UCI
Universidad de las Ciencias Informáticas

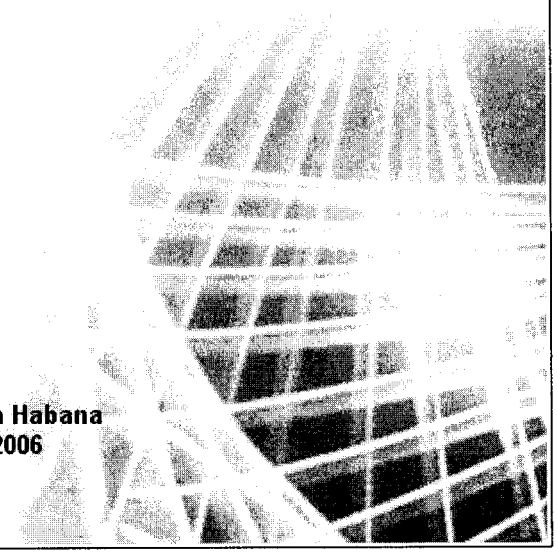
Cuba Search

Araña del Buscador Web CubaSearch

Trabajo para optar por el título de Ingeniero Informático

Autor
Ramsés Ibarrola Suárez

Tutor
José Albert Cruz Almaguer



Ciudad de La Habana
Abril 2006

RESUMEN

Viajar por Internet es una tarea fácil gracias a los Buscadores. A través de ellos es posible recuperar información de cualquier país del mundo. Actualmente en nuestro país no existe ningún motor de búsqueda, por tanto los internautas cubanos para realizar sus búsquedas acerca de nuestro país dependen de buscadores internacionales, como Google o Yahoo, esto conlleva a ver resultados no deseados de sitios que ni siquiera son cubanos, como por ejemplo páginas cuyos servidores están en otros países y su contenido no refleja nuestra realidad.

El sitio cubano CubaSI desea desarrollar el primer buscador web cubano y en un convenio realizado con la Universidad de las Ciencias Informáticas, surge el proyecto CubaSearch, y como uno de sus eslabones fundamentales, la Araña del buscador Web CubaSearch que tiene como **objetivo concreto** diseñar e implementar un robot indexador de sitios web, además de proponer una versión de robot distribuido.

Como principales resultados se ha logrado que el software sea capaz de indexar unos 60 documentos por minuto aproximadamente, además de generar reportes. Se explican los conceptos relacionados con el mismo, se hace un análisis de la propuesta del sistema, y se dejan algunas recomendaciones para el mejoramiento futuro del mismo.

INDICE

INTRODUCCIÓN

Con la aparición del WWW, a principios de los años 90 del pasado siglo, se produjo un crecimiento vertiginoso del número de usuarios y recursos de información en Internet. Una de las principales valoraciones que se han escuchado siempre acerca de la Internet es la dificultad de encontrar la información específica que se desea. Aunque exagerados, estos criterios no dejan de tener algo de cierto, sobretodo durante los primeros años de la Red, cuando las búsquedas de información podían complicarse enormemente con el uso de los primeros sistemas[1], que permitían la interconexión de computadoras y la descarga de documentos y archivos. Cuando se necesita de Internet, normalmente buscamos información sobre un tema concreto, y es difícil acceder a una página que la contenga, simplemente pinchando vínculos. Como solución a este problema surgieron los buscadores. Su uso facilita enormemente la obtención de un listado de páginas web que contienen información sobre el tema que nos interesa.

Existen principalmente dos tipos de buscadores: los buscadores temáticos o directorios[2], y los buscadores por palabra clave, que están formados por un motor de búsqueda[3], y un robot de búsqueda.

Actualmente en nuestro país no existe ningún buscador web, a penas en algunos portales como www.cuba.cu y www.islagrande.cu solo contienen directorios webs, trayendo como consecuencia la siguiente **situación problémica**: no se tiene conocimiento de la cantidad

[1] Antes de que se popularizara la World Wide Web (la Web, en corto) a mediados de la década de los años Noventas, los usuarios de Internet contaban básicamente con tres herramientas para intercambiar información: *E-mail* (para el envío personalizado de mensajes electrónicos), *Telnet* (para conectarse a una computadora remota) y *FTP* (para depositar o recuperar archivos de texto o binarios). En el mejor de los casos, el "cibernauta" podía tener acceso a listados de los distintos archivos y carpetas en un servidor en particular conectado a la Red.

[2] Los directorios son listados de recursos organizados en categorías temáticas.

[3] Se trata de un programa que busca a través de una base de datos, en el contexto de la Web, se refiere usualmente a búsquedas de bases de datos de documentos Html, recopilados por un robot.

de sitios que actualmente están en la intranet cubana, el portal CubaSI no tiene ningún buscador web, la forma de buscar información en nuestros sitios es mediante buscadores internacionales (Google, Yahoo, etc.), no es posible determinar estadísticas acerca de las páginas cubanas debido que no existe ningún robot de búsqueda especializado en Cuba.

El portal cubano CubaSI, entidad subordinada a la Empresa de Telecomunicaciones de Cuba S.A. (ETECSA) líder en la provisión de servicios de Internet en Cuba, dedica sus acciones a la comercialización de productos y servicios, en conjunto con la UCI, conformaron un grupo de trabajo por estudiantes de la Universidad y liderados por los diseñadores del portal CubaSI para desarrollar un proyecto denominado ***CubaSearch***, que tiene como objetivo principal hacer un buscador web para páginas cubanas, por todo esto surge la **necesidad** de desarrollar un sistema que permita realizar las funciones de un robot de búsqueda.

Para la implementación del sistema se escogió el desarrollo de la aplicación con una interfaz de ventanas debido a que el sistema solo interactúa con un solo usuario, el administrador del sistema, y por lo tanto no necesita ser publicado en ningún servidor web, además de ser un sistema que no pertenece a la cara del buscador, es decir que los usuarios del buscador solo interactúan con el motor de búsqueda y el directorio.

Una de las principales metas de este trabajo es crear un sistema automatizado que sea capaz de moverse por todos los sitios webs cubanos y recopilar datos de cada URL[4] visitada, tales como el título, los enlaces relacionados a él, los metadatos[5] y las palabras en forma de texto, formando parte así del primer buscador cubano, propiciando además facilidades al administrador con una interfaz sencilla y amigable. El principal beneficio del sistema, sin embargo, será que va a propiciar que se pueda tener una idea, al menos aproximada, de la cantidad de información que se mueve por los sitios cubanos y por lo tanto se tendrá almacenada gran parte de ella, además de ser el punto de partida para el motor de búsqueda.

[4] (Uniform Resource Locutor). La dirección global de documentos y otras fuentes en el Internet

[5] Son datos altamente estructurados que describen información, describen el contenido, la calidad, la condición y otras características de los datos, en este caso de los datos contenidos en una página web.

El sistema a medida que "visita" cada URL va indexando[6] cada vínculo relacionado a este y así sucesivamente, haciendo posible que todos sean procesados, también y para evitar chequear vínculos repetidamente, cada URL procesada será registrada. Se toma como datos de gran importancia el título y los meta-datos, ya que estos definen el documento como tal; las palabras en forma de texto, - es decir, las que no forman parte de ningún tag[7] HTML-, se almacenarán junto con los demás datos del enlace recogido, pero no se incluirán las denominadas "stop words".

Para el comienzo del sistema se parte de una lista de enlaces que será introducida por el administrador, los vínculos contenidos en ella deben recoger la mayor cantidad de enlaces ya que este será el punto de partida del sistema. Por parte de la información persistente, el sistema está concebido para la asignación por parte del usuario del sistema de asignar un servidor de base de datos y en este se almacenarán todos los datos recogidos, siendo estos utilizados luego por el motor de búsqueda. A su vez aquellas páginas que hayan tenido algún problema de conexión en el momento de recopilación de datos, serán archivadas para una posterior indexación.

Por tanto el **objeto de estudio** de este trabajo es lo referente a la información contenida en todos los sitios y portales web cubanos.

Como **hipótesis** se parte de la idea de que si se desarrolla una aplicación con interfaz de ventanas, basada en un gestor de base de datos potente, y en una plataforma poderosa; es posible lograr la indexación de la mayor parte de los sitios cubanos de una manera rápida y eficiente, creando así la base para el motor de búsqueda.

Los **objetivos generales** de este trabajo son:

- *Diseñar un robot indexador de sitios Web.*
- *Implementar un robot indexador de sitios Web.*
- *Proposición de una versión de robot distribuido.*

[6] Simboliza la acción de ordenar registros mediante la utilización de índices

[7] Etiqueta del lenguaje HTML.

De aquí se derivan los siguientes **objetivos específicos**:

A. Hacer un estudio de las técnicas de indexación de sitios webs para elegir la más adecuada a las condiciones del sistema de intranet cubano.

B. Brindar información sobre el estado del sistema y los datos recogidos, mediante reportes.

C. Implementación de un módulo para indexar sitios webs.

D. Almacenar toda la información que se extraiga de los sitios webs.

Para cumplir los objetivos trazados se desarrollaron las siguientes **tareas**:

- ✓ Realizar un estudio del entorno de trabajo.
- ✓ Identificar las necesidades de la institución y del buscador en general.
- ✓ Declarar los requisitos que debe cumplir el sistema de acuerdo con sus necesidades.
- ✓ Descripción de los procesos que se van a implementar en el sistema.
- ✓ Declaración de los ciclos de desarrollo.
- ✓ Descripción de las clases del diseño.
- ✓ Diseño de la Base de Datos.
- ✓ Diseño de la interfaz.
- ✓ Implementación de la aplicación.

Este trabajo ha sido organizado de la siguiente manera:

Capítulo 1: Describe cómo se realiza el proceso de búsqueda actualmente en nuestro país. Se mencionan los principales problemas que generaron la necesidad del cambio, se analizan las soluciones existentes y como conclusión se obtienen los objetivos generales y específicos a cumplir.

Capítulo 2: Describe las técnicas y tendencias a desarrollar en la aplicación, se fundamenta la metodología utilizada y las especificaciones de software y hardware.

Capítulo 3: Describe el dominio mediante un Modelo de Dominio, se hace el análisis del sistema a desarrollar. Se definen las funcionalidades del sistema y se describen detalladamente, utilizando herramientas de modelación, los principales procesos del mismo.

Capítulo 4: Explica la fase de elaboración de la solución mediante diagramas de clases, se describen también los principios para el diseño y la implementación del sistema, se detallan y explican las funcionalidades que se definieron en el capítulo anterior.

Capítulo 1

FUNDAMENTACIÓN DEL TEMA

1.1 Introducción.

En este capítulo se brinda una visión general de los aspectos relacionados con los buscadores web, los conceptos necesarios para el estudio de los mismos, y las características de cada tipo de buscador. Los temas fundamentales en los cuáles se profundiza son: Los buscadores con robot de búsqueda y estructura y funcionamiento de un robot de búsqueda. Además se describen los principales conceptos asociados al dominio del problema que son necesarios para entender la propuesta de solución.

1.2 Recuperación de Información.

1.2.1 Concepto.

La Recuperación de Información es una actividad que el ser humano realiza, consciente e inconscientemente, casi continuamente y en el marco de cualquier actividad. La necesidad de resolver una duda, o de documentar una afirmación o estudio, son expresiones clásicas de los procesos de recuperación de información. Con el desarrollo de los sistemas digitales de procesamiento de datos y de tratamiento de información, las técnicas de recuperación de información han ido desarrollando un conjunto de teorías y

aplicaciones prácticas que subyacen en la actualidad a cualquier búsqueda y recuperación de información en Internet.

La Recuperación de la Información, RI, puede definirse como la representación, almacenamiento, organización y el acceso a elementos de información. El campo de RI envuelve un conjunto bastante grande de conceptos, estructuras y métodos.

Conceptualmente, la Recuperación de Información (RI) es una operación en la que se interpreta una necesidad de información de un usuario y se seleccionan los documentos más relevantes capaces de solucionarla, es decir, consiste en buscar documentos que exhiban un mayor parecido a la pregunta formulada. En el contexto de la WWW, se puede definir el objetivo de la recuperación como la identificación de una o más referencias de páginas web que resulten relevantes para satisfacer una necesidad de información.

Un sistema de recuperación de información (RI) es aquel que, con distintas técnicas, proporciona acceso rápido automático a grandes colecciones de objetos que contienen información de cualquier tipo. Estos objetos pueden tomar muchas formas, ya sean textos (artículos científicos, mensajes de correo electrónico, informes médicos, manuales de usuario, etc.), pero también imágenes, archivos de sonido, etc., asociando un texto a cada uno de ellos. En general se emplea el término documento para referirse a dichos objetos, y el término descripción del documento para referirse a su significado [1].

1.2.2 Características generales de los Sistemas de Recuperación de Información.

Según la clasificación de Lewis, dentro de los sistemas basados en textos podemos distinguir dos tipos:

• Sistemas de clasificación: Incluyen a:

- Los sistemas de recuperación de documentos que proporcionan como resultado un conjunto de textos (que puede ser vacío) en respuesta a una consulta de un usuario (por ejemplo, ver los documentos que contengan las palabras deseadas).
- Los sistemas de categorización de textos, cuyo objetivo es clasificar los documentos en una serie de categorías pre-establecidas.

• Sistemas de entendimiento: son aquellos destinados a generar resúmenes de los documentos, responder preguntas determinadas sobre un documento, y extraer datos del mismo.

Aquí se incluyen:

- Los sistemas de extracción de información, en los que se pretende extraer una fracción de todo el texto y en los que los matices del texto completo no son muy relevantes.
- Los sistemas de comprensión, cuyo objetivo es extraer el significado del documento completo y captar los matices del lenguaje empleado y los objetivos del autor.

En cualquier caso, los límites entre ambos sistemas son muy difusos, existiendo sistemas que logran varias o todas las funciones de ambas categorías.

Un sistema de recuperación de información comprende una serie de pasos básicos:

1. Generar una representación del significado o contenido de cada documento incluido en la colección a partir de su descripción. Este proceso es la indexación.
2. Generar una representación del significado de la consulta de usuario, llamado formulación.
3. Comparar estas dos representaciones para seleccionar aquellos documentos que tienen mayor probabilidad de satisfacer la consulta (comparación).

En la siguiente ilustración se muestran estos pasos básicos.

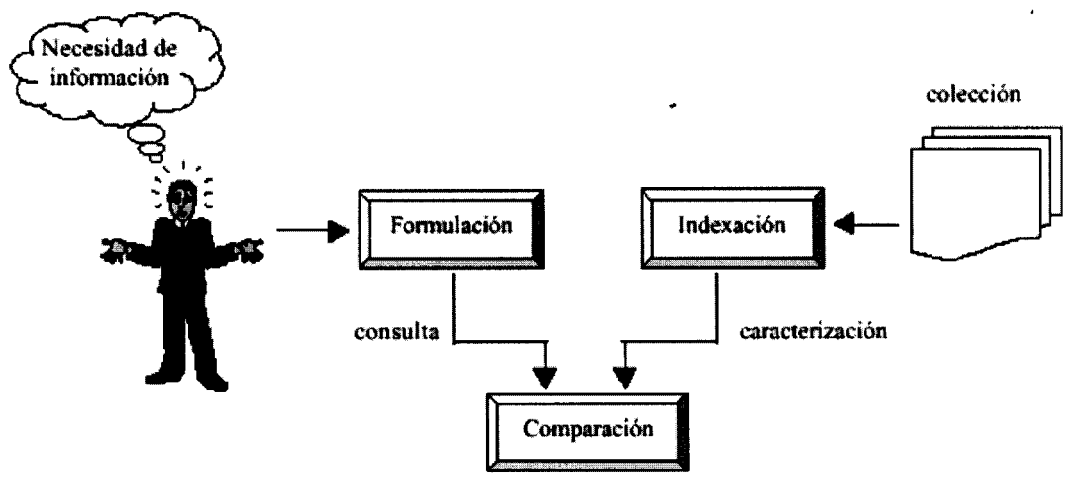


Ilustración 1.1 *Paradigma de la recuperación de información.*

Las fases del tratamiento de la información: el modelo conceptual, la indexación, la transformación de consultas, las operaciones sobre los términos y la gestión de documentos pueden verse en la figura 1.1.

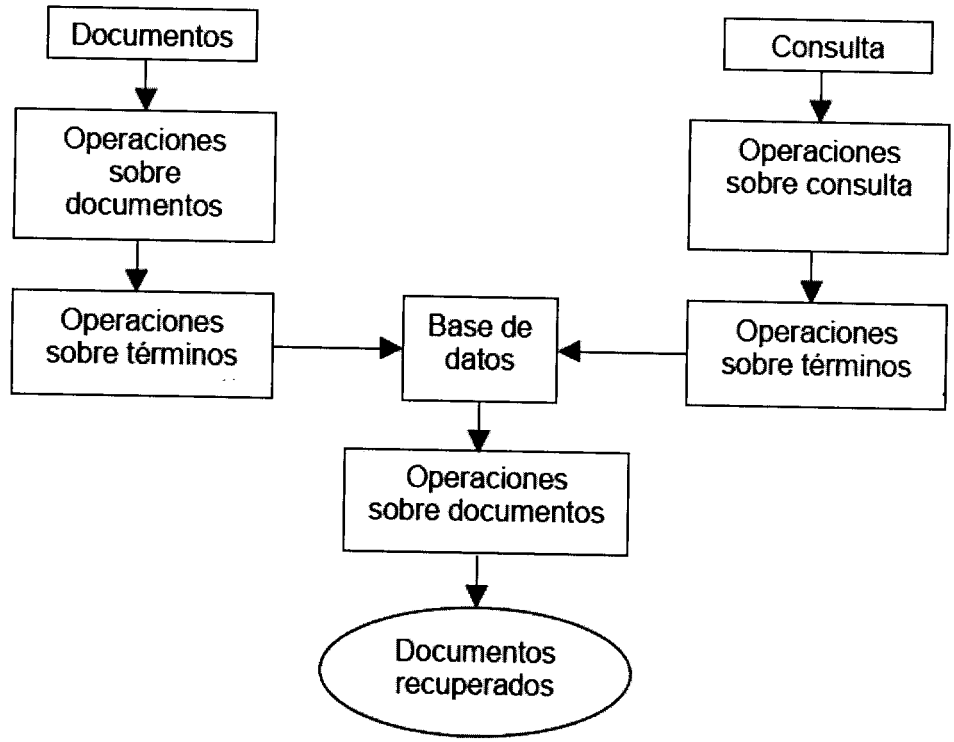


Figura 1.1 *Fases del tratamiento de la información.*

La indexación consiste en obtener representaciones de los documentos que componen la colección. En primer lugar, hay que recorrer todos y cada uno de ellos buscando las características más relevantes de los mismos para su posterior empleo en el modelado y comparación con la consulta realizada.

La representación de los documentos debe ser lo más breve posible, con poca redundancia, relacionada semánticamente con el documento y manteniendo la especificidad (es decir, manteniendo las características propias que permiten distinguir un documento de otro). Para ello, es generalmente admitido que la frecuencia de asignación de cada característica debe ser moderada: si la frecuencia de asignación es alta para todos los documentos de la colección, la característica no permitirá diferenciar unos de otros, y si es demasiado baja, no tendrá efecto estadísticamente.

En los sistemas de recuperación de documentos textuales las características del texto se escogen dependiendo de la funcionalidad requerida por el sistema. Si, por ejemplo, el sistema se piensa para realizar búsquedas bibliográficas en una biblioteca, cada documento estará caracterizado por su título, autor, editorial, año de publicación, ISBN,

formato, precio, etc. y éstos serán los términos de indexación escogidos. Si, por el contrario, se piensa en hacer búsquedas sobre el contenido de los documentos (por ejemplo, manuales técnicos), los términos de indexación serán, en este caso, todas las palabras que contienen [1].

1.3 Los buscadores.

La gran cantidad de información que se engloba en la Red Internet presenta, paradójicamente, el inconveniente de la dificultad que supone su localización. La primera medida que se tomó para paliar este inconveniente fue la de incorporar a la Red los servidores DNS, que transforman las direcciones URL en IP y viceversa. Aún así, el problema no estaba solucionado. Surgen, pues, los primeros servicios de búsqueda de información en la Red, todos ellos históricos en la actualidad, como Archie, Verónica, Hytelnet, Wais, etc. Su uso hoy es muy marginal, aunque siguen funcionando, puesto que la información que sirven está integrada en los actuales buscadores, de potencia mucho mayor y manejo más simple. Esto se debe al desarrollo del más popular de los servicios de Internet, la WWW, para la que no estaban preparados los antiguos servicios de búsqueda de información.

1.3.1 Primeros instrumentos de búsqueda.

Archie: buscar ficheros

Es una herramienta que permite localizar programas y los principales sitios FTP anónimos de Internet. Está formada por ficheros de fondos bibliográficos contenidos en los servidores de FTP con la ruta adecuada para llegar a ellos.

Hytelnet

Hytelnet (Hypertext browser for telnet-accesible sites) es un programa de hipertexto que contiene las direcciones Internet y otros datos accesibles vía telnet. Este programa se utiliza para ayudar a localizar las direcciones vía telnet de:

- Catálogos de acceso público en línea (OPACs)
- Bases de datos y bibliografías
- Servicios de correo electrónico
- Revistas electrónicas
- Servicios de información general de universidades
- Libros en formato electrónico
- Servidores (Archie, Gopher, Wais, WWW)
- Servicios directorio y páginas amarillas
- Acceso a otras redes
- Cualquier otro servicio accesible en Internet

Verónica

Verónica (Very Easy Rodent-Oriented Net-wide Index to Computerized) es una herramienta que permite acceder a la información contenida en la mayoría de los menús de un servidor Gopher, a través de preguntas por palabras clave y con el empleo de operadores booleanos. Las respuestas pueden limitarse a un número de datos o a algunos tipos de documentos [2].

1.3.2 Herramientas de búsqueda en Internet.

Dado que actualmente existen miles de nodos web, la dificultad para encontrar los que nos interesan sería enorme si no fuera por las herramientas de búsqueda que se han ido desarrollando, entre las que destacan los *directorios* y los *motores de búsqueda*, que facilitan el acceso a la información y a los documentos que existen en la red. Además de estos motores y directorios tenemos como herramientas de búsqueda las *fuentes de información* (bases de datos, bibliografías, biografías, boletines oficiales, catálogos, congresos, diccionarios, directorios, enciclopedias, periódicos, revistas, tesauros y tesis doctorales.

Un motor de búsqueda es una herramienta web que localiza de forma rápida la información en Internet y está formado por tres elementos bien diferenciados: un *interface* (página web a la que accede el usuario y en la que realiza la búsqueda), un *robot* (programa que recorre la web analizando páginas web) y una *base de datos* (Índice de

palabras, frases y datos asociados con la dirección URL de las páginas web). Existe un gran número de términos para referirnos a estas herramientas (robot, spiders, search engines, agents, rastreadores, motores de búsqueda, buscadores, índices, directorios...). El término motor de búsqueda es una traducción libre del americano search engines y que se puede identificar también con la palabra "buscadores".

Entendemos por un "**buscador**" en Internet una dirección más de la WWW que, mediante el uso de una base de datos, ofrece al usuario las direcciones URL de otras páginas o servicios, u otro tipo de informaciones en el caso de algunos buscadores especializados, atendiendo al criterio de búsqueda que se haya seleccionado. Además de ofrecer esas direcciones, permiten, en la mayoría de los casos, acceder a los recursos localizados mediante enlaces. El primero de estos buscadores hace su aparición en 1994 y se trata del buscador Yahoo.

1.3.3 Tipos de buscadores.

Hay tres tipos principales de buscadores en la Web: los índices temáticos (también conocidos como catálogos, directorios o buscadores por categorías), los motores de búsqueda (buscadores por contenido) y los buscadores híbridos.

Los *índices temáticos* mantienen su base de datos "manualmente", utilizando para incluir las direcciones a sus propios empleados o a los internautas que dan de alta sus páginas. Se estructuran por temas o categorías principales que a su vez contienen otras subcategorías, y así sucesivamente hasta que al final ofrecen enlaces directos a otras páginas de Internet.

Los *motores de búsqueda* emplean un robot de búsqueda. Estos robots son potentes programas que recorren la web automáticamente recopilando e indexando todo el texto que encuentran, formando de esta manera grandes bases de datos en donde se efectúan las búsquedas mediante la inclusión de palabras clave. Los robots recorren los distintos servidores a través de los enlaces que proporcionan las páginas que allí se encuentran, descendiendo como si de un árbol se tratara a través de las distintas ramas de cada servidor. Periódicamente visitan de nuevo las páginas para comprobar si ha habido

incorporación o si las páginas siguen activas, para que su base de datos se mantenga actualizada.

Estos buscadores pueden efectuar sus búsquedas en una o varias bases de datos con lo que ofrecen a los usuarios posibilidades muy amplias de resultados. Así tenemos:

**Una única base de datos*: a este tipo pertenecen la mayoría de los buscadores de Internet, ya sean motores o índices. La información se localiza a partir de la propia base de datos. Como inconvenientes podemos señalar que sólo se encontrará lo que esté publicado en la misma, obviando las direcciones que, aún existiendo, no lo estén.

**Varias bases de datos no simultáneas (multibuscadores)*: buscan la información en varias bases de datos diferentes aumentando la posibilidad de encontrarla. Los multibuscadores realizan esta operación manualmente ya que ponen a disposición del usuario varias bases de datos, pero es él mismo quien debe elegir cuál de ellas desea utilizar en cada momento y realizar desde allí su consulta. El inconveniente es que hay que buscar de una en una ya que no permite buscar en varias al mismo tiempo.

**Varias bases de datos simultáneas (metabuscadores)*: son buscadores múltiples que llevan a cabo la consulta solicitada por el usuario en varias bases de datos al mismo tiempo en un número no superior a diez. Si el sistema está bien depurado, el metabuscador comparará los enlaces devueltos para evitar repeticiones y colocará en primer lugar los enlaces más repetidos, ya que se consideran más importantes al estar dados de alta en más buscadores.

Los *buscadores híbridos* son aquellos que disponen de ambos métodos (directorio y motor de búsqueda). Los índices para el buscador se captan por la araña, y el catálogo se llena por los moderadores del sistema. De forma tal que los usuarios puedan realizar las búsquedas por palabras claves o navegar a través de las distintas categorías hasta llegar a las páginas deseadas. Este tipo de sistemas aunque hereda las ventajas de cada uno también requiere de un gran costo de mantenimiento debido al factor humano que presenta. *Google* es un ejemplo claro de este tipo de buscadores [3].

1.4 Robots o Arañas de búsqueda.

Un robot es un programa que rastrea la Web (visitando recursivamente aquellos documentos enlazados), recogiendo información sobre las páginas que encuentra. Esa información se indexa y se introduce en una base de datos que será explorada posteriormente utilizando un motor de búsqueda. A este tipo de robots se les denomina también *spiders* (arañas) o *Web crawlers*. El robot actualiza la base de datos visitando periódicamente las páginas para comprobar si ha habido alguna modificación o si aún siguen activas; incluso, pueden "aprender" a visitar y reexaminar con mayor frecuencia aquellos servidores que cambian rápidamente o que son citados en otras muchas páginas. Tales robots pueden recoger más de 10 millones de páginas por día y actualizan la información tras un período que puede ir desde unos minutos hasta 3 meses.

El funcionamiento de los robots o arañas no siempre ha respondido a los mismos parámetros. Cada robot busca a su manera en la Web, de ahí que la información almacenada en cada base de datos sea diferente. Se calcula que la intersección de las bases de datos de los servicios de búsqueda más conocidos es tan sólo de un 2%.

Al principio sólo obtenían las palabras de determinadas partes del documento Web, como el título o la descripción; luego, del documento completo aplicando las operaciones sobre los términos estudiadas anteriormente; posteriormente comenzaron a hacer uso de la información recogida en las metaetiquetas HTML, etc.

Las arañas utilizan diversas estrategias para rastrear la red y recopilar información que, una vez indexada, entra a formar parte de la base de datos del buscador. Si bien las empresas o instituciones encargadas del mantenimiento de estos programas no suelen dar a conocer el algoritmo exacto que utilizan los robots para operar en el ciberespacio por ser secreto comercial, se sabe, al menos, que algunos utilizan un método probabilístico para viajar por la Web sin sobrecargar los servidores incluyendo en su base de datos páginas muy citadas o con las direcciones (*path*) más cortas, ya que se suponen que éstas corresponden a los directorios principales de una colección de recursos. Esta última premisa no siempre se cumple, con lo que el robot puede incurrir en imperdonables olvidos al no localizar documentos que quizás sean de gran importancia para los usuarios. Otros optan por realizar una primera búsqueda "por extensión" para asegurarse de que en su índice están al menos representados tantos servidores como sea posible, aunque no indexen todas sus páginas. Este tipo de robots cuando recuperan una URL, examinan sus enlaces directos o internos, es decir, los ubicados en ese mismo servidor, así como los

externos y crean una cola de espera para volver a examinar cada recurso o bien para obtener una muestra representativa de los mismos. Generalmente parten de una lista determinada y a partir de ahí, realizan un rastreo recursivo de los documentos que se referencian en un documento. Además hay robots que también utilizan, complementariamente, listas de URLs que se han registrado en el servicio de búsqueda. Estas URLs pueden haber sido registradas manualmente por los usuarios a través de un formulario o automáticamente por servicios que se dedican a promover páginas o sitios Web.

1.4.1 Algoritmo general de un robot.

Analicemos un algoritmo general para recuperar documentos de la Web utilizado por los robots.

1. El algoritmo usa una lista de URLs conocida. Esta lista contiene al menos una URL con la cual empezar.
2. Se coge una URL de la lista (usando heurísticas que son distintas para cada robot), se verifica si ya había sido almacenada o si ha cambiado y se recupera el correspondiente documento de la Web.
3. El documento es analizado para recuperar información y los diferentes enlaces a otros documentos que se indexarán en la base de datos.
4. Las URL de los enlaces encontrados en el documento son añadidas a la lista de URL conocidas. El orden y la posición en las cuales las URL son añadidas a las listas difieren en cada robot.
5. Si la lista está vacía o se excede algún límite (número de documentos recuperados, tamaño del índice de la base de datos, tiempo transcurrido desde el comienzo, etc.), el algoritmo termina. En otro caso, el algoritmo continúa en el paso 2.

Los robots toman como punto de partida una URL inicial y recupera un fichero en formato HTML que transfiere al sistema local, de forma similar a como lo hace un cliente Web; pero una vez recuperado, en lugar de proceder a su visualización, el sistema se sirve de él para generar nuevos registros en una base de datos. Cada entrada de esta base de datos recogerá la URL completa del documento y una serie de palabras significativas extraídas, bien de los fragmentos con un mayor contenido informativo (<TITLE>, <H1>, etc.), a partir de su frecuencia de aparición en el documento, etc.

Una vez indexado el documento, el robot apoyado por el subsistema localizador y analizador identifica las referencias hipertextuales que contiene el documento y que dirigen a otras unidades informativas en el mismo o en otros servidores de la red; de forma recursiva, el robot procede a recuperar los documentos referenciados en estos nexos, procediendo a su indexación, obtención de nuevas referencias, etc.
Evidentemente, las estrategias de selección de *URL*s iniciales, extracción de contenido de los documentos y asignación de valores a estos términos de indexación están abiertas a numerosas posibilidades y cada implementación puede optar por distintas alternativas.
Cada vez que un robot llega a una página comprueba si ya la había visitado anteriormente o si es nueva para él. En el caso de que haya recogido la página, examina si ha sufrido modificaciones desde la última vez que la visitó y, en tal caso, actualiza la información que almacena sobre ella. El CINDOC (Centro de Información y Documentación Científica) calcula que la media de cambio de una página web es de 44 días.

1.4.2 Principio de exclusión de los Robots.

Con la intención de prevenir que los robots rastreasen zonas de la Web no deseadas, Martijn Koster creó en 1994 el denominado *Standard for Robot Exclusion*. Además, esto nace con el objetivo de evitar que un robot cause serios problemas e interfiera en la capacidad de respuesta del servidor WWW. Para ello no hay muchas posibilidades: la única forma que se puede garantizar la exclusión de un determinado agente consiste en la exclusión de esa máquina en la configuración del servidor. Se puede llegar a saber si un Robot ha accedido al servidor mediante la información registrada en los ficheros log (*User-Agent*); siempre que se encuentre un *User-Agent* que con más frecuencia de lo habitual, ha recuperado un elevado número de documentos en un corto espacio de tiempo se puede afirmar haber identificado un Robot y proceder a su exclusión si ha generado efectos colaterales en el rendimiento del servidor. Evidentemente, ésta no es una buena solución porque la aparición de nuevos Robots es impredecible y se podría causar efectos colaterales impidiendo el acceso de usuarios que acceden al servidor con otros fines.
La solución a estos problemas sólo puede obtenerse mediante la concientización de los administradores de este tipo de agentes, por lo que se han propuesto una serie de directrices así como el estándar para la exclusión de Robots que permita a los administradores de servidores Web proteger total o parcialmente su servidor de la acción de los Robots. El problema reside en que los encargados de implementar el protocolo son

los propios creadores de los Robots, por lo que a las potenciales "víctimas" no se les brindan muchas posibilidades.

El protocolo para la exclusión de Robots consiste en un acuerdo al que llegaron los miembros del foro electrónico robots-request@nexor.co.uk en junio de 1994 y, a pesar de su denominación, no se encuentra avalado por ningún tipo de organización de Standard. El protocolo se basa en la presencia de un fichero /robots.txt, localizado en el directorio raíz del servidor WWW. Los Robots que decidan corroborar y cumplir el protocolo, deberán leer la información registrada en este fichero para saber si su acceso está autorizado y si se ha establecido alguna restricción sobre algún conjunto de ficheros en la estructura de directorios.

La estructura del fichero /robots.txt debe ceñirse a las siguientes restricciones:

1. Cualquier comentario deberá aparecer precedido por el carácter #.
2. La información sobre restricciones se consignará mediante las etiquetas:
   - *User-agent:*
   - *Disallow:*
3. A continuación de *User-agent:* se consignará el identificador del robot que se desea excluir; el carácter * se utiliza a modo de comodín para referenciar la totalidad de los agentes a los que no se ha hecho mención particular. Se debe tener presente que la cadena de caracteres especificada tras *User-agent:* será interpretada por los agentes como una subcadena, es decir, la cadena Sodion restringirá el acceso a cualquier robot cuya identificación incluya esa subcadena: Sodion, SodionPlus, etc.
4. Después de *Disallow:* se establecerá qué conjunto de ficheros debe ser respetado.
   - Si después de *Disallow:* no se consigna ningún dato, el agente identificado en la etiqueta *User-agent* inmediatamente anterior no tendrá asignada ninguna restricción.
   - Si después de *Disallow:* se encuentra el caracter /, el robot tendrá restringido el acceso a cualquier directorio dependiente de la raíz, incluido éste, es decir, no podrá acceder a nuestro servidor.
   - Si se quiere evitar que el Robot acceda a un determinado directorio, tras la etiqueta *Disallow:* se especificará el camino completo hasta ese directorio.

A pesar de su amplia discusión teórica, la aplicación de la *Norma para la Exclusión de Robots* es más bien reducida, ya que sólo una mínima parte de los servidores visitados por los robots contienen el fichero /robots.txt. Esto puede deberse a:

- Un desconocimiento de la norma por parte de quienes mantienen los servidores conectados a la red y de los creadores de recursos Web.
- Que realmente no sea necesario vetar el acceso de las arañas a ninguno de los ficheros de documentos.
- Que la norma sea un tanto ambigua y confusa y que sea necesario estudiar a fondo su eficacia y mejorarla.

Una alternativa al uso del fichero robots.txt es la etiqueta <META> de la siguiente manera: <META NAME="ROBOTS" CONTENT="NOINDEX, NOFOLLOW"> la cual indica que tal documento no se ha de indexar ni analizar para seguir los enlaces [1].

1.4.3 Limitaciones de los robots.

No obstante, los robots presentan actualmente las siguientes limitaciones:

• La obtención de páginas objeto de indexación se verá restringida y condicionada por las páginas seleccionadas inicialmente.

• La acción de los Robots puede generar una sobrecarga y saturación en los servidores WWW de los que extrae los documentos, produciendo serios problemas en el rendimiento del servidor.

• Los Robots generan también sobrecargas insospechadas en la infraestructura de la red de comunicaciones. La medida de la cantidad de información que puede transmitirse físicamente a través de una línea de transmisión en un período de tiempo determinado se denomina *ancho de banda* y, normalmente, se mide en *bits por segundo* (bps). Para trabajar los robots necesitan un ancho de banda considerable porque actúan de forma continuada en períodos de tiempo prolongado y porque, para acelerar el proceso, muchos robots realizan una recuperación en paralelo, de tal modo que incluso partes remotas de la red pueden acusar excesiva tensión si el robot hace un gran número de recuperaciones en un breve período, lo que se conoce como *fuego rápido* (*rapid fire*). Este bombardeo al que se ven sometidos los servidores provoca una escasez temporal de ancho de banda para otros usos y usuarios.

• En el diseño e implementación de los robots debe ponerse especial cuidado en evitar errores o problemas en la ejecución del programa que perjudiquen a otros participantes en el proceso de búsqueda y recuperación de información, desde los servidores hasta los usuarios.

• Insuficiente calidad en la selección e indexación: cualquier procedimiento de indexación automático es incapaz de distinguir entre documentos con mayor o menor riqueza informativa.

• La información cambia con frecuencia y las arañas únicamente actualizan las bases de datos de los buscadores con cierta periodicidad. Muchas páginas web no son ficheros estáticos, sino que recogen información dinámica, perecedera y en constante cambio, lo cual dificulta el que puedan ser analizadas e indexadas por los programas.

• La justificación de sistemas de indexación mediante robots sería difícil si la red cambiara hacia un uso no gratuito de los recursos.

• El tipo de datos recogido por los robots no es útil, ya que aún hoy las arañas presentan un funcionamiento demasiado simple.

• Los sistemas de comprensión del lenguaje natural no están lo suficientemente avanzados como para extraer el significado de los recursos.

La indexación automática tiende a una perspectiva simplista, poco selectiva, que provoca que la localización de recursos en la red y la recuperación de la información solicitada llegue a ser cada vez menos factible.

• Frente a los indexadores humanos, los programas automatizados tienen dificultades para identificar características de un documento web como el contexto o temática general en la que se engloba y el género, por ejemplo, una comunicación científica, información profesional o informal, al que ese recurso pertenece.

• El WWW carece de reglas para facilitar la indexación automática. Los documentos no están estructurados de forma que los programas puedan obtener de modo fiable la información, conocida como metadatos - autor, título, longitud del texto, materia que un indexador humano detectaría fácilmente tras una rápida revisión.

• Los editores y/o creadores de estos recursos a veces abusan del carácter indiscriminado de la indexación automática. Un servidor web puede falsear el proceso de indexación con el fin de atraer la atención de los usuarios, mediante la repetición en el documento de una palabra como *sex*, muy utilizada en las búsquedas, aunque sea otro su contenido.

• Tanto la pérdida de semántica (a nivel de documento y de requerimiento de información) y el problema de la relevancia (recuperación de documentos irrelevantes y la no

recuperación de documentos relevantes) surgen como consecuencia de tener un espacio de términos de índices poco precisos y, muchas veces, de la formulación de consultas muy pobres y/o defectuosas por parte de los usuarios, como por ejemplo, sucede en las consultas web.

1.4.4 Estrategias a seguir para el diseño y uso de un robot de búsqueda.

Para intentar minimizar estos problemas, en 1993 Koster enunció unas directrices, *Guidelines for Robots Writers* (http://info.webcrawler.com/mak/projects/robots/guidelines.html), donde, a modo de orientación, se indicaba a sus creadores los daños que podían causar al lanzar su robot a la red.

El documento, que tuvo amplia difusión, ofrecía algunas sugerencias como las siguientes:

• Reconsiderar la necesidad de un nuevo robot. Los robots consumen recursos a nivel mundial, quizás ya haya uno que realice ese mismo trabajo y que se pueda utilizar sin necesidad de crear uno nuevo. Si finalmente se decide a llevar a cabo el proyecto, no debe plantearse recorrer la Web completa, sino que sólo llegue a unos pocos niveles de profundidad.

• Ser responsable: anunciarse, identificarse e informar. Quienes mantienen los servidores deben poder identificar al robot y contactar fácilmente con el responsable del mismo. Además, un nuevo robot debe anunciarse enviando un mensaje al foro *Usenet comp.infosystems.W3.providers* o a la lista de correo robots@nexor.co.uk o solicitar su integración en la base de datos de robots activos *The Web Robots Database* de Koster (http://info.webcrawler.com/mak/projects/robots/active.html). Por último, hay que informar a los gestores de los sistemas, ya que quizás sean los primeros en detectar algún error en el funcionamiento del robot y puedan, así, comunicárselo a su creador.

• Realizar comprobaciones previas en ficheros de datos locales. Para comprobar la efectividad de un prototipo que puede tener un comportamiento erróneo, se deben probar exhaustivamente en los servidores locales antes de lanzarlo a la red.

• Moderar el consumo de recursos. Una buena política que se puede seguir es evitar el fuego rápido y eliminar recuperaciones redundantes. Moderar la velocidad y frecuencia de acceso a cualquier servidor y recuperar únicamente aquello que realmente pueda gestionar –por el tipo y por el volumen de datos – son medidas razonables que todos los robots deberían respetar.

• Supervisar el funcionamiento. El responsable del robot debe analizar continuamente las conexiones a los servidores, corregir los posibles errores y estar preparado para responder y actuar cuando sea necesario.

• Compartir los resultados. La información generada por el robot debe ser de carácter público y estar a disposición de cualquier usuario. Los robots utilizan gran cantidad de recursos, lo que se justifica si el resultado de sus operaciones pueden disfrutarlo todos aquellos que sufren las consecuencias negativas, es decir, todos los usuarios de la red. Deben informar a los administradores de los servidores de los enlaces hipertextuales obsoletos y deben ofrecer la posibilidad de consultar los recursos que, en el transcurso de su actividad, hayan localizado.

Aunque el número de buscadores ha crecido considerablemente, los problemas que en ellos se encuentran siguen sin resolverse en su totalidad, por lo que es necesario seguir investigando en esta temática.

1.5 Sistemas distribuidos. Aplicación.

La computación desde sus inicios ha sufrido muchos cambios, desde los grandes ordenadores que permitían realizar tareas en forma limitada y de uso un tanto exclusivo de organizaciones muy selectas, hasta los actuales ordenadores ya sean personales o portátiles que tienen las mismas e incluso mayores capacidades que los primeros y que están cada vez más introducidos en el quehacer cotidiano de una persona.

Los mayores cambios se atribuyen principalmente a dos causas, que se dieron desde las décadas de los setenta:

1. El desarrollo de los microprocesadores, que permitieron reducir en tamaño y costo a los ordenadores y aumentar en gran medida las capacidades de los mismos y su acceso a más personas.
2. El desarrollo de las redes de área local y de las comunicaciones que permitieron conectar ordenadores con posibilidad de transferencia de datos a alta velocidad.

Es en este contexto que aparece el concepto de "Sistemas Distribuidos" que se ha popularizado tanto en la actualidad y que tiene como ámbito de estudio las redes como

por ejemplo: Internet, redes de teléfonos móviles, redes corporativas, redes de empresas, etc [4].

1.5.1 Concepto y características.

Definición:

"Sistemas cuyos componentes hardware y software, que están en ordenadores conectados en red, se comunican y coordinan sus acciones mediante el paso de mensajes, para el logro de un objetivo. Se establece la comunicación mediante un protocolo prefijado por un esquema cliente-servidor" [4].

Características:

- Concurrencia.- Esta característica de los sistemas distribuidos permite que los recursos disponibles en la red puedan ser utilizados simultáneamente por los usuarios y/o agentes que interactúan en la red.
- Carencia de reloj global.- Las coordinaciones para la transferencia de mensajes entre los diferentes componentes para la realización de una tarea, no tienen una temporización general, esta más bien distribuida a los componentes.
- Fallos independientes de los componentes.- Cada componente del sistema puede fallar independientemente, con lo cual los demás pueden continuar ejecutando sus acciones. Esto permite el logro de las tareas con mayor efectividad, pues el sistema en su conjunto continua trabajando.

1.5.2 Ventajas y desventajas.

Ventajas.

- Economía, es muy barato añadir servidores y clientes cuando se requiere aumentar la potencia del procesamiento.
- Trabajo en conjunto, por ejemplo: en una fábrica de ensamblado, los robots tienen sus CPUs diferentes y realizan acciones en conjunto, dirigidos por un sistema distribuido.
- Confiabilidad, al estar distribuida la carga de trabajo en muchas máquinas la falla de una de ellas no afecta a las demás, el sistema sobrevive como un todo.

- Crecimiento incremental, se puede añadir procesadores al sistema incrementando su potencia en forma gradual según sus necesidades.

Desventajas.

- Diseño e implantación del software distribuido.
- Redes de comunicación, Ej.: saturación del tráfico, pérdida de mensajes, etc.
- Al compartir datos peligra la seguridad de los mismos.

1.6 Objeto de estudio.

El objeto de estudio de este trabajo es un mecanismo automático de recopilación de información de los sitios web con dominio .cu, que serán parte del buscador de CubaSi.

1.6.1 Descripción del proceso actual.

Actualmente la mayoría de las búsquedas de información en nuestro país se realizan mayormente por buscadores internacionales, mediante Google, que tiene un dominio local en nuestro país, Yahoo, AltaVista, etc. a pesar de la existencia de algunos directorios nacionales, éstos no cumplen con las exigencias de nuestros internautas.

1.6.2 Situación problémica.

Actualmente en Cuba existen aproximadamente unos 400 sitios y unas 20 000 webs, distribuidas en distintas categorías, donde las artes y humanidades y el turismo son las categorías que más páginas aportan a la red cubana. Además existen 4 portales generales, 15 temáticos y 13 portales provinciales. Los portales generales son:

- Islagrande. Portal Cubano en Internet. http://www.islagrande.cu
- CUBAWEB. http://www.cubaweb.cu
- Portal CubaSí. http://www.cubasi.cu
- Cuba.cu. http://www.cuba.cu

Entre los principales portales temáticos están INFOMED (http://www.infomed.cu) y CUBARTE (http://www.cubarte.cu). [5]

Actualmente los usuarios de la red cubana cuentan con directorios, estos tienen como principal dificultad que son actualizados manualmente, lo que implica que en caso de surgir un nuevo sitio o portal cubano habría que esperar por los moderadores del directorio para la inclusión de este.

Uno de los objetivos del portal CubaSI es incluir un buscador web, para esto se necesita un robot de búsqueda que sea capaz de indexar la mayor cantidad de páginas cubanas.

1.7 Sistemas automatizados vinculados al campo de acción.

El portal CubaSI solo presenta un buscador interno de su sitio, no tienen implementado ningún robot de búsqueda ni tampoco un buscador externo.

1.8 Soluciones existentes.

Luego de realizar una búsqueda de información sobre otros robots de búsqueda, se llegó a la conclusión de que en el mundo existen numerosos de estos robots. La mayoría son muy caros y otros simplemente no se adaptan a la necesidad de información que se requiere.

Los más conocidos son:

- GoogleBot. Robot del buscador Google.
- Scooter. Robot del buscador AltaVista.

1.9 Propuesta de solución

Después de realizar un análisis sobre la situación actual sobre el objeto de estudio que tiene este trabajo, se concluye que se hace necesario implementar un sistema que pueda garantizar todas las funciones de un robot de búsqueda para el portal CubaSI.

Existen muchos robots de búsqueda, la mayoría son caros, y no se ajustan a las funcionalidades requeridas por CubaSI, por esto se ha decidido efectuar la confección de dicho sistema automático, de forma que garantice seguridad, confiabilidad y que sea el punto de partida de todo el buscador.

La aplicación se integrará junto con un motor de búsqueda, un directorio y un sistema de publicidad asociado al buscador, conformando así un buscador híbrido que es el objetivo principal del portal CubaSI.

1.9.1 Propuesta de robot distribuido.

Actualmente los grandes robots de búsqueda, como por ejemplo el robot de Google o el de AltaVista, son sistemas distribuidos, ya que al ver las ventajas de estos tipos de sistemas los robots se convierten en sistemas más robustos y eficaces, además de que la velocidad con que ejecutan sus funciones se multiplica a medida que aumenten los procesadores.

Para la propuesta actual, no es necesario un software distribuido, ya que la dimensión de la intranet de Cuba no exige tantos recursos, pero para un posterior estudio de una nueva versión que abarque mucho más que la actual, se recomienda la utilización de estos tipos de sistemas.

1.10 Fundamentación de los objetivos que se propone el trabajo.

Para darle respuesta a la situación problémica planteada, se propone para este trabajo un conjunto de objetivos:

1.10.1 Objetivos generales.

El objetivo es brindar una propuesta general integral, para el diseño e implementación de un robot indexador de sitios web, además de proponer una versión de robot distribuido para un posterior seguimiento del estudio de esta propuesta.

1.10.2 Objetivos específicos.

Este trabajo tiene los siguientes objetivos específicos:

- Hacer un estudio de las técnicas disponibles en el mercado acerca de la indexación de sitios webs para elegir la más adecuada a las condiciones del sistema de intranet cubano.

- Brindar información sobre el estado del sistema y los datos recogidos, mediante reportes.
- Confección de un sistema para indexar sitios webs en Cuba.
- Almacenar toda la información que se obtenga de los sitios webs cubanos.

1.11 Conclusiones.

En este capítulo se detallaron las condiciones y problemas que rodean al objeto de estudio; y a través de los conceptos y definiciones planteadas, se determinaron las condiciones específicas que rodean al problema y en base a esto se obtuvieron los objetivos generales y específicos para este trabajo.

Capítulo 2

TENDENCIAS Y TECNOLOGÍAS ACTUALES A CONSIDERAR.

2.1 Introducción.

En este capítulo se hace un análisis de las tendencias y tecnologías en la arena internacional en el desarrollo de robots de búsqueda. Se analizan las herramientas a utilizar y la metodología que se va a seguir y el lenguaje en que se va a implementar la propuesta.

2.2 Estrategias de navegación.

La estrategia de navegación es el proceso de devolver una URL de la lista de ellas a recuperar y de las que se va a servir el Robot para recuperar de Internet. De la misma forma, involucra el añadir nuevas direcciones (procedentes del sistema de análisis de un documento) y situarlas en su lugar correspondiente en la lista.
Ante la aparición de nuevas URLs encontradas, existen diversas estrategias:

- La estrategia *primero en profundidad*. Las URLs son añadidas al principio de la lista. Da en conjunto la mejor distribución de URLs en la Web, lo cual es importante solamente cuando se quiere recuperar una pequeña porción de documentos. Plantea problemas al robot en el aspecto que se puede encontrar con ciclos, cuando, por ejemplo existen documentos que se referencian a sí mismos o que se vuelven a referenciar por medio de documentos que han salido

de él. Este es un aspecto que tiene que tener en cuenta el Robot para no caer en ciclos infinitos.

- La *estrategia primero en anchura.* Las URLs son añadidas al final de la lista. Cuando es usada con una lista inicial de servidores registrados, se pueden obtener resultados excelentes al principio, porque se pueden alcanzar muchos servidores. En general, esta estrategia es menos eficiente para explorar la Web más allá de los puntos de partida.

  También se puede dar el caso de que gran parte de los documentos que se estén analizando en un determinado momento pertenezcan al mismo servidor, para lo cual se establecen múltiples conexiones en poco tiempo lo que puede acarrear graves problemas de rendimiento a los servidores.

- Las *estrategias manuales.* No se pueden considerar como una fuente importante pero, a veces, es necesario para poder llegar a distintos puntos de la Web que no son accesibles a través del robot. Éstas son:

  - *Administrador*: el propio administrador del buscador puede introducir manualmente nuevos enlaces de interés.
  - *Usuarios registrados y temporales*: Ciertos usuarios pueden tener permisos para introducir sus URLs por medio de unos formularios que deben cubrir y enviar al administrador el cual se encarga de añadirlos a su lista de enlaces [1].

2.3 Métodos de indexación.

2.3.1 Concepto de indexación.

El sistema es el encargado de la indexación de todos los documentos obtenidos.

La indexación, es la operación destinada a representar los resultados del análisis del contenido de un documento o de una parte del mismo, mediante elementos (denominados genéricamente *términos de indexación*) de un lenguaje documental o natural, generalmente para facilitar la recuperación.

La indexación puede realizarse utilizando diversos métodos:

- Indexación manual por vocabulario controlado: consiste en la asignación por expertos de los términos de indexación a partir de un conjunto finito de los mismos.
- Indexación automática con vocabulario controlado: automáticamente se escogen los términos de los existentes en un conjunto finito.
- Indexación libre: son los propios autores de los textos los que asignan los términos de indexación pensando en los términos que utilizarían los usuarios si buscasen información sobre los temas de sus textos.
- Indexación a través del lenguaje natural: los términos de indexación los elige la computadora a partir de un análisis de los textos. Este último tipo aparece hoy en día debido a la gran variedad de textos disponibles en formato electrónico y es el centro de atención en las investigaciones actuales en Recuperación de Información.

El método más empleado en este tipo de indexación es tomar como índices las palabras del texto. La gran dificultad de este enfoque es que si indexamos un texto basándonos en las palabras del lenguaje natural, podemos obtener decenas de miles de características, una cantidad más allá de la que podríamos caracterizar como óptima. Además, en una representación basada en palabras hay una redundancia considerable, en el sentido de que aparecerán conjuntos de características sinónimos o casi sinónimos. Si dos palabras son sinónimas podríamos pensar que deberían estar en los mismos documentos y añadirlas como características. Otro efecto perjudicial es la ambigüedad en su significado, entendiéndola como la existencia de dos o más significados para una misma palabra o expresión lingüística. Esto se puede ver como la disyunción entre dos o más conceptos inconexos o no relacionados. La ambigüedad debemos evitarla para las representaciones de los textos, puesto que es de suponer que es poco probable que todos los significados sean de interés en un texto determinado. Por ejemplo,"planta" tiene varios significados y es poco probable que a un usuario que quiera recuperar documentos en los que la palabra tiene uno de esos significados (por ejemplo: factoría), le agrade que aparezcan documentos en los que tiene los otros significados (por ejemplo: vegetal o parte inferior del pie).

A diferencia de los directorios, cuya indexación es intelectual o manual, la mayoría de los buscadores de la Web realiza una indexación automática.

2.3.2 Niveles de indexación.

Los buscadores utilizan distintos métodos para indexar los recursos que incorporan a su base de datos. La indexación puede plantearse en cuatro niveles:

• Submorfológico.
La *indexación en el nivel submorfológico*, esto es, sin análisis morfológico, sintáctico ni semántico, ofrece un método muy flexible para la recuperación. Así las fuentes dè información se indexan como patrones de bits (*bit patterns*) de manera que el texto, el sonido y las imágenes en movimiento, pueden indexarse y recuperarse usando la misma forma de representación. Algunas herramientas de consulta comienzan a incorporar sistemas como, por ejemplo, Excalibur Visual RetrievalWare, que ofrecen recuperación de imágenes y de textos.

• Por palabra clave.
La *indexación por palabra clave* es la forma más común de indexación de textos en la Web.
Se crean índices inversos de raíces y palabras clave, direcciones, ubicación y frecuencia de apariciones. Este enfoque, esencialmente morfológico y estadístico, basa la Recuperación de Información en la similitud formal de las palabras y las estadísticas de su presencia en documentos y colecciones de documentos.

• Por conceptos.
Existen varios procedimientos para construir bases de datos basadas en conceptos, algunas de ellas muy complejas y basadas en sofisticadas teorías lingüísticas y de inteligencia artificial. En otros casos, como Excite, se basan en una aproximación numérica, calculando la frecuencia de aparición de ciertas palabras significativas. A partir de análisis estadísticos el buscador determina qué conceptos aparecen juntos o relacionados en textos que se centran en un tema concreto. Mediante este sistema se pueden recuperar recursos que tratan un tema dado, incluso aunque las palabras incluidas en el documento no coincidan formalmente con las de la pregunta.

• Por hiperenlaces.

La Web, como conjunto de páginas HTML relacionadas mediante enlaces o hipervínculos, puede ser representado como un grafo, en el que cada página constituye un nodo y cada enlace puede considerarse un arco. Dado que los enlaces o hipervínculos parten de páginas o nodos concretos y apuntan a otra página concreta, se puede hablar de grafo dirigido, de manera que los arcos tienen un sentido o dirección determinada, independientemente de que los navegadores puedan tener utilidades de vuelta atrás o mecanismos similares.

Aunque las páginas HTML contienen, desde luego, más cosas, además de los enlaces o arcos, éstos constituyen elementos informativos de importancia, que pueden ayudar, además, a caracterizar dichas páginas. En efecto, los hipervínculos establecen una relación entre unas páginas y otras. De una manera intuitiva, se puede pensar que dos páginas que reciben enlaces desde los mismos nodos deben tratar acerca de los mismos o parecidos temas. *Páginas que apuntan o enlazan con los mismos nodos podrían ser más o menos similares en su temática o contenido.*

La utilización de los hipervínculos como términos de indexación de las páginas Web pueden ser usadas en la recuperación de dichas páginas, aplicando las mismas técnicas que se emplean habitualmente en Recuperación de Información basada en palabras o términos.

Esto permitiría eliminar las diferencias idiomáticas, dado el carácter multilingüe de la Web y reducir significativamente la capacidad de proceso y almacenamiento necesarios.

2.3.3 Operaciones sobre los términos.

Existen diferentes enfoques para considerar los términos de indexación. Están los que consideran elementos de indexación a las palabras que aparecen en el texto, mientras que otros modelos toman como elementos de indexación aquellos correspondientes a entidades distintas de las palabras.

No todas las palabras o términos que componen un documento deben ser considerados términos de indexación. Los términos sufren una serie de transformaciones antes de incluirse en el índice.

Si se toman como elementos de indexación las palabras que aparecen en el texto analicemos, entonces, las operaciones que pueden realizarse sobre los términos.

- Identificación de los signos de puntuación y espaciados, eliminación de los acentos (uno de los eternos problemas en el procesamiento del lenguaje natural en español), reducción de las mayúsculas y las minúsculas, reconocimiento del formato del documento (por ejemplo, si es una página HTML se eliminan las etiquetas), reconocimiento de las palabras, etc. Cuando termina esta etapa tenemos el texto plano y las palabras identificadas en él.
- Eliminación de palabras vacías (*stop words*[8]).
  La lista de palabras vacías o antidiccionario es una relación de términos considerados como valores no indexables, usados para eliminar potenciales términos de indexación. Los términos de una lista vacía están carentes de todo significado a la hora de recuperar información, como, por ejemplo, el artículo "la" no posee ninguna funcionalidad a la hora de recuperar documentos, ya que en todos los documentos de la base de datos aparecerá este término de forma casi segura y no resalta nada el contenido del documento almacenado. Así, cada término potencial de indexación es comprobado previamente, verificándose su presencia en la lista de palabras vacías y es descartado si se encuentra en ella. Esta lista de parada está formada por las preposiciones, conjunciones, artículos, pronombres, así como aquellas palabras que no son discriminatorias por su elevada frecuencia de aparición en el conjunto de documentos de la Web. Algunos trabajos incluyen los verbos dentro de la lista de parada, aunque esto no está generalizado, pues en ocasiones los verbos sí tienen significación. Con la eliminación de las palabras vacías se logra una reducción del documento entre un 30 y un 50%.

- Identificación de estructuras multipalabra, como por ejemplo, frases sustantivas, nombres propios, etc.
- Extracción de raíces o lemas (*stemming*).

[8] Son aquellas palabras que son muy comunes en un lenguaje y por tanto son inútiles para hacer búsquedas o ser indexadas. Ejemplo de estas palabras en español serían: de, en, el, tu, los, un, etc.

Los algoritmos de extracción de raíces se encuentran orientados a obtener un único término a partir de diferentes palabras que constituyen esencialmente variaciones morfológicas con un mismo significado. Como, por ejemplo, se puede considerar la obtención del término "niño" a partir de "niños" y "niñita". El resultado del algoritmo debe ser una misma forma canónica para las diferentes variaciones morfológicas de una palabra, que no tiene por qué ser, necesariamente, la raíz lingüística. Este proceso comprende la eliminación de los plurales, de ciertos prefijos y sufijos, de las conjugaciones verbales y su reducción al infinitivo, etc. Un error que puede cometerse en la eliminación de sufijos es, por ejemplo, que queremos remover el sufijo "ual" de la palabra "factual" pero no de "equal". Para ello se aplican algunas reglas como, por ejemplo, si la raíz que queda tiene menos de 2 caracteres no eliminar el sufijo o que la raíz que resulta no termine en un cierto caracter.

Se pueden encontrar diferentes tipos de algoritmos de extracción de raíces. Como caso concreto, el algoritmo de Porter[9], utiliza un autómata de estados finitos e incluye 88 reglas de eliminación de sufijos. Las reglas se encuentran en 9 conjuntos, de forma tal que cada grupo de reglas se utiliza para la eliminación de un determinado tipo de sufijos, aplicándose los grupos de reglas en un determinado orden. En cada regla, se incluye, básicamente, una pareja de sufijos o terminaciones. Si se encuentra el primero de ellos en una palabra es sustituido por el segundo. La aplicación sucesiva de los grupos de reglas a una palabra, permite obtener el término en forma canónica que la representa.

Se pueden introducir dos tipos de errores en este tipo de algoritmos. El primero de ellos es el error de *infrarradicación* (*understemming*), que resulta de obtener diferentes formas canónicas para palabras que deberían proporcionar una misma por tener un mismo significado; por ejemplo para "hope" y "hopping" obtener HOP y HOPP, respectivamente. El segundo error es el de *sobrerradicación* (*overstemming*), que es su complementario: obtener la misma forma canónica para palabras que deberían tenerlas distintas, por diferir no en variaciones morfológicas, sino en su significado; por ejemplo, para "capital", "capitular" y "capitolio" obtener "capit".

La elección de un algoritmo concreto depende de la política seguida en la construcción de un sistema concreto. Para evitar errores en el proceso de

[9] Algoritmo de stemming desarrollado en 1980 por M. F. Porter.

indexación interesan algoritmos que minimicen la sobrerradicación y la infrarradicación. Algoritmos menos sofisticados que tiendan a la sobrerradicación y eviten la infrarradicación (se disminuye el número de índices) pueden adecuarse a sistemas en que aspectos de eficiencia en tiempo y espacio juegan un papel prioritario pero traen consigo una disminución en la efectividad del proceso.

- Truncamiento.

  El truncamiento es una "mezcla" manual de términos usando caracteres especiales en la palabra, por lo que el término truncado formará múltiples palabras; en este caso, se refiere a las operaciones de localización de términos con una raíz común, por ejemplo: la localización de todos aquellos documentos que comiencen por "informa", sería una búsqueda por truncamiento que ofrecería como resultado términos tales como: información, informaciones, informativo, informacional, informador, informadores, etc.

- Utilización de un tesauro.

  Un tesauro proporciona una agrupación o clasificación de términos en un determinado dominio o área en categorías denominadas clases. Pueden utilizarse diversos tipos de tesauros en la indexación automática o manual de los documentos. En una aproximación basada en el análisis automático de textos y consultas, la utilización del tesauro permite identificar términos lexicográficamente diferentes pero equivalentes semánticamente (sinónimos). De esta forma, se puede conseguir disminuir el problema que plantea la sinonimia en el proceso de recuperación: documentos que son adecuados a una consulta del usuario no se recuperan por no aparecer en ellos los términos de la consulta, aunque sí aparezcan sinónimos suyos.

  La utilización del tesauro se puede realizar durante el proceso de indexación de los documentos (utilizando como términos de indexación las clases del tesauro) o durante el análisis de las consultas (realizando una expansión de los términos de la consulta mediante sus equivalentes).

  La construcción manual de un tesauro es un proceso costoso. Existen métodos de construcción de tesauros semiautomáticos y completamente automáticos. Existen métodos de construcción de listas de asociación de términos, equivalentes a las clases del tesauro anteriormente comentadas, a partir del análisis de los

documentos. Las listas de asociación son conjuntos de palabras con significado parecido a efectos de recuperación. La suposición fundamental en que se basa la construcción de estas listas es que términos semejantes aparecen en contextos "semejantes".

De esta forma, términos que aparecen simultáneamente en un número importante de documentos resultan con valores de similitud altos, mientras que términos que no aparecen en los mismos contextos, resultan con valores de similitud bajos o nulos. A modo de ejemplo, si las palabras "carro" y "automóvil" aparecen a la vez en un gran número de documentos, se obtiene una similitud para ellas importante.

La formación de las asociaciones o grupos de términos se realiza a partir de sus valores de similitud. En la actualidad, los métodos automáticos no permiten la generación de tesauros de calidad equivalente a los construidos manualmente.

No necesariamente siempre se aplican todas las operaciones. Una vez que un elemento de indexación ha pasado el filtro de las palabras vacías, puede ir directamente al índice, o ser indexado después de sufrir algún tipo de transformación dirigida a aumentar su representatividad y a homogeneizar la base de términos a incluir en las consultas. Así se podría aplicar a los elementos de indexación la extracción de raíces o *stemming*, incluyendo en el índice la raíz de los términos a indexar en lugar de los términos mismos. También se pueden homogeneizar los términos de indexación considerándolos en mayúsculas o en minúsculas, etc.

En la figura 1 se muestra un texto de un documento correspondiente a una noticia de un periódico y los términos de indexación resultantes de aplicar las operaciones de preprocesamiento, eliminación de las palabras vacías, la identificación de estructuras multipalabras y la extracción de raíces.

Casi siempre que los elementos de indexación sufren una transformación, ésta debe ser realizada también con los términos que compongan las consultas, para que el comportamiento del sistema de Recuperación de Información sea el correcto.

PROTESTAS EN VENEZUELA POR LOS RECORTES DEL GASTO SOCIAL
L.WINOGRADOFF, Caracas

Los venezolanos han comenzado ya a notar la caída internacional de los precios del petróleo, principal fuente de ingresos del país. En lo que va de año, el Gobierno de Rafael Caldera ha tenido que recortar dos veces su capítulo de gastos, en total un 15% del presupuesto. Las protestas en la calle, que ayer comenzaron con estudiantes y trabajadores del sector de la salud, no se han hecho esperar, y su evolución es potencialmente explosiva, dado el grado de descontento de los venezolanos.

PROTESTAS VENEZUELA RECORTES GASTO SOCIAL
L.WINOGRADOFF Caracas

venezolanos comenzar notar caída internacional precios petróleo principal fuente ingresos país año Gobierno Rafael Caldera tener recortar dos veces capítulo gastos total 15% presupuesto protestas calle ayer comenzar estudiantes trabajadores sector salud no se han hecho esperar evolución potencialmente explosiva grado descontento venezolanos

(venezolanos [2], comenzar [1], notar [1], ... , descontento [1])

Figura 1 *Un ejemplo de extracción de términos de indexación.*

2.4 Tendencias actuales.

Tal como se trató en el capítulo anterior, las principales características de un robot de búsqueda son: sistema multihilos, rápido y eficaz tratamiento de cadenas de texto y manejo y almacenamiento de grandes volúmenes de información.

Entre los lenguajes que más han sobresalido por el auge que han alcanzado en esta rama están, Java, C, Python, etc.

2.4.1 Lenguaje C

El lenguaje de programación C está caracterizado por ser de uso general, con una sintaxis sumamente compacta y de alta portabilidad.
Es común leer que se le caracteriza como un lenguaje de "bajo nivel", lo correcto es decir de "medio nivel", es decir puede entenderse como lenguaje de máquina, con las ventajas del lenguaje casi común. No debe confundirse el término "bajo" con "poco", ya que el significado del mismo es en realidad "profundo", en el sentido que C maneja los elementos básicos presentes en todas las computadoras: caracteres, números y direcciones.

Esta particularidad, junto con el hecho de no poseer operaciones de entrada-salida, manejo de arreglo de caracteres, de asignación de memoria, operaciones multihebras, etc., puede al principio parecer un grave defecto; sin embargo el hecho de que estas operaciones se realicen por medio de llamadas a Funciones contenidas en Librerías externas al lenguaje en sí, es el que confiere al mismo su alto grado de portabilidad, independizándolo del "Hardware" sobre el cual corren los programas [6].

El robot de búsqueda de Google, GoogleBot está implementado en C, con algunas secciones de código assembler.

2.4.2 Lenguaje Java

El lenguaje de programación Java, fue diseñado con el propósito de crear un lenguaje que pudiera funcionar en redes computacionales heterogéneas (redes de computadoras formadas por más de un tipo de computadora, ya sean PC, MAC's, estaciones de trabajo, etc.), y que fuera independiente de la plataforma en la que se vaya a ejecutar. Esto significa que un programa de Java puede ejecutarse en cualquier máquina o plataforma.

El lenguaje Java se diseño con las siguientes características:

- Simple. Elimina la complejidad de los lenguajes como "C", mantiene las facilidades básicas de lenguaje en un mínimo y proporciona un gran número de extras con las librerías de clases.
- Familiar. La sintaxis de Java es muy similar al lenguaje C++, además se han eliminado ciertas características, como los punteros.

- Robusto. El sistema de Java maneja la memoria de la computadora por lo que no es necesario preocuparse por apuntadores, memoria que no se esté utilizando, etc.
- Seguro. El sistema de Java tiene ciertas políticas que evitan se puedan codificar virus con este lenguaje. Existen muchas restricciones, especialmente para los applets, que limitan lo que se puede y no puede hacer con los recursos críticos de una computadora.

- Portable. Como el código compilado de Java (conocido como byte code) es interpretado, un programa compilado de Java puede ser utilizado por cualquier computadora que tenga implementado el interprete de Java.
- Multithreaded (multihilos) El lenguaje soporta la concurrencia a través de hilos, o sea que puede ejecutar varios procedimientos en paralelo.
- Interpretado y compilado a la vez. Java es compilado, en la medida en que su código fuente se transforma en una especie de código máquina, los bytecodes, semejantes a las instrucciones de ensamblador. Por otra parte, es interpretado, ya que los bytecodes se pueden ejecutar directamente sobre cualquier máquina a la cual se hayan portado el intérprete.
- Dinámico. Java no requiere compilar todas las clases de un programa para que funcione, si se realiza una modificación a una clase, Java se encarga de realizar un Dynamic Bynding o un Dynamic Loading para encontrar las clases [7].

Java es un lenguaje multipropósito y uno de los más utilizados en el mundo de la informática, hay numerosos robots de búsqueda implementados en este lenguaje, entre ellos MOMSpider, VisualWebSpider, etc.

2.4.3 Lenguaje Python

Python es un lenguaje interpretado, orientado a objetos de propósito general. Python permite mantener de forma sencilla interacción con el sistema operativo, y resulta muy adecuado para manipular archivos de texto. Está disponible en MS-Windows, GNU/Linux, Mac y cualquier entorno.

Python es un lenguaje de scripts, sencillo pero potente. Hablamos de scripts pero... ¿qué es un script? Un script es un conjunto de instrucciones que se ejecutan paso a paso, instrucción a instrucción. Esto significa que Python no genera ejecutables, si no que es

Python el encargado de ejecutar nuestro código. Es por tanto un lenguaje interpretado, no compilado. Esto le dota de ventajas, pero también de algunos inconvenientes:

Ventajas

- Desarrollo más rápido: Puedes escribir un programa, salvarlo y ejecutarlo. En un lenguaje compilado tienes que pasar por los pasos de compilar y ligar el software, lo cual puede ser un proceso lento.
- Multiplataforma: El mismo código funciona en cualquier arquitectura, la única condición es que disponga del intérprete del lenguaje. No es necesario compilar el código una vez para cada arquitectura.

Inconvenientes

- Lentitud: Los programas interpretados son más lentos que los compilados. Sin embargo los programas interpretados suelen ser cortos, en los que la diferencia es inapreciable.

Python es uno de los lenguajes que tiene mejor tratamiento de cadenas de texto, por lo tanto es de gran utilidad para desarrollar sistemas de minería de textos o analizadores de cadenas [8].

2.5 Fundamentación del lenguaje a utilizar.

Se han expuesto muchas de las tendencias actuales en la construcción de robots, haciendo hincapié en las estrategias de navegación y en los métodos de indexación, luego de un estudio de cada una de éstas características, se ha decidido la estrategia de navegación ha utilizar será la de *primero en anchura*, ya que es muy eficiente y se asemeja al sistema de intranet cubano, dentro de los niveles de indexación se utilizará en el método de *indexación por palabra clave* ya que es la más utilizado actualmente y basa la Recuperación de Información en la similitud formal de las palabras y las estadísticas de su presencia en documentos y colecciones de documentos. Dentro de las operaciones

con los términos de indexación se aplicará la identificación de los signos de puntuación y la eliminación de las palabras vacías ("stop words").

CubaSI es un portal totalmente renovado y dinámico, desarrollado en .NET Framework, implementado en el lenguaje C#, por tanto se elige este lenguaje para el desarrollo del sistema, a continuación se explican algunas características de la plataforma y del lenguaje.

2.5.1 Plataforma .NET.

.NET Framework es una nueva plataforma informática que simplifica el desarrollo de aplicaciones en un entorno altamente distribuido como es Internet. El diseño de .NET Framework está enfocado a cumplir los objetivos siguientes:

- Proporcionar un entorno coherente de programación orientada a objetos, en el que el código de los objetos se pueda almacenar y ejecutar de forma local, ejecutar de forma local pero distribuida en Internet o ejecutar de forma remota.
- Proporcionar un entorno de ejecución de código que reduzca lo máximo posible la implementación de software y los conflictos de versiones.
- Ofrecer un entorno de ejecución de código que garantice la ejecución segura del mismo, incluso del creado por terceras personas desconocidas o que no son de plena confianza.
- Proporcionar un entorno de ejecución de código que elimine los problemas de rendimiento de los entornos en los que se utilizan secuencias de comandos o intérpretes de comandos.
- Ofrecer al programador una experiencia coherente entre tipos de aplicaciones muy diferentes, como las basadas en Windows o en el Web.
- Basar toda la comunicación en estándares del sector para asegurar que el código de .NET Framework se puede integrar con otros tipos de código.

.NET Framework soporta múltiples lenguajes de programación y, es posible desarrollar cualquier tipo de aplicación con cualquiera de estos lenguajes: C# (C Sharp), Visual Basic, C++, J# (Java #), Jscript, además, se han adaptado lenguajes existentes, como Perl, Phyton, Fortan, Cobol, entre otros.

La idea central detrás de la plataforma .NET es la de servicio. Más concretamente software como servicio y de cómo construir, instalar, consumir, estos servicios para que puedan ser accedidos mediante Internet [9].

2.5.2 Lenguaje C#.

C# es un moderno lenguaje orientado a objetos con el que los programadores podrán crear con facilidad una amplia gama de aplicaciones para la nueva plataforma Microsoft .NET, y que dispone de herramientas y servicios para aprovechar al máximo la informática y las comunicaciones.

La sintaxis y estructuración de C# es muy similar a la C++, ya que la intención de Microsoft con C# es facilitar la migración de códigos escritos en estos lenguajes a C# y facilitar su aprendizaje a los desarrolladores habituados a ellos. Sin embargo, su sencillez y el alto nivel de productividad son equiparables a los de Visual Basic.

El lenguaje proporciona la capacidad de generar componentes de sistema duraderos en virtud de las siguientes características:

- ✓ Es un lenguaje orientado a objetos.
- ✓ Total compatibilidad entre COM y plataforma para integración de código existente.
- ✓ Gestión automática de memoria, presenta un recolector de basura, esto significa que no es necesario incluir instrucciones de destrucción de objetos.
- ✓ Es orientado a componentes.
- ✓ El entorno en C# inicializa automáticamente las variables.
- ✓ Las variables disponen de seguridad de codificación estricta de tipos.
- ✓ Plena compatibilidad con conceptos de metadatos extensibles.
- ✓ Ofrece un amplio sistema de clases y librerías para el tratamiento de cadenas de texto.

Además, es posible interactuar con otros lenguajes, entre plataformas distintas, y con datos heredados, en virtud de las siguientes características:

- ✓ Plena interoperabilidad por medio de los servicios de COM+ 1.0 y .NET Framework con un acceso limitado basado en bibliotecas.

- ✓ Compatibilidad con XML para interacción con componentes basados en tecnología Web.
- ✓ Capacidad de control de versiones para facilitar la administración y la implementación.

2.5.3 Estudio de la concurrencia.

El sistema ha sido concebido para conseguir la mayor concurrencia posible, requisito indispensable para proveer un buen servicio interactivo. El ThreadPool de .NET proporciona un sistema concurrente para atender simultáneamente las peticiones de los distintos clientes. Gracias a esa concurrencia, se podrá aprovechar la potencia de un sistema multiprocesador para albergar el servicio de búsqueda de páginas web.

Este pool no sólo es accesible a las aplicaciones que quieran hacer uso de él, sino que también está integrado con la mayoría de las clases incluidas en el entorno. Por si esto fuera poco, gran parte de la funcionalidad de .NET utiliza directamente el mismo pool. Por ejemplo, *.NET Remoting* hace uso de él para atender las peticiones sobre objetos remotos.

Cuando una aplicación manejada por .NET es arrancada, el entorno pone a su disposición un único pool de hilos que será creado cuando el programa acceda a él por primera vez. Este pool estará asociado con el proceso físico donde resida nuestra aplicación, detalle importante si utiliza la posibilidad que ofrece .NET para ejecutar múltiples aplicaciones (denominadas "application domains") sobre el mismo proceso.

Cada uno de los vínculos que serán procesados estará representado por un hilo del "pool", así se garantizará una mayor eficiencia y rapidez en todo el proceso [10].

2.5.4 La manipulación de texto.

Como un robot de búsqueda debe hacer grandes análisis de texto, .NET Framework ha puesto a disposición de todos los desarrolladores de la plataforma un conjunto de clases especializadas en tratamiento de cadenas, éstas son expresiones regulares.

Las expresiones regulares son muy populares y son usadas comúnmente en aplicaciones escritas en Perl. Sin embargo, puede encontrar esta característica en el .NET Framework y en System.Text.RegularExpressions, que tiene clases especiales a tal efecto.

2.5.5 SQL.

Debido a la diversidad de lenguajes y de bases de datos existentes, la manera de comunicar entre unos y otras sería realmente complicada de gestionar a no ser por la existencia de estándares que nos permiten el realizar las operaciones básicas de una forma universal.

Es de eso de lo que trata el SQL (Structured Query Language) que no es más que un lenguaje estándar de comunicación con bases de datos. Hablamos por tanto de un lenguaje normalizado que nos permite trabajar con cualquier tipo de lenguaje en combinación con cualquier tipo de base de datos (MS Access, SQL Server, MySQL, etc.). Aquí tenemos una lista de algunas características proporcionadas por SQL que no forman parte del álgebra y del cálculo relacional:

- Comandos para inserción, borrado o modificación de datos.
- Capacidades aritméticas: En SQL es posible incluir operaciones aritméticas así como comparaciones, por ejemplo A < B + 3. Nótese que ni + ni otros operadores aritméticos aparecían en el álgebra relacional ni en cálculo relacional.
- Asignación y comandos de impresión: es posible imprimir una relación construida por una consulta y asignar una relación calculada a un nombre de relación.
- Funciones agregadas: Operaciones tales como *promedio (average)*, *suma (sum)*, *máximo (max)*, etc. se pueden aplicar a las columnas de una relación para obtener una cantidad única.

2.6 Fundamentación del gestor de BD.

Los principales objetivos de un Gestor de Base de Datos (SGBD) son: Evitar la redundancia de los datos, eliminando así la inconsistencia de los mismos, mejorar los mecanismos de seguridad de los datos y la privacidad.

Por la naturaleza de los datos que se almacenarán en el sistema propuesto y por la estrecha relación que existe con el .NET Framework, se ha decidido utilizar Microsoft SQL Server 2000 como gestor de base de datos, por ser una aplicación poderosa, robusta, que permite gran seguridad de los datos, ostenta marcas de referencia en cuanto a escalabilidad y confiabilidad, que son críticas para el éxito de bases de datos de gran tamaño. SQL Server permite lograr una gran velocidad en el procesamiento de transacciones, y agilidad en todas sus operaciones, además, es el utilizado por CubaSi.

2.7 Fundamentación de la metodología utilizada.

2.7.1 RUP (Racional Unified Process).

Para desarrollar un software se necesita de un método ordenado de trabajo. Un proceso que integre las múltiples facetas del desarrollo, que cumpla las siguientes características:

- ✓ Debe servir como guía a todos los participantes, clientes, usuarios, desarrolladores, etc.
- ✓ Proporcione normas para el desarrollo eficiente de software de calidad.
- ✓ Capture y presente las mejores prácticas que el estado actual de la tecnología permita.
- ✓ Debe ser capaz de evolucionar durante muchos años.

"El Proceso Unificado es un proceso de desarrollo de Software. Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. Sin embargo, el Proceso Unificado es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organización, diferentes niveles de aptitud y diferentes tamaños de proyecto...El Proceso Unificado está *basado en componentes*, lo cual quiere decir que el sistema software en

construcción está formado por componentes software interconectados a través de interfaces bien definidas." [11]

"El Proceso Unificado utiliza el *Lenguaje Unificado de Modelado* (Unified Modeling Language, UML) para preparar todos esquemas de un sistema software , De hecho, UML, es una parte esencial del Proceso Unificado – sus desarrollos fueron paralelos".[11]

Lo que hace único al Proceso Unificado (RUP) son los aspectos que lo definen, que están divididos en tres fases fundamentales, dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental.

2.7.2 UML (Unified Modeling Language).

"UML son las siglas de Unified Modeling Language (Lenguaje Unificado de Modelado), notación (esquemática en su mayor parte) con que se construyen sistemas por medio de conceptos orientados a objetos". [11]

El Lenguaje de Modelamiento Unificado (UML - Unified Modeling Language) es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables.

UML (Unified Modeling Language) permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos. Se ha convertido en el estándar de facto de la industria, debido a que ha sido impulsado por los autores de los tres métodos más usados de orientación a objetos: Grady Booch, Ivar Jacobson y Jim Rumbaugh. Con UML se fusiona forma una herramienta compartida entre todos los ingenieros de software que trabajan en el desarrollo orientado a objetos.

El Lenguaje Unificado de Modelado prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Mientras que ha habido muchas notaciones

y métodos usados para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación.

2.8 Herramientas utilizadas.

Para la confección de la aplicación se utilizaron diversas herramientas, **Visual Studio .NET 2005** por ser la herramienta más poderosa para programar cualquier aplicación en .NET, para la modelación de la aplicación se utilizó **Rational Rose**, líder en este propósito, así como **Embarcadero Studio** para la modelación de la Base de datos, este software permite modelar conceptualmente la base de datos y genera el código del modelo físico, minimizando las posibilidades de error.

2.9 La propuesta

Teniendo en cuenta los elementos antes expuestos de algunas de las herramientas disponibles y las tendencias actuales, se decidió implementar el sistema en C# y se utilizó Visual Studio .NET 2005 para el desarrollo de la aplicación, ya que permite aprovechar al máximo las ventajas de la plataforma .NET.

Como gestor de base de datos se utilizó Microsoft SQL Server 2000, por su capacidad para grandes volúmenes de información y fortaleza, y estar disponible en los servidores de CubaSI y ser la herramienta que utilizan.

Es importante resaltar que aunque Visual Studio .NET y Microsoft SQL Server no son gratis, el portal CubaSi tiene licencia para la utilización de dichas herramientas.

2.10 Conclusiones.

En este capítulo se profundizó en el conocimiento de algunos conceptos necesarios para la comprensión de este trabajo. Además se realizó un análisis completo de las tecnologías que serán utilizadas a lo largo del desarrollo del sistema propuesto, se fundamentaron las elecciones de lenguaje, sistema gestor de bases de datos, y la metodología a utilizar. Una vez conocidas las herramientas óptimas, y los conceptos a utilizar, se puede empezar a desarrollar la propuesta de sistema.

Capítulo 3

DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA

3.1 Introducción

En el presente capítulo se hace la descripción de la propuesta que trae este trabajo, para ello se describen los procesos del dominio que tienen que ver con el objeto de estudio. Además, se enumeran los requisitos funcionales y no funcionales que debe tener el sistema que proponemos, lo que permite hacer una concepción general del sistema, e identificar mediante un Diagrama de Casos de Uso, las relaciones de los actores que interactúan con el sistema, y las secuencias de acciones con las que interactúan.

3.2 Modelo de Dominio

Teniendo en cuenta que en el portal CubaSI solo existe un buscador interno, por tanto no tienen implementado ningún robot de búsqueda.

Para ello el trabajo está basado en un modelo del dominio, ya que permite de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio del sistema en desarrollo. Esto ayuda a los usuarios, clientes y desarrolladores e interesados, a utilizar un vocabulario común para poder entender el contexto en que se emplaza el sistema. Para capturar correctamente los requisitos y poder construir un sistema correcto se necesita tener un firme conocimiento del funcionamiento del objeto de estudio. Este

modelo va a contribuir posteriormente a identificar algunas clases que se utilizarán en el sistema. Primeramente se va a identificar todos los conceptos que se utilizarán en el diagrama, mediante un glosario de términos sobre los nombres:

- Se le denominará **Usuario** a cualquier persona que opere el sistema.
- Un **Servidor** es, como su nombre lo dice, un servidor de base de datos que será el que utilizará el sistema.
- Se considera un **Filtro** a la lista de sitios web que no se incluirán en los resultados del sistema, así como el dominio en el que se va a enmarcar.
- Se le denomina **Conexión** al tipo de conexión que tendrá el sistema, que puede ser directa o bajo un servidor Proxy.
- Se le denomina **Vínculo** a la dirección URL de una página web.
- Un **Documento** es la representación de los datos que contiene una página web, la dirección URL propia, los hipervínculos contenidos en ella, los metadatos, las palabras y el tamaño en Kbytes de la página.

El modelo del dominio se describe mediante diagramas UML, específicamente con un diagrama clases conceptuales significativas en el dominio del problema.

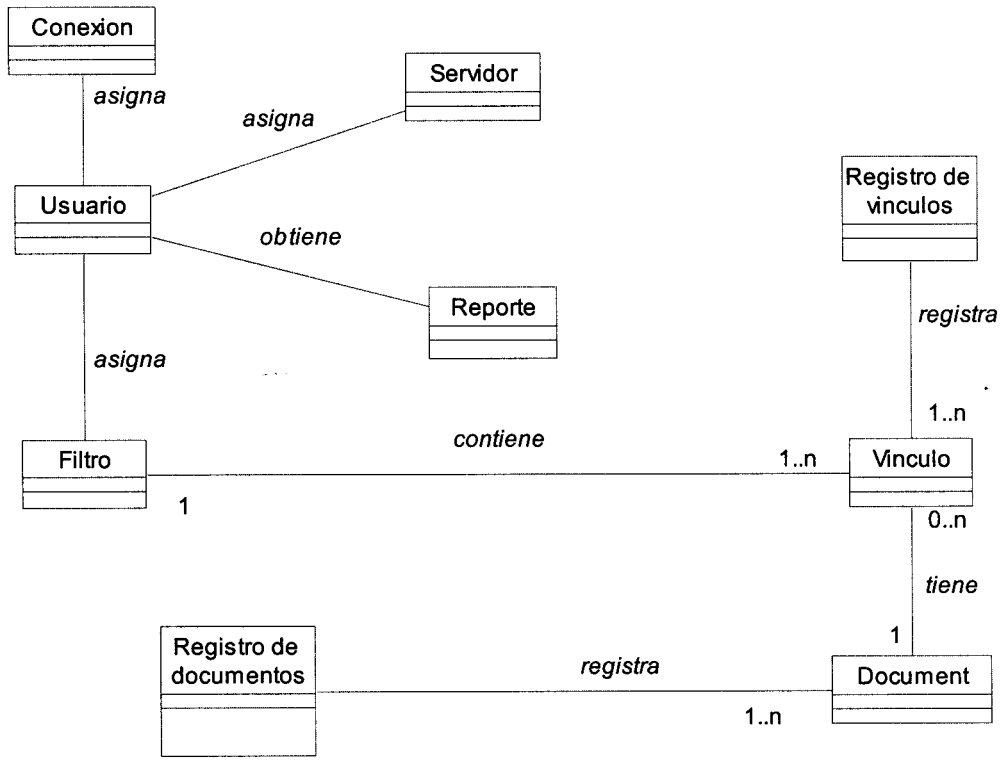


Figura 3.1. *Diagrama de clases del Modelo de Dominio.*

3.3 Requerimientos funcionales.

Una vez conocidos los conceptos que rodean al objeto de estudio, podemos empezar a analizar ¿Qué debe hacer el sistema para que se cumplan los objetivos planteados al inicio de este trabajo?, para ello se enumerará a través de requerimientos funcionales las funciones que el sistema deberá ser capaz de realizar. Dentro de ellos se incluyen las acciones que podrán ser ejecutadas por el usuario, las acciones ocultas que debe realizar el sistema, y las condiciones extremas a determinar por el sistema. De acuerdo con los objetivos planteados el sistema debe ser capaz de:

R1. Introducir vínculos iniciales.

1.1. Para introducir vínculos se necesita un vínculo o una lista de vínculos o URL.

1.2. Se registra el vínculo o la lista.

R2. Extraer los datos necesarios de un vínculo.

2.1 Se toma el primer vínculo disponible.

2.2 Conectarse al vínculo especificado.

2.3 Descargar su código fuente.

2.4 Extraer los siguientes datos:

- Titulo del vínculo.
- Meta datos.
- Hipervínculos.
- Palabras (sin incluir las "stop words")
- Todo el código fuente.
- Fecha de la última actualización.
- Tamaño en Kbytes.
- Tipo de servidor de aplicaciones web.
- Dominio web.
- Cuerpo del código HTML.
- Versión del protocolo de conexión.
- Lenguaje de programación.
- Método de descarga.

2.5 Se registran los vínculos enlazados a este vínculo.

2.6 Se registra el vínculo como un documento.

R3. Indexar un documento.

3.1 Se toma el primer documento disponible.

3.2 Se eliminan los signos de puntuación y las "stop words".

3.3 Para indexar un documento se necesita los siguientes datos:

- Titulo de la URL.
- Meta datos.
- Hipervínculos.

- Palabras (sin incluir las "stop words")
- Todo el código fuente.
- Fecha de la última actualización.
- Tamaño en Kbytes.
- Tipo de servidor de aplicaciones web.

R4. Mostrar reporte después de una indexación.

4.1 Un reporte debe contener la siguiente información.

- Cantidad de sitios con servidores Apache y Microsoft IIS.
- Cantidad de sitios que tienen menos de 10, entre 10 y 40, y mayores de 40 Kb.
- Todos los errores de conexión que existieron.
- Mostrar opciones para ver la cantidad de sitios actualizados desde una fecha dada.
- Mostrar opciones para ver la cantidad de sitios que tienen una cantidad de Kbytes dada.

R5. Asignar un filtro.

5.1 Para asignar un filtro se necesita el dominio web mediante el cual va a filtrar el sistema y el o los sitios que no serán procesados por el sistema.

R6. Configurar las opciones de conexión.

6.1 Mostrar las opciones de conexión: Conexión directa o Bajo un servidor Proxy.

6.1.1 Conexión directa.

- Se necesita saber el tipo de conexión directa.

6.1.2 Bajo un servidor Proxy.

- Se necesita saber el nombre o numero IP del servidor, usuario con que se conectara y su contraseña.

R7. Asignar un servidor de base de datos.

7.1 Mostrar las opciones para asignar el servidor de base de datos.

7.1.1 Mediante autentificación de Windows.

- Se necesita solo el nombre del servidor SQL.

7.1.2 Mediante autentificación de SQL Server.

- Se necesita el nombre del servidor, usuario y contraseña con que se conectará.

R8. Almacenar reporte.

8.1 El sistema debe ser capaz de almacenar el reporte generado después de una indexación en un mecanismo de almacenamiento persistente.

3.4 Requerimientos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

Interfaz externa

1. El sistema interactúa con el usuario mediante una interfaz de ventanas muy fácil de utilizar.

Rendimiento

2. Se requiere de un tiempo de descarga del código fuente de una URL no sea mayor de 45 segundos.
3. Hacer uso riguroso y eficiente de dispositivos externos conectados a la PC.

Portabilidad

4. El sistema deberá funcionar sobre plataforma Windows.

Hardware

5. Requiere estar instalada en una PC Pentium II, 256Mb de RAM y una tarjeta de red de 100Mbps pues el sistema en todo su tiempo de vida accede a la red.

Software

6. Se debe disponer en el servidor con Windows XP, Windows 2000 Server o 2000 Advanced Server. Se utilizará como lenguaje de programación: C# y como gestor de Base de Datos: SQL Server 2000.

Funcionalidad

7. Mínima cantidad de ventanas para ejecutar todas las funciones posibles.

3.5 Descripción del Sistema propuesto.

Para cumplir los objetivos propuestos al inicio de este trabajo, y teniendo en cuenta todos los requerimientos planteados, y para utilizar las funcionalidades del sistema, se considera la existencia de un solo rol, o sea, un usuario que se va a comportar como el Administrador.

El Administrador es el que inicia el sistema mediante una lista de URL, además de que es el responsable de asignar el servidor de base de datos que se va a utilizar, las páginas o sitios web que el robot no visitará o excluirá en sus resultados, y configurar el tipo de conexión a Internet.

También el Administrador podrá parar cuando desee el sistema o darle una pausa y verificar como se encuentra la ejecución del sistema, así mismo tendrá la opción de volver a reanudar el sistema.

El sistema se conecta simultáneamente a varios sitios web y descarga su código fuente para luego extraer los hipervínculos, las palabras, el título y los metadatos de cada uno, luego de procesar cada sitio se almacenara cada uno de los datos extraídos.

3.6 Modelo de casos de uso del sistema

En este epígrafe se enumeran los actores del sistema así como daremos una breve descripción de sus principales casos de uso.

3.6.1 Definición de los actores del sistema.

Tabla. Actores del sistema

Actores	Descripción
Administrador	Es el único operador del sistema, por tanto es el iniciador de este, tiene todos los privilegios para operar con la Base de datos que utiliza el sistema. Puede asignar un filtro y además la lista de vínculos.

3.6.2 Casos de uso del sistema.

A continuación se presentan los casos de uso determinados para satisfacer los requerimientos funcionales de sistema.

CU-1	Configurar sistema.
Actor	Administrador (inicia)
Descripción	El administrador debe incluir la lista inicial de vínculos al sistema, asignar un filtro, un servidor de base de datos y el tipo de conexión a Internet.
Referencia	R1, R5, R6, R7

CU-2	Procesar vínculos
Actor	Administrador (inicia)
Descripción	Dado un vínculo se descarga su código fuente y se extraen los datos necesarios de dicha Url, son adicionados sus hipervínculos al sistema y luego se registra como documento.
Referencia	R2

CU-3	Indexar documentos.
Actor	
Descripción	Se indexará un documento que haya sido registrado anteriormente.
Referencia	R3

CU-4	Mostrar reporte
Actor	Administrador (inicia)
Descripción	El administrador desea ver un reporte del sistema y este lo muestra.
Referencia	R4

CU-5	Almacenar reporte.
Actor	
Descripción	Se almacenará un reporte después de terminada la indexación de todos los documentos.
Referencia	R8

3.6.3 Diagrama de casos de uso del sistema

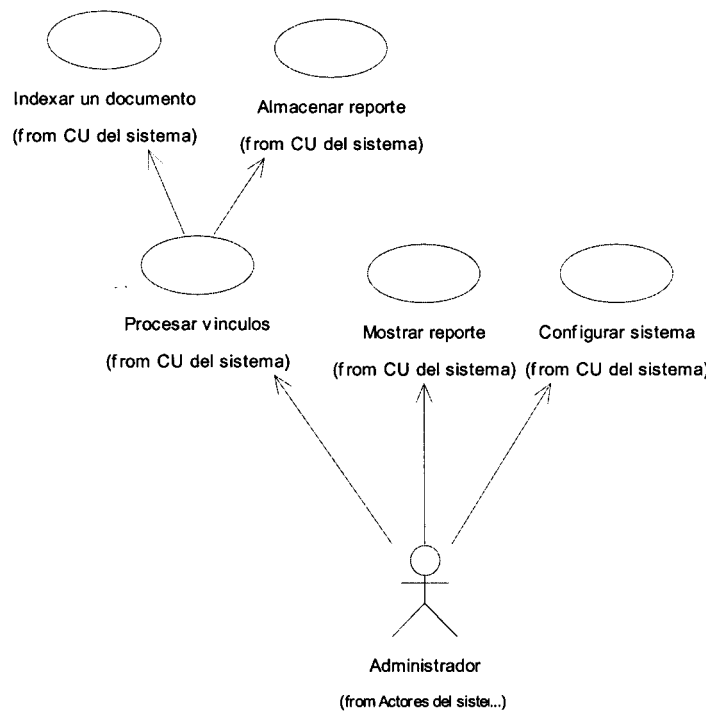


Figura 3.2 *Diagrama de casos de uso del sistema.*

3.6.4 Expansión de los casos de uso.

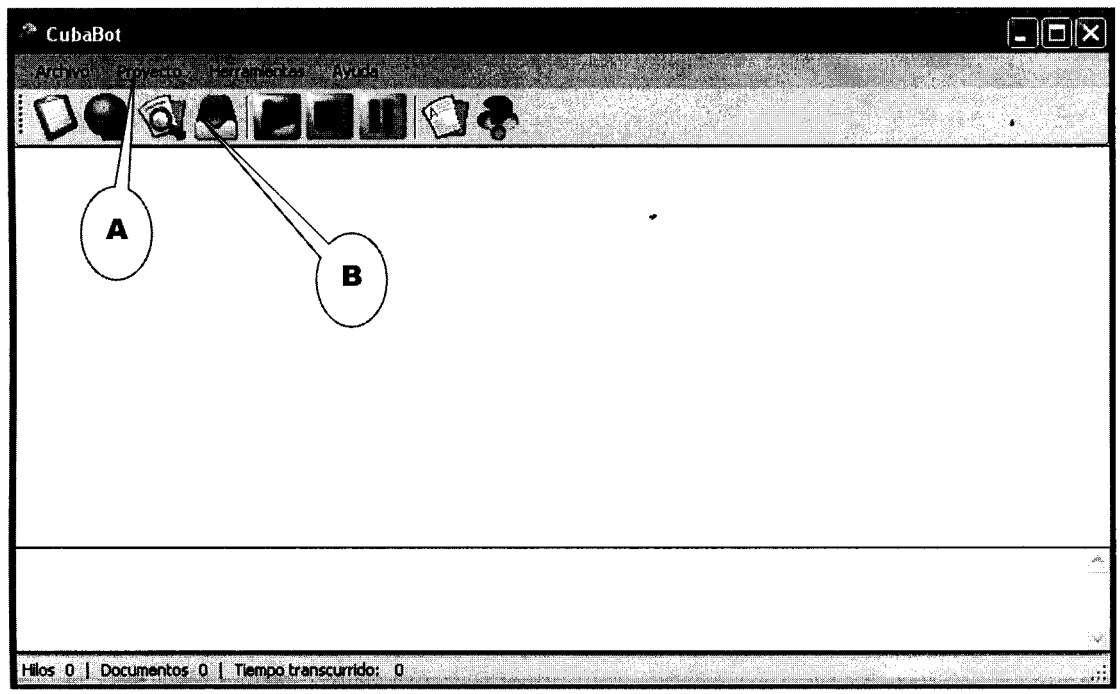
Mediante los casos de uso expandidos se describe paso a paso la secuencia de eventos que los actores utilizan para completar un proceso a través del sistema. Este sería el último paso en el análisis, para pasar a la construcción de la solución propuesta.

Caso de Uso:	Configurar sistema
Actor(es):	Administrador (inicia)
Propósito:	Posibilitar que el sistema quede bien configurado para el inicio del éste.
Resumen:	El administrador debe incluir la lista inicial de vínculos al sistema, asignar un filtro, un servidor de base de datos y el tipo de conexión a Internet.
Referencias:	R1, R5, R6, R7
Precondiciones:	El administrador accede a la interfaz principal.

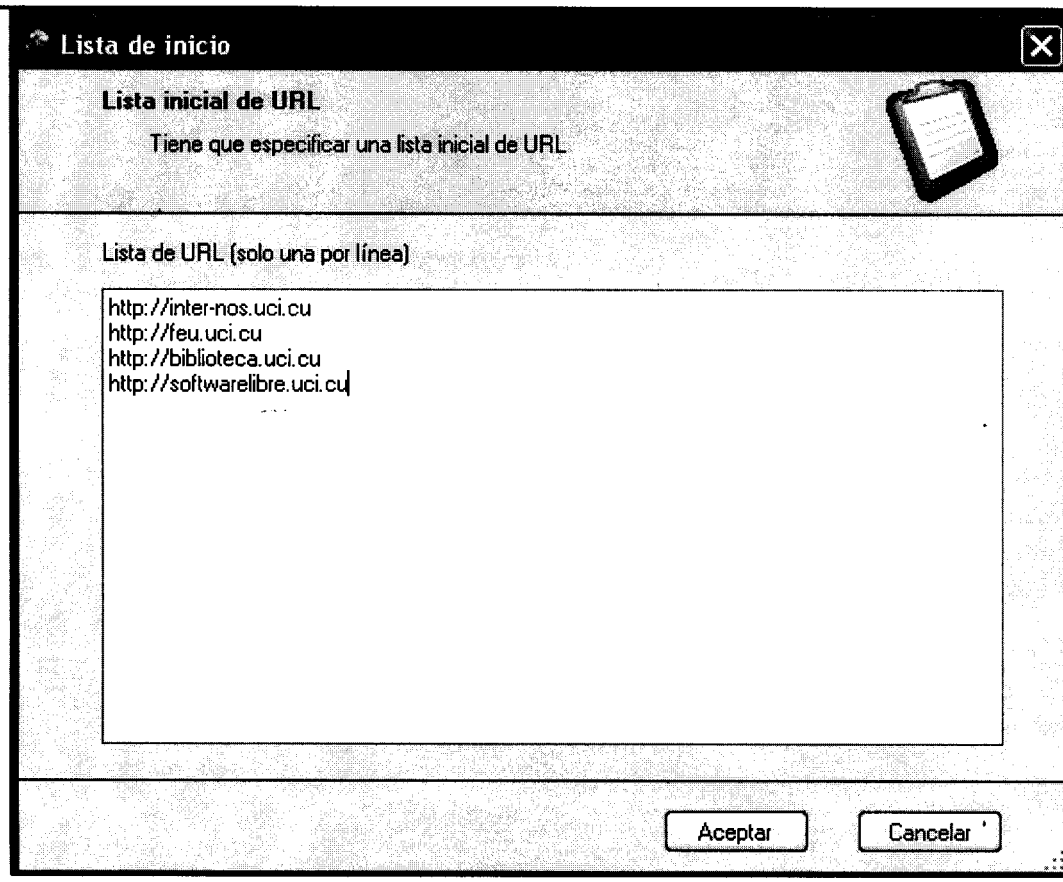
Acción del Actor	Respuesta del Sistema
1. El administrador selecciona el tipo de configuración que realizará (Pantalla 1).	2. a) Si elige "Incluir lista inicial de vínculos", ir a la sección Lista inicial. b) Si elige "Asignar filtro", ir a la sección Asignar filtro. c) Si elige "Servidor de base de datos", ir a la sección de Servidor Base de Datos. d) Si elige "Conexión a Internet", ir a la sección de Conexión a Internet.
Sección: "Lista Inicial"	
Acción del Actor	**Respuesta del Sistema**
3. Introduce el vínculo o el listado de vínculos inicial (Pantalla 2).	4. Valida las URL entradas y registra la lista insertada. En caso de que los datos no sean correctos, ver CA1.
Sección: "Conexión a Internet"	
Acción del Actor	**Respuesta del Sistema**
5. Introduce los datos del servidor Proxy y de autentificación, en caso de la conexión así lo requiera (Pantalla 3, A, B).	6. Valida los datos introducidos. En caso de que los datos no sean correctos, ver CA2.
Sección: "Servidor de Base de Datos"	

Acción del Actor	Respuesta del Sistema
7. Introduce los datos del servidor de base de datos, en caso de ser necesario los datos de autentificación (Pantalla 4, A, B).	8. Valida los datos introducidos. En caso de que los datos no sean correctos, ver CA3.
Sección: "Asignar Filtro"	
Acción del Actor	**Respuesta del Sistema**
9. Introduce el dominio web y las páginas web que no desea que el sistema procese (Pantalla 5, A, B).	10. Valida que el dominio entrado y las páginas webs introducidas estén escritos correctamente. En caso de que los datos no sean correctos, ver CA4.

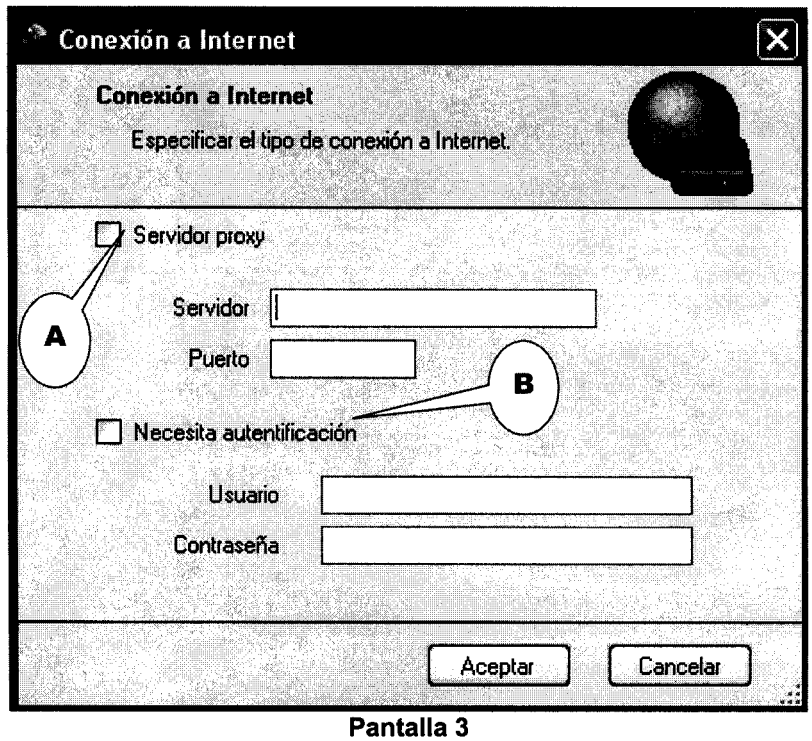
Interfaz



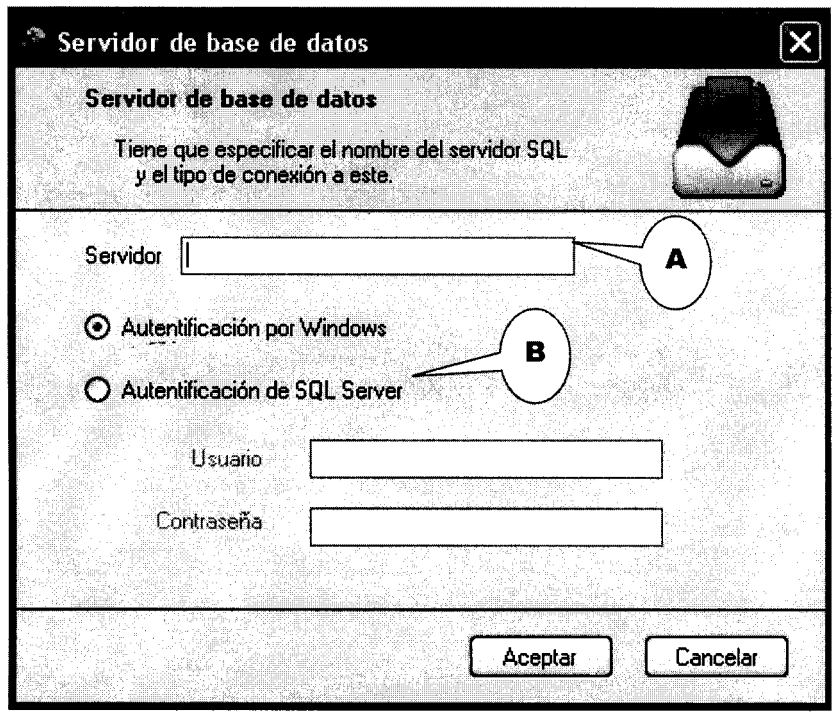
Pantalla 1



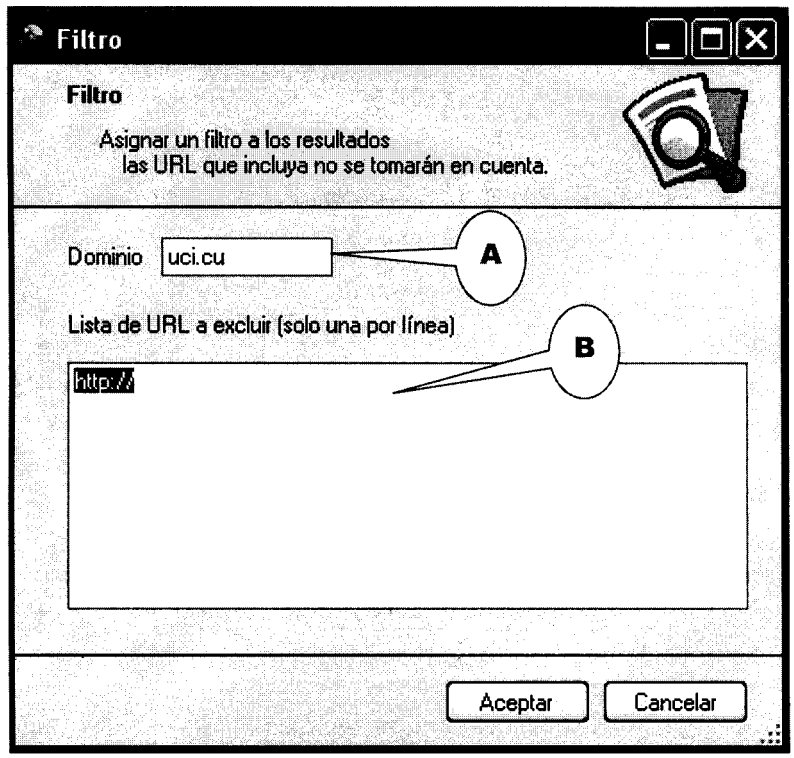
Pantalla 2



Pantalla 3


Servidor de base de datos
Servidor de base de datos
Tiene que especificar el nombre del servidor SQL y el tipo de conexión a este.
Servidor
A
Autentificación por Windows
B
Autentificación de SQL Server
Usuario
Contraseña
Aceptar
Cancelar

Pantalla 4


Filtro
Filtro
Asignar un filtro a los resultados las URL que incluya no se tomarán en cuenta.
Dominio
uci.cu
A
Lista de URL a excluir (solo una por línea)
B
http://
Aceptar
Cancelar

Pantalla 5
Cursos Alternativos:

Caso de Uso:	Procesar un vínculo	
Actor(es):	Administrador (inicia)	
Propósito:	Posibilitar que se procese un vínculo y se guarde su nombre y todos sus datos.	
Resumen:	Primeramente se establece conexión a dicho vínculo, luego se procede a la descarga del código fuente, después a la extracción del título, hipervínculos, palabras y meta datos, para al final almacenar toda esa información.	
Referencias:	R1, R2	
Precondiciones:	El sistema ha sido configurado correctamente.	
Acción del Actor		**Respuesta del Sistema**
1. Da en el botón de Ejecutar y así comienza el funcionamiento del sistema.		2. Toma el primer vínculo disponible. 3. Establece conexión con el vínculo. En caso de que ocurra algún problema en la red, o se demore más de 45 segundos u ocurra alguna excepción web, ver CA1. 4. Descarga el código fuente para que sea escaneado por el sistema. 5. Comienza con la extracción del título del vínculo, hipervínculos, palabras y meta datos, a partir de su código fuente. 6. Procede a registrar la referencia de dicho documento. 7. Registra los vínculos relacionados

	a este documento si antes ya no han sido registrados o procesados.
Interfaz	
Cursos Alternativos: CA1: Se reporta el error al sistema y termina el caso de uso.	

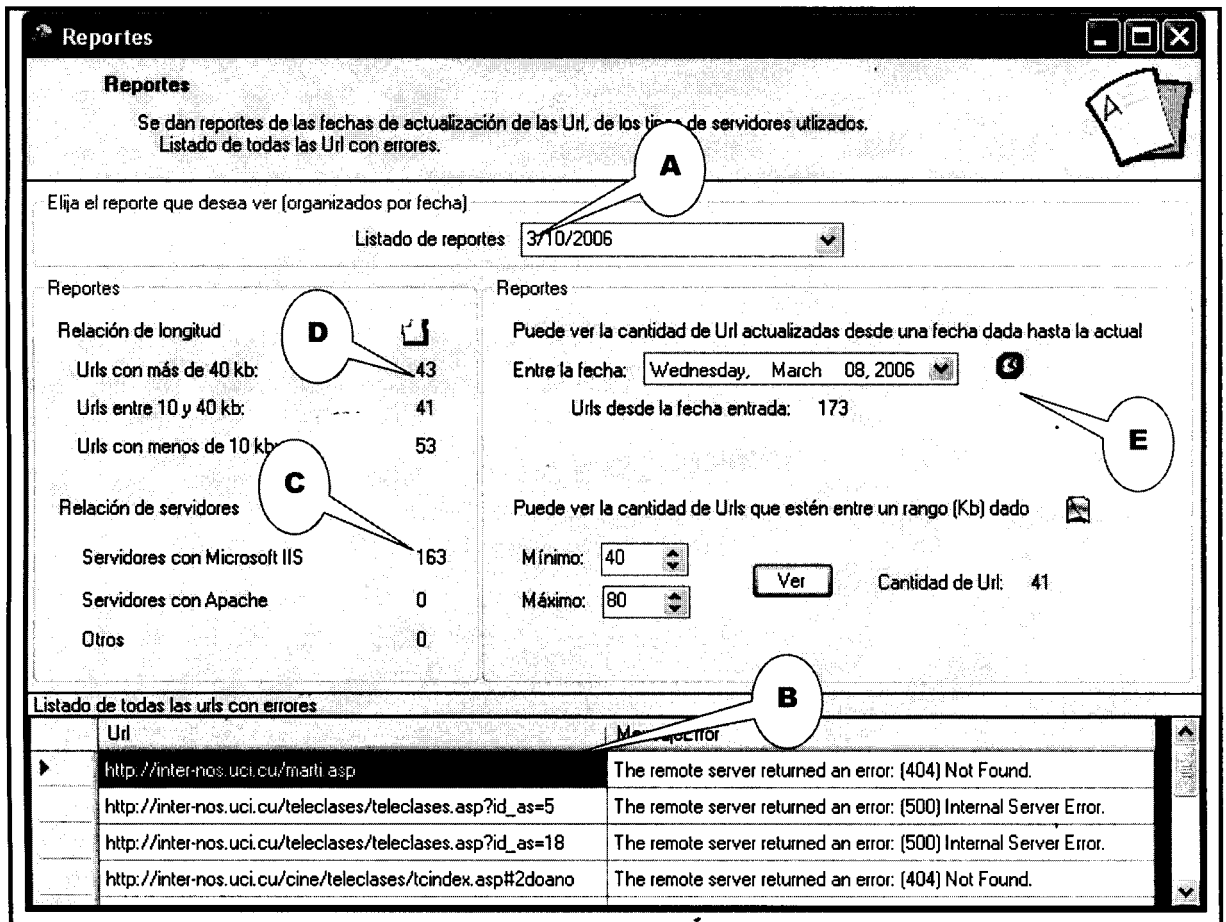
Caso de Uso	Indexar un documento
Actor(es):	
Propósito:	Posibilitar que se indexe un documento con todos sus datos.
Resumen:	Primeramente se toma el primer documento registrado y luego se procede a indexar todos sus datos.
Referencias:	R3
Precondiciones:	El sistema debe tener registrados 1000 documentos.

Acción del Actor	Respuesta del Sistema
	1. Se conecta al servidor de base de datos. 2. Se toma le primer documento disponible. 3. Se extraen todos los signos de puntuación de la lista de palabras. 4. Se excluyen las "stop words" del listado de palabras 5. Se indexan todos los datos en el servidor de almacenamiento persistente.
Interfaz	

Cursos Alternativos:

Caso de Uso	Mostrar reporte
Actor(es):	Administrador (inicia).
Propósito:	Posibilitar que el administrador pueda ver reportes.
Resumen:	Primeramente el sistema debe estar en estado de pausa o detenido y luego tiene la opción de ver el reporte.
Referencias:	R3
Precondiciones:	El administrador debe estar en la ventana de Reportes.

Acción del Actor	Respuesta del Sistema
1. Da clic en el botón de reportes o accede mediante el menú de opciones. 3. Elige el reporte que desea ver.	1. Muestra el listado de reportes disponibles ordenados por fecha. 4. Muestra el reporte elegido por el administrador (A), con todas las Url que dieron errores de conexión (B), además de datos estadísticos como el tipo de servidor (C) y la cantidad de Url que tienen un cierto tamaño en Kb (D). Además de brindar otras opciones, como buscar por fecha (E).
Interfaz	



Cursos Alternativos:

Caso de Uso: Almacenar reporte	
Actor(es):	
Propósito:	Posibilitar que se almacene un reporte.
Resumen:	Se almacenará un reporte después de terminada la indexación de todos los documentos.
Referencias:	R8
Precondiciones:	El sistema debe tener registrados 1000 documentos.
Acción del Actor	**Respuesta del Sistema**
	1. Se conecta al servidor de base

	de datos. 2. Genera el reporte con todos los errores ocurridos en esos documentos, y todos los datos registrados por estos, como el tipo de servidor web, la fecha de actualización y otros. 3. Se almacena el reporte con la fecha actual.
Interfaz	
Cursos Alternativos:	

3.7 Conclusiones.

En este capítulo se comenzó el desarrollo de la propuesta de solución, analizándose los requisitos que debe tener el sistema, llegando a los casos de uso necesarios para satisfacer a estos requisitos. Partiendo de esto, se puede empezar a construir el sistema, a partir de los casos de uso.

Capítulo 4

CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA.

4.1 Introducción

En este capítulo se modelan los artefactos que constituyen las clases de cada paquete de la aplicación, y las relaciones entre ellos. Se muestra además, el modelo de datos, para una mejor comprensión de qué parte de la información manipulada es almacenada.

Se exponen las pautas seguidas para el diseño de la capa de presentación de la aplicación. Se muestra además, cómo ha sido elaborada la interfase, la estructura de la ayuda y los reportes realizados.

Además, se especifican los estándares a seguir en la codificación, para lograr una mejor organización y un estilo propio del software y por último se presentará el modelo de implementación, donde se definen los nodos y componentes que conforman la estructura física de la aplicación.

4.2 Diagrama de clases

Para una mejor comprensión del diagrama de clases del presente trabajo, se han separado las mismas en 4 paquetes atendiendo a su funcionalidad.

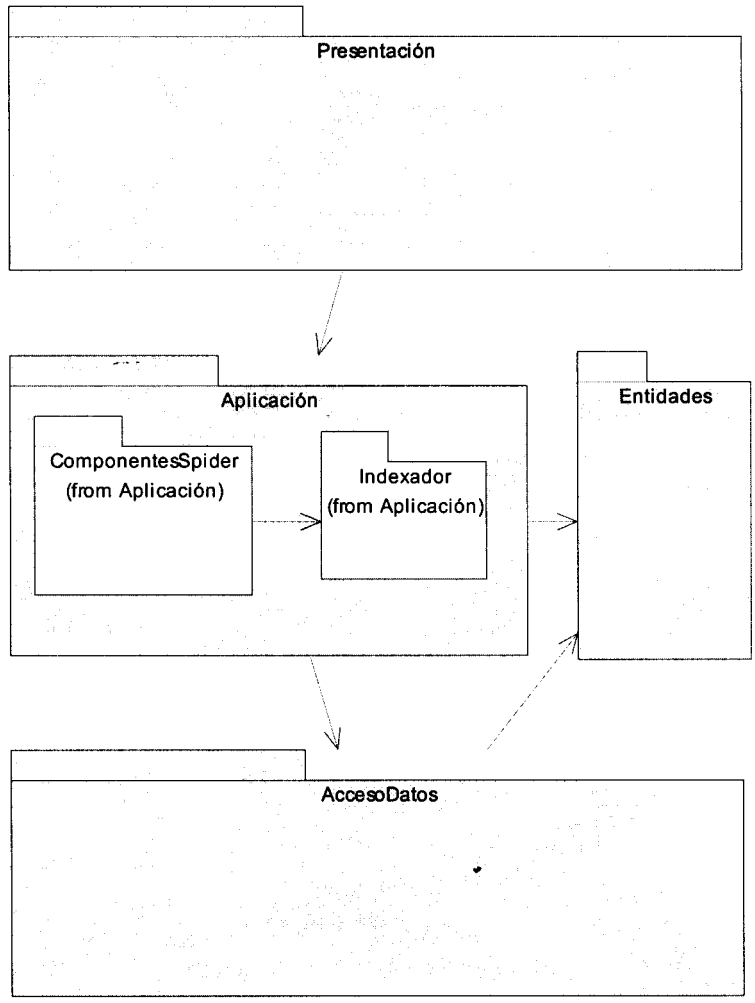


Figura 4.1 *Diagrama de clases de diseño.*

El paquete ***AccesoDatos*** contiene las clases para hacer posible la persistencia y recuperación de objetos. Es el que contiene las clases encargadas de acceder a la base de datos.

El paquete ***Aplicación*** contiene la lógica del dominio de la aplicación y es el que contiene las clases controladoras relacionadas con las entidades. Está dividido en dos subpaquetes: ***ComponentesSpider*** que abarca las clases para el control y extracción de los datos necesarios de cada documento web. El subpaquete ***Indexador*** es el que tiene las clases necesarias para la indexación de todos los documentos.

El paquete ***Entidades*** contiene clases que no tienen comportamiento, sólo propiedades y son representaciones de entidades reales del dominio, son además, clases persistentes que son accedidas por las clases de los paquetes *Aplicación* y *AccesoDatos*.

El paquete ***Presentación*** contiene las clases de presentación del sistema, en este caso, una interfaz de ventanas de escritorio. Aquí están todas las ventanas relacionadas con la configuración del sistema, gestión de los datos y los reportes.

Esta división por paquetes obedece a la arquitectura dividida por capas en que se ha diseñado la aplicación y también a la funcionalidad de las clases, las cuales han sido agrupadas de esta forma para lograr mayor desacoplamiento, reutilización y legibilidad de los diagramas.

Paquete Presentación.

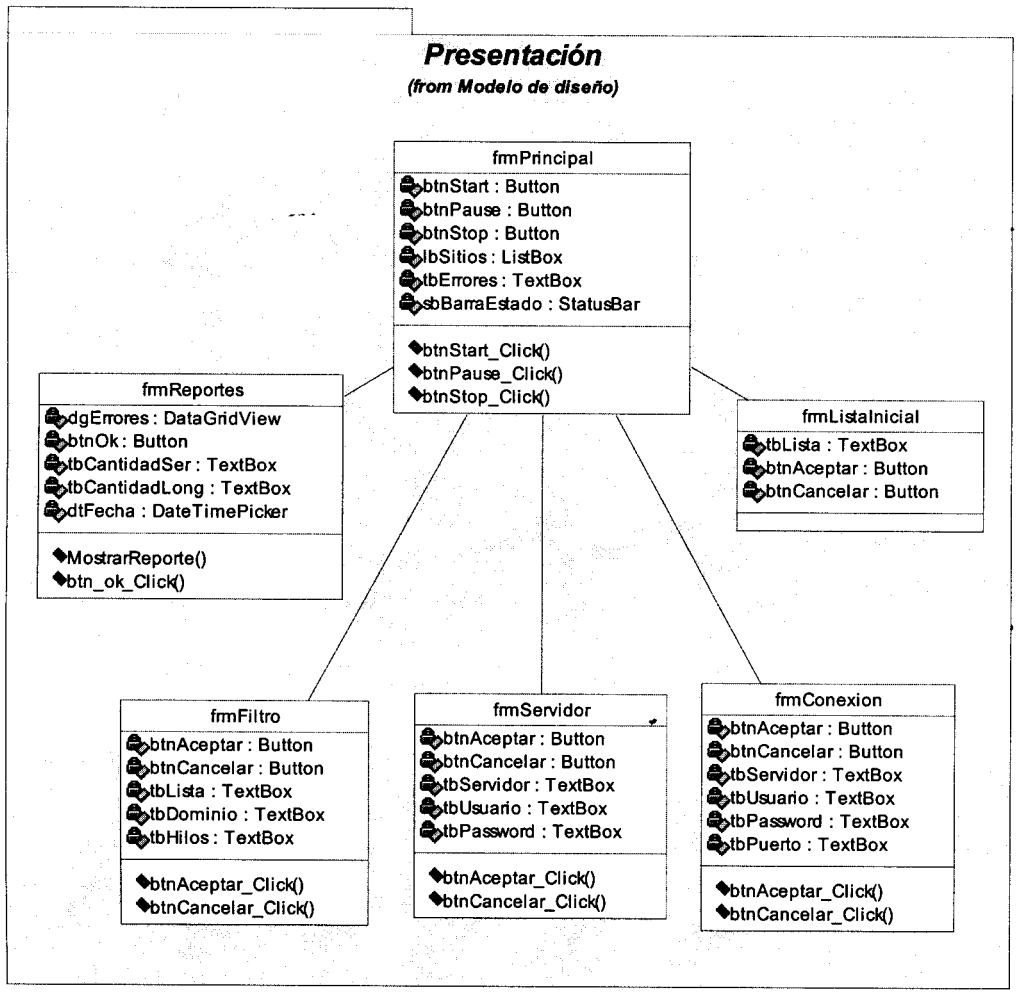


Figura 4.2 *Diagrama de clases del paquete Presentación.*

Paquete ComponentesSpider

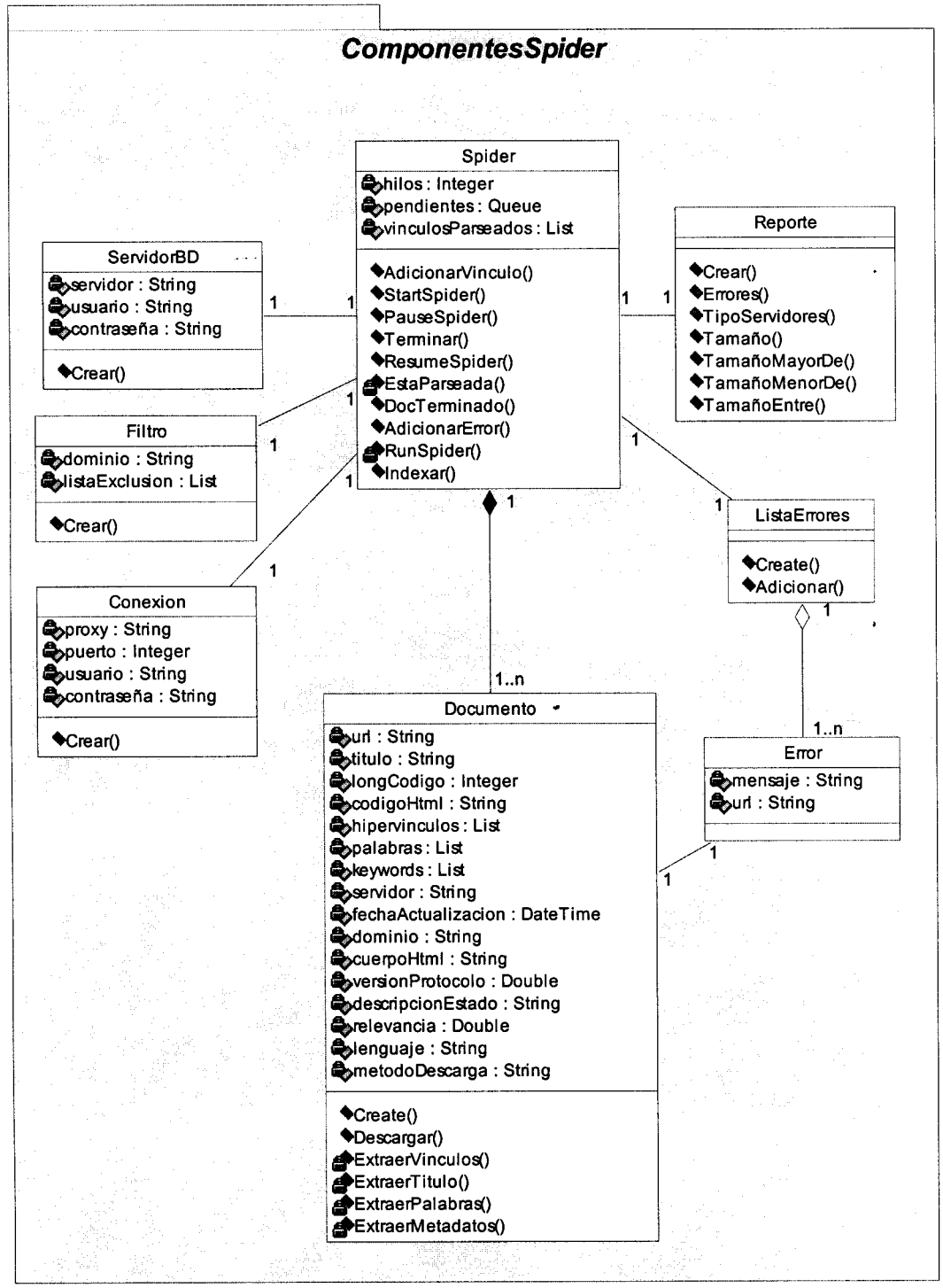


Figura 4.3 *Diagrama de clases del paquete Dominio.*

Paquete Indexador

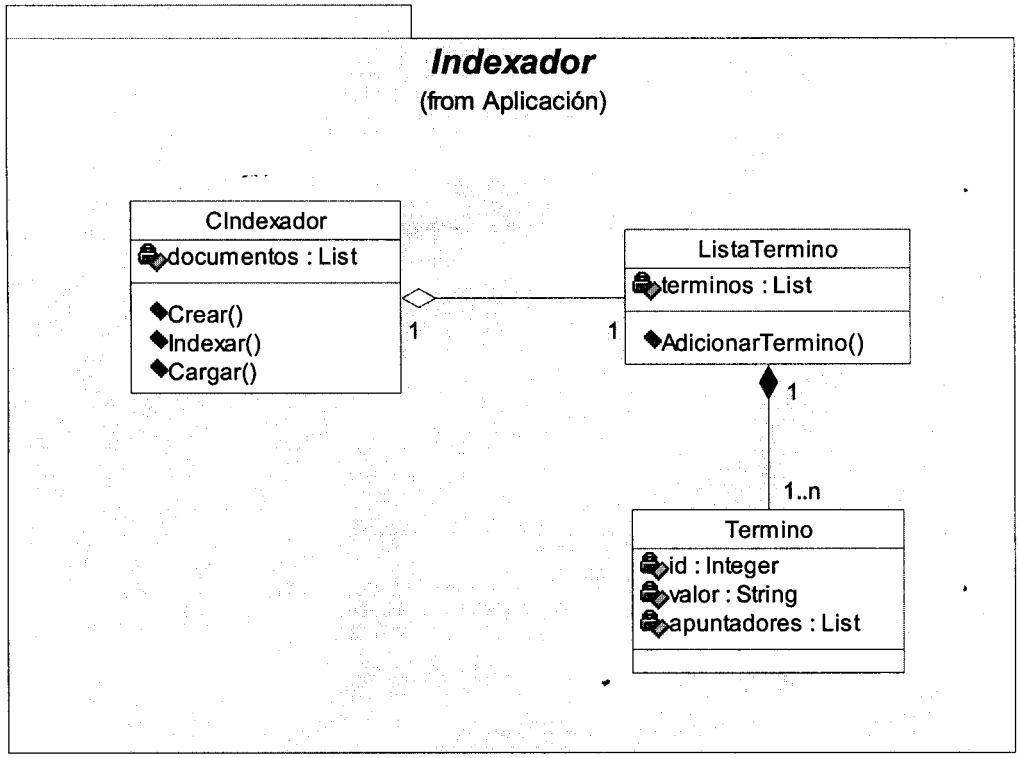


Figura 4.4 *Diagrama de clases del paquete Indexador.*

Paquete Entidades.

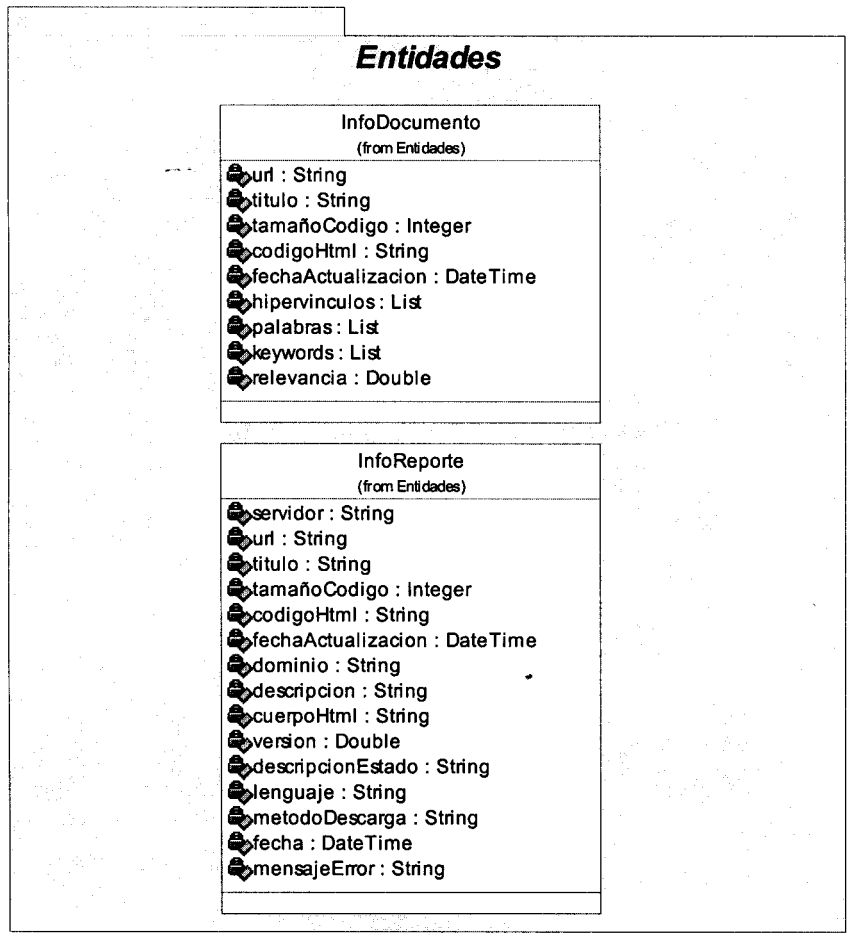


Figura 4.5 *Diagrama del subpaquete Entidades.*

Paquete AccesoDatos

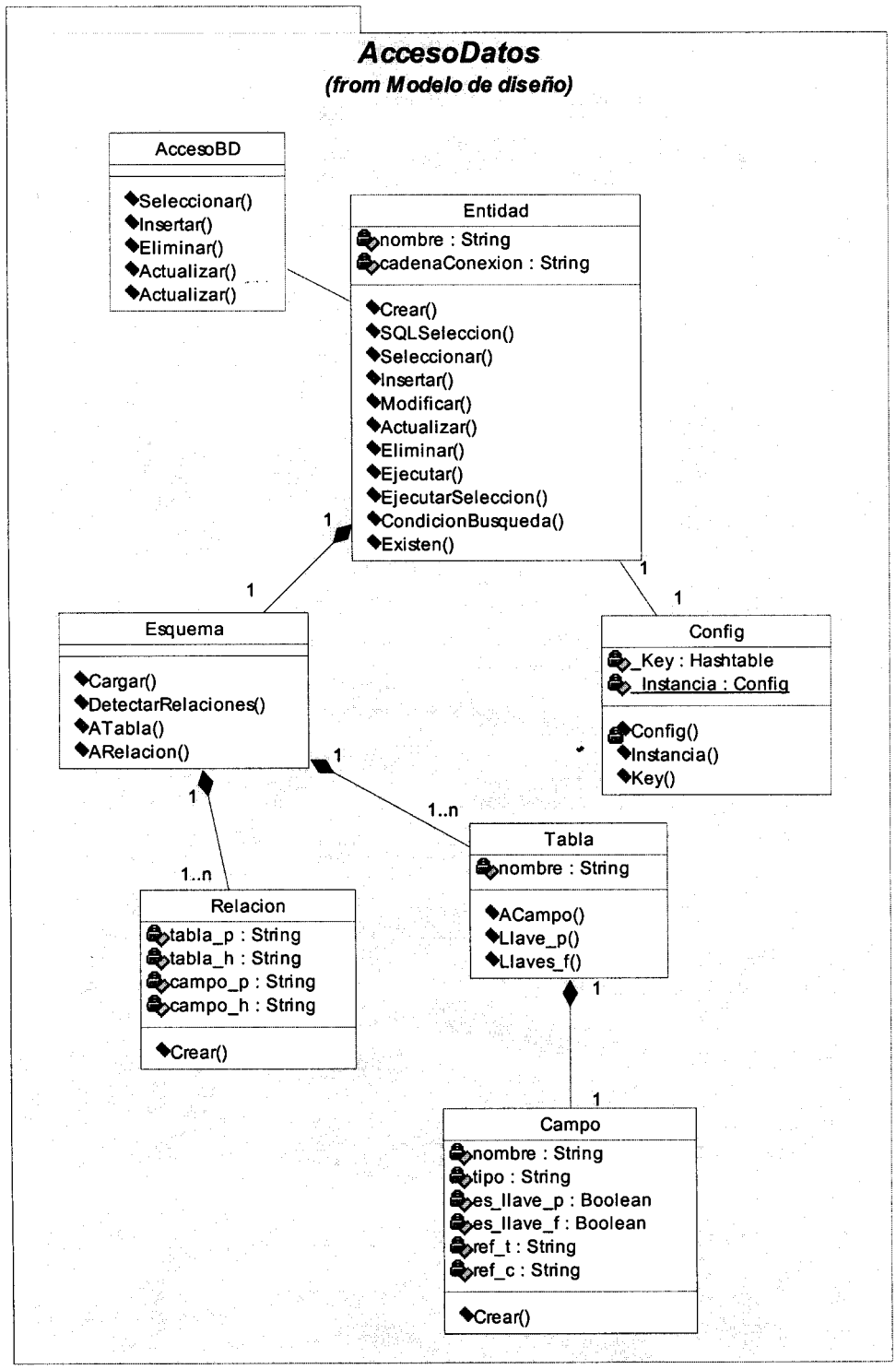


Figura 4.6 *Diagrama de clases del paquete AccesoDatos.*

4.3 Diseño de la base de datos

Para asistir en la construcción de la base de datos, se utiliza el diagrama de clases persistentes, - que es el mismo que el diagrama del paquete *Entidades*, por tanto se decidió omitirlo en este epígrafe -, con la representación física de la misma.

4.3.1 Modelo de datos.

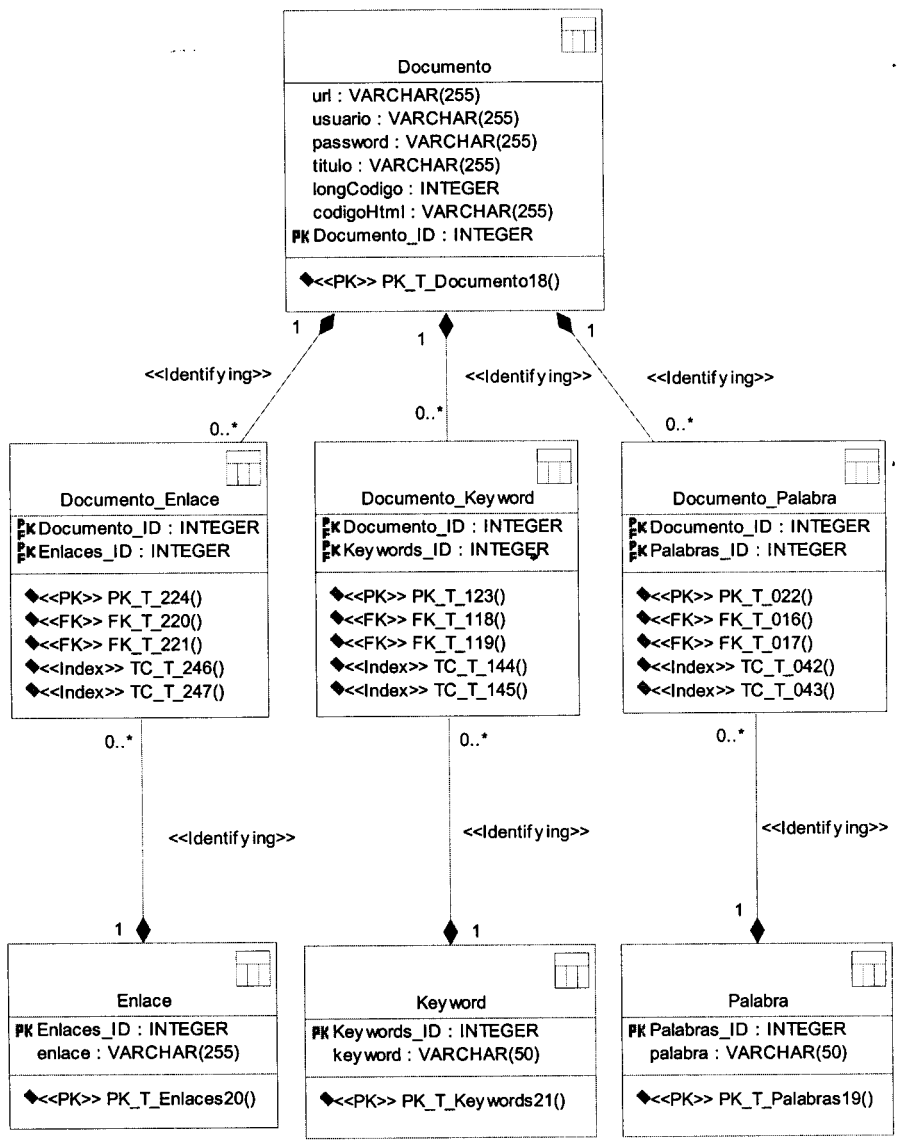


Figura 4.7 *Modelo de datos.*

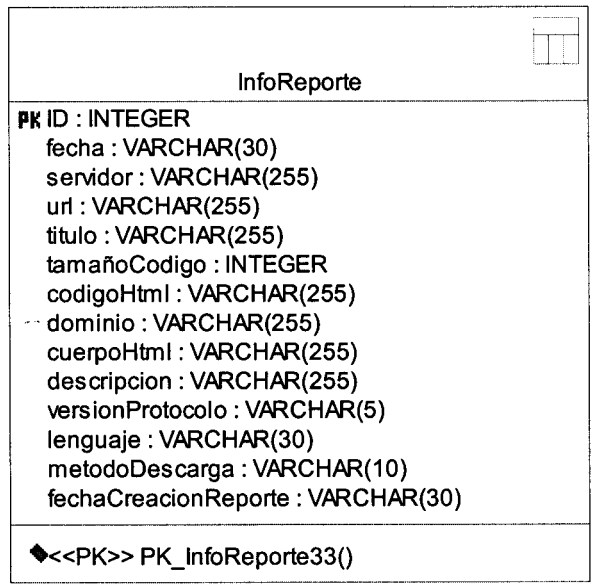


Figura 4.8 *Modelo de datos (cont.)*

4.4 Principios de diseño.

4.4.1 Estándares de la interfaz de aplicación.

El diseño ha sido elaborado pensando en los usuarios finales, que serán: administradores de redes, otros usuarios, los cuales, aunque deben tener conocimientos de computación, se ha elegido una interfaz amigable e intuitiva. Se ha mantenido un diseño consistente en todas las ventanas, para lograr que el usuario se sienta cómodo y logre acostumbrarse rápidamente a la aplicación.

Todas las ventanas de configuración tienen una estructura similar, contienen una imagen identificativa y un texto que explica brevemente su funcionalidad (Figura 4.9, A, B).

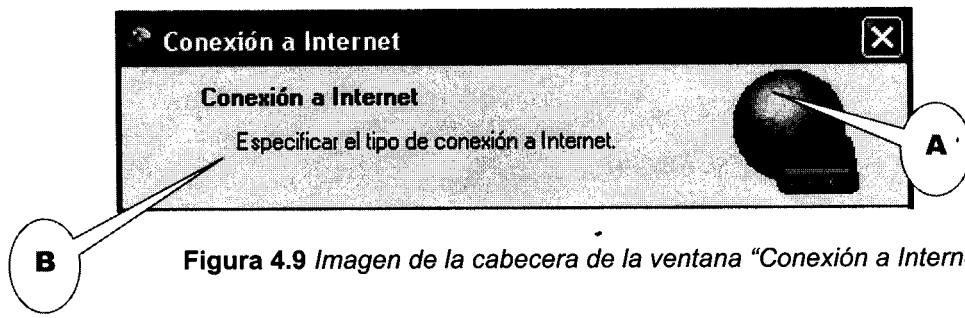


Figura 4.9 *Imagen de la cabecera de la ventana "Conexión a Internet"*

La ventana principal contiene un vínculo a cada una de las ventanas de opciones de configuración, como "Conexión a Internet", se puede acceder a estas mediante botones o por un menú de opciones (Figura 4.10, A, B).

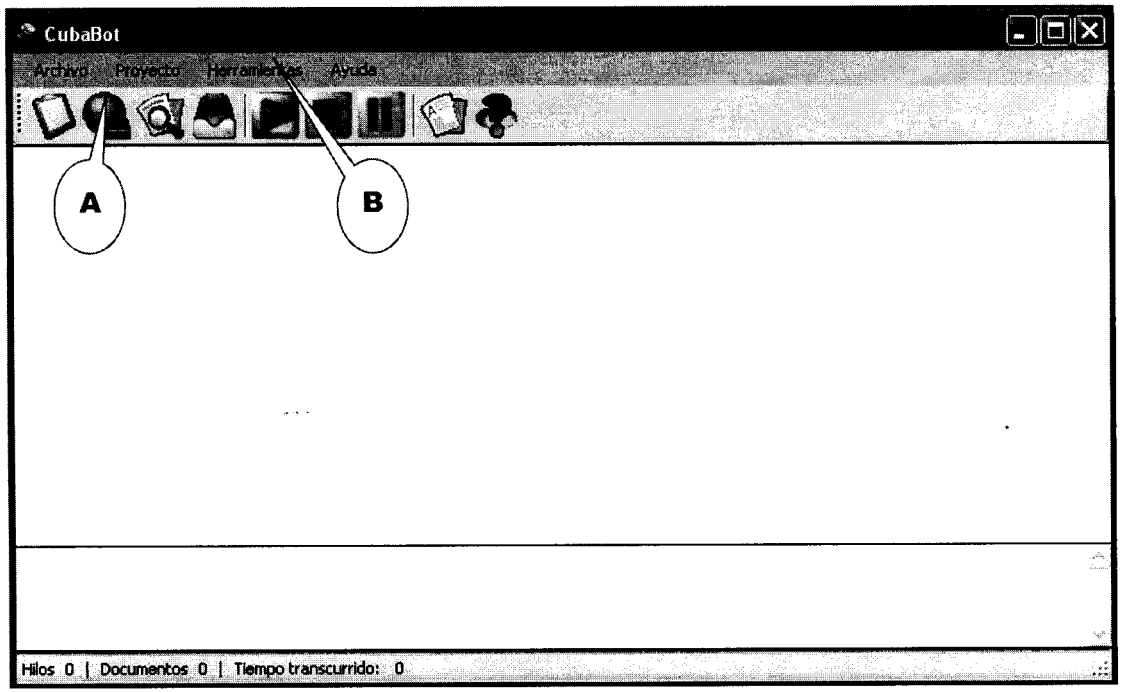


Figura 4.10 *Ventana principal de la aplicación.*

Los colores utilizados son en su mayoría los tonos de anaranjado claro, blanco, y amarillo claro. Se ha utilizado anaranjado claro porque es un color agradable a la vista y se utiliza mayormente para resaltar secciones importantes. El blanco se utiliza de fondo para mostrar información, así es mucho más legible para el usuario. El amarillo se utiliza como fondo para los reportes de errores.

4.4.2 Formato de reportes.

El sistema brinda dos tipos de reportes, los reportes cortos o rápidos que son aquellos que el administrador necesita saber constantemente, como la cantidad de hilos que se están ejecutando, el tiempo transcurrido desde el inicio del sistema, etc. El otro tipo de reportes son más detallados, cantidad de sitios con errores, el tipo de servidor que usa cada url, etc. Los reportes cortos se dan en la barra de estado de la ventana principal y los reportes de errores de cada sitio están dados de color rojo en la ventana principal (Figura 4.11, A, B, C), el resto de los reportes detallados están en la ventana "Reportes" que son pedidos por el administrador.

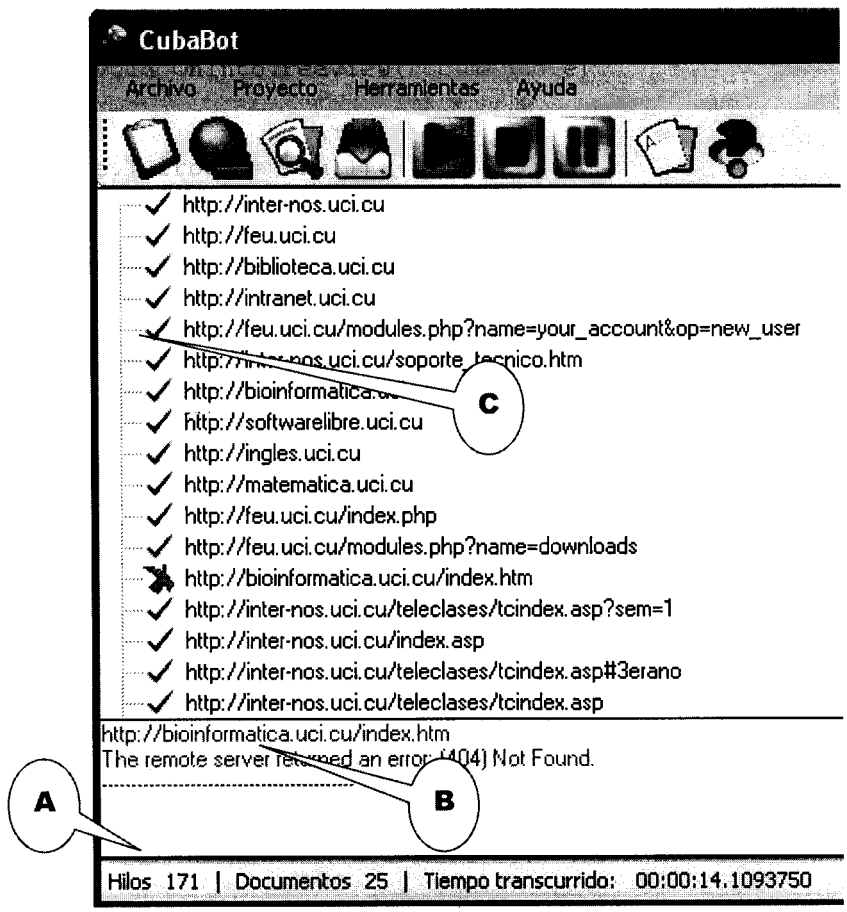


Figura 4.11 *Parte de la ventana principal.*

Los reportes más detallados y que necesitan de una mayor atención del administrador están en la ventana "Reportes", donde se muestra la cantidad de páginas que ocasionaron errores, el tipo de servidor que utilizan las urls, el peso en Kb de las urls, divididas por secciones, y se brinda la posibilidad de ver que cantidad de páginas han sido actualizadas desde una fecha entrada, (Figura 4.12, A, B, C).

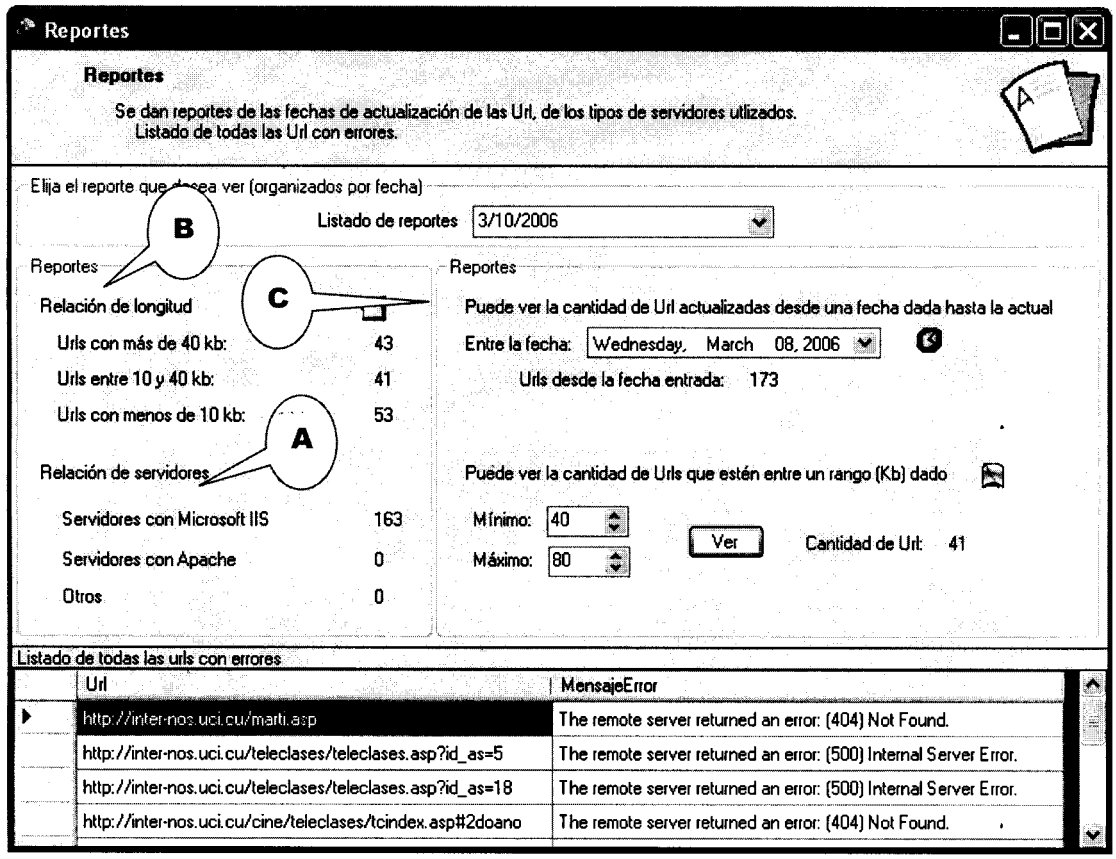


Figura 4.12 *Ventana de reportes.*

4.4.3 Concepción de la ayuda.

El sistema cuenta con un manual de usuario que esta disponible desde cualquier página del mismo, en dicho manual, se explica su funcionamiento, requerimientos, y se muestra una lista con preguntas frecuentes y sus respuestas.

4.4.4 Tratamiento de excepciones

Para prevenir errores por parte del usuario, sólo se le brindan las opciones mínimas necesarias, a la hora de efectuar cualquier operación, por ejemplo, se deshabilitan ciertos botones si el usuario no tiene que utilizarlos en ese momento.

Se manipulan las excepciones que podrían ocurrir en tiempo de ejecución de la aplicación, ya sean excepciones del tipo web, es decir, excepciones en el momento en

que se establece conexión a un sitio determinado, así como datos erróneos entrados por el usuario, generándose un registro en la base de datos para una eventual revisión del administrador.

Existen dos clases que manipulan los errores de conexión a los sitios web, una de ellas es la clase *Error* que se genera a partir del levantamiento de alguna excepción web, y la clase controladora de estos errores es *ListaErrores* que es la que almacena todos los errores generados.

En la figura 4.13 (A) se muestra la captura de una excepción en la ventana "Servidor de base de datos" en el cuál el usuario no puso ningún servidor.

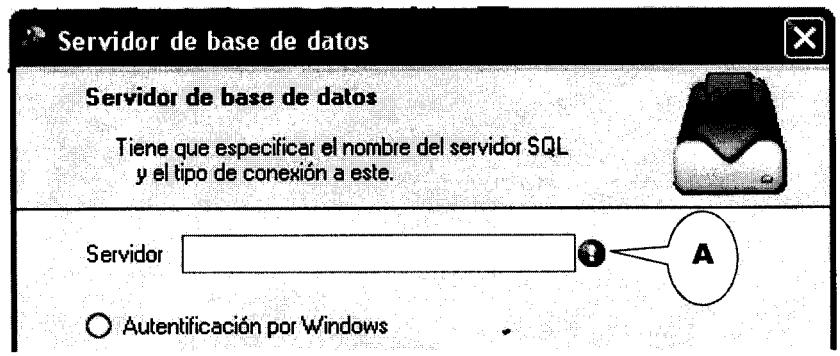


Figura 4.13 *Ejemplo de captura de errores.*

4.4.5 Estándares de codificación.

Es muy ventajoso utilizar un estándar para escribir código, entre éstas ventajas, tenemos las siguientes:

1. Reducir errores
2. Escribir un código comprensible y fácil de leer
3. Garantizar una buena comunicación entre los programadores del equipo
4. Facilitar el mantenimiento del software

En esta aplicación se ha utilizado el estándar de codificación del grupo de programadores del portal CubaSI, que utiliza el estilo "Camel Case"[10] que principalmente tiene que ver con la capitalización de los caracteres (Figura 4.14)

```
private void ExtraerPalabras(string code, Regex regPalabra)
        {
            MatchCollection colPalabras = regPalabra.Matches(code);

            foreach (Match m in colPalabras)
                palabras.Add(m.Result("$0"));
        }
```

Figura 4.14 *Ejemplo de codificación utilizada.*

Se ha tenido especial cuidado para nombrar clases, variables, y demás elementos, precediendo cada nombre con un prefijo para su fácil identificación, así, tenemos los más frecuentes en este sistema:

- Campos de edición: **txt**Nombre.
- Campos de selección: **lbl**Nombre o **cb**Nombre
- Botones de acción: **btn**Nombre.
- Variables de control de ciclos: **i**, **j**
- Treeviews: **tv**Nombre
- Listbox: **lb**Nombre

En el diseño de la base de datos, las tablas se han nombrado igual que la entidad que almacenan. Los campos de estas tablas, siguen la misma regla, y están nombrados igual que las propiedades de las entidades.

[10] Estilo de escritura de código, se basa en capitalizar las iniciales de las todas las palabras que forman el nombre de un método o variables, menos la primera.

4.5 Modelo de despliegue

El modelo de despliegue contiene los nodos que conforman la topología de hardware sobre la que se ejecuta el sistema.

El sistema se ha construido siguiendo la arquitectura de tres capas, o sea, la separación de los componentes en Lógica de Negocio o Dominio, Acceso a Datos, Presentación.

Existe un servidor SQLServer 2000 que contiene la base de datos. El sistema se encuentra en otra PC, que debe ser potente y debe tener instalada la plataforma .NET 2.0.

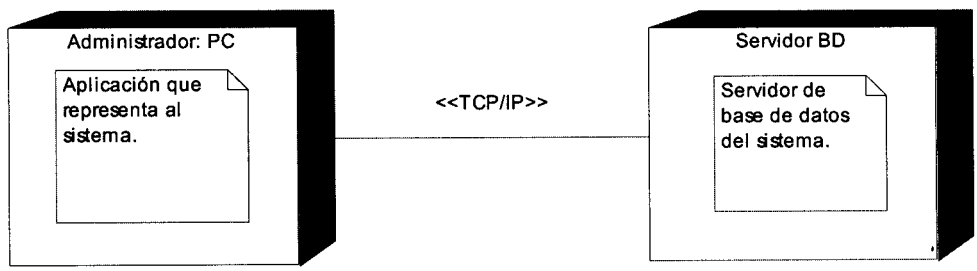


Figura 4.15 *Diagrama de despliegue.*

4.6 Modelo de componentes.

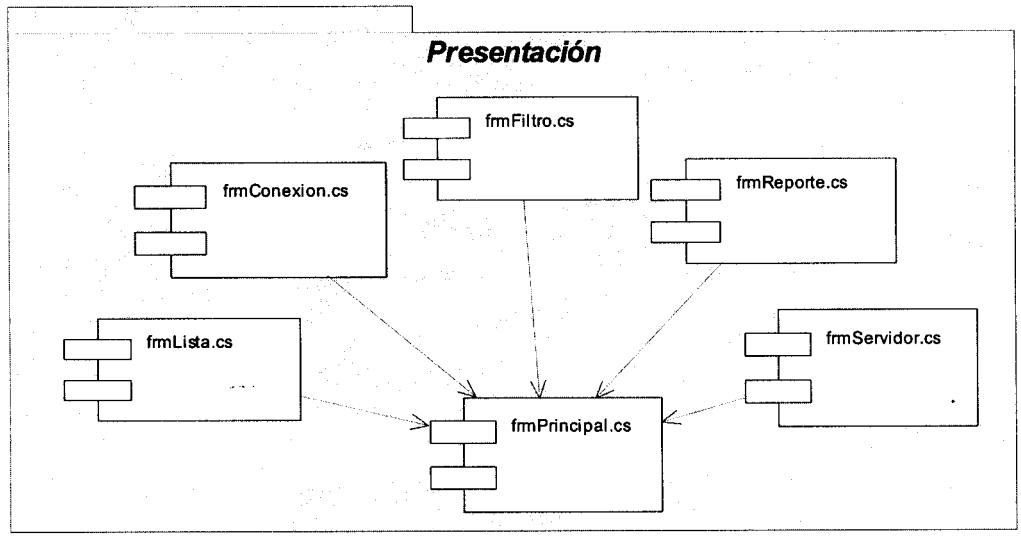


Figura 4.16 *Diagrama de componentes.*

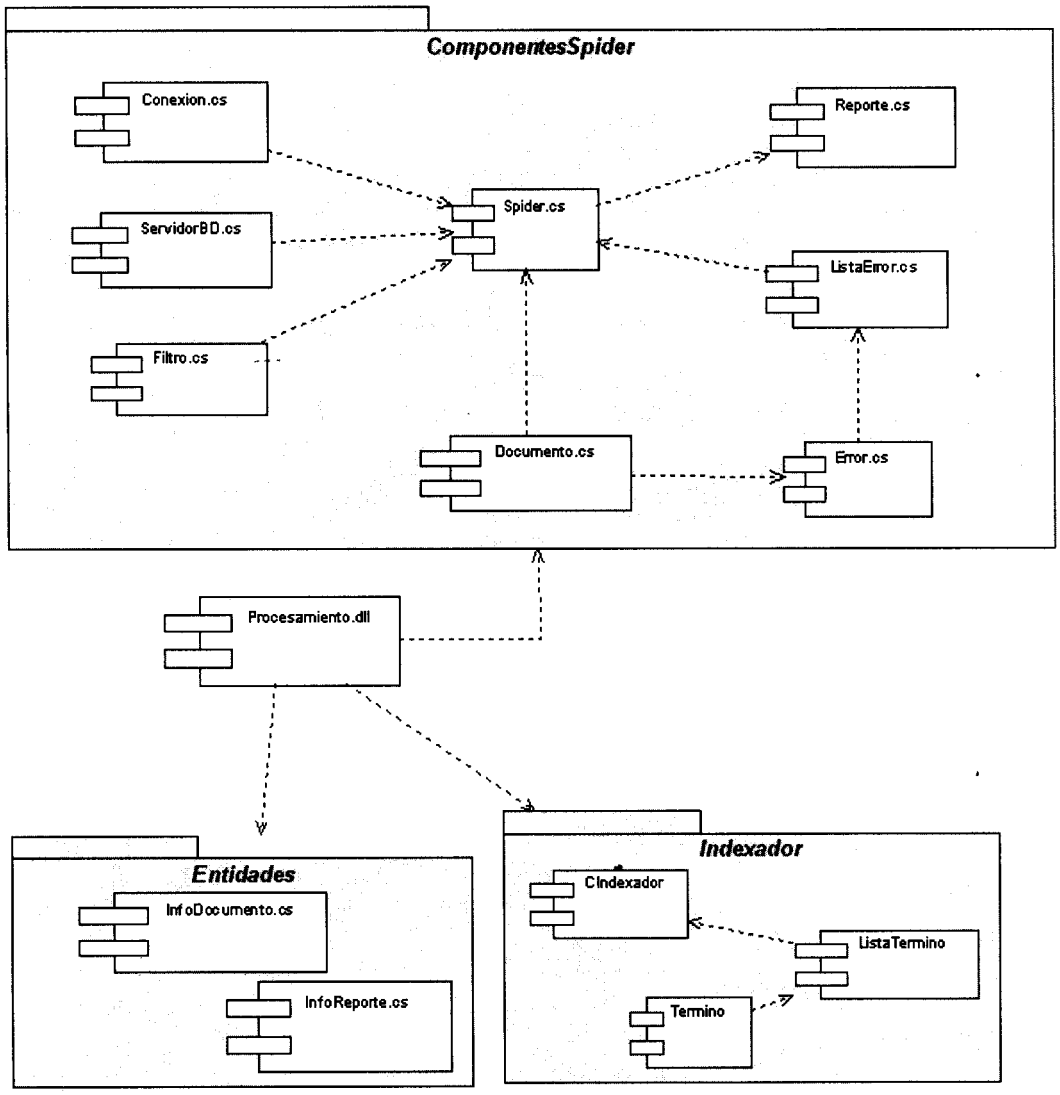


Figura 4.17 *Diagrama de componentes. (cont.)*

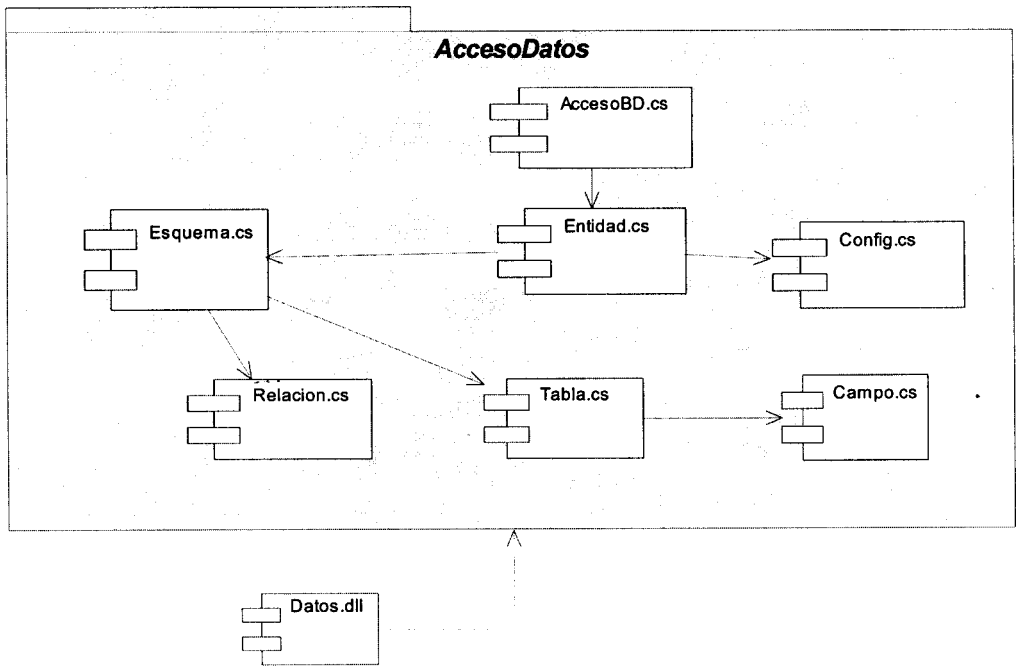


Figura 4.18 *Diagrama de componentes. (cont.)*

4.6.1 Explicación de los componentes

Se seguirá el mismo orden en que se han presentado los paquetes, para una mayor claridad en las explicaciones de cada componente. Se implementó cada una de las clases en un fichero separado, con el mismo nombre, por tanto, cada componente tiene el mismo nombre de la clase que contiene, con el objetivo de lograr una mayor limpieza y rapidez al encontrar las clases. Si desea información sobre el contenido de los componentes, remítase al diagrama de clases del paquete correspondiente. Los componentes que cumplen esta regla son los de los siguientes paquetes:

- Paquete **ComponentesSpider**

Aquí se encuentran los componentes que contienen las implementaciones de clases controladoras de los documentos web.

- Paquete **Indexador**

Aquí se encuentran los componentes que contienen las implementaciones de las clases encargadas de la indexación de los documentos.

- Paquete **Entidades**

En este paquete se encuentran los componentes que tienen las implementaciones de las clases entidades del sistema.

- Paquete **AccesoDatos**

En este paquete se encuentran los componentes que contienen las implementaciones de clases que se ocupan de la persistencia en base de datos de las entidades del sistema.

Otros componentes que no cumplen con la regla de nomenclatura antes mencionada son:

➢ **Procesamiento.dll**

Es la librería necesaria para controlar la lógica del negocio del sistema, contiene la implementación de las clases de los paquetes Aplicación y Entidades, ver diagrama de componentes de dichos paquetes.

➢ **Datos.dll**

Librería necesaria para el acceso a datos desde una BD, contiene la implementación de las clases del paquete AccesoDatos, ver diagrama de componentes de dicho paquete.

➢ **Config.xml**

Archivo XML[11] donde se guarda la conexión a la base de datos.

[11] Es el acrónimo de eXtensible Markup Language (lenguaje de marcado extensible) desarrollado por el World Wide Web Consortium (W3C).

- **Esquema.xml**

Archivo XML donde se guarda la estructura relacional de la base de datos.

- **SQLQuery.txt**

Este archivo contiene la consulta SQL que se va a ejecutar para crear las tablas en la base de datos.

4.7 Conclusiones

En este capítulo se ha llevado a cabo la descripción de las clases y demás elementos necesarios para la implementación. Se obtuvo el diagrama de clases del sistema, separado por paquetes. Se definieron, a partir del mismo, cuáles son las clases que serán persistentes, luego, a partir de esto, se construyó el modelo de datos. Se expusieron las pautas seguidas para el diseño de la interfaz, y se explicó cómo está estructurada la aplicación físicamente, mediante los modelos de despliegue y de componentes.

CONCLUSIONES

Con este trabajo se propone una solución al problema de la indexación de sitios webs en Cuba y la confección del primer buscador web cubano, se hicieron estudios de las diferentes técnicas de indexación y navegación para aplicar las más adecuadas a nuestra intranet nacional, además de un análisis de los sistemas distribuidos existentes asociados a los requerimientos funcionales y no funcionales del sistema para un posterior seguimiento del mismo. Se presenta una aplicación capaz de indexar sitios web y que brinda reportes sobre los mismos. Esta aplicación puede ser una poderosa herramienta ya que brinda un banco de datos que puede ser reutilizado por otras futuras aplicaciones.

El sistema se desarrolló siguiendo la metodología RUP, y se utilizaron representaciones UML para la modelación de todas las fases del proyecto.

El sistema resultante está provisto por un ambiente cómodo, que cumple con los estándares de diseño, utilizando técnicas modernas de programación orientada a objetos.

Por todo lo anterior se concluye que los objetivos propuestos para el presente proyecto han sido cumplidos satisfactoriamente. Se incluyen una serie de recomendaciones que deben tenerse en cuenta para el trabajo futuro.

RECOMENDACIONES

Los objetivos generales de este trabajo han sido cumplidos, pero a lo largo de su desarrollo han ido surgiendo ideas que podrían implementarse en un futuro, por tanto se recomienda:

- Estudiar con más profundidad los sistemas distribuidos y extender la aplicación a un sistema distribuido, debido al crecimiento constante de nuestra intranet nacional y la necesidad de recuperar toda la información contenida en ella y hacerlo de la forma más rápida y eficaz.
- Promover el uso de estándares internacionales de redacción de metadatos, como Dublin Core.
- Estudiar con más detalle los diferentes algoritmos de indexación existentes a fin de fortalecer el actual y estar a la par del crecimiento actual de la intranet cubana.

REFERENCIAS BIBLIOGRAFICAS

[1] Diseño e implementación de una Arquitectura multiplataforma para el estudio de Motores de búsqueda en Internet.
http://acoruna.tuportal.com

[2] Tramullas, Dr. Jesús: Bibliotecas y Centros de Documentación: *"Internet para bibliotecarios y documentalistas"*.

[3] Soto López, Nilet María, Saborit Ramírez, Yunier: *"Propuesta para un sistema de catalogación y recuperación de recursos de información"*. Trabajo para optar por el título de Ingeniería Informática, Instituto Superior Politécnico "José Antonio Echeverría", Ciudad de La Habana, junio 2004.

[4] Sistemas Distribuidos.
http://www.monografias.com/trabajos16/sistemas-distribuidos/sistemas-distribuidos.shtml

[5] Escalona Guerra, Caridad I.: *"Los portales en Internet, un nuevo recurso para la gestión de Información. Experiencia cubana"*. Conferencia Científica 40 Aniversario del IDICT, Capitolio de La Habana, Abril 15 del 2003.

[6] Lenguaje C
http://www.monografias.com/trabajos4/lenguajec/lenguajec.shtml

[7] Lenguaje Java
http://www.monografias.com/trabajos12/lenguajejava/lenguajejava.shtml

[8] Lenguaje Python
http://www.monografias.com/trabajos15/lenguajepython/lenguajepython.shtml

[9] Características de Visual Studio .NET. Microsoft Corp. 2005.
http://www.microsoft.com/latam/vstudio/producto/caracteristicas.asp

[10] Ayuda del Visual Studio .NET 2005
http://www.microsoft.com/spanish/msdn/latam (12/12/2005)

[11] Booch, G., Rumbaugh, J., Jacobson, I. *"El Lenguaje Unificado de Modelado"*. Addison-Wesley. 1999.

BIBLIOGRAFÍA

1. Schmuller, Joseph: "Aprendiendo UML en 24 horas".
2. Booch, G., Rumbaugh, J., Jacobson, I.: "El Proceso Unificado de Desarrollo de Software". Addison-Wesley 2000.
3. Larman, Craig: "UML y Patrones: Introducción al análisis y diseño orientado a objetos".
4. Charte Ojeda, Francisco: "Visual C# .NET". Anaya Multimedia 2002.
5. Carmona, David: "Programación en el pool de hilos de .NET" http://www.microsoft.com/spanish/msdn/articulos/archivo/080302/voices/programacion.asp (15/1/2006)
6. "Arquitectura de aplicaciones de .NET: Diseño de aplicaciones y servicios". Diciembre 2002. http://www.microsoft.com/spanish/msdn/latam/ (12/12/2005)
7. Diseño e implementación de una Arquitectura multiplataforma para el estudio de Motores de búsqueda en Internet. 2003.
8. Soto López, Nilet María, Saborit Ramírez, Yunier: *"Propuesta para un sistema de catalogación y recuperación de recursos de información"*. Trabajo para optar por el título de Ingeniería Informática, Instituto Superior Politécnico "José Antonio Echeverría", Ciudad de La Habana, junio 2004.
9. Koster, Martijn: "Guidelines for Robots Writers". 1993. http://www.robotstxt.org (16/1/2006)
10. Koster, Martijn: "Robots in the Web: threat or treat?". Abril 1995. http://www.robotstxt.org (16/1/2006)
11. "All About Search Indexing Robots and Spiders" http://www.searchtools.com (18/1/2006)
12. "Search Indexing Robots and the Robots META Tag" http://www.searchtools.com/robots/robots-meta.html (19/1/2006)