UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
"A R A G O N"

INTEGRACION DE HERRAMIENTAS PARA LA RED LATINOAMERICANA DE ENFERMEDADES METABOLICAS HEREDITARIAS Y SU PUBLICACION EN INTERNET

T E S I S

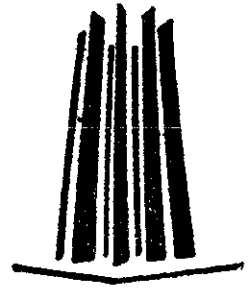
QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A :

CESAR FRANCISCO GERMAN ROSAS

Incluye disquete



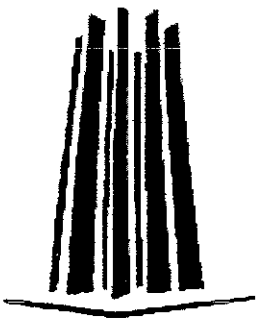
SAN JUAN DE ARAGON, NEZAHUALCOYOTL, ESTADO DE MEXICO

1999

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ARAGÓN

Integración de herramientas para la Red Latinoamericana de Enfermedades Metabólicas Hereditarias y su publicación en Internet



Tesis que para obtener el título de:
Ingeniero en Computación

Presenta:
César Francisco Germán Rosas

Director de tesis: Ing. Donaciano Jiménez Vázquez

San Juan de Aragón, Nezahualcoyotl, Estado de México. ~~1998~~

1999

Integración de herramientas para la Red Latinoamericana de Enfermedades Metabólicas Hereditarias y su publicación en Internet

Agradecimientos

Gracias Infinitas...

A **Dios**, por tenerme bajo su luz.

A la **Universidad Nacional Autónoma de México**,
por ser la más grande, por ser la mejor.

A mis pa´s **Lourdes** y **Manuel**, por la oportunidad de vivir, por su tiempo,
por su esfuerzo, por sus enseñanzas, por todo.

A **Karen**, por nacer, por ser mi inspiración.

A mis ángeles, **Manuel**, **Jesús** y **Fernando**.

A **Celia** y **Ma. de Jesús**, por su ejemplo de fuerza.

A **Martha**, **Maru**, **Ade** y **Paco**, por su amistad y apoyo.

A ti, por dejarme amarte, por esos momentos.

Contenido

Prólogo

Sin duda la globalización que se vive actualmente, en todos los ámbitos de la vida, ha determinado una modificación mas que significativa de la conducta humana. El mundo ya no es el mismo, hace solo unos cuantos años, el mundo estaba dividido por barreras casi infranqueables. Ahora el mundo pretende ser uno solo, las fronteras ya nos son geográficas, ahora son ideológicas, el ser humano se agrupa por afinidades. Se pretende interactuar en igualdad de circunstancias, aunque existan factores económicos, políticos, sociológicos, religiosos, etc. que entorpecen la materialización del ideal humano "ser todos iguales".

Uno de los pilares más importantes de la globalización es Internet. Este gran avance tecnológico ha logrado lo inimaginable: entrelazar a todos los países del mundo y permitir una comunicación directa entre los hombres.

En tan solo 10 años, se pasó de una comunicación complicada, básicamente por la tecnología disponible; limitada, en la cantidad de servicios que se ofrecían; y la manera en que el usuario podía disponer de la información, solo texto, que carecía del impacto visual de las imágenes.

Con el surgimiento de la tecnología Web, se integró, a la creciente telaraña de máquinas distribuidas alrededor de todo el mundo, la capacidad de transmitir imágenes, audio y video. Esto, aunado a los servicios ya disponibles, permite tener, por fin, la tan anhelada interrelacion humana a través de la tecnología.

Como ya se mencionó, las necesidades de cada persona varían por diversos factores. Siguiendo el anterior principio, al surgir grupos de personas afines, surge implícitamente la necesidad de comunicarse, para interactuar, en todas las personas del grupo.

El presente trabajo es un esfuerzo por describir las herramientas básicas para la creación de un sito en el Internet de comunicación integral. Se basa en el desarrollo práctico del sistema de información REDLAEM. Este sistema integra todas las herramientas para permitir que los Especialistas Latinoamericanos en Enfermedades Metabólicas Hereditarias se comuniquen y compartan experiencias, que no en pocos casos, pueden salvar vidas, principalmente de niños.

REDLAEM es solo uno de los cientos de miles de grupos que existen en el mundo. Los médicos especialistas en estas enfermedades, encuentran en REDLAEM la posibilidad de interactuar.

En el **capítulo 1** se explican algunos conceptos básicos para entender lo que es Internet y su historia. Conceptos como red, protocolos puertos son abordados de una manera breve y sencilla. Finalmente, se presenta una introducción al Web.

En el **capítulo 2** se describen amplia y claramente las herramientas computacionales necesarias para crear y mantener un sistema de información en el Web. Herramientas como HTML, HTTP, *Servidor Web, lenguajes de programación y* SQL SYBASE son abordadas de manera que se tenga una idea general y sólida de lo que se requiere para elaborar un sitio Web determinado.

En el **capítulo 3** se plantean los antecedentes del proyecto REDLAEM y se describe la manera, tradicional, en que se comunicaban los especialistas médicos antes de la creación de REDLAEM en Web y como se soluciona con el sistema.

En el **capítulo 4** se realiza el análisis del proyecto, se intenta seguir alguna metodología (no establecida para sistemas Web), y se consideran etapas como *objetivo, planteamiento y diseño.*

En el **capítulo 5** se cumple con la etapa de la metodología, *el desarrollo.* Se describe la estructura general del sitio, la manera en que funciona el sistema y se presenta una descripción amplia del sistema, pantallas de interfaz con el usuario y los programas que se ejecutan. Además se anexa un disquete que contiene el código fuente de todos los programas que forman el sitio de REDLAEM en Web.

1. Introducción

1.1 Historia y panorama general.

Actualmente es difícil imaginar cómo alguien puede perder la oportunidad de aprender por lo menos algo acerca de World Wide Web e Internet. Los periódicos, revistas y medios de difusión presentan a Internet con frecuencia. A menudo, se ven direcciones de páginas Web en comerciales de televisión y anuncios impresos. Las Universidades, los negocios y otras instituciones se han preocupado por formar parte de Web, por su parte los empresarios han aprovechado esta fiebre para instalarse en las laderas de la supercarretera de la información, vendiendo de todo, desde conexiones a Internet hasta asesoría para la creación de páginas Web y conferencias relacionadas con Web.

La explosión de interés en Internet está impulsada por un crecimiento aún más explosivo de Web. Sin embargo, Internet estuvo aquí antes y ha estado durante más de veinte años; Podemos definirla como el conjunto de redes y computadoras de todo el mundo que se interconectan por medio de TCP/IP (Protocolo de Control de Transmisión / Protocolo Internet).

La Agencia de Proyectos de Investigación Avanzada, (ARPA, por sus siglas en inglés) comenzó a trabajar con una tecnología de red de redes a mediados de los años setenta; su arquitectura y protocolos tomaron su forma actual entre 1977 y 1979. En ese tiempo, ARPA era conocida como la principal agencia en proporcionar fondos para la investigación de redes de paquetes conmutados y fué pionera de muchas ideas sobre la conmutación de paquetes con su conocida ARPANET. ARPA también ofreció fondos para la exploración de conmutación de paquetes a través de redes de radio y mediante canales de comunicación por satélite.

La disponibilidad de ARPA en cuanto a fondos para la investigación, atrajo la atención y la imaginación de muchos grupos de investigación, en especial de los investigadores que ya tenían experiencia previa utilizando conmutación de paquetes en ARPANET. ARPA llevo a cabo reuniones informales de investigadores para compartir ideas y discutir los resultados de los experimentos. 1979, había tantos investigadores involucrados en los esfuerzos, que ARPA forma un comité informal para coordinar y guiar el diseño y la arquitectura del Internet que surgía.

La Internet global se inició alrededor de 1980 cuando ARPA comenzó a convertir las máquinas conectadas a sus redes de investigación en máquinas con el nuevo protocolo. ARPANET se convirtió en la columna vertebral del nuevo Internet. La transición hacia la tecnología Internet se completó en enero de 1983, cuando la Oficina del Secretario de Defensa ordenó que todas las computadoras conectadas a redes de largo alcance utilizaran el nuevo protocolo TCP/IP. Al mismo tiempo, la Agencia de Comunicación de la Defensa (DCA), dividió ARPANET en dos redes separadas, una para la investigación futura y otra para la comunicación militar. La parte para la investigación conservó el nombre de ARPANET; la parte militar, que era un poco más grande, se conoció con el nombre de MILNET.

Para alentar a los investigadores universitarios a que adoptaran y utilizaran los nuevos protocolos, ARPA puso a su disposición una implementación de bajo costo. En ese tiempo, la mayor parte de los departamentos universitarios de ciencias de la computación utilizaban una versión del sistema operativo UNIX. Al proporcionar fondos a Bolt Beranek de Newman, Inc., para implementar sus protocolos en la utilización de UNIX y al proporcionar fondos para la integración de los protocolos al sistema operativo UNIX, ARPA fué capaz de llegar a más del 90% de los departamentos universitarios de ciencias de la computación.

En 1986 aumentaron los esfuerzos para el enlace de redes al proporcionar fondos para una nueva red de columna vertebral de área amplia, llamada NSFNET, que eventualmente alcanzó todos los centros de supercomputadoras y los unió a ARPANET. Por último, en 1986, la NSF proporcionó fondos para muchas redes regionales, cada una de las cuales conecta en la actualidad importantes instituciones científicas de investigación en cierta área.

La mayoría de la gente está familiarizada con la idea general de una red de computación: varias computadoras en una oficina u otro ambiente común, conectadas por medio de cables para permitir un uso compartido de impresoras y archivos, o bien, que haya una comunicación entre ellas. La idea de Internet es muy similar, solo que a una escala mucho mayor, y también tiene un elemento adicional importante. La conectividad de red de TCP/IP no sólo permite la conexión de computadoras locales entre sí, sino también que las redes se conectan con otras redes. Estas conexiones crean internets, en las cuales, para los usuarios parece que las computadoras de todas las redes conectadas son parte de una sola y enorme interred.

Las redes interconectadas no necesitan estar en la misma ubicación ni en el mismo edificio; pueden estar en lugares remotos entre sí, desde el punto de vista físico, con conexiones que utilizan líneas de datos de propósitos especiales, radio por satélite, enlaces de radio infrarrojo, por cable o incluso líneas telefónicas ordinarias y módems.

1.2 Red.

Una red es una colección de computadoras y otros dispositivos interconectados que pueden enviar y recibir datos desde o hacia cualquier otra, casi en tiempo real. Una red está normalmente conectada mediante cables, y los bits de datos son convertidos en electrones que viajan a través de los cables. Además, redes inalámbricas que transmiten los datos a través de rayos infrarrojos o microondas están comenzando a aparecer, y muchas de las transmisiones a larga distancia ahora se realizan mediante cables de fibra óptica, se mandan rayos de luz visible a través de filamentos de vidrio.

Cada máquina conectada a la red es llamada un nodo. Muchos nodos son computadoras; pero, impresoras, ruteadores, bridges, gateways y terminales tontas también son nodos. Los nodos que son computadoras que tienen todas las capacidades también se llaman hosts. Es conveniente referirse como nodo a cualquier dispositivo de la red, y como host a las computadoras de propósito general que controlan lógicamente a otros nodos.

Cada nodo de la red tiene una dirección: una serie única de bytes que lo identifican.

Existen dos direcciones para cada nodo: una dirección lógica (IP) formada por 4 bytes, que es asignada por el organismo internacional dedicado a administrar las direcciones de todos los nodos de Internet, el InterNIC (Internet Network Information Center). Este organismo asigna las direcciones dependiendo del tamaño de la red particular y las subredes que la conforman. Así tenemos que existen clases de redes: clase A, para redes con 2^{24} hosts; clase B, para redes con 2^{16} hosts, clase C, para redes con 2^{8} hosts. Actualmente están disponibles solo clases B y C. Un bloque clase C especifica los tres primeros bytes para la dirección y permite 254 nodos. Un bloque

clase B especifica los dos primeros bytes para la dirección y permite 65536 nodos. Estas direcciones se escriben normalmente en el formato 132.248.71.14, donde cada uno de los cuatro números es uno de los bytes sin signo entre al rango de 0 a 255. Y una dirección física (MAC Media Access Control) formada por 6 bytes, los tres primeros bytes son asignados por la IEEE a cada fabricante de tarjetas Ethernet, y los 3 últimos son asignados por el fabricante, asegurando así una única dirección MAC para cada tarjeta fabricada en el mundo.

En algunas redes, los nodos tienen asignados nombres, con algún significado humano, asociados a las direcciones IP. De cualquier manera, los nombres pueden cambiar en cualquier momento y ser asignados a otros nodos. Esta información es almacenada como una tabla de datos en un servidor de nombres (DNS Domain Name System). Este sistema fué desarrollado para convertir los hostnames en direcciones numéricas. Este servicio distribuye los nombres simbólicos a través de peticiones entre la comunidad de servidores en Internet.

Todas las redes de computadoras son de switcheo de paquetes. Esto quiere decir que los datos que viajan en la red, son divididos en varias partes llamados paquetes, y cada paquete es manejado individualmente. Por esta razón, cada paquete contiene información sobre quien lo envió y hacia donde va, además de otros datos. La ventaja más importante de dividir los datos en paquetes individuales es que se pueden enviar datos desde muchos nodos a la vez sin interferencia en un solo cable. En cambio, cuando se hace una llamada telefónica, se reserva una línea individual mientras dure la llamada. Otra ventaja es que se puede detectar si el paquete sufrió o no daños en el trayecto.

Para lograr la comunicación entre los nodos de una red, y lograr el switcheo[1] de paquetes, existen conjuntos precisos de reglas que definen como se logra la

[1] El switcheo de paquetes consiste en formar pequeños grupos de datos, y enviarlos a través de la red por distintas rutas, a fin de reducir el tiempo de transmisión.

comunicación entre máquinas y el intercambio de información. Estas reglas se conocen como protocolos.

1.3 Suite de protocolos TCP/IP.

Como se mencionó anteriormente, el protocolo que hace posible que todas las máquinas conectadas a Internet se comuniquen en el protocolo TCP/IP. Este protocolo es el más robusto de todos; desde su concepción, se contempló que cumpliera con las especificaciones del modelo OSI (Open Systems Interconnection), que son los protocolos, específicamente estándares de ISO (International Organization for Standarization), que es la organización internacional que bosqueja, discute, propone y especifica estándares a nivel mundial, para la interconexión de sistemas de computadoras. Aunque son diferentes el protocolo TCP y el protocolo IP, se acordó nombrar al conjunto de protocolos del modelo OSI como: TCP/IP.

Idealmente, el modelo OSI está formado por siete capas:

- Capa Física.
- Capa de Enlace de datos.
- Capa de Red.
- Capa de Transporte.
- Capa de Sesión.
- Capa de Presentación.
- Capa de Aplicación.

En la práctica, TCP/IP reúne estas características, pero solo se consideran cinco capas:

- Capa física (Hardware)
- Capa de Interfaz de Red (protocolos ARP y RARP).
- Capa Internet (protocolos IP e ICMP).
- Capa de Transporte (protocolos TCP y UDP).
- Capa de Aplicación (protocolos TELNET, TFP y SMTP).

La figura 1.1 presenta una comparación más precisa entre los modelos OSI y TCP/IP.

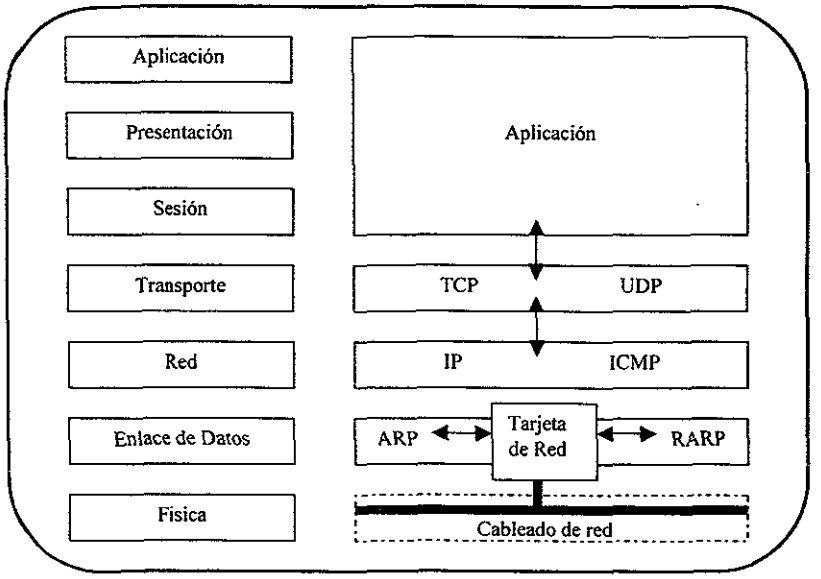


Figura 1.1 Modelos OSI y TCP/IP

1.3.1 ARP y RARP.

El Protocolo de Asociación de Direcciones (ARP) permite que un nodo encuentre la dirección física de otro nodo dentro de la misma red física con solo proporcionar la dirección IP de su objetivo.

En el arranque del sistema, una computadora que no tenga un disco permanente debe contactar a un servidor para encontrar su dirección IP antes de que se pueda comunicar por medio del TCP/IP. El Protocolo de Asociación de Direcciones por Réplica (RARP), utiliza el direccionamiento físico de red para obtener la dirección de red de redes de la máquina. RARP proporciona la dirección física de la máquina de destino para identificar de manera única el procesador y transmite por difusión la solicitud RARP. Los servidores en la red reciben el mensaje, buscan la transformación en una tabla y responden al transmisor. Una vez que una máquina obtiene su dirección IP, la guarda y no vuelve a utilizar RARP hasta que se inicia de nuevo.

1.3.2 IP e ICMP.

El protocolo que define el mecanismo de entrega sin conexión y no confiable es conocido como Protocolo Internet (IP). El protocolo IP proporciona tres definiciones importantes. Primero, define la unidad básica para la transferencia de datos utilizada a través de una red de redes TCP/IP. Es decir, especifica el formato exacto de todos los datos que pasarán a través de la red. Segundo, el software IP realiza la función de ruteo, seleccionando la ruta por la que los datos serán enviados. Tercero, además de aportar especificaciones formales para el formato de los datos y el ruteo, el IP incluye un conjunto de reglas que le dan forma a la idea de entrega de paquetes no confiable. Las reglas caracterizan la forma en que los anfitriones y ruteadores deben procesar los paquetes, cómo y cuándo se deben generar los mensajes de error y las condiciones bajo las cuales los paquetes pueden ser descartados. El IP es una parte fundamental

del diseño de la red de redes TCP/IP, que a veces se conoce como tecnología basada en el IP.

En el sistema sin conexión que se ha descrito, cada ruteador opera de manera autónoma , ruteando o entregando los datagramas que llegan sin coordinarse con el transmisor original. El sistema trabaja bien si todas la máquinas funcionan de manera correcta y si están de acuerdo respecto a las rutas. Por desgracia, ningún sistema funciona bien todo el tiempo. Además de las fallas en las líneas de comunicación y en los procesadores, el IP tiene fallas en la entrega de datagramas[2] cuando la máquina destino está desconectada temporal o permanentemente de la red, o cuando los ruteadores intermedios se congestionan tanto que no pueden procesar el tráfico entrante.

Para permitir que los ruteadores de una red de redes reporten los errores o proporcionen información sobre circunstancias inesperadas, los diseñadores agregaron a los protocolos TCP/IP un mecanismo de mensajes de propósito especial. El mecanismo, conocido como Protocolo de Mensajes de Control Internet (ICMP), se considera como parte obligatoria del IP y se debe incluir en todas las implementaciones IP.

El ICMP permite que los ruteadores envíen mensajes de error o de control hacia otros ruteadores o anfitriones; el ICMP proporciona comunicación entre le software del Protocolo Internet en una máquina y el mismo software en otra.

[2] Se denomina de esta forma a los bloques de datos en los que se ha dividido la información original para ser enviada a través de la red.

1.3.3 TCP y UDP.

Para mejorar el esquema básico, TCP (Transmission Control Protocol), fué agregado para que cada conexión tuviese la capacidad de checar la integridad de los datagramas IP y pedir su retransmisión en caso de haberse perdido. Más aún, TCP permite unir y ordenar los paquetes recibidos.

Cuando el orden de los datos no es particularmente importante, y cuando la perdida de algunos paquetes no afecta el flujo de dados, los paquetes son enviados sin las garantías que TCP ofrece. Esto se hace utilizando UDP (User Datagram Protocol), UDP es un protocolo inseguro que no garantiza que los paquetes lleguen a su destino, o que lleguen en orden. No obstante esto debe ser un problema para el uso de algunas aplicaciones como la transferencia de archivos, esto es perfectamente aceptable para las aplicaciones en que la perdida de algunos datos no es percibida por el usuario, como una señal de audio o video, que no se distorsionará significativamente.

1.3.4 TELNET, FTP y SMTP.

El conjunto de protocolos TCP/IP incluye un protocolo de terminal remota sencillo, llamado TELNET. TELNET permite al usuario de una localidad establecer una conexión TCP con un servidor de acceso a otro. TELNET transfiere las pulsaciones de teclado directamente desde el teclado del usuario a la computadora remota como si hubiesen sido hechos en un teclado unido a la máquina remota. Telnet también transporta la salida de la máquina remota de regreso al pantalla del usuario. El servicio se llama transparente porque da la impresión de que el teclado y el monitor del usuario están conectados de manera directa a la máquina remota.

Dado un protocolo de transporte confiable de extremo a extremo como el TCP, la transferencia de archivos podría parecer trivial. Sin embargo, los detalles de

autorización, el nombre y la representación entre máquinas heterogéneas hace que el protocolo sea complejo. El Protocolo de Transferencia de Archivos (FTP, File Transfer protocol) ofrece muchas facilidades que van más allá de la función de transferencia misma, dicha facilidades son:

- *Acceso interactivo.* Proporciona una interfaz interactiva que permite a las personas interactuar fácilmente con los servidores remotos.

- *Especificación de formato.* El FTP permite al cliente especificar el tipo y formato de datos almacenados o contenidos en los archivos.

- *Control de autenticación.* El FTP requiere que los clientes se autoricen a si mismos con el envío de un nombre de conexión y una clave de acceso al servidor antes de pedir la transferencia de archivos.

Además del formato de los mensajes, el conjunto de protocolos TCP/IP especifica un estándar para el intercambio de correo entre las máquinas. Es decir, especifica el formato exacto de los mensajes a un cliente en una máquina que lo utiliza para transferir correo hacia el servidor en otra. El protocolo de transferencia estándar se conoce como SMTP, Simple Mail Transfer Protocol (Protocolo de Transferencia de Correo Simple). El protocolo se enfoca específicamente en cómo transfiere el sistema de entrega de correo subyacente los mensajes a través de un enlace en una máquina a otra. No especifica de qué manera acepta el sistema de correo los mensajes de correo de un usuario o cómo presenta al usuario la interfaz de usuario el correo entrante. El SMTP tampoco especifica en qué forma se almacena el correo o con qué frecuencia el sistema de correo trata de enviar mensajes.

1.4 Puertos.

Una dirección puede ser todo lo que se necesite si cada computadora no realiza más de una cosa a la vez. Pero las computadoras actuales realizan muchas cosas al mismo tiempo. El correo electrónico debe estar separado de las peticiones FTP, que necesitan estar separadas del tráfico Web. Esto se logra con los *puertos*. Cada computadora con una dirección IP tiene varios miles de puertos lógicos (65,535 para ser precisos). Estos no son más que secciones de memoria asignadas a cada uno de los puertos, y no representan nada físico como un puerto serial o paralelo. Cada puerto es identificado por un número entre 1 y 65535. Cada puerto puede ser asignado a un servicio particular.

Por ejemplo, el servicio HTTP, que es utilizado por el Web, generalmente corre en el puerto 80. SMTP o correo electrónico corre en el puerto 25. Cuando los datos son enviados a un servidor Web en una máquina en particular con una dirección IP, estos son enviados también a un puerto en particular en esa máquina. El receptor revisa cada paquete para ambos, la dirección y el puerto. Si la dirección coincide, los datos son enviados al programa adecuado que los enviará a ese puerto. Esta es la forma en que el tráfico de datos es manejado.

Los puertos entre 1 y 1023 están reservados para servicios conocidos como: finger, FTP, HTTP, y correo electrónico. Algunos servidores Web corren en puertos diferentes al 80, por que múltiples servidores deben correr en una misma máquina o por que la persona que instaló el servidor no tiene privilegios de root[3] necesarios para correrlo en el puerto 80. En máquinas UNIX, está completamente claro la lista de puertos asignados, esto se almacena en el archivo */etc/services*.

[3] Se denomina así al administrador del equipo y de los recursos del mismo. El administrador tiene todos los permisos sobre todos los archivos.

1.5 Introducción a Web.

Como ya se mencionó, antes que Web ya existía Internet, una red mundial de redes interconectadas por medio de TCP/IP . Algunas de las características principales de esta Internet anterior a Web incluyen:

- *Correo electrónico* para enviar mensajes entre usuarios en computadoras remotas.

- *Transferencias de archivos* entre computadoras remotas por medio del Protocolo de Transferencia de Archivos (FTP).

- *Servicio de conexión remota* (TELNET) que permiten a los usuarios conectarse a computadoras remotas y emplearlas como si fueran locales.

Además de estos principales servicios de Internet, se han desarrollado muchos otros en este lapso de tiempo, algunos de los cuales utilizan combinaciones de los servicios mencionados. Los servicios de devolución de correo electrónico transfieren archivos, de manera muy similar a los servicios de respuesta de fax, cuando son solicitados vía correo electrónico. Se han desarrollado listas de correo electrónico de interés especial para personas con preferencias similares que desean entrar en debate sobre algún tema en especial, sin importar la índole del mismo.

Cada uno de estos servicios, y muchos otros, son herramientas útiles y poderosas, y todas con un uso bastante amplio. Incluso antes de la existencia de Web, la necesidad de capacidades de correo electrónico era un impulso sustancial para el crecimiento de Internet. Sin embargo, cada servicio de Internet anterior a Web tiene su propia interfaz de usuario. Muchas de estas interfaces son de fácil manejo para usuarios sin conocimientos técnicos.

En 1993, Tim Berners-Lee y otros investigadores del Laboratorio Europeo de Física de Partículas (CERN[4], por sus siglas en Francés) en Ginebra, Suiza, desarrollaron un medio para compartir datos entre sus colegas con el uso de algo que llamaron *hipertexto*. Los usuarios de CERN podían ver documentos en las pantallas de sus computadoras mediante el nuevo software de *navegador*. Además, algunos códigos especiales incrustados en estos documentos electrónicos permitían a los usuarios saltar de un documento a otro en la pantalla, seleccionando tan solo un *hipervínculo*. Se integraron capacidades para Internet en estos navegadores. De la misma manera como un usuario podía cambiar de un documento de texto en una computadora a otro, podría cambiar de un documento en una computadora a otra computadora remota. Además, cada uno de los principales servicios de Internet antes listados fué agregado al software de navegador. Un investigador podía transferir un archivo desde una computadora remota a su sistema local, o conectarse a un sistema remoto con solo hacer clic en un hipervínculo, en lugar de utilizar los poco inteligibles mecanismos FTP o Telnet. El trabajo innovador del CERN es la base para la World Wide Web actual y su software de navegador y servidor Web fué el primero en su categoría.

El CERN ahora ha vuelto a su misión principal de realizar investigaciones sobre física de partículas, pero su herencia relacionada con Web se ha transferido al World Wide Web Consortium (W^3, un grupo de organizaciones académicas y comerciales dedicadas al avance de Web. El W^3, como se le conoce, sigue activo en el desarrollo de Web, y Berns-Lee sigue al pendiente de los asuntos del W^3.

[4] *Conseil Européen pour la Recherche Nucléaire*

1.6 URLs, URIs, y URNs

Un Uniform Resource Identifier (URI) es el significado de la localización única de un *recurso* en Internet. El recurso es generalmente un archivo, pero también puede ser una dirección de correo electrónico, un programa CGI, o algo más. Existen dos tipos de URIs: Uniform Resource Locators (URLs), y los Uniform Resource Names (URNs). Un URL es un puntero hacia un recurso particular en Internet en una localización particular. Por ejemplo *http://www.unam.mx* o *ftp://zeus.dgsca.unam.mx/pub*. Un URN es un puntero hacia un recurso particular pero sin referencia a una localización particular. La idea detrás de los URNs es que ellos puedan manejar convenientemente recursos que son copiados en diferentes localizaciones o que han sido movidos de un sitio a otro; ellos identifican al recurso en sí, no al lugar en donde se encuentren. Desafortunadamente los URNs se encuentran en una etapa de investigación y desarrollo, y no son usados por el software actual.

El URL especifica el protocolo usado para acceder al servidor (ftp, http), el nombre del servidor, y la localización de un archivo en ese servidor. Un URL típico se ve así: *http://www.unam.mx/redlaem/Admin/historia.html* Esto especifica que existe un archivo llamado *historia.html* en un directorio llamado *Admin* en el servidor *www.unam.mx*, y que ese archivo puede ser accedido vía el protocolo HTTP. La sintaxis de un URL es: *protocolo:// servidor(:puerto)/ruta/archivo(#sección)*

Los URLs que no están completos, pero que tienen partes inherentes a su predecesor se llaman URLs *relativos*. En contraste un URL completo se llama *absoluto*. En un URL relativo, todas las partes que faltan, se asume que son las mismas que corresponden al URL donde se encontró el documento.

2. Herramientas

2.1 HTTP.

HTTP (HyperText Transfer Protocol) es un estándar que define como se comunica un cliente web con el servidor, y como los datos son transferidos de regreso desde el servidor hacia el cliente. HTTP es un protocolo sin estado (no guarda el estado anterior inmediato de una petición). HTTP se auxilia de otros dos estándares: el Multipurpose Internet Mail Extensions (MIME), y el HyperText Markup Language (HTML). MIME es una manera para codificar diferentes clases de datos, tales como sonido y texto, para ser transmitidos a través de una conexión ASCII de 7 bits; esto permite también saber que clase de datos son recibidos y desplegarlos adecuadamente. MIME fué diseñado originalmente para facilitar el envío multimedia de correo electrónico, y permitir que los datos codificados puedan ser enviados a través de cualquier programa de correo. HTML es un estándar simple para describir el valor semántico de datos textuales. Esto significa que se puede decir que "esto es un título", "esto es una lista", etc. Pero es el *navegador* quien interpreta ese formato. HTML es un lenguaje de marcación de *hipertexto* porque incluye una manera para especificar enlaces a otros documentos identificados por un URL.

HTTP y muchos de los protocolos utilizados para transferir datos se valen de programas que se ejecutan constantemente, escuchando peticiones para una conexión o atender un servicio en particular; estos programas se conocen como "*demonio*".

HTTP 1.0 es la versión aceptada actualmente. Usa MIME para codificar los datos. El protocolo básico define una secuencia de cuatro pasos para cada solicitud del cliente al servidor.

1. Realizando la conexión

El cliente establece una conexión TCP con el servidor, en el puerto 80 por defecto; otros puertos pueden ser especificados en el URL.

2. Realizando una petición

El cliente envía un mensaje al servidor solicitando la página especificada en el URL. El formato de esta petición aparece generalmente como:

```
GET /index.html HTTP/1.0
```

GET es una palabra llave. /index.html es un URL relativo a un archivo en el servidor. Se asume que el archivo referido se encuentra en la máquina que recibe la petición. HTTP/1.0 es la versión del protocolo que el cliente entiende. Dos pares de retorno/salto de línea termina la petición, si importar como son terminadas las líneas en la plataforma del cliente o del servidor.

Aunque la línea GET es todo lo que se necesita, una petición puede incluir otra información.

La palabra llave más común es Accept, que indica al servidor que clase de datos puede manipular el cliente. Por ejemplo, las líneas abajo indican que el cliente puede manipular cuatro tipos MIME, estos tipos son: documentos HTML, texto plano, imágenes GIF e imágenes JPEG.

```
Accept: text/html
Accept: text/plain
Accept: image/gif
```

User-Agent es otra palabra llave común, esta indica al servidor cuál browser se está utilizando. Esto permite al servidor enviar archivos optimizados para ese tipo de browser en particular. La línea abajo indica que la petición viene de un browser Lynx versión 2.4:

```
User-Agent: Lynx/2.4 libwww/2.1.4
```

Finalmente, una línea en blanco termina la petición; esto es, dos pares de retorno/salto de línea termina la petición. Un ejemplo de una petición completa es:

```
GET /index.html HTTP/1.0
Accept: text/html
Accept: text/plain
User-Agent: Lynx/2.4 libwww/2.1.4
```

3. La respuesta

El servidor envía una respuesta al cliente. La respuesta comienza con un código de respuesta, seguido por información MIME del encabezado, una línea en blanco y posteriormente el documento solicitado o un mensaje de error (estos errores básicamente son: entre el 200 y 299 indican éxito, entre el 300 y 399 indican redireccionamiento, entre el 400 y 499 indican error en el cliente, y entre el 500 y 500 indican error en el servidor). Asumiendo que el archivo solicitado se localizó, una respuesta típica seria algo como lo siguiente:

```
HTTP/1.0 OK 200
Server: NCSA/1.4.2
MIME-version: 1.0
Content-type: text/html
Content-length: 107

<html>
<Head>
<Title>
Un archivo cualquiera
</Title>
</Head>
<body>
El resto del documento HTML
</body>
</html>
```

La primera línea indica el protocolo que está usando el servidor (HTTP 1.0), seguida por un código de respuesta. OK 200 es el más común, e indica que la petición fué exitosa. Las otras líneas del encabezado identifican el software del servidor (el servidor NCSA versión 1.4.2), la versión de MIME usada, el tipo de contenido MIME (Content-type), y la longitud del documento enviado (sin contar el encabezado), en este caso, 107 bytes.

4. Cerrando la conexión

La conexión puede ser cerrada por el cliente, por el servidor, o por ambos. Así, una conexión de red por separado es usada por cada petición. Si el cliente se conecta de nuevo, el servidor no guarda en memoria la conexión previa o sus resultados. Un protocolo que no guarda en memoria la petición anterior se llama sin estado; en contraste, un protocolo como FTP puede procesar varias peticiones antes de que se cierre la conexión. La falta de estado es una fuerza y una debilidad de HTTP.

HTTP 1.0 no es un éstándar oficial Internet por haber sido desarrollado inicialmente fuera de IETF (Internet Engineering Task Force). HTTP 1.1 es un estándar propuesto y está siendo desarrollado por el W3C y el grupo de trabajo HTTP de la IETF. Este provee una comunicación entre el cliente y el servidor mucho más poderosa y flexible.

Actualmente se está desarrollando un protocolo llamado HTTP-NG (Next Generation). HTTP-NG está diseñado para mejorar el desempeño de HTTP 1.x, primeramente usando el estado. HTTP-NG permite al browser enviar diferentes peticiones en una sola conexión; la conexión permanece abierta hasta que se indica su cierre. Las peticiones y respuestas son asíncronas. El browser no necesita esperar la respuesta para enviar una segunda o tercera petición. De manera similar el servidor puede contestar las peticiones en cualquier orden. Este también puede enviar las respuestas a muchas peticiones en paralelo usando un concepto de la capa de aplicación llamado canales. Un canal es esencialmente una manera de dividir un puerto en subpartes. HTTP-NG está lejos de convertirse en un estándar Internet, pero muestra como los avances en el desarrollo de Internet son muy acelerados y amplios.

2.1.1 Software para un servidor Web.

Sólo dos paquetes de software de servidor Web dominan la participación de UNIX en el mercado: los servidores NCSA httpd (demonio del http) y CERN httpd. El primero, del Centro Nacional de Aplicaciones de Supercomputación (que también desarrollo el navegador Web NCSA Mosaic) tiene, de hecho, la participación más grande que cualquier servidor Web, aproximadamente el 39% del mercado mundial. El paquete del CERN, ahora respaldado por el W^3 Consortium, mantiene casi el 17%. Ambos paquetes están disponibles de manera gratuita.

Recientemente el servidor Apache ha recibido mucho apoyo por parte de NCSA. Apache ha demostrado ser estable, y capaz de soportar módulos de seguridad. Apache también está disponible de manera gratuita.

2.2 HTML y SGML.

HTML es el formato usado para escribir documentos en el Web; se considera como la parte más simple para la interacción entre el usuario y los sitios web. Como se mencionó anteriormente, HTML es un estándar simple para describir el contenido semántico de los datos textuales. Esta idea surgió a partir de un estándar anterior llamado Standard Generalized Markup Language (SGML). HTML es una instancia de SGML. SGML fué creado a principios de los años 70's por el Departamento de Defensa de los Estados Unidos. SGML ahora es un estándar de la International Standards Organization (ISO).

SGML y, por consiguiente, HTML, están basados en la noción de diseñar por significado en vez de por apariencia. El usuario no le indica al *browser* (Navegador web) que se desea que las letras aparezcan de 18 puntos, el usuario solo le indica que se necesita sean del mayor tamaño disponible (<h1> en HTML). Así mismo, no se le

indica que determinada palabra debe aparecer en itálicas, se le indica que se necesita cierto énfasis para esa palabra (en HTML). Al browser se le deja la tarea de determinar la mejor manera de desplegar el texto.

Los tags[5] usados para marcar el texto son insensibles de minúsculas o mayúsculas. Todos los tags deben ser cerrados por otro tag concluyente; este es el mismo tag abierto, pero con una diagonal inicial entre los brackets[6]. Por ejemplo: este texto</STRONG>.

Los tags pueden anidarse[7], pero no traslaparse. La primera línea abajo es un estándar. La segunda no lo es, pero algunos browsers lo aceptan.

```
<strong><em>esta linea es un estandar</em></strong>
<strong><em>esta linea no lo es</strong></em>
```

Algunos tags tienen parámetros adicionales. Por ejemplo: <h1 align=center>. El valor del parámetro debe estar entre comillas dobles cuando contiene espacios en blanco.

HTML continúa evolucionando rápidamente. La versión original (ahora conocida como HTML 1.0, antes llamada solo HTML) incluyó tags para encabezados, varios tipos de listas, texto preformateado, texto fuerte y enfatizado, y algunas otras clases de contenido. HTML 1.0 es de hecho un estándar bien definido para casi todos los browsers, pero no es, ni nunca lo ha sido, un estándar oficial de ningún grupo u organización.

[5] Se conoce como tag a las marcas utilizadas en HTML para dar formato a las páginas Web.
[6] Símbolos < y > usados para delimitar un tag en HTML.
[7] Término empleado para instrucciones contenidas en otras instrucciones.

HTML 2.0 agrega imágenes y formas interactivas al lenguaje. Esto fué dirigido por varias personas en NSCA para desarrollar un browser gráfico que pudiese desplegar gráficas. HTML 2.0 es un estándar del IESG (INTERNET ENGINEERING STEERING GROUP). Todos los browsers actuales soportan HTML 2.0 muy bien.

HTML 3.0 incorpora tablas, ecuaciones matemáticas, figuras y estilos para páginas. Algunas de estas extensiones han sido implementadas en diferentes browsers, pero no soportan toda la gama de las extensiones propuestas en HTML 3.0. de cualquier manera, HTML 3.0 no es un estándar, y nunca lo será; los esfuerzos para crear estándares, muchas veces se ven superados por los avances tan rápidos en el desarrollo de web.

Muchas compañías han agregado extensiones no-estandares a HTML. Estas incluyen texto intermitente, frames y applets. Algunas de estas extensiones, como las tablas, están basadas en las propuestas para HTML 3.0.

Nunca existió un HTML 3.1. El estándar actualmente en desarrollo es HTML 3.2, que codifica las características existentes en la mayoría de los browsers; y elimina algunas otras. La adición más importante son los estilos para páginas, que permiten aplicar a los documentos un formato específico para diferentes tags. El Consorcio World Wide Web (W3C), la organización que desarrolla los estándares HTML, decidió saltar directamente a la versión 3.2 desde la inconclusa 3.0.

HTML 4.0 apenas está siendo discutido, y básicamente piensa incorporar ya las ecuaciones matemáticas que originalmente estaban planeadas para HTML 3.0.

2.3 MIME.

MIME (Multipurpose Internet Mail Extensions) es un estándar abierto para enviar datos multimedia a través del correo electrónico en Internet. Los datos pueden ser binarios, o pueden usar múltiples caracteres ASCII y no-ASCII. Aunque MIME fué planeado originalmente para correo electrónico, se convirtió en una técnica ampliamente utilizada para describir el contenido de archivos, así que el software del cliente puede notar la diferencia entre los diferentes tipos de datos. Por ejemplo, un browser usa MIME para saber si un archivo se trata de una imagen GIF o un archivo PostScript.

MIME soporta casi cien tipos predefinidos de contenido. Los tipos de contenido son clasificados en dos niveles: tipos y subtipos. Los tipos muestran de una manera muy general que clase de dato estácontenido: imagen, texto, video. El subtipo identifica específicamente el tipo de dato: imagen GIF, imagen JPEG, imagen TIFF. Por ejemplo, el tipo de contenido (Content-type) de HTML es text/html; el tipo es texto, y el subtipo es html. En casi todos los sistemas, un simple archivo de texto mantiene un mapeo entre tipos MIME y la aplicación usada para procesar ese tipo de datos. En UNIX, este archivo es llamado *mime.types*.

Los datos regresados por un servidor HTTP 1.0 o 1.1 es enviado con formato MIME. Casi todos los servidores y clientes web entienden al menos dos tipos MIME de texto: text/html y text/plain, y dos formatos de imagen, image/gif e imag/jpeg.

2.4 CGI.

CGI (Common Gateway Interface), la interfaz de pasarela común, es usada para generar páginas Web dinámicas; el browser invoca un programa en el servidor que crea una nueva página. Esta página web puede procesar los resultados de una forma

enviada por el cliente. Se pueden escribir programas CGI en muchos lenguajes como Perl, C, Shell o Java. Los programas CGI corren como procesos independientes, iniciados por el servidor HTTP cada vez que se recibe una petición de servicio. Esto tiene tres importantes consecuencias: Primero, los programas CGI son relativamente seguros al correr, un programa CGI puede no ejecutarse satisfactoriamente, sin dañar al servidor en sistemas de memoria protegida como UNIX. Segundo, los programas CGI tienen estrictamente limitado el acceso al servidor. Tercero, los programas CGI exigen un sacrificio de rendimiento relativo a servir un archivo estático, debido la elevado número de procesos separados generados por cada petición.

Los programas CGI más simples corren sin una entrada de datos del usuario. Desde el punto de vista del cliente, estos son accesados como cualquier página web. La diferencia entre una página web producida por un CGI que no toma ninguna entrada de datos, y una página web escrita en HTML estático reside en el lado del servidor.

Un CGI contiene suficiente inteligencia para garantizar una comunicación fructífera entre el servidor y una aplicación externa. En otras palabras, un CGI permite a un servidor web suministrar a los clientes información que de otro modo no podrían conseguir en una manera legible. Por ejemplo, esto podría permitir a un cliente web mandar una petición a una base de datos SQL y recibir una respuesta apropiada en forma de documento web. Es por esto que se considera que el propósito principal del CGI es mantener la comunicación con información y servicios fuera del alcance normal de Web, fundamentalmente para crear objetos HTTP a partir de que no lo son.

2.4.1 La especificación CGI.

La especificación CGI fué creada y documentada por los principales autores del servidor HTTP (Tony Sanders, Ari Loutonen, George Phillips y John Franks). Qienes

se dieron cuenta de que no querían seguir añadiendo funcionalidad a sus servidores en función de una u otra actividad particular web. Por lo tanto decidieron crear un núcleo claramente definido de la funcionalidad del servidor web y facilitar un modo de ampliar los servicios y capacidades desde este planteamiento.

Necesitaban una interfaz de programación de aplicación (API, Aplication Programming Interface) para cualquiera que trabajase en lenguajes como Perl, C o sobre una shell. De ahí la creación de CGI como una especificación rigurosa y formal.

2.4.2 Variables de entorno.

Las variables de entorno son entidades que existen dentro del entorno informático particular del usuario. Muchas de estas variables consiguen sus valores siempre que un usuario accede a un ordenador o este está bajo un sistema operativo no multitarea como puede ser DOS. Las variables de entorno están generalizadas dentro de UNIX, donde se utilizan para pasar información desde un entorno de ejecución a las aplicaciones.

Las variables de entorno actúan en ocasiones como canastas para pasar datos de una aplicación a otra en una misma sesión. En el caso de los CGI, las variables de entorno conocidas por el servidor y por el CGI se utilizan para transmitir datos acerca de una petición HTTP de un servidor a la aplicación CGI. Estas variables son asequibles tanto para el servidor como para cualquier aplicación CGI que las pueda solicitar.

El servidor web también puede usar argumentos de líneas de comando, además de variables de entorno, para transmitir datos sobre una petición de información de un cliente web. Esto permite un control específico sobre los parámetros o sobre la realización de los programas que sean obra del servidor, además de proporcionar un

mecanismo para que los clientes pasen pequeñas secuencias de entradas a los programas CGI relacionados.

2.4.3 Acceso a las variables CGI.

Los requisitos de un servidor web específico controla como las aplicaciones CGI acceden a las variables. Si una variable no tiene valor, o no está definida, esto indica que tiene un valor cero (NULL, nulo, en UNIX). Al igual que ocurre con el acceso de variables, las indicaciones del sistema para el servidor web dictan el modo de representar los valores para las variables CGI.

AUTH_TYPE

Esta variable se utiliza para suministrar varios niveles de seguridad de acceso al servidor. Si su servidor soporta verificación de acceso, este valor de variable de entorno indica el método de verificación específico del protocolo que se ha utilizado para validar a un usuario.

```
AUTH_TYPE=Basic
```

CONTENT_LENGTH

El valor de esta variable es igual al número decimal de bytes de la entidad adjunta, si es que hay alguna. Si no hay entidad adjunta, el valor es NULL.

```
CONTENT_LENGTH=142
```

CONTENT_TYPE

El valor de esta variable indica el tipo de la entidad asociada, si es que hay alguna. Si no hay entidad, también tiene el valor NULL. HTTP 1.0 utiliza tipos de contenido MIME, que facilitan un mecanismo abierto y extensible para escribir a máquina los datos y para la negociación de tipos de datos. Con HTTP se pueden representar diferentes medias como pueden ser: textos, imágenes, audio y video.

```
CONTENT_TYPE=image/gif
```

GATEWAY_INTERFACE

Esta variable de entorno representa la versión de la especificación CGI a la cual obedece el servidor que proporciona este valor. Esta variable no es especifica de una solicitud en concreto definida para todas las peticiones HTTP.

```
GATEWAY_INTERFACE=CGI/1.1
```

HTTP_*

Las variables de entorno CGI que comienzan con la cadena "HTTP", contienen una información de cabecera HTTP suministrada por el cliente. Son tipos de contenido MIME que el cliente aceptara como cabeceras de petición HTTP. Cada tipo de dato en esta lista debería estar separado por una coma, según se requiere en la especificación 1.0 de HTTP.

```
HTTP_ACCEPT=audio/basic, image/gif, text/html, www/mime.
```

QUERY_STRING

Esta variable representa una cadena de búsqueda codificada bajo un URL. El valor de esta variable sigue al caracter "?" en el URL que referenciaba una aplicación

de búsqueda CGI. El servidor no decodifica el valor de esta variable, pero lo pasa a la aplicación CGI sin ningún cambio.

El valor de esta variable está codificado en el formato URL estándar en el que se reemplaza el espacio con un signo más, ("+") y se codifican caracteres no imprimibles con el esquema de codificación hexadecimal '%dd', en el que 'd' representa un dígito.

Esta variable siempre se coloca cuando se produce alguna pregunta de un cliente, independientemente de que la aplicación CGI decodifique cualquier línea de comandos.

http://www.unam.mx/cgi-bin/place.pl?Apellido_Pat

QUERY_STRING=Apellido_Pat

REQUEST_METHOD

El valor de esta variable de entorno CGI representa el método que se utilizó para realizar la petición del cliente. Para HTTP 1.0 este método es GET, HEAD, POST, PUT, DELETE, LINK, y UNLINK. El nombre del método es caso sensible.

REQUEST_METHOD=POST

SCRIPT_NAME

El valor de esta variable es la ruta del URI que identifica una aplicación CGI. Es la ruta virtual hacia una aplicación CGI que realiza un servidor.

http://www.unam.mx/cgi-bin/place.pl

SCRIPT_NAME=/cgi-bin/place.pl

SERVER_NAME

Esta variable representa el nombre del host del servidor, alias DNS, o IP según aparece en los URL. Esta variable no es especifica de una solicitud determinada y su valor se asigna en todas las peticiones o solicitudes que recibe el servidor.

SERVER_NAME=www.unam.mx

SERVER_PORT

Esta variable es el puerto en el cual se recibe la petición del cliente. La mayor parte de implementaciones de servidores HTTP utilizan por defecto el puerto 80. Los URL que no especifiquen claramente un puerto pueden realizar la implementación a través del puerto 80.

SERVER_PORT=80

SERVER_PROTOCOL

El valor de esta variable representa el nombre y la versión del protocolo de información que el cliente solicitante utiliza. El protocolo es similar al esquema URL utilizado por el cliente y se trata de un caso no sensible.

SERVER_PROTOCOL=HTTP/1.0

SERVER_SOFTWARE

Esta variable representa el nombre y la versión del software del servidor. Esta variable no es específica de una solicitud determinada y su valor se asigna en todas las peticiones que recibe el servidor.

SERVER_SOFTWARE=NCSA/1.4

2.4.4 Entrada de datos en CGI.

Para peticiones con datos añadidos que siguen a la cabecera de petición HTTP como son HTTP POST o PUT, los datos se llevan a la aplicación CGI utilizando un descriptor estándar de ficheros de entrada. Todas las aplicaciones CGI siguen este método para analizar los datos. El sistema operativo UNIX utiliza un mecanismo llamado de entradas estándar y al que se le conoce con el nombre de stdin.

El servidor envía un número de bytes de información, siempre y cuando el valor este especificado por la variable CONTENT_LENGTH, utilizando el descriptor de ficheros stdin para pasar los datos a la aplicación CGI. La aplicación CGI no tiene que intentar leer más datos que los indicados en la variable CONTENT_LENGTH.

El servidor también suministra la variable CONTENT_LENGTH para los datos pasados a la aplicación CGI. Esto permite a la aplicación suministrar esta información para decidir como interpretar los datos que recibe. Al final de estos datos, al servidor no se le pide que transmita una marca de finalización de fichero. Tanto el servidor como la aplicación, asumen que la lectura se terminará inmediatamente después de que la aplicación CGI lea el número de bytes especificados por el valor de la variable CONTENT_LENGTH.

2.4.5 Salida de datos en CGI.

Una aplicación CGI siempre entrega cierta información al servidor o al cliente que la invoca. La aplicación CGI envía su salida a un descriptor de ficheros estándar, UNIX lo llama stdout. La salida de la aplicación CGI podría ser un documento generado por la aplicación o podrían ser instrucciones de datos recuperados para el servidor, es decir, transferir un fichero al cliente. Las salidas de información se devuelven, normalmente, de una o dos maneras. La primera es en una salida de cabecera no analizada. Utilizando esta forma, una aplicación CGI debe devolver un mensaje de respuesta HTTP completo. La segunda forma es una cabecera de salida estructurada. Se requiere un servidor para mantener esta segunda manera. En este caso, la aplicación CGI devuelve un mensaje de respuesta CGI. Esta respuesta consiste en varias cabeceras y un cuerpo separado por una línea en blanco. Estas cabeceras pueden ser cabeceras CGI, que se interpretarían por un servidor, o bien cabeceras HTTP que se incluyen dentro de un mensaje de respuesta. Aquí el cuerpo es opcional, pero si aparece debe estar precedido por un cabecera de tipo MIME.

2.5 Lenguajes de programación de CGI.

Al igual que difieren sustancialmente las filosofías de los lenguajes, también difiere el estilo de los programas resultantes. Los cuatro tipos básicos de lenguajes de programación que se analizarán son:

Procedurales, describen los pasos de un algoritmo.
Orientados a objetos, describen las interacciones entre objetos.
Lógicos, deducen la solución de sus predicados.
Funcionales, describen las funciones de transformación.

La elección del lenguaje de programación adecuado pasa por muchas consideraciones y análisis. En algunas ocasiones la elección de un lenguaje se realiza simplemente porque el programador conoce ese lenguaje y puede obtener ventajas de su sintaxis y funciones.

En otras ocasiones, se elige un lenguaje porque se ha demostrado que es el más comprensible, eficiente, seguro y el más adecuado para realizar una tarea especifica.

Sin embargo, existen cinco consideraciones básicas que deben ser tomadas en cuenta a la hora de elegir un lenguaje para programar CGI y son las siguientes:

- La cantidad de código fuente público que hay en servidores de fácil acceso.
- La disponibilidad de herramientas de infraestructura y apoyo como los depuradores, compiladores, intérpretes, libros, clases y editores; a esto se le conoce como soporte de lenguaje.
- El nivel personal de conocimiento sobre un lenguaje en particular.
- El rendimiento deseado de los datos, para ofrecer apoyo en operaciones específicas.
- La utilidad, modularidad y capacidad de extensión.

Código fuente público

Quizá, la consideración más popular a la hora de elegir un lenguaje de programación. Desde el principio se puede visitar muchas localizaciones Internet donde se encuentran librerías de código fuente públicas.

Estas localizaciones contienen códigos que cualquier programador de CGI podría modificar y compilar en un momento y, tendría una aplicación CGI casi inmediata.

Sin embargo, esto tiene un cierto riesgo. Este código puede tener en su interior líneas de código que dañen la información o todo el sistema.

Soporte de lenguaje

Primero se debe determinar la disponibilidad de lenguajes de programación que existen en su sistema. En la mayoría de sistemas UNIX se pueden encontrar lenguajes como Perl, C, C++ o Tcl. En caso de no contar con alguno de estos lenguajes, existen sitos web que tienen versiones gratuitas de dichos interpretes o compiladores.

Finalmente se debe determinar la disponibilidad y la calidad del material de referencia del lenguaje de programación.

Nivel personal de conocimiento

Cuando se esté considerando un lenguaje CGI, se debe analizar las experiencias pasadas, su nivel actual de conocimiento sobre ese lenguaje y su capacidad de aprender uno nuevo.

Cabe mencionar que el cambio de un lenguaje a otro, normalmente no supone una gran diferencia para un programador experimentado.

Rendimiento deseado de datos

Desde que el WWW se desarrollo en UNIX, muchos de los servidores contienen lenguajes basados en UNIX como Perl, C shell, Tcl, Bourne shell y C/C++. Cada uno de estos lenguajes se puede encontrar en casi todos los sistemas UNIX. Entre estos seis lenguajes hay que hacer dos distinciones: Perl, Tcl y las shells Bourne y C son lenguajes interpretados, esto quiere decir que sus objetos fuente no se compilan en formato binario y no se cargan y se ejecutan después, cada uno de estos tiene un

intérprete que lee cada línea de código fuente cada vez y la ejecuta, esto produce una disminución del rendimiento de la aplicación y de los datos.

Por el contrario, C y C++ se compilan en un ejecutable binario, se cargan en memoria y se ejecutan a una velocidad mayor, con lo que el rendimiento es también mayor.

Finalmente tenemos el lenguaje Java que es una combinación de lenguajes compilados e interpretados. Java requiere un sistema runtime, incluyendo un intérprete propio de una plataforma y un objeto binario compilado sobre una arquitectura neutral. La velocidad de ejecución de Java no difiere mucho de la encontrada en programas en C o C++.

2.5.1 Lenguajes de programación CGI compilados.

Un compilador es típicamente nativo de una arquitectura particular y puede asumir que los objetos binarios de una arquitectura no se ejecutaran en otra. Esto es, un programa compilado bajo plataforma PC no correrá bajo una plataforma UNIX.

2.5.1.1 C.

C es un lenguaje procedural que describe los pasos de un algoritmo, hay dos ventajas importantes al elegir C como lenguaje de programación: Primero, se puede compilar en un objeto binario que ocupa un espacio mínimo comparado con los lenguajes interpretados, segundo, los objetos binarios se ejecutan normalmente más rápido que los lenguajes interpretados.

La desventaja principal de utilizar C para la programación en CGI es que el manejo de cadenas es difícil con los constructores de C, cerca del 90% de las

aplicaciones CGI implican el manejo intensivo de cadenas. Esto significa que los caracteres y los datos en cadenas se deben transformar, convertir o trasladar de un formulario a otro.

2.5.1.2 C++

C++, sucesor de C, es miembro del paradigma orientado a objeto dentro de los lenguajes de programación. Los lenguajes orientados a objetos ofrecen muchas ventajas. Dan un reutilización de clases superior lo cual reduce el costo de desarrollo para aplicaciones similares. También permiten la capacidad de extensión de clases comunes permitiendo a los programadores añadir funcionalidad a desarrollos existentes.

La principal desventaja de utilizar C++ es que es orientado a objetos. El desarrollo el código fuente requiere una formación sustancial. Entre los conceptos nuevos que se pueden encontrar están: las clases y el polimorfismo de funciones.

2.5.2 Lenguajes de programación de CGI interpretados.

Estos lenguajes se interpretan en el momento en que son llamados a ejecutarse. Al crearse, no es necesario compilarlos, no se genera un programa objeto ni un programa ejecutable.

Como se describe más adelante, el sistema REDLAEM se programó casi en su totalidad en Perl, por eso se ahonda en la descripción de este lenguaje de programación de CGI´s.

2.5.2.1 Perl.

Perl (the Practical Extraction and Report Language) es un lenguaje interpretado optimizado para el manejo sencillo de ficheros, texto y procesos. Perl combina algunas de las características de C, sed awk y sh. Los programadores que estén familiarizados con estos lenguajes tendrán pocas dificultades a la hora de aprender y aplicar Perl.

Perl se considera el mejor de los lenguajes interpretados dada su habilidad para el manejo intensivo de cadenas. Perl utiliza técnicas sofisticadas para examinar rápida y eficientemente grandes cantidades de texto; también puede manejar datos binarios con la misma facilidad.

Su autor, Larry Wall (lwall@netlabs.com) realizó Perl casi como una obra altruista, de modo que hasta PERL 5.X su distribución es gratuita, sin que por eso tenga menos poder o consistencia.

2.5.2.1.1 ¿Para qué sirve?

Perl surgió como una opción para una gran cantidad de herramientas de UNIX en las cuales basa su propia sintaxis, buscando el mínimo sacrificio de su desempeño por una máxima facilidad de programación e integración, sigue la filosofía de mantener un ambiente que sea capaz de detectar y corregir pequeñas omisiones del programador, y de proporcionarle una forma abreviada de realizar múltiples tareas. Lo cual significa que, es una utilería que pretende facilitar el proceso de grandes volúmenes de información sin minimizar el rendimiento.

2.5.2.1.2 ¿Donde puede usarse?

Las plataformas donde Perl se ha desarrollado mas son los servidores UNIX, por sus necesidades de administración y lo robusto de su manejo de memoria y de procesos (requisitos de PERL hacia el S.O.) además de la facilidad de Perl para realizar los así llamados CGIs, interfaces para comunicar recursos del servidor con un servicio de Internet particular (como podría ser WWW o gopher), En otras plataformas, PC en particular, se han desarrollado versiones que mantienen un razonable grado de funcionalidad, pero en realidad, el sistema DOS no tiene un manejo óptimo de los procesos o de la memoria para permitir a Perl dar un buen desempeño, además de que no es común ver en PC necesidades de administración de la magnitud de un servidor institucional. Sin embargo, puede practicarse la programación en PERL de PC, o incluso elaborar programas de reporte en él , sin embargo, es algo que no se ha popularizado hasta hoy.

2.5.2.1.3 La filosofía de Perl.

Perl no establece ninguna filosofía de programación (de hecho, no se puede decir que sea orientado a objetos, modular o estructurado aun cuando soporta directamente todos estos paradigmas), los objetivos que se tuvieron en cuenta al diseñar la sintaxis de Perl fueron la facilidad de aprendizaje y de uso y la claridad de código, las cuales, considero que son necesarias (aunque pueden escribirse programas en Perl complejos e inteligibles si así se desea).

Por si fuese poco, Perl no es ni un compilador ni un intérprete, está en un punto intermedio, cuando mandamos a ejecutar un programa en Perl, se compila el código fuente a un código intermedio en memoria, se le optimiza (como si fuésemos a elaborar un programa ejecutable) pero es ejecutado por un motor, como si se tratase de un intérprete. El resultado final, es que utilizamos algo que se comporta como un

intérprete pero que tiene un rendimiento comparativo al de programas compilados. Sin embargo, ya existen compiladores de Perl con la versión 5.

2.5.2.1.4 Diferencias entre Perl 4.x y 5.x

Actualmente existen dos versiones altamente populares de Perl, la 4.x y la 5.x, de hecho hay diferencias importantes entre una versión y otra. Seguramente el lector se pregunta porque surge la duda entre usar una versión vieja y una nueva, por regla general las nuevas versiones son mejores que las anteriores de modo que las opacan en todo sentido, Perl no es la excepción a esta regla, el único factor que impide una transición inmediata es que no son 100% compatibles. La versión 5 de Perl es una reescritura total que ya incluye un manejo de estructuras abstractas de datos mucho más poderoso, incluso, soporta la orientación a objetos a su manera (tema que no trato en esta introducción). De modo que las librerías, por ejemplo para creación de CGI's no funcionan de una función a otra por lo que la migración es poco práctica.

Así pues, la decisión sobre que versión utilizar depende del trabajo que haya sido realizado con anterioridad, si ya se tiene un sistema completo o grande en Perl 4, es recomendable mantenerlo en Perl 4 por el resto de su vida útil, pero para desarrollos nuevos es más recomendable iniciarlos con Perl 5 o en su caso, la versión más reciente que este disponible, por experiencia se que la capacitación para adaptarse a la nueva versión es extraordinariamente corta dada la importancia de las mejoras. Una tercera opción (que por sus ventajas es la más recomendable en caso de contar con los recursos) consiste en instalar las dos versiones de modo que convivan en el sistema.

En la actualidad, la mayoría de las aplicaciones CGI utilizan Perl por sus características positivas.

2.5.3 Lenguajes de creación de scripts en UNIX.

La shell es la interfaz de usuario más importante en un ambiente UNIX. La shell es simplemente otro programa. No tiene un estado especial. Puede ser modificada y actualizada con bastante facilidad, dando lugar a varias shells. Cada shell funciona como un programa separado y cada usuario puede elegir qué shell utilizar cuando está trabajando en un sistema UNIX. En este caso analizaremos dos shells: la "C shell" y la "Bourne shell". Estas shells no son sintácticamente compatibles. Cada comando interactivo y lenguaje de escritura shell difieren y cada uno tiene comportamientos y características únicas. Cada shell tiene su propio lenguaje de escritura, el cual puede ser la base para los programas CGI.

2.5.3.1 La Bourne shell.

La Bourne shell es parte de una configuración estándar para cada versión de UNIX; por su menor tamaño, el procesamiento sobre ella es más eficiente.

La Bourne shell es compacta, se ejecuta rápidamente y requiere fuentes mínimas. Desarrollada en AT&T, se ha convertido en la shell más usada. Entre las principales ventajas de la Bourne shell encontramos que:

- Permite el manejo de excepciones.
- Soporta variables locales y globales.
- Utiliza tuberías con nombre del System V.

El lenguaje de escritura es parecido a C, pero funciona de forma semejante a un lenguaje de comandos convencional. Una shell aunque resulta excelente para actividades pequeñas, no tiene la flexibilidad y la capacidad de extensión de un lenguaje de programación.

2.5.3.2 La C shell

La shell que se ejecuta bajo la mayor parte de los sistemas UNIX es la C shell, desarrollada por Bill Joy en la Universidad de California en Berkeley. Es un intérprete de comandos y suministra un interfaz de usuario muy importante al sistema operativo UNIX.

Entre las ventajas de la C shell se incluyen:

- La función history, que mantiene registros de comandos, mientras se realizan y permite volver a ejecutarlos sin necesidad de volver a teclearlos.
- Evaluación directa de los comandos nativos de UNIX incorporados.
- Mecanismos a través de los cuales puede crear nombres mnemotécnicos para comandos, caminos y otros objetos del sistema.
- Control de tareas en primer y segundo plano.
- Sintaxis similar a la del lenguaje de programación C.

Aunque la escritura de la C shell es parecida a la de C, trabaja más como un lenguaje de comandos convencional que como un lenguaje de programación completo. Mientras que es útil para actividades y aplicaciones pequeñas, la C shell no tiene el poder ni la capacidad de extensión que contiene un lenguaje de programación.

2.5.4 Lenguajes de programación CGI interpretados y compilados.

Estos lenguajes, al compilarse, generan un código binario que luego puede interpretarse en varias plataformas indistintamente.

2.5.4.1 JAVA.

Java es un lenguaje de programación orientado a objetos y su entorno proviene de Sun Microsystems. Como con C y C++, Java se compila en un objeto binario y luego se interpreta dependiendo de la arquitectura específica en la que se va a ejecutar; por lo anterior se dice que Java es Independiente de la plataforma, se basa en el principio de compilarse una sola vez, y ejecutarse en cualquier máquina que tenga instalado un intérprete Java. Lo anterior lo convierte en el lenguaje del futuro, y marcará el camino a seguir en la expansión de Internet.

Netscape Communications Corporation ha licenciado el lenguaje Java para implementarlo dentro del navegador Netscape Navigator. Su mayor motivación es incrementar el campo de extensión de Navigator y facilitar la creación de nuevos tipos de aplicaciones en red teniendo en cuenta la estructura cliente/servidor.

2.6 Introducción al modelo cliente/servidor.

El modelo cliente/servidor abarca un amplio campo de funciones, servicios, y otros aspectos dentro del ambiente distribuido. Estos incluyen cómputo en redes locales y de área amplia, datos distribuidos, procesamiento distribuido, manejo y procesamiento distribuido de transacciones, y sistemas abiertos.

2.6.1 Procesamiento cliente/servidor.

El modelo de procesamiento cliente/servidor ha surgido como el nivel más alto del procesamiento por compartición de dispositivos encontrado típicamente en redes de área local (LAN).

En un procesamiento por compartición de dispositivos en un ambiente LAN, las computadoras personales (PC's) se conectan a través de un sistema que les permite compartir recursos comunes, como son: archivos, disco duro, impresoras, etc. En la terminología LAN, los dispositivos compartidos se llaman servidores (un servidor de archivos, un servidor de impresión). El término servidor es válido porque estos dispositivos compartidos reciben peticiones de un servicio desde las PC's. en este procesamiento, las peticiones de las PC's se limitan generalmente a servicios relacionados con la lectura o la impresión de un archivo.

El modelo de procesamiento cliente/servidor es una extensión natural del procesamiento por compartición de dispositivos. Gradualmente, los servidores han sido capaces de atender eficientemente a un número mayor de estaciones de trabajo. Al mismo tiempo, el rol de las estaciones de trabajo también cambió, las estaciones se convirtieron en clientes de los servidores.

En este modelo, el procesamiento de la aplicación es dividido entre el cliente y el servidor, los dos cooperan para la ejecución satisfactoria de la aplicación. Un servidor de bases de datos como SYBASE® es un ejemplo de un ambiente de procesamiento cliente/servidor.

Para ejemplificar esto, tenemos que: en el caso de un servidor de bases de datos, una aplicación corriendo en una máquina envía una petición para leer un registro al servidor de la base de datos. El servidor de la base de datos procesa

localmente la base de datos y envía únicamente el registro solicitado a la aplicación cliente.

Para resumir, un procesamiento bajo arquitectura cliente/servidor requiere:

- Una comunicación robusta entre cliente y servidor.
- Interacción cooperativa entre cliente y servidor que es iniciada por el cliente.
- Procesamiento de la aplicación distribuido entre el cliente y su servidor.
- Servidor esforzado con control sobre que servicios o datos puede solicitar el cliente.
- Servidor basado en el control de conflictos entre solicitudes de clientes.

2.6.2 La especialización del cliente.

En el paradigma cliente/servidor, la aplicación y los datos están distribuidos a través de la red. Los nodos de la red pueden ser clasificados en clientes (aquellos que solicitan servicios) y servidores (aquellos que proporcionan los servicios solicitados). Los clientes y los servidores cooperan en una relación establecida para cada par cliente/servidor, además también puede establecerse una relación de muchos a uno entre una colección de clientes y su servidor.

En una arquitectura cliente/servidor, un sistema individual es designado como un cliente o como un servidor dependiendo de la actividad (solicitando un servicio o proporcionando un servicio) que realiza en una relación nodo a nodo en un momento dado.

2.6.2.1 El rol del cliente.

Desde que los nodos cliente son diseñados para interactuar con el usuario final, su implementación y funcionalidad pueden ser especializadas para esas interacciones. Las funciones más realizadas por un sistema cliente son las de presentación. Un usuario final interactúa con la aplicación a través de la interfaz, además de que sigue la lógica de la presentación. La lógica de la presentación es la capa de la arquitectura cliente/servidor que, por un lado interactúa con la lógica de la aplicación, y , por el otro, interactúa con el usuario. Después se incluyen todas las interacciones con el dispositivo físico y el manejo de la entrada/salida.

2.6.3 La especialización del servidor.

La especialización del servidor se ve bien reflejada en la funcionalidad y diseño de servidores de bases de datos. Básicamente, los servidores de bases de datos son capaces de proveer grandes volúmenes de almacenamiento, significante poder de procesamiento, y la habilidad de correr muchas aplicaciones (clientes) simultáneamente. De cualquier forma, la tecnología sigue evolucionando, la especialización se extiende a muchas funciones como comunicaciones, emulación de terminal, fax, manejo de librerías y correo electrónico.

2.6.3.1 El rol del servidor.

Dentro de la arquitectura, un servidor es un proceso lógico que provee servicios a procesos que los solicitan. En el procesamiento cliente/servidor, un cliente inicia la interacción enviando una petición a su servidor. Las funciones que el servidor debe ejecutar, están determinadas, en gran parte, por los tipos de peticiones que los clientes pueden enviar a los servidores. Si un servidor es incapaz de ejecutar una función solicitada por un cliente, entonces ese servidor no puede participar en la interacción.

Idealmente, un cliente no debe estar enviando peticiones no soportadas al servidor. En general, de cualquier manera, una vez que en cliente y el servidor están interconectados en una red, las siguientes funciones pueden ser solicitadas a un servidor:

- Compartición de archivos.
- Compartición de impresoras.
- Acceso a Bases de Datos.
- Servicios de comunicación.

2.7 Manejo distribuido de datos.

El manejo distribuido de datos es uno de los estilos de procesamiento distribuido compartido implementado en la arquitectura cliente/servidor. La arquitectura cliente/servidor ofrece una solución ideal a los requerimientos del manejo distribuido de datos. Muchas de las implementaciones de arquitectura cliente/servidor disponibles en la actualidad son sistemas de manejo distribuido de datos, o DDBMS. El ambiente del manejo distribuido de datos se caracteriza por una distribución de dos sentidos:

- Los datos y el manejador de bases de datos están distribuidos en múltiples nodos, incluyendo el nodo con la aplicación lógica.
- Las funciones del manejo de datos están distribuidas entre un sistema front-end (lógica de manejo de datos y el lenguaje), y el servidor back-end (funciones de base de datos).

2.7.1 ¿Por qué distribución de datos?

El manejo distribuido de datos negocia con datos (Bases de Datos) distribuidos a través de múltiples nodos. Distribuyendo datos a través de múltiples sitios ofrece los siguientes beneficios:

- Acomodo cercano de datos a su fuente de una manera apropiada.
- Alta disponibilidad de datos, colocando múltiples copias de datos críticos en diferentes localidades, lo que elimina el riesgo de pérdida si se presenta una falla.
- Acceso más eficiente a los datos, lo que mejora el desempeño.
- Facilidad de crecimiento de las aplicaciones y las demandas del usuario final.

2.7.2 El modelo relacional.

Un modelo relacional de datos tiene todos los datos organizados en tablas. Las filas representan registros de datos, mientras que las columnas (atributos) representan campos en el registro. No hay filas repetidas y el orden de las filas no es significante. Las tablas representan hechos y valores del mundo real. La columna que únicamente identifica un hecho particular en el que se basa la tabla, representa la llave primaria de esa tabla. Típicamente, cada tabla contiene una llave primaria.

La relación entre la llave primaria y las llaves externas presenta un interesante problema cuando una (o más) de las tablas relacionadas es modificada. Consideremos, por ejemplo, que una fila conteniendo una llave primaria, referenciada a una llave externa, es borrada. Entonces, la llave externa también debe modificarse o borrarse. De lo contrario se perdería la correspondencia de datos, y la consistencia en la información. De igual manera, si se inserta una nueva fila que contiene una llave

externa, el valor de la nueva llave debe coincidir con el de la llave primaria de la tabla relacionada.

Las bases de datos soportan el lenguaje relacional, SQL[8] (Structured Query Language). Este lenguaje es usado para formular operaciones que define y manipula datos en forma relacional. SQL es la única manera de proveer acceso a los datos en una base de datos relacional. SQL contiene un conjunto pequeño de operaciones, por lo que su aprendizaje y utilización son sencillos.

Una de las principales ventajas de SQL y el modelo de datos relacional es su acceso a los datos de manera no navegable. En un sistema manejador no relacional de base de datos, el usuario tiene que decirle al sistema no solo que dato se necesita, sino también cómo obtenerlo, el cómo se logra seleccionando una ruta de acceso a los datos y navegando a través de todos ellos en la implementación física del modelo de datos. Por ejemplo, en el modelo jerárquico de datos, el programador usa el lenguaje apropiado para el sistema manejador de base de datos para moverse de arriba hacia abajo y de izquierda a derecha para localizar el dato deseado.

2.7.3 La evolución de SQL.

Por años, el Structured Query Language (lenguaje estructurado de consulta) a crecido en visión y poder. El primer paso hacia SQL fué el modelo de datos relacional desarrollado por el Dr. E. F. Codd en IBM, que había probado que las operaciones lógicas del álgebra podían ser usadas para extraer cualquier dato de una base de datos relacional, y que no es necesario un lenguaje de programación para efectuar esa tarea.

[8] Es un lenguaje de programación especializado para enviar consultas a bases de datos.

SQL ha sido robustecido por la ANSI, que en 1986 produce el primer estándar para SQL: SQL-86. Rápidamente se crea el Consorcio SQL (SQL Access Group, o SAG). Luego surge la versión SQL-92, y actualmente el estándar es SQL3, que incluye Extensiones Masivas Orientadas a Objetos (MOOSE). Actualmente se realizan esfuerzos para extender las habilidades de SQL a la multimedia en su versión SQL/MM.

Antecedentes del sistema

3. Antecedentes del sistema

En esta tesis desarrollo un sistema, que funciona en el Web, para la Red Latinoamericana de Enfermedades Metabólicas Hereditarias que agrupa especialistas en este tipo de enfermedades; es por esto que en el presente capítulo se tratan aspectos relacionados con este grupo de especialistas, tales como: las enfermedades, la situación anterior a REDLAEM, las necesidades de comunicación, los antecedentes del sistema, y los recursos que se destinaron para el proyecto.

3.1 Las enfermedades.

Las enfermedades metabólicas hereditarias son poco comunes y muy complejas; hay cerca de 500 descubiertas hasta ahora, aunque se estima que son más de 2,000, y su número conocido aumenta cada año.

Estas constituyen fallas en cada una de las reacciones químicas del organismo; por lo que constituyen un grupo enormemente diverso. Por ello, y por la muy baja frecuencia de cada una de ellas en lo individual, los especialistas solo pueden tener experiencia amplia en apenas una fracción de la totalidad, lo que hace imprescindible la colaboración para su diagnóstico, tratamiento e investigación, es por esto que se requiere de mecanismos eficientes de comunicación.

3.2 La situación.

En el caso de especialistas que residen en los países latinoamericanos, hasta ahora las interacciones se han dado fundamentalmente con colegas del llamado "primer mundo", principalmente Estados Unidos y Europa, empleando como idioma el inglés.

Pero las diferencias en el idioma limitan la comunicación y las posibilidades de entrenamiento y capacitación.

Aún más importante aunque menos obvio, son las grandes diferencias en las condiciones socioeconómicas y culturales entre las naciones latinoamericanas y países más desarrollados, que confieren grandes diferencias al entorno en el que estas enfermedades se desarrollan y a los medios para su diagnóstico y tratamiento, lo que disminuye la utilidad de las consultas y obstaculiza las colaboraciones potenciales.

Por lo anterior, la comunicación y colaboración entre especialistas en Latinoamérica resulta conveniente. América Latina llegó tarde al estudio y manejo de estos padecimientos, pero en la última década ha habido un aumento substancial del número de profesionistas e investigadores dedicados a ellos, con creciente interés y experiencia. Pero existía un problema práctico que parecía insoluble hasta el advenimiento de la globalización de la información: las enormes distancias que nos separan en una área geográfica tan extensa como la nuestra.

3.3 La comunicación.

Hasta antes del nacimiento de REDLAEM, la comunicación entre los especialistas inscritos en esta organización era muy limitada; básicamente esta se daba a través de medios como: el teléfono, correo electrónico, o personalmente, solo una vez cada año, en los congresos organizados por los especialistas, buscando solventar la necesidad de comunicación.

Esta comunicación, tan limitada, obviamente presenta muchas inconveniencias. El teléfono, por ejemplo, implica un elevado costo por cada llamada de larga distancia. El correo electrónico es más eficiente, pero se depende del tiempo que tarde la otra persona en contestar. En los congresos anuales, solo se abordan temas de interés general y los asuntos particulares de cada especialista se dejan un tanto al margen de la reunión.

3.4 Antecedentes de REDLAEM.

La idea de una red de especialistas latinoamericanos en enfermedades metabólicas hereditarias surgió en octubre de 1991, durante el Congreso Internacional de Genética Humana en Washington, en pláticas entre los doctores Carlos de Céspedes (Costa Rica), Fanny Cortés (Chile), Roberto Giugliani (Brasil) y Antonio Velázquez (México). La idea original incluía únicamente las personas que interactuarían entre sí, pero no los medios de comunicación. En el Congreso Latinoamericano de Genética en Puerto Vallarta, México, el Dr. Víctor Guerra Ortiz, a la sazón Director General de Cómputo Académico (DGSCA) de la Universidad Nacional Autónoma de México (UNAM), propuso utilizar la naciente *World Wide Web* (WWW) en el Internet, como la plataforma para la comunicación de los miembros de la red. Al mismo tiempo, la Dra. Susan Mize ofreció los protocolos de la Red de Información Metabólica (MIN – *Metabolic Information Network*), que ella y sus

colaboradores habían desarrollado en los Estados Unidos. Especialistas en enfermedades metabólicas de otros países de la región se sumaron a este esfuerzo, y se empezó a trabajar en el proyecto de cómputo en la DGSCA, junto con la Unidad de Genética de la Nutrición (UGN) de los Institutos de Investigaciones Biomédicas de la UNAM y Nacional de Pediatría.

3.4 Recursos destinados al sistema.

- El sistema es realizado por César Francisco Germán Rosas, del Departamento. de Incorporación y Extensión de Servicios, de la Coordinación de Servicios de Red, de la DGSCA-UNAM.

- La máquina en la que se alojará el servidor Web, el servidor de bases de datos y los programas creados, es: dragon.dgsca.unam.mx con dirección IP: 132.248.104.10

  Esta máquina tiene la las siguientes características:

  - Sun SparcCenter 2000 E
  - 512 MB en RAM
  - 8 Procesadores
  - Arreglo para 30 Discos Duros de 1.03 GB c/u

- El espacio en disco duro asignado en esta máquina para el proyecto es de: 35 Mbytes

- El servidor de base de datos y SQL es: SYBASE

- La base de datos se llama: DATOS_MEDICO; su tamaño es de 5Mbytes

- El URL asignado para el sitio Web REDLAEM es: http://www.unam.mx/redlaem

3.5 El entorno de desarrollo

Aquí se trata de documentar brevemente el entorno de desarrollo que se ha utilizado para construir el sitio Web REDLAEM.

- La plataforma es UNIX. La mayoría de los servidores Web en uso hoy en día funcionan con algún formato para UNIX. Además, esta plataforma ha demostrado ampliamente ser la más estable en un entorno de red. El sistema operativo instalado en la máquina asignada al proyecto es Sun Solaris ver. 2.5

- El lenguaje de programación elegido para crear los programas CGI´s, es Perl5. Se eligió este lenguaje por sus sobresalientes capacidades de manejar cadenas y por su excelente depurador, el cual permite la ejecución paso a paso, porque se detiene en la inspección de variables, y cumple con su cometido interactivo. Todas estas cosas le facilitan el tratamiento con el uso de HTTP para el manejo de la entrada y la salida durante la ejecución del programa. Perl es el lenguaje más eficiente y conveniente hoy en día para la programación CGI.

- El lenguaje de programación elegido para crear los programas que consultarán, insertarán y borrarán los registros de las bases de datos, es C; en su versión GCC 2.7 2. El lenguaje C ofrece las librerías (sybfront.h, sybdb.h, syberror.h) adecuadas que sirven para trabajar directamente con la conexión a SYBASE. Estos programas servirán como interfaz entre las bases de datos y los programas CGI´s.

- Trabajamos con HTTPD de NCSA. Esta implementación del demonio de HTTP se encuentra disponible para uso no comercial, es de fácil configuración y ha demostrado ser lo suficientemente estable en el manejo de peticiones de más de un cliente a la vez.

4. Análisis

4.1 Principios de una metodología.

Con el auge de los sistemas de información publicados en Internet, a través del web, los especialistas en desarrollar estos sistemas han intentado, sin éxito, establecer una metodología general estándar para los sitios web.

Por diferentes razones, especialmente por la libertad que en todos sentidos se tiene o se debe tener en el mundo de la información a través de Internet, es imposible decirle a los desarrolladores que su sitio debe tener o no ciertas características, o que debe presentar alguna determinada apariencia.

Por eso, aunque no existe una metodología para desarrollar sitios web reconocida por alguna autoridad en Internet, se consideraron, para la construcción del sitio Web de REDLAEM, aspectos como:

- Objetivo
- Planteamiento
- Diseño
- Desarrollo

En el desarrollo de este sistema de información (Red Latinoamericana de Enfermedades Metabólicas Hereditarias) se tienen las siguientes consideraciones:

4.1.1 Objetivo.

Desarrollar un sistema integral de información que, a través de web, permita a los especialistas en Enfermedades Metabólicas de Latinoamérica, tener comunicación directa; saber qué especialistas y qué laboratorios se encuentran en cada país; intercambiar experiencias, dudas y comentarios.

4.1.2 Planteamiento.

El sistema debe tener un directorio de Especialistas y Enfermedades, que permita consultar los datos generales de los especialistas, además un índice de todas las enfermedades, y los especialistas con experiencia en cada una de ellas.

El sistema debe tener también un directorio de Laboratorios y Pruebas, que permita consultar los datos de los laboratorios, además de un índice de todas las pruebas, y los laboratorios que las realizan.

Se requiere contar con un sistema de mensajes, para que cualquier usuario pueda enviar un mensaje de interés general a todos los demás usuarios y especialistas, y así se mantenga una comunicación entre ellos. Estos mensajes podrán ser leídos por todos los demás usuarios.

Para lograr un uso eficiente de las cualidades de Internet, es indispensable contar con una sala de discusión en tiempo real (CHAT).

Debido a que se trata de un servicio abierto a toda la comunidad Internet, se debe tener una opción para solicitar registro en el sistema, tanto para los especialistas como para los laboratorios.

Como en todo sistema de información, se debe tomar en cuenta que los datos pueden cambiar, para tal efecto se debe ofrecer una opción para que los usuarios puedan modificar los datos de su registro.

El sistema cuenta además con un administrador de información, este será capaz de dar de alta registros, modificar información existente, eliminar registros, tanto de especialistas como de laboratorios; y eliminar mensajes antiguos o inapropiados.

4.1.3 Diseño.

Antes de cualquier desarrollo o implementación del sistema, debe tenerse un diseño; una serie de consideraciones a tener en cuenta para que el sistema cumpla con las expectativas de los usuarios y de las instituciones involucradas en el proyecto.

4.1.3.1 Imagen.

Por ser un sistema dirigido al uso casi exclusivo de médicos, el sitio presenta una imagen de pulcritud distintiva de estos especialistas.

El sitio debe presentar un logotipo propio de REDLAEM, diseñado conjuntamente con el representante de REDLAEM en México y la Coordinación de Servicios de Red. Además de presentar el escudo de la Universidad Nacional Autónoma de México, por ser un proyecto avalado por el Instituto de Investigaciones Biomédicas de la UNAM, además de que el sistema es desarrollado por personal y con recursos de la UNAM.

4.1.3.2 Descripción.

La página está integrada por un directorio de especialistas y de enfermedades, otro de laboratorios, un boletín de mensajes, formas para solicitar inscripción a REDLAEM y una "sala" de discusión. Contiene, además, ligas a otros sitios existentes en el Web sobre este tipo de enfermedades. Existe un sistema paralelo para la administración de la información La característica más original consiste en que el directorio de especialistas está entrelazado con una lista de las principales enfermedades metabólicas hereditarias (poco más de 200), de forma que permite conocer la experiencia de cada especialista, o identificar a aquellos con experiencia en una enfermedad dada. Esta lista proviene de la utilizada por la MIN para su directorio de médicos. En REDLAEM, los especialistas están agrupados por países, para facilitar la consulta. Esto permite, por ejemplo, buscar médicos en un cierto país, con experiencia en alguna de estas raras y complejas enfermedades, ya sea para la atención de un paciente, para buscar un colaborador en un proyecto de investigación, o para enviar a un estudiante para entrenamiento y capacitación. De forma similar están entrelazados el directorio de laboratorios con una lista de las principales pruebas de laboratorio para el diagnóstico y el seguimiento de estos pacientes, pudiendo también practicarse las búsquedas por países.

El boletín de mensajes (BBS) permite, por ejemplo, realizar consultas sobre pacientes, solicitar apoyos, proporcionar información sobre cursos y congresos, etc. En la sala virtual de discusión se hace posible el diálogo en tiempo real entre especialistas. Cada especialista (o laboratorio) es su propio administrador, de forma que la información de cada uno sólo puede ser actualizada, ampliada o corregida por el propio especialista o laboratorio, quien es el único responsable de la información referente a él.

Cuando, a través del sistema, se envían solicitudes de inscripción; estas son enviadas automáticamente, vía correo electrónico, al administrador del sistema. En caso de aprobarse la solicitud de registro, el administrador es capaz de realizar esta tarea con el uso del sistema administrador. Este sistema, además, ofrece las opciones de modificar la información registrada, borrar registros y eliminar los mensajes que ya hayan sido atendidos.

Por último, aunque esta página está dirigida fundamentalmente a especialistas de habla española, cuenta también con una versión en inglés, lo que hace posible su consulta por personas que no hablen nuestro idioma.

4.1.3.3 Navegación (uso del sistema).

Todos los sitios desarrollados para Web hacen uso del hipertexto, esto permite al usuario pasar de una página a otra o de un sitio a otro con solo hacer click en la liga deseada. Por eso se debe buscar que el usuario llegue a la información que busca, lo más precisamente posible.

Pensando en que los usuarios del sistema tienen mínimos conocimientos de computación, y posiblemente de los servicios ofrecidos en Internet, como Web, la navegación a través del sitio deberá ser lo más clara y fácil posible, evitando al máximo mostrar demasiada información u opciones que puedan confundir al usuario.

El usuario debe contar siempre con la opción de regresar al menú principal o al menú anterior que presente más de una opción a elegir; esto pensando en coartar lo menos posible la capacidad de decisión del usuario, y previendo que el usuario elija una opción que no le sea útil en ese momento.

4.1.3.4 Servicios.

- *Información.*

En el sitio se puede localizar información sobre especialistas y sobre laboratorios.

Se ofrece información sobre datos generales de cada especialistas, además de las enfermedades en las que ese especialista en particular tiene experiencia.

En cuanto a los laboratorios, se ofrece información general y las pruebas que cada laboratorio realiza.

- *Búsqueda.*

La información sobre especialistas se puede buscar por el nombre del especialista o por el país en donde radica el especialista.

Además, se puede localizar información sobre los especialistas con experiencia en determinada enfermedad, por el nombre de la enfermedad.

La información sobre laboratorios se puede buscar por el nombre del laboratorio o por el país en el que se localiza el laboratorio.

Se puede localizar información sobre los laboratorios que realizan determinada prueba, por el nombre de la prueba.

- *Actualización.*

El usuario que decida modificar o actualizar los datos que tiene registrados, lo puede hacer; se mostrará una forma conteniendo los datos registrados; el usuario puede modificar o eliminar los datos que considere necesario, y grabar su registro con los datos adecuados.

El usuario puede actualizar su experiencia registrada, ya que se muestra una lista con todas las enfermedades metabólicas hereditarias, y el usuario selecciona las enfermedades en las cuales tenga experiencia.

Se ofrecen las mismas opciones para los laboratorios y las pruebas que realiza cada uno.

- *Administración.*

Un representante de REDLAEM en México se encargará de administrar la información contenida en el sistema, además de vigilar que los especialistas hagan buen uso del sistema, y lo más importante, cuidar la consistencia de la información.

Entre otras actividades se debe cuidar que la base de datos contenga información de especialistas y laboratorios que aun pertenezcan a REDLAEM, que los registros no presenten demasiados campos en blanco, eliminar los mensajes atrasados o no relacionados con los temas que se abordan en REDLAEM, revisar y atender las solicitudes de registro al sistema; además de mantener una relación de trabajo estrecha con la Coordinación de Servicios de Red para reportar posibles deficiencias en la prestación del servicio, y se tomen las medidas pertinentes.

4.1.3.5 Estructura del sitio

El diagrama 4.1 muestra la estructura general del sistema REDLAEM, en la parte del usuario y el diagrama 4.2 muestra la estructura del módulo de administración del sistema, al cual tiene acceso solo el representante de REDLAEM en México.

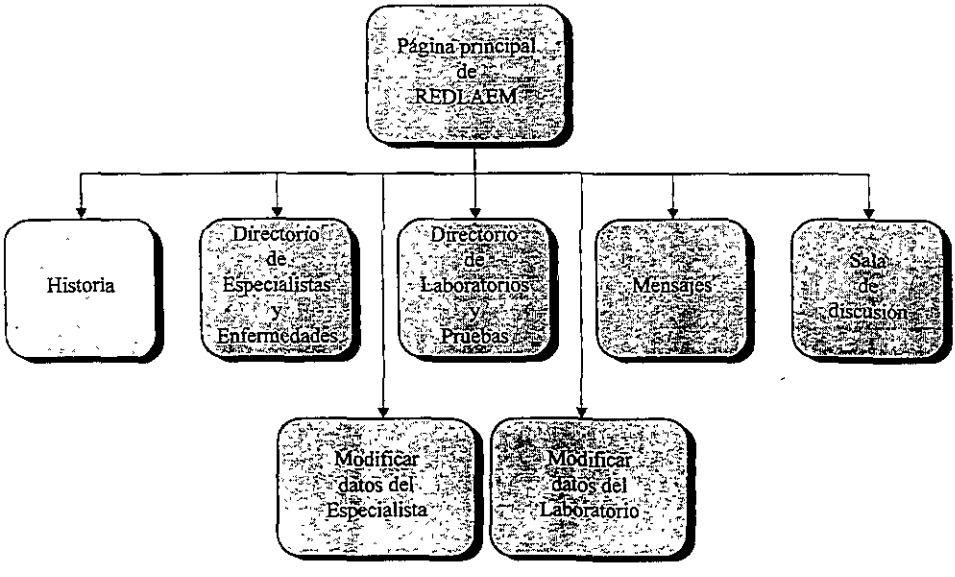


Diagrama 4.1 Estructura general del sistema REDLAEM, se muestran las opciones que tiene el usuario.

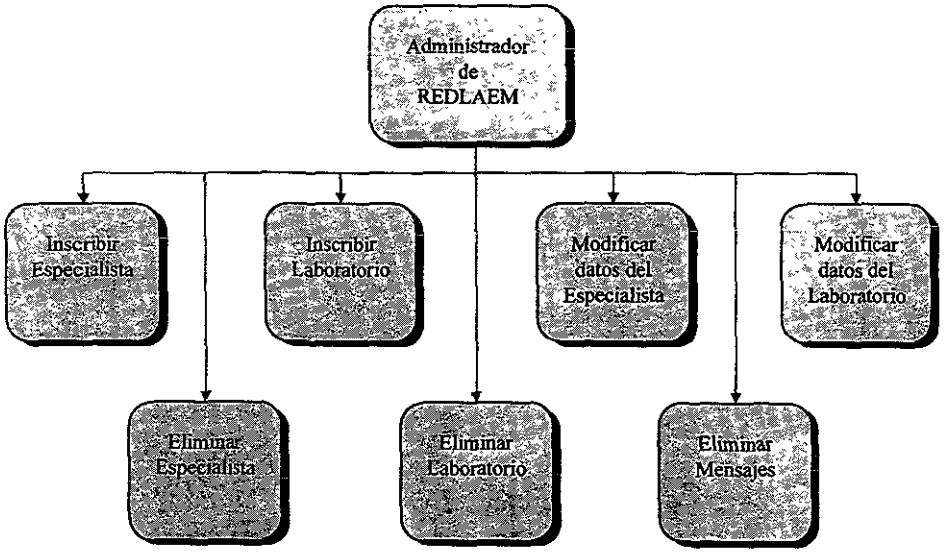


Diagrama 4.2 Estructura general del módulo de administración del sistema.

Opción Directorio de Especialistas y Enfermedades

El diagrama 4.3 muestra la estructura general del servicio de Directorio de especialistas y enfermedades.

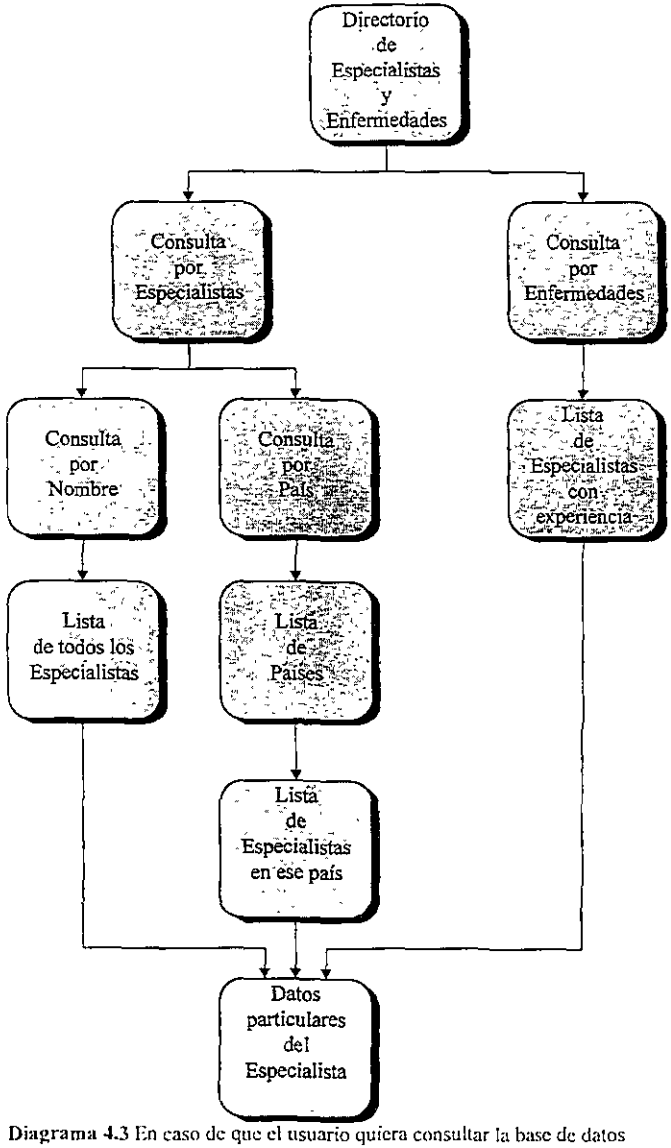


Diagrama 4.3 En caso de que el usuario quiera consultar la base de datos por Especialistas y Enfermedades

Opción: Directorio de Laboratorios y Pruebas

El diagrama 4.4 muestra la estructura general del servicio de Directorio de Laboratorios y Pruebas.

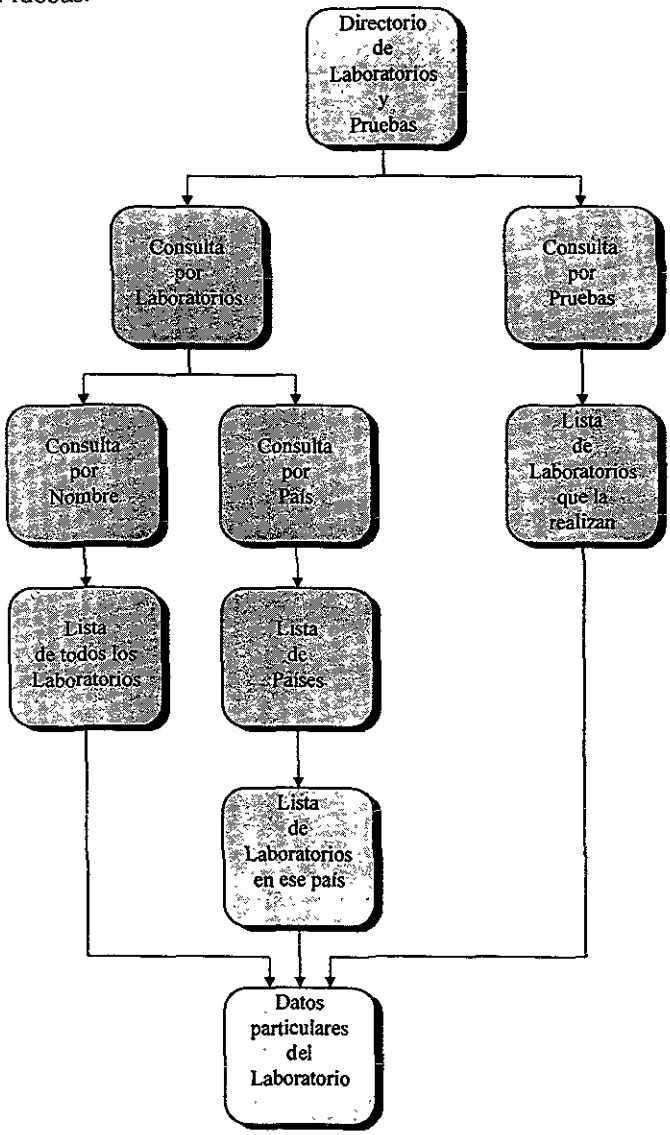


Diagrama 4.4 En caso de que el usuario quiera consultar la base de datos por Laboratorios y Pruebas.

Opción: Boletín de mensajes

El diagrama 4.5 muestra la estructura general del servicio de Boletín de Mensajes.

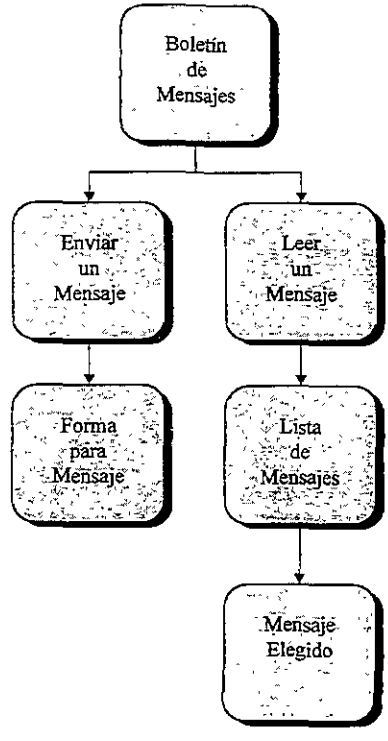


Diagrama 4.5 En caso de que el usuario quiera utilizar el servicio de

Opción: Modificar datos del Especialista.

El sistema REDLAEM ofrece al usuario la posibilidad de modificar sus datos, ver diagrama 4.6.

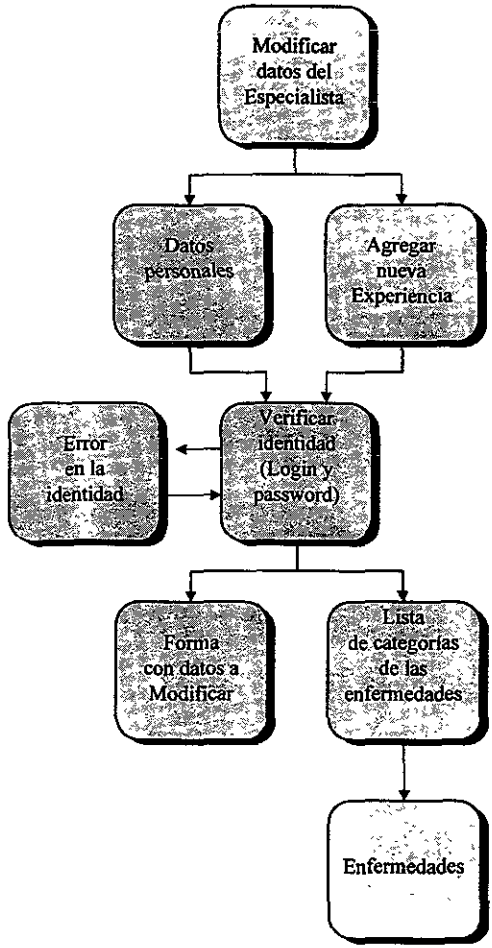


Diagrama 4.6 Estructura general para la opción de Modificar Datos de Especialista.

Opción: Modificar datos del Laboratorio.

El sistema REDLAEM ofrece al usuario la posibilidad de modificar sus datos, ver diagrama 4.7.

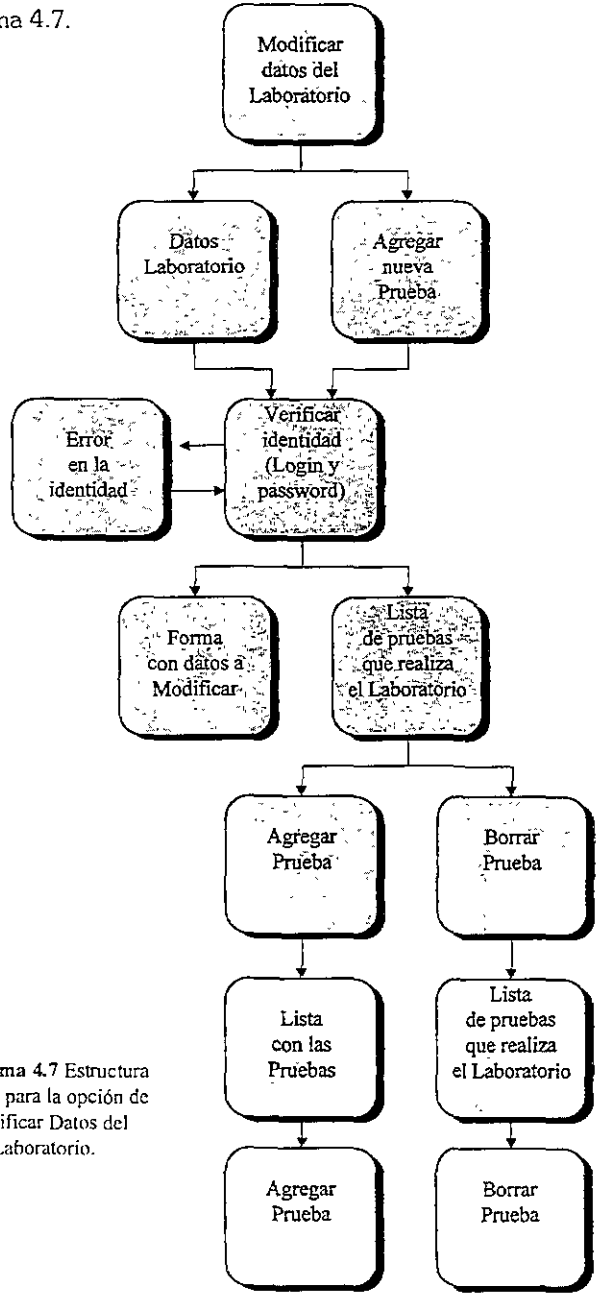


Diagrama 4.7 Estructura general para la opción de Modificar Datos del Laboratorio.

4.1.3.6 La Base de Datos

La información recopilada, que será consultada y modificada por los usuarios del sistema; y depurada por el administrador, se almacena en una sola base de datos (**DATOS_MEDICO**). Esta base de datos contiene las siguientes tablas:

- **DATOS_LAB** (ver Tabla 4.1)
- **DATOS_MEDICO** (ver Tabla 4.2)
- **ENFERMEDAD** (ver Tabla 4.3)
- **EXPE_ENFE** (ver Tabla 4.4)
- **MENSAJES** (ver Tabla 4.5)
- **PRU_LAB** (ver Tabla 4.6)

A continuación se describe la estructura de cada tabla de datos:

DATOS_LAB

campo	tipo	longitud
cve	char	5
jefe	char	80
departamento	char	40
dependencia	char	40
direccion	char	60
pais	char	5
estado	char	20
telefono	char	25
fax	char	20
email	char	60
pruebas	char	200
ciudad	char	40
password	char	10

Tabla 4.1 Muestra el nombre, el tipo y la longitud de los campos que fueron declarados en SYBASE.

DATOS_MEDICO

campo	tipo	longitud
cve	char	5
nombre	char	40
pais	char	15
especialidad	char	35
direccion	char	70
estado	char	15
codigo	char	15
telefono	char	25
fax	char	15
email	char	40
ciudad	char	40
password	char	10

Tabla 4.2 Muestra el nombre, el tipo y la longitud de los campos que fueron declarados en SYBASE.

ENFERMEDAD

campo	tipo	longitud
nombrelab	char	40
enfermedad	char	60

Tabla 4.3 Muestra el nombre, el tipo y la longitud de los campos que fueron declarados en SYBASE.

EXPE_ENFE

campo	tipo	longitud
Cve	char	5
Enfermedad	char	100
Casos	char	3

Tabla 4.4 Muestra el nombre, el tipo y la longitud de los campos que fueron declarados en SYBASE.

MENSAJES

campo	tipo	longitud
cve	char	5
nombre	char	40
tema	char	110
fecha	char	25
cuerpo	char	250
nada	char	3

Tabla 4.5 Muestra el nombre, el tipo y la longitud de los campos que fueron declarados en SYBASE.

PRU_LAB

campo	tipo	longitud
cve	char	5
prueba	char	80

Tabla 4.6 Muestra el nombre, el tipo y la longitud de los campos que fueron declarados en SYBASE.

5. Integración

5.1 Descripción.

Este capítulo abarca el último punto de la metodología para el desarrollo de un sitio Web, *el desarrollo*. En esta etapa se integran todas las herramientas ya descritas en capítulos anteriores para materializar el sistema de información en Web para REDLAEM.

Este sistema de información es dinámico, la información contenida en la base de datos puede modificarse en cualquier momento. Por eso es necesario desarrollar, no sólo páginas Web (html), sino también programas CGI´s (Perl) y programas que sirvan como interfaz con las Bases de Datos (C).

El formato que se adoptó para ilustrar el sitio Web de REDLAEM es el siguiente:

- Pantalla o página que el usuario ve en cada opción (imagen).

- Nombre de los programas que se ejecutan en cada caso y una breve explicación del funcionamiento de estos, como: parámetros que necesita, funciones y/o subprogramas que se llaman a ejecución, y alguna nota en caso de ser necesario.

- El código fuente de los programas (Disquete anexo).

5.2 El funcionamiento.

En el siguiente diagrama (ver diagrama 5.1) se explica como interactuan los programas creados, para realizar todas las tareas ordenadas por el usuario.

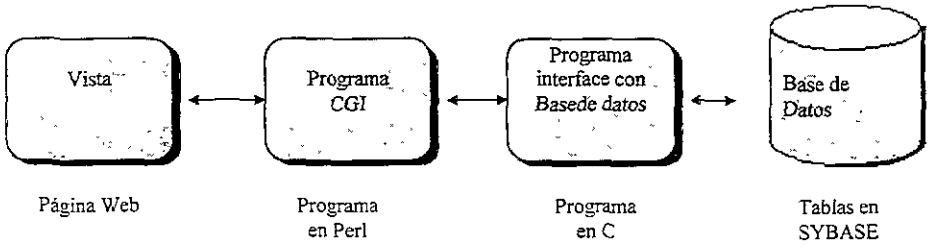


Diagrama 5.1 Para el usuario, la interacción entre los programas y las tareas que cada uno de ellos realiza es transparente.

5.3 El sitio.

Página principal.

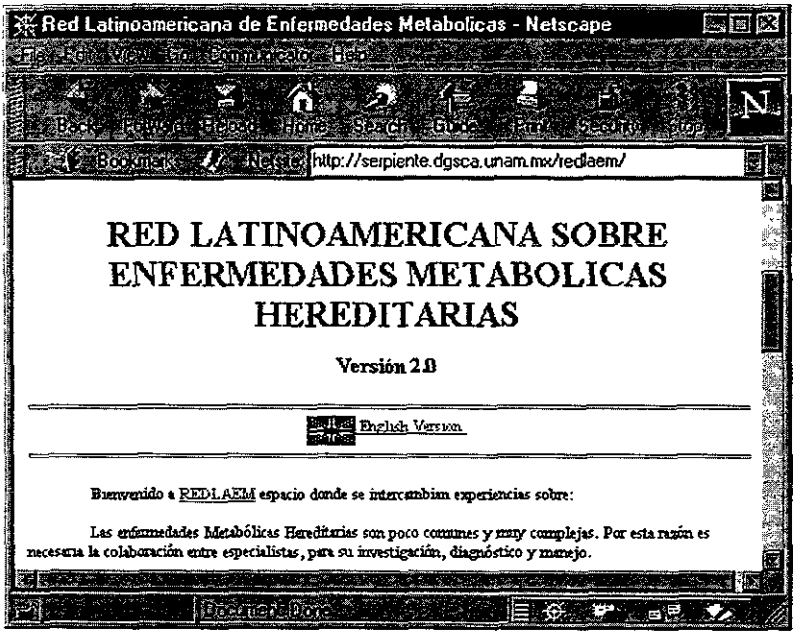


Imagen 5. 1

El archivo que da forma a la página es:

- **index.html** (ver Imagen 5.1)

Cada opción es presentada de manera subrayada; estos son hipervínculos[9] (ligas) hacia otras páginas, por ejemplo: la opción **REDLAEM** es un hipervínculo hacia el archivo **redlaem.html**.

[9] Se conoce como hipervínculo a la liga virtual que une a dos documentos en Web. Basta hacer click sobre el hipervínculo, para pasar de un documento web a otro.

Historia de REDLAEM.

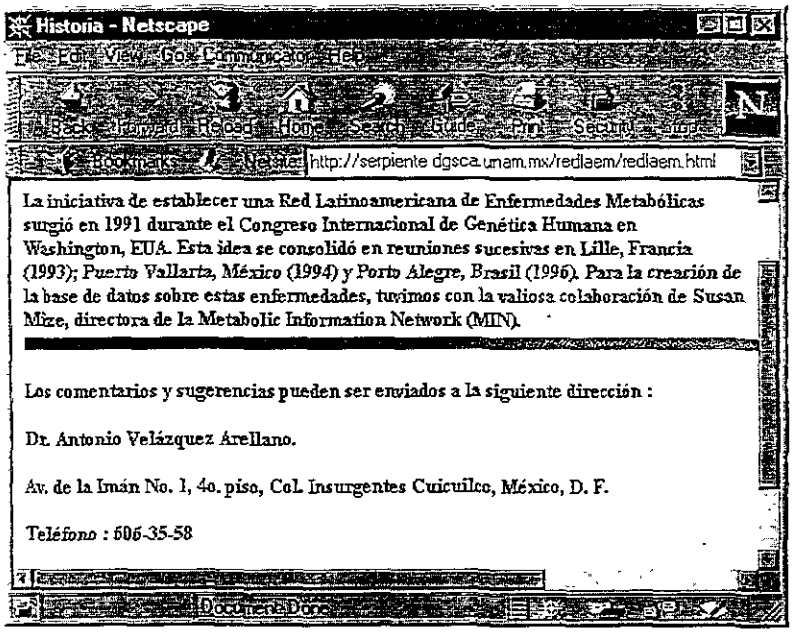


Imagen 5. 2

El archivo que contiene la historia es:

- ***redlaem.html*** *(ver Imagen 5.2)*

1) Directorio de Especialistas y Enfermedades.

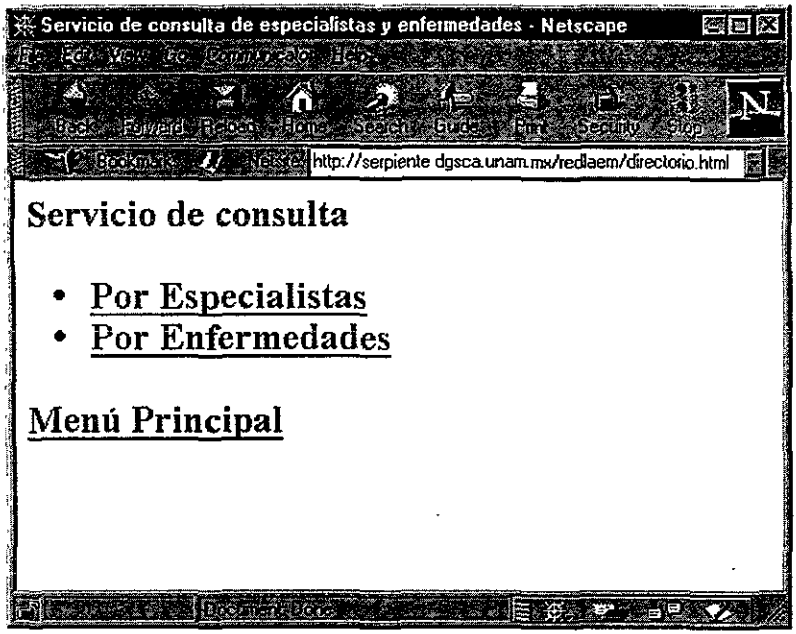


Imagen 5. 3

El programa que forma la página es:

- **directorio.html** (ver Imagen 5.3)

El sistema ofrece dos posibles tipos de búsqueda de la información:

- Por Especialistas
- Por enfermedades

a) Buscar por especialistas.

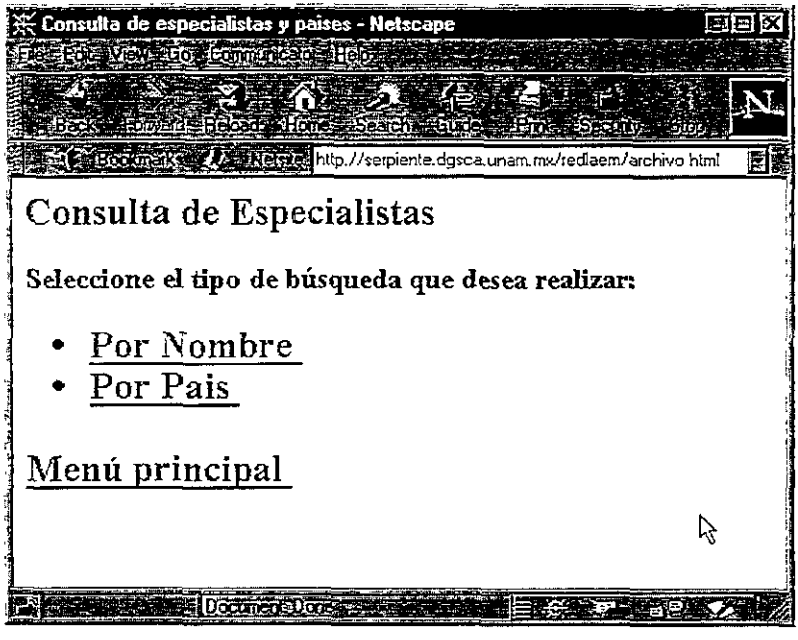


Imagen 5. 4

El archivo que forma la página es:

- **archivo.html** (ver Imagen 5.4)

El usuario puede buscar el especialista por dos criterios distintos:

- Por Nombre
- Por País

i) Buscar Especialistas por nombre.

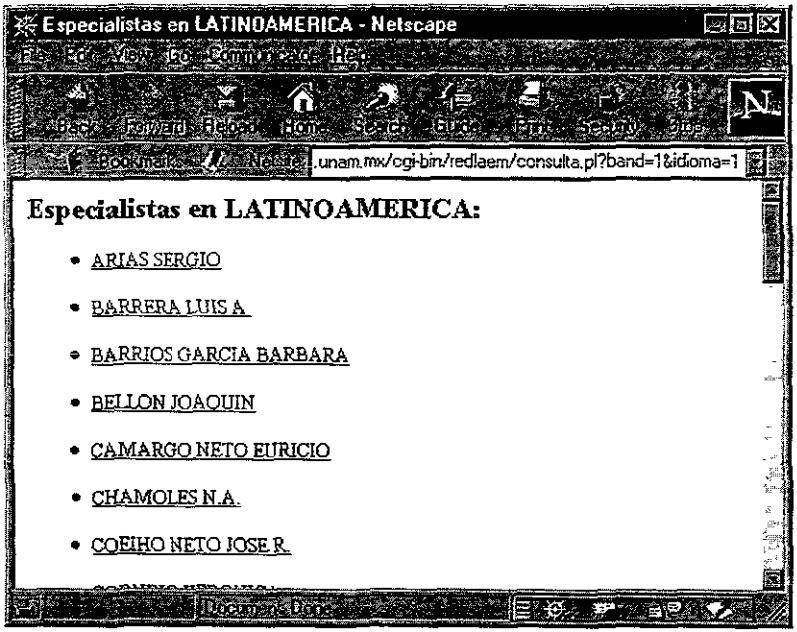


Imagen 5. 5

El programa CGI que se ejecuta es:

- **consulta.pl** Recibe como parámetros: el tipo de consulta y el idioma.

El programa, al recibir estos parámetros, hace una llamada de ejecución del programa **hacetod_ya**. Este busca en la tabla **DATOS_MEDICO** y regresa la clave y el nombre de todos los especialistas, ordenados alfabéticamente.

Se muestra una lista con los nombres de los especialistas (ver Imagen 5.5). Si se quieren ver los datos registrados, se ejecuta el programa:

- **restr.pl** Recibe como parámetro: El nombre del especialista.

El programa hace una llamada de ejecución del programa **hacecon**. Este busca en la tabla **DATOS_MEDICO** y regresa todos los datos del especialista (ver Imagen 5.6).

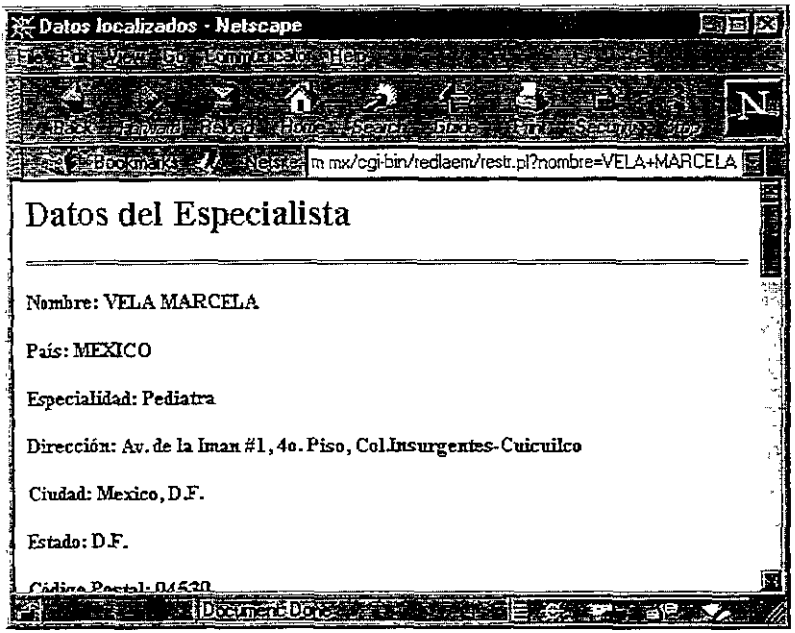


Imagen 5. 6

Además, se llama a ejecución el programa **hacexpe_ya** con la clave del especialista como parámetro. Este busca en la tabla **EXPE_ENFE** y regresa el nombre de todas las enfermedades registradas por ese especialista (ver Imagen 5.7).

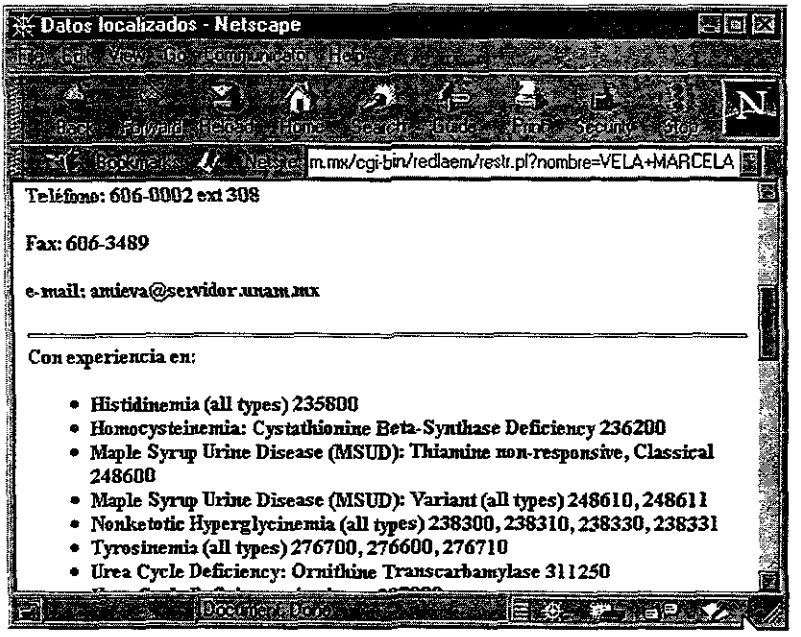


Imagen 5. 7

ii) Buscar Especialistas por país.

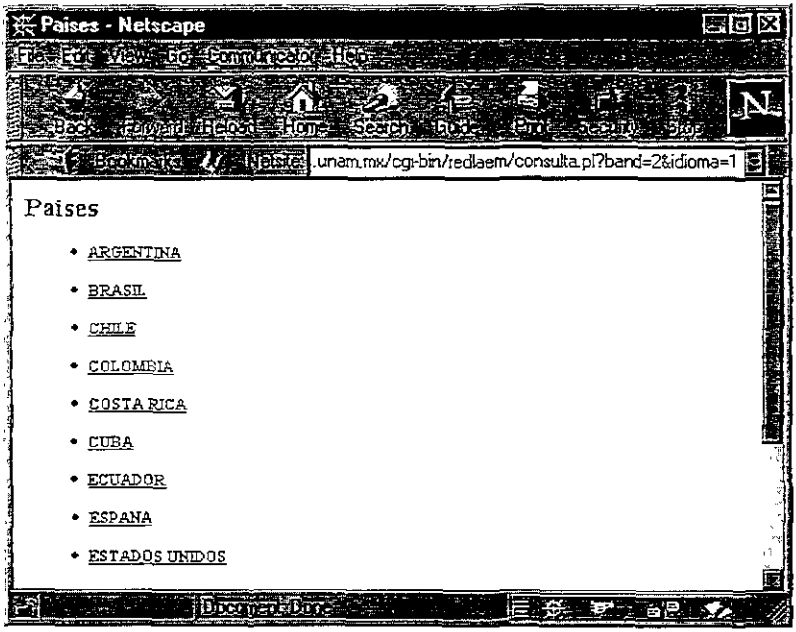


Imagen 5. 8

El programa CGI que se ejecuta es:

- **consulta.pl** Recibe como parámetros el tipo de consulta y el idioma.

Se llama a ejecución el programa **hacepais**, este busca en la tabla **DATOS_MEDICO** y regresa todos los países, sin repetirlos y ordenados alfabéticamente (ver Imagen 5.8).

Se muestra una lista con los países que tienen al menos un especialista en REDLAEM. Si se desea ver una lista de los especialistas en cada país, se ejecuta el programa:

- **lista_pais.pl** Recibe como parámetros el país y el idioma (1).

Se llama a ejecución el programa **hacepaisnom**. Este busca en la tabla **DATOS_MEDICO** y regresa el nombre y el país de todos los especialistas, ordenados por el nombre

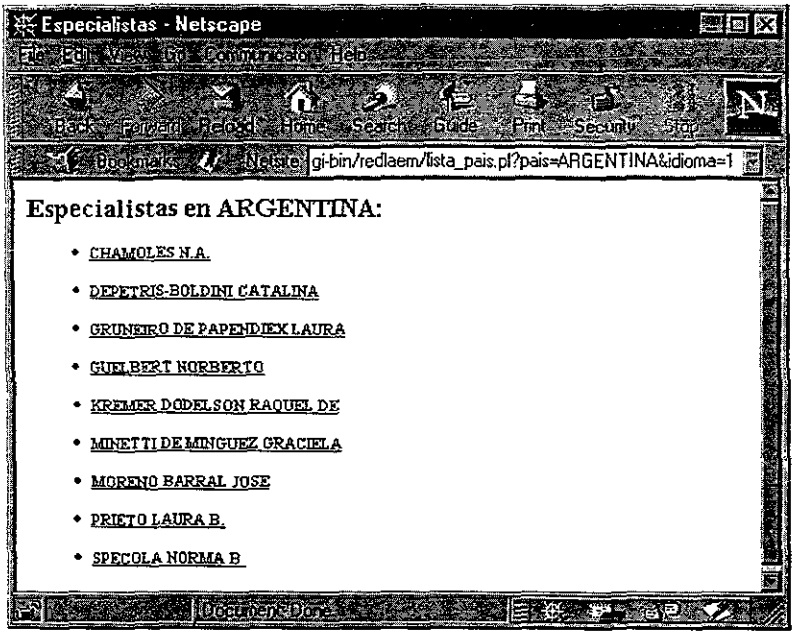


Imagen 5. 9

Se muestra una lista con los nombres de los especialistas (ver Imagen 5.9). Si se quieren ver los datos registrados, se ejecuta el programa:

- **restr.pl** Recibe como parámetros: El nombre del especialista.

El programa hace una llamada de ejecución del programa **hacecon**. Este busca en la tabla **DATOS_MEDICO** y regresa todos los datos del especialista (ver Imagen 5.6). Además, se llama a ejecución el programa **hacexpe_ya** con la clave del especialista como parámetro. Este busca en la tabla **EXPE_ENFE** y regresa el nombre de todas las enfermedades registradas por ese especialista (ver Imagen 5.7).

b) Por enfermedades.

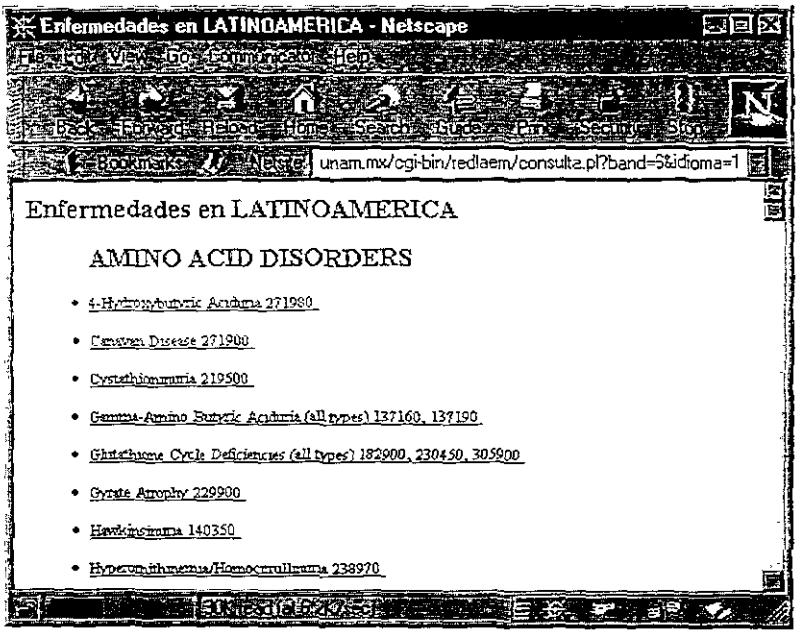


Imagen 5. 10

El programa CGI que se ejecuta es:

- **consulta.pl** Recibe como parámetros: el tipo de consulta y el idioma.

Se llama a ejecución el programa **hacenfelist**. Este busca en la tabla **EXPE_ENFE** y regresa la clasificación y el nombre de todas las enfermedades registradas.

Se muestra una lista de todas las enfermedades (ver Imagen 5.10). Si se desea ver una lista de los especialistas que tienen experiencia en esa enfermedad (ver Imagen 5.11), se ejecuta el programa:

- **rest.pl** Recibe como parámetro: el nombre de la enfermedad y el idioma.

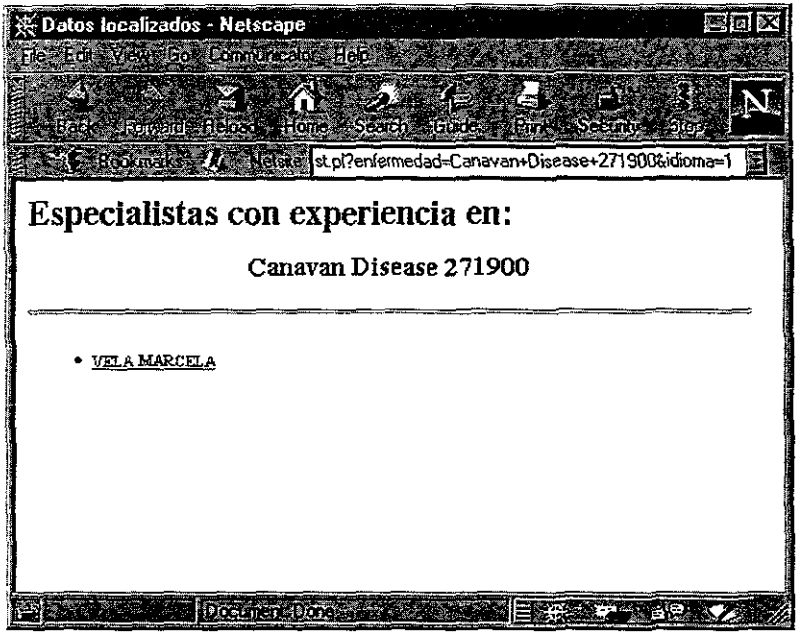


Imagen 5. 11

Si se quieren ver los datos registrados, se ejecuta el programa:

- **restr.pl** Recibe como parámetro: El nombre del especialista.

El programa hace una llamada de ejecución del programa **hacecon** .Este busca en la tabla **DATOS_MEDICO** y regresa todos los datos del especialista (ver Imagen 5.6). Además, se llama a ejecución el programa **hacexpe_ya** con la clave del especialista como parámetro. Este busca en la tabla **EXPE_ENFE** y regresa el nombre de todas las enfermedades registradas por ese especialista (ver Imagen 5.7).

2) Directorio de especialistas y enfermedades.

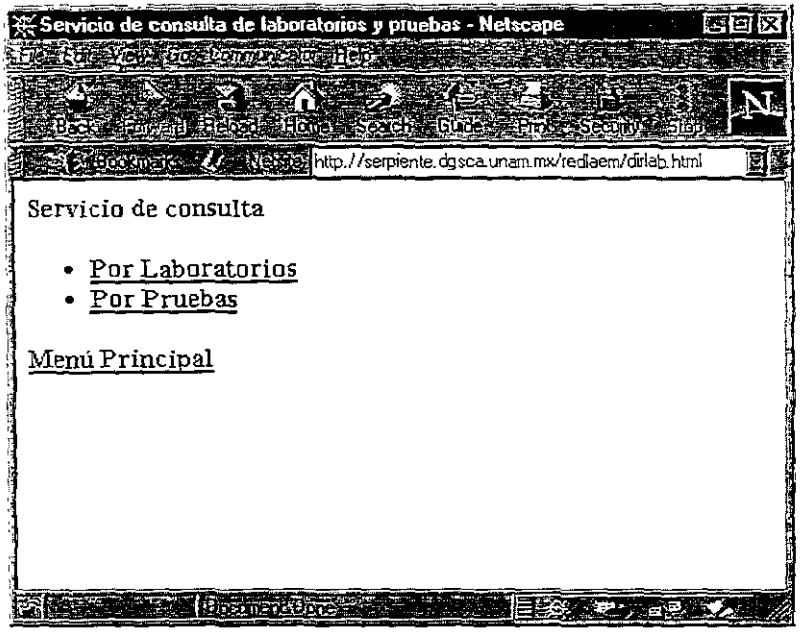


Imagen 5. 12

El archivo que forma la página es:

- **dirlab.html** (ver Imagen 5.12).

El sistema ofrece 2 posibles tipos de búsqueda de la información:

- Por Laboratorio
- Por Pruebas

b) Buscar por laboratorios.

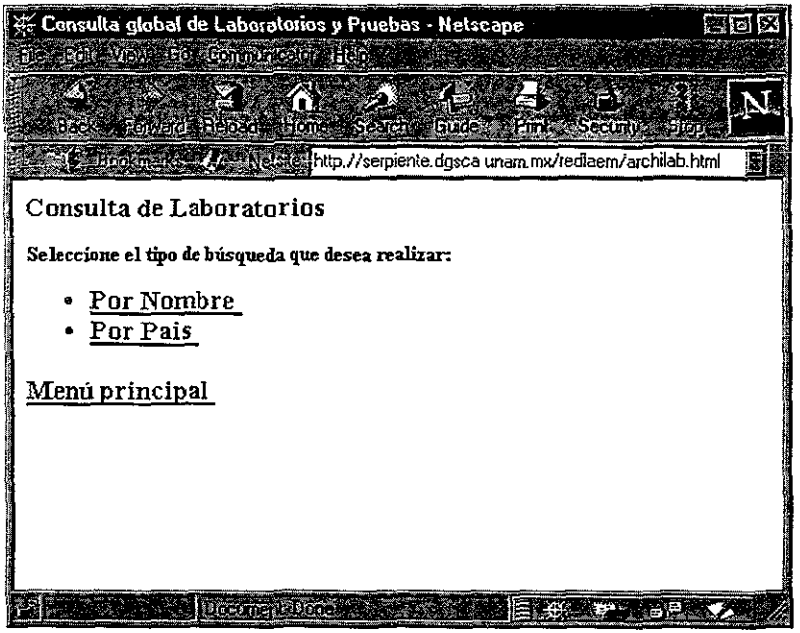


Imagen 5. 13

El archivo que forma la página es:

- **archilab.html** (ver Imagen 5.13).

El usuario puede buscar el laboratorio por dos criterios diferentes.

i) **Buscar laboratorios por nombre.**

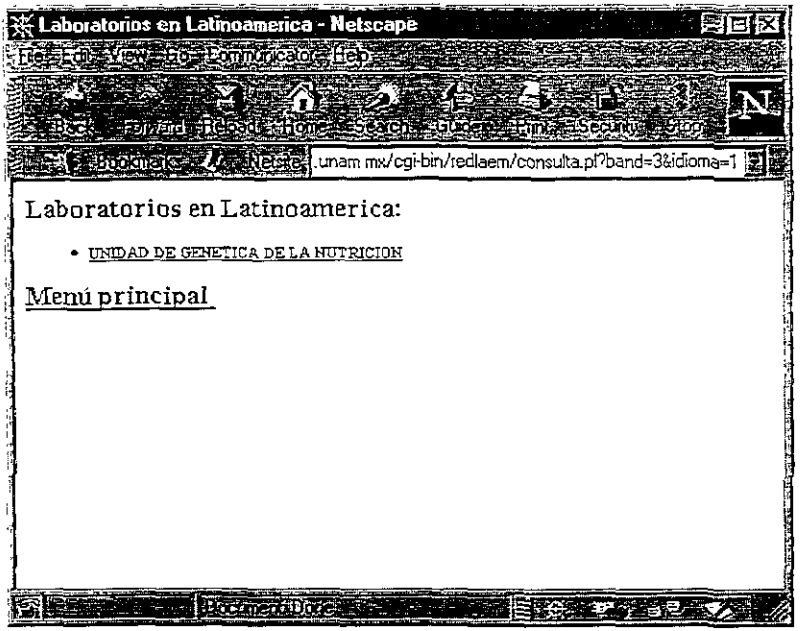


Imagen 5.14

El programa CGI que se ejecuta es:

- **consulta.pl** Recibe como parámetros el tipo de búsqueda y el idioma.

Se realiza una llamada de ejecución al programa **todlab_ya**. Este busca en la tabla **DATOS_LAB** y regresa el nombre de todos los laboratorios registrados, ordenados alfabéticamente por nombre.

Se muestra una lista con todos los laboratorios registrados (ver Imagen 5.14). Si se desea ver los datos particulares del laboratorio, se ejecuta el programa:

- **labora.pl** Recibe como parámetro el nombre del laboratorio

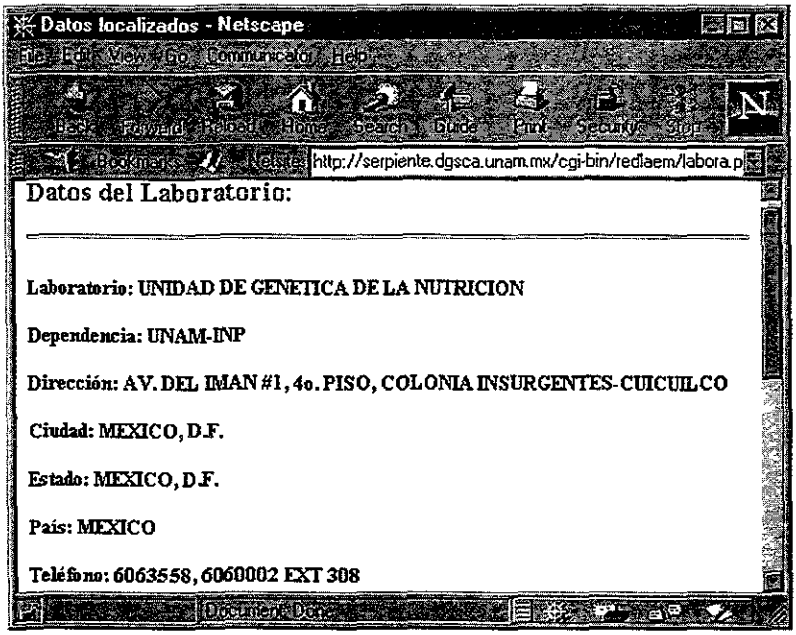


Imagen 5.15

El programa hace una llamada de ejecución del programa **con-lab**, este busca en la tabla **DATOS_LAB** y regresa todos los datos particulares del laboratorio (ver Imagen 5.15).

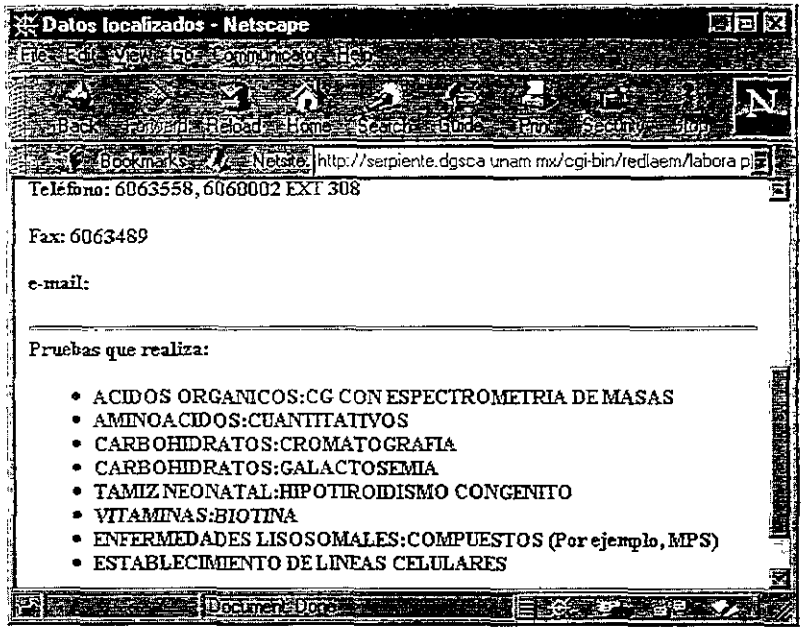


Imagen 5.16

Además, se llama a ejecución el programa **conpru_lab** con la clave del laboratorio como parámetro. Este busca en la tabla **PRU_LAB** y regresa el nombre de todas las pruebas que el laboratorio realiza (ver Imagen 5.16).

b) Buscar laboratorios por país.

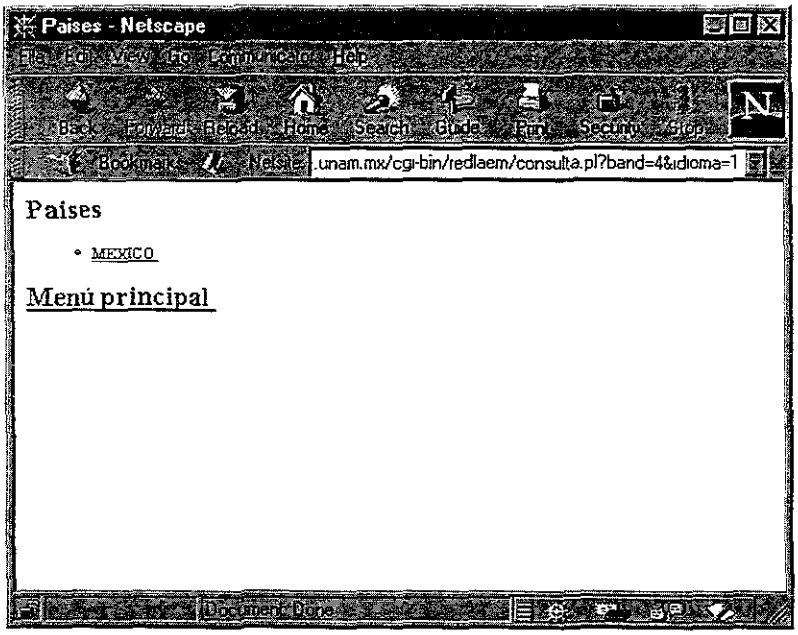


Imagen 5.17

El programa CGI que se ejecuta es:

- **consulta.pl** Recibe como parámetros el tipo de búsqueda y el idioma.

Se realiza una llamada de ejecución del programa **labpais**, este busca en la tabla **DATOS_LAB** y regresa todos los países, sin repetirlos y ordenados alfabéticamente (ver Imagen 5.17).

Se muestra una lista con los países que tienen al menos un laboratorio registrado. Si se desea ver los laboratorios de ese país en particular, se ejecuta el programa:

- **labs_pais.pl** Recibe como parámetros el país y el idioma.

Se llama a ejecución el programa **conpais_ya**, el cual busca en la tabla **DATOS_LAB** y regresa los laboratorios de ese país ordenados alfabéticamente por nombre (ver

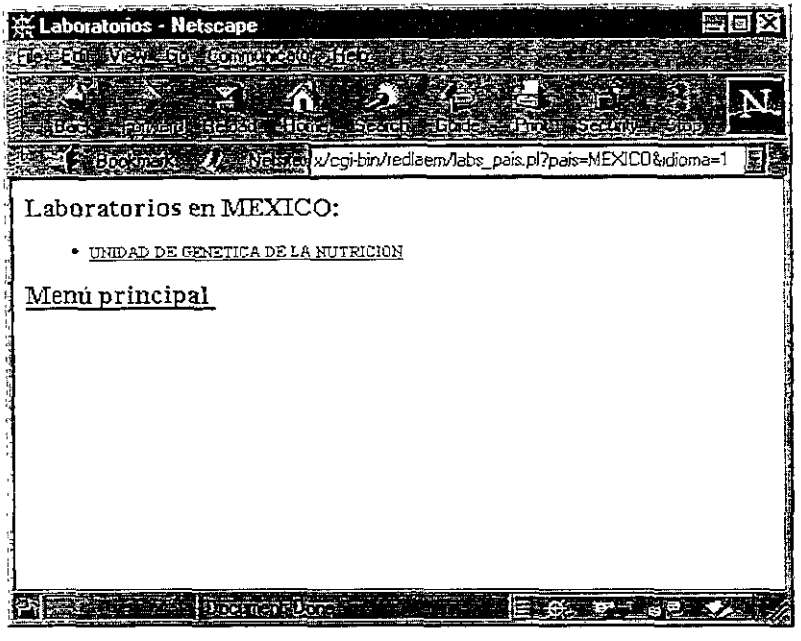


Imagen 5.18).

Imagen 5.18

Si se desea ver los datos particulares del laboratorio (ver Imagen 5.15), se ejecuta el programa:

- **labora.pl** Recibe como parámetro el nombre del laboratorio.

b) Buscar laboratorios por pruebas.

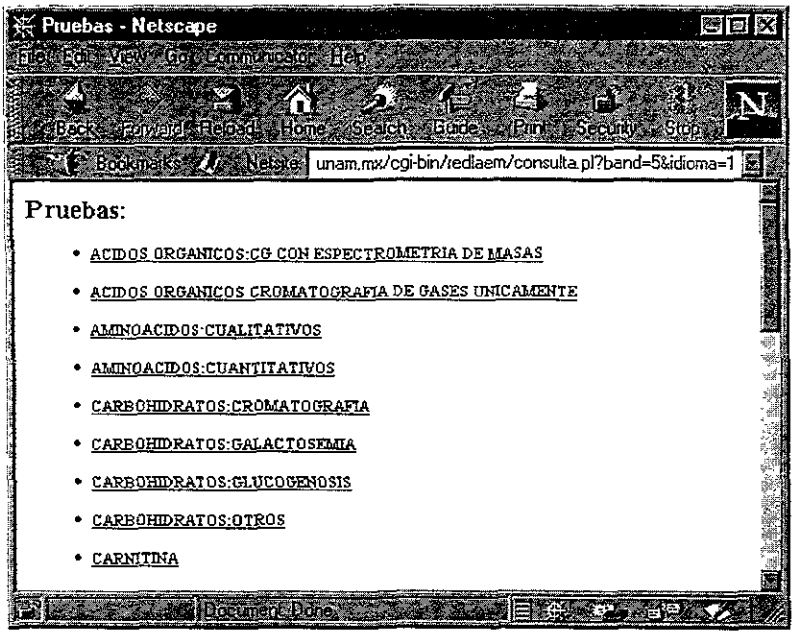


Imagen 5.19

El programa CGI que se ejecuta es:

- **consulta.pl** Recibe como parámetros el tipo de búsqueda y el idioma.

Se lee el archivo **lis_pruebas.txt**, aparece una lista de todas las pruebas que se realizan para este tipo de enfermedades (ver Imagen 5.19). Si se desea ver los laboratorios que realizan esa prueba en particular (ver Imagen 5.20), se ejecuta el programa:

- **rest2.pl** Recibe como parámetros el nombre de la prueba y el idioma.

Se llama al programa **hacepru_ya**, este busca en la tabla **PRU_LAB** y regresa las claves de los laboratorios que realizan esa prueba.

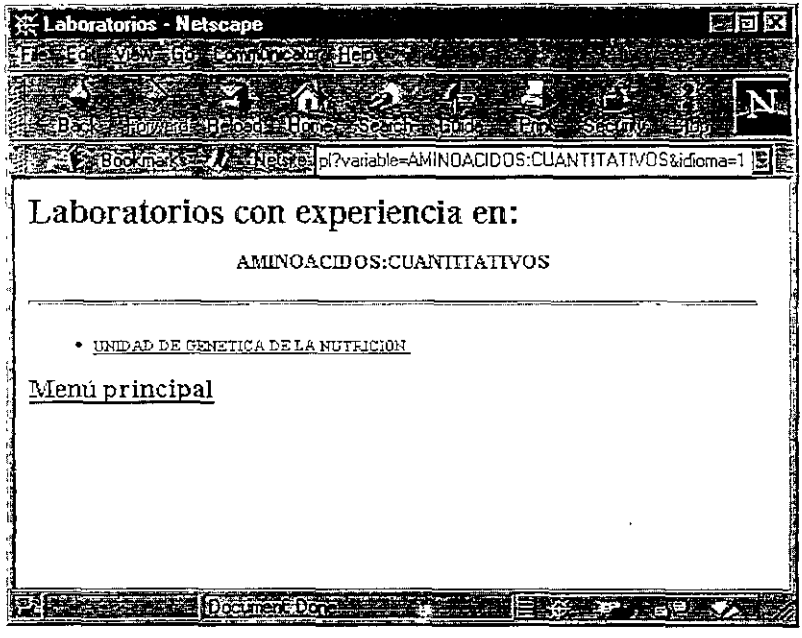


Imagen 5.20

Si se desea ver los datos particulares del laboratorio (ver Imagen 5.15), se ejecuta el programa:

- **labora.pl** Recibe como parámetro el nombre del laboratorio

3) Sala de discusión (Chat).

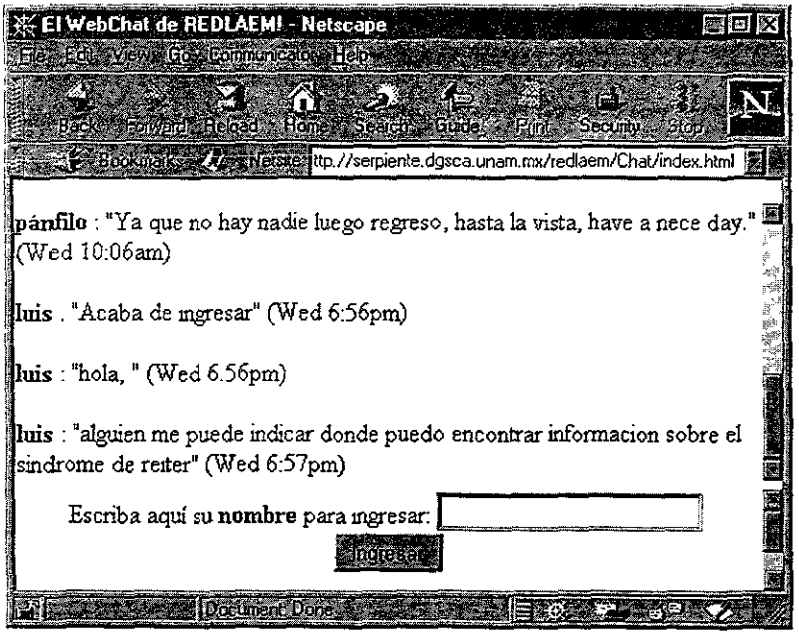


Imagen 5.21

El programa que da forma a la página es: **Chat/index.html** (ver Imagen 5.21).

El programa que recibe el nombre del usuario y los mensajes es: **quickchat.cgi**

Los mensajes son desplegados en la página formada por el programa: **messages.html**

4) Solicitud de ingreso para especialistas.

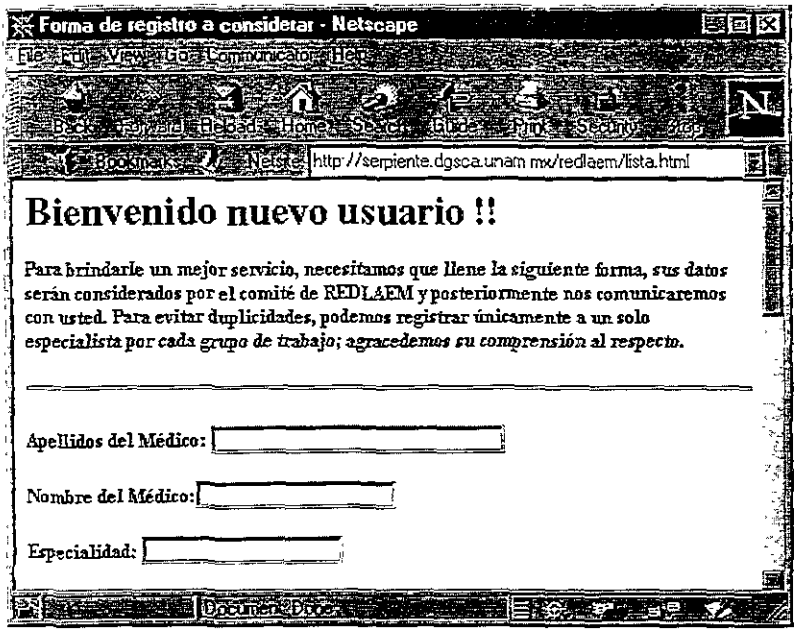


Imagen 5.22

El programa que contiene la forma de registro es:

- **lista.html** (ver Imagen 5.22).

Los datos enviados son recibidos por el programa:

- ***for_temp_cap.pl***

Este programa da, a los datos, el formato adecuado para el manejo de la información del especialista.; manda los datos con formato al programa **aviso.pl**. Este envía los datos por correo electrónico automáticamente al administrador de REDLEM utilizando el programa **sendmail**.

5) Solicitud de ingreso para laboratorios.

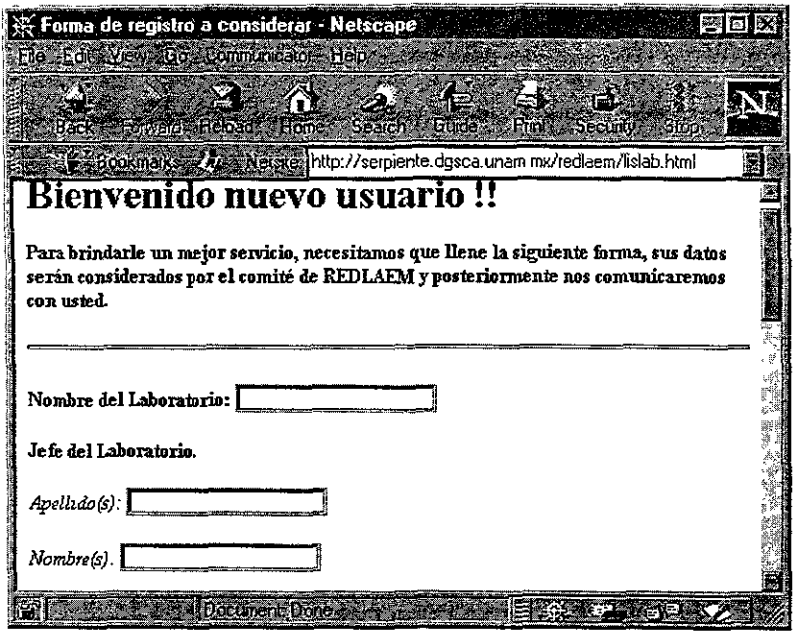


Imagen 5.23

El programa que contiene la forma de registro es:

- **lislab.html** (ver Imagen 5.23)

Los datos enviados son recibidos por el programa:

- **for_temp_lab.pl**

Este programa da, a los datos, el formato adecuado para el manejo de la información del especialista.; manda los datos con formato al programa **aviso2.pl**. Este envía los datos por correo electrónico automáticamente al administrador de REDLEM. Utilizando el programa **sendmail**.

6) Ingresar nueva información sobre especialistas.

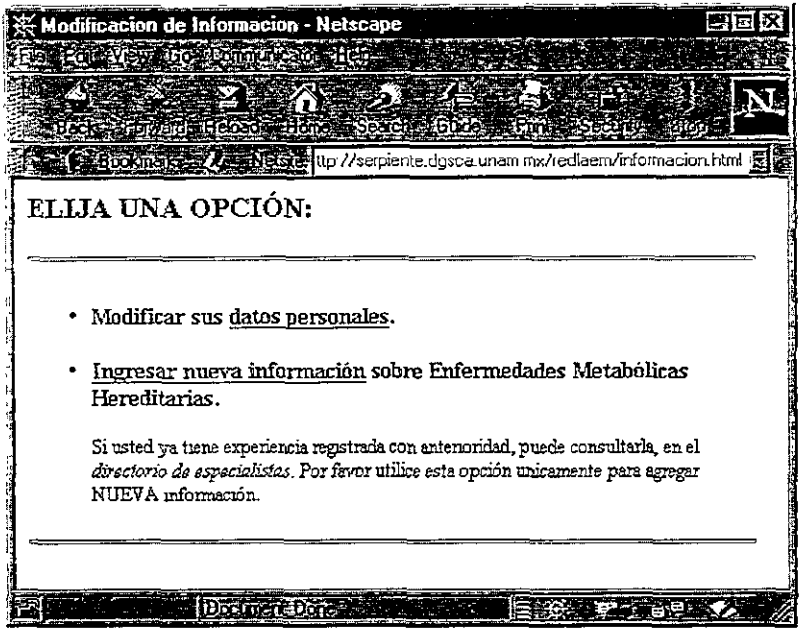


Imagen 5.24

El programa que da forma a la página es:

- **informacion.html** (ver Imagen 5.24)

a) Modificar los datos personales del especialista.

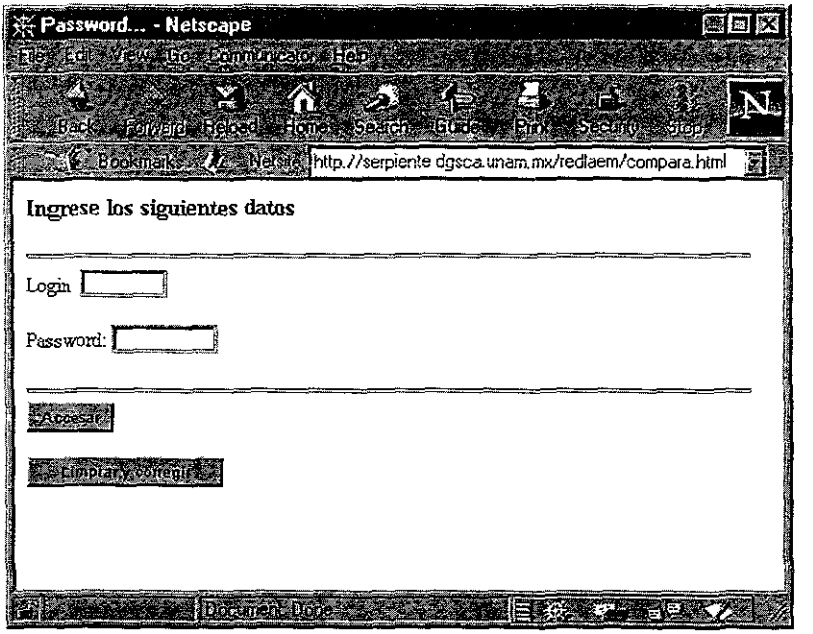


Imagen 5.25

Es necesario validar la identidad del usuario; el programa **compara.html** es una forma que el usuario debe llenar (ver Imagen 5.25). Los datos son enviados al programa **valida.pl**; este llama a ejecución al programa **hacepass** que busca en la tabla **DATOS_MEDICO** y regresa la clave y el password de todos los especialistas. En caso de existir un error, el programa **valida.pl** envía un mensaje.

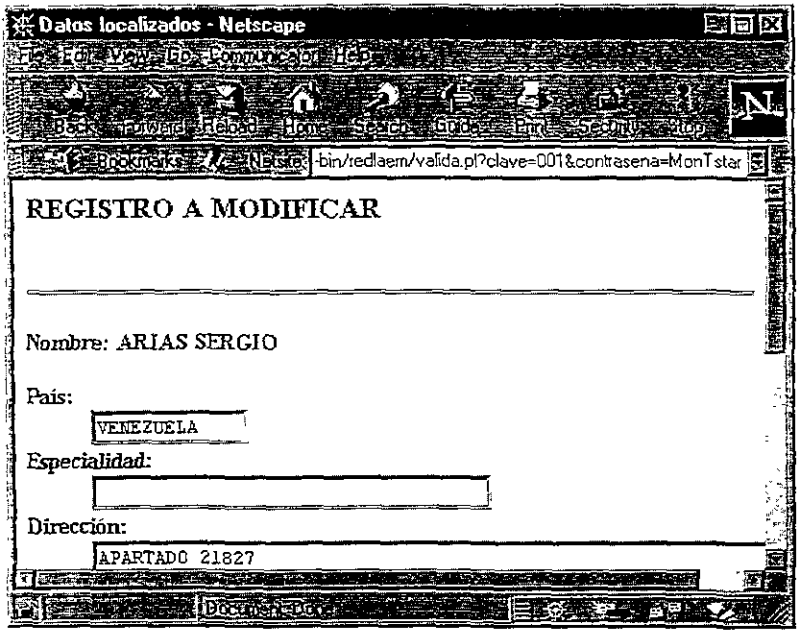


Imagen 5.26

Si se trata de un usuario válido, el programa presenta una forma con los datos a modificar (ver Imagen 5.26). Los datos son enviados al programa: **actualiza.pl** este llama a ejecución a los programas **borramed**, que borra el registro de la tabla **DATOS_MEDICO** por la clave del especialista; y **grabamed**, que graba el registro con los nuevos datos en la tabla **DATOS_MEDICO**.

b) Agregar nueva experiencia.

Es necesario validar la identidad del usuario. El programa **compara3.html** es una forma que el usuario debe llenar. Los datos son enviados al programa **valida3.pl**; este llama a ejecución al programa **hacepass** que busca en la tabla **DATOS_MEDICO** y regresa la clave y el password de todos los especialistas. En caso de existir un error, el programa **valida3.pl** envía un mensaje.

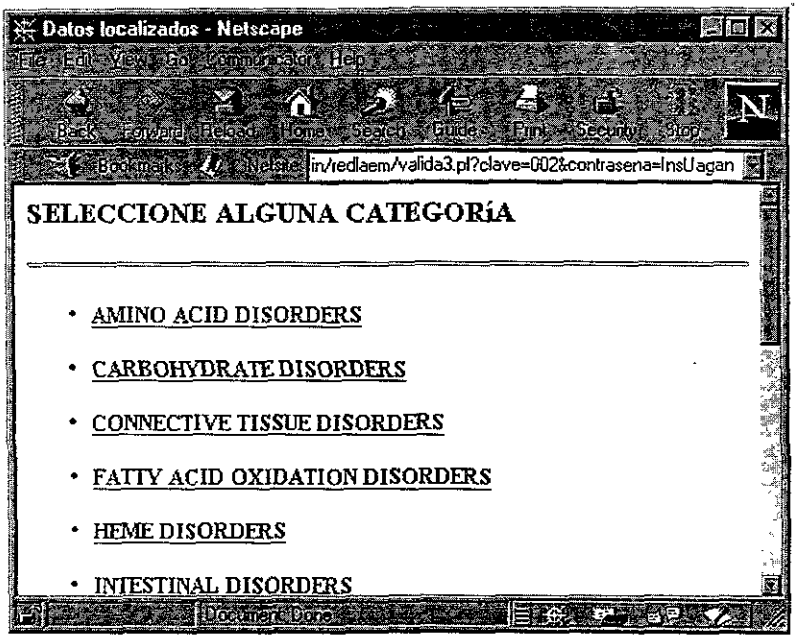


Imagen 5.27

Si se trata de un usuario válido, el programa presenta una lista de las clasificaciones de las enfermedades (ver Imagen 5.27). Para cada clasificación, existe un programa que muestra todas las enfermedades, y el usuario selecciona las que desea agregar (ver Imagen 5.28).

Estos programas son:

amino.pl

carbo.pl

connect.pl

fatty.pl

hemme1.pl

intes1.pl

lyso1.pl

membrane1.pl

metal1.pl

organic1.pl

peroxi1.pl

porphy1.pl

puri.pl

skin.pl

stero1.pl

vita1.pl

Todos estos programas reciben como parámetros: la clave y el password del usuario.

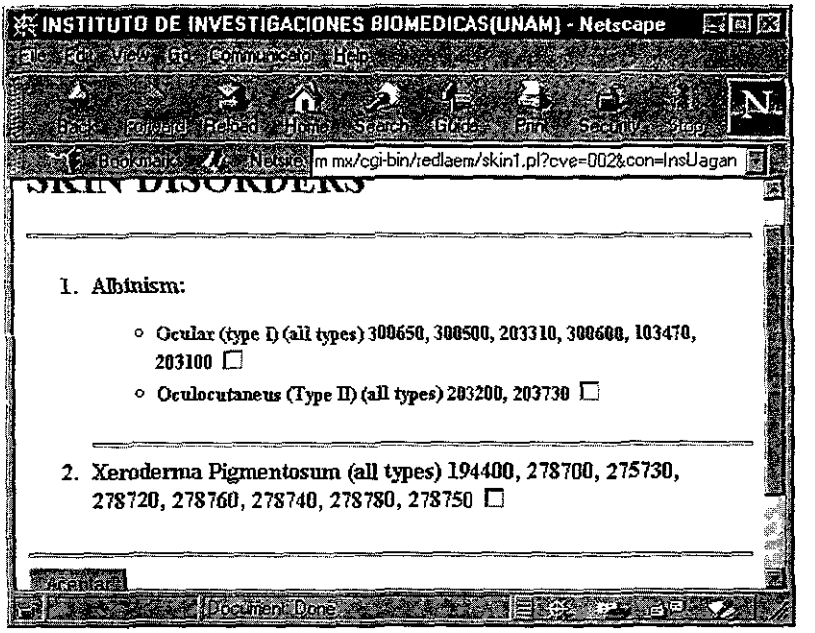


Imagen 5.28

El usuario elige las enfermedades que desea agregar como nueva experiencia. Estos datos son enviados al programa **grabadoc.pl** que recibe como parámetros un número 1 por cada enfermedad agregada; se envía como parámetro el nombre de la enfermedad y se ejecuta el programa **grabadoc** que actualiza la tabla **EXPE_ENFE**.

7) Ingresar nueva información sobre laboratorios.

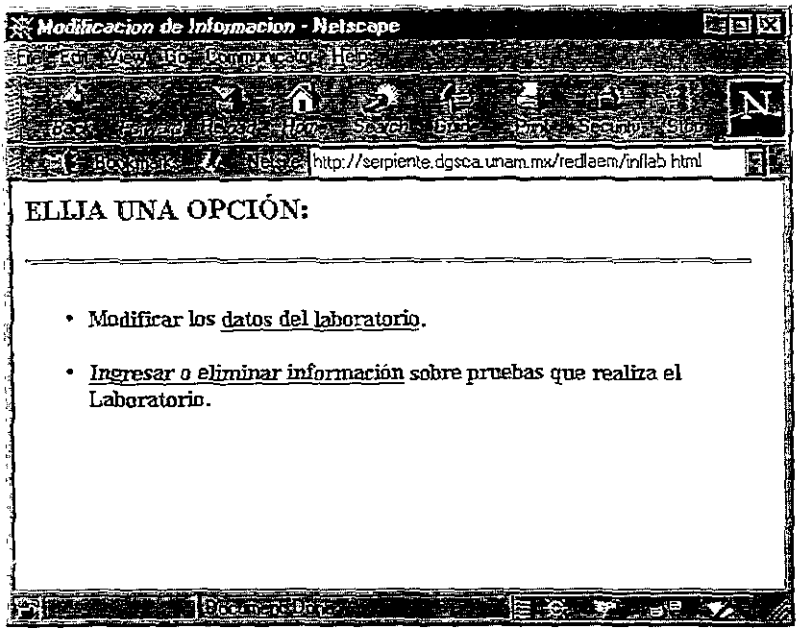


Imagen 5.29

El programa que da forma a la página es:

- **inflab.html** (ver Imagen 5.29)

Las opciones que presenta son:

- Modificar los datos del laboratorio.
- Ingresar o eliminar información sobre pruebas que realiza el laboratorio.

a) Modificar datos del laboratorio.

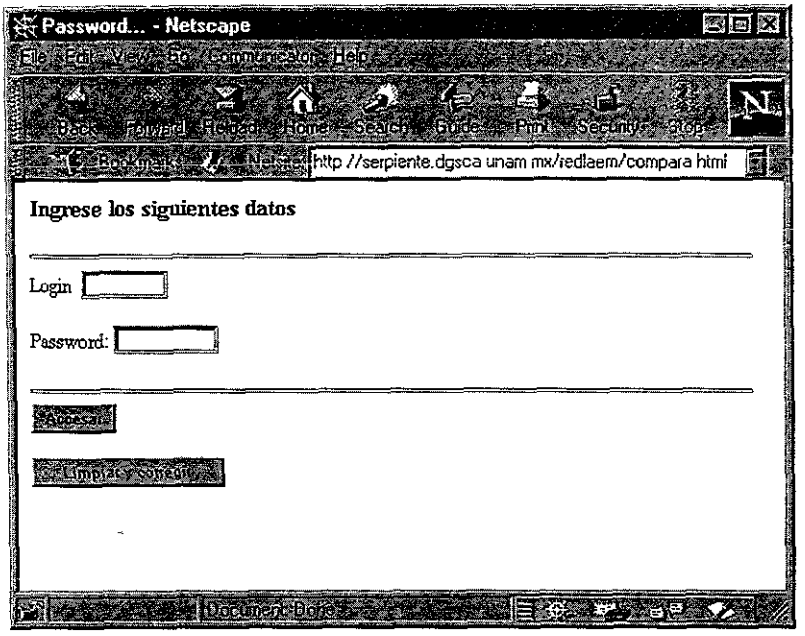


Imagen 5.30

Es necesario validar la identidad del usuario. El programa **comparalab.html** es una forma que el usuario debe llenar (ver Imagen 5.30). Los datos son enviados al programa **validalab.pl**; este llama a ejecución al programa **hacepasslab** que busca en la tabla **DATOS_LAB** y regresa la clave y el password de todos los laboratorios. En caso de existir un error, el programa **validalab.pl** envía un mensaje.

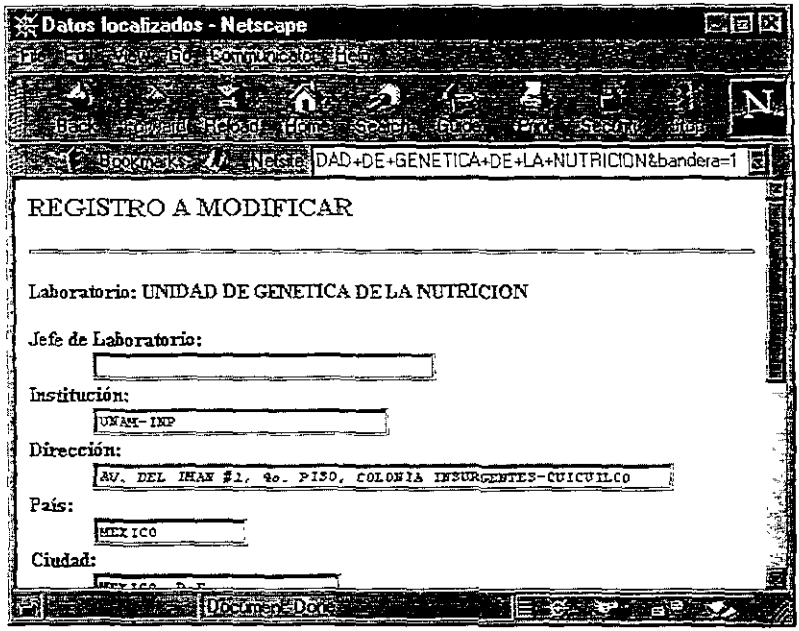


Imagen 5.31

Si se trata de un usuario válido, el programa presenta una forma con los datos a modificar (ver Imagen 5.31). Los datos son enviados al programa: **actualab.pl** este llama a ejecución a los programas **borralab**, que borra el registro de la tabla **DATOS_LAB** por la clave del especialista; y **grabalab**, que graba el registro con los nuevos datos en la tabla **DATOS_LAB**.

b) Agregar nuevas pruebas.

Es necesario validar la identidad del usuario; el programa **comparalab3.html** es una forma que el usuario debe llenar. Los datos son enviados al programa **validalab2.pl**; este llama a ejecución al programa **hacepasslab** que busca en la tabla **DATOS_LAB** y regresa la clave y el password de todos los laboratorios. En caso de existir un error, el programa **validalab2.pl** envía un mensaje.

Si se trata de un usuario válido, el programa presenta una lista de las pruebas que realiza el laboratorio. Si se desea agregar nuevas pruebas, se ejecuta el programa: **validalab3.pl**, este presenta la lista de pruebas que el usuario puede, para ello tan sólo es necesario que las seleccione.

El usuario elige las pruebas que desea agregar como nueva experiencia. Estos datos son enviados al programa **grabadoclab.pl** que recibe como parámetros un número 1 por cada prueba agregada; se envía como parámetro el nombre de la prueba y se ejecuta el programa **grabadoclab** que actualiza la tabla **PRU_LAB**.

Conclusiones

Actualmente REDLAEM agrupa a 35 especialistas de 11 países latinoamericanos, así como algunos que residen en Estados Unidos y en España.

En términos generales, se puede decir que ha cumplido con éxito sus propósitos de enlazar, ser el medio por el que se resuelven problemas difíciles de diagnóstico y manejo de enfermos, y constituir un foro para cursos y congresos.

En particular, de REDLAEM surgió la primera reunión de los miembros de la Red, que se llevó a cabo en Porto Alegre, Brasil, en agosto de 1996, el cual, a su vez, dio origen al Primer Congreso y a la fundación de la Sociedad Latinoamericana de Errores Innatos del Metabolismo y de Pesquisa Neonatal, en septiembre de 1997 en Cuba.

REDLAEM se convirtió, de punto de partida, en el órgano de comunicación de la nueva sociedad científica.

Aún más importante, un número creciente de niños con estas enfermedades han sido atendidos en forma eficaz gracias a esfuerzos colaborativos internacionales, que en algunos casos literalmente los libraron de la muerte.

Cabe señalar que un comité de la Sociedad Latinoamericana será, de ahora en adelante, el que decida sobre las solicitudes de admisión a REDLAEM y se encargue de su administración.

La tecnología actual permite desarrollar una nueva versión, lo cual se contempla para el futuro próximo.

Existen otros sitios en el Web sobre estos padecimientos, todos ellos en idioma inglés, cabe señalar que su desarrollo fué posterior a REDLAEM.

El conjunto de todos estos sitios permite el enlace de los especialistas en todo el mundo, y para aquellos que sólo pueden utilizar el idioma español, REDLAEM representa una ventaja para comunicarse entre sí.

En cuanto a las herramientas seleccionadas para la construcción del sitio Web de REDLAEM podemos decir que:

- Perl es el mejor lenguaje de programación de CGI's; ofrece una gran capacidad en el manejo de expresiones regulares (cadenas).Su sintaxis es sencilla, lo que permite una fácil lectura y comprensión del código de cada programa. Además de ofrecer un gran soporte de funciones, un muy buen manejo de archivos y programas externos, y la gran ventaja de ser un lenguaje interpretado, lo que permite realizar modificaciones casi de inmediato en el código y evitar la creación de programas binarios.

- La eficiencia de C en cuanto al manejo de argumentos enviados y recibidos a una base de datos es, hoy en día, la mejor.

- El servidor Web de NCSA maneja un programa demonio muy estable y de fácil configuración.

- SYBASE es un muy buen producto para el manejo de bases de datos relacionales. Protege la integridad de la información y cuenta con algunos procesos almacenados que manejan todas las peticiones de información sin reducir el rendimiento del sistema operativo.

El sistema ofrece un nivel aceptable de seguridad. Aunque el acceso al sitio es totalmente libre por cualquier usuario de Internet, la información contenida en la base de datos solo puede ser modificada únicamente por los usuarios registrados en el sistema.

La seguridad ofrecida por cualquier sistema en Web, puede abarcar aspectos como:

- El canal de comunicación (canal cifrado).
- La certificación de usuarios (certificados y firmas digitales).
- La validación de usuarios (login y password).
- Veracidad de información.
- Buen uso del sistema.

Los aspectos arriba mencionados varían de acuerdo a las necesidades de los propietarios del sitio, y a las políticas de uso que se establezcan de acuerdo a la importancia y valor de la información.

El sitio ha sido presentado ante la comunidad médica internacional, con muy buena aceptación en dos ocasiones: Primero en la Habana, Cuba, en Enero de 1998 y más recientemente, en Noviembre de 1998, en la Ciudad de México, en el Congreso General de Cómputo, conmemorativo de los cuarenta años de cómputo en México.

Bibliografía

Aplique SQL
James R groff, Paul N. Weinberg
McGraw-Hill, 1991.

Fundamentos de programación con HTML & CGI
Ed Tittle, Mark Gaither
Ediciones Anaya Multimedia, 1996.

JAVA Network Programing
Elliotte Rusty Harold
O'Reilly, 1997.

Redes globales de Información con Internet y TCP/IP, principios básicos, protocolos y arquitectura
Douglas E. Comer
Prentice Hall, 1996.

SYBASE and Client/Server Computing
Alex Berson, George Anderson
McGraw-Hill, 1991.

The Metabolic and Molecular Basis of Inherited Disease

Scriver, C.

McGraw-Hill, 1996.

Referencias Web:

http://www.ncsa.uiuc.edu

http://www.w3.org/Protocols

http://home.netscape.com

http://www.sun.com

http://www.sybase.com

http://laran.waisman.wisc.edu/fv/www/lib_meta.htm

http://www.unam.mx/redlaem

http://www.ncl.ac.uk/~nchwww/SSIEM/ssframe.html

http://www.franken.de/lists/metab-l/

http://www.mit.edu/perl/

http://www.nic.mx

http://www.isoc.org

http://www.apache.org

http://java.sun.com

http://www.ludd.luth.se/

http://www.merseyworl.com

Glosario

Alias. Nombre de un sistema informático que indica el nombre de otro en vez del objeto fundamental. La mayoría de los URL del Web son general o parcialmente alias (para resguardar el sistema de archivos subyacente en el servidor Web al que señalan).

ANSI. (American National Standards Institute, instituto de estándares nacionales americanos). Uno de los grupos de ajuste de estándares principales de la tecnología informática en Estados Unidos.

Atributo. En SGML, HTML y la mayoría de los lenguajes de programación orientados a objeto, un atributo es un componente nombrado (con nombre) de un objeto o termino, con un valor tipográfico específico, definiciones de elementos y con requisitos y estado por defecto.

Back-end. Jerga informática que designa un servicio que se ejecuta en una máquina en cualquier ligar de la red y que normalmente es conducido por una interfaz desde otra máquina en otro lugar de la red (front-end).

Backbone. Conexión de alta velocidad diseñada para interconectar varias redes. El objetivo de un backbone es conectar equipos terminales. Los nodos de red normal no se conectan al backbone principal sino que lo hacen a estos nodos terminales, accediendo de este modo a toda la red.

BIND (Berkeley Internet Name Domain). BIND es la implementación más popular del servicio de nombres de dominio Internet en uso. Escrita por Kevin Dunlap para 4.3 BSD UNIX, BIND proporciona capacidad de base de datos distribuida que permite a varios servidores DNS cooperar para separar los nombres de Internet en direcciones IP correctas residiendo en máquinas distintas.

Bourne shell. Las máquinas UNIX tienen normalmente una shell o interprete de comandos nativo. En muchas máquinas, este interprete en la Bourne shell, que toma su nombre de S.R. Bourne en 1975. La Bourne shell es parte de la configuración estándar para UNIX.

BSDI (Berkeley Software Distribution, Inc.). BSDI permanece como una de las implementaciones más importantes de UNIX disponibles hoy en día, pero ahora es distribuido por una empresa indirecta y no por la universidad de California en Berkeley.

C. lenguaje de programación desarrollado por algunos de los fundadores de UNIX, Bryan Kernighan y Dennis Ritchie. Es, posiblemente, el lenguaje con el que encontrará mayor cantidad de código fuente público disponible en Internet.

C++. Lenguaje de programación desarrollado por Bjarne y Stroustrup. C++ es el sucesor de C. es una implementación orientada a objeto de C.

Cabecera de petición. Es el preámbulo a una petición. La cabecera debe identificar a quien solicita la información y facilitar información de formato y verificación donde sea pertinente. Esto permite al servidor saber donde tiene que enviar la respuesta, si es que hay tal respuesta, y que formatos de presentación se deben utilizar a la hora de codificarlo.

Cabecera de respuesta. Es el preámbulo de una respuesta. La cabecera identifica a quien envía la petición y la aplicación a la que se debe suministrar la respuesta.

Cadena de petición. Los parámetros facilitados a una máquina de búsqueda basada en Web que normalmente utiliza para el paso de parámetros el método GET (puesto que las cadenas de búsqueda son casi siempre cortas, esto es bastante seguro).

Campo. En una base de datos, un componente mencionado de un registro y sus valores asociados. En un formulario HTML, una widget de entrada nombrado o un área de texto y sus valores asociados.

Caso Sensible. Significa que las mayúsculas y las minúsculas no son equivalentes (es decir, los nombres de archivo en UNIX son caso sensibles pues no es lo mismo "TEXT.TXT" que "text.txt").

CERN (Centre European pour la Researche Nucleare). El centro europeo de partículas físicas donde comenzó el WWW.

CGI (Common Gateway Interfece, interfaz de pasarela común). El paso de parámetro y la técnica de invocación utilizados para permitir a los clientes del Web enviar entradas a los servidores Web (y a programas específicos escritos para la especificación CGI).

Cliente. Se puede interpretar de dos formas. (a) como sinónimo de navegador Web (es decir, como cliente Web) o (b) como miembro front-end solicitante de una aplicación cliente/servidor (como WWW).

Cliente/servidor. Paradigma informático en donde el proceso se divide entre una aplicación front-end gráfica ejecutada sobre una máquina de escritorio de un usuario y entre un servidor back-end que realiza tareas de procesamiento intensivo de almacenamiento o de datos en respuesta a las peticiones de un cliente.

Codificación URL. Se trata de un método para pasar peticiones de información y especificaciones del URL a los servidores Web desde los navegadores. La codificación URL reemplaza los espacios por un signo (+) y sustituye los códigos hexadecimales por un intervalo de caracteres irreproducibles. Este método se utiliza para transmitir peticiones de un documento (utilizando el método GET), de un navegador a los servidores (y a los CGI´s).

Compilador. Un programa de software que lee el código fuente de un lenguaje de programación y que crea una versión de un ejecutable binario de ese código.

Contestador automático. Un programa de correo electrónico que envía una respuesta predeterminada a cualquiera que envíe un mensaje a una dirección de correo electrónico particular.

Correo electrónico. Servicio que permite a los usuarios intercambiar mensajes dentro de la red. La tecnología más importante de correo electrónico actualmente en uso dentro de Internet está basada en SMTP (Simple Mail Transfer Protocol).

Correo normal. Es la antítesis del correo electrónico. El correo normal necesita sobres y sellos y tarda muchísimo más en llegar a su destino.

DARPA. (Defense Advanced Research Projects Agency, agencia de proyectos de investigación avanzada de defensa. En un principio conocida como ARPA). La sección del DoD que funda la investigación avanzada, incluyendo el trabajo inicial que conduce al desarrollo y despliegue de Internet.

Datagrama. Es la unidad de información independiente más pequeña dentro de los protocolos IP y TCP/IP.

DBMS (Database Management System, sistema de manejo de base de datos). Sistema complejo de programas y utilidades utilizadas para definir, mantener y controlar el acceso a amplias colecciones de datos en línea.

Demonio. Término de UNIX para un programa que se ejecuta constantemente, escuchando peticiones para una conexión o atender un servicio en particular.

Dirección del puerto. En términos de TCP/IP, una dirección de puerto se refiere al identificador de toma que un programa o un servicio buscan para dirigirse a un tipo específico de comunicaciones. La mayor parte de los protocolos TCP/IP tienen direcciones de puertos muy conocidas asociadas a ellos (por ejemplo el de HTTP es el puerto 80), pero las configuraciones del sistema permiten que se utilicen otras direcciones de puertos.

DNS. (Domain Name Service, servicio de nombre de dominio). Servicio de Internet que distribuye nombres simbólicos a direcciones IP a través de peticiones entre la comunidad de los servidores DNS.

DoD. (Department of Defence, departamento de defensa). Las personas que propiciaron indirectamente Internet, entre otras cosas (ver DARPA).

FRONT END. El lado de una aplicación cliente/servidor de una interfaz de usuario. El front-end es lo que los usuarios ven y con lo que interactúan.

FTP (File Transfer Protocol, protocolo de transmisión de archivos). Servicio de protocolo de Internet que facilita la transmisión de archivos de red entre dos nodos de red de información para los cuales el usuario deben obtener derechos de acceso a dichos archivos.

gcc (GNU C Compiler, compilador C de GNU).

GET. Método HTTP para pasar datos del cliente al servidor. GET distribuye todos los parámetros de la línea del comando.

Hipertexto. Método de organizar texto, gráficos y otros tipos de datos para uso informático, el cual permite que los elementos de datos individuales se señalen unos a otros. Método no lineal de organizar información, especialmente el texto.

HTTP (HyperText Transfer Protocol, protocolo de transmisión de hipertexto). Protocolo de comunicaciones, basado en TCP/IP. Define cómo se comunican los clientes y los servidores en el Web y cómo intercambian información

httpd (demonio HTTP). Programa que se ejecuta sobre un servidor Web que escucha (también se le denomina oyente) y está preparando a responder las preguntas de documentos Web o de servicios basados en CGI.

Independiente de la aplicación Se dice que un formato es independiente de la aplicación cuando trabaja en varios en entornos y no depende de una aplicación específica para comprender o utilizar sus contenidos.

Independiente de la plataforma. Indica que un programa o mecanismo funcionará en cualquier ordenador independiente del modelo, tipo, procesador o sistema operativo utilizado.

Interfaz. Las subrutinas particulares, los mecanismos de paso de parámetros y los datos que definen el modo en el que dos sistemas (que pueden estar en el mismo ordenador o en ordenadores diferentes). Se comunican entre sí o con el usuario.

Internet. Nombre de una comunidad informática en red basada en TCP/IP conocida mundialmente como la red de redes, la cual tiene millones de usuarios y que es capaz de unir investigación, industria, negocios, etc.

Intérprete. Programa de software que lee el código fuente de un lenguaje de programación cada vez que se ejecuta, para interpretar las instrucciones que contiene. La alternativa es utilizar un compilador, el cual traduce el código fuente en forma binaria con el objeto de ejecutarlo posteriormente.

IP. (Internet Protocol, protocolo Internet). Protocolo de red primordial para el grupo de protocolo TCP/IP. IP es probablemente el protocolo de red más utilizado en el mundo.

Java. Entorno y lenguaje de programación orientado a objetos de Sun Microsystems. Junto con C y C++, Java se compila en un objeto binario de arquitectura neutra y después se interpreta como Perl o Tcl para una arquitectura informática específica.

Lan (Local Area Network, red de área loca). Una red unida por cables físicos o conexiones de corto recorrido con una extensión que es generalmente inferior a una milla.

Librería. *Conjunto de programas o módulos de código que los programadores* pueden unir a su propio código para facilitar una funcionalidad predefinida y estándar utilizando un enlazador.

METHOD. Enfoque HTML para pasar datos de entrada de un navegador a un *servidor (y posiblemente a un CGI).*

MIME (Multipurpose Internet Mail Extensions, extensiones de correo Internet multipropósito). Extensiones del formato de mensaje de correo RFC822 para permitir datos más complejos y tipos de archivos de texto sencillo. Hoy en día, los tipos MIME *incluyen sonido, vídeo, gráficos, PostScript y HTML entre otros.*

Navegador. Aplicación de Internet que permite a los usuarios acceder a servidores WWW y navegador por la red utilizando http como protocolo de transporte e interpretando los documentos html.

NCSA (National Center for Supercomputing Applications, centro nacional para aplicaciones de supercómputo).

NFS (Network File System, sistema de ficheros en red).
Sistema de ficheros distribuidos creado por Sun Microsystems y que hoy en día se utiliza mucho en entornos de red TCP/IP. NFS permite a los usuarios acceder a sistemas de ficheros lejanos como si se tratara de extensiones de sus discos duros.

NIC (Network Interface Card, tarjeta de interfaz de la red).
El hardware que permite a su ordenador comunicarse con una red y viceversa independientemente del protocolo utilizado.

Parámetro. Valor que se lleva dentro y fuera de un programa, subrutina, o a través de una interfaz, cuando los componentes del código se comunican unos con otros.

Pasarela. Programa o servicio que sabe cómo pasar entradas de un tipo de sistema a otro tipo de sistema. Esta palabra está relacionada con CGI ya que maneja las entradas de los clientes del Web como extensiones del servidor Web y facilita salidas a esos mismos clientes.

Perl. (Practical Extraction and Report Language). Lenguaje de programación interpretado desarrollado por Larry Wall semejante al awk. Perl ofrece un magnífico manejo de cadenas y la capacidad de equiparar modelos y además es el lenguaje favorito entre los programadores CGI.

Petición. Mensaje en red emitido por un cliente y dirigido a un servidor con el objeto de solicitar un servicio o información sobre un asunto determinado.

POST. Método HTTP por el que se pasa al servidor una lista de valores y nombres asociados para un manejo y análisis más detallado. POST permite de forma arbitraria grupos complejos y extensos de parámetros al contrario de lo que sucede con GET, el cual se limita a un máximo de 255 caracteres para la mayoría de versiones UNIX existentes en la actualidad.

PostScript. Lenguaje de descripción de página definido por Adobe Systems. Los archivos PostScript normalmente tienen la extensión ".ps" dentro de UNIX y son una manera común de intercambiar ficheros de impresión formateados satisfactoriamente.

PPP (Point-to-Point Protocol, protocolo punto a punto).
Protocolo TCP/IP asíncrono más eficaz y nuevo designado específicamente por usuarios que desean sacar el máximo de las conexiones con Internet.

Puente (bridge). Pieza del equipo de trabajo del Internet que opera en el nivel 2 del modelo ISO y que dirige los paquetes de un segmento de red a otro sin comprobar la dirección.

Respuesta. Mensaje en red de un servidor a un cliente que contiene una contestación a una petición de servicio.

Router. Programa, sistema hardware o mecanismo interno de la red que lee las direcciones de los paquetes nuevos y los guía a su destino o a otros indicadores que les conduzcan cerca de su destino.

Script. Sinónimo de programa. Los programas normalmente se refieren a su trabajo como script cuando está escrito en un lenguaje interpretado porque, al igual que sucede con script, se lee completamente cuando se ejecuta.

Servidor. Una máquina en red que sirve peticiones de clientes.

SGML (Standard Generalized Markup Language, lenguaje estándar de marcas generalizado). Metalenguaje apropiado para describir todo tipo de lenguajes de marcas, incluyendo HTML.

Shell. Lenguaje de interfaz de usuario UNIX. Una shell facilita el entorno de comando básico para el sistema operativo de UNIX.

Sin estado. Un protocolo sin estado no necesita información de lo que ha ocurrido anteriormente o de lo que se espera que suceda en un futuro, con relación a las comunicaciones entre el emisor y el receptor. Es el tipo de comunicación en red más eficaz y sencillo de implementar.

Síncrono. Método de comunicación en el que todas las partes intervienen en la misma, interactúan unas con otras al mismo tiempo, la comunicación está controlada por un reloj.

SQL (Structured Query Language, lenguaje de peticiones estructurado).
Lenguaje de peticiones de base de datos desarrollado por IBM que tiene un uso muy extendido en todas las bases de datos mundiales (se ha definido una versión estándar).

Stdin. Mecanismo de entrada de UNIX. Stdin es la fuente de entrada por defecto para los programas y facilidades, incluidos los servidores Web y los clientes.

Stdout. Mecanismo de salida estándar de los sistemas basados en UNIX. Stdout es la fuente de salida por defecto para los programas y facilidades, incluidos los servidores Web y los clientes.

T1. Enlace de transmisión digital con una capacidad de 1544 Mb/sg. T1 (también se puede escribir T-1), es el estándar para la transmisión digital en Estados Unidos, Canadá, Hong Kong y Japón.

TCP/IP (Transmision Control Protocol/Internet Protocol). Grupo de protocolos básicos sobre los que se ejecuta Internet.

Telnet. Servicio y protocolo TCP/IP que permite a un usuario en un ordenador emular a un terminal adjunto a otro ordenador.

Text/html. Tipo de contenido MIME para documentos HTML (utilizado normalmente por los programas CGI y HTTP para la salida de información y por los navegadores Web como entrada de información).

Text/plain. Tipo de contenido MIME para texto sencillo (se representará tal y como es por la mayoría de los navegadores).

Tiempo de respuesta. Periodo de tiempo que pasa entre la transmisión de una petición de servicio y la llegada de la respuesta correspondiente.

UNIX. Sistema Operativo desarrollado por Brian Kernihan y Dennis Ritchie como entretenimiento en el Bell Labs a finales de los años sesenta y que todavía está vigente hoy.

URI (Uniform Resource Identifier, identificador de una fuente uniforme). Uno de los tipos de objeto que identifican las fuentes disponibles dentro del Web. Tanto los URL como los URN son instancias de un URI determinado.

URL (Uniform Resource Locator, localizador de fuente uniforme). Idea primaria de nombramiento utilizada para identificar las fuentes Web. Los URL definen los protocolos que se deben utilizar, el nombre del dominio del servidor Web donde reside la fuente, la dirección del puerto usada en la comunicación y la ruta del directorio para acceder a un documento Web o fuente determinada.

URN (Uniform Resource Name, nombre de fuente uniforme).
Nombre permanente e inalterable para una fuente Web (muy raras veces se utilizan en el entorno Web actual).

Variables de entorno. Como muchos otros programas UNIX, los CGI obtienen y almacenan sus entradas en vez de leerlas cada vez que son necesarias. Esta información almacenada en forma de variables de entorno, se transmiten al programa por el servidor HTTP. Por lo tanto, una variable de entorno en un valor transmitido a un programa o script por las variables en tiempo de ejecución sobre el sistema en el que se están ejecutando.

W3 (World Wide Web).

W3C (World Wide Web Consortium). Es el consorcio que engloba CERN, MIT y otras organizaciones que actualmente tienen custodia sobre HTTP, HTML y otros estándares y software relacionados con el Web.