Buscadores de Contenidos para Bibliotecas Digitales: Desarrollo de una Arquitectura para un Buscador XML

Enrique Sánchez Villamil[1] y Rafael C. Carrasco[2]

[1] Biblioteca Virtual Miguel de Cervantes,
Universidad de Alicante,
E-03071, Alicante, España
enrique.sanchez@cervantesvirtual.com
http://www.cervantesvirtual.com/
[2] Departamento de Lenguajes y Sistemas Informáticos,
Universidad de Alicante,
E-03071, Alicante, España
carrasco@dlsi.ua.es
http://www.dlsi.ua.es/

Resumen. El gran crecimiento del volumen de datos en las bibliotecas digitales hace necesario el desarrollo de buscadores de información que permitan al usuario encontrar rápida y eficazmente aquello que requieren. El etiquetado XML de los textos permite incorporar información estructural en los contenidos digitales. Las enormes posibilidades de explotación de estos contenidos hacen posibles servicios y facilidades de cara al usuario que hace unos años serían impensables. Este artículo explica los requisitos que debe reunir un buscador para contenidos en bibliotecas digitales y propone una arquitectura específica para el mismo. La arquitectura está diseñada para el trabajo con grandes volúmenes de texto, con un marcado estructural, para estar accesible vía web y atender un gran número de accesos. Esta arquitectura ha sido implementada y probada con éxito en la Biblioteca Virtual Miguel de Cervantes.

1 Introducción

En los últimos años, el volumen de información digital disponible se ha multiplicado, hasta el punto de que ningún usuario pueda leer toda la información, ni siquiera a lo largo de toda su vida. Se hace indispensable el desarrollo de herramientas que permitan a un usuario acceder a la información que necesite, de un modo sencillo y efectivo.

Una biblioteca digital dispone de gran cantidad de textos que ofrece al usuario lector de una forma más o menos accesible. En muchas ocasiones el usuario no accede a la biblioteca digital para leer un texto sino para *consultarlo.*

En las bibliotecas clásicas[1], la información no está siempre accesible (alquiler de libros), y además el método de consulta es absolutamente de prueba y error. Por ejemplo: buscamos información sobre un tema determinado, así que vamos al estante de ese tipo de libros y uno a uno ojeamos los índices de libros que hay disponibles a ver si contienen dicha información.

La biblioteca digital debe aprovechar que no tiene a priori restricciones de ubicación ni de disponibilidad (de las obras digitalizadas), para ofrecer al usuario una experiencia más satisfactoria. Siguiendo con el ejemplo anterior: buscamos las obras de un tema determinado y aprovechando que siempre estarán disponibles, podemos consultar rápidamente y desde cualquier sitio la información que requerimos.

Pero, ¿por qué quedarnos ahí? Una vez que la información está digitalizada se abre el abanico de posibilidades de explotación de dicha información. El avance en accesibilidad es el desarrollo de buscadores de contenidos de los textos, que librarán al usuario de la tediosa tarea de buscar lo que necesita manualmente.

El siguiente paso es apoyarnos en la estructura de los textos digitalizados. Al digitalizar, es interesante no quedarnos simplemente con el texto, sino guardarnos el formato de la obra, así como algo de contenido estructural y semántico, para generar nuestro contenido digital, usualmente en XML. La estructura arborescente del XML permite unas búsquedas de nivel más alto.

Actualmente existen herramientas de búsqueda en textos XML como el *Fxgrep* [2] o el *Xset* [3] . *Fxgrep* permite realizar búsquedas muy potentes, pero el tiempo de respuesta es lento haciendo difícil su integración en un servicio accesible vía web. *Xset* por el contrario tiene poca potencia en las búsquedas pero es bastante rápido y trabaja con colecciones relativamente pequeñas de documentos.

En este artículo proponemos una arquitectura que está diseñada para tratar gran cantidad de información, permitir realizar un conjunto de búsquedas más que aceptable y a una velocidad muy elevada. También trataremos los requisitos que debería cumplir un buscador y como se han resuelto con la arquitectura propuesta. Por un lado estudiaremos la arquitectura de sistema distribuido propuesta para sistema, y por otro lado la arquitectura interna del buscador.

2 Requisitos de un buscador

Cuando nos planteamos el diseño de un buscador tenemos que proponernos una serie de objetivos, a menudo contrapuestos, entre los que tendremos que llegar a un compromiso en función de las necesidades de la biblioteca digital en cuestión.

Actualmente algunos de esos objetivos o características se han hecho básicos como por ejemplo el ordenar los resultados de una búsqueda, el mostrar temas relacionados a lo que buscamos, o la posibilidad de restringir búsquedas a un determinado conjunto de documentos.

Podemos clasificar las características deseables en dos grupos, por un lado las de funcionalidad como las mencionadas anteriormente, y por el otro las de prestaciones como fiabilidad, velocidad, seguridad... Las características más deseables en orden de importancia [4] son:

1. *Eficacia.* Encontrar lo que estamos buscando es primordial. La eficacia influye en gran medida en el nivel de calidad del buscador y el de satisfacción del usuario. Solo puede conseguirse el nivel de eficacia adecuado si el usuario puede transmitirnos lo que quiere encontrar mediante una interfaz adecuada, que deberá ser lo suficientemente potente como para ser muy específica, pero lo suficientemente simple como para que la pueda utilizar el mayor número de usuarios.

2. *Velocidad.* Una búsqueda debe ser rápida para que el usuario no se canse de esperar. Dado que muchos usuarios puedan utilizar el buscador de forma simultánea, necesitamos un sistema extremadamente rápido. La contrapartida de la velocidad es la potencia de las búsquedas, puesto que las búsquedas más complejas requieren mayor cantidad de tiempo para resolverse. La velocidad no debe decrecer sustancialmente a medida que incrementamos el volumen de las colecciones de datos, que podrá crecer hasta el orden de varios GBs.
3. *Extracción de información*. No solo debemos decirle al usuario dónde está lo que busca, sino, en la medida de lo posible, mostrarle directamente lo que busca. En el caso de una biblioteca digital, si buscamos una frase no basta con decirle al usuario en qué obra está la frase. Tendremos que mostrarle cada vez que aparezca la frase, el contexto en el que aparece, así como dirigirle al punto exacto de la obra publicada en el que se encuentra el resultado de su búsqueda.
4. *Seguridad*. El buscador necesita para su funcionamiento información privada de la biblioteca que debe esconder al usuario. En nuestro caso tenemos que preservar los textos evitando que puedan copiarse a través del buscador. La seguridad está muy relacionada con la extracción de información, puesto que cuando se accede a las obras para mostrar el contexto de lo que se ha buscado es para satisfacer dicho requisito.
5. *Multimedia.* Las bibliotecas digitales *multimedia* tienen requerimientos extra, debido a que los datos que manejan son mucho más extensos en cuanto a su tamaño en bytes, así como en cuanto al tiempo necesario para procesarlos. Este tipo de datos quedan fuera del ámbito de nuestro buscador en textos XML, aunque podrían integrarse sin dificultad en la arquitectura que proponemos.
6. *Expansión de búsqueda*. Consiste en que cuando el usuario nos solicita una búsqueda, se realizan muchas búsquedas similares para aportar al usuario una respuesta más completa. Los mecanismos de expansión se basan en tesauros y en heurísticas basadas en procesamiento del lenguaje natural.

En definitiva, la arquitectura para el buscador debe tener en cuenta todos estos aspectos generales, puesto que son ellos los que permiten evaluar la calidad de la misma, junto con las funcionalidades que facilita.

3 Arquitectura del buscador

El sistema de búsquedas que proponemos es un sistema distribuido, que se divide en tres subsistemas, como vemos en la Figura 1. Estos subsistemas son bastante independientes y se comunican vía TCP/IP, por lo que es aconsejable que se ejecuten en máquinas distintas para un mayor rendimiento[5]. Cuando hablamos aquí de rendimiento nos referimos al nº máximo de búsquedas que podemos atender por segundo.

Los usuarios acceden al servicio de búsquedas a través del servidor web, que es quien ejecutando el programa cliente buscador, envía la petición de búsqueda al servidor de usuarios. El servidor de usuarios se encarga de distribuir la petición al servidor de consultas más adecuado y de devolverle la respuesta a la petición al cliente. Los servidores de consultas trabajan de un modo independiente, por lo que necesitan tener el índice completo.

Esta arquitectura ofrece el máximo rendimiento, pero podemos perfectamente ejecutar todos los servidores en una misma máquina y obtener un rendimiento más que aceptable.

Para hacernos una idea, veamos un ejemplo real de la implementación del sistema en la BVMC [6]. En una colección de ficheros XML de 300 MBs, y ejecutando todos los servidores en una misma máquina se consigue una media de aproximadamente 20000 búsquedas por hora. Por tanto, si no necesitamos rendimientos superiores o trabajamos con un volumen de información mucho mayor, no es necesario dedicar más que una máquina al sistema buscador.

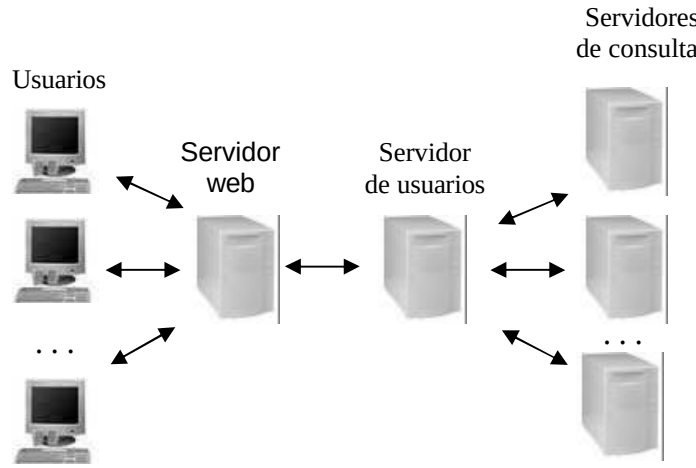

Figura 1. Arquitectura de subsistemas que conforman el sistema de búsquedas.

A continuación explicamos el cometido de los servidores más detalladamente.

3.1 Cliente buscador en servidor Web

El cliente buscador es el subsistema volátil de la arquitectura, puesto que cada cliente solo atiende una búsqueda. Su cometido es construir un paquete de información con los datos sobre la búsqueda solicitada, el usuario que la solicita, y datos varios.

El cliente también debe realizar una adaptación entre codificaciones distintas. Los datos sobre la búsqueda solicitada deben convertirse a la codificación ASCII, puesto que la interfaz web utiliza una codificación hexadecimal para ciertos caracteres.

La búsqueda se especifica mediante un protocolo específicamente diseñado para posibilitar cualquier tipo de búsqueda que permite el índice, pero de un modo optimizado para que la resolución de la búsqueda pueda efectuarse rápidamente. Esto se consigue no solo especificando lo que queremos buscar, sino además cómo buscarlo.

El paquete así construido debe enviarse al servidor de usuarios, y cuando conteste que ha recibido el mensaje, el cliente estará a la espera de la respuesta del servidor de usuarios. La respuesta consiste en un fragmento de página web, que el cliente devolverá al usuario, no sin antes haberle realizado las modificaciones oportunas, como por ejemplo insertar la cabecera y el pie a la página. Tras devolver el resultado al usuario el cliente buscador finaliza su ejecución.

3.2 Servidor de usuarios

El servidor de usuarios es el encargado de la coordinación del sistema. Recibe peticiones de todos los clientes y se encarga de atender directamente las que pueda y de mandar a uno de los servidores de consultas disponible el resto. Como es lógico tras recibir las respuestas de los servidores de consultas las envía de vuelta a los clientes que las solicitaron.

Su funcionamiento está basado en una base de datos heterogénea que contiene gran diversidad de información como vemos en la Tabla 1. Toda esta información nos permite una coordinación global del sistema, así como la generación de estadísticas para analizar el funcionamiento del sistema.

Tabla 1. Información contenida en la base de datos del servidor de usuarios

Tipo	Datos
Servidores de consultas	Disponibilidad, carga de trabajo, cola de búsquedas pendientes, tiempo inactivo, peticiones servidas
Usuarios	Direcciones IP, búsqueda(s) solicitada(s), tiempo en espera
Búsquedas	Usuarios que la solicitan, servidor de consultas encargado de atenderla, estado, coincidencias encontradas, tiempo que tardó en realizarse, resultado devuelto por el servidor de consultas
Estadísticas	Tiempo de actividad, nº de búsquedas, tiempo medio por búsqueda, tamaño de los resultados devueltos

La información referente a búsquedas permite que el servidor de usuarios actúe como una caché, almacenando temporalmente los resultados de las mismas, y permitiendo la devolución directa del resultado, si está disponible, al cliente. Además nos permite desechar búsquedas que lleven mucho tiempo encoladas (caso de saturación)

Por otro lado, la información sobre los usuarios nos permite restringir sus peticiones de búsqueda, mediante un sencillo sistema de contraseñas. Así podremos fácilmente distinguir entre peticiones públicas y peticiones privadas. La configuración del sistema nos permite diferenciar entre varios niveles de privilegios, para de ésta forma limitar el acceso a los servicios de administración del buscador.

La otra tarea del servidor de usuarios es la de la gestión del registro del sistema y la generación on-line de estadísticas. En el registro se anotarán todas las peticiones que se reciban, y se reflejarán todas las incidencias que se produzcan, como peticiones rechazadas o peticiones descartadas.

El servidor de usuarios es único en el sistema, puesto que es la única forma de centralizar el sistema de estadísticas, de registro, y en definitiva de maximizar el rendimiento de todo el sistema, teniendo en cuenta los volúmenes de información que se mueven en bibliotecas digitales. El único límite de rendimiento de la arquitectura es el marcado por el servidor de usuarios, puesto que todas las búsquedas pasan por él, aunque, dado el escaso procesamiento que realiza, el rendimiento normalmente se verá limitado por el número de servidores consultas que utilicemos.

3.3 Servidor de consultas

El servidor de consultas atiende las peticiones de búsqueda que recibe del servidor de usuarios. Su funcionamiento se basa en encolar las peticiones que llegan y en ir resolviéndolas una por una y devolviendo los resultados al servidor de usuarios.

La resolución de las búsquedas consta de dos fases. En la primera fase se analiza la búsqueda solicitada y se calcula todas las veces que aparece (ocurrencias) lo que se busca en los textos XML. En la segunda fase se accede a los ficheros XML para generar el resultado con el contexto de las ocurrencias solicitadas y se añaden las expansiones de la búsqueda. Siempre se solicita un rango de ocurrencias a mostrar, es decir, se dice que queremos las ocurrencias de la 1 a la 50, o lo que necesitemos.

La fase más costosa es el cálculo de los resultados de una búsqueda, en la que se calculan, por lo general, muchísimas más ocurrencias de las que vamos a mostrar al usuario. Por eso tras acabar la primera fase almacenamos todas las ocurrencias en una caché de ocurrencias, que evitará repetir la búsqueda si el usuario solicita ver más ocurrencias de esa misma búsqueda.

La segunda fase también tiene partes costosas, como el acceso a los ficheros XML puesto que están en disco. Tras dicho acceso se realizará la generación de expansiones de búsqueda. Las expansiones consisten en sugerencias para el usuario de búsquedas relacionadas con la solicitada. Los mecanismos de generación de expansiones se describen brevemente en el apartado 4.3.

4 Sistema buscador en textos XML

El sistema buscador es el encargado de atender todas las peticiones de búsqueda. Su funcionamiento se apoya en la generación previa de unos índices para proporcionar acceso directo a todas las palabras, etiquetas y atributos incluidos en la colección de archivos XML en los que vamos a realizar las búsquedas.

Con esos índices la realización de las búsquedas se realiza siempre en un tiempo constante, independientemente del tamaño de la colección de archivos XML, a pesar de que para cualquier búsqueda solicitada sea necesario realizar más de un acceso a este índice.

En la Figura 2 mostramos la estructura de módulos que conforman el sistema de búsquedas. Básicamente, el buscador analiza la petición de búsqueda para ver qué debe hacer. Una vez decidida la estrategia a seguir, se realizan las búsquedas necesarias en el índice y se analizan las expansiones para sugerirlas al usuario. Veamos los módulos en más detalle.

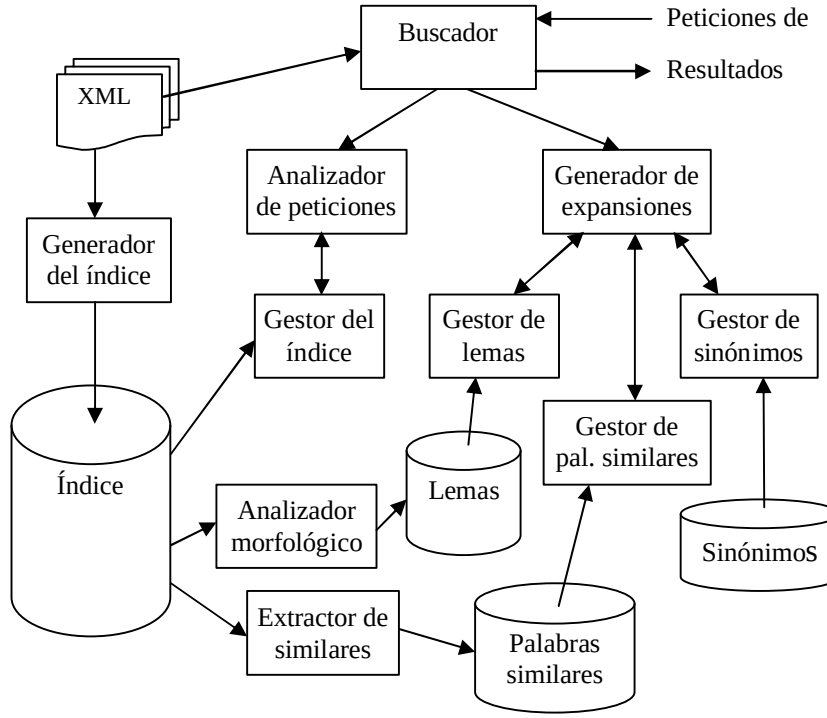

Figura 2. Esquema de funcionamiento del buscador. Contiene todos los módulos que lo componen así como las bases de datos que utiliza.

4.1 Generador y gestor del índice

El módulo generador del índice es el encargado de extraer la información de la colección de ficheros XML acerca de dónde (posiciones) se encuentra cada una de las palabras, etiquetas y atributos.

La estructura de datos del índice, diseñada originalmente por Rafael Carrasco y Sergio Ortiz [7], dispone de una tabla hash para cada tipo de elemento (palabra, etiqueta, atributo) así como un fichero de índice para cada uno. Los ficheros índice contienen para cada elemento todas las posiciones dentro del texto en el que aparece dicho elemento.

Cada una de las tablas hash tendrá una entrada para cada elemento distinto, para cuando lo busquemos obtener directamente el número de veces que aparece, así como la posición dentro del fichero de índice en el que empiezan las ocurrencias del elemento.

Dado que el tamaño del grupo de ficheros de índice es aproximadamente el mismo que el de toda la colección de ficheros XML (no se utiliza ningún tipo de compresión), se hace inviable mantener todo el índice en memoria. Este diseño permite mantener en memoria solo las tablas hash, y así necesitar un único acceso a disco para acceder al

fichero de índice. El tiempo de generación del índice es de aproximadamente 20 minutos en un Pentium IV 2Ghz con una colección de 400 MBs de ficheros XML.

El módulo gestor del índice se encarga de ofrecer una interfaz simple para las búsquedas de palabras, de etiquetas o de atributos, así como de aportar mecanismos para realizar búsquedas de elementos dentro de otros elementos, aprovechando la estructura arborescente del XML.

4.2 Analizador de peticiones

Este módulo se encarga de estudiar la petición de búsqueda que nos ha enviado el cliente y de acceder al índice y a los ficheros XML para generar el resultado a dicha búsqueda. Su funcionamiento interno esta íntimamente ligado a la interfaz escogida para el buscador, por lo que es uno de los módulos más dependiente de la biblioteca digital con la que trabaja.

Para generar resultados se necesita el acceso a la colección de ficheros XML, lo que ralentiza bastante las búsquedas, sobre todo si dicha colección no se encuentra disponible localmente.

Por otro lado, la biblioteca digital suele tener publicada en Internet las obras en las que se desea buscar. Todos los resultados de las búsquedas deben disponer de un enlace a la obra y posición dentro de la obra en la que aparece cada ocurrencia, para que el usuario pueda llegar directamente a lo que busca.

4.3 Expansión de búsquedas

La tarea de este módulo es sugerir nuevas búsquedas parecidas a las que el usuario ha solicitado. Esta tarea no es imprescindible en el buscador pero aporta un mayor grado de satisfacción al usuario al permitir una corrección semiautomática de fallos tipográficos del usuario y al disminuir el tiempo que tarda el usuario en encontrar aquello que realmente busca. La expansión de búsquedas se apoya en tres módulos que le aportan toda la información que necesita para llevar a cabo su labor. Estos módulos son el gestor de lemas, el de palabras similares y el de sinónimos.

El gestor de lemas provee acceso a los análisis morfológicos de todas las palabras existentes en los textos XML. Estos análisis se calculan previamente tras la generación del índice, para no perder tiempo cuando expandimos.

El gestor de palabras similares se encarga de darnos todas las palabras similares a una palabra dada. Por similar entendemos una palabra que se escribe prácticamente igual a otra. Previamente a la ejecución se calculan todas las palabras similares entre sí, de entre las existentes en los textos XML, para de esa forma no tener que calcularlo en tiempo de búsqueda. Sin embargo, si la palabra dada no está, nos vemos obligados a utilizar diversas heurísticas para calcular sus similares en tiempo real.

El gestor de sinónimos permite obtener sinónimos de las palabras más comunes que están contenidas en los textos XML. Dichos sinónimos se almacenan en una base de datos a la que accedemos durante la búsqueda. Obviamente, cuanto más completa sea la base de datos mayor será la eficacia a la hora de obtenerlos.

La expansión de búsquedas funciona de manera muy distinta en función del número de palabras que está buscando el usuario. Cuando el usuario busca una sola palabra, se apoya sobre todo en las palabras similares y en los sinónimos a la hora de sugerir

búsquedas. En cambio cuando se buscan más palabras, se analizan para ver si concuerdan en género, número y persona (si es un verbo), y las expansiones son relativas a estas concordancias.

5 Conclusiones

Tras analizar la arquitectura desde un punto de vista global, vemos que cumple bastante bien con los objetivos que nos marcamos previamente, siempre en la medida de lo posible.

La eficacia de la arquitectura, es muy alta, puesto que siempre buscamos lo que quiere el usuario y como mucho sugerimos, pero no decidimos por él. Sin embargo, la eficacia es muy dependiente de la interfaz web que le demos a nuestra aplicación, puesto que si el usuario no puede expresar bien lo que quiere la eficacia será menor.

Puesto que el sistema de búsquedas planteado siempre extrae la información de alrededor de las ocurrencias de las palabras buscadas, cumple con el requisito de la extracción de información. Además dado que genera enlaces de las ocurrencias encontradas a la dirección del texto en la que se encuentra permite al usuario el acceso directo a toda la información referente a lo que está buscando.

La velocidad de esta arquitectura es muy dependiente a la implementación, pero a priori es muy elevada, puesto que si necesitamos mayor rendimiento basta con añadir más servidores de consultas: es una arquitectura muy escalable. Dado que el procesamiento realizado en el servidor de usuarios es mínimo, no supondrá ningún problema atender cualquier número de peticiones que lleguen. Aquí se juega con la velocidad de la red, ya que es la que determina el número de peticiones máximas que pueden llegar por segundo. Si el servidor de usuarios es capaz de atenderlas todas entonces el rendimiento global nunca dependerá de él.

La seguridad de la información con esta arquitectura es muy alta, puesto que para que un usuario accediera debería pasar a través de un servidor web, luego a través del servidor de usuarios, que lleva un registro de peticiones y restringe el acceso cuando es necesario, y por último a través del servidor de consultas, que es quien accede a la colección XML y solo devuelve al usuario texto plano (sin etiquetas), para preservar el XML original.

Por último, la expansión de búsquedas, que se realiza basándonos en criterios gramaticales, en similaridad de palabras y en sinonímias, facilita al usuario el trabajo con el sistema.

Por otro lado, la arquitectura propuesta está preparada para integrar fácilmente nuevos servicios que se necesiten en un futuro, como por ejemplo el trabajo con datos multimedia, que dentro de unos años se prevé que sea lo habitual.

Referencias

1. Oya Rieger: "Zoom In - Zoom Out: Imaging to Asset Management" (Marzo 2003).
2. Fxgrep: “A XML Querying Tool” http://www.informatik.uni-trier.de/~aberlea/Fxgrep/
3. Ben Y. Zhao, Anthony Joseph: “XSet: A Lightweight XML Search Engine for Internet Applications” : http://www.cs.berkeley.edu/~ravenben/xset/html/xset-saint.pdf

4. W. B. Croft: “What Do People Want from Information Retrieval?” D-Lib Magazine, (November 1995).
5. Martínez Barco, Saiz y Clavel: “Sistemas Informáticos Distribuidos” (Ed. Club Universitario 1999).
6. Biblioteca Virtual Miguel de Cervantes http://www.cervantesvirtual.com
7. Sergio Ortiz: "Trabajo de investigación tutelada" (tutor Rafael Carrasco 2003).
8. Witten, Moffat y Bell: "Managing Gigabytes" (1999 2ª Ed).
9. Manning y Schütze: "Foundations of Statistical Natural Language Processing" (2000).
10. Baeza-Yates: "Modern Information Retrieval" (1999).