60

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ingeniería

Desarrollo de un sistema de consulta a catálogos bibliográficos a través de Web

(con un procedimiento automático)

T E S I S

que para obtener el título de

INGENIERO EN COMPUTACIÓN

presenta

Eva Edith López López

directora

Ing. Laura Sandoval Montaño

M é x i c o, D. F., 2 0 0 0

Dedico con mucho cariño este trabajo a Dios, Julia y Juan por dejarme Ser..

A Gaby y Mariano, por contribuir con los tomos dos y tres de Tesis de Ingeniería en casa.

A mis abuelos y tíos por siempre estar ahí.

A mi pequeño Rolando por desvelarse conmigo tantas noches sin fallar.

Agradecimientos

A Carlos Taboada, Thelma del Castillo y al equipo de la CSR-DGSCA por lo viejos tiempos de trabajo y amistad.

Y especialmente a:
Laura Sandoval, Daniel Sol, Antonio González y Germán Santos por su apoyo para esta tesis, además de darme su cariño y amistad.

A todos ellos, Gracias..

Prefacio

Este trabajo, pretende describir las actividades efectuadas para la realización del Sistema Automático en la generación de servicios de consulta a catálogos publicados en Web *"GENÉRICO"* y su respectivo sistema de actualización automática de catálogo de datos. Para este propósito escribí cuatro capítulos.

En el capítulo I (antecedentes), pretendo situar dentro del marco de Internet y su evolución, a la publicación de catálogos en Web, así como su importancia y trascendencia como un servicio en Internet.

El capítulo III (software libre), forma parte de una confirmación del ¿por qué, el sistema por automatizar, está basado en software libre?. La propuesta que yo presento, se enfoca particularmente hacia la automatización del servicio, aunque en el análisis realizado para cumplir este objetivo, me confirmó el éxito del uso de las herramientas previamente elegidas, además, me guió en la correcta elección sobre el lenguaje de programación a utilizar para obtener el producto esperado.

El capítulo II y IV, son propiamente las fases de análisis y desarrollo de *GENÉRICO*, describiendo finalmente los resultados ya esperados, confirmando, los beneficios otorgados por la decisión del uso del software libre.

Finalmente agrego 2 Anexos. En el anexo1, incluyo el total de código del sistema *GENERICO*, resaltando su mínimo tamaño en código generado, su portabilidad transparente para otros sistemas operativos como son: Solaris, Linux (en sus múltiples versiones), Irix, e incluso, para Windows con Perl. En el anexo2, se incluyen los tres manuales de usuario correspondientes a: el manual de usuario del sistema *GENÉRICO*; el manual de usuario del servicio de consulta, producto de GENÉRICO ; y el manual de usuario para el sistema de actualización automática de los catálogos de datos.

Eva López, enero del 2000

Capítulo I **Antecedentes**

Capítulo I **Antecedentes**

A. Introducción a Internet y al Universo del Word Wide Web.

1. Definición Internet

Ya se ha escrito mucho acerca de lo que significa Internet, pero no está de más definir el término desde el punto de vista significado y funcional.

a) Definición oficial[1] del término Internet:

El término Internet está definido oficialmente desde el 24 de Octubre de 1995 por la FNC (*Federal Networking Council*) "Internet" se refiere al sistema global de información que está lógicamente interconectado en una unidad por un espacio único de direcciones basado en el Protocolo de Internet (IP) o sus subsecuentes ampliaciones/versiones (IPv6); es capaz de adoptar comunicaciones usando el conjunto de aplicaciones del Protocolo de Control de Transmisión / Protocolo de Internet (TCP/IP) o sus subsecuentes ampliaciones/versiones, y/o otros protocolos compatibles con IP; y provee, usa o hace accesible, ya sea de modo público o privado, servicios de alto nivel que se asientan en las comunicaciones e infraestructura relacionada descrita en la presente.

b) Definición funcional:

Llamamos Internet al sistema global de redes de computadoras que se comunica con un mismo protocolo (TCP/IP).

Esta red de computadoras está interconectada por canales de alta velocidad, ya sea circuitos telefónicos, fibra óptica, enlaces satelitales e incluso vía microondas.

Su protocolo estándar de interconexión, permite que cada red que integra este sistema, sea diversa, es decir tener cada una diferentes características como: topología de red, plataforma de operación, tamaño, ubicación geográfica, velocidad de comunicación, medio de enlace, etc.

A diferencia de las tecnologías desarrolladas por una compañía, consorcio o por encargo de alguna entidad gubernamental, la Internet se ha nutrido del esfuerzo de muchos especialistas del ramo y es regulado por muchas instituciones simultáneamente. Quizá la mayor parte del desarrollo de lo que hoy constituye Internet fue realizado por universidades y personas en

[1] La resolución se puede encontrar en: http://www.fnc.gov/Internet_res.html

instituciones que no pretendían lucrar con ella, lo que ha sido determinante en su conformación y desarrollo.

2. Historia de Internet.

Sin embargo, la Internet tuvo sus orígenes mucho antes, en la Red conocida como ARPANET[2].

En 1964 la RAND (Contracción de "Research and Development"), una organización no lucrativa dedicada a mejorar las políticas y toma de decisiones gubernamentales mediante investigaciones y análisis, inventó un nuevo tipo de red orientada hacia la solidez y tolerancia a fallas en el equipo. La propuesta escrita por Paul Baran se basó en la idea de que la nueva estructura de red no tendría ninguna autoridad central. Además, estaba diseñada para operar aún estando desmembrada. Para esto, todos los nodos serían iguales cualitativamente, cada uno podría enviar y recibir mensajes. Todos los mensajes se enviarían en paquetes, cada uno con su propia dirección. Estos paquetes podrían ser enviados a un nodo y llegar a otro. Lo innovador era que la forma en que los paquetes atravesaban la red no era importante. Eso significa que si un nodo era destruido, el resto de los nodos seguirían siendo capaces de comunicarse. Esto es ineficiente y lento, pero extremadamente confiable.

La Internet todavía usa este método hoy día, y sólo ha habido una suspensión de servicio colectiva a la fecha. Sin embargo, la estructura actual de Internet ya no está construida con la supervivencia a la destrucción de sus partes, ahora responde una comunicación más eficiente, por lo que es más frágil de lo que su idea original especifica.

La primera red de prueba construida sobre estos principios estuvo en el Laboratorio Nacional de Investigaciones (National Research Laboratory) en la Gran Bretaña en 1968. Poco tiempo después, ARPA, Agencia de Proyectos de Investigación Avanzados (*Advanced Research Projects Agency*) del Pentágono, quiso instalar una red más avanzada basada en los mismos principios en los Estados Unidos. La red consistió de cuatro computadoras de alto rendimiento.

En 1969, se instaló el primer nodo en la UCLA, Universidad de California en Los Angeles (*University of California at Los Angeles*). Para 1971 habia 23 nodos de ARPANET, que fue el nombre dado a esta red, después del de la UCLA, se establecieron nodos en el Instituto de Investigaciones de Stanford (*Stanford Research Institute*), en la universidad de Utah y en la UCSB Universidad de California, Berkeley (*University of California Santa Barbara*).

---

[2] Fuente: "The history of Internet and the WWW" por David Mayr
http://ourworld.compuserve.com/homepages/dmayr/history.htm

ARPANET se construyó porque el tiempo de procesamiento tenía un costo y un valor muy alto, además ARPANET ofreció a los investigadores la posibilidad de emplear a las computadoras de otras instituciones y compartir las propias con cómputo a distancia.

En Marzo de 1972 Ray Thomlinson de la BBN (Bolt Beranek and Newman, compañía de cómputo) inventó el primer programa de correo electrónico. Hoy día uno de los principales usos de la Internet es el intercambio de mensajes, y la forma de éstos no ha cambiado mucho desde estos primeros intentos.

Los primeros nodos internacionales se establecen en 1973. Estos estaban localizados en Escocia y en Noruega. El crecimiento de la ARPANET fue posible porque podía ser empleado por casi cualquier plataforma, gracias a que el código fue puesto a disposición gratuitamente y se incluía con el sistema operativo UNIX.

En 1974 Vint Van Cerf y Bob Kahn publican "A protocol for Packet Network Internetworking" (Un protocolo para Interconexión de Redes de Paquetes) en el que se especifica el diseño de un TCP (Protocolo de Transferencia de Paquetes).

En 1979 se establece la Usenet, basado en, UUCP Unix to Unix Copy (Copia de UNIX a UNIX) con fines de intercambio académico de información.

El protocolo TCP/IP se establece para ARPANET en 1982. Este protocolo se estandarizaría el 1 de enero de 1983 (reemplazando al NCP o Network Control Protocol, Protocolo de Control de Red). La palabra "Internet" comienza a utilizarse. Esta transición fue una de las más grandes pruebas a la capacidad de auto organización de la comunidad de Internet, y se llevó a cabo sin que se reportaran incidentes de consideración.

También en 1983 ARPANET se divide en ARPANET y el segmento militar MILNET. MILNET se integra con la red de datos de la defensa (Defence Data Network) creada el año anterior. El nuevo protocolo estándar creado el año anterior y esta división son factores importantes en la desmilitarización de la ARPANET. Este mismo año se desarrolla el servidor de nombres en la universidad de Wisconsin, el que se usa todavía y permite identificar a los nodos de la red por un nombre en vez de una secuencia de números.

En 1984 se rebasan los 1000 nodos de ARPANET y se adopta el servicio de nombres con el servicio llamado DNS Domain Name System (Sistema de Nombres de Dominio).

La NSF, National Science Foundation (Fundación Nacional de Ciencia) decide ligar cinco de sus centros de supercómputo (cómputo de alto rendimiento) para que pudiesen ser empleados en proyectos de investigación, pero la burocracia y falta de personal de ARPANET impide que se integren a esta red. De este modo se construye una nueva red basada en el protocolo IP de ARPANET, la NSF enlaza los cinco centros con líneas de 56 Kbps (56x1024 bits por segundo)

pero no establecen las ligas con las universidades por falta de recursos económicos para tender el cableado necesario. Las escuelas e instituciones se comunican en grupos regionales y comunican estos grupos con alguna de las supercomputadoras. La carga de información por estas líneas se incrementó paulatinamente y pronto las líneas fueron insuficientes para manejar la cantidad de datos enviados.

En 1987 la NSF firma un contrato con Merit Networks para incrementar la capacidad de la red. Desde ese entonces las líneas y servidores han sido continuamente mejorados por diversos proveedores. Se rebasan los 10,000 servidores. Para 1989 se rebasan los 100,000 servidores.

En 1990 ARPANET deja de existir, pero los usuarios difícilmente lo notan por mantenerse los servicios que ésta prestaba. De hecho, todavía se conservan.

3. Desarrollo de Internet en México: Orígenes de REDUNAM[3]

Aún cuando la Internet, y la propia REDUNAM son fenómenos que se han presentado en la presente década, la comunicación electrónica de datos tuvo sus comienzos desde la década de los 60 y 70, con la naciente infraestructura telefónica de la UNAM, la cual se empleó ampliamente para intercomunicar terminales de diversos tipos por medio de las líneas telefónicas.

En octubre de 1985, la UNAM e IBM de México subscribieron un convenio con el cual se puso en marcha un proyecto conjunto de investigación y desarrollo en el que se contemplaba: La instalación de una red universitaria de cómputo para apoyo a la docencia que permitiera el acceso remoto a los sistemas de procesamiento de datos actuales y futuros de las dependencias de la UNAM y la creación de un laboratorio para el diseño y la manufactura apoyado por computadoras. Dicho convenio fue firmado por el rector de la UNAM, el presidente y director general de IBM y por el secretario de Comercio y Fomento Industrial. A raíz del convenio se integraron dos grupos de especialistas de la UNAM e IBM con objeto de definir detalladamente el plan de trabajo y así cumplir con los objetivos propuestos.

El planteamiento original se adaptó a las cambiantes condiciones de las redes de computadoras y se alejó paulatinamente del esquema de redes cerrado de IBM hacia sistemas académicos más abiertos basados en software gratuito y estándares públicos.

En 1987 se establece la primera conexión a la red Académica Bitnet por medio de un enlace telefónico, desde la Ciudad Universitaria hasta el Instituto Tecnológico de Estudios Superiores de Monterrey (ITESM) y de ahí hasta San Antonio, Texas en los EUA.

[3] Fuente: Centro de Información de REDUNAM http://www.nic.unam.mx/REDUNAM/historia.html

Bitnet es la abreviatura de "*Because it's Time*" y fue una de las primeras redes de área amplia del mundo empleada primordialmente por universidades con fines académicos.

En un esfuerzo por consolidar su presencia en Bitnet, se utilizó la recién adquirida computadora IBM 4381 como residencia del correo electrónico y demás servicios de Bitnet. Y mediante terminales IBM con emulación 3270 por un enlace con la red TELEPAC de la SCT se logra dar una cobertura nacional, aunque limitada a este servicio.

No fue sino hasta 1989 cuando la UNAM a través del instituto de Astronomía establece un convenio de enlace con la red NSF de los E.U.A, el que empleaba al satélite Morelos II para enlazar al instituto de Astronomía de la UNAM con el UCAR-NCAR con residencia en Boulder Colorado.

Por primera vez en la Universidad se comunica usando fibra óptica a las redes locales del Instituto de Astronomía y la Dirección General de Servicios de Cómputo Académico. Siendo inaugurada oficialmente en ese mismo año la REDUNAM por el entonces rector José Sarukhán Kérmez.

En este periodo de tiempo, la UNAM comenzó una auténtica revolución en cuanto al manejo de sus recursos de cómputo y comunicaciones que todavía sigue transformando la forma en que los universitarios nos comunicamos entre nosotros. Se inició la adquisición masiva de computadoras personales, la construcción de centros de cómputo de alto rendimiento y la ampliación y el fortalecimiento de la red de comunicaciones de la universidad que estableció el enlace central de fibra óptica y la telefonía digital con que hoy contamos.

En 1990 la UNAM fue la primera institución en Latinoamérica que se incorpora a la red mundial [4] Internet, la que constituye uno de los fenómenos sociales más importantes del siglo originados en avances tecnológicos.

Los servicios de Telnet, FTP (*File Transfer Protocol*), correo electrónico y listas de correo en Internet comenzaron a operar en 1991. En 1992 se establecen los primeros servidores de *Gopher* en la universidad, lo que se complementa en 1993 con la creación de servicios electrónicos de Bases de Datos y revistas electrónicas, servicios hemerográficos y publicación de artículos íntegros en 1994. Servicio de traducción automática español-inglés, establecimiento de la red de videoconferencias e inicio de los servidores *World Wide Web* universitarios en 1995. Bases de datos de imágenes, servicios específicos *Web* y de bases de datos para las dependencias que los solicitan y almacenamiento masivo de información en 1996.

Transmisión de Radio UNAM por Internet, colaboración como sitio de consulta de los resultados preliminares en las elecciones del 6 de junio, integración de la red telefónica digital con Red

[4] Fuente: Departamento de Operación de la Red (DGSCA)

UNAM, comienzo de la operación de los laboratorios "Fundación UNAM" con una fuerte orientación hacia Internet y se comienza la migración de la red central de fibra óptica de tecnología *FDDI (Fiber Distributed Data Interface)* a ATM (*Asynchronous Transfer Mode*).

Actualmente la infraestructura de REDUNAM[5] transmite indistintamente voz y datos, mediante sistemas digitales basados en las más modernas normas internacionales. Las principales instalaciones de la Universidad están integradas a la Red. Esto significa que a nivel licenciatura, posgrado e investigación, alrededor del 90% de sus miembros se encuentran en instalaciones cubiertas por la Red, independientemente de su ubicación geográfica.

El sistema es descentralizado redundante y está integrado por 31 Nodos de Cómputo y Telecomunicaciones enlazados entre si, via fibra óptica. Asi mismo tiene una infraestructura instalada para más de 170 redes locales de cómputo. La Red enlaza a cerca de 9000 computadoras en la UNAM entre sí y alrededor de un millón de computadoras en el resto del mundo.

4. Internet2 en México.

La evolución de Internet en el mundo fué enfocada hacia su rendimiento y capacidades.

La iniciativa de internet2 propone una red de alto rendimiento inicialmente para Universidades y centros de investigación científica, con el objetivos de facilitar y fomentar la nueva generación de aplicaciones que demandan calidad de servicio *(QoS)*, ancho de banda, retardo mínimo (low latency) y garantía de recursos.

La UNAM a través de Dirección de Cómputo Académico, en conjunto con 7 de las principales Universidades del país, forma parte en la Corporación Universitaria para el desarrollo de Internet (CUDI), que pretende ser un órgano regulador para el desarrollo de Internet2 en México.

El 8 de abril de 1999 se oficializó la constitución de CUDI y se firmó un convenio de colaboración internacional[6] con la University Corporation for Advanced Internet Development (UCAID) el 20 de mayo del mismo año. Además se establecen fuertes lazos de colaboración con las organizaciones en el mundo enfocadas hacia el mismo propósito, tales como: Asian Pacific Advanced Network (APAN), Singapure Advanced Research and Education Network (SingAREN) y CANARIE entre otras.

Esta nueva red de alto rendimiento, aporta nuevos horizontes a las tecnologías ya implementadas y otras por implementar tales como: colaboratorios virtuales, control remoto de

[5] Fuente: Coordinación de Prospección Tecnológica (DGSCA)

[6] http://www.intrenet2.edu/international/html/cudi.html acuerdo de entendimiento para la colaboración México – E.U.A.

instrumentos, procesamiento masivo en tiempo real, tele-inmersión, vídeo en demanda, almacenamiento masivo y videoconferencia con estándares de calidad, entre otras.

Hoy en día la REDUNAM brinda conexiones a 33 dependencias universitarias fuera de la Ciudad Universitaria y a 71 organismos privados y de gobierno a lo largo y ancho del territorio nacional.

B. Los servicios de Internet

WAIS Wide Area Information Service (Servicio de Información de Area Amplia) y *Gopher* (contracción de Go-For y mascota de la universidad de Minnessota donde se desarrolló) se liberan, y pronto se convierten en dos de los servicios principales de Internet.

WWW, World Wide Web (Telaraña de amplitud mundial) uno de los principales servicios de hoy en día se libera en 1992 por la CERN "*Conseil Europeen pour la Recherche Nucleaire*" (Consejo europeo de investigaciones nucleares, llamado actualmente Laboratorio Europeo de Física de Partículas). Más adelante hablaremos un poco sobre el surgimiento de *Web*, creador e historia.

El número de servidores para estas fechas rebasa el 1,000,000. Para 1993, el primer programa lector de WWW (*Browser*) llamado *Mosaic*, se libera. La tasa de crecimiento de Internet se estima en 341% y aunque a un ritmo menor, continúa creciendo.

Con la popularización de la Internet y la nueva presencia comercial en ella, comienzan a hacerse evidentes algunas deficiencias en los protocolos básicos principalmente respecto a ciertos aspectos de seguridad y direccionamiento de servidores, en respuesta a estos problemas.

Los Directores del Area de IP de la IETF, Internet Engineering Task Force (Fuerza de Ingeniería de Internet) en su reunión del 25 de julio de 1994 [RFC 1752]. Recomiendan el desarrollo y adopción gradual de un nuevo protocolo, conocido como IPng, *Next Generation Internet Protocol* o IP versión 6, y se lanza su propuesta de estándar ese mismo año. Este evento marca el inicio de la así llamada Internet II (basada en una nueva infraestructura radicalmente más poderosa que la anterior) y la adopción gradual de los nuevos protocolos, en contraste con la adopción del TCP/IP el 1° de enero de 1983.

En 1995 se culmina la privatización de la infraestructura de red a cargo de la NSF planteado desde 1985. Los fondos recabados de la adquisición de derechos por compañías privadas se redistribuyen entre las redes regionales para adquirir enlaces de escala nacional y alto rendimiento con las ahora numerosas compañías que podían proveer ese servicio.

C. Arquitectura cliente/servidor

Muchos de los servicios de Internet están basados en la arquitectura cliente-servidor.

Esta arquitectura puede definirse claramente desde su punto de vista funcional. Partimos de dos elementos básicos: el cliente y el servidor; el cliente realiza peticiones, su contra parte, el servidor, está en espera de alguna de ellas, una vez realizada dicha petición, el servidor la resuelve y envía la solución al cliente.

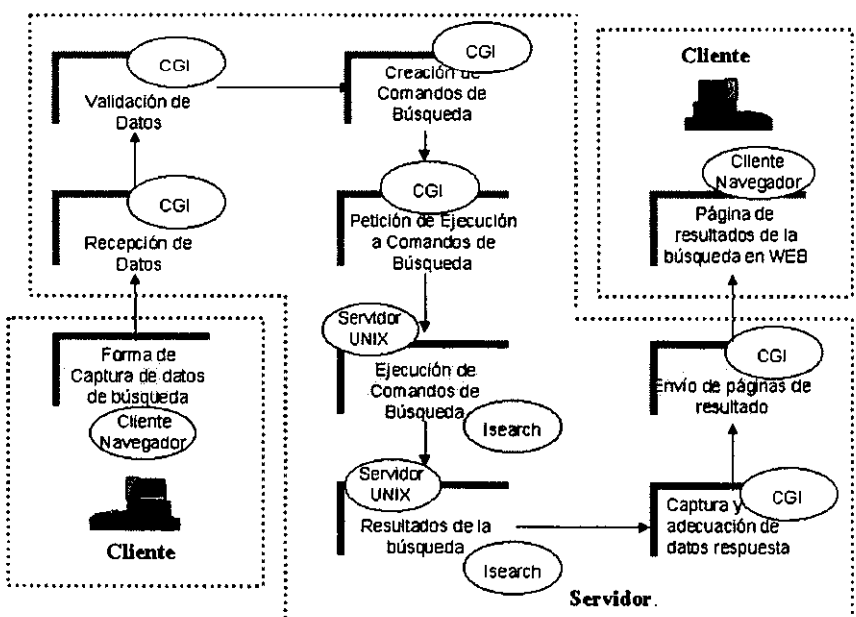
En el siguiente esquema podemos analizar con detalle el proceso a seguir en la arquitectura cliente – servidor del servicio de consulta resultante.

- El cliente es el encargado de interactuar con el usuario, por ejemplo en el llenado de una forma de *Web*, el usuario teclea su información, el cliente se encarga de captarla y desplegarla en la pantalla.

- El servidor es el responsable de responder a las peticiones del cliente, para el ejemplo anterior, los datos son recibidos por el servidor, el efectuará las operaciones que requieran como validación de datos, operaciones y obtendrá resultados.

- Una vez obtenidos los resultados por parte del servidor, el cliente los recibirá y mostrara al usuario por medio de una interfaz

- El cliente y el servidor pueden correr en la misma computadora, aun que la mayoría de las veces no es así, el servidor y el cliente se encuentran en una máquina por separado y su medio de comunicación es la conexión a red.



El servicio cliente acepta información del usuario, por lo general desde el teclado, *mouse*, o cualquier otro dispositivo de entrada, una vez que se tiene la información, el cliente se la envía al servidor, junto con las instrucciones de transformación (*query*). El servidor responde a la petición del cliente. Una vez recibida la información, el cliente, a través de un dispositivo de salida, entrega la respuesta al usuario; por lo general esta respuesta se da por un monitor o *display*.

La arquitectura cliente-servidor hace posible proveer los servicios de Internet a cualquier máquina que esté conectada a la red (configurando lo más básico).

La contra parte de este tipo de red es una red donde todos los servicios (cliente y servidor) corren desde cada máquina, y sólo algunos servicios como el correo se da a partir de un servidor.

1. Protocolo de comunicación entre el cliente y el servidor.

Cada cliente, como cada servidor, para poder comunicarse necesitarán hablar el mismo lenguaje, éste se establece y se le denomina protocolo, cada servicio tiene su propio protocolo, ejemplos de éste es el protocolo de *Web* es HTTP (protocolo de transferencia de *hipertexto*), el de *GOPHER* es *gopher*, para transferencia de datos se utiliza el FTP (protocolo de transferencia de datos).

El propósito básico del protocolo, ademas de controlar errores y comandos, acarrea las peticiones, del cliente, así como las respuestas o documentos que el servidor entregue. La comunicación se lleva a cabo a través de un puerto de comunicación asignado por el administrador del equipo.

En UNIX, el cliente al requerir de una comunicación con el servidor verifica el archivo `etc/services`, este archivo le indica cómo se debe comunicar con su respectivo servidor como por ejemplo :

```
# Network services, Internet style
#
tcpmux          1/tcp
ftp-data        20/tcp
ftp             21/tcp
gopher          23/tcp
```

La última línea indica que *gopher* se puede comunicar por el puerto 23 con el protocolo TCP.

El cliente de telnet se encarga de encapsular el mensaje, es decir, de transformar la petición en uno o más paquetes basados en el protocolo TCP (protocolo de control de transporte) los envía a través del puerto 23, los recibe *ined* quien acude al mismo archivo, verifica el destino del paquete y lo entrega al proceso servidor correspondiente.

El proceso servidor, recibe la serie de paquetes TCP, los desencapsula, interpreta la petición y al evaluarla envía la respuesta a la red, también en una configuración como la que recibió.

D. *World Wide Web*

1. Breve historia del nacimiento del *Word Wide Web.*

World Wide Web es uno de los más poderosos servicios de Internet, además de ser de los servicios más atractivos para el usuario por su interfaz tan amigable. De ahí que, algunos usuarios llegan a pensar que *Web* es Internet. Y ésto es comprensible si tomamos en cuenta que mediante el servicio de *Web* podemos acceder a muchos otros servicios de Internet como FTP, correo electrónico, *Gopher*, bases de datos, buscadores de *Web*, listas de discusión, etc. Aún por dichas caracteristicas, es un grave error confundir estos términos.

El sueño de *Web* inició sin lugar a dudas con la universalidad del espacio de información, la información hallada formaría parte de cualquier espacio del mundo y sería accedida mediante una red.

El autor del *Web*, Tim Berners Lee, con el sueño de la universalidad de información, además con otra idea central: "la organización de la información en la computadora no está permitiendo registro de asociaciones aleatorias entre dicha información"; pensó en desarrollar un "*Identificador universal de documentos*".

En 1990, Tim Berners Lee programó un editor de texto llamado "*WolrDwidEweb*" mientras trabajaba en un laboratorio Europeo de Partículas Físicas conocido como CERN, ubicado en Suecia frontera con Francia

Propuso un espacio universal de información, donde el usuario coloca una marca personal o una liga en la que consultará posteriormente de manera instantánea a dicha información. Así llegamos a la definición[7] de su mismo creador "*el Web es un universo global de información accedido por red*" y en éste espacio abstracto, el usuario podrá interactuar y comunicarse con otros usuarios.

[7] fuente: http://www.w3.org/People/Berners-Lee/1996/ppf.html

Este primer servidor de *WEB*. Consistió en el desarrollo de una utileria llamada wwwlib generada en 1992.

Posteriormente un estudiante Nicola Pellow, puso en "línea" un *Browser* que funcionaba en casi cualquier computadora, con algunos datos como UDI´s (ahora llamado URL localizador Universal de Recursos) lenguaje HTML y además utilizando el Protocolo de Transferencia de Hipertexto (HTTP).

Para cumplir con la segunda parte del sueño sería necesario que todos trabajarán un poco, construyendo hipertexto desde la casa o la oficina.

En los tres años siguientes surgieron los primeros Navegadores de Web como *Erwise*, *Viola*, *Cello* y eventualmente *Mosaic*.

En 1994, después de muchas discusiones entre la academia y la industria se decidió el formar el consorcio *World Wide Web* el mes de septiembre, conformado inicialmente por USA (MIT), Francia (INRIA) y Japón (Universidad de Keio).

El consorcio seria entonces un foro neutral donde se discutiria el futuro del *Web*, protocolos, estándares y más...

La cara de este servicio tendría como rasgos, ligas de hipertexto (caminos de acceso rápido a información complementaria) e imágenes; hoy en día contamos además con animaciones, audio y video mostrando mundos en tercera dimensión y videoconferencias.

2. Objetivos del *Web*

Word Wide Web tiene como objetivos básicos:

- Crear un lenguaje estándar, sencillo para transformar los archivos tipo *ASCII* y publicarlos vía Internet.

- Generar un mecanism innovador de organización de la información, que permita dentro del documento asociarle información complementaria mediante una marca, algo similar a en un libro colocar los separadores en un libro que me permitan llegar a partes específicas del libro en forma rápida.

- Proveer la información y asegurar que ésta puede ser consultada en todas partes del mundo

- Garantizar la diversidad y "libertad de consulta" de la información para cualquier usuario.

✓ Soluciones

Con la finalidad de crear un formato estándar para desplegar los documentos de información en Internet, se transformaron los documentos tradicionales tipo ASCII (texto plano) por documentos con marcas tipo "*tag*" <marcas>, a este tipo de codificación se le denominó HTML (*hipertext markup lenguage*) lenguaje de marcas de hipertexto. Cada parte del documento y cada característica especial en el documento se indica mediante una marca de texto o tag específico. Más adelante mencionaré los *tags* básicos para un documento en HTML.

"La información a publicar en *Web* por todo el mundo es realmente voluminosa, y entonces ahora será necesario tener una forma ingeniosa de organizarla", así se desarrolló un mecanismo que permitiera, que a través de un documento y específicamente de una liga, llegar a otros documentos relacionados con él y que a su vez estos documentos permitan regresar a los documentos iniciales y a otros. Esto dio como resultado una verdadera telaraña.

[imagen] un documento con ligas a otros

3. Visiones del *Word Wide Web*

a) Desde el punto de vista del lector

El *www.* consiste en un mundo de documentos y ligas. Estos documentos se encuentran indexados en una forma especial, tal que mediante un software de navegación (*browser* gráfico)

al seguir en camino de un texto ligado en mi documento (por lo general haciendo un click con el apuntador del *mouse*), puedo encontrar más información correspondiente a otro u otros documentos.

Las páginas de *Web* están desarrolladas en un leguaje de programación llamado HTML (HyperText Markup Lenguage) o lenguaje de marcas de hipertexto, el protocolo utilizado para la transferencia de este tipo de documentos es llamado HTTP (Hipertext Transfer Potocol) protocolo de transferencia de hipertexto.

b) Desde el punto de vista del proveedor de información

Los *browsers* del *www* mediante diferentes protocolos como es HTTP (FTP, transferencia de correo electrónico, etc.) y un *gateway* pueden acceder rápidamente a una masa crítica de datos por todo el mundo.

4. Impacto del *World Wide Web*

El impacto por la información en *Web*, es definitivo, ya que la interfaz de este sistema es amigable y muy atractiva, los proveedores de información toman en cuenta factores como:

Captar la atención del usuario con atractivos y dinámicos diseños.

Captar la atención del usuario con información de alto nivel de interés general.

Llevarle al usuario la información directamente a su hogar o trabajo, que en particular a él le interese.

Retener al usuario mediante información actualizada.

5. Aspectos técnicos del *WWW*

World Wide Web, *W.W.W*, *W3* o también conocido como *Web*, es uno de los servicios más gráficos y más poderosos de Internet, dió pauta a sus clientes (*browsers* gráficos) a aprovechar sus cualidades y adoptar a otros servicios de Internet como FTP, correo electrónico, etc.

Estas características han hecho del *Web*, el servicio de más rápido crecimiento (en todo sentido) en Internet.

World Wide Web es una frase con un significado espectacular, esta tecnología de comunicación facilita la conectividad global. Es un medio funcional para que los usuarios de todo el mundo localicen información, compartan conocimientos y además interactúen entre sí.

Web es un sistema de navegación para Internet, un sistema de administración y distribución de información con un formato dinámico para comunicación masiva y personal.

En *Web* podemos encontrar cualquier tipo de información, como por ejemplo educación, ciencia, política, noticias, deportes, diversión, música, etc. Todo gracias a la diversidad de pensamientos de la humanidad que aporta información para el *Web*.

6. ¿Dónde radica la importancia de *Web*?

¿Dónde radica la importancia de *Web*?, la respuesta es sencilla, la importancia radica en la diversidad, universalidad, y accesibilidad de la información. Este servicio dio a Internet una revolución de información muy importante, ya que:

Tenemos en *Web* sitios virtuales de los museos con las exhibiciones completas, mapas, descripciones, e incluso guías en varios idiomas, como el museo de Lourve en Paris, el Museo Nacional de Historia Natural de Estados Unidos http://www.mnh.si.edu/collections.html.

Podemos encontrar a todas las principales compañías periodísticas, llegando con sus noticias a todas partes del mundo.

Las transacciones comerciales se llevan a cabo por Internet, donde sólo selecciono lo que deseo adquirir, doy el número de identificación o tarjeta comercial bancaria y algunos días después llegará el producto hasta mi casa por alguna compañía de paquetería.

Los anuncios de empleo y propiamente los negocios pueden ser realizados en forma electrónica.

Las universidades en línea, informan todo lo relacionado con sus planes de estudio, instalaciones, conferencias, nuevas investigaciones, y hasta realizan sus inscripciones vía Internet.

En el sector público y particularmente en instituciones gubernamentales se aprovecha la infraestructura de red, para tener comunicación inmediata entre los diferentes estados del país.

Así podemos citar un sin número de nuevas tendencias nacientes a partir de la evolución tecnológica de Internet y específicamente del *Web*.

a) El Hipertexto

Al texto que nos permite asociarle más información le llamamos hipertexto, con los nuevos *browsers* gráficos al hipertexto que nos permite llegar automáticamente con otro documento le llamamos hiperliga o también se les conoce simplemente como ligas. Las ligas son texto subrayado con un color diferente al resto del texto, que con sólo dar click en ellas, nos llevan a información complementaria o concerniente al texto. Una vez visitada una liga, cambiará de color para resaltarlo.

El concepto de hipertexto es ahora muy utilizado, un ejemplo de ello es la ayuda en línea que ofrece Microsoft en sus nuevos productos, si colocas tu ratón sobre un icono de la barra de herramientas mostrará un recuadro con el nombre o descripción correspondiente al elemento en cuestión.

b) Concepto de URL

Cada documento tendrá asociado un URL (*Uniform Resource Locator*) localizador uniforme de recursos, o también conocido como "dirección *Web*".

Describe el protocolo utilizado, el nombre del servidor donde se encuentra alojado, la ruta y el nombre del archivo, por ejemplo

http://wwww.unam.mx/dependencias/info/avisos.html

ftp://sunsite.unam.mx/pub/lenguajes/Java/javafaq.html

Cada hiperliga deberá tener indicado el URL correspondiente al documento a ligar.

E. El *browser* y el servidor de *Web*

Los primeros servidores y *browsers* de *Web* fueron desarrollados a partir de la invención del *World Wide Web* en la CERN enfocados al protocolo cliente/servidor HTTP, dando lugar a una simple librería, denominada ***wwwlib***, en 1992 esta librería fue puesta al dominio público.

Posteriormente la comunidad de Internet con base a esta librería desarrollaron nuevas características hasta llegar a los *Browsers* y servidores de *Web* que hoy en día soportan casi cualquier arquitectura y sistema operativo.

Algo que es importante resaltar es que a partir de desarrollos simples y de la disposición al dominio público, se han podido hacer mejoras considerables dando como resultado herramientas de alto nivel de calidad, refinamiento y seguridad.

Uno de los desarrollos más importantes de los *Browsers* de *Web* es el realizado por NCSA (*National Center for Supercomputer Applications*) en la universidad de Illinois.

Led de Marc Anderssen, (ahora *Mosaic Comunications, Inc.*) son un grupo de desarrolladores de software. Ellos crearon uno de los principales servidores de *Web* conocido con el nombre de *Mosaic*, inicialmente para plataforma *UNIX*.

En el ambiente cliente/servidor el control del *Web* recae en el *Web browser.*

El *browser* de *Web*, o también llamado Navegador, realiza varias funciones; de las principales :

Recuperar del servidor de *Web* mediante el URL, el documento invocado por el usuario ya sea dado directamente en la caja de texto *Location:* o indirectamente indicado en una hiperliga.

Interpretar el documento escrito en código HTML

Mostrar al usuario el documento deseado con una interfaz amigable y gráfica casi ya por default.

El poner a disposición los documentos de Web, es tan fácil como "correr " el servidor de Web e indicarle la estructura de directorios donde se encontrará la información a publicar.

Los servidores de *Web* son los encargados de poner a disposición de los *Browsers* los documentos HTML mediante su protocolo de transporte llamado HTTP.

El *browser* de *Web*, utiliza el protocolo HTTP (*hipertext transfer protocol*) protocolo de transferencia de hipertexto para comunicarse con su respectivo servidor, pero además pueden comunicarse con servidores de otros tipos, como los servidores de FTP ó servidores de *gopher.*

Cada respuesta del servidor de *Web* hacia el *browser* es realizada mediante una nueva conexión. Es decir, cuando el *browser* hace la petición al servidor de mostrar un documento HTML mediante su URL establecen una conexión entre ellos, que posteriormente se cerrará una vez transferido el documento.

El protocolo de comunicación ha tenido *también* un desarrollo importante entre versiones, como de la versión HTTP / 0.9 a la versión HTTP / 1.0.

F. La codificación de páginas HTML

El código HTML consiste en marcas, etiquetas o *tags* agregadas dentro de un texto normal *ASCII*. Éstas son identificadas por el protocolo y por tanto por el *Browser* de *Web* y permiten el correcto despliegue de los documentos.

Cada parte del documento, así como sus características especiales tales como formatos de texto, indicaciones de saltos de línea, nuevos párrafos, tablas, listas numeradas, colores, hiperligas, etc., son definidos mediante una etiqueta.

Todas las etiquetas se delimitan por un juego de pico paréntesis <ETIQUETA>, la mayoría de los formatos se aplican por secciones, es decir si yo deseo aplicar algún formato especial a un bloque de texto, por ejemplo un bloque de texto color rosa, tendré un juego de etiquetas para marcar el inicio y el fin del formato: <etiqueta_inicio> bloque de texto </etiqueta_fin>.

G. Haciendo formas Interactivas en *Web*

Una de las características más útiles en la evolución del protocolo de *Web* es la introducción de medios interactivos con el usuario, como son las formas de *Web*.

Las formas permiten a cada usuario introducir información en una página *Web* desde su *Browser* y así lograr múltiples objetivos como:

- **Camptura de datos.** Con la finalidad de alimentar una base de datos.

- **Comunicación en Intrenets.** Mediante el uso de una forma interactiva en web , un usuario o un grupo de usuarios, por ejemplo una empresa, podrán a través de una intranet tener sus sistemas corporativos utilizando como interfaz gráfica formularios en páginas *Web* vistos por un navegador. Llenar un formulario de registro, proporcionar datos personales de alguna índole en particular (como datos personales, opiniones ó comentarios, etc.)

- **Captura de encuestas**. Muchos de los sitios de Internet proporcionan al usuario cuestionarios de registro u opinión con el fines estadísticos o de cartera de clientes, esto es gracias a los formularios en *Web*.

- **Realizar comercio electrónico.** Las formas interactivas en *Web* permitieron también la realización del comercio electrónico que consiste en realizar una petición de algún producto llenando una forma con los datos del artículo y datos personales como nombre del comprador, número de tarjeta bancaria y forma en que prefiero se realice el envío.

- **Publicar algún tipo de información.** Otros posibles proyectos a realizar principalmente basados en la interacción del usuario es la publicación de material educativo, proporcionando un medio que permita captar la información del investigador, la valide y la publique directamente.

H. CGI

1. Programación de CGI

En el tema anterior de "Elaboración de formas para *Web*" mencionamos la importancia del CGI, pero ¿qué es el CGI?.

CGI (*Common Gateway Interface*) es una interfaz común de intercambio de información, el CGI es un programa que permite al servidor de *Web* (*Web* server) comunicarse con otros programas y demás recursos del sistema (*host*).

El CGI es un programa que reside en un directorio determinado en la configuración del servidor de *Web*. Cada CGI es diseñado para realizar una tarea especifica dentro de un sistema, y en realidad es lo que hace la diferencia entre varios sistemas de *Web*.

Cada sistema en *Web* puede tener el mismo servidor de *Web*, e incluso la misma base de datos, pero las actividades o funciones que realice se determinan al programar cada CGI.

El programa CGI es en realidad el alma del sistema y es reflejado por el siguiente ciclo de vida:

El CGI desde una forma de *Web* es invocado por un URL como podría ser: http://www.unam.mx/cgi-bin/saludos.pl

Al hacer el llamado, el servidor de *Web* ya configurado reconocerá el tipo de programa invocado como un CGI y le entregará los datos que el usuario le envía mediante la forma de *Web*.

El CGI recibe los datos, les da validación y el formato adecuado para ser entregados a algún, o algunos otros programas (incluso otros programas CGI) que casi siempre son llamados y despedidos por el mismo CGI.

Posteriormente recibe la respuesta a los datos, los acondiciona y nuevamente comunicándose con el servidor de *Web*, se los entrega al usuario.

En el esquema de bases de datos, el CGI mediante algunos programas adicionales (utilerías) hace las peticiones de operación como consultas, introducción y corrección de datos. Recibe las respuestas y las adecua para mostrarlas al usuario en *Web*.

En cada sistema de bases de datos el CGI determina mediante su programación qué sentencias deben enviarse al manejador de bases de datos, llámese ORACLE, Sysbase, MSQL, etc.

Dependiendo de la operación que el usuario elija, el CGI se encargará de formularla, de pedir los datos de respuesta al mismo manejador, y finalmente adecuará la respuesta para un despliegue atractivo al usuario a través de *Web*.

I. Los sistemas de consulta a catálogos en *Web*

Web ha dado a los sistemas de catalogación un giro muy importante ya que ahora el poder consultarlos a nivel global y con un nuevo esquema de búsqueda le da un valor agregado a la información que contiene el catálogo.

El sistema de búsqueda en catálogos publicados en *Web* pretende ayudar al usuario a encontrar la información que necesita de forma sencilla y utilizando un medio totalmente visual.

La información que encontramos en un catálogo por lo general está organizada en registros y cada registro tiene un número variable de campos.

Por ejemplo en el caso de un catálogo bibliográfico, seguramente tendrá un número grande de registros dependiendo del tamaño de la biblioteca, y cada registro tendrá la información por campos, por ejemplo el campo título del libro, el autor, la localización, un resumen, el número de IBSN, comentarios, etc es posible que en algunos registros existan campos sin información o que existan dos registros muy parecidos porque en la biblioteca tengan varios ejemplares de un libro y solo con algunos campos que los distinguen entre si.

Este esquema de catalogación es muy flexible, ya que puede ser aplicado a muchos otros rubros como revistas, noticias de periódicos, videotecas, descripción y venta de productos, etc.

La información es concentrada en un archivo de texto

En una forma de *Web* el usuario escribe la información que desea buscar en el catálogo, es posible que el usuario recuerde que su información está asociada con algún campo en especial y recomiende hacer la búsqueda por algún campo en especia

Capítulo II **Análisis de la problemática del proyecto y alternativas de solución**

Capítulo II **Análisis de la Problemática del Proyecto y Alternativas de Solución.**

A. Análisis de la problemática a solucionar.

1. Necesidades.

La información generada en el país como resultado de las actividades diarias en distintas áreas como educación, investigación, tecnología, industria, comercio, entretenimiento, etc., nos muestra una marcada necesidad de organizar dicha información, además darle una correcta difusión. Con el advenimiento de las nuevas tecnologías de Red, estas necesidades se han visto ampliamente beneficiadas, ya que cada vez es más fácil llegar hasta el público que lo demanda.

En el caso específico de la Universidad Nacional Autónoma de México, el volumen de acervos publicados, es masivo, tanto bibliográficos como hemerográficos y videográficos nos llevan a la gran labor de catalogación.

Otra necesidad no menos importante es el aspecto de la difusión, ya que estos acervos deben llegar a cada rincón de nuestro país y además darse a conocer en todo el mundo.

Una alternativa ya implementada para solucionar esta necesidad es **el Servicio de Publicación de Catálogos a través de Internet**, ya que proporciona un medio atractivo y totalmente masivo para mostrar amplios volúmenes de información a través del mundo.

La Dirección General de Servicios de Cómputo Académico de la UNAM, como responsable directa de proporcionar nuevas alternativas de solución tecnológica basadas en el cómputo, proporciona a través de su Coordinación de Servicios de Red (CSR) entre otros servicios, el de publicación de catálogos en Internet, además de proporcionarle al usuario de Internet una herramienta sencilla que le facilite a encontrar la información que necesita de manera rápida y efectiva. A este servicio que proporcionan se le llama "Servicio de búsqueda en Catálogos publicados en *Web*".

Recursos utilizados para este sistema:

Para proporcionar este servicio se cuenta con recursos tanto de infraestructura como de personal académico especialmente capacitado.

En el esquema de bases de datos, el CGI mediante algunos programas adicionales (utilerías) hace las peticiones de operación como consultas, introducción y corrección de datos. Recibe las respuestas y las adecua para mostrarlas al usuario en *Web*.

En cada sistema de bases de datos el CGI determina mediante su programación qué sentencias deben enviarse al manejador de bases de datos, llámese ORACLE, Sysbase, MSQL, etc.

Dependiendo de la operación que el usuario elija, el CGI se encargará de formularla, de pedir los datos de respuesta al mismo manejador, y finalmente adecuará la respuesta para un despliegue atractivo al usuario a través de *Web*.

I. Los sistemas de consulta a catálogos en *Web*

Web ha dado a los sistemas de catalogación un giro muy importante ya que ahora el poder consultarlos a nivel global y con un nuevo esquema de búsqueda le da un valor agregado a la información que contiene el catálogo.

El sistema de búsqueda en catálogos publicados en *Web* pretende ayudar al usuario a encontrar la información que necesita de forma sencilla y utilizando un medio totalmente visual.

La información que encontramos en un catálogo por lo general está organizada en registros y cada registro tiene un número variable de campos.

Por ejemplo en el caso de un catálogo bibliográfico, seguramente tendrá un número grande de registros dependiendo del tamaño de la biblioteca, y cada registro tendrá la información por campos, por ejemplo el campo título del libro, el autor, la localización, un resumen, el número de IBSN, comentarios, etc es posible que en algunos registros existan campos sin información o que existan dos registros muy parecidos porque en la biblioteca tengan varios ejemplares de un libro y solo con algunos campos que los distinguen entre si.

Este esquema de catalogación es muy flexible, ya que puede ser aplicado a muchos otros rubros como revistas, noticias de periódicos, videotecas, descripción y venta de productos, etc.

La información es concentrada en un archivo de texto

En una forma de *Web* el usuario escribe la información que desea buscar en el catálogo, es posible que el usuario recuerde que su información está asociada con algún campo en especial y recomiende hacer la búsqueda por algún campo en especia

2. Planteamiento de la problemática.

El servicio de búsqueda en catálogos publicados en *Web*, es uno de los servicios con más demanda últimamente, como resultado de una actualización tecnológica entre las diferentes dependencias de la UNAM y en general en el sector público y privado. Proporcionando una necesidad apremiante, ya que se tiene la respuesta a las necesidades de catalogación y difusión ahora.

Por tales motivos se nos presenta una problemática bien definida, se tiene una sobre demanda del **"Servicio de búsqueda en Catálogos publicados en *Web*", además de, la necesidad de mantener y crear nuevos servicios con los mismos recursos *materiales y humanos*. *Ante esta problemática nos inclina hacia la solución de automatización de este servicio con el fin de generarlo en forma eficiente y a la brevedad posible y con mínimos recursos.***

La automatización de este servicio nos da grandes ventajas, entre las más importantes, la rápida respuesta a las peticiones de éste servicio, así como el ahorro sustancial de recursos económicos y de personal.

Una vez implementada la solución ante la sobre demanda del servicio, se pretende ver reflejado el beneficio directamente en una reducción sustancial del tiempo de respuesta ante la petición de un nuevo servicio, que el desarrollo del servicio no requiera de personal altamente calificado, y de ser posible implementar un mecanismo que permita al usuario (dueño de los datos) no depender de la CSR para las actualizaciones y mantenimiento de sus datos.

El proceso para poner en línea que consume este servicio, es aproximadamente de cuatro a cinco semanas, asignando a un líder de proyecto y a un becario de apoyo.

Los recursos requeridos para este servicio son variados:

En hardware:

- Infraestructura de red (red local con conexión a Inernet permanentemente).
- Servidor UNIX con un espacio (cuota de cuenta de usuario) para almacenamiento superior al tamaño del catálogo del servicio.
- Un servidor UNIX para desarrollo y diseño.

En software:

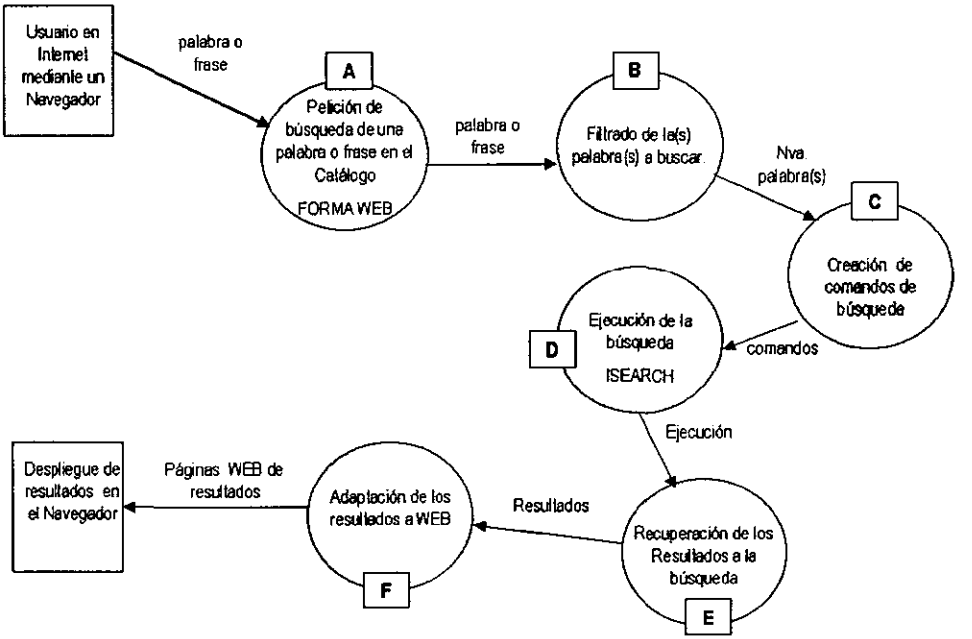
- Ambiente de desarrollo: Sistema Operativo UNIX (Solaris, Irix, Linux)
- Servidor httpd (servidor de *Web*)
- Manejador para datos.
- Lenguajes de programación para CGI en UNIX.

En personal:

- Coordinador: Coordinador para efectuar el contrato y recepción del cliente.
- Líder de proyecto: ingeniero en cómputo, informático o afín.
- Desarrollador: Programador experimentado en servicios de Internet (becario).
- Diseñador: personal de diseño especializados en sitios de *Web*.
- Administrador: Administrador del servidor UNIX, administrador del servidor de *Web*.

Además de las instalaciones correspondientes para mantener las actividades diarias.

•DFD Funcionamiento del sistema a través de la consulta de un usuario.



B. Análisis y diseño de las alternativas de solución.

1. Características del Servicio a Automatizar.

Cada sistema de búsqueda, tendrá características muy particulares, que dependerán del tipo de información y del giro de la empresa que proporcione el servicio (comercio, educación, gobierno,etc.)

El proceso tradicional de implementación de cada servicio de búsqueda, fue desarrollado con anterioridad a este trabajo de automatización, es el resultado de la realizar un exhaustivo análisis de las herramientas, procedimientos y alternativas óptimas para este tipo de servicios.

En general, el servicio de búsqueda de palabras en catálogos publicados en *Web*, funciona bajo la filosofía cliente servidor (ampliamente explicada en el capítulo 1). El funcionamiento esperado de cada servicio es muy similar, lo podemos visualizar en el siguiente esquema:

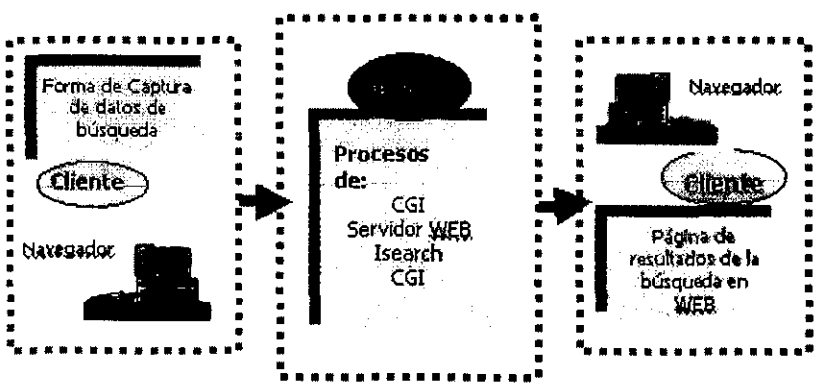
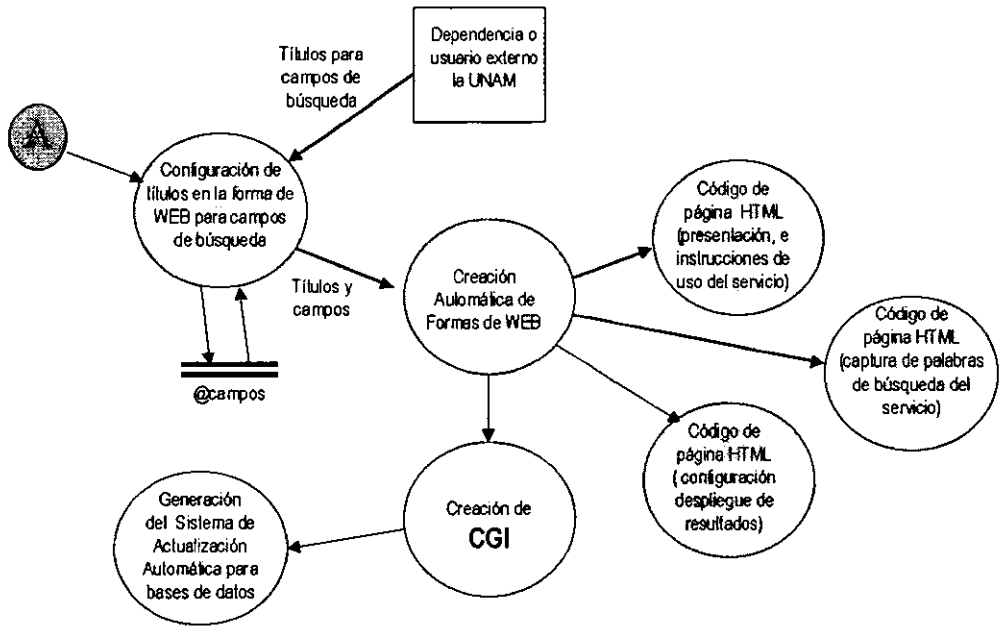


Diagrama del funcionamiento de un sistema de búsqueda en un catálogo por Internet.

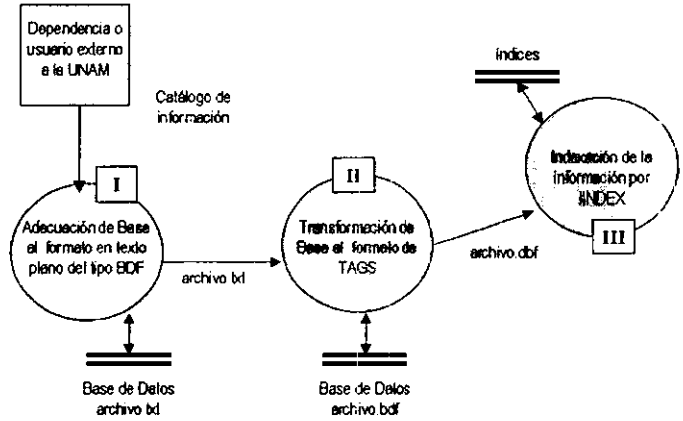
Descripción del funcionamiento:

El diagrama de flujo de Datos (***DFD) funcionamiento del sistema a través de una consulta de usuario.***, nos muestra el funcionamiento de cada uno de los servicios de búsqueda, en él, se describen los procesos que se efectúan en cada consulta que hace un usuario.

- **DFD Desarrollo de un Sistema de Búsqueda para catálogos publicados en Web** (continuación)



- **DFD Adecuación del Catálogo de Datos para cada Servicio.**



Así mismo, el diagrama ***DFD Desarrollo de un Sistema de Búsqueda para catálogos publicados en Web y el diagrama DFD Adecuación del catálogo de datos para cada servicio***, nos representa a grandes rasgos el procedimiento que efectuaría de la manera tradicional, el grupo de trabajo asignado para desarrollar cada uno de los sistemas de búsqueda que son pedidos a la Dirección de Cómputo Académico.

Cliente Interfaz-Usuario-Web.

En el lado del cliente, tenemos el Navegador de *Web* que está instalado en la computadora de cada usuario que accede a Internet

En el URL definido por el servicio podemos visualizar la interfaz del sistema de búsqueda. A esta interfaz la componen los siguientes elementos :

- Información de la empresa que proporciona el servicio.
- Instrucciones de uso del servicio de búsqueda
- Cajas de introducción de texto. En estas cajas de texto se colocarán las palabras a buscar en el catálogo de datos.
- Una lista de selección de campos. En cada catálogo la información se divide en registros y a su vez cada registro se divide en un número variable de campos. Las búsquedas de palabras se efectúan a partir de uno o varios campos.
- Botones de configuración del tipo de búsqueda requerida. Cada búsqueda de palabras se puede realizar de un modo estricto (tipo *and*) o de un modo más libre a la incidencia (tipo *or*)
- Datos para contactar al responsable del servicio en el caso de dudas más especificas.

Procesos Servidor-UNIX.

- **CGI:** este programa es invocado mediante la forma de WEB y se encuentra activo a la espera de una petición de búsqueda por parte del usuario, su primera labor es recibir e identificar los datos que el usuario le proporcione, verificar la integridad de ellos, así como su validez.

  Una vez identificados los datos deberá con ellos crear los comandos correspondientes a cada tipo de búsqueda.

Los comandos serán invocados directamente en el CGI y serán recibidas por el sistema operativo.

- **Servidor Web:** El servidor WEB es el responsable de la interfaz del usuario. Es el encargado de desplegar la forma donde el usuario configura los datos de la búsqueda y de desplegar los resultados que le envia el CGI como resultado de la operación.

- **Isearch :** Es el motor encargado de realizar las búsquedas a través de todo el documento. Una vez invocado, realiza un "análisis sintáctico" de la información y envía las incidencias en modo monitor.

CGI: Nuevamente es la interfaz entre el sistema y WEB, captura los resultados emitidos por el motor de búsqueda, los depura y adecúa para ser desplegados por WEB.

Mediante una interfaz gráfica en Internet, un usuario en cualquier parte del mundo podrá acceder a la información de un catálogo, ayudado por un sistema que permite realizar búsquedas sencillas de una o más palabras ligadas a campos específicos correspondientes a los registros de la base de datos.

La interfaz utilizada para el usuario en este tipo de sistemas es gráfica e interactiva, por medio de formas de WEB, en ellas además de contener la herramienta de búsqueda, se muestran características del catálogo que se publica tales como: ¿A quién le pertenece?, ¿Cuáles fueron las últimas fechas de actualización de la base de datos? ¿Cómo utilizar efectivamente la herramienta de búsqueda?.

El sistema de búsquedas está controlado por una serie de programas CGI, que se encargan de recibir los datos y la información del tipo de búsqueda a realizar, por ejemplo el campo donde se requiere la incidencia a la información buscada, cuántos registros se desea visualizar.

El proceso de búsqueda y recuperación de datos del catálogo está basado en el protocolo z.3950 Isearch, software de tipo gratuito *del Center for Networked Information Discovery & Retrieval* (CNRI); éste, a partir de las peticiones del cliente efectúa un barrido de la información por medio de índices que le permite localizar rápidamente las incidencias. Además también es el encargado de entregar los resultados para ser desplegados al usuario nuevamente en forma gráfica.

2. Formas de automatización para el servicio.

Una vez realizado el Análisis general de las actividades que se tienen que efectuar para cada servicio de búsqueda en específico (con ejemplos desarrollados anteriormente o supuestos,

definimos actividades de tipo estándar generalizadas para cualquier caso), que permitan el cubrir todas las necesidades de cada sistema particular.

Actividades a realizar para cada Sistema de búsqueda y publicación de catálogos:

- Contacto con el cliente respondiendo a un proceso de solicitud formal del Servicio búsqueda en catálogos publicados en Internet.

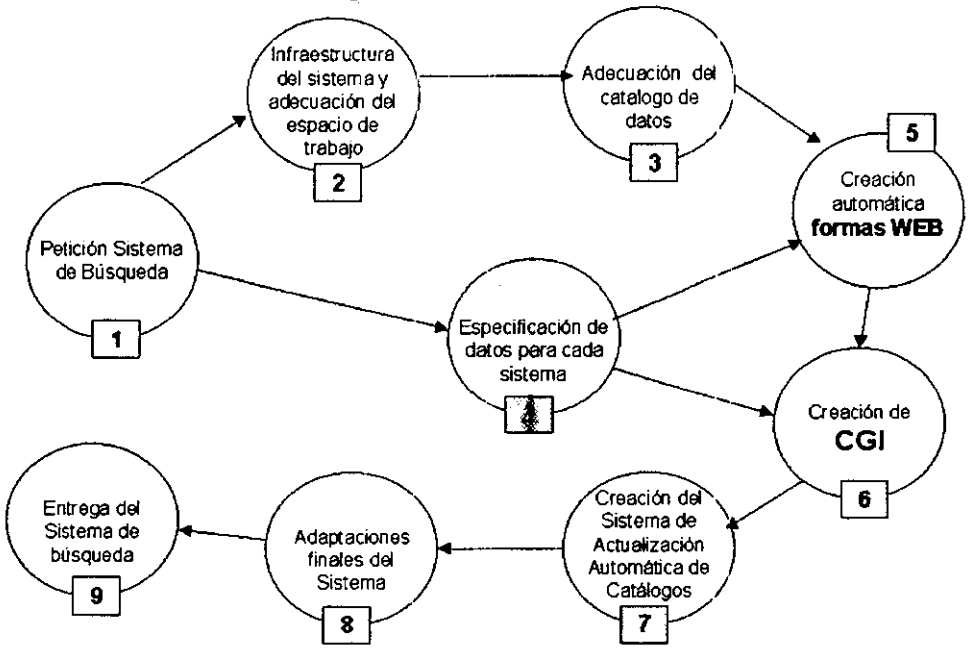
- Recopilar la información de la instancia que lo solicita, entre otras cosas, identidad, responsable de la información publicada, preferencias de diseño en el servicio y propiamente el catálogo a publicar contenido ya en una base de datos del tipo BDF (DBASE) o en un formato estándar para bases de datos.

- Hacer la petición de los espacios de trabajo, la cuenta final en la que se alojará el sistema resultante y solicitud de una cuenta de desarrollo.

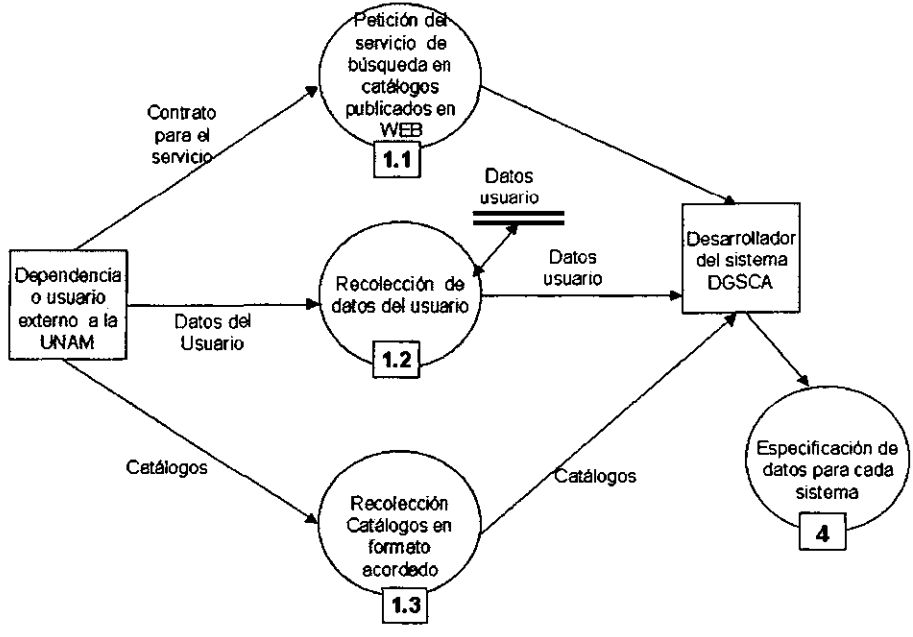
- Adecuar los datos mediante un programa de conversión al protocolo z3950.

Diseño y creación de la forma principal HTML. Esta forma de consultas deberá contener la información acordada por el usuario además de un mecanismo de selección de los campos donde se requerirá la incidencia de palabras y le permita al usuario poder elegir entre una búsqueda rígida y una optativa.

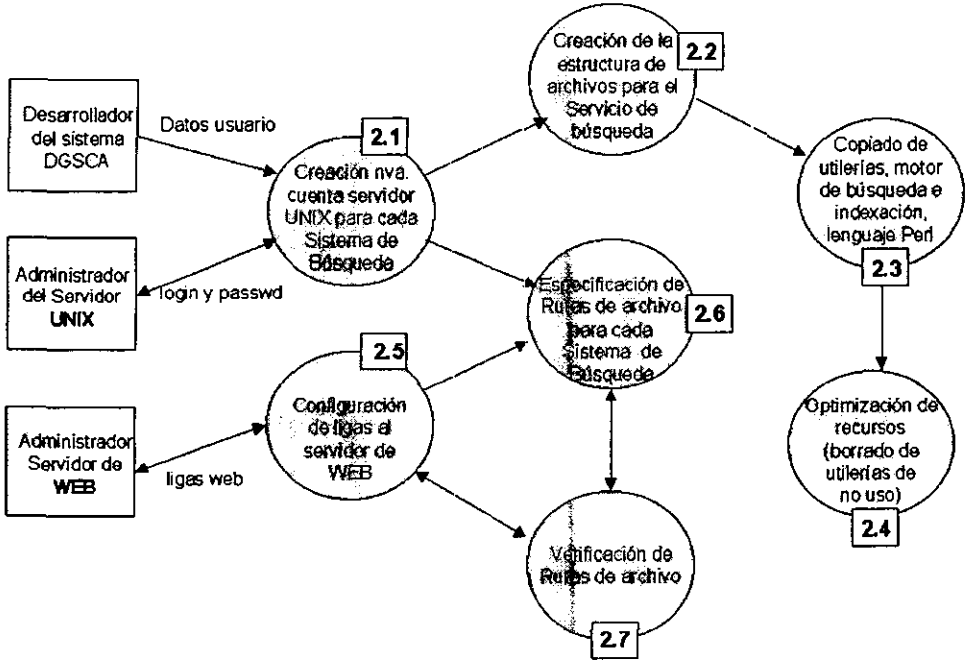
- Proceso de Generación Automática de Servicios de Búsqueda de Palabras en Catálogos Publicados en WEB.



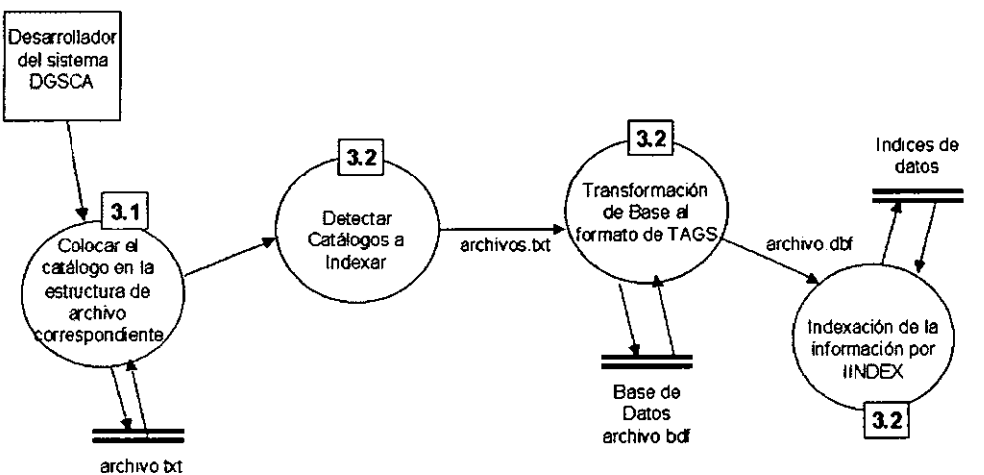
• **Proceso** 1 **Petición del Sistema de Búsqueda .**



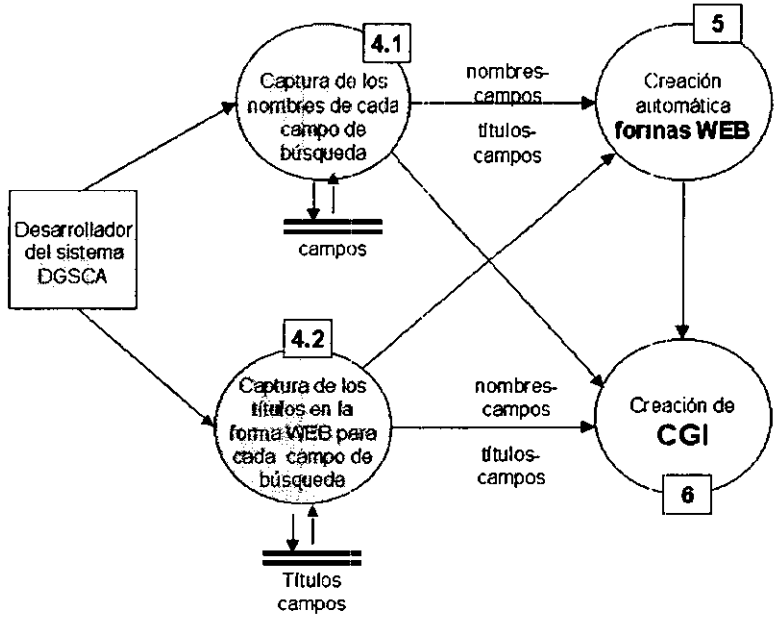
- **Proceso** 2 Infraestructura del sistema y adecuación del espacio de trabajo.



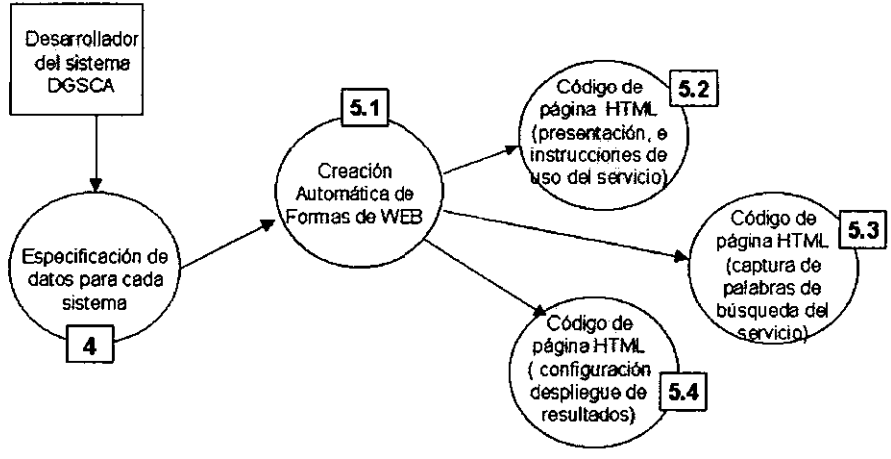
- **Proceso** 3 Adecuación de los datos para cada Servicio.



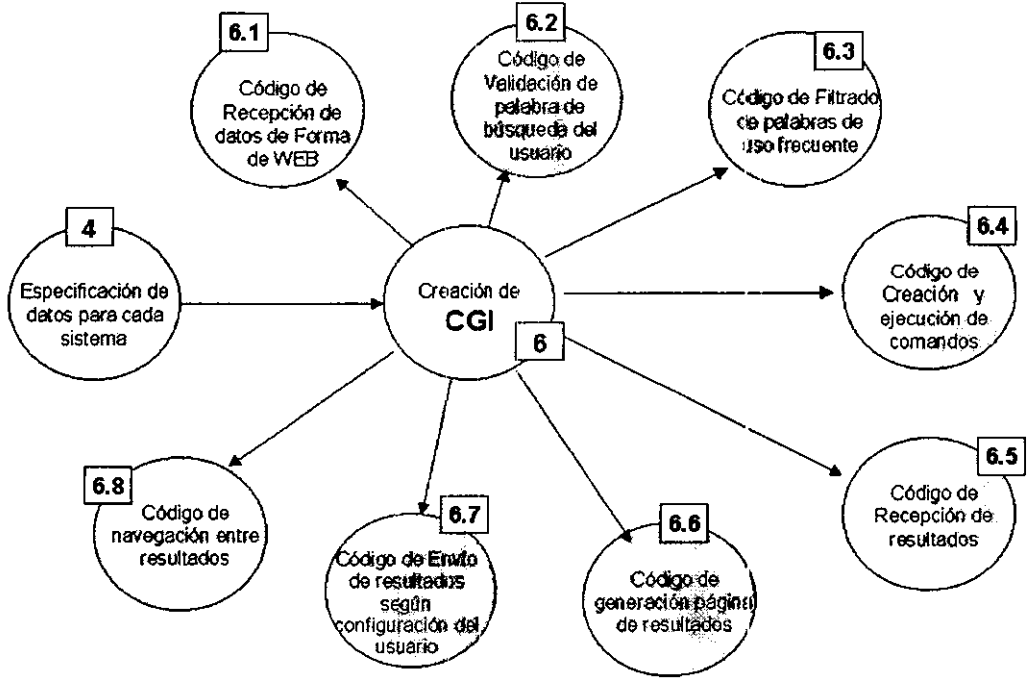
- **Proceso** 4 Especificación de los datos en cada Sistema de Búsqueda .



• **Proceso** 5 Creación Automática de páginas y formas de WEB.

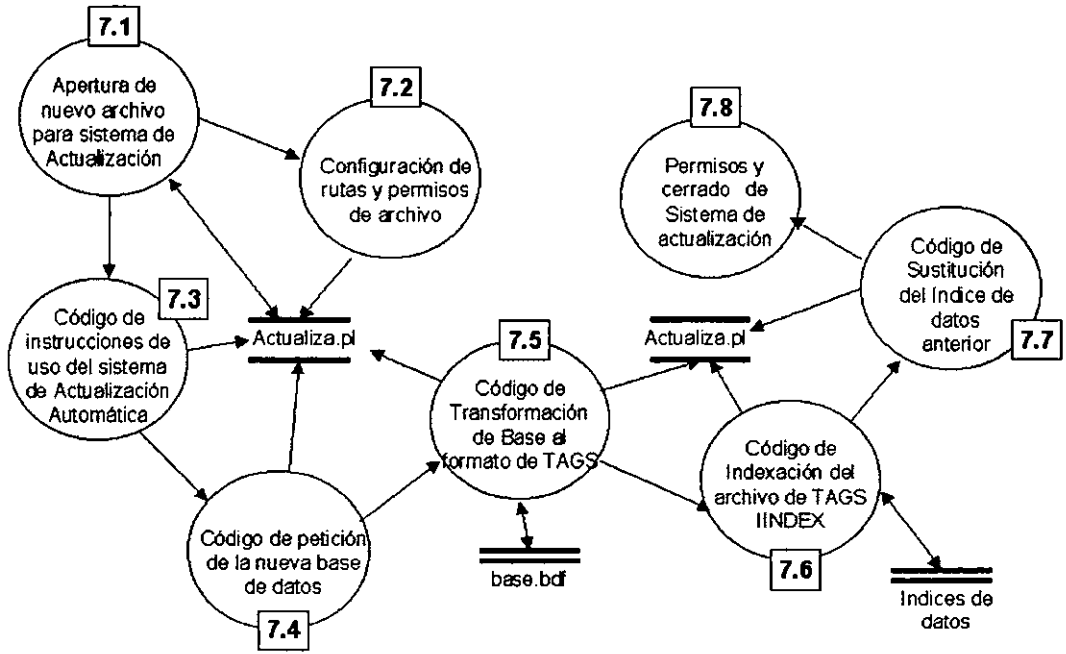


• **Proceso** 6 Creación de CGI.

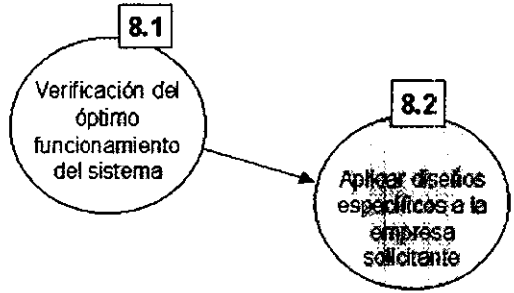


- **Proceso** 7 Generación de código del

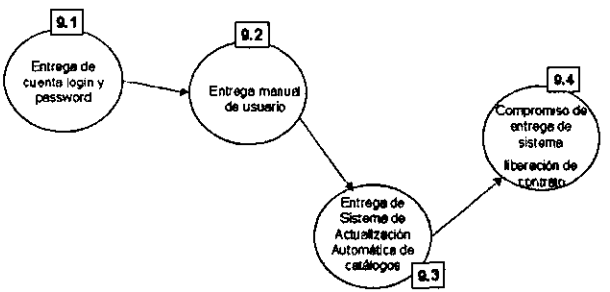
"Sistema de Actualización Automática de bases de Datos para Usuarios".



- **Proceso** 8 Adaptaciones finales del Sistema.



• **Proceso** 9 **Entrega del Sistema de búsqueda en Catálogos para WEB .**



C. Conclusiones de la fase de análisis y diseño.

La metodología utilizada en este proyecto nos llevó claramente a diseñar un sistema simple con necesidades bien definidas, es un paso clave en el desarrollo del proyecto, ya que como resultado tenemos consideradas todas las actividades a programar o integrar para el sistema final.

Finalmente concluimos que mediante el análisis del funcionamiento particular del producto esperado, podemos bosquejar fácilmente la generalidad de este tipo de sistemas de búsqueda y al mismo tiempo modelamos mediante procesos todo lo que de antemano necesitamos ofrezca el *sistema global de generación automática Genérico.*

En la siguiente fase describiremos el desarrollo de nuestro sistema partiendo de los recursos generados en esta fase del proyecto

.

Capítulo III **Software libre**

Capítulo III "Software Libre"

A. Uso de software de libre distribución

En el análisis previo para la implementación tradicional del servicio de búsqueda y el de este nuevo procedimiento automatización, nos mostró los grandes beneficios que otorgaba el uso de los desarrollos de "software libre", en lugar del software comercial.

La alternativa de desarrollar o utilizar implementaciones de software libre, nos lleva a un sin número de ventajas que trataré de mostrar en este capítulo.

1. Definición de software de Libre distribución (*Open Source*)

La venta del software es uno de los negocios más importantes del cómputo en el mundo, pero no perder de vista que no es el propósito fundamental que se tiene para desarrollarlo.

Tradicionalmente el software tiene el propósito fundamental de interactuar entre el usuario y la máquina, proporcionándole un ambiente sencillo de utilizar los recursos del equipo, aunque la actualidad marca una necesidad imperante de la existencia del hardware con su respectivo software, resultando inútil uno sin el otro. Resultando el Software, uno y el mismo que el cómputo.

Con la revolución de la información provocada con el surgimiento de Internet, el desarrollo de software también se vió muy favorecido. Internet proporcionó el medio adecuado de comunicación entre desarrolladores de software en todo el mundo, vinculando a las universidades del mundo en donde se realizan la mayor parte de los proyectos tecnológicos.

El surgimiento del software libre nace con la Internet misma, ya que para que todo el mundo pudiera comunicarse entre sí, fue necesario distribuir el software apropiado. Además ya que la filosofía de Internet es por definición "abierta" y cooperativa el desarrollo del software también adquirió estas caracteristicas.

De los desarrollos más sobresalientes tenemos al sistema operativo UNIX que a través de la red fue robusteciéndose mucho más, los servidores *Web*, los lenguajes de programación como Perl, etc.

Todos estos desarrollos de software pertenecen al software libre bajo distintas modalidades, pero definamos a que se le considera software libre.

Es considerado un software como libre si cumple con los siguientes puntos.

a) Libre Redistribución

La licencia no debe restringir a nadie vender o entregar el software como un componente de una distribución de software que contenga programas de distintas fuentes. La licencia no debe requerir "*royalty*[8]" ni ningún tipo de cuota por su venta.

b) Código Fuente

El programa debe incluir el código fuente y se debe permitir su distribución tanto de código fuente como código compilado.

Cuando de algún modo no se distribuya el código fuente junto con el producto, deberá proveerse un medio conocido para obtener el código fuente sin cargo a través de Internet. El código fuente es la forma preferida en la cual un programador modificará el programa.

Este tipo de licencia no permite el código fuente deliberadamente confundido. Tampoco se permiten formatos intermedios, como la salida de un pre-procesador, o de un traductor.

c) Trabajos Derivados

La licencia debe permitir modificaciones y trabajos derivados, y debe permitir que éstos se distribuyan bajo las mismas condiciones de la licencia del software original.

d) Integridad del Código Fuente del Autor.

La licencia puede restringir la distribución de código fuente modificado sólo si se permite la distribución de "*patch files*[9]" con el código fuente, esto con el propósito de modificar el programa en tiempo de construcción.

La licencia debe permitir explícitamente la distribución de software construido en base a un código fuente modificado.

La licencia puede requerir que los trabajos derivados lleven un nombre o número de versión distintos a los del software original.

[8] Concesión de derechos de uso, distribución, etc.
[9] Archivos parches. Archivos de robustecimiento o actualización de errores.

e) No discriminar personas o grupos.

La licencia no debe hacer discriminación de personas o grupos de personas.

f) No discriminar campos de aplicación.

La licencia no debe restringir el uso del programa en un campo específico de aplicación. Por ejemplo, no puede restringir su uso en negocios, o en investigación genética.

g) Distribución de la Licencia.

Los derechos concedidos deben ser aplicados a todas las personas a quienes se redistribuya el programa, sin necesidad de obtener una licencia adicional.

h) La Licencia No Debe Ser Específica a un Producto.

Los derechos aplicados a un programa no deben depender de la distribución particular de software de la que forma parte. Si el programa es extraído de esa distribución y usado o distribuido dentro de las condiciones de la licencia del programa, todas las personas a las que el programa se redistribuya deben tener los mismos derechos que los concedidos en conjunción con la distribución original de software.

i) La licencia no debe afectar otro software.

La licencia no debe imponer restricciones sobre otro software que sea distribuido junto con él.

Por ejemplo, la licencia no debe insistir en que todos los demás programas distribuidos en el mismo medio deben ser software *Open Source*. De esta manera, los desarrollos que se incluyan con un software de este tipo podrán tener el tipo de licencia que su creador decida conveniente.

2. Variaciones de Software libre

a) Software Libre

Este tipo de software tiene permisos de uso, copia, modificación y distribución de cualquier persona. Una característica fundamental es que tiene disponible su código fuente, si no cumple con esta restricción, no califica como software libre. Respecto al costo de él, por lo general el término "*free*" (libre) tiene la implicación de cero costo, pero esto, se debe aclarar en los términos de la licencia de cada producto.

El software libre a diferencia del *freeware* tiene una distribución binaria y al mismo tiempo de su correspondiente código fuente.

b) Software del Dominio Público

Este tipo de software es aquel que no tiene registro de derechos de autor, es decir no cuenta con registro de copyright "si un software está en el dominio público, no pertenece a nadie". Este tipo de software no tiene restricciones en cuanto a su uso o distribución, modificación o copias.

No existe ninguna protección para el autor (programador), cualquier persona puede copiar su producto, modificarlo y posteriormente venderlo sin distribuir el código fuente. La licencia de dominio público es la que da más libertad al usuario, y es la que menos garantías o seguridad da a sus usuarios, además pierde en gran parte el objetivo inicial de *Open Source*.

c) Software Propietario.

En este tipo de software no tiene libertad absoluta de su uso "semi-libre", copia o distribución, para tales efectos se deberá llegar a una negociación con el autor.

Podemos identificar dos categorías importantes de este en shareware y freeware.

- ***Freeware*:**

  Esta es una categoría muy utilizada en licencias de software, aun que no es del todo clara en su definición en el sentido que solo se refiere a la libertad de costo (sin costo) pero no así, a su disponibilidad de código fuente.

  Freeware permite uso, copia y redistribución, pero no permite la modificación de él. Una característica importante es que no tiene disponible su código fuente, solo se distribuye el binario listo para ser instalado en la computadora. Un ejemplo de este tipo de software es el Internet Explorer que a estas fechas se distribuyen su binario sin costo alguno, pero no su código fuente.

- ***Shareware*:**

  Los programas de *Shareware* se distribuyen libremente, pero para su uso se deberá aportar una donación o pago al autor del

programa en cuestión después de un período de tiempo determinado para prueba, habilitándole posteriormente todas las potencialidades del software. En la distribución del shareware se provee el binario listo para su instalación y casi nunca estará acompañado de su código fuente.

Un ejemplo claro de este tipo de software es aquel que podemos encontrar para funcionar por un período de tiempo, en forma de ``demostración", o con solo parte de sus capacidades, cuando el usuario pague por la licencia, el software funcionará de manera óptima.

d) Software Comercial:

Este software se distribuye en forma binaria, es decir listo para ejecutarse en la computadora.

El código fuente prácticamente es un secreto en el archivo ejecutable, esto con el objetivo de que no sea fácil de modificar. La licencia prohibe explícitamente la ingeniería inversa o modificación alguna.

Su característica primordial es que los propietarios de este software cobran por el uso de éste, actualizaciones y a veces hasta por soporte técnico. Este es el modelo que puede ser visto como el ``clásico", si pensamos que el software como un producto de venta. No hay libertad de compartir, ni de modificar y muchas veces ni de ejecutar el software en circunstancias especificas.

Ejemplos de Licencias.

Las licencias GNU GPL, BSD, X *Consortium*, y Artistic son ejemplos de licencias que consideramos que cumplen con la definición de *Open Source*. También la licencia MPL cumple con la definición.

La siguiente tabla, ilustra las diferencias y semejanzas entre los más importantes esquemas de licenciamiento de software.

Característica de la licencia vs. Tipo de software	Precio nulo	Redistribución	Uso Ilimitado	Código fuente disponible	Código fuente modificable	Todos los derivados free
Open Source (Linux/GNU style)	✓	✓	✓	✓	✓	✓
Open Source (Apache Style)	✓	✓	✓	✓	✓	✗
Open Source (BSD-Style)	✓	✓	✓	✓	✓	✗
Freeware Software Binario	✓	✓	✓	✗	✗	✗
Librerías freeware	✓	✓	✓	✓	✗	✗
Shareware (varia según la licencia de uso)	✓	✓	✓	✗	✗	✗
Software no usado en términos comercial (Non-Commercial)	✓**depende del uso**	✓	✗	✗	✗	✗
Software de periodo de prueba (Trial Software)	✓**versión limitada**	✓	✗	✗	✗	✗
Software Comercial licencia en Inglés	✗	✗	✗	✗	✗	✗

3. Soporte de Software.

El software de uso comercial en ocasiones justifica su precio bajo la promesa de que no estará solo, que se tendrá soporte y asistencia en caso de problemas, además que la marca lo respalda. La realidad es que casi en todos los casos la asistencia técnica se reduce a instrucciones telefónicas en el mejor de los casos y además no sabemos ¿cuánto más? Nos costará y si nos podrán atender personal y rápidamente, se agudiza más el problema si se trata de un software comercial que no cuenta con oficinas en nuestro país.

Una visión o mito muy común, creado en contra del software libre " bájalo de la red y estás solo contra el mundo...". Analizando la situación del software se concluye que no es del todo cierto éste mito, ya que gracias al interés global y la accesibilidad de desarrollar y perfeccionar el software, encontraremos bastante ayuda.

La ayuda la podemos traducir a listas de correo, foros de consulta y discusión casi siempre dedicados especificamente a algún desarrollo particular, casi en cualquier universidad se encuentra un servidor de FTP con toda la documentación, librerías, utilerías y demás aportaciones.

Un ejemplo de este tipo de apoyo al software libre en México, es la lista de discución de usuarios en Web de Linux, de Perl, de Postgresql, de Java, y otras entre las más importantes.

Estas listas de correo son lo más cercano a soporte inmediato y personalizado. En ellas, podemos plantear las experiencias u obstáculos que surgen al utilizar o incluso ampliar el software, en cuestión de algunos minutos gente muy dedicada al tema planteará la respuesta o una solución alternativa.

Algo muy importante que debemos resaltar, es que, el apoyo proporcionado para el software libre no tiene costo alguno, ni requisito de ninguna índole, solo será necesario encontrar la información.

Adicionalmente, existen pequeñas y no tan pequeñas empresas que proporcionan servicios de asesoría y asistencia especializada en algunos productos de software libre.

4. Ventajas y desventajas del software-libre

Una de las garantías que el software libre tiene sobre los productos comerciales es sin duda, el hecho que los desarrollos son enriquecidos y puestos a prueba por millones de usuarios, dando garantía de estandarización, seguridad y además estabilidad.

Las prácticas monopólicas de software nos han generado una dependencia clara a las actualizaciones forzosas, y continuamente se tiene que actualizar el hardware ya se convierte en insuficiente he inestable en respuesta de un software de baja calidad.

En el enfoque del software libre lo importante es enriquecer los desarrollos y ahí procurar la seguridad y robustecimiento, quedando en segundo plano el generar nuevos productos "remunerables".

Un ejemplo muy claro de esto es LINUX, el sistema operativo UNIX para computadoras personales, Linux es un sistema operativo de amplia confiabilidad, seguridad, con multiproceso y multiusuario. Se puede usar como una estación de trabajo UNIX o como servidor para tareas que abarcan desde servidor de WEB altamente eficiente hasta estaciones de trabajo de bajo costo para redes cliente/servidor.

La lista de características de Linux es extensa. Ofrece un ambiente estable de multitarea y multithreading en todas las plataformas que soporta, soporte para procesamiento simétrico (SMP), y controladores para el hardware más diverso. Sin embargo, los factores que aseguran el éxito a largo plazo de Linux tienen poco que ver con la lista de características, sino más bien con su licencia. Linux es el resultado de un desarrollo mundial a través de Internet al igual que muchos otros ejemplos de las mismas dimensiones como:

Perl, Apache, GNU Emacs, GCC y los programas GNU, Netscape, Java y Solaris más recientemente.

Estabilidad real entre aplicaciones

- La importancia del uso del software de libre distribución para países en vías de desarrollo.

En los países en vías de desarrollo como lo es México, se requieren de muchos esfuerzos para lograr alcanzar a los países exportadores de tecnología, esta tecnología que es traducida en ingresos económicos y en beneficios en general para el país, de ahí la importancia de alcanzarla. Uno de los principales obstáculos que tenemos es el alto costo de la tecnología y las severas crisis económicas que sufre el país.

La tecnología de la información es sin duda uno de los pilares en el desarrollo económico, ya que todos sus sectores dependen de mecanismos que faciliten el análisis, creación y movimiento de datos con el fin de toma de decisiones correctas y oportunas.

La globalización económica nos obliga a la actualización tecnológica, con el fin de tener un papel favorable para el país.

Internet ha dado pauta a gran revolución informática y ha puesto al alcance de todo el mundo información de primer nivel para todos los ámbitos: educación, medicina, industria, ingeniería, física, teorías económicas, etc.

El campo de la computación sin duda, él más cambiante y beneficiado, ya que los nuevos desarrollos para la información son muy abundantes, uno de los desarrollos más notables y que más contribuyen por sus características ya antes definidas es sin duda el software de libre distribución. El software de libre distribución por sus características ya antes mencionadas, derrumba la barrera económica, proporciona las herramientas y la oportunidad a personas con la preparación adecuada y sobre todo con talento (que existen en todo el mundo) de desarrollar la tecnología de su propio país. Un ejemplo claro de esto es el desarrollo tecnológico en nuestra máxima casa de estudios.

Este tipo de soporte tiene varias cualidades que merecen ser destacadas

Participación activa del Desarrollador: Autodidacta y dispuesto a buscar la información adecuada.

¿Qué podría ser considerada como una desventaja al utilizar el software libre? El costo real de implementación de este tipo de software es la inversión inicial necesaria para la capacitación del personal que se encargará de ocupar y desarrollar este software, ya que no es "evidente" ni mucho menos enfocado a usuarios promedio. Si será necesario el tener conocimientos previos de cómputo.

B. Protocolo z.3950

1. Definición.

Z.3950 es un estándar que define a un protocolo cliente/servidor para la recuperación de datos.

Este estándar especifica los procedimientos y las estructuras para que el cliente realice una búsqueda y recupere la información en una base de datos proporcionada por el servidor.

Este estándar también definirá el control de acceso, el control de reenvío de datos, y algunos otros servicios.

El protocolo trata la comunicación entre las aplicaciones responsables de la recuperación de datos, el cliente y el servidor, pero no trata la interacción entre el cliente y el usuario final.

Isearch es un motor de búsqueda basado en el estándar z.3950. Isearch es un software cuya función es Indexar y buscar en documentos de texto, permitiendo las búsquedas en todo tipo de bases organizadas por campos,

Isearch es un motor de búsqueda en texto desarrollado por CNIDR Center for Networked Information Discovery and Retrieval (CNIDR) fundado inicialmente por la National Science Foundation (NSF), este motor de búsqueda es capaz de realizar incidencias de palabras de texto en grandes volúmenes de datos.

2. Isite

Es una integración de software que incluye a Isearch plus además de su indexador de texto Index, Iutil utilerías para eliminar fácilmente Indices ya no operables y una interfaz CGI.

Isearch soporta búsquedas por campos y subcampos, por rangos relevantes, también realiza búsquedas en algunos tipos especiales de archivos como lo son los archivos HTML, SGML, listas de discusión, y puede extenderse el tipo de archivos que soporta mediante la creación de clases en C++ que definan la estructura del documento, pare esto deberá de obtenerse el código fuente y adecuarlo a las necesidades propias.

Isearch es un software no comercial, es desarrollado por un exitoso grupo de desarrolladores donde su primera versión es atribuida a Nassib Nassar en 1994 mientras trabajaba para CNIDR.

Algunas de las ventajas inmediatas que podemos considerar son:

3. Costo y disponibilidad.

Libre distribución sin costo alguno, además podemos encontrarlo incluso con su código fuente. Las diferentes versiones de Isearch así como Isite podemos encontrarla en

ftp://ftp.cnidr.org/pub/software/Isite/

ftp://ftp.cnidr.org/pub/software/Isite/

http://www.etymon.com/pub/software/Isearch/

Además de estar en casi todos los Sunsites.

Podemos encontrar el software en dos formas, binaria (ya compilada para plataformas determinadas) listos para instalar o en código fuente, que será necesario tener algunas utilerías

adicionales para compilarlo dependiendo del sistema operativo que se tenga y asegurarnos de instalarle los patches desarrollados para su mejora y seguridad.

Es muy modular y ajustable a las necesidades propias de cada lenguaje, ya que en este motor de búsqueda podemos encontrar incluso palabras con caracteres especiales al lenguaje Ingles, por ejemplo palabras acentuadas, o con la letra ñ,... etc.

4. Soporte técnico.

Es muy significativa la comunidad que utiliza Isearch en el mundo, dando como resultado algunas derivaciones comerciales basadas en este motor, dando como consecuencia natural gran apoyo comunidad usuaria, tanto desarrolladores como usuarios en general. El apoyo es manifestado en listas de discusión y sitios WEB donde podemos encontrar manuales, tutoriales, y casi cualquier tipo de ayuda, incluso por parte de especialistas. Por ejemplo la lista de correo:

5. Portabilidad.

Su portabilidad, podemos compilarlo casi de manera transparente para cualquier tipo de plataforma UNIX, como puede ser: Solaris y SunOS, Irix, HP/UX, AIX, para las diferentes versiones Linux y últimamente hasta puede ser compilado en Windows 95/NT mediante C++.

6. Eficiente uso de recursos.

Uso eficiente de los recursos del disco: Los índices para las referencias a cada palabra en la base de datos son relativamente compactos, generalmente más pequeña que la colección original.

7. Mantenimiento.

Mantenimiento de la base de datos: los viejos documentos pueden ser borrados automáticamente y actualizar la colección nueva sin necesidad de nuevamente clasificar la colección entera

8. Aspectos Legales.

C. El uso de Perl para la realización de CGI.

1. Perl.

Perl es un lenguaje de programación. El lenguaje escogido por la mayoría de los programadores de CGI, y parte indispensable de la administración de casi todos los sitios de Internet, Perl ha sido descrito como "la cinta adhesiva de Internet". Originalmente creado por Larry Wall, Perl ahora es mantenido por un grupo de cientos de programadores distribuidos por la red, que se comunican por una lista de correo. Larry mantiene un "control artístico" sobre el lenguaje, pero la mayoría del desarrollo es hecho por otros. Existen mecanismos de extensiones que permiten a la gente hacer modificaciones libremente.

2. Licencia.

Al igual que Linux, fue desarrollado basado en el trabajo de una persona distribuido bajo una licencia modificada de material artístico de uso gratuito sin restricciones lo que atrajo la colaboración de una gran cantidad de desarrolladores. Actualmente sigue en desarrollo pero los cambios mayores se dan en las librarías dedicadas a simplificar todo tipo de problemas y no en el lenguaje o módulos principales de este. La licencia sigue siendo de código abierto y uso gratuito sin restricciones.

3. Conclusiones

Perl es el lenguaje para desarrollo de sistemas *Web* que ha sido mas ampliamente probado y cuenta con una gran comunidad de usuarios, lo que lo hace una de las alternativas más seguras para el desarrollo de sistemas de Internet bajo plataforma UNIX.

Cuenta con un robusto manejo de cadenas, experiones regulares y de estructuras complejas de datos lo que lo hace una alternativa natural para los sistemas que manejan la información no como un conjunto de operaciones a aplicar sobre los datos sino como una estructura de parámetros que determinan las operaciones y los datos a utilizar en los procedimientos que definen al programa.

D. Servidor de *Web* Apache.

1. Apache.

Apache es un proyecto de desarrollo colaborativo. Apache es el servidor de *Web* para sistemas UNIX mas utilizado en Internet desde 1996 hasta la fecha (1999) varias estadísticas de estados

unidos nos indican que el servidor apache cubre el 53% de los servidores en Internet en todo el mundo. [http://www.apache.org/] Apache es un desarrolló bajo la filosofía de *open-source* del Grupo Apache.

Retomando el servidor NCSA httpd 1.3 e integrando todos los *patches* publicados para sus correspondientes *bugs*, nació Apache como una derivación de *PATCHE*. Actualmente grupo apache acaba de liberar la versión 1.3.4 incluyendo esta versión una para win32 bits (no tan estable como la versión de UNIX)

2. Características

a) Configuración.

La configuración de apache es muy simple, básicamente tiene un archivo donde debo indicar los parámetros como el puerto donde recibe los datos (por default es el puerto 80) la ruta del directorio base de archivos HTML, del directorio cgi-bin, así como el directorio donde el servidor alojará los mensajes de error y bitácoras.

Adicionalmente Apache puede proporcionar seguridad mediante la configuración de un reten a un nivel de archivos, donde solo podrá ingresar aquellos que cuenten con una clave y contaseña de usuario.

b) Bitácoras.

Reporte de errores, accesos, estadistico por medio de bitácoras, autenticación de usuario. El grupo Apache con la colaboración de grupos de usuarios por todo el mundo, están en constante mejoría del servidor de WEB.

c) Licencia de uso.

Apache tiene una licencia especial de *open source* en la que cuenta con todas las características correspondientes al código fuente disponible, libre distribución, sin costo, sin restricciones de uso, y además con la característica adicional de que sus derivados no tienen que ser libres.

E. Navegador de Internet Netscape.

1. Características.

Los directivos de la empresa Netscape en 1998, al darse cuenta que perdían mercado ante Microsoft Explorer en el mercado de los Navegadores, ya que Microsoft regalaba su navegador

(más no el código fuente), e inclusive pretendía incluirlo por defecto en su nuevo sistema operativo Windows 98 decidieron liberar el código fuente de Netscape y cambiar la licencia a una no-restrictiva del tipo Open Source. Esta decisión se tomó después de que consultaron a Eric Raymond, el autor the "*The Cathedral and the Bazaar*" y analizaron los comentarios de los entusiastas de software libre en diversos foros de discusión de Internet. , donde la comunidad de software libre sugería una apertura del código para poder lograr una mejoría en las calidades de los Navegadores de esta compañía.

Netscape creó una licencia parecida a la GNU/GPL pero que adecua más a las condiciones que deseaban conservar para su software, ésta licencia se propone inclusive para otros proyectos que no sean de Netscape. Esta licencia se conoce como licencia NPL (*Netscape Public License*). La decisión de Netscape dio mucha publicidad al software libre, y ha generado mucha discusión. Hay gente que la apoya y hay gente que cree que la fragmentación destruirá a los productos de la empresa. Netscape como empresa gana la mayor parte de sus entradas en el mercado de servidores de WEB (en los que compite con productos de software libre como Apache y con productos comerciales como los de Microsoft) y en el mercado de valor agregado en Internet, compitiendo con los "WEB Portals" (Altavista, Yahoo, Lycos) e inclusive con Microsoft.

Capítulo IV **Sistema de generación automática de servicios de búsqueda de palabras en catálogos publicados en Web, GENÉRICO**

Capítulo IV **Sistema de Generación Automática de Servicios de Búsqueda de Palabras en Catálogos Publicados en WEB (Genérico). "**

"Mediante el análisis del funcionamiento particular del producto esperado, resultado del sistema automático, podemos bosquejar fácilmente la generalidad de este tipo de sistemas de búsqueda y al mismo tiempo modelamos mediante procesos todo lo que de antemano necesitamos ofrezca el sistema global de generación automática Genérico...". Esta fue la conclusión a la que llegamos en la fase de análisis.

Partiendo de la fase de Análisis y Diseño, emprendemos la fase de desarrollo. En esta fase del proyecto se detallan los procesos diseñados en la etapa anterior y comentaremos las características adicionales de implementación que nos llevaron al sistema esperado.

El desarrollo se efectúa partiendo de los esquemas de diseño, de cada proceso esperado, cada sección de esquema diseñado corresponde a una clara necesidad y por supuesto a una implementación de código. Se considera importante incluir la sección de código ya que pretendo resaltar la simplicidad de programación que nos otorga Perl y así demostrar los beneficios que presenta en general al combinarlo con UNIX.

A. Desarrollo del sistema Genérico.

1. Petición del Sistema de búsqueda.

El proceso de generación automática inicia con la petición **(proceso 1.1)** de una dependencia de la UNAM o una empresa externa. La petición consiste en publicar un catálogo de información, por ejemplo un catálogo de productos en venta con sus características principales como por ejemplo: funcionalidad, presentación, precio, disponibilidad, marca, etc. Además se pretende publicar en WEB con el apoyo de una herramienta que le ayude al cliente a identificar el producto que busca, ya sea por su funcionalidad, por su color, o por la marca.

La petición es cotizada, tomando en cuenta el tamaño requerido para almacenamiento, y las características especiales que el cliente prefiera, en diseño, URL, etc.

Una vez aceptada la petición, el sistema es asignado al líder de proyecto que se encargará de coordinar la generación del servicio.

El encargado de recolectar los datos del usuario **(proceso 1.2)** obtendrá la siguiente información:

- **¿Quién es el responsable de la información?** Una parte muy importante, ya que la información que se publica en los servidores de la UNAM deberá ser fidedigna, de carácter serio y no lucrativa. Si se trata de una solicitud externa, también se efectúa una evaluación de que la empresa sea de carácter serio.

- La información sobre ¿qué es lo que se desea como producto final, en términos de diseño, logos, colores, etc.?

Catálogo de datos **(proceso 1.3)**, en el formato especificado:

Un archivo de texto con un número variable de registros, cada uno de esos registros con un numero no fijo de campos, cada uno de los campos tendrá una etiqueta o label que permitirá realizar búsquedas específicas por cada campo.

La integridad y el correcto formato del catálogo es responsabilidad del usuario, ya que se pretende independizar al usuario de Cómputo Académico, basando las actualizaciones de información en sólo los datos de usuario. Se le proporcionará apoyo para generar el primer archivo de datos.

2. Infraestructura del sistema y adecuación del espacio de trabajo.

El siguiente paso es la creación de una nueva cuenta en un servidor **(proceso 2.1)** que ya éste en producción, que cuente con un sistema operativo UNIX ya sea Solaris, Irix o Linux, asegurando el funcionamiento óptimo del servidor *Web* y una versión de PERL5.0. Este proceso lo efectúa el administrador del sistema, ya que se requiere cuenta de *root* para crear nuevas cuentas en servidor UNIX.

La cuenta se creará en dragon.dgsca.unam.mx para el caso de dependencias de la UNAM, o en otro servidor de producción según el tipo de petición. Si el usuario ya cuenta con un espacio se deberá asegurar además de los requerimientos de infraestructura de tener las facilidades de red para acceder remotamente.

Una vez establecido el espacio de trabajo, en el directorio *home* de la nueva cuenta, se genera la estructura de archivos **(proceso 2.2)** necesaria como el directorio html-docs base para servir los archivos HTML, el directorio cgi-bin para ejecutar los *scrips* (programas CGI) relacionados con el funcionamiento del sitio, el directorio para la el catálogo de datos y las utilerías.

Instalación o copiado (si se cuenta con la versión adecuada al sistema operativo) de utilerías, motor de búsqueda e indexación y de Perl5.0 **(proceso 2.3)**.

Optimización de recursos, se refiere al borrado de las utilerías que ya existieran en el servidor, esto con la finalidad de no duplicar las herramientas utilizadas.

Esto puede ocurrir si copiamos en un gran bloque comprimido las herramientas de una sola vez (que es muy recomendable).

Configuración de ligas al servidor de WEB **(proceso 2.5)** es un proceso que efectúa el administrador del equipo. Cada equipo tendrá instalado un cliente de WEB, encargado de proporcionar el servicio de WEB, en este caso es utilizado el Servidor Apache, se levanta uno o más demonios httpd que se encargarán de atender a las peticiones en WEB.

El servidor de WEB cuenta con un directorio central para servir las páginas html por lo general se llama "html-docs" o "htdocs", el demonio de web **httpd** escucha las peticiones en el puerto configurado (8080 por lo general) y busca las páginas en el directorio indicado.

A partir de este directorio central, se accede a los directorios respectivos en cada cuenta de usuario, se deberá colocar ligas suaves a los respectivos directorios html-docs y al cgi-bin con los permisos apropiados.

Comando: ln -s <ruta> <nombre_liga>

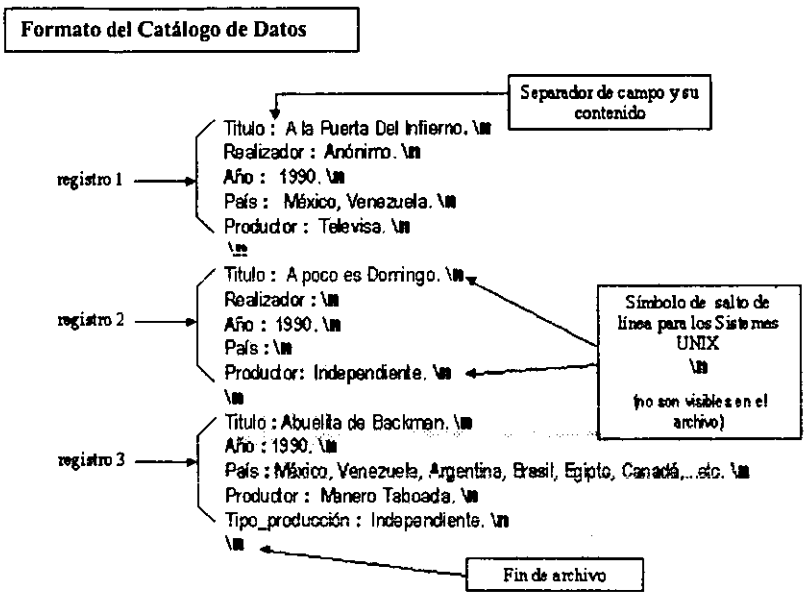
Especificación de Rutas de archivo para cada Sistema de Búsqueda, en este proceso, que depende de la obtención de datos y adecuación del espacio de trabajo, se configuran todas las nuevas rutas para posteriormente utilizarlas en la generación instantánea de las formas de web y los programas cgi. Las rutas a identificar son:

Home de la cuenta de usuario	/users/user/filmoteca
Ubicación del catálogo de datos p.ej.	/users/user/filmoteca/baseweb/base1
Ruta del directorio htm-docs	/usrs/user/filmoteca/html-docs
Liga al servidor web de htm-docs	/filmoteca
Ruta del directorio cgi-bin	/users/user/filmoteca/cgi-bin
Liga al servidor web de cgi-bin	/cgi-bin/filmoteca

El sistema genérico deberá ser capaz de enviar una verificación de rutas en tiempo de corrida **(proceso 2.7)**, ya que el éxito del producto se basa en gran medida en la estructura de archivos.

3. Adecuación del catálogo de datos.

Al ser entregado el archivo con el formato acordado, el desarrollador deberá hacer una revisión



exhaustiva del formato, ya que alguna pequeña variación en el formato del archivo repercutirá en el correcto funcionamiento del sistema.

El siguiente proceso es de vital importancia en el correcto funcionamiento ya que el sistema basa su funcionamiento en el reconocimiento de la estructura del catálogo, asignando un índice que permitirá posteriormente la recuperación de la información por cada campo elegido.

El desarrollador del proyecto, deberá colocar el catálogo en el espacio asignado para la base de datos **(proceso 3.1)** en la estructura de archivos previamente definida, en este lugar el sistema podrá identificarlo y posteriormente reconocer su estructura **(proceso 3.2)**. Es decir, una vez

verificada la base de datos el archivo debe colocarse en el directorio */baseweb/bases* y además de proporcionar los permisos adecuados para su uso *> chmod 750 nombre.txt*

NOTA: si los campos son nulos, podemos poner la etiqueta (nombre del campo) y enseguida el salto de línea o simplemente podemos omitir el campo completamente.

El orden de los campos, tampoco es muy importante, a excepción del primer campo que siempre deberá ser el mismo.

a) Transformación de la base a formato de TAGS **(proceso 3.3)**

El siguiente proceso consiste en la transformación del formato entregado por el usuario al formato de TAGS (SGMLTAG) uno de los formatos que reconoce el software encargado de Indexar la información : lindex.

En el formato de TAGS se indica el nombre de cada campo encerrándolo en picoparéntesis formando una etiqueda idéntica a las etiquetas utilizadas en la codificación HTML, por ejemplo el campo titulo tendrá una etiqueta <Título> el campo realizador tendrá una etiqueta <Realizador> , etc.

El contenido del campo se limitará entre dos etiquetas.

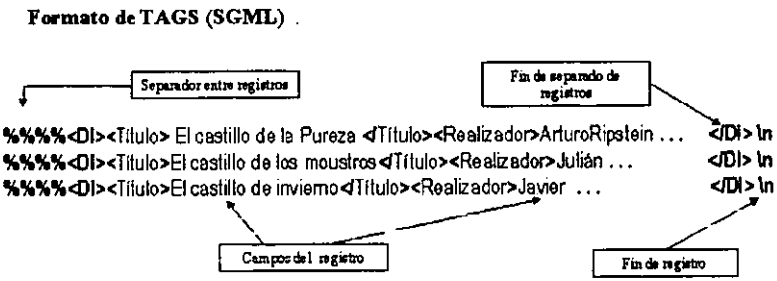
La etiqueta de inicio indica que la siguiente información forma parte del campo , el final estará delimitada por otra etiqueta idéntica, pero inicia con una línea diagonal.

Ejemplo:

<Titulo> El castillo de la Pureza </Título>

Todos los campos pertenecientes a un registro deberán estar a renglón seguido y estarán delimitados por un campo adicional %%%%<DI> y finalizados por </DI> , este campo adicional nos servirá para el despliegue del registro incidente a las palabras de búsqueda.

Para esta transformación, necesitaremos un programa que reconozca el archivo inicial txt, lo descomponga identificando cada registro, cada campo, y finalmente lo imprima a un nuevo archivo.



Debo resaltar en este punto las grandes ventajas de utilizar Perl para este propósito, ya que el manejo que le da a los archivos y a las expresiones regulares es realmente transparente.

El nuevo archivo deberá tener el mismo nombre que el archivo original, pero con extensión .bdf para su posterior indexación.

EL proceso de indexación del catálogo de datos **(proceso 3.4)** lo lleva a cabo el lindex, incluido en la suite de Isearch.

La indexación se realiza por medio de una instrucción desde línea de comandos en UNIX, el comando:

lindex ruta/base.txt opciones lugar

El indexador de archivos de texto realiza su labor a partir de reconocer el formato SGMLTAG (entre otros posibles reconocidos) o denominado en su nueva versión el SGMLNORM , estos formatos son considerados como estándares.

Una vez reconocidas los elementos del archivo de texto, registros, campos y su respectivo contenido se genera una tabla de índices para localización de incidencia de palabras.

Los índices creados son almacenados en el mismo directorio donde se encuentre el archivo de datos.

Ejemplos de espacios requeridos para el almacenamiento de índices de archivo.

Propietario Sistema	archivo txt (bytes)	archivo bdf (bytes)	Indices (bytes)
Filmografía Nacional		11,513,068	12,925,784
CEIICH Investigación	2,090,635	3,027,792	3,476,005

La herramienta de indexación genera una serie de archivos (variable el número de ellos, proporcionales al tamaño del archivo de datos)

archivo.mdt

archivo.inx y una serie de índices con una extensión formado por un número progresivo.

Ejemplo : Listado de Indices de datos del catálogo de datos -

BASES.001	BASES.015	BASES.029	BASES.043	BASES.057	BASES.071
BASES.002	BASES.016	BASES.030	BASES.044	BASES.058	BASES.072
BASES.003	BASES.017	BASES.031	BASES.045	BASES.059	BASES.073
BASES.004	BASES.018	BASES.032	BASES.046	BASES.060	BASES.074
BASES.005	BASES.019	BASES.033	BASES.047	BASES.061	BASES.075
BASES.006	BASES.020	BASES.034	BASES.048	BASES.062	BASES.076
BASES.007	BASES.021	BASES.035	BASES.049	BASES.063	BASES.077
BASES.008	BASES.022	BASES.036	BASES.050	BASES.064	BASES.078
BASES.009	BASES.023	BASES.037	BASES.051	BASES.065	BASES.dfd
BASES.010	BASES.024	BASES.038	BASES.052	BASES.066	BASES.inx
BASES.011	BASES.025	BASES.039	BASES.053	BASES.067	BASES.mdt
BASES.012	BASES.026	BASES.040	BASES.054	BASES.068	angel15.bdf
BASES.013	BASES.027	BASES.041	BASES.055	BASES.069	
BASES.014	BASES.028	BASES.042	BASES.056	BASES.070	

Especificación de los datos para cada sistema de búsqueda **(proceso 4)**

En esta serie de procesos se definen las particularidades de cada sistema, como los nombres de cada campo del catálogo por el cual se efectuarán las búsquedas **(proceso 4.1)** así como los datos y rótulos que aparecerán en la forma de web **(proceso 4.2)**

A partir de estos procesos pasamos al 5 y 6.

4. Creación automática de páginas y las formas de WEB.

Ya que el sistema cuenta con la información necesaria como los nombres de los campos de búsqueda, mediante perl, fácilmente se podrá generar el código HTML de la interfaz de usuario.

El programa creará un nuevo archivo con extensión html en el lugar indicado en la estructura de archivos. colocando la siguiente información:

Información que identifique el tipo de servicio que se proporciona

Una forma que permita al usuario introducir la información de las búsquedas que desea realizar.

Esta forma principal que recibe los datos del usuario (interfaz de usuario) será creada automáticamente aunque sólo se cubrirá el aspecto funcional, el aspecto de diseño (fondo de la página, logotipos, tipos de letra, etc.) deberá ser adecuado de la forma tradicional para cada usuario, pero cabe señalar que en versiones posteriores, el sistema podría contar con establecimiento de diseño automático adecuado para cada aplicación.

a) Las instrucciones de uso de la forma.

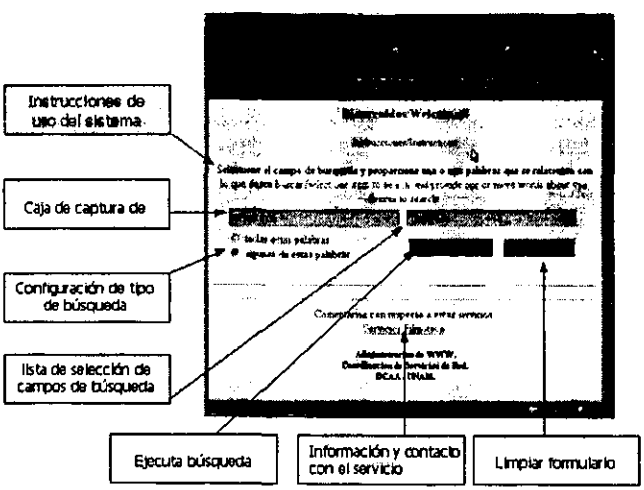


Figura. Pantalla principal del sistema de búsqueda al catálogo "Filmografía Nacional" de la Filmoteca de la UNAM.

El despliegue de campos que contiene la forma (o los indicados por el usuario)

Las otras páginas que se deben generar automáticamente son las de despliegue de resultados. Y deben considerar si el usuario desea un despliegue de un número de incidencias predefinido.

5. Creación del CGI (proceso 6)

La recepción, validación y filtrado de las palabras que proporciona el usuario, la creación y ejecución de los comandos de búsqueda, la recepción y adecuación de los resultados a la petición son las labores que se encargara de realizar el CGI todas estas acciones serán adecuadas a cada sistema de búsqueda en específico.

a) Código de recepción de datos de forma WEB **(proceso 6.1)**

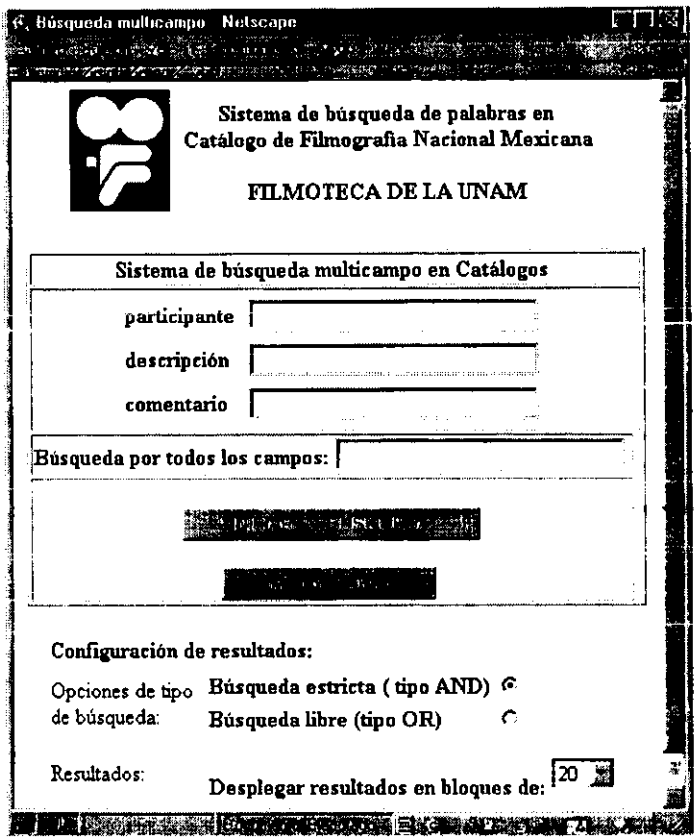
La recepción de datos la realiza una forma de web, esta forma contiene campos donde el usuario configurará sus búsquedas.

La generación de código corresponde a generar una forma HTML que tendrá los datos específicos a cada sistema, como por ejemplo las instrucciones de uso, nombres de los campos de búsqueda con sus respectivas cajas de introducción de texto, el cuadro de configuración de tipo de búsqueda y de resultado, además de los botones de ejecutar y limpiar forma.

b) Código de validación de palabra de búsqueda del usuario

(proceso 6.2) y Código de filtrado de palabras de uso frecuente **(proceso 6.3)**

Una vez activado el botón de ejecucción en la forma, los datos viajan al servidor a través de httpd y son recibidos por el cgi que los validará con el fin de evitar que estos datos sean nulos y con la finalidad de eliminar las palabras de uso frecuente tales como: de, la, los, las, un, y, o alguna palabra muy repetida como la palabra "universidad" en un catálogo de universidades, además de eliminar caracteres irregulares como: #, $, %,&, etc. Este filtrado se hace a partir de un "análisis sintáctico" de las variables enviadas por el usuario y la eliminación de las palabras coincidentes como de uso ilegal.



Forma de captura del servicio de Búsqueda Web

Este filtrado de palabras es muy conveniente para darle seguridad al sistema, así el usuario no podrá enviar comandos ilegales.

c) Código de creación de comandos de búsqueda **(proceso 6.4)**

La creación de los comandos de búsqueda se efectúa una vez realizado el filtrado y validación de los datos que proporcione el usuario.

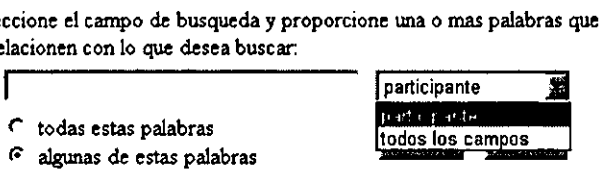
Estos comandos consisten en una línea que se ejecutará llamando al sistema.

Todas las posibles combinaciones de tipo de búsqueda deben considerarse:

- El usuario coloca una palabra en un solo campo.
- El usuario coloca dos o más palabras en un solo campo con el conector AND
- El usuario coloca dos o más palabras en un solo campo con el conector OR

Y así todas las posibles combinaciones

♦Para el modo 1 "búsqueda simple"



El CGI generará al tiempo de corrida las 4 posibles configuraciones de comandos de búsqueda.

La estructura de los comandos está definida por el motor de incidencia de palabras en el catálogo ISEARCH.

Comando:

Isearch [opciones de búsqueda] **palabra1/campo1 or palabra2/campo1 or....**

Buscar los registros incidentes con la **palabra1** por el **campo1** (un campo elegido de la lista de campos posibles búsqueda) **O** buscar a **palabra2** por el **campo1** (el mismo campo elegido) **O...**

Comando:

Isearch [opciones de búsqueda] **palabra1/campo1 and palabra2/campo1 and....**

Buscar los registros incidentes con la **palabra1** por el **campo1** (un campo elegido de la lista de campos posibles búsqueda) **Y** buscar a **palabra2** por el **campo1** (el mismo campo elegido) **Y...**

Comando:

lsearch [opciones de búsqueda] **palabra1/todos_los_campos or palabra2/ todos_los_campos or....**

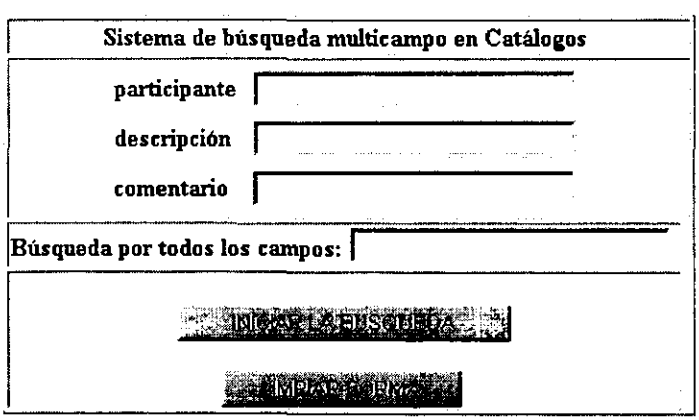
Buscar los registros incidentes con la **palabra1** por **Todos** los campos del catálogo **O** buscar a **palabra2** por **Todos** los campos del catálogo **O...**

Comando:

lsearch [opciones de búsqueda] **palabra1/todos_los_campos and palabra2/ todos_los_campos and....**

Buscar los registros incidentes con la **palabra1** por **Todos** los campos del catálogo **Y** buscar a **palabra2** por **Todos** los campos del catálogo **Y...**

♦Para el modo 2 "Búsqueda por varios campos"



El CGI generará también al tiempo de corrida las 4 posibles configuraciones de comandos de búsqueda.

Comando:

lsearch [opciones de búsqueda] **palabra1/campo1 or palabra1/campo2 or palabra2/campo1 or palabra2/campo2 or....**

Buscar los registros incidentes con la **palabra1** por el **campo1** O buscar la **palabra1** por el **campo2** O buscar la **palabra2** por el **campo1** O la **palabra2** por el **campo2** O... **etc.**

Comando:

lsearch [opciones de búsqueda] **palabra1/campo1 and palabra1/campo2 and palabra2/campo1 and palabra2/campo2 and ...**

Buscar los registros incidentes con la **palabra1** por el **campo1** Y buscar la **palabra1** por el **campo2** Y buscar la **palabra2** por el **campo1** Y la **palabra2** por el **campo2** Y... **etc.**

Comando:

lsearch [opciones de búsqueda] **palabra1/todos_los_campos or palabra2/ todos_los_campos or....**

Buscar los registros incidentes con la **palabra1** por **Todos** los campos del catálogo O buscar a **palabra2** por **Todos** los campos del catálogo **O**. . .

Comando:

lsearch [opciones de búsqueda] **palabra1/todos_los_campos and palabra2/ todos_los_campos and....**

Buscar los registros incidentes con la **palabra1** por **Todos** los campos del catálogo **Y** buscar a **palabra2** por **Todos** los campos del catálogo **Y**. . .

d) Código de recepción de resultados **(proceso 6.5)**

Una vez que el sistema ejecuta el comando, lsearch entrega los resultados de las incidencias encontradas. Estos resultados son redireccionados por el sistema hacia un archivo de texto.

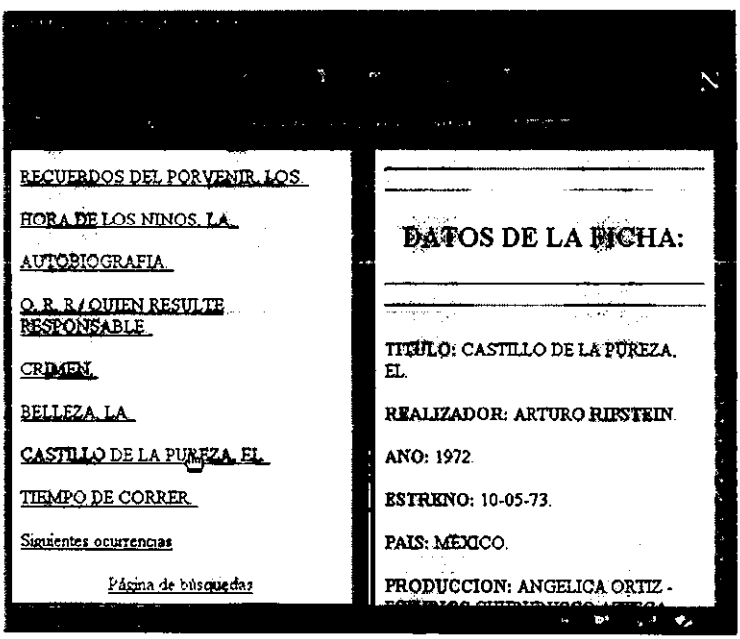
Con el archivo de texto guardado en un directorio temporal, un *scrip* en Perl se encarga de acondicionarlos para enviarlos por web.

e) Código de generación de página de resultados **(proceso 6.6)** y de envío de resultados según configuración de usuario **(proceso 6.7)**

Las páginas HTML con los resultados se generan a partir de un archivo de texto plano, éstas pueden ser de dos tipos, por el modo de frames que permite al usuario sólo identificar una parte del resultado y seleccionar el contenido de esa ficha, y la segunda es por medio de una configuración de despliegue en bloques de un número seleccionado.

♦Modo1 frames

Una vez seleccionada la hiperliga de preferencia, en el lado derecho de la página aparecerá el correspondiente registro con todos los datos que tiene registrado el catálogo.



De esta manera, se proporciona una navegación aleatoria a los registros incidentes ya que el usuario podrá discriminar los registros que no le sean familiares con lo que busca partiendo de la breve información que proporciona la hiperliga.

Este tipo de despliegue de resultados tiene gran aceptación en catálogos del tipo bibliográficos, hemerográficos o videográficos, ya que el título del libro, artículo o película puede dar

información muy importante al usuario y tener una forma mas breve de llegar a la información que se necesita.

♦Modo 2 despliegue en bloques previamente configurados.

La segunda configuración en el despliegue de resultados consiste en una pagina que muestra los resultados en forma secuencial ordenados en bloques con un número configurable por el usuario de registros.

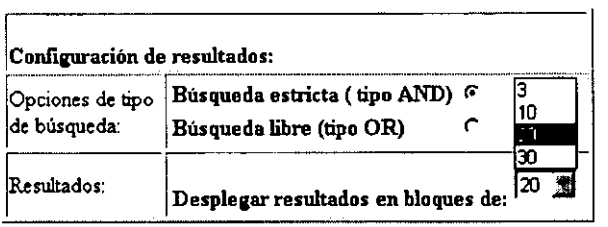


Fig. Cuadro de configuración de resultados en bloques.

Esta segunda forma de despliegue tiene algunas ventajas sobre el modo 1 de visualización de resultados; la principal de ellas es que no requiere de la generación de archivos temporales, ya que para el despliegue de cada bloque de resultados el CGI se ejecuta a si mismo aprovechando que la herramienta de búsqueda Isearch permite también la configuración en el despliegue de resultados.

Otra característica de este despliegue es que podemos navegar entre los bloques de resultados en una forma que no necesita de muchos recursos del cliente.

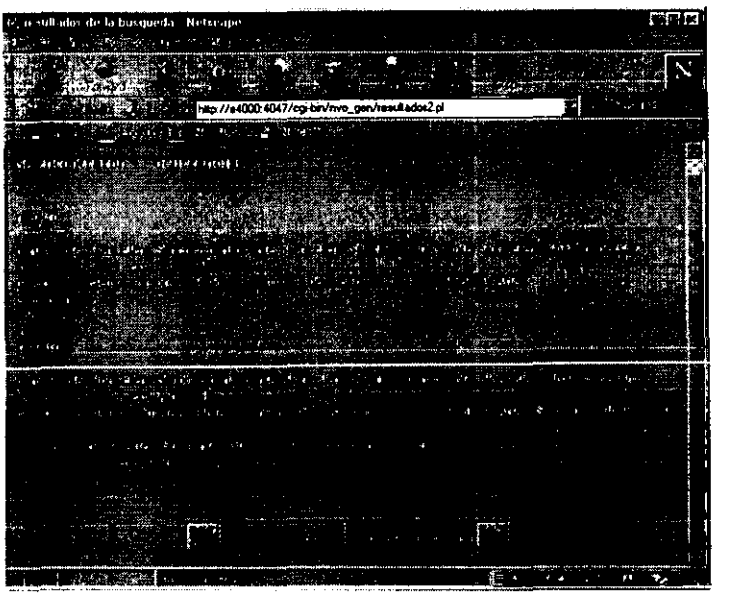


Fig. Página de resultados a bloques de 2 incidencias de una consulta a catálogo

La única desventaja que podemos encontrar es que el usuario no podrá discriminar los resultados que desea ver y tendrá que recorrer uno a uno de los bloques hasta encontrar la información deseada, aunque esto no es problema si no se tienen mas pistas de la búsqueda.

f) Código de navegación de resultados **(proceso 6.7)**

En el código de navegación de resultados es especifico para el modo 2 de despliegue, ya que requiere hacerse un cálculo dinámico del total de resultados o incidencias obtenidas para cada búsqueda y generar a tiempo de corrida las correspondientes páginas HTML que el usuario va examinando. Esto por supuesto ocupará más recursos del servidor implicando mayor comunicación con el cliente y tiempo de espera.

6. Creación del Sistema de Actualización Automática de Catálogos.

En la fase de análisis se contempló la necesidad de independizar al usuario proporcionándole una herramienta que le permita actualizar él mismo su catálogo de datos, ya que por lo regular los catálogos cambian más frecuentemente que una base de datos convencional.

En el mismo programa se incluye código que genera un sistema de actualización automática de catálogos. Este sistema consiste en un conjunto de comandos encargados de repetir los procesos que ejecutaría el encargado de actualizar los catálogos en forma tradicional "a mano", que son básicamente la adecuación de los datos, su indexación y colocación en el lugar adecuado.

Este sistema parte de la misma premisa "el archivo catálogo de datos tiene el formato que reconoce el sistema, el formato acordado" ya que cualquier variación en el, causará que el buscador no reconozca los datos.

Este sistema adicional le da un valor agregado, ya que disminuye considerablemente la carga de trabajo en Cómputo Académico evitando que los usuarios dependan de la dependencia para cada una de sus actualizaciones.

7. Adaptaciones finales del sistema.

Las adaptaciones finales se refieren a las consideraciones especiales de diseño que el usuario requiera, ya que el sistema es de carácter general, será necesario personalizar dando los toques finales en cada sistema, en posteriores versiones se puede agregar una suite de diseños varios a elegir en la misma ejecución del sistema Genérico.

8. Entrega del sistema de búsqueda y capacitación del personal para el uso del sistema de actualización automática.

El paso final es la entrega del servicio funcional ya directamente comprobable a partir de su URL y un navegador.

El procedimiento último es la entrega del oficio donde se indican los datos del nuevo servicio, del sistema de actualización automática con su respectivo manual de uso, y además se menciona el compromiso por parte del usuario a mantenerlo actualizado.

B. Conclusiones Generales

1. Resultados del sistema

La metodología para el desarrollo de software es parte esencial para lograr obtener un producto de alta calidad, la fase de diseño nos permite considerar todas aquéllas posibles situaciones que debe satisfacer nuestro sistema, además de esa manera obtenemos un código claro y de fácil adecuación a posteriores mejoras del mismo.

La metodología utilizada para el desarrollo de "Genérico", particularmente la fase de diseño de procesos, fue de vital importancia, ya que formó parte medular en la realización del código y en la prospección de otras necesidades.

El diseño de cada uno de los procesos a seguir nos ayudó considerablemente a no perder de vista los posibles aspectos que facilitan la labor de los desarrolladores y del mismo usuario final. En esta fase del proyecto surgió la necesidad de incluir a la par del sistema de búsqueda, la creación de su propio Sistema de Actualización Automática de Catálogos, con el objetivo de independizar un poco al usuario permitiéndole libertad de actualizar sus catálogos de datos (ya incluido en esta versión de Genérico).

El uso del software libre sin duda fue parte esencial del éxito obtenido, ya que todas las ventajas que ofrece este tipo de software mencionadas en el capítulo 3 nos permitieron desarrollar un sistema sencillo en términos de código y de apoyo sobre todo en el motor de búsqueda de cada sistema. Además mencionar claramente el considerable ahorro económico que implicó el uso de estos brillantes desarrollos altruistas.

Otra parte muy valiosa de este proyecto es sin duda su flexibilidad, ya que puede fácilmente ser adaptado a toda clase de plataforma UNIX, incluso a plataformas Win-Intel con consideraciones especiales. Esto fue posible sin duda gracias a su concepción inicial, y a la adecuada elección de las herramientas para su desarrollo, como los son Perl, Apache e Isite.

En general, el desarrollo del sistema Genérico logró satisfacer el objetivo primordial de encontrar una solución sencilla eficaz y además económica para la generación de los Sistemas de búsqueda de catálogos publicados en Web.

El sistema "Genérico" actualmente se encuentra ya trabajando en la producción de los sistemas en serie, instalado en varios equipos con diferentes plataformas en la Coordinación de Servicios de Red de la Dirección General de Servicios de Cómputo Académico DGSCA.

Los sistemas que ha generado hasta estas fechas, en general son muy estables en su funcionamiento, confiables en cuanto a la seguridad y muy variables en su tipo de aplicación, desde el Catálogo de Filmografía Nacional de la Filmoteca de la UNAM que ha tenido mucho reconocimiento internacional, el registro de investigadores de CEIICH, registro de publicaciones e investigaciones, catálogos de noticias periodísticas, entre otros.

Anexo 1 **Código fuente, sistema Genérico**

Anexo 1 Código en Perl del Sistema Genérico.

```
#!/usr/bin/perl

use CGI;
print "Content-type: text/html\n\n";

$correctos="n";
print "\n";
while ($correctos eq "n")
{
        system ("clear");
        print " ************************************************************\n\n";
        print " Estructura basica nesesaria:\n\n
             home_________html-docs
                        |        |
                        |        |__baseweb (formas html)
                |       |__tempo(resultados temporales)
                |
                |__baseweb
                |       |__bases (base de datos indexada)
                |       |__reportes (reportes generados)
                |
                |__utilerias
                        |        |__perl, Isite (programas necesarios)
                        |__cgi-bin (cgi-bin) ";


        print "siguinete -> presiona cualquier tecla\n";
        <stdin>;
        system ("clear");

        print " \n*****  CONFIGURACION DE RUTAS******\n\n\n";
        print "Dar la ruta de home [incluyendo el directorio base generico]\n";
        $home= <STDIN>;
       chop($home);
        print "Dar  la ruta del directorio html-docs \n";
        $ruta_html=<STDIN>;
        chop($ruta_html);
        print "Proporcione la liga del servidor de web  html-docs \n";
```

```
        $html=<STDIN>;
        chop($html);
        print " Dar la ruta del directorio cgi-bin \n";
        $ruta_cgi= <STDIN>;
        chop($ruta_cgi);
        print " Dar la liga del servidor de Web  para CGI \n";
        $cgi_bin= <STDIN>;
        chop($cgi_bin);
        print "\n\n Mensaje importante: \n";

    print "Esta version generadora, cuenta con su propia version de perl5\n";
    print "probablemente ya se encuentre instalada alguna version en el\n";
    print "el servidor, desea utilizar la version del Servidor ? [s/n/?]\t";
        $desea=<STDIN>;
        chop($desea);

# el siguinete ciclo while valida que solo sea si o no

        while($desea  ne "s" && $desea ne "S" && $desea ne "n" && $desea ne "N")
        {
        print "\nresponda [n] si desea utilizar la version integrada al  sistema\n";
        print "responsa [s] si desea utilizar la version \"perl 5\" ya instalada\n";
        $desea=<STDIN>;
        chop($desea);

    }

       if ($desea eq "s" || $desea eq "S")
       {
         print "\n Dar la nueva ruta para Perl version 5  [/usr/bin/perl]\t";
         $ruta_perl=<STDIN>;
         chop($ruta_perl);
       }
       else
       {
         $ruta_perl="$home/utilerias/perl";
       }

         system ("clear");
         print " ************************************************************\n\n";
         print " *****          VERIFICANDO LA RUTAS          ******\n\n";
         print " HOME PARA CUENTA        $home \n\n";
```

```
        print " ruta servidor html      $ruta_html \n";
        print " liga servidor html      $html \n\n";
        print " ruta servidor cgi-bin   $ruta_cgi \n";
        print " liga servidor cgi-bin   $cgi_bin \n";
       print " ruta al binario de Perl version 5   $ruta_perl\n\n";
        print " SON CORRECTOS LOS DATOS?    [y][n]\n\n";
        $correctos= <STDIN>;
        chop($correctos);
}

if ($desea eq "s" || $desea eq "S")
  {
        print " Ya que utilizara Perl5 instalado en el servidor, desea borrar la version integrada";
        print " \nen el generador? [s / n ]\t";
        $se_borra=<STDIN>;
        chop($se_borra);
        if ($se_borra eq "s" ||  $se_borra eq "S" )
        {
                open (BORRA_PERL, "rm  $home/utilerias/perl |");
                close (BORRA_PERL);
        }
        else
        {
                print " \nSi desea borrarlo porteriormente la ruta es $home/utilerias/perl";
    }
  }

#**********copiando archivos**********

open (TEMPO, "mkdir $ruta_html/tempo |");
close (TEMPO);
open (PERTEMPO,"chmod 775 $ruta_html/tempo |");
close (PERTEMPO);
open (COPIA1, "cp $home/CGI.pm $ruta_cgi/CGI.pm |");
close (COPIA1);
open (COPIA2, "cp $home/shellwords.pl $ruta_cgi/shellwords.pl |");
close (COPIA2);
open (PERCGI, "chmod 755 $ruta_cgi/CGI.pm |");
close (REPCGI);
open (PERSHE, "chmod 755 $ruta_cgi/shellwords.pl |");
close (PERSHE);
```

```
#********************  FORMATO A MULTIPLES BASES  **********************

#Este programa transforma la base (archivo de texto) a un archivo con
#formato de "tags", para que se pueda ser indexado por el programa
#lindex, necessito el nombre de las bases que estaran de antemano
#ubicadas en el directorio  ..generico/baseweb/bases

print "\nEl sistema detecto las siguientes bases en el directorio\n";
print "correspondiente:\n\n";

@bases=();
$cont=0;

open (NOM_BASES, "ls baseweb/bases/*.txt|");

while(<NOM_BASES>)
{
chop($_);
$bases[$cont]=$_;
print "$cont. $bases[$cont]\n";
$cont ++;
}

#******* Nombres de los campos y rotulos de los mismos en web ********

print "\n::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::";
print "\n\nEl sistema se dispondra a darle formato a las bases de";
print "\ndatos disponibles en el directorio ../baseweb/bases";
print "\nCada base de datos tendra algun campo donde se desee realizar";
print "\nlas busquedas, a continuacion necesito que me proporcione el";
print "\nnombre de cada campo y de su respectivo rotulo en la pagina HTML";
print "\nbusquedas \n\n";
print "::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::\n";

print "Se realizaran busquedas en las siguinetes bases de datos\n";
print "\nEn cuantos campos se realizaran busquedas:>";
$numcamp=<STDIN>;
chop($numcamp);

print "\nNota:\n";
print ":::::::\n\n";
```

```
print "Es importante que el nombre del campo sea identico a como
aparece\n";
print "en su respectiva base de datos\n";
print "En el rotulo para WEB, no olvide colocar el codigo
correspondiente\n";
print "a caracteres especiales [titulo = T&iacute;tulo]\n\n";

@campos=();
@rotulos=();

for ($i=1; $i<$numcamp+1; $i++)
{
    print "\n campo numero $i :>";
    $campo=<STDIN>;
    chop($campo);
    $campos[$i]=$campo;
    print "\nRotulo del campo:$campos[$i] para Web:";
    $rotulo=<STDIN>;
    chop($rotulo);
    $rotulos[$i]=$rotulo;
}

#*****************************************************************

print "\n\n::::: El sistema esta dando formato a sus bases de datos :::::\n";

for ($i=0; $i<$cont; $i++)
{
  $archivotxt = $home."\/$bases[$i]";
  #print "$archivotxt\n";

  #debe ponerse la ruta donde esta el archivo a modificar
  print "\n\n\n\n:::::: abriendo:: $archivotxt\n\n";
  open (ARCHIVO,"$archivotxt") || die "Archivo da~ado: $archivotxt";
  ($nombre,$extension) = split(/\./,$archivotxt,2);

  #Toma el nombre del archivo y lo separa en nombre y su extension
  $archivobdf = $nombre . ".bdf";  # modifica el archivo con todo u ruta y cambia .txt.bdf
  $bandera=0;

  #el siguiente codigo es solo para asegurar que el nuevo archivo .bdf no
  #tenga nada y ademas tenga el permiso adecuado
```

```
open (BASE1,">$archivobdf");
print "el nuevo archivo se llama $archivobdf\n";
open (PERBASE1, "chmod 755 $archivobdf |");
close (BASE1);
close  (PERBASE1);


# la siguiente rutina cambia los caracteres especiales como acentos por letras no acentuadas

while (<ARCHIVO>)
{
 if ($_ ne "\n")
   {
    chop($_);
    $cambia = $_;
    #$cambia =~ s/\341/a/g;
    #$cambia =~ s/\351/e/g;
    #$cambia =~ s/\355/i/g;
    #$cambia =~ s/\363/o/g;
    #$cambia =~ s/\372/u/g;
    #$cambia =~ s/\361/ni/g;
    #$cambia =~ s/\321/NI/g;
    #$cambia =~ s/\301/A/g;
    #$cambia =~ s/\311/E/g;
    #$cambia =~ s/\315/I/g;
    #$cambia =~ s/\323/O/g;
    #$cambia =~ s/\332/U/g;
    #$cambia =~ s/\015/ /g;
    $cambia =~ s/\\s+/\s/g;

    ($tag,$frase) = split(/:/,$cambia,2);
    $inicio ="<$tag>";
    $fin = "</$tag>";
    $bandera=1;
   }
 open (BASE, ">>$archivobdf") || "no lo pude abrir";
 #if ($inicio eq "<TITULO>" || $inicio eq "<Titulo>")
 if ($inicio eq "<$campos[1]>")
  {
   print BASE  "%%%%<DI>";
  }
```

```
    print BASE "$inicio $frase $fin";
    if ($_ eq "\n" && $bandera == 1)
    {
      print BASE "</DI>\n";
      $bandera = 0;
    }

    $inicio = "";
    $frase = "";
    $fin = "";
  }
  close(ARCHIVO);
  close(BASE);
  print "\n\n:::::formato finalizado para $bases[$cont] :::::\n\n";
}

#**************************** INDEXANDO B.D.*****************************

print "\n\n:::::::::::::::: Indexando las bases ::::::::::::::::::\n\n";
$index= "$home/utilerias/lindex -d $home/baseweb/bases/BASES -m 5 -s \"%%%%\" -t
SGMLTAG $home/baseweb/bases/*.bdf";
print "$index\n";
system($index);

#****************** Sistema de Actualizacion Automatica  de base de datos *********

open (ACTUAL, ">ACTUALIZA.pl");
open (PER_ACTUAL, "chmod 750 ACTUALIZA.pl|");
close (PER_ACTUAL);

print ACTUAL <<"fin_archivo";
#!/usr/bin/perl

# Este porograma tiene por objetivo actualizar el catalogo (la base de datos) del usuario,
# en forma automatica.

#Este programa sirve para que el archivo de texto sea pasado a un archivo con
#formato de \"tags\" y ademas para que se pueda ser indexado por el programa
#lindex.

print \"Dar el nombre de la base de datos >\";
```

```
\$archivotxt=<STDIN>; #recibe el nombre del archivo a formatear
chop(\$archivotxt);
#\$home=\"/usr/home4/filmo/nuevo_gen/gen1\";
print \"home de la cuenta \$home\\n\";
print \"tu ruta es \$home/baseweb/bases/\$archivotxt\";
#debe ponerse la ruta donde esta el archivo a modificar
open (ARCHIVO,\"\$home/baseweb/bases/\$archivotxt\") || die \"No existe el archivo: archivotxt\";

(\$nombre,\$extension) = split(/\./,\$archivotxt,2);

print \"nombre \$nombre y extension \$extension\";
# Toma el nombre del archivo y lo separa en nombre y su extension
\$archivobdf = \$nombre . \".bdf\";  # modifica el archivo y cambia .txt .bdf
\$bandera=0;

#el siguiente codigo es solo para asegurar que el nuevo archivo .bdf no
#tenga nada y ademas tenga el permiso adecuado

open (BASE1,\">\$home/baseweb/bases/\$archivobdf\");
open (PERBASE1, \"chmod 750 \$home/baseweb/bases/\$archivobdf |\");
close (BASE1);
close  (PERBASE1);
open (CPY, \"cp /home/servicios/ceiich/gen/baseweb/bases/\$archivotxt tempo|\");
close (CPY);
open (PERTMP, \"chmod 777 tempo|\");
close (PERTMP);
open (CAT, \"cat tempo|tr -d \'[\015]\'>/home/servicios/ceiich/gen/baseweb/bases/\$archivotxt|\");
close(CAT);
open (RM, \"rm -r tempo|\");
close (RM);


# la siguiente rutina cambia los caracteres especiales como acentos por
#letras no acentuadas


while (<ARCHIVO>)
{
 if (\$_ ne \"\\n\")
 {
   chop(\$_);
   \$cambia = \$_;
```

```
  #\$cambia =~ s/\\012\\101\\116\\117/\\012\\101\\116\\111\\117/g;
  #\$cambia =~ s/\\341/a/g;
  #\$cambia =~ s/\\351/e/g;
  #\$cambia =~ s/\\355/i/g;
  #\$cambia =~ s/\\363/o/g;
  #\$cambia =~ s/\\372/u/g;
  #\$cambia =~ s/\\361/ni/g;
  #\$cambia =~ s/\\321/NI/g;
  #\$cambia =~ s/\\301/A/g;
  #\$cambia =~ s/\\311/E/g;
  #\$cambia =~ s/\\315/I/g;
  #\$cambia =~ s/\\323/O/g;
  #\$cambia =~ s/\\332/U/g;
  \$cambia =~ s/\\015//g;
  \$cambia =~ s/\\\\s+/\\s/g;

  (\$tag,\$frase) = split(/:/,\$cambia,2);
  \$inicio =\"<\$tag>\";
  \$fin =\"</\$tag>\";
  \$bandera=1;
 }

  open (BASE,\">>$home/baseweb/bases/\$archivobd\") || \"no lo pude abrir\";
  if (\$inicio eq \"<$campos[1]>\" || \$inicio eq \"<$campos[1]>\")
   {
   print BASE \"%%%%<DI>\";
   }
   print BASE \"\$inicio \$frase \$fin\";

   if (\$_ eq \"\\n\" \&\& \$bandera == 1)
   {
     print BASE \"</DI>\\n\";
     \$bandera = 0;
   }
#  print \"#\";
  \$inicio = \"\";
  \$frase = \"\";
  \$fin = \"\";
}
close(ARCHIVO);
close(BASE);
print \"ya termine de darle formato para ser utilizado por el lindex\\n\";
```

```
#open (BORRA, \"rm $home/baseweb/bases/\$archivotxt |\");
#close(BORRA);
#*************************** INDEXANDO B.D.*************************

\$index= \"$home/utilerias/lindex -d $home/baseweb/bases/BASES -m 5 -s \\\"%%%%\\\" -t
SGMLTAG $home/baseweb/bases/*.bdf\";
print \"\\n\$index\\n\";
system(\$index);
print \"\\nya lo indexe 8)\\n\\n\";
print \"\\n*********************************************FIN DEL SISTEMA\\n\\n\";

fin_archivo

close (ACTUAL);
#****************** FIN Sistema de Actualizacion Automatica ************



#************************Generando forma*******************************
print"\n\n:::::::::::: Generando formas WEB y CGI ...  ::::::::::::::\n\n";

#el siguiente codigo se encarga de generar el codigo html de la forma
#principal de busqueda, sera necesario decirle que tipo de campos debemos
#contemplar en la forma.

open (FORMA, ">$ruta_html/form_gen.html");
open (PERFORMA, "chmod 755 $ruta_html/form_gen.html |");
close (PERFORMA);
#es importante que el usuario ademas de la liga html-docs nos de la
#ruta total del mismo directorio


open (FORMA, ">$ruta_html/form_gen.html");

&genera;

sub genera
{
print FORMA "<html>
<head> <title>Consulta: Base de DatosUNAM</title></head>
<body> <center>
```

```
<table border=10> <tr>
<td><font size=+2><br>
<center>CONSULTA GENERAL: BASES DE DATOS</center>
<td><font size=+2><br>
</td></tr></table></center>
<br>
<form method=\"POST\" action=\"$cgi_bin/buscador.pl\"> <br>
<CENTER><h1>BIENVENIDOS!</h1></CENTER>
<p>
<P>
INTRUCCIONES:<P>
Seleccione el campo de busqueda y proporcione una o
mas palabras que se relacionen con lo que desea buscar:
<br><center>
<table>
<tr><td><input name=\"palabra\" type=\"text\" size=30>
<td colspan=2>
<select name=\"opcion\">\n";

# qui los campos dependen del usuario

for ($i=1; $i<$numcamp+1; $i++)
{
   print FORMA "<option value=\"$campos[$i]\">$rotulos[$i]\n";
}

print FORMA "<option value=\"todos\">todos los campos";
print FORMA "</select>
<tr>
<td>
<input type=\"radio\" name=\"tipo\" value=\"y\"> todas estas palabras<br>
<input type=\"radio\" name=\"tipo\" value=\"o\" checked> algunas de estas palabras
<td><center>
<input type=\"submit\" name=\"aceptar\"  value=\"buscar\">
<td>
<input type=\"reset\" name=\"limpiar\" value=\"limpiar\"></center>
</td></tr></table>
</center>
</form>
<p>
<p>
<hr></center>Comentarios y sugerencias con respecto a estos servicios:<br>
```

```
<a href=\"mailto:edith\@servidor.unam.mx\">eva\@servidor.unam.mx</a>
<br>
<h5>
Coordinaci&oacute;n de Servicios de Red.<br>
DCAA - UNAM.<br></h5>
</hr>
</body>
</html>";

close (FORMA);
}

#************************genera aviso html*****************************

open (AVISO, ">$ruta_html/aviso.html");
open (PERAVISO, "chmod 755 $ruta_html/aviso.html |");
close (PERAVISO);
print AVISO "<hr><br>";
print AVISO "<H3> <center>BIENVENIDOS A LA BUSQUEDA</center><H2>";
print AVISO "<hr><br>";
close (AVISO);

#************************ generador de cgi busquedas ********************
#el siguiente codigo se encarga de generar el cgi buscador.pl que
#requiere la forma principal forma.html es necesario que se cuente con la
#ruta exacta de donde esta el directorio cgi-bin para en dicho directorio
#crear el cgi.

open (CGI, ">$ruta_cgi/buscador.pl");
open (PERBUSCA, "chmod 755 $ruta_cgi/buscador.pl |");
close (PERBUSCA);

#open (CGI, ">buscador-prueba.pl");

print CGI <<"label1";
#!$ruta_perl
use CGI;
\$query = new CGI;
print \"Content-type: text/html\\n\\n\";

\$PALABRA  = \$query->param('palabra');
\$OPCION = \$query->param('opcion');
```

```
\$TIPO = \$query->param('tipo');
\$ENUNCIA=\$PALABRA;
\$ENUNCIA=~s/\\s/\\+/g;
\$contador = 1;
open(LOGS,\">errores.log\");


#impresiones de verificion de datos (solo para pruebas)--------
#print \"enunciado \$ENUNCIA\";
#print \"\$PALABRA\\n\";
#print \"\$OPCION \\n\";
#print \"\$TIPO \\n\";

&palabras_uso_frecuente;
&valida_entrada;
&genera_comandos;
&comandos_busqueda;
&frames;

#-----------------------------------------------------------------
#pasar por un filtro a la palabra, quitando variables de uso frecuente

sub palabras_uso_frecuente
{
 \$PALABRA=~ s/\\s+/\\040/g;
 \$PALABRA=~ s/^\\s//g;
 \$PALABRA=\" \".\$PALABRA;
 \$PALABRA=~ s/ de / /g;
 \$PALABRA=~ s/ DE / /g;
 \$PALABRA=~ s/ la / /g;
 \$PALABRA=~ s/ La / /g;
 \$PALABRA=~ s/ LA / /g;
 \$PALABRA=~ s/ con / /g;
 \$PALABRA=~ s/ los / /g;
 \$PALABRA=~ s/ Los / /g;
 \$PALABRA=~ s/ las / /g;
 \$PALABRA=~ s/ Las / /g;
 \$PALABRA=~ s/ LOS / /g;
 \$PALABRA=~ s/ LAS / /g;
 \$PALABRA=~ s/ lo / /g;
 \$PALABRA=~ s/ Lo / /g;
 \$PALABRA=~ s/ LO / /g;
```

```
  \$PALABRA=~ s/ el / /g;
  \$PALABRA=~ s/ EL / /g;
  \$PALABRA=~ s/ El / /g;
  \$PALABRA=~ s/ su / /g;
  \$PALABRA=~ s/ SU / /g;
  \$PALABRA=~ s/ En / /g;
  \$PALABRA=~ s/ EN / /g;
  \$PALABRA=~ s/ en / /g;
  \$PALABRA=~ s/ A / /g;
  \$PALABRA=~ s/ a / /g;
  \$PALABRA=~ s/ Y / /g;
  \$PALABRA=~ s/ y / /g;
  \$PALABRA=~ s/ un / /g;
  \$PALABRA=~ s/ UN / /g;
  \$PALABRA=~ s/ UNA / /g;
  \$PALABRA=~ s/ una / /g;
  \$PALABRA=~ s/ mi / /g;
  \$PALABRA=~ s/ MI / /g;
  \$PALABRA=~ s/ Mi / /g;
  \$PALABRA=~ s/^.//;
  #para quitar el primer caracter que agregue
  print LOGS \"palabra a buscar=\$PALABRA \\n\";
  print LOGS \"campo de busqueda=\$OPCION \\n\";
  print LOGS \"tipo de busqueda=\$TIPO \\n\";
}

#-------------------------------------------------------------

#-------------------------------------------------------------

sub genera_comandos
{
  \$comando1=\"\";
  \$comando2=\"\";
  #dividir la palabra (el enunciado) en un arreglo de palabras
  \$_=\$PALABRA;
  \@frase=split(/ /,\$_);

  foreach (\@frase)
  {
  \$PALABRAS=\$PALABRAS.\$_.\"+\";
  }
```

```
chop(\$PALABRAS);
print LOGS \"envia al CGI> \$PALABRAS\\n\";
foreach (\@frase)
{
#----por un solo campo -----
\$comand_or=\$comand_or.\$OPCION.\"/\".\$_.\" or \";
\$comand_and=\$comand_and.\$OPCION.\"/\".\$_.\" and \";
#----todos los campos-------
\$comand_or_all=\$comand_or_all.\$_.\" or \";
\$comand_and_all=\$comand_and_all.\$_.\" and \";

}
chop(\$comand_or);
chop(\$comand_or);
chop(\$comand_or);

chop(\$comand_or_all);
chop(\$comand_or_all);
chop(\$comand_or_all);

chop(\$comand_and);
chop(\$comand_and);
chop(\$comand_and);
chop(\$comand_and);

chop(\$comand_and_all);
chop(\$comand_and_all);
chop(\$comand_and_all);
chop(\$comand_and_all);

\$busca1=\"\$home/utilerias/lsearch -d \$home/baseweb/bases/BASES -p DI -q -infix
\$comand_and_all\";
\$busca2=\"\$home/utilerias/lsearch -d \$home/baseweb/bases/BASES -p DI -q -infix
\$comand_and\";
\$busca3=\"\$home/utilerias/lsearch -d \$home/baseweb/bases/BASES -p DI -q -infix
\$comand_or_all\";
\$busca4=\"\$home/utilerias/lsearch -d \$home/baseweb/bases/BASES -p DI -q -infix
\$comand_or\";

print LOGS \"todos_and\\n\$busca1\\n\\n\";
print LOGS \"campo/and\\n\$busca2\\n\\n\";
print LOGS \"todos_or\\n\$busca3\\n\\n\";
```

```
  print LOGS \"csmpo_or\\n\$busca4\\n\\n\";

}


#-------------------------------------------------------------
#funcion que verifica que la palabra a buscar exista

sub valida_entrada
{
   if ((\$PALABRA eq \"\"))
   {
#      print \"<body background=\"/images/fondo1.gif\">\";
       print \"<br><br><br><center>\";
       print \"<h1> Favor de introducir un t&eacute\\;rmino para iniciar la b&uacute\\;squeda.
<h1><br>\";
       print \"<a href= \\"$html/form_gen.html\\">Regresar al Sistema</a>\";
       print \"</center></body>\";
       exit(1);
   }
   open (COM, \"rm $ruta_html/tempo/resultado.txt |\") || die \"No puedo realizar la operacion \$!\";
   close (COM);
}


#-------------------------------------------------------------------------
#funcion que verifica que no se ponga palabra a buscar

sub comandos_busqueda
{

  if(\$PALABRA ne \"\")
  {
     if(\$TIPO eq \"y\")
     {
        if( \$OPCION eq \"todos\")
        {
                open(BUSCA,\"\$busca1|\") || die  \"no lo ejecute\";
        }else{
                open(BUSCA,\"\$busca2|\") || die  \"no lo ejecute\";
        }
     }
```

```
    if(\$TIPO  eq \"o\")
    {
       if( \$OPCION eq \"todos\")
       {
                  open(BUSCA,\"\$busca3|\") || die  \"no lo ejecute\";
         }else{
                  open(BUSCA,\"\$busca4|\") || die  \"no lo ejecute\";
       }
    }

  \$numero = \$\$;
  print LOGS \"numero de proceso \$numero\\n\";

   while(<BUSCA>)
   {
      if(/<$campos[1]>/)
      {
        \$ban=1;
        &imprime;
            \$TOTAL=\$contador-1;
            print LOGS \"contador final \$contador\\n\";
      }

#     if (/0 document/)
      if(index(\$_,\"0\")==0)
         {
         print \"<body>\";
         print \"<br><br><br><center>\";
         print \"<h1>El termino solicitado\";
         print \" no se encontr&oacute\\;,<p>\";
         print \"favor de modificar su b&uacute\\;squeda \\n<br>\";
         print \"<a href=\\\"$html/form_gen.html\\\">Regresar al Sistema</a>\";
         print \"</center></h1></body>\";
         exit(1);
         }
    }
   close(BUSCA);
   close(TEMPO);
  }
}
```

```
#-------------------------------------------------------------------
#subrutina que imprime el resultado de la busqueda en el directorio resultado.txt

sub imprime
{
        chop(\$_);
        \@lista=split(/></,\$_);
        foreach \$i (\@lista)
        {
            if (\$i ne \\n)
            {
             (\$cadena,\$desecho)=split(/<\\//,\$i);
             \$cadena =~ s/<//;
             \$cadena =~ s/>//;
                        open(TEMPO,\">>$ruta_html/tempo/resultado.txt\");
                       open (PERMISO, \"chmod 755 $ruta_html/tempo/resultado.txt |\");
                            close (PERMISO);
             if(\$ban)
             {
              print TEMPO \"Contador:\$contador\\n\";
                        }
              print TEMPO \"\$cadena \\n\";
              \$ban=0;
            }
        }
        print TEMPO \"Numero:\$numero\\n\";
        print TEMPO \"Contador:\$contador\\n\";
        print TEMPO \"\\n\";
\$numero++;
\$contador++;

}
#-------------------------------------------------------------------

#funcion que genera la pagina html de respuesta

sub frames
{
 print <<FIN;
<HTML>
 <HEAD><TITLE> SERVICIO DE BUSQUEDA </TITLE></HEAD>
  <FRAMESET cols = 40%,60%>
```

```
 <NOFRAME>
   <FRAME
SRC="$cgi_bin/indice.pl?contador=1&opcion=\$OPCION&palabra=\$PALABRAS&tipo=\$TIPO&t
otal=\$TOTAL\">
   <FRAME SRC = \"$html/aviso.html\" name=\"izq\">
 </FRAMESET>
 </NOFRAME>
</HTML>
FIN
}
#-------------------------------------------------------

label1
close (CGI);



#*************************  genera-indice  *************************

open (INDX, ">$ruta_cgi/indice.pl");
open (PERINDX, "chmod 755 $ruta_cgi/indice.pl |");
close (PERINDX);
#open (INDX, ">indice.pl");

print INDX <<"ETIQUETA";
#!$ruta_perl
use CGI;

\$query = new CGI;
\$CONT=\$query->param('contador');
\$OPCION=\$query->param('opcion');
\$PALABRA=\$query->param('palabra');
\$TIPO=\$query->param('tipo');
\$TOTAL=\$query->param('total');
\$RUTA_PRINCIPAL = \"$html/form_gen.html\";

open(LOGS,\">errores_indice\");
print LOGS \"total \$TOTAL\\n\";
\@frase=split(/ /,\$PALABRA);

&resultado;
```

```
#-------------------------------------------------------------
open (COM, \"$ruta_cgi/borrar.pl|\") || die \"No puedo realizar la operacion \$!\";
close (COM);

#-------------------------------------------------------------
#funcion que imprime los archivos temporales

\$x=\$CONT;
\$bandera=\$CONT;
\$z=1;
\$siga=0;

#-------------------------------------------------------------------
open (ARCH, \"$ruta_html/tempo/resultado.txt\");
while (<ARCH>)
{
  #chop(\$_);
  if(\$_ eq \"Contador:\$CONT\n\")
  {
  \$siga = 1;
  }

  #El siguiente ciclo If nos permite desplegar los campos de 15 o de 10 segun lo que se
necesite.
  if((\$x < (\$bandera+10)) && (\$x >= \$bandera) && (\$siga == 1))
  {
    if (\$_ ne \"\n\")
    {
      chop(\$_);
      (\$nombre[\$z],\$valor[\$z]) = split(/:/,\$_,2);
      if (\$nombre[\$z] eq \"Numero\")
      {
        \$ruta=\$valor[\$z]; #se asigna a ruta el identificador del proceso
      }
      \$z++;
    }
    else
    {
      \$r=\$z;
      &encabezados;
      &despliega_ligas;
```

```
        \$x++;
        #\$CONT++;
        \$z=1;
        \$PALABRA=~ s/\\s/+/ig;
        if ((\$CONT%10) == 0)
        {
          \$CONT++;
          print \"<a
href=\\\"$cgi_bin/indice.pl?contador=\$CONT&opcion=\$OPCION&palabra=\$PALABRA&tipo=\$T
IPO&total=\$TOTAL\\\">Siguientes ocurrencias</a>\";
        }
        \$CONT++;
    }
  }
}
&liga_regreso_pagina;
close(ARCH);



#-------------------------------------------------------------
#funcion que genera frame de ligas de resultado
sub resultado
{
 print \"Content-type: text/html\\n\\n\";
 #print \"<body background=\\\"/Images/fondo1.gif\\\">\";
 print \"<H4><CENTER> RESULTADO DE LA BUSQUEDA \";
 print \"PARA EL CAMPO \\\" \$OPCION \\\"\\</H4\\>\\<\\>- \$PALABRA -\\</\\>\</CENTER>\";
 #print \"<P>\";
 print \"<HR>\";
 print \"<h5>fichas<br>encontradas \$TOTAL</h5><br>\";
 #print \"el contador tiene \$CONT\";
}
#-------------------------------------------------------------

#-------------------------------------------------------------
# funcion  encabezado frame de datos de la ficha (archivos temporales)
sub encabezados
{
   open (TEMP,\">$ruta_html/tempo/temp\$ruta.html\") || die \"No puedo crear los archivos \$!\";
   open (PERMISO, \"chmod 755 $ruta_html/tempo/temp\$ruta.html\");
   close(PERMISO);
```

```
    print TEMP \"<body>\";
    print TEMP \"<P>\\n\";
    print TEMP \"<hr>\\n\";
    print TEMP \"<hr>\\n\";
    print TEMP \"<P>\\n\";
    print TEMP \"<center>DATOS DE LA FICHA:</center><p><hr>\\n\";
    print TEMP \"<HR>\\n\";
    print TEMP \"<P>\\n\";
    print TEMP \"<P>\\n\";
}
#-----------------------------------------------------------------

#-----------------------------------------------------------------
sub despliega_ligas
{
   for (\$tmp=1; \$tmp<\$r ; \$tmp++)
   {
      foreach(\@frase)
      {
         \$valor[\$tmp] =~ s/(\$_)/<b>\$1<\/b>/ig;
      }
      if ((\$nombre[\$tmp] ne \"Numero\") && (\$nombre[\$tmp] ne \"Contador\"))
      {
        #el siguinete If es para restringir los campos que deseo desplegar
        #if((\$nombre[\$tmp] eq  \"$campos[1]\")||( \$nombre[\$tmp] eq \"$campo[2]\")||( eq
\"comentario\")||( eq ... \" los necesarios..\"))
              #{
                 #print TEMP \"<b>\$nombre[\$tmp]:</b>\";
                 print TEMP \"<tt>\$nombre[\$tmp]:</tt>\";
           print TEMP \"\$valor[\$tmp]<P>\\n\";
        #}
      }
      #if((\$nombre[\$tmp] eq \"$campos[1]\") || (\$nombre[\$tmp] eq \"$campos[2]\"))
      if((\$nombre[\$tmp] eq \"$campos[1]\"))
      {
        #imprime el nombre del campo antes de la liga
        #print  \"\$nombre[2]: \";
        print \"<a href=\\\"$html/tempo/temp\$ruta.html\\\"target=izq>\$valor[\$tmp]</A><p>\\n\";
      #}
      #else
      #{
```

```
        #print \"\$OPCION: \";
        #print \"<ahref=\\\"$html/tempo/temp\$ruta.html\\\"target=izq>\$valor[\$tmp]</A><p>\\n\";
      }
   }
   print TEMP \"</body>\";
   close(TEMP);
}
#-----------------------------------------------------------------

#-----------------------------------------------------------------
#liga de regreso a pagina de busquedas

sub liga_regreso_pagina
{
print \"\\<P\\>\\<CENTER\\>\";
print \"\\<A HREF=\\\"\$RUTA_PRINCIPAL\\\" TARGET=\\\"_top\\\"\\>\";
print \"P\\&aacute\\;gina de b\\&uacute\\;squedas\";
print \"\\</A\\>\";
print \"\\</CENTER\\>\";
print \"</body>\";
}
#-----------------------------------------------------------------
ETIQUETA
close(INDX);


#************************genera borra.pl ********************************
#Esta parte  del codigo genera  un programa que borra automaticamente por
#medio de comparacion de fecha que obtiene del formato ls -t donde varia
#segun el numero de campos que tenga.

open (BORRA, ">$ruta_cgi/borrar.pl");
open (PERBORRA, "chmod 755 $ruta_cgi/borrar.pl |");

print BORRA <<"FINBORRA";
#!$ruta_perl

\$cont=0;
open(LISTA,"ls -lt $ruta_html/tempo |");
 while(<LISTA>)
 {
  chop(\$_);
```

```
\$cont++;
(\$p1,\$p2,\$p3,\$p4,\$p5,\$p6,\$p7,\$p8,\$p9) = split(/\s+/,\$_,9);
unless(/total/)
{
 if(\$cont == 2) {
    \$mes = \$p6;
    \$dia = \$p7;
    \$hora =\$p8;
  }

  if(\$mes ne \$p6) #ojo debe ser diferente del mes
  {
    &borrar (\$p9);
  }else{
   if ( \$p7 ne \$dia ) #ojo debe ser diferente el dia
   {
    &borrar (\$p9);
   } else {

    (\$hp,\$mp) = split(/:/,\$hora,2);
    (\$hh,\$mm) = split(/:/,\$p8,2);
    \$ultimo = ((int(\$hp) * 60) + int(\$mp));
    \$compara = ((int(\$hh) * 60) + int(\$mm));
     if((\$ultimo - \$compara) > 2)
     {
      &borrar (\$p9);
     }
   }
  }
 }
}#while
close(LISTA);

sub borrar
{
 print"\$_[0]";
 open (BORRAR,"rm $ruta_html/tempo/\$_[0] |");
 close(BORRAR);
}

FINBORRA
close(BORRA);
```

```
close(PERBORRA);

print "\n\n\n\n\n\n";
print "::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::\n";
print "::          EL SISTEMA HA FINALIZADO                    ::\n";
print ":: Ya puede consultar su servicio directamente en WEB    ::\n";
print "::      dudas del sistema eva\@servidor.unam.mx          ::\n";
print "::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::\n";
```

Anexo 2 **Manuales de usuario**

MANUALES

GENERADOR DE SERVICIO DE BÚSQUEDAS PARA CATÁLOGOS PUBLICADOS EN WEB UTILIZANDO Z3950 Y PERL

Realizado por: Eva Edith López López

fecha de finalización: enero 1998

Índice General

1. Panorama general del sistema generador.

Servicios de búsqueda en bases de datos para WEB

El servicio de búsqueda en *Web* es uno de los servicios con mas demanda últimamente, ya que es una forma muy atractiva de mostrar un gran volumen de información por medio de Internet.

Este servicio también funciona bajo la filosofia cliente servidor. Cada cliente tiene con el una interfaz de usuario la cual se encarga de recibir la palabra o frase a buscar por el usuario. El cliente realiza la petición de búsqueda al servidor que además de validar los datos de entrada, genera los procesos de búsqueda, finalmente el servidor se encarga de emitir los resultados a la petición del cliente.

Cada sistema de búsquedas en catálogo cuenta de las siguientes partes.

Forma de captura en WEB

Realiza mediante palabras que el usuario provee desde una forma de *Web*, búsquedas con querys sencillos tipo "*and* " y tipo "*or*" a una base de datos que se encuentra alojada en un servidor.

La interfaz de este tipo de sistemas es gráfica e interactiva por medio de formas, apoyadas por programas CGI y su despliegue *por word wide web*.

Este servicio consiste en una forma con un campo y varias opciones de búsqueda, la información que contendrá el campo será proporcionada por el usuario y consiste en una o unas palabras a buscar en la base de datos, las opciones de búsqueda consisten en búsquedas tipo and y tipo or, es decir las palabras exactamente como se escribe o alguna de las palabras escritas

Problemática.

El servicio de búsqueda es una de los mas demandados a últimas fechas por instituciones de la UNAM y en general, el tiempo de desarrollo para cada uno de los sistemas entre configuración y adecuación de la base de datos es aproximadamente de 4 semanas.

La base de este servicio es el protocolo z3950 Isearch (software de tipo gratuito de CNRI) y programas en Perl versión 5 (software de tipo gratuito), Inicialmente se cuenta con un grupo de programas que forman parte del servicio, a continuación se describen:

- Programa de conversión al formato z3950.
- Forma principal HTML: forma de consultas CGI de búsquedas: encargado de responder a la forma principal y hacer las peticiones a lsearch.
- CGI de resultados: es invocado por el CGI principal y se encarga de general los resultados.
- Avisos HTML de respuesta.
- Programa encargado de borrar los archivos temporales.

La problemática de este servicio se presenta por la necesidad de reconfigurar rutas, permisos, estructuras de archivo, creación de nuevos directorios, conversión e indexación de cada base de datos, nueva generación de la forma principal y sus nuevos campos, verificación del funcionamiento de los nuevos programas CGI y pruebas del sistema completamente integrado.

Sistema generador de servicios de búsqueda para bases de datos en *Web*

Con la finalidad de reducir el tiempo de respuesta para proporcionar el servicio de búsqueda, además de optimizar de los recursos utilizados, se propuso un sistema de tipo genérico, que se encargará de resolver todas las labores reconfiguración para cada sistema.

El sistema genérico realizado completamente en Perl, realizará las siguientes actividades:

- Transformación de la base de datos txt a bdf, formato utilizado por el protocolo z3950.
- Indexación de la base de datos, por medio de utilerías de z3950
- Generación de la forma principal, las formas de respuesta y los programas CGI correspondientes, pidiendo al usuario los datos necesarios,
- Adecuación de los permisos y estructuras de archivos necesarias, además de colocar los diferentes programas en sus respectivos directorios html-docs y cg-bin.

2. Datos técnicos del sistema.

El sistema genérico primera versión, fue desarrollado para las plataformas de Sun Solaris 5.5.1, Silicon Graphis IRIX 5.3, y Linux versión

Para la versión inicial, se decidió que el genérico contara con sus propios recursos (lindex, lsearch y Perl5.0) y que así, no dependiera de recursos previamente instalados en el servidor o por instalar. Con motivos de optimización de los recursos en el sistema, el programa de configuración de genérico, proporciona una opción de borrado los programas autocontenidos, si ya se cuenta con las versiones necesarias en el servidor (para mayor referencia ver manual de uso).

Entradas del sistema: la base de datos para realizar las búsquedas, las configuraciones de rutas pertinentes (ligas de configuración al servidor de *Web* y rutas físicas de la cuenta).

Salidas del sistema: El sistema proporcionará las formas generales en HTML para realizar las búsquedas, así como sus respectivos programas CGI para el funcionamiento de la misma. Adicionalmente proporciona un programa que encarga de realizar limpieza de archivos no necesarios. El sistema se encarga de proporcionar los permisos adecuados para el funcionamiento, y dar como resultado un sistema funcional para consultar por *Web* por medio de un Navegador.

El formato específico que debe tener la base de datos al entrar al sistema genérico es enunciado a continuación, éste no debe tener ningún tipo de error, ya que de lo contrario repercutirá en el mal funcionamiento del sistema de búsqueda.

Formato proporcionado por el usuario:

- La base de datos deberá tener un nombre con extensión .txt
- Será colocada en el directorio adecuado (ver manual de uso)
- Dicha base contendrá cada registro separado entre sí, por un salto de línea en blanco "\n"
- Cada nombre de campo y su respectiva información estará separado por dos puntos
- El contenido de cada campo tendrá que estar a renglón corrido, sin ningún tipo de salto de línea.
- Cada campo contenido en el registro deberá de estar separado con el siguiente, por un salto de línea ""\n"
- El archivo preferentemente será un archivo tipo UNIX, en su defecto será un archivo de texto plano, sin diseño de ningún tipo.

Ejemplo de un catálogo con tres registros y un numero variable de campos en cada uno, en un archivo.txt:

TITULO: A LA PUERTA DEL INFIERNO.
REALIZADOR: ANONIMO.
ANIO: 1990.
PAIS: MEXICO, VENEZUELA.
PRODUCCION: TELEVISA.

TITULO: A POCO ES DOMINGO.
REALIZADOR:
ANIO: 1990.
PAIS:
PRODUCCION: INDEPENDIENTE.

TITULO: ABUELITA DE BACKMAN.
ANIO: 1990.
PAIS: MEXICO, VENEZUELA, ARGENTINA, BRASIL, EGIPTO, CANADA,...ETC .
PRODUCCION: MANERO TABOADA.
TIPO DE PRODUCCION: INDEPENDIENTE.

NOTA: si los campos son nulos, podemos poner las etiquetas o simplemente no ponerlos. También incluso ponerlos en desorden, a excepción del primer campo, ya que se necesita que este siempre exista (puede estar solo la etiqueta) y además deberá estar siempre en primer lugar. Esta base de datos deberá de colocarse en el directorio **/baseweb/bases** además de proporcionar los permisos adecuados para su **uso > chmod 755 nombre.txt**

lindex versión 1.20 compilada para Solaris 5.5.1

lsearch versión 1.20 compilada para Solaris 5.5.1

Perl versión 5.001 compilada para Solaris 5.5.1

notas:

El uso de otra versión de lserch y de lindex no garantiza el buen funcionamiento.

Para las versiones de Perl deberá ser Perl 5 o posteriores

3. ¿Cómo migrar a otras plataformas?

Es posible el migrar fácilmente este sistema a otras plataformas UNIX (diferentes versiones en sistemas operativos de Silicon, Solaris, LINUX) lo único que necesito es compilar las siguientes utilerías y colocarlas en el lugar indicado.

Perl versión 5 o posterior.	colocación: ../generico/utilerias/ cualquier otra ruta, es configurable
Isearch (z3950) versiones 1.14 y anteriores	colocación: .../generico/utilerias/
Isearch deberá contener lindex debe aceptar el formato SGMLNORM o en su defecto SGMLTAG, pero deberá ser configurado editando el archivo generador y cambiando en la opción de indexación el tipo de formato.	colocación: .../generico/utilerias/
Recomendación: hacer pruebas que aseguren el buen funcionamiento de Z 3950	

MANUAL DE USO: SISTEMA AUTOMÁTICO GENÉRICO

Este manual tiene el propósito de guiarle en el uso adecuado del sistema generador de servicios de búsqueda, le guiará paso a paso lo que debe hacer para obtener el resultado óptimo, cualquier duda o sugerencia comunicarla a Eva Edith López López eva@servidor.unam.mx, gracias.

1. Verificar que la versión del sistema genérico sea la adecuada al sistema que utilizará en la cuenta de uso.

2. Descomprima la versión requerida según la plataforma correspondiente a la cuenta de usuario asignada al servicio. Para las siguientes plataformas:

   - Silicon graphics (lince): genIRIX_ver5.3.tar
   - Sun (e4000): genSolaris_ver5.5.1.tar
   - Linux : genLinux_ver .tar

3. Descomprimir el archivo genérico. Al descomprimirse el archivo generara una estructura de archivos, con un directorio inicial **/generico** a partir de este directorio genérico trabajará.

4. Verificar que se tenga la base de datos con el formato adecuado. (si hay dudas del formato verificar en la descripción técnica)

5. Colocar la base o las bases de datos en el directorio **/generico/baseweb/bases/** con extensión txt, ya que el sistema detecta solo las bases con extensión ".txt"

6. Tenga a la mano las rutas de configuración del *home* de su cuenta, las rutas de donde se encuentran los directorios del servidor de *Web* y de *cgi-bin*, además sus respectivas ligas al servidor de *Web*.

   - Cambiar al subdirectorio /generico encontrará un archivo llamado *genlinux.pl*

     ejecute> genlinux.pl

   - Si no se ejecuta de forma autónoma, ejecútelo con la ruta de Perl (verificar la ruta del interprete de Perl)

ejecute>/usr/bin/perl genlinux.pl

Al ejecutarlo le mostrará la estructura de archivos que el sistema genere, presione cualquier tecla para continuar.

7. Configure las rutas que el sistema le pide, es importante poner las rutas correctamente, de lo contrario no encontrará sus recursos.

   ejemplo:

```
***** CONFIGURACIÓN DE RUTAS ******
Dar la ruta de home [incluyendo el directorio base generico]
>/home/usuario1/generico
Dar la ruta del directorio html-docs
>/home/usuario1/htdocs
Proporcione la liga del servidor de web html-docs
>/usuario1
Dar la ruta del directorio cgi-bin
>/home/usuario1/cgi-bin
Dar la liga del servidor de Web para CGI
>/cgi-bin/usuario1
```

8. El siguiente mensaje que envía el sistema, se refiere a que la versión del sistema genérico cuenta con su propia versión de Perl 5 en el directorio ***/generico/utilerias/.*** Se recomienda que si ya se cuenta con una versión de Perl5 (o posterior) instalada en el servidor, trabaje con ella y cuando el sistema le de opción borre la adicional.
9. Si trabaja con la versión propia del servidor deberá configurar la ruta de ejecución de Perl cuando el sistema lo indique,
   por ejemplo:

```
Dar la ruta de ejecución de PERL en el servidor.
>/var/opt/bin/perl
```

10. A continuación el sistema le mostrará las rutas que configuró, verifique si son correctas, si no lo son, el sistema dará oportunidad de reconfiguración de las mismas.
11. Configuración de campos es el siguiente paso, deberá de indicar el nombre del campo por el que realizarán las búsquedas e indicar el rotulo con el que aparecerá en la interfaz de usuario.

Para realizar las búsquedas, no es necesario el poner todos los campos que tiene la base de datos, solo en los que se necesiten realizar búsquedas.

Es importante que los campos se capturen tal y como aparecen en la base de datos, esto evitará muchos posibles errores

Para los rótulos de cada campo en la interfaz de usuario, capturen el texto que prefiera aparezca y además se debe de poner el código HTML para los acentos o caracteres especiales.

12. Una vez configurados todos estos datos, Felicidades el sistema esta concluido y puede verificarse directamente en Web, por medio de un navegador con el URL asignado según la cuenta de usuario por ejemplo:
http://www.unam.mx/filmoteca/filmo.html

MANUAL DE USO: SISTEMA DE ACTUALIZACIÓN AUTOMÁTICA DE CATÁLOGO DE DATOS

Este breve manual pretende proporcionarle una guía rápida para orientarle en los pasos a seguir para actualizar el catálogo de datos de su Sistema de búsqueda para catálogos publicados en Web.

COLOCAR LA NUEVA BASE DE DATOS :

1. Ya que tiene su base datos (catálogo de datos en archivo.txt) con el formato adecuado (ver archivo FORMATO), mediante FTP de windows transfiera la cuenta en el servidor, colocándola en el directorio

   /usr/users/filmoteca/filna/bases

**nota. Le recomendamos tener respaldo y borrar todos los archivos que en el directorio /usr/users/filmoteca/filna/bases antes de copiar la nueva base (por cuestiones de espacio)

Una vez borrados los archivos existentes, transfiera la nueva base de datos que deberá tener la extension .txt (esto, es muy importante) no importando el nombre del archivo.

REALIZAR LA ACTUALIZACIÓN:

2. Si usted trabaja en una PC, realice una conexión al servidor.

   Telnet dragon.dgsca.unam.mx

3. Pedirá el servidor el nombre de la cuenta y la contraseña.

4. Cambie al directorio filna

   ```
   >cd filna
   ```

5. En este directorio encontrará el archivo de actualización **ACTUALIZA.pl**

6. Teclee el nombre de archivo:

   ```
   >ACTUALIZA.pl
   ```

7. Enseguida aparecerá el siguiente mensaje:

   Dar el nombre de la base de datos>

   Teclee el **nombre del archivo** (catalogo a actualizar) que colocó en el paso 1 y enseguida teclee **enter.**

8. Espere unos cuantos minutos a que el sistema automático actualice el archivo y los indices del catalogo, el sistema le enviara un mensaje de final del procedimiento.

Dudas: eva@servidor.unam.mx

Coordinación de servicios de red, Admón. De Servicios Institucionales. Tel 6228522

DGSCA-DCAA

Referencias

Tema: Internet y Word Wide Web.

http://www.cs.indiana.edu/docproject/zen/zen-1.0_toc.html

http://www.isoc.org/internet-history/

http://members.magnet.at/dmayr/history.htm

historia e impacto en la red http://www.columbia.edu/~hauben/netbook/

http://info.med.yale.edu/caim/manual/contents.html

http://www.fnc.gov/Internet_res.html

http://ourworld.compuserve.com/homepages/dmayr/history.htm

Centro de Información de REDUNAM http://www.nic.unam.mx/REDUNAM/historia.html

http://www.dgsca.unam.mx/

http://www.w3.org/People/Berners-Lee/1996/ppf.html

http://www.mnh.si.edu/collections.html.

http://wmaestro.com/webmaestro/

http://www.internet2.unam.mx/

http://www.internet2.edu.mx/

http://www.internet2.edu

Tema: Perl

http://www.perltk.com/

ftp://sunsite.dgsca.unam.mx/pub/lenguajes/Perl/manuales/tutorial_perl.html

http://www.perl.com/

http://www.republic.perl.com/

ftp://ftp.rge.com/pub/languages/perl/CPAN.html

Tema: z.3950

http://lcweb.loc.gov/z3950/

http://vinca.cnidr.org/software/lsearch/

http://www.etymon.com/pub/software/lsearch/

ftp://ftp.cnidr.org/pub/NIDR.tools/lsite/

ftp://ftp.cnidr.org/pub/software/lsite/

Tema: Software Libre

http://www.apache.org/

http://www.gnu.org/philosophy/categories.html

http://www.gnu.org/gnu/manifesto.html

http://www.debian.org/social_contract

http://www.opensource.org/

http://www.shareware.com/

http://www.freeware.com/

http://www.tucows.com/

http://shareware.netscape.com/computing/shareware/

http://www.iac.com.mx/computo_98/

http://www.iac.com.mx/computo_98/gnu-library.html

http://www.iac.com.mx/computo_98/apache.html

http://www.iac.com.mx/computo_98/bsd.license.html

http://www.iac.com.mx/computo_98/comercial.html

http://eros.pquim.unam.mx/~aspuru/softwarelibre/

http://euch6h.chem.emory.edu/services/aix-faq/c4.index.html

http://www.sgmlu.com/

Bibliografía

Client/server strategies, William Marion, McGraw-Hill Series on Computer Comunications, *Toronto E.U.,1994, 311 pp.*

Client/Server System Design & Implementations, Larry T.Vaughn, McGraw-Hill Series on Computer Comunications, Toronto E.U., 1994, 311 pp.

Managing Cliente/server Environments, Tools and strategies for building Solutions, John McConnell, Prentice Hall, New Jersey, E.U., 1996. 360 pags.

Wais and Gopher Servers, A guide for Internet end-users, Eric Lease Morgan, Mecklermedia, London U.K., 1994,118 pp.

Perl a través de ejemplos, David Medinets, Prentice Hall-QUE, México 1997, 657 pp.