UNIVERSIDAD SAN FRANCISCO DE QUITO

Personalización de Búsquedas en la Web

Sergio Páez Bimos

Tesis de Grado presentada como requisito para la obtención del título de Ingeniero en Sistemas

Quito, Marzo 2011

UNIVERSIDAD SAN FRANCISCO DE QUITO
Colegio Politécnico

HOJA DE APROBACIÓN DE TESIS

Personalización de Búsquedas en la Web

Sergio Páez Bimos

Enrique Vinicio Carrera, D.Sc. .....
Director de Tesis (firma)

Fausto Pasmay, MS .....
Coordinador de Ingeniería de
Sistemas (firma)

Quito, Marzo 2011

Este trabajo esta dedicado para mis padres y hermanos, quienes en el calor del hogar siempre me han dado su apoyo y cariño. Éste es simplemente un símbolo de agradecimiento.

Además una dedicación muy especial para Malena y Emma, mi nueva familia. Ellas han sido la gran motivación y me han dado el empuje final para culminar este proyecto.

Agradecimientos

A mis padres por toda la entrega y apoyo que me han dado, por todos aquellos esfuerzos, consejos, paciencia, cariño, gratos momentos, las oportunidades de aprender, sus guías de como tratar de vivir bien y luchar contra la adversidad . Sin su apoyo la consecución de este trabajo no hubiera llegado a este punto.

A mis hermanos y a mi esposa por todo el apoyo, empuje y cariño que me han brindado en las distintas etapas de mi vida y de este proyecto.

A mis profesores que con su experiencia han sido una magnífica guía para mi vida estudiantil y profesional, pues ya que sea por estar de acuerdo o no en las clases y discusiones en las aulas, las experiencias de haber estudiado con ellos han sido enriquecedoras desde cualquier punto de vista, todas las clases en las que estuve presente me dejaron un gran valor.

Agradezco a Vinicio Carrera y Fausto Pasmay por representar un gran aporte en el área de Sistemas de la USFQ, y por lo que han representado durante la ejecución de mis estudios, por toda la exigencia que han tenido con sus alumnos, por transmitir sus experiencias de forma transparente y sincera.

Un agradecimiento especial a Vinicio por darme todo el apoyo como tutor para este trabajo, siempre ha estado pendiente del progreso de este proyecto aún cuando en circunstancias determinadas mi entrega a éste no ha sido continua, su soporte académico es invaluable en esta tesis.

Tengo presente en este agradecimiento a mis compañeros y amigos, con los cuales hemos compartido y afrontado las exigencias de los estudios, malas noches, estrés, presiones, fiestas, una cancha de fútbol, risas y demás. Estos años de universidad han sido un tiempo inolvidable y en gran parte es gracias a ellos.

Resumen

Siendo la Web la base de datos más grande del mundo, la información que encontramos en ella es diversa y dinámica, se encuentra en cantidades significativamente grandes, por lo que se puede decir que porciones considerables de esta información existen de forma no estructurada y volátil. Esto deriva en una serie de dificultades para los usuarios que navegan a través de la Web, tratando de buscar información relevante para sus intereses. Para las personas encontrar lo que buscan no siempre es una tarea fácil, la información presentada después de que una búsqueda es realizada no siempre está mostrada de la forma más útil, en parámetros de relevancia y de las áreas de interés de los usuarios.

La manera de enfrentar este problema es mediante la personalización de búsquedas, es decir mostrarle al usuario resultados que espera a través del conocimiento de sus áreas de interés en la Web. Esta tesis realiza un análisis sobre la efectividad de diferentes formas de personalizar una búsqueda, basándose en dos perfiles de usuario: *perfil fijo o explícito* y *perfil adaptativo*, aplicados sobre dos mecanismos de selección de páginas relevantes: *filtrado de resultados* y *refinamiento de consulta*.

Abstract

The Web is the largest database in the world, it's information is diverse and dynamic, and it is found in significantly large quantities, but also we can say that considerable portions of this information exist in an unstructured and volatile manner. This leads to several difficulties during navigation through the Web trying to find information relevant to the user's interests. Therefore finding information is not always an easy task. The data presented after a search it's not always shown in the most useful and relevant way concerning the areas of user's interest.

The best way to face this problem is by personalizing web searches, it means to show the results that the user expects, through knowledge of the user's interesting areas in the Web. This thesis makes an analysis of the effectiveness of different ways to personalize a search, based on two user profiles, fixed (or explicit) profile and adaptive (or implicit) profile, applied on two selection mechanisms of relevant pages; page filtering and query refinement.

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

1.1. Antecedentes

La inmensa cantidad de información que encontramos en la Web es diversa y dinámica. De hecho, se puede afirmar que la Web es la base de datos más grande del mundo, que aunque es altamente escalable, contiene información no-estructurada y volátil en una gran diversidad de formatos. Estas características crean una serie de problemas para los usuarios que buscan información en la Web. Entre los principales inconvenientes podemos mencionar:

- La información encontrada no siempre es relevante. Al realizar una búsqueda por determinado tema, muchos de los enlaces que se despliegan como resultado no tienen contenidos de importancia para el usuario final.
- La información cambia tanto en calidad como en cantidad. Resultados de búsquedas similares presentan enlaces distintos en función del tiempo, motor de búsqueda o estado de la red.
- La información existente no es personalizada. Los usuarios generalmente tienen preferencias de como la información tiene que ser visualizada y organizada, además que no siempre la información se encuentra en la manera más óptima para su navegación.

- El área de interés de cada usuario cambia. Así como la información presente en la Web cambia en función del tiempo, así también cambian las preferencias o intereses de los usuarios.

Los motores de búsqueda convencionales se basan en el envío de una respuesta al usuario como producto de una consulta realizada, dicha consulta consiste en una secuencia de palabras clave específicas relacionadas al tema de interés del usuario. La respuesta que se obtiene en este tipo de motores de búsqueda tiene un dominio general con relación a la petición hecha, lo que en definitiva significa que el motor atraviesa todas las páginas indexadas buscando la relación con la consulta y devuelve todas las que coinciden en un orden de relevancia propio del motor de búsqueda. Así, al ingresar una misma consulta dos usuarios con distintas áreas de interés, obtienen como resultado las mismas páginas, con información de áreas que probablemente no les interesa a ninguno de los dos.

1.2. Entorno Actual

Con la finalidad de solucionar esta problemática, existen algunos proyectos y productos comerciales los cuales usan distintos tipos de mecanismos y metodologías para lograr su propósito.

En la actualidad existen motores de búsqueda que ofrecen al usuario una serie de categorías para elegir y las cuales se usan para filtrar la información en cada búsqueda realizada, o también otro segmento que se los conoce como los *Web Groups*, donde el usuario puede navegar a través de categorías y subcategorías hasta encontrar una página de su interés. De hecho los motores más importantes y populares cuentan con este tipo de mecanismos (8), (21) y (4). Adicionalmente, algunos motores de búsqueda ofrecen interfaces que permite la selección de varios parámetros (preferencias) de búsqueda (7), como fechas, formatos de documentos, idiomas, dominios, etc.

De igual manera, existen motores de búsqueda que personalizan los resultados basándose únicamente en el historial de navegación de los usuarios. Un ejemplo de ello es *Web History* de *Google* (9), donde los usuarios deben tener una cuenta registrada para que se realice el monitoreo de su actividad Web, y en base a ésto la obtención de resultados personalizados. Esta funcionalidad de *Google* está disponible a través del *Google Toolbar* que es el ente encargado de administrar la información asociada con cada cuenta de usuario.

Existen trabajos relacionados con este proyecto de tesis donde se presentan, discuten y analizan los diferentes mecanismos de personalización de búsquedas aplicados a diferentes ámbitos y prácticas. Mecanismos y metodologías que rondan y encajan dentro de los tópicos del data mining, web mining, text mining, personalización, obtención de información, interacciones con el usuario, entre otros.

Así pues en algunos de estos trabajos han sido considerados como material de apoyo y fuente de consulta de esta tesis, estos tratan de; el filtrado de información (14), investigaciones de web mining (12), web mining para el descubrimiento de información y patrones en la web (16), búsquedas web adaptativas basadas en el uso de perfiles de usuario construidas sin esfuerzo del usuario (11), algunos planteamientos que proponen nuevos mecanismos de personalización como es el caso de CubeSVD un nuevo alcance para la personalización de búsquedas (1) y búsquedas sobre dominios específicos basadas en la determinación de palabras clave (17).

1.3. Propuesta

Construir el prototipo de un sistema de búsqueda en la Web que permita la personalización de la búsqueda de manera manual y automática usando tanto refinamiento de consultas como filtrado de resultados. El prototipo servirá de base para el análisis de efectividad de los diversos mecanismos de personalización de búsquedas.

Este trabajo además de el estudio de las técnicas de personalización de búsquedas

mencionadas, propone crear una infraestructura base sobre la que se podrían evaluar y perfeccionar nuevas técnicas de personalización y web mining.

1.3.1. Objetivos de la Tesis

- Investigar los mecanismos de personalización de búsquedas basados en formularios de preferencias.
- Investigar los mecanismos de personalización de búsquedas basados en el comportamiento histórico del usuario.
- Investigar las técnicas de personalización de búsquedas que usan refinamiento de consultas.
- Investigar las técnicas de personalización de búsquedas que usan filtrado de resultados.
- Analizar los requerimientos informáticos que exigen cada una de los mecanismos y técnicas mencionadas.
- Definir las herramientas informáticas que mejor se ajustan al desarrollo de este proyecto.
- Implementar el prototipo que permita trabajar con los diferentes mecanismos y técnicas.
- Evaluar el prototipo con un número limitado de usuarios.
- Concluir elementos de juicio que lleven a proponer una combinación de mecanismos y técnicas de alta efectividad en la búsqueda de información.
- Documentar el estudio realizado incluyendo los resultados más relevantes.

Capítulo 2

Fundamento Teórico

2.1. Introducción

Como ya fue mencionado previamente esta tesis analiza la efectividad de los mecanismos de personalización de búsquedas (filtrado de información vs. refinamiento de consulta), sobre el funcionamiento de dos tipos de perfiles, el fijo y el adaptativo. Siendo así es importante discutir:

De tal manera en este capítulo veremos:

1. Establecimiento de la preferencias de usuario a través de los perfiles.

2. Personalización de búsquedas mediante el filtrado de resultados y el refinamiento de consultas.

3. Clasificación de páginas como soporte de el filtrado de resultados y la determinación de palabras clave por categorías.

4. Comunicación del Prototipo con los motores de búsqueda convencionales.

Detalles específicos de como fueron aplicados los diferentes conceptos en el prototipo real se los encontrará en el capítulo de implementación.

2.2. Construcción de Perfiles

Para que la personalización de búsquedas pueda ser llevada a cabo es necesario de alguna forma saber que es lo que el usuario quiere ver, es decir, saber cuales son sus intereses dentro del mundo web. Para lograrlo se recurre a la creación de perfiles, siendo estos un conjunto de preferencias los cuales son almacenados y representan el interés del usuario en diferentes áreas, para este caso categorías.

2.2.1. Tipos de Perfiles

El perfil de usuario se puede construir básicamente de dos maneras; fija y adaptativa. En el primer caso la participación explícita de usuario es requerida, mientras que en el segundo la interacción es implícita. En las siguientes subsecciones hay una explicación más detallada de cada perfil.

2.2.1.1. Perfil Fijo

El perfil fijo está construido en base a la interacción explícita del usuario, de tal manera en este tipo de perfil los usuarios registran sus intereses, información demográfica o algún tipo de retro-alimentación al sistema. En cualquiera de los casos el ingreso de información es realizado a través de canales en donde el usuario puede hacer selección de sus preferencias, típicamente a través de un formulario.

La creación del perfil, actualización y administración dependerán siempre de lo que el usuario explícitamente manifieste, a través de los formularios donde la información esta definida. Refiérase a (13), (1) y (11) donde se ha trabajado con diferentes perfiles de usuario.

- Ventajas: Simplicidad.
- Desventajas: El usuario necesariamente debe saber que es lo que quiere.

2.2.1.2. Perfil Adaptativo

Por otro lado tenemos el perfil adaptativo llamado así puesto que aprende automáticamente y se adapta a las preferencias del usuario, en el cual el usuario no necesita llenar formularios o evaluar los resultados de las búsquedas. En este caso la información referente al usuario llega de diferentes fuentes implícitas, de manera que puede llegar a representar diferentes aspectos del usuario.

Existen distintas maneras de construir un perfil de este tipo, este aprendizaje automático de las preferencias siempre dependerá altamente de parámetros y métricas que implícitamente adaptan el comportamiento del sistema, así podemos encontrarnos con soluciones que estén basadas en el historial de navegación, el historial de búsquedas, links en donde se ha dado un click, tiempo de permanencia en las páginas, similaridad entre distintos usuarios, entre otros. (13), (11) y(1).

- Ventajas: Descubre lo que el usuario quiere.
- Desventajas: Necesita de un periodo de adaptación.

2.3. Personalización de Búsquedas

Una vez creado el perfil de usuario (explícito o adaptativo) es necesario filtrar la información de acuerdo a dicho perfil. Para ello también existen diversas técnicas y/o soluciones. Entre las opciones que más destacan podemos mencionar el refinamiento de consultas y el filtrado de páginas (o resultados) basado en categorización. (14), (12) y (17).

2.3.1. Filtrado de Resultados Basado en Categorías de Interés

El filtrado de páginas basado en categorización, consiste en identificar la categoría de cada resultado presentado por un motor de búsqueda convencional y eliminar aquellos

que no concuerdan con el perfil de usuario correspondiente. La categorización de las páginas es realizada mediante el uso del clasicador, generalmente se aplican un conjunto de reglas para establecer la relevancia del resultado.(14).

2.3.2. Refinamiento de Consulta

El refinamiento de consultas es la técnica más simple, consiste en adicionar términos o palabras clave a la consulta original realizada por el usuario. Los términos adicionados deben identificar las categorías de preferencia del usuario y de esta forma enfocar la búsqueda de un motor convencional en las páginas de interés del usuario.(17).

Existe un trabajo inicial que es la determinación de la palabra asociada a cada categoría, la cual puede ser provista por un clasificador como se describe en la próxima sección.

2.4. Clasificador

Es importante entender el funcionamiento de un clasificador, puesto que es un medio fundamental para la aplicación de los mecanismos de personalización de búsquedas, ya que sea cuales fueren estos, es necesario enmarcar a la información dentro de categorías (grupos, etiquetas, etc), las cuales nos permitirán luego determinar si son categorías o grupos que el usuario quiere ver.

Cabe destacar que el clasificador en este proyecto no es parte del alcance, al construir el prototipo se ha creado un clasificador básico, el cual permite que los mecanismos de personalización sean aplicados. El principio de este prototipo es servir como base para el soporte de nuevas investigaciones que puedan llevarse a cabo en el ámbito de web mining.

2.4.1. ¿Que es un Clasificador?

Un clasificador es una forma de análisis que puede ser usada para extraer modelos descriptivos de importantes clases de datos. El clasificador lo que hace es predecir etiquetas categóricas, éste análisis puede proveer un mejor entendimiento de cantidades de datos grandes. Un ejemplo de esto sería la construcción de un modelo de clasificación para categorizar aplicaciones de préstamo bancarias, y saber si estas son seguras o riesgosas (10).

La clasificación de datos es un proceso de dos pasos, la fase de aprendizaje toma lugar primero, se construye el clasificador basándose en el aprendizaje del conjunto de datos de entrenamiento o *training set*, el cual es una base de datos que contiene las tuplas y sus etiquetas de clases asociadas. Cada tupla es representada por los atributos que la definen, por la medición de dichos atributos se le puede otorgar a la tupla la pertenencia a una clase y etiquetarla una vez identificada su categoría, las tuplas de datos individuales involucradas en este primer paso son llamadas tuplas de entrenamiento (10).

En el segundo paso, el modelo puede ser usado para clasificación, para lo cual primero es determinada la exactitud predictiva del clasificador, ésto usando un conjunto de datos de prueba, estas tuplas son seleccionadas del conjunto de datos general y son independientes de las tuplas de entrenamiento, es decir, tuplas que no han sido usadas para construir el clasificador. Existen varios métodos para estimar la exactitud con la que un clasificador trabaja, en esencia esta exactitud es medida considerando el porcentaje de tuplas bien clasificadas de acuerdo al conjunto de pruebas usado, luego de lo cual si se considera conveniente el clasificador será utilizado para clasificar nuevos datos (10).

En nuestro caso vamos a usar un clasificador basado en términos frecuentes (conocido como TFIDF), se ha escogido este clasificador puesto que permite el filtrado de páginas y el refinamiento de consultas.

El clasificador de términos frecuentes busca encontrar la relevancia de palabras por cada categoría, es decir, las palabras que definen cada categoría. Esto lo logra analizando contenidos de páginas web que han sido previamente clasificadas, pesando la aparición de las palabras por categoría y comparando estas apariciones con las apariciones en el resto de las categorías. Luego esta información será usada para clasificar páginas que se encuentran fuera del banco de datos usados para el aprendizaje del clasificador y para el filtrado de páginas. En esta sección se muestra paso a paso como este clasificador fue construido.

Existen varios tipos de clasificadores entre, los más importantes se puede mencionar (10):

- Árbol de Decisión
- Clasificador Bayesiano
- Clasificador Basado en Reglas
- Redes Neuronales
- Clasificación Asociativa
- Clasificadores Perezosos

2.4.2. Frecuencia del término y frecuencia inversa de documento (TF/IDF)

Debido al tipo de clasificador que deseamos implementar, es importante definir cómo se determinará la importancia de los términos que son parte de las páginas web, e identificar los que definan cada categoría.

Esto se realiza a través del método TFIDF (frecuencia de término y la frecuencia inversa del documento), la cual es una técnica de minado de textos para la obtención

de información, y ha sido una pieza básica en lo que corresponde a la clasificación de contenido no estructurado (10).

Método que es una medida estadística usada para evaluar que tan importante es una palabra a un documento en una colección de documentos. La importancia se incrementa proporcional al número de veces que una palabra aparece en el documento pero también considerando la frecuencia de aparición en la colección (20) y (10). Existen algunas variaciones en el uso del TFIDF, pero la que se ha usado en este prototipo esta expresada en la fórmula 2.1.

$$TFIDF = (\frac{TotalCategorias}{NumeroCategoriasenlasqueAparecelaPalabra}) * PesoPalabra \quad (2.1)$$

En la fórmula 2.1 se muestra el cálculo del TFIDF para encontrar la importancia de un término en relación a una categoría, la fórmiula entonces dice que el número de categorías incolucradas se lo divide para el número de categorías en las cuales término estudiado aparece, resultado que se lo multiplica para el número de veces que dicho término aparece en la categoría que se esta analizando. Así se determina que tan importante es esa palabra para dicha categoría.

2.5. Comunicación con los motores de búsqueda convencionales

Una vez explicada la manera que funciona el clasificador y como los perfiles son creados, es el momento de mostrar como el prototipo se comunica con los motores de búsqueda convencionales, hay que destacar que la tecnología utilizada permite que diferentes formas de interacción con populares motores sea usada, así para este caso se han utilizado dos conocidos motores sobre los cuales se ha trabajado en este proyecto (en sus distintas etapas).

2.5.1. Google API

El prototipo funciona (en su primera etapa) sobre el conocido motor de búsqueda *Google*, la manera en la que el programa realiza peticiones a este motor es mediante el uso de un API, el nombre de esta aplicación es *Google SOAP Search API Reference* ver (6). Este es un servicio web que ofrece google para realizar consultas directas a las bases de datos del motor mediante la interfaz que el API ofrece, dicha interfaz está desarrollada en JAVA lo que facilitó su integración con el prototipo.

Para usar este API se necesita registrar una cuenta en google y pedir una clave que es necesario enviar al servicio desde el programa que implementa la interfaz, se crea un objeto de la interfaz y se envía una serie de parámetros que son; clave, consulta al motor, desde donde empieza a mostrar resultados y número de resultados a mostrar. Una vez que esta petición es enviada el servicio retorna los resultados considerando los parámetros que ha recibido y los encapsula en un objeto.

Este objeto es manejado en el prototipo y la información necesaria de los resultados es extraída para sea integrada con el clasificador y los perfiles de forma que nos permita filtrar las búsquedas. La información que se extrae es: URL a la página resultado, título y descripción.

La vigencia de este API fue descontinuada durante la ejecución de este proyecto, sin embargo Google ofrece la misma funcionalidad a través de otra tecnología (Javascript).

2.5.2. Bing API

El API Versión 2.0 de Bing, es un aplicativo que permite agregar un flexible y poderoso motor de búsqueda como un componente buscador personalizado en sitios web y aplicaciones. Entre sus ventajas se encuentra que puede entregar resultados para XML como para JSON, adicionalmente tiene soporte para SOAP, permite agregar publicidades y tiene una interfaz para trabajar con RSS. Por favor refiérase a (2) para mayor información.

Los pasos básicos generales para la integración de de este API son:

1. Obtener el AppID
2. Determinar valores de los parámetros requeridos.
3. Trabajar con parámetros opcionales.
4. Escoger un protocolo.
5. Enviar una petición.
6. Trabajar con los resultados.

Para una mayor referencias sobre como trabajar con este API y para ejemplos sobre como integrarlo en aplicaciones JAVA por favor mire las referencias;(2) y (3).

Una vez que se ha explicado cómo funciona el ensamblaje del motor de búsqueda con el prototipo tenemos todas las piezas para empezar a realizar la personalización de búsquedas, el detalle de como se realiza esto esta descrito en las siguientes subsecciones.

Capítulo 3

Implementación

3.1. Introducción

Recordemos que el objetivo final de este proyecto es el análisis de efectividad del filtrado de resultados y del refinamiento de consulta, sobre dos tipos de perfil de usuario, fijo y adaptativo.

Este capítulo contiene la descripción de como se encuentra implementado el prototipo de personalización de búsquedas. El programa que se ha desarrollado en diferentes módulos y componentes, los cuales interactuan para cumplir con los objetivos planteados.

La figura 3.1 muestra el diagrama de bloques propuesto para el desarrollo del prototipo, allí se ilustra como funcionaría el prototipo, donde como primer paso el usuario elige el tipo de perfil que quiere usar, luego se muestran los diferentes módulos necesarios para llegar a personalizar las búsquedas usando el filtrado de resultados y el refinamiento de consulta, para lo cual es necesario obtener los perfiles de usuario u usar el clasificador (que a su vez se alimenta de un módulo de aprendizaje, proceso que se ejecuta offline).

La descripción de la implementación se la realizará de acuerdo al esquema propuesto en la figura 3.1, el orden de análisis de cada uno de los componentes será el siguiente:

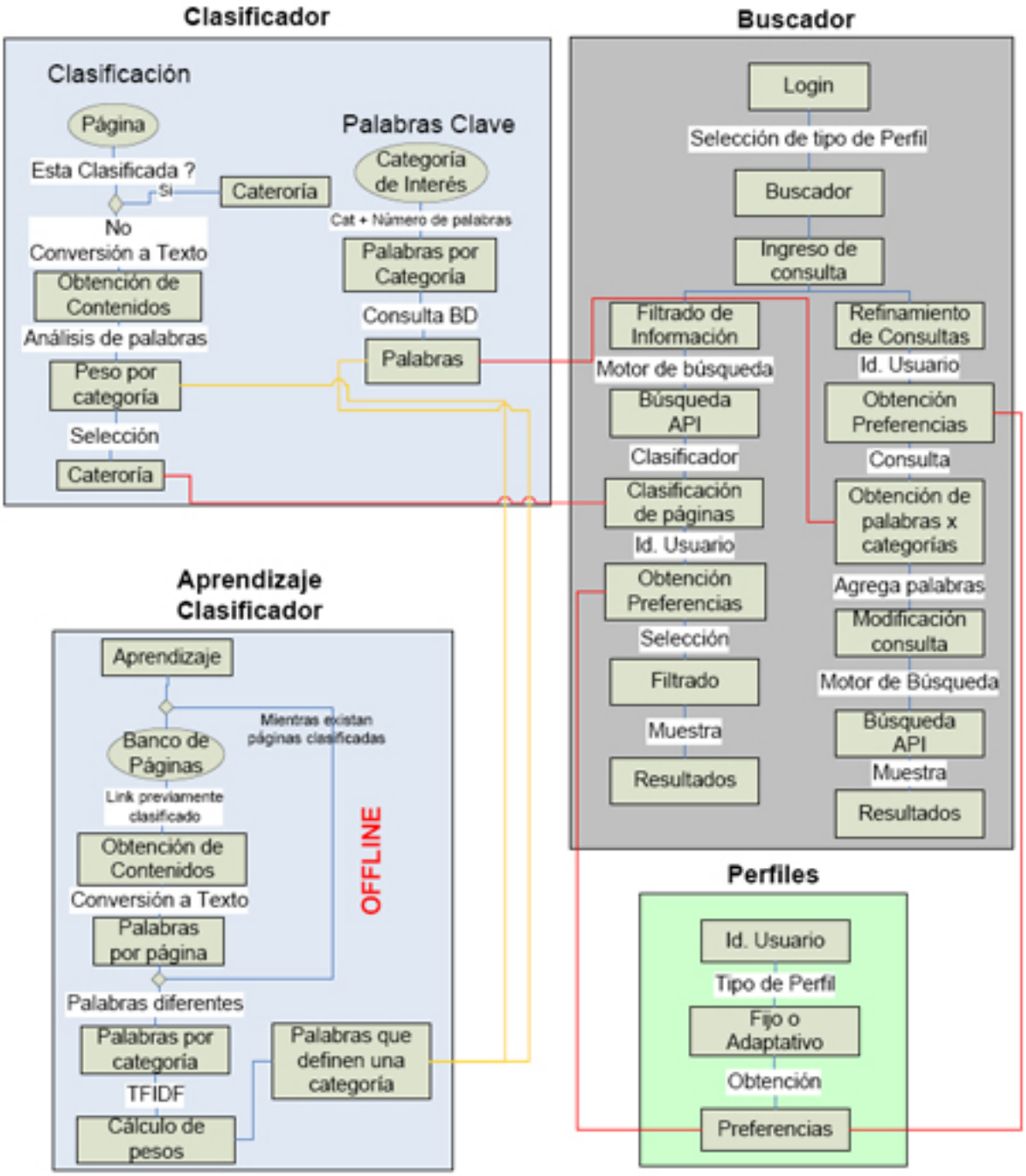

Figura 3.1: Diagrama de Bloques propuesto

- Construcción de perfiles, cómo en este prototipo se construye y manejan los perfiles.
- Mecanismos de Personalización de Búsquedas, basándose en el perfil de usuario.
- Clasificador básico, implementado en el prototipo para la categorización de páginas, su aprendizaje y funcionamiento.
- Estructura del prototipo, herramientas usadas, estructura de la base de datos, funcionamiento general y presentación de resultados.

3.2. Perfiles de Usuario

El perfil es construido para saber las preferencias del usuario frente a las categorías que son utilizadas en este prototipo (ver tabla 3.1), así sabemos cuáles son las categorías en las que el usuario tiene interés, esta información es utilizada en combinación con el clasificador para filtrar las páginas. El perfil fue construido de dos maneras; fija y adaptativa, a continuación se explica cómo cada una de estas opciones están implementadas. La sección 2.2 del capítulo del marco teórico se refiere a la creación y tipos de perfiles.

3.2.1. Perfil Fijo

Es el más sencillo de implementar y administrar, involucra la interacción explícita del usuario, éste a través de un formulario realiza una selección directa de las categorías que le interesan. Así de esta forma se construye una lista por cada usuario donde un booleano es prendido cuando una categoría es de interés. El formulario en el prototipo es construido de tal manera que las selecciones que el usuario haya realizado puedan ser modificadas cada vez que éste lo considere necesario. Entonces si más adelante el usuario considera que los resultados que se le presentan abarcan temas que no son

útiles, se dirige a la sección *Mi Perfil* y cambia las preferencias que tiene seleccionadas. La figura 3.2 muestra como el formulario es manejado.

3.2.2. Perfil Adaptativo

El perfil adaptativo consiste en almacenar las preferencias del usuario a través del aprendizaje de su comportamiento mientras utiliza el buscador. Lo que esto quiere decir es, que se va alimentando la tabla de categorías de interés basándonos en las elecciones de resultados que el usuario haga mientras usa el prototipo, así este perfil se va adaptando a las preferencias del usuario. Para construir este perfil el usuario no necesita mostrar interés explícito por las categorías, sino que de las páginas que el vaya explorando se construye el perfil. Para el marco teórico específico de este tipo de perfil por favor refiérase a la sección 2.2.1.2.

En este perfil cada categoría tendrá un valor que define que tan interesado esta el usuario en ella. A este valor lo llamamos *preferencia* de la categoría, este valor es de tipo *double*, el máximo que puede ser asignado para la preferencia es *1*. Este valor se irá alterando dependiendo de cómo el usuario vaya navegando por los resultados, y es modificado y usado de la siguiente forma:

1. Las preferencias del perfil se inician en 1: Cuando el usuario no tiene un historial de búsqueda, es decir, es la primera vez que utiliza el buscador en el modo *Perfil Adaptativo*. Todos los valores de las preferencias para cada categoría son establecidos en 1. Así el usuario la primera vez que usa el buscador está interesado en todas las categorías. Punto 1 ejemplo 3.3.

2. El usuario da click en algún link de interés: Una vez que una búsqueda es realizada el usuario irá navegando por los resultados, cada vez que de un click en una página, la categoría a la que esta página pertenece se vuelve una categoría de interés, aún no podemos decir que tanto interés tiene el usuario en ella, esto lo veremos en los siguientes pasos. Punto 2 ejemplo 3.3.

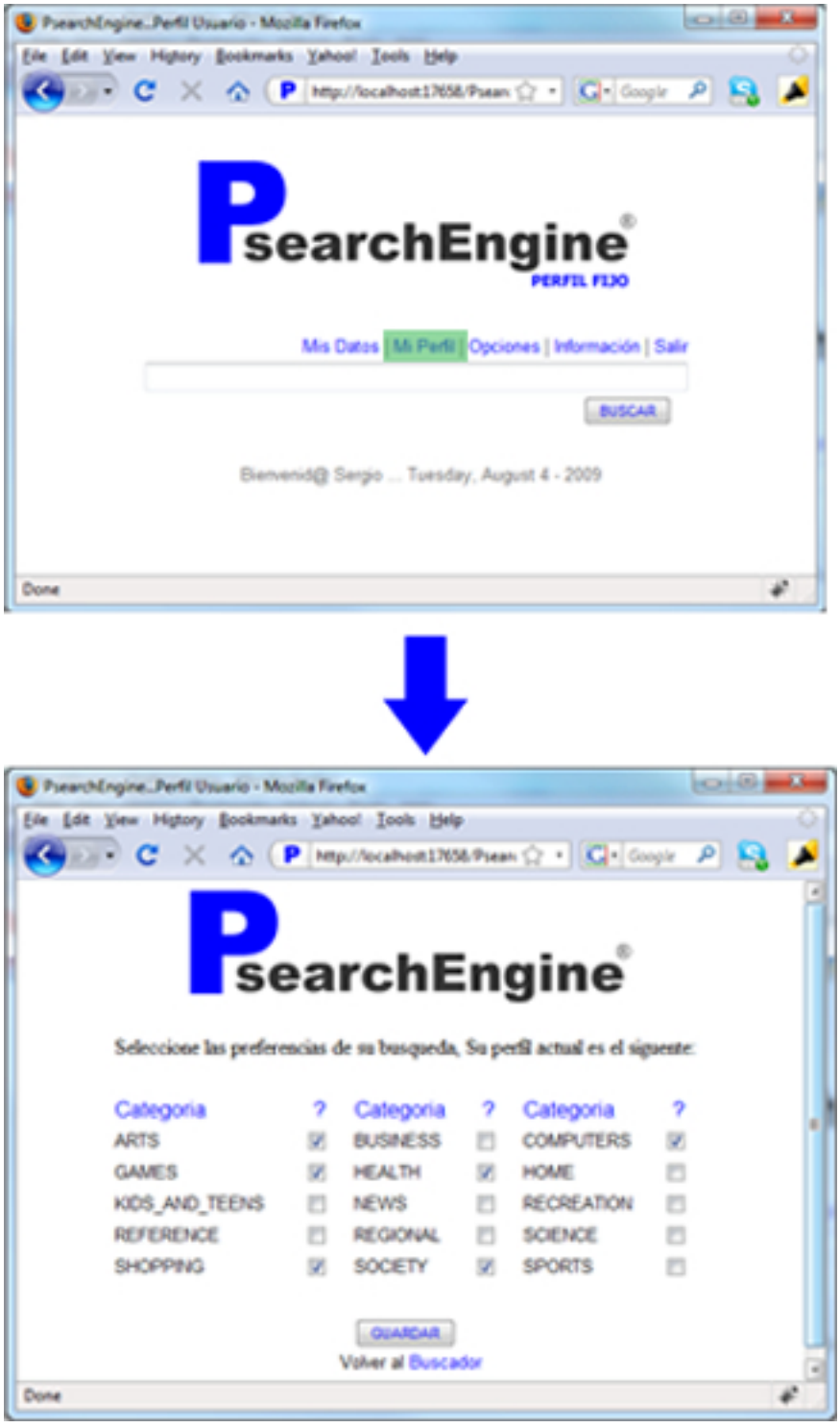

Figura 3.2: Construcción del Perfil Fijo.

3. Se incrementa el valor de la preferencia: Al saber que una categoría es de interés, el valor de su preferencia es aumentado en una cantidad, esta cantidad es una variable en el programa, por defecto el prototipo arranca con un valor de 0.1 para esta variable. Parte de esta tesis es ir ajustando el valor de manera que sea el más óptimo para la construcción del perfil. Punto 3 ejemplo 3.3.

4. Normalización de los valores de preferencia: Una vez que se ha incrementado la categoría de interés, se procede a normalizar los valores de preferencias de todas las categorías, para esto se divide cada valor de preferencia para el valor mayor de preferencia que exista entre todas las categorías. Así mientras el usuario elije páginas de varias categorías estas son incluidas en el perfil y dependiendo del número de veces que elija cierta categoría, esta será más o menos importante en el perfil global. Punto 4 ejemplo 3.3.

5. Escoger las categorías preferidas, usando un umbral: Ahora si bien cada categoría tiene su valor de preferencia cómo se eligen aquellas que son las preferidas, es decir, cómo decimos que un usuario está interesado en determinada categoría o no lo está. Para esto usamos un umbral, no es más que un valor, que resulta nuevamente ser una variable del programa que nos dice, que si el valor de preferencia de la categoría es mayor que el valor establecido como umbral entonces la categoría es de interés. El umbral por defecto es de 0.8, sin embargo este valor es también sometido a cambios y este proyecto tratará de determinar cuál debería ser su valor óptimo, esto a resolverse en el capítulo de evaluación. Punto 5 ejemplo 3.3.

3.3. Mecanismos de Personalización de Búsquedas

Con la implementación descrita en las secciones anteriores estamos listos para iniciar el proceso de personalización de búsquedas, como ya se ha mencionado en este

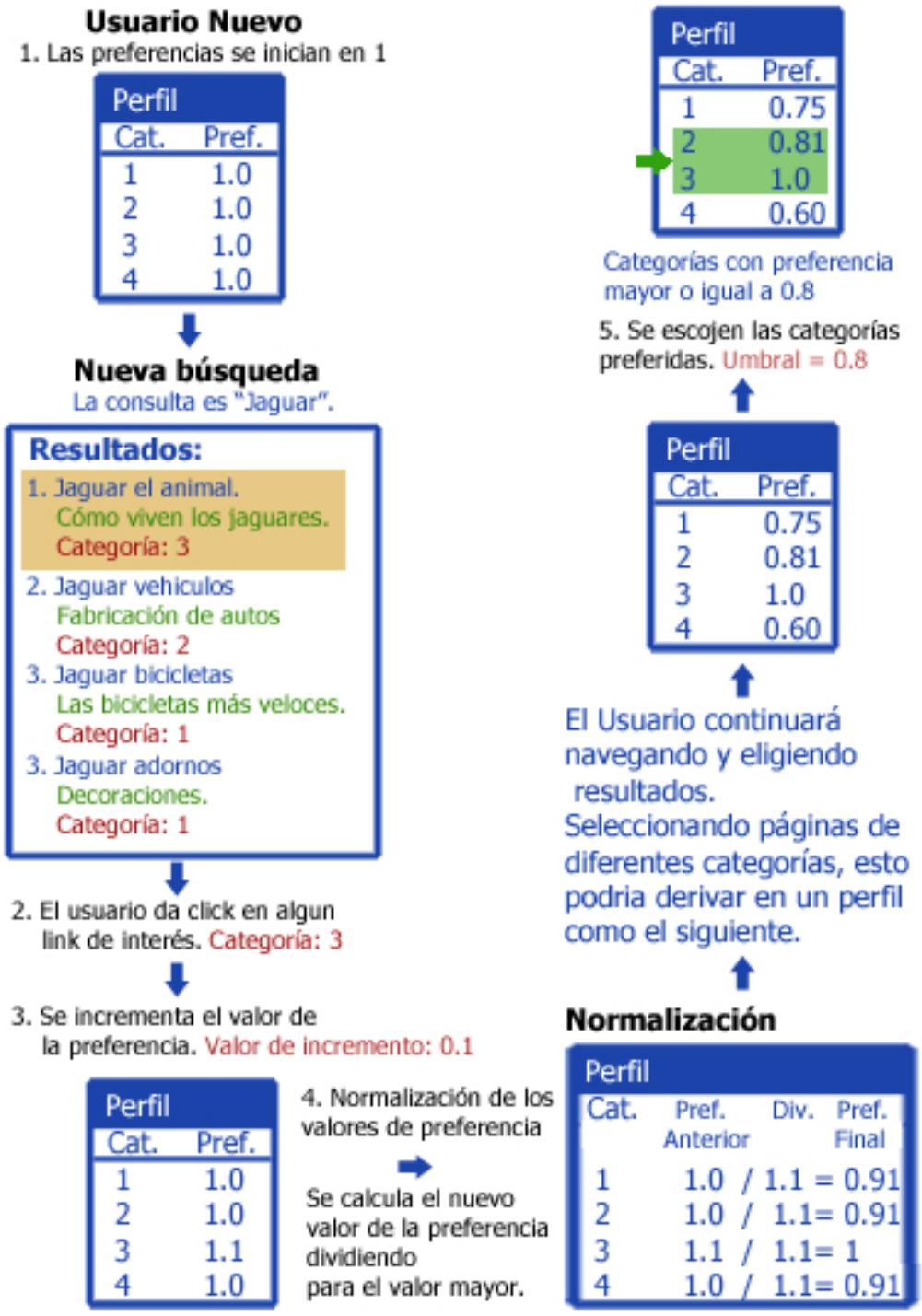

Figura 3.3: Construcción del Perfil Adaptativo.

documento lo realizamos de dos formas; en base a categorías de interés y mediante refinamiento de consulta. El detalle de como esto se realiza se explica en esta sección.

3.3.1. Filtrado de Páginas

Este filtrado se lo realiza sobre los resultados obtenidos de la búsqueda normal usando el motor, esto quiere decir que lo primero que se hace es enviar la consulta tal y como el usuario la ha introducido en el prototipo al motor de búsqueda (usando el API), el filtrado se lo realizará sobre estos resultados. Hay que mencionar que en el prototipo los resultados de una búsqueda son mostrados de diez en diez (por defecto, se puede cambiar). El filtrado se lo va a ir realizando sobre los resultados que se están mostrando en ese momento. El marco teórico correspondiente se encuentra en 2.3.1.

Entonces se realiza una consulta y el API devuelve los resultados, a continuación el prototipo clasificará cada una de las páginas que han sido retornadas (en el primer caso los primeros diez links y seguirá así dependiendo de página de resultados que se esté mirando), la clasificación de la página es hecha como esta explicado en la sección 3.4.2.

Una vez que sabemos las categorías a las que pertenecen las páginas resultado, obtenemos las categorías de preferencia del usuario, ya sea que se esté usando el perfil adaptativo o el perfil fijo (por favor refiérase a las secciones 3.2.2 y 3.2.1 para más información), con el conocimiento de las categorías que le interesan al usuario entonces se procede a elegir los resultados a mostrar de los que fueron retornados por el API, solamente se mostraran aquellas páginas que pertenezcan a categorías que al usuario le interesan.

La figura 3.4 ilustra el proceso de filtrado a través de categorías de interés.

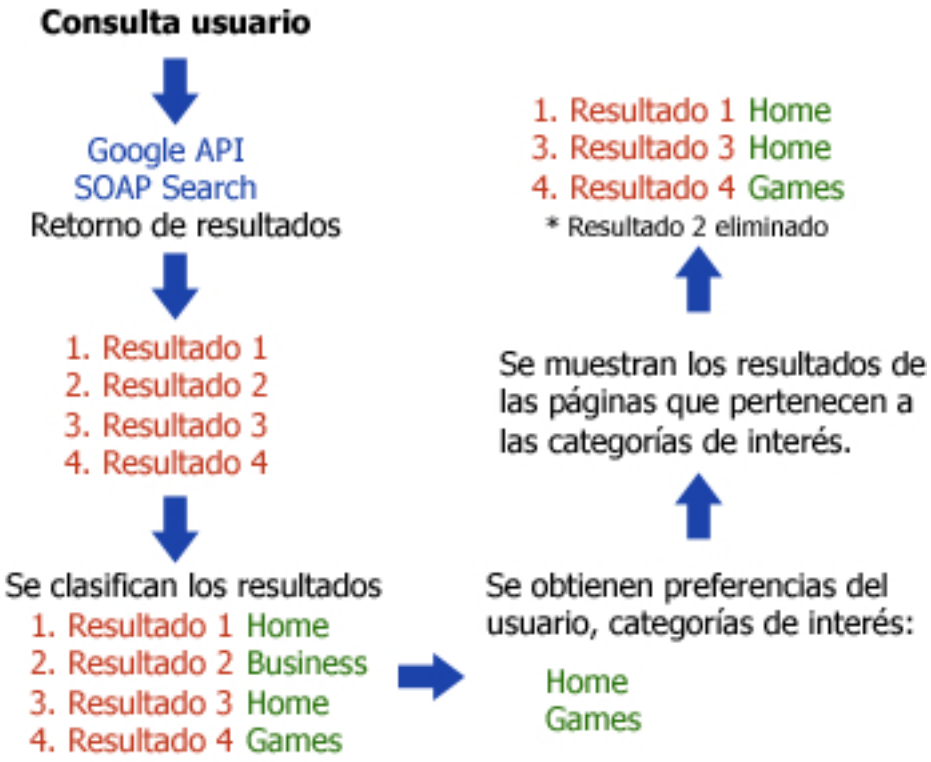

Figura 3.4: Filtrado de Resultados.

3.3.2. Refinamiento de Consulta

En el caso del refinamiento de consulta el mecanismo es diferente, puesto que la consulta al motor de búsqueda es modificada antes de que la petición al motor sea realizada, es decir, el usuario ingresa la consulta en el prototipo después a ésta se le modifica considerando las preferencias del usuario y el resultado es enviado al motor. Por favor vaya a la sección 2.3.2 para mayor información.

Lo primero que se realiza en este caso es obtener las categorías de interés del usuario, sin importar si se está usando el perfil fijo o el perfil adaptativo (par ver información de perfiles diríjase a las secciones 3.2.2 y 3.2.1). Una vez que se obtienen las categorías preferidas se consulta las palabras importantes para cada una de éstas, el número de palabras que se obtendrán por categoría es una variable del programa, en el capítulo 4 de evaluación se harán una serie de pruebas para determinar cuál es el óptimo número de palabras a obtener por categoría para este filtrado. Por defecto el número de palabras

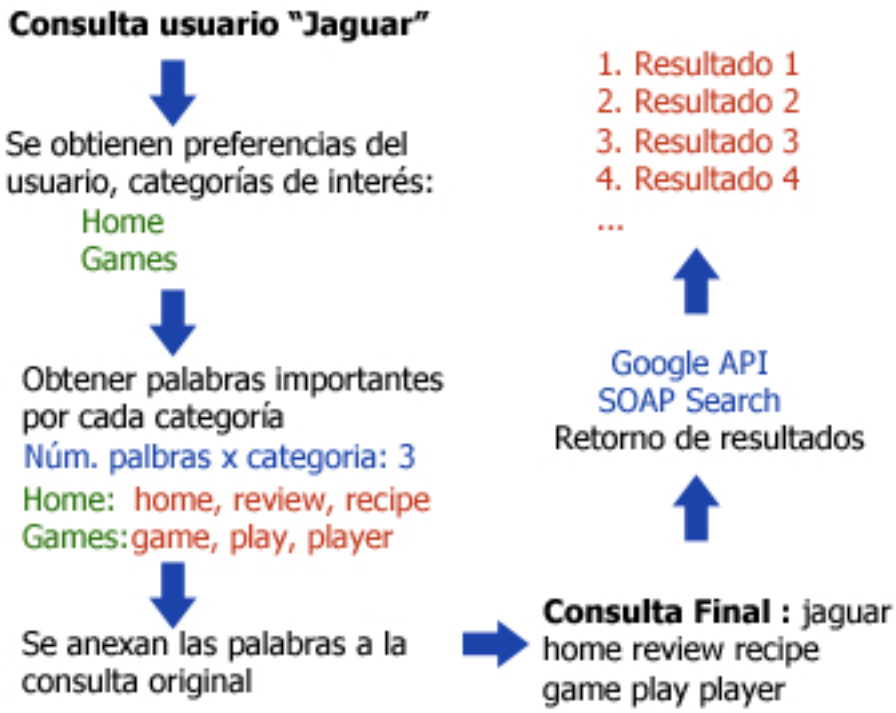

Figura 3.5: Refinamiento de Consulta.

importantes a extraer es tres.

Ahora las palabras importantes por cada una de las categorías de interés son anexadas a la consulta original que el usuario realizó y la búsqueda es enviada al motor, el API retorna los resultados y estos son presentados al usuario en el orden que el buscador originalmente ha establecido, es decir simplemente son renderizados.

La figura 3.5 ilustra este proceso de filtrado de información mediante el refinamiento de consultas.

3.4. Clasificador

Cabe mencionar que el clasificador es simple pero eficiente en los mecanismos de filtrado de resultados y refinamiento de información. Este prototipo puede ser utilizado como base para la prueba y estudio de otras formas de clasificación.

3.4.1. Aprendizaje del Clasificador

3.4.1.1. Conjunto de datos de Entrenamiento (Links Clasificados)

El primer paso para construir el clasificador fue proveerle al programa de un banco de datos conocido sobre páginas web que habían sido ya previamente encasilladas en una categoría, para esto se utilizó la base de datos que se ofrece en el *Open Directory Project* (5), este es un proyecto que consta de aproximadamente 83 mil editores voluntarios quienes se encargan de clasificar páginas que se encuentran en la Web, consta con 590,000 categorías (este número es considerando subcategorías), existen 16 categorías principales y aproximadamente 4,500,000 páginas clasificadas.

Por esto se dice que el *Open Directory Project* (5) es el directorio editado por humanos más largo y completo, debido a la vasta comunidad de editores voluntarios. Es posible también agregar cualquier página a la lista de sitios a ser clasificados a través del sitio web del proyecto.

De las categorías existentes para este trabajo se tomaron en cuenta las primeras 15 debido a que el clasificador funciona para un solo lenguaje que es Inglés, por lo que se descartó el uso de la categoría *World* ya que involucra páginas en otros idiomas. De igual manera no fueron consideradas las subcategorías de cada una de las categorías. En la tabla 3.1 se ilustran las categorías usadas en el clasificador.

La información que *Open Directory Project* ofrece está dada en un XML, la figura 3.6 muestra un ejemplo de como este XML esta estructurado en la parte que es de interés para este proyecto.

De la información mostrada en el XML (figura 3.6) se extraen tres piezas fundamentales para ser insertadas en la base de datos que luego serán usadas por el clasificador; se usa el *Topic* que es un identificador descriptivo para la categoría (o subcategoría), el *catId* que es un identificador numérico único para dicha categoría (o subcategoría) y finalmente la información que se encuentra dentro del tag *link* que especifica la di-

Id	Categoría	Identificador en XML	Número de Links
1	Arts	Top/Arts	263871
2	Business	Top/Business	241973
3	Computers	Top/Computers	123144
4	Games	Top/Games	56315
5	Health	Top/Health	61148
6	Home	Top/Home	30306
7	Kids and Teens	Top/Kids and Teens	43975
8	News	Top/News	8380
9	Recreation	Top/Recreation	111384
10	Reference	Top/Reference	58473
11	Regional	Top/Regional	1110133
12	Science	Top/Science	104696
13	Shopping	Top/Shopping	101061
14	Society	Top/Society	246582
15	Sports	Top/Sports	104762

Tabla 3.1: Categorías usadas en el Prototipo

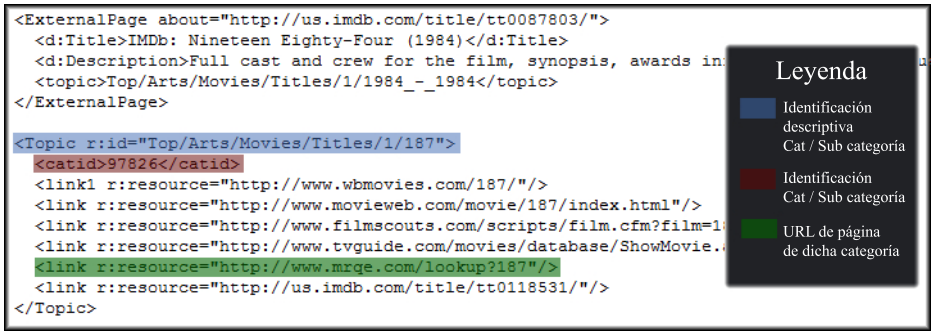

Figura 3.6: XML content ODP Example

Figura 3.7: Links ODP insertados en la Base de Datos

rección de sitios web que pertenecen a la categoría del tag *Topic* que lo contiene. La figura 3.7 muestra la información insertada ya en la base de datos.

Cabe destacar que la información que está almacenada en el *rId* que corresponde a *Topic* en el XML identifica las categorías y subcategorías, aquí se encuentran descritos los niveles de las distintas categorías separados por el caracter *'/'*, por ejemplo todas las categorías pertenecen a un nivel superior definido como *Top*, en el caso de la figura 3.6 tenemos que *Topic* (señalado en azul) pertenece a la categoría *Arts* y *Movies/Titles/1/187* es una subcategoría de esta categoría, lo que quiere decir también es que todos los tags *link* dentro de este *Topic* pertenecen a esta categoría y subcategoría. Es importante tomar en cuenta esto puesto que más adelante esta información será usada para obtener determinado número de links por subcategoría.

Ahora la cantidad de links que se tiene por cada categoría (expresadas en la tabla 3.1) es demasiado grande para su procesamiento, es decir, obtener los contenidos de cada una de estas páginas resulta una tarea demasiado pesada en términos de tiempo de procesamiento computacional. Por este motivo se buscó un número de links por categoría aceptable que nos permita obtener los contenidos de las páginas en un tiempo realizable y llevar a cabo los trabajos necesarios para que el clasificador funcione, los cuales son explicados a lo largo de este capítulo. Luego de múltiples pruebas sobre la cantidad de links y cálculos de tiempo se llegó a que *5000 links* por categoría es el número apropiado, esto nos permitió procesar completamente la información para

cada categoría en un tiempo aproximado de entre 3 y 4 días, complementariamente nos permitió tener un número aceptable de páginas para que el clasificador identifique los términos importantes por categoría.

Sin embargo el hecho explicado en el párrafo anterior no es un limitante para el prototipo como tal, pues en el caso de existir más recursos, es factible que el prototipo trabaje con una cantidad de datos mayor o con una base de datos diferente sin ningún problema.

Dado que no se pueden usar todos los links por página, una problemática adicional surgió, puesto que cada categoría tiene subcategorías a varios niveles no se pueden tomar los primeros 5000 links de una categoría, ya que probablemente se estaría excluyendo un gran número de páginas de varias subcategorías, ahora lo que se hace en este prototipo es sacar los 5000 links por categoría pero tratando de tomar en cuenta todas las subcategorías para esa categoría. Para esto se usa el numero de subcategorías por categoría y el número de links que se quiere obtener por categoría, así se puede calcular el número links que es necesario extraer por subcategoría, con el objetivo de llegar a completar el número de páginas que se quiere alcanzar globalmente en la categoría. De esta manera la muestra de páginas por categoría se vuelve mucho más representativa ya que involucra a todas las subcategorías que conforman dicha categoría.

Después de correr el algoritmo que ha sido descrito en el párrafo anterior se crea una nueva tabla en la base de datos, esta tabla contiene un identificador de los links a ser tomados en cuenta por categoría, esta tabla es la que será recorrida para obtener los contenidos de cada una de las páginas, la figura 3.8 muestra un ejemplo de cómo está almacenada la información en esta tabla.

3.4.1.2. Obtención de Contenidos (Conversión a Texto)

Una vez que tenemos la tabla de los links a analizar por categoría, el siguiente paso es la extracción de contenidos. Para esto se atraviesa por cada uno de los links y se

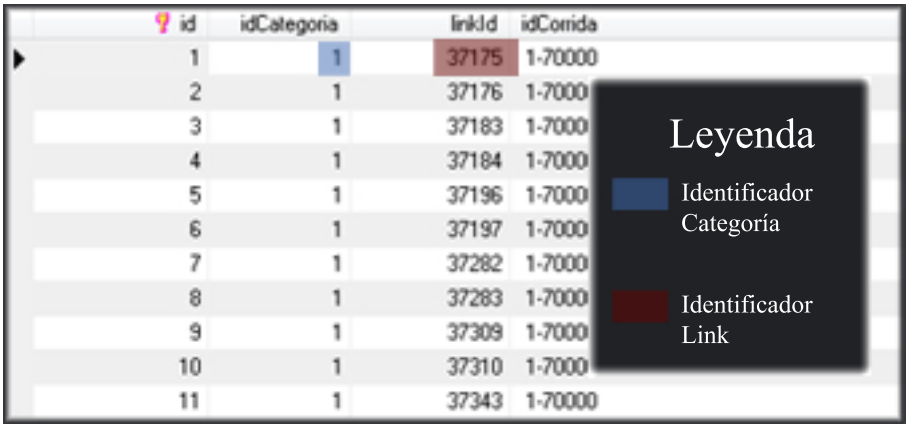

Figura 3.8: Links finales por categoría

realiza una petición *HTTP* usando el *URL* de cada página. El objetivo de esto es tener la información útil para el clasificador que está disponible en cada uno de los sitios, es decir, lo que los usuarios en realidad leen. La figura 3.9 muestra este proceso.

La petición es realizada a través de *Hilos*, fue implementado usando esta técnica para básicamente tener control sobre el tiempo de espera de la petición HTTP, pues en muchas ocasiones la petición puede quedarse en espera por siempre, como por ejemplo en el caso de que la página a la que se esté tratando de acceder sea un *jarro de miel* o el servidor se encuentre con sobrecarga, si no hubiese control sobre esta problemática todo el programa de extracción de contenido se paralizaría. Así al usar hilos se pueden seguir obteniendo contenidos si el link actual que se está tratando de acceder excede el tiempo de espera permitido, es decir, para el prototipo si una petición a determinada página no se responde en un tiempo máximo de 60 segundos, se descarta la obtención de contenidos para esa página y se procede con la siguiente en la lista. De igual manera esto se encuentra ilustrado en la figura 3.9.

Para obtener esta llamada información útil para el clasificador es necesario remover información adicional que es retornada cuando una petición *HTTP* es resuelta. Esta información adicional es la siguiente y ha sido removida en el siguiente orden:

1. Extra HTML: En este paso todo texto que pueda surgir de errores en la creación de la pagina es eliminado, es decir todo aquello que este fuera de los tags <*html*>

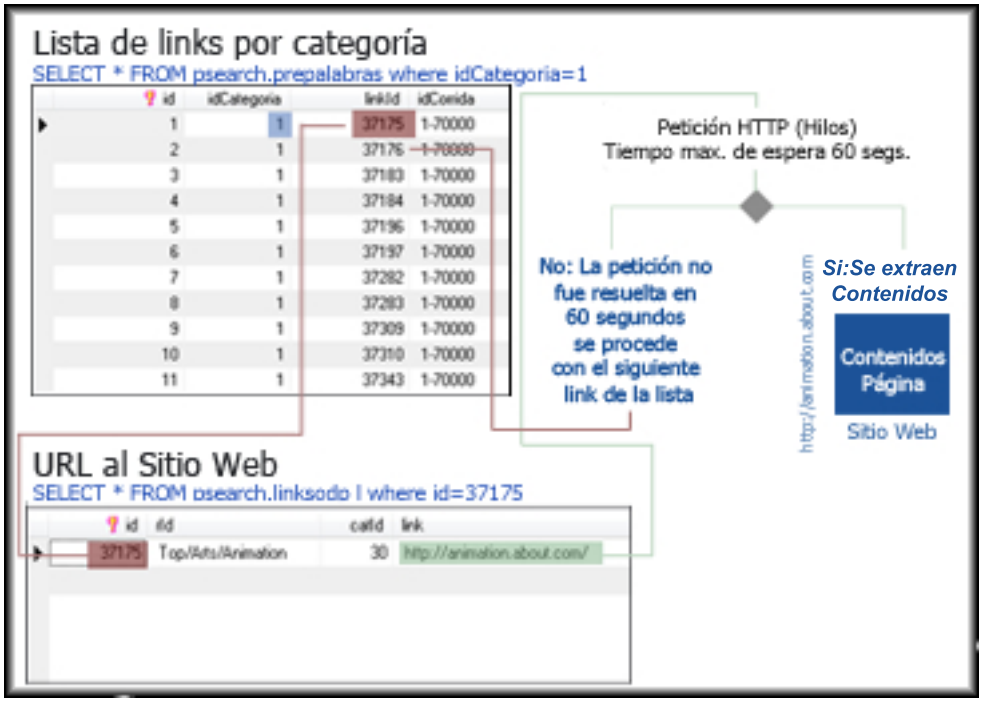

Figura 3.9: Extracción Contenido Web

y *</html>*. Ver la parte superior de la figura 3.10.

2. Tag Style: Todo lo que este dentro de los tags *<style>* y *</style>* se descarta, pues esto enmarca los estilos de la página que sirven solamente para la visualización de la misma, por lo tanto no es relevante para obtener los contenidos.Ver la parte superior de la figura 3.10.

3. Scripts: De igual manera cadenas de caracteres dentro de los tags *<script>* *</script>* se remueven, ya que en la mayoría de los casos todo lo que este adentro de estos involucra código de programación en javascript para darle funcionalidad extra a la página, por lo que no es de interés para el clasificador.Ver la parte superior de la figura 3.10.

4. Tags HTML: Tags HTML no son relevantes para analizar contenidos en el prototipo, por lo tanto cualquier tag que represente código HTML en el texto retornado por la página es borrado, por ejemplo, <body>, <head>, <p>,
, etc. Ver la parte superior de la figura 3.10.

5. Limpieza de caracteres especiales: En este paso se limpian todos los caracteres que son considerados especiales, tanto en el lenguaje HTML como en el inglés, ya que estos no son útiles para encontrar palabras relevantes en los contenidos. Entre éstos encontramos por ejemplo: '&', '"', '?', '.', ':', saltos de línea, tabulaciones, etc. Ver la parte media de la figura 3.10.

6. Palabras comunes: Se eliminan las palabras comunes propias del inglés, palabras como; of, the, away, are, they, we, I, etc. Esto se lo realiza con el uso de de la clase commonWords que se basa en el código publicado en (19), no todas las funcionalidades de este código fueron usadas, solamente aquellas necesarias para el prototipo fueron acopladas. Ver la parte inferior de la figura 3.10.

7. Reducción de palabras a su raíz gramatical: Una vez que tenemos solamente las palabras que nos van a servir para el análisis de relevancia, se reducen todas las palabras a su raíz gramatical, por ejemplo, las palabras *searching* y *searched* serán convertidas a *search*, y solamente se tomará en cuenta una sola palabra. Para esta parte se usó el código de Porter (15). Ver la parte inferior de la figura 3.10. Con esto se evita que se someta a un análisis redundante la misma palabra, por estar conjugada de diferentes maneras.

Una vez que todos estos pasos han sido ejecutados lo que obtenemos son las palabras relevantes por cada una de las páginas que se han analizado, éstas palabras son separadas por comas y almacenadas en la base de datos con un identificador de link y categoría a la que pertenecen. Esto se muestra en la parte final de la figura 3.10. Una vez que esta información es almacenada lo siguiente que se debe hacer es calcular los pesos para cada palabra, esto quiere decir saber que tan relevante es la palabra dentro de su categoría, este procedimiento es explicado a detalle en la próxima sección.

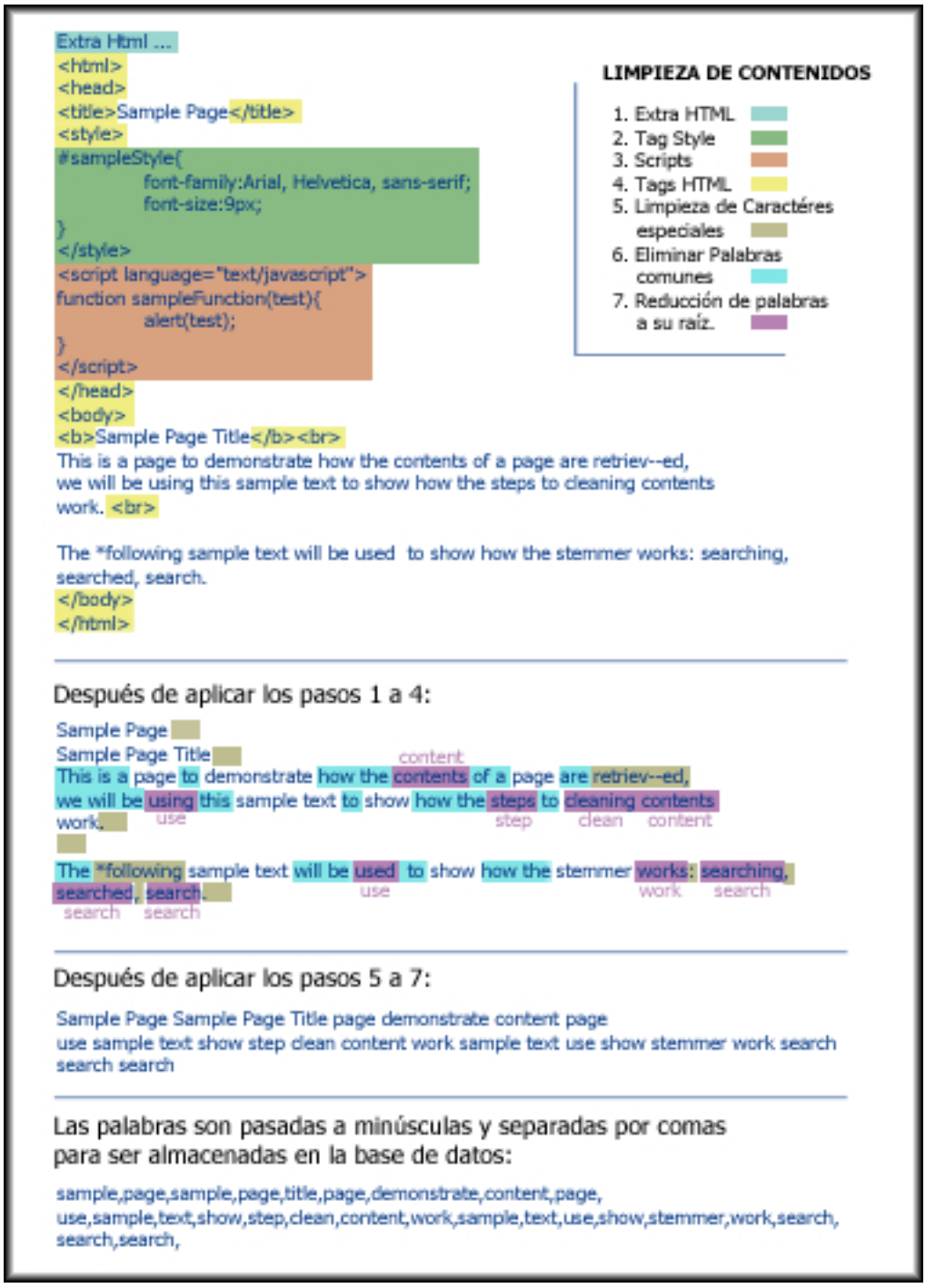

Figura 3.10: Proceso de obtención (limpieza) de contenidos de una página.

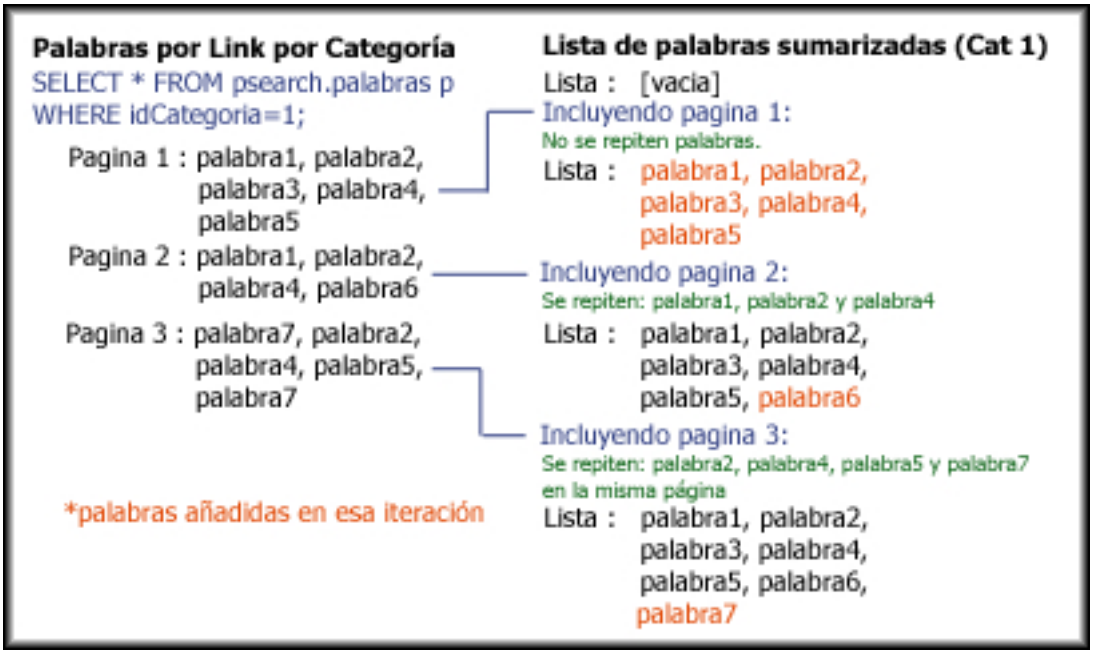

Figura 3.11: Palabras por Categoria sumarizadas.

3.4.1.3. Cálculo de Pesos / TFIDF

Como ya se había mencionado antes el objetivo del cálculo de pesos es saber que tan relevante es una palabra dentro de la categoría a la que pertenece. Esto se logra en varios pasos.

El primero es reunir las palabras diferentes por categoría, esto se obtiene de la lista que contiene las palabras relevantes por cada link en determinada categoría. Para esto es necesario recorrer esta lista y de cada uno de los links obtener las palabras correspondientes, estas palabras son añadidas a una lista global (lista de palabras por categoría), donde no se almacenan palabras repetidas. Esto desde el punto de vista de programación se lo realiza usando un *HashSet*, ésta es una estructura de datos que permite añadir items de determinado tipo, una de sus características es que si el item ya existe en la lista este no será añadido. La figura 3.11 muestra un ejemplo de la sumarización de palabras por categoría.

Una vez que las palabras han sido sumarizadas éstas son almacenadas en una tabla similar a la tabla 3.2, nótese que los datos son correspondientes a los de la figura 3.11

(continuamos con el mismo ejemplo). Cómo se obtiene el campo *Peso* será explicado más adelante en esta subsección.

Id	idCategoría	Palabra	Peso
1	1	palabra1	2
2	1	palabra2	3
3	1	palabra3	1
4	1	palabra4	3
5	1	palabra5	2
6	1	palabra6	1
7	1	palabra7	2

Tabla 3.2: Tabla de almacenamiento palabras por categoría

Encontradas las palabras por categoría ahora se calcula que tan relevante es cada una de estas palabras dentro de su categoría, este es el cálculo del peso. Para esto se recorre la lista de palabras por categoría y se cuenta cuantas veces aparece la palabra en los links de dicha categoría. Así el peso es la ocurrencia de la palabra en los links de la categoría que se esta analizando. La figura 3.12 muestra como esto es realizado para el ejemplo que hemos venido manejando en esta subsección.

Los pesos calculados corresponden a la columna *Peso* de la tabla 3.2, la cual es almacenada en la base de datos. El peso de la palabra es un factor fundamental para el cálculo del *TFIDF*.

El peso de la palabra nos permite saber la relevancia de la misma dentro de la categoría a la que pertenece, pero esto no es suficiente para saber que tan importante es la palabra con respecto a las otras categorías, es decir, si la palabra es lo suficientemente importante para definir a la categoría. Para esto se usa el TFIDF, por favor refiérase a la sección 2.4.2 para detalles del cálculo de esta métrica.

El total de categorías en el caso del prototipo es 15 (refiérase a la tabla 3.1), este valor es dividido para el número de categorías en las que la palabra analizada aparece, y a este resultado se le multiplica el peso de la palabra, así se obtiene el TFIDF.

La figura 3.13 ilustra este cálculo para nuestro ejemplo, el total de categorías en este

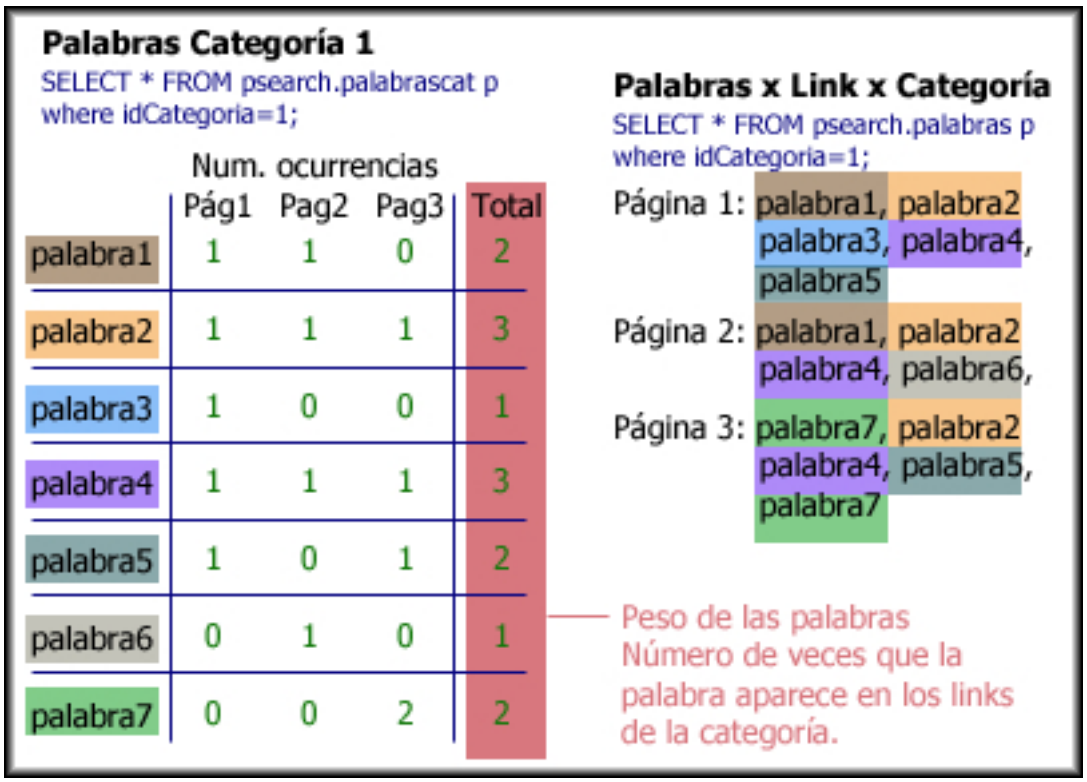

Figura 3.12: Cálculo de pesos palabra.

caso es 4. Nótese que si la palabra existe en la categoría esta listada, en este ejemplo no es necesario saber el peso de la palabra en las otras categorías sino solamente los pesos para la categoría que se está analizando, para el ejemplo es la número 1. Aquí se lista las palabras de la primera categoría, el peso, número de las categorías en las que aparece y el TFIDF calculado, de acuerdo a la fórmula 2.1.

De esta forma con el valor de TFIDF se puede determinar que palabras son las más importantes de la categoría, es decir, que palabras son las que pueden ser usadas para definir la categoría. Para nuestro ejemplo (véase la figura 3.13) decimos que las palabras: palabra2, palabra5 y palabra6; son las más importantes. Mientras más grande es el valor de este cálculo más importante es la palabra. La tabla 3.3 muestra la tabla final de las palabras por categoría que sería almacenada en la base de datos para este ejemplo.

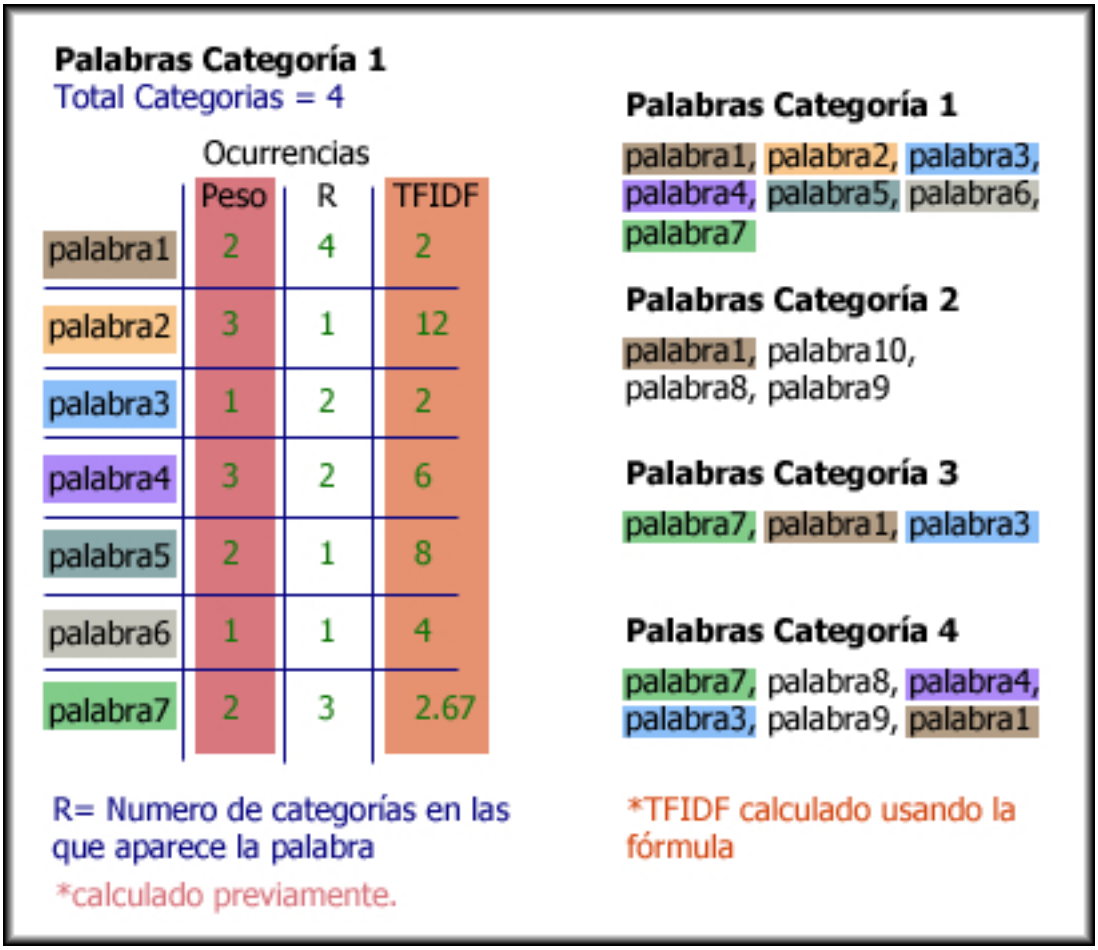

Figura 3.13: Cálculo de TFIDF.

Id	idCategoría	Palabra	Peso	TFIDF
1	1	palabra1	2	2
2	1	palabra2	3	12
3	1	palabra3	1	2
4	1	palabra4	3	6
5	1	palabra5	2	8
6	1	palabra6	1	4
7	1	palabra7	2	2.67

Tabla 3.3: Tabla de almacenamiento palabras por categoría

Después de correr estos algoritmos en nuestro prototipo encontramos los resultados para las 15 categorías que estamos manejando. En la tabla 3.4 se muestran las primeras 10 palabras que la definen basándose en el valor del TFIDF, los valores de peso y TFIDF no son mostrados en esta tabla.

Id	Categoría	Palabras
1	Arts	page, animation, site, work, show, art, time, link, movie, voice
2	Business	service, farm, product, sale, alpaca, contact, home, site, information, business
3	Computers	computer, system, informacion, page, network, software, file, service, program, web
4	Games	game, player, play, card, page, chess, casino, online, site, rule
5	Health	alcohol, health, drug, treatment, life, information, meet, addict, service, site
6	Home	review, recipe, car, price, home, product, site, side, make, store
7	Kids and Teens	game, music, review, video, site, cheat, work, time, play, feature
8	News	time, media, post, comment, world, read, home, report, student, story
9	Recreation	car, page, site, club, engine, post, part, show, member, link
10	Reference	student, universe, program, information, college, texas, service, home, research, center
11	Regional	site, information, home, service, africa, contact, pag, link, uk, country
12	Science	plant, language, product, information, site, research, inverse, page, crop, development
13	Shopping	product, price, home, order, free, custom, phone, design, accesories, model
14	Society	people, indian, american, time, site, page, page, state, member, information, home
15	Sports	team, baseball, game, basketball, league, player, sport, home, schedule, season

Tabla 3.4: Tabla de almacenamiento palabras por categoría

Estas son las palabras que son usadas por el clasificador, con esto se catalogan nuevas páginas para el mecanismo de filtrado de páginas, la forma de hacerlo esta descrita en

la próxima sección. Adicionalmente éstas son las palabras que serian añadidas a la consulta, en el mecanismo de refinamiento de ser el caso.

3.4.2. Clasificación de nuevas páginas

Con la lista de palabras que definen cada categoría catalogamos páginas nuevas, es decir, páginas que no se encuentran en el banco de links que fue utilizado para construir el clasificador. Para realizar esta tarea se necesitan dos parámetros; el url de la página y el número de palabras importantes por categoría que se van a utilizar. Una vez que se conocen estas dos variables se realizan los siguientes pasos:

1. Comprobar si la página ha sido clasificada previamente: Al momento en que llega un nuevo URL a ser clasificado lo primero que se verifica es si este URL no ha sido clasificado anteriormente, para esto usamos una tabla llamada *paginasclasificadas*, aquí se almacenan todas las páginas que ya han sido clasificadas. De esta forma si un URL de determinada página viene nuevamente a ser clasificado ya se sabe a qué categoría pertenece y no se gasta tiempo catalogando la página. Así si este paso se cumple y la página ya ha sido asociada con una categoría, ninguno de los siguientes pasos es realizado, simplemente se devuelve a que categoría dicha página pertenece. Ver el punto 1 del gráfico 3.14.

2. Obtener los contenidos de la página: En caso de que la página no esté en la tabla *paginasclasificadas* se procede a clasificarla, para esto lo primero es obtener los contenidos de la página, ver sección 3.4.1.2. Ver el punto del gráfico 3.14.

3. Determinar el peso de la nueva página en cada categoría: Con los contenidos de la página obtenidos calculamos la similitud de la página con cada una de las categorías, para esto se obtienen las palabras importantes para cada categoría, hay que considerar el número de palabras que se van a manejar (el parámetro mencionado arriba), se cuenta cuantas veces aparece cada una de las palabras en

la página y se suman las ocurrencias. Así se va obtener un valor que lo llamamos *peso* (en este caso de la página) y cada categoría va a tener un peso relacionado con la página en cuestión. Punto 3 del gráfico 3.14.

4. Verificar la categoría con la que la página se identifica más: Ahora se decide cual de las categorías tiene el peso más elevado con relación a la página, y esto determinara que la página pertenece a esa categoría. En caso de que la página tenga el mismo peso para dos o más categorías, se decide aleatoriamente por una de ellas. Punto 4 del gráfico 3.14.

5. Almacenar el url y la categoría correspondiente: Una vez que sabemos a qué categoría pertenece la página se procede a almacenar esta información en la tabla *paginasclasificadas*, así se va alimentando la base de datos en el caso de que la página sea nuevamente solicitada. Punto 5 del gráfico 3.14.

3.5. Estructura del Prototipo

3.5.1. Herramientas Usadas

El prototipo se encuentra desarrollado sobre JAVA, esta conformado de dos partes básicas:

1. Clasificador: El clasificador es un programa normal en JAVA que accede a archivos y a la base de datos.

2. Buscador: Es un proyecto Web que contiene toda la funcionalidad para la construcción de perfiles y el filtrado de páginas.

Para lograr darle vida a la funcionalidad del prototipo se usaron las siguientes tecnologías:

- JAVA: jdk1.6.0.02, jre1.6.0.02.

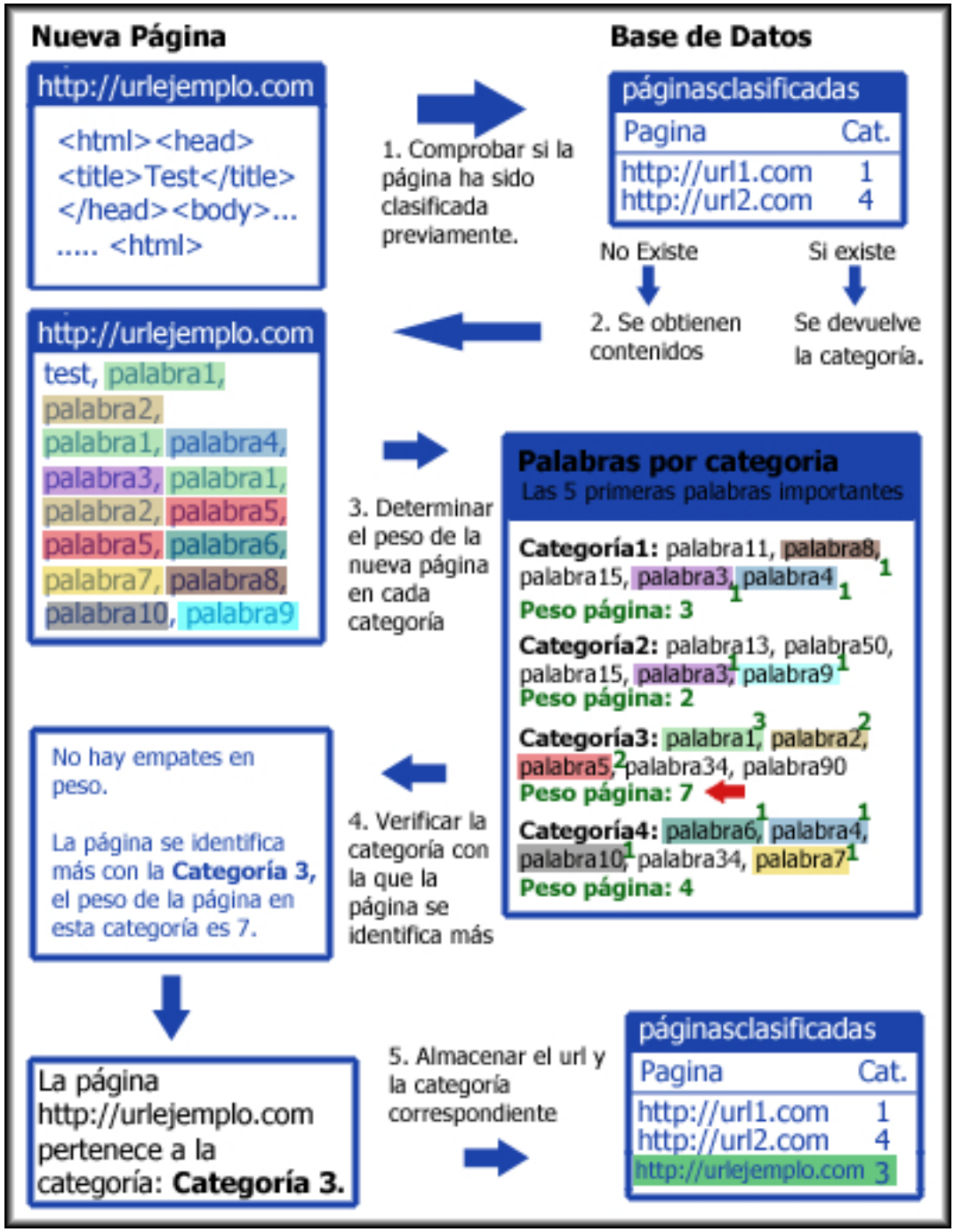

Figura 3.14: Ejemplo clasificación de una página nueva.

- Base de Datos: MySQL (Versión 5), MySQL GUI tools.
- Servidor de Aplicaciones: GlassFish V2 (Versión incluida con NetBeans).
- IDE: Netbeans (Versión 6.5.1).
- Sistema Operativo: Desarrollo sobre Windows 7 Ultimate (Versión Beta), procesamiento de información para clasificador sobre Solaris 10.

La parte del clasificador del prototipo por la demanda de recursos que requería para su procesamiento fue realizado en el Laboratorio de Computación de Alto Desempeño del Colegio de Ciencias e Ingeniería de la USFQ (18), esto facilitó el procesamiento permitiendo aumentar el número de links que fueron utilizados para que el clasificador determinara las palabras importantes para cada una de las categorías que se usaron, esto debido a las capacidades físicas de estas computadoras, las cuales se encuentran descritas en (18). Básicamente la carga de procesamiento se dividió en dos máquinas, las de más recursos del laboratorio (Valdivia y Tolita). Detalles de cómo funciona el clasificador están descritos en la sección 3.4.

3.5.2. Estructura de la Base de Datos

En la figura 3.15 se muestra un diagrama de como la base de datos se encuentra estructurada, a continuación se encuentra una descripción de lo que la figura ilustra.

Sección a: Clasificación

1. **linksodp**

   Utilizada para almacenar la información de los links del *Open Directory Project*, extraídos del XML. Ver sección 3.4.1.1.

2. **prepalabras**

   La información de los links a ser tomados en cuenta dependiendo de cuantos

links por categoría van a ser utilizados es almacenada aquí, esto porque la cantidad de links existente es demasiado grande, véase la sección 3.4.1.1 para más información.

3. palabras

Se almacenan en esta tabla todas las palabras que se han extraído correspondientes a los links de cada categoría, más información sección 3.4.1.2 (obtención de contenidos).

4. palabrascat

Aquí se almacenan las palabras sumarizadas de cada categoría e información relacionada, como sus respectivos valores de peso y TFIDF, esta tabla es usada por el clasificador. Ver secciones 3.4.1.3, 3.4.1.3 y 3.4.2.

5. categorias

Se encuentran almacenadas las categorías usadas en el prototipo con sus respectivos identificadores, ver tabla 3.1.

Sección b: Banco de datos de nuevas páginas.

9. paginasclasificadas

Esta tabla es utilizada para almacenar las páginas nuevas que van siendo clasificadas por el prototipo, así si la página es parte de algún nuevo resultado en el futuro esta no necesita ser catalogada nuevamente, ver sección 3.4.2.

Sección c: Buscador e información personalizada.

6. usuario

Guarda toda la información relacionada con el usuario, esta tabla se encuentra relacionada con otras para darle identificación a los perfiles y realizar funcionalidades como el *login*.

7. preferenciausuario

Tabla que almacena la información del **perfil fijo** del usuario (3.2.1).

8. preferenciausuarioadp

Tabla que almacena la información del **perfil adaptativo** del usuario (3.2.2).

3.5.3. Funcionamiento General

El diagrama presentado en la figura 3.16 ilustra el funcionamiento general del prototipo, los cuadros en azul muestran las diferentes opciones que están disponibles, los enlaces azules por otro lado muestran las elecciones que se pueden realizar para llegar a las opciones disponibles, los cuadros negros muestran tablas de la base de datos y los enlaces negros la extracción o inserción de información que se haga en las tablas (ver 3.5.2).

3.5.4. Presentación de resultados

En la figura 3.17 se muestra como los resultados de la búsqueda son renderizados y los componentes de navegación del prototipo. El *Modo (fijo o adaptativo)* es seleccionado en un paso previo al momento en que el usuario ingresa al sistema. El menú de resultados sirve para navegar por los distintos tipos de filtrado que tenemos en el prototipo, la primera pestaña muestra los resultados devueltos por el API sin ser modificados, la segunda pestaña muestra los datos después de que el filtrado por categorías de interés es realizado y la última ventana renderiza los resultados del refinamiento de consultas.

En el caso del refinamiento los resultados varían un poco, pues la categoría no es mostrada a continuación de la descripción de la página, esto no es necesario ya que cualquier proceso de clasificación de páginas para el refinamiento es realizado después de que los resultados son mostrados.

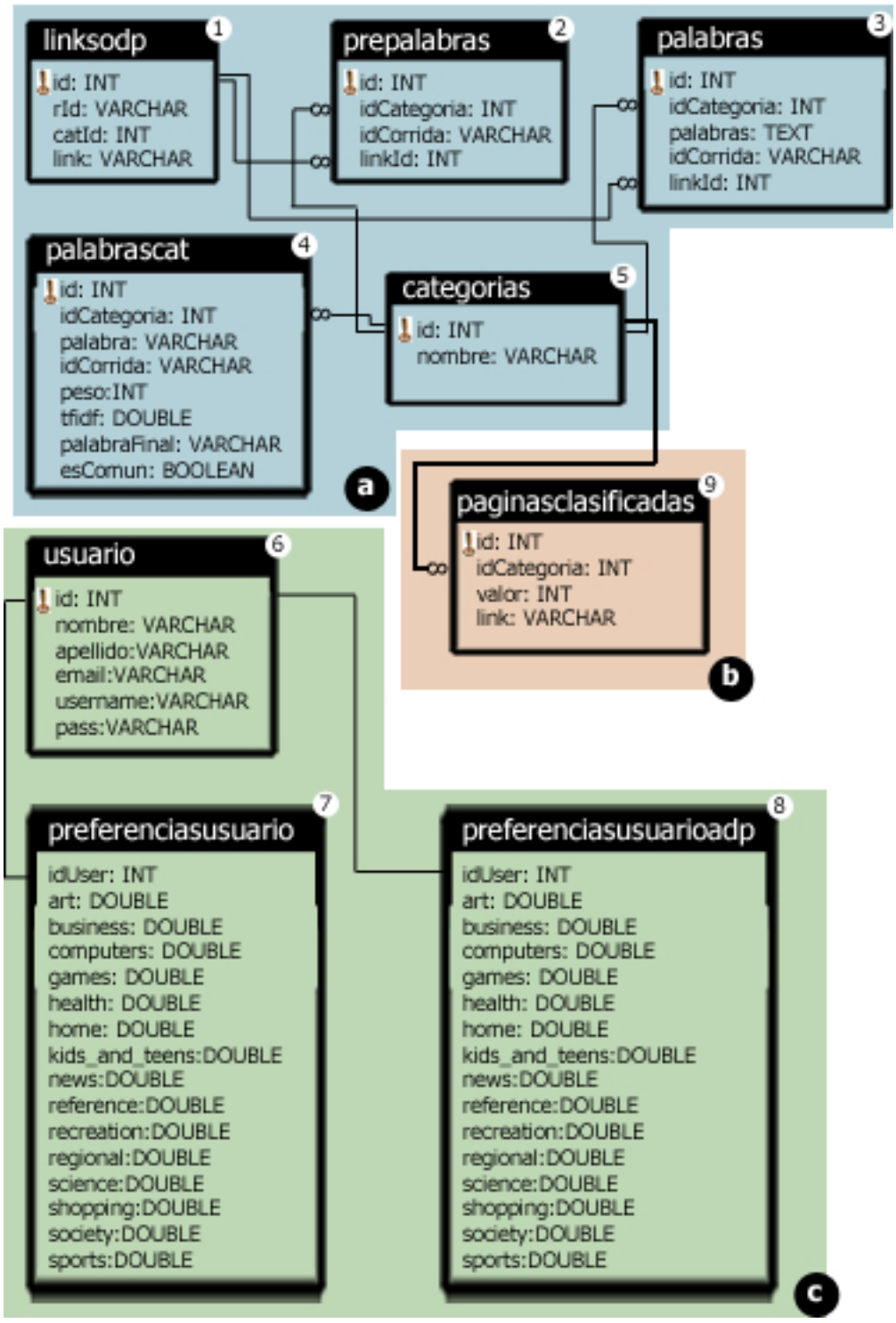

Figura 3.15: Esquema de la Base de Datos.

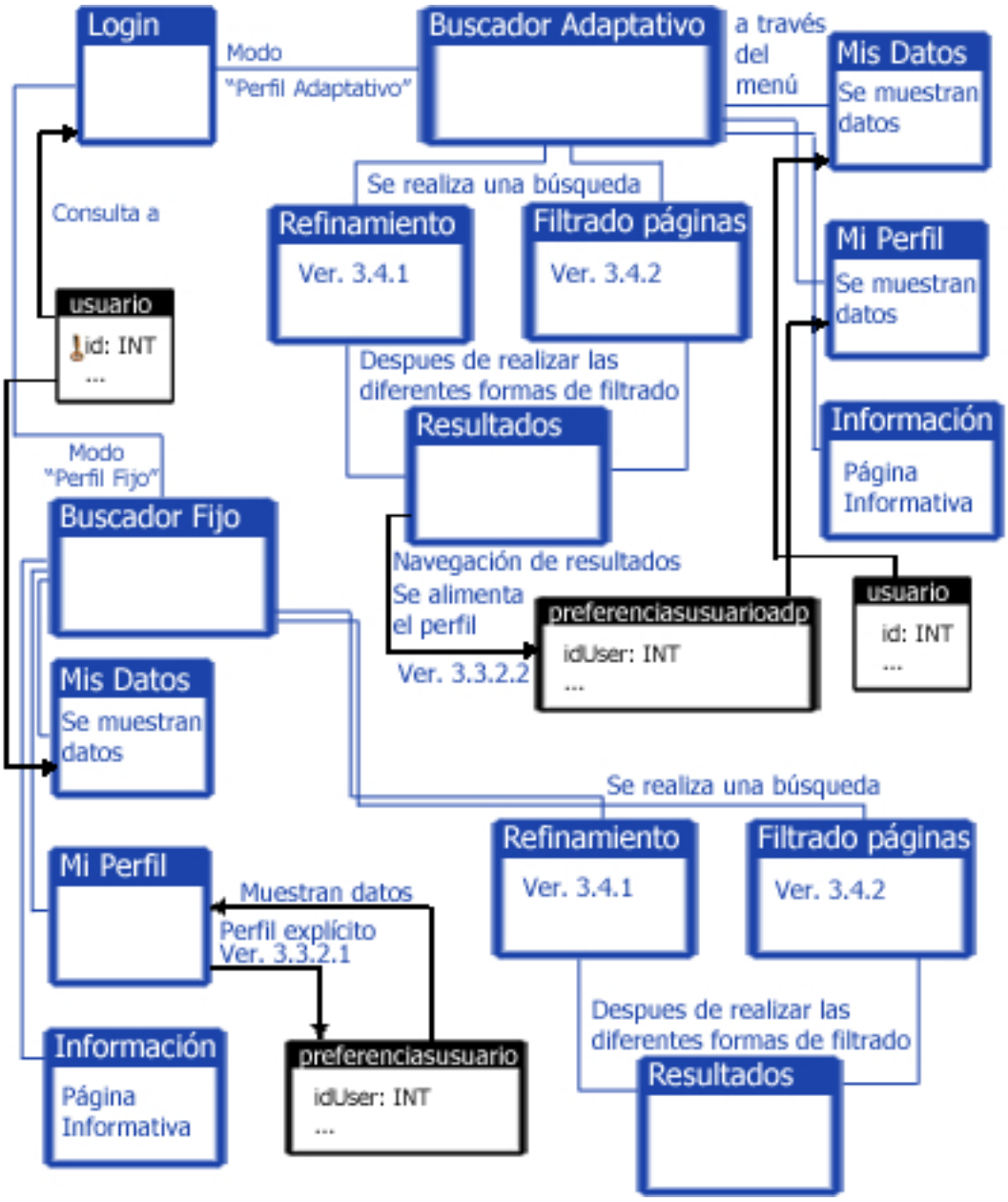

Figura 3.16: Prototipo: Funcionamiento General.

Figura 3.17: Renorno de Resultados

En la figura 3.17 podemos observar una pantalla real del prototipo que involucra todas las funcionalidades ya integradas.

Capítulo 4

Evaluación

4.1. Introducción

En este capítulo encontramos el análisis realizado sobre el prototipo de personalización, análisis que nos permite estudiar la efectividad de los resultados presentados después de usar los distintos mecanismos.

Se creó diferentes usuarios y se hizo pruebas para el filtrado de resultados y el refinamiento de consultas (y las diferentes variantes dentro de cada uno, tal como lo es el número de términos), esto para el caso del perfil fijo. Mientras que para el perfil adaptativo el análisis realizado fue de cuánto le toma al prototipo llegar a construir el perfil de cada usuario, de manera que llegue a funcionar como perfil fijo. Adicionalmente se ha realizado un estudio complementario para los mecanismos que comprende el análisis de tiempos.

4.2. Creación de Usuarios

Se han creado cuatro usuarios para este análisis, el criterio de creación ha sido el que nos ha permitido evaluar de una forma integral el prototipo, con preferencias de usuarios que abarquen las posibles combinaciones que pueden darse, y con un alcance

realizable.

Cuando se habla de las posibles combinaciones, se ha tratado de encontrar los casos generales iniciales de lo que pudiera ocurrir con un usuario. Por lo que el enfoque que se les ha otorgado a estos usuarios de prueba es más bien sobre la relación o NO relación que pudieran tener las categorías de interés entre si. Los usuarios fueron definidos como se muestra en la tabla 4.1.

No.	Nombre	Categorías Preferidas
1	Usuario 1	Computers
2	Usuario 2	Computers and Science
3	Usuario 3	Computers and Health
4	Usuario 4	Games, Arts and Home

Tabla 4.1: Preferencia de Usuarios de Prueba

Como vemos en la tabla 4.1 el *usuario1* solo tiene interés en una categoría, el *usuario 2* prefiere dos categorías que se encuentran bastante relacionadas, mientras el *usuario 3* tiene como categorías de preferencia dos que no son muy relacionadas y por último el *usuario 4* ha elegido tres categorías que no tienen relación entre ellas.

Cabe también mencionar que las preferencias de estos usuarios son categorías conocidas por mí, esto facilitó la selección de las páginas que aparecen en las distintas búsquedas.

4.3. Consultas Realizadas

Para cada usuario se definió 5 consultas a realizarse, relacionadas con las áreas de interés que fueron definidas en cada caso. Para cada consulta se hizo el estudio sobre los primeros 30 resultados mostrados. También se estudió diferentes posibilidades de número de términos clave para cada uno de los mecanismos. En la tabla 4.2 se muestra a detalle esta información.

Nombre	Consultas
Usuario 1	Laptop, algorithm, program, animation and phone
Usuario 2	Laptop, algorithm, program, computational development and research
Usuario 3	Laptop, eating, exercises, fitness table and business
Usuario 4	Laptop, console, purchase, animation and games

Tabla 4.2: Consultas Realizadas

4.4. Evaluación de Mecanismos de Personalización

En esta sección se muestran los resultados resumen obtenidos de las pruebas realizadas con el prototipo. Se ha realizado para el Perfil Fijo; análisis de efectividad del filtrado de resultados y el refinamiento de consultados. Luego en el perfil adaptativo cuánto le toma al mismo llegar a comportarse como el perfil fijo en cada caso.

4.4.1. Perfil Fijo

4.4.1.1. Filtrado de Resultados

Es esta sección se analiza la efectividad del prototipo cuando se usa el mecanismo de filtrado de resultados y el perfil fijo de usuario.

Para analizar que tan efectivo es este mecanismo, se han considerado los usuarios y consultas mencionados en las anteriores secciones, adicionalmente se ha tomado en cuenta tres diferentes posibilidades en cuanto al número de términos usados por el clasificador para catalogar nuevas páginas: 5, 10 y 15 términos.

El análisis realizado se lo ha enfocado en dos aspectos: Encontrar los falsos positivos (FP) y los falsos negativos (FN), el primer caso se refiere a todas aquellas páginas que fueron marcadas como relevantes pero no lo son, y el segundo cuando las páginas fueron enmarcadas como no relevantes y lo son.

Las figuras 4.1, muestran el porcentaje de falsos (FP y FN).

En forma de resumen el gráfico 4.2 muestran el porcentaje de efectividad obtenido

Filtrado de Resultados						
Porcentaje de Falsos Positivos y Negativos						
	5 Términos		10 Términos		15 Términos	
Usuario	FP	FN	FP	FN	FP	FN
Usuario1	0.00%	75.27%	2.38%	58.59%	3.33%	69.15%
Usuario2	4.26%	54.08%	5.08%	45.63%	0.00%	37.14%
Usuario3	14.81%	71.60%	0.00%	66.67%	0.00%	66.67%
Usuario4	4.84%	16.90%	3.39%	19.72%	5.63%	5.63%

Figura 4.1: Filtrado de Resultados - Porcentaje de Falsos Positivos y Negativos

(1 - FP).

Del cuadro y el gráfico mostrados referentes al desempeño del filtrado de resultados, podemos decir que para el enfoque de los Falsos Positivos:

1. En la mayoría de los casos las páginas mostradas después de que el filtrado ha sido aplicado, tienen un porcentaje de FP bastante bajo, ésto en cuanto a la relevancia de páginas que el usuario quiere ver. Para los usuarios 2,3 y 4 esto se cumple de gran forma, en estos tres casos el porcentaje más alto es de 14.81 por ciento.

2. A pesar de que no se muestra en este análisis es necesario mencionar que para el caso 1, en una de las consultas realizadas no existieron resultados devueltos por el filtrado. Este caso se evidenciará de mejor forma en el segundo enfoque (FN), es importante tomar en cuenta en esta parte dado que finalmente distorsiona la efectividad vista desde este punto de vista también.

3. Es muy importante para este prototipo las palabras que definen un categoría, cuando páginas que han retornado como resultados desde el motor convencional son de un dominio más general estas pueden caer dentro de la categoría(s) de interés. Así mismo cuando una consulta involucra una palabra que define fuertemente a otra categoría, los resultados que se muestran después de filtrar son pocos y pueden llegar a ser ninguno.

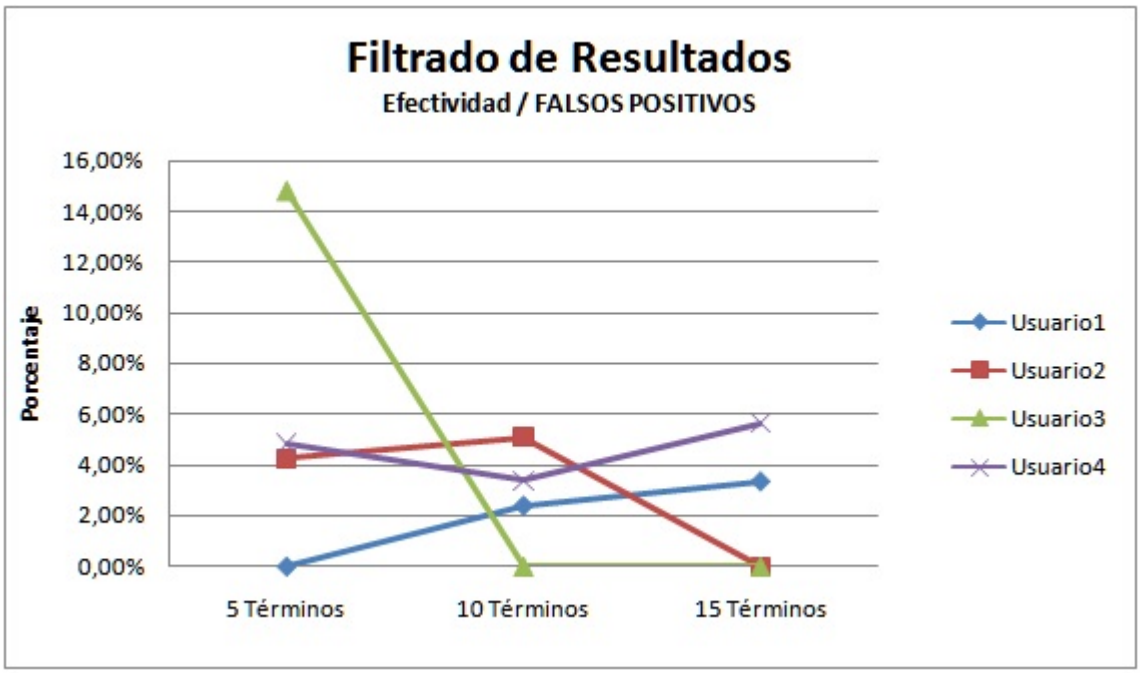

Figura 4.2: Gráfico de Efectividad para Filtrado de Información (FP)

4. En lo relacionado a la variación del número de términos, no se puede concluir un patrón que identifique un número de términos ideal a usarse en un análisis por usuario individual.

Por otro lado para el caso de los Falsos Negativos:

1. El porcentaje de aciertos dado este enfoque es considerablemente más bajo, esto puesto que el porcentaje de falsos negativos es alto, es decir, que de los resultados expresados por el motor de búsquedas convencional no se esta mostrando las páginas que el usuario realmente esta interesado en ver (hay perdida de páginas relevantes).

2. Nuevamente aquí se evidencia el tema de las páginas de dominio general (por asi llamarlo), muchos FN caen en estas categorías, entre las que más frecuentemente están en este plano tenemos: *shopping* y *business*, y en un grado menor *reference*. Un análisis más profundo deberá llevarse a cabo para determinar el peso real de esta situación en el filtrado de búsquedas.

3. Cuando una búsqueda contiene como parámetro de consulta una palabra que define a una categoría, las páginas resultado de la búsqueda mostradas tienden a ser en su mayoría páginas que el clasificador las ubica en la categoría a la que dicha palabra representa. En el estudio esto sucedió por ejemplo al hacer la consulta de *animation* en el contexto de usuarios cuyas categorías de interés no eran *arts*, puesto que *animation* es una palabra que define a la categoría *arts* el número de aciertos fue muy bajo o nulo.

4. La identificación de FN se la hizo basándose en la breve descripción que presenta el motor de búsqueda asociado con cada página.

5. Hay ciertas búsquedas que coinciden con nombres propios de instituciones lo que hace en algunos casos que el número de aciertos se reduzca en gran cantidad y que incluso se pueda volver nulo, pues si la consulta coincide con un nombre propio y dicho nombre propio se relaciona a una categoría distinta a la de interés buscada, entonces las páginas resultado que se desplieguen no se convertirán aciertos (en su gran mayoría).

Con este análisis termina la exposición de los resultados para este mecanismo, hay que mencionar que el análisis en la siguiente sección (para el refinamiento de consultas), será un poco distinto, puesto que el análisis de FN no es posible realizarlo.

4.4.1.2. Refinamiento de Consultas

En la presente sección se resume los resultados generados por el prototipo cuando el refinamiento de consultas es aplicado como mecanismo de personalización.

Como ya se había mencionado, no fue posible la consideración de los falsos negativos, esto debido a que no es posible obtener información previa al refinamiento sobre los resultados de la búsqueda. De tal manera en esta sección se analizan todos aquellos falsos positivos y el porcentaje de acierto.

Refinamiento de Consulta				
Porcentaje de Aciertos y Falsos Positivos				
	1 término	2 términos	3 términos	4 términos
Usuario	FP	FP	FP	FP
Usuario1	13.33%	29.33%	29.33%	39.33%
Usuario2	56.67%	64.67%	84.67%	84.67%
Usuario3	63.33%	68.67%	86.00%	92.67%
Usuario4	28.67%	59.33%	69.33%	78.00%

Figura 4.3: Refinamiento de Consulta - Porcentaje de Aciertos Vs. Número de Términos

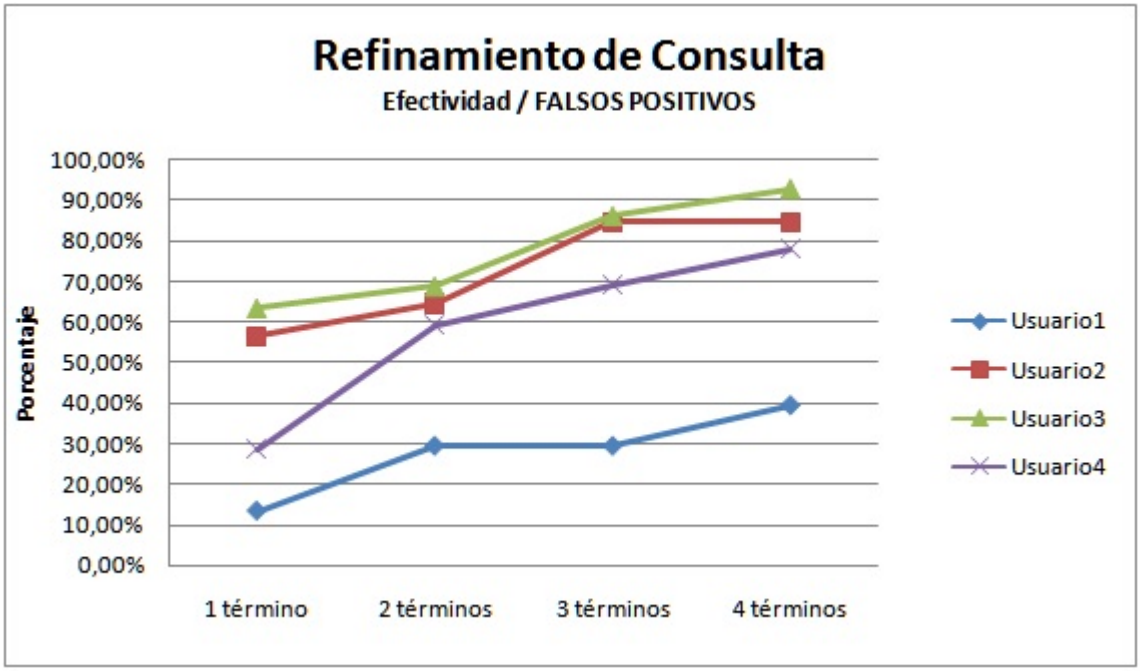

Figura 4.4: Gráfico de Efectividad de Refinamiento de Consulta (FP)

Los escenarios para el caso de refinamiento de consultas se definen por el número de términos que se añaden a la consulta original, así en este caso particular encontramos cuatro posibilidades distintas, cuando los términos añadidos van de 1 a 4. Tabla 4.3.

De los resultados detallados en la tabla 4.3 se ha realizado un gráfico que ayuda a ilustrar los ilustrar los datos, gráfica 4.4.

Del análisis de la información resumen mostrada sobre los resultados de las consultas realizadas se puede decir:

1. Sin importar cualquiera de los cuatro usuarios que se usaron para el análisis,

los resultados son más efectivos cuando el número de términos que se añaden a la misma son menores. En el caso particular cuando solamente se ha usado un término agregado a la consulta original. Es más el agregar más términos a la consulta distorsiona los resultados mostrados sobre el tema consultado.

2. La efectividad fue mayor para el primer escenario (es decir una sola categoría de interés), cuando menor es el dominio de categorías de páginas a mostrar, el refinamiento de consulta funciona de forma más eficiente (mientras menos categorías de interés existan, menor es el número de términos que acompañan a la consulta original).

3. Hay casos que han retornado resultados realmente efectivos en términos de contenidos, se han mostrado páginas que se ajustan muy bien a las categorías de interés y a la combinación de las mismas. Por ejemplo se ha dado cuando se han empleado 2 términos de refinamiento para los usuarios 2 y 3. Para el usuario 3 en este caso el número de resultados efectivos es menor para dos términos, pero las páginas mostradas muestran contenidos más ricos relacionados con la consulta y las categorías.

4. Una vez más, tal y como se registra en el caso de personalización por filtrado de resultados, las palabras que definen cada categoría son sumamente importantes. Los resultados se han visto un poco desviados cuando las palabras que definen las categorías en este prototipo son de un dominio general, por ejemplo esto ha pasado en nuestro estudio para las categorías science y health.

4.4.2. Perfil Adaptativo

En esta sección se muestran los resultados en lo que se refiere al prototipo de personalización cuando el perfil adaptativo es usado. Ya que como se ha mencionado antes en este trabajo, se ha separado el estudio de cada mecanismo de personalización

Perfil Adaptativo				
Número de selecciones o clicks para llegar al comportamiento de perfil fijo.				
	Umbral 0.2	Umbral 0.4	Umbral 0.6	Umbral 0.8
Paso Incremental:	0.2	0.2	0.2	0.2
Clicks:	13	9	3	2
Paso Incremental:	0.1	0.1	0.1	0.1
Clicks:	21	18	11	5
Paso Incremental:	0.01	0.01	0.01	0.01
Clicks:	199	132	91	25

Figura 4.5: Usuario1: Perfil Adaptativo, Número de clicks para ajuste. (Computers)

para cada perfil. En esta sección solamente se muestra cuanto le cuesta al prototipo llegar a adaptarse para que su comportamiento sea similar al caso del perfil fijo.

Aquí se analiza cuantos clicks el usuario da en páginas de su preferencia, de forma que, el prototipo llegue a comportarse (en términos de filtrado y refinamiento de información) de la misma manera que lo ha hecho para el perfil fijo.

Para el caso del perfil adaptativo se consideran dos variables; el *paso incremental* y *el umbral*.

De manera que para cada una de las variables los valores son los siguientes:

Umbral: 1:0.2 / 2:0.4 / 3:0.6 / 4:0.8

Paso Incremental: 1: 0.2 / 2:0.1 / 3: 0.01

Recordemos que el umbral determina el límite para seleccionar una categoría como preferida, mientras que por otro lado el paso incremental, es el peso que se le da a la categoría a la que pertenece una página cuando se da click sobre ella. Así pues estos dos parámetros junto con la normalización, son los que le dan al perfil adaptativo la velocidad y precisión de aprendizaje.

Entonces en este estudio tenemos 12 diferentes niveles de análisis (4 umbrales y 3 pasos incrementales). Los gráficos 4.5, 4.6, 4.7 y 4.8, Muestras los resultados derivados de las pruebas.

De los resultados mostrados en las figuras de clicks vs paso incremental vs umbral,

Perfil Adaptativo				
Número de selecciones o clicks para llegar al comportamiento de perfil fijo.				
	Umbral 0.2	Umbral 0.4	Umbral 0.6	Umbral 0.8
Paso Incremental:	0.2	0.2	0.2	0.2
Clicks:	17	6	4	3
Paso Incremental:	0.1	0.1	0.1	0.1
Clicks:	25	13	8	4
Paso Incremental:	0.01	0.01	0.01	0.01
Clicks:	231	156	86	21

Figura 4.6: Usuario2: Perfil Adaptativo, Número de clicks para ajuste.(Computers and Science)

Perfil Adaptativo				
Número de selecciones o clicks para llegar al comportamiento de perfil fijo.				
	Umbral 0.2	Umbral 0.4	Umbral 0.6	Umbral 0.8
Paso Incremental:	0.2	0.2	0.2	0.2
Clicks:	11	7	5	3
Paso Incremental:	0.1	0.1	0.1	0.1
Clicks:	20	13	8	4
Paso Incremental:	0.01	0.01	0.01	0.01
Clicks:	176	110	79	21

Figura 4.7: Usuario3: Perfil Adaptativo, Número de clicks para ajuste.(Computers and Health)

Perfil Adaptativo				
Número de selecciones o clicks para llegar al comportamiento de perfil fijo.				
	Umbral 0.2	Umbral 0.4	Umbral 0.6	Umbral 0.8
Paso Incremental:	0.2	0.2	0.2	0.2
Clicks:	12	8	6	4
Paso Incremental:	0.1	0.1	0.1	0.1
Clicks:	22	14	10	8
Paso Incremental:	0.01	0.01	0.01	0.01
Clicks:	158	96	65	43

Figura 4.8: Usuario4: Perfil Adaptativo, Número de clicks para ajuste.(Games, Arts and Home)

se puede determinar que independientemente del umbral, mientras más pequeño es el paso incremental más demora el perfil en llegar a considerar una categoría como preferida, o en el caso contrario de sacar una categoría de las categorías preferidas.

Hay un tendencia lógica marcada dentro del análisis para un perfil, la cual nos indica que mientras el umbral es menor (lo que significa que es más exigente) mayor es el número de clicks necesario necesario para construir el perfil, esto sin importar el paso incremental que estemos usando.

Umbrales menores, más exigentes determinan o aseguran la calidad de las páginas mostradas acorde a las preferencias, pero decrementan la rapidez con la que el perfil adaptativo aprende. Umbrales más altos aceleran el proceso de aprendizaje pero así mismo si el usuario ingresa a páginas que no corresponden a sus preferencias por equivocación, el riesgo de que los resultados mostrados no sean los deseados es más grande.

Otra determinación resultado de este análisis corresponde a que no se ha encontrado una correlación incremental del número de clicks dado para los perfiles que tiene más categorías de interés. La lógica quizás indicaría que mientras más categorías sean de interés para el usuario más clicks tendrá que darse para que el perfil adaptativo logre aprender, sin embargo, eso no es lo que se ha evidenciado en este estudio, pues el ejercicio realizado no da ningún indicio de que esto ha ocurrido, así por ejemplo el usuario cuatro en el caso mas crítico (paso incremental 0.01 y umbral 0.2) aprende en menos clicks que cualquiera de los otros casos a pesar de tener tres categorías no relacionadas como preferencias.

Derivado de lo mencionado en el párrafo anterior podemos también decir, que si bien no se encuentra una relación númerica de ningún tipo aparente entre la cantidad de clicks dados versus el número de categorías elegidas como preferidas por el usuario (relacionadas o no), la rapidez con la que el perfil adaptativo aprende depende también de la experiencia que el usuario tenga como buscador en las áreas de interés. Es decir

que si un usuario esta explorando una nueva área de interes es muy probable que el perfil adaptativo demore más en aprender.

Otro factor sumamente importante mencionado ya en otras secciones es el clasificador, si el clasificador no cataloga las páginas de una manera correcta el perfil no podrá aprender, pues el usuario va a elegir páginas que entrarían en otras categorías, y alejarían el perfil adaptativo de la realidad.

Por otro lado el clasificador debe ser lo suficientemente efectivo como para poder determinar que hacer cuando una página cae en varias categorías, en este prototipo el clasificador no tiene esa característica. Sin embargo por lo evidenciado en este estudio éste es un aspecto fundamental a considerar, pues pienso que definitivamente hay páginas cuyo domino general nos obliga a catalogarlas dentro de varias categorías, es posible afirmar esto basándome en la forma en que el clasificador fue implementado en este prototipo. En el capítulo de conclusiones se sugieren posibles soluciones para poder mejorar este aspecto.

Las variables que conforman la construcción del perfil adaptativo para el prototipo son ingresadas y modificadas a través de un archivo de texto, lo cual debe ser considerado, ya que debería el usuario de quererlo tener la opción de configurar a través de interfaces amigables las variables con las cuales se construye su perfil, o quizás otra opción es una autodetección del sistema adaptativo de cambios bruscos del perfil por las selecciones que el usuario realiza,con esta detección se podría cambiar las variables automáticamente para evitar esto o se podría alertar al usuario.

4.5. Medición de tiempos

En esta sección el tema a tratar es los tiempos en los cuales el prototipo ejecutó las acciones principales que son pieza fundamental de este trabajo de tesis, así pues se han incluido segmentos de código en el prototipo que permiten medir los tiempos en los que se completaron las diferentes etapas y mecanismos.

Busqueda (Google o Bing)	
Busqueda	**Tiempo (ms)**
PROMEDIO (MS)	**4407.75**
PROMEDIO (S)	**4.41**
Mostrando 10 páginas	

Figura 4.9: Tiempos, Google o Bing 10 páginas

Como antecedente a la presentación de los tiempos es necesario especificar cuales son las características físicas del computador donde estos datos fueron tomados y almacenados. En la tabla 4.3 se describen las características más importantes.

Característica	Valor
Procesador	Intel(R) Core(TM)2 CPU T5600 1.83 GHz
RAM	2 GB
HDD	150 GB
Sistema Operativo	Windows 7 de 32 bits.
Conexion a internet	512 Kbps

Tabla 4.3: Características de computador de toma de tiempos

Ahora bien, en lo que respecta a la búsqueda como tal, se ha realizado la medición del tiempo en el cual concluye una búsqueda normal a través del prototipo usando el API de Google o Bing, la figura 4.9 ilustra esta información. Esta medición se la ha hecho para tener un parámetro de comparación entre una búsqueda sin aplicar ni filtrado y refinamiento.

La figura 4.10 muestra los tiempos ejemplo requeridos y su promedio, en la cual el prototipo concluye búsquedas al usar el mecanismo de refinamiento de consulta, específicamente cuando se han desplegado 10 páginas a la persona que esta realizando la búsqueda.

Por otro lado la gráfica 4.11 muestra un resumen de los tiempos promedio necesarios para completar las búsquedas cuando se usa el mecanismo de filtrado de resultados, por

Refinamiento	
No. Busqueda	**Tiempo (ms)**
PROMEDIO (MS)	**8447.00**
PROMEDIO (S)	**8.45**
Mostrando 10 páginas	

Figura 4.10: Tiempos, Refinamiento de Consulta 10 páginas

Filtrado			
Busqueda	**5 Términos**	**10 Términos**	**15 Términos**
Promedios * Paginas Mostradas (ms)	389674.00	367791.14	435302.48
Promedios * Paginas Mostradas (s)	389.67	367.79	435.30
Promedios * Paginas Mostradas (min)	6.49	6.13	7.26
Páginas Mostradas	10	10	10

Figura 4.11: Tiempos, Filtrado de Información 10 páginas

favor refiérase a la sección 3.3.1 para los detalles con lo cuales se realiza este proceso, estos tiempos han sido analizados y almacenados de igual manera para cuando 10 páginas le son mostradas al usuario y considerando cuando se utilizan cinco, diez y quince términos de clasificación.

Por último para concluir con la presentación de la información de tiempos, se presenta en la figura 4.12 un análisis resumen de los tiempos para cada uno de los mecanismos de búsqueda, así pues queda claramente evidenciado en este gráfico que en términos de tiempo y bajo las condiciones de este estudio:

1. La diferencia de tiempos entre la búsqueda normal de Google o Bing, y el refinamiento de consultas es aproximadamente del doble, sin embargo no es una referencia tan significativa y perceptible ante el usuario dado que se le estarían mostrando la páginas de interés.

2. Los tiempos entre el refinamiento de consultas y el filtrado de información (para cualquiera de los tres casos) varían significativamente, desde el punto de vista del usuario es ciertamente bastante mas largo esperar por que los resultados de su

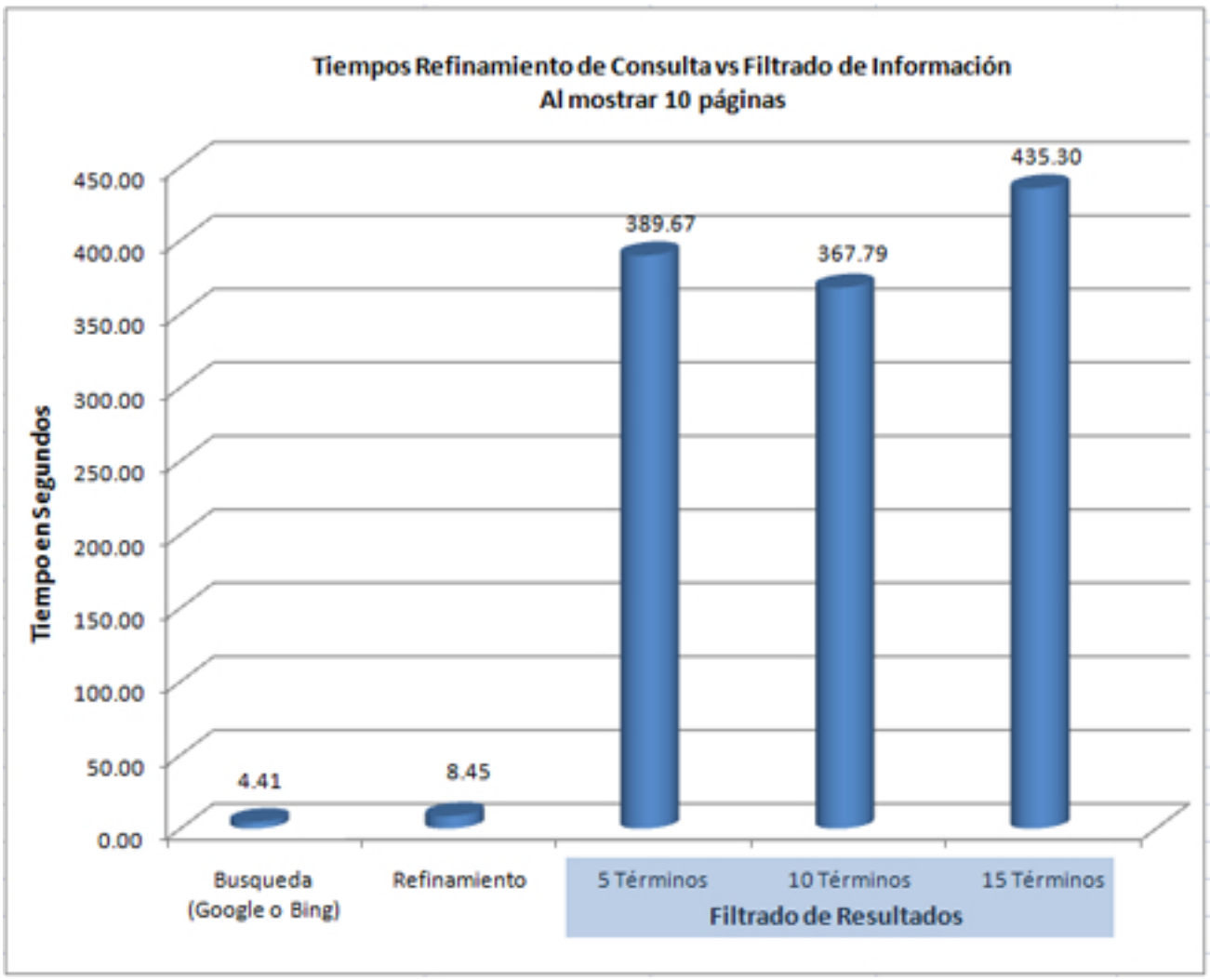

Figura 4.12: Resumen de Tiempos

búsqueda se desplieguen cuando se usa el filtrado de información.

3. Las diferencias de tiempos para el mecanismo de filtrado de información cuando se usan los distintos números de términos no es mayormente significativa para cuando se usan 5 y 10 términos, pero se vuelve un tema mas grave cuando usamos 15 términos de clasificación.

4.6. Análisis de Efectividad

En esta sección se analiza comparativamente los resultados presentados en cada una de las secciones en términos de efectividad.

4.6.1. Perfil Adaptativo vs Perfil Fijo

Para el perfil fijo podemos decir que es muy rápida la forma en el cual se construye y se aplica, pero siempre requerirá de la interacción explícita del usuario para su creación y mantenimiento, además el usuario deberá tener pleno conocimiento de las categorías en las cuales los contenidos que busca pueden estar enmarcados, la falta de conocimiento o de experiencia del usuario como buscador pueden derivar en serios problemas para el uso de este tipo de perfil.

Por otro lado el perfil adaptativo tiene la ventaja de que no requiere la participación explícita del usuario, es cambiante de acuerdo a las preferencias del usuario buscador, las cuales a su vez son cambiantes dependiendo de las circunstancias y de su entorno. El hecho de que tenga la capacidad de aprender de las selecciones del usuario es una fortaleza para este tipo de perfil.

Los aspectos que hay que cuidar en el caso del perfil adaptativo son los parámetros con los cuales se considera que un usuario ha seleccionado una página y le ha parecido interesante, así también los parámetros con los cuales el perfil define que una categoría es interesante y la velocidad con la que aprende. Quizás sería conveniente probar con darle al usuario la opción de manejar dichos parámetros en caso de que lo considere necesario.

Adicionalmente hay que decir que es más pesado en cuanto al uso de recursos el perfil adaptativo sobre el perfil fijo.

4.6.2. Refinamiento de Consultas versus Filtrado de Resultados

Para el análisis comparativo respectivo se presenta la tabla 4.4, el cual recoge la información final de los mecanismos, dadas las selecciones de número de términos a usar que les dan el mejor funcionamiento.

Usuario	FR(1-FP)	FR(1-FN)	RC(1-FP)	Característica de Categoría(S)
1	96.67	30.85	86.67	Una sola
2	100	62.86	43.33	Dos relacionadas
3	100	33.33	36.67	Dos sin mayor relación
4	94.37	94.37	71.33	Tres aleatorias

Tabla 4.4: Filtrado de Resultados - FR (15 términos) vs Refinamiento de Consulta - RC (1 término)

De lo mostrado en la tabla 4.4 tenemos dos escenarios de análisis, lo más lógico sería la comparación del Falsos Positivos de los dos mecanismos (puesto que es la medida mutua), aquí observando los datos de la tabla se llega a determinar que el FR es bastante más efectivo que el RC, ésto independientemente del perfil de usuario usado para el estudio y considerando los mejores casos de variables para cada mecanismo.

Sin embargo, el análisis de FN para FR versus el RC y los FP obtenidos es necesario hacerlo, puesto que aquí se evidencia la pérdida de páginas, que está tomando lugar para el mecanismo de filtrado de resultados. Entonces comparando los porcentajes de acierto en este caso vemos que para dos de los usuarios el refinamiento de consultas es más efectivo (usuarios 1 y 3). En este caso claramente se observa un mejor funcionamiento cuando solamente existe una categoría de interés.

Sin embargo es necesario ir más profundo cuando se analiza esta información, así pues a la tabla 4.4 se le ha agregado una columna que indica la característica (s) de la (s) categorías que conforman el perfil de cada usuario. Considerando esta columna mencionada en la párrafo anterior podemos decir que para este estudio el

refinamiento de consultas resultó ser mas efectivo cuando en el perfil existe una sola categoría de interés y cuando hay dos categorías no relacionadas. Esto sucedió puesto que las palabras clave que definen la categoría son de un dominio más específico entorno a las búsquedas realizadas, la tabla 4.5 muestra un resumen de las palabras clave por categoría de los perfiles usados en las pruebas, así pues las palabras que definen computer y health son más pesadas para el refinamiento que las palabras que definen art, games y home. Estas últimas tienen un dominio más general al ser enviadas con la consulta original al buscador, esa es la razón por la cual cae en desventaja ante el filtrado de resultados.

Usuario	Categorías	Palabra(s)
1	Computers	computer, system, informacion
2	Computers and Science	S:plant, language, product
3	Computers and Health	H: alcohol, health, drug
4	Arts, Games and Home	A:page, animation, site G:game, player, play, H:review, recipe, car,

Tabla 4.5: Resumen Usuarios, Categorías y Palabras Clave

Por otro lado en el caso del filtrado de información tenemos que a pesar de que las palabras que definen las categorías sean de dominio más general, este mecanismo es mas efectivo puesto que usa una mayor cantidad de palabras clave para definir la categoría a la que pertenece la página, es decir reacciona mejor ante la problemática de las categorías de dominio general.

4.6.3. Un poco de Eficiencia

A pesar de que no esta contemplado en el alcance de este trabajo, es importante dar una reseña de que tan diferentes son los mecanismos de personalización en cuanto al

uso de recursos, esto esta claramente evidenciado en la sección 4.5, donde se muestran los tiempos de ejecución de las etapas y casos.

Entonces dada la información presentada en dicha sección se concluye que en uso de recursos el refinamiento de consultas es bastante más eficiente que el filtrado de información, para que en este contexto puedan ser mecanismos que compitan a la par, sería necesario hacer una implementación física mas adecuada dados los requerimientos para el filtrado, o quizás considerar el uso de alguna otra metodología para extraer la información de las categorías a las que pertenecen las paginas.

Capítulo 5

Conclusiones

5.1. Conclusiones

En este trabajo se ha realizado un estudio que nos ayude a determinar las fortalezas y debilidades de cada uno de los mecanismos de personalización de búsquedas, así dadas las condiciones y particularidades de este estudio, se ha encontrado que los dos mecanismos son capaces de mostrar un alto número de páginas relevantes para el usuario dependiendo del caso y tenemos también un número de aciertos relativamente bajo en otros.

Sin embargo al realizar una comparación de los aciertos en los dos mecanismos se puede evidenciar que el filtrado de información puede atender por la forma en que analiza y pesa los contenidos de las páginas, casos en los cuales probablemente es más complejo encontrar una página relevante con el RC, esto básicamente es cuando las categorías de interés son de dominio general, la sección 4.6 muestra el análisis completo.

Como esta mencionado en el párrafo anterior se ha evidenciado durante la ejecución de este estudio los problemas de catalogación de una página, hay varios puntos a considerar que pueden mejorar este aspecto, a pesar de que no es del alcance de esta tesis el análisis e implementación de un clasificador, este prototipo usa un clasificador propio desarrollado para este trabajo y por lo tanto podemos mencionar los aspectos en los cuales se puede optimizar para obtener mejores resultados al momento de aplicar

los mecanismos de personalización, de tal manera:

1. La cantidad de información usada para encontrar palabras clave puede crecer, mejorando la implementación del clasificador e instalándolo en un hardware mucho más poderoso.

2. Debe tener un trabajo constante de actualización y aprendizaje de las palabras claves, realizando catalogaciones comprobatorias y alimentándose de nuevas páginas periódicamente.

3. Existen páginas que de hecho pertenecen a varias categorías, son de dominio general, las cuales deberían ser tratadas de diferente forma. Se evidenció que en algunos de los casos si las páginas hubieran pertenecido a varias categorías el número de aciertos en el caso de filtrado de resultados (Falsos Negativos) hubiese sido más alto. Entonces si sería interesante en otras etapas de estudio implementar un sistema con el cual una página dependiendo de los pesos obtenidos pueda pertenecer a varias categorías o quizás trabajar con diferentes pesos de palabras para clasificación.

4. Al encontrar las palabras clave por categoría vemos también una razón para desviar los resultados mostrados, esto sucede cuando palabras de dominio general forman las que definen una categoría, habría que trabajar en la forma en cómo identificar y no considerar estas palabras o darles menos peso al momento de usarlas para pesar una pagina. Esto básicamente fue evidenciado cuando el RC fue usado.

Para mejorar estos aspectos se puede trabajar en una segunda etapa o quizás combinar con otros proyectos para construir un clasificador efectivo y eficiente, la cantidad de información que se maneja con este propósito es bastante grande.

En lo que respecta a los perfiles se ha expuesto la evidencias de las funcionalidades en la sección 4.6, sobre el perfil adaptativo es también importante mencionar que el

trabajo del clasificador es esencial, un buen clasificador ayudará a construir un perfil de forma más efectiva, para esto hay que considerar los factores que se han mencionado ya en esta sección.

Adicionalmente se puede pensar en darle al usuario las opciones para que de forma amigable pueda manejar las variables de adaptación, esto si el considera que el perfil aprende muy rápido o muy lento para su gusto. Puede también pensarse que la implementación de una funcionalidad que permita de forma automática al sistema saber si esta aprendiendo muy rápido, muy lento o si hay cambios dramáticos en el perfil y realizar un autoajuste de las variables de adaptación.

Así mismo otro aspecto a considerar es que se puede mejorar en cuanto al aprendizaje del perfil, es definir de otra manera que es lo que es relevante para el usuario, pues para el caso de este prototipo el simple hecho de que de un click ya hace a la página relevante, un nuevo y mejorado mecanismo sería agregar una temporalización para saber cuanto tiempo navegó por la misma, u otras metodologías que se consideren interesante de implementar y probar.

En el caso de filtrado de información se ha evidenciado que los resultados de las búsquedas pueden ser más o menos efectivos dependiendo del número de términos empleados para la clasificación de las páginas (para este estudio 5, 10 y 15 términos), como se analizó en 4.6, si se seleccionara el mejor caso las relevantes mostradas pudieran aumentar significativamente, esto el prototipo no lo hace por si solo y se llegó a esta conclusión después de analizar los resultados entregados en cada uno de los casos. Por lo cual sería interesante plantear la posibilidad de añadir esta funcionalidad al sistema, claro involucra el análisis de los recursos necesarios y que tan viable es hacerlo.

Por otro lado dada la manera en que trabajar el mecanismo de filtrado de información, al analizar los contenidos de la página y pesar las palabras, puede ser usado también como una herramienta de protección de privacidad y contenidos muy efectiva, por ejemplo páginas con contenidos pornográficos, violentos, etc.

Dada la efectividad mostrada por cada mecanismo y expuesto el hecho de que cada uno de ellos tiene sus ventajas sobre el otro en diferentes casos, nos da a pensar que quizás un sistema combinado puede darnos resultandos significativamente mejores, así pues se puede pensar para el futuro hacer un sistema que sea un híbrido entre los dos mecanismos, como por ejemplo que primero se aplique el RC y luego sobre los resultados entregados el FI, considero que un estudio de este tipo podría dar resultados importantes para el avance de este estudio.

Este sistema prototipo ha sido creado con el objetivo de plantear una base para el estudio de los mecanismos de personalización de búsquedas. Pruebas a mayor escala, mejoras de implementación, algoritmos más poderosos, uso de sistemas o información ya implementados, mejor uso de recursos, uso de recursos más poderosos, pueden llegar a convertir a este prototipo y estudio en una gran herramienta de análisis de diferentes motores de búsqueda, clasificadores y mecanismos de personalización. Es necesario mencionar que el hecho de que una mayor cantidad de personas se involucren en este trabajo puede hacerlo crecer de forma considerable, claro sería importante sumarle la posibilidad de que en un futuro se pueda considerar a esto como un proyecto universitario.

El prototipo de esta tesis ha sido utilizado ya en dos semestres consecutivos de la case de Data Minning de la USFQ para estudios de proyecto final, este es un referente importante para el trabajo y los resultados obtenidos podrían ser considerados para ampliar el alcance, y trabajar en las áreas donde se encuentre problemática o nuevos casos de estudio.

Referencias

[1] Jian-Tao Sun adn et all. Cubesvd: A novel approach to personalized web search. *IW3C2*, 1(1):382 – 390, May 2005. 3, 6, 7

[2] Bing - Microsoft. Using the API Version 2 Beta with Java and the APIs XML interface. `http://www.bing.com/toolbox/blogs/developer/archive/2009/05/28/using-the-api-version-2-0-beta-with-java-and-the-api-s-xml-interface.aspx`, 2009 - accedido el 10/08/2009. 12, 13

[3] Bing - Microsoft. Bing API Version 2 Basics. `http://www.bing.com/developers/s/APIBasics.pdf`, accedido 30/01/2011. 13

[4] Bing - Microsoft. Bing results by categories. `http://searchengineland.com/meet-bing-microsofts-new-search-engine-20093`, accedido el 30/01/2011. 2

[5] dmoz. Open Directory Project. `http://www.dmoz.org/`, 1998-2009 / accedido 30/01/2010. 24

[6] Google. Google SOAP Search API Reference. `http://code.google.com/apis/soapsearch/reference.html`, accedido el 10/08/2009. 12

[7] Google. Google Advanced Web Search. `http://www.google.com.ec/advanced_search?hl=es`, accedido el 30/01/2011. 2

[8] Google. Google Directory - Seaching By Categories. `http://www.google.com/dirhp`, accedido el 30/01/2011. 2

[9] Google. Personilized Web Search of google based on web history search. `http://www.google.com/psearch`, accedido el 30/01/2011. 3

[10] Jiawei Han and Michele Kamber. *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers, second edition, 2006. 9, 10, 11

[11] Kenji Hatano Kazunari Sugiyama and Masatoshi Yoshikawa. Adaptative web search based on user profile constructed without any effort from users. *ACM Org*, 16(1):675–684, May 2004. 3, 6, 7

[12] Raymond Kosala and Hendrik Blockeel. Web mining research: a survey. *ACM SIGKDD Explorations Newsletter*, 2(1):1–15, June 2000. 3, 7

[13] Fang Liu, Clement Yu, and Weiyi Meng. Personalized web search for improving retrieval effectiveness. *IEEE Transactions on Knowledge and Data Engineering*, 16(1):28–40, January 2004. 6, 7

[14] Jacob Palme. Information filtering. *Department of Computer and Systems Sciences, Stockholm University/KTH*, 2(1):–, July 1998. 3, 7, 8

[15] Porter. An algorithm for suffix stripping. `http://tartarus.org/~martin/PorterStemmer/java.txt`, 1980 - accedido 30/01/2011. 30

[16] B. Mobasher R. Cooley and J. Srivastava. Web mining: Information and pattern discovery on the world wide web. *Department of Computer Science and Engineering, University of Minnesota*, 1(1):588, January 1997. 3

[17] y Toru Ishida Satoshi Oyama, Takashi Kokubo. Domain-specific web search with keyword spices. *IEEE Transactions on Knowledge and Data Engineering*, 16(1):17–27, January 2004. 3, 7, 8

[18] Universidad San Francisco de Quito. Laboratorio - COMPUTACIÓN DE ALTO DESEMPEÑO. `http://latolita.usfq.edu.ec/`, accedido 30/01/2010. 40

[19] University of Waikato, Hamilton, New Zealand. Stop words algorithm. `http://www.java2s.com/Open-Source/Java-Document/Science/weka/weka/core/Stopwords.java.htm`, 2001 - accedido 30/01/2011. 30

[20] Wikipedia. Term Frequency Inverse Document Frequency. `http://en.wikipedia.org/wiki/Tf-idf`, accedido 30/01/2010. 11

[21] Yahoo. Yahoo Directory - Seaching By Categories. `http://dir.yahoo.com/`, accedido el 30/01/2011. 2

Capítulo 6

Anexos

6.1. Manual de Usuario

Manual de Usuario

Introducción

El obtetivo de este manual es proveer de todos los pasos para instalar el prototipo de esta tesis en un sistema nuevo y que la misma funcione de forma independiente. Las instrucciones en este caso estan descritas para plataforma Windows, sin embargo el prototipo puede perfectamente trabajar sobre plataformas basadas en Unix. A continuación se presenta la información necesaria para instalar el prototipo.

Nota: Como anexo a esta tesis se deja en digital toda la información necesaria para que el prototipo pueda ser instalado, así mismo el código fuente, proyecto Netbeans, éste documento y demás material de apoyo.

Herramientas necesarias

Los siguientes programas listados son necesarios, existen otras herramientas que pueden ser usadas también para hacer funcionar el proyecto (dependerá de las preferencias de la persona que esté instalando), en esta guía se pretende mostrar como instalar el prototipo procurando que opciones de debuggeo y de visualización de código fuente esten habilitadas, las herramientas presentadas son las que fueron usadas en el desarrollo del prototipo:

1. Plataforma Java, *http://www.sun.com/software/index.jsp?cat=Java %20 Technologies&tab=3& subcat=Java %20Products.*

2. Netbeans IDE (paquete de descarga con perfil Java): Con servidor de aplicaciones incluido (GlassFish), en caso de querer bajar una version más reciente que la proporcionada en el DVD por favor diríjase a *http://www.netbeans.org/downloads/.*

3. MySql: Instalacion del manejador de la base de datos y de las herramientas de administración gráficas (GUI Tools). Para versiones más recientes *http://dev.mysql.com/downloads*

4. Conector de MySql para Java.

5. Imagen de la Base de Datos del prototipo: Aquí se incluye información tanto de la construcción del clasificador (fase primera del proyecto), como del buscador que implementa los diferentes mecanismos de filtrado y perfiles.

6. Proyecto *Web* del prototipo.

Toda esta información se encuentra en el DVD de instalación.

Instalación del Ambiente de Trabajo

1. Instalación de la plataforma java: Utilizar el ejecutable proporcionado en el DVD (dentro del directorio *Java Jdk 6*) con la version 6 de la plataforma Java SE, para mayor documentación sobre esto refiérase a *http://www.sun.com/documentation/javase/index.jsp*.

2. Instalación de Netbeans IDE: Usar el instalador que esta proporcionado en el DVD (directorio *Netbeans*). Para instrucciones de instalación de Netbeans vea *http://www.netbeans.org/community/releases/36/install.html*. Puesto que esta version de Netbeans tiene el servidor de aplicaciones (GlassFish) incluido, es necesario tomar en cuenta la configuración con la que éste va a funcionar, es decir, puertos, nombre de usuario y clave de administración.

3. MySql: Usar los ejecutables entregados para la instalación del MySql Server y de las herramientas de administracipón gráficas (MySQL GUI tools), estos se encuentran en el directorio *MySql 5* , mayor información en *http://dev.mysql.com/doc/refman/5.1/en/installing.html*. Es importante tomar en cuenta la configuración del servidor puesto que esta debe ser especificada en el archivo de configuración

del prototipo de forma que éste pueda conectar a la base de datos, la información clave es el *host* donde esta corriendo el servidor y la clave de *root*.

4. Conector de MySql para Java: Ubicar el archivo *.jar* del conector de la base de datos en una ubicación específica del computador (el archivo se encuentra en el directorio *Librerias Jar y Conector*) , luego apuntarlo desde la variable de entorno *CLASSPATH*, en el caso específico de este manual el archivo se ecuentra ubicado en *C:/mysql-connector-java-3.1.13-bin.jar*.

Instalación del Prototipo

1. Copiar la carpeta del proyecto *PsearchApp* en una ubicación específica del computador (carpeta que se encuentra en la raíz del DVD).

2. Copiar los archivos *googleapi.jar* y *xerces.jar* (ubicados en *Librerias Jar y Conector*) en el sistema de archivos de la máquina. Estos son archivos (librerias) indispensables para la compilación y funcionamiento del prototipo.

3. Copiar el archivo de configuraciones: El archivo *configuracionPsearchDEFAULT.txt* ubicado la raíz del DVD contiene las principales variables del programa, entre estas estan por ejemplo, el numero de palabras para clasificar páginas en el filtrado de las mismas, así como también el número de palabras utilizado en el refinamiento de consultas, entre otras. El archivo debera ser ubicado en una locación de su preferencia y renombrado a *configuracionPsearch.txt*.

4. Cambiar el archivo de configuraciones: El archivo de configuraciones necesita ser alterado en la seccion – *Database* –, aquí se encuentran las directivas para que el programa pueda encontrar el servidor de MySql y pueda acceder a los archivos de la base de datos. Por lo general es necesario cambiar las variables; *DatabaseURL, DatabaseUser y DatabasePass.* Los valores por defecto para estas variables son; *jdbc:mysql://localhost/, root y adminadmin.* Por favor cámbielos conforme a como

la instalación de MySql ha sido realizada en su computador.

Además las variables que son importantes en este archivo para el caso del búscador personalizado son: **MaxPalabrasFiltrado**, que es el total de palabras que van a ser tomadas en cuenta para clasificar páginas nuevas que lleguen al sistema. **MaxPalabrasRefinado,** el número de palabras que se utilizaran para realizar refinamiento de consultas. **ValorSumaAdaptativo,** este es el valor que va a ser incrementado en determinada categoría en modo perfil adaptativo cuando el usuario de un click sobre una página de una categoría específica (demostrándo así su interés en la misma, luego esto es normalizado). **Umbral,** variable utilizada en modo perfil adaptativo para determinar que categoría es de interés para el usuario, así en el caso del archivo DEFAULT todas aquellas categorías que tengan un valor de interés mayor a 0.2 serán consideradas categorías preferidas por el usuario.

Cambiar estas variables determina el objeto de estudio de esta tesis.

5. Añadir el proyecto en netbeans: En el IDE ir al menu *File* y elegir la opción *Open Project*. Ubicar la carpeta *PsearchApp* en donde se la haya copiado localmente (Paso 1) y abrirla, esta carpeta es un proyecto de Netbeans. Al momento de añadir el proyecto probablemente tenga la notificación de que existen referencias incorrectas, esto será solucionado en el siguiente paso. Ver gráfico 6.1.

6. Añadir las librerias necesarias: Expanda el árbol de navegación del proyecto, en el primer nivel se encuentra la sección *Libraries*, dar click derecho sobre esta sección y elegir la opción *Add JAR/Folder*, luego de esto ubicar el archivo *googleapi.jar* (paso 2) en donde este haya sido almacenado y añadirlo. Hacer lo mismo para los dos archivos *jar* restantes, es decir, *mysql-connector-java-3.1.13-bin.jar* y *xerces.jar*. Con esto se solucionan los errores de referencias del proyecto. Ver gráfico 6.2.

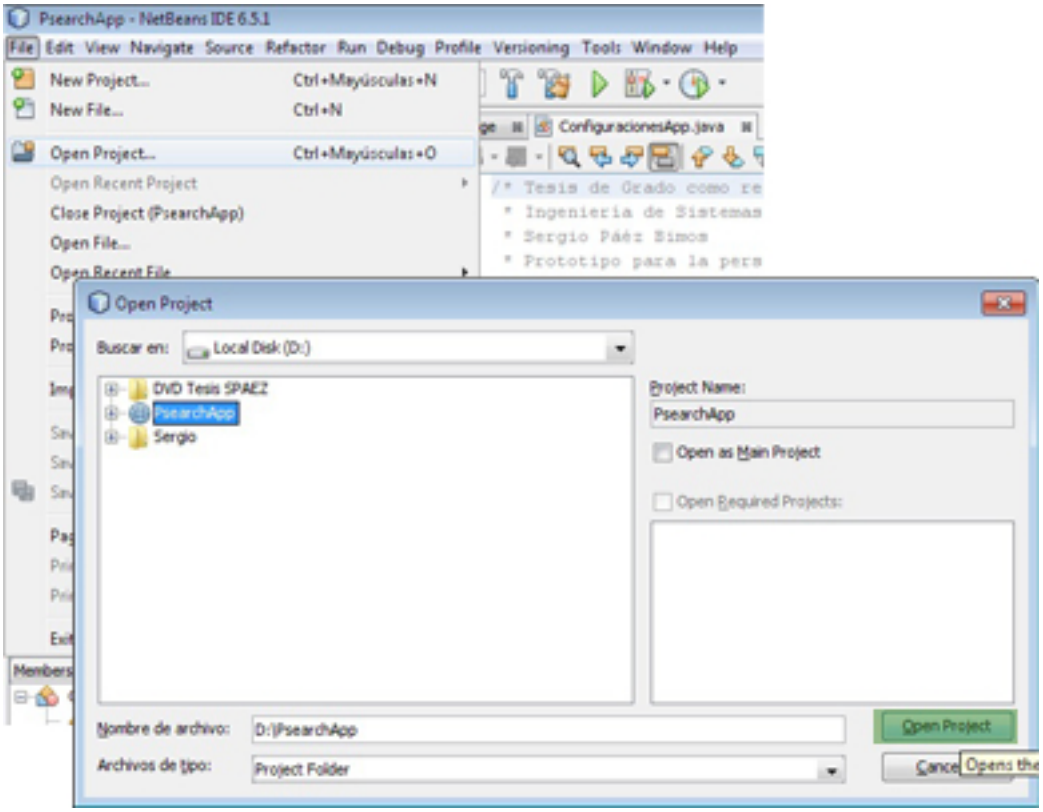

Figura 6.1: Netbeans: Abrir el proyecto

7. Apuntar el prototipo al archivo de configuraciones: Cabe destacar que el proyecto se encuentra divido en tres partes (ver en el primer nivel de navegación la seccion *Sources Packages*), estas son: *Buscador*, aqui se encuentran todas las clases propias del proyecto web. *Clasificador*, aquí estan todas las clases que construyen la información necesaria para el clasificador (de hecho ya no son usadas por el buscador, lo que si es usada es la información que esta sección retornó). *Generales*, es la parte donde se encuentran las clases que son usadas en común por las dos anteriores secciones. En esta ultima sección de *generales* se encuentra la clase *ConfiguracionesApp* la cual es encargada de leer el archivo de configuraciones (paso 3 y 4), en la línea 14 se encuentra la variable *archivo* la cual deberá ser cambiada de tal manera que apunte a donde el archivo de configuraciones, previamente almacenado en el sistema local (paso 3).

8. Hacer restore de la Base de datos: En el DVD dentro de la carpeta *Restore BD* se

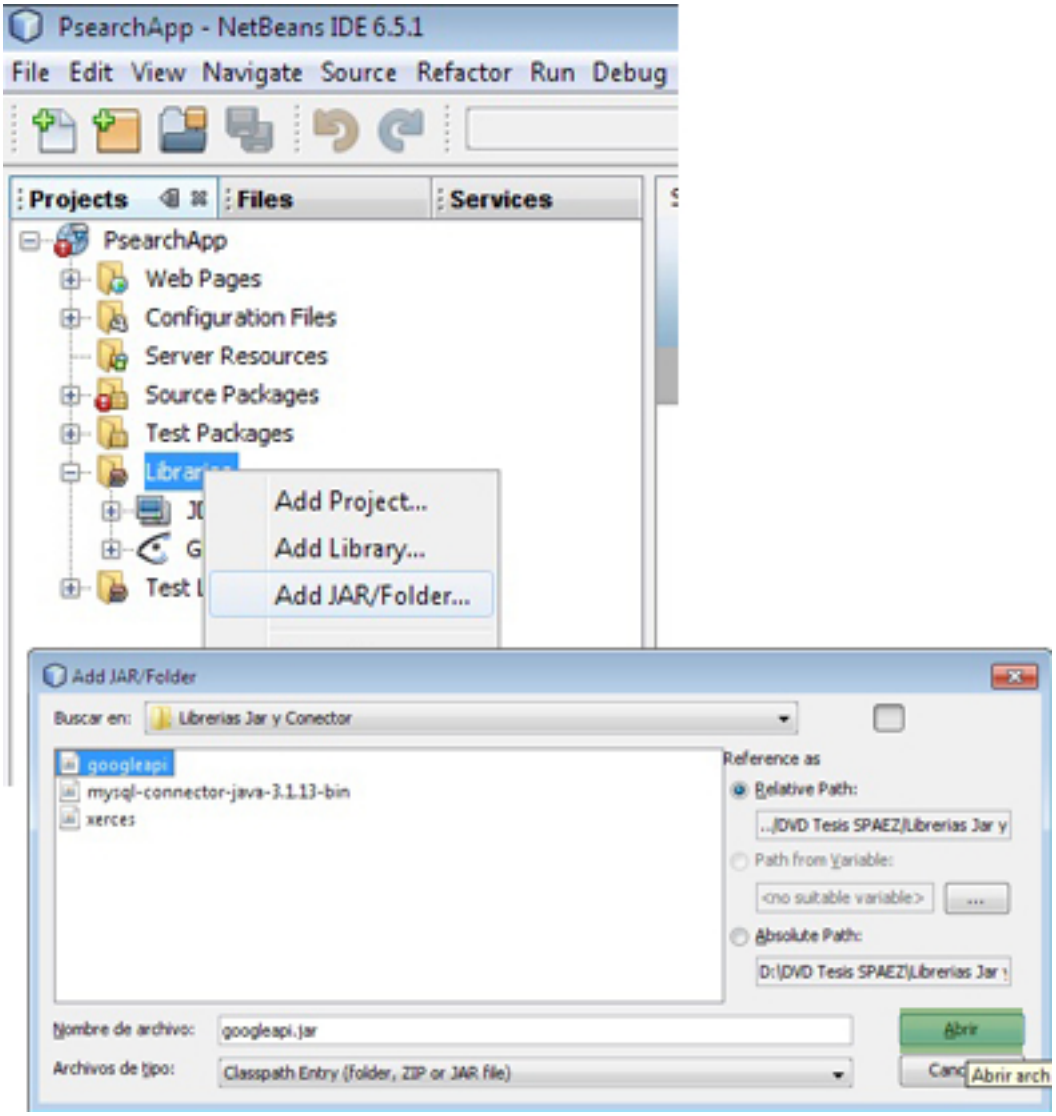

Figura 6.2: Netbeans: Añadir una libreria

encuentra dos archivos *.sql* que contienen un respaldo de la información necesaria para el uso de este prototipo. El archivo *Psearch Buscador* contiene todo lo que se necesita para correr el buscador y deberá ser recuperado primero, por otro lado el archivo *Psearch Clasificador* tiene toda la información correspondiente al clasificador y todo lo que se ha usado para llegar a tener palabras que definan a cada una de las categoriías (hacer restore de este archivo es opcional para el uso del buscador como tal). El restore de la base se lo puede hacer usando la línea de comandos o usando las herramientas gráficas que hemos instalado, para esto corra el programa *MySQL Administrator* (instalado con GUI Tools, lo debe encontrar en el menú Inicio y en la subcarpeta MySql de no haber cambiado esto al instalar), una vez abierto el programa dirígase la pestaña de *Restore* y en la parte de *File to restore* seleccione el archivo de backup de la base de datos que viene con el DVD (para hacer este proceso más rápido es recomendable copiar el archivo desde el DVD hasta el disco duro de la máquina). Ver gráfico 6.3.

9. Correr el Programa: Ahora ya que todo lo necesario para inicial el proyecto esta realizado de click derecho sobre el proyecto en netbeans y elija la opción *Run*, esto inicia el proyecto y abre una ventana del explorador de internet que tiene predeterminado con la aplicación lista para usar. Ver gráfico 6.4.

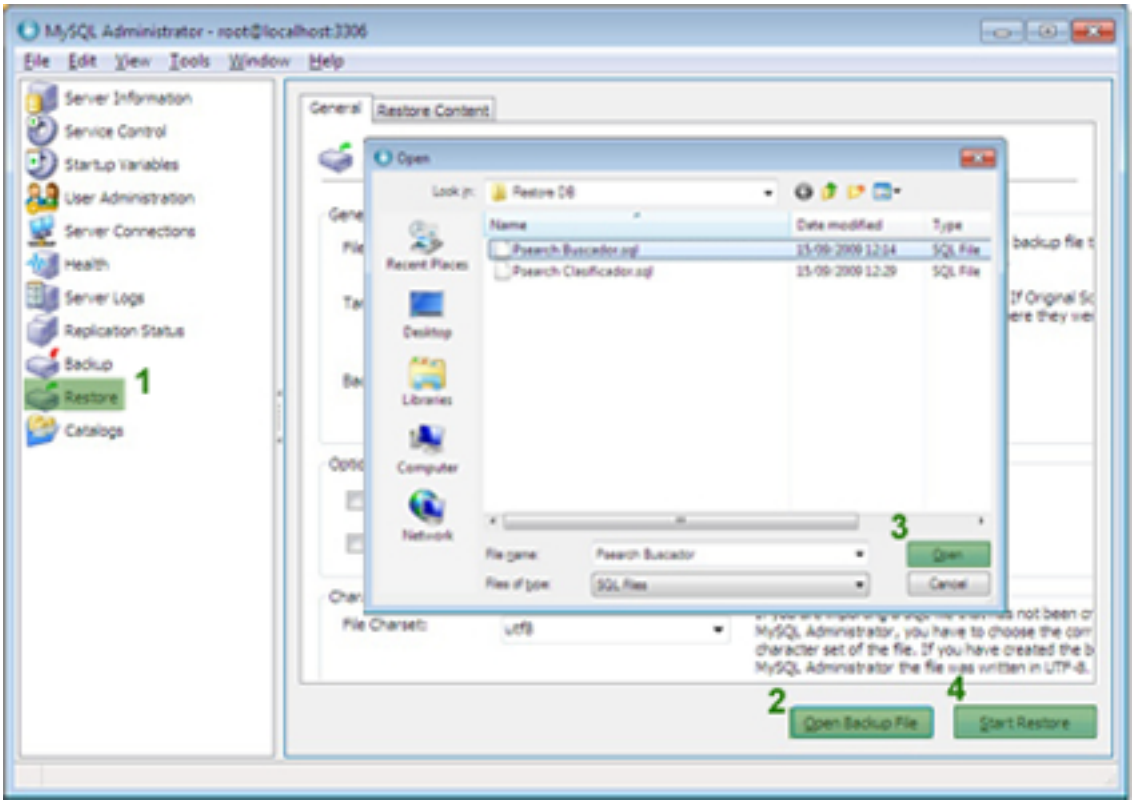

Figura 6.3: MySQL Administrator: Hacer restore

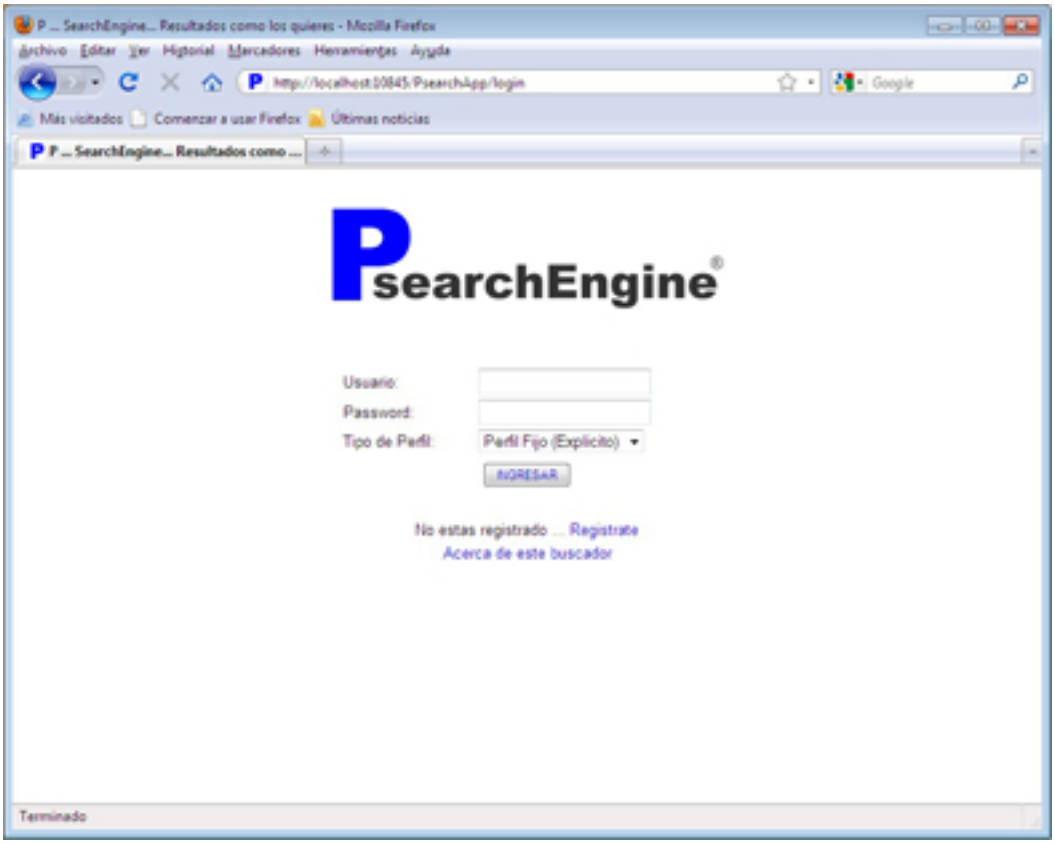

Figura 6.4: Aplicacion PSearch abierta.