UNIVERSIDAD NACIONAL DE COLOMBIA

SISTEMA DE GENERACIÓN, ADMINISTRACIÓN Y CONSULTA DE UNA LIBRERÍA DIGITAL DE DOCUMENTOS PARA UN PORTAL WEB

Ing. Isabel Andrea Mahecha Nieto

Universidad Nacional de Colombia

Facultad de Ingeniería

Departamento de Ingeniería de Sistemas e Industrial

Bogotá, Colombia

2011

SISTEMA DE GENERACIÓN, ADMINISTRACIÓN Y CONSULTA DE UNA LIBRERÍA DIGITAL DE DOCUMENTOS PARA UN PORTAL WEB

Ing. Isabel Andrea Mahecha Nieto

Tesis presentada como requisito parcial para optar al título de:

Magister en Ingeniería de Sistemas y Computación

Directora:

Ph.D. Elizabeth León Guzmán

Línea de Investigación:

Minería de Documentos y Librerías Digitales

Grupo de Investigación:

Grupo de Investigación en Minería de Datos - MIDAS

Universidad Nacional de Colombia

Facultad de Ingeniería

Departamento de Ingeniería de Sistemas e Industrial

Bogotá, Colombia

2011

A mis padres Carmenza y Jairo

A mis hermanas Laura y Sara

A mi abuelita Carmen

Agradecimientos

Gracias a todas las personas que estuvieron conmigo a lo largo de este camino, siempre los llevaré en mi corazón.

A mis padres, por creer en mí y por su amor infinito.

Laura y Sara Mahecha por su colaboración y su cariño fraterno.

Un especial agradecimiento a todas los integrantes del Laboratorio de Investigación en Sistemas Inteligentes – LISI, por sus enseñanzas, su compañía y su amistad.

A la profesora Elizabeth León por su orientación y conocimiento.

Agradezco a todos los profesores de la Maestría en Ingeniería de Sistemas y Computación, que dejaron en mí su conocimiento, consejo y sabiduría.

A mis amigos, por su apoyo incondicional: Alix Rojas, Ana María Rodríguez, Camilo Pino, Daniel Restrepo, David Becerra, Iván Mendoza, José Moreno, Juan Carlos Galeano, Juan Carlos Mendivelso, Mario Linares, Miguel Dussán, Wilson Soto y Yamile Jaime,

Jesús Burbano por sus valiosos aportes en lo conceptual, lo técnico y lo personal.

Juan David Cruz, gracias por tu presencia en momentos que necesité de tus palabras y de tú compañía.

To Kevin M., thank you for showing me your world and being in my way.

RESUMEN

SISTEMA DE GENERACIÓN, ADMINISTRACIÓN Y CONSULTA DE UNA LIBRERÍA DIGITAL DE DOCUMENTOS PARA UN PORTAL WEB

por

ING. ISABEL ANDREA MAHECHA NIETO

MSc. en Ingeniería de Sistemas y Computación

UNIVERSIDAD NACIONAL DE COLOMBIA

Director: Ing. Elizabeth León Guzmán. PhD.

Los portales web son grandes repositorios de información desorganizada. Muchas veces mantener una política de publicación de documentos es difícil, y finalmente cada parte de la organización publica sus documentos en lugares distintos en una estructura de carpetas particular. El acceso a los documentos se vuelve tedioso porque las personas que necesitan un documento en particular casi siempre desconocen su ubicación. Y a pesar de los hiper-vínculos entre documentos, muchas veces el texto que identifica el enlace no es suficiente para identificar el documento al que hace referencia. Por su parte, las librerías digitales se enfocan en la organización de la información a través de colecciones que permiten agrupar documentos similares lo cual agiliza el proceso de búsqueda de documentos específicos a quienes los consultan. La creación de librerías digitales se ha convertido en una opción económica y ágil, para la organización de documentos sobre diferentes temáticas, almacenados en diversos formatos (texto, audio, imagen, vídeo) en los sitios web.

En este trabajo de tesis, se propone un sistema de administración de librerías digitales web de documentos que es capaz de: 1) recuperar e indexar documentos publicados en un portal web, 2) generar una estructura de librería digital y 3) clasificar y permitir la consulta de los documentos a través de una interfaz de navegación. La recuperación e indexación se hace a través de un sistema de recuperación de información. El agrupamiento de los documentos se hace a través de un componente de minería de texto que contiene herramientas para el preprocesamiento de los documentos, y un algoritmo de agrupamiento conceptual. La interfaz de usuario se compone de dos vistas principales, la interfaz de navegación y un módulo de administración. La interfaz de navegación está compuesta por un árbol de categorías y un grafo que relaciona las diferentes categorías entre sí a través de los documentos que pertenecen a cada una de ellas. La interfaz de administración permite el acceso a los parámetros del sistema para la ejecución de los procesos de indexación y agrupamiento de los documentos.

El caso de estudio propuesto para probar el prototipo fueron los portales ubicados dentro del dominio web de la Universidad Nacional de Colombia, el cual ofrece el escenario de una organización con múltiples servidores.

Palabras Clave: Minería de documentos web, Recuperación de información, Portales web, Librerías digitales.

ABSTRACT

GENERATION SYSTEM, MANAGEMENT AND SEARCH OF A DIGITAL LIBRARY OF DOCUMENTS FOR A WEBSITE

by

ING. ISABEL ANDREA MAHECHA NIETO

MSc. en Ingeniería de Sistemas y Computación

UNIVERSIDAD NACIONAL DE COLOMBIA

Advisor: Ing. Elizabeth León Guzmán. PhD.

Web portals are great repositories of information disorganized. Many times maintain a policy of publishing documents is difficult, and eventually each part of the organization publishes its documents in different places in a particular folder structure. Access to documents becomes tedious for those who need a particular document is almost always unknown their location. And although the hyperlinks between documents, often the text that identifies the link is not sufficient to identify the referenced document. For its part, digital libraries focus on the organization of information through libraries that allow you to group similar documents which streamlines the process of searching for specific documents to those who consult them. The creation of digital libraries has become an economical and flexible for the organization of documents on various topics, stored in various formats (text, audio, image, video).

In this thesis a management system web digital library of documents is proposed. This system is able to: 1) retrieving and indexing documents on a website, 2) generating a digital library structure based on the content of the web documents and 3) classifying and allowing navigation of the web documents. The indexing and retrieval of the web documents is done through a system of information retrieval extended of the Nutch utility. For generating the structure of the digital library and future classifications, the documents are grouping through a text mining techniques, the system uses tools for preprocessing such as a Lucene, and a conceptual clustering algorithm called LINGO. For exploring and navigating the documents, the system provides a user interface that is composed by a category tree and a graph that relates the different categories with each other through the documents. Additionally, the system has and a management interface that allows setting the system parameters for the processes of indexing and clustering documents.

The system of the web digital library proposed was applied to a web site located within the National University of Colombia, which has an environment with multiple servers.

Keywords: Document web mining, Information retrieval, Websites, Digital libraries.

Contenido

Resumen ii

Abstract iii

Contenido vi

Lista de Figuras xi

Lista de Tablas xii

Lista de Algoritmos xiii

Glosario xiv

1 INTRODUCCIÓN 1
- 1.1 DESCRIPCIÓN DEL PROBLEMA . . . 2
- 1.2 OBJETIVOS . . . 2
  - 1.2.1 Objetivo General . . . 2
  - 1.2.2 Objetivos Específicos . . . 2
- 1.3 ALCANCES Y LIMITACIONES . . . 3
- 1.4 CONTRIBUCIONES . . . 3
- 1.5 PUBLICACIONES . . . 4
- 1.6 ESTRUCTURA DEL DOCUMENTO DE TESIS . . . 4

2 ESTADO DEL ARTE EN LIBRERÍAS DIGITALES DE DOCUMENTOS 5
- 2.1 LIBRERÍA DIGITAL . . . 5
  - 2.1.1 Componentes Básicos de una Librería Digital . . . 6
  - 2.1.2 Librería Digital Web . . . 7
  - 2.1.3 Sistemas de Software para Creación de Librerías Digitales Web . . . 8
- 2.2 MODELOS PARA CREACIÓN DE LIBRERÍAS DIGITALES . . . 9
  - 2.2.1 Modelos Preliminares . . . 9
  - 2.2.2 Modelo Formal: Plataforma 5S . . . 9
    - 2.2.2.1 Flujos . . . 9
    - 2.2.2.2 Estructuras . . . 10
    - 2.2.2.3 Espacios . . . 10
    - 2.2.2.4 Escenarios . . . 11
    - 2.2.2.5 Sociedades . . . 11
- 2.3 SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN WEB . . . 11
  - 2.3.1 Rastreador Web . . . 12
  - 2.3.2 Análisis de Formato . . . 13
  - 2.3.3 Índice de Documentos . . . 14

2.3.3.1 *Tries* . . . 15
2.3.3.2 Archivos de firmas . . . 15
2.3.3.3 Índice Invertido . . . 15
2.3.3.4 Representación de índices invertidos . . . 16
2.3.3.5 Modelo de espacio vectorial . . . 16
2.3.3.6 Medidas para encontrar la relación de términos y documentos . . . 17
2.3.3.7 Matriz de términos y documentos . . . 18
2.3.3.8 *Latent Semantic Indexing* . . . 18
2.3.4 Procesador de Consultas de Usuario . . . 20
2.3.5 Interfaz de Usuario . . . 22
2.3.6 Motor de Búsqueda Nutch . . . 23
2.4 PREPROCESAMIENTO AUTOMÁTICO DE DOCUMENTOS . . . 25
2.4.1 Análisis Léxico . . . 25
2.4.2 Corrección de Ortografía . . . 26
2.4.3 Eliminación de *Stop Words* . . . 26
2.4.4 Identificación de Lenguaje . . . 27
2.4.5 *Stemming* y Lematización . . . 27
2.4.6 Procesamiento de Lenguaje Natural . . . 28
2.5 AGRUPAMIENTO DE DOCUMENTOS . . . 28
2.5.1 Medidas de Similitud . . . 29
2.5.2 *Suffix Tree Clustering* . . . 31
2.5.3 *Semantic Online Hierarchical Clustering* . . . 31
2.5.3.1 Frases completas . . . 33
2.5.3.2 Conjuntos continuos . . . 33
2.5.4 LINGO . . . 33
2.5.4.1 Preprocesamiento . . . 34
2.5.4.2 Extracción de frases frecuentes . . . 35
2.5.4.3 Inducción de las etiquetas de conjunto . . . 35
2.5.4.4 Descubrimiento del contenido de los conjuntos . . . 35
2.5.4.5 Formación final de los conjuntos . . . 35

3 MODELO DE LIBRERÍA DIGITAL DE DOCUMENTOS PARA UN PORTAL WEB 36
3.1 IDENTIFICACIÓN DE NECESIDADES DE USO . . . 36
3.1.1 Sociedades . . . 36
3.1.2 Escenarios . . . 37
3.1.3 Espacios . . . 37
3.1.4 Flujos . . . 37
3.1.5 Estructuras . . . 37
3.1.6 Esquema 5S para LiD-UN . . . 38
3.2 DISEÑO DEL SISTEMA . . . 38
3.2.1 Diseño Lógico del Sistema . . . 39
3.2.1.1 Sistema de recuperación de información . . . 39
3.2.1.2 Componente de minería de texto . . . 41
3.2.1.3 Interfaz de usuario . . . 42
3.2.2 Proceso de Funcionamiento del Sistema . . . 45
3.2.3 Arquitectura Funcional del Sistema . . . 46
3.2.4 Modelo físico . . . 46
3.2.5 Escenarios de Uso . . . 48
3.2.5.1 Instalar e iniciar el sistema para primer uso . . . 48
3.2.5.2 Registrar el usuario administrador . . . 49
3.2.5.3 Autenticar usuario administrador . . . 51

3.2.5.4 Finalizar sesión de usuario administrador . . . 51
3.2.5.5 Mostrar interfaz de configuración . . . 52
3.2.5.6 Crear índice de documentos . . . 52
3.2.5.7 Actualizar índice de documentos . . . 52
3.2.5.8 Ver detalles del índice de documentos actual . . . 53
3.2.5.9 Eliminar índice de documentos . . . 53
3.2.5.10 Agrupar documentos para identificar etiquetas de los grupos de documentos . . . 54
3.2.5.11 Etiquetar documentos con la categoría a la que pertenecen . . . 54
3.2.5.12 Mostrar interfaz de consulta . . . 55
3.2.5.13 Consultar árbol de categorías . . . 55

4 APLICACIÓN, VALIDACIÓN Y ANÁLISIS DE RESULTADOS 56
4.1 EVALUACIÓN DE COMPONENTES DEL MODELO . . . 56
4.1.1 Evaluación del Sistema de Recuperación de Información . . . 56
4.1.1.1 Método de evaluación . . . 56
4.1.1.2 Parámetros de evaluación . . . 59
4.1.1.3 Resultados . . . 59
4.1.2 Evaluación del Componente de Minería de Texto . . . 61
4.1.2.1 Método de evaluación . . . 62
4.1.2.2 Parámetros de evaluación . . . 63
4.1.2.3 Resultados . . . 63
4.2 EVALUACIÓN DE USABILIDAD DE LA LIBRERÍA DIGITAL WEB LID-UN . . . 65
4.2.1 Aspectos de usabilidad evaluados . . . 65
4.2.2 Instrumento de evaluación de usabilidad . . . 66
4.2.3 Método de evaluación . . . 67
4.2.3.1 Población . . . 67
4.2.3.2 Librería digital de evaluación . . . 67
4.2.3.3 Infraestructura utilizada . . . 68
4.2.3.4 Resultados de la evaluación de usabilidad . . . 68

5 CONCLUSIONES Y TRABAJO FUTURO 75
5.1 MODELO PROPUESTO . . . 75
5.2 HALLAZGOS . . . 76
5.3 TRABAJOS FUTUROS . . . 76

Bibliografía 78

Índice alfabético 86

A MANUAL DE USUARIO 87
A.1 Requerimientos de hardware . . . 87
A.2 Requerimientos de software . . . 87
A.3 Ejecución del robot (crawler) del buscador Nutch . . . 90
A.4 Configuración del buscador Nutch . . . 94
A.5 Generación del archivo de entrada para el proceso de agrupamiento de documentos . . . 98
A.6 Ejecución del proceso de agrupamiento de documentos . . . 100
A.7 Interfaz de navegación . . . 100

B POLÍTICA DE SEGURIDAD PARA USO DE MOTORES DE BÚSQUEDA EN LOS PORTALES WEB DE LA UNIVERSIDAD NACIONAL DE COLOMBIA 104
B.1 Política de seguridad propuesta . . . 105

B.2 Estándar propuesto 105
B.3 Mejor práctica 105
B.4 Guía propuesta 105

C ARTÍCULO "AN EXPLORATORY STUDY OF OPEN SOURCE SEARCH ENGINES: EVALUATION OF SUPPORTABILITY, USABILITY, FUNCTIONALITY AND PERFORMANCE". 108

Lista de Figuras

2.1 Componentes de una Librería Digital 7
2.2 Meta-Modelo de una librería digital desarrollada con la plataforma 5S 10
2.3 Componentes de un Sistema de Recuperación de Información Web básico 12
2.4 Arquitectura básica de un rastreador web 13
2.5 Archivo de estructura invertida 16
2.6 Ejemplo de representación gráfica de tres documentos en espacio vectorial 17
2.7 Matriz de términos y documentos 19
2.8 Ejemplo de LSI - Descomposición SVD 20
2.9 Ejemplo de LSI - Aproximación Rango - 2 20
2.10 Interfaces de motores de búsqueda 22
2.11 Librería Digital Greenstone 23
2.12 Interfaz de búsqueda del motor de búsqueda Nutch 23
2.13 Arquitectura motor de búsqueda Nutch 24
2.14 Corrector de ortografía de Google 26
2.15 Ejemplo de la representación vectorial del cálculo de la similitud del coseno 30
2.16 Árbol generalizado de sufijos para tres frases [109, 119] 32

3.1 Esquema de aplicación de la plataforma 5S a la librería digital Lid-UN 38
3.2 Modelo de 4+1 vistas 39
3.3 Diseño Lógico 40
3.4 Interfaz de navegación de Lid-UN 44
3.5 Árbol de categorías de documentos 44
3.6 Grafo de categorías y documentos 45
3.7 Módulo de administración 46
3.8 Proceso de funcionamiento del sistema 47
3.9 Diagrama de arquitectura de Capas 48
3.10 Modelo Físico del sistema 49
3.11 Diagrama de Casos de Uso 50

4.1 Precisión para diferentes consultas en dos niveles 61
4.2 Precisión de Nutch por tipo de consulta 61
4.3 Tiempo de indexación 62
4.4 Tamaño del índice 62
4.5 Número de conjuntos por tamaño de conjuntos 64
4.6 Promedio de documentos por conjunto 64
4.7 Promedio de documentos por conjunto sin el conjunto “otros temas” 65
4.8 Distribución de opiniones para la característica de simplicidad: ¿La librería digital tiene opciones simples de usar? 68
4.9 Distribución de opiniones para la característica de confort: ¿Me sentí a gusto dentro de la librería digital? 69

4.10 Distribución de opiniones para la característica de facilidad de uso: ¿La librería digital es fácil de usar? . . . 69
4.11 Distribución de opiniones para la característica de **Control**: ¿Percibí que yo controlaba la utilización de la librería digital? . . . 70
4.12 Distribución de opiniones para la característica de **Legibilidad**: ¿La información que se accede desde la librería digital fue ordenada y legible? . . . 70
4.13 Distribución de opiniones para la característica de **Información adecuada**: ¿La información que se accede desde la librería digital es adecuada? . . . 71
4.14 Distribución de opiniones para la característica de **Navegabilidad**: ¿La navegación a través de la biblioteca digital se ejecuta con facilidad? . . . 71
4.15 Distribución de opiniones para la característica de **Reconocimiento**: ¿Entendí rápidamente las características y funciones disponibles en el entorno de la librería digital? . . . 72
4.16 Distribución de opiniones para la característica del **tiempo de acceso**: ¿La información que solicitó dentro de la librería digital de carga en un tiempo razonable? . . . 72
4.17 Distribución de opiniones para la característica de **Consistencia**: ¿La consistencia de los términos, palabras y acciones utilizados en todo el librería digital era evidente? . . . 73
4.18 Distribución de opiniones para la característica de **Presentación visual**: ¿El Texto en negrillas, cursivas, subrayado y estuvieron presentes en la librería digital para llamar mi atención? . . . 74
4.19 Distribución de opiniones para la característica de **Relevancia**: ¿La información que obtuve de la biblioteca digital era pertinente con mi necesidad de información? . . . 74

A.1 Cygwin . . . 87
A.2 Java JDK . . . 88
A.3 Configure Tomcat . . . 88
A.4 Iniciar Tomcat . . . 89
A.5 Tomcat . . . 89
A.6 Nutch . . . 90
A.7 Configuración carpeta de ejecución de Nutch . . . 90
A.8 Carpeta de direcciones . . . 91
A.9 Lista de direcciones . . . 91
A.10 Restricción del Dominio . . . 92
A.11 Salida comando crawl . . . 94
A.12 Aplicación web . . . 95
A.13 Administrador de Tomcat . . . 96
A.14 Aplicación web . . . 96
A.15 Configuración del site . . . 97
A.16 Ejemplo de consulta . . . 98
A.17 Configuración carpeta de ejecución de Nutch . . . 98
A.18 Ubicación de la carpeta “segments” . . . 99
A.19 Ejemplo generación de nuevo segmento con comando “Segment Merger” . . . 100
A.20 Ubicación del nuevo segmento . . . 101
A.21 Ejemplo generación de archivo . . . 101
A.22 Carpeta “Agrupamiento” . . . 102
A.23 Ejemplo interfaz de navegación . . . 103

Lista de Tablas

3.1 Comparación de algoritmos de agrupamiento de documentos de texto 43

Lista de Algoritmos

2.1 Algoritmo *Suffix Tree Clustering* 32
2.2 Algoritmo SHOC 34

Glosario

- **Afijos derivativos:** un elemento que se añade a una palabra y que tiene una función gramatical o semántica Hay distintos tipos de afijos, en el idioma español se emplean sufijos que son afijos que siguen la raíz y los prefijos que son afijos que preceden la raíz.
- **DNS** (sigla en inglés Domain Name Service, en español Servicio de Nombres de Dominio): este servicio permite traducir una dirección IP a un nombre de dominio y vice-versa. Un nombre de dominio corresponde a un nombre en palabras dado a una dirección IP, estos nombres facilita que las personas recordar la dirección de una página o documento web.
- **Colección:** Conjunto ordenado de elementos, en general de una misma clase y agrupadas por un interés particular o característica.
- **Espacio Euclidiano:** Euclides (Grecia, 300 a.C) desarrolló "la geometría plana" que trata de la geometría de objetos bidimensionales en una superficie plana. Luego desarrollaría la "geometría sólida", con que analizó la geometría de objetos tridimensionales. Todos los axiomas de Euclides fueron codificados en un espacio matemático abstracto conocido como espacio euclidiano bi o tridimensional. Estos espacios matemáticos pueden ser extendidos a cualquier dimensión, y tal espacio es llamado espacio euclidiano n-dimensional o una n-espacio.
- **Forma gramatical:** Son los diferentes tipos de palabras que al ser representados en un orden específico conforman una oración bien formada dentro de un lenguaje. Los tipos de formas gramaticales para el idioma español son ocho: nombre (e.g. gato), adjetivo (e.g. alto), verbo (e.g. nadar), adverbio (e.g. así), pronombre (e.g él), determinante (e.g. el, mi), preposición (e.g. de), conjunción (e.g. y)
- **Forma flexiva:** Formas gramaticales que se generan a partir de los cambios de morfemas flexivos añadido a la raíz de adjetivos, sustantivos, pronombres y verbos.
- **Metadatos:** Son datos estructurados sobre la información, es decir información sobre información, o datos sobre datos. Los meta-datos en el contexto de la Web, son datos que se pueden guardar, intercambiar, procesar y están estructurados para permitir la identificación, descripción clasificación y localización del contenido de un documento o recurso web y por tanto también sirven para su recuperación.
- **Morfema flexivo:** Morfema que caracteriza a la palabra los rasgos gramaticales que constituyen la información pertinente en el establecimiento de relaciones con otras palabras en la oración. En el idioma español, esta información se refiere al género, número, grado, tiempo, modo, aspecto y persona. Por ejemplo, en la palabra niñitos, los morfemas flexivos son -o (género masculino), -s (número plural).
- **Normalización de un vector**: Para un vector de un espacio vectorial con una norma en definida, la normalización es el cálculo a partir del cual se obtiene otro vector que tiene idéntica dirección y sentido que el primero y cuya norma es igual a la unidad.

- **Proceso heurístico:** Los procesos heurístico son métodos empíricos para la resolución de problemas que se basa en el aprendizaje por descubrimiento. Este concepto nace en el campo de la matemática y se compone de cuatro fases: 1. Comprender el problema; 2. Proponer un plan para resolver el problema; 3. Ejecutar el plan y 4. Verificar los resultados.
- **Software de código abierto:** este tipo de software es aquel cuyo código fuente es de libre acceso y está públicamente disponible en medios como la Web. Los acuerdos de licencia específicos varían en lo que se puede hacer con ese código (e.g. GNU General Public License, GNU Lesser General Public License, BSD License, Mozilla Public License, MIT License, the Apache License).
- ***Stop words:*** Término en idioma inglés usado para identificar las palabras que son utilizadas con mucha frecuencia. En español a veces se traduce como palabras de parada, sin embargo no hay una traducción oficial.
- **URL:** Localizador uniforme de recursos (en inglés Uniform Resource Locator), es la dirección única de documentos y de otros recursos publicados en la Web.
- **Web graph*:*** es el gráfico que incluye las páginas web que son representadas por los nodos y los hiper-vínculos representados por las conexiones entre los nodos.
- ***World Wide Web*** (WWW): también conocida como "la Web", es un subconjunto de Internet que consiste en páginas a las que se puede acceder usando un navegador. Internet es la red de redes donde reside toda la información. Por ejemplo, el correo electrónico, como los FTP, juegos, etc. son parte de Internet, pero no de la Web. La web es un sistema de hipertexto que utiliza Internet como su mecanismo de comunicación, es decir es una forma gráfica de explorar Internet.

1 INTRODUCCIÓN

Con el crecimiento de la Web y el gran aumento en la publicación de documentos en portales web, de diferente índole en todo tipo de formato textual, el interés por la búsqueda de información ha cobrado cada vez más importancia dentro de los usuarios que consultan dichos portales [60]. En el caso particular de los portales web de instituciones educativas como el de una universidad, el volumen de documentos que se publican a diario es considerable, debido a la diversidad de actividades que se desarrollan dentro de la misma, tanto en la administración como en el desarrollo académico. Los documentos que son publicados se refieren a: la normatividad (resoluciones, actas, comunicados, etc.), las noticias del día y trabajos referentes a actividades académicas (material de cursos, tesis, artículos, informes, etc.). Los usuarios de dichos documentos son todas las personas que pertenecen a la institución, las cuales tienen diferentes roles: profesores, estudiantes, personal administrativo y también personas que buscan información a través del sitio web público.

En general los portales web son grandes repositorios de información desorganizada. Muchas veces mantener una política de publicación de documentos es difícil, y finalmente cada parte de la organización publica sus documentos en lugares distintos en una estructura de carpetas particular. El acceso a los documentos se vuelve tedioso porque las personas que necesitan un documento en particular casi siempre desconocen su ubicación. Y a pesar de los hiper-vínculos entre documentos, muchas veces el texto que identifica el enlace no es suficiente para identificar el documento al que hace referencia. En este contexto, los buscadores web permiten descargar parte del contenido de dichos portales con el fin de facilitar el acceso a los documentos a partir de su recuperación y la creación de un índice con el contenido de los mismos. Además, estos sistemas utilizan cálculos de relevancia para cada página Web lo cual permite conocer su relevancia en una determinada consulta de usuario a partir de diferentes criterios: autor, número de páginas que la tienen enlazada, etc. Dentro de los buscadores web comerciales los cinco más usados son [73]: Google Search, Yahoo Search, MSN/Windows Live/Bing Search, AOL Search, y Ask.Com Search.

Como alternativa de los buscadores web comerciales están los buscadores web de código abierto[1], cuya oferta asciende a 50 buscadores publicados en la Web. Debido a los elevados costos y el enfoque a gran escala de los buscadores comerciales, la mayoría de los sitios web no tienen acceso a este tipo de herramientas, por lo cual los buscadores web de código abierto son opciones favorables porque ofrecen ventajas similares a las que tienen los buscadores web comerciales [69]. La adaptación de motores de búsqueda a las condiciones particulares de los portales web, es un primer acercamiento hacia la construcción de sistemas más robustos que permitan no solo la recuperación, indexación y consulta de documentos, sino también la clasificación y administración de la publicación de documentos en línea.

Por su parte, las librerías digitales se enfocan en la organización de la información a través de colecciones que permiten agrupar documentos similares lo cual agiliza el proceso de búsqueda de documentos específicos a quienes los consultan [110]. La creación de librerías digitales se ha convertido en una opción económica y ágil, para la organización de documentos sobre diferentes temáticas, almacenados en diversos formatos (texto, audio, imagen, vídeo) en los sitios web [66]. Para aquellos que administran

[1] En el portal web de "Search Tools for Web Sites and Intranets", está publicado un listado de los buscadores disponibles tanto comerciales como de código abierto. La dirección en linea es: `http://www.searchtools.com/tools/tools.html`

sitios web, las librerías digitales representan una nueva forma de llevar a cabo las funciones de administración de documentos, que incluye la gestión de los diferentes tipos de recursos de información, tales como nuevos métodos de almacenamiento y preservación; nuevos criterios para la catalogación y clasificación, y los nuevos modos de interacción con y para los usuarios [25].

Además las librerías digitales proponen el uso de una interfaz de usuario que integra diferentes opciones de búsqueda dentro de textos completos, frases, una estructura de navegación jerárquica ó un collage de visualización dinámica de imágenes, que incluye enlaces a documentos pertinentes [116]. A diferencia de una página web que está enlazada entre sí porque se insertan manualmente hiper-vínculos, los nuevos documentos de una Librería Digital se pueden añadir con sus meta-datos e integrar de forma automática a una colección para ser consultados. Una comunidad puede utilizar una librería digital para crear un repositorio de información que permita a los usuarios cargar nuevos documentos [110, 112], que sean fácilmente consultados a través de una estructura de documentos digitales indexados que a su vez están están integrados con un buscador, de tal manera que permita reducir el espacio de búsqueda por tema y/o tipo de documento dentro de publicaciones de subdivisiones de una institución [78, 115].

A continuación se presenta la justificación de este proyecto con sus objetivos, alcances y limitaciones.

1.1 DESCRIPCIÓN DEL PROBLEMA

En la actualidad, la mayoría de portales web tiene publicados una gran cantidad de documentos textuales que se refieren a múltiples temáticas, estos documentos se encuentran en diferente formato (html, pdf, etc.). El acceso a documentos específicos, en dichos portales, cada vez es más difícil por dos razones simples: 1) su estructura que no es la más adecuada para agrupar los diferentes documentos [17] y 2) los buscadores web que se utilizan para consultar dichos portales no ofrecen al usuario las opciones suficientes para efectuar búsquedas precisas y eficientes de acuerdo a necesidades específicas de información de los usuarios [60].

Diferentes trabajos de investigación han planteado alternativas para generar librerías digitales, que utilizan estrategias de organización de documentos y adición de meta-datos [114, 111, 17], como también la implementación de técnicas de recuperación de información [105], agrupamiento de documentos [30, 105], taxonomías [95] y ontologías [11]. En cuanto a los buscadores aplicados a librerías digitales se han propuesto diferentes estrategias como la extracción e indexación de frases [38, 39] y búsqueda de palabras clave [23, 84].

En este trabajo de tesis, se propone un sistema de administración de librerías digitales web de documentos que es capaz de: 1) indexar documentos publicados en un portal web, 2) generar una estructura de Librería Digital y 3) clasificar y permitir la consulta de los documentos en la librería digital propuesta.

1.2 OBJETIVOS

1.2.1 Objetivo General

Diseñar un sistema para generar, administrar y consultar librerías digitales de documentos publicados en un portal web.

1.2.2 Objetivos Específicos

1. Diseñar un modelo de librería digital que permita recuperar e indexar los documentos del portal web de una organización.

2. Recuperar e indexar los documentos de un portal web de una organización, utilizando un buscador de código abierto.

3. Proponer e implementar un modelo de clasificación de documentos digitales usando una técnica de agrupación de documentos.

4. Diseñar un módulo de administración de documentos en la estructura de documentos digitales.

5. Adaptar un buscador de código abierto, que permita realizar búsquedas de documentos en el modelo de librería digital definido.

6. Validar el sistema de librería digital, con los documentos que se encuentran publicados en el portal web de la Facultad de Ingeniería de la Universidad Nacional de Colombia.

1.3 ALCANCES Y LIMITACIONES

Algunos aspectos que restringen el alcance de los resultados de este proyecto son:

- Se utilizarán herramientas de código abierto, con el objetivo de aprovechar las ventajas de este tipo de software para adaptarse a las necesidades de funcionalidad del proyecto propuesto.
- El alcance del modelo de librería digital se desarrollará como un prototipo que integra el uso de varios prototipos, siguiendo la lógica funcional del modelo.
- El desarrollo de la herramienta que cumpla con los requerimientos del sistema completo no se contempla como objetivo del proyecto, sin embargo el diseño propuesto se validará con los prototipos para un desarrollo de software futuro.

1.4 CONTRIBUCIONES

Los aportes que tiene este trabajo son:

1. Diseño de una librería digital que integra una técnica de recuperación de información y una técnica de minería de texto, el cual se describe en la sección 3; el diseño lógico del modelo se muestra en la sub-sección 3.2.1 y en la sub-sección 3.2.3 se plantea la arquitectura del modelo.

2. Método de selección de un buscador de código abierto para recuperar e indexar documentos de un portal web de una organización. La descripción del proceso se describe en la Sección 4.1.1 y fue publicado en [63].

3. Adaptación de un algoritmo de agrupamiento de documentos para la generación de la estructura de la librería digital propuesta. En la sección 3.2 se describe como se integró el algoritmo de agrupamiento de documentos dentro de la librería digital. En la Sección 4.1.2 se describe la validación de la implementación del algoritmo. El diseño de la generación de la estructura de la librería digital y la validación se publicó en [62].

4. Aplicación de un método de validación de usabilidad del sistema de librería digital propuesto. En la sección 4.2se describen la validación desarrollada con los documentos que se encuentran publicados en el portal web del Departamento de Ingeniería de Sistemas e Industrial de la Universidad Nacional de Colombia.

1.5 PUBLICACIONES

Las publicaciones obtenidas durante el desarrollo de esta tesis son:

- Mahecha-Nieto, Isabel A., León Guzmán, Elizabeth. "Buscadores Web Open Source". Tendencias en Ingeniería de Software e Inteligencia Artificial. Volumen 2. Medellín, Colombia. Julio de 2008. ISBN: 978-958-44-3666-5. pp 147-152
- Mahecha-Nieto, Isabel A., León Guzmán, Elizabeth. "An exploratory study of open source search engines: Evaluation of supportability, usability, functionality and performance". Quinto Congreso Colombiano de Computación 2010. Cartagena, Colombia. Abril de 2010. ISBN: 978-958-8387-40-6.
- Mahecha-Nieto, Isabel A., León Guzmán, Elizabeth. "Digital Web Library of a Website With Document Clustering". Proceedings of the 12th Ibero-American Conference on AI, IBERAMIA 2010. Noviembre de 2010. Bahía Blanca, Argentina. Lecture Notes in Artificial Intelligence Series, Volumen 6433. pp. 214-223. Springer.

1.6 ESTRUCTURA DEL DOCUMENTO DE TESIS

Este documento de tesis se organiza de la siguiente forma:

- En el capítulo 2 se describe el estado del arte en el área de las Librerías Digitales de Documentos. Se presentan las diferentes definiciones que se han hecho del término Librería Digital y se enumeran algunos de los sistemas de Librerías Digitales disponibles actualmente. Además, se describen los principales componentes de este tipo de sistemas y las técnicas que se han utilizado para el tratamiento de procesamiento automático de documentos.
- En el capítulo 3 se presenta el modelo propuesto a partir de los conceptos utilizados para su desarrollo. Se describen los componentes involucrados en los procesos de: Recuperación de Información, Preprocesamiento Automático de documentos, Agrupamiento de Documentos y Visualización de Conjuntos de Documentos.
- En el capítulo 4, se describen los resultados obtenidos usando el sistema propuesto en un portal de una organización académica. La evaluación del sistema se desarrollo en dos fases, la primera incluyó la evaluación del sistema de recuperación de información y el componente de minería de texto por separado. La segunda fase consistió en la evaluación del sistema a través de una prueba de usabilidad.
- Finalmente, en el capítulo 5 se describen las conclusiones obtenidas del trabajo desarrollado y se muestran futuras perspectivas para nuevos trabajos.

2 ESTADO DEL ARTE EN LIBRERÍAS DIGITALES DE DOCUMENTOS

En este capítulo se muestra una visión general de las propuestas tecnológicas que han surgido en el campo de las Librerías Digitales, en las que se incluyen técnicas de Recuperación de Información y Minería de Texto. En los procesos de desarrollo de sistemas de librerías digitales, se han utilizado diferentes clases de tecnologías para la recopilación de documentos como lo son los componentes de recuperación de información para el rastreo e indexación de documentos y también interfaces para procesar búsquedas de usuarios. También se utilizan técnicas de preprocesamiento de documentos para la reducción de contenido como el análisis léxico, la eliminación de *stop words*, la lematización y *stemming*, corrección de ortografía, identificación de lenguaje y procesamiento de lenguaje natural. Para la creación de colecciones de documentos dentro de librerías digitales se usan algoritmos de minería de texto para agrupamiento y clasificación de documentos.

2.1 LIBRERÍA DIGITAL

La definición de librería digital ha sido propuesta desde diferentes puntos de vista como el histórico y el tecnológico, también desde diferentes perspectivas al interior de distintas comunidades de investigación en áreas como las Ciencias de la información, Recuperación de Información e Interacción Hombre-Computador y hasta el momento no se ha llegado a un consenso definitivo [32]. Licklider en [58] define librería digital como una colección de versiones digitales de contenidos disponibles a través de equipos de cómputo interconectados. Levy y Marshall en [57] proponen la librería digital como la integración de documentos, tecnología y trabajo. Lesk en [56] analiza los esfuerzos que se han hecho por entender las palabras librería y digital y concluye que dichos esfuerzos están divididos entre la comprensión de las necesidades del usuario y el uso de los recursos que las librerías digitales proveen.

Otra definición fue propuesta en el estudio Delphi [51] sobre librerías digitales, el cual concluye con una definición amplia: "colección de recursos organizados, mecanismos de exploración y búsqueda, entornos de red distribuidos, y conjuntos de servicios creados para identificar las necesidades del usuario".

El panel en librerías digitales *President's Information Technology Advisory Committee* (PITAC)[1], define librería digital como un conjunto de colecciones en red de objetos digitales como textos, documentos, imágenes, sonidos, datos científicos y software; además estos sistemas se convierten en el núcleo de la Internet de hoy y los repositorios universales del mañana de acceso digital al conocimiento de la humanidad [85].

Los sistemas de librerías digitales tienen componentes interdisciplinarios como: el hipertexto, la recuperación de información, minería de texto, los servicios multimedia, la administración de bases de datos y la interacción hombre computador [25]. El enfoque actual en el desarrollo de este tipo de sistemas, busca permitir la integración de todas estas perspectivas en sistemas que sean construidos

[1] *President's Information Technology Advisory Committee* (PITAC) `http://archive.cra.org/Policy/pitac.html`

con arquitecturas capaces de ínter-operar con otros sistemas y su adaptación a los cambios requeridos con el paso del tiempo.

Desde el punto de vista de la recuperación de información, una librería digital se define como un sistema extendido [5], que contiene colecciones de documentos almacenados en formatos digitales que residen en bases de datos, bases de conocimiento, bases de texto, o la World Wide Web (WWW) [25] y dentro de sus funciones básicas están el soporte a grandes colecciones de documentos, las búsquedas, la catalogación y la indexación [4].

En el contexto de las librerías y bibliotecas tradicionales, las librerías digitales permiten el control de las bibliotecas físicas, a través de una infraestructura tecnológica, que incluye equipos de cómputo, software especializado para digitalización de documentos y adquisición de servicios electrónicos como almacenamiento de contenidos en medios magnéticos (CD-ROM, Discos Duros, etc.), implementación de catálogos en línea de acceso público y bases de datos en línea [25].

Para los administradores de bibliotecas físicas y portales web, las librerías digitales representan una forma novedosa de ejecutar las funciones de administración de documentos, lo cual incluye la gestión de diferentes tipos de recursos de información tales como: nuevos enfoques de la adquisición (sobre todo con más intercambio y servicios de suscripción); nuevos métodos de almacenamiento y preservación; nuevos enfoques para clasificación y catalogación; y nuevos modos de interacción con y para los usuarios [25].

La dimensión social de las librerías digitales es un aspecto que se ha ido incluyendo dentro de las investigaciones en el área, porque las librerías digitales se convierten en construcciones colectivas –coleccionadas y organizadas– por una comunidad de usuarios. Las funciones de dichas librerías deben proveer el soporte adecuado a las necesidades de información de los usuarios, a través de la integración de una variedad de instituciones de información en las que las diferentes tipos de fuentes de información son seleccionados, coleccionados, organizados, preservados y están dispuestos para su consulta [5].

2.1.1 Componentes Básicos de una Librería Digital

En la figura 2.1 se presentan los componentes básicos de una librería digital, dentro de los que se incluyen: repositorio de documentos, componentes de indexación, clasificación, búsqueda, y una interfaz de usuario. Estos componentes son solo los que en esencia debe tener una librería digital, sin embargo pueden incluirse otros componentes e implementarse en diversas arquitecturas de acuerdo a los requerimientos lógicos y físicos del sistema.

La arquitectura básica de una librería digital incluye los siguientes elementos [113, 79]:

- Repositorio de documentos: este componente almacena todos los objetos digitales que contienen datos e información. Estos objetos pueden ser documentos en diferentes formatos como textos, imágenes y vídeos, y también bases de datos.
- Componente de indexación: el objetivo de este componente es construir un índice con el contenido de los documentos [110]. Un índice permite la ejecución de búsquedas alfabéticas rápidas de cada una de las palabras de búsqueda. En promedio, para encontrar un elemento en un índice de tamaño N toma Log2N pasos, lo cual es mucho más rápido que leer un archivo completo [56].
- Componente de clasificación: es el componente que se usa para identificar y clasificar las colecciones de documentos con contenidos comunes [113]. La clasificación obtenida permite que los usuarios puedan navegar a través de las diferentes colecciones de una librería digital y así puedan encontrar rápidamente documentos relacionados entre sí [113]. Los procesos de clasificación que se han implementado hasta el momento incluyen procesos de clasificación manuales y asistidos desarrollados por expertos o personas que tienen conocimiento amplio en un conjunto de datos particular. También se han implementado procesos de clasificación semi-automática utilizando algoritmos de minería de texto y ontologías.

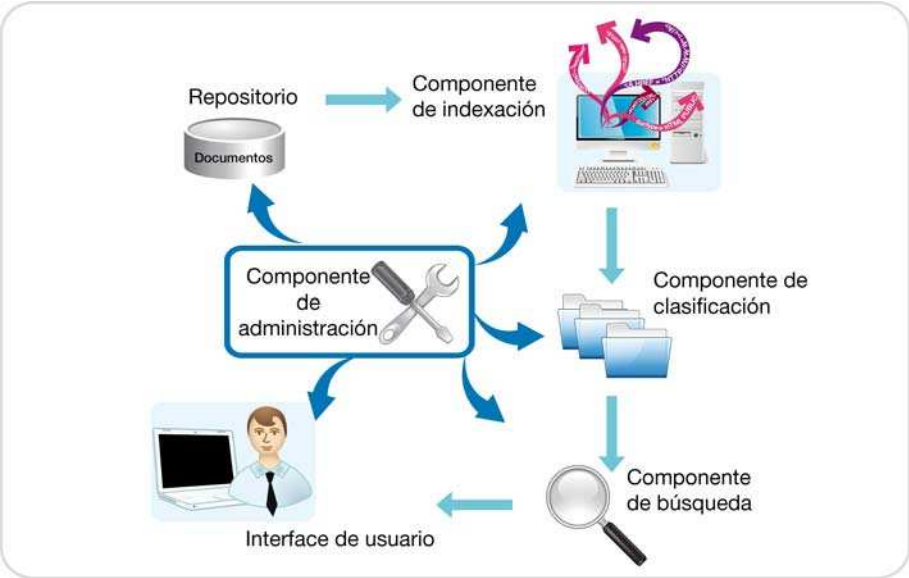


Figura 2.1: Componentes de una Librería Digital

- Componente de búsqueda: este componente permite hallar documentos con información específica de una consulta de información enunciada por un usuario [4, 19] de la librería digital. Para el procesamiento de la consulta se utilizan diferentes estrategias probadas en diferentes sistemas de recuperación de información y motores de búsqueda entre las cuales se destacan las que usan: Cálculos de ponderación de términos para modelo vectorial, análisis local automático con técnicas de agrupamiento de documentos, análisis global automático con el uso de tesauros y cálculos de similitud [4].

- Interfaz de usuario: en general se incluyen dos tipos de interfaz de usuario dentro de las librerías digitales, la interfaz de usuario final y la interfaz de administración. La primera es usada para mostrar la forma como funciona la librería digital, su organización y las funciones para efectuar búsquedas de información dentro de las colecciones de documentos. La interfaz de administración tiene como objetivo el acceso a las funciones de instalación, configuración y mantenimiento de los diferentes componentes incluidos en una librería digital.

- Componente de administración: este componente incluye una vista de los diferentes parámetros del sistema que deben ser configurados para el correcto funcionamiento del sistema, así como las funciones necesarias para actualizar el repositorio de documentos y la construcción de índices y colecciones con dichos documentos.

2.1.2 Librería Digital Web

Las librerías digitales web tienen tienen como objetivo la recuperación y clasificación de conjuntos de documentos publicados dentro de portales tanto en la Web como en una red Intranet. La creación de este tipo de librería digital de documentos, se hace a partir de las páginas web y documentos electrónicos de diferente formato (texto plano, .doc, .pdf, hojas electrónicas, etc.) los cuales son normalizados y transformados a un formato estándar, para luego ser almacenados y clasificados [17].

El uso de protocolos estándar y tecnologías para archivo de documentos permiten la organización y acceso adecuado a librería digital al facilitar la incorporación de herramientas de extracción de datos [17], que son utilizadas para agregar meta-datos a los documentos almacenados en librería digital.

La ventaja de utilizar dichos meta-datos, es la posibilidad de efectuar búsquedas más precisas de documentos dentro de las librerías digitales a partir de la información que almacenan [17, 112].

2.1.3 Sistemas de Software para Creación de Librerías Digitales Web

Los sistemas de software para la creación de librerías digitales web se han desarrollado durante los últimos veinte años y su enfoque ha sido la administración de repositorios que satisfacen los requerimientos de usuarios y organizaciones. Estos sistemas usualmente tienen funciones de recuperación de información [64] y sus colecciones de documentos son construidas con el conocimiento de un experto quien selecciona el nombre para cada colección y luego clasifica los documentos de uno o varios repositorios. Sin embargo, a pesar de la intensa actividad de investigación en el campo de las librerías digitales aún se encuentran pocas soluciones web para la creación y mantenimiento de dichas librerías [113] y para conjuntos de documentos no-estructurados o semi-estructurados las opciones son bastante limitadas.

Para la implementación de este tipo de soluciones se debe considerar características básicas de un sitio web para librerías digitales, como la interfaz de usuario que debe ser atractiva visualmente, ergonómica y fácil de usar; además de incorporar funciones de búsqueda de gran alcance con opciones de uso sencillas para el usuario; en cuanto al su contenido, ante todo debe ser fácil de mantener y actualizar, lo cual debido a su complejidad, se ha abordado utilizando técnicas de Recuperación de Información y estándares de meta-datos [113, 17]. Algunos de los sistemas de software para librerías digitales web más famosos son: Fedora [55], DSpace [100], EPrints [37], OpenDLib [18] y Greenstone [113].

Fedora es un sistema enfocado en la administración de repositorios de objetos complejos a través de un conjunto de servicios web para la búsqueda y el acceso a diferentes vistas de dichos objetos [55, 80]. Actualmente, Fedora Commons Repository Software[2] es una plataforma robusta que incluye las siguientes características: almacena todo tipo de contenidos y sus meta-datos, puede administrar y mantener contenido digital de cualquier tipo, puede ser escalado a millones de objetos, se puede tener acceso a los datos vía API Web, tiene una utilidad de recuperación en caso de desastres y migración de datos, el repositorio total puede ser reconstruido desde objetos digitales y contenidos de los archivos, y finalmente, tiene un servicio para búsqueda en texto e interfaces para múltiples usuarios[3].

Greenstone es un sistema de software para construir y acceder colecciones de contenido digital almacenado en diferentes medios como páginas web y dispositivos de almacenamiento removibles [113]. Esta plataforma de investigación incluye diferentes áreas como: minería de texto dinámico [114], visualización de consultas [88], servicios de alertas [15], búsqueda espacial [49] y ontologías [14].

Otras aproximaciones para la clasificación y agrupamiento de documentos están basados en modelos de minería de datos como lo son MARIAN [30], BT Digital Library [11], Somelib [84] y Hypia System [24]. Por otro lado, la investigación en Web Semántica ha propuesto otro sistemas soportados con ontologías, meta-datos y taxonomías, como el sistema propuesto en [95] que describe una metodología para construir taxonomías automáticamente dentro de una librería digital a través de un ordenamiento previa la generación de la librería.

Todos estas aproximaciones han tratado de resolver el problema de la clasificación de documentos, pero la principal dificultad s que los buenos procesos de clasificación no son del todo automáticos y necesitan de un experto que refine las etiquetas de los conjuntos de documentos y las relaciones entre ellos. En general estos sistemas usan contenidos estructurados como publicaciones científicas y por tanto se requieren nuevas aproximaciones para el manejo de otro tipo de documentos semi-estructurados y no-estructurados como los que se publican portales web.

[2] Fedora.org `http://www.fedora-commons.org/`
[3] Fedora features `http://www.fedora-commons.org/about/features/`

2.2 MODELOS PARA CREACIÓN DE LIBRERÍAS DIGITALES

Para especificar y obtener un entendimiento de las características, estructura y el comportamiento de sistemas de información son esenciales los modelos formales y las teorías. Los modelos formales tienen como objetivo la abstracción de las características tanto generales como las que tienen en común un conjunto de sistemas desarrollados para problemas semejantes, y permiten explicar sus estructuras y procesos [33].

En general, los modelos formales para sistemas de información son usados como herramientas para el diseño de sistemas reales ya que permiten el desarrollo de una especificación precisa de los requerimientos contra los cuales la implementación final del sistema puede ser probada y verificada. La mayoría de los sistemas de información actuales tienen algún modelo formal subyacente [33].

2.2.1 Modelos Preliminares

Para el caso de los sistemas de librerías digitales, los modelos propuestos en la literatura son bastante escasos debido en gran medida a la complejidad de este campo. Un primer intento de proponer un modelo de librería digital es propuesto por Wang en [107]. Este modelo describe una "aproximación híbrida" en la que integra una base de datos con objetivos especiales y una interfaz de usuario multimedia, además el modelo es formalizado en términos del lenguaje formal Z [96].

Otro modelo es propuesto por Kalinichenko *et. al.* en [50], el cual integra un modelo canónico para sistemas de información y una aproximación de composición, este sistema es implementado para proveer una solución parcial para la interoperatibilidad de librerías digitales.

2.2.2 Modelo Formal: Plataforma 5S

Los modelos de Wang [107] y Kalinichenko *et. al.* [50] son la base para formalizar una librería digital propuesto en la plataforma 5S propuesta por Gonzalves en[32], la cual es el primer modelo formal de librería digital que incluye todo un marco de referencia para el desarrollo de este tipo de sistemas. La plataforma 5S es la integración de cinco conceptos: Flujos, Estructuras, Espacios, Escenarios y Sociedades (en inglés, *Streams, Structures, Spaces, Scenarios and Societies*). La plataforma 5S permite general una base formal para entender la complejidad de las librerías digitales a través del entendimiento de los diferentes elementos que intervienen en estas (flujos, estructuras, espacios, escenarios y sociedades) y el diseño de los componentes necesarios para responder a las expectativas de uso de esta clase de sistemas.

La plataforma 5S fue probada a través de la implementación de un sistema generador de librerías llamado *"The 5SLGEN Digital Library Generator"* [32], en el cual se desarrolla un meta-modelo de una librería digital compuesta por servicios básicos: servicio de manejo de colecciones de documentos, servicio de hipertextos, servicio de indexación y un servicio de clasificación. En la figura 2.2 se describe la relación entre los conceptos de la plataforma 5S y el diseño de los servicios de un sistema de librería digital.

A continuación se describen los conceptos básico de la plataforma 5S propuesta en [32].

2.2.2.1 Flujos

Los flujos son secuencias de elementos de un tipo arbitrario (bits, caracteres, imágenes, etc). Pueden modelar tanto contenido estático (e.g. textos) como dinámico (video digital, dato de secuencia de tiempo y posición de un objeto en movimiento). Los flujos dinámicos permiten representar la comunicación que existe dentro de una librería digital. Estos flujos dinámicos son entendidos a través de su naturaleza temporal. Por su parte, los flujos estáticos corresponden a algún tipo de contenido

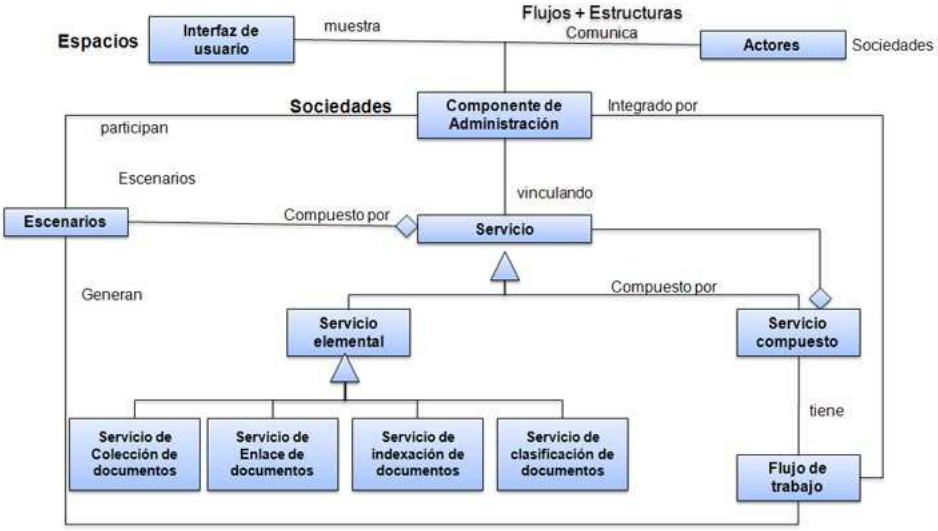


Figura 2.2: Meta-Modelo de una librería digital desarrollada con la plataforma 5S

informativo que se interpreta como una secuencia de elementos básicos, casi siempre de igual tipo. El flujo estático más usado es el texto (secuencia de caracteres). El tipo de flujo determina su semántica y área de aplicación: un texto, un conjunto de textos, como artículos científicos o libros se pueden considerar como flujos estructurados [32].

2.2.2.2 Estructuras

Las estructuras especifican la manera como cada parte de un todo está ordenado u organizado. En el caso de las librerías digitales, estas estructuras representan hipertextos, taxonomías, conexiones entre sistemas, relaciones entre usuarios y contenidos. La estructura permite orientar al lector dentro de la información de un documento, por ejemplo en el caso de los libros estos están estructurados lógicamente en capítulos, secciones, sub-secciones, párrafos [27]. Los lenguajes de marcado (markup languages, SGML, XML, HTML) son la forma primaria para presentar la estructura de los documentos digitales para propósitos de recuperación y presentación [28, 20, 29]. Las bases de datos relacionales y orientadas son estructuras en las cuales los datos son representados a través de tablas y grafos como unidades estructurales [25].

2.2.2.3 Espacios

Los espacios son el conjunto de objetos y las operaciones sobre dichos objetos que obedecen a ciertas condiciones. En este sentido, las operaciones y sus condiciones con un espacio definido definen sus propiedades. Uno de los autores que incluye el concepto de espacios dentro de la definición de librería digital es Licklider en [58], quien habla de espacios para información. También, los espacios pueden ser definidos por un lenguaje regular aplicado a una colección de documentos. Las librerías digitales pueden utilizar muchos tipos de espacios para indexación, visualización y clasificación de documentos, entre otros servicios. Los espacios más utilizados dentro de las librerías digitales son: espacios de medición, espacios de probabilidad, espacios de vectores y espacios topológicos [32].

Para Gonzalves [32], el conocimiento de la humanidad puede ser plasmado en usando espacios conceptuales debido a que existen medios multimedia que brindan la posibilidad de representar sistemas

reales de forma simulada en un ambiente virtual en una o varias dimensiones limitados por espacios métricos o de presentación como lo son las ventanas, vistas y proyecciones [71, 123].

2.2.2.4 Escenarios

Los escenarios son descripciones de las posibles formas de uso de un sistema que cumple con alguna función que el usuario desea. Estos escenarios hacen parte del proceso de diseño de los sistemas de información [54]. Los escenarios se convierten en herramientas que facilitan a quienes desarrollan librerías digitales el entendimiento de la complejidad de dichos sistemas porque describen lo que sucede con los flujos, en los espacios a través de las estructuras. En conjunto, los escenarios describen los servicios, actividades, tareas, el funcionamiento general [32] de los sistemas y como estos afectan las organizaciones y las sociedades.

Para entender los escenarios es necesario comprender los conceptos de estado y evento. Un estado está determinado por el contenido en una ubicación específica, los cuales son definidos dentro de la particularidad de cada sistema. En consecuencia de esto, un evento denota una transición o cambio entre estados [32]. Entendiendo el concepto de evento se desarrollan los escenarios, los cuales especifican secuencias de eventos, que involucran acciones y que modifican estados en el cómputo de e influencia en la ocurrencia y resultado de eventos futuros [32].

2.2.2.5 Sociedades

Las sociedades son el conjunto de entidades y sus relaciones entre sí. Estas entidades incluyen personas como también los componentes de hardware y software presentes en los sistemas de librerías digitales[32]. Las sociedades de personas pueden ser catalogadas de acuerdo a los roles presentes en una organización (e.g. clientes, autores, editores, desarrolladores). Durante el desarrollo de las actividades ejecutadas por los miembros de las sociedades crean frecuentemente artefactos de información que pueden ser gestionados en la librería digital [32].

Las sociedades son holísticas y como tales deben ser vistas como un todo, no sólo como la suma de sus partes y las relaciones entre estas. Por tanto, las sociedades son el nivel más elevado de los componentes de la librería digital, que existe para proveer las necesidades de información de las sociedades y describe el contexto de su uso. Las librerías digitales básicamente son herramientas para recolectar, preservar y compartir artefactos de información entre los miembros de las sociedades [32].

2.3 SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN WEB

La Recuperación de Información, entendida como un campo de estudio se define como la búsqueda de contenidos (e.g. texto, imágenes, multimedia) de una naturaleza no estructurada que satisface una necesidad de información dentro de grandes colecciones de documentos digitales almacenados en diferentes formatos [64]. Los sistemas de recuperación de información de acuerdo al contexto en el cual se implementan pueden ser clasificados como locales para el caso de una sola terminal y web cuando los repositorios de contenidos se encuentran publicados en-línea [4].

La recuperación de información enfocada en las librerías digitales web tiene como objetivo básico la recolección e indexación de documentos de uno o varios repositorios web de documentos digitales y la disposición de una interfaz de usuario para permitir la búsqueda de información dentro de una estructura de colecciones [53]. En la figura 2.3 se muestran los componentes básicos de un sistema de recuperación de información los cuales se describen a continuación.

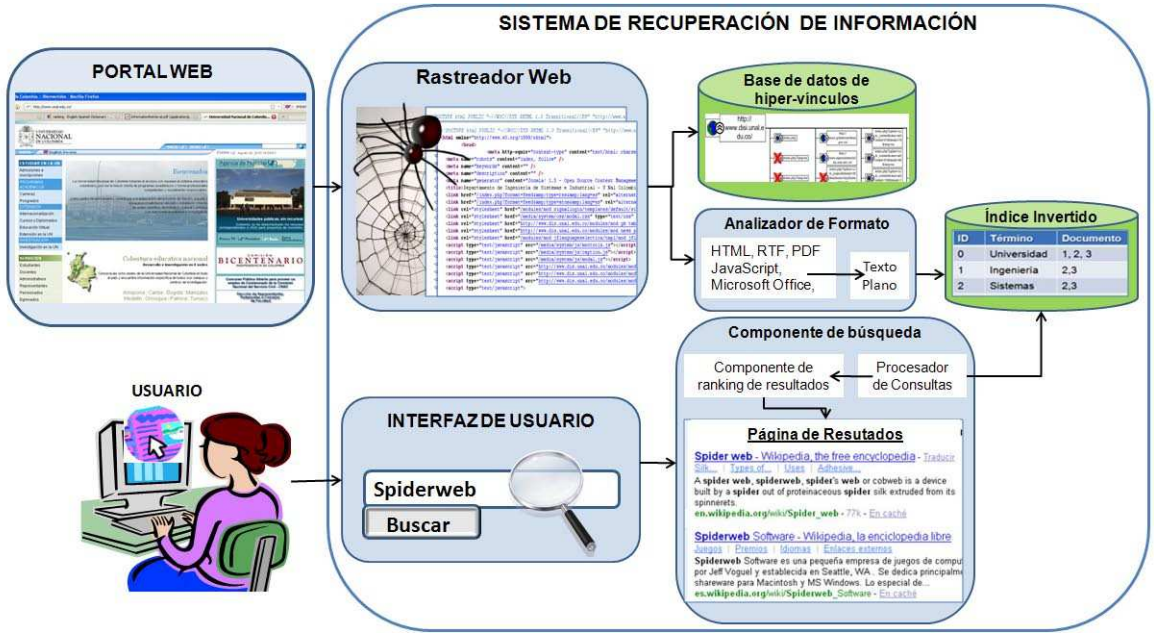


Figura 2.3: Componentes de un Sistema de Recuperación de Información Web básico

2.3.1 Rastreador Web

Para que un sistema de recuperación de información pueda extraer el contenido de un conjunto de páginas web primero se requiere conocer el listado de las URL de los documentos que serán almacenados [19]. Para recolectar las URL se debe navegar dentro de la Web o una Intranet, a partir de un conjunto de URL inicial y hacia los que enlaces que esta contenga. El componente que se encarga de este proceso es el rastreador web (*crawler* en inglés), el cual se encarga del almacenamiento de las nuevas URL en una base de datos para su posterior rastreo. Luego que una documento web es rastreado, se envía a otro componente analizador de formato para la extracción de contenido de texto y metadatos que serán almacenados en el índice[19].

La complejidad de un rastreador web depende del alcance que tenga el sistema. Para el caso de los motores de búsqueda disponibles en la Web cuyo enfoque es el rastreo constante de grandes porciones del contenido público de la Web [19, 64], los rastreadores deben ser componentes robustos y cuyo funcionamiento debe ser de forma distribuida como los usados por Alta Vista [42] y Google [13]. Para sistemas enfocados en porciones pequeñas de la Web como portales o repositorios de contenido en-línea, existen rastreadores disponibles en la Web, los cuales pueden ser adaptados a sus características como el utilizado por el motor de búsqueda de código abierto llamado Nutch [16] y los rastreadores Heritrix [70] y Ubicrawler [12].

Los rastreadores web de gran alcance pueden manejar varios hilos de procesamiento a la vez aprovechando al máximo los recursos de hardware que se dispongan. El rastreador web se ejecuta de acuerdo a los parámetros que se le asignen como pueden ser el número de páginas y niveles de profundidad dentro de un dominio [19, 64]. En la figura 2.4 se muestra un diagrama con el proceso básico que ejecuta un rastreador web. Las tres funciones más importantes que debe ejecutar este componente son [19]: 1. Resolver los nombres de host en la URL a una dirección IP usando un DNS (Domain Name Service), 2. Permitir la conexión con el puerto del servidor al cual se envía una solicitud para rastrear páginas web y 3. Recibir las páginas de respuesta a una solicitud de rastreo.

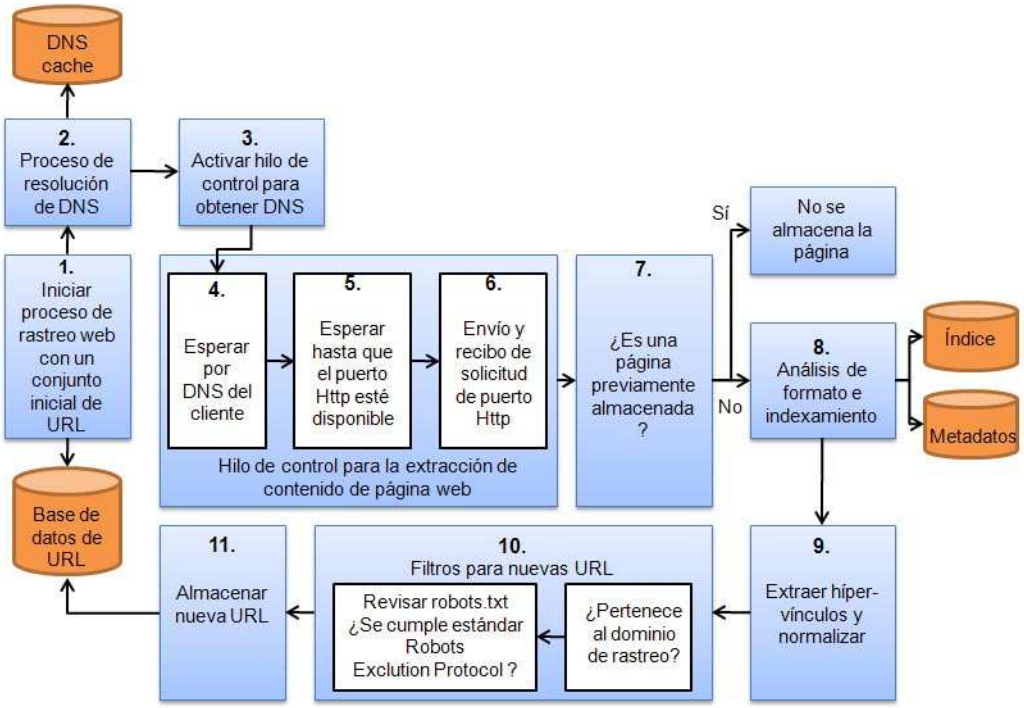


Figura 2.4: Arquitectura básica de un rastreador web

2.3.2 Análisis de Formato

Los documentos digitales son la entrada para el proceso de indexación, estos documentos son conjuntos de bytes almacenados en un archivo dentro de un servidor web. El primer paso para procesar estos archivos es la conversión de los bytes en una secuencia lineal de caracteres [64]. El análisis de formato permite la obtención del flujo de caracteres que componen el contenido de los archivos digitales a través del uso de unos filtros específicos para cada tipo de formato de archivo. De acuerdo al enfoque del sistema de recuperación de información pueden existir diferentes necesidades para la extracción tanto del contenido del documento como de su estructura, esquema de presentación y metadatos. En general los sistemas enfocados en la Web, almacenan el contenido de texto, información sobre los hipervínculos, ubicación URL del documento y los metadatos.

Hay una amplia variedad de filtros tanto comerciales como de código abierto disponibles en la Web los cuales se han ido adaptando a las nuevas exigencias del mercado. Los filtros comerciales más usados son: Stellent parte de los productos para búsqueda de Oracle[4], KeyView[5], y IFilter[6] incluido en algunos de los productos de Microsoft [7]. En cuanto a las opciones de código abierto, existen diferentes herramientas para el procesamiento de múltiples formatos y desarrollados en diferentes lenguajes de programación disponibles para su uso como los que se listan en portales que continuamente están actualizando información acerca de este tipo de herramientas como Search Tools[7].

El análisis de formato utiliza un conjunto de filtros que ejecutan un proceso básico previo al indexación de un documento [4], compuesto por dos fases, primero se identifica el tipo de formato que puede ser texto plano, Microsoft Word, PDF, HTML, XML, Frame Maker, etc. Luego se selecciona el filtro

[4] Stellent `http://www.oracle.com/stellent/index.html`
[5] KeyView `http://www.autonomy.com/content/Products/idol-modules-connectors/index.en.html`
[6] IFilter `http://www.ifilter.org/`
[7] Search Tools `http://www.searchtools.com/guide/file-format-parsing.html`

adecuado al formato para extraer el contenido del documento en forma de un flujo de caracteres identificando todo contenido de texto codificado, el título y metadatos adicionales como la URL, el autor, fecha de creación, fecha de actualización [4]. Los datos identificados se almacenan en archivos locales intermedios llamados segmentos [64].

Codificación de texto

Para la extracción del texto contenido en los documentos se tienen en cuenta dos aspectos la codificación y el formato. Los textos digitales están codificados en dígitos binarios adoptando diferentes esquemas. Los primeros esquema fueron EBCDIC y ASCII, este último se convirtió en el estándar de ocho bits (ISO-Latin) el cual se adapta a muchos idiomas incluyendo los acentos y otros signos diacríticos. Para las lenguas orientales como el chino y el japonés se usa la codificación Unicode (ISO-10616) que tiene 16 bits [4].

Metadatos

La mayoría de los documentos y colecciones de textos digitales han asociado con ellos lo que se conoce como metadatos. Los metadatos son información sobre la organización de los datos, los campos de datos diferentes, y la relación entre ellos. Por definición los metadatos son "datos acerca de los datos". Las formas comunes de metadatos asociados con el texto incluye el autor, la fecha de publicación, la fuente de la publicación, la longitud del documento (páginas, palabras, bytes, etc), y el género del documento (libro, artículo, nota, etc.) [4].

El primer esquema para la creación de metadatos fue el desarrollado por el Dublin Core Metadata Iniciative [22], el cual propones 15 campos para describir una fuente bibliográfica y han sido desarrollados en estándares como el ISO Standard 15836:2009 [97] y el ANSI/NISO Standard Z39.85-2007 [74]. Marchionini [65], se refiere a este tipo de información como metadatos descriptivos, los metadatos externos al significado del documento y que describe la forma en que fue creado. Otro tipo metadatos son los que se encuentran dentro del contenido del documento, los cuales son llamados metadatos semánticos [4]. Estos metadatos se asocian a los términos clave los cuales en muchas publicaciones indexadas corresponden a ontologías específicas de un área [4].

El enfoque más reciente para desarrollo de metadatos son las especificaciones propuestas por el *World Wide Web Consortium* (W3C), desarrolladas en el grupo *Semantic Web Activity* (antes llamado *Metadata Activity*). El principal interés de este grupo es el desarrollo de la plataforma para descripción de recursos web llamada *Resource Description Framework* (RDF™) y la plataforma PICS™ (*Platform for Internet Content Selection*). La especificación RDF [117] consta de un conjunto de recomendaciones del W3C, publicadas en 2004 que incluye temas de diseño web y arquitectura de metadatos. Otras tecnologías, como OWL (*Web Ontology Language*)[8] o SKOS (*Simple Knowledge Organization System*)[9], se basan en RDF y proveen un lenguaje estructurado, ontologías basadas en la Web que permiten una integración robusta e interoperabilidad de datos entre las comunidades descriptivas.

2.3.3 Índice de Documentos

En sistemas de recuperación de información y librerías digitales se requiere almacenar los documentos que las conforman para su posterior consulta. Para llevar a cabo esta tarea se utilizan diferentes estrategias que incluyen la creación de diversas estructuras de almacenamiento y técnicas de búsqueda, dentro de las cuales están: búsquedas lineales, índices invertidos, tablas *hash*, *tries y* archivos de firmas [56]. Las rutinas de búsqueda lineal desde el comienzo van carácter por carácter a través del texto

[8] OWL `http://www.w3.org/2001/sw/wiki/OWL`
[9] SKOS `http://www.w3.org/2001/sw/wiki/SKOS`

de los documentos completos en busca de la información requerida, y al final todos los archivos de los documentos han sido procesados. Con los índices invertidos, la búsqueda consiste en la creación de una estructura de datos que recuperar sólo los bloques que contienen partes de contenido de una consulta hecha por un usuario [56]. Las tabla *hash* utilizan algoritmos de búsqueda para calcular una ubicación adecuada de los elementos que se requieren dentro de una base de datos [56].

2.3.3.1 *Tries*

Los *tries* son similares a los índices invertidos, porque utilizan las letras del alfabeto para asociar las palabras que empiezan por cada una de ellas lo cual hace que la recuperación de términos sea muy rápida como si se tratara de un diccionario. Sin embargo esta estrategia tiene desventajas considerables porque se requiere muchos recursos de tiempo y memoria para la construcción de los *tries*, por lo cual esta estrategia no es popular en la construcción de librerías digitales [56].

2.3.3.2 Archivos de firmas

Los archivos de firmas requieren una exploración lineal de nuevo, pero estos condensan el material de búsqueda más rápido porque cada registro corresponde a un código corto que describe el contenido. Los códigos utilizados son generados con rutinas hash. Sin embargo si la búsqueda es en archivos lineales esta estrategia no es tan eficiente como lo pueden ser los índices invertidos [56].

2.3.3.3 Índice Invertido

Los índices invertidos son las estructuras más usadas en grandes sistemas y librerías digitales actuales, debido en gran parte a que los índices permiten almacenar colecciones de archivos de giga-bytes o terabytes y se puede obtener tiempos de respuesta de búsqueda muy cortos. Los índices invertidos se pueden comparar con los índices en la parte trasera de los libros: los elementos buscados se extraen y ordenan alfabéticamente para que luego sean más fáciles de encontrar y así dar respuesta a diferentes consultas. En promedio, para encontrar un elemento en una lista de artículos toma $Log_2 N$ pasos, lo que es mucho más rápido que la lectura la totalidad de un largo archivo [4, 56].

Este tipo de índice se compone del vocabulario compuesto por el conjunto de palabras presentes en un conjunto de documentos, y de una lista de desplazamiento (en inglés *posting list*) que consiste en una serie de apuntadores donde se almacenan la frecuencia de ocurrencia de un término del vocabulario y los números que identifican los documentos donde aparece cada término [4, 69]. Existen diferentes métodos para almacenar las listas de desplazamiento, su elección depende de las necesidades de la aplicación donde sea implementado el índice. En [124] se encuentra una descripción detallada de cada una de los métodos para la implementación de los vectores y la construcción índices invertidos.

En cuanto a recursos de almacenamiento utilizados, el espacio necesario para un índice invertido es proporcional al tamaño del conjunto de documentos por lo cual existen técnicas para su reducción y optimización [8]. La mayoría de los casos requiere un espacio reducido para vocabulario, pero el espacio utilizado por la lista de desplazamiento es mucho más considerable [8]. Hay que resaltar que la cantidad de información que se almacena depende de las funcionalidades ofrecidas por un sistema específico, y en consecuencia hay una relación directa entre el espacio necesario y las funcionalidades que se implementen [69].

Por ejemplo, en el caso de algunos motores de búsqueda se almacena el texto completo de los documentos, con el fin de presentar al usuario en el listado de los resultados los fragmentos o resúmenes de cada documento donde aparecen los términos de búsqueda; mientras que otros aplican técnicas de preprocesamiento de texto y solo almacenan el vocabulario reducido del contenido de los documentos, pero no ofrecen los fragmentos o resúmenes en los resultados de búsqueda [69]. Otro aspecto importante a ser tenido en cuenta es el tiempo de respuesta cuando se recorren las listas de desplazamiento,

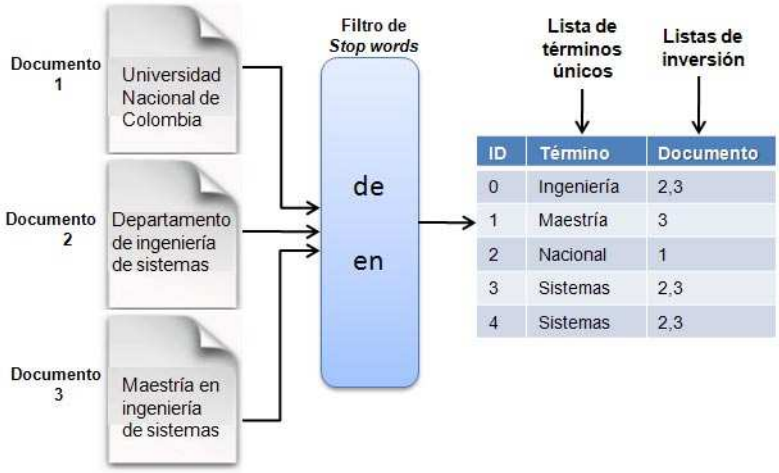


Figura 2.5: Archivo de estructura invertida

porque cuando se trata de sistemas web el tiempo para responder una consulta de usuario debe ser mínimo [69].

2.3.3.4 Representación de índices invertidos

Las representaciones mas usadas para la creación de índices invertidos en sistemas de recuperación de información son los archivos de estructuras invertidas (en inglés, *inverted file structures*) , el modelo de espacio vectorial [4] y Indexación Semántica Latente (por sus siglas en inglés, LSI), debido a que su implementación es sencilla y permite que se ejecuten procesos de recuperación de información como el indexación y la búsqueda de consultas de usuario de forma rápida comparada con otros modelos de recuperación de información [4].

Archivos de estructuras invertidas

Los archivos de estructuras invertidos integran tres componentes [8]: 1. Archivo del documento: es la estructura donde se almacenan el identificador y todos los términos de un documento, 2. Diccionario: es una lista ordenada de todos los términos únicos presentes en la colección de documentos y 3. Listas de inversión: contienen los apuntadores desde los términos a los documentos que contienen cada término. En la figura 2.5 se describe el proceso de creación de un archivo de estructura invertida. En la figura 2.5 se muestra un ejemplo de la construcción de un archivo de estructura invertida a partir de un conjunto de documentos.

2.3.3.5 Modelo de espacio vectorial

El modelo de espacio vectorial fue propuesto por George Salton en 1971 e implementado en el sistema SMART [90]. En este modelo los documentos son representados como vectores en un espacio euclidiano multidimensional. Cada componente del vector representa un palabra clave o término relacionado con el documento dado [93]. El valor para cada componente depende del grado de relación entre el término asociado y el respectivo documento, este concepto fue propuesto por primera vez en [61]. En este modelo un documento D_i se representa como un vector multidimensional que contiene t términos así [90]: $D_i = (a_{i1}, a_{i2}, \ldots, a_{ij}, \ldots, a_{it})$ donde a_{ij} Representa el grado de relación entre un

documento ***i*** y un término ***j***. En la figura 2.6 se muestra un ejemplo de la representación vectorial de tres documentos que contienen tres términos, es decir son de tres dimensiones.

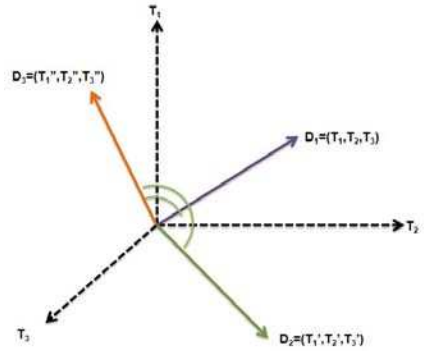


Figura 2.6: Ejemplo de representación gráfica de tres documentos en espacio vectorial

2.3.3.6 Medidas para encontrar la relación de términos y documentos

Existen diferentes esquemas de cálculo para determinar la relación de los términos y los documentos de una colección, comúnmente se le conoce como "ponderación de términos" (en inglés ***term weighting***). A continuación se describen los tres esquemas más usados.

1. **Ponderación Binaria:** Esta ponderación indica acerca del hecho que un término está presente en un documento pero no sugiere el grado de relación del término con el documento [106]. Por tal razón, se identifica como el caso más sencillo para la ponderación de términos, la cual consiste en asignar el valor uno si la palabra clave ***j*** aparece en el documento ***i,*** $a_{ij} = 1$; y cero en otro caso $a_{ij} = 0$ [106].

2. **Ponderación por frecuencia de términos:** Éste esquema de ponderación denota cuántas veces el término ***j*** ocurre en el documento ***i*** [91].

$$a_{ij} = tf_{ij} \tag{2.1}$$

   Éste cálculo es más informativo que la ponderación binaria, sin embargo tiene algunas deficiencias [91, 75]. La medición que se hace es con respecto a un conjunto de ocurrencias del término de forma local dentro de un documento, pero no se puede establecer la relevancia del término dentro de todo el conjunto de documentos. Puede suceder que el término aparezca en un documento de forma significativa, digamos en el 80% del documento lo cual indicaría que él término es muy significativo en dicho documento [75]. Caso contrario puede darse cuando un término aparece en todos los documentos del conjunto, lo cual puede indicar que el término no es tan significativo [75].

3. **Ponderación Tf-idf:** El esquema de ponderación Tf-idf (del inglés ***term frequency–inverse document frequency***) se enfoca en equilibrar la ponderación tanto local como global de un término dentro de un conjunto de documentos [91]. La ecuación de cálculo en este esquema es el siguiente [91].

$$a_{ij} = tf_{ij} \bullet idf_i \tag{2.2}$$

Donde tf_{ij} es la frecuencia del término $\boldsymbol{j}$ en el documento $\boldsymbol{i}$ y el factor idf_i se refiere a la frecuencia inversa de los documentos en los que aparece el término $\boldsymbol{j}$. El termino idf_i corresponde a $log(N/df_i)$, donde N es el número de documentos de la colección y df_ies el número de documentos en los cuales aparece el término $\boldsymbol{j}$. Por tanto la ecuación queda de la siguiente manera [91]:

$$a_{ij} = tf_{ij} \bullet log(N/df_i) \tag{2.3}$$

Usando éste esquema, cuando un término aparece en todos los documentos de la colección, $df_i = N$, entonces el factor $idf_i = 0$ lo cual indica que el término no es útil como un discriminante de documentos.

2.3.3.7 Matriz de términos y documentos

En el modelo de espacio vectorial, los vectores con la ponderación de cada término en cada documento se almacenan en una estructura llamada la matriz de términos y documentos (en inglés, ***term-document matrix***) de tamaño $m \times n$ donde m representa el número de términos y n el número de documentos. En esta matriz los vectores columna representan los vectores de documentos es decir cada columna representa cada documento de la colección y los vectores de las filas representan los términos usados en el proceso de indexación del conjunto de documentos [91]. Cada elemento de la matriz a_{ij} es el valor ponderado del término calculado con los esquemas de ponderación descritos anteriormente: binario, frecuencia de términos ó Tf-idf.

Este esquema permite determinar el contenido semántico de la colección de documentos y también aprovechar las relaciones geométricas entre los vectores de los documentos y los términos con el objetivo de explicar la similitud y la diferencia entre diferentes conceptos o temas presentes dentro del conjunto de documentos [8].

Para el caso de colecciones heterogéneas de documentos, como es el caso de aquellas que describen diferentes temas como los periódicos, las enciclopedias y portales web, el número de términos (m) es por lo general mucho más grande que el número de documentos ($n \ll m$). En la Web, la situación es inversa debido a la gran cantidad de páginas que están publicadas. Para el caso del idioma español se tiene un estimado de 150.000 palabras, en inglés 300.000. Para el caso de los portales se han planteado cuatro tamaños aproximados de las colecciones de documentos: 10 o menos, 11 a 50, 51 a 100, más de 100 [1]. Debido a que los documentos solo contienen un subconjunto pequeño de las palabras del diccionario de un idioma, la mayoría de los elementos de una matriz de términos y documentos serán igual a cero.

En la figura 2.7 se describe el proceso de creación de una matriz de términos y documentos, llamada la matriz A a partir de un conjunto de documentos compuesto por siete documentos que contienen el título del documento. Se observa que no todos los términos se incluyen, solo se consideran los términos que aparecen más de una vez y se eliminan las *stop words*. La matriz de términos y documentos es una matriz dispersa, debido a la gran cantidad de ceros que aparecen en ella. Para el almacenamiento de este tipo de matrices se han propuesto diferentes métodos para simplificarla [6, 8] como es el caso del método de compresión de almacenamiento por fila (por sus siglas en inglés CRS) y el método de compresión de almacenamiento por columna por sus siglas en inglés CCS) [8].

2.3.3.8 *Latent Semantic Indexing*

El modelo de espacio vectorial ha sido uno de los modelos más populares pero no se ocupa de muchos de los problemas de la actual recuperación de información. En primer lugar, el modelo se basa en la coincidencia literal entre los términos de un documento y los de una consulta [75]. Sin embargo, debido a los sinónimos es frecuente encontrar muchas maneras para expresar una misma idea, y por tanto los términos de búsqueda no concuerdan con documentos relevantes que tienen sinónimos. De esta

Términos	**Documentos**
t=5	d=7
T1: Recuperación	D1: Recuperación de información en librerías
T2: Información	D2: Organización automática de información
T3: Librerías	D3: Análisis de valores singulares
T4: Valores	D4: Clasificación de documentos en librerías
T5: Singulares	D5: Recuperación de información bibliográfica
	D6: Descomposición de matrices en valores singulares
	D7: Sistemas de librerías digitales

La **matriz de términos y documentos** de 5 x 7 antes de la normalización por longitud de columna donde el elemento a_{ij} es el número de veces que aparece el término i en el documento j.

$$A = \begin{pmatrix} 1{,}0 & 0{,}0 & 0{,}0 & 0{,}0 & 1{,}0 & 0{,}0 & 0{,}0 \\ 1{,}0 & 1{,}0 & 0{,}0 & 0{,}0 & 1{,}0 & 0{,}0 & 0{,}0 \\ 1{,}0 & 0{,}0 & 0{,}0 & 1{,}0 & 0{,}0 & 0{,}0 & 1{,}0 \\ 0{,}0 & 0{,}0 & 1{,}0 & 0{,}0 & 0{,}0 & 1{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 & 1{,}0 & 0{,}0 & 0{,}0 & 1{,}0 & 0{,}0 \end{pmatrix}$$

La **matriz de términos y documentos** de 5 x 7 después de la normalización por longitud de columna:

$$A' = \begin{pmatrix} 0{,}58 & 0{,}0 & 0{,}0 & 0{,}0 & 0{,}71 & 0{,}0 & 0{,}0 \\ 0{,}58 & 0{,}0 & 0{,}0 & 0{,}0 & 0{,}71 & 0{,}0 & 0{,}0 \\ 0{,}58 & 1{,}0 & 0{,}0 & 1{,}0 & 0{,}0 & 0{,}0 & 1{,}0 \\ 0{,}0 & 0{,}0 & 0{,}71 & 0{,}0 & 0{,}0 & 0{,}71 & 0{,}0 \\ 0{,}0 & 0{,}0 & 0{,}71 & 0{,}0 & 0{,}0 & 0{,}71 & 0{,}0 \end{pmatrix}$$

Figura 2.7: Matriz de términos y documentos

manera, algunos de los documentos no aparecen en los resultados de búsqueda a pesar de que deberían [75]. En segundo lugar, el modelo de espacio vectorial supone que las palabras clave utilizadas para indexar la colección de documentos son independientes entre sí. Sin embargo, en aplicaciones reales no siempre puede ser el caso. Por esta razón, un volumen significativo de información redundante está presente en la matriz de términos y documentos [75]. Por último, el modelo clásico de VSM utiliza una sola palabra para describir los documentos, mientras que son los ***conceptos*** lo que la mayoría de usuarios buscan [75].

Indexación Semántica Latente (LSI) es técnica novedosa de recuperación de información que se diseñó para abordar las deficiencias del modelo de espacio vectorial clásico [75]:

> LSI trata de superar los problemas de coincidencia léxica utilizando índices conceptuales derivados estadísticamente, los en lugar de palabras individuales para la recuperación. LSI asume que hay algún tipo de estructura subyacente o latente en el uso de las palabras que está parcialmente oculto por la variabilidad en la selección de palabras. La Descomposición en Valores Singulares (por sus siglas en inglés, SVD) de forma reducida se utiliza para estimar la estructura en el uso de la palabras a través de los documentos. Luego, la recuperación se ejecuta usando una base de datos de los valores singulares y vectores obtenidos del resultado de la descomposición en valores singulares reducido. Los datos de rendimiento muestran que estos vectores derivados estadísticamente, son indicadores más sólidos del significado de términos individuales [10].

Descomposición en valores singulares

La construcción matemática subyacente del método LSI es la descomposición en valores singulares de la matriz de términos y documentos. Esta descomposición divide una matriz A de tamaño $t \times d$ en

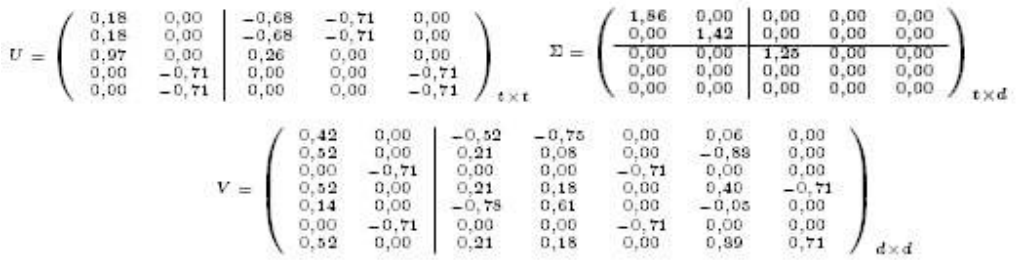


Figura 2.8: Ejemplo de LSI - Descomposición SVD

$$A_2 = \begin{pmatrix} 0{,}14 & 0{,}17 & 0{,}0 & 0{,}17 & 0{,}05 & 0{,}0 & 0{,}17 \\ 0{,}14 & 0{,}17 & 0{,}0 & 0{,}17 & 0{,}05 & 0{,}0 & 0{,}17 \\ 0{,}76 & 0{,}94 & 0{,}0 & 0{,}94 & 0{,}25 & 0{,}0 & 0{,}94 \\ 0{,}00 & 0{,}0 & 0{,}72 & 0{,}0 & 0{,}0 & 0{,}72 & 0{,}0 \\ 0{,}00 & 0{,}0 & 0{,}72 & 0{,}0 & 0{,}0 & 0{,}72 & 0{,}0 \end{pmatrix}$$

Figura 2.9: Ejemplo de LSI - Aproximación Rango - 2

tres matrices U, Σ y V donde A $= U\Sigma V^T$. La matriz U de tamaño $t \times t$, es una matriz ortogonal cuyos vectores columna determinan los vectores singulares derechos de A. La matriz Σ de tamaño $t \times d$ es una matriz diagonal que tiene los valores singulares de A ordenados es forma decreciente $\left(\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{min(t,d)}\right)$ a lo largo de su diagonal. El rango r_a de la matriz A es igual al número de sus valores singulares diferentes de cero. Las primeras r_a columnas de U conforman una base ortogonal para el espacio columna de A [10].

Los vectores columna de la matriz U al ser vectores vectores de la base de la matriz A, también se expresan en el mismo espacio y por tanto pueden ser tratados como pseudo-documentos que de alguna manera un resumen de los **conceptos abstractos** más importantes presentes en la colección. El vector de la primera columna de la matriz U representa el concepto abstracto "Recuperación de Información", mientras que el segundo vector es un pseudo-documento que se ocupa de "Valores Singulares". De hecho, estas son las dos ideas principales en torno al cual giran los documentos del ejemplo. En el algoritmo LINGO el modelo LSI se utiliza para el agrupamiento de documentos [75].

2.3.4 Procesador de Consultas de Usuario

A partir de un índice de documentos, es posible realizar consultas de manera muy eficiente. El proceso básico para responder a una consulta hecha por un usuario sigue tres pasos básicos [4, 69]: identificar el vocabulario de la consulta de búsqueda, recuperar las listas de desplazamiento y la manipulación de las ocurrencias para responder la consulta.

1. **Identificar el vocabulario de búsqueda**

   La consulta se divide en palabras que son buscadas en el vocabulario del índice. Este se ejecuta de forma eficiente si el vocabulario está ordenado alfabéticamente.

2. **Recuperación de ocurrencias**

   Todas las listas de desplazamiento de los términos que aparecen en el vocabulario de la consulta hecha por el usuario, se seleccionan dentro del índice de documentos y se almacenan en la memoria temporal.

3. **Manipulación de ocurrencias**

   Las listas donde aparecen los términos de la consulta son procesadas para que respondan al tipo de consulta que puede ser de tipo booleana, de proximidad y con el uso de símbolos comodines (en inglés, *wildcards*). Dependiendo del tipo de consulta el procesamiento de la consulta es diferente. Por ejemplo, las búsquedas booleanas consisten en un conjunto de términos que se combinan utilizando un conjunto de operadores (como "y", "o", "no") con el fin de recuperar documentos que coinciden con estas condiciones. Este tipo de preguntas son muy sencillas de resolver mediante un índice invertido, ya que la única manipulación es necesario fusionar el desplazamiento listas y seleccionar sólo aquellos que cumplan las condiciones. Por su parte las búsqueda por frases y por proximidad, consisten en preguntas más difíciles de resolver porque requieren una compleja manipulación de los términos. La frase se refiere a una consulta que contiene un conjunto de palabras que aparecen en un patrón particular, mientras que la búsqueda por proximidad es una versión más relajada donde las palabras pueden estar a cierta distancia. Por este tipo de consultas, es necesario contar con la lista de ocurrencias ordenadas por la posición de la palabra y el patrón de ocurrencia en lista de resultados, lo cual hace el proceso de recuperación más complicado que una búsqueda booleana [4, 69].

4. **Ranking de resultados de búsqueda**

   Después de realizar la búsqueda en el índice, es necesario ordenar los documentos de acuerdo a su relevancia con respecto a la consulta hecha por el usuario. Esta etapa de clasificación podría ser opcional, dependiendo de los requerimientos del sistema, pero en la Web se ha convertido en un elemento importante en la resolución de consultas de usuario. Este proceso tiene en cuenta varios factores adicionales, además de si la lista de documentos satisface la consulta o no [4].

   El primer modelo utilizado fue el modelo booleano que comparaba los términos de la consulta con los conjuntos de términos usados para identificar el contenido de cada documento [4]. Luego se planteó el modelo de retroalimentación de relevancia en el cual los usuarios validan los documentos de las consultas y así se afina el cálculo de la relevancia [87]. El siguiente modelo fue el probabilístico que se basa en el cálculo de la probabilidad para una colección de documentos [102]. En los años setenta, se propuso el modelo de espacio vectorial que utiliza conjuntos de términos para representar tanto las consultas de los usuarios como los documentos y para calcular la relevancia utiliza operaciones de álgebra lineal para hallar la similitud entre una consulta y un documento específico [90].

   En 1998 se publicó la primera versión del algoritmo *Page Rank* usado por Google, el cual calcula la "popularidad" de una página recuperada a partir de una combinación del número de enlaces que referencian una página y el número de enlaces que salen de la misma página [13], este algoritmo ha demostrado su eficiencia colocando a Google como el mejor buscador web más usado en todo el mundo [73].

   En colecciones específicas de documentos y sistemas de recuperación de información específicos de un contexto (e.g. uno o varios portales web), la técnica más popular es el modelo de espacio vectorial. En este modelo, cada consulta es representada como un pseudo-documento, es decir en forma de un vector de términos de la consulta de igual manera como se almacenan los documentos [91, 75]. El proceso para encontrar los documentos con mayor similitud al vector de consulta incluye el cálculo geométrico de la distancia entre el vector de cada documento y el vector de la consulta. Se han propuesto varios métodos de cálculo de similitud entre los que se resaltan tres: similitud del coseno, índice DICE, coeficiente Jaccard y la distancia euclidiana [91] (Ver subsección 2.5.1), los cuales también son usados para el agrupamiento de documentos. En los sistemas de recuperación en general la medida de similitud más utilizada es el coeficiente del coseno [4].

(a) 1990 - Archie el primer buscador web

(b) 2010 - Google el buscador web más usado del mundo

Figura 2.10: Interfaces de motores de búsqueda

2.3.5 Interfaz de Usuario

En la mayoría de los sistemas de recuperación de información, las las interfaces de búsqueda permitir a los usuarios la expresión de sus necesidades de información a través de la formulación de consultas y también la presentación de un listado de resultados ordenados[41]. Estas interfaces no han cambiado mucho desde la aparición de los primeros motores de búsqueda en 1990, básicamente porque no ha cambiado el estándar que deben cumplir este tipo de interfaces. Este estándar propone lo siguiente [41]:

1. La búsqueda de información es un medio para lograr algún otro fin, por tanto solo pocas veces una búsqueda es un objetivo en sí mismo. En el momento en el que una persona está buscando información, la mayoría de las veces está desarrollando una tarea más grande, y por tanto debe evitar que su flujo de pensamiento interrumpido por distracciones en una interfaz.

2. Retomando la idea anterior, la búsqueda es una tarea mental intensiva por lo cual cuando una persona lee algún texto, se centra en esa tarea y no es posible leer y pensar en otra cosa al mismo tiempo. En consecuencia, se busca que la interfaz sea lo más minimalista posible y cause la menor distracción durante la lectura.

3. Debido a que una gran mayoría de personas usan búsquedas web, el diseño de la interfaz debe ser comprensible y atractivo para una amplia variedad de usuarios de todas las edades, culturas y orígenes, aplicado a una enorme variedad de necesidades de información.

En la figura 2.10 se muestra una comparación de las interfaces gráficas de dos motores de búsqueda, en la figura 2.10a está Archie[10] el primer motor de búsqueda web ubicado en 1990 y en la figura 2.10b está Google[11] el motor de búsqueda web más usado en la actualidad. En los sistemas de librerías digitales, las interfaces de usuario están enfocadas en presentar gráficamente opciones para la navegación y la búsqueda de documentos dentro de una estructura organizada de colecciones de documentos. En la figura 2.11a se muestra la interfaz principal de una de las librerías digitales más populares llamada Greenstone[12] compuesta por la estructura de navegación a través de las colecciones de documentos, en la figura 2.11b se muestra la interfaz para búsqueda de Greenstone que se compone de un campo de inserción de consultas a través de texto y las opciones de selección para búsquedas avanzadas.

[10] Archie `http://archie.icm.edu.pl/archie-adv_eng.html`
[11] Google `http://www.google.com`
[12] Greenstone `http://www.greenstone.org/`

(a) Interfaz principal de acceso compuesta de acceso a las colecciones

(b) Interfaz de búsqueda

Figura 2.11: Librería Digital Greenstone

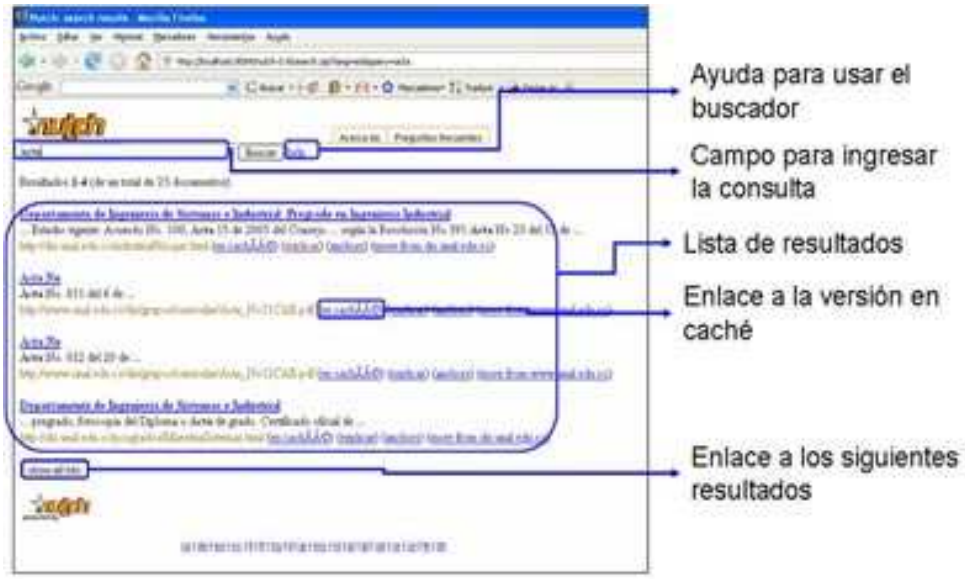


Figura 2.12: Interfaz de búsqueda del motor de búsqueda Nutch

2.3.6 Motor de Búsqueda Nutch

Nutch[13] es un motor de búsqueda de código abierto y es un proyecto amparado por *The Apache Software Foundation*[14]. Está construido con la librería de indexación más utilizada llamada Lucene[15], y contiene componentes específicos para la Web, tales como un rastreador, una base de datos de enlaces, analizadores de formato de archivos como HTML, texto plano y PDF [16]. Nutch tiene una arquitectura altamente modular que permite a los desarrolladores crear módulos adicionales de acuerdo a los requisitos específicos de diferentes portales o conjuntos de documentos [81]. La figura 2.12 muestra la interfaz de usuario de Nutch, cuya estructura cuenta con funciones básicas similares a las de los motores de búsqueda disponibles en la Web como Google. Este tipo de interfaz es fácil de entender por los usuarios, ya que tiene campos intuitiva para escribir una consulta y la lista de resultados muestra un pequeño extracto de cada documento.

[13] Nutch `http://nutch.apache.org/`
[14] *The Apache Software Foundation* `http://www.apache.org/`
[15] Lucene `http://lucene.apache.org/java/docs/index.html`

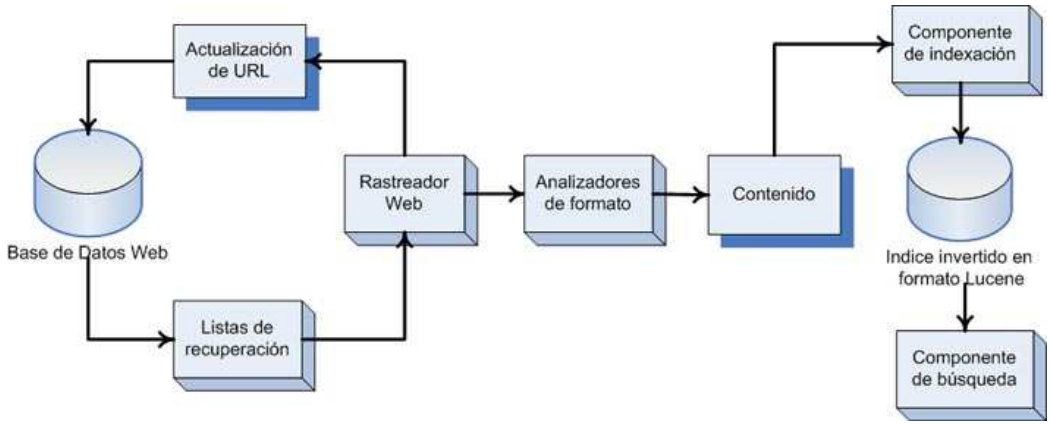


Figura 2.13: Arquitectura motor de búsqueda Nutch

Funcionamiento de Nutch

En la figura 2.13, se muestra un diagrama de bloques de la arquitectura del motor de búsqueda Nutch. A continuación se describirá cada uno de sus componentes y como interactúa con los demás [16].

- **La Base de Datos Web**: Es una estructura de datos especializada en persistencia para almacenar la estructura y las propiedades recuperadas de los documentos web. Dentro de la estructura se almacena tanto el contenido de las páginas web como sus respectivos hiper-vínculos. Las páginas web se indexan por su URL y para el almacenamiento del contenido dentro del índice se utilizan funciones hash MD5. En cuanto a los enlaces, se representa el enlace de una página web (fuente) a otra (objetivo) [16].
- **Listas de recuperación:** contiene todas las URL que se van a rastrear.
- **Rastreador Web:** Recorre las listas de recuperación y gestiona la descarga de los documentos web en el servidor donde están publicados. Las dos salidas que tienen dichos procesos son: por un lado son las actualizaciones de la base de datos web, que contendrán las páginas que han aparecido o desaparecido desde el último intento de búsqueda; la segunda salida son los documentos en su formato original que son enviados a los analizadores de formato [16].
- **Analizadores de formato:** como se mencionó en la subsección 2.3.2 , estos analizadores se encargan de extraer el contenido de los documentos, y luego dicho contenido es usado para generar el índice invertido (Ver subsección 2.3.3). Nutch cuenta con analizadores para diferentes tipos de archivo texto plano, HTML, JavaScript, Microsoft Excel, Microsoft Word, Open Office, PDF, RSS, SWF y ZIP [101]
- **Componente de indexación:** crean el índice invertido de todas las páginas que el sistema ha recuperado y es creado por combinación de los índices de los segmentos individuales. Dichos procesos utilizan funciones de la librería de recuperación de información llamada Lucene [16].
- **Librería Lucene**: es un API de código abierto que sirve para la implementación de componentes de indexación en sistemas de recuperación de información, la primera versión fue desarrollada en lenguaje Java por Doug Cutting [40]. Actualmente es un proyecto amparado por *The Apache Software Foundation*[16] y se distribuye bajo la Apache Software License. Existen diferentes

[16] Apache Software Foundation `http://www.apache.org/`

versiones de Lucene para otros lenguajes incluyendo C#[17] y Python[18]. Es útil para cualquier aplicación que requiera indexación y búsqueda de texto completo. Lucene ha sido ampliamente usado por sus capacidades robustas en la implementación de funciones de búsqueda en portales web y también motores de búsqueda, los cuales suman un total de 300 reportados en el portal web de Lucene[19]. El centro de la arquitectura lógica de Lucene se encuentra el concepto de Documento (Document) que contiene Campos (Fields) de texto [40].

- **Componente de búsqueda:** Dada una consulta en forma de un conjunto de términos, este componente se encarga de encontrar rápidamente el conjunto de documentos más relevante y muestra un listado con los documentos ordenados de acuerdo a dicha relevancia con el título, resumen y enlace de cada documento [16].

2.4 PREPROCESAMIENTO AUTOMÁTICO DE DOCUMENTOS

Dentro de los procesos de recuperación de información y agrupamiento de documentos dentro de los sistemas de librerías digitales de documentos, se requiere un procesamiento bastante complejo y extenso debido a gran cantidad de términos condensados en los conjuntos de documentos. Por esta razón, se utilizan diferentes técnicas para eliminar información textual no relevante. Algunas de estas técnicas son esenciales dentro de este tipo de sistemas como es el caso del análisis léxico que es fundamental para el indexación de los documento textuales, así como también existen técnicas opcionales que permiten mejorar los resultados obtenidos como el procesamiento de lenguaje natural. A continuación se describen las técnicas más usadas para el procesamiento de texto.

2.4.1 Análisis Léxico

El análisis léxico consiste en un proceso de conversión de un flujo de caracteres, que se extraen del texto contenido en un documento, en un flujo de palabras. Las palabras que se descubren son las que utilizan para la creación de los índices invertidos de una colección de documentos. Un aspecto importante en la implementación de éste tipo de analizadores es el reconocimiento de los espacios es blanco como pueden ser los cambios de línea, tabulaciones y los espacios normales entre palabras [75]. Otros analizadores pueden incluir funciones más sofisticados para el tratamiento de caracteres numéricos, guiones y signos de puntuación [4]. Para el caso de los datos numéricos, por lo general son malos términos para indexación por si solos para que sean de utilidad necesitan del contexto que los rodea, por tanto es muy frecuente que sean omitidos en los procesos de indexación. Otro caso es el de los dígitos cuando son usados en acrónimos, por ejemplo MP3[20], en donde los dígitos se convierten en parte importante del término y por tanto debe incluirse en el índice invertido.

Los analizadores léxicos que se enfocan en los documentos web, además tienen que incluir características específicas como los caracteres especiales y las etiquetas HTML. Para el caso del símbolo arroba "@", debe ser analizado dentro de su contexto y si se trata de correos electrónicos debe almacenarse el nombre completo. Las etiquetas HTML no tienen un significado conceptual, pero pueden ayudar a identificar diferentes partes de un documento como el título, el contenido, la dirección URL y el nombre del documento o página web. Luego de identificar la estructura es posible la identificación de los términos presentes para ser almacenados en el índice invertido [86].

La efectividad de un proceso de indexación y el posterior proceso de clasificación de documentos se debe en gran medida al apropiado análisis léxico, en la actualidad se encuentran disponibles analizadores

[17] Lucene.net `http://lucene.apache.org/lucene.net/`
[18] PyLucene `http://lucene.apache.org/pylucene/`
[19] Powered by Lucene `http://wiki.apache.org/lucene-java/PoweredBy`
[20] MP3 - MPEG-1 Audio Layer-3

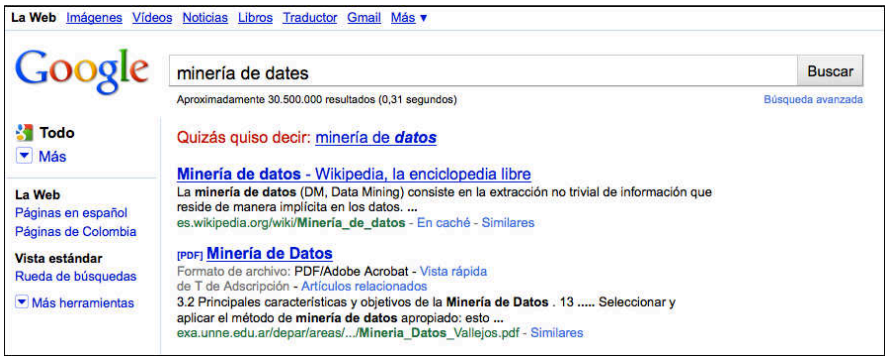


Figura 2.14: Corrector de ortografía de Google

léxicos (e.g. Flex[21], JLex[22]) y analizadores de expresiones regulares (e.g. Expresso 3.0[23]), estas herramientas permiten la implementación de herramientas robustas para el análisis léxico complejo [75].

2.4.2 Corrección de Ortografía

En el momento en que un usuario de un sistema de recuperación de información ingresa una consulta textual en forma de una frase para la búsqueda de documentos, se utilizan diferentes formas gramaticales de las palabras y a veces se desconoce como escribir determinadas palabras. Por esta razón, es necesaria la corrección de ortografía de las consultas usuario, de esta forma se asegura que se recuperen los documentos relacionados con las necesidades de información del usuario.

Sistemas como el motor de búsqueda Google tienen implementados diferentes algoritmos para la corrección de ortografía [48] y en la interfaz de usuario muestran sugerencias para efectuar búsquedas con las sugerencias de corrección como se muestra en la figura 2.14. Los algoritmos que se han desarrollado para la corrección de ortografía han evolucionado integrando nuevas métricas y estrategias como el propuesto en [44].

2.4.3 Eliminación de *Stop Words*

Las ***stop words*** son palabras específicas para un idioma y dentro de conjuntos de documentos de un mismo idioma dichas palabras son muy frecuentes porque corresponden a preposiciones (e.g. a, ante, bajo, cabe, con, contra, etc.), artículos (el, la, los, las), adjetivos determinativos (e.g. este, ese, aquel, mi, tu, su, nuestro, varios, algún, uno, dos, tres, primero, segundo, qué, cuál, quién), pronombres (e.g. yo, mí, me, conmigo, nosotros, nos, tú, ti, te, contigo, etc.), adverbios (e.g. aquí, allá, debajo, bien, mal, sí, también, etc.) y conjunciones (e.g. y, e, ni, o, u, bien, mas, pero, etc.). Las palabras que se presentan en el 80% de un documento como el caso de las ***stop words***, no son de gran utilidad porque no son factores diferenciadores de los documentos [75].

Los sistemas que efectúan eliminación de *stop words*, utilizan listas que contienen dichas palabras y son llamadas *stop lists*. Estas *stop lists* pueden encontrarse publicadas en la Web para diferentes idiomas [75] y pueden ser complementadas con palabras frecuentes identificadas.

Una ventaja de la eliminación de las *stop words* es la reducción del tamaño del índice en un factor aproximado del 40% [3, 34]. Sin embargo otros estudios sugieren que las *stop words* son un factor

[21] Flex: The Fast Lexical Analyzer `http://flex.sourceforge.net/`
[22] JLex: A Lexical Analyzer Generator for Java(TM) `http://www.cs.princeton.edu/~appel/modern/java/JLex/`
[23] Expresso 3.0 - The premier regular expression development tool `http://www.ultrapico.com/Expresso.htm`

importante dentro del análisis semántico de las frases presentes dentro de un documento, por ejemplo si se eliminan todas las *stop words* de un documento que contenga la frase "ser o no ser" sería imposible de encontrar [75].

2.4.4 Identificación de Lenguaje

Para que se puedan efectuar procesos de lematización, corrección de ortografía y eliminación de *stop words*, es necesario identificar el idioma en el cual se encuentra escrito el documento. Es poco frecuente que se conozca a priori el idioma de todos los documentos de una colección o que el documento tenga registrado el idioma en el que fue escrito. También se presenta el caso que una colección contenga documentos en idiomas diferentes lo cual es frecuente con documentos publicados en la Web. Es entonces cuando la identificación de lenguaje se convierte en un factor importante en los sistemas de recuperación de información, por esta razón se han desarrollado diferentes algoritmos.

Grefenstette en [36] compara dos métodos para identificación automática de lenguaje. Uno de los métodos es el *trigram* el cual se basa en las terminaciones de las palabras para identificar el idioma, usa una base de datos con dichas terminaciones para lenguajes como el inglés, francés, danés, holandés, alemán, italiano, noruego, portugués, español y sueco. El segundo método es llamado la técnica de la palabra pequeña (en inglés, *small word technique*), la cual se basa en el conteo de *stop words* que aparecen en la mayoría de los documentos, y las cuales son comparadas con listas de *stop words* para diferentes idiomas. El estudio concluye que ambos métodos funcionan igualmente bien para documentos con más de treinta palabras con un índice de error del 0.5% [36].

Otros estudios posteriores muestran otros métodos con otros enfoques para la clasificación de documentos de acuerdo a la identificación del idioma como[47, 82].

2.4.5 *Stemming* y Lematización

Los documentos de texto contienen gran cantidad de palabras de las cuales muchas son diferentes formas gramaticales de una misma palabra, tal es el caso de las conjugaciones de los verbos por ejemplo las palabras construirse, construyen y construyeron corresponden al verbo construir [64]. Además, existen familias de palabras derivadas que se relacionan con significados similares, tales como la palabra metal, metálico y metalmecánico. En muchos casos, es útil para la búsqueda de una de estas palabras a devolver los documentos que contengan además otras palabras del conjunto de palabras relacionadas [4, 64]. El objetivo tanto de la técnica de *stemming* como de la lematización es reducir las formas flexivas y las formas a veces derivadas relacionadas de una palabra a una forma común de base [64]. Por ejemplo. las palabras: soy, es, fue, se agrupan en el verbo ser; y las palabras: carro, carros, carrocería, se agrupan en la palabra carro. Las reglas para la lematización y el *stemming* de las palabras dependen del lenguaje. A pesar que tanto la lematización como el *stemming* tengan el mismo objetivo de reducir las palabras, lo hacen a través de procesos diferentes [64].

Stemming

El proceso de *stemming* consiste en un proceso heurístico que elimina las terminaciones de las palabras con el objetivo de reducir las palabras correctamente adecuadamente la mayoría de las veces. Es muy frecuente encontrar algoritmos de stemming que remueven afijos derivativos [64]. Los algoritmos de *stemming* se han desarrollado de acuerdo a las particularidades de cada idioma, como es el caso del algoritmo de Porter [104] que se utiliza para la lematización de palabras en idioma inglés. Para el caso de otros idiomas como las lenguas romances (español, francés, portugués, italiano y rumano), las lenguas germanas (alemán, holandés), lenguas escandinavas (sueco, noruego, danés), ruso y finlandés

se encuentran disponibles librerías de lematización que contienen algoritmos y listas de lemas como lo es la librería de código abierto llamada Snowball[24].

Lematización

La lematización se define como el proceso de reducción automática de elementos no esenciales de las palabras como lo son los sufijos y los prefijos, para obtener la parte esencial llamado lema [4]. Este proceso permite agrupar en un solo lema las variantes de las palabras como lo son los plurales, gerundios y formas temporales (presente, pasado, futuro, presente perfecto, etc.). A diferencia de los procesos de *stemming*, la lematización se efectúa a través del uso de listas de vocabulario y análisis morfológicos de las palabras con el objetivo de remover las terminaciones flexivas sólo si en el diccionario específico del idioma de la palabra existe una forma base o lema [64]. Para ejecutar procesos de lematización se han desarrollado lematizadores, los cuales son herramientas desarrollados desde el área del procesamiento de lenguaje natural [64],

2.4.6 Procesamiento de Lenguaje Natural

El procesamiento de lenguaje natural es una rama de la inteligencia artificial que busca facilitar la comunicación hombre-máquina por medio del lenguaje humano (lenguaje natural) por medio escrito o hablado. La aplicación del procesamiento de lenguaje natural en el campo de los sistemas de recuperación de información, ha permitido el desarrollo de aplicaciones para la respuesta de preguntas y la obtención de conocimiento específico a través de la interpretación de las preguntas del usuario y la búsqueda específica de información para presentar respuestas concretas. En motores de búsqueda como Google[25] es posible encontrar documentos que contienen la frase completa referente a una pregunta de usuario. Un sistema novedoso es Wolfram-Alpha[26] que es un motor de conocimiento el cual permite responder consultas de usuario a través de datos, imágenes, definiciones y datos históricos. Otros buscadores como START[27] y Answerbus[28] son sistemas experimentales que permiten la búsqueda de respuestas a preguntas concretas.

2.5 AGRUPAMIENTO DE DOCUMENTOS

Los algoritmos que permiten la identificación de conjuntos de documentos es una rama de investigación muy activa dentro del área de la minería de texto. Dichos algoritmos se dividen entre los algoritmos de clasificación y los algoritmos de agrupamiento. Los algoritmos de clasificación que consisten en métodos para asignar los documentos a los grupos identificados *a priori* por un experto o personas que conocen las características y los temas de dichos documentos. Por el contrario, los algoritmos de agrupamiento tienen como objetivo el descubrimiento de las colecciones de documentos similares de forma automática [46]. La idea original de los algoritmos de agrupamiento de texto se basa en los algoritmos de agrupamiento de datos que surgen a partir de la integración de diferentes enfoques de diferentes áreas: 1. Muestreo, estimación y prueba de hipótesis de la estadística, 2. algoritmos de búsqueda, técnicas de modelamiento y teorías de aprendizaje de la inteligencia artificial, reconocimiento de patrones y aprendizaje de máquina [99].

El objetivo del agrupamiento de documentos es formar una serie de conjuntos (grupos, clases, en inglés ***clusters***) que cumplan las propiedades de [17]:

[24] Snowball `http://snowball.tartarus.org/`
[25] Google `http://www.google.com/`
[26] Wolfram-Alpha `http://www.wolframalpha.com/`
[27] START `http://start.csail.mit.edu/`
[28] Answerbus `http://www.answerbus.com/`

- Homogeneidad dentro de los conjuntos, lo cual significa que los documentos que pertenecen al mismo grupo y deben ser muy similares entre sí.
- Heterogeneidad entre conjuntos, lo cual debe reflejarse con el hecho que cuando los documentos pertenecen a grupos diferentes deben ser muy distintos entré sí.

La definición formal de agrupamiento es la siguiente: dada una colección de "n" objetos descritos por un conjunto de "p" atributos, el objetivo del agrupamiento es derivar una división útil de los "n" objetos en un número de conjuntos. Un conjunto es una colección de objetos similares entre sí, basado en los valores de sus atributos, y puede ser tratado colectivamente como un grupo. El agrupamiento es útil para obtener una distribución interna del conjunto de datos [2].

Los primeros sistemas de agrupamiento de documentos incluían la adaptación de los algoritmos de agrupamiento de datos para el agrupamiento de documentos de texto. Los algoritmos más usados fueron el algoritmo de agrupamiento jerárquico aglomerativo (por sus siglas en inglés, HAC) [89] y el K-Means [99, 46]. Estos algoritmos de agrupamiento utilizan medidas de similitud para calcular la cercanía entre dos objetos del conjunto. Otros algoritmos para agrupamiento de documentos han sido desarrollados basados en la identificación de frases y conceptos que comparten diferentes documentos como es el caso del *Suffix Tree Clustering* (STC) [120], Semantic Online Hierarchical Clustering (SHOC) [121] y LINGO [76].

2.5.1 Medidas de Similitud

Para el cálculo de la similitud entre dos documentos de texto se han planteado diferentes medidas, las más usadas para agrupamiento de documentos son: distancia de Hamming, distancia de edición, similitud del coseno, índice DICE, coeficiente Jaccard y la distancia euclidiana [4]. Strehl en [98] expone un estudio comparativo de diferentes medidas de similitud para documentos; la conclusión obtenida indica que los coeficientes del coseno y Jaccard son los más idóneos que la distancia euclidiana para espacios de gran dimensionalidad y con datos dispersos, como es el caso del agrupamiento de documentos.

Distancia de Hamming

Esta medida requiere que los documentos de entrada sean de l misma longitud. La distancia entre cada uno de estos textos se calcula como el número de posiciones que tienen caracteres diferentes. Dos textos son iguales si su distancia es igual a cero [4].

Distancia de edición

También llamada distancia de Levenstein, se define como el mínimo número de inserciones, eliminaciones y sustituciones de caracteres, que deben hacerse para que un documento sea igual a otro. Por ejemplo la distancia de edición entre la palabra "soñar" y la palabra "sueños" es igual a tres [4].

Coeficiente de Jaccard

Los valores posibles de similitud se encuentran en el rango [0,1]. El coeficiente de Jaccard es calculado con la ecuación 2.4, donde a_j es el j-ésimo documento, t es el número de términos, q es el vector consulta. Este coeficiente toma valores intermedias entre el coeficiente del coseno y la distancia Euclideana, que se describe a continuación [92]

$$Sim(a_{ij}, q) = \frac{a_{ij}^T \bullet q}{\|a_{ij}\|^2 + \|q\|^2 - (a_{ij} \bullet q)} = \frac{\sum_{i=1}^{t} a_{ij} q_i}{\sum_{i=1}^{t} (a_{ij})^2 + (q_i)^2 - (a_{ij} \bullet q_i)} \tag{2.4}$$

Índice DICE

Éste índice es calculado con la ecuación 2.5, donde a_j es el j-ésimo documento, t es el número de términos, q es el vector consulta. La ecuación se reduce notando que C es el número de términos que a_j el j-ésimo documento tiene con el vector consulta$q$, A es el número de términos contenidos en el documento y B el número de términos contenidos en el vector de consulta [83].

$$Sim(a_{ij}, q) = \frac{2\sum_{i=1}^{t} a_{ij}q_i}{\sum_{i=1}^{t}(a_{ij})^2 + \sum_{i=1}^{t}(q_i)^2} = \frac{2C}{A+B} \tag{2.5}$$

Distancia Euclidiana

La distancia euclidiana es usado para calcular la magnitud del segmento que une dos puntos. La ecuación 2.6 es la más utilizada, donde a_j es el j-ésimo documento, t es el número de términos, q es el vector consulta. Los valores posibles para la distancia euclidiana se encuentran en el rango entre [0,1], un documento tiene semejanza igual a 1 sólo consigo mismo. Para un documento cualquiera, el vector que lo representa es un punto en el espacio [92].

$$dis - euc(a_{ij}, q) = \|a_{ij} - q\| \tag{2.6}$$

Similitud del coseno

El cálculo de la similitud del coseno entre la consulta y cada uno de los documentos es el calcular el ángulo entre el vector consulta y cada uno de los vectores de los documentos. Los documentos más paralelos al vector de la consulta serán aquellos que responden mejor a la misma. La representación vectorial de éste cálculo se muestra en la figura 2.15, donde se muestran dos documentos d_1 y d_2, y la similitud de estos documentos y una consulta q es el cálculo del ángulo entre cada vector del documento y el vector de la consulta.

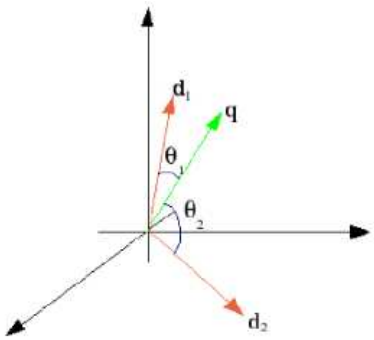


Figura 2.15: Ejemplo de la representación vectorial del cálculo de la similitud del coseno

El cálculo se efectúa utilizando la ecuación 2.7 [94], donde a_j es el j-ésimo documento, t es el número de términos, q es el vector consulta y $\|a\|$ es la magnitud del vector a. Debido a que la multiplicación tanto del vector a_{ij} como del vector q por una constante no modifica el cálculo del coseno, los vectores pueden ser escalados por un valor que sea conveniente. Es frecuente que tanto los vectores de documentos

como los de las consultas sean normalizados a una unidad de longitud

$$Sim(a_j,\, q) = cos(\theta_j) = \frac{a_j^T q}{\|a_j\|\,\|q\|} = \frac{\sum_{i=1}^{t} a_{ij} q_i}{\sqrt{\sum_{i=1}^{t}(a_{ij})^2 \sum_{i=1}^{t}(q_i)^2}} \tag{2.7}$$

2.5.2 *Suffix Tree Clustering*

El algoritmo *Suffix Tree Clustering* (STC) se basa en el agrupamiento de de un conjunto de documentos de texto de acuerdo a las frases idénticas que estos comparten [120]. La idea fue propuesta por primera vez en 1973 por Weiner [108] y las publicaciones posteriores sobre el algoritmo perfeccionaron la idea inicial [67, 103].

El algoritmo STC se fundamenta en el hecho que las frases, a comparación del caso de las palabras claves individuales, tienen mayor poder descriptivo para identificar los conjuntos de documentos. Esto se debe a que se conservan las relaciones de proximidad y orden de las palabras. Las frases se convierten en un factor de ventaja porque se utilizan tanto en la conformación de los conjuntos de documentos como en la descripción de los grupos resultantes [75].

En el algoritmo 2.1 se describe el pseudo-código del algoritmo STC como se plantea en [109], esta descripción no es incremental para facilitar el entendimiento, la versión completa incremental se encuentra descrita en [119]. En la figura 2.16 se describe un ejemplo de la construcción del árbol de sufijos que incluye tres documentos que tienen cada uno una frase:

- Documento A: Gato comió queso
- Documento B: Ratón comió queso también
- Documento C: Gato comió ratón también

Los nodos en la figura 2.16 se etiquetan con un número secuencial y las letras, que corresponden a frases indexadas en subárbol del nodo. Para el nodo, el camino que va a la raíz es "gato se comió" y de la etiqueta de inmediato puede decir que las oraciones contienen esta frase (A y C). Los árboles de sufijos requieren que el último elemento que se inserta sea único (en caso contrario los sufijos pueden terminar en los nodos implícitos, la explicación completa se encuentra en [103, 72]. Para facilitar el entendimiento de este ejemplo se renunció a este supuesto lo que causó el sufijo de la última frase, A - la palabra "queso" - finaliza en el nodo 3, sin que la ramificación. Lo mismo se aplica al nodo 5, que debería haber sido dividido en dos nodos diferentes [109]. Los detalles del procedimiento para la generación completa de un árbol de sufijos se encuentra en [21].

2.5.3 *Semantic Online Hierarchical Clustering*

El algoritmo *Semantic Online Hierarchical Clustering* (SHOC) publicado en [121], se propuso como un algoritmo para el agrupamiento de resultados de consultas web para un sistema de recuperación de información enfocado en idioma chino. Su planteamiento incluye una variación del modelo de espacio vectorial llamado *Latent Semantic Indexing* (LSI) y también incluye un proceso de agrupamiento con frases que es muy diferente a los algoritmos planteados antes de su publicación [75]. De hecho supera ampliamente la calidad de las frases obtenidas con el algoritmo STC al introducir dos conceptos nuevos: frases completas y conjuntos continuos [75]. El algoritmo 2.2 describe el pseudo-código del algoritmo SHOC [121].

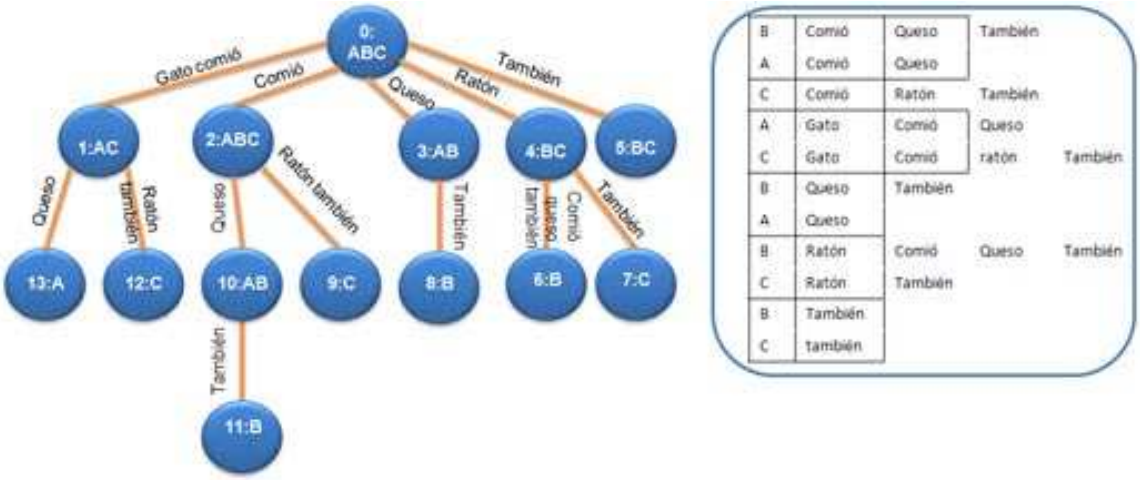


Figura 2.16: Árbol generalizado de sufijos para tres frases [109, 119]

Algoritmo 2.1 Algoritmo *Suffix Tree Clustering*

procedure DIVIDIR TEXTO(*Conjunto de Documentos*)
    Documento ← *Conjunto de frases de palabras*
end procedure
    ▷ Fase 1a. Creación del árbol de sufijos genaralizado para todas las oraciones
for cada documento **do**
    for cada frase **do**
        if longintud de la frase > 0 **then**
            Insertar la frase y todas sus sub-cadenas dentro de un árbol de sufijos generalizado y actualizar los nodos internos con el índice del documento actual mientras se reordena el árbol
        end if
    end for
end for
    ▷ Fase 1b. Construir la lista de los conjuntos iniciales
for cada nodo del árbol **do**
    if número de documentos en el nodo del sub-árbol > 2 **then**
        if puntuación del conjunto base candidato > Mínima puntuación del conjunto base **then**
            Añadir un conjunto base a la lista de conjuntos base
        end if
    end if
end for
    ▷ Fase 2. Combinar los conjuntos base
if número de documentos comunes indexados por A y B > Umbral de combinación **then**
    construir un grafo donde los nodos son los conjuntos base y un enlace entre el nodo A y B los conjuntos son sub-grafos del grafo
end if

2.5.3.1 Frases completas

Este concepto se crea para evitar la extracción de frases parciales que no tienen significado, lo cual ocurre en el idioma chino e idiomas occidentales como el inglés y el español. Dado un documento T de longitud N, S - es la subcadena de T definida como [121]:

> Definición: S es la subcadena completa de T cuando S ocurre en K en diferentes posiciones p_1, p_2, ..., p_k, en T y el $(p_j - 1) - ésimo$ caracter par al menos un par *(i, j)* , $1 \leq i < j \leq k$ (completitud por la izquierda), y el $(p_i + |S|) - ésimo$ caracter es diferente desde el $(p_i + |S|) - ésimo$ caracter por al menos un par de *(i, j)*, $1 \leq i < j \leq k$ (completitud por la derecha).

2.5.3.2 Conjuntos continuos

El concepto de conjunto continuo permite identificar la asociación de los documentos a los conjuntos a través de diferentes intensidades de pertenencia. Debido a esto, la definición se ajusta a la necesidad de la superposición de los conjuntos y también permite el ordenamiento de los documentos dentro de un conjunto. La definición de conjunto continuo es la siguiente [75]:

> Definición: Un conjunto C_g de m objetos t_1, t_2, ..., t_m se define por un vector $m - dimensional$ (vector del conjunto) x_g, cuya norma es igual a uno $|x_g| = 1$. Cada componente de $x_g(i)$ del vector del conjunto representa la intensidad con la que el objeto $t_i$pertenece al conjunto C_g.

2.5.4 LINGO

LINGO es un algoritmo enfocado en el problema del agrupamiento automático de los resultados de consultas de búsqueda como las que generan motores de búsqueda como Yippy[29] (antes llamado Clusty). Los documentos de entradas que procesa son los resúmenes que se presentan al usuario de las páginas y documentos web donde se encuentran las ocurrencias de las palabras o frases de la consulta. Este algoritmo se caracteriza por utilizar agrupamiento conceptual [76], basado en la descomposición en valores singulares de matrices (SVD) y un proceso *a priori* de etiquetamiento de conjuntos. En general, los algoritmos tradicionales de agrupamiento ejecutan un proceso básico que consiste en: primero se descubre el contenido de los conjuntos y luego de acuerdo a este contenido se etiquetan los conjuntos [76]. El problema con los algoritmos clásicos de agrupamiento es la dificultad para encontrar razones que puedan ser comprendidas por los seres humanos del porque los documentos se encuentran agrupados en el mismo conjunto. Por esta razón, LINGO propone un esquema novedoso para resolver el problema de agrupamiento de documentos.

Este algoritmo ejecuta primero un proceso para encontrar las etiquetas humanamente comprensibles con las cuales se identificarán los conjuntos, posteriormente los documentos se asocian a los conjuntos de acuerdo a la similitud de las etiquetas con el contenido de los mismos [76]. El proceso de agrupamiento de LINGO se compone de cinco fases: 1. Preprocesamiento, 2. Extracción de frases frecuentes, 3. Inducción de las etiquetas de conjunto, 4. Descubrimiento del contenido de los conjuntos y 5. Formación final de los conjuntos [76].

[29] Yippy `http://www.yippy.com/`

Algoritmo 2.2 Algoritmo SHOC

```
1: procedure DIVIDIR TEXTO(Conjunto de Documentos)
2:     Documento ← Conjunto de frases de palabras
3: end procedure
       ▷ Fase 1. Descubrir las frases completas
4: procedure CONSTRUIR ARREGLO(Conjunto de Documentos)
5:     Arreglo de sufijos ← Conjunto sufijos del texto
6: end procedure
7: procedure IDENTIFICAR FRASES COMPLETAS(Arreglo de sufijos)
8:     Conjuntodefrasescompletas ← frases completas del arreglo de sufijos
9: end procedure
       ▷ Fase 2. Descubrir los conjuntos base
10: procedure CONSTRUIR UN MATRIZ BASE(Arreglo de sujijos)
11:     Matriz de frases − documentos ← Frases completas del arreglo de sufijos
12: end procedure
13: procedure DESCOMPOSICIÓN SVD(Matriz d frases − documentos)
14:     Matriz SVD ← SVD(Matriz d frases − documentos
15:     Los vectoes singulares derechos de la matriz son los vectores base del conjunto;
16: end procedure
       ▷ Fase 3. Construir la jerarquía de los conjuntos
17: procedure SELECCIONAR CONJUNTOS(Matriz SVD)
18:     Conjunto X ← Primer conjunto representativo base de Matriz SVD
19:     Conjunto Y ← Segundo conjunto representativo base de Matriz SVD
20: end procedure
       ▷ sum(X,Y) denota la suma de los conjuntos X y Y
       ▷ intersec(X,Y) denota la intersección de los conjuntos X y Y
       ▷ tamaño(A) denota el número de elementos del conjunto A
21: if tamaño(sum(X,Y))/tamaño(intersect(X,Y)) > Umbral de Combinación 1 then
22:     combinar X y Y en un nuevo conjunto;
23: else
24:     if tamaño(X) > tamaño(Y then
25:         Y se convierte en hijo de X
26:     else
27:         if size(intersect(X,Y)) / tamaño(X)> Umbral de Combinación 2) then
28:             X se convierte en hijo de Y
29:         end if
30:     end if
31: end if
```

2.5.4.1 Preprocesamiento

Durante el proceso de indexación de los documentos, todos los términos son almacenados en un índice invertido incluyendo las *stop words*, solo las etiquetas HTML son removidas y también son filtradas los objetos que no son caracteres (imágenes, videos, elementos multimedia). Luego las entradas para el algoritmo LINGO son el contenido, la URL y el título de cada documento. El siguiente paso es la identificación del lenguaje de los documentos a través de la identificación y conteo de las *stop words,* las cuales son comparadas con las listas de *stop words* de diferentes lenguajes. El lenguaje es seleccionado de acuerdo al mayor número de *stop words* que coinciden con una lista de *stop words* de un lenguaje específico [76, 75].

El proceso siguiente es la lematización de palabras, en el cual se utiliza el algoritmo de Porter para idioma inglés, y para otros idiomas como el español se utiliza la librería Snowall.

2.5.4.2 Extracción de frases frecuentes

La extracción de frases frecuentes permite identificar las etiquetas de los conjuntos que se formarán con los documentos. Las frases frecuentes son las frases y términos que tienen un alto potencial de tener un significado o ser comprendidas fácilmente por seres humanos, lo cual permite identificar de forma rápida el contenido de los conjuntos [75, 76]. Los detalles completos del algoritmo de identificación de frases se encuentran descritos en [75].

Las frases candidatas tienen las siguientes características [75]: 1. Se presentan en el conjunto de documentos de entrada un mínimo de veces, 2. No traspasa las fronteras oraciones, es decir los lugares donde termina una oración como los puntos seguidos o puntos aparte y 3. No empiezan ni terminan con una *stop word*.

2.5.4.3 Inducción de las etiquetas de conjunto

El objetivo de esta fase es encontrar las descripciones del contenido de cada uno de los conjuntos usando la descomposición en valores singulares. Este proceso incluye cuatro pasos [75]: 1. Construcción de la matriz de términos y documentos. Dicha matriz se construye con el cálculo del peso tf-idf para cada término en cada uno de los documentos, 2. Descubrimiento de los conceptos abstractos, 3. Correspondencia de etiquetas con los conceptos abstractos y 4. Selección de etiquetas de los conjuntos.

2.5.4.4 Descubrimiento del contenido de los conjuntos

Los conjuntos se forman usando el modelo de espacio vectorial, pero LINGO encuentra la correspondencia de los documentos de entrada contra cada una de las etiquetas individualmente.

2.5.4.5 Formación final de los conjuntos

La formación de los conjuntos, se efectúa utilizando un cálculo de relevancia para cada uno de los documentos en cada uno de los conjuntos. Y también se calcula la relevancia de cada uno de los conjuntos a través de la multiplicación de cada una de las etiquetas y el peso hallado en el modelo de espacio vectorial para cada uno de los documentos.

Resumen del capítulo

En este capítulo se desarrolló el estado del arte en librerías digitales de documentos a través de tres temas principales: el concepto de librería digital, las aproximaciones para el desarrollo desde el modelo formal 5S y los enfoques de diferentes tecnologías para recuperación de información, preprocesamiento automático de texto y agrupamiento de documentos.

3 MODELO DE LIBRERÍA DIGITAL DE DOCUMENTOS PARA UN PORTAL WEB

En este capítulo se desarrolla un modelo de una librería digital que puede ser generada a partir de un conjunto de documentos publicados en un portal web. El modelo se aplicó en el portal web de la Universidad Nacional de Colombia, en el cual se encuentran diferentes tipos de documentos que son un activo importante dentro de las actividades diarias de las personas que pertenecen a dicha institución. El sistema es propuesto desde su concepción de funcionamiento lógico por componentes, luego su arquitectura por módulos de software que integra un primer componente para la generación de una estructura de librería digital a través de la recuperación de documentos; un segundo componente que se encarga del proceso de agrupamiento de documentos; el tercer componente que consiste en un módulo para la administración de documentos dentro de la librería digital y finalmente un módulo que permite efectuar búsquedas de los documentos que contiene la librería generada. Para identificar la necesidades de uso del sistema se aplicó la plataforma 5S. El diseño fue creado utilizando el modelo de 4+1 vistas, el cual permite examinar diferentes partes de la arquitectura por separado para facilitar la visualización de la complejidad del sistema. El modelo propuesto fue publicado en [62].

3.1 IDENTIFICACIÓN DE NECESIDADES DE USO

La Librería Digital Universidad Nacional de Colombia (LiD-UN), se propone como un sistema enfocado en la gestión, organización y acceso a las publicaciones digitales de la Universidad en las diferentes sedes (Bogotá, Medellín, Manizales, Palmira, Caribe y Amazonía), facultades (e.g Ingeniería, Economía, Artes, Medicina, Ciencias), Departamentos (e.g. Ingeniería de Sistemas e Industrial, Ingeniería Electrónica), Escuelas (e.g. Cine y televisión, Música), Institutos (e.g IBUN, ICTA, IECO) y todas las dependencias administrativas nacionales, de las sedes, de las facultades y de las carreras.

Dentro de los documentos que son publicados en los diferentes portales web de toda la Universidad se encuentran: páginas informativas relativas a las diferentes instancias de la institución, páginas personales y publicaciones de los miembros de la comunidad (profesores, estudiantes, grupos de investigación, grupos de trabajo, etc.), noticias y documentos relativos a la reglamentación (acuerdos, resoluciones, actas, etc.) en diferentes niveles de la administración nacional, de sede, de facultad, por departamento o escuela, comité curricular, etc.

Para identificar la necesidades de uso del sistema se aplicó la plataforma 5S descrita en la sección 2.2.2, originalmente propuesta en [31]. A continuación se describen las sociedades, escenarios, espacios, flujos y estructuras de la librería digital Lid-UN.

3.1.1 Sociedades

- Estudiantes de pregrado
- Estudiantes de posgrado

- Grupos de investigación
- Egresados
- Profesores
- Personal Administrativo de la Universidad
- Personal encargado de mantener el sistema

3.1.2 Escenarios

Cada una de las comunidades involucradas en la sociedad de la LiD-UN, necesita un servicio particular de la librería digital. Cada uno de estos servicios se relacionan con los documentos digitales y sus respectivos escenarios.

Se identificaron los siguientes servicios básicos:

1. Servicio de Consulta de Documentos: permite consultar los documentos electrónicos publicados.
2. Servicio de Capacitación: tiene como objetivo entrenar a los usuarios del sistema para hacer uso del mismo a través de diferentes estrategias como tutoriales, manuales y contenido de ayuda a lo largo de todos los componentes del sistema.
3. Servicio de Administración: este servicio es el encargado de actualizar periódicamente el sistema con los nuevos documentos publicados, su respectiva clasificación, y el mantenimiento y mejoramiento de los diferentes servicios que componen el sistema.

3.1.3 Espacios

De acuerdo a la ubicación física de los documentos, se establece un espacio físico de acuerdo a la sede, unidad administrativa o académica en la cual son publicados.

Los espacios conceptuales que se usan para el indexación, agrupamiento y clasificación corresponden a la tipología de documentos publicados de acuerdo a la unidad organizacional de la Universidad que lo emite, ó de acuerdo al tema que trata.

En cuanto al uso de la librería digital se puede plantear el espacio de interfaces que tiene el sistema para el funcionamiento de sus diferentes servicios.

3.1.4 Flujos

Debido a la heterogeneidad de documentos que se publican en la Universidad, se identifican varios tipos de flujos: caracteres para los textos, píxeles para las imágenes, imágenes y secuencias de audio para vídeos.

3.1.5 Estructuras

Los documentos que son publicados en los portales web de toda la Universidad, en su mayoría son documentos no estructurados, como lo son las páginas web que utilizan lenguajes de marcas tradicionales como HTML y XML. También se encuentran documentos de texto en diferentes formatos como texto plano, Microsoft Word, PDF, Micrsoft Excel y Microsoft Power Point.

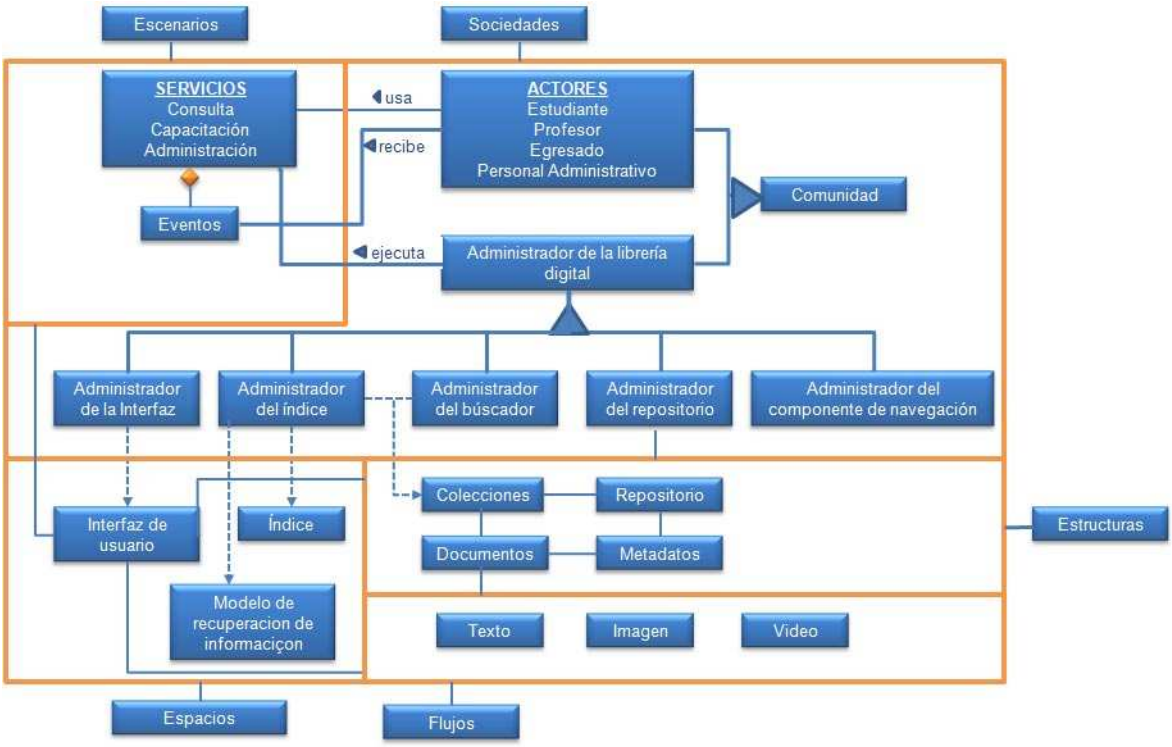


Figura 3.1: Esquema de aplicación de la plataforma 5S a la librería digital Lid-UN

3.1.6 Esquema 5S para LiD-UN

La aplicación de la plataforma 5S a la librería digital Lid-UN, a través de la identificación de sus elementos (sociedades, escenarios, espacios, flujos y estructuras), permitió la construcción del modelo lógico, la arquitectura funcional, el modelo físico y los escenarios de uso. Las sociedades permiten identificar las diferentes comunidades que utilizarán el sistema, los escenarios describen los servicios que proveerá el sistema. Los espacios definen la topología del sistema y la arquitectura funcional del sistema. Los flujos y las estructuras identifican los tipos de documentos y sus correspondientes formatos los cuales son la principal entrada del sistema. En la figura 3.1 se muestra la relación entre cada uno de los elementos de la plataforma 5S aplicados a la librería digital Lid-UN, donde también se muestra la relación con los componentes del diseño del sistema de software.

3.2 DISEÑO DEL SISTEMA

El diseño fue creado utilizando el modelo de 4+1 vistas descrito en [54], el cual permite examinar diferentes partes de la arquitectura por separado para facilitar la visualización de la complejidad del sistema. El modelo del sistema está compuesto por cinco vistas que son: el modelo lógico, la descripción del proceso funcional, la arquitectura del sistema de software, el modelo físico y los escenarios de uso. Además el diseño se basa en componentes de software de código abierto debido a la posibilidad de utilización y modificación de herramientas robustas disponibles bajo este tipo de licencia. Debido al enfoque en desarrollo de software orientado a objetos y el escenario de uso de la herramienta que es la Web, se escogió como plataforma de desarrollo Java J2EE. En la figura 3.2 se muestra una visión general de como se relacionan cada una de las vistas propuestas del sistema.

Figura 3.2: Modelo de 4+1 vistas

3.2.1 Diseño Lógico del Sistema

El diseño lógico describe el sistema en su conjunto como la interacción de diferentes unidades de software que interactúan entre sí pero que no están acoplados, es decir cada componente puede ser desarrollado y/ó cambiado de forma independiente de los demás sin que se afecte el funcionamiento [45]. El diseño propuesto presenta una alta cohesión de funciones de acuerdo a la lógica de cada unidad de software lo cual también favorece el desacoplamiento entre diferentes unidades de software [68]. Se identificaron tres unidades de software básicas [62]:

1. **Sistema de recuperación de información**: Este sistema agrupa los componentes necesarios para el rastreo e indexación de los documentos que están publicados en un portal web y se almacenan en diferentes tipos de formato. El contenido de los documentos se almacena en un índice invertido.

2. **Componente de minería de texto:** Este componente tiene como finalidad el preprocesamiento del contenido textual de los documentos recuperados y su agrupamiento en diferentes conjuntos de contenidos similares. Al finalizar se obtiene un conjunto de colecciones de documentos y los documentos asociados a cada una.

3. **Interfaz de usuario**: Este componente permite la interacción de los usuarios con el sistema. Se propone con un diseño gráfico para su despliegue dentro de una aplicación web, la cual permite que su acceso sea a través de un navegador web de un equipo de cómputo conectado a Internet.

En la figura 3.3, se muestra el diseño lógico del sistema a través del conjunto de componentes que lo integran: sistema de recuperación de información, el componente de minería de texto y la interfaz de usuario [62].

3.2.1.1 Sistema de recuperación de información

El sistema de recuperación de información requerido dentro de la librería digital web de documentos, tiene el funcionamiento básico descrito en la sección 2.3. Los componentes principales que integran el

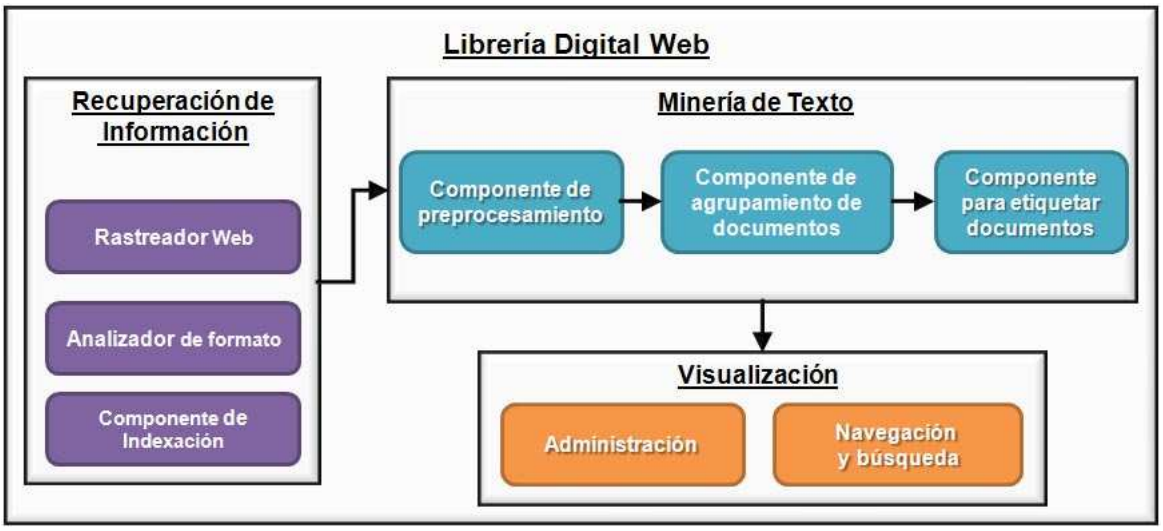


Figura 3.3: Diseño Lógico

sistema de recuperación de información son: rastreador web, analizador de formato y componente de indexación. En este sistema el componente de resolución de consultas se plantea de diferente forma debido al enfoque de agrupamiento de documentos y navegación a través de una jerarquía de categorías.

Existe una amplia oferta de herramientas robustas de recuperación de información disponibles en la Web, por lo cual se optó por la selección de un buscador web que integra los componentes requeridos de recuperación de información. Para efectuar dicha selección se analizó la información sobre los motores de búsqueda de código abierto disponible de las páginas web de los proyectos a los que pertenecen, así como de los artículos publicados por los grupos responsables de dichos motores de búsqueda y se evaluó el rendimiento de los motores de búsqueda que más se ajustaban a las necesidades del sistema [63]. La metodología propuesta de selección se publicó en [63].

El motor de búsqueda seleccionado fue Nutch, el funcionamiento de este buscador web de código abierto se describió en la subsección 2.3.6. Los resultados de la evaluación y proceso de selección del sistema de recuperación de información se describen en la sección 4.1.1.

Rastreador Web

Este componente tiene como objetivo recorrer las direcciones URL de los documento publicados en el portal web, estas direcciones corresponden a un dominio específico y se convierte en la entrada base para la construcción de las listas de recuperación (Ver la subsección 2.3.1, donde se describe el funcionamiento básico de un rastreador web). A través de las listas de recuperación, el rastreador web gestiona la descarga de los documentos en el servidor donde están publicados. Las dos salidas que tienen dichos procesos son: por un lado son las actualizaciones de la base de datos web, que contendrán las páginas que han aparecido o desaparecido desde el último intento de búsqueda; la segunda salida son los documentos en su formato original que son enviados a los analizadores de formato.

Base de Datos Web

La base de datos web, como se mención en la subsección 2.3.6 permite el almacenamiento de la estructura que conforman las URL y los hiper-vínculos presentes en el contenido de los documentos, se guarda la representación del enlace de un documento origen a otro objetivo.

Analizadores de formato

Como se describió en la subsección 2.3.6, este componente se encargan de extraer el contenido de los documentos, dicho contenido es usado para generar el índice invertido. Nutch cuenta con analizadores para diferentes tipos de archivo texto plano, HTML, JavaScript, Microsoft Excel, Microsoft Word, Open Office, PDF, RSS, SWF y ZIP [101].

Componente de indexación

En el sistema propuesto el componente de indexación lo integran cuatro elementos presentes en el buscador Nutch que son: constructor del índice de documentos utilizando la librería Lucene, el índice invertido en formato Lucene, los índices de los segmentos y la base de datos web.

1. **Constructor del índice de documentos**: La forma como Nutch crea un índice de documentos con la librería Lucene se describió en la en la subsección 2.3.6.
2. **Índices de segmentos:** estos índices intermedios, almacenan el contenido extraído por lo analizadores de formato. Se les llama índices intermedios porque se crean a medida que el rastreador web recorre el sitio o un conjunto de sitios.
3. **Índice invertido:** este índice es el resultado de la ejecución del constructor del índice de documentos tomando como entrada los índices de segmentos. Está en formato Lucene, y por tanto utiliza el concepto de "documento" el cual se compone por "campos" de texto [40]. En el índice se almacenaron dos partes de los documentos originales, el contenido y los metadatos. Al final cada documento encontrado en el portal web se almacena en el índice en forma de un "documento" que tiene los siguientes campos: contenido, y los campos de metadatos que son la URL, el título y el autor.

3.2.1.2 Componente de minería de texto

El componente de minería de texto requerido para el sistema propuesto en este trabajo de tesis, debe cumplir con las siguientes características:

1. Algoritmo de agrupamiento y etiquetamiento automático de documentos de texto.
2. Debido al contexto de prueba, el algoritmo debe procesar documentos en idioma español e inglés.
3. Las etiquetas generadas por el algoritmo deben ser fácilmente entendibles y adecuados de acuerdo al contexto en el cual están publicados los documentos de texto.

Debido a las características listadas anteriormente, se optó por una revisión del estado del arte de los algoritmos disponibles para agrupamiento de texto en [46, 53, 9]. Se encontraron varios estudios de investigación enfocados en la comparación de rendimiento de los algoritmos clásicos de agrupamiento de documentos en [118, 120, 119, 122].

Estudios más recientes muestran varios desarrollos en algoritmos enfocados en el agrupamiento de resultados de consultas, dentro de los cuales se encontró el STC y LINGO, descritos en las subsecciones 2.5.2, 2.5.3 y 2.5.4. Estos algoritmos se enfocan en análisis semánticos, es decir identifican relaciones de significado entre los términos de diferentes documentos con lo cual también se identifican frases que representan los conceptos más relevantes que se incluyen en dichos documentos. Este enfoque semántico permite la selección automática de etiquetas a cada uno de los conjuntos de documentos encontrados de forma que sean más entendibles para las personas, lo cual con otros algoritmos solo era posible a través de técnicas supervisadas por expertos.

La selección de un algoritmo de agrupamiento de documentos se hizo a partir de la comparación de las características básicas de los algoritmos disponibles para agrupamiento automático de documentos de texto, esta comparación se resumen en la tabla 3.1. De este análisis se pudo observar que los algoritmos LINGO, STC y SHOC tienen ventajas con respecto a los demás algoritmos en cuanto al etiquetamiento de los conjuntos de documentos que se generan. LINGO (ver descripción completa en la sección 2.5.4) se seleccionó como la mejor opción debido a su enfoque multilenguaje y también por las mejoras que introdujo a partir de los algoritmos STC y SHOC. LINGO tiene esquema novedoso para la identificación de conjuntos de documentos relacionados a través de la identificación de conceptos presentes en los documentos los cuales son representados por las frases frecuentes y que se convierten en las etiquetas de cada uno de los conjuntos. Este proceso de inducción de etiquetas se enfoca en que sean humanamente comprensibles, es decir ofrece al usuario del sistema un esquema de clasificación de documentos que es fácil de entender. Además LINGO a demostrado un buen desempeño en colecciones de documentos de tamaños medianos [77] lo cual es aplicable a los portales web de instituciones como una universidad o sus dependencias.

Para la implementación del componente de minería de texto del sistema propuesto se seleccionó la plataforma de agrupamiento de resultados de búsqueda llamado Carrot2[1], la cual es un sistema de código abierto disponible en la Web. Se seleccionó esta plataforma debido a los componentes robustos que lo conforman, entre ellos el algoritmo de agrupamiento LINGO que permiten la ejecución de procesos de minería de texto desde el pre-procesamiento de los documentos, la creación de la matriz de términos y documentos, descomposición SVD y el agrupamiento de documentos basado en conceptos. Carrot2 ha sido probado en un meta-buscador disponible en la Web donde presenta el agrupamiento de resultados de búsquedas de diferentes motores.

3.2.1.3 Interfaz de usuario

Esta interfaz permite dos funciones básicas, la primera es la administración del sistema a través del acceso a las funciones de configuración de cada uno de los componentes. La segunda función es ofrecer al usuario una aplicación de navegación dentro del conjunto de colecciones de documentos de la librería digital web.

Interfaz de navegación

Esta interfaz es que se muestra a los usuarios que usarán la librería digital web de documentos para navegar sobre una colección de documentos. Para facilitar el proceso de navegación, se propone una interfaz en forma de árbol jerárquico que se construye a partir de las categorías de documentos y los documentos asociados a cada una de las categorías. En la figura 3.4, se muestra la composición de la interfaz de navegación con dos tipos de componentes de navegación que son el árbol de categorías y un grafo de los conjuntos y sus relaciones con los documentos que pertenecen a cada uno y su relación entre documentos. En la figura 3.5 se muestra en detalle el componente de árbol de categorías, donde cada nodo principal es una categoría, y sus nodos asociados son los documentos asociados a la categoría. Se muestra tanto el nombre del documento como su URL. A través de un clic en la URL se abre en el navegador web la página o documento original. En la figura 3.6, se muestra el grafo de categorías y documentos, donde las categorías se identifican por el color verde.

Interfaz de administración

La interfaz de administración es la vista del sistema que permitirá la configuración del sistema para la ejecución de los procesos necesarios para la creación del índice de documentos y el agrupamiento en colecciones de documentos. Además de la actualización periódica de dichos procesos. En la figura

[1] Carrot2 `http://project.carrot2.org/`

Algoritmo	Calidad de los conjuntos	Etiquetamiento de conjuntos	Requerimientos de memoria RAM	Tiempo de procesamiento	Tipo de procesamiento
LINGO	Buena	Automático	Directamente proporcional al tamaño del conjunto de documentos	Directamente proporcional al tamaño del conjunto de documentos	Cálculo sobre matrices de gran tamaño
STC	Buena	Automático	Directamente proporcional al tamaño del conjunto de documentos	Directamente proporcional al tamaño del conjunto de documentos	Recorrido de árboles de sufijos
SHOC	Buena	Automático	Directamente proporcional al tamaño del conjunto de documentos	Directamente proporcional al tamaño del conjunto de documentos	Cálculo sobre matrices de gran tamaño
Algoritmo K-Means	Buena	Requiere adicionar un método supervisado	Depende de los parámetros seleccionados	Depende de los parámetros seleccionados	Cálculo sobre matrices de gran tamaño
Algoritmos de agrupamiento jerárquico aglomerativo	Buena	Requiere adicionar un método supervisado	Depende de los parámetros seleccionados	Depende de los parámetros seleccionados	Cálculo sobre matrices de gran tamaño
Algoritmos de agrupamiento particionales	Buena	Requiere adicionar un método supervisado	Depende de la cantidad de los parámetros seleccionados	Depende de los parámetros seleccionados	Depende del patrón de partición.

Table 3.1: Comparación de algoritmos de agrupamiento de documentos de texto

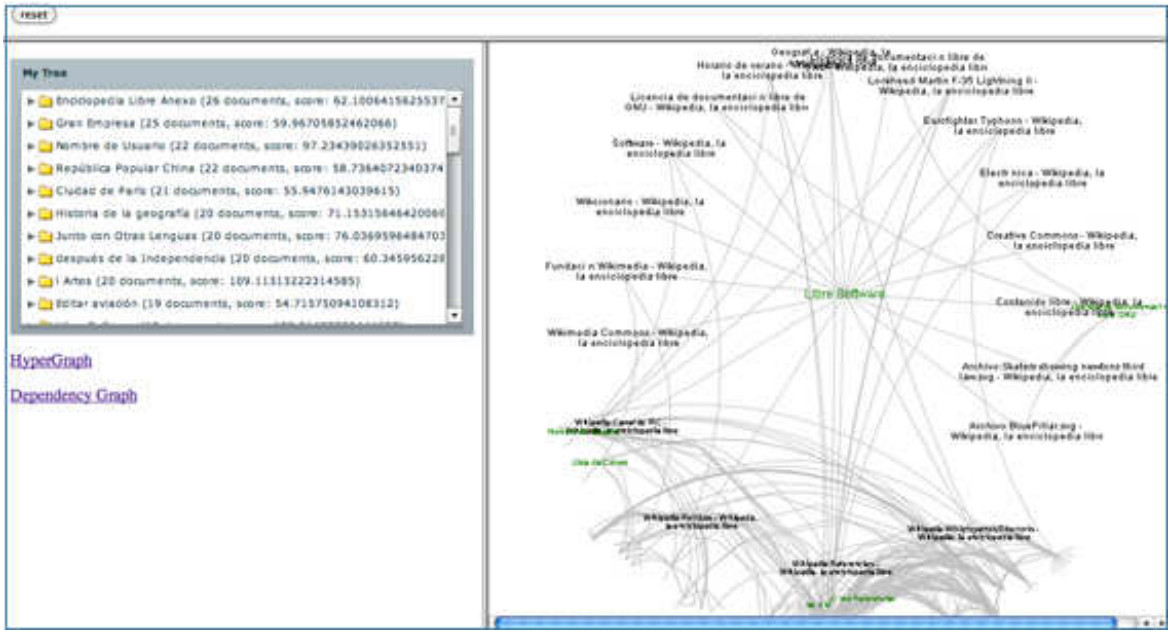


Figure 3.4: Interfaz de navegación de Lid-UN

Figure 3.5: Árbol de categorías de documentos

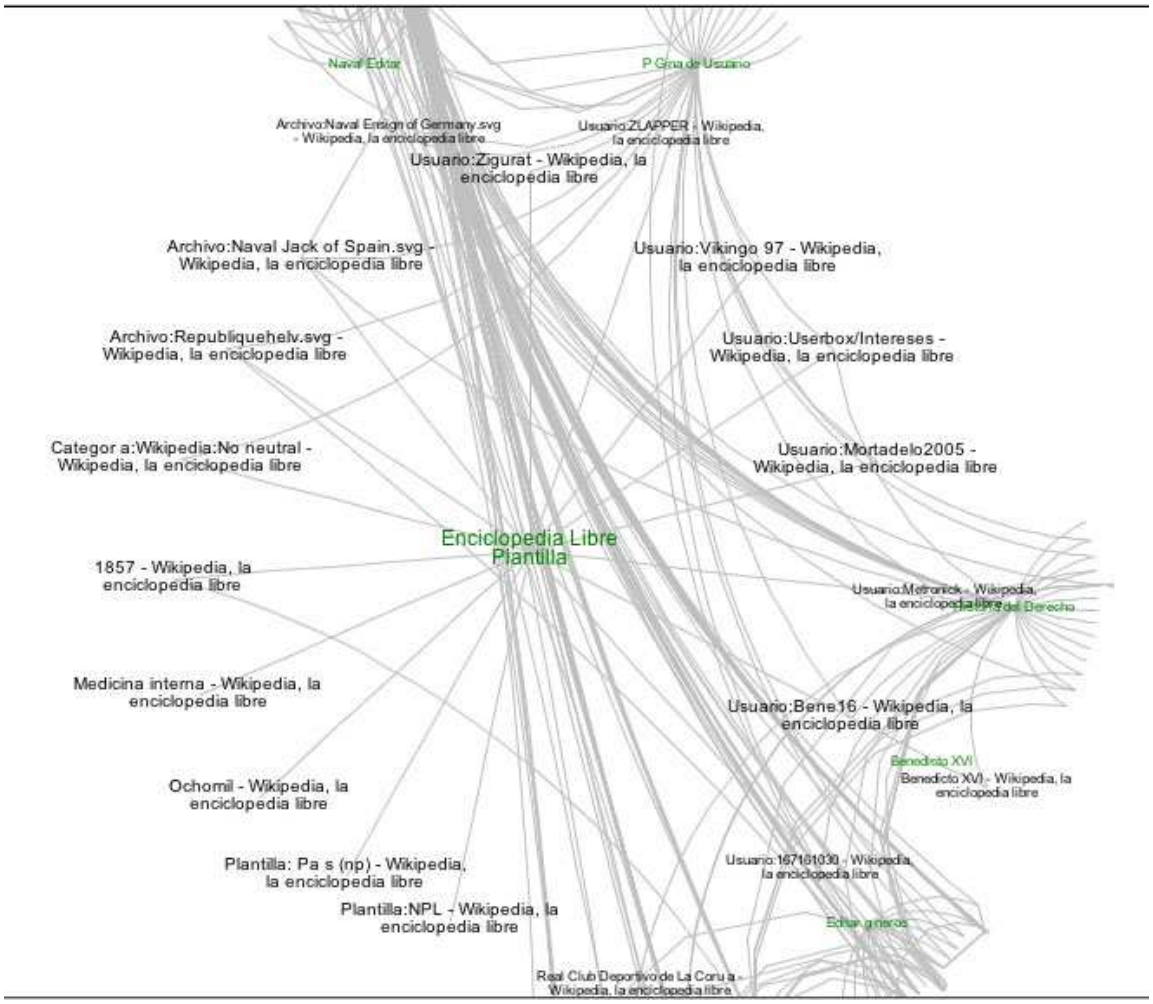


Figure 3.6: Grafo de categorías y documentos

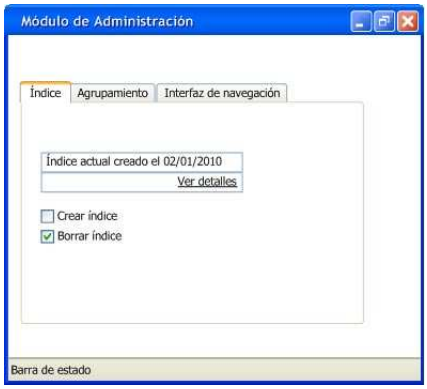
3.7, se muestra el diseño del módulo de administración que incluye tres pestañas, la primera es para la administración del índice de documentos en la figura 3.7a, la segunda para ejecutar el proceso de agrupamiento de documentos en la figura 3.7b, y en la tercera para configurar la interfaz de navegación en la figura 3.7c.

3.2.2 Proceso de Funcionamiento del Sistema

En la figura 3.8 se muestra el proceso de funcionamiento del sistema que incluye los diferentes repositorios de documentos y sus respectivas transformaciones a través del funcionamiento de los componentes del sistema. Se ejecutan tres etapas básicas:

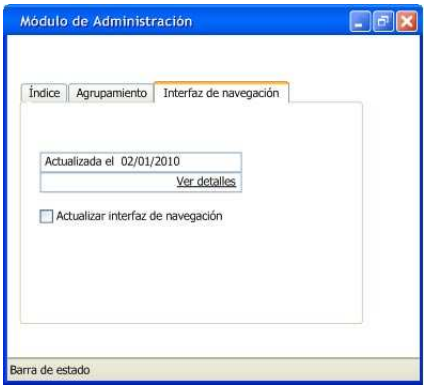
1. Los documentos del portal web son rastreados e indexados con lo cual se genera un índice invertido con el contenido de los documentos y sus metadatos (título, URL y autor).

2. A partir del índice invertido se construye la matriz de palabras y términos, la cual es la entrada para el algoritmo de agrupamiento conceptual, LINGO.



(a) Opciones para la administración del índice de documentos

(b) Opciones para ejecutar el proceso de agrupamiento de documentos



(c) Configuración de la interfaz de navegación

Figure 3.7: Módulo de administración

3. Las categorías de documentos encontradas con LINGO y los documentos asociados a cada categoría son almacenados en una estructura de almacenamiento que es utilizada para la generación de la interfaz de navegación.

3.2.3 Arquitectura Funcional del Sistema

La arquitectura funcional del sistema se basa en el modelo cliente-servidor y se descompone en las diferentes capas del sistema en el cliente, el servidor de aplicaciones y el servidor de bases de datos. Se usó esta clase de arquitectura porque ofrece ventajas en cuanto al control central de la aplicación, la escalabilidad del número de clientes y servidores y el mantenimiento se facilita en la medida que los cambios y actualizaciones del sistema afectan a los servidores y no a los clientes [26].

En la figura 3.9 se muestra la visión general del sistema con su descomposición en las capas de: presentación, lógica, datos y plataforma.

3.2.4 Modelo físico

El modelo físico del sistema permite describir la topología del sistema dentro de una red institucional. En la figura 3.10, se muestra la descripción del modelo físico donde se identifican dos divisiones básicas

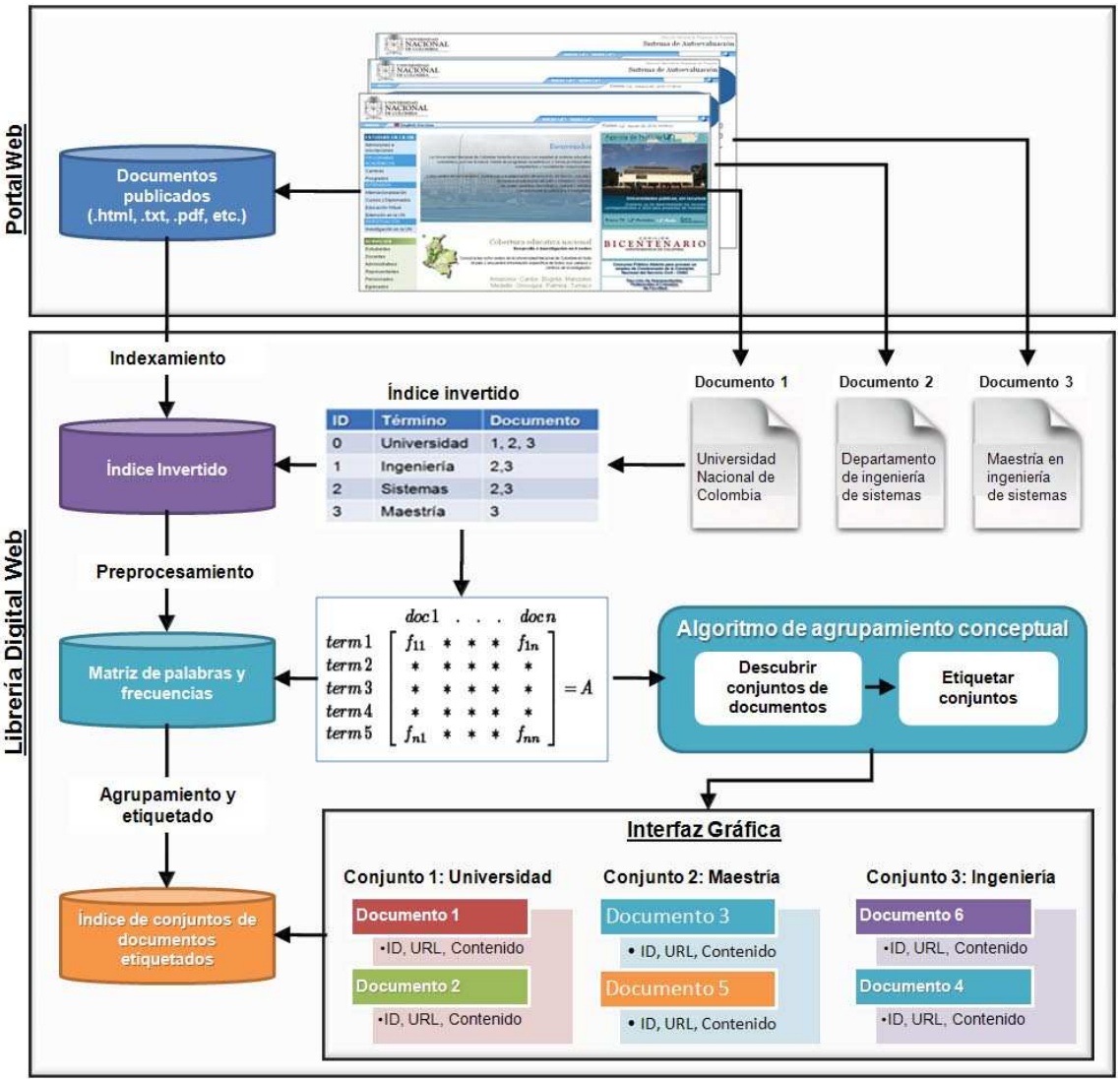


Figura 3.8: Proceso de funcionamiento del sistema

dentro de una organización, la primera es un centro de cómputo central y la segunda son los edificios de las divisiones particulares de la organización que pueden ser institutos, centros de investigación, etc. La topología se caracteriza porque el centro de cómputo agrupa muchos de los componentes de administración principales de la red y también servidores de las bases de datos, de aplicaciones y de administración. En los edificios, es frecuente encontrar servidores con los portales de las diferentes dependencias de una organización.

La topología del sistema permite identificar las necesidades de acceso al contenido público que está disponible en los portales web de cada una de las instancias de una organización. El sistema al utilizar un rastreador web, el cual es un robot es decir un programa de software que accede rápidamente a las páginas web debe tener el permiso adecuado para poder ejecutar sus tareas de rastreo.

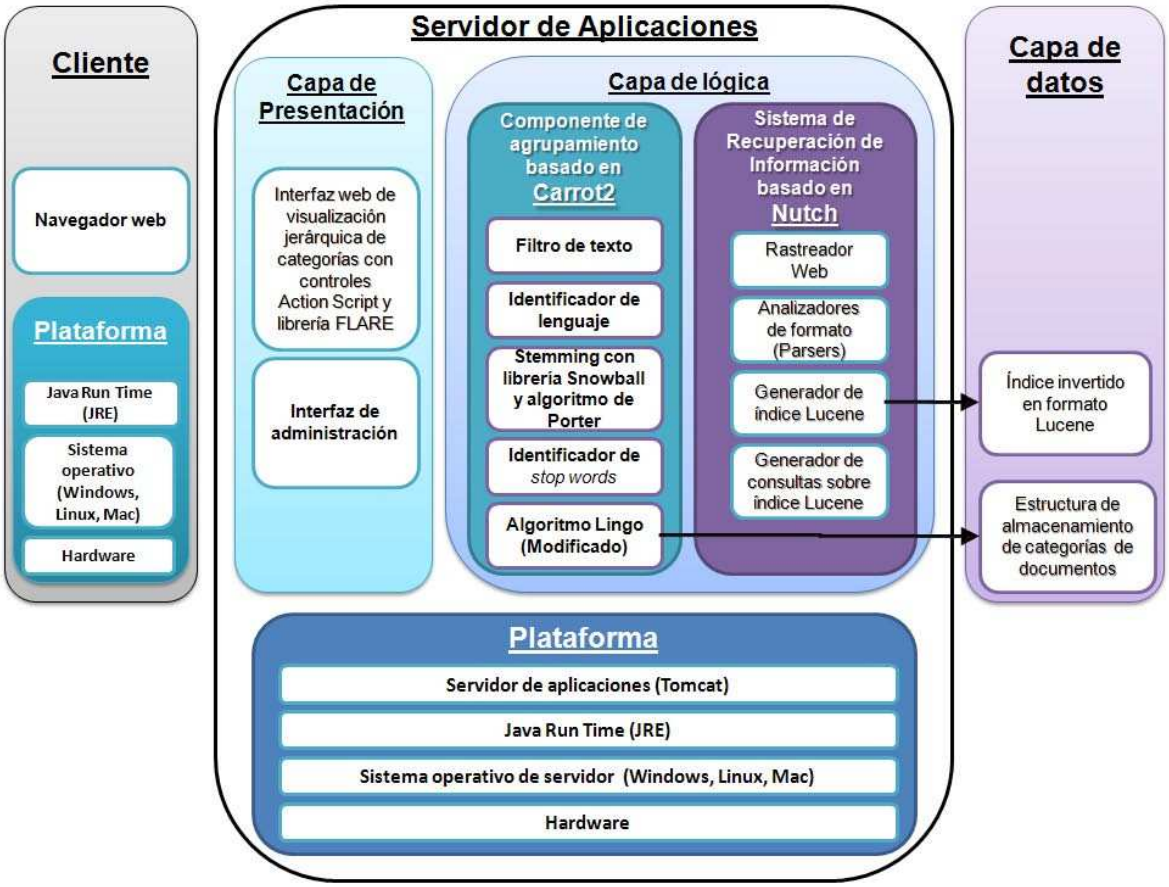


Figura 3.9: Diagrama de arquitectura de Capas

3.2.5 Escenarios de Uso

Los escenarios de uso se describirán a través de casos de uso diseñados con lenguaje UML (*Unified Modeling Language*)[2]. La ventaja de este tipo de diagramas es que permiten la validación del sistema. En la figura 3.11, se describen los casos de uso asociados al sistema propuesto y la interacción de los actores que intervienen.

3.2.5.1 Instalar e iniciar el sistema para primer uso

- Descripción: Permite instalar el sistema dentro del servidor y se configuran los parámetros para iniciar su uso.
- Actores: Usuario administrador
- Flujo Normal:

  Sistema Windows

  1. Iniciar ejecutable para instalar los componentes y crear las carpetas necesarias.

[2] El Unified Modeling Language ™(UML), es la especificación más utilizada del OMG (*Object Management Group*), que propone la forma de diferentes modelos para la descripción del diseño, comportamiento, y la arquitectura de aplicaciones de software, y también de procesos de negocio y estructura de datos.`http://www.uml.org/`

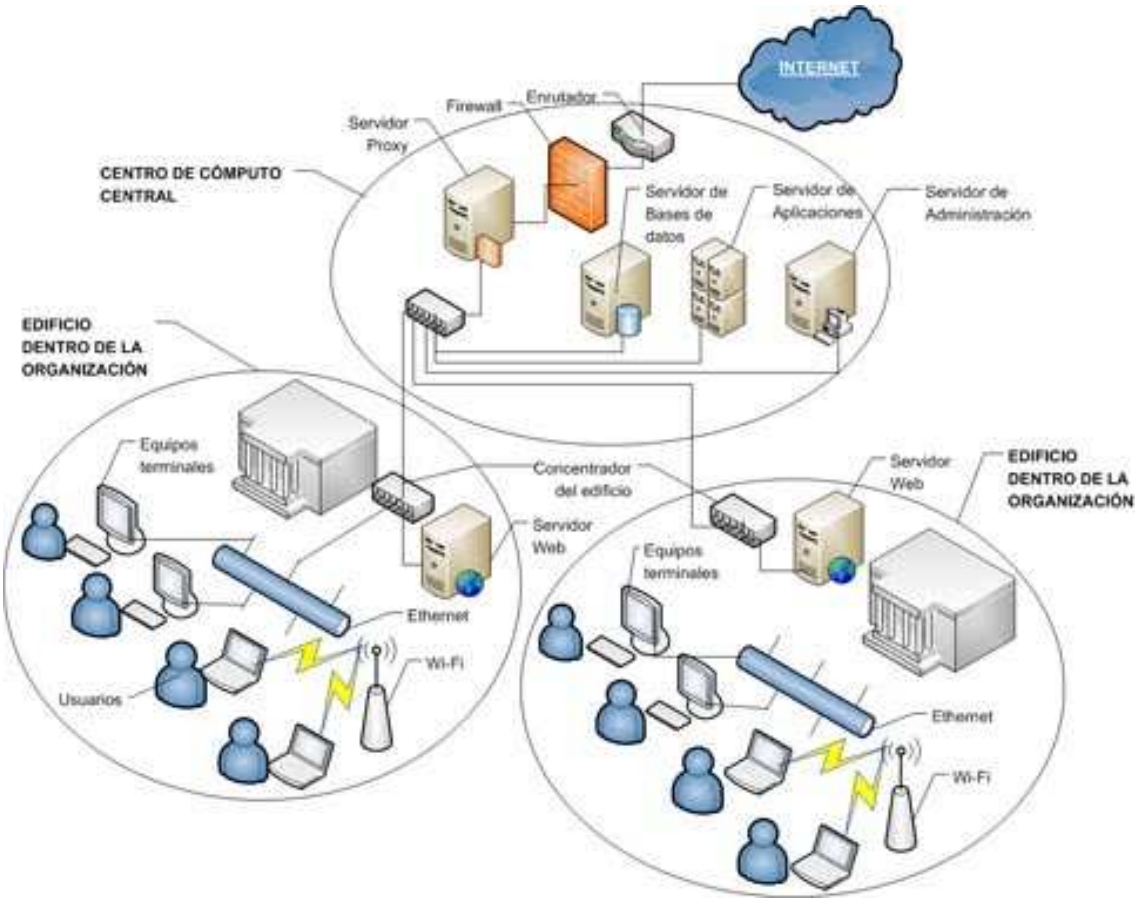


Figura 3.10: Modelo Físico del sistema

2. Caso de uso "Registrar el usuario administrador".

- Flujo alternativo: 1, 2 Salir del programa de instalación.
- Poscondiciones: Inicia el flujo para el registro de usuario administrador.

3.2.5.2 Registrar el usuario administrador

- Descripción: Permite registrar el nombre y contraseña del usuario administrador.
- Actores: Usuario administrador
- Precondiciones: El flujo de instalación de la aplicación comenzó con éxito.
- Flujo Normal:

1. Solicitar nombre, contraseña y verificación de contraseña del usuario administrador. Se verifica que sea una clave segura:
   - Tiene ocho caracteres como mínimo.
   - No contiene el nombre de usuario, el nombre real o el nombre de la empresa.
   - No contiene una palabra completa.

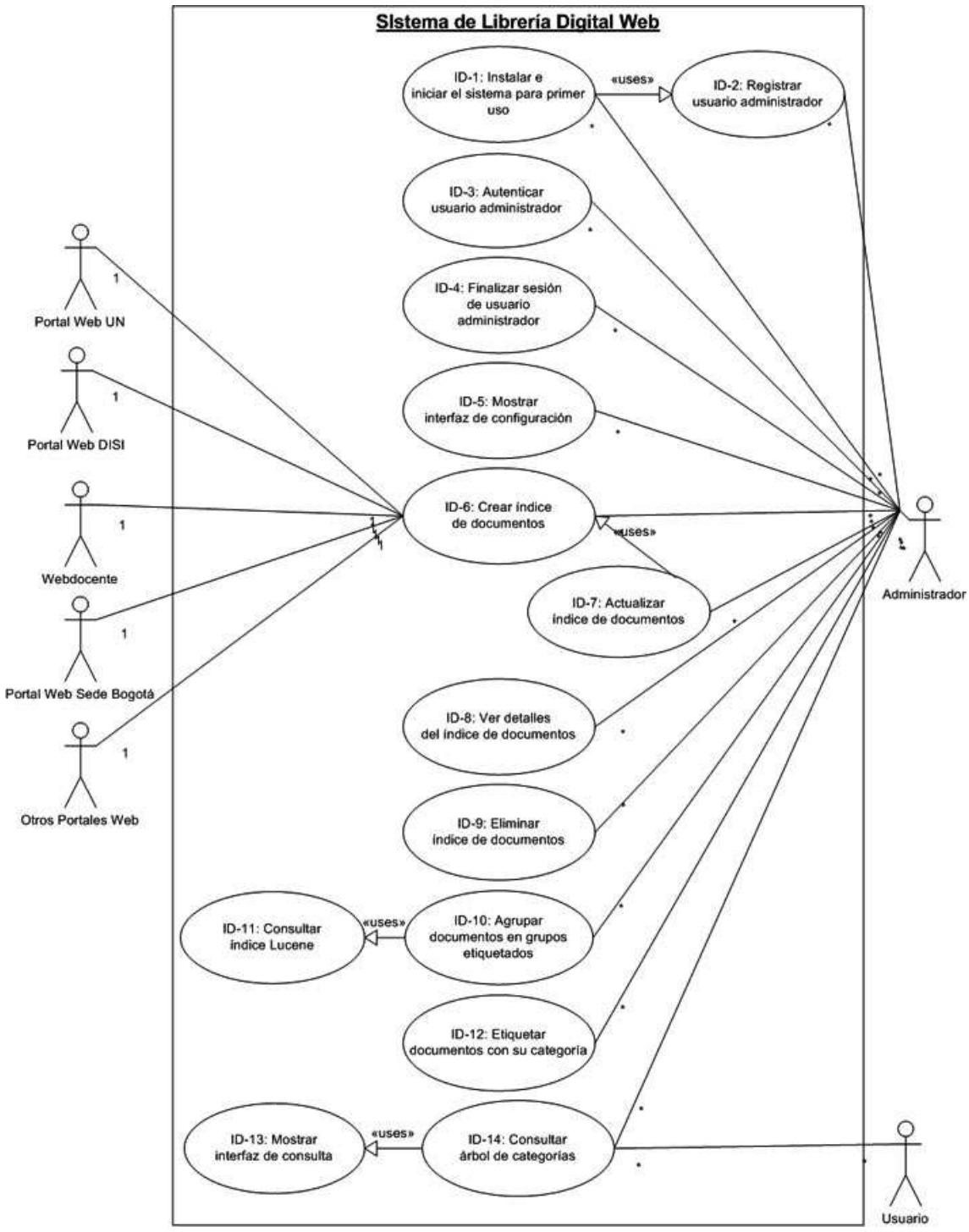


Figura 3.11: Diagrama de Casos de Uso

- Es significativamente diferente de otras contraseñas anteriores.
- Está compuesta por caracteres de cada una de las siguientes cuatro categorías: letras mayúsculas (e.g. A, B, C), letras minúsculas (e.g. a, b, c), números (0, 1, 2, 3, 4, 5, 6, 7, 8, 9), símbolos del teclado (todos los caracteres del teclado que no se definen como letras o números) y espacios (‘ ~ ! @ # $ % ^ & * ( ) _ - + = { } [ ] \ | : ; " ’ < > , . ? /).

2. Salir del flujo de instalación.

- Flujo Alternativo: 1. El sistema comprueba la validez de los datos de nombre de usuario y contraseña, si no son válidos solicita de nuevo el nombre de usuario, contraseña y verificación de contraseña del usuario administrador.
- Poscondiciones: Mensaje instalación completada.

3.2.5.3 Autenticar usuario administrador

- Descripción: Permite autenticar al usuario administrador a través de un nombre de usuario y contraseña.
- Actores: Usuario administrador
- Precondiciones: El usuario debe haberse registrado durante la instalación del sistema.
- Flujo Normal:

1. El actor pulsa sobre la opción “Administrador”.
2. El sistema muestra la interfaz para ingresar datos de autenticación: nombre de usuario y contraseña.
3. El sistema verifica que los datos sean válidos.
4. El sistema muestra la interfaz de configuración.

- Flujo Alternativo: 2. El sistema comprueba la validez de los datos, si los datos no son correctos, se avisa al actor de ello permitiéndole que los corrija
- Poscondiciones: Mensaje de bienvenida

3.2.5.4 Finalizar sesión de usuario administrador

- Descripción: Permite finalizar la sesión de usuario administrador a través de un botón.
- Actores: Usuario administrador
- Precondiciones: El usuario administrador debe haber iniciado una sesión.
- Flujo Normal:

1. El actor pulsa sobre la opción “Administrador”.
2. El sistema muestra la interfaz para ingresar datos de autenticación: nombre de usuario y contraseña.
3. El sistema verifica que los datos sean válidos. 4.El sistema muestra la interfaz de configuración.

- Flujo Alternativo: 3. Si los datos no son correctos, se avisa al actor con un mensaje y se muestra de nuevo el formulario para ingresar el nombre de usuario y la contraseña.
- Poscondiciones: Mensaje de bienvenida

3.2.5.5 Mostrar interfaz de configuración

- Descripción: Permite iniciar la interfaz de configuración de todo el sistema.
- Actores: Usuario administrador
- Precondiciones: El usuario administrador debe haber iniciado una sesión.
- Flujo Normal:

  1. El actor pulsa sobre la opción "Configuración".
  2. El sistema muestra dos menús con las opciones para: crear índice, actualizar índice, ver detalles del índice actual, eliminar índice, crear árbol de categorías, modificar árbol de categorías y borrar árbol de categorías

- Flujo Alternativo: El usuario termina la sesión y no habilita ninguna de las opciones.
- Poscondiciones: Se habilita una de las opciones.

3.2.5.6 Crear índice de documentos

- Descripción: Permite crear el índice de los documentos web que son encontrados por el rastreador web.
- Actores: Usuario administrador
- Precondiciones: 1. El usuario administrador debe haber iniciado una sesión. 2. El actor pulsa sobre la opción "Crear índice".
- Flujo Normal:

  1. Se configuran los archivos: urls/dirección.txt y conf/crawl-urlfilter.txt con la dirección del portal a indexar.
  2. Se configuran los parámetros del rastreador web: profundidad y número de páginas.
  3. Verificar que el proceso de indexación se realizo correctamente. Mostrar mensaje de terminación exitosa o falla.

- Flujo Alternativo:
- Poscondiciones: Se crea el índice para búsqueda.

3.2.5.7 Actualizar índice de documentos

- Descripción: Permite actualizar el índice de búsqueda
- Actores: Usuario administrador
- Precondiciones:

  1. El usuario administrador debe haber iniciado una sesión.
  2. Existe un índice creado con éxito.
  3. El actor pulsa sobre la opción "Actualizar índice".

- Flujo Normal:

1. Mostrar mensaje: “El índice creado en la fecha dd/mm/aaaa, se actualizará”, mostrar opciones: Aceptar y Cancelar.
2. Si se elige la opción “Aceptar”, se ejecuta el script de actualización.
3. Verificar que el proceso de indexación se realizo correctamente. Mostrar mensaje de terminación exitosa o falla.

- Flujo Alternativo: 4. Si se elije la opción “Cancelar”, se muestra de nuevo la interfaz de configuración.
- Poscondiciones: Se actualiza el índice de búsqueda.

3.2.5.8 Ver detalles del índice de documentos actual

- Descripción: Permite visualizar los detalles del índice de búsqueda
- Actores: Usuario administrador
- Precondiciones:

1. El usuario administrador debe haber iniciado una sesión.
2. Existe un índice creado con éxito.
3. El actor pulsa sobre la opción “Ver detalles del índice actual”.

- Flujo Normal:

1. Mostrar mensaje: “Detalles del índice creado en la fecha dd/mm/aaaa”, los detalles del índice a mostrar son: Ubicación del índice, Nombre del índice, Número de campos del índice, Número de documentos indexados, Número de términos, Última modificación: dd/mm/aaaa, hora.
2. Mostrar opción: Cerrar.

- Flujo Alternativo: 2. Si se elije la opción “Cerrar”, se muestra de nuevo la interfaz de configuración.
- Poscondiciones: No hay cambios en el índice

3.2.5.9 Eliminar índice de documentos

- Descripción: Permite eliminar un índice de búsqueda
- Actores: Usuario administrador
- Precondiciones:

1. El usuario administrador debe haber iniciado una sesión.
2. Existe un índice creado con éxito.
3. El actor pulsa sobre la opción “Eliminar índice”.

- Flujo Normal:

1. Mostrar mensaje: “El índice creado en la fecha dd/mm/aaaa, se eliminará”, mostrar opciones: Aceptar y Cancelar.

2. Si se elige la opción “Aceptar”, se elimina el contenido índice.
3. Verificar que el proceso de indexación se realizo correctamente. Mostrar mensaje de terminación exitosa o falla.

- Flujo Alternativo: 4. Si se elije la opción “Cancelar”, se muestra de nuevo la interfaz de configuración.
- Poscondiciones: Se actualiza el índice de búsqueda.

3.2.5.10 Agrupar documentos para identificar etiquetas de los grupos de documentos

- Descripción: Aplicar el algoritmo de agrupamiento al índice de documentos con el objetivo de identificar los grupos de documentos y sus respectivas etiquetas.
- Actores: Usuario administrador
- Precondiciones:
  1. El usuario administrador debe haber iniciado una sesión.
  2. Existe un índice creado con éxito.
  3. El actor pulsa sobre la opción “Crear árbol de categorías”.
- Flujo Normal:
  1. Mostrar mensaje: “Se creará el árbol de categorías sobre el índice de documentos creado en la fecha dd/mm/aaaa”, mostrar opciones: Aceptar y Cancelar.
  2. Si se elige la opción “Aceptar”, se ejecuta el algoritmo de agrupamiento sobre el índice seleccionado
  3. Verificar que el proceso de agrupamiento terminó exitosamente. Mostrar mensaje de terminación exitosa o falla.
  4. Mostrar el árbol de categorías en forma de árbol.
- Flujo Alternativo: 2. Si se elije la opción “Cancelar”, se muestra de nuevo la interfaz de configuración.
- Poscondiciones: Se crea el árbol de categorías.

3.2.5.11 Etiquetar documentos con la categoría a la que pertenecen

- Descripción: De acuerdo a los grupos obtenidos para formar el árbol de categorías, etiquetar cada documento en el índice con la categoría a la que pertenecen.
- Actores: Usuario administrador
- Precondiciones: 1. El usuario administrador debe haber iniciado una sesión. 2. Existe un índice creado con éxito. 3. El árbol de categorías de documentos se creó correctamente.
- Flujo Normal:
  1. Mostrar mensaje: “Se etiquetarán los documentos de acuerdo a la categoría a la que pertenecen”.

2. De acuerdo a los resultados obtenidos al finalizar la ejecución del algoritmo de agrupamiento, etiquetar cada documento en un nuevo campo del índice de documentos con la categoría a la que pertenece.
3. Mostrar el árbol de categorías en forma de árbol, en los nodos de categorías al dar clic en ellos se muestren en forma de hojas los documentos que pertenecen a la categoría.

- Flujo Alternativo:
- Poscondiciones: Se etiquetan los documentos con sus respectivas categorías.

3.2.5.12 Mostrar interfaz de consulta

- Descripción: A través del navegador de internet, el usuario abre la dirección donde se encuentra la aplicación.
- Actores: Usuario
- Precondiciones: 1. El usuario abre una ventana en el navegador de internet y digita correctamente la dirección.
- Flujo Normal:

1. La interfaz de consulta que se muestra se compone del árbol de categorías con los documentos asociados a cada categoría, y un campo de texto para digitar una palabra o frase de búsqueda.

3.2.5.13 Consultar árbol de categorías

- Descripción: En la ventana del navegador de Internet se muestra el árbol de categorías de documentos el cual puede ser consultado a través de la explicación de cada uno de las categorías para abrir los documentos que se asocian a cada una.
- Actores: Usuario
- Precondiciones: 1. El árbol de categorías se muestra en la interfaz.
- Flujo Normal:

1. El usuario selecciona una categoría directamente en el árbol de categorías y abre el documento de interés, el documento se visualiza en una nueva pestaña o nueva ventana en el navegador.
2. El usuario cierra una categoría y consulta otra categoría.
3. Flujo Alternativo: 2. El usuario finaliza su consulta y cierra la ventana.

Resumen del capítulo

En este capítulo se presenta la descripción del modelo propuesto para el desarrollo de una librería digital web a partir de los documentos publicados en un portal web. El modelo se describe utilizando el modelo 4+1 vistas, donde se muestran diferentes aspectos del sistema el diseño lógico, la arquitectura funcional, el modelo físico y los escenarios de uso. El modelo de librería digital web propuesto se enfoca en los portales web organizacionales, para proveer una interfaz de navegación dentro de un conjunto de colecciones de documentos.

4 APLICACIÓN, VALIDACIÓN Y ANÁLISIS DE RESULTADOS

En este capítulo se describe el proceso de evaluación del sistema de librería digital web Lid-UN, dentro del portal web del Departamento de Ingeniería de Sistemas e Industrial de la Universidad Nacional de Colombia. El sistema se validó a través de la evaluación de sus componentes principales: el sistema de recuperación de información, el componente de minería de texto y la interfaz de usuario de la librería digital. El objetivo principal de la evaluación del sistema consistió en la verificación del funcionamiento de cada uno de los componentes y de la usabilidad del sistema completo. Los resultados parciales obtenidos durante la evaluación del sistema se publicaron en dos artículos: [63, 62].

En la sección 4.1 se describe la evaluación del sistema de recuperación de información y del componente de minería de texto. En la sección 4.2 se describe la evaluación de usabilidad de Lid-UN.

4.1 EVALUACIÓN DE COMPONENTES DEL MODELO

4.1.1 Evaluación del Sistema de Recuperación de Información

La evaluación del sistema de recuperación de información se publicó en [63], en el cual se plantea un método de selección de un buscador web para un portal web y también se describen las pruebas de rendimiento que se hicieron con Nutch. Este buscador web de código abierto fue seleccionado a través de la evaluación de los requisitos no funcionales y el rendimiento. A continuación se describe el método de selección y evaluación de Nutch y los resultados obtenidos al implementar este buscador web de código abierto dentro del portal del Departamento de Ingeniería de Sistemas e Industrial de la Universidad Nacional de Colombia.

En el anexo C, se encuentra el artículo publicado en [63], que contiene toda descripción del estudio exploratorio para la selección del buscador web de código abierto, donde están los detalles y la totalidad de buscadores que se incluyeron en el estudio.

4.1.1.1 Método de evaluación

El sistema de recuperación de información de Lid-UN descrito en la subsección 3.2.1.1, fue escogido a través de un proceso de selección que consistió en la evaluación de los requisitos no funcionales y el rendimiento de los motores de búsqueda de código abierto que se encontraron disponibles en la Web; los resultados se publicaron en [63]. Los criterios de selección fueron tomados de FURPS [35], un modelo desarrollado en Hewlett-Packard por Grady y Caswell para la clasificación de los atributos de calidad del software. FURPS es un acrónimo de:

- **Funcionalidad:** integra todo el conjunto de características, capacidades y seguridad.
- **Usabilidad:** incluye los factores humanos, la estética, la consistencia, la documentación.

- **Fiabilidad:** agrupa factores como la frecuencia y severidad de los fallos, la recuperación, la previsibilidad, la exactitud, tiempo medio entre fallos.
- **Rendimiento:** puede ser medido como la velocidad, la eficiencia, el consumo de recursos y el tiempo de respuesta.
- **Compatibilidad:** incluye propiedades como la extensibilidad, adaptabilidad, capacidad de mantenimiento, compatibilidad, capacidad de configuración, la capacidad de servicio, capacidad de instalación y la portabilidad.

Se encontró un total de 48 motores de búsqueda listados en portales web enfocados en el campo de la recuperación de información, donde se publican noticias y artículos de análisis sobre nuevos desarrollos y herramientas, estos portales fueron: Search-Tools[1], Search Engine Watch[2] y java-source.net[3].

Los motores de código abierto se evaluaron en términos de las siguientes cuatro características de calidad: la compatibilidad, usabilidad, funcionalidad y rendimiento[63].

- **Compatibilidad:** En [35], la característica de compatibilidad combina tres factores: 1) extensibilidad, la capacidad de extender el sistema, 2) adaptabilidad: el grado en que un sistema de software se adapta a los cambios en su entorno y 3) servicio: la facilidad y la eficacia del mantenimiento del sistema. La evaluación de la compatibilidad de los motores de búsqueda de código abierto incluyó características de disponibilidad y de mantenimiento. Estos aspectos garantizan que el software y sus modificaciones futuras se publicarán en los sitios web de los proyectos que se desarrollaron los motores de búsqueda de código abierto. Para la evaluación de estos criterios, se incluyeron los siguientes datos: estado del proyecto, fecha de la última actualización, la versión, el tipo de licencia, y el número de sitios web donde se utiliza. Los motores de búsqueda se filtraron según los siguientes criterios: el estado del proyecto debe estar activo, la fecha de la última actualización no debería haber ocurrido hace más de 3 años y la licencia debe permitir modificaciones y añadir nuevas funcionalidades [63].
- **Usabilidad**: El modelo FURPS propone la evaluación de la usabilidad a través de los factores humanos, la estética general, la coherencia y la documentación [35]. En este caso, el motor de búsqueda debe cumplir con una documentación adecuada para su instalación, configuración y administración. Esta condición garantiza que hay suficiente información disponible para los usuarios para llevar a cabo sus actividades de consulta, y para los administradores de sistemas que son responsables de la instalación y la gestión de posibles errores. Por otra parte, la documentación relativa al código fuente es necesario para el desarrollo de nuevas funciones para el software de código abierto. Para evaluar este aspecto, se incluyeron los siguientes datos: autor, el lenguaje del desarrollo, manual del usuario, documentación de desarrollo y lenguajes para la creación del índice de documentos. Los criterios para el filtrado de los motores de búsqueda, de acuerdo a la usabilidad fueron: tiene publicado un manual del usuario y documentación técnica, y debido a la lengua de la colección de documentos de prueba, en nuestro caso el motor de búsqueda debe permitir palabras de indexación en español.[63].
- **Funcionalidad:** La funcionalidad se evalúa el conjunto de características y capacidades del sistema [35]. El motor de búsqueda debe adaptarse a las condiciones específicas del sitio, por lo que tiene que ser capaz de crear un índice a partir de diferentes tipos de archivo en los que los documentos están publicados en el portal web. La evaluación de la funcionalidad se dividió en dos etapas [63]:

[1] Search-Tools`http://www.searchtools.com/`
[2] Search Engine Watch`http://searchenginewatch.com/`
[3] java-source.net`http://java-source.net/open-source/`

1. **Evaluación de las características de indexación**: Estas características permiten la evaluación de las opciones que cada motor de búsqueda tiene que almacenar los documentos de acuerdo a los tipos de archivos, lenguajes y estrategias para crear y mantener un índice que se utiliza para procesar las consultas del usuario. Para evaluar las características de indexación se incluyeron los siguientes datos: estrategia de almacenamiento, los tipos de archivos que puede indexar, y manejo de: *stemming*, *stop word*, sinónimos y acrónimos. Los criterios para el filtro en esta parte de la evaluación fueron los siguientes: el motor de búsqueda debe ser capaz de indexar documentos de texto que están almacenados en diferentes formatos como texto plano, Microsoft Word, RTF y PDF [63].
2. **Evaluación de las características de la búsqueda**: Estas características permiten la verificación de los métodos aplicados para procesar las consultas de los usuarios. Estos métodos dependen del tipo de búsqueda que se ejecute y el ranking de resultados utilizado. Además, esta evaluación incluye el uso de plantillas que pueden ser usadas para mostrar los resultados de las consultas. Para la evaluación de las características de búsqueda, se incluyeron los siguientes datos: tipos de búsqueda permitidos (booleano, frases, comodines, difusa), función de ranking de resultados, resumen del contenido de los documentos resultantes de una consulta y la plantilla de presentación de resultados. Los criterios para el filtro en esta evaluación fueron: el motor de búsqueda tiene que tener por lo menos de búsqueda booleana y búsqueda de frases, debe tener una función de ranking para manejar una lista de resultados relevantes de una consulta [63].

- **Rendimiento:** Según la evaluación preliminar de la compatibilidad, usabilidad y funcionalidad, se eligieron los tres mejores motores de búsqueda de código abierto, los cuales podrían ser adaptados a las condiciones específicas de un portal web de tamaño pequeño o mediano. Las pruebas de rendimiento incluyen medidas de eficacia y eficiencia:

  1. **Métrica de Eficacia**: Para medir la eficacia se escogió la precisión, esta métrica muestra en que porcentaje el motor de búsqueda puede rechazar de los documentos no relevantes. Para medir la precisión, lo primero que considera un punto de corte del listado de los resultados de la consulta, en este caso se tomaron los primeros 20 resultados que los que normalmente aparecen en la primera página de los resultados de búsqueda [116]. En el listado de los resultados relevantes, cada página o documento encontrado tiene el título, la URL y un pequeño resumen donde aparecen los términos de búsqueda. Este estudio sólo se considera la precisión debido a otras medidas como el *recall* requieren de una exploración completa de la colección de documentos y luego de un especialista para decidir cuáles son relevantes en diferentes consultas[116], y en este caso este factor era una restricción. La precisión se calculó con la ecuación 4.1 [116], donde el número de documentos relevantes recuperados corresponde al número de documentos de la lista de resultados que contienen los términos de búsqueda, y el número total de documentos recuperados en este caso como se explicó anteriormente es igual a 20.

$$Precisión \quad = \quad \frac{Número\ de\ documentos\ relevantes\ recuperados}{Número\ total\ de\ documentos\ recuperados} \tag{4.1}$$

  2. **Métricas de Eficiencia**: para medir la eficiencia del motor de búsqueda se utilizaron dos métricas: 1) Tiempo transcurrido de indexación, que mide la cantidad de tiempo necesario para construir un índice de documentos en un sistema en particular [116]; y 2) El tamaño del índice, que mide la cantidad de almacenamiento necesario para almacenar los archivos de índice [116].

4.1.1.2 Parámetros de evaluación

1. **Sitio web de prueba:** El sitio de prueba fue el portal web de la Universidad Nacional de Colombia[4]. Este sitio web fue seleccionado porque es un ejemplo de un sitio web de una institución, con una organización típica de documentos y una distribución de temas específicos de la institución. Los usuarios de este sitio web son: estudiantes, profesores, personal administrativo y el personas en general. El sitio web contiene documentos relativos a las normas de la institución y los programas académicos, la información y los documentos que los cursos de los profesores, información de grupos de investigación, servicios de información y datos de contacto de las personas que componen el departamento.

2. **Profundidad:** Éste parámetro se refiere al límite en profundidad (cantidad de niveles dentro de la *web graph*) para efectuar la recuperación de las páginas. En este caso se le dio el valor de 10 para efectuar una recuperación total del sitio.

3. **Número de páginas por nivel:** Ese es el límite de número de páginas que se recuperan en cada nivel de la *web graph*. En este estudio, hemos utilizado tres cantidades de documentos: 50, 100 y 1.000.

4. **Tamaño de los conjuntos de documentos de prueba:** de acuerdo a la profundidad y al número de páginas recuperadas por nivel, se obtuvieron tres conjuntos: 500 documentos, 1000 documentos y 10.000 documentos.

5. **Consultas de prueba:** La muestra de consultas se seleccionó de acuerdo con el tipo y el formato de los documentos que se publican en el portal web de prueba. Estas consultas fueron utilizadas para probar características del motor de búsqueda Nutch, incluyendo los diferentes tipos de búsqueda que tiene. Consultas de búsqueda:

   - Búsqueda con las palabras: actas, resolución, pregrado, maestría.
   - Búsqueda con comodines: *. pdf, *. doc
   - Búsqueda de frases: Grupos de investigación, Maestría en Ingeniería, "Maestría en Ingeniería", "Departamento de Ingeniería de Sistemas e Industrial", "Departamento de Ingeniería de Sistemas e Industrial".

4.1.1.3 Resultados

Luego de aplicar el método de evaluación descrito en la subsección 4.1.1.1, a partir del conjunto de 48 buscadores web de código abierto, se seleccionó Nutch porque es un sistema robusto que cumplió con los criterios tanto funcionales como no funcionales y de rendimiento como se describe a continuación[63]:

Mantenimiento

Nutch está publicado en la web, como un proyecto cubierto por Apache Software Foundation, el cual es mantenido continuamente.

Usabilidad

Nutch tiene información suficiente para la instalación, uso y desarrollo de nuevas características. Cuenta con un foro que facilita la solución de problemas y la corrección de errores de uso y ejecución en diferentes ambientes. Se adapta fácilmente a cualquier sitio web y es utilizado por los usuarios de

[4] Universidad Nacional de Colombia `http://http://www.unal.edu.co/`

manera intuitiva, porque presenta características visuales similares a las que tienen los motores de búsqueda disponibles en la Web (e.g. Google, Yahoo. Bing).

Funcionalidad

Nutch tiene implementado un proceso para indexación a través de la creación de índices en formato Lucene y puede ser utilizado para la mayoría de idiomas occidentales. Permite efectuar búsqueda por frases, *wilcards* y algunas de tipo booleano. Maneja una función robusta de asignación de relevancia utilizando el modelo de espacio vectorial. Tiene opciones para mostrar el resumen del contenido del documento y tiene una plantilla de prueba para la ejecución de consultas. Nutch es un sistema robusto debido a que integra todos los componentes necesarios para buscar en la Web: un rastreador, un índice, una base de datos de enlaces e hipervínculos, analizadores de formato texto plano, HTML, JavaScript, Microsoft Excel, Microsoft Word, Open Office, PDF, RSS, SWF y ZIP [101].

Rendimiento

La evaluación de rendimiento se hizo para dos buscadores web de código abierto que presentaron las mejores características funcionales y no funcionales, que fueron Nutch y Terrier[5]. Terrier es un proyecto del Departamento de Informática de la ciencia de la Universidad de Glasgow. El objetivo de este motor de búsqueda es proveer una plataforma comprensible, adaptable y robusta. Este buscador web está escrito en lenguaje Java y se ha utilizado para sistemas de búsqueda en la Web, sistemas empresariales y aplicaciones de escritorio. Además, esta herramienta se utiliza para desarrollar y evaluar nuevas técnicas de texto a gran escala de recuperación de información.

1. **Evaluación de efectividad**

   La medida de la precisión fue la métrica escogida para evaluar la efectividad, en la figura 4.1 se muestran los resultados obtenidos, donde se compara el nivel de precisión de las consultas de ejemplo en tres índices diferentes. Se observó que para el primer caso del índice de 500 documentos, la precisión para algunas consultas no es muy buena, pero para el caso de 1000 y 10000 documentos la precisión es igual a 100 % para todas las consultas de prueba, para ambos buscadores. Se concluyó que para índices pequeños o menores a 500 documento, la precisión en la recuperación de los documentos disminuye significativamente.

   En la figura 4.2, se muestra la precisión de Nutch de acuerdo al tipo de consulta, donde se puede observar que la precisión es bastante buena en diferentes tipos de consulta como frases completas, palabras y comodines, pero en el conjunto de 500 documentos la precisión disminuye cuando se trata de frases, varios términos y comodines.

2. **Evaluación de eficiencia**

   El tiempo estimado en indexación y el tamaño del índice fueron las dos métricas utilizadas para medir la eficiencia del sistema. La figura 4.3 muestra la relación entre el número de documentos recuperados y la cantidad de tiempo usado para el indexación. Se observó que el proceso de indexación de Terrier es más corto para los tres conjuntos, pero el tiempo utilizado por Nutch no es significativamente más amplio.

   La figura 4.4 muestra la relación entre el número de documentos recuperados y el tamaño del índice. El tamaño del índice de Terrier es aproximadamente el 50 % del volumen de los documentos originales. Nutch por su parte, crea índices con tamaños más reducidos que se aproximan al 30 % del volumen de los documentos originales.

[5] Terrier `http://ir.dcs.gla.ac.uk/terrier/`

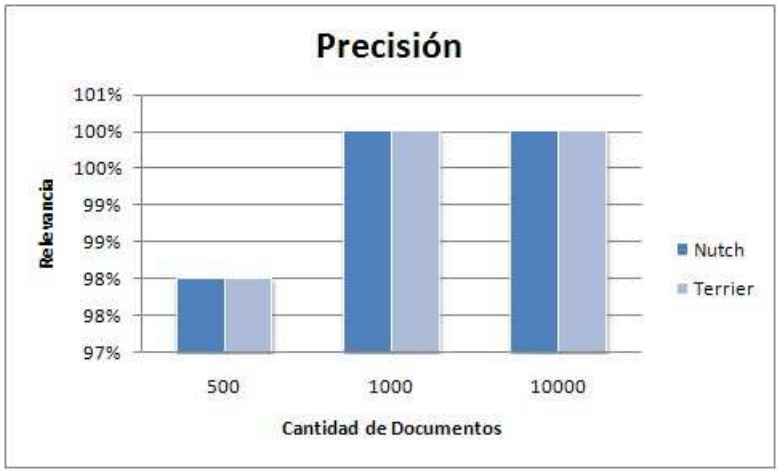


Figura 4.1: Precisión para diferentes consultas en dos niveles

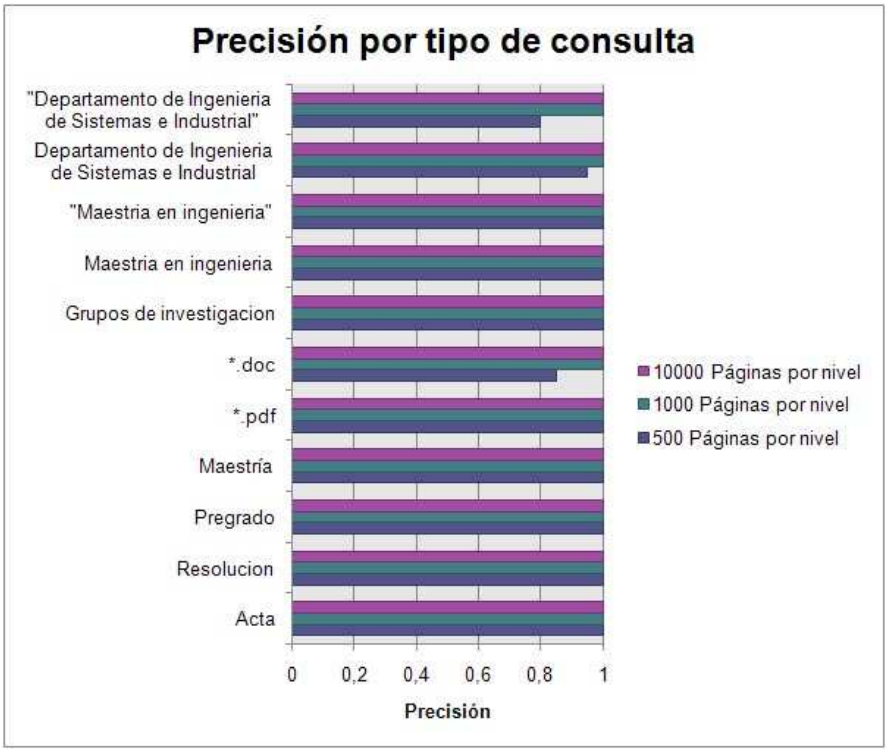


Figura 4.2: Precisión de Nutch por tipo de consulta

4.1.2 Evaluación del Componente de Minería de Texto

La evaluación del componente de minería de texto se baso en la implementación de un prototipo con los componentes del sistema de recuperación de información y el componente de minería de texto descritos en el capítulo 3. Los resultados de las pruebas del prototipo se publicaron en [62].

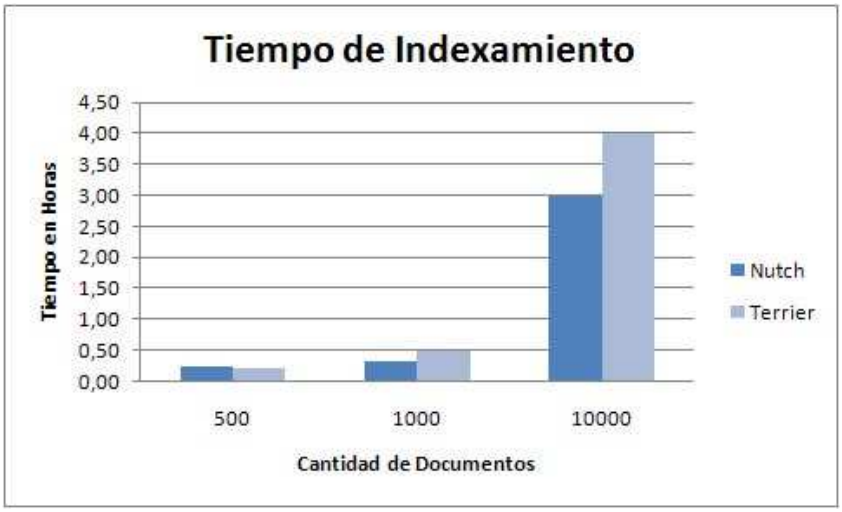


Figura 4.3: Tiempo de indexación

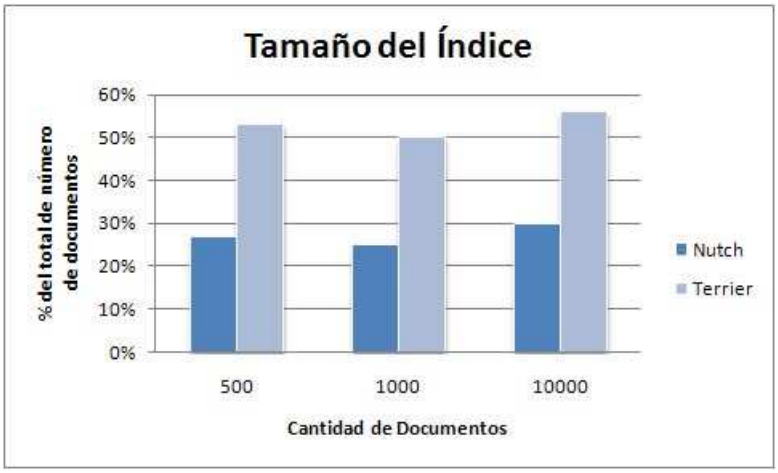


Figura 4.4: Tamaño del índice

4.1.2.1 Método de evaluación

El componente de minería de texto fue probado en el sitio web del Departamento de Ingeniería de Sistemas e Industrial de la Universidad Nacional de Colombia[6]. La mayoría de los documentos en este sitio web están escritos en idioma español, solo unos pocos están escritos en idioma inglés. Los documentos publicados en el sitio web fueron indexados en diferentes conjuntos de prueba con el fin de examinar el comportamiento del sistema con los conjuntos de tamaños incrementales. Se obtuvieron diecisiete índices en total.

[6] Departamento de Ingeniería de Sistemas e Industrial `http://www.disi.unal.edu.co/`

4.1.2.2 Parámetros de evaluación

Parámetros de Rastreo e indexación

- **Profundidad:** Este parámetro se refiere a la profundidad de recorrido del rastreador. En este caso, las diferentes medidas de este parámetro se utilizaron: 2, 3 y 10 para un rastreo completo del portal web.
- **Número de páginas por cada nivel:** Este es el límite para el número de páginas de cada nivel de la *web-graph.* Las pruebas se realizaron con seis niveles: 10, 50, 100, 200, 500 y 1.000 páginas.

Parámetros del algoritmo de agrupamiento de documentos

La descripción completa de cada parámetro del algoritmo de LINGO se puede encontrar en [75]. El algoritmo de agrupamiento fue probado con diferentes valores para los parámetros siguientes:

1. **Umbral de etiquetas candidatas:** Permite identifica el número máximo de etiquetas de conjuntos candidatas. Corresponde a la aproximación k-rango usada para la simplificación de la matriz de términos y frecuencias con el método de descomposición en valores singulares (Ver subsección 2.3.3.8). Los valores oscilan entre 0,7 y 0,9. Los mejores resultados se obtienen con 0.75.
2. **Umbral de asignación de documento**: Es el umbral de similitud que el peso calculado de un documento debe superar para ser agregado a un conjunto. Los valores oscilan entre 0,1 y 0,3. Los mejores resultados se obtienen con 0,2.
3. **Umbral de frecuencia de término**: Es el número de veces que un término debe aparecer en los documentos de entrada con el fin de ser incluidos en la matriz de términos y frecuencias. Los valores van 2-5. Los mejores resultados se obtienen con 3.
4. **Umbral de similitud de etiqueta:** Es el nivel aceptable de similitud con la distancia coseno entre las etiquetas de los conjuntos. Si la similitud excede éste umbral, la etiqueta con el puntaje más bajo es descartada. Los valores oscilaron entre 0,2 y 0,5. Los mejores resultados se obtienen con 0,3.

4.1.2.3 Resultados

En la figura 4.5, se muestra la relación entre los conjuntos de documentos y el número de conjuntos. Los mejores juegos de grupos se encontraron en grupos entre 733 y 2.401 documentos.

Composición de conjuntos

La composición de los conjuntos obtenidos se analizó en función del número medio de documentos por conjunto. En la figura 4.7, se muestra una comparación del promedio de documentos por conjunto incluyendo todos los grupos que se obtuvieron. En la figura 4.7 se muestra el promedio de documentos por conjunto sin incluir el grupo "otros temas". Esta comparación se hizo porque el algoritmo de agrupamiento LINGO tiene una tendencia a incluir muchos documentos dentro del grupo "otros temas".

Los mejores composiciones de conjunto se encontraron en los conjuntos con tamaños que oscilan entre 414 y 1.930 documentos, porque el número promedio de documentos por conjunto están entre rangos de 15 y 20. En conjuntos con más de 1.930 documentos, el conjunto "otros temas" es significativamente

grande, lo que significa que el proceso de agrupación tiene deficiencias para la creación de conjuntos cuando se disponen conjuntos de tamaños grandes.

Etiquetas de Conjuntos

Los resultados obtenidos en este trabajo muestran que las etiquetas adecuadas se identifican en alrededor del 80% de los conjuntos de tamaños que oscilan entre 4 y 2401 documentos. En general se observó que las etiquetas de los conjuntos tienen relación con el título y con el contenido de los documentos que se asocian a cada conjunto.

Figure 4.5: Número de conjuntos por tamaño de conjuntos

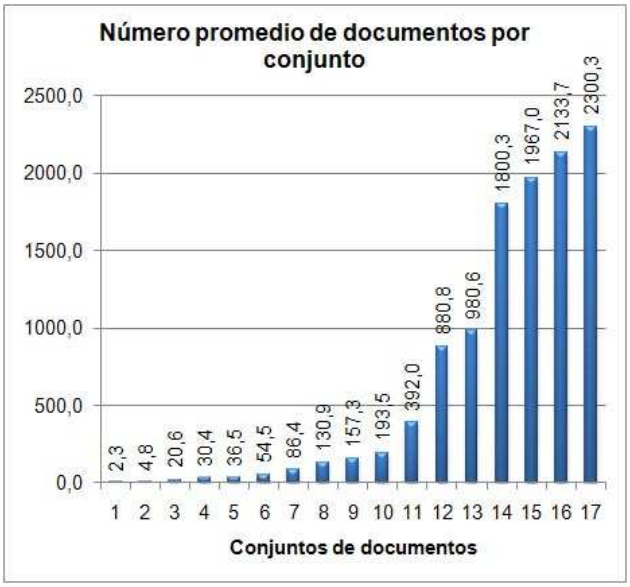


Figure 4.6: Promedio de documentos por conjunto

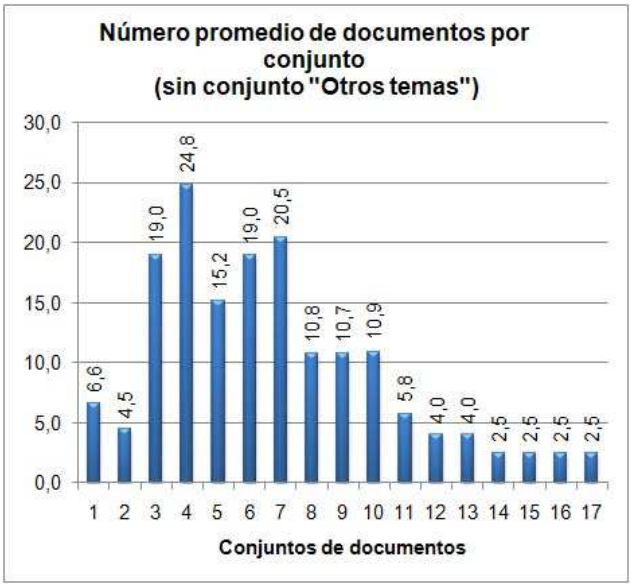


Figure 4.7: Promedio de documentos por conjunto sin el conjunto "otros temas"

4.2 EVALUACIÓN DE USABILIDAD DE LA LIBRERÍA DIGITAL WEB LID-UN

La evaluación de usabilidad de la librería digital web LID-UN se hizo a través de un primer estudio exploratorio estadístico utilizando el método de ***Human Computer Interaction*** (HCI) para la recolección de datos a través de una encuesta. A continuación se describen los aspectos evaluados y los resultados obtenidos.

4.2.1 Aspectos de usabilidad evaluados

Para la evaluación de usabilidad de Lid_UN se utilizó un instrumento propuesto en [52] el cual fue validado en un caso real y tiene un alto índice de confiabilidad para medir la usabilidad en librerías digitales. Este instrumento se compone por doce aspectos, tomados del estándar ISO 9241-11 referente a la usabilidad de los sistemas de software. A continuación se describen los aspectos de usabilidad evaluados:

1. **Simplicidad**: Una librería digital debe ser fácil de usar. La simplicidad se asocia a características de usabilidad como: facilidad de aprendizaje, eficiencia y satisfacción.

2. **Comfort**: Los usuarios deben sentirse cómodos usando una librería digital. Las características de usabilidad asociados al confort pueden ser: facilidad de aprendizaje, eficiencia y satisfacción.

3. **Facilidad de uso**: Una librería digital debe ser fácil de usar. Los atributos relacionados con esta propiedad puede incluir: atractivo, facilidad de aprendizaje, eficiencia y satisfacción.

4. **Control**: Los usuarios deben tener el control de la utilización de una librería digital. La usabilidad atributos que afectan a esta propiedad puede incluir operatividad, facilidad de aprendizaje, la eficacia, la eficiencia y la satisfacción.

5. **Legibilidad**: La información debe estar accesible a los usuarios de una librería digital y esta debe ser ordenada y legible. La atributos de usabilidad vinculados a la legibilidad incluyen la comprensión, atracción, facilidad de aprendizaje, la eficacia, eficiencia, y la satisfacción

6. **Información adecuada**: La información a la que se accede desde una librería digital debe ser adecuada. La usabilidad atributos relacionada con esta propiedad puede incluir la eficacia, eficiencia y la satisfacción.

7. **Navegabilidad**: Los usuarios deben ser capaces de navegar con facilidad, libremente, y sin problema a través de una librería digital. La usabilidad atributos vinculados a esta propiedad puede incluir operatividad, facilidad de aprendizaje, la eficacia, eficiencia y la satisfacción.

8. **Reconocimiento**: Los usuarios deben ser capaces de entender rápidamente las características y funciones disponibles en una librería digital. Los atributos vinculados a esta propiedad pueden ser recordación, facilidad de aprendizaje, la eficiencia y la satisfacción.

9. **Tiempo de acceso**: La información consultada en una librería digital debe ser cargado en un tiempo razonable. Los atributos vinculados a esta propiedad puede incluir la flexibilidad, eficacia, eficiencia y la satisfacción.

10. **Relevancia**: La información que se accede desde una librería digital debe ser relevante. Los atributos vinculados a esta propiedad puede incluir la eficacia, eficiencia, y la satisfacción.

11. **Consistencia**: A lo largo de una librería digital debe existir consistencia de los términos, palabras y acciones. Los atributos vinculados a esta propiedad puede incluir recordación, facilidad de aprendizaje, la eficiencia y la satisfacción.

12. **Presentación visual**: En una librería digital la presentación debe ser adecuada a través del uso de texto en negrilla, subrayado, cursiva. Los atributos vinculados a esta propiedad puede incluir la eficacia, eficiencia, atractivo y la satisfacción.

4.2.2 Instrumento de evaluación de usabilidad

El instrumento usado para la evaluación de usabilidad utiliza el método de ***Human Computer Interaction*** (HCI), el cual se enfoca en la evaluación participativa mediante un cuestionario. El objetivo de la evaluación con este instrumento fue la recolección de datos sobre la usabilidad de librerías digitales directamente desde el punto de vista subjetivo de los usuarios. Este método tiene la capacidad de adquirir todas las características relacionadas con las necesidades de los usuarios, los deseos, los procesos de pensamiento y experiencias, aspectos que son difíciles de evaluar de otra manera [43].

El instrumento usado para la evaluación de usabilidad utiliza la técnica de medición propuesta por Likert [59], en donde se tiene una escala de uno a cinco para la asignación de calificación en las respuestas. Las posibles respuestas son:

- Totalmente en desacuerdo = 5
- En desacuerdo = 4
- Ni de acuerdo ni en desacuerdo = 3
- De acuerdo = 2
- Totalmente de acuerdo = 1

Las preguntas relacionadas con cada uno de los aspectos de usabilidad evaluados son las siguientes:

1. **Simplicidad**: ¿La librería digital tiene opciones simples de usar?
2. **Confort**: ¿Me sentí a gusto dentro de la librería digital?
3. **Facilidad de uso**: ¿La librería digital es fácil de usar?
4. **Control**: ¿Percibí que yo controlaba la utilización de la librería digital?
5. **Legibilidad**: ¿La información que se accede desde la librería digital fue ordenada y legible?
6. **Información adecuada**: ¿La información que se accede desde la librería digital es adecuada?
7. **Navegabilidad**: ¿La navegación a través de la biblioteca digital se ejecuta con facilidad?
8. **Reconocimiento**: ¿Entendí rápidamente las características y funciones disponibles en el entorno de la librería digital?
9. **El tiempo de acceso**: ¿La información que solicitó dentro de la librería digital de carga en un tiempo razonable?
10. **Relevancia**: ¿La información que obtuve de la biblioteca digital era pertinente con mi necesidad de información?
11. **Consistencia**: ¿La consistencia de los términos, palabras y acciones utilizados en todo el librería digital era evidente?
12. **Presentación visual**: ¿El Texto en negrillas, cursivas, subrayado y estuvieron presentes en la librería digital para llamar mi atención?

4.2.3 Método de evaluación

4.2.3.1 Población

La población a la cual está dirigida la librería digital está compuesta por todas las personas que integran una comunidad de una organización, en este caso una universidad; por tanto se escogió una muestra de 40 usuarios que tienen los siguientes perfiles:

1. Estudiantes de posgrado de la Universidad Nacional de Colombia, quienes son usuarios expertos en el acceso a información publicada en la Web, que conocen y manejan de forma adecuada diferentes herramientas como buscadores web, librerías digitales de artículos científicos y acceso a portales web.
2. Profesionales que conocen herramientas de búsqueda en la Web.

4.2.3.2 Librería digital de evaluación

La librería digital Lid-UN se adaptó al portal del Departamento de Ingeniería de Sistemas e Industrial[7] que es uno de los portales dentro de la Universidad Nacional de Colombia[8].

[7] Departamento de Ingeniería de Sistemas e Industrial `http://dis.unal.edu.co/`
[8] Universidad Nacional de Colombia `http://www.unal.edu.co/index.html`

4.2.3.3 Infraestructura utilizada

El prototipo de Lid-UN fue instalado de forma local en los equipos ubicados dentro del campus de los usuarios y el acceso se hizo utilizando el navegador web Firefox versión 3.6[9].

4.2.3.4 Resultados de la evaluación de usabilidad

Simplicidad: ¿La librería digital tiene opciones simples de usar?

En la figura 4.8 se observa que el 62.5 % de las personas encuestadas están de acuerdo en que la librería digital tiene opciones sencillas de entender y de usar, lo cual muestra que los componentes de la interfaz gráfica invitan al usuario a usar la librería digital.

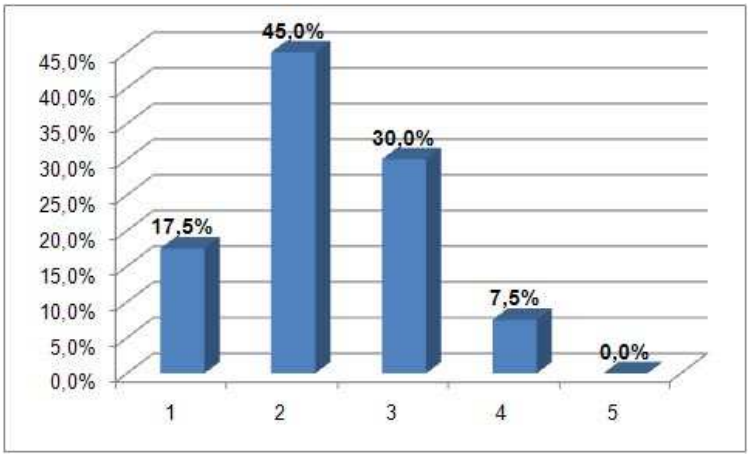


Figura 4.8: Distribución de opiniones para la característica de simplicidad: ¿La librería digital tiene opciones simples de usar?

Confort: ¿Me sentí a gusto dentro de la librería digital?

En la figura 4.9 se observa que el 67.5 % de las personas encuestadas están de acuerdo en que la librería digital permite que los usuarios se sientan a gusto dentro de la librería digital, lo cual indica que la interfaz gráfica es amigable, cómoda y llamativa.

Facilidad de uso: ¿La librería digital es fácil de usar?

En la figura 4.10 se muestra que el 57.5 % de las personas encuestadas están de acuerdo en que la librería digital es fácil de usar, lo cual se confirma con el hecho que a los usuarios les tomó muy poco tiempo para comprender como usar el sistema y efectuar consultas exitosas.

Control: ¿Percibí que yo controlaba la utilización de la librería digital?

En la figura 4.11 se muestra que el 65 % de las personas encuestadas están de acuerdo en que percibieron que tenían el control mientras utilizaban la librería digital, lo cual indica que los usuarios se

[9] Firefox `http://www.mozilla.com/`

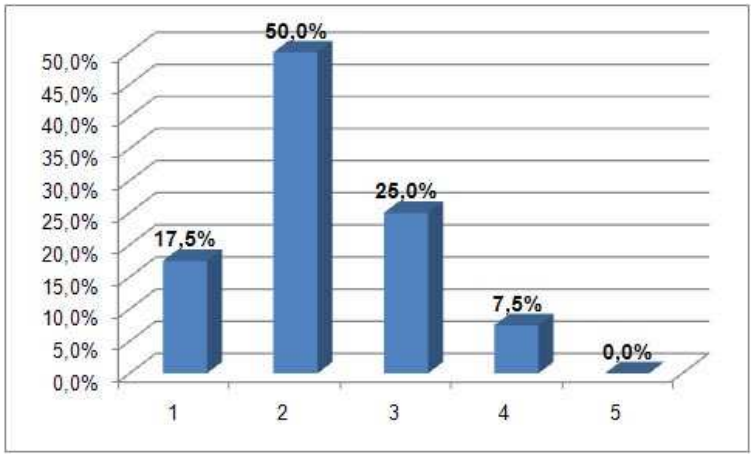


Figura 4.9: Distribución de opiniones para la característica de confort: ¿Me sentí a gusto dentro de la librería digital?

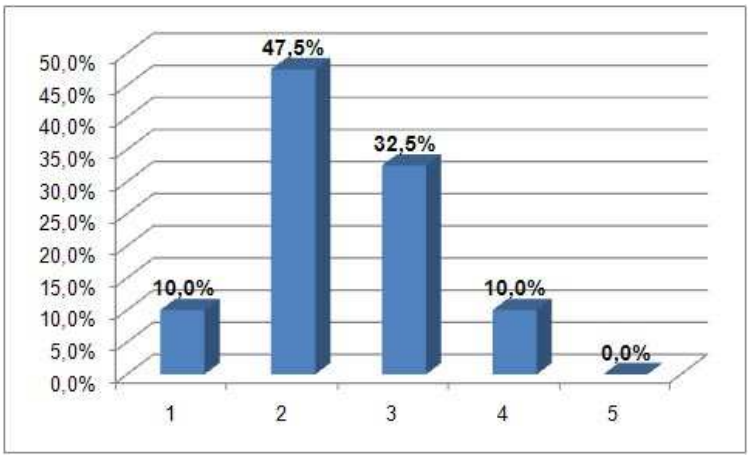


Figura 4.10: Distribución de opiniones para la característica de facilidad de uso: ¿La librería digital es fácil de usar?

sienten libres dentro del sistema para manejar las diferentes funciones que este provee sin que perciban limitaciones o restricciones para su uso.

Legibilidad: ¿La información que se accede desde la librería digital fue ordenada y legible?

En la figura 4.12 se observa que el el 67.5 % de las personas encuestadas están de acuerdo en que la librería digital ofrece legibilidad en la información que se accede en el sistema, lo cual indica que tanto la interfaz gráfica del sistema como la estructura de las etiquetas de las categorías de documentos y la distribución de los documentos dentro de las categorías es fácil de leer y tiene un orden que es posible percibir.

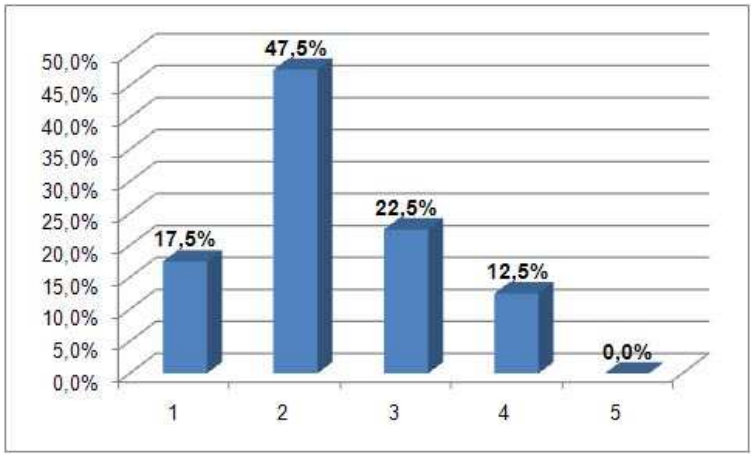


Figura 4.11: Distribución de opiniones para la característica de **Control**: ¿Percibí que yo controlaba la utilización de la librería digital?

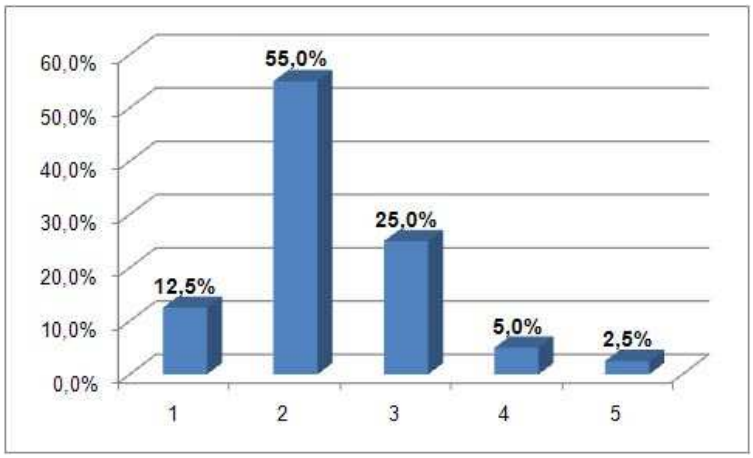


Figura 4.12: Distribución de opiniones para la característica de **Legibilidad**: ¿La información que se accede desde la librería digital fue ordenada y legible?

Información adecuada: ¿La información que se accede desde la librería digital es adecuada?

En la figura 4.13 se observa que el 70 % de las personas encuestadas están de acuerdo en que la información que se accede desde la librería digital es adecuada dentro del contexto en el cual se probó, lo cual indica que el sistema puede ser adaptado a un contexto específico para facilitar el acceso de la información.

Navegabilidad: ¿La navegación a través de la biblioteca digital se ejecuta con facilidad?

En la figura 4.14 se observa que el 75 % de las personas encuestadas están de acuerdo en que la navegación dentro de la librería digital se ejecuta de forma fácil, lo cual muestra que los usuarios pueden utilizar el sistema a través de la función de navegación

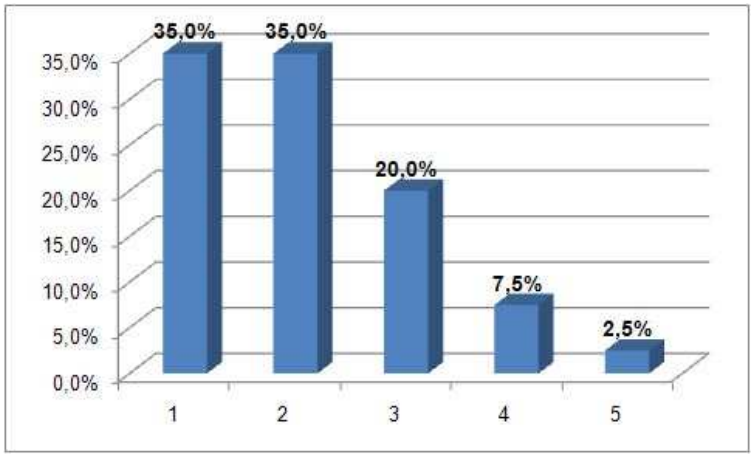


Figura 4.13: Distribución de opiniones para la característica de **Información adecuada**: ¿La información que se accede desde la librería digital es adecuada?

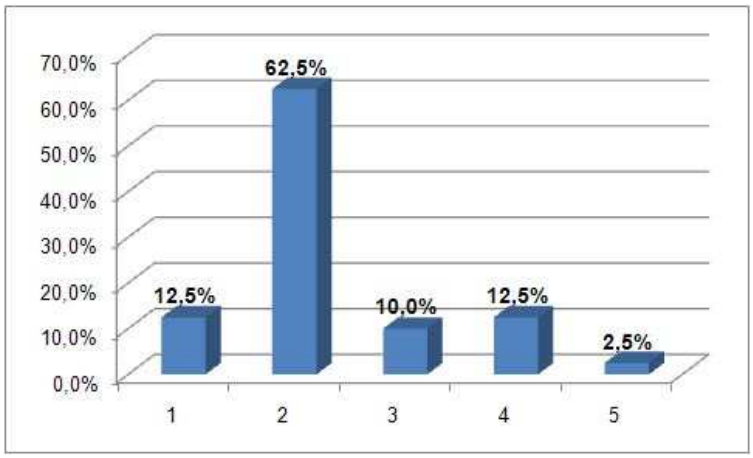


Figura 4.14: Distribución de opiniones para la característica de **Navegabilidad**: ¿La navegación a través de la biblioteca digital se ejecuta con facilidad?

Reconocimiento: ¿Entendí rápidamente las características y funciones disponibles en el entorno de la librería digital?

En la figura 4.15 se observa que el 65 % de las personas encuestadas están de acuerdo en que las características y las funciones disponibles de la librería digital se reconocen rápidamente. A cada usuario se le enseño el funcionamiento básico del sistema, pero no se les dieron muchos detalles, lo que se pudo observar fue que los usuarios además pudieron descubrir de forma amplia como funciona el sistema y las opciones que tienen para utilizar cada una con el objetivo de utilizarlas para efectuar consultas de información.

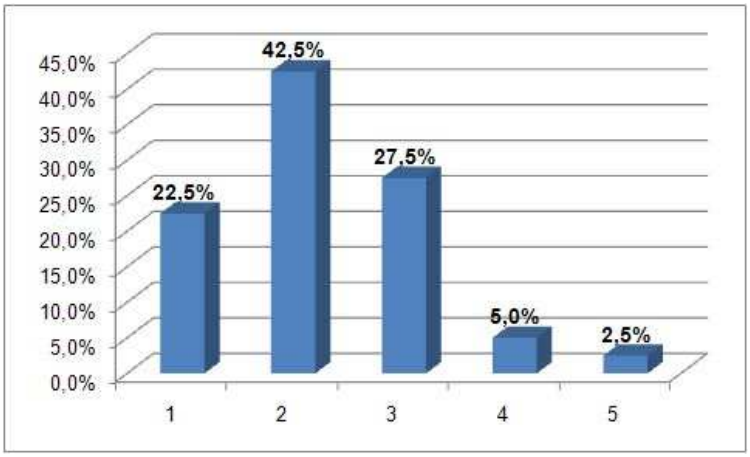


Figura 4.15: Distribución de opiniones para la característica de **Reconocimiento**: ¿Entendí rápidamente las características y funciones disponibles en el entorno de la librería digital?

El tiempo de acceso: ¿La información que solicitó dentro de la librería digital de carga en un tiempo razonable?

En la figura 4.16 se observa que el 90 % de las personas encuestadas están de acuerdo en que el tiempo de acceso es adecuado cuando efectúan consultas de información. Debido a que las consultas se hicieron a fuentes externas en la Web, el 10 % de la muestra tuvo problemas para acceder a los documentos buscados, lo cual se debió a varias razones inherentes a la navegación en la Web que son la dependencia de las condiciones de la conexión a Internet y también de la disponibilidad de las páginas en el momento en el que usuario las consulta.

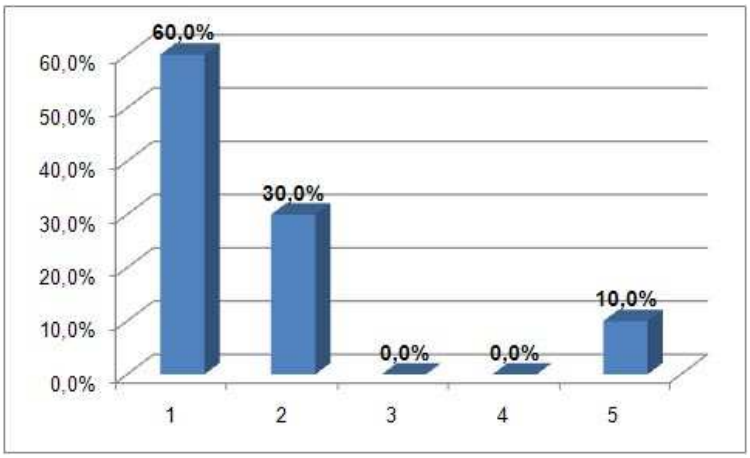


Figura 4.16: Distribución de opiniones para la característica del **tiempo de acceso**: ¿La información que solicitó dentro de la librería digital de carga en un tiempo razonable?

Consistencia: ¿La consistencia de los términos, palabras y acciones utilizados en todo el librería digital era evidente?

En la figura 4.17 se observa que el 62.5 % de las personas encuestadas están de acuerdo en que se maneja un conjunto de términos adecuado dentro de las etiquetas de categorías y nombres de documentos mostrados en la librería digital. El restante 37.5 % de las personas percibieron inconsistencia en los términos debido a que muchas páginas del portal de prueba, no tienen títulos adecuados y también tienen títulos repetidos lo cual no permite identificar las páginas de forma fácil desde el sistema.

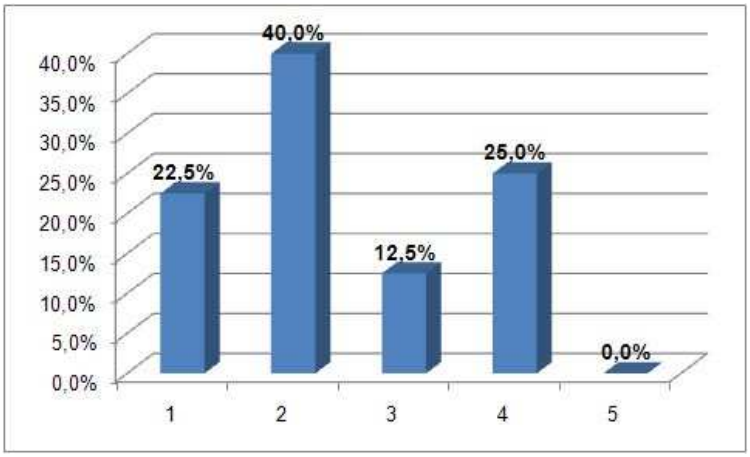


Figura 4.17: Distribución de opiniones para la característica de **Consistencia**: ¿La consistencia de los términos, palabras y acciones utilizados en todo el librería digital era evidente?

Presentación visual: ¿El Texto en negrillas, cursivas, subrayado y estuvieron presentes en la librería digital para llamar mi atención?

En la figura 4.18 se observa que el 52 % de las personas encuestadas están de acuerdo en que la presentación visual es llamativa y utiliza diferentes elementos del texto para fijar puntos de atención en la información. La presentación visual tiene componentes que pueden ser mejorados para llamar la atención en información más relevante, como en los títulos de las categorías para resaltarlos en la estructura de navegación. También los usuarios sugirieron que puede existir mayor integración de los componentes de visualización para ofrecer diferentes opciones de visualización a través de diferentes criterios.

Relevancia: ¿La información que obtuve de la biblioteca digital era pertinente con mi necesidad de información?

En la figura 4.19 se observa que el 52.5 % de las personas encuestadas están de acuerdo en que la información obtenida a través de la consulta de información específica es relevante. En este sentido, se observó que todos los usuarios encontraron el documento solicitado, y lo hicieron utilizando las opciones que provee el sistema.

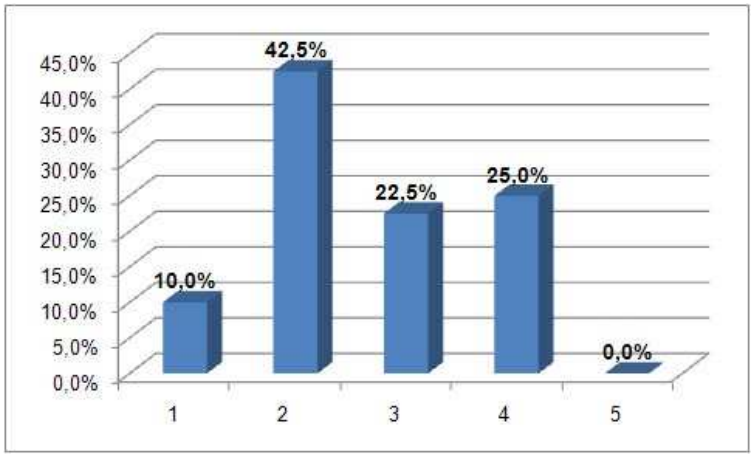


Figura 4.18: Distribución de opiniones para la característica de **Presentación visual**: ¿El Texto en negrillas, cursivas, subrayado y estuvieron presentes en la librería digital para llamar mi atención?

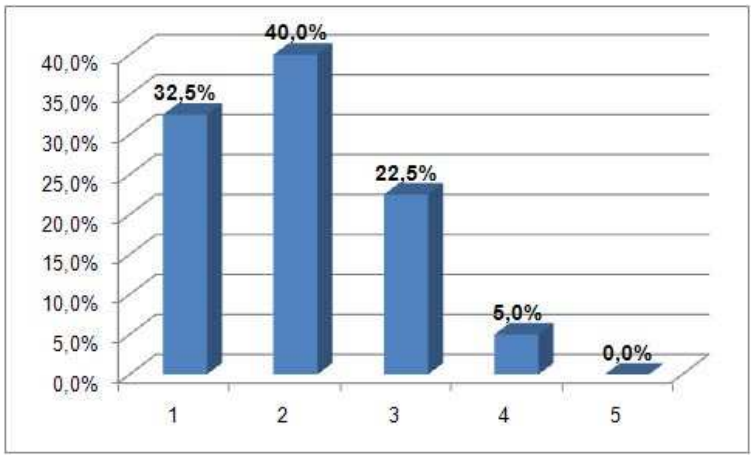


Figura 4.19: Distribución de opiniones para la característica de **Relevancia**: ¿La información que obtuve de la biblioteca digital era pertinente con mi necesidad de información?

Resumen del Capítulo

En este capítulo se describieron las pruebas desarrolladas para los componentes del sistema de librería digital web propuesto, en tres etapas principales, la primera consistió en la evaluación funcional y de rendimiento del sistema de recuperación de información. La segunda fue la evaluación del componente de minería de texto. Finalmente, la última etapa fue la prueba de usabilidad del sistema a través de interfaz de navegación que se propuso para la búsqueda de documentos a través de una jerarquía de categorías de documentos.

5 CONCLUSIONES Y TRABAJO FUTURO

En esta tesis se planteó el diseño de un sistema para generar, administrar y consultar librerías digitales de documentos publicados en un portal web. En este capítulo se desarrolla la discusión de los resultados obtenidos del modelo propuesto, y se proponen algunos trabajos futuros a partir de los hallazgos encontrados en el desarrollo del sistema.

5.1 MODELO PROPUESTO

El sistema de recuperación de información implementado a partir del buscador de código abierto llamado Nutch, permitió el aprovechamiento de características robustas para el rastreo, análisis de formato e indexación de documentos publicados en sitios web públicos. La librería utilizada para el indexación de documentos llamada Lucene, provee funciones para la creación de índices invertidos que almacenan tanto el contenido textual como los metadatos de los documentos. Estos índices creados con el formato Lucene permiten la ejecución de consultas de forma rápida y respondiendo a los términos buscados de forma eficaz.

El componente de minería de texto basado en el algoritmo LINGO, se probó con diferentes tamaños de conjuntos de documentos web, con lo cual se evidenciaron sus limitaciones debido al enfoque con el cual fue creado para el agrupamiento de resultados de búsqueda. El algoritmo funciona bien para conjuntos de pequeño y mediano alcance, encuentra etiquetas que describen conceptos lógicos y entendibles por los seres humanos. El agrupamiento de los documentos alrededor de las etiquetas o los conceptos, propuso una nueva visión en el campo de la minería de texto buscando que los procesos de etiquetamiento de los conjuntos obtenidos sean de forma automática. Sin embargo, se observó el conjunto llamado "Otros temas", en el cual se asignan los documentos que no se pudieron asignar a otro conjunto, es de un tamaño considerable, lo cual puede mejorarse con documentos mejor estructurados y también con medidas de asociación a los conjuntos que tengan parámetros más robustos.

Los recursos de tiempo y de memoria RAM requeridos para el desarrollo del proceso de agrupamiento con LINGO, son considerables debido al enfoque en *Latent Semantic Indexing* y la descomposición por valores singulares de la matriz de términos y documentos. Sin embargo, para procesos de agrupamiento como el propuesto en el modelo de la librería digital web donde la ejecución del proceso de agrupamiento de documentos se hace fuera de línea, la necesidad de tiempo de ejecución no es crítica y la cantidad de memoria RAM requerida puede ser soportada por equipos de cómputo que puedan disponer de al menos dos Gigabytes para el uso de la máquina virtual de Java.

El modelo de librería digital web de documentos propuesto se probó con el caso real de un portal web de una institución educativa, lo cual planteó el reto del procesamiento de documentos de texto propios de la vida diaria de una comunidad y que se encuentran en su mayoría escritos en idioma español. Las pruebas de los diferentes componentes con dichos documentos mostraron que se logró un adecuado rastreo web, indexación y posterior agrupamiento en conjuntos etiquetados automáticamente.

La interfaz de navegación que se integró en el modelo de librería digital, incluye dos vistas diferentes una que tiene la jerarquía de categorías de documentos y el listado de los documentos que pertenecen a

cada categoría con su respectivo título y enlace al documento original. En la segunda vista se despliega una red donde los nodos principales son las categorías de documentos y se muestra la navegación dentro de los conjuntos de categorías y sus relaciones entre categorías para guiar la navegación. La propuesta de esta interfaz integrada permitió probar una idea novedosa para el acceso a la información de forma rápida en documentos distribuidos en forma a veces desconocida para los usuarios.

5.2 HALLAZGOS

Durante las fases de rastreo, indexación, procesamiento de texto y agrupamiento de documentos se identificaron gran cantidad de páginas web y documentos del portal web de prueba, que carecen del título y que al final el algoritmo de agrupamiento los ubicaba en un conjunto adecuado. Para mejorar los procesos ejecutados al interior de la librería digital web, los documentos de entrada deberían estar escritos con un estándar de campos obligatorios como el título y el autor.

Durante las pruebas del sistema de recuperación de información se observó que el rastreador web utilizado, con frecuencia no podía recorrer el contenido público dentro de los portales de la Universidad. Para mejorar los procesos de recuperación de información de páginas web y documentos dentro de los portales con acceso público es necesaria la implementación de una política organizacional que integre la aplicación de estándares como el uso del archivo "robots.txt", el cual le permite a los rastreadores web tener acceso a partes del portal web que son públicas y respetar las que están restringidas. En el anexo B se muestra un primer acercamiento para plantear dicha política de seguridad.

Durante la prueba de usabilidad de la librería digital se observó que la propuesta de interfaz gráfica llama la atención de los usuarios y además invita a descubrir su funcionamiento. El tiempo de aprendizaje para entender el funcionamiento del sistema es muy corto y los usuarios solo requieren de una descripción básica del funcionamiento para poder usarlo y efectuar búsquedas de información exitosas.

5.3 TRABAJOS FUTUROS

En esta tesis se propuso un modelo para el desarrollo de una librería digital de documentos, el cual propone una estrategia novedosa para la generación de una jerarquía de categorías de documentos que luego puede ser consultada por usuarios para facilitar el acceso a la información. Sin embargo, se tienen algunos problemas que pueden ser mejorados para obtener mejores resultados a lo largo de la ejecución de todo el proceso propuesto en el modelo. A continuación se presentan algunas sugerencias para el desarrollo de trabajos futuros relacionados con el tipo de sistemas propuestos en este trabajo:

1. **Algoritmo de agrupamiento jerárquico**

   La utilización de un algoritmo de agrupamiento de documentos que pueda generar una estructura de sub categorías, con lo cual se puede facilitar la navegación dentro de categorías muy grandes. Pueden plantearse mejoras al algoritmo LINGO, para la iteración del agrupamiento de documentos dentro de unas categorías base que en principio identifican conceptos muy generales. También se puede proponer el uso de algoritmos jerárquicos e incrementales que permitan encontrar tanto categorías de documentos como sub categorías asociadas a los mismos.

2. **Algoritmo para etiquetamiento de categorías de documentos**

   Para mejorar las etiquetas que se asignan automáticamente para identificar las categorías de documentos, pueden ser mejoradas a través de algoritmos que permitan un mejor aprovechamiento de las *stop words*, que son palabras que permiten identificar conceptos dentro de documentos de texto. Además este tipo de palabras son necesarias para la formación de etiquetas de categorías con términos compuestos lo cual permite que sean mejor entendidas por los seres humanos.

3. **Descripción del contenido de las categorías de conjuntos**

   Como opción para la identificación de las categorías de documentos, se puede proponer el uso de reglas de asociación, para encontrar relaciones entre términos que permitan la identificación del contenido textual presente en los documentos.

4. **Minería de estructura web**

   A partir de bases de datos web como las utilizadas por los sistemas de recuperación de información para el almacenamiento de las URL y de la relación de vínculos entre sí, se pueden aplicar técnicas de minería de estructura web para encontrar relaciones entre las temáticas publicadas dentro de grupos de personas específicos lo cual puede identificar redes sociales que pueden ser de expertos, de temas de investigación o también relacionados con los diferentes roles de la estructura organizacional.

5. **Lematización**

   Integración de un componente de lematización para multilenguaje usando una ontología de idiomas como "Eurowordnet"[1], lo cual puede ayudar a mejorar la asociación de los sinónimos dentro de un mismo idioma y en diferentes idiomas. Las bases multilenguaje como Eurowordnet, permitirían mejorar el preprocesamiento de los documentos de texto antes de su agrupamiento automático de documentos.

6. **Corrección de ortografía**

   Uso de herramientas para la corrección de ortografía como el API de Google para mejorar las consultas hechas por los usuarios cuando ingresan términos dentro de la interfaz de usuario de un buscador web. Con esta estrategia el usuario puede asegurar que si está buscando lo que necesita debido a que el sistema ofrecería sugerencias de palabras de búsqueda que si están escritas con la ortografía correcta, mitigando así el error humano.

7. **Web Semántica**

   El uso de estándares de metadatos en páginas del portal web para uso de web semántica podría ser una estrategia para la implementación de algoritmos de agrupamiento mejorados que puedan procesar metadatos semánticos. El uso de metadatos semánticos permiten la identificación de ontologías y taxonomías de los temas tratados en conjuntos de documentos, por lo cual se podrían identificar relaciones conceptuales entre diferentes documentos de formas más certeras y ofrecer a los usuarios nombres de categorías de documentos más cercanos al vocabulario manejado por quienes se relacionan con el conjunto de documentos. El agrupamiento de acuerdo a metadatos semánticos puede llegar a mostrar relaciones entre la información a través de diferentes elementos como los autores, los temas y las palabras clave.

8. **Interfaz de Navegación**

   La interfaz de navegación puede ser mejorada a través de la implementación de métricas que permitan describir de forma gráfica las relaciones entre diferentes documentos y entre diferentes categorías de documentos, de tal forma que los usuarios puedan identificar gráficamente el nivel de relación entre diferentes elementos de la colección de documentos.

[1] Eurowordnet `http://www.illc.uva.nl/EuroWordNet/`

Bibliografía

[1] Adamic, Lada A. ; Huberman, Bernardo A.: The Web's Hidden Order [En línea]. [Estados Unidos]. Hewlett-Packard Labs / Disponible desde Internet: <URL: http://www.hpl.hp.com/research/papers/weborder.pdf>. 2001. – Informe de Investigación

[2] Arco, L. ; Bello, R. ; Mederos, J.M. ; Pérez, Y.: Agrupamiento de Documentos Textuales mediante Métodos Concatenados. En: *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial* (2006), p. 43–53

[3] Baeza-Yates, R. ; Navarro, G.: XQL and proximal nodes. En: *Journal of the American Society for Information Science and Technology* 53 (2002), Nr. 6, p. 504–514

[4] Baeza-Yates, R. ; Ribeiro-Neto, B. [u. a.]: *Modern information retrieval*. Addison-Wesley. Harlow, Inglaterra, 1999

[5] Baker, J. *UCLA-NSF Social Aspects of Digital Libraries Workshop*. [En línea][California, Estados Unidos]: enero 1998. [Citado 8 Agos., 2007] Disponible desde Internet: <URL: http://is.gseis.ucla.edu/research/dig_libraries/index.html>

[6] Barrett, R.: *Templates for the solution of linear systems: building blocks for iterative methods*. Society for Industrial Mathematics. Filadelfia, Estados Unidos, 1994

[7] Bennett, Mark: Where Have All the Filters Gone? En: *New Idea engineering Inc.* 4 (2007), Junio, Nr. 3

[8] Berry, M.W. ; Browne, M.: *Understanding search engines: mathematical modeling and text retrieval*. Society for Industrial Mathematics for Industrial & Applied Math. Filadelfia, Estados Unidos, 1999

[9] Berry, M.W. ; Castellanos, M.: *Survey of text mining II: clustering, classification, and retrieval*. Springer-Verlag New York Inc, 2007

[10] Berry, M.W. ; Dumais, S.T. ; O'Brien, G.W.: Using linear algebra for intelligent information retrieval. En: *SIAM review* 37 (1995), Nr. 4, p. 573–595

[11] Bloehdorn, S. ; Cimiano, P. ; Duke, A. ; Haase, P. ; Heizmann, J. ; Thurlow, I. ; Völker, J.: Ontology-based question answering for digital libraries. En: *Research and Advanced Technology for Digital Libraries* (2007), p. 14–25

[12] Boldi, P. ; Codenotti, B. ; Santini, M. ; Vigna, S.: Ubicrawler: A scalable fully distributed web crawler. En: *Software: Practice and Experience* 34 (2004), Nr. 8, p. 711–726

[13] Brin, Sergey ; Page, Lawrence: The anatomy of a large-scale hypertextual Web search engine. En: *Computer Networks and ISDN Systems* 30 (1998), April, p. 107–117

[14] Buchanan, George.: FRBR: enriching and integrating digital libraries. En: *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*. Chapel Hill, NC, USA : ACM, 2006. – ISBN 1–59593–354–9, p. 260–269

[15] Buchanan, George ; Bainbridge, David ; Don, Katherine J. ; Witten, Ian H.: A new framework for building digital library collections. En: *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*. Denver, CO, Estados Unidos : ACM, 2005. – ISBN 1–58113–876–8, p. 23–31

[16] Cafarella, M. ; Cutting, D.: Building Nutch: Open Source Search. En: *ACM Queue* 2 (2004), Nr. 2, p. 54–61

[17] Calado, Pavel P. ; Gonzalves, Marcos A. ; Fox, Edward A. ; Ribeiro-Neto, Berthier ; Laender, Alberto H. F. ; da Silva, Altigran S. ; Reis, Davi C. ; Roberto, Pablo A. ; Vieira, Monique V. ; Lage, Juliano P.: The Web-DL environment for building digital libraries from the Web. En: *Third ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'03)*. Houston, Texas, Estados Unidos : IEEE Computer Society, 2003. – ISBN 0–7695–1939–3, p. 346–357

[18] Castelli, D. ; Pagano, P.: OpenDLib: A digital library service system. En: *Research and Advanced Technology for Digital Libraries* (2002), p. 327–340

[19] Chakrabarti, S.: *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann. San Francisco, Estados Unidos, 2003

[20] Coombs, J.H. ; Renear, A.H. ; DeRose, S.J.: Markup systems and the future of scholarly text processing. En: *Communications of the ACM* (1987)

[21] Dorohonceanu, B.: A Practical Suffix-Tree Implementation for String Searches. En: *Dr. Dobb´s Journal* (2000), p. 133

[22] Dublin Core Metadata Initiative, DCMI. *Dublin Core Metadata Element Set, Version 1.1.* [En línea][Universidad de Tsukuba, Japón] [Citado 13 de julio, 2008] Disponible desde Internet: <URL: http://dublincore.org/documents/dces/ >

[23] Ferrucci, A.: UIMA: an architectural approach to unstructured information processing in the corporate research environment. En: *Natural Language Engineering* 10 (2004), Nr. 3-4, p. 327–348

[24] Finn, A. ; Kushmerick, N. ; Smyth, B.: Fact or fiction: Content classification for digital libraries. En: *DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*, 2001

[25] Fox, Edward A. ; Akscyn, Robert M. ; Furuta, Richard K. ; Leggett, John J.: Digital Libraries. En: *Communications of the ACM* 38 (1995), Nr. 4, p. 22–28

[26] Franklin, M.J. ; Jónsson, B.T. ; Kossmann, D.: Performance tradeoffs for client-server query processing. En: *Proceedings of the 1996 ACM SIGMOD international conference on Management of data* ACM, 1996, p. 160

[27] Furuta, R.: Defining and using structure in digital documents. En: *Proceedings of the First Annual Conference on the Theory and Practice of Digital Libraries*, 1994

[28] Furuta, R. ; Quint, V. ; Andre, J.: Interactively editing structured documents. En: *Electronic Publishing* 1 (1988), Nr. 1, p. 19–44

[29] Goldfarb, C.F. ; Prescod, P.: *XML handbook*. Prentice Hall PTR Upper Saddle River, NJ, Estados Unidos, 2000

[30] Gonçalves, M. ; France, R. ; Fox, E.: MARIAN: Flexible interoperability for federated digital libraries. En: *Research and Advanced Technology for Digital Libraries* (2001), p. 173–186

[31] Gonçalves, Marcos ; Fox, Edward ; Krowne, A. ; Calado, P. ; Laender, AHF ; da Silva, AS ; Ribeiro-Neto, B.: The effectiveness of automatically structured queries in digital libraries. En: *Digital Libraries, 2004. Proceedings of the 2004 Joint ACM/IEEE Conference on* (2004), p. 98–107

[32] Gonçalves, Marcos A.: *Streams, Structures, Spaces, Scenarios, and Societies: A Formal Framework for Digital Libraries and Its Applications: Defining a Quality Model fopr Digital Libraries.* Blacksburg, Virginia, Virginia Tech CS Department, Tesis de Grado, 2004. – Disponible desde Internet: <URL: http://scholar.lib.vt.edu/theses/available/etd-12052004-135923/ >

[33] Gonçalves, Marcos A. ; Fox, Dr. Edward A. ; Watson, Dr. Layne T. ; Kipp, Neill A.: Streams, Structures, Spaces, Scenarios, Societies (5S): A Formal Model for Digital Libraries / Computer Science, Virginia Tech. 2001. – Informe de Investigación

[34] Gorelov, Vadim Paz M. ; Ángel F. Zazo ; Figuerola, Carlos G. ; Berrocal, José Luis A.: Librerías Lucene y dotLucene para Recuperación de Información. Estudio y desarrollo de casos prácticos / Departamento de Informática y Automática - Universidad de Salamanca. 2007. – Informe de Investigación

[35] Grady, Robert B. ; Caswell, Deborah L.: *Software metrics: establishing a company-wide program.* Upper Saddle River, NJ, USA : Prentice-Hall, Inc., 1987. – ISBN 0–13–821844–7

[36] Grefenstette, Greg: Comparing Two Language Identification Schemes. En: *The proceedings of 3rd International Conference on Statistical Analysis of Textual Data (JADT 95)*, 1995

[37] Gumpenberger, Christian: The EPrints story: Southampton as the cradle of institutional self-archiving. En: *GMS Medizin - Bibliothek - Information* 9 (2009), Nr. 1, p. 1–6

[38] Gutwin, Carl ; Paynter, Gordon ; Witten, Ian ; Nevill-Manning, Craig ; Frank, Eibe: Improving browsing in digital libraries with keyphrase indexes. En: *Decision Support Systems* 27 (1999), November, p. 81–104

[39] Hammouda, K.M. ; Kamel, M.S.: Efficient phrase-based document indexing for Web document clustering. En: *Knowledge and Data Engineering, IEEE Transactions on* 16 (2004), p. 1279–1296. – ISSN 1041–4347

[40] Hatcher, E. ; Gospodnetic, O.: *Lucene in action.* Manning. Greenwich, CT, Estados Unidos, 2005

[41] Hearst, Marti A.: *Search User Interfaces.* Cambridge University Press. Cambridge, Reino Unido, 2009

[42] Heydon, A. ; Najork, M.: Mercator: A scalable, extensible web crawler. En: *World Wide Web* 2 (1999), Nr. 4, p. 219–229

[43] Hilbert, D.M. ; Redmiles, D.F.: Extracting usability information from user interface events. En: *ACM Computing Surveys (CSUR)* 32 (2000), Nr. 4, p. 384–421

[44] Hodge, V.J. ; Austin, J.: A comparison of standard spell checking algorithms and a novel binary neural approach. En: *IEEE Transactions on Knowledge and data engineering* (2003), p. 1073–1081

[45] Hopkins, Jon: Component primer. En: *Commun. ACM* 43 (2000), Nr. 10, p. 27–30. – ISSN 0001–0782

[46] Hotho, A. ; Nürnberger, A. ; Paass, G.: A brief survey of text mining. En: *LDV Forum-GLDV Journal for Computational Linguistics and Language Technology* Vol. 20, 2005, p. 19–62

[47] Hull, D.A. ; Grefenstette, G.: Experiments in multilingual information retrieval. En: *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* Citeseer, 1996, p. 49–57

[48] Jacquemont, S. ; Jacquenet, F. ; Sebban, M.: Correct your text with Google. En: *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence* IEEE Computer Society, 2007, p. 170–176

[49] Jones, S. ; Jones, M. ; Barr, M. ; Keegan, T.T.: Searching and browsing in a digital library of historical maps and newspapers. En: *Journal of Digital Information* 6 (2004), Nr. 2

[50] Kalinichenko, LA ; Briukhov, DO ; Skvortsov, NA ; Zakharov, VN: Infrastructure of the subject mediating environment aiming at semantic interoperability of heterogeneous digital library collections. En: *Procedings of the Second Russian National Conference on"Digital Libraries: Advanced Methods and Technologies, Digital Collections* Citeseer, 2000

[51] Kochtanek, T.R. ; Hein, K.K.: Delphi study of digital libraries. En: *Information Processing and Management* 35 (1999), Nr. 3, p. 245–254

[52] Koohang, A.: Development and validation of an instrument for assessing users' views about the usability of digital libraries. En: *Issues in Informing Science and Information Technology* 1 (2004), p. 55–63

[53] Kosala, Raymond ; Blockeel, Hendrik: Web mining research: a survey. En: *SIGKDD Explor. Newsl* 2 (2000), p. 1–15

[54] Krutchen, P.B.: The 4+ 1 view model of architecture. En: *IEEE Software* 12 (1995), Nr. 6, p. 42–50

[55] Lagoze, Carl ; Payette, Sandy ; Shin, Edwin ; Wilper, Chris: Fedora: an architecture for complex objects and their relationships. En: *Int. J. Digit. Libr.* 6 (2006), Nr. 2, p. 124–138. – ISSN 1432–5012

[56] Lesk, M.: *Understanding Digital Libraries.* Morgan Kaufmann. San Francisco, CA, Estados Unidos, 2004

[57] Levy, D.M. ; Marshall, C.C.: Going digital: A look at assumptions underlying digital libraries. En: *Communications of the ACM* 38 (1995), Nr. 4, p. 77–84

[58] Licklider, JCR: *Libraries of the Future.* The MIT Press, 1965

[59] Likert, R.: A technique for the measurement of attitudes. En: *Archives of Psychology* 22 (1932), Nr. 140, p. 1–55

[60] Lossau, Norbert ..: Search Engine Technology and Digital Libraries. En: *D-Lib Magazine* 10 (2004), Nr. 6

[61] Luhn, H.P.: The automatic creation of literature abstracts. En: *IBM Journal of research and development* 2 (1958), Nr. 2, p. 159–165

[62] Mahecha-Nieto, Isabel A. ; León, Elizabeth.: Digital Web Library of a Website With Document Clustering. En: *Proceedings of the 12th Ibero-American Conference on AI, IBERAMIA 2010* Vol. 6433, 2010, p. 214–223

[63] Mahecha-Nieto, Isabel A. ; León, Elizabeth.: An exploratory study of open source search engines: Evaluation of supportability, usability, functionality and performance. En: *Quinto Congreso Colombiano de Computación 2010* (2010)

[64] Manning, Christopher D. ; Raghavan, Prabhakar ; Schütze, Hinrich: *Introduction to Information Retrieval*. Cambridge University Press, 2008

[65] Marchionini, G.: *Information seeking in electronic environments*. Cambridge University Press, 1997

[66] Marcum, Deanna.: Requirements for the Future Digital Library. En: *The Journal of Academic Librarianship* 29 (2003), p. 276

[67] McCreight, Edward M.: A Space-Economical Suffix Tree Construction Algorithm. En: *J. ACM* 23 (1976), Nr. 2, p. 262–272. – ISSN 0004–5411

[68] McNatt, W.B. ; Bieman, J.M.: Coupling of design patterns: Common practices and their benefits. En: *Computer Software and Applications Conference, 2001. COMPSAC 2001. 25th Annual International* IEEE, 2002, p. 574–579

[69] Middleton, Christian. ; Baeza-Yates, Ricardo ..: A Comparison of Open Source Search Engines / Universitat Pompeu Fabra Departamento de Tecnología. 2007. – Informe de Investigación

[70] Mohr, G. ; Stack, M. ; Rnitovic, I. ; Avery, D. ; Kimpton, M.: Introduction to Heritrix. En: *4th International Web Archiving Workshop*, 2004

[71] de Moura, E.S. ; Navarro, G. ; Ziviani, N. ; Baeza-Yates, R.: Fast and flexible word searching on compressed text. En: *ACM Transactions on Information Systems (TOIS)* 18 (2000), Nr. 2, p. 113–139

[72] Nelson, M.: Fast string searching with suffix trees. En: *Dr. Dobb´s Journal* (1996), p. 115–119

[73] Nielsen Mega View Search, NMVS: Top 10 U.S. Search Providers, Home & Work August 2010 [En línea]. [Estados Unidos]: Agosto de 2010 [Citado 12 sept., 2010]. En: *Disponible desde Internet: <URL: http://en-us.nielsen.com/content/nielsen/en_us/insights/rankings/internet.html>*

[74] NISO, the National Information Standards O. *ANSI/NISO Z39.85 - The Dublin Core Metadata Element Set*. 2007

[75] Osinski, S.: *An algorithm for clustering of web search results*, Poznan University, Tesis de Grado, 2003

[76] Osiriski, S. ; Stefanowski, J. ; Weiss, D.: Lingo: Search results clustering algorithm based on singular value decomposition. En: *Intelligent information processing and web mining: proceedings of the International IIS: IIPWM'04 Conference held in Zakopane, Poland, May 17-20, 2004* Springer Verlag, 2004, p. 359

[77] Osiriski, S. ; Weiss, D.: Conceptual clustering using lingo algorithm: Evaluation on open directory project data. En: *Intelligent information processing and web mining: proceedings of the International IIS: IIPWM'04 Conference held in Zakopane, Poland, May 17-20, 2004* Springer Verlag, 2004, p. 369

[78] Palotai, Zsolt ; Gábor, Bálint ; Lörincz, András: Adaptive highlighting of links to assist surfing on the Internet. En: *International Journal of Information Technology & Decision Making* 4 (2005), März, p. 117–139. – ISSN 02196220

[79] Pandey, Richa: Digital Library Architecture. En: *DRTC Workshop on Digital Libraries: Theory & Practice, Bangalore, India* (2003), March

[80] Payette, S. ; Staples, T.: The Mellon Fedora Project Digital Library Architecture Meets XML and Web Services. En: *Research and Advanced Technology for Digital Libraries* (2002), p. 87–106

[81] Perera, Tyrell: Nutch - The Java Search Engine. [En línea] / The Apache Software Foundation. [Citado 4 marzo, 2008] Disponible dede Internet: <URL http://wiki.apache.org/nutch/Nutch_-_The_Java_Search_Engine >. – Informe de Investigación

[82] Prager, J.M.: Linguini: Language identification for multilingual documents. En: *Journal of Management Information Systems* 16 (1999), Nr. 3, p. 71–101

[83] Rasmussen, Edie M.: Clustering Algorithms. En: *Information Retrieval: Data Structures & Algorithms*. 1992, p. 419–442

[84] Rauber ; Merkl: Text Mining in the SOMLib Digital Library System: The Representation of Topics and Genres. En: *Applied Intelligence* 18 (2003), Mai, p. 271–293

[85] Reddy, R. ; Wladawsky-Berger, I.: Digital Libraries: Universal Access to Human Knowledge-A Report to the President. En: *PITAC, Panel on DLs* (2001)

[86] Riboni, D.: Feature selection for web page classification. En: *EURASIA-ICT 2002 Proceedings of the Workshop* Citeseer, 2002, p. 473–477

[87] Rocchio, J. J.: Relevance feedback in information retrieval. En: Salton, G. (Ed.): *The Smart retrieval system - experiments in automatic document processing*. Englewood Cliffs, NJ: Prentice-Hall, 1971, p. 313–323

[88] Rüger, S.: Putting the user in the loop: visual resource discovery. En: *Adaptive Multimedia Retrieval: User, Context, and Feedback* (2006), p. 1–18

[89] Sahami, M. ; Yusufali, S. ; Baldonaldo, M.Q.W.: Sonia: A service for organizing networked information autonomously. En: *Proceedings of the third ACM conference on Digital libraries* ACM, 1998, p. 200–209

[90] Salton, G.: The SMART Retrieval System, Experiments in Automatic Document Processing. (1971)

[91] Salton, G. ; Buckley, C.: Term-weighting approaches in automatic text retrieval. En: *Information processing & management* 24 (1988), Nr. 5, p. 513–523

[92] Salton, G. ; McGill, M.J.: *Introduction to modern information retrieval*. McGraw-Hill New York, 1983

[93] Salton, G. ; Wong, A. ; Yang, C.S.: A vector space model for automatic indexing. En: *Communications of the ACM* 18 (1975), Nr. 11, p. 620

[94] Salton, Gerald (Ed.): *Automatic text processing*. Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 1988. – ISBN 0–2:1–1227–8

[95] Sanchez, D. ; Moreno, A.: Bringing taxonomic structure to large digital libraries. En: *International Journal of Metadata, Semantics and Ontologies* 2 (2007), Nr. 2, p. 112–122

[96] Spivey, J.M: *Introducing Z: A Specification Language and its Formal Semantics*. Cambridge University Press, 1988

[97] International Organization for Standardization, ISO. *ISO 15836:2009*. 2009

[98] Strehl, A. ; Ghosh, J. ; Mooney, R.: Impact of similarity measures on web-page clustering. En: *Workshop on Artificial Intelligence for Web Search (AAAI 2000)*, 2000, p. 58–64

[99] Tan, P.N. ; Steinbach, M. ; Kumar, V.: *Introduction to data mining*. Pearson Addison Wesley Boston, 2006

[100] Tansley, Robert ; Bass, Mick ; Smith, MacKenzie: DSpace as an Open Archival Information System: Current Status and Future Directions. En: *Research and Advanced Technology for Digital Libraries*. Springer, 2003, p. 446–460

[101] The Apache Software Foundation, TASF: Nutch. [En línea] [Estados Unidos][Citado 20 mayo 2007] Disponible desde Internet: <URL: http://wiki.apache.org/nutch/>. – Informe de Investigación

[102] Turtle, H. ; Croft, W.B.: Evaluation of an inference network-based retrieval model. En: *ACM Transactions on Information Systems (TOIS)* 9 (1991), Nr. 3, p. 187–222

[103] Ukkonen, E.: On-line construction of suffix trees. En: *Algorithmica* 14 (1995), p. 249–260. – 10.1007/BF01206331. – ISSN 0178–4617

[104] Van Rijsbergen, CJ ; Robertson, SE ; Porter, MF: New models in probabilistic information retrieval. En: *British Library Research and Development Report* 5587 (1980)

[105] Waegel, Daniel B. ; Kontostathis, April: TextMOLE: text mining operations library and environment. En: *SIGCSE BULLETIN*. Houston, Texas, USA : ACM Press, 2006. – ISBN 1–59593–259–3, p. 553–557

[106] Waller, WG ; Kraft, D.H.: A mathematical model of a weighted Boolean retrieval system. En: *Information Processing & Management* 15 (1979), Nr. 5, p. 235–245

[107] Wang, B.: A hybrid system approach for supporting digital libraries. En: *International Journal on Digital Libraries* 2 (1999), Nr. 2, p. 91–110

[108] Weiner, P.: Linear pattern matching algorithms. En: *IEEE Conference Record of 14th Annual Symposium on Switching and Automata Theory, 1973. SWAT'08*, 1973, p. 1–11

[109] Weiss, D.: *A clustering interface for web search results in polish and english*, Poznan University of Technology, Tesis de Grado, 2001

[110] Witten, Ian H.: How to build a digital library using open-source software. En: *Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries*. Portland, Oregon, USA : ACM, 2002. – ISBN 1–58113–513–0, p. 416–416

[111] Witten, Ian H. ; Bainbridge, David: Building digital library collections with greenstone. En: *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*. Denver, CO, USA : ACM, 2005. – ISBN 1–58113–876–8, p. 425–425

[112] Witten, Ian H. ; Bainbridge, David: A retrospective look at Greenstone: lessons from the first decade. En: *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*. Vancouver, BC, Canada : ACM, 2007. – ISBN 978–1–59593–644–8, p. 147–156

[113] Witten, Ian H. ; Boddie, Stefan J. ; Bainbridge, David ; McNab, Rodger J.: Greenstone: a comprehensive open-source digital library software system. En: *Proceedings of the fifth ACM conference on Digital libraries*. San Antonio, Texas, United States : ACM, 2000. – ISBN 1–58113–231–X, p. 113–121

[114] Witten, Ian H. ; Don, Katherine J. ; Dewsnip, Michael ; Tablan, Valentin: Text mining in a digital library. En: *International Journal on Digital Libraries* 4 (2004), p. 56–59. – ISSN 14325012

[115] Witten, Ian H. ; Loots, Michel ; Trujillo, Maria F. ; Bainbridge, David: The promise of digital libraries in developing countries. En: *Commun. ACM* 44 (2001), p. 82–85

[116] Witten, I.H. ; Gori, M. ; Numerico, T.: *Web Dragons: Inside the Myths of Search Engine Technology*. Morgan Kaufmann, 2007

[117] World Wide Web Consortium, W3C. *Resource Description Framework (RDF) [En línea]*. Disponible desde Internet: <URL: http://www.w3.org/standards/techs/rdf#w3c_all >. 2004

[118] Zamir, O. ; Etzioni, O.: Web document clustering: A feasibility demonstration. En: *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* ACM, 1998. – ISBN 1581130155, p. 46–54

[119] Zamir, O. ; Etzioni, O.: Grouper: a dynamic clustering interface to Web search results. En: *Computer Networks* 31 (1999), Nr. 11-16, p. 1361–1374

[120] Zamir, O.E.: *Clustering web documents: a phrase-based method for grouping search engine results*, University of Washington, Tesis de Grado, 1999

[121] Zhang, D. ; Dong, Y.: Semantic, hierarchical, online clustering of web search results. En: *Advanced Web Technologies and Applications* (2004), p. 69–78

[122] Zhao, Y. ; Karypis, G. ; Fayyad, U.: Hierarchical clustering algorithms for document datasets. En: *Data Mining and Knowledge Discovery* 10 (2005), Nr. 2, p. 141–168. – ISSN 1384–5810

[123] Ziviani, N. ; Silva de Moura, E. ; Navarro, G. ; Baeza-Yates, R.: Compression: A key for next-generation text retrieval systems. En: *IEEE Computer* 33 (2000), Nr. 11, p. 37–44

[124] Zobel, Justin ; Moffat, Alistair: Inverted files for text search engines. En: *ACM Comput. Surv.* 38 (2006), Nr. 2, p. 6. – ISSN 0360–0300

Índice alfabético

A
Afijos derivativos, xiv, 27

B
Buscador web, 1, 2, 21, 40, 41, 56, 60, 67, 77, 87, 92–94, 97, 105

C
Código abierto, xv, 3, 12, 13, 23, 24, 28, 38, 40, 42, 56–58
Colección, xiv, 2, 5, 8, 10, 17, 18, 20, 27, 29
Colección de documentos, 10, 16, 19, 21, 25, 42, 57, 58

D
DNS, xiv, 12

E
Espacio Euclidiano, xiv, 16

F
Forma flexiva, xiv
Forma gramatical, xiv

G
Google, 1, 12, 21–23, 26, 28, 60, 77, 105

L
Librerías digitales, 1, 2, 4–11, 14, 15, 22, 25, 35, 65–67, 75
Librerías digitales web, 2, 7, 8, 11

M
Metadatos, xiv, 12–14, 41, 45, 75, 77
Morfema flexivo, xiv

N
Normalización de un vector, xiv

P
Proceso heurístico, xv, 27

R
Recuperación de Información, 2–8, 11–14, 16, 21, 24–28, 31, 39, 40, 57

S
Stop words, xv, 5, 18, 26, 27, 34, 76

U
URL, xv, 12–14, 24, 25, 34, 40–42, 58, 77, 97

W
Web, xv, 1, 6, 7, 12–14, 21, 23, 26, 40, 42, 56, 60, 67, 87, 103
Web graph, xv, 59, 93

A MANUAL DE USUARIO

A.1 Requerimientos de hardware

Los mismos que se exigen para Microsoft Windows XP.

A.2 Requerimientos de software

Antes de instalar Nutch se deben descargar desde la Web las siguientes herramientas:

- Nutch 0.9: `http://www.apache.org/dyn/closer.cgi/lucene/nutch/`
- JAVA JDK 6 o superior: `http://java.sun.com/javase/downloads/index.jsp`
- Apache web server 6: `http://tomcat.apache.org/download-60.cgi`
- Para sistemas operativos que no son UNIX se necesita de Cygwin: `http://www.cygwin.com/`

Se deben instalar cada una de las herramientas una por una en el siguiente orden:

1. Primero se instala Cygwin, ejecutando el archivo cygwinSetup.exe. Si se instaló de forma adecuada, cuando se inicia Cygwin, el sistema muestra una ventana como la siguiente:

Figura A.1: Cygwin

2. Segundo se instala JAVA JDK, ejecutando el archivo llamado dk-6u3-windows-i586-p.exe (este nombre puede variar de acuerdo al número de la versión). Luego de efectuar la instalación, de acuerdo la aplicación queda disponible en una carpeta dentro de los Archivos de programa o en la carpeta que haya configurado el usuario, como se muestra en la siguiente imagen. La ruta de dicha carpeta de instalación la debe tener presente el usuario cuando se va a ejecutar el buscador web.

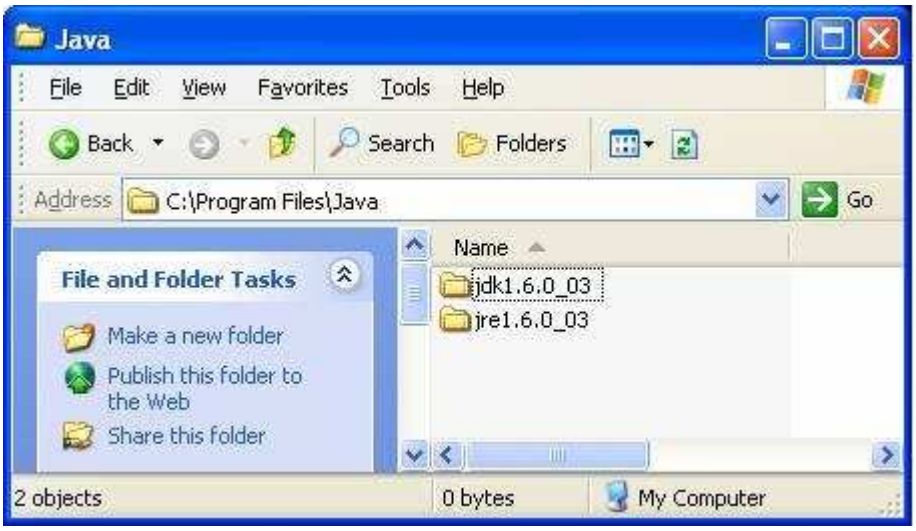


Figura A.2: Java JDK

3. En tercer lugar se instala el servidor de aplicaciones Tomcat, se debe ejecutar el archivo llamado run apache-tomcat-6.0.14.exe (este nombre puede variar de acuerdo al número de la versión). Luego de instalar la aplicación, el servidor se configura para activar el servicio, esto se hace a través de la siguiente ruta: Inicio -> Todos los programas -> Apache Tomcat 6.0 -> Configure Tomcat, como se observa en la siguiente imagen:

Figura A.3: Configure Tomcat

a) Se da Click en el botón “Start” para iniciar Apache Tomcat Service.

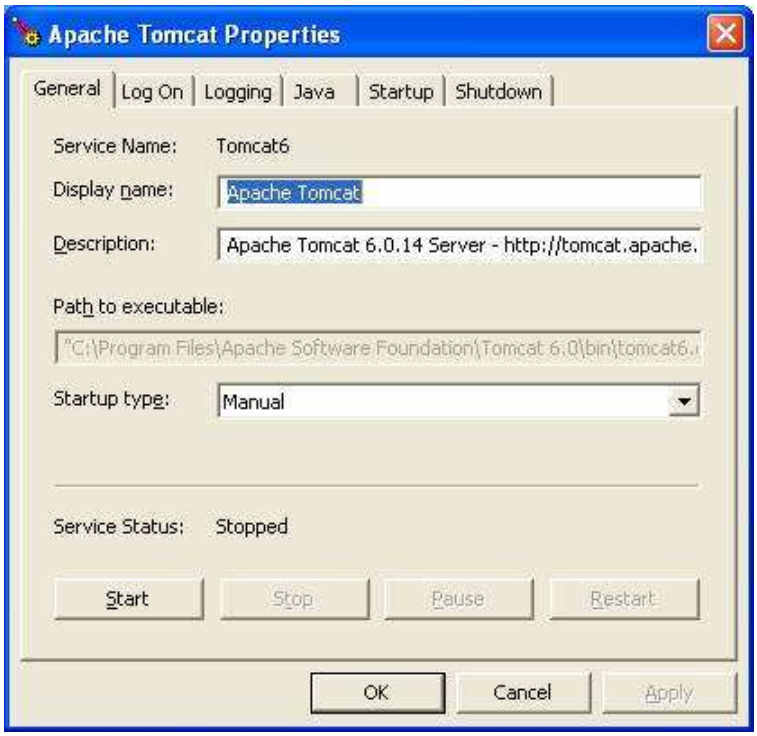


Figura A.4: Iniciar Tomcat

b) Para probar que el servicio de Tomcat está funcionando, en una ventana del navegador web se consulta la dirección: http://localhost:8080 y se debe acceder a la página principal de Tomcat cuya interfaz debe ser similar a la que se observa en la siguiente imagen:

Figura A.5: Tomcat

4. Cuarto, se descomprime el archivo de Nutch que se llama nutch-0.9.tar.gz, y se guardan los archivos en una dirección en un dispositivo local como por ejemplo: c:\nutch.

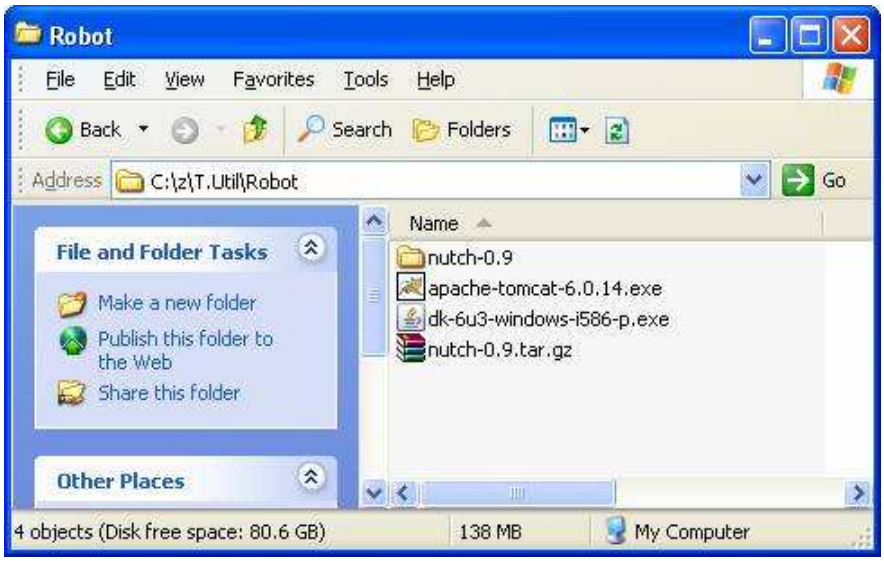


Figura A.6: Nutch

A.3 Ejecución del robot (crawler) del buscador Nutch

1. En una ventana de la aplicación Cygwin, se debe ubicar la carpeta que contiene los archivos de Nutch, y luego se configura la dirección donde se ubica la instalación de JAVA, es decir el "JAVA_HOME" así como se muestra en la siguiente imagen:

Figura A.7: Configuración carpeta de ejecución de Nutch

2. A continuación se debe crear una carpeta dentro de la carpeta de archivos de Nutch, que va a contener el archivo de direcciones que serán recuperadas, en este caso de ejemplo la carpeta creada se llama "urls".

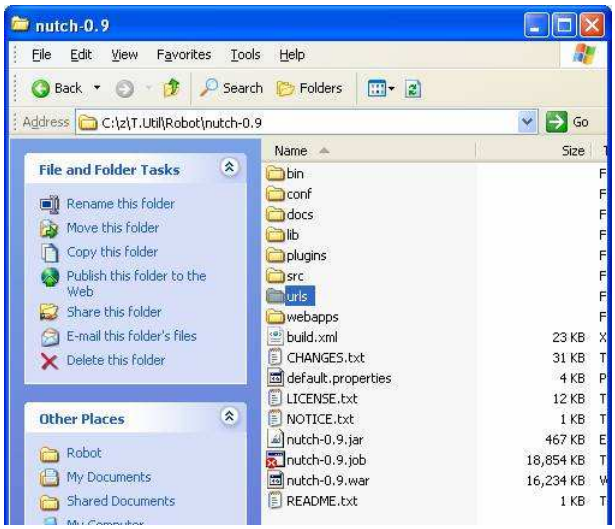


Figura A.8: Carpeta de direcciones

3. En la carpeta que se creó, se debe generar un archivo tipo texto en el cual se encuentre el listado de direcciones como el que se muestra en la siguiente imagen:

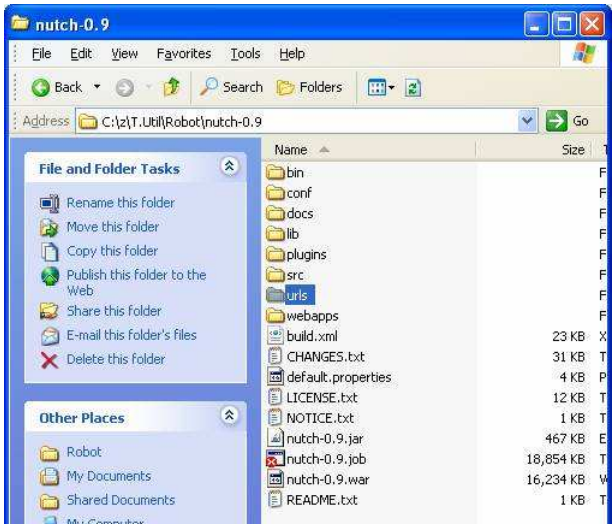


Figura A.9: Lista de direcciones

4. Luego se edita el archivo crawl-urlfilter.txt, que se encuentra en la carpeta "conf" dentro de la carpeta de archivos de Nutch, la ruta debe ser similar a esta: c:\nutch\conf\crawl-urlfilter.txt. Lo que se debe cambiar es la línea que dice "MY.DOMAIN.NAME" con el dominio al cual se limitará el proceso de recuperación. Por ejemplo, si se quiere limitar al dominio apache.org, la línea debe serl la siguiente: +^http://([a-z0-9]*\.)*apache.org/

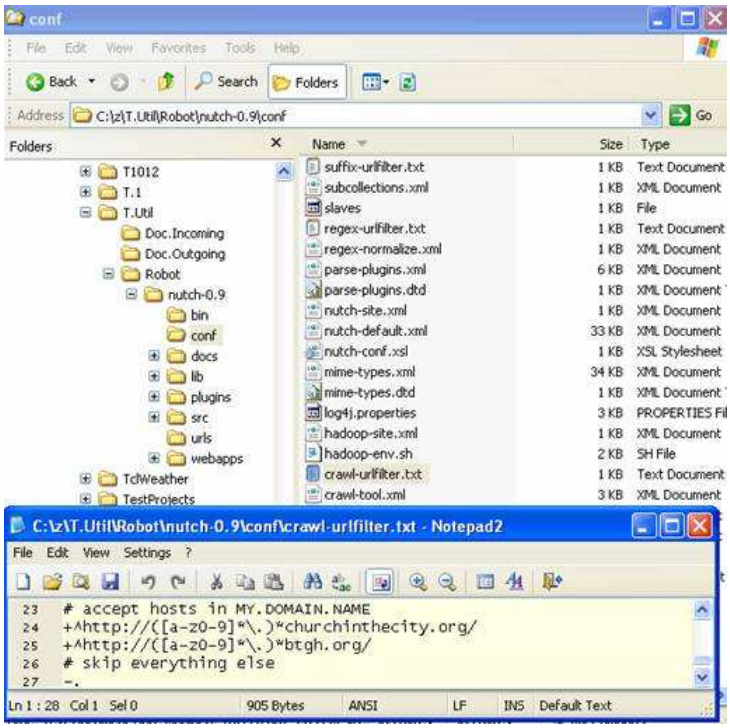


Figura A.10: Restricción del Dominio

5. A continuación se debe editar el archivo que configura el sitio web del buscador, el archivo se llama nutch-site.xml y se ubica en la carpeta "conf" que se ubica dentro de la carpeta de archivos de Nutch, la ruta debe ser similar a esta: c:\nutch\conf\nutch-site.xml, se deben insertar por lo menos las propiedades y sus respectivos valores respecto a quien es el usuario del buscador así como se muestra en el siguiente ejemplo:

<?xml version="1.0"?>

<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!– Put site-specific property overrides in this file. –>

<configuration>

<property>

<name>http.agent.name</name>

<value>Pepe Perez</value>

<description>Pepe Perez

</description>

</property>

<property>

<name>http.agent.description</name>

<value>Nutch buscador</value>

<description> Nutch buscador

</description>

</property>

<property>
<name>http.agent.url</name>
<value>http://pepeperez.paginasweb.com </value>
<description>http://pepeperez.paginasweb.com
</description>
</property>
<property>
<name>http.agent.email</name>
<value>MyEmail</value>
<description>pepeperez@yahoo.com
</description>
</property>
</configuration>

6. Ejecución del robot (crawler).

   a) Luego de configurar todo, la ejecución del robot es sencilla, se utiliza el comando crawl el cual incluye los siguientes parámetros:

      i. -dir: *dir* es el nombre de la carpeta donde se almacenarán los archivos generados durante la ejecución del comando crawl, que son el índice, la base de datos web y los índices intermedios.

      ii. -threads: *threads* es el parámetro que determina el número de hilos de ejecución que serán utilizados en paralelo.

      iii. -depth: *depth* este parámetro indica el grado de profundidad dentro de la *web graph* que será rastreado por el buscador.

      iv. -topN: *N* determina el número máximo de páginas que serán recuperadas por cada nivel en profundidad.

   b) Un ejemplo del uso del comando "crawl" es el siguiente: bin/nutch crawl urls -dir crawl -depth 10 -topN 20

   c) Es recomendable efectuar pruebas para determinar cuales son los valores de los parámetros son los más adecuados según corresponda al caso. Se aconseja comenzar con profundidades pequeñas limitando drásticamente el número de páginas obtenidos en cada nivel (-topN), y evaluando la salida para comprobar que las páginas son las deseadas. La profundidad adecuado para el rastreo completo de un sitio Web es de alrededor de 10. El número de páginas por nivel (-topN) para un rastreo completo puede ser de decenas de miles a millones, dependiendo del caso. A continuación en la siguiente imagen se muestra un ejemplo de lo que aparece en pantalla cuando se ejecuta el comando "crawl":

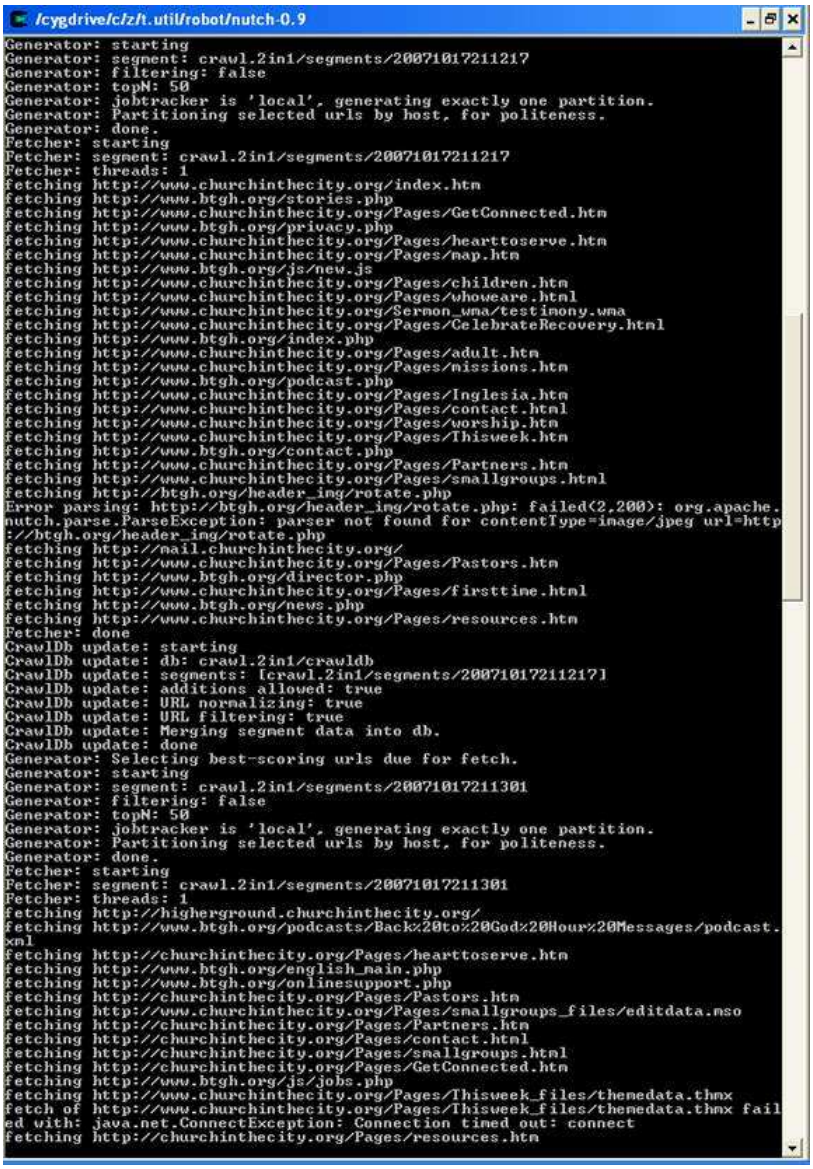


Figura A.11: Salida comando crawl

A.4 Configuración del buscador Nutch

1. Para configurar el buscador Nutch para resolver consultas de acuerdo al contenido que se recupero a través del comando "crawl", se debe configurar la aplicación Web que se ejecuta dentro del servidor Web, que para este caso es Tomcat. Primero se ubica el administrador de Tomcat en la siguiente dirección local: http://localhost:8080/manager/html a través del navegador de Internet. En el campo "WAR file to deploy", se debe ubicar el archivo nutch0.9.war que se encuentra dentro de los archivos de Nutch en una dirección similar a esta: c:\nutch-0.9.

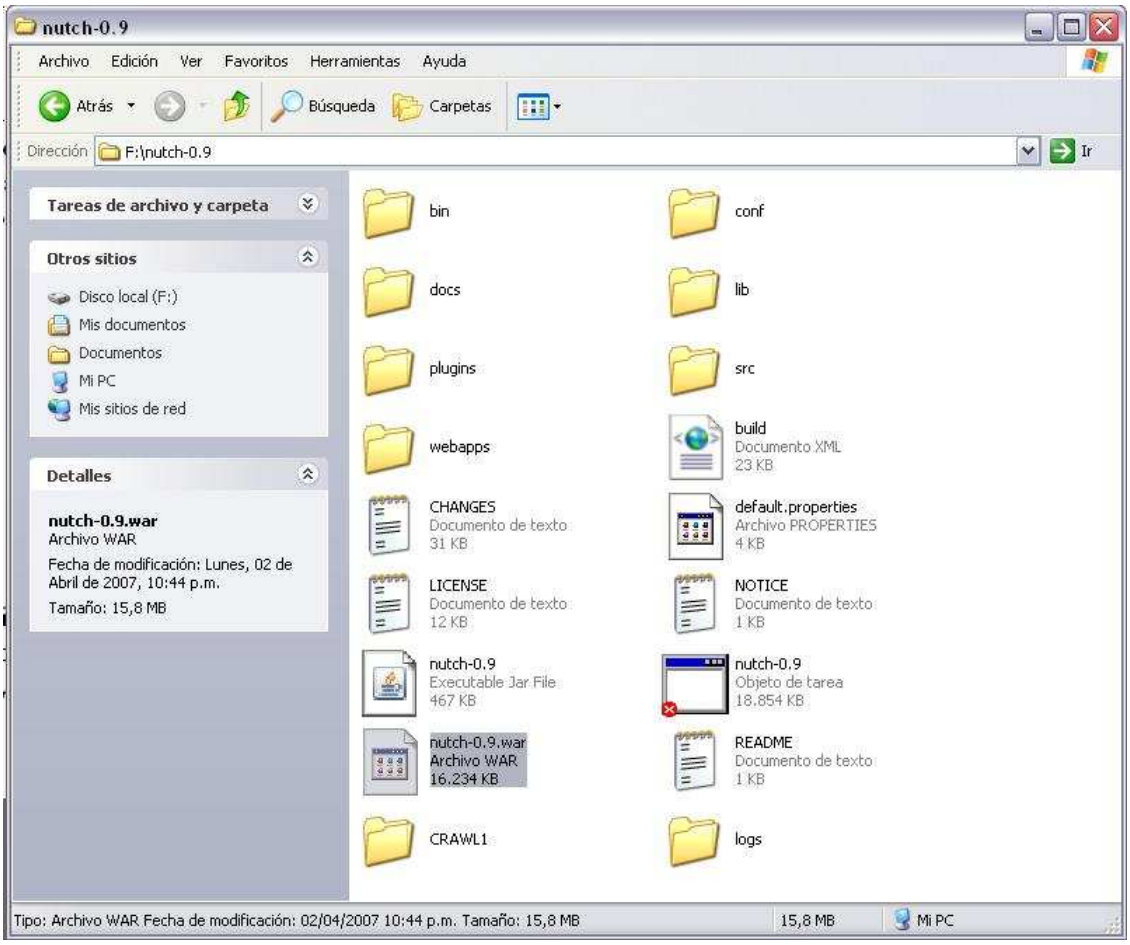


Figura A.12: Aplicación web

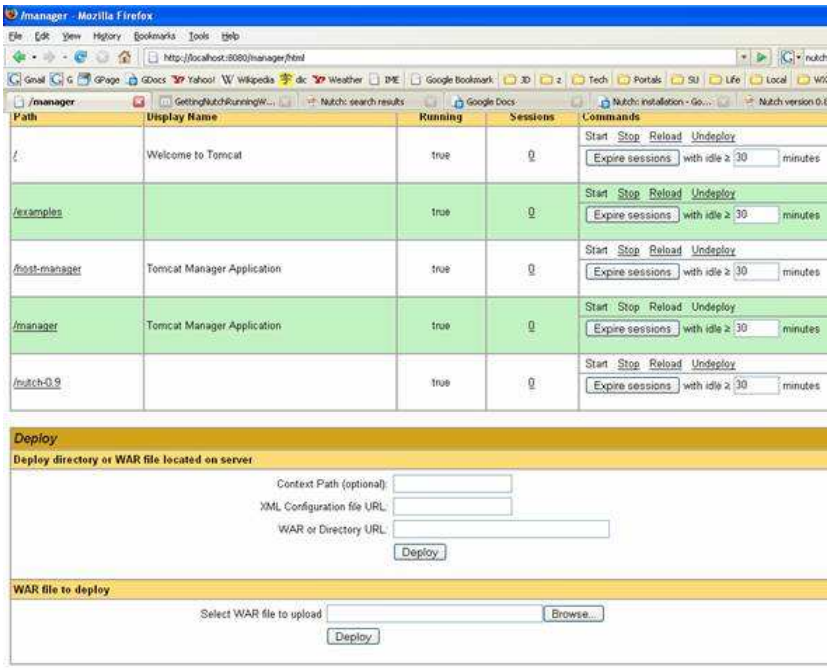


Figura A.13: Administrador de Tomcat

2. Se debe verificar que el archivo nutch-0.9.war ha sido correctamente copiado, dicho archivo debe estar en la carpeta de instalación de Tomcat en la carpeta "webapps", la dirección debe ser similar a esta: C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps.

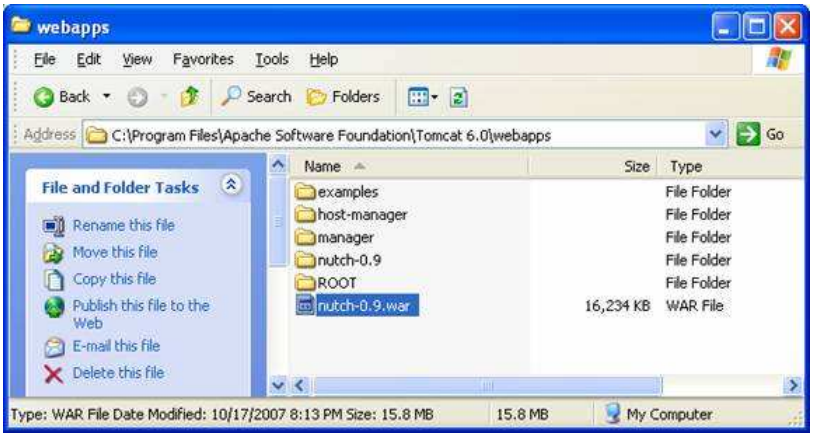


Figura A.14: Aplicación web

3. En el navegador, se escribe la dirección: http://localhost:8080/manager/html, y luego se da click en el botón "Start" que se encuentra en la fila correspondiente a nutch-0.9. A continuación se mostrará la carpeta llamada "nutch-0.9" la cual fue creada en la carpeta " webapps" mostrada en el numeral anterior.

4. Configuración de la carpeta de búsqueda:

a) Primero se ubica la carpeta WEB-INF/classes, que se encuentra en la carpeta webapps\nutch-0.9. En dicha carpeta se encuentra el archivo nutch-site.xml el cual debe ser editado con los siguientes cambios:

<configuration>

<property>

<name>searcher.dir</name>

<value>Dirección de la carpeta que contiene el crawl</value>

</property>

</configuration>

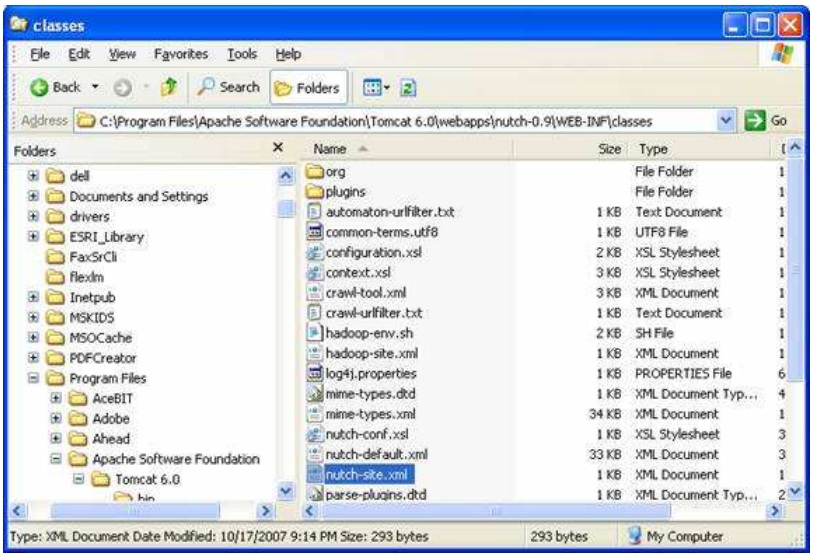


Figura A.15: Configuración del site

b) Por ejemplo, si la carpeta de Nutch está ubicada en la dirección: C:\nutch-0.9.0, y la carpeta que se especificó dentro del comando "crawl" fue: C:\nutch-0.9.0\crawl\, ésta dirección se cambia la frase: "Dirección de la carpeta que contiene el crawl".

5. Luego se debe recargar la aplicación web de Nutch dentro de Tomcat, esto se hace a través de la interfaz de el Administrador de Tomcat en la dirección: http://localhost:8080/manager/html y simplemente se hace click en el botón "Recargar" en Nutch, o reiniciar Tomcat utilizando la herramienta de servicios del servidor.

6. Finalmente para probar el buscador web, se abre una ventana en el navegador web y se introduce la URL: http://localhost:8080/nutch-0.9. A continuación se ejecutará la interfaz para ingresar las consultas de búsqueda como se muestra en la siguiente imagen.

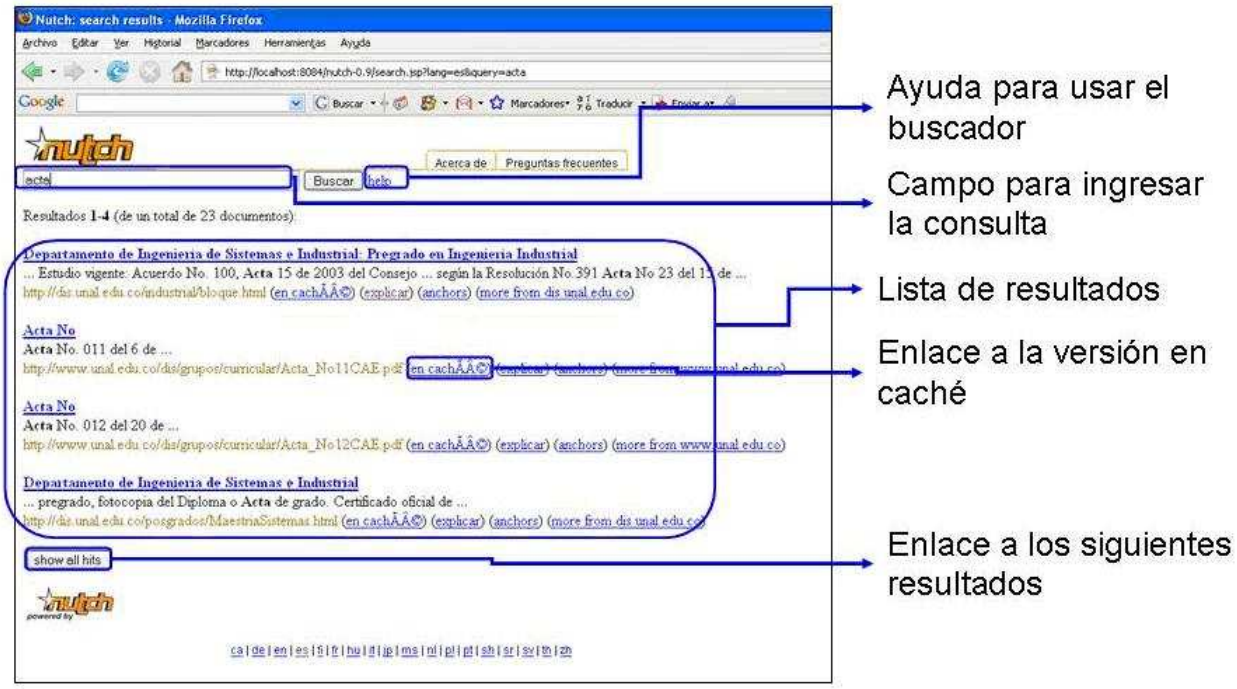


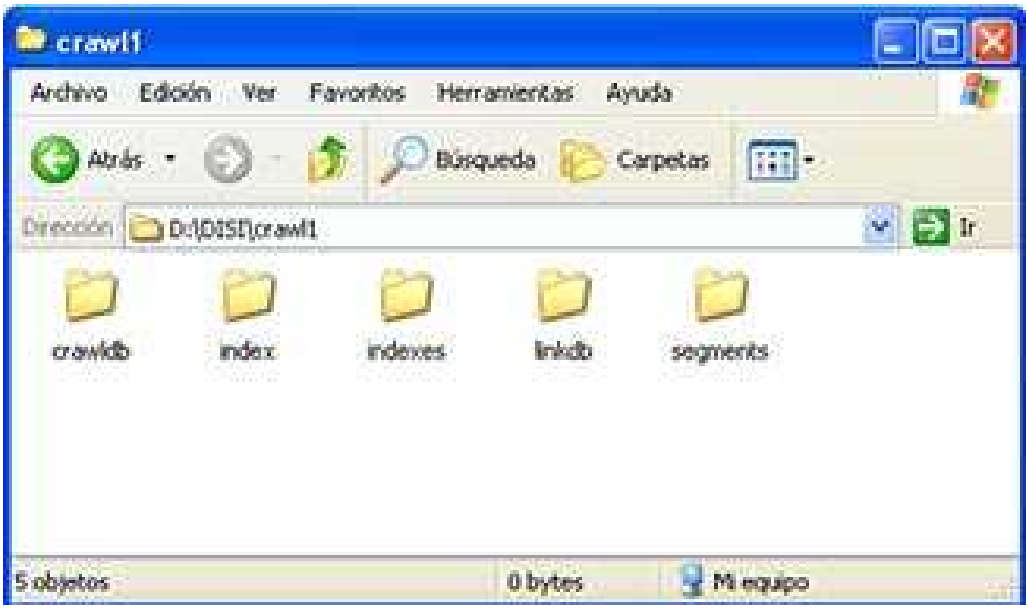
Figura A.16: Ejemplo de consulta

A.5 Generación del archivo de entrada para el proceso de agrupamiento de documentos

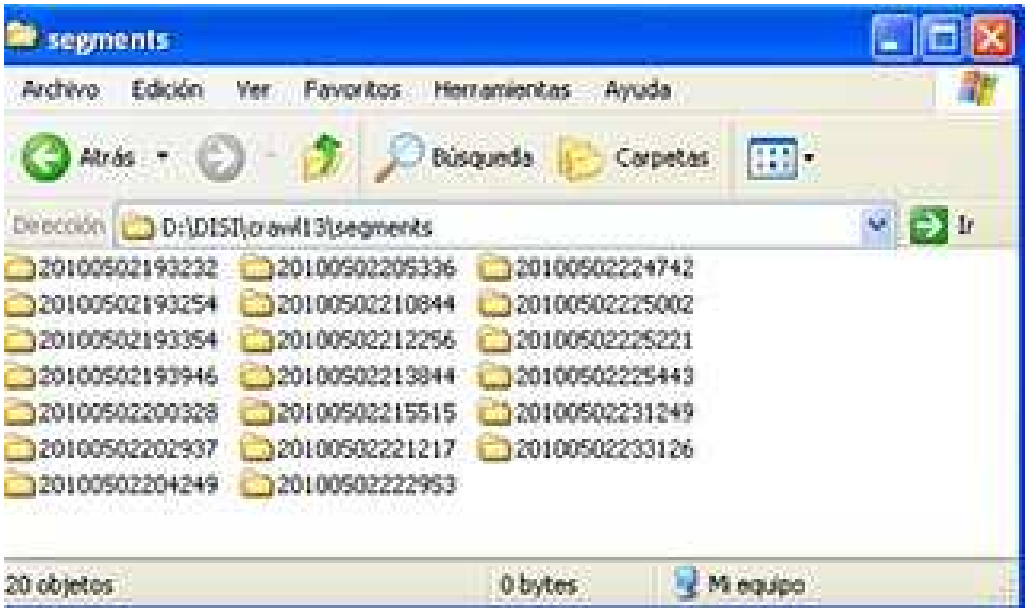
1. En una ventana de la aplicación Cygwin, se debe ubicar la carpeta que contiene los archivos de Nutch, y luego se configura la dirección donde se ubica la instalación de JAVA, es decir el "JAVA_HOME" así como se muestra en la siguiente imagen:

Figura A.17: Configuración carpeta de ejecución de Nutch

2. Se ejecuta el comando *"Segment Merger"* con la siguiente sintaxis: ubicación-del-comando-segment-merger **mergsegs** tipo-de-salida ubicación-de-cada-segmento-del-índice ubicación-del-archivo-de-salida. Los segmentos del índice se ubican dentro de la carpeta "index" ubicada donde



(a) Carpeta de los archivos generados con el comando crawl



(b) Carpeta con los segmentos del índice

Figura A.18: Ubicación de la carpeta "segments"

se almacenaron los archivos generados durante la ejecución del comando crawl (Ver sección A.3), cómo se muestra en la figura A.18, donde la figura A.18a muestra la ubicación de la carpetas generadas durante la ejecución del comando crawl y la figura A.18b muestra un ejemplo de los segmentos de un índice.

3. La línea de comando que debe ejecutarse siguiendo con el ejemplo anterior es: bin/nutch mergesegs crawl/nuevo-segmento crawl/segments/*. Luego de ejecutarse el comando si se ejecutó

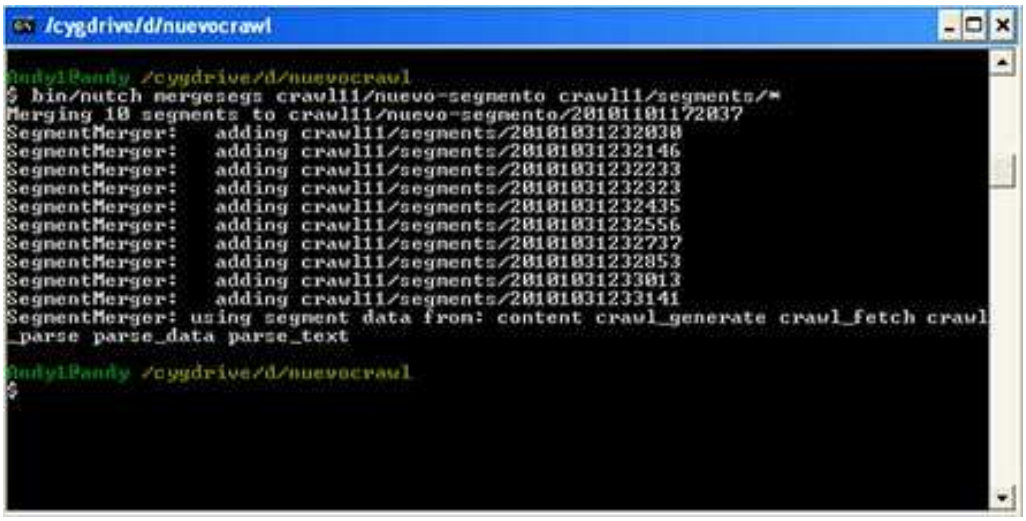


Figura A.19: Ejemplo generación de nuevo segmento con comando "Segment Merger"

de forma exitosa se muestra un mensaje como el que se muestra en la figura A.21. En la figura A.20 se muestra la ubicación del nuevo segmento.

4. Se ejecuta el comando *"Segment Reader"* con la siguiente sintaxis: ubicación-del-comando-segment-reader **readseg** tipo-de-salida ubicación-del-segmento-del-índice ubicación-del-archivo-de-salida opciones-generales. El segmento del índice se ubica donde se creó en el paso anterior. La línea de comando que debe ejecutarse siguiendo con el ejemplo anterior es: bin/nutch readseg -dump crawl/nuevo-segmento crawl/carpeta-de-salida -nofetch -nogenerate -noparsedata -noparsetext. Luego de ejecutarse el comando si se ejecutó de forma exitosa se muestra un mensaje como el que se muestra en la figura A.21.

A.6 Ejecución del proceso de agrupamiento de documentos

1. En el CD que acompaña este documento, en la carpeta donde se encuentra el software llamada "agrupamiento", en dicha carpeta se encuentran los archivos para el proceso de agrupamiento y la interfaz de navegación, en la figura A.22 se muestra el contenido de la carpeta "Agrupamiento".
2. Para ejecutar el proceso de agrupamiento se utiliza una consola, y se ubica dentro de la carpeta descrita en el paso anterior.
3. Se ejecuta la siguiente línea de comando: java -Xms128m -Xmx2000m -jar DocClustering.jar
4. En la consola se muestra cada uno de los archivos generados.

A.7 Interfaz de navegación

1. Luego de la ejecución del proceso de agrupamiento, se ejecuta el archivo index.html ubicado en la carpeta "agrupamiento" como se indicó en la sección anterior. Se aconseja el uso del navegador Firefox 3.6.

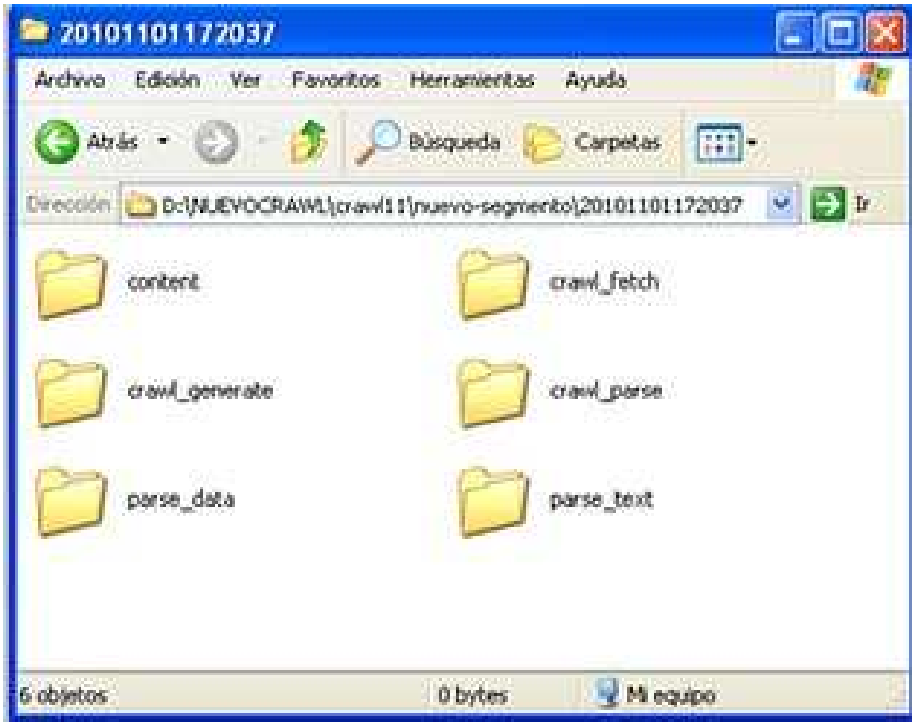


Figura A.20: Ubicación del nuevo segmento

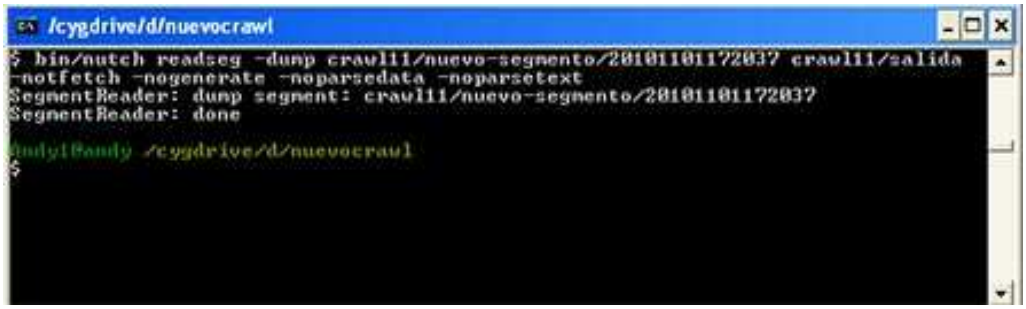


Figura A.21: Ejemplo generación de archivo

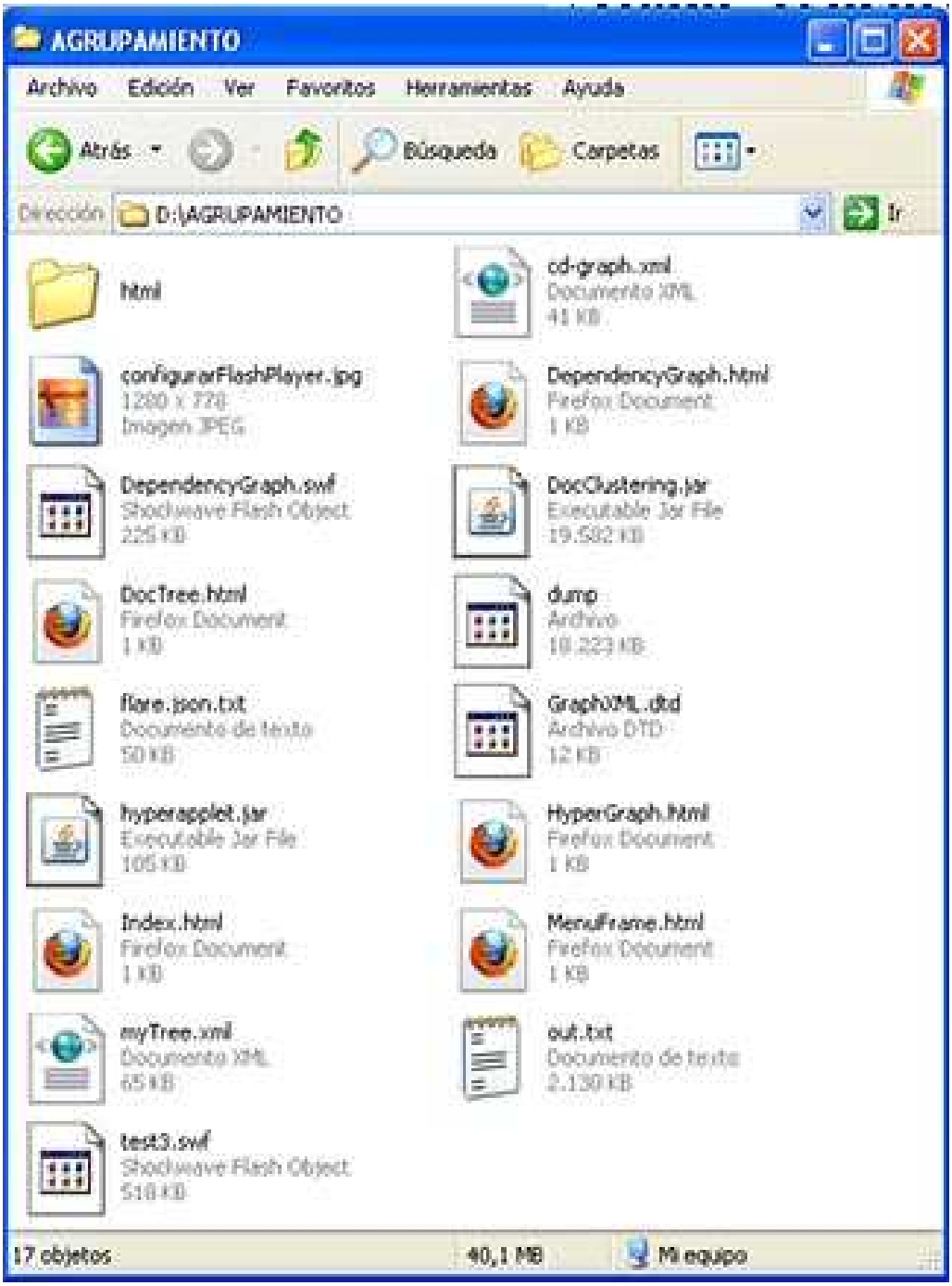


Figura A.22: Carpeta “Agrupamiento”

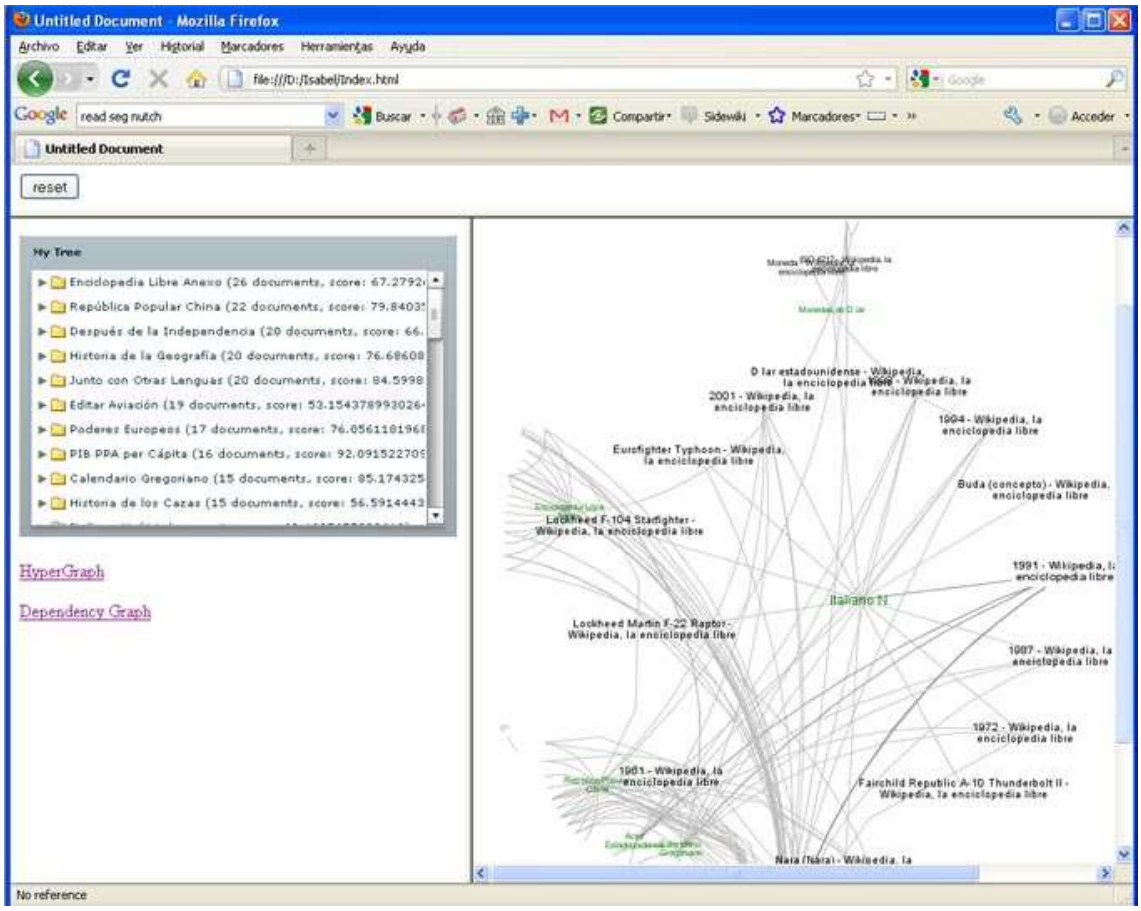


Figure A.23: Ejemplo interfaz de navegación

2. La interfaz de navegación se debe mostrar como el ejemplo de la figura A.23.

3. Para publicar el archivo index.html en la Web o en una intranet, debe utilizarse un servidor web.

B POLÍTICA DE SEGURIDAD PARA USO DE MOTORES DE BÚSQUEDA EN LOS PORTALES WEB DE LA UNIVERSIDAD NACIONAL DE COLOMBIA

Para la propuesta de la política de seguridad para el uso de motores de búsqueda en los portales web de la Universidad Nacional de Colombia se tuvieron en cuenta los siguientes conceptos descritos en la "Guía para elaboración de políticas de seguridad" de la Dirección Nacional de Informática y Comunicaciones[1] de la Universidad Nacional de Colombia[2]:

- Política: Declaración general de principios que presenta la posición de la administración para un área de control definida. Las políticas se elaboran con el fin de que tengan aplicación a largo plazo y guíen el desarrollo de reglas y criterios más específicos que aborden situaciones concretas. Las políticas son desplegadas y soportadas por estándares, mejores prácticas, procedimientos y guías. Las políticas deben ser pocas (es decir, un número pequeño), deben ser apoyadas y aprobadas por las directivas de la universidad, y deben ofrecer direccionamientos a toda la organización o a un conjunto importante de dependencias. Por definición, las políticas son obligatorias y la incapacidad o imposibilidad para cumplir una política exige que se apruebe una excepción.

- Estándar: Regla que especifica una acción o respuesta que se debe seguir a una situación dada. Los estándares son orientaciones obligatorias que buscan hacer cumplir las políticas. Los estándares sirven como especificaciones para la implementación de las políticas: son diseñados para promover la implementación de las políticas de alto nivel de la organización antes que crear nuevas políticas.

- Mejor práctica: Es una regla de seguridad específica a una plataforma que es aceptada a través de la industria al proporcionar el enfoque más efectivo a una implementación de seguridad concreta. Las mejores prácticas son establecidas para asegurar que las características de seguridad de sistemas utilizados con regularidad estén configurados y administrados de manera uniforme, garantizando un nivel consistente de seguridad a través de la organización.

- Guía: Una guía es una declaración general utilizada para recomendar o sugerir un enfoque para implementar políticas, estándares y buenas prácticas. Las guías son, esencialmente, recomendaciones que deben considerarse al implementar la seguridad. Aunque no son obligatorias, serán seguidas a menos que existan argumentos documentados y aprobados para no hacerlo.

[1] Dirección Nacional de Informática y Comunicaciones `http://www.dnic.unal.edu.co/index.php`
[2] "Guía para elaboración de políticas de seguridad `http://www.dnic.unal.edu.co/docs/guia_para_elaborar_politicas_v1_0.pdf`

B.1 Política de seguridad propuesta

Todo portal web dentro de la Universidad Nacional de Colombia, debe tener los mecanismos necesarios para que los buscadores que se utilizan dentro de los portales puedan acceder a todo el contenido público de dichos portales con el fin de facilitar el acceso a la información a través de dichas herramientas.

B.2 Estándar propuesto

El estándar propuesto para permitir el uso de buscadores dentro de los portales web es el ***Robots Exclusion Standard***[3], el cual es un protocolo que permite dar acceso a los robots o programas de rastreo a las páginas web públicas y su posterior indexación para permitir el procesamiento de consultas de texto dentro de los buscadores. Las ventajas de usar el ***Robots Exclusion Standard*** son las siguientes:

- Se impide el acceso a robots determinados: los rastreadores web (crawlers, robots) pertenecen a diferentes motores de búsqueda como Google, Yahoo, Bing, o buscadores de código abierto que pueden ser adaptados dentro de los portales web. Sin embargo existen robots cuyo objetivo es ocasionar problemas, en el portal `http://www.robotstxt.org/`, se encuentra actualizado un listado de los robots malos, los cuales son identificados al rededor de todo el mundo.
- Se reduce la sobrecarga del servidor: Se puede controlar el flujo de acceso de algunos robots. Algunos de ellos no tienen mucho control y por ello el número de peticiones pueden llegar a saturar el servidor.
- Se pueden prohibir zonas dentro del portal web: se puede dar el caso que existan zonas disponibles de un portal web, que sean accesibles para algunos robots, pero que no aparezca en algunos buscadores.
- Se pueden fijar mapas del sitio: es posible adaptar un sitemap para indicar una ruta adecuada para que los robots puedan recorrer un portal web.

B.3 Mejor práctica

La mejor práctica para la implementación del ***Robots Exclusion Standard***, es la utilización del archivo robots.txt en el directorio de nivel superior de su servidor web por ejemplo la dirección "http://www.example.com/" es el nivel superior de un portal web, por tanto el archivo robots.txt debe ir así ""http://www.example.com/robots.txt". Para el nombre del archivo siempre se usan letras minúsculas "robots.txt" no "ROBOTS.TXT".

B.4 Guía propuesta

Procedimiento de ejemplo para la creación de un archivo robots.txt es el siguiente[4]:

1. El encabezado debe llevar estas dos líneas:

   User-agent: *

   Disallow:

[3] ***Robots Exclusion Standard*** `http://www.robotstxt.org/robotstxt.html`

[4] Tomado de "Robots.txt : Todo lo que debería saber" `http://www.emezeta.com/articulos/robots-txt-todo-lo-que-deberia-saber`

2. El parámetro “User-agent” corresponde al nombre del robot.

3. El parámetro “Disallow” corresponde a las rutas donde se restringirá el acceso. Algunos ejemplos:

   Disallow: / prohibe la entrada a todo el sitio.

   Disallow: /foro/ prohibe la entrada a los documentos del directorio foro.

   Disallow: permite la entrada a todo el sitio.

4. En algunos casos suele utilizarse en lugar de Disallow, la palabra Allow. Aunque por definición es correcta, es conveniente no utilizarla, puesto que las rutas omitidas se asumen que están permitidas por defecto, y algunos rastreadores web no entienden la palabra Allow. Es posible utilizar varios Disallow bajo un mismo User-agent, pero no se pueden usar varios User-agent encima de un Disallow, como el siguiente ejemplo:

   # Crawler de MSN

   User-agent: msnbot

   Disallow: /links.html

   Disallow: /private/

   Disallow: /photos/

   Este código impide al rastreador web del buscador de Live (MSN) acceder a la página links.html, y las carpetas private y photos (y todo su contenido) de un sitio web. Añadiendo el carácter # al principio de una linea se pueden incluir comentarios que no interpretará el rastreador web.

5. Para diferentes rastreadores web se pueden tener conjuntos de reglas diferentes, formando un robots.txt más largo y completo. Cada vez que se escriban las reglas para un “User-agent” se debe dejar una línea en blanco de separación.

6. Se pueden usar comodines ($ y *) en las rutas en algunos rastreadores web como Googlebot y Slurp:

   User-agent: Slurp

   Disallow: /*.js$

   Disallow: /2006/*

   Disallow: /2007/*

   Disallow: /articulos/*/pagina/*

   Se está indicando al robot de Yahoo, que no indexe los archivos que terminan en .js (javascript), direcciones que empiecen por 2007 o 2006 (fechas), ni artículos con la palabra pagina (paginado de comentarios). Estos casos pertenecen a la idea de no indexar contenido duplicado.

7. Se debe tener mucho cuidado con usar cosas como Disallow: /pagina o Disallow: /*pagina, puesto que en lugar de bloquear algo como una carpeta, una pagina o artículos paginados, se termina bloqueando las direcciones como /primera-pagina o /paginas-de-interes/.

8. A veces algunos rastreadores web revisan con frecuencia los sitios web, y pueden generar gran cantidad de peticiones al servidor. Existe una manera de gestionar el comportamiento de estos robots con el parámetro “Crawl-delay” como se muestra en el siguiente ejemplo:

   User-agent: noxtrumbot

   Crawl-delay: 30

   Con esto le decimos al robot de noxtrum que espere 30 segundos entre cada acceso. Cuidado, porque Crawl-delay no lo soportan todos los crawlers (al menos MSNBot y Slurp si lo soportan,

y Googlebot desde el panel de webmasters también). Finalmente, podemos también incluir un mapa del sitio en nuestro robots.txt de la siguiente forma:

Sitemap: http://www.ejemplo.com/sitemap.xml

C ARTÍCULO "AN EXPLORATORY STUDY OF OPEN SOURCE SEARCH ENGINES: EVALUATION OF SUPPORTABILITY, USABILITY, FUNCTIONALITY AND PERFORMANCE".

El artículo titulado "An exploratory study of open source search engines: Evaluation of supportability, usability, functionality and performance", fue presentado en el Quinto Congreso Colombiano de Computación 2010, Cartagena- Colombia (abril de 2010) [63].

En este artículo se describe el procedimiento que se ejecutó para la selección de un motor de búsqueda open source, que se utilizó dentro del sistema de librería digital propuesto en este trabajo de tesis.

An exploratory study of open source search engines: Evaluation of supportability, usability, functionality and performance

Isabel Andrea Mahecha-Nieto
Universidad Nacional de Colombia
Departamento de Ingeniería de Sistemas e Industrial
Bogotá, Colombia
iamahechan@unal.edu.co

Elizabeth León Guzmán
Universidad Nacional de Colombia
Departamento de Ingeniería de Sistemas e Industrial
Bogotá, Colombia
eleonguz@unal.edu.co

ABSTRACT

Search Engines are applications for organizing and facilitating access to documents stored in different media (e.g. websites, hard disks, and repositories). There are many open source search engines published in the World Wide Web, and each of them has different advantages and features that are important for deciding which one to use in a specific website. The present work compares 48 open source search engines in terms of performance in retrieving, indexing and searching for documents on a specific website. The objective of this work was to have a frame of reference for selecting an open source search engine for the website of an institution. The selection was done based on features of usability, adaptability, functionality and performance according to the specific requirements of the website.

Categories and Subject Descriptors

H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance evaluation (efficiency and effectiveness)*; D.2.7 [**Software Engineering**]: Distribution, Maintenance, and Enhancement—*Documentation, Portability*

General Terms

Documentation, Experimentation, Performance

Keywords

Search Engines, Website, Open source

1. INTRODUCTION

The World Wide Web has allowed people to create and organize their own information spaces and share them with others. Its main purposes are to organize information into hypertext documents, to publish documents and to have useful access to these documents [24]. With the continuous growth of the Internet, websites have turned into huge repositories of unorganized information, therefore making it necessary to develop search engines that help users find information on the Web [16, 24].

Search Engines are applications for organizing and facilitating access to documents stored in different media (e.g. websites, hard disks, and repositories). They use crawlers that continually download huge portions of the Web and index them. Users can make queries to find specific information in such indexed documents, and the results are presented with a summary and linked to the original source documents [23]. As an alternative to implementing web search engines within web portals, there is the possibility of using commercial search engines (such as Google, Yahoo, MSN, etc.) or open source ones [16]. However, for most websites the use of a commercial search engine is too expensive and can be troublesome because they are best suited for large websites. On the contrary, open source search engines fulfill the same functions of commercial search engines [16], and have the advantages of the open source philosophy: they are distributed free of charge, their software remains active and they can be adapted to different needs [5].

There is a large number of open source search engines published in the Web, and each of them has different advantages and features that should be taken into account for deciding which one to implement in a specific website. Search engines can be classified according to their indexing features (inverted file, database, and other file structures), searching features (Boolean operators, fuzzy search, use of stemming, etc.), and features of support, update and technical documentation. The present study describes the process followed for selecting an open source search engine for the website of an institution. The selection process was divided into four parts; each part evaluated open source search engines in terms of one of the following features: supportability, usability, functionality and performance.

This paper proceeds as follows: In the second section, we describe a research study dealing with comparing open source search engines, in the third section we give a brief background on open source search engines, in the fourth section we describe the methodology followed for comparing different open source search engines, in the fifth section we present the results of the selection process, and finally, in

Quinto Congreso Colombiano de Computación 2010 Cartagena, Colombia
.

the last section we conclude with the main findings of the study.

2. RELATED WORK

Christian Middleton and Ricardo Baeza-Yates reported a study in which they compared 17 open source search engines that were publicly available on the Web around 2007 [16]. The study was divided into three parts. The first part consisted in an initial characterization for evaluating the availability of the different open source search engines according to the following characteristics: date of last update, version, observations about the continuity of the project and technical details. These are important aspects to consider because a search engine that has not been updated recently could present problems when being adapted to the requirements of a particular website. In the second part the authors show the main characteristics of the 17 open source search engines analyzed. These characteristics can be used to make an extensive classification of the search engines, which in turn can be used to reduce the options when selecting an open source engine. The main characteristics proposed by the authors were [16]:

- Storage: Indicates the way in which the indexer stores the index which can be by, either using a database engine or a simple file structure (e.g. an inverted index).
- Incremental Index: Indicates if the indexer is capable of adding files to an existing index without the need of generating the whole index again.
- Results Excerpt: If the engine gives an excerpt ("snippet") with the results.
- Results Template: Some engines give users the possibility of using a template for parsing the results of a query.
- Stop words: Indicates if the indexer can use a list of words as stop words in order to discard too frequent query terms.
- File type: Types of files that the indexer is capable of parsing. The common file type among the engines analyzed in [16] was HTML.
- Stemming: If the indexer/searcher is capable of doing stemming operations over the words.
- Fuzzy Search: Ability of solving queries in a fuzzy way, i.e. not necessarily matching the query exactly.
- Sort: Ability to sort the results by several criteria.
- Ranking: Indicates if the engine gives the results based on a ranking function.
- Search Type: The type of searches that the search engine is capable of doing, and whether or not it accepts query operators.
- Indexer Language: The programming language used to implement the indexer. This information is useful in order to extend the functionalities or integrate the search engine into an existent platform.
- License: Determines the conditions for using and modifying the indexer and/or the search engine.

In the third part of the study, the authors compare the performance of the search engines with different loads of data. The analysis was carried out based on metrics for indexing (indexing time, index size, and resource consumption) and performance during the retrieval stage (searching time, precision and recall). In present paper, we took Christian Middleton and Ricardo Baeza-Yates' study as a benchmark study and we adapted the methodology proposed by them to address the problem of selecting an open source search engine for the website of an institution. The features selected for comparing open source search engines are related in terms of usability, adaptability and performance.

3. OPEN SOURCE SEARCH ENGINES

3.1 Basic features of Search Engines

A search engine is a type of information retrieval system. The difference between a search engine and a standard information retrieval system is that a search engine replicates the Web, and it answers to the users' queries using indexes without having to access the web pages [2, 4]. The basic architecture of a search engine is composed of the following four elements [2]:

- Crawler: It is a program or software agent whose function is to navigate the Web by following the hyper-links to discover new pages and check for changes in crawled pages. Pages are sent to the main server where they are indexed.
- Indexer: This component builds an index with the content of pages and documents founded [25]. The index allows making a fast alphabetical lookup of each query word, followed by a direct retrieval of the relevant part of the string. On average, to find an item in a list of N items takes $Log_2 N$ steps; this is much faster than reading an entire long file [14].
- User interfaces: Search engines have two basic user interfaces, the query interface and the answer interface. The query interface is a page that has a text area where a user enters his queries and such queries are received as a string. The answer is the page where the search results are shown. It usually shows a list of the ten-top ranking pages [2]. Retrieved documents are shown as a list that contains the page's title, an excerpt of the content of the page and the link to the source document [25].
- Component for searching: The retrieving task follows these steps:
  1. Vocabulary Search: The query is divided into words that are searched in the vocabulary's index.
  2. Retrieval of occurrences: The lists from where the words of the query are recovered.
  3. Manipulation of occurrences: This task is carried out to answer phrases, proximity and Boolean operations.

  Figure 1 shows an overview of a search engine where the relationships between its different components are illustrated.

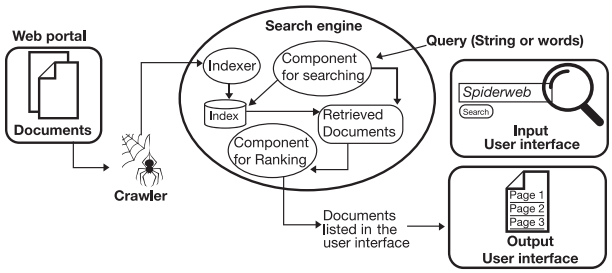


Figure 1: Components of a search engine

- Component for ranking: After searching in the index, the results of a query have to be ranked. The ranking of each found page or document can be calculated according to different parameters, such as the number of times that the query words are found in a document. Others parameters are based on factors such as the popularity of a page and their calculus considers the number of links from other pages to that page; such is the case of the Page Rank algorithm used by Google [3].

3.2 Open source search engines available on the Web

There are many open source search engines for downloading data from the Web, which are used either by personal and companies' projects to improve searching for documents in local sites and websites. We found 48 open source web search engines listed in some websites, which are focused on the field of information retrieval, publishing new developments and information on new tools constantly: Searchtools[1], Searchenginewatch[2] and java-source.net[3].

4. METHODOLOGY FOR SELECTING AN OPEN SOURCE SEARCH ENGINE FOR A WEBSITE

We took the work of Christian Middleton and Ricardo Baeza-Yates reported in [16] as a benchmark study and we focused the methodology on features of usability and extensibility for selecting an open source search engine for the website of an institution. The institution could be a small organization with some levels and areas in its organizational structure. This kind of organizations often have a website where they publish a large number of documents. The present study focused on selecting an open source search engine for the web portal of a university. The volume of documents that are published every day by a university is considerable given the diversity of activities that take place within it, and concern both the government of the institution and its academic development. The volume of documents that are published on the web portal include: regulations (resolutions, minutes, notices, etc.), the daily news and work on academic activities (lecture notes, theses, articles, reports, etc.). Users of these documents include mostly people who belong to such institution and have different roles such as: teachers, students, staff and people seeking for information through the university's web site.

To select the open source search engine that best fitted to the query needs of the web Portal users, information about open source search engines was gathered directly from the websites of the projects to which they belong as well as from the articles published by the groups responsible for such search engines. The selection process consisted of evaluating nonfunctional requirements and the performance of the search engines; the selection criteria were taken from FURPS [11], a model developed at Hewlett-Packard by Grady and Caswell for classifying software quality attributes. FURPS is an acronym for:

- Functionality: feature set, capabilities, generality, security.
- Usability: human factors, aesthetics, consistency, documentation.
- Reliability: frequency/severity of failure, recoverability, predictability, accuracy, mean time to failure.
- Performance: speed, efficiency, resource consumption, throughput, response time.
- Supportability: testability, extensibility, adaptability, maintainability, compatibility, configurability, serviceability, installability, localizability, portability.

In this work, open source engines were evaluated in terms of the following four quality features: supportability, usability, functionality and performance. The methodology followed for evaluating each of these features is described in subsections 4.1, 4.2 , 4.3 and 4.4, respectively.

4.1 Supportability

In [11] the feature of supportability combined three factors: 1) Extensibility, the ability to extend the program; 2) Adaptability: the extent to which a software system adapts to changes in its environment and 3) Serviceability: the easiness and effectiveness of the search engine's maintenance and repair system.

The evaluation of the supportability of the open source search engines included features of availability and maintenance; these aspects ensure that the software and its future modifications will be published on the websites of the projects that developed the open source search engines. For evaluating these criteria, we included the following data: state of the project; date of the last update, version, type of license, and number of websites where it is used. The search engines were first filtered according to the following criteria: the state of the project should be active, the date of the last update should not have occurred more than 3 years ago and the license should allow adding modifications and new functionalities.

4.2 Usability

The FURBS model proposes evaluating usability by considering human factors, overall aesthetics, consistency, and documentation [11]. In our case, the usability of a search engine must comply with an adequate documentation for its installation, configuration and management. This condition guarantees that there is enough information available for users to carry out their query activities, and for the systems

[1] Searchtools `http://www.searchtools.com/`

[2] Searchenginewatch `http://searchenginewatch.com/`

[3] java-source.net `http://java-source.net/open-source/search-engines`

administrators who are responsible for installing and managing potential errors. On the other hand, documentation regarding the source code is necessary for developing new functions for the open source software. For evaluating this aspect, we included the following data: author, language of development, user's manual, documentation for development, and languages of indexing. The criteria for filtering open source search engines according to features of usability were: the search engine must have published a user's manual and technical documentation, and due to the language of the collection of test documents , in our case the search engine must use indexing words in Spanish.

4.3 Functionality

The functionality evaluates the feature set and capabilities of the program and the generality of the functions that are delivered [11]. The search engine must adapt to the specific conditions of the site, so it has to be able to index the types of files in which the documents are published on the website. The evaluation of functionality was divided into two stages:

1. Evaluation of Indexing features: These features allow evaluating the options that each search engine has to store documents according to file types, languages and strategies for creating and maintaining an index that is then used to process the user's queries. For evaluating this features, we included the following data: storage strategy, language, option to maintain an incremental index, file types that are indexed, and management of: stemming, stop words, synonyms and acronyms. The criteria for the filter in this part of the evaluation were following: the search engine should be able to index text documents in formats such as plain text, MS Word, RTF and Tex. Indexing of PDF files was considered mandatory because too many of the files that are published on the web portal of the case study are created in this format.

2. Evaluation of Search features: These features allow verifying the methods applied for processing queries. These methods may be related to the kind of search that can be run and the relevance ranking it uses. Also, this evaluation includes the use of templates that allow users to select how they want query results to be displayed. For evaluating these features, we included the following data: search type permitted (Boolean, phrase, wildcards, fuzzy), assessment of relevance, kind of ranking, results excerpt and results template. The criteria for the filter in this evaluation were: the search engine has to have at least Boolean search and search with phrases; and the search engine has to have a ranking function to handle a list of relevant results for a query.

4.4 Performance

According to the preliminary evaluation of supportability, usability and functionality, we choose the three best open source search engines that could be adapted to the specific conditions of a small website.

The performance tests included measures of effectiveness and efficiency:

1. Effectiveness metric:

   - Precision: This measures shows how well the search engine is doing at rejecting non-relevant documents. To measure the accuracy, we first considered a cut-off point in the ranking of query results, in this case we took the first 20 results which that are normally displayed on the first page of the search results. The relevant results have a link to the documents that contain the word or phrase of the query [24]. This study only considered precision because other metrics such as recall need of a complete exploration of the entire document collection and then for a specialist to decide which ones are relevant [24]. So for our case that was a restriction. The precision was calculated as follows [24]:

$$Precision = \frac{Number\ of\ relevant\ documents\ retrieved}{Total\ number\ of\ retrieved\ documents} \quad (1)$$

2. Efficiency metrics:

   - Elapsed indexing time: Measures the amount of time necessary to build a document index on a particular system.
   - Index size: Amount of storage necessary to store the index files [24].

5. SELECTING AN OPEN SOURCE SEARCH ENGINE

This section describes the process followed to select one open source search engine. The selection process was divided into four parts: supportability evaluation, usability evaluation, functionality evaluation and performance evaluation.

This exploratory study analyzed a set of 48 published open source search engines published and referrals in the literature on the Web and research papers. The results are presented for assessment in accordance with the criteria set out in section 4: supportability, usability, functionality and performance.

5.1 Supportability evaluation

Table 1 shows information about 22 open source search engines selected based on the criteria of supportability. These search engines satisfied the criteria described in the subsection 1. These search engines are active, their last update does not fall more than 3 years ago; and the licenses allow the inclusion of new functionalities. The licenses of these search engines are: GNU GPL, Apache 2.0, BSD, Mozilla Public License, and LGPL. The open source search engines most commonly used are: Nutch, Solr and SWISH-E.

5.1.1 Usability evaluation

The list of 22 open source search engines was reduced to 10 when search engines were filtered according to the following criteria: the search engine must have published a user manual and technical documentation, and due to the language of the test documents collection, in this case the search engine must include indexing words in Spanish; Table 2 shows information about the support and documentation of these 10 search engines.

The technical support of an application is very important; the team that is developing the project of the search engine

Table 1: Set of open source search engines which meeting the criteria for supportability

Search Engine	Update	Version	License	# of sites
Amberfish®[9]	01/06/2007	1.6.4	GNU GPL	Unavailable
Compass	06/04/2009	2.2.0	Apache 2.0	6
Datapark[8]	20/04/2009	4.52	GNU GPL	22
Egothor	20/05/2006	1.3.004	BSD	Unavailable
Hounder	29/07/2008	1.1.0	Apache 2.0	6
Indri[20]	06/04/2009	2.9	BSD	Unavailable
KSearch[13]	27/05/2008	1.5b	GNU GPL	20
LIUS	06/03/2007	1.0	Apache 2.0	Unavailable
MG4J[22]	05/05/2008	2.1.2	GNU GPL	Unavailable
mnoGoSearch for Unix[6]	13/02/2009	3.3.8	GNU GPL	Unavailable
Namazu[19]	12/03/2009	2.0.19	GNU GPL	10
Nutch[10]	23/03/2009	1.0	Apache 2.0	44
Omega[21]	19/04/2009	1.0.12	GNU GPL	Unavailable
Perlfect Search	30/05/2007	3.37	GNU GPL	Unavailable
regain	24/04/2009	1.6	LGPL	1
Solr	15/09/2008	1.3.0	Apache 2.0	44
SWISH++[15]	25/01/2008	6.1.5	GNU GPL	Unavailable
SWISH-E[17]	05/04/2007	2.4.7	GNU GPL	162
Terrier[1]	29/01/2009	2.2.1	Mozilla Public License	Unavailable
YaCy	21/01/2009	0.7	GNU GPL	4
Zebra[7]	03/04/2008	2.0.28	GNU GPL	12
Zettair[12]	08/09/2006	0.9.3	BSD	Unavailable

needs to have enough ways to communicate corrections of errors and enough assistance to solve problems of the users and developers. In the field of research, Indri is a search engine that is part of the project Lemur [20], which is developed jointly by the Carnegie Mellon University and the University of Massachusetts, and has an open and very active forum of discussion. Other prominent cases are Nutch and Solr, which are supported by the Apache Software Foundation[4]. These projects have a fairly large list of developers around the world and are constantly running error corrections. They also contain a wide variety of functions for indexing text documents and their wikis are constantly being updated.

5.2 Functionality evaluation

5.2.1 Index features

Table 3 shows information about the index features of the 4 search engines selected based on functionality in features of indexing criteria. At this stage, search engines were filtered according to the following criteria: the search engine should index text documents in formats such as plain text, MS Word, RTF and Tex. Indexing of PDF files was mandatory. The list of the 5 open source search engines was reduced to 4 after applying this filter.

5.2.2 Search features

Table 4 shows information about the search features of 4 search engines selected. The criteria for filtering search engines according to search features were: the search engine has to have at least Boolean search or search with phrases; and the search engine has to have a ranking function to handle a list of relevant results for a query.

[4]Apache Software Foundation http://www.apache.org/

5.3 Performance evaluation

In this subsection, we present the results of the evaluation of performance for two selected open source search engines. We selected: Nutch and Terrier, because they have good features of supportability, usability, indexing and search. Also, Nutch and Terrier use the Snowball[5] library for indexing different languages including English and Spanish. This section proceeds as follows: in the first subsection we show the tests parameters and in the second subsection we present the description of the results in terms of effectiveness and efficiency.

5.3.1 Nutch

Nutch is an open source web search engine and is a project covered by The Apache Software Foundation[6]. It is built with the most commonly used indexing library, Lucene[7], and contains web-specifics such as a crawler, a link-graph database, parsers for HTML and other document formats, etc [10]. Nutch has a highly modular architecture that allows developers to create plug-ins for activities such as media-type parsing, data retrieval, querying and clustering of results [18].

Figure 2 shows the user interface of Nutch. Its structure has basic functions similar to the ones of commercial search engines such as Google[8]. This kind of interface is easily understood by users because it has intuitive fields for writing a query and the list of results shows a small excerpt from each document.

5.3.2 Terrier

[5]Snowball http://snowball.tartarus.org/

[6]Nutch http://lucene.apache.org/nutch/about.html

[7]Lucene http://lucene.apache.org/java/docs/

[8]Google http://www.google.com

Table 2: Set of open source search engines that meet the criteria of usability

Search Engine	Language of development	User manual	Technical support	Languages of indexing
Compass	Java	Yes	Yes	***
Indri	C, Java	Yes	Yes	Language independent tokenization of UTF-8 encoded documents(any language)
Hounder	Java	Yes	Yes	***
MG4J	Java	Yes	Yes	***
mnoGoSearch	C	Yes	Yes	700 languages, which includes most of the frequently used language in the world.
Namazu	C++	Yes	Yes	
Nutch	Java	Yes	Yes	***
Solr	Java	Yes	Yes	***
Terrier	Java	Yes	Yes	Encoding of documents (UTF), to facilitate multi-lingual retrieval.
Zettair	C	Yes	Yes	

***Search engines that used Snowball, these indexing documents in English, French, Spanish, Portuguese, Italian, Romanian, German, Dutch, Swedish, Norwegian, Danish, Russian, Finnish, Hungarian and Turkish

Table 3: Set of open source search engines that meet the criteria of functionality in features of indexing

Search Engine	Type file Storage	File Types			Stemming	Stop-words	Synonyms	Acronyms
		Text	Web pages	Other				
Indri	File	MS Word, MS Power Point, Pdf	Html, Xml		Yes	Yes	Yes	Yes
Namazu	File	Text files, email messages, MS Word, TEX, Pdf	usenet netnews, mhonarc	gzip, RFC	No	Yes	No	No
Nutch	Data Base	Plain text, MS Word, RTF	Html, JavaScript, RSS, Pdf	Microsoft Power Point, ZIP, Mp3	No	Yes	Yes	No
Terrier	File	Plain text, MS Word, Pdf	Html	MS Excel, MS PowerPoint	Yes	Yes	No	No

Table 4: Set of open source search engines that meet the criteria of functionality in features of searching

Search Engine	Search Type						Relevance	Ranking	Results Excerpt	Results Template
	Boolean	Phrase	Wilcards	Fields	Proximity	Fuzzy				
Indri	Yes	Yes	Yes	No	Yes	No	Yes	Pseudo Relevance Feedback	No	No
Nutch	No	Yes	Yes	Yes	No	No	Yes	Calculation using Vector Space Model (VSM) of Information Retrieval	Yes	Yes
Terrier	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Randomness (DFR) document ranking models, Okapi BM25, language modeling and TF-IDF	No	No

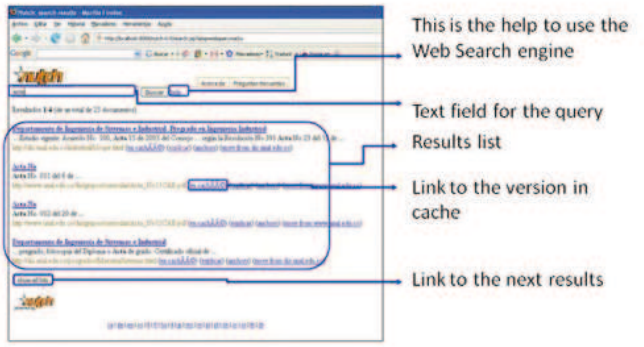


Figure 2: Nutch's Interface

Terrier is a project of the Department of Computing Science from the University of Glasgow. The goal of this search engine is to provide an understanding, adaptable and robust platform. Terrier is written in Java and has been used for the Web and Enterprise search, Desktop, Intranet and Vertical search engines. Also, this tool is used for developing and evaluating novel large-scale text information retrieval techniques and applications[9].

5.3.3 *Parameters settings*

1. **Website of tests:** The search engines Nutch and Terrier were tested in the website of the National University of Colombia[10]. This website was selected because it is an example of an institution's website, with a typical organization of documents and a distribution of topics specific to the institution. The users of this website are mostly students, teachers, administrative staff and people in general. The website contains documents related to the rules of the institution, academic programs, information and documents for courses, information from research groups, information services and contact details of the department.

2. **Depth:** This parameter refers to the limit of how far the crawler goes. In this case, we use a depth 10 for a full crawl.

3. **Number of pages per level:** This is the limit for the number of pages fetched at each level of the web graph. In this study, we used three amounts of documents: 50, 100 and 1,000.

With the parameters described below, we tested the search engines and selected three indexes from each one in order to test their effectiveness and efficiency. The selection of the three indexes showed three possible scenarios in a web site: a small local site with about 500 documents, a medium site with about 1,000 documents and a big website with about 10,000 documents.

5.3.4 *Effectiveness*

Precision was the metric used for measuring the effectiveness of the queries. We calculated the precision by using the equation described in subsection 4.4. The sample queries were selected according to the type and format of the documents that are published on the web portal. These questions were intended to test the characteristics of the search engine, with different kinds of search. The search queries were:

- Search with words: Actas, Resolución, Pregrado and Maestría.
- Search with wildcards:*.pdf, *.doc
- Search with phrases: Grupos de investigación, Maestría en Ingeniería, "Maestría en Ingeniería", "Departamento de Ingeniería de Sistemas e Industrial" and "Departamento de Ingeniería de Sistemas e Industrial".

Figure 3 shows the precision of sample queries in three different indexes of each search engine. It was noted that the precision in the first index of 500 pages for some queries is not good, at the rate of 1,000 pages precision is equal to 1 for every sample query. The conclusion is that for low levels of retrieved pages the accuracy decreases significantly for both search engines.

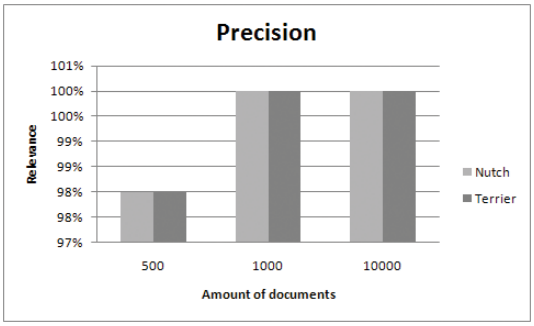


Figure 3: Precision of Nutch and Terrier for different queries with different amounts of documents

5.3.5 *Efficiency*

The elapsed indexing time and index size were metrics to measure efficiency. Figure 4 shows the results for different amounts of documents retrieved. It was noted that the relationship is almost linear. Nutch showed a better performance than Terrier for the three indexes.

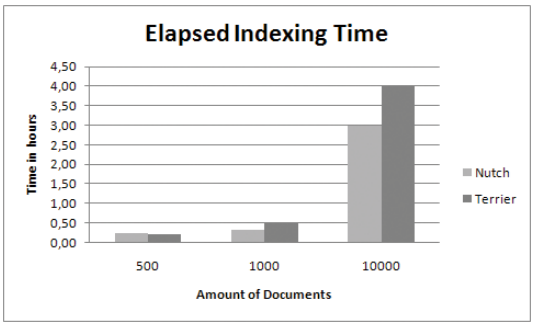


Figure 4: Elapsed Time indexing of Nutch and Terrier

Figure 5 shows the relationship between the amount of documents retrieved and the size of the indexes. It was noted that the relationship is almost linear. Nuth has smaller indexes than Terrier.

[9] Terrier `http://ir.dcs.gla.ac.uk/terrier/`

[10] Universidad Nacional de Colombia `http://http://www.unal.edu.co/`

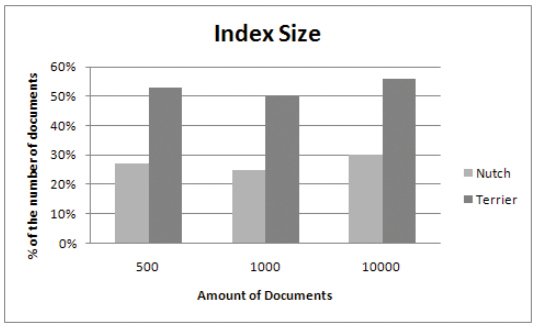


Figure 5: Query Index Size of Nutch and Terrier

6. CONCLUSIONS

In the field of research, the offer of open source search engines shows a favorable outlook with the aim of improving and generating new tools for organizing and classifying documents. In this article, we described a basic comparison of different open source web search engines using criteria such as: availability and maintenance, support and documentation, index features and search features. This comparison enabled us to select of a web search engine that was adapted to a specific website.

The web search engine chosen was Nutch, which is an option that offers good conditions in terms of support, maintenance and ability to be adapted easily to any website. Nutch, same as other open source search engines, facilitates access to information to any person or organization, requiring only of an investment in training for simple installation and configuration of the software tools. Nutch has an advantage because it is an integrated system that contains all the components necessary to search the Web: a crawler, an index, a link-graph database, parsers for HTML and other document formats, etc.[10]. It is easy to install in a web portal and is used by the users in an intuitive way. On the other hand, Terrier is built on a robust platform that allows developers to build search engines but is not a search engine itself. A software developer can use Terrier to build search engines.

In the current approach web search engines are aimed to generate more robust systems for the organization and classification of documents such as digital libraries.

7. FUTURE WORKS

The research group Midas, adscribed to the Faculty of Engineering of the National University of Colombia, Bogotá; it has within its projects the selection of a search engine to suit the specific conditions of the website of an institution, which should allow indexing public documents related to its regulations, operation and those generated by its members. The second part of the project consists in proposing a model of document classification through the addition of automatic meta-data and using clustering techniques that will allow finding patterns of relationships between documents and set trends for publication. Finally, the model will facilitate users to access information contained in documents.

8. REFERENCES

[1] Terrier - http://ir.dcs.gla.ac.uk/terrier/, 2007.

[2] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*. Addison-Wesley Harlow, England, 1999.

[3] S. Brin, L. Page, R. Motwami, and T. Winograd. The PageRank citation ranking: bringing order to the web. Technical report, 1999.

[4] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, 2003.

[5] K. Coar. The open source definition. http://opensource.org/docs/osd, July 2006.

[6] L. Corp. mnogosearch web search engine software - http://www.mnogosearch.org/, 2008.

[7] I. Data. Zebra - http://www.indexdata.dk/zebra/, 2008.

[8] D. Engine. Dataparksearch engine - http://www.dataparksearch.org/, 2008.

[9] I. Etymon Systems. Text retrieval, Amberfish and Isearch - http://www.etymon.com/tr.html, 2005.

[10] T. A. S. Foundation. Nutch - http://wiki.apache.org/nutch/, 2007.

[11] R. B. Grady and D. L. Caswell. *Software metrics: establishing a company-wide program*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1987.

[12] S. E. Group. The Zettair Search Engine - http://www.seg.rmit.edu.au/zettair/, 2006.

[13] KScripts.com. Kscripts - http://www.kscripts.com/, 2008.

[14] M. Lesk. *Understanding Digital Libraries*. Morgan Kaufmann, 2004.

[15] P. J. Lucas. SWISH++: Simple Web Indexing System for Humans: C++ version - http://swishplusplus.sourceforge.net/, 2008.

[16] C. Middleton and R. . Baeza-Yates. A comparison of open source search engines. Technical report, Universitat Pompeu Fabra Departamento de Tecnología, October 2007.

[17] W. H. of Famer Kevin Hughes. Swish-e: Simple Web Indexing System for Humans - Enhanced - http://swish-e.org/index.html, 2007.

[18] T. Perera. Nutch: The java search engine. http://wiki.apache.org/nutch/Nutch_-_The_Java_Search_Engine, 2007.

[19] N. Project. Namazu: a full-text search engine - http://www.namazu.org/, 2008.

[20] T. L. Project. Indri hints - http://ciir.cs.umass.edu/ strohman/indri/, 2007.

[21] X. project. Omega overview - http://xapian.org/docs/omega/overview.html, 2008.

[22] S. Vigna. Mg4j (managing gigabytes for java). http://mg4j.dsi.unimi.it/, 2008.

[23] I. Witten and D. Bainbridge. *How to Build a Digital Library*. Morgan Kaufmann, 2002.

[24] I. Witten, M. Gori, and T. Numerico. *Web Dragons: Inside the Myths of Search Engine Technology*. Morgan Kaufmann, 2007.

[25] I. H. Witten. How to build a digital library using open-source software. pages 416–416, Portland, Oregon, USA, 2002. ACM.