Universidad de las Ciencias Informáticas

"Facultad de Entornos Virtuales"

Título: Sisweb

Sistema Bot-Web Buscador e Indexador de Información

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Carlos de Brito Salazar

Ernesto Macías Romero

Tutor: Alain Hernández Castillo

Ciudad de La Habana, Junio del 2008

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____	_____
Carlos de Brito Salazar	Ernesto Macías Romero
Firma del Autor	Firma del Autor

_____

Alain Hernández Castillo

Firma del Tutor

DATOS DE CONTACTO

Ing. Alain Hernández Castillo (alainhc@uci.cu)

Graduado de Ingeniería Informática en la CUJAE en el año 2006, con más de 4.76 puntos de promedio general.
Ha ejercido como profesor de las asignaturas de Gestión de Software, Bases de Datos, Programación web, y Teleinformática 2.
El curso 2006-2007 fue cotutor de una tesis de diploma.

AGRADECIMIENTOS

A la dirección de la Revolución Cubana, a nuestro Comandante Fidel Castro y a la UCI, por permitirnos formar parte de este proyecto y contribuir a nuestra formación profesional.

Carlos y Ernesto

A mi madre, que ha dado todo por mí, a mis abuelos Mirta y Fernando que han hecho posible la familia que es hoy.

A mis Tíos Arelis y José, por su apoyo incondicional.

A mi Primo Fernando.

A los amigos de aula, de grupo, de proyecto y de siempre Yunerkis y Lázaro, a los compañeros que aportaron al desarrollo de este trabajo y a las personas que influyeron en su realización.

Carlos

A mi madre, a mis segundos padres Antonia y Eusebio que son los mejores abuelos del mundo, que siempre me dieron su apoyo y comprensión cuando más lo necesitaba, y a toda mi familia que siempre me apoyaron y ayudaron.

A mi novia Mailyn por estar siempre a mi lado y tener una sonrisa para mí.

A mis compañeros de aula, en especial a William, Alejandro, Amaury, Julio y Gelson que me aclararon las dudas en los momentos difíciles.

A todas las personas que de una forma u otra han hecho posible este sueño.

Ernesto

DEDICATORIA

A mi madre por el apoyo brindado, a toda la familia que siguió de cerca mis estudios en la universidad.

A los amigos que no dejaron de influir en cada momento.

Carlos

A mi madre y toda mi familia que siempre me ayudaron e hicieron posible este sueño.

A mi novia Mailyn que me apoyó siempre.

A mis compañeros de aula y profesores, en fin a todas las personas que he conocido en éstos 5 años y que han aportado su granito de arena.

Ernesto

RESUMEN

En la actualidad en la Universidad de Ciencias Informáticas el uso de las tecnologías de la información y las comunicaciones orientadas a la actividad educativa, ha posibilitado el surgimiento de una intranet y diversos sitios de difusión de información, por este motivo existe un gran volumen de información que debe ser localizada y organizada.

Producto a lo expuesto anteriormente surge la idea de crear un sistema bot-web buscador e indexador de información con capacidad de integración en diferentes gestores de base de datos basado en un bot-web de búsqueda e indexación automática de sitios web publicados en la intranet de la UCI.

El sistema contiene un conjunto de funcionalidades orientadas a la gestión de URLs y el contenido relacionado, implementa los dos tipos de rastreos de red existentes (en amplitud y en profundidad) y almacena el contenido en una estructura de índice invertido, además contiene funcionalidades de gestión que optimizan y muestran el estado de los datos como resultado de los procesos del sistema.
Se desarrolló un bot-web donde se integran las funcionalidades de búsqueda e indexación de información web que constituyen el principal objetivo de solución.

La realización de este trabajo ha posibilitado obtener un producto para contribuir al desarrollo de sistemas de búsquedas más abiertos y eficientes que permitan cumplir las necesidades actuales de la comunidad universitaria UCI.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad el desarrollo de tecnologías de información y comunicación junto al surgimiento y evolución de las redes de computadores ha propiciado una amplia difusión de información. La Universidad de Ciencias Informáticas (UCI), por medio de una intranet orientada al desarrollo de actividades educativas posibilita un conjunto de sitios web cada ves más amplio como resultado de la publicación de información necesaria para un gran número de usuarios, por este motivo existe un volumen de información que aumenta y cambia continuamente junto a la necesidad de obtener información localizada y organizada.

Debido a lo anteriormente planteado la creación de un sistema bot-web buscador e indexador de información con capacidad de integración en diferentes gestores de base de datos basado en un bot-web de búsqueda e indexación automática de sitios web publicados en la intranet de la UCI, facilitaría a sistemas informáticos buscadores procesos más abiertos y eficientes orientados a disminuir el tiempo de búsqueda de la información mas actual en la web de la comunidad universitaria UCI.

Los **antecedentes** de este trabajo se originan en el año 1980 cuando surgió una primera impresión de la web y Tim Berners-Lee se propuso lo que diez años después sería lo que hoy conocemos como sitios web. A finales de 1990 comenzó a aplicar sus ideas creando el primer servidor web NeXT, el primer navegador de internet, llamado World Wide Web (que también era un editor HTML) y la primera página web.

El contenido de la web fue aumentando continuamente imposibilitando la recuperación de información, en junio de 1993 desde el MIT(Massachusetts Institute of Technology) y con Matthew Gray a la cabeza, se desarrolló World Wide Web Wanderer, el primer robot que rastreaba la red con el objetivo de medir su longitud. Ese robot se amplió posteriormente a una versión mejorada llamada Wandex, que podía leer direcciones URL. Esta versión tuvo serios problemas de infraestructura y velocidad debido a lo que en aquella época se consideraba un tráfico enorme: cientos de visitas diarias. El próximo buscador, surgió en octubre del mismo año, fue AliWeb (Archie Like Indexing on the Web). Creado por Martijn Koster, lo que le hacía novedoso era el hecho de que indexaba los metatags de las páginas que se le daban a su índice. Tras estos primeros rastreos en internet, Koster propuso alguna que otra sugerencia para lo que sería el fichero robots.txt, que limita la acción de búsqueda en sitios web.

A partir de este momento comenzaron a desarrollarse los primeros robots rastreadores modernos, como Jumpstation, que indexaba el título, URL y cabecera del sitio, como lo hacía también el robot World Wide Web Worm creado por Oliver McBryan en 1994. Aunque era un gran avance que indexaran, aún quedaba un proceso pendiente, y el más importante en la concepción de búsqueda futura. Encontrarlo todo es fácil, porque está, pero su posicionamiento es más difícil. Es decir, no aplicaban ningún algoritmo en la ordenación de resultados, sino que los mostraban según el orden de indexación. En diciembre de 1994 RBSE (Repository-Based Software Engineering) comenzó a aplicar un ranking según la relevancia de las páginas en relación con las palabras buscadas.

Al mismo tiempo iban apareciendo algunos directorios como EINet Galaxy. Pero fue en abril de aún 1994 cuando se crearía por parte de Jerry Yang y David Filo lo que hoy es una de las potencias de internet: Yahoo, anteriormente conocido con el nombre de Jerry's Guide to the World Wide Web, y que era una colección de las páginas webs favoritas. El gran problema de Yahoo es que de buen comienzo era un directorio que actualizaba su base de datos con webs nuevas únicamente a partir de las introducidas por sus constructores, haciéndose rápidamente necesario la creación de un software que clasificase autónomamente las respectivas páginas.

De todas formas, estos buscadores eran aún demasiado sencillos y se caracterizaban por ser simples experiencias y no fue hasta el 20 de Abril de 1994 que Brian Pinkerton, desde la Universidad de Washington, presentase WebCrawler. La gran diferencia que presentaba este buscador era que indexaba los sitios que rastreaba de forma íntegra, siendo la relevancia de resultados mucho mayor. Gracias a la adquisición de dos patrocinadores, DealerNet y Starware, esta empresa podía mantenerse exclusivamente de la publicidad.

Lycos surgió en julio del mismo año. Creado por Michael Mauldin en la Universidad de Carnegie Mellon, incluía un algoritmo interesante que estudiaba la proximidad entre palabras. Una desventaja de Lycos (quizás era una estrategia de ahorro de memoria, espacio y dinero) era que no indexaba de forma completa las páginas, sino que sólo indexaba las veinte primeras frases, las doscientas primeras de la cabecera y un grupo de las cien más relevantes de todo el documento. De todas formas, Lycos se destacaba por la cantidad de documentos indexados: se lanzó con 54.000 documentos; en agosto de 1996 tenía 394.000, en enero de 1995 1,5 millones, y en noviembre de 1996 llegó a la cifra de sesenta millones de documentos, convirtiéndolo en el motor de búsqueda más destacado.

En diciembre de 1995 seis estudiantes de Stanford lanzaron Excite, gracias al proyecto Architext iniciado en 1994 que introdujo nuevos conceptos en la búsqueda. Este buscador se basaba en un complicado algoritmo que intentaba crear un sistema parecido a los sinónimos mediante estadísticas entre las relaciones de palabras, de forma que se podían hacer búsquedas obteniendo resultados aunque la misma palabra no apareciese en los resultados.

El siguiente gran lanzamiento fue AltaVista ya que, con su aparición, proponía unas mejoras increíbles, ofreciendo un ancho de banda prácticamente ilimitado, permitiendo búsquedas en lenguaje natural , consultas avanzadas admitiendo operadores lógicos (AND, OR...), añadir o eliminar direcciones URL en un plazo máximo de veinticuatro horas, comprobar los enlaces entrantes a un sitio web y proponía algo muy novedoso, y era la posibilidad de buscar a través de los nombres de las imágenes y de otros archivos multimedia.

En esta época comenzaron a aparecer también los llamados meta-buscadores, o sea, buscadores que recopilaban información de otros buscadores. En 1995 apareció el primero de ellos a manos de Eric Selberg y Oren Etzioni, ambos estudiantes de la Universidad de Washington. En este caso, este buscador cuyo problema era la velocidad, devolvía información de Lycos, AltaVista, Yahoo!, Excite!, WebCrawler e InfoSeek, y su nombre era MetaCrawler.

Poco después, el 20 de mayo de 1996, Paul Gauthier y Eric Brewer, desde la Universidad de Berkeley, lanzan HotBot que, con su motor Inktomi, consiguieron llegar a un acuerdo con la empresa Wired, que lo llevó a la fama. Se consideró el primer buscador capaz de indexar los millones de sitios web que había en ese momento.

El proyecto Google comenzó a desarrollarse en 1996 en la Universidad de Stanford a manos de Larry Page y Sergey Brin, llamándose inicialmente BackRub debido a la tecnología que utilizaba. En septiembre de 1997 el dominio google.com fue comprado, y el 7 de septiembre de 1997 se creaba Google Inc. Los hechos de que Google se hiciese tan rápidamente famoso fueron dos: una interfaz muy sencilla y unos resultados de búsqueda perfectos gracias a la tecnología PageRank, ideada por el mismo Larry Page (de ahí el nombre) y patentada el 4 de septiembre de 2001. Radicalizando e innovando el mundo de los buscadores, Google implementaba un sistema mediante el cuál no sólo se

tenían en cuenta los factores de la página en la que se buscaba, sino que se le daba importancia también a los factores externos a la página pero que tuvieran relación a ella.

En 1998 apareció por fin el buscador de Microsoft, MSN Search, que utilizaba los datos de Inktomi. Nació también una revolución, o sea, el Open Directory Project (ODP), primer directorio elaborado de forma conjunta por personas. Creado por Rich Skrenta y Bob Truel y llamado inicialmente Huno, pasó a llamarse Newhoo el 5 de junio de 1998. Pero sólo cuando fue adquirido por Netscape en octubre del mismo año pasó a ser el ODP, con un total de 100.000 direcciones y 4.500 editores.

A mediados de 1999 apareció en el mercado AllTheWeb. Utilizaba la tecnología de Fast, una empresa noruega que venía de la Norwegian University of Science and Technology. En febrero de 2003 fue comprado por Overture y ésta, a su vez, por Yahoo! en marzo de 2004, reduciendo alguna de sus aplicaciones. Su base de datos, cuando fue adquirido por su ya mencionado actual propietario, llegaba a la cifra de 3.300 millones de páginas web indexadas.

En 1999 aparecía también otro gigante de la red: Baidu, motor de búsqueda chino y punto de referencia hasta la actualidad debido a la presión que ejerce el gobierno chino sobre internet.
Acercándonos al desenlace de esta historia, en el 2000 apareció Teoma, tecnología punta creada por Apostolos Gerasoulis, utilizando un sistema para organizar los sitios en base al Subject-Specific Popularity (actualmente Expert Rank). Este sistema de ranking de sitios difería del PageRank de Google en que analizaba los enlaces hacia un sitio web en un contexto en el que se daba una calificación a una página según el tema tratado. El 11 de septiembre de 2001 fue comprado por AskJeeves.

Los buscadores anteriores a Google (salvo Altavista), eran directorios de páginas web, esto significaba que debías dar de alta tu sitio en el directorio, indicar por qué palabras clave deseabas ser encontrado, tenias que redactar la definición de tu página web, entonces cuando alguien buscaba una palabra que estuviera incluida en las palabras clave o en la definición dada el buscador mostraba tu página. En la actualidad los buscadores funcionan con unos índices automáticos que se actualizan continuamente mediante arañas web.

- Como **aportes prácticos** esperados del trabajo, se pretende obtener un sistema con capacidad de integración en diferentes gestores de base de datos compuesto por un bot-web que

posibilite la búsqueda e indexación automática y eficiente de sitios web publicados en la intranet de la UCI.

El **objeto de estudio** de este trabajo es el desarrollo de buscadores web de información basados en bots-web de búsqueda. A partir de este podemos determinar que el **campo de acción** es el desarrollo de bots-web de búsqueda e indexación de información para buscadores de internet.

Objetivos Generales:

- Diseñar e implementar un sistema bot-web buscador e indexador de información con capacidad de integración en diferentes gestores de base de datos basado en un bot-web de búsqueda e indexación automática y eficiente de sitios web publicados en la intranet de la UCI.

Objetivos Específicos:

- Analizar el funcionamiento de buscadores web basados en bots web de búsqueda.
- Analizar el funcionamiento de bots-web de búsqueda.
- Analizar los procesos de búsqueda e indexación de información.
- Desarrollar un sistema basado en un bot-web para buscar e indexar documentos web.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En este capítulo se explica el estado actual de los buscadores de internet basados en bot-web a diferentes escalas, y se describen los conceptos principales que han sido objeto de investigación, así como las principales tendencias, tecnologías, metodologías, plataformas de desarrollo y herramientas que hicieron posible la realización del presente trabajo.

1.1 Los buscadores.

Los buscadores son programas que permiten buscar en documentos previamente analizados por aplicaciones que encuentran de forma exacta palabras en textos extensísimos, los mismos, deben diferenciarse de los directorios. Estos últimos son listas compuestas por links que hacen referencia a diferentes páginas web organizadas jerárquicamente y por temas: ciencia, arte, cultura, etc. No son motores de búsqueda y por lo tanto no muestran listas de páginas según palabras clave. Pueden llegar a tener un millón de secciones y son muy útiles si se tiene paciencia. Es decir, los directorios no dejan de ser listas enormes creadas por personas ayudadas o no de algún software específico de catalogación de sitios por temas, por ejemplo. Muchos de ellos son muy generales, y ofrecen distinciones entre idiomas, regiones, etc., y sus listas son elaboradas según distintos métodos:

- Libre suscripción: El directorio no cobra por contener hipervínculos hacia páginas.
- Links recíprocos: el sitio listado debe tener un link que apunte hacia el directorio para que sea incluido en la lista.
- Links premiados: los links de los sitios que el directorio considera más importantes serán más visibles.
- Links que no podrán ser seguidos por los motores de búsqueda.

Este último factor es el que interesa para entender la distinción. Los buscadores se dedican a analizar páginas web para que luego todo usuario pueda buscar y encontrar palabras o hasta frases en el texto íntegro de dichos documentos. Analizan sitios de la WWW, cuyas direcciones URL no podrán inventar sino que las extraerán a partir de los hipervínculos de las páginas ya analizadas que apuntan hacia otras no analizadas. Obviamente, una buena fuente de links se puede encontrar en los directorios, que

catalogan las páginas por temas pero no muestran su contenido. Al ser competencia directa de los buscadores, pueden no permitir que éstos los analicen: todo link en un directorio apunta hacia una página que la ha indexado por considerarla interesante.

Los motores de búsqueda son, en definitiva, programas informáticos formados por complejos algoritmos de búsqueda e indexación y enormes bases de datos que abarcan gran parte del contenido de la inmensa red que nos rodea, internet, y que son utilizados para que todo usuario pueda acceder a una información global. Pablo Castells comenta que (1) "aunque es sumamente difícil medir el tamaño de la web, se estima que hoy en día unos 1.000.000.000 usuarios utilizan la WWW, y que ésta contiene del orden de 4.000.000.000 documentos, un volumen equivalente a entre catorce y veintiocho millones de libros.

Como dato comparativo, la asociación American Research Libraries, que agrupa a unas cien bibliotecas en los Estados Unidos, tiene catalogados 3.700.000 libros. Estas cifras incluyen sólo datos relacionados con la llamada web superficial, formada por los documentos estáticos accesibles en la WWW. Se ha calculado que la web profunda constituida por las bases de datos cuyos contenidos, no directamente accesibles, se hacen visibles mediante páginas generadas dinámicamente, puede contener un tamaño de información varios cientos de veces mayor, y de mucha mejor calidad, que la superficial, y crece a un ritmo aún mayor que ésta.

Se estima que el tamaño de la profunda ha superado ya al volumen total de información impresa existente en todo el planeta" (2). La eficiencia de los buscadores es altísima ya que no sólo consiguen abarcar gran cantidad de información sino que saben reconocer qué se está buscando. ¿Se imaginan el tiempo que necesitaría una persona para encontrar todos los libros que contengan una palabra determinada en una pequeña biblioteca de cien mil libros? Que los buscadores lo consigan y además sepan procesar la información que encuentran es lo que los hace indispensables. Otro estudio, How Much Information 2003, muestra también que el tamaño de la WWW es inabarcable, admitiendo una cantidad de 532.897 terabytes, y esto es la misma cantidad de información que en aproximadamente $1.2*10^{13}$ trabajos iguales que éste, o sea: 12.000.000.000.000 trabajos. Los buscadores deberán tratar con distintos formatos de datos. En la siguiente figura se muestra el esquema de los porcentajes:

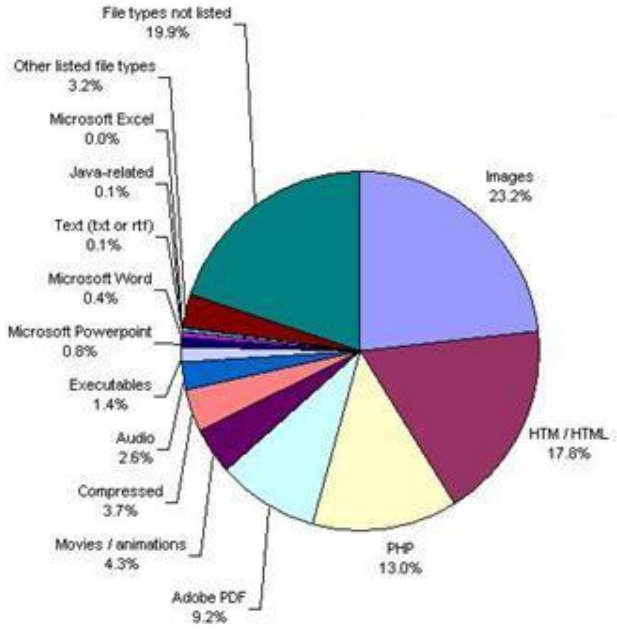

Figura 1 Composición de la web

El mismo estudio demuestra que el 29% de la población americana con acceso a internet utiliza los motores de búsqueda, realizándose según el portal SearchEngineWatch.com un total de 213 millones de búsquedas al día y 6.400 millones en Marzo del 2006, en Estados Unidos. Por lo tanto, como vemos los buscadores son una herramienta cada vez más indispensable, y no sólo en internet puesto que, como decía un artículo de El Periódico aparecido el mes de octubre de 2006, "Microsoft ha integrado la búsqueda de información en todo el sistema operativo (Windows Vista). Desde un documento o el escritorio se puede buscar por palabras clave o medias palabras", de manera que cuando el usuario desee encontrar un documento que se llame de tal manera o contenga una palabra, el motor de búsqueda del sistema lo encontrará.

En palabras de Larry Page (cofundador de Google), "ahora se puede buscar a lo equivalente a 70 millas de altura en papel en menos de un segundo. Creemos que es fantástico". De esta forma se observa que, como explica Pablo Gallardón en El secreto de Google y el Álgebra Lineal, el diseño de un buscador a gran escala es un problema de ingeniería matemática, cuestiones computacionales, resumido en: cómo almacenar la información, cómo actualizarla, cómo buscar eficientemente en las bases de datos, etc.

1.2 El funcionamiento de los buscadores

Los buscadores, normalmente, no buscan en internet directamente. Cada uno de ellos busca en su extensa base de datos residente en sus servidores. En esas bases de datos almacenan íntegramente el texto de cada página, de manera que cuando uno usa un buscador está buscando de manera indirecta en una copia idéntica de la página original. Esta búsqueda se hace mediante una interfaz, que es el elemento que permite el acceso al motor de búsqueda y en la que el usuario introducirá la información a buscar y donde posteriormente recibirá los resultados. Dicha información será tratada por el programa de búsqueda, yendo éste a analizar la base de datos.

Estas bases de datos son actualizadas automáticamente por los llamados robots de búsqueda. Los mismos son programas residentes en un ordenador desde el que envían peticiones a otros servidores. Como no saben llegar a una URL de forma autónoma, comienzan la tarea de rastreo a partir de una página web determinada. De ella extraen todos los hipervínculos hacia otras páginas que serán analizadas de igual forma y de las que también extraerán sus links hacia otras webs para posteriormente analizarlas. El encargado de detallar la base de datos, el programa que trabaja directamente con ella, es el indexador. Su objetivo principal es el de ordenar la información recolectada por el robot de forma específica y detallada para luego facilitar la ordenación de resultados en una búsqueda concreta. En la siguiente figura se muestra el funcionamiento de un buscador.

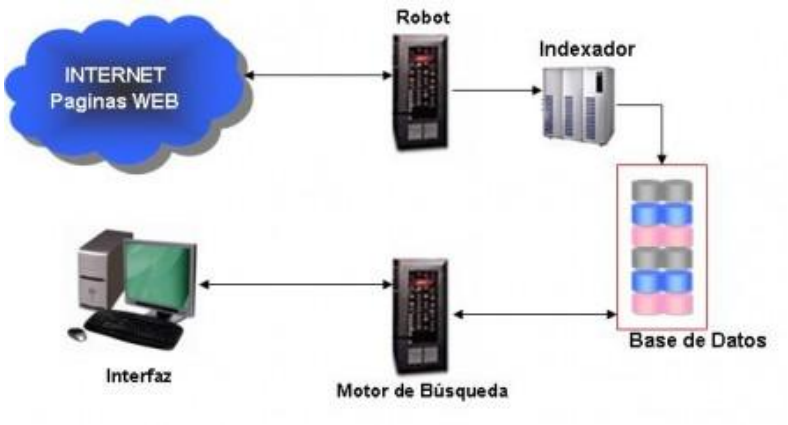

Figura 2 Esquema de funcionamiento de un buscador

1.3 Los componentes de los buscadores

Los buscadores son sistemas informáticos orientados a búsquedas de información web, un ejemplo son los buscadores de internet (algunos orientados a búsquedas sólo en la WWW y otros a búsquedas además en News, Gopher, FTP, etc.), están formado por cinco componentes:

- Interfaz: Es el componente en el que el usuario podrá introducir las palabras a buscar y donde posteriormente se mostraran los resultados.
- Programa Buscador: Implementa métodos de búsqueda de información
- Programa Rastreador: Implementa métodos de exploración de archivos almacenados en servidores web (bots de búsqueda)
- Programa Indexador: Implementa métodos de indexación de información
- Base de Datos: Contenedor de datos escalable que posibilita tratar con gran cantidad de información. Debe presentar una función básica, y es la existencia de una interfaz que todos los componentes que vayan a hacer uso de ella (web crawler, indexador y programa de búsqueda) sepan interpretar.

1.3.1 Bot-web buscador o Crawler web

Un bot-web buscador también denominado web crawler, web spider o web wanderer es un programa orientado a inspeccionar las páginas del World Wide Web de forma automática y metódica. Los bot - web rastreadores son utilizados por los motores de búsqueda para actualizar sus bases de datos, creando una copia de todas las páginas visitadas para poder luego analizarlas y ofrecer una búsqueda más rápida.

El proceso de rastreo consiste en darle al programa un grupo de direcciones URLs iniciales, la araña web descarga estas direcciones, analiza las páginas y busca enlaces a páginas nuevas. Luego descarga las mismas, analiza sus enlaces, y así sucesivamente.

1.3.2 Crawler parcial o completo

Un crawler basado en una estructura Batch-mode puede estar configurado tanto para llevar a cabo un rastreo completo como para analizar sólo ciertas páginas. Hay básicamente dos causas que dificultan el proceso de rastreo de la red: su gran volumen de información y su volatilidad, porque constantemente se crean, eliminan o cambian sitios web. Esto provoca que la araña rastreadora pueda sólo descargar porciones de la red dentro de un tiempo determinado, cosa que hace necesario establecer un cierto orden de prioridad. Además, el constante cambio de internet puede provocar que cuando un robot buscador está descargando la nueva información de un sitio determinado ya en ese momento se añadan otros contenidos nuevos.

Como el proceso de rastreo no puede ser infinito y exige un determinado coste, es absolutamente necesario que el proceso ejecute de un modo eficiente, debiendo saber los spiders en cada paso cual será el siguiente. Además, tal y como explican los fundadores de Google, implementar eficientemente un rastreador de la red de redes es una tarea muy complicada, pues en cada momento y en cada página puede suceder un problema que cause un comportamiento inesperado del robot. De esta manera, el comportamiento de los bots web buscadores está determinado por cuatro procesos, con tal de garantizar su correcto funcionamiento:

- El proceso de selección, que determinará qué páginas descargar y en qué momento.
- El de actualización, que establece cada cuanto tiempo deberá visitarse una página web para comprobar si ha sufrido cambios.
- El proceso que evita la sobrecarga de los sitios web.
- El proceso que coordina la distribución de los diferentes spiders.

1.3.3 Crawler Batch-mode o constante

Un Batch-mode crawler es ejecutado periódicamente (cada mes, por ejemplo), y se le permite rastrear durante un tiempo determinado o simplemente hasta que ha realizado toda la tarea programada. En contraste, un crawler constante rastrea sin ninguna pausa, enviando continuamente actualizaciones o páginas nuevas al almacén de datos.

1.4 Procesos de Crawlers

Los procesos de crawlers se pueden clasificar principalmente en cuatro tipos que definen el funcionamiento del robot web.

1.4.1 Proceso de selección

Debido a la gran cantidad de información contenida en la red, los motores de búsqueda cubren solo una parte de ella. Un estudio realizado en 1998 por Lawrence y Giles (NEC Research Institute) demostraba que, tras analizar a seis buscadores, ninguno de ellos cubría más que el 57.5% de la web. Por lo tanto, como un agente de búsqueda sólo descarga determinadas páginas, es absolutamente necesario que sean las más relevantes. Esto requiere el establecimiento de unos parámetros de importancia para priorizar descargas.

La importancia de una página vendrá determinada por una función que incluirá factores como su calidad, su popularidad en links o visitas, entre otros. Obviamente el *spider* deberá tener en cuenta estos factores trabajando no con toda la información sino simplemente con la que conoce. Existen algunos métodos para analizar la web cuando no se dispone de un historial de información de los sitios a analizar:

- Breadth-first (BFS)*:* Es una teoría que se explica muy esquemáticamente con un gráfico. Propone empezar el proceso de rastreo por el primer nodo y explorar a todos sus vecinos de forma secuencial hasta llegar al último.

- Depth first search (DBS)*:* Esta estrategia propone expandir uno de los nodos a su nivel más profundo y sólo cuando llega a un camino sin salida regresa a niveles menos profundos. En la siguiente figura se muestran los recorridos BFS y DBS.

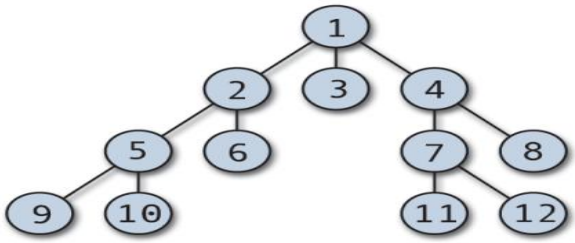

Figura 3 Espacio de recorridos
Recorrido en amplitud de la web (1,2,3,4,5,6,7,8,9,10,11,12)
Recorrido en profundidad de la web (1,2,5,9,10,6,3,4,7,11,12,8)

- Backlink-count: Esta estrategia rastrea primero las páginas a las que más links apuntan, de manera que la página que se rastreará será la más señalada por las páginas ya descargadas.

- Batch-PageRank*:* Esta estrategia calcula una estimación del que será el *PageRank* de las K siguientes páginas a analizar, siendo generalmente K = 100.000.

- Partial-PageRank*:* Parecido al anterior, realizado durante el proceso de actualización de los actuales PageRanks, asignando de manera temporal un ranking correspondiente en un sitio a la suma de los PageRanks de las web que apuntan hacia éste dividido por el número de links.

- Online Page Importance Computation (OPIC)*:* Estrategia complicada que se basa en una cantidad de 'crédito' asignado de forma igual a las páginas. Al analizar una, se distribuye uniformemente entre los sitios hacia los que hace referencia, de manera que el que sume más 'efectivo' será el siguiente en ser analizado.

Si en cambio se cuenta con un historial de la información, existen otros métodos de rastreo, a partir o no de la clasificación anterior. Un estudio llevado a cabo por Ntoulas (UCLA Computer Science), Junghoo Cho (UCLA Computer Science) y Christopher Olston (Carnegie Mellon University) pone de manifiesto que la estructura de los links en la WWW es más dinámica que ella misma.

Por lo tanto, a la hora de hacer el rastreo se podrá suponer un PageRank aleatorio para cada web, uno que valga cero y rastrear según la fecha de indexación, o hasta establecer también relación entre los diversos PageRanks: el valor de prioridad de una página será el mismo que el de aquella hacia la que apunta. Se puede también comenzar a analizar por las páginas que ocupen más, o por las que ocupen menos.

1.4.2 Proceso de actualización

Como se ha dicho, el rasgo más notorio de la web es su alto dinamismo, cosa que dificulta el correcto rastreo y lo ralentiza: cuando una araña rastreadora ha finalizado su proceso de análisis, ya otros cambios (actualizaciones, creaciones o eliminaciones) pueden haber sucedido. De manera que, cuando un crawler ha seleccionado, inspeccionado y descargado las páginas que considera importantes, debe revisar periódicamente la veracidad (si existe o no y de qué manera) de esa información.

Junghoo Cho y Héctor García-Molina (actual director del centro de investigación de Yahoo en Barcelona) presentan en el año 2000 un estudio sobre dos conceptos ciertamente muy razonables: la edad de una página, que la antigüedad de una web, y su frescura, que nos indica si la copia del sitio es exacta o no. Intuitivamente, es lógico pensar que una colección de páginas es exacta cuando tiene muchas web actualizadas. Consideremos dos colecciones A y B, ambas conteniendo las mismas veinte páginas. Si A mantiene diez páginas actualizadas, mientras que B quince, podemos considerar más exacta la colección B. Pero también existe el factor ‘edad’. En cambio, la colección A será más actual si el crawler la revisa y actualiza una vez al día mientras que a B lo hace cada tres días. Como se puede comprobar son dos conceptos altamente relacionados y son el claro objetivo en el proceso de actualización: las webs contenidas en las bases de datos deben estar al día.

De esta manera, hay dos estrategias principales en el proceso de actualización: la revisión uniforme y la revisión según unos grados de proporcionalidad. La primera establece que el spider visitará todos los sitios según una misma frecuencia de tiempo. En cambio, la segunda hará que el crawler visite cierta página según el grado de probabilidad de que ésta cambie. Por ejemplo, si una araña rastreadora visita una web cada día durante un mes y detecta diez cambios, notará que un cambio se produce cada tres días, pudiendo establecer así cuándo volverá a analizarla.

1.4.3 Proceso para la no-sobrecarga de sitios web

Los web crawlers pueden reunir y tomar información mucho más rápido que un simple usuario, pudiendo repercutir directamente en los sitios web. Por lo tanto, tienen que, de alguna manera, obedecer a ciertas restricciones. La más importante y que deben tener siempre en cuenta es que no han de saturar las páginas que visitan. Como se verá más adelante, estos programas se ejecutan en muchos ordenadores a la vez, y se establece así una conexión en paralelo, pudiendo con sus peticiones bloquear a los servidores. Para evitar esta sobrecarga están en cierta manera obligados a descargar cierta página de determinado sitio en un tiempo establecido, y esperar otro tiempo para volver a entrar y descargar de esa misma página.

Se supuso inicialmente que 1 minuto era el tiempo ideal entre cada acceso a la página deseada. Este tiempo implica esperar meses para descargar miles de páginas, así que en un estudio realizado por Heydon y Najork en 1999 utilizaban un crawler que adaptaba sus tiempos de espera: si se tardaba t segundos para descargar un documento de una página, el crawler esperaría 10t segundos para la segunda visita. Por otra parte, Cho y García-Molina redujeron ese tiempo de espera a 10 segundos en una investigación que llevaron a cabo en 2003.

Hasta aquí se ha analizado cuándo o cada cuánto pueden rastrear las webs los robots, pero, ¿qué páginas pueden rastrear? Existe el Protocolo de Exclusión de Robots. Fue desarrollado para determinar el comportamiento de las arañas rastreadoras en la WWW por parte de sus propietarios para definir qué podían analizar y qué no podían. El protocolo permite a los propietarios definir instrucciones en un fichero que llamarán robots.txt que marcarán el comportamiento en esa web de los robots de búsqueda.

1.4.4 Proceso de coordinación

En esta sección se estudiarán los métodos para coordinar a distintos crawlers para mejorar su rendimiento; es necesario, tener claro que usar más de un robot rastreador puede presentar tres problemas:

1- Cuando múltiples procesos se ejecutan en paralelo bajo el objetivo de descargar páginas web más eficientemente, es posible que distintos procesos descarguen la misma página más de una vez.

2- Los crawlers analizan primero las páginas que consideran importantes para mejorar la calidad del contenido de su base de datos; sin embargo, cuando se ejecutan crawlers en paralelo, cada proceso puede no saber qué parte de la web se ha descargado entre todos y, por este motivo, cada araña buscadora puede analizar la red según la imagen que por sí sola tiene de la web, siguiendo los pasos que tenga programados sin tener en cuenta los movimientos que han realizado los demás.

3- Para prevenir la repetida descarga de las páginas, o para mejorar la calidad del contenido, los procesos de rastreo necesitan comunicarse cada cierto tiempo entre ellos, comunicación que puede crecer considerablemente según el número de procesos.

El rastreo en paralelo presenta ventajas muy considerables:

1- Mayor escalabilidad, pues debido a la gran cantidad de información de la web es necesario establecer diversos procesos de crawling para agilizar el rastreo.

2- La dispersión de la carga de la red, porque robots que se ejecuten en paralelo son necesarios cuando una red no puede soportar un rastreo a gran escala: por ejemplo, asumamos que un crawler en Francia analiza y descarga una página de los Estados Unidos. Para ser descargada, esta página deberá recorrer múltiples redes para viajar de Estados Unidos a Francia. En cambio, si un proceso de crawling en Europa descarga todas las páginas europeas y uno en Asia descarga todas las páginas de su zona, la carga general de la red se reducirá porque las páginas se desplazarán por redes locales.

Múltiples procesos de crawling, todos ellos conectados en paralelo, hacen necesaria una arquitectura bien definida. Cada procedimiento funciona independientemente, es decir, no deja de ser un crawler más cuyo comportamiento vendrá determinado por o determinará el comportamiento de otros *spiders*. Estos procedimientos pueden llevarse a cabo bajo una misma red o de forma distribuida. Los primeros se comunican entre ellos a partir de una conexión rápida (LAN) y se caracterizan por utilizar la misma red local cuando descargan páginas remotas. Por otro lado, cuando los diversos crawlers actúan distantes geográficamente conectados entre sí por internet o por una red de área amplia decimos que son procesos que se ejecutan de forma distribuida y en los que la comunicación entre ellos deberá realizarse cada cierto tiempo, siendo muy importante el cada cuándo y el cuánto deben comunicarse.

Cuando diversos procedimientos de rastreo descargan páginas simultáneamente, puede existir el caso en que una misma página sea descargada más de una vez. Para evitarlo deberán coordinarse correctamente. Hay tres formas de hacerlo:

- Independiente: Sencillamente, cada procedimiento podrá descargar páginas independientemente de lo que hagan los demás y bajo ninguna coordinación. Esto es, cada procedimiento empezará su tarea de rastreo a partir de las URLs iniciales y seguirá los *links* sin consultarlo con los otros procedimientos.

- Asignación dinámica: Se sigue este tipo de coordinación cuando existe un coordinador central que divide de forma lógica la web en particiones muy pequeñas y dinámicamente asigna cada una de ellas a un procedimiento. Por ejemplo, imaginemos que tenemos un coordinador que divide la red según el nombre del dominio de la *URL*, de forma que http://ejemplo.com/Index.html y http://ejemplo.com/Contenidos.html pertenecerían a la misma partición. Luego, durante un rastreo de la red, el centro de coordinación decide en cada momento qué partición será la siguiente en ser analizada (en este caso, http://ejemplo.com) y la envía a uno de los procedimientos. Hecha esta petición, el crawler analizará y descargará el sitio y de éste extraerá todos sus hipervínculos. Si uno de ellos apunta hacia otra partición, el procedimiento informará al coordinador. Cabe decir que la red puede ser fraccionada de muchos modos, cosa que repercutirá notablemente en la comunicación entre procesos y centro de coordinación o simplemente en la forma de rastrear la WWW.

- Asignación estática: Entendemos por asignación estática cuando la web es fraccionada y asignada a cada procedimiento antes de empezar a rastrear. En este caso no se necesita un coordinador, porque cada crawler sabe qué debe analizar.

  Existen tres modos de analizar la web a partir de este tipo de asignación:

  1- Cada crawler descargará sólo las páginas de su partición.

  2- El crawler descargará primero las páginas de su partición, pero seguirá también los hipervínculos que apunten hacia otra.

  3- Los distintos procedimientos se intercambian *links* correspondientes a páginas de sus respectivas particiones.

Por otra parte, hay dos formas posibles para la correcta comunicación de procesos mucho más eficientes que la comunicación directa y sin ningún rigor. La primera establece que, en vez de informar directamente al correspondiente proceso cuando se encuentra un link apuntando hacia otra partición, el crawler puede esperar cierto tiempo con tal de conseguir una lista de URLs apuntando hacia una partición que no es la suya.

Cuando ha recolectado una lista de *k* páginas, las envía al proceso correspondiente. La segunda, en cambio, se entiende ante todo sabiendo que pocas páginas son las que tienen una gran cantidad de links apuntando hacia ellas. Por ésto, se reduciría notablemente el número de URLs intercambiadas si se enviasen únicamente las más importantes, es decir, aquellas con más hipervínculos que las señalan. (El segundo método es sólo posible cuando ya se tiene una imagen de la estructura de la web).

1.5 La Indexación

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 (términos de indexación) de un lenguaje documental o natural, orientados a facilitar la posterior recuperación de los documentos indexados.

El problema fundamental que se halla al intentar indexar la web es que es necesario representar gráficos de la red con millones de nodos y, en consecuencia, crear índices a partir de millones de páginas. Por lo tanto, se deberá tener mucho cuidado a la hora de crear un índice a gran escala para conseguir su eficiencia. Se presenta a continuación una breve descripción de cada tipo de índice:

- **Índice basado en los hipervínculos:** Para construir un índice basado en los links una porción de la web es tratada como a un esquema con nodos y vectores. Cada nodo en esa estructura es una página y un vector que desde una página *A* apunta a otra página *B* representa un hipervínculo en la página *A* que señala a la página *B*. Normalmente, la estructura más utilizada por los algoritmos de búsqueda es la información del vecino; por ejemplo, dada una página *P*, ordena la colección de páginas señaladas por *P* o la colección de páginas que señalan a *P*. En la siguiente figura se muestra la web basada en hipervínculos.

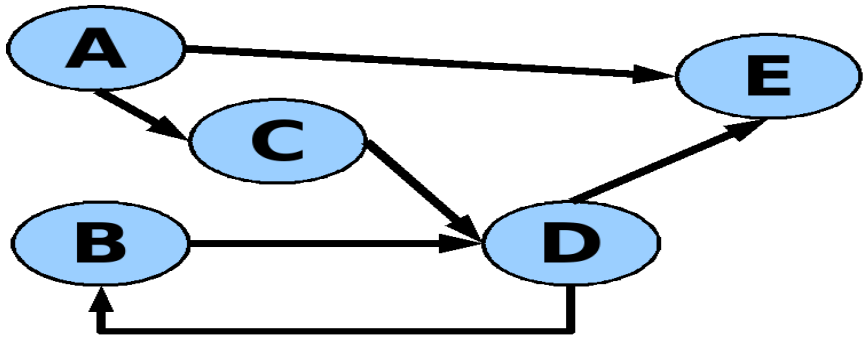

Figura 4 Web basada en hipervínculos

- **Índice de texto:** Aunque los índices basados en los hipervínculos entre páginas son utilizados mucho para mejorar la calidad y la relevancia de los resultados de búsqueda, los índices basados en largas colecciones de texto (que sirven por ejemplo para poder identificar ciertas palabras con páginas

almacenadas) continúan siendo primordiales a la hora de definir la relevancia de los resultados. Pueden ser implementados utilizando varias estructuras, y hay dos muy importantes: signature files, que son índices estructurados implementando un modelo que asigna claves a bloques de texto, e índices invertidos, de los que se hablará más adelante.

- **Índices útiles:** El número y el tipo de índices útiles construidos depende de las características del algoritmo de búsqueda y de lo que ofrezca cada motor de búsqueda en particular. Por ejemplo, un buscador que otorgue la posibilidad de restringir las búsquedas a un dominio concreto o a un sitio específico, se beneficiará de un índice que relacione el nombre de tal dominio o sitio con la lista de páginas que le pertenecen. Del mismo modo, utilizando la neighborhood information, un algoritmo iterativo (que se repite bajo unas condiciones) puede computar y almacenar el valor de la relevancia que se le otorga a cierta página (por ejemplo, el algoritmo PageRank). Un índice de este tipo será utilizado en el momento en el que se necesiten ordenar los resultados obtenidos por una cierta consulta.

1.5.1 Índice invertido

Los índices invertidos son los más utilizados hoy en día por los buscadores. Un índice invertido de una colección de páginas web consiste en un conjunto de listas invertidas, una para cada palabra (o término del índice). Una lista invertida para un término consiste en una lista de localizaciones de la colección donde el término aparece. En el más simple de los casos, una localización será el identificador de la página y la posición del término en la página. Sin embargo, los algoritmos de búsqueda se sirven usualmente de información adicional sobre la ocurrencia de términos en una página. Por ejemplo, términos que aparezcan en el cuerpo de una página, en los encabezamientos o en los hipervínculos serán tratados de forma distinta a la hora de evaluar el ranking de la página para ése término. Para conseguir que esta distribución sea más eficiente, se crean campos donde la información adicional de un término se almacena. Por ejemplo, la mayoría de los índices de texto mantienen lo que se llama lexicon, que lista todos los términos que aparecen en el índice junto con estadísticas sobre cada uno de ellos (por ejemplo, número total de documentos en los que el término aparece) que son usados por los algoritmos de catalogación de relevancia para los sitios web.
Conceptualmente, elaborar un índice invertido implica procesar cada página. Junghoo Cho y Héctor García-Molina, desde la Computer Science Department, Stanford University, admiten que tras haber

fragmentado el texto en términos indexados, se analizarán sus localizaciones, y finalmente se almacenará tal contenido en el disco duro. Pero tratar con colecciones de páginas a escalas tan grandes no hace la cosa tan fácil. Se necesitan recursos muy optimizados, pudiendo llevar días la tarea de indexamiento. Además, como el contenido de la WWW cambia rápidamente, periódicos procesos de crawleo y reconstrucción de los índices son necesarios para mantener la colección actualizada. Finalmente, los formatos de almacenaje para un índice invertido deben ser elegidos muy cuidadosamente.

Un índice muy comprimido mejora la calidad de respuesta del motor de búsqueda porque puede almacenar porciones largas de web en el caché de la memoria. Sin embargo, presenta ciertas desventajas, y la principal es la velocidad de respuesta.
Por lo tanto, construir un índice eficiente y a escala de la red de redes requiere una altísima escalabilidad y una arquitectura muy distribuida. Bajo estos esquemas, existen dos maneras básicas para dividir los índices a través de una colección de nodos:

- **Índice invertido local:** En este tipo de organización, cada nodo es responsable de una desunión de listas de páginas en una colección. De alguna manera, el algoritmo de búsqueda será en este caso el que se difunda a través de los nodos, que devolverán listas de páginas que contengan las palabras buscadas. A modo de ejemplo, consideremos la estructura E_1, formada por las páginas {P_1, P_2,…, P_n}, una posible distribución siguiendo este método sería admitiendo una o varias páginas iniciales (P_i, P_j,..., P_k), y que de cada una de estas páginas se deriva una o más de una, y así sucesivamente.

- **Índice invertido global:** Esta organización esquematiza los términos indexados de manera que cada servidor almacena listas de sólo ciertos términos de la colección. Por ejemplo, en un sistema en el que hay dos servidores S_1 y S_2, el primero podrá almacenar una lista de términos que empiecen por las letras [a,b,…,q] y el segundo las restantes, de manera que el algoritmo de búsqueda accederá a S_1 cuando esté buscando la palabra ‘proceso’.

1.5.2 Arreglo de Sufijos

Los índices invertidos asumen que el texto puede verse como una secuencia de palabras. Esto restringe el tipo de consultas que se pueden responder. Por otro lado, las consultas de frases son caras de resolver. Además, debe tenerse en cuenta que el concepto de palabra no existe en aplicaciones como las base de datos (BD) genéticas.

Los arreglos de sufijos son una implementación eficiente en espacio de los ´arboles de sufijos. Este tipo de índice permite responder eficientemente a consultas más complejas. Sus principales desventajas son el coste del proceso de construcción, que el texto debe estar disponible en el momento de la consulta y que los resultados no se recuperan en el mismo orden que su posición en el texto. Esta estructura puede utilizarse para indexar palabras (con o sin stopwords) o caracteres, lo que la hace adecuada para un amplio espectro de aplicaciones como las BD genéticas. Sin embargo, para aplicaciones basadas en palabras, los índices invertidos suelen resolver mejor las consultas complejas.

El arreglo de sufijos ve el texto como una cadena larga. Cada posición se considera como un sufijo de texto (i.e., una cadena que va desde dicha posición hasta el final). Dos sufijos que comienzan en posiciones diferentes serán lexicográficamente diferentes (se asume que existe una marca de final de texto). De este modo, cada sufijo se identifica de forma única por su posición en el texto.

No todas las posiciones del texto se tienen que indexar. Lo habitual es seleccionar puntos de índices en el texto que apuntan al comienzo de las posiciones del texto que serán recuperadas. Por ejemplo, es posible indexar solo los comienzos de palabra para tener una funcionalidad similar a los índices invertidos. Los elementos que no son puntos índice no son recuperables (al igual que en un índice invertido no es posible recuperar el medio de una palabra).

1.6 Buscadores de internet

En la actualidad se aprecia una tendencia por parte de los principales buscadores de internet a dar el salto hacia entornos móviles creando una nueva generación de buscadores web. A continuación se describen algunos buscadores que fueron objeto de estudio, resaltando sus principales características y funcionalidades.

1.6.1 Google

Google, es el motor de búsqueda en internet más grande y más usado actualmente (2007). La innovadora tecnología de búsqueda y su diseño de interfaz de usuario diferencian a Google de los actuales buscadores. Se basa en los hipertextos, analizando todo el contenido de cada web y la posición de todos los términos en cada página. Le da prioridad a los resultados de acuerdo con la proximidad de los términos de la búsqueda, favoreciendo los resultados en los que los términos de búsqueda están próximos entre sí, sin perder tiempo analizando resultados irrelevantes.

Google se basa en la tecnología PageRank, lo que asegura que los resultados más importantes se muestran primero. PageRank mide objetivamente la importancia de las páginas web y se calcula que resuelve una ecuación de 500 millones de variables y más de 2.000 millones de términos. Los complejos mecanismos automáticos de búsqueda de Google permiten prescindir de la interferencia humana. Está estructurado de manera que nadie puede comprar un lugar privilegiado en la lista ni alterar los resultados con fines comerciales (nadie puede comprar un PageRank más elevado, por ejemplo).

Su robot de búsqueda es Googlebot. Colecciona documentos desde la WWW para construir una base de datos para el motor de búsqueda Google. Tiene dos versiones, Deepbot y Freshbot. Deepbot investiga profundamente, tratando de seguir cualquier enlace en esa página, además de poner esta página en el caché, y dejarla disponible para Google. En Marzo del 2006, completaba este proceso en casi un mes. Freshbot investiga la web buscando por contenido nuevo. Visita sitios que cambian frecuentemente. Idealmente, el Freshbot visitará la página de un periódico todos los días, mientras que la de una revista cada semana, o cada 15 días. Googlebot descubre enlaces a otras páginas, y se dirige hacia ellos también, así puede abarcar toda la WWW fácilmente.

El proceso de análisis de la red de redes en Google se basa en el robot Googlebot, programado en Python e implementado en plataformas Linux. Para poder dar abasto a tal cantidad de información, Google tiene un sistema distribuido de rastreo. Un servidor general emite a cada rastreador (suelen ejecutar hasta 3 al mismo tiempo) una lista de páginas, de manera que cada araña rastreadora mantiene unas 300 conexiones abiertas. Sus fundadores especificaron que si, en cambio, se utilizaran 4 *web* crawlers, Google sería capaz de analizar 100 páginas en 1 segundo. De esta manera, en 9 días

es capaz de analizar 26 millones de sitios. Puede también leer contenido de ficheros en un formato distinto al HTML, como por ejemplo PDF.

1.6.2 Yahoo

Yahoo lanzó su propio buscador basado en una combinación de tecnologías de sus adquisiciones y proporcionando un servicio en el que ya prevalecía la búsqueda en webs sobre el directorio. En la actualidad utiliza un motor de búsqueda más intuitivo bautizado "Search Assist" (asistente de búsqueda), incluye una función que permite ver sugerencias mientras el usuario se decide entre las opciones encontradas por el buscador. También permite incluir varios tipos de búsquedas en una misma página, combinando textos, fotos, videos o audio.

Yahoo Slurp es el robot rastreador de Yahoo! para el indexado de páginas web, el mismo recopila documentos de la WWW para construir un índice rastreable para servicios de búsqueda que usan el motor de búsqueda de Yahoo. Estos documentos son descubiertos y rastreados porque otros sitios web contienen enlaces que dirigen hacia ellos.

Como parte del sistema de rastreo, Yahoo Slurp tomará en cuenta los estándares robots.txt para asegurarse de que no se rastrean e indexan las páginas que no quieres que aparezcan en resultados de búsqueda a través de Yahoo Search Technology. Si una página está protegida por un fichero robot.txt no será considerada para inclusión ni indexación en la base de datos de nuestro motor de búsquedas. Este spider, basado en la tecnología de Inktomi, se prevé que podrá hacerse con una importante fracción del mercado, alrededor del 40%.

Yahoo Slurp, rastrea la web más rápidamente que su anterior versión, de manera que, tal y como comentan sus creadores, los propietarios de los sitios notarán en un 25% la reducción de peticiones de descarga y el ancho de banda utilizado por este rastreador. Almacena toda la información que recoge durante el proceso de rastreo y luego la pasa a la base de datos. Diariamente rastrea la web para mantener sus páginas actualizadas y dos veces por semana se encamina en busca de nuevos contenidos.

1.6.3 Live Search

El más reciente de los grandes buscadores es Live Search (antes MSN Search), de Microsoft, que previamente dependía de otros para listar sus búsquedas. En 2004 debutó una versión beta con sus propios resultados, impulsada por su propio robot (llamado MSNBot). Al principio de 2005 comenzó la versión definitiva.

MSNBot tiene un funcionamiento similar a GoogleBot. Rastrea la web a través de los links establecidos de una página a otra y a partir de esta información realizará una clasificación a partir de los enlaces recibidos y la importancia de las páginas que los enlazan. La importancia de esta nueva incursión de MSN en este terreno es que la gran mayoría de los equipos informáticos del mundo utilizan Windows y alguna de las versiones de Internet Explorer, que enlazan habitualmente a una pagina de búsqueda cuando no se encuentra la URL indicada.

1.6.4 AltaVista

AltaVista, que significa "una visión desde las alturas", basa su ranking, más o menos, en los siguientes factores:

- Las páginas largas con mucho texto significativo.

- Páginas con un buen sistema de navegación, con un montón de vínculos a páginas con contenido relacionado.

- La conectividad de las páginas, incluyendo no sólo cuantos vínculos hay hacia una página sino también desde dónde vienen los vínculos; el número de distintos dominios y la "calidad" de esos sitios desde los que apuntan los vínculos. Un sitio o página es "bueno" si muchas páginas apuntan a ella y especialmente si muchos buenos "sitios" apuntan a ella.

- El nivel de directorio donde se encuentra la página. Los más altos son considerados como más importantes. Si una página está muy al fondo, el espía no irá tan abajo y nunca la encontrará. Estos factores estáticos son re calculados una vez a la semana, y según vaya mejorando la página irá subiendo en el ranking.

El índice de AltaVista se construye enviando "espías" (programas robot) que capturan texto y lo almacenan. En este proceso no interviene ninguna acción humana ni juicio. Lo que ven es lo que almacenan.

El principal espía, "Scooter", recoge miles de peticiones HTTP simultáneamente como si fueran miles de usuarios picando texto, almacenándolo y enviándolo a las máquinas indexadoras para que el texto pueda ser clasificado. "Scooter" tiene "primos", otros espías que realizan tareas específica para ayudar a mantener el índice actualizado, cómo, por ejemplo, comprobar vínculos rotos - páginas que se han movido o borrado y no serán indexadas.

¿Cómo sabe Scooter dónde tiene que ir? Sigue los vínculos que se encuentra en las páginas que visita. Cuando una página es capturada, los vínculos desde esa página se almacenan en una lista. En teoría, no es necesario describir a AltaVista su sitio: el resto del sitio se encontrará automáticamente. En un día normal, Scooter y sus primos visitan más de 10 millones de páginas.

1.6.5 Buscador Cubano 2x3

El primer buscador cubano en internet fue presentado en La Habana por la Oficina para la Informatización (INFOSOC), en el pabellón que representa a Cuba en la Exposición Internacional Informática 2007, como uno de los proyectos más avanzados en los que trabaja dicha entidad para facilitar el uso masivo, ordenado y eficiente de las Tecnologías de la Información y las Comunicaciones (TICs) a escala nacional.

La función principal de esta herramienta es facilitar la búsqueda, revisión y consulta de los más diversos contenidos sobre nuestro país, publicados en páginas y sitios web del patio, o sea, en la red de Cuba. El mismo contiene indexadas en su base de datos más de 100 mil direcciones de sitios cubanos en internet correspondientes al dominio .cu, pero próximamente se irá ampliando esta cobertura a todos los sitios de entidades cubanas o mixtas basadas en el país. El robot de búsqueda diariamente indexa aquellos sitios que poseen mayor nivel de actualización, como son los medios de prensa, el resto de los sitios se revisan con menor frecuencia proporcionando una actualización semanal a su base de datos.

El sistema cuenta con la opción de introducir manualmente nuevos sitios por los usuarios para luego ser recorridos por su robot, aunque para ser incorporados tienen que cumplir con ciertos criterios, el primero es que el sitio exista, o sea, que esté en funcionamiento. Y después, que esté en el dominio .cu o que cumpla con las condiciones de pertenecer a entidades de cubanas o mixtas basadas en el territorio nacional, con independencia de la ubicación de los servidores donde están hospedadas. Además, en todos los casos se revisará que su contenido no sea ofensivo de las normas de conductas y educación aceptadas.

Entre sus principales funcionalidades ofrece la búsqueda por palabras o por frases, el sistema no busca por todas las palabras introducidas ya que el mismo posee un listado de palabras tales como preposiciones, conjunciones, artículos y otras que no aportan significado por sí mismas y que son conocidas como “palabras vacías” o “stopwords”. También permite realizar búsquedas especiales en sitios de medios de prensa así como en los discursos del compañero Fidel. Además ofrece la búsqueda de imágenes, en la misma se puede seleccionar diferentes tamaños para las imágenes. El criterio de ordenamiento para devolver los resultados de las búsquedas aparecen primero las páginas que contienen las palabras tecleadas en la dirección URL, a continuación cuando aparecen en el título de la página, luego las que aparecen en los metadatos y por último, las que aparecen en el cuerpo o contenido principal de la página.

1.6.6 Buscador UCI GPI++

El Grupo de Procesamiento de Imágenes (GPI), es uno de los proyectos que se desarrolla en la Universidad de Ciencias Informáticas (UCI) con el objetivo proveer con productos de software de alta calidad y de elevado valor agregado; por su carácter científico, en el tema de Procesamiento Digital de Imágenes y Señales, al Sistema Nacional de Salud y a otros Centros e Instituciones.

En la actualidad GPI constituye uno de los proyectos líderes de la UCI su más conocida identificación es el portal GPI basado en el CMS Joomla que entre sus objetivos promueve una continua publicación de noticias sobre tecnologías desarrolladas en el mundo y un servicio de búsqueda de información en la intranet de la UCI el cual se basa en un módulo de este CMS. El funcionamiento del mismo realiza procesos irregulares de búsqueda debido a la información desactualizadas en sus bases de datos y la estructura de sus índices no permite la eliminación de redundancia para los criterios de búsqueda establecidos por los usuarios.

1.7 Aplicaciones web

El surgimiento de internet y de la WWW cambiaron la forma de pensar de muchos programadores a nivel mundial, ya que anteriormente las aplicaciones eran desarrolladas solamente para entornos de escritorio, con la aparición de la nueva infraestructura de comunicación se diseñaron los lenguajes de programación web con el propósito de contribuir a la creación de aplicaciones más eficientes y funcionales que permitan utilizar los recursos de internet.

Una aplicación web es un sistema informático que los usuarios utilizan accediendo a un servidor web a través de internet o de una intranet con un conjunto de páginas estáticas y dinámicas.
Una página web estática es aquélla que no cambia cuando un usuario la solicita: el servidor envía la página al navegador solicitante sin modificarla. Por el contrario, el servidor modifica las páginas dinámicas antes de enviarlas al navegador solicitante. La naturaleza cambiante de este tipo de página es la que le da el nombre de dinámica.

1.7.1 Arquitectura de las Aplicaciones web

La arquitectura general de las aplicaciones web está basada en la arquitectura Cliente/Servidor con notables distinciones. Una de sus ventajas más significativas es su forma de instalación y distribución.
La esta arquitectura es la integración distribuida de un sistema en red, con los recursos, medios y aplicaciones que, definidos modularmente en los servidores, administran, ejecutan y atienden las solicitudes de los clientes; todos interrelacionados física y lógicamente, compartiendo datos, procesos e información. Se establece así un enlace de comunicación transparente entre los elementos que conforman la estructura. Esta arquitectura tiene entre sus características que el servidor presenta a todos sus clientes una interfaz única y bien definida y éste no necesita conocer la lógica del servidor, sólo su interfaz externa. El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra. Los cambios en el servidor implican pocos o ningún cambio en el cliente.

Instalar una aplicación web consiste en configurar los componentes del lado del servidor en la red y no es necesaria una instalación o configuración en el lado cliente. La comunicación del cliente y el servidor es vía HTTP, protocolo principal de comunicación en una aplicación web, el cual funciona normalmente desconectado, es decir, el cliente hace una petición al servidor, este la procesa y le devuelve el resultado, terminando la comunicación entre estos.

1.7.2 Principales ventajas de las Aplicaciones web

- Compatibilidad multiplataforma, ya que varias tecnologías incluyendo Java, Flash, ASP, Ajax, PHP entre otras permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales.

- Menos requerimientos de memoria, porque al residir y correr en los servidores del proveedor, a esas aplicaciones basadas en web usa en muchos casos la memoria de las computadoras que ellos corren.

- Inmediatez de acceso. Las aplicaciones web no necesitan ser descargadas, instaladas y configuradas.

- Múltiples usuarios concurrentes. Las aplicaciones web pueden realmente ser utilizada por múltiples usuarios al mismo tiempo.

- Alta disponibilidad. Se puede acceder a ellas a cualquier hora y desde cualquier parte del mundo si se tiene conexión a internet.

1.7.3 Principales desventajas de las Aplicaciones web

- Diferencias de presentación entre plataformas y navegadores. La falta de estándares ampliamente soportados dificulta el desarrollo de las aplicaciones.

- Acceso limitado, la necesidad de conexión permanente y rápida a internet hacen que el acceso a estas aplicaciones no esté al alcance de todos

- La interactividad no se produce en tiempo real, en las aplicaciones web cada acción del usuario conlleva un tiempo de espera excesivo hasta que se obtiene la reacción del sistema.

- Elementos de interacción muy limitados. En comparación con el software de escritorio, las posibilidades de interacción con el usuario que ofrecen las aplicaciones web (mediante formularios principalmente) son muy escasas.

1.8 Metodología de desarrollo de software utilizada

En la actualidad el uso de metodologías es un factor importante en el desarrollo de sistemas informáticos. El principal propósito es contar con un marco de trabajo claramente definido y estandarizado, que permita obtener productos que garanticen los requerimientos de calidad, que cumplan con las expectativas del cliente y se desarrollen en el tiempo estimado y bajo los costos presupuestados. Para la realización del sistema se decidió el uso del Proceso Unificado de Desarrollo de Software, RUP (del inglés: Rational Unified Process).

1.8.1 Proceso Unificado de Desarrollo de Software

RUP constituye una forma disciplinada de asignar tareas y responsabilidades en una empresa de desarrollo (quién hace qué, cuándo y cómo), y tiene como objetivo asegurar la producción de software de calidad dentro de plazos y presupuestos predecibles. Dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. El proceso de ciclo de vida de RUP se divide en cuatro fases. Esas fases se dividen en iteraciones, cada una de las cuales produce una pieza de software demostrable (3).

UML (Lenguaje Unificado de Modelado)

UML es un lenguaje que proporciona un vocabulario y reglas para permitir una comunicación, el mismo nos indica cómo crear y leer los modelos, pero no dice cómo crearlos. UML sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas software como para la arquitectura hardware donde se ejecuten, es además un método formal de modelado aporta un mayor rigor en la especificación y permite realizar una verificación y validación del modelo realizado. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

1.9 Plataforma de desarrollo de software utilizada

Una plataforma web tiene cuatro componentes básicos: un sistema operativo, un servidor web, una base de datos y un lenguaje de programación. Buena parte de las empresas comerciales presentan los cuatro componentes bien sea empaquetados, o de fácil incorporación e integración entre ellos, ventaja que no se tiene en el mundo abierto ya que cada uno produce por su lado. Existen muchas y nuevas plataformas para el desarrollo de aplicaciones web, como son los casos de Zope y TurboGears, ambas son plataformas de desarrollo web implementadas en Python. Sin embargo las plataformas más utilizadas son .NET, J2EE y XAMP.

1.9.1 XAMP

XAMP traduce un conjunto de aplicaciones que permite establecer una plataforma web bien sea para desarrollo o para producción, y todas las aplicaciones de libre disposición en la red.

- La "X" cubre los sistemas operativos Linux, Windows, Mac Os X o Solaris. En la actualidad el Linux se ha posicionado fuertemente en el sector de los servidores y está siendo utilizado cada día más para soluciones de misión crítica.

- La "A" denota Apache como servidor web. El mismo simplemente se encarga de servir páginas estilo web hacia quien lo solicita, que por lo general es un PC o dispositivo con un visor de web. El Apache es el servidor de mayor número de instalaciones en la red. Inició como el reemplazo al servidor de la NSF (National Science Foundation) y su curva de crecimiento es exactamente opuesta a la de decrecimiento del mismo. Este servidor está inclusive incorporado en muchas de las soluciones que brindan las diferentes casas comerciales como parte de su oferta y como manejador de servicios web, dándole un respaldo total al servidor.

- Hoy en día, con el desarrollo de las tecnologías, no se percibe que un sitio que pretenda efectuar transacciones comerciales, brindar personalización y creación de comunidad, funcione sin una base de datos como esquema de almacenamiento. La plataforma XAMP incluye la "M" de MySQL, una base de datos de muy fácil manejo y de relativamente buen desempeño. Es importante también aquí recordar que también se cuenta con otra base de datos de excelentes características de estabilidad llamada PostgreSQL la cual también es de carácter gratuito, y

aunque un poco más compleja de manejar que MySQL, también tiene desarrolladas recientemente un número considerable de herramientas que facilitan esta tarea.

- El último componente está para los lenguajes de programación. Las ofertas que propone XAMP, todas iniciando por la P son PHP, Perl, y Python. De estos tres, el que hasta hace poco tenía la delantera y todavía presenta buena acogida entre programadores es el Perl. Normalmente permite la elaboración de programas de una manera sencilla, y permite la interacción y la manipulación de información. Recientemente, con el avance que ha tenido el PHP en su versión 4 y la reciente liberación de la versión 5(actualmente se trabaja en la versión 6) que incorpora un nuevo modelo de POO (Programación Orientada a Objetos) y extensiones para el trabajo con MySQL entre otras funciones; es muy utilizado para soluciones web en la plataforma abierta. Tiene también la posibilidad de ser instalado en múltiples plataformas, lo que lo convierte en una buena elección para empresas que quieran desarrollar a bajo costo. El PHP se caracteriza en términos generales por proveer funciones de conexión con las bases de datos y funciones de presentación para que el resultado sea presentado al usuario en formato de página web. La cantidad de aplicaciones ya disponibles en PHP es amplia.

Las soluciones que ofrecen las casas comerciales se encargan de hacer que los cuatro componentes que hemos analizado: sistema operativo, servidor web, base de datos y lenguaje de programación; funcionen sincronizada y adecuadamente dentro de un conjunto de equipos. La propuesta XAMP pretende ser la respuesta a las casas comerciales con software que hasta ahora ha dado buenos resultados en forma separada y que promete siempre y cuando se logre consolidar el soporte para cada una de las herramientas (base de datos y lenguajes en especial) y garantizar la facilidad de instalación de cada uno de los componentes para permitir una integración transparente.

XAMP provee un robusto y flexible ambiente para el desarrollo de aplicaciones web, es un servidor independiente de plataforma, software libre, fácil de usar y capaz de interpretar páginas dinámicas Traduce un conjunto de aplicaciones que permite establecer una plataforma web bien sea para desarrollo o para producción, y todas las aplicaciones de libre disposición en la red.

XAMP es regularmente actualizado para incorporar las últimas versiones de Apache/MySQL/PHP y Perl. También incluye otros módulos como OpenSSL y phpMyAdmin , viene con un administrador de

servicios integrado que se instala como icono en la bandeja del sistema, que permite administrarlo fácilmente y acceder a todos los servicios. El programa esta liberado bajo la licencia GNU.

Las aplicaciones web debido a las exigencias de sus propósitos funcionales imponen la necesidad de usar para su desarrollo lenguajes de programación con características que faciliten su optimo desempeño. La incorporación de bibliotecas o paquetes de clases, la disponibilidad de documentación son factores fundamentales en la selección de una plataforma de desarrollo determinada.
Para el desarrollo de la aplicación se decidió el uso de la plataforma XAMP y el lenguaje de programación PHP.

1.10 Lenguajes de programación utilizados

Hoy en día la red de redes se ha convertido en el canal de comunicación más usado del mundo, por las grandes ventajas y potencialidades que brindan todos los sistemas que soporta, permitiendo la interacción con los usuarios y la personalización. Esto es posible por un conjunto de lenguajes de programación que le dan gran interactividad a las aplicaciones web, tanto del lado del cliente como del lado del servidor. Los lenguajes de programación más usados en software libre, que corren en el servidor y procesan toda la lógica del negocio son PHP, JAVA y Perl. Las técnicas de desarrollo web y lenguajes del lado del cliente más utilizados, encargados de visualizar la información en el navegador y la validación de la información en los formularios, son HTML, Java Script y Ajax.

1.10.1 HTML

HTML es el acrónimo de Hypertext Markup Language (lenguaje de marcas hipertextuales) que fue creado en el año 1990 por Tim Berners-Lee y diseñado principalmente para mostrar información, animaciones en forma de hipertexto. Algunas ventajas que presenta, son la facilidad con que se pueden actualizar los contenidos y que permite utilizar estilos en formato CCS (hojas de estilos en cascada) en las páginas para una mayor facilidad en su modificación. En la actualidad, es el lenguaje que utilizan todos los navegadores para mostrar la información final.

1.10.2 PHP

El lenguaje de programación PHP denominado preprocesador de hipertexto, (del ingles “Hypertext Pre-processor”), es un lenguaje libre y multiplataforma. Posee una amplia documentación en su página oficial posibilitando gran comprensión del mismo, se sustenta en la actualidad bajo el paradigma más difundido actualmente en el mundo que es programación orientado a objeto, incluye también la programación estructurada y servicios web.

Presenta excelente integración con todos los motores de base de datos. Cuenta con una biblioteca que trae un conjunto de funciones para realizar cualquier operación (acceso a base de datos, encriptación, envío de correo, XML, creación de PDF, entre otros). Su código es libre y se sustenta bajo la licencia GPL.

PHP, es un lenguaje de programación usado normalmente para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios web, es la versión libre del sistema equivalente de Microsoft ASP. Es un lenguaje encapsulado dentro de los documentos HTML, de forma que se pueden introducir instrucciones PHP dentro de las páginas. gracias a esto el diseñador gráfico de la pagina puede trabajar de forma independiente al programador.

Debido a la naturaleza open-source de PHP, si hay algo que actualmente no se pueda hacer en este lenguaje de programación no existe ningún impedimento para que se pueda escribir un módulo o una extensión en código C para extender la funcionalidad y hacer lo que se desee. Esto es posible por la buena documentación de la API que esta disponible para todos.

Principales ventajas de PHP

- Sintaxis cómoda: PHP cuenta con una sintaxis similar a la de C, C++ o Perl. Lo más destacado ocurre a nivel semántico: el tipado es muy poco estricto. Es decir, cuando creamos una variable no tenemos que indicar de qué tipo es, pudiendo guardar en ella datos de cualquier tipo. Esto es muy flexible y cómodo para el desarrollador, aunque los errores que se cometen pueden ser muchos más graves y difíciles de corregir al reducirse mucho las posibilidades del intérprete para detectar incompatibilidades entre variables.

- Soporta objetos y herencia: Tiene soporte para la programación orientada a objetos, es decir, es posible crear clases para la construcción de objetos, con sus constructores, etc. Además soporta herencia, aunque no múltiple.

- Ejecución en Servidor: Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web justo antes de que se envíe la página a través de internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página web con el código HTML resultante de la ejecución del código PHP compatible con todos los navegadores.

- Se puede incrustar código PHP con etiquetas HTML.

- Compatibilidad con bases de datos: Amplio soporte para una gran cantidad de bases se datos. Tiene acceso un gran número de gestores de bases de datos: Adabas D, dBase, Empress, Ingress, InterBase, FrontBase, DB2, Informix, mSQL, MySQL, ODBC, Oracle, PostgreSQL, Sybase, etc.

- Se puede hacer de todo lo que se pueda transmitir por vía HTTP.

- Multiplataforma: Funciona tanto en sistemas Unix o Linux con servidor web Apache como en sistemas Windows con Microsoft Internet Information Server, de forma que el código generado por cualquiera de éstas plataformas no debe ser modificado al pasar a la otra.

- Licencia de software libre: Es un lenguaje basado en herramientas con licencia de software libre, es decir, no hay que pagar licencias, ni existen límites en su distribución y, es posible ampliarlo con nuevas funcionalidades si así lo deseamos.

- Extensa librería de funciones: Cuenta con una extensa librería de funciones que facilitan enormemente el trabajo de los desarrolladores.

Principales desventajas de PHP

- Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.

- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.

- La orientación a objetos es aún muy deficiente para aplicaciones grandes

La aparición de soluciones más adecuadas y sencillas hacen que PHP se convierta en la mejor opción actual para multitud de necesidades., actualmente es uno de los paquetes para programación de internet más utilizados.

1.11 Servidor web utilizado

Un servidor web es un programa que implementa el protocolo HTTP. Este protocolo está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML: textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música. Este servidor se encarga de mantenerse en espera de las peticiones HTTP llevadas a cabo por un cliente. Este realiza una petición al servidor, quien le responde con el contenido solicitado.

Para alojar la aplicación se usará el servidor Apache, es un servidor de páginas web de código abierto multiplataforma y modular, el mismo se ha adaptado siempre a una gran variedad de entornos a través de su diseño modular. Este diseño permite a los administradores de sitios web elegir que características van a ser incluidas en el servidor seleccionando que módulos se van a cargar, ya sea al compilar o al ejecutar el servidor. Trabaja sobre múltiples plataformas (Unix, Linux, MacOSX, Vms, Win32, OS2) entre otras, soporta CGI, Perl, PHP, permite soporte SSL para transacciones seguras. Desde su origen a evolucionado hasta convertirse en uno de los mejores servidores en términos de eficiencia, funcionalidad y velocidad. Apache ha demostrado ser substancialmente mas rápido que muchos otros servidores libres y compite con los mejores servidores comerciales.

1.12 Sistemas Gestores de Base de Datos utilizados

1.12.1 MySQL

Es uno de los gestores de bases de datos más populares de internet para aplicaciones pequeñas, que no realicen muchas transacciones al mismo tiempo, con PHP hace la combinación perfecta. Está desarrollado bajo la filosofía de código abierto y es multiplataforma. Su velocidad, estabilidad y seguridad es alta. Presenta como gran desventaja que no implementa integridad de los datos, ni soporte de vistas, ni sub-consultas, esto lo hace un poco deficiente a la hora de desarrollar un sistema grande, con muchos clientes y muchas transacciones a la vez.

1.12.2 PostgreSQL

Es el motor de base de datos libre más avanzado hasta estos momentos, al que se le puede acceder a su código fuente. Es usado para manejar grandes cantidades de información y está basado en el modelo relacional, aunque incorpora conceptos del modelado orientado a objeto. Se pueden definir consultas anidadas, vistas, crear funciones por el usuario, no sólo en el lenguaje natural SQL, sino en varios más, entre ellos C, Pl-PgSQL, lenguaje nativo PostgreSQL, Perl, PHP y Java. Es multiplataforma, soporta múltiples transacciones, integridad de datos, presenta una estabilidad muy alta, gran seguridad de los datos, soporta la réplica y procedimientos almacenados. Propone un tamaño ilimitado para las base de datos y de 64 Tb para las tablas, lo que da la medida de un gestor de base de datos robusto, y con grandes funcionalidades. Presenta como desventaja que por sus grandes potencialidades consume muchos recursos y carga el sistema.

1.13 Herramientas de desarrollo utilizadas

Las herramientas de desarrollo en la actualidad con el elevado número de prestaciones que ofrecen, posibilitan la obtención del producto con mayor calidad. Proporcionan un alto grado de personalización, lo que da lugar a un conjunto mayor de soluciones escalables. Las avanzadas funcionalidades de depuración de códigos permiten desarrollar sistemas informáticos más acabos y en menores tiempos de desarrollo.

1.13.1 Herramienta de modelado UML

Para el modelado de las artefactos y diagramas generados a lo largo del ciclo de vida del proyecto se decidió el uso de Visual Paradigm, pues su uso está muy estandarizado a nivel mundial y constituye una herramienta multiplataforma muy madura y acabada, además, el grupo de desarrollo se encuentra familiarizado con el uso de esta herramienta.

Visual Paradigm

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite modelar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

1.13.2 Entorno Integrado de desarrollo

Los entornos integrados de desarrollo, IDE (del inglés: Integrated Development Environment) constituyen programas compuestos por un conjunto de herramientas para los programadores. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios. Los IDEs proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación. Para el desarrollo de la aplicación se decidió usar Zend Studio, debido a las múltiples funcionalidades que ofrece.

Zend Studio

Es un programa de la casa Zend, impulsores de la tecnología de servidor PHP, orientada a desarrollar aplicaciones web, en lenguaje PHP. Además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código. Tiene versiones para diferentes sistemas operativos Windows, Linux y MacOS. Consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Las dos partes se instalan por separado, la del cliente contiene la interfaz de edición y la ayuda. Permite además hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración hay disponer de la parte del servidor, que instala Apache y el módulo PHP

o, en caso de que estén instalados, los configura para trabajar juntos en depuración. Contiene una ayuda contextual con todas las librerías de funciones del lenguaje que asiste en todo momento ofreciendo nombres de las funciones y parámetros que deben recibir. Aunque esta ayuda contextual no solo se queda en las funciones definidas en el lenguaje, sino que también reporta ayudas con las funciones que vaya creando el programador.

1.13.3 Herramienta para el control de versiones

El control de versiones es el proceso mediante el cual se pueden gestionar todas las versiones de los elementos de configuración que forman la línea base de un producto o configuración del mismo. Los sistemas de control de versiones facilitan la administración de los cambios que se realizan sobre los componentes. El mismo es utilizado principalmente en la industria de la informática para el control de las versiones por las que transita el código fuente de una aplicación, aunque el concepto es aplicable a otras partes del software y del mundo informático, como documentos, imágenes, entre otras (4).
Este proceso puede ser desarrollado de forma manual, aunque en la actualidad existen varias herramientas que facilitan el proceso.

Un sistema de control de versiones proporciona:

- Mecanismo de almacenamiento de cada objeto bajo control de versiones.

- Posibilidad de edición para cada objeto bajo control.

- Almacenamiento de la línea de cambios por los que ha pasado el objeto, de forma tal que posibilite ir a una versión anterior en condiciones determinadas.

Es aconsejable la generación de informes que almacenen los cambios producidos entre versiones, informes de estado, especificando un nombre que identifique a la persona que realizó los cambios.
Para el desarrolló de aplicación se decidió el uso de SVN, herramienta capaz de automatizar todos las funcionalidades relacionadas con el control de versiones, es de libre acceso y el grupo de desarrollo cuenta con experiencia en su uso.

SVN

Software diseñado para mantener el control de versiones sobre los sistemas. Una característica importante de Subversion es que todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo (5).

Algunas de sus principales ventajas son:

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.
- Las modificaciones son automáticas.
- Permite la creación de ramas y etiquetas.
- Se envían sólo las diferencias en ambas direcciones.
- Puede ser servido mediante Apache. Esto permite que sea accedido desde la web.
- Maneja eficientemente archivos binarios.
- Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En este capítulo se presentan los temas referentes al objeto de estudio de la aplicación, haciendo énfasis en los procesos involucrados en el campo de acción, así como un análisis de la ejecución del mismo. Se describen las características del sistema que fueron obtenidos durante la modelación del dominio y que recogen los temas referentes a las necesidades que promovieron la realización del presente proyecto, así como las acciones que se deben de llevar a cabo para su solución y que tributan a que se realice una propuesta del sistema.

2.2 Propuesta del Sistema

La implementación del sistema se basa en los siguientes componentes:

- **Servidor web:** Es la aplicación orientada a gestionar las peticiones de las aplicaciones web y de generar la respuesta a cada acción enviada. Es capaz de manejar la concurrencia, seguridad y persistencia de los datos en el servidor. Cuenta con servicios que se ejecutan en un tiempo definido por el desarrollador de las aplicaciones para controlar las pérdidas de conexión con el servidor. Periódicamente realiza comprobaciones para eliminar recursos que no están en uso.

- **Sistema Sisweb:** Sistema Indexador de Información web que posibilita a través de una interfaz un conjunto de funcionalidades, como, Indexar URL, Listar URL, Gestionar URL y Gestionar estadísticas del sistema. Permite datos referentes al estado del servidor y los principales procesos del sistema.

Funcionamiento del Sistema

El Sistema de Indexación de Información es una aplicación web de interfaz de usuario que se ejecuta mediante el protocolo HTTP. Se instala en un servidor Apache de uso general y gestiona los datos en un servidor MySQL o PgSQL En la siguiente figura se muestra la propuesta del sistema.

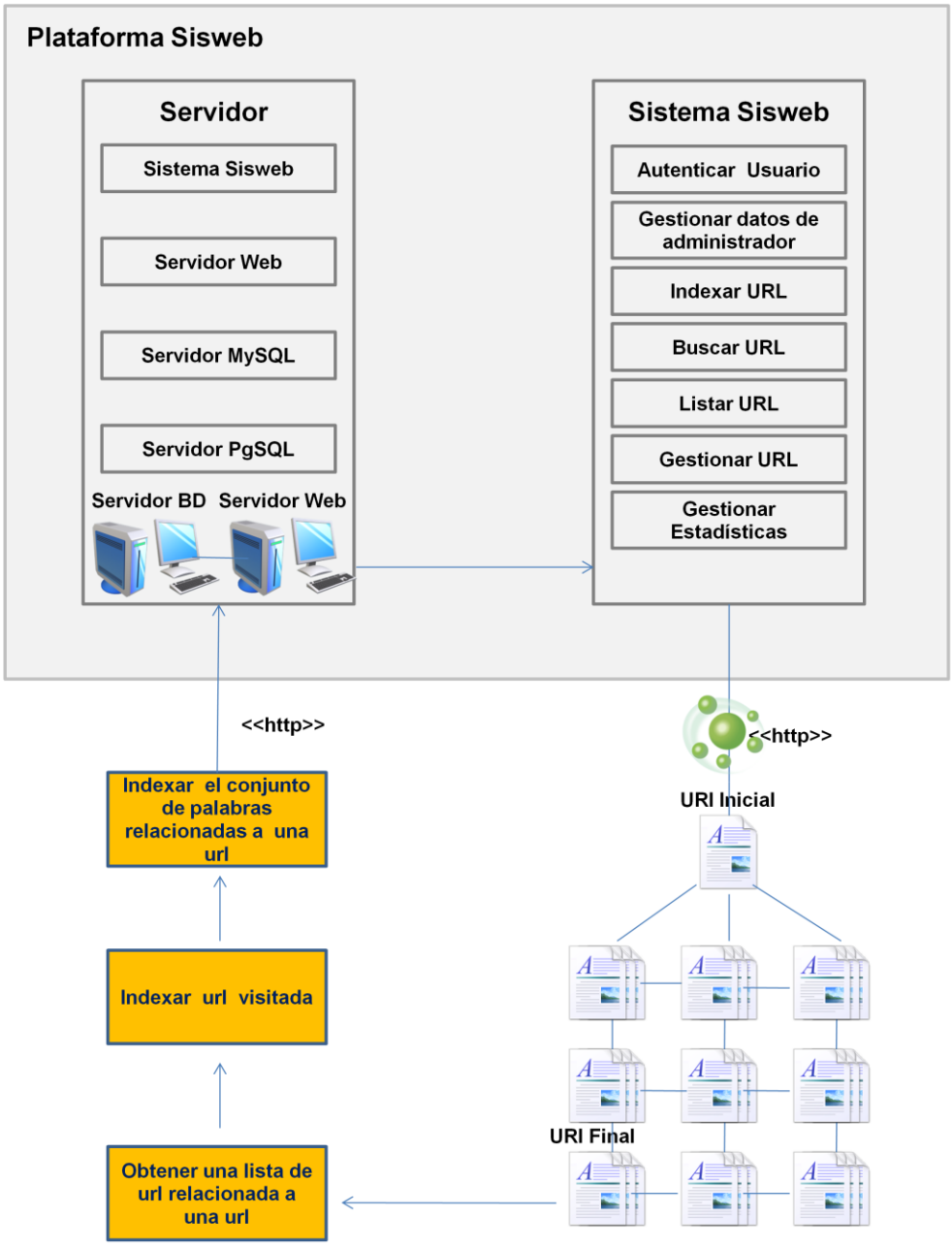

Figura 5 Propuesta del sistema

2.3 Modelo de negocio

Un sistema, generalmente es complicado. Por eso es necesario dividirlo en piezas si se pretende comprenderlo y gestionar su complejidad. Esas piezas se pueden representar a través de modelos que permitan abstraer sus características esenciales. Entre estos se encuentra el modelo de negocio en el cual se refuerza la idea de que sea el propio negocio lo que determine los requisitos.

Dependiendo de la situación o escenario que se presente, hay varias alternativas de desarrollar este proceso, no siempre es necesaria o posible, la completa realización de un modelo de negocio. Si se determina que no es necesario un modelo completo del negocio se realizará lo que se conoce como un modelo del dominio.

2.4 Modelo de dominio

El modelo del dominio captura los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema. El modelo del dominio se considera un subconjunto del llamado modelo de objetos del negocio.

Durante el desarrollo de la aplicación se detectaron problemas en el origen de los flujos de informaciones, resultando en algunos casos difuso. Se hizo difícil determinar los elementos más importantes del sistema y sus interconexiones, así como el establecimiento de las reglas de funcionamiento por lo que se hizo necesario un modelado del domino, mostrado en la siguiente figura.

Figura 6.2.4 Modelo de dominio

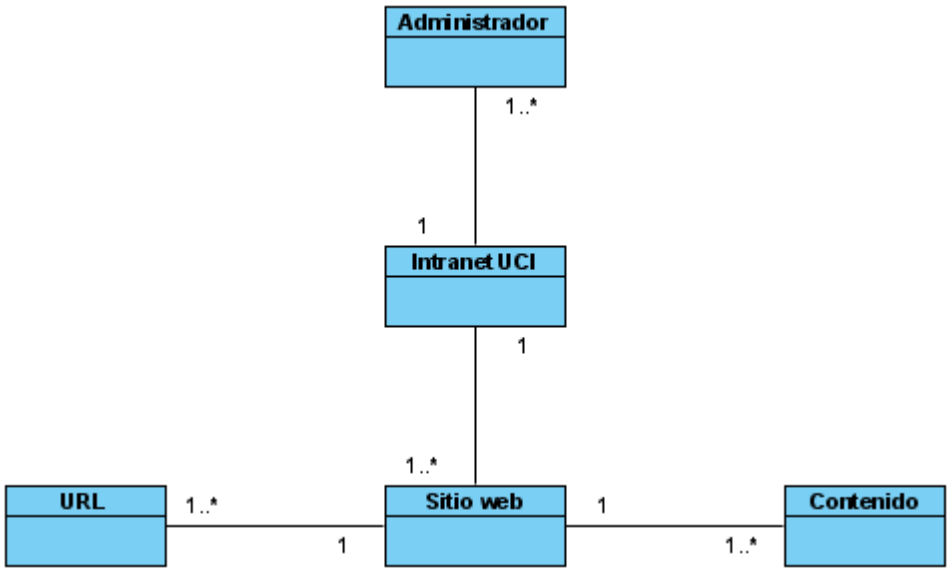

Figura 6 Modelo de dominio

Glosario de términos del dominio

Administrador: Usuario con rol de administrador del sistema.

Intranet UCI: Infraestructura de comunicación de computadores de la UCI.

Sitio web: Conjunto de documentos web relacionados.

URL: Localizador de recurso uniforme.

Contenido: Información.

2.5 Requisitos del sistema

Los requerimientos constituyen la capacidad o condición que necesita un usuario para resolver un problema o lograr un objetivo y que debe ser alcanzada o poseída por un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente (6).

En el desarrollo de la aplicación se lleva a cabo un profundo análisis de los principales conceptos y temas de importancia relacionados con su funcionamiento, el proceso de captura de requisitos arrojó un listado de las principales funcionalidades que debe cumplir el sistema, así como todos los aspectos relacionados con su funcionamiento. La identificación de los requisitos tuvieron en cuenta las necesidades que promovieron la realización del presente proyecto.

A continuación se describen los requerimientos funcionales y no funcionales identificados, empleándose un nombre y una breve explicación en cada caso.

2.5.1 Requisitos Funcionales

RF.1- Gestionar usuario: Define funciones de control para la cuenta de usuario administrador del sistema.

RF.1.1- Registrar usuario: Define la posibilidad de crear una cuenta de usuario administrador del sistema.

RF.1.2- Autenticar usuario: Define el mecanismo para controlar el acceso al sistema.

RF.1.3- Cambiar datos de usuario: Define la posibilidad de actualizar los datos del usuario administrador del sistema.

RF.2- Gestionar Base de Datos: Define funciones de control de la base de datos del sistema

RF.2.1- Crear base de datos del sistema.

RF.2.2- Eliminar datos de cache: Define la posibilidad de eliminar datos del cache de la base de datos.

RF.2.3- Limpiar índice: Define la posibilidad de limpiar el índice de la base de datos.

RF.2.4- Limpiar palabras comunes: Define la posibilidad de limpiar las palabras comunes de la base de datos.

RF.2.5- Limpiar arranques: Define la posibilidad de limpiar los datos de arranque de la base de datos.

RF.3- Indexar URL: Define el mecanismo para indexar URL.

RF.3.1- Listar URL indexadas: Define el mecanismo de listar las URL indexadas.

RF.3.2- Mostrar tiempo de ejecución del proceso de indexación.

RF.4- Gestionar URL: Define las funcionalidades relacionadas con la gestión de URL.

RF.4.1- Actualizar URL indexadas: Define la posibilidad de actualizar las URL indexadas.

RF.4.2- Eliminar URL indexadas: Define la posibilidad de elimina las URL indexadas.

RF.5- Mostrar estadísticas: Define la posibilidad de mostrar estadísticas.

RF.5.1- Mostrar palabras claves: Define la posibilidad de mostrar palabras claves almacenadas en la base de datos.

RF.5.2- Mostrar URL por cantidad de información: Define la posibilidad de mostrar las URL indexadas por la cantidad de información.

RF.6- Mostrar estado de la Base de Datos: Define el mecanismo para mostrar el estado de la base de datos.

RF.6.1- Mostrar número de entradas de hosts: Define el mecanismo para mostrar números de entradas de URL.

RF.6.2- Mostrar número de entradas de URL: Define el mecanismo para mostrar número de entradas de índice.

RF.6.3- Mostrar número de entradas de índice: Define el mecanismo para mostrar el número de entradas de índice.

RF.6.4- Mostrar número de entradas de palabras claves: Define el mecanismo para mostrar el número de palabras claves.

RF.6.5- Mostrar número de entradas de tablas temporales: Define el mecanismo para mostrar el número de entradas de tablas temporales.

2.5.2 Requisitos No Funcionales

- **Usabilidad:** La aplicación deberá presentar un diseño centralizado en una interfaz principal que posibilite la ejecución de las funcionalidades del sistema, la distribución espacial debe estar orientada a la zona de visión del usuario para facilitar la operatividad.

- **Rendimiento:** El sistema se desarrollará sobre la arquitectura Cliente / Servidor por lo que es de importancia la tasa de transferencia entre ambos extremos. Por las características del sistema es necesario definir el tiempo límite de demora en la transferencia de datos entre el cliente y el servidor, esto está sujeto a las peculiaridades de cada equipo, así como la distancia entre ambos y la tecnología utilizada en cada caso.

- **Soporte:** La aplicación incorporará la documentación relacionada que describe las funcionalidades del sistema y una guía para su uso.

- **Portabilidad:** El sistema debe ser desarrollado sobre una plataforma multi-sistema que posibilita la ejecución en varios sistemas operativos.

- **Seguridad:**
- Confidencialidad: La información manejada por el sistema debe estar protegida de acceso no autorizado y divulgación.
- Integridad: La información manejada por el sistema debe ser objeto de cuidadosa protección contra la corrupción y estados inconsistentes, de la misma forma será considerada igual a la fuente o autoridad de los datos. Pueden incluir también mecanismos de chequeo de integridad y realización de auditorías.

- **Disponibilidad:** A los usuarios autorizados se les garantizará el acceso a la información y que los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.
- **Software:** La aplicación requiere para su ejecución MySQL 4.x o PgSQL 8.2 o superior y PHP 5.x o superior.

- **Hardware:** La aplicación debe ser compatible con bajas capacidades de hardware, a continuación se muestran datos de estos requerimientos.

- **Capacidades de hardware mínimas:**

  RAM = 128 MB
  CPU = 1.0 GHz

- **Capacidades de hardware recomendadas:**

  RAM = 512 MB
  CPU = 2.0 GHz

2.6 Modelo del sistema. Definición de casos de uso

Los casos de uso son una técnica para especificar el comportamiento de un sistema, es una secuencia de interacciones entre un sistema y alguien o algo que usa alguno de sus servicios. El modelo de casos de uso describe lo que hace el sistema para cada tipo de usuario.

2.6.1 Definición de actores

Los actores del sistema constituyen personas u otros sistemas que eran trabajadores del negocio y que interactúan de alguna forma con el sistema y que están asociadas al cumplimiento de los requisitos funcionales o procesos que responden a las funcionalidades definidas en los mismos.

Durante el desarrollo de la aplicación se definió un grupo de actores que en un momento determinado desencadenarán un grupo de acciones en el sistema, los mismos se describen a continuación:

Tabla 1 Definición de actores

Actores	**Justificación**
Usuario del Sistema	Actor con permisos de ejecución del casos de uso de autenticación de identidad
Administrador del Sistema	Actor con máximo nivel de permiso en el sistema

2.6.2 Casos de uso del sistema

Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema.

Tabla 2 Casos de Uso del Sistema

CU-1	**Autenticar usuario**
Actor	Usuario
Descripción	Caso de uso que define funciones de control de acceso al sistema
Referencia	RF.1
CU-2	**Gestionar usuario**
Actor	Administrador del Sistema
Descripción	Caso de uso que define funciones de control de datos para la cuenta de usuario administrador del sistema
Referencia	RF.1
CU-3	**Gestionar Base de Datos**
Actor	Administrador del Sistema
Descripción	Caso de uso que define funciones de control de la base de datos del sistema
Referencia	RF.2
CU-4	**Indexar URL**
Actor	Administrador del Sistema
Descripción	Caso de uso principal del sistema, define funciones de indexación de URL en el sistema
Referencia	RF.3
CU-5	**Gestionar URL**
Actor	Administrador del Sistema
Descripción	Caso de uso que define funciones de Actualizar y eliminar URL del sistema
Referencia	RF.4
CU-6	**Mostrar estadística**
Actor	Administrador del Sistema
Descripción	Caso de uso que define funciones para mostrar estadísticas del sistema
Referencia	RF.5, RF.6

2.6.3 Diagrama de Casos de Uso del Sistema

El Diagrama de casos de uso del sistema a automatizar, define las relaciones entre los actores y los casos de uso. Para la realización de la aplicación se definieron un conjunto de acciones que deben ser ejecutadas por distintos actores y que desencadenan un conjunto de operaciones. Las interrelaciones entre las acciones y los actores de la aplicación son agrupadas en el diagrama de casos de uso del sistema que se muestra a continuación.

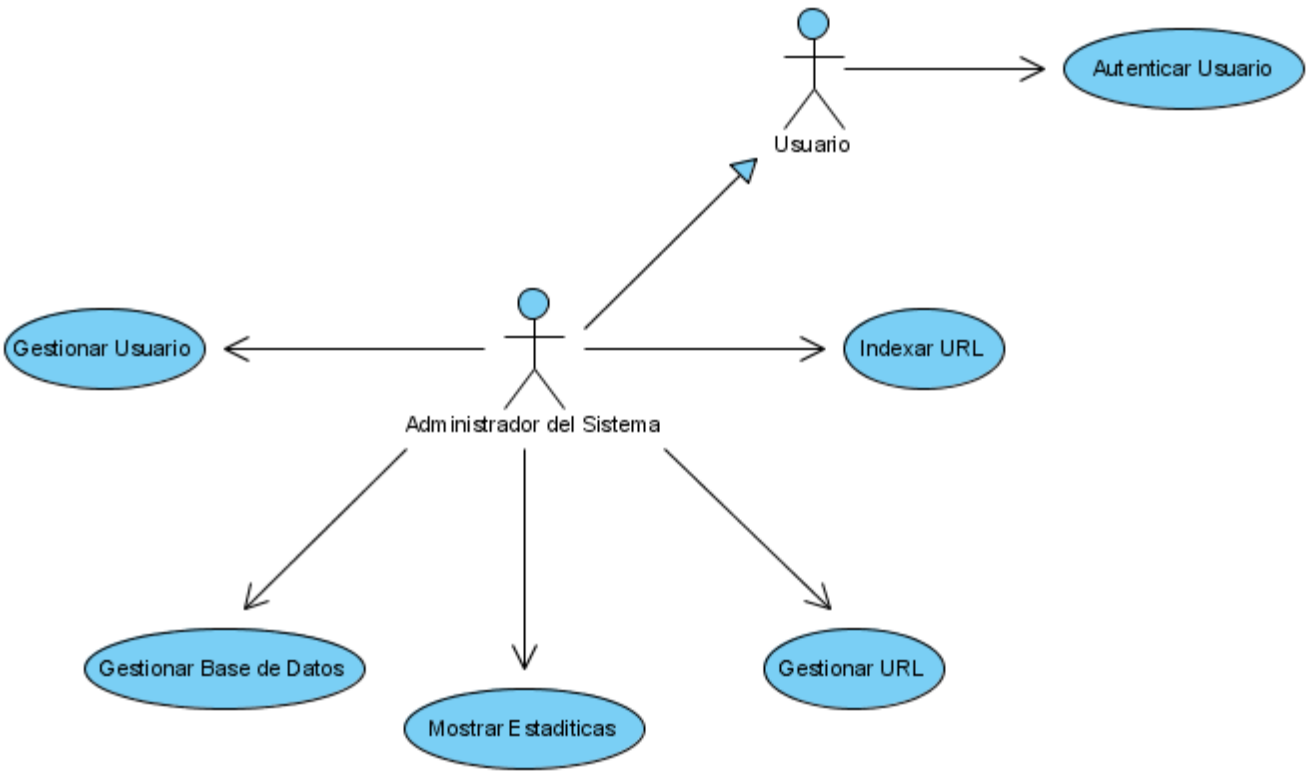

Figura 7 Diagrama de Casos de Uso del Sistema

2.6.4 Casos de Uso expandidos

Para entender la funcionalidad asociada a cada caso de uso no es suficiente con la representación gráfica del diagrama de casos de uso. Se debe elaborar una descripción que capte las principales características de los mismos. La descripción puede ser elaborada de forma breve o extendida. A continuación se muestra la descripción de los principales casos de uso.

Tabla 3 Caso de Uso expandido Autenticar Usuario

Caso de uso	
CU-3	**Autenticar Usuario**
Propósito	Autenticar la identidad presentada por un usuario
Actores Usuario	
Resumen: Caso de uso que define funciones de control de acceso al sistema, inicia cuando un usuario introduce una identidad y solicita entrar al sistema, el proceso requiere que la identidad sea válida para posibilitar el acceso a la interfaz principal del sistema.	
Referencias	
Pre condiciones	El usuario debe iniciar el sistema
Pos condiciones	
Acción del actor	**Respuesta del sistema**
1- Un usuario entra una identidad valida, compuesta por un nombre y una clave y solicita entrar al sistema	2- El sistema recibe la solicitud y comprueba si la identidad es válida verificando que esté compuesta por los datos requeridos y que existe en el sistema
	3- El sistema posibilita el acceso a la interfaz principal
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
1- Un usuario entra una identidad no valida y solicita entrar al sistema	2- El sistema recibe la solicitud y deniega el acceso notificando un mensaje de identidad no válida

Tabla 4 Caso de Uso expandido Gestionar Usuario

Caso de uso	
CU-3	**Gestionar Usuario**
Propósito	Cambiar los datos de usuario administrador del sistema
Actores Administrador del Sistema	
Resumen: Caso de uso que define funciones de cambio de datos para la cuenta de usuario administrador del sistema, inicia cuando un usuario autentificado como administrador del sistema solicita cambiar los datos de la identidad de usuario administrador, el proceso requiere que los datos de entrada sean válidos para completar el cambio de identidad en el sistema	
Referencias	
Pre condiciones	El usuario debe autenticarse como administrador del sistema
Pos condiciones	El usuario debe iniciar el gestor de datos de usuario administrador
Acción del actor	**Respuesta del sistema**
1- El usuario entra datos válidos compuestos por un nombre, una clave y una confirmación de clave y solicita cambiar la identidad de usuario administrador del sistema	2- El sistema recibe la solicitud y comprueba si los datos de entrada son válidos verificando si son los requeridos para la nueva identidad
	3- El sistema cambia los datos de la identidad de usuario administrador del sistema
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
1- El usuario entra datos no válidos y solicita cambiar la identidad de usuario administrador del sistema	2- El sistema recibe la solicitud y deniega el proceso mostrando un mensaje de datos no válidos

Tabla 5 Caso de Uso expandido Indexar URL

Caso de uso	
CU-3	**Indexar URL**
Propósito	Indexar URL disponibles en la web de la UCI
Actores Administrador del Sistema	
Resumen: Caso de uso principal que define funciones de indexación, inicia cuando el usuario administrador del sistema solicita ejecutar el proceso de indexación, el proceso posibilita entrar un conjunto de URL iniciales para su ejecución	
Referencias	CU Buscar URL
Pre condiciones	El usuario debe iniciar la sección de administrador del sistema
Pos condiciones	Proceso de indexación finalizado
Acción del actor	**Respuesta del sistema**
1- El administrador del sistema entra una lista de URL válidas y solicita iniciar el proceso de indexación	2- El sistema recibe la solicitud e inicia el proceso de indexación con la lista de URL de entrada
	El sistema procede a comprobar si la URL inicial es valida, verificando su estructura y su existencia indexa la URL, el contenido relacionado y crea una lista de URL relacionadas a las que ejecuta recurrentemente el proceso
	El sistema muestra cada URL indexada en el progreso del proceso de indexación y notifica la finalización del proceso con un conjunto de estadísticas relacionadas
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
1- El administrador del sistema entra una lista de URL no válidas y solicita iniciar el proceso de indexación	2- El sistema procede a comprobar si la URL inicial es valida, verificando su estructura y su existencia no indexa la URL y continua el proceso para otra URL de entrada

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

3.1 Introducción

En este capítulo se presentan los temas referentes al análisis y diseño del sistema, Se hace referencia a las principales tareas que se llevan a cabo en este flujo de trabajo, así como la descripción de los artefactos que se generan.

3.2 Modelo de Análisis

El modelo de análisis constituye un modelo que se utiliza para obtener una visión del sistema sobre los requisitos funcionales, expresados en un lenguaje técnico, es el resultado de la actividad de analizar los casos de uso. El modelo de análisis constituye la primera aproximación al modelo de diseño.

Durante la realización del Modelo de Análisis se precisa la elaboración de una serie de artefactos vinculados a este, el Diagrama de Clases del Análisis constituye uno de los pilares más importantes en esta etapa desarrollo. El Diagrama de Clases del Análisis es un artefacto en el que se representan los conceptos en un dominio del problema.

Una clase de análisis representa una abstracción de una o varias clases y/o subsistemas del diseño. Se centra en el tratamiento de requisitos funcionales y pospone los no funcionales para el diseño, según RUP siempre se ajusta a alguno de los estereotipos siguientes: interfaz, control o entidad (6).

- Clase interfaz: Modelan la interacción entre el sistema y sus actores.

- Clase controladoras: Coordinan la realización de uno o unos pocos casos de uso, relacionando las actividades de los objetos que implementan sus funcionalidades.

- Clases entidad: Modelan información que posee larga vida y que es a menudo persistente.

Para el desarrollo del sistema se realizó un diagrama de clases del análisis para cada caso de uso. La figura 8 que se muestra a continuación ilustra el diagrama de clases del análisis del caso de uso autenticar usuario.

3.2.1 Diagramas de Clases del Análisis.

Los diagramas de clases son diagramas de estructura estática que muestran las clases del sistema y sus interrelaciones. Estos diagramas son los más utilizados en el modelado de sistemas orientados a objetos por constituir el pilar básico del modelado con UML para mostrar lo que el sistema puede hacer (análisis), como para mostrar cómo puede ser construido (diseño).

3.2.2 Diagrama de Clases del Análisis. CU Autenticar Usuario

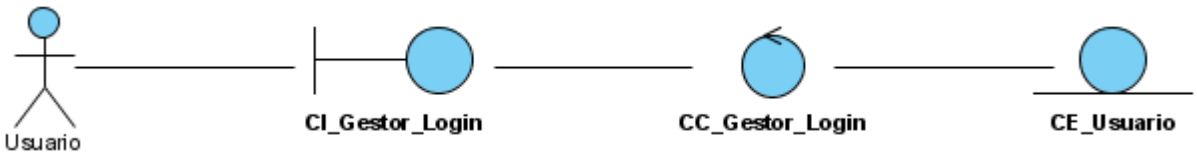

Figura 8 Diagrama de Clases del Análisis. CU Autenticar Usuario

3.2.3 Diagrama de Clases del Análisis. CU Gestionar Usuario

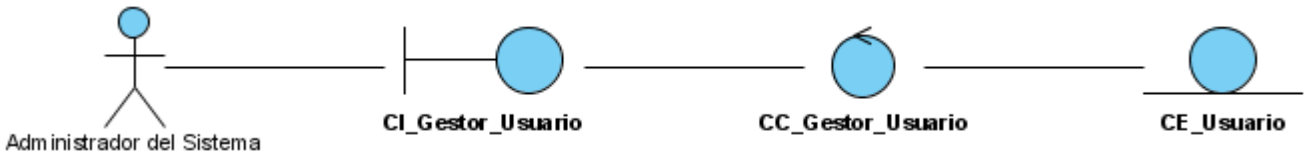

Figura 9 Diagrama de Clases del Análisis. CU Gestionar Usuario

3.2.4 Diagrama de Clases del Análisis. CU Indexar URL

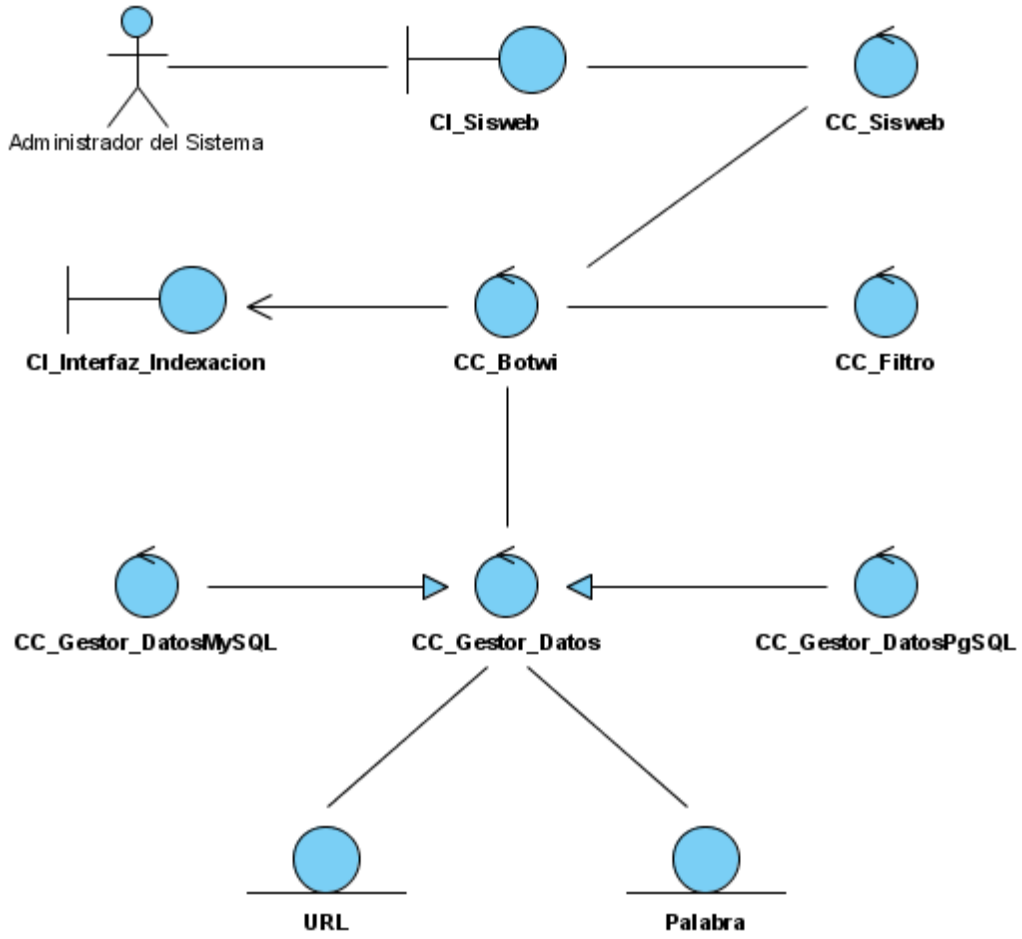

Figura 10 Diagrama de Clases del Análisis. CU Indexar URL

3.2.5 Diagramas de Colaboración del Análisis.

Los diagramas de colaboración son una forma alternativa a los diagramas de secuencia de mostrar un escenario. Estos tipos de diagramas muestran las interacciones entre objetos organizadas entorno a los objetos y los enlaces entre ellos contribuyendo a la representación de aspectos dinámicos del sistema.

3.2.6 Diagrama de Colaboración del Análisis. CU Autenticar Usuario

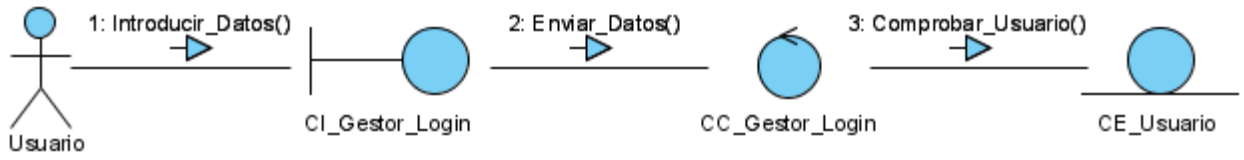

Figura 11 Diagrama de Colaboración del Análisis. CU Autenticar Usuario

3.2.7 Diagrama de Colaboración del Análisis. CU Gestionar Usuario

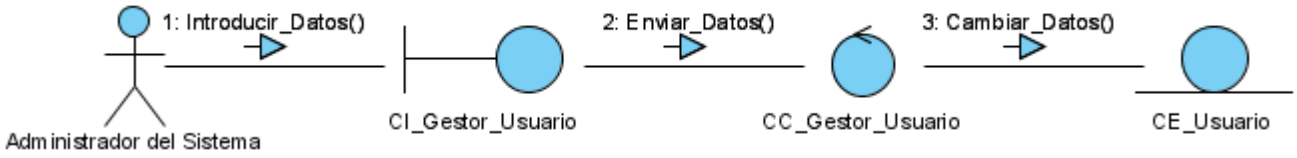

Figura 12 Diagrama de Colaboración del Análisis. CU Autenticar Usuario

3.2.8 Diagrama de Colaboración del Análisis. CU Indexar URL

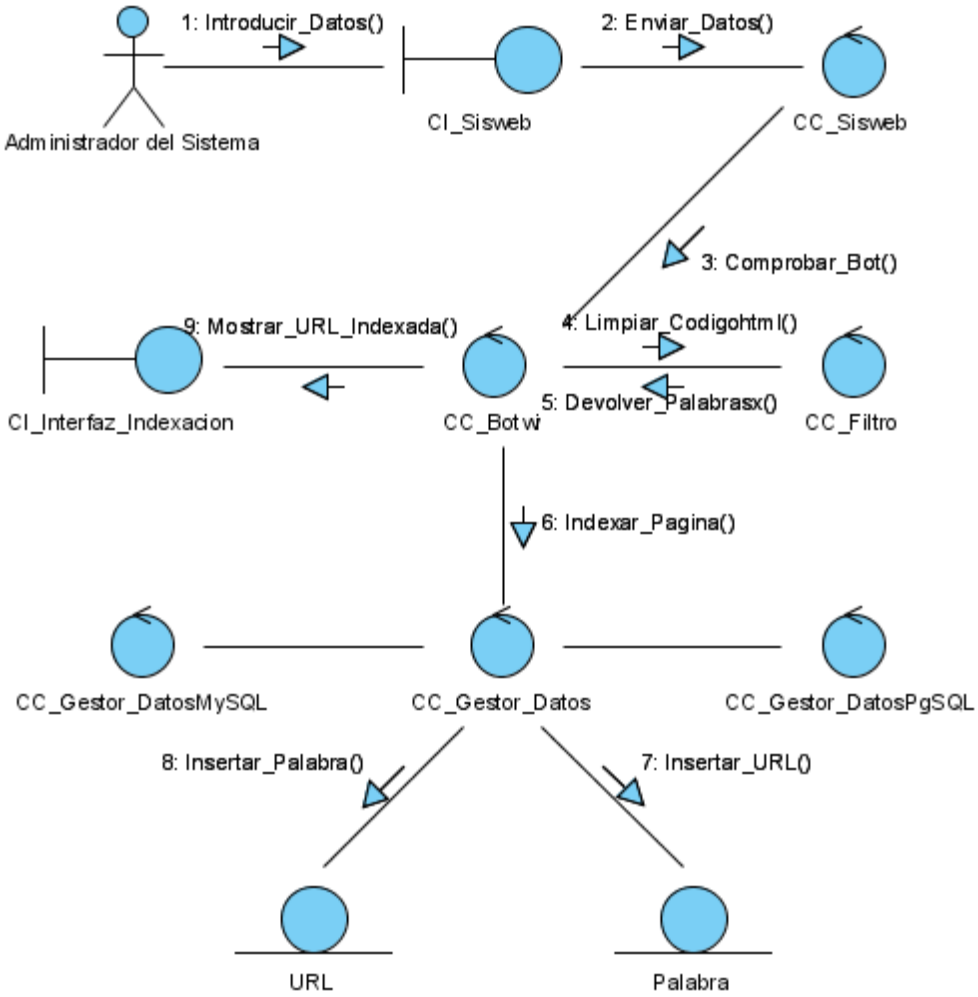

Figura 13 Diagrama de Colaboración del Análisis. CU Indexar URL

3.3 Modelo de Diseño

El modelo de diseño describe la realización física de los casos de uso centrándose tanto en los requisitos funcionales como en los no funcionales. En el diseño se modela el sistema y se confecciona su estructura (arquitectura), que sirve de soporte para todos los requisitos. Las abstracciones del modelo de diseño tienen una correspondencia directa con los elementos físicos del ambiente de implementación.

La realización de caso de uso del diseño contiene una descripción de flujos de eventos textuales, diagramas de clases y diagramas de interacción. Los diagramas de clases son los más utilizados en el modelado de sistemas orientados a objetos. Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones.

Una clase de diseño es una abstracción sin costuras con una clase o construcción similar en la implementación del sistema. Se especifican en un lenguaje de implementación (PHP, JSP, C#, etc.), además se le atribuyen visibilidad a sus atributos y operaciones (6).

3.3.1 Diagramas de Clases del Diseño

Durante el flujo de trabajo de diseño, se modela el sistema de manera que soporte todos los requerimientos, incluyendo a diferencia del análisis, a los requerimientos no funcionales. Este modelo se puede utilizar para visualizar la implementación y para soportar las técnicas de programación gráfica de la aplicación.

En el diseño de la propuesta de solución, se tienen en cuenta esencialmente los patrones Experto y Creador. El Experto establece que se debe asignar una responsabilidad a la clase experta a la información necesaria para llevarla a cabo. El Creador por su parte, indica que se le debe dar la responsabilidad a una clase para crear una instancia de otra, siempre y cuando se agregue, contenga los objetos, registre las instancias de los objetos, o tenga los datos de inicialización que serán enviados cuando sea creado el objeto.

La forma tradicional de modelar las clases del diseño, no es factible a la hora de diseñar una aplicación web. Por ese motivo, se utiliza una extensión de UML para web, que se adapta a la arquitectura de este tipo de sistemas.

A continuación se encuentran los diagramas de clases web para cada caso de uso del sistema, de forma tal que se facilite la comprensión de las relaciones entre los distintos componentes.

3.3.1 Diagrama de Clases web. CU Autenticar Usuario

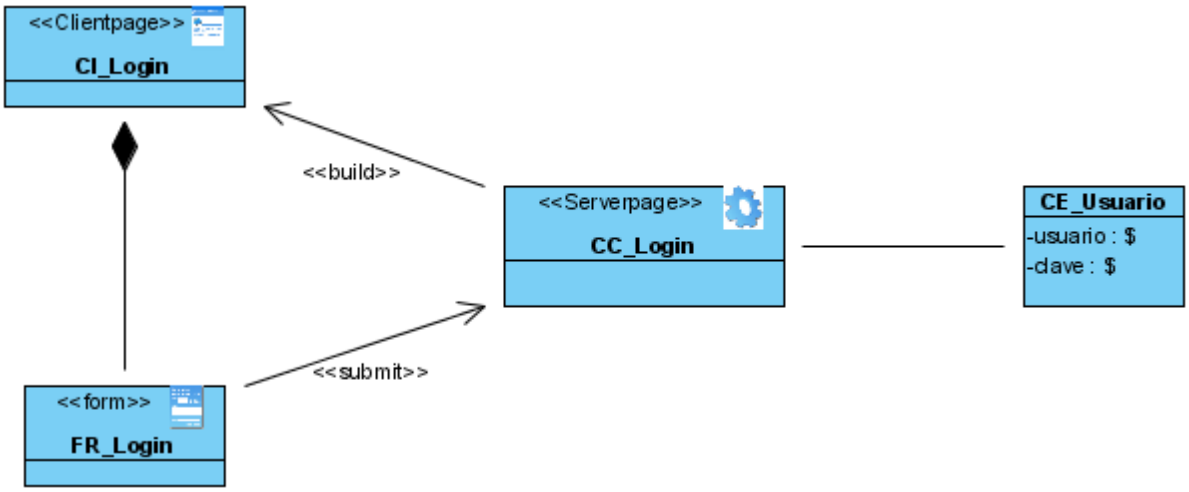

Figura 14 Diagrama de Clases web. CU Autenticar Usuario

3.3.2 Diagrama de Clases web. CU Gestionar Usuario

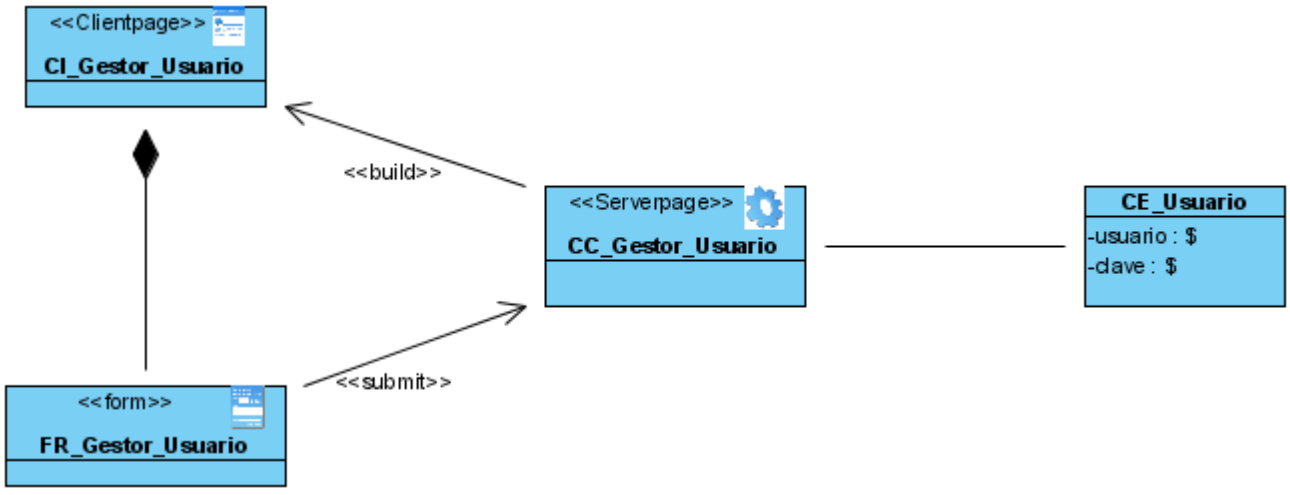

Figura 15 Diagrama de Clases web. CU Gestionar Usuario

3.3.3 Diagrama de Clases web. CU Indexar URL

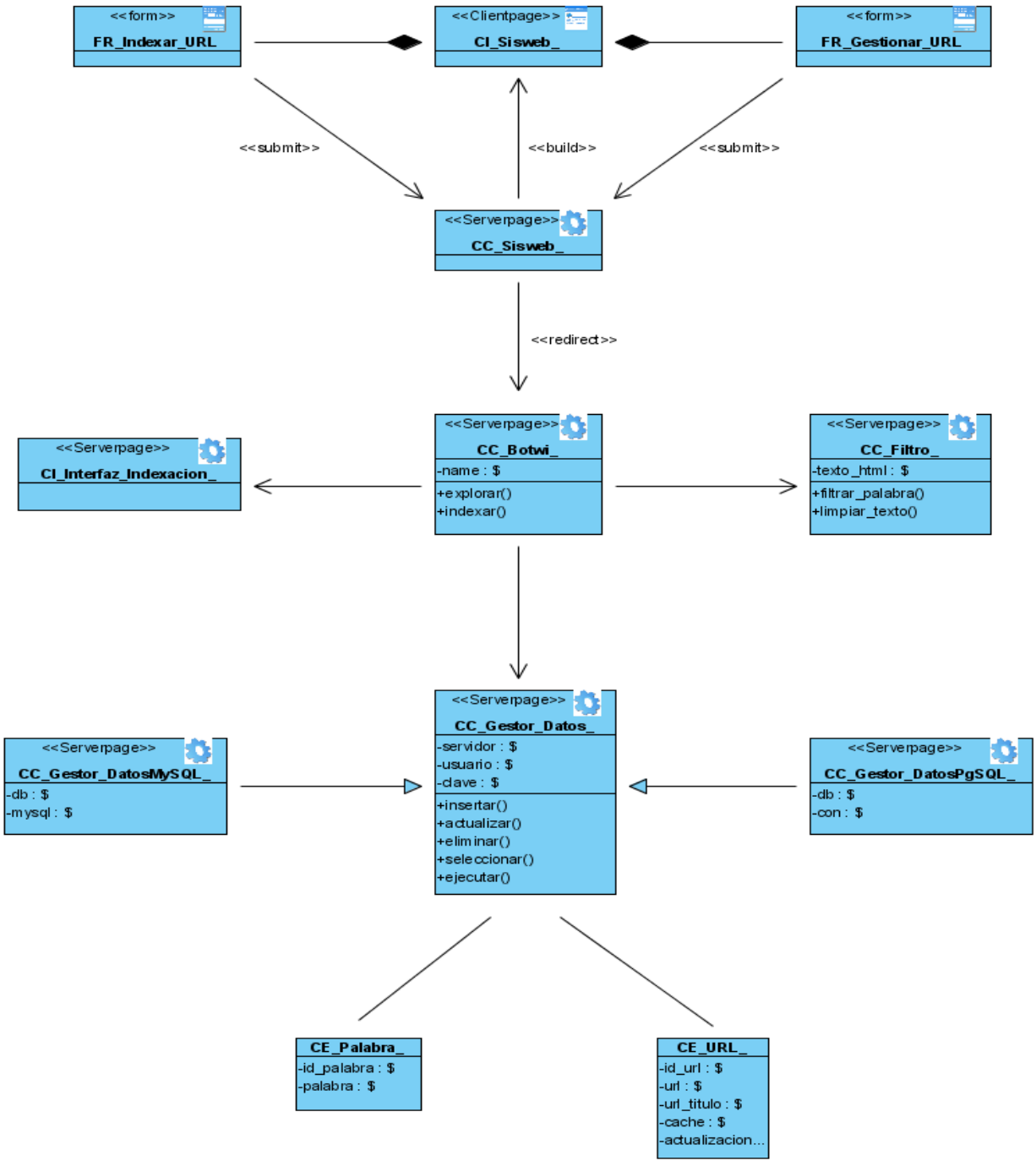

Figura 16 Diagrama de Clases web. CU Indexar URL

Tabla 6 Descripción de la clase CC_Botwi

Nombre: CC_Botwi	
Tipo de clase (controladora)	
Atributo	**Tipo**
urls	$
tipo	$
acceso	$
filtro	$
tipo_gestor	$
Responsabilidad:	
Nombre:	CrawlingInfProgresivo
Descripción:	Funcion orientada a inspeccionar la web del dominio uci, inicia la ejecución a partir de la página inicial y, logicamente, a través de ella se debe poder acceder a toda la web visible del dominio (web que no requiere una clave de acceso). Se basa en la sentencia iterativa while (o sea, mientras se cumpla una o más condiciones “c”, todo lo que esté dentro del bloque se ejecutará), que se repetirá siempre y cuando el programa tenga más páginas por inspeccionar.

3.3.4 Patrones de diseño utilizados

Con el desarrollo del modelo orientado a objeto, OOP (del inglés: Object Oriented Programming) el uso de los patrones de diseño ha tomado un auge importante, hoy en día son muy populares y están presentes en casi todos los sistemas informáticos a nivel mundial.

Un patrón de diseño constituye la descripción de clases y objetos involucrados en la solución adaptada de un problema de diseño general en un contexto particular, hace especial énfasis en la forma de comunicarse de los objetos. Identifica las clases, instancias, roles, colaboraciones, que contribuyen a lograr que el código obtenido sea flexible y que satisfaga los criterios especificados. Un patrón de diseño es una manera más práctica de describir aspectos organizacionales de la organización de un problema (7).

En el desarrollo de la solución planteada, se usaron varios patrones de diseño con el objetivo de lograr un software más acabado, robusto, flexible y de mayor calidad. Entre los distintos patrones se pueden citar los encargados de describir los principios fundamentales durante la asignación de responsabilidades a objetos GRASP (del inglés: General Responsibility Assignment Software Patterns). A continuación se describen las principales características de los patrones usados (8).

GRASP:

Experto: Al implementar un sistema es importante saber asignarle a cada clase el grupo de operaciones que debe llevar a cabo, el patrón experto recomienda asignarle a una clase solo aquellas responsabilidades para las que ella contenga la información necesaria para su cumplimiento. Durante el desarrollo de la aplicación se llevó a cabo el uso de este patrón de diseño con el propósito de disminuir el tiempo de respuesta a cada petición realizada por el cliente así como la organización del contenido.

Creador: La peculiaridades de sistema obligan a hacer un uso óptimo de la memoria del sistema, es determinante saber asignar la responsabilidad de crear una instancia de una clase solo aquella clase que la requiera y así evitar que se creen objetos repetidos.

3.3.5 Tratamiento de errores

El sistema posibilita un control interno para el tratamiento de los posibles errores que puedan ocasionarse durante la ejecución de alguna acción en el servidor o en el proceso de comunicación con el mismo, con el fin de garantizar la integridad y confiabilidad de la información que en ella se maneja. Se lleva a cabo un análisis del error ocurrido detectando la gravedad del mismo iniciando una serie de acciones y medidas en pos de mantener la integridad y funcionalidad del servidor.

En el desarrollo del sistema se han detectado una serie de acciones que pueden provocar un mal funcionamiento del servidor, especificando para cada una de ellas un mecanismo de recuperación. Los mensajes de errores son traducidos a un lenguaje de fácil comprensión para el cliente.

Algunos de los mensajes de error que pueden llegar a la aplicación cliente son los siguientes:

Tabla 7 Descripción de errores

Error	Causa	Respuesta del Sistema
ERROR_USUARIO_DUPLICADO	El cliente intenta acceder al sistema con un usuario que se encuentra autenticado.	Notificar al usuario que intenta acceder al sistema que las credenciales se encuentran en uso.
ERROR_USUARIO_DESCONOCIDO	El cliente trata de acceder al sistema con un usuario desconocido, que no ha sido registrado en el sistema.	Notificar al usuario que el usuario proporcionado no se encuentra autentificado en el servidor.
ERROR_CONEXION_PERDIDA	Puede suceder por múltiples causas que involucran principalmente problemas en la red del cliente que afectan la conexión con el servidor.	Notificar al cliente que ha perdido la conexión con el servidor.
ERROR _ DEL_SISTEMA	Error grave del sistema que impide su correcto funcionamiento.	El sistema intenta recuperar la información y restablecer los servicios.

3.3.6 Seguridad

Es de vital importancia el tema de la seguridad en el desarrollo del sistema, para ello se definió un grupo de medidas encaminadas a evitar la transformación o suplantación de la información referente al sistema.

Una de las medidas implementadas es el proceso de encriptación de todos los datos que se hacen persistentes para el sistema y que contienen información relacionada con el usuario administrador a través de una encriptación que utiliza el algoritmo MD5, este proceso también se extiende al intercambio de información entre el cliente y el servidor

Además de las medidas expuestas, se utilizaron los mecanismos y reglas de seguridad definidas en la plataforma XAMP.

3.3.7 Interfaz

Para el desarrollo del sistema se creó una interfaz de usuario que integra simplicidad y facilidad de uso, lo que aumenta el tiempo de respuesta a las solicitudes realizadas. La misma cuenta con un conjunto de botones en la parte superior izquierda que posibilitan el acceso a funcionalidades del sistema.

La interfaz del sistema cumple con las exigencias de navegación para sistemas informáticos web. A continuación se muestra la interfaz en las siguientes imágenes.

3.3.8 Interfaz de autenticación

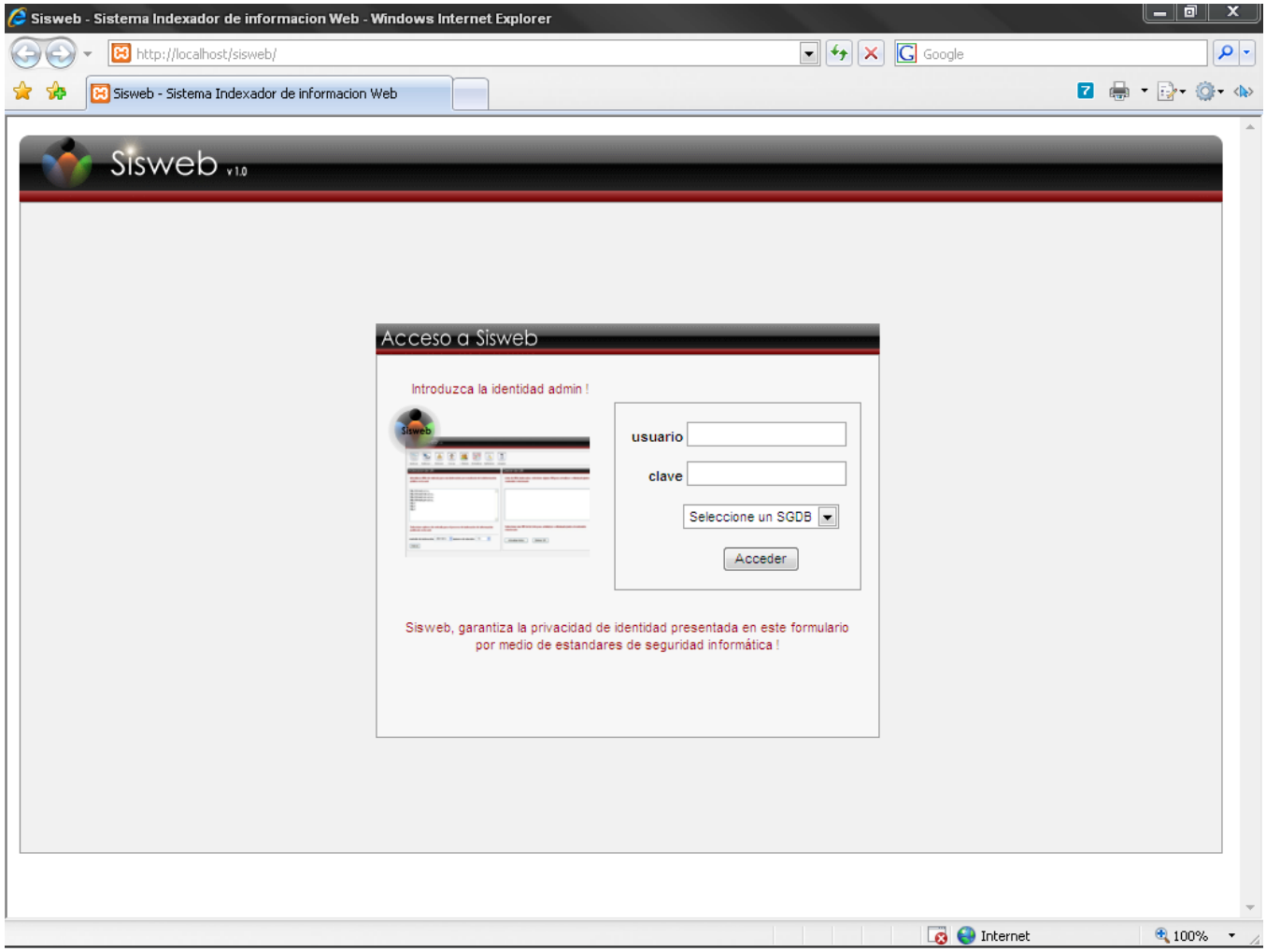

Figura 17 Interfaz de autenticación

3.3.9 Interfaz principal

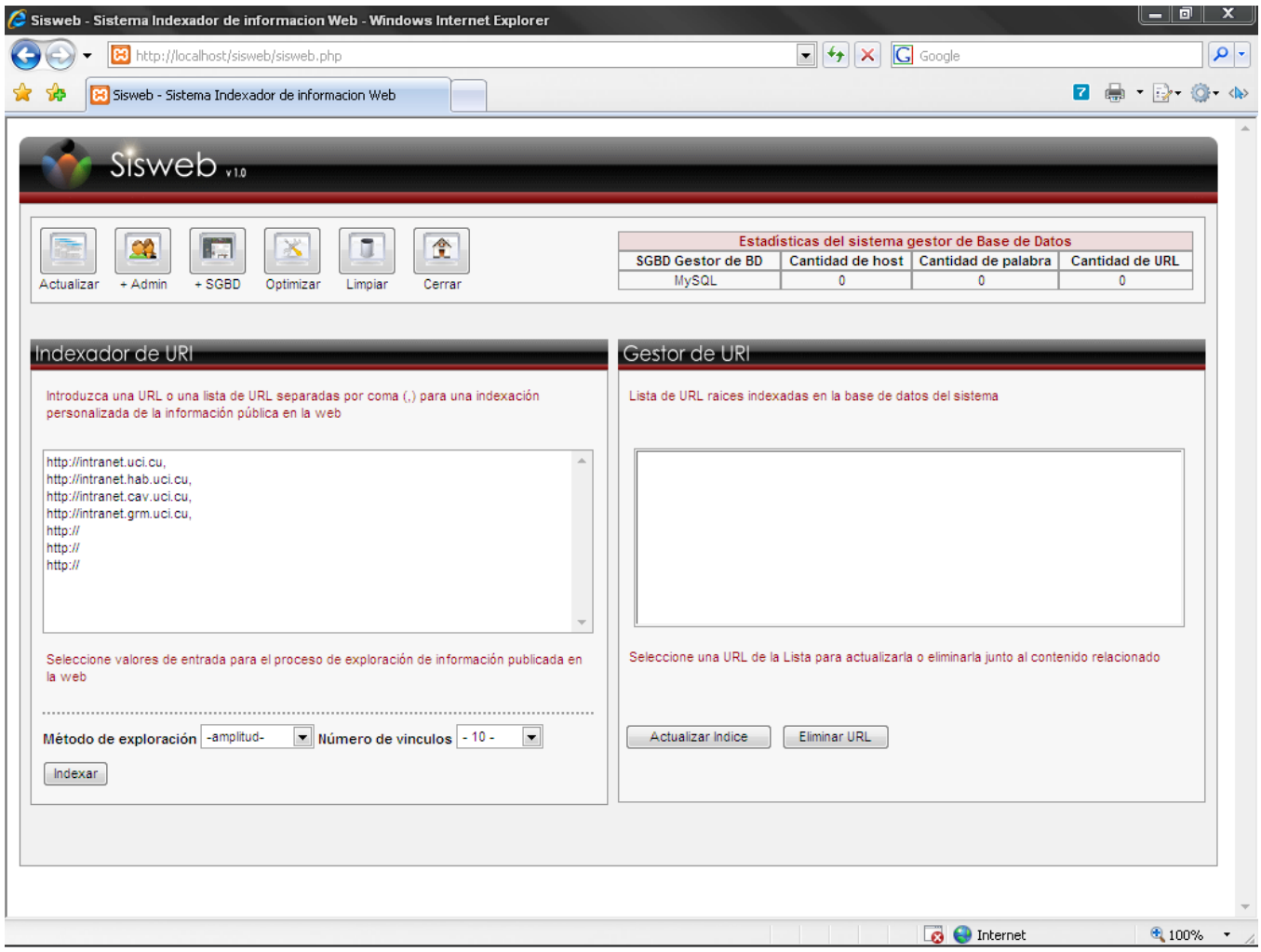

Figura 18 Interfaz principal

3.3.10 Diseño de Base de Datos

La propuesta de base de datos que se expone a continuación, satisface las necesidades de persistencia de los datos que el sistema requiere, en cumplimiento de sus requerimientos funcionales y de la propia integración al resto de la plataforma. Para diseñar la base de datos del sistema, se utilizan los modelos lógico y físico de datos.

3.3.11 Diagrama de Clases Persistentes

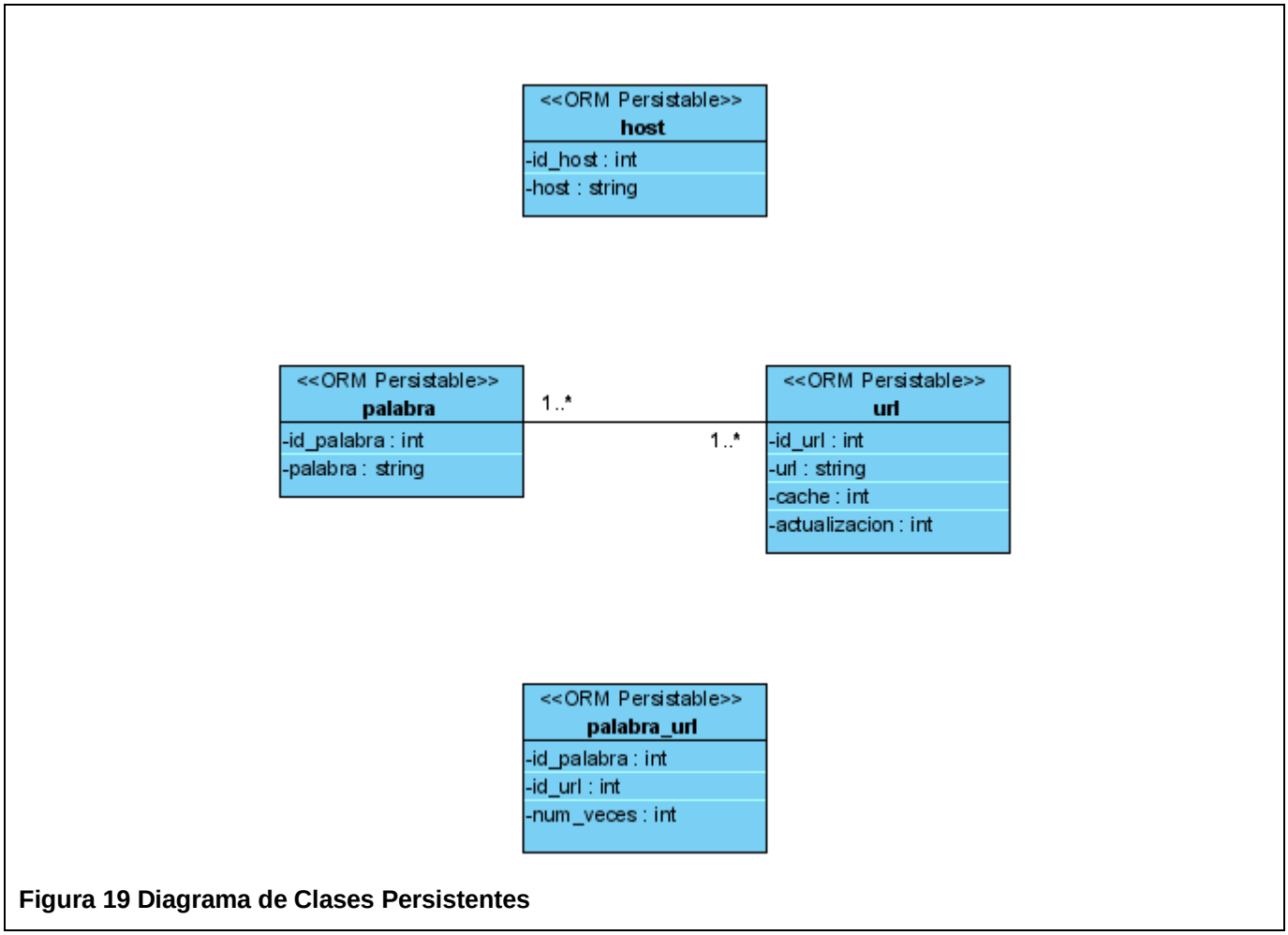

Figura 19 Diagrama de Clases Persistentes

3.3.12 Diagrama Entidad-Relación

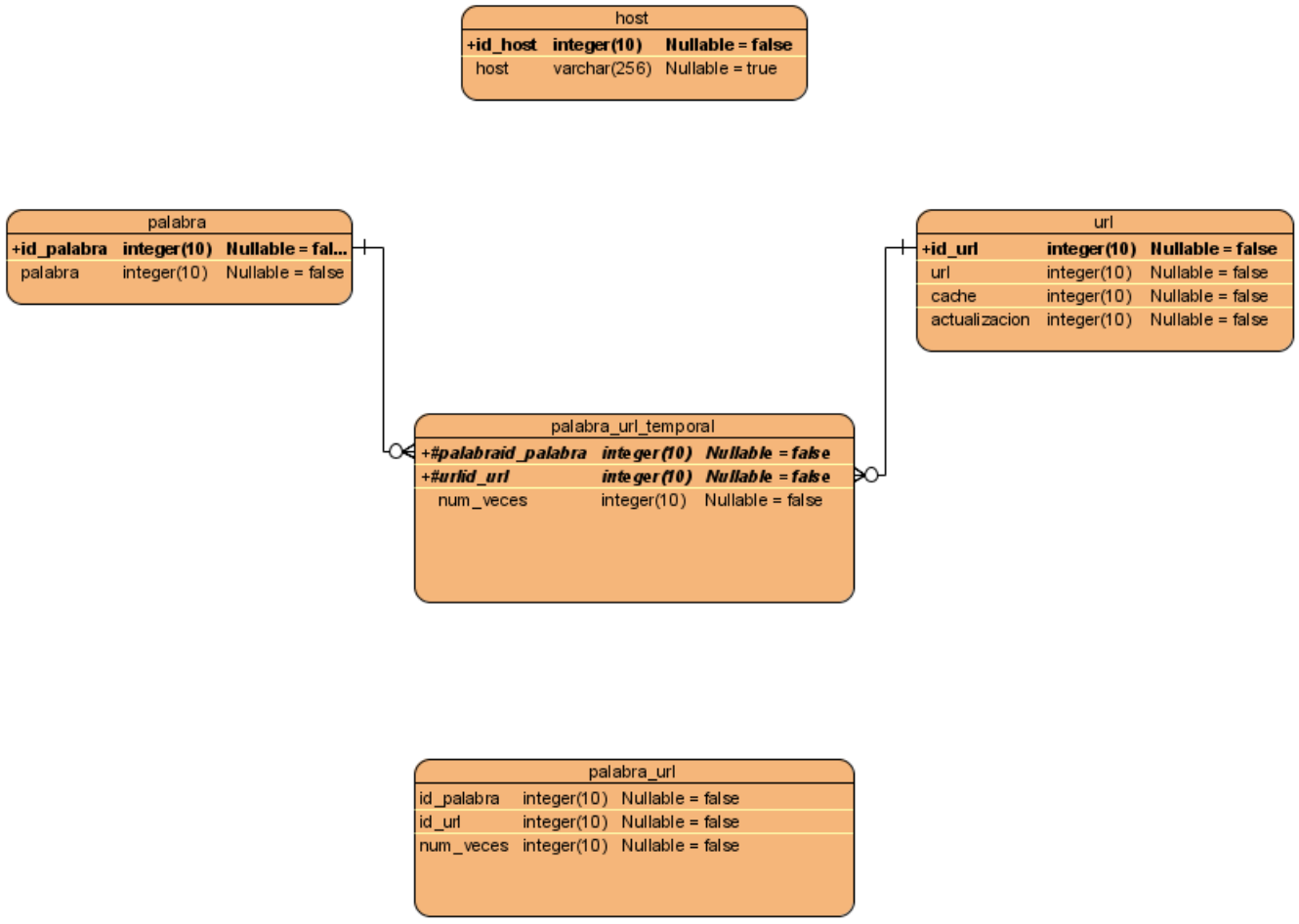

Figura 20 Diagrama Entidad-Relación

3.3.13 Descripción de las tablas.

Tabla 8 Descripción de la Tabla host

Nombre: Host		
Descripción: La tabla host, contiene la dirección web de todas las páginas raíces rastreadas. Está formada por los campos idhost y host. El primero admite valores de números enteros y es una función Auto Increment, de manera que cada vez que se inserta una página se genera el valor n del campo idhost en la fila n, valor que el algoritmo de búsqueda utilizará como identificador del segundo campo, host, que contiene los nombres (con una extensión máxima de 256 caracteres)		
Atributo	**Tipo**	**Descripción**
idhost	int(11)	
host	varchar(256)	

Tabla 9 Descripción de la Tabla Palabra

Nombre: Palabra		
Descripción: La tabla palabra, almacena todas las palabras encontradas en el proceso de rastreo. Se sirve también de dos campos, idPalabra y Palabra. El primero admite valores de números enteros y es una función Auto Increment, de manera que cada vez que se inserta una página se genera el valor n del campo idPalabra en la fila n, valor que el algoritmo de búsqueda utilizará como identificador del segundo campo, Palabra. El segundo campo es el que contiene las palabras encontradas por el rastreador y podrá tener una longitud de hasta 256 caracteres (por si falla algo y se encuentra una cadena de texto larguísima).		
Atributo	**Tipo**	**Descripción**
idpalabra	int(11)	
palabra	varchar(256)	

Tabla 10 Descripción de la Tabla URL

Nombre: URL		
Descripción: La tabla url, contiene la dirección web de todas las páginas rastreadas. Está formada por los campos idURL y URL. El primero admite valores de números enteros y es una función Auto Increment, de manera que cada vez que se inserta una página se genera el valor n del campo idURL en la fila n, valor que el algoritmo de búsqueda utilizará como identificador del segundo campo, URL, que contiene los nombres (con una extensión máxima de 512 caracteres) de las URLs analizadas.		
Atributo	**Tipo**	**Descripción**
idurl	int(11)	
url	varchar(512)	
url_titulo	varchar(256)	
cache	longtext	
actualizacion	timestamp	

Tabla 11 Descripción de la Tabla Palabra_URL

Nombre: Palabra_URL		
Descripción: La tabla palabras_url, relaciona las tablas palabra y url. Está compuesta por tres campos: idURL (que se corresponderá con el idURL de una página “P” en la tabla urls), idPalabra (que tendrá el mismo valor que el idPalabra de una palabra “W” en la tabla palabras) y Num_Veces, que será el campo determinante a la hora de buscar, pues contendrá una fila para cada idPalabra e idURL con un entero “K” que se corresponderá con el número de veces que la palabra W aparece en la página “P”.		
Atributo	**Tipo**	**Descripción**
idpalabra	int(8)	
palabra	int(8)	
num_veces	int(11)	

Tabla 12 Descripción de la Tabla Palabra_URL_Temporal

Nombre: Palabra_URL_Temporal		
Descripción: La tabla palabras_url_temporal, relaciona las tablas palabra y url, almacena la información de forma temporal obtenida en el proceso de indexación que posteriormente será transferida a la tabla palabra en el proceso de optimización de la base de datos. Está compuesta por tres campos: idURL (que se corresponderá con el idURL de una página P en la tabla urls), idPalabra (que tendrá el mismo valor que el idPalabra de una palabra W en la tabla palabras) y Num_Veces, que será el campo determinante a la hora de buscar, pues contendrá una fila para cada idPalabra e idURL con un entero K que se corresponderá con el número de veces que la palabra W aparece en la página P.		
Atributo	**Tipo**	**Descripción**
idpalabra	int(8)	
palabra	int(8)	
num_veces	int(11)	

CAPÍTULO 4: IMPLEMENTACIÓN DEL SISTEMA

4.1 Introducción

En el presente capítulo se describen los temas referentes a la implementación y prueba del sistema, se hace referencia a las principales tareas que se llevan a cabo en este flujo de trabajo, así como la descripción de los artefactos que se generan en el mismo.

4.2 Modelo de Implementación

En el flujo de trabajo de diseño se propone crear un plano del modelo de implementación, por lo que sus últimas actividades están vinculadas a la creación del modelo de despliegue. El flujo de trabajo de implementación describe cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue.

Los diagramas de despliegue y componentes, que son artefactos generados en este flujo de trabajo, conforman lo que se conoce como un modelo de implementación, al describir los componentes a construir y su organización y dependencia entre nodos físicos en los que funcionará la aplicación.

Los componentes constituyen la parte modular del sistema, encapsulan implementación y un conjunto de interfaces y proporciona la realización de los mismos. Un componente típicamente contiene clases y puede ser implementado por uno o más artefactos (ficheros ejecutables, binarios, etc.).

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación (6).

Para la realización de la solución propuesta se desarrollaron una serie de componentes, los cuales se encuentran referenciados en el diagrama que se ilustra a continuación.

4.2.1 Diagramas de Componentes

Los diagramas de componentes se utilizan para modelar la vista estática de un sistema. Muestra la organización y las dependencias lógicas entre un conjunto de componentes de software, sean éstos componentes de código fuente, librerías, binarios o ejecutables. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema.

4.2.2 Diagrama de Componentes General

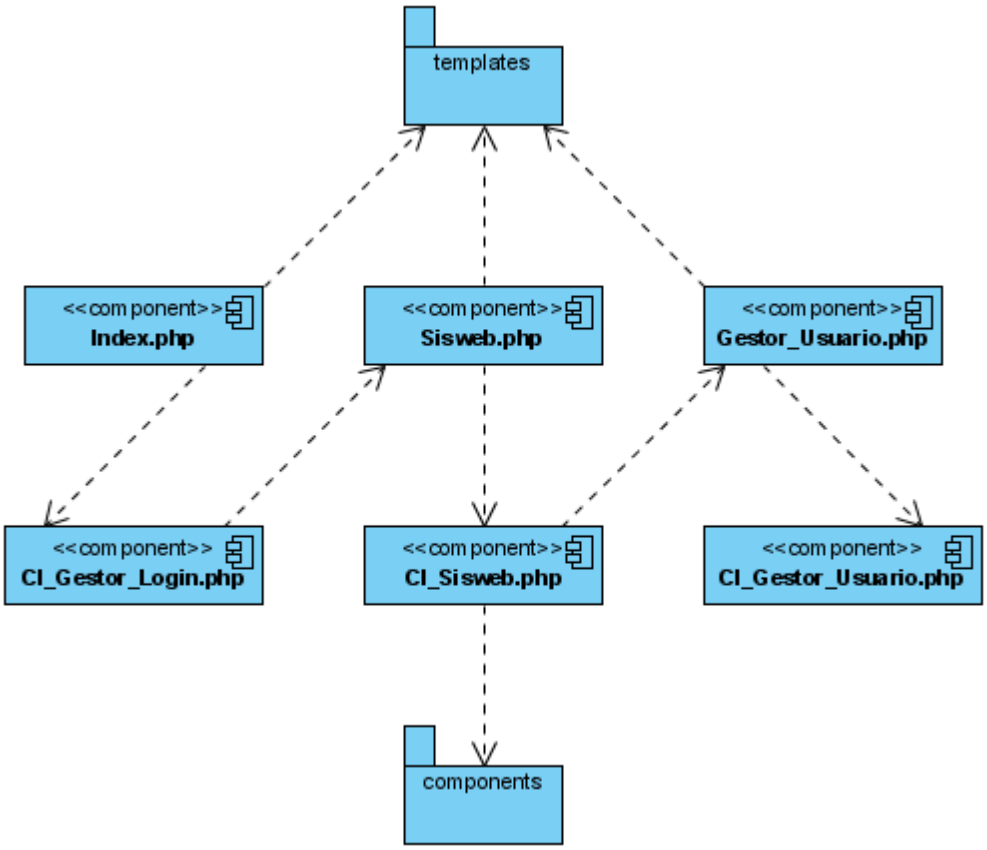

Figura 21 Diagrama de Componentes General

4.2.3 Diagrama de Componentes. Paquete components

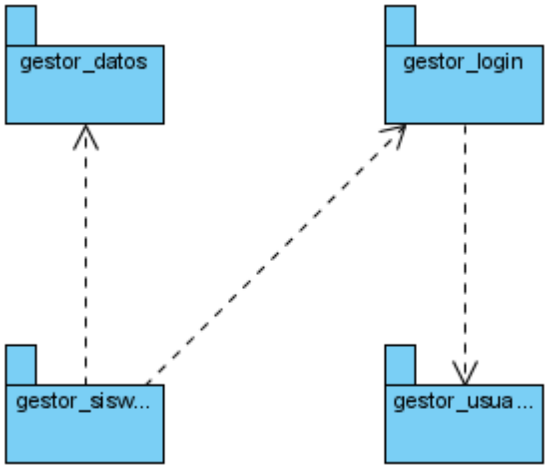

Figura 22 Diagrama de Componentes. Paquete components

4.2.4 Diagrama de Componentes. Paquete gestor_datos

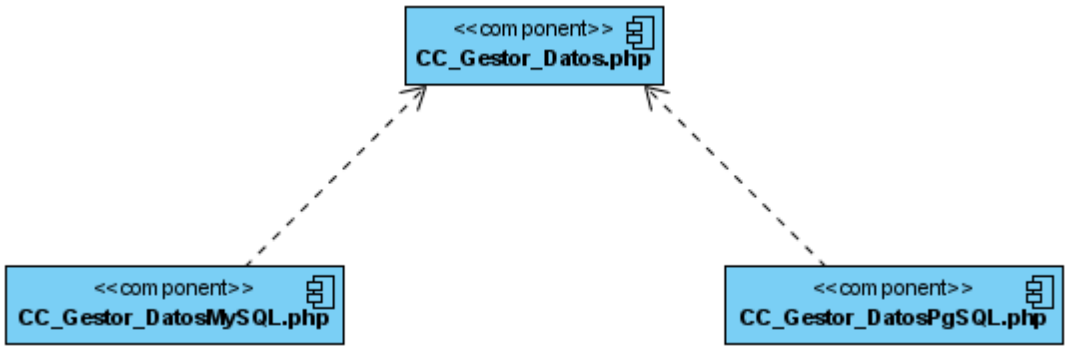

Figura 23 Diagrama de Componentes. Paquete gestor_datos

4.2.5 Diagrama de Componentes. Paquete gestor_login

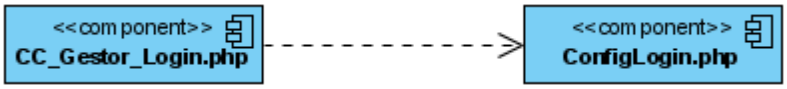

Figura 24 Diagrama de Componentes. Paquete gestor_login

4.2.6 Diagrama de Componentes. Paquete gestor_sisweb

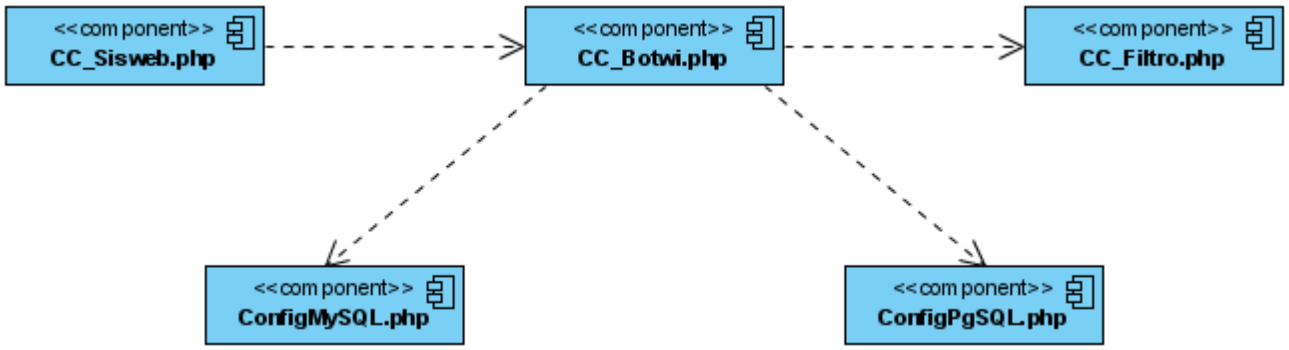

Figura 25 Diagrama de Componentes. Paquete gestor_sisweb

4.2.7 Diagrama de Componentes. Paquete gestor_usuario

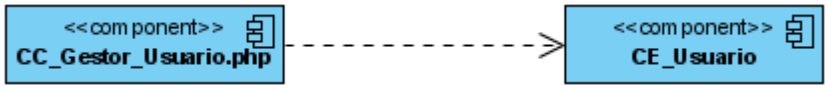

Figura 26 Diagrama de Componentes. Paquete gestor_usuario

4.2.8 Diagrama de Componentes. Paquete template

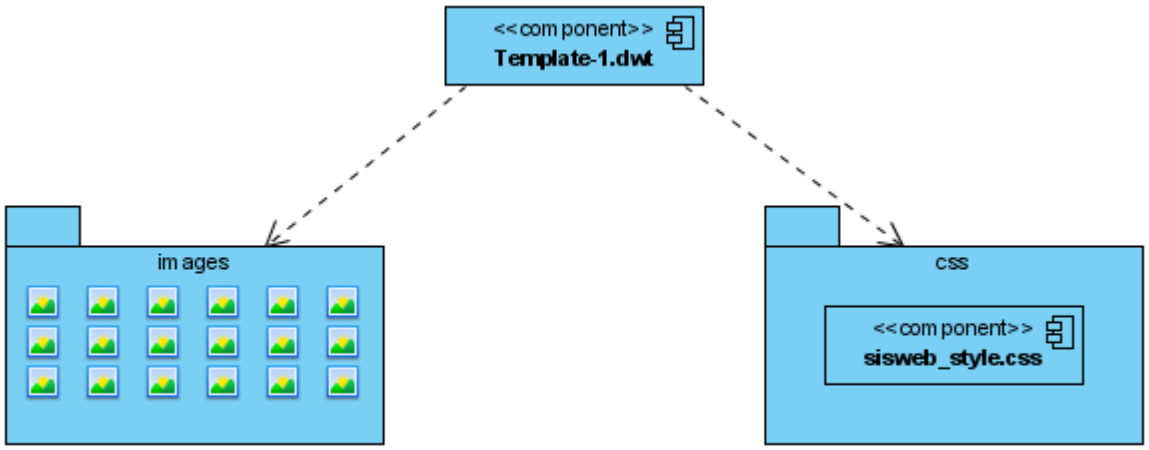

Figura 27 Diagrama de Componentes. Paquete template

4.2.9 Diagrama de Despliegue

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Es decir, se sitúa el software en el hardware que lo contiene. Cada hardware se representa como un nodo.

Un nodo se representa como un cubo, es un elemento donde se ejecutan los componentes, representa el despliegue físico de estos componentes (6).

Atendiendo a las características del sistema antes planteadas se realizó el diagrama de despliegue que se muestra a continuación, el cuál satisface todas las necesidades y exigencias de la aplicación.

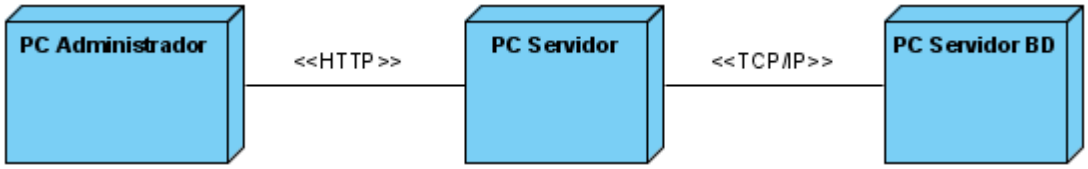

Figura 28 Diagrama de Despliegue

CONCLUSIONES

En el presente trabajo se dio cumplimiento a los objetivos propuesto, se realizaron un conjunto de actividades con el propósito de desarrollar el proyecto planteado. Se profundizó en el análisis del funcionamiento de las tecnologías relacionadas a los procesos de búsqueda e indexación de información web y se establecieron las bases para el desarrollo de nuevos proyectos.

El sistema informático desarrollado implementa los dos tipos de rastreo de información web existentes (exploración en amplitud y en profundidad) utilizados por los principales sistemas de este tipo, debido a los niveles de exigencia para la implementación de los procesos del sistema se utilizó RUP como metodología para organizar el desarrollo del software.

La solución informática que se presenta posibilita la gestión de información relacionada a URLs del dominio uci.cu establecido para el sistema, la organización y localización de la información obtenida como resultado de los procesos de rastreo fue concebida para la estructura de almacenamiento índice invertido que elimina redundancia de datos y se desarrollaron procesos para la presentación de estadísticas relacionadas al estado de la base de datos.

Hay que tener en cuenta que proyectos como el propuesto en este documento constituyen una contribución al desarrollo de buscadores verticales. Estos buscadores encontrarán un determinado espacio en la red que hará de las búsquedas especializadas un gran competidor de los grandes, y sobretodo el uso de buscadores internos de información.

Sin duda, este proyecto será muy útil si se sigue desarrollando y se aplica a nivel productivo, ya que el futuro está en la red.

RECOMENDACIONES

En este trabajo se obtuvo un sistema informático en una primera versión expuesto a mejoras que se determinaron en el ciclo de desarrollo del sistema y se exponen a continuación para el desarrollo de versiones posteriores:

- **Optimizar el rastreador:** Analizar posibles vías de optimización del algoritmo de rastreo. Por ejemplo, en la realización de los procesos de rastreo de la red sería conveniente analizar los log del servidor para cada URL visitada con el objetivo de obtener información sobre la fecha de modificación del contenido y de esta forma optimizar el tiempo de ejecución de los procesos de rastreo. Además, se podría considerar la posibilidad de incorporar al rastreador un funcionamiento genérico para toda la red de manera que, pudiese analizar todo tipo de páginas. En futuras versiones de este sistema también se espera que pueda tratar con el protocolo de exclusión de robots y tener la capacidad de cubrir cualquier otro dominio.

BIBLIOGRAFÍA

1. **Castells, Pablo.** La Web Semántica. 2003.
2. La Sociedad de la Información. [En línea] 2007. Disponible en: http://www.sociedadelainformacion.com/20011103/invisible/internetprofundo.htm.
3. **Pressman.** Ingeniería del Software un enfoque práctico. 2001.
4. **Izquierdo; y otros.** Ahijado's blog. [En línea] 2008.
5. Tigris.org. [En línea] 2006. http://subversion.tigris.org/.
6. **Jacobson, Ivar; y otros.** El Proceso Unificado de Desarrollo de Software. 2000.
7. **Larman, Craig.** UML y Patrones. 2004.
8. **Erich Gamma; y otros.** Gang of Four. 1994.
9. **Arasu, Arvind; Cho, Junghoo; García-Molina, Hector.** *Searching the Web*. 2001. Disponible en: http://oak.cs.ucla.edu/~cho/papers/cho-toit01.pdf.
10. **Cho, Junghoo; García-Molina, Hector.** *Parallel Crawlers*. 2002. Disponible en: http://www2002.org/CDROM/refereed/108/.

    Cho, Junghoo; García-Molina, Hector; Page, Lawrence. *Efficient Crawling Through URL Ordering*. *1998.* Disponible en: http://dbpubs.stanford.edu:8090/cgibin/makehtml.cgi?document=1998/51&format=0&page=1#page1.
11. **Battelle, John.** *The Search*. 2005.
12. **Hernández Orallo, José; Ramírez Quintana, José; Ramírez, César.** *Introducción a la minería de datos*. España, 2004 ED: Pearson.
13. **Belew, Richard.** *Finding out about: a cognitive perspectibe on Search Engine technology and the WWW*. EEUU, 2000 ED: Cambridge University Press.
14. **Checkland, Peter.** *Information, Systems and Information Systems*. Inglaterra, 2004 ED: Wiley.

    Cho, Junghoo; García-Molina, Hector. *The Evolution of the Web and Implications for an Incremental Crawler*. 2000. Disponible en: http://oak.cs.ucla.edu/~cho/papers/cho-evol.pdf.
15. **Senn, James.** *Information technology*. 2004 ED: Pearson.
16. **Oates, Briony.** *Researching Information Systems and Computing*. Inglaterra, 2006 ED: Sage.
17. **Milstein, Sarah; MacDonald, Matthew.** *Google: the missing manual*. Marzo 2004, ED: O'REILLY.
18. **Ntoulas, Alexandros; Cho, Junghoo; Olston, Christopher.** *What's new on the Web?: the evolution of the web form a search engine perspective*. Disponible en: http://www.cs.cmu.edu/~olston/publications/webstudy.pdf.
19. **Gallardón, Pablo.** *El secreto de Google y el álgebra lineal*.
20. **Heydon, Allan; Najork, Marc.** *Mercator: A Scalable, Extensible Web Crawler*. Disponible en: http://cgi.di.uoa.gr/~ad/MDE519.docs/scalable-crawler.pdf.

21. **Page , Lawrence; Brin, Sergey.** *The anatomy of a Large-Scale Hypertextual Web Search Engine*.2003. Disponible en: http://infolab.stanford.edu/~backrub/google.html.
22. **Gulli, Antonio; Signorini, Alessio.** *The indexable Web Size is more than 11.5 billion pages*. 2005. Disponible en: http://www.cs.uiowa.edu/~asignori/web-size/.
23. *Search Engine Optimization*. [en linea] 2008. http://www.seochat.com.
24. *OJOBuscador.* [en linea] 2008. http://www.ojobuscador.com.
25. *The Source For Search Engine Marketing.* [en linea] 2007. http://www.searchenginewatch.com.
26. *The Best Search Engines.* [en linea] 2007. http://www.lib.berkeley.edu/TeachingLib/Guides/Internet/SearchEngines.html.

ANEXOS

En las siguientes gráficas se representan estadísticas obtenidas de los procesos de indexación para los métodos de exploración en amplitud en el tiempo de 1 hora con parámetros cantidad de host (22), cantidad de URL (1034) y cantidad de vínculos (10) y en profundidad en el mismo tiempo con los parámetros cantidad de host (8), cantidad de URL (1245) y cantidad de vínculos (10).

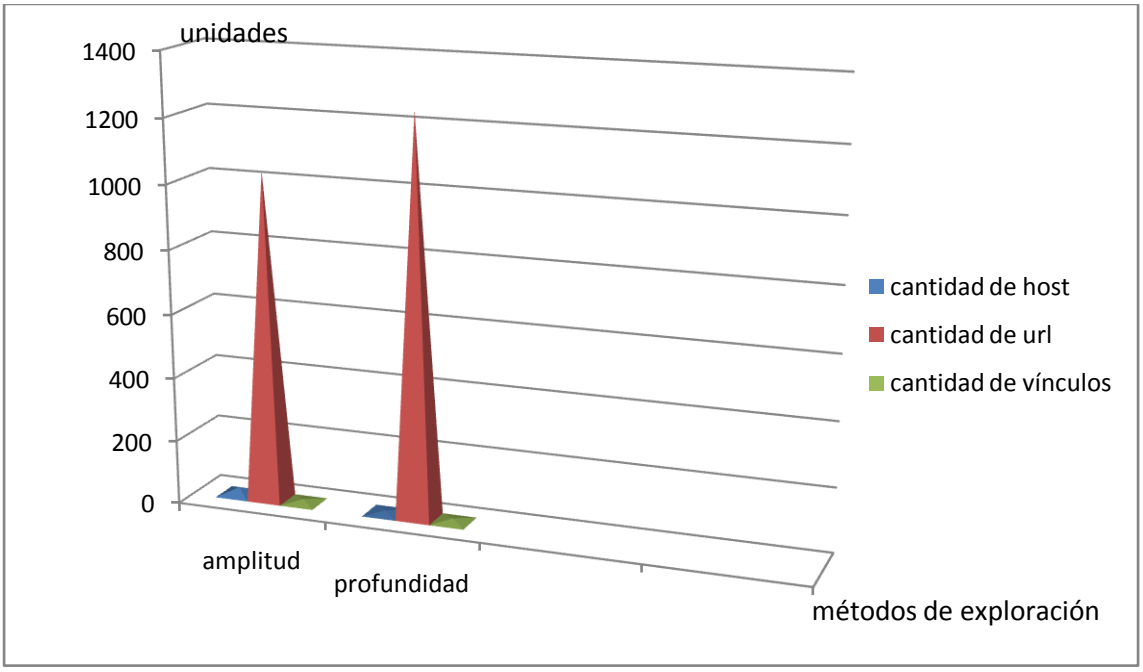

Figura 29 Gráfica de estadísticas en unidades de URL

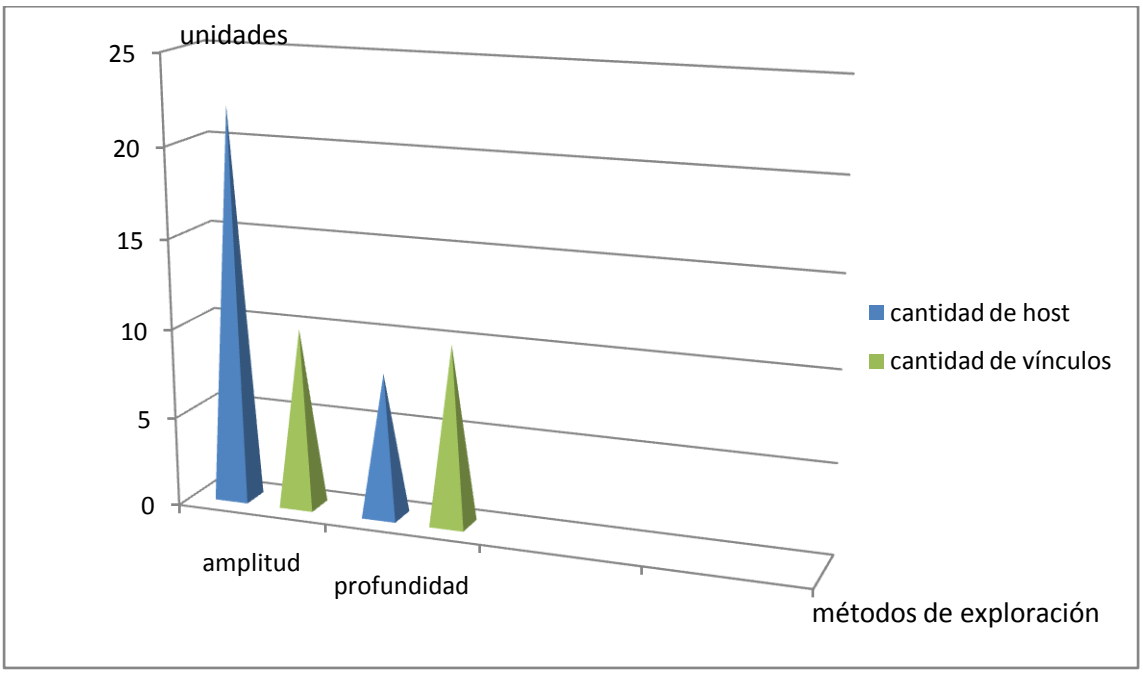

Figura 30 Gráfica de estadísticas en unidades de host

GLOSARIO

El argot informático es un lenguaje que a primera vista asusta. Por este motivo, creo que es necesario que, antes de comenzar a leer mi trabajo, se tengan unas nociones básicas de los conceptos relacionados con los motores de búsqueda.

- **Red:** Una red es un conjunto de ordenadores y otros dispositivos periféricos conectados unos a otros para comunicarse y transmitir datos entre ellos.

- **Servidor:** Ordenador que actúa como unidad de archivo central en una determinada red, que puede ser local o Internet.

- **Internet:** Internet es una red de redes a escala mundial que se basa en la interconexión pública de muchos ordenadores que funcionan independientemente pero que permiten compartir documentos (las páginas web, por ejemplo) hallados en los distintos servidores que comparten un mismo protocolo de comunicación (TCP/IP).

- **Intranet:** Red de computadoras dentro de una red de área local (LAN) privada, empresarial o educativa que proporciona herramientas de Internet.

- **Uniform Resource Locator (URL):** Dirección de una cierta página en Internet.

- **Link:** Enlace, hipervínculo. Conexión con otro documento web por medio de la dirección URL. Los enlaces aparecen en el texto de un sitio web generalmente en forma de texto subrayado o de distinto color.

- **Hyper Text Markup Language (HTML):** El HTML es un lenguaje de marcas diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas web. Es uno de los formatos más populares para la creación de documentos.

- **Algoritmo:** Un algoritmo es un conjunto finito de instrucciones o pasos que sirven para ejecutar una tarea o resolver un problema.

- **Interfaz de Programación de Aplicaciones (API):** Una API es un conjunto de especificaciones de comunicación entre componentes *software*.

- **Robot de búsqueda:** Programa diseñado para recorrer la web siguiendo sus enlaces. Se les llama de diversos modos: arañas, *crawlers*, *spiders*, rastreadores.

- **Clúster:** Porción de una unidad de almacenamiento en la que se almacenan los datos de los ficheros.

- **Nodo:** Por definición, punto donde convergen más de dos líneas pero, en términos informáticos, cualquier ordenador conectado a una red, cualquier punto de confluencia en una red.

- **Dominio:** Los dominios son el sistema de distribución de direcciones en Internet. Por ejemplo, *uao.es* constituye un dominio y cada página cuya *URL* derive de este término formará parte de dicho dominio; la parte de la derecha (*.es*) es lo que se llama un dominio de alto nivel y cada país tiene uno que le corresponde.

- **Stopwords**: Palabras que son utilizadas con mucha frecuencia. Estas palabras no son consideradas por ningún buscador, sino que son filtradas quedando fuera de cualquier indexación.