Descripción	Documento	Respuesta del sistema	Flujo central
EC 1.1 Calcular la relevancia de la información de los documentos.	El sistema calcula la relevancia de la información de los documentos obtenidos referidos a la consulta realizada	V Documento contiene términos asociados a la consulta realizada.	El sistema devuelve una puntuación de relevancia diferente de 0.	El usuario realiza una consulta en el buscador. El sistema obtiene los términos y / o conceptos de la consulta y los documentos referidos a esta.
EC 1.2 Calcular la relevancia de la información de los documentos.	El sistema calcula la relevancia de la información de los documentos obtenidos referidos a la consulta realizada.	V Documento no contiene términos asociados a la consulta realizada.	El sistema devuelve una puntuación de relevancia igual a 0.	El usuario realiza una consulta en el buscador. El sistema obtiene los términos y / o conceptos de la consulta y los documentos referidos a esta.
EC 1.3 Calcular la relevancia de la información de los documentos.	El sistema no calcula la relevancia de la información de los documentos porque no obtiene ninguno referido a la consulta realizada.	I No existe documento referido a la consulta realizada	El sistema no devuelve ninguna puntuación	El usuario realiza una consulta en el buscador. El sistema no obtiene los términos y / o conceptos de la consulta y los documentos referidos a esta.

Tabla 10: Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Documento	Campo de texto.	No	Este campo permite todos los caracteres.

A continuación, se muestran los resultados de las pruebas realizadas, se realizaron 4 iteraciones, en las 3 primeras se detectaron un total de 14 no conformidades que fueron corregidas satisfactoriamente, no detectándose ninguna en la 4 iteración.

Tabla 11: Errores por cada iteración de las pruebas funcionales.

Errores	Primera iteración	Segunda iteración	Tercera iteración	Cuarta iteración
Detectados	10	3	2	0
Resueltos	10	2	2	0
Pendientes	0	1	0	0

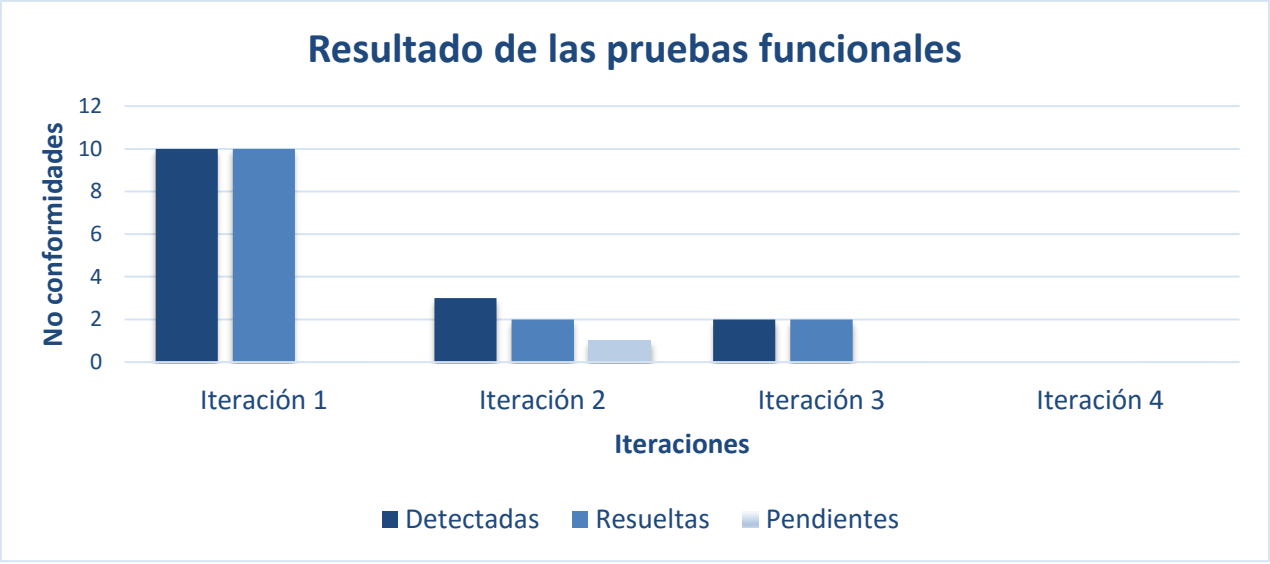


Figura 9: Resultados de las pruebas funcionales.

3.5 Validación de la hipótesis de investigación

Con el propósito de evaluar los indicadores: exhaustividad o de respuesta, y precisión o indicador de pertinencia, en el componente para el cálculo de la relevancia de la información con anotación semántica, correspondiente a la variable dependiente definida como parte de la hipótesis de investigación, se realizó un experimento donde se compararon los resultados obtenidos en el buscador Orión actual y el componente implementado. Para el experimento se tomaron 50 documentos de los cuales solo 20 eran relevantes al criterio de prueba.

Al realizar la prueba se recuperaron 22 documentos de los cuales solo 16 eran relevantes, el porcentaje que se obtuvo de los resultados obtenidos por el componente desarrollado referentes a los indicadores descritos (Ver fórmulas para el cálculo de indicadores en el Anexo 3) se representan en la siguiente tabla:

Tabla 12: Resultados de la medición del indicador "Exhaustividad".

Componente	Resultado
Componente desarrollado	80%

Tabla 13: Resultados de la medición del indicador "Precisión".

Componente	Resultado
Componente desarrollado	72%

Teniendo en cuenta que obtener un porcentaje de respuesta por los indicadores precisión y exhaustividad entre 60% y 70% se considera un buen resultado (Moreiro, 2002), y que actualmente el buscador Orión presenta muy bajo porcentaje respecto a estos. Se evidencia que utilizando el componente desarrollado se obtienen resultados con mayor relevancia para el usuario con respecto a Orión, siendo esto satisfactorio para la validación de la hipótesis de la presente investigación.

3.6 Conclusiones parciales

En el presente capítulo se abordó todo lo referente a la fase de diseño, implementación y pruebas del componente, haciendo una descripción de cada uno de los artefactos generados durante el desarrollo del mismo, arribando a las siguientes conclusiones:

- ✓ La ejecución de pruebas al componente permitió detectar las deficiencias presentes, y corregirlas en el menor tiempo posible y ofrecer una aplicación con mayor calidad.
- ✓ La utilización de estándares de codificación de código permitió adoptar una estructura homogénea que facilita la comunicación y asegura la calidad, menos errores y fácil mantenimiento.
- ✓ La aplicación del método experimental y la realización de cálculos estadísticos, aportaron elementos sustanciales que permitieron validar la hipótesis de investigación planteada con anterioridad, y con ello la factibilidad del componente planteado.

Conclusiones Generales

Con la realización y culminación del presente trabajo, se llegaron a las siguientes conclusiones:

- ✓ El enfoque ágil propuesto por la metodología AUP- UCI y el estudio de las herramientas, tecnologías y metodologías permitieron analizar y describir los subprocesos que se debían ejecutar, determinando cuáles se ajustaban a las características del sistema a desarrollar.
- ✓ El diseño contribuyó a definir la estructura del sistema, teniendo en cuenta distintos patrones, lo cual propició la aplicación de las buenas prácticas del desarrollo de *software*.
- ✓ La evaluación de las pruebas de *software* realizadas permitió erradicar las insuficiencias detectadas en el componente desarrollado logrando así un producto funcional conforme al objetivo general de la presente investigación.
- ✓ La validación de la hipótesis a través de un experimento demostró la calidad de la herramienta desarrollada, resultado que fue satisfactorio para la autora quedando demostrada la hipótesis de la investigación.

Por todo lo anteriormente expuesto, se concluye que el objetivo planteado para el presente trabajo se cumplió satisfactoriamente, poniendo en práctica todas y cada una de las tareas propuestas para el desarrollo del componente para el cálculo de la relevancia de la información con anotación semántica.

Recomendaciones

Teniendo en cuenta los resultados obtenidos en la realización del presente trabajo de diploma se recomienda:

- Integrar el componente para el cálculo de la relevancia de la información en el entorno real de Orión y Red Cuba.
- Permitir que el buscador Orión pueda realizar un completo procesamiento semántico de los documentos teniendo en cuenta el componente desarrollado en la presente investigación.

Referencias Bibliográficas

Baeza Yates, R., Ribeiro Neto, B. 1999. Modern information retrieval. New York : ACM:press, 1999. 0-201-39829-X.

Belloch, Consuelo. 2012. Las Tecnologías de la Información y Comunicación en el aprendizaje. Departamento de Métodos de Investigación y Diagnóstico en Educación. Valencia : s.n., 2012.

Comeche, J., A., M. Los modelos clásicos de recuperación de información y su vigencia. [PDF].e-printsin library & informtaion science, 2014.[Citado el: 3 de diciembre de 2016]. Disponible en: [http://eprints.rclis.org/9662/].

Converse, Tim, et al. 2008. Powerset's natural language wikipedia search engine. Wikipedia and Artificial Intelligence. San Francisco : s.n., 2008.

Croft, W. Bruce. 1987. Approaches to intelligent information retrieval. [book auth.] Croft Bruce W. Information Processing & Management. Massachusetts : Board, 1987.

fergarciac. 2013. Entorno de Desarrollo Integrado (IDE). [Online] enero 25, 2013. [Cited: diciembre 3, 2016.] https://fergarciac.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/.

Foundation, The Apache Software. Apache Tomcat Specifications. [Online] mayo 11, 2017 . [Cited: mayo 15, 2017.] http://tomcat.apache.org/.

Genbeta Dev. 2014. Eclipse IDE. [Online] enero 10, 2014. [Cited: diciembre 3, 2016.] https://www.genbetadev.com/herramientas/eclipse-ide/.

Grainger, T.,Potter,T. 2014. Solr in Action. s.l. : Manning, 2014. 1617291021.

Hernández, L. Modelo de implementación. [Online] Modelo de implementación, 2013. [Cited: 15 de Marzo de 2017]. Disponible en: http://ithleovi.blogspot.com/2013/06/unidad-5-modelo-deimplementacion-el.html

Herrera, A., G., L. Modelos de Sistemas de Recuperación de Información Lingüística Difusa. 2006.

Java. 2016. Conozca más sobre la tecnología Java. [Online] 2016. [Cited: diciembre 3, 2016.] https://www.java.com/es/about/.

Korfhage, R.R. 1997. Information Storage and Retreival. New York : Wiley Computer Publishing, 1997. 0-471-14338-3..

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.

Leacock, Claudia,Chodorow, Martin. 1998. Combining local context and WordNet similarity for word sense identification. WordNet: An electronic lexical database. Massachusetts : s.n., 1998, Vol. 49, pp. 265-283.

Letelier, Patricio, Sánchez, Emilio. 2003. Metodologías ágiles en el desarrollo de software. Alicante : s.n., 2003.

Levensthein, V. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. s.l. : Publicaciones Soviet Physics Doklady, 1966. pp. 707-710. Vol. 10.

Moreiro, José Antonio. Criterios e indicadores para evaluar la calidad del análisis documental de contenido. Ciência da informaçao, 2002, vol. 31, no 1.

NetBeans. 2015. Welcome to NetBeans. [Online] 2015. [Cited: diciembre 3, 2016.] https://netbeans.org.

Pessman, Roger S. 2010. Ingenierís de Software. Un enfoque práctico. New York : Mac Graw Hill, 2010. 978-607-15-0314-5.

Pinto, Maria. 2015. e-COMS:Electronic Content Managment Skills. e-COMS:Electronic Content Managment Skills. [Online] diciembre 13, 2015. [Cited: noviembre 30, 2016.] http://www.mariapinto.es/e-coms/.

Platero Alcón, Alejandro. 2016. El derecho al olvido en internet. El fenómeno de los motores de búsqueda. Medellín : Sello, 2016. pp. 243-260.

Rada, R., Mili,H. , Bicknell ,E. , Blettner ,M. 1989. Development an Application of a Metric on Semantic Nets. s.l. : IEEE Transactions on Systems, Man and Cybernetics, 1989. pp. 17-30. Vol. 19.

Reynoso, Carlos; Kicillof, Nicolás. Estilos y Patrones en la Estrategia de Arquitectura de Microsoft. *Buenos Aires: Universidad de Buenos Aires*, 2004.

Resnik, P. 1993. Selection and information: a class-based approach to lexical relationships. Pensilvania : IRCS Technical Reports Series, 1993. p. 200.

Ríos, S. S. (2007). Análisis y UML. Modelo conceptual. 2007, Disponible en: http://www.uvmsf.cl/~ssanchez/images/Metodologias/Unidad4_MAD.pdf.

Rodríguez Ávila, Abel. 2007. Iniciación a la red Internet. Concepto, funcionamiento, servicios y aplicaciones de Internet. Vigo : Ideaspropias, 2007. p. 104 . 978-84-9839-139-8.

Rojas, Manuel. A semantic association page rank algorithm for web search engines, 2012. *1211.6159*.

Salton, G., McGill, M. J. 1983. Introduction to modern information retrieval. New York : Mc Graw-Hill Computer Series, 1983. 0-07-054484-0.

Sánchez, L. y Fernández, N. La web semántica: fundamentos y breve "estado del arte". *Novatica*, 2005, vol. XXXI, Nº 178, ISSN 0211-2124.

Servicios especializados de testing. [En línea] [Citado el: 11 de abril de 2016.] http://pedrosebastianmingo.com/para-que-sirven-las-pruebas-de-rendimiento-i-introduccion/.

Sommerville, I. INEGINERÍA DE SOFTWARE. Madrid : Pearson Educación S.A, 2005. 84-7829-074-5.

Schamberg, L., Einseberg, B. and Nilo. 1990. Re-examination of relevance: toward a dynamic, situational definition Information Procesing and Management. s.l. : S. A, 1990. pp. 755-775.

SparxSystems. Diagrama de Despliegue UML 2. [En línea]. Sparx Systems - Tutorial UML 2 - Diagrama de Despliegue, 2014. [Citado el: 24 de Febrero de 2015.] Disponible en: [http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html].

Tolosa, G. H., Bordignon, F. R. A. 2008. Introducción a la recuperación de información: conceptos, modelos y algoritmos básicos. Buenos Aires : UNlu, 2008.

Torres Remón, Manuel. FUNDAMENTOS DE PROGRAMACIÓN g LENGUAJES Y TÉCNICAS DE PROGRAMACIÓN. 2012.

Venegas, Rene. 2006. La similitud léxico-semántica en artículos de investigación científica en español.Una aproximación desde el Análisis Semántico Latente.60, Valparaíso : s.n., 2006, Vol. 39, pp. 75-106. 0718-0934.

Visual Paradigm. 2014. What is Visual Paradigm? [Online] 2014. [Cited: diciembre 1, 2016.] https://www.visualparadigm.com/features/.

Wanson, D. R. 1986. Subjetive versus objetive relevance in bibliografic retrieval system. s.l. : Library Quartely, 1986. pp. 389-398.

Zamora., J. 2011. Análisis de los procesos de verificación y validación en las organizaciones. [En línea] 2011. http://orff.uc3m.es/handle/10016/12880.

Bibliografía

CASTELLS, Manuel. Internet y la sociedad red. *La factoría*, 2001, vol. 14, p. 15.

HONRUBIA LÓPEZ, F. J. Introducción a las Ontologías. Escuela Universitaria Politécnica de Albacete. [Fecha de consulta: 25/05/2011]. http://www. dsi. uclm. es/asignaturas/42551/trabajosAnteriores/Trabajo-Ontologias. pdf, 2009.

JIMÉNEZ, JJ Domínguez, et al. El reto de los servicios web para el software libre. *Este libro se distribuye bajo licencia Creative Commons, España*, 2007, p. 117.

MORA, Sergio Luján. *Programación de aplicaciones web: historia, principios básicos y clientes web*. Editorial Club Universitario, 2002.

PEREIRA, Rosalba Talavera; AULAR, Yelitza Josefina Marcano. Los lenguajes de representación semántica y su uso en la construcción de ontologías. *Revista de Ciencias Sociales*, 2007, vol. 13, no 1.

VILARINO, Darnes, et al. Un modelo para detectar la similitud semántica entre textos de diferentes longitudes. *Avances en la Ingeniería del Lenguaje y del Conocimiento*, 2014, p. 57.

Anexos

Anexo 1

Anexo 2

Tabla 14: Historia de usuario # 1.

Historia de usuario	
Número: HU_1	**Nombre Historia de Usuario:** Obtener la consulta realizada por el usuario, luego de ser procesada.
Prioridad en negocio: Alta	
Descripción: El sistema obtiene la consulta una vez que ha sido procesada, de forma tal que pueda obtener la información que contiene la misma.	
Prototipo	

Tabla 15: Historia de usuario # 2.

Historia de usuario	
Número: HU_2	**Nombre Historia de Usuario:** Obtener los documentos, referidos a la consulta procesada.
Prioridad en negocio : Alta	
Descripción: El sistema debe ser capaz de obtener los documentos referidos a la consulta procesada, de forma tal que pueda realizar el subgrafo de cada documento basándose en los conceptos y relaciones de la ontología.	
Prototipo	

Anexo 3

Nombre	**Fórmulas**
Fórmula para medir exhaustividad.	$$Exhaustividad = \frac{Documentos\ relevantes\ recuperados}{Documentos\ relevantes}$$
Fórmula para medir precisión.	$$Precisión = \frac{Documentos\ relevantes\ recuperados}{Documentos\ recuperados}$$