UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES

ARAGÓN

"LA INTERACTIVIDAD CON EL WEB A TRAVES DE LOS SCRIPT – CGI (INTERFAZ DE PASARELA COMÚN) APLICADA A LOS PROCESOS DE EVALUACION ACADEMICA."

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A :

MARIA FERNANDA APARICIO LOPEZ

ASESORA:

ING. JUAN GASTALDI

MEXICO **2001.**

PAGINACIÓN DISCONTINUA

AGRADECIMIENTOS

No tengo palabras para agradecer a todos aquellas personas que se involucraron en este proyecto a través de sus consejos, criticas e ideas que proporcionaron para enriquecer este trabajo.
Estoy segura de que siempre recordare los nombres de los profesores Marcelo y Juan.
Gracias.

Un agradecimiento especial a la profesora Almánzar por enseñarme el valor de un compromiso y de la profesionalidad.

A mis padres:
"*No importa la adversidad ni el obstáculo siempre estaremos juntos*".

La misión del conocimiento no es iluminar a un alma que es oscura de por sí, ni hacer ver a un ciego. Su misión no es descubrir los ojos del hombre, sino guiarlo gobernarlo y dirigir sus pasos a condición de que tenga piernas y pies para caminar. (*Montaigne*)

INDICE

INTRODUCCIÓN

Este trabajo de tesis muestra como a través de la diversidad de aplicaciones de los script CGI, que sean popularizado gracias al florecimiento y expansión de la tecnología de Internet, es posible beneficiar las distintas actividades que realiza el hombre tanto en su trabajo como en el hogar; particularmente, este trabajo se dirige a la docencia y alumnado de la Universidad Nacional en el plantel Aragón, donde se imparte la asignatura de Graficación por Computadora.

En la enseñanza de esta asignatura ha nacido la necesidad por parte del profesor de emplear nuevos recursos y herramientas para llevar acabo de forma eficiente la evaluación de los exámenes de los alumnos. Por lo tanto, basándose en esta necesidad se propone el uso de una herramienta que facilite este labor, donde se incluya la elaboración de los reactivos (cuestionarios) para el diseño de los exámenes y la evaluación de los mismos.

Esta herramienta se desarrolla y construye empleando el recurso de los script CGI, que será accesible gracias al Internet y que bajo la presentación de una pagina Web podrá estar en uso en cualquier parte donde el profesor o los alumnos tenga acceso a Internet.

La herramienta que se presenta se ha nombrado *Sistema Auto - evaluación de Graficación por Computadora*, este sistema incluye una pagina principal de menú de opciones a través del cual se elige la actividad a realizar. Las opciones que se presentan son: Actualización de un catalogo de preguntas, Modificación del catalogo de preguntas, presentación de un catalogo y presentación de un examen; con estas opciones es posible generar un catalogo de preguntas para diseñar nuevos exámenes que podrán ser aplicados a los alumnos y entregarles resultados de forma inmediata.

Este trabajo se compone de cinco capítulos, que llevan al lector de lo general a lo particular concluyendo con la presentación del Sistema Auto- Evaluación.

En el primer capitulo denominado Fundamentos Teóricos de Internet, se explican los conceptos y términos generales que se manejan alrededor de la tecnología de Internet.

En el segundo capítulo nombrado El servicio gráfico de Internet: el World Wide Web, se describe el funcionamiento y los elementos que interviene en la construcción de una pagina Web.

En el tercer capítulo que lleva por nombre El Web interactivo a través de los script CGI, se explica que es un script CGI y los aspectos que se relacionan con su funcionamiento, y se identifican los principales usos y aplicaciones que se hacen de los CGI.

En el cuarto capítulo denominado Fundamentos del Lenguaje Perl, se describen de forma general los elementos básicos para la programación del lenguaje Perl.

El último capitulo de este trabajo concluye con el desarrollo y presentación del Sistema de Auto – Evaluación, en esta sección se definen los elementos estructurales del sistema, de cómo se desarrollaron y de los procesos de prueba que se realizaron para la puesta en marcha del sistema.

La creación de esta herramienta es una muestra de cómo la ingeniería en computación permite generar soluciones optimizadas y eficientes de las problemáticas que se presentan en las actividades de las diversas áreas o disciplinas del conocimiento, a través del diseño e implementación de sistemas de aplicación, que en esta ocasión benefician el área de la pedagogía.
Donde la tarea principal del ingeniero será emplear su creatividad, su inventiva y su análisis haciendo uso de todos los recursos del alrededor computacional que estén a su alcance para presentar de forma concreta una solución.
De ahí su vinculo tan importante que lo relaciona con todas las áreas en que se desenvuelve el hombre, al resolver problemas que sean factibles de solucionar por medios sistemáticos y por ende computacionales.

CAPITULO I

FUNDAMENTOS TEORICOS DE INTERNET

En este capitulo se explican los términos y conceptos comúnmente usados en la "jerga" del Internet, permitiendo que el lector conozca los antecedentes y los elementos que acompañan a esta tecnología sobre la cual se diseñara la aplicación de los scritp CGI.

1. Antecedentes históricos de Internet

Actualmente, el concepto de Internet en términos generales, comprende a las miles de computadoras interconectadas entre sí distribuidas alrededor del mundo, cada una con una identificación o dirección propia y única, que le permite a la gente que las usa comunicarse y compartir información.

Sin embargo, para que esta tecnología estuviera al alcance de cualquier persona fue necesario recorrer un largo camino de avances tecnológicos e investigaciones que permitieran hacer realidad esta nueva forma de comunicación.

Dando un vistazo hacia atrás, fue gracias a los sistemas telegráficos que nacen las bases que dieron la pauta para la creación de sistemas capaces de enlazar computadoras, es decir para el uso de lo que actualmente se conoce como Redes.
Los sistemas telegráficos de entonces consistían en términos modernos, en enlaces punto a punto, en el cual se tiene una línea que será ocupa por dos usuarios que realizan una conexión, dicha línea queda ocupada hasta que estos finalicen su conexión. Con una industria creciente fue necesario crear centros conmutadores que actuaran como estaciones de transmisión, donde las cintas perforadas fueron el medio a través del cual las personas ruteaban (encaminaban) la información que se transmitía de un enlace a otro.[1]

En el año de 1969 ocurrieron grandes acontecimientos, no solamente para la NASA al poner los primeros astronautas sobre la luna sino también, para el Departamento de Proyectos de Investigación Avanzada de Defensa (ARPA) que junto con Bolt, Baranek y Newman se dan a la tarea de desarrollar una Red basada en las ideas de Paul Baran, quien planteó la necesidad de construir una Red de Computadora que pudiera sobrevivir a un ataque nuclear. El proyecto inicial enlazaba computadoras de la Universidad de California

[1] Breedlove, Bod, et. al. Web Programming unleashed. Ediciones Sams Net, México, 1999. p. 4.

en Los Angeles (UCLA) , el Instituto de Investigaciones Stanford (SRI) en Menlo Park, California y la Universidad de Utah en Salt Lake City, Nevada , esto significo el nacimiento de lo que se llamo con el nombre de ARPAnet y que constituyo la primera Red Experimental lo cual podría llamársele el padre de lo hoy conocemos como INTERNET.
El ARPAnet proveía originalmente tres tipos de servicios: login remoto (telnet), transferencia de archivos e impresión remota. Así para 1972, ARPAnet estaba constituida por 37 sitos de enlace (sites) y muy pronto los e-mail formaron parte de los servicios de la nueva Red. En octubre de 1972 ARPAnet fue presentada públicamente en la Conferencia Internacional sobre Comunicación por Computadora en Washington, D.C. En los siguientes años la tarea principal fue establecer las normas o el protocolo estándar para ARPAnet. [2]

Históricamente, los primeros indicios sobre redes tratan alrededor de los años 70 ´s cuando la computadora ya se había convertido en una herramienta indispensable en las labores del trabajo. A pesar de que se contaba con computadoras potentes estas fueron rebasadas por las crecientes necesidades de los usuarios, por tanto, sé penso en crear un sistema con el cual se lograran comunicar desde una misma terminal a varias computadoras, esto dio como resultado la creación de las Redes LAN o Redes de Área Local, en donde las terminales se conectaron a lo que se conoce como Servidores de Red. El Servidor es una computadora cuya misión consiste en establecer la conexión entre las terminales y las grandes computadoras.[3]

La tecnología LAN (Redes de Área Local) por si sola no fue suficiente, puesto que la longitud de los cables y el número de unidades conectadas se limitaron por las características técnicas, por ello fue inevitable realizar un nuevo avance que permitiera ampliar las posibilidades de dicha red. En los años 80´s, el desarrollo de las redes locales se vio influenciado por la introducción al mercado de las Computadoras Personales ocasionando un aumento en el número de unidades activas en las redes locales, con esto creció la necesidad de redes de mayor capacidad, como consecuencia las Computadoras Personales ganaban terreno en el área de aplicación y el uso de las redes se diversifico.

La conexión de las LAN´s marca el verdadero nacimiento de la interconectividad de las redes. Para entonces, una LAN individual se podía conectar con otra LAN para formar una red más grande, bajo este principio, al conectar las redes LAN a una red más grande o red global se fue construyendo la gran red de Internet.

[2] Breedlove, Bod, et. al. Web Programming unleashed. Ediciones Sams Net, México, 1999. p. 5.
[3] Todo sobre Internet. Ediciones Marcombo – DATA BECKER. México, 1587. pp. 22-23.

A) Conceptos básicos sobre Redes.

La estructura básica sobre la que descansa Internet o también llamada la red de redes se construye a partir de las ya tan mencionadas redes, por ello es necesario marcar algunos conceptos sobre éstas que servirán para entender ideas de capítulos posteriores.

Para comenzar es necesario precisar que una *Red de computadoras*, es un conjunto de computadoras conectadas entre sí, a través de medios de comunicación (líneas telefónicas, cable coaxial, fibra óptica y microondas), cuyos objetivos son tres:

- Compartir Información.
- Comunicar Usuarios.
- Tener Flexibilidad en el Manejo de Información.

A continuación se puntualizaran algunos conceptos sobre las Redes LAN.

Una Red de Area Local (LAN) esta constituida por: Estaciones de Trabajo (PC 's), el Servidor de la Red, los Cables de Comunicación, las Tarjetas de Interfaces y un Sistema Operativo. En donde las estaciones de trabajo se interconectan permitiendo una comunicación entre ellas para compartir recursos de forma integral y coordinada.

Existen tres tipos básicos de topologías que se pueden adoptar como son:

- Red Tipo Anillo.
- Red Tipo Bus o Lineal.
- Red Tipo Arbol o Estrella.

Una *topología de Red* es la forma fisica de conectar las estaciones de trabajo, mientras que las estaciones de trabajo se comunican a la Red a través de un método de acceso específico que depende del tipo de red. Los Métodos de Acceso son técnicas utilizadas por las estaciones de trabajo, para compartir el canal de comunicación (medio de transmisión).

Actualmente, los tipos de redes más representantes dentro del mercado son: Ethernet orientado al Bus, que hoy en día es casi un estándar generalizado, Token- Ring marca registrada de IBM, que cuenta con productos importantes para redes de topología de anillo y ARCNET registrada por Standard Microsystems Corp, que describe una red tipo estrella.

La información que se transmite dentro de una red se envía y se recibe por medio de "*paquetes*", estos paquetes deben incluir las direcciones del destinatario y la de quien remite. Las formas de intercambio de información varían según el tipo de red.

Un nuevo reto aparece, cuyo objetivo es lograr una unión entre tecnologías de red diferentes o lejanas, es entonces cuando se adelanta un paso hacia la *interconectividad*. Gracias a esto, nuevos elementos aparecen en escena para franquear las barreras de Conectividad, que son: Repetidores, Puentes, Router y Gateways.

Los repetidores son dispositivos electrónicos del tamaño de una vídeo que regeneran o repiten los paquetes de datos que llegan a ellos hacia otro segmento de la red, de esta forma se unen dos segmentos de red del mismo tipo, así, su función principal es incrementar la extensión física de la red.

Los Puentes (Bridges) constituyen una unidad de enlace más sofisticado que enlazan dos o más LAN's para formar inter- redes Estos equipos de comunicación operan entre redes de distintas topologías (ARCNET con una Ethernet por ejemplo), así como, en redes de la misma tecnología. Básicamente se encargan de regular el tráfico en la red, gracias a un filtrado, los paquetes de datos de acuerdo a la información contenido en el campo dirección del paquete se dirigirán, ya sea, a una estación de la red local permitiendo que continúe con su trayectoria, o enviándolo hacia la otra red optimizando el tráfico local. Los puentes, funcionan independientemente del protocolo de transporte usado por la red: (TCP/IP o IPX)

Los ruteadores (Routers) ayudan a resolver el problema de conectar dos o más redes. Propiamente un ruteador es una computadora dedicada de propósito especial que enruta los datagramas IP[4]. Cada ruteador direcciona un datagrama hacia otro ruteador hasta que pueda alcanzar su destino final.

Los Gateways, se utilizan para conectar computadoras de diferente arquitectura, ya que funcionan como convertidores o traductores de protocolos totalmente distintos como X.25 y TCP/IP. Dependiendo del nivel de incompatibilidad los gateways se ubican en los niveles 4 al 7 del Modelo OSI. A través de un gateway es posible comunicar una Red Local con una Minicomputadora o con un Mainframe.

A continuación se muestran figuras de los dispositivos de conectividad mencionados

[4] Datagrama IP.- paquetes de datos enviados a través de Internet. Cada datagrama IP contiene la dirección de la computadora que lo envía, la de la computadora destino y los datos que se están enviando.

Figuras de diferentes dispositivos de conectividad

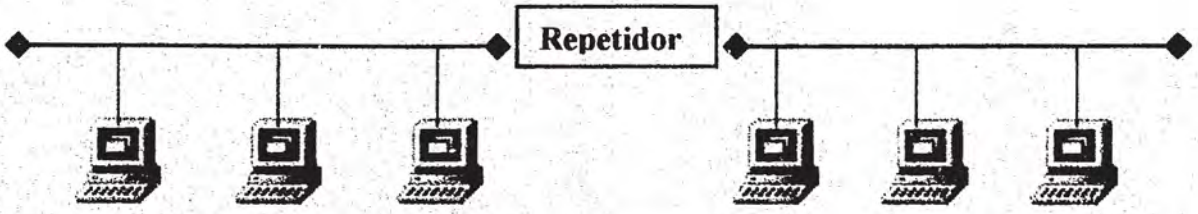


Figura 1.1. - Repetidor

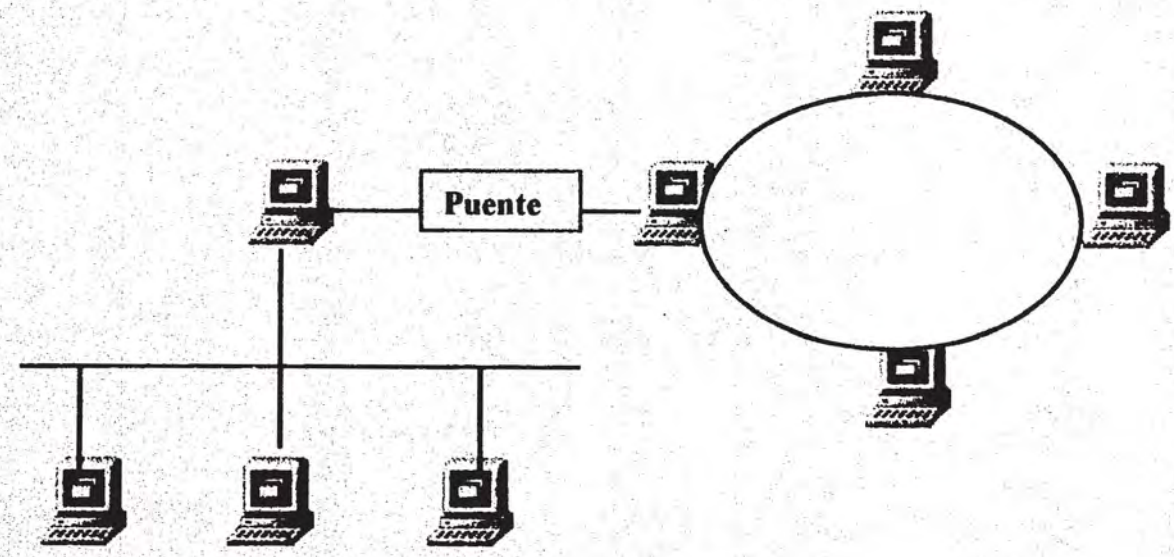


Figura 1.2. – Puente

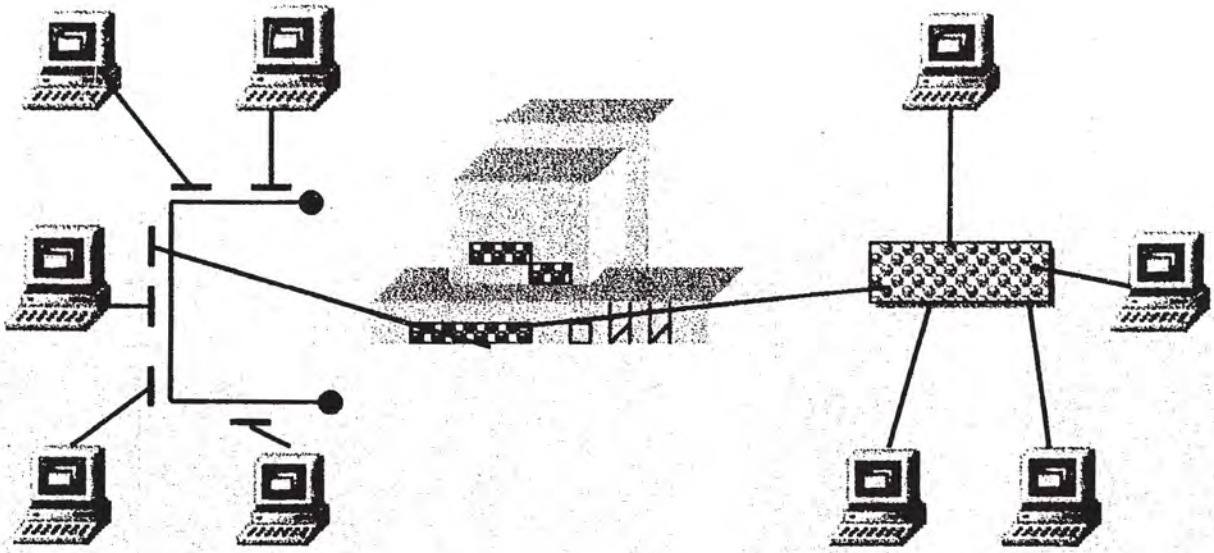


Figura 1.3. – Routers

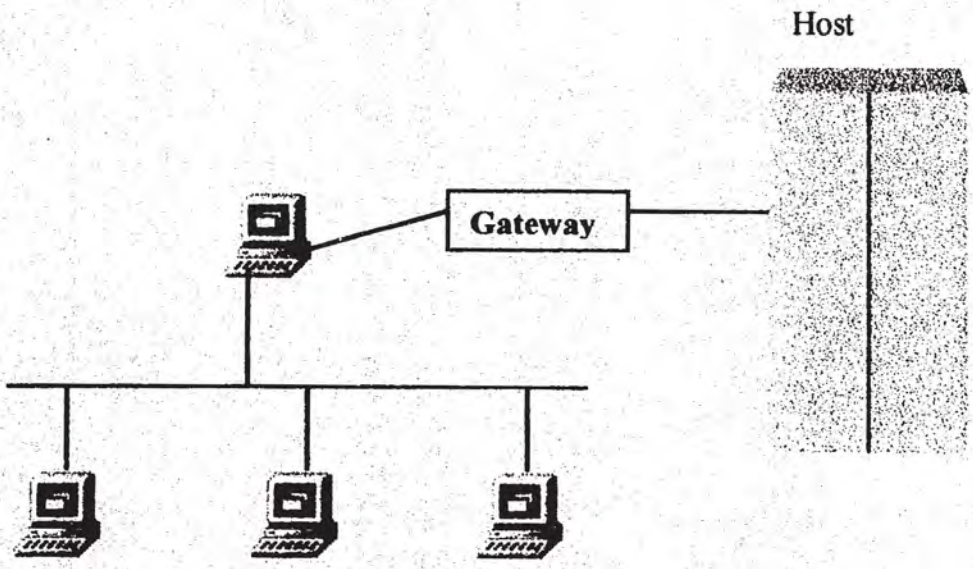


Figura 1.4. - Gateways

Con el aumento de la popularidad de los sistemas basados en redes surgió la necesidad de crear un conjunto de normas para el diseño y construcción de equipos que garantizara la compatibilidad entre computadoras de distintas marcas y modelos. La Organización Internacional de Normas (ISO), fue uno de los organismos que se encargó de resolver este problema estableciendo un Modelo de Interconexión de Sistemas Abiertos (heterogéneos), conocido también como " Modelo OSI ".

El modelo OSI tiene como finalidad lograr una comunicación eficiente de datos entre las computadoras, este modelo consta de 7 niveles que son:

1. Capa Física
2. Capa de Enlace de Datos
3. Capa de la Red
4. Capa de Transporte
5. Capa de Sesión
6. Capa de Presentación
7. Capa de Aplicación.

En estos 7 niveles jerárquicos se define y rige la estructura estándar que deben cumplir las computadoras para intercomunicarse.

2. Uso de los protocolos en Internet

Los protocolos juegan un papel importante dentro de la comunicación y una forma de observarlo es cuando se realizan actividades cotidianas, por ejemplo, cuando se tiene una conversación telefónica " se oye y se habla ", ambos interlocutores hablan y se escuchan, si no se entiende lo que se escucha se interrumpe y se pide que se repita.
En todo esto hay un conjunto implícito de normas que reglamentan la comunicación, que en la terminología de redes éste conjunto de normas reciben el nombre de *Protocolos*.
Cualquier proceso de comunicación independientemente del sistema que se trate y del nivel de comunicación, presupone la existencia de un Protocolo.

Cuando se establece una comunicación entre dos computadoras es necesario contemplar aspectos tales como: concordancia con el tipo de transmisión de datos, los niveles de tensión, que corresponden a los estados binarios de ambas computadoras, así como, el número de bits de información por segundo. Estos primeros aspectos deberán definirse de acuerdo a un protocolo de la capa más inferior.

En una capa de protocolo del siguiente nivel superior, para la representación de datos alfanuméricos que se agrupan en ocho bits por un byte, se proporcionaran las normas sobre como separar los bytes entre sí, además de incluir bits de control.
En el nivel inmediatamente superior, se asignan los contenidos a las series de caracteres que se transmitieron con el protocolo de la capa anterior. En este nivel se diferenciara entre las direcciones para el routing y los datos propiamente.[5]

Cuando se transmiten datos en una red compleja y heterogénea, los paquetes de datos se transportan utilizando diferentes protocolos de niveles inferiores, mientras que la información en sí se oculta en los niveles más altos del protocolo. De esta forma la información se protege sin ser afectada por ninguna operación que se realice en un nivel inferior del protocolo.

A raíz de la creación de redes tan heterogéneas, la invención de un Protocolo unitario para redes de diferente tipo se convirtió en la tarea principal para el desarrollo de la tecnología de Internet. La solución a dicho problema comprendía la creación de protocolos de transmisión de datos muy robustos y eficientes. Un avance de importancia fundamental fue la creación de un protocolo para Internet, tarea que estuvo a cargo del Departamento de Defensa, cuyo resultado fueron dos protocolos, TCP (Transmission Control Protocol) e IP (Internet Protocol), o también conocidos como TCP/IP.

TCP/IP actúan en el tercer y cuarto nivel del Modelo OSI, cada uno se encarga de llevar acabo un trabajo particular. Por un lado esta el protocolo IP, cuya función es definir el formato básico de un paquete de datos, además de ser el responsable del transporte de los datos dentro de toda la red. Un paquete IP o datagrama IP está provisto de una dirección de emisor y otra dirección de receptor, ambas direcciones se especifican en forma numérica considerando que el tamaño máximo de un paquete IP es de 1500 caracteres, incluyendo los datos que se envían. Cuando la unidad de información rebasa los limites establecidos es necesario dividirla previamente en unidades menores para que esta se pueda transmitir. Al llegar a su destino estas unidades menores deberán volver a agruparse para formar la unidad original de información. Este mecanismo funciona gracias a que los paquetes de datos incluyen información adicional, como puede ser un nombre lógico, común a todos los paquetes de datos de una misma unidad de información, y de un numero en orden consecutivo (serial), esta tarea especifica la realiza el protocolo TCP, el cual reclamará automáticamente un paquete dañado o perdido.

Por lo tanto, mientras el protocolo TCP se encarga de corregir faltas, de repartir los datos en paquetes y de reorganizarlos cuando llegan al destinatario, el protocolo IP, es responsable del transporte de datos (Routing) dentro de toda la red. Básicamente, el Routing describe la ruta inequívoca de un paquete de datos, es decir, a través de que estaciones debe ir el paquete hasta llegar al receptor.

[5] Todo sobre Internet. Ediciones Marcombo – DATA BECKER, México, 1587. pp. 31-32.

Cuando se establece una línea de enlace punto a punto entre dos redes a través de líneas telefónicas se recurre al uso de los protocolos PPP y/o SLIP, ya que las líneas que proveen las compañías telefónicas están desprovistas de cualquier protocolo de la capa física, ambos protocolos se emplean principalmente como métodos para transferir datos entre dos ruteadores.[6]

SLIP (Protocolo Internet de Línea en serie), fue el primer protocolo para ampliar Internet a través de líneas de acceso telefónico estándar. Llevar TCP/IP a través de las líneas de teléfono ha sido práctico únicamente en los últimos años gracias a la introducción del módem de alta velocidad. Poco tiempo después, apareció en escena PPP (Protocolo Punto a Punto), convirtiéndose en una opción más fiable que SLIP y en ocasiones ligeramente más rápido. Una de las principales diferencias entre SLIP y PPP, es que SLIP solamente puede transportar TCP/IP a través de una línea en serie, mientras que PPP puede transportar muchos protocolos además de TCP/IP en una variedad de conexiones.

Por otro lado, los programas de aplicaciones de Internet ocupan protocolos que se asignan a un nivel superior. A continuación se presenta una tabla donde se observan una lista de protocolos de alto nivel que es posible encontrar en Internet.

Tabla 1. Lista de protocolos de alto nivel en Internet

Protocolo	*Descripción*
AUTH	Autentificación
ECHO	Revisa al servidor para ver si está ejecutando
FINGER	Le permite recuperar información sobre un usuario
FTP	Protocolo para Transferencia de Archivos
NNTP	Protocolo para Transferencia de Noticias en Red - Grupos de noticias Usenet
POP	Protocolo de Oficina Postal – Correo entrante
SMTP	Protocolo Simple para Transferencia de Correo
TIME	Servidor de hora
TELNET	Le permite conectarse a una computadora anfitriona (host) y usarla como si fuera una terminal conectada directamente
HTTP	Protocolo de Transporte de Hipertexto*

[6] Breedlove, Bod, et. al. Web Programming unleashed. Ediciones Sams Net, México, 1999. p. 27.

* Sistema para almacenar páginas en el Web, donde estas páginas contiene información textual que incluyen referencias a otras páginas de información.

3. El sistema de direccionamiento y dominios en Internet

"Las direcciones en Internet son como las direcciones de una entidad (persona o empresa) donde los índices de su ubicación correcta son: país, estado o provincia, calle, número y número interior. Esta manera de direccionamiento la convierten en una entidad única. De la misma forma, se trata de definir a cada una de las computadoras y usuarios de la red como un ente único y localizable." [7]

La existencia de una dirección para cada computadora conectada dentro de Internet permite establecer una comunicación entre los miembros de la gran red. Estas direcciones se conocen como direcciones IP. Una dirección IP se compone de cuatro cifras comprendidas entre 0 y 255, separadas entre sí por un punto. Por ejemplo observe las siguientes direcciones IP:

170.12.0.165 o 200.12.165.19

La representación de la dirección decimal 170.12.0.165 en forma binaria se vería así:

170	12	0	165	Decimal
10101010	00001100	00000000	10100101	Binario

Como se puede observar una representación binaria de una dirección IP se compone de 32 bits divididos en cuatro octetos.

Un numero IP, a pesar de estar estructurado por cuatro cifras se puede dividir en dos partes principales que indican, por un lado, la dirección de la red local a la que pertenece la computadora y por otro lado, la dirección con la que se ubica la computadora dentro de la red. Para la asignación de direcciones es necesario considerar las tres clases generales de direcciones que son: la clase A, la clase B y la clase C.

Las direcciones de la clase A se asignan para las grandes redes, donde la cantidad de direcciones máximas que se asignan para las redes de esta clase son 16777216 direcciones en total. El número de direcciones posibles en la clase B es de 65536. Las direcciones de la clase C emplean un total de 256 nodos para direccionar.

[7] Ferreyra C., Gonzalo. Internet paso a paso hacia la autopista de la información. Ediciones Computec, México, 1222 p. 81.

En la siguiente tabla se muestran las categorías de clases con sus respectivos rangos de direcciones de redes que son posibles asignar.

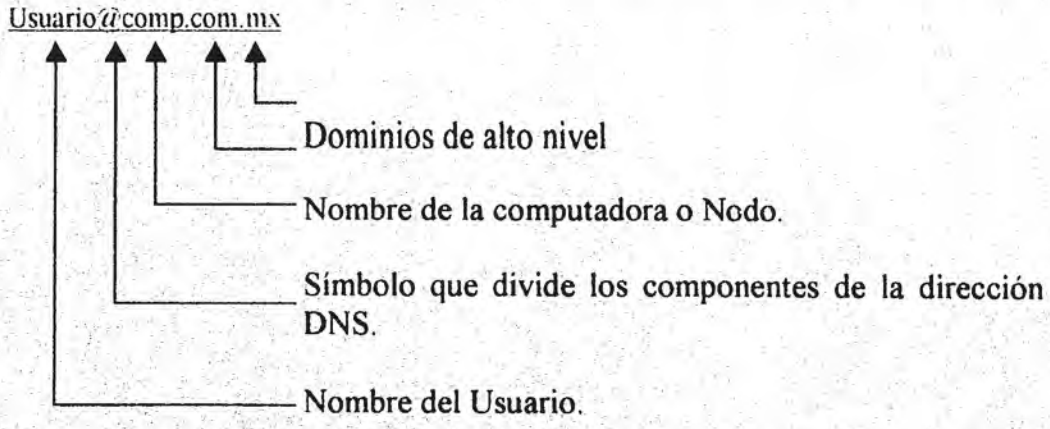
Tabla 2. Rangos de direcciones de red

TIPO DE CLASE	*DIRECCIONES DE REDES*
Clase A	1 – 127
Clase B	128 – 191
Clase C	192 – 223
Clase D	224 – 254

A través de las direcciones IP, es posible ubicar físicamente sin equivoco donde se encuentra una computadora dentro de la gran red, sin embargo, resulta incomodo para los usuarios tener que memorizar esta serie de números, por esta razón se pensó en una convención de nombres que permitiera una forma fácil de comunicación entre los usuarios. Gracias a esta convención de nombres denominada Sistema de Nombres de Dominios (DNS) resulta más cómodo recordar nombres que recordar números. Actualmente, el Centro de Información de Red Internet o también llamado InterNIC, a través de la Autoridad para la Asignación de Números en Internet se encargan de determinar los estándares para la asignación de nombres de dominios a las redes.

De forma similar a las direcciones IP, los nombres de dominios se conforman como: Usuario@dominios, donde los dominios se separan entres sí por un punto. Típicamente estos dominios son el nombre de la computadora, el nombre de la institución y un dominio de alto nivel.

La estructura completa de una dirección de Internet en el sistema de dominios se vería así:



Los dominios designados como los de más alto nivel por el InterNIC. Son:

Dominios genéricos	Para su uso en :
Edu	Instituciones educativas
Com	Organizaciones comerciales
Net	Redes y centros de Información
Org	Organizaciones no lucrativas
Gob	Instituciones de gobierno
Mil	Instituciones militares
Int	Instituciones internacionales

Las abreviaturas para los dominios de países se basan en la norma estandarizada de ISO-3166, donde se identifican los códigos para la representación de nombres de países.

Dominios de Países			
at	Austria	**it**	Italia
au	Australia	**jp**	Japón
ca	Canadá	**mx**	México
ch	Suiza	**se**	Suecia
de	Alemania	**uk**	Inglaterra
es	España	**va**	Vaticano
fr	Francia	**us**	Estados Unidos

4. Los servicios, conexiones y proveedores de Internet

Uno de los aspectos que siempre resultan de gran interés son los beneficios que puede conseguir un usuario conectado a Internet a través de los servicios que este proporciona. Tres son los servicios básicos que ofrece Internet a sus usuarios, independientemente de las aplicaciones utilizadas para explotarlos.

Telnet:

Constituye el servicio de acceso remoto de Internet, que permite conectar dos computadoras de forma transparente para el usuario, con este servicio un usuario puede utilizar una computadora situada en cualquier parte del mundo como si estuviera en su propia casa.

Con Telnet todo lo que teclea el usuario se envía a la computadora remota interactuando con el sistema operativo remoto como si fuera el propio. Para lograr esto, basta que el usuario conozca la dirección de la computadora a utilizar, tenga una cuenta de acceso o nombre identificador de usuario (USERID) y de preferencia un password.

FTP:

Con el servicio FTP el usuario tiene la posibilidad de transferir copias de archivos de una computadora a otra, esta tarea se logra mediante el empleo del protocolo FTP. Los programas FTP permiten transferir archivos de texto, programas, gráficos, archivos de sonido, etc.
De manera similar al Telnet, establecer una conexión remota vía FTP requiere de un nombre de identificación del usuario y de una clave de acceso; algunos servidores FTP permiten accesar archivos públicos con un identificador especial que es anonymous.
Los servidores FTP anónimos aportan muchos beneficios, puesto que constituyen uno de los principales medios de distribución de Software e información en Internet. En estos existe gran cantidad de información que se puede obtener gratuitamente.

Correo electrónico (e-mail)

Consiste en la posibilidad de intercambiar mensajes personales entre los usuarios de la red. Basta con conocer la dirección electrónica para entrar en contacto con el destinatario en cualquier punto de la red.

Entre otros servicios que resultan de particular interés dentro de Internet se encuentran los siguientes:

Usenet:

Este término se aplica al grupo de computadoras que intercambian noticias en la red.
En cada computadora que dispone de este servicio se realizan básicamente dos tareas, una encargada de recibir, memorizar y distribuir los mensajes o también llamados artículos, a otros puntos de la red, y una segunda tarea donde la computadora actúa como un lector de noticias que permite a los usuarios locales recuperar de la computadora Usenet más próxima los artículos de su interés.

IRC:

El Internet Relay Chat o Chat como se conoce, es un servicio que permite a múltiples usuarios mantener una conversación interactiva sobre un tópico determinado. Cuando el usuario se conecta al Chat puede seleccionar en que conversación de las que en ese momento se está llevando acabo se va a incluir. A partir de entonces podrá escuchar, participar en la conversación o saltar a otra diferente. Existen tres tipos de canales para conectarse a una conversación los cuales son:

Públicos, a los que cualquiera puede entrar.
Privados, que se reservan a ciertos usuarios.
Secretos, que son invisibles para el resto de los usuarios.

World Wide Web:

El servicio Web representa uno de los servicios más populares dentro de Internet. Su probable demanda tiene que ver en gran medida por la forma fácil y atractiva con la que recuperar información basada en el hipertexto.

Los servicios mencionados anteriormente muestran la gama de posibilidades a las que puede aspirar el usuario dentro de Internet. Sin embargo, para tener alcance a cualquiera de estas opciones es imprescindible tener un acceso a Internet. Es aquí donde se recurre a los Proveedores de Servicio Internet (ISP) o bien al proveedor de acceso a Internet, quienes se encargaran de proporcionar al usuario una conexión a Internet.

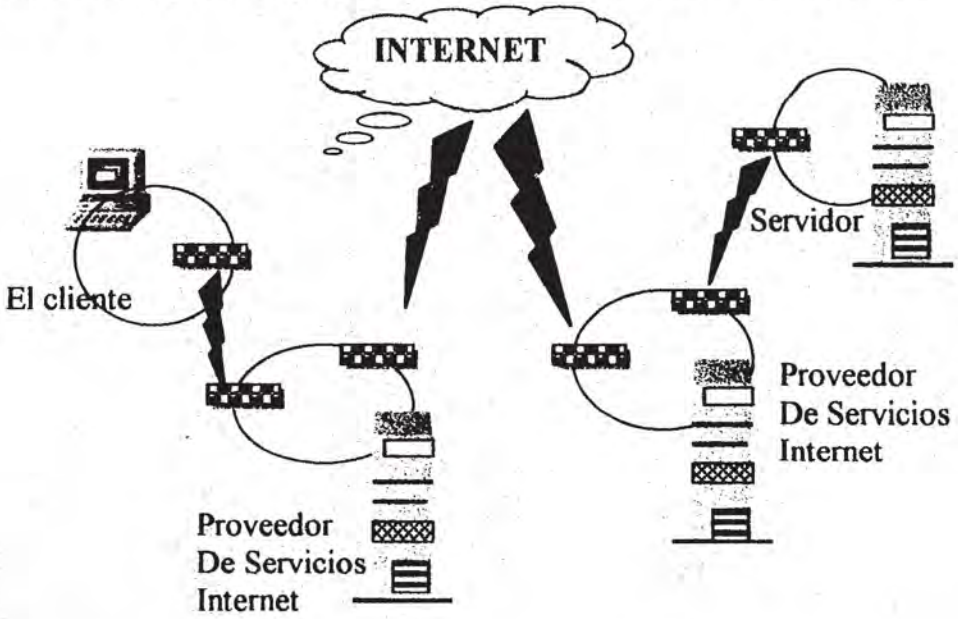


Figura 1.5. – Conexión Típica de Internet

El acceso a Internet través de la línea telefónica se ha convertido en el tipo de acceso de mayor demanda que aparte de necesitar una línea telefónica, también es necesario un Módem con una velocidad de 9600 baudios o superior.

Normalmente, cuando se establece una conexión a Internet a través de una cuenta PPP o SLIP, el proveedor proporciona lo siguiente: un nombre o cuenta de usuario, una contraseña de acceso para iniciar una sesión, el número de teléfono al que deberá llamar su computadora (es decir el número del ruteador o computadora que permite establecer la conexión a Internet), un número IP y quizás un número extra como mascara de subred, y un número para el servidor de nombre de dominio e información para la cuenta de correo.

El número IP de la computadora es su única dirección en Internet. Algunos proveedores asignan un número IP permanente, otros harán que su computadora asigne un número a la computadora del usuario cada vez que se conecta. Si el proveedor asigna un número permanente a la computadora del usuario, es posible que el usuario tenga que elegir un nombre único para que su computadora esté asociada a su número IP.

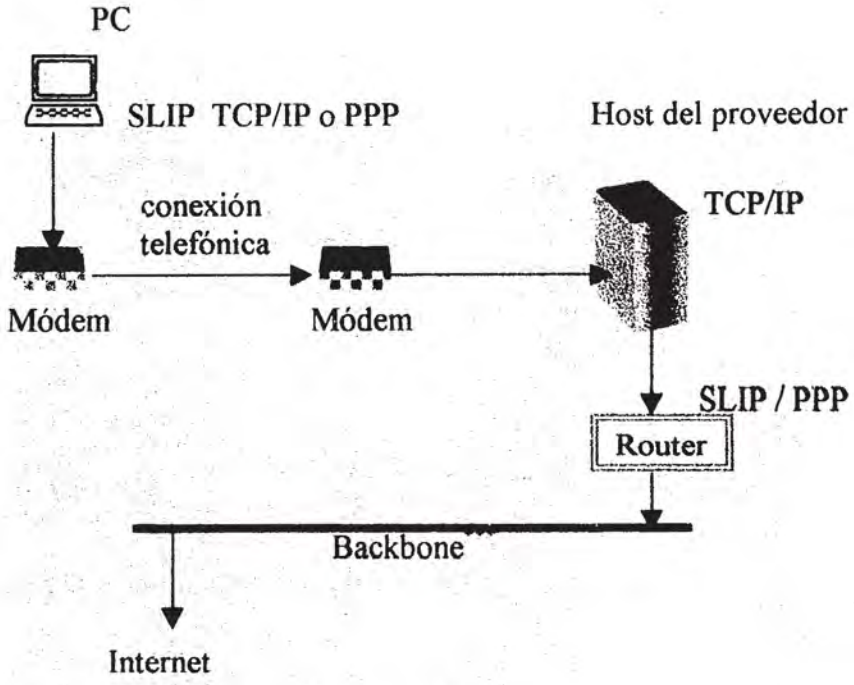


Figura 1.6. - Acceso a Internet vía módem y TCP/IP.

Cuando se entra a Internet con una conexión mediante un módem utilizando protocolos PPP y SLIP las ventajas son claras, pues los equipos y conexiones no son tan costosas como la de una red ligada físicamente. Al entrar a la red la computadora hace una emulación de un nodo Internet ya que se le asigna su propia dirección IP mientras dura el enlace.

*** Backbone (red de columna vertebral) esta expresión se emplea para referirse a una red central que tiene muchos ruteadores conectados. En Internet, una red de columna vertebral utiliza tecnología de Redes de Area Amplia. Cada compañía en particular se refiere también a su red central como una red vertebral, en este caso una red de columna vertebral podría ser una LAN.

El acceso a Internet a través de línea directa es otra opción que prescinde de un módem y constituye el medio más rápido, aunque también el más caro. Con esta opción es posible manejar un numero mayor de usuarios a velocidades más altas de las que puede ofrecer un módem, donde una de las alternativas seria el uso de una línea telefónica digital especifica conocida como (ISDN) o Red Digital de Servicios Integrados. Un acceso de este tipo es utilizado principalmente por las Universidades o Instituciones y por Empresas Comerciales grandes.

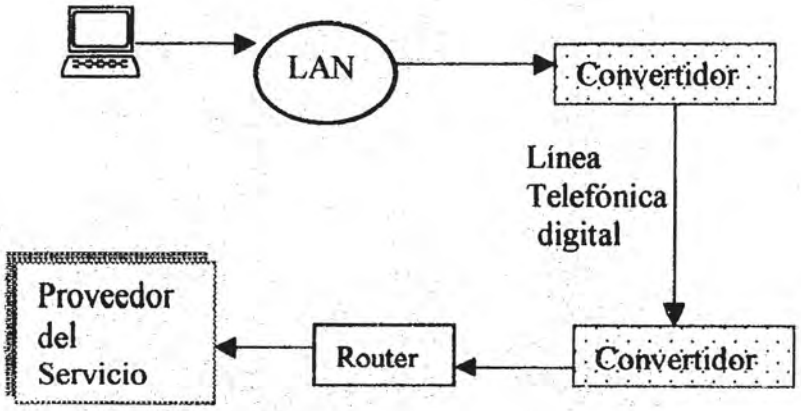


Figura 1.7. - Acceso a Internet a través de línea directa.

Actualmente, existe gran variedad de proveedores de Internet en el mercado como: CompuServe, America On-Line, Prodigy, Infosel, Telmex, por mencionar algunos; bastara con acercarse a ellos para conocer los servicios, tipos de conexiones que ofrecen y los costos de los mismos, de esta forma el usuario estará en posibilidad de elegir el más conveniente a sus necesidades y a su bolsillo.

CAPITULO II

EL SERVICIO GRAFICO DE INTERNET EL WORLD WIDE WEB

El servicio gráfico del Web consagra la boga del Internet mostrando información de todo tipo con imágenes y sonido que cautivaron a los usuarios navegadores. Actualmente el Web se ha convertido en un medio de información poderosa al cual es fácil tener acceso en casi cualquier parte del mundo, la versatilidad de este medio permite usar herramientas como los script CGI que se conocerán más adelante, para la solución de las necesidades generadas en las labores del hombre.

1. Características y elementos del Web.

El Web fue desarrollado por un grupo de investigadores a cargo de Tim Berners Lee en el Laboratorio Europeo de Física en Partículas (CERN) de Suiza en 1989, y constituye una herramienta de recuperación y distribución de información basada en el concepto de Hipertexto, es decir, documentos compuestos de texto, imágenes y sonidos, además de contener enlaces a otros documentos con temas relacionados.
El Web alcanzaría su máxima popularidad en el año de 1993 con la aparición del primer software de fácil uso para navegar en el Web. Esta historia tiene su inicio en un lugar conocido como Centro Nacional de Aplicaciones de Supercómputo (National Center for Supercomputing Applications, NCSA) de la Universidad de Illinois, donde Marc Andreessen trabajaba en un proyecto cuyo propósito era leer las paginas del Web que estaban en formato HTML, pero no en modo texto, sino en forma gráfica y con capacidades de hipermedia. El producto fue MOSAIC, y que por las siglas de este centro de investigación se conoció a esta primera versión como NCSA Mosaic. [1]

A mediados de ese mismo año ya existían unas 1500 paginas del Web, cantidad que ha ido creciendo en forma abrumadora, de tal manera que actualmente hay más de 500 000 lugares con Web (Web sites) y sigue creciendo hasta nuestros días. Posteriormente se crea una nueva versión del programa navegador, denominada Netscape, que incluyo funciones muy útiles para cualquier usuario de la gran red. Esto vino a apoyar aún más la postura comercial del World Wide Web en la gran red Internet.

[1] C, Gonzalo Ferreyra. Internet paso a paso hacia la autopista de la información. Editorial Computec, México. p.145.

El Web representa un sistema de localización de computadoras anfitrionas (host) o lugares con servidores Web (Web sites), donde se ofrecen documentos escritos en formato HTML, (Lenguaje de Marcas de Hipertexto) que incluyen textos, imágenes y sonido; a estos documentos se les conoce como *Pagina Web*, que obviamente contienen referencias o enlaces a otras páginas ubicadas en otros Web sites.

Ingresar al gran mundo del Web es bastante fácil si se cuenta con un programa navegador, que sirva como ventana a través de la cual se mostraran las Paginas Web. Actualmente, los navegadores más populares son: Internet Explorer de Microsoft y Netscape que ya es un estándar. La mayoría de las empresas dedicadas a ofrecer una conexión a Internet proporcionan el programa adecuado en su versión shareware o incluso gratis, en la modalidad de freeware.[2]

El elemento responsable en gran medida del auge del Web, fue precisamente la aplicación del concepto Hipertexto, que hace uso de las ligas o enlaces para dirigirse de un lugar a otro del documento, o hacia otros sitios con documentos Web. El Hipertexto (Hypertext) es el texto contenido en un documento provisto de ligas o enlaces que permiten leerlo en forma no lineal, este concepto fue utilizado por primera por Ted Nelson. Cuando estas ligas llevan a otros documentos de texto, o a ver gráficos, animaciones o vídeo, y a escuchar archivos de sonido, entonces el documento toma el nombre de *hipermedia*.

El gran "boom" del Web sé a dirigido a todas las áreas en las que incursiona el hombre dígase en la educación, en la ciencia y la tecnología, en los negocios y por supuesto, en el entretenimiento. Por esta razón no es de sorprenderse que se encuentre una gran cantidad de paginas Web, donde es posible encontrar todo tipo de información que se pueda imaginar. Esta nueva alternativa de comunicación a hecho a Internet el gran escaparate donde hay lo suficiente y más... , para todos aquellos que se introducen al mundo del Web.

2. EL Web bajo el entorno cliente - servidor.

El proceso de búsqueda de las páginas Web se observa completamente invisible ante el usuario, pero implica una serie de actividades que son posibles realizar bajo el entorno *Cliente / Servidor*. En forma general estos dos términos hacen referencia a los programas o computadoras que trabajan en red, donde se denotan dos computadoras, una solicita los servicios y la otra se dedica a ofrecer los servicios, bases de datos o programas que solicitan los diferentes clientes.

[2] El freeware son programas de creadores de software que no tienen ninguna intención de lucro con su producto, aunque piden respeto para que no sean modificado y distribuido de manera diferente a la original, y que sea utilizado sin fines comerciales. Por otro lado, el shareware constituyen programas cuyos fines sí son comerciales.

Por tanto se define al cliente, como el programa que envía una petición de servicio a una computadora conectada a la red esperando una respuesta. Mientras que el servidor, es un programa de aplicación que ofrece un servicio o paquete de servicios que son solicitados por un programa cliente.

Bajo este principio Cliente / Servidor es posible visualizar una pagina Web en el navegador en donde:

- Un servidor Web es una computadora conectada a la red que recibe las peticiones que los programas navegadores (clientes) le dirigen, las procesa y devuelve como resultado la página Web solicitada.

- Para acceder a una pagina Web desde cualquier navegador conectado a Internet es preciso que se encuentre grabada en un servidor Web.

- Para acceder a la pagina disponible en Internet, debemos indicar al navegador la dirección que tiene asignada, esta dirección se le conoce como: Localizador Universal de Recursos o URL.

El URL es una cadena de caracteres que se le ha llamado universal porque describe todo lo que necesita el navegador sobre un recurso en particular para solicitarlo y mostrarlo.
El URL o dirección de una pagina Web consta de:

Protocolo +	Nombre del Servidor Web	+	trayectoria y nombre del documento html
http://	www.geocities.com		/silicon valley/heights/1779/hoja.htm

Como se observa, la primera parte del URL comienza con HTTP:// , que significa Hypertext Transport Protocol o Protocolo de Transporte de Hipertexto, estas iniciales se usan cuando se solicita un recurso Web. Este protocolo define un conjunto de reglas que sigue el navegador y un servidor remoto para solicitar y facilitar un documento, respectivamente. Un documento HTML utiliza casi siempre el protocolo HTTP, pero los navegadores Web pueden utilizar otros protocolos que variaran de acuerdo al tipo de recurso que se transfiera, como son:

- FTP (para protocolo de transferencia de archivos)
- Gopher (para acceder a los servidores gopher)
- Mailto (para acceder a un servidor de correo SMTP)
- NNTP (para protocolos de transferencia de noticias de la red)
- Telnet (para sesiones de terminal remoto)

La segunda parte del URL es la dirección de la pagina Web, en ella se indica el nombre del servidor Web que proporciona el recurso, más, el directorio donde se ubica el documento Web junto con su respectivo nombre. Los directorios se dividen con una diagonal (/) y no invertida (\) como en el caso de DOS, los archivos o documentos que se muestran pueden ser HTML, imágenes Gif o JPEG, o de texto ASCII.

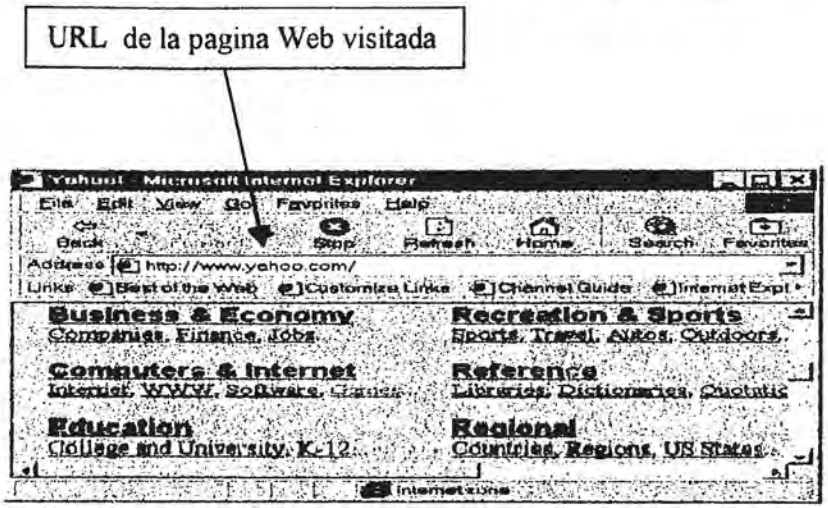


Figura 2.1. La dirección de la pagina Web o URL se especifica en el cuadro address del navegador Web y se oprime [enter] para situarse en la pagina deseada.

El proveedor que realiza la conexión a Internet cuenta con una computadora de propósito especial (Servidor de nombre de dominio) configurada para traducir los nombres de dominio de la computadora o servidor a sus números IP correspondientes. Por ejemplo, supóngase que se desea ver la pagina de alfaomega en el navegador Web, para ello se escribirá **http://www.alfaomega.com** en el cuadro localizador (address) del navegador Web. Antes de que el navegador pueda cargar la pagina solicitada, tiene que consultar a su servidor de nombre de dominio para obtener el número IP del servidor Web: **www.alfaomega.com**. El servidor de nombres de dominio informará el número IP 192.150.170.2 de dicha computadora para localizarla y mostrarla.

3. Paginas Web con HTML (Lenguaje de Marcas de Hipertexto)

Una pagina Web es un documento diseñado para ser visualizado en la pantalla de la computadora a través de un navegador, y que se escribe empleando el Lenguaje de Marcas de Hipertexto, más conocido como HTML.

Históricamente, HTML aparece en el año de 1986 cuando la Organización Internacional de Estándares (ISO) realiza la publicación ISO 88879, donde define un lenguaje de marcas para la creación de documentos estandarizados, dicho lenguaje se denominó: Lenguaje de Marcas Generalizado Estándar (SGML). SGML define muchos tipos de documentos, pero el tipo que resulta de evidente interés es el documento de Hipertexto: HTML cuyas siglas significan, Lenguaje de Marcas de Hipertexto. Desde el punto de vista de SGML, un documento HTML es simplemente un documento en particular (documento de hipertexto) entre otras muchas Definiciones de Tipo de Documento (DTD). Por lo tanto, las tipografías y fuentes para cada componente de un documento HTML ya están definidas. El modo en que los navegadores componen o muestran un documento depende de las capacidades del propio navegador y de la computadora en particular sobre la que se ejecuta.

Contrariamente, más que ser un lenguaje, HTML representa un conjunto de etiquetas o marcas que se escriben junto con el texto para proporcionar información sobre el aspecto que debe presentar. Las etiquetas HTML son palabras y símbolos encerrados entre los signos "< >".

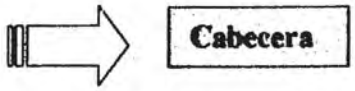
Los navegadores (Netscape y/o Internet Explorer), reciben información en formato HTML e interpretan las etiquetas para presentar el documento según las instrucciones que en ellas se indican.

La construcción de un documento HTML requiere básicamente, de un editor de documentos HTML que propiamente podría ser cualquier editor de textos, que permita guardar el documento como un archivo de solo texto ASCII. Aunque en la actualidad se cuenta con una gran variedad de editores de documentos HTML que facilitan su construcción.

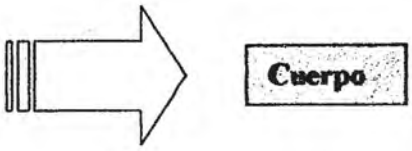
A continuación, se conocerá como se estructura en general un documento HTML cualquiera sin indagar en las etiquetas especificas que podrían usarse, puesto que solo se pretende dar una referencia rápida basándose en que ya se está familiarizado con el diseño de documentos de este tipo.

Un documento HTML se compone de forma general de la siguiente manera:


```
<!doctype HTML public "- / /w30 / / DTD W3 HTML 3.0/ /EN">
<HTML>
<HEAD>
<TITLE>Esta es la pagina de inicio</TITLE>
</HEAD>
<BODY BGCOLOR= "#FFFFFF">
```



..... Aquí se define el cuerpo principal del documento.




```
</BODY>
</HTML>
```

❑ EL PROLOGO

El prólogo es un elemento adicional que puede o no encontrase en un documento. Un prologo es una línea de texto situada al principio del documento HTML que indicara a los futuros navegadores a que versión HTML pertenece. En esta línea de prologo se identifica la DTD (Definiciones de Tipo de documento) según la cual se conforma un documento HTML.

La etiqueta <HTML> confirma al navegador que el texto que sigue esta marcado con HTML, y generalmente viene después del prólogo. La etiqueta </HTML> de terminación es normalmente la última línea de un documento HTML.

❑ LA CABECERA

Todos los documentos poseen una sección de cabecera en donde se define el título para la pagina. Esta sección se identifica con la etiqueta <HEAD> y termina con la etiqueta </HEAD>, dentro de éstas se configura el título con la ayuda de las etiquetas <TITLE> y </TITLE>. El navegador muestra el título del documento en la barra de titulo de la ventana del navegador.

Elementos de información y de cabecera del documento:

Etiqueta	Descripción	Terminador
<BODY>	Define parte de la pagina que pertenece al cuerpo.	</BODY>
<HTML>	Indica el principio de una pagina Web	</HTML>
<HEAD>	Indica el principio de la cabecera, en esta se define el título del documento	</HEAD>
<TITLE>	Especifica el título que mostrará el explorador	</TITLE>
<ISINDEX>	Informa al navegador que se trata de un documento índice en el que se busca por una palabra clave	

▫ EL CUERPO

En la sección de cuerpo se construye y se da formato al texto que se visualizara. Para ello se recurre al uso de cabeceras (palabra o frase que aparece resaltada de algún modo para indicar el título de una sección de texto), texto o párrafos con algún formato, listas ordenadas, listas de menú, así como, la creación de tablas. Otras opciones que son posibles manipular para la construcción de una pagina, es el color del fondo y el color del texto, y con más imaginación, agregar imágenes será ideal para una pagina Web atractiva.

La etiqueta <BODY> puede tomar ciertos atributos para crear un documento HTML más atractivo. Uno de estos atributos es BACKGROUND, en donde se especifica un archivo de imagen que el navegador copiara para crear un fondo personalizado. Este es un ejemplo:

<BODY BACKGROUND= "nubes.gif">

Otro atributo útil es BGCOLOR, para configurar el color del fondo y, TEXT para configurar el color del texto. Los parámetros de estos atributos se representan con 3 números hexadecimales.
La sintaxis para los atributos BGCOLOR y TEXT es la siguiente:

<BODY BGCOLOR= "#5C3317" TEXT= "#FFFF00">

Algunos códigos de colores interesantes se muestran en la siguiente tabla:

Tabla. Código de colores

Código	Color
000000	Negro
0000FF	Azul
00FF00	Verde
FF0000	Rojo
FFFFFF	Blanco
FFFF00	Amarillo
5C3317	Chocolate
8E236B	Granate
FF7F00	Naranja
4F2F4F	Violeta

Elementos para el formateado físico:

Etiqueta	Descripción	Terminador
<B>	Texto en negrita	</B>
<I>	Texto en cursiva	</I>
<S>	Tachado	</S>
_{	Subíndice	}
^{	Superíndice	}
<U>	Texto subrayado	</U>

Elementos de formateado lógico:

Etiqueta	Descripción	Terminador
<ADDRESS>	Fuente cursiva especial para mostrar direcciones	</ADDRESS>
<BLOCKQUOTE>	Marca texto entre comillas	</BLOCKQUOTE>
<CITE>	Cita	</CITE>
<CODE>	Código de lenguaje de programación	</CODE>
<H1>	Marca texto con estilo de cabecera 1	</H1>
<H2>	Marca texto con estilo de cabecera 2	</H2>
<H3>	Marca texto con estilo de cabecera 3	</H3>
<H4>	Marca texto con estilo de cabecera 4	</H4>
<H5>	Marca texto con estilo de cabecera 5	</H5>
<H6>	Marca texto con estilo de cabecera 6	</H6>

Elementos de menú y lista

Etiqueta	Descripción	Terminador
<DD>	Define un término en una lista de definición	
<DL>	Lista de definición	</DL>
<LI>	Elemento de lista de línea para cualquier tipo de lista	
<OL>	Lista ordenada	</OL>
<UL>	Lista sin ordenar	</UL>

A) El uso de las Tablas

Un elemento que resulta bastante conveniente para el diseño de paginas, es incluir el uso de tablas, en donde será posible agregar texto, datos, imágenes, ligas o enlaces y cualquier cosa que el lenguaje reconozca.

Para la creación de una tabla se emplea la etiqueta <TABLE>, si se desea, el atributo BORDER ayuda a especificar un borde para la tabla y con esto darle mayor vista. Toda tabla posee columnas y filas, para su manipulación y diseño se utilizan las siguientes etiquetas: <TH>, esta etiqueta da un formato particular a una cabecera de columna; <TR>, define el inicio de una fila y </TR> marca el fin de la fila; con <TD>, se marcan los datos a colocarse dentro de una celda, así como, especificar un formato en particular para los datos de la celda, y su terminador a usar es </TD>.

Existen otros atributos con los que se manipula la estructura de una tabla como son:

COLSPAN y ROWSPAN son atributos para la etiqueta <TD>, donde los parámetros que toman corresponden al numero de columnas y filas sobre las que se debe extender la celda. Una celda puede abarcar múltiples filas, múltiples columnas o ambas.

Dos atributos para la etiqueta <TABLE> son: CELLSPACING y CELLPADDING, con las que se controla el espaciado de las celdas. CELLSPACING, ajusta la cantidad de espacio que el navegador coloca entre las celdas, y CELLPADDING permite ajustar la cantidad de espacio entre el borde de la celda y su contenido.

De forma general una tabla se construye con las siguientes etiquetas:

```
<TABLE>
<TR><TH>Cabecera de columna 1</TH><TH>Cabecera de columna 2</TH></TR>
<TR><TD>Datos de la celda</TD><TD>Datos de la celda</TD></TR>
<TR><TD>Datos de la celda</TD><TD>Datos de la celda</TD></TR>
</TABLE>
```

B) Vínculos de Hipertexto o ligas.

Cuando se emplea un vínculo de hipertexto en un documento HTML, éste se convierte en un documento de hipertexto, generando la posibilidad de viajar a otras paginas dentro del Web pulsando sobre los vínculos o ligas.

Un vinculo de hipertexto se crea utilizando la etiqueta <A>, donde su atributo más importante es HREF, que define el objeto (documento, Web site, imagen) al que se enlaza el vinculo. El texto que se coloca entre las etiquetas <A> y </A> se convierte en la marca o foco, donde se pulsa para realizar el enlace o vinculo hacia el objeto definido.

Los vínculos se pueden realizan hacia una pagina local, es decir, dentro de la misma pagina o hacia otra parte de un mismo documento.

Cuando el usuario pulsa sobre este vinculo lo lleva directamente al archivo o documento especificado haciendo una búsqueda a través del directorio raíz y sus subdirectorios.

<A HREF= "/index.html"> INDICE </A>

Otra opción, es cuando el usuario se ubica en un documento determinado y desea establecer un enlace hacia una parte especifica de este mismo documento. Estos enlaces se pueden establecer como sigue.

<A HREF= "#uno"> Tema I. La economía mundial </A>
<A HREF= "#intro"> Introducción </A>

Al asignar un nombre a la parte especifica del documento al que se quiere saltar, se identifica fácilmente la sección seleccionada cuando el usuario pulsa sobre el vínculo. Esto es:

<A NAME= " uno" > La economía mundial </A>
<A NAME= "intro"> Introducción </A>

Para viajar por Internet visitando distintos Web sites o paginas Web en cualquier parte del mundo que se ubique un usuario, basta con la siguiente sentencia:

<A HREF= http://www.tectron.com/info/bienve.html> visite tectron </A>

Las imágenes, iconos o gráficos son una opción más para realizar vínculos a través de éstos. Dicha liga se vería de la siguiente forma:

<A HREF= "prevpage.html" ><IMG SRC= "flechaizq.gif" ALT= "pagina anterior"></A>

Atributos de liga o enlace:

Símbolo	Descripción
<A>	Marca el principio de una etiqueta de ancla que termina con </A>
HREF	Define una referencia hipertexto a otra pagina
NAME	Nombre opcional que se puede asignar a la liga para hacerla destino de la liga HREF

Códigos variados:

Símbolo	Descripción
 	Salto de línea
<HR>	Crea una línea horizontal
<IMG>	Inserta una imagen
<P>	Marca inicio de párrafo
<	Representa el símbolo "<" dentro de un documento
>	Representa el símbolo ">" dentro de un documento
&quot	Representa la doble comilla (") dentro de un documento
	Espacio de no- terminación o ruptura
Align= left	Tabula hacia la izquierda el texto
Align= center	Centra el texto
Align = right	Tabula hacia la derecha el texto

Los comandos de este lenguaje están en un constante proceso de evolución, de tal manera que en poco tiempo se han definido los estándares HTML de nivel 1 al nivel 4.

Atributos de liga o enlace:

Símbolo	Descripción
<A>	Marca el principio de una etiqueta de ancla que termina con </A>
HREF	Define una referencia hipertexto a otra pagina
NAME	Nombre opcional que se puede asignar a la liga para hacerla destino de la liga HREF

Códigos variados:

Símbolo	Descripción
 	Salto de línea
<HR>	Crea una línea horizontal
<IMG>	Inserta una imagen
<P>	Marca inicio de párrafo
<	Representa el símbolo "<" dentro de un documento
>	Representa el símbolo ">" dentro de un documento
&quot	Representa la doble comilla (") dentro de un documento
	Espacio de no- terminación o ruptura
Align= left	Tabula hacia la izquierda el texto
Align= center	Centra el texto
Align = right	Tabula hacia la derecha el texto

Los comandos de este lenguaje están en un constante proceso de evolución, de tal manera que en poco tiempo se han definido los estándares HTML de nivel 1 al nivel 4.

CAPITULO III
EL WEB INTERACTIVO A TRAVES DE LOS SCRIPT CGI

Hasta hace poco cuando se entraba a una Página Web, era común encontrar un documento con imágenes, texto, quizás sonido y obviamente ligas o enlaces a otras paginas, mientras que el usuario, se mostraba contemplativo y receptor de esa gran cantidad de información.

Sin embargo, la postura pasiva del usuario tomo un cambio importante resultado de las necesidades inmediatas de comunicación, donde las partes involucradas usuario y servidor Web, debían retroalimentarse para generar una dinámica. El cambio implicaba modificar la forma convencional de como se diseñaban las paginas Web para generar documentos Web activos. Un documento Web activo, consiste en un documento Web que permita al usuario intercambiar datos o información con el sitio Web (Web site), de esta manera, se proporciona una respuesta inmediata a las inquietudes del usuario que visita la pagina Web.

El nuevo recurso que ayudo en esta transformación se nombro: **Interfaz de Pasarela Común**, que corresponde a las siglas CGI, (en inglés Common Gateway Interface) que en la jerga informática se le conoce como: **script CGI**.

1.Concepto y funcionamiento del Interfaz de Pasarela Común (CGI).

"Un script CGI es un programa ejecutado bajo los auspicios de un servidor Web y solicitado por el navegador, bien directamente o bien referenciado en el documento HTML que se está procesando en ese momento."[1]

De forma precisa un script se interpreta como un programa. Aunque en una definición llana un "programa" se refiere a un código ejecutable en formato binario, la palabra "script" se emplea para referirse a archivos de texto que encierran una secuencia de instrucciones. El objetivo principal del script es actuar como intermediario entre el programa navegador que solicita datos, y el lugar en el que estos se encuentran. Las paginas Web interactivas que envían y/o reciben datos suelen comunicarse con scripts situados en el servidor Web, el servidor se encarga de darle tratamiento correspondiente a la información recibida consultando archivos, bases de datos, etc. para enviar los resultados hacia el navegador que solicita la información.

---

[1] Palacio B., Juan; Perl paginas web interactivas; Ediciones ra – ma, Madrid, 1998, p. 17

Una forma de concretar y comprender que es la interfaz CGI, es definiendo cada una de las palabras del acrónimo Interfaz de Pasarela Común.

- ✓ **Interfaz** – utiliza un método bien definido para interactuar con un servidor Web.
- ✓ **Pasarela** – proporciona a los usuarios una forma de tener acceso a diferentes programas que proporcionan bases de datos, imágenes, archivos, etc.
- ✓ **Común** – interactúa con diferentes sistemas operativos.

Con la interfaz CGI se define un método para que el servidor Web a través del empleo de un script acceda a aplicaciones externas desde el contexto de un documento Web interactivo.

El funcionamiento de un script o programa CGI implica un proceso a través del cual se logran paginas Web interactivas.

El proceso de interacción inicia cuando el lector de una pagina Web envía información al sitio Web, propiamente al servidor que hospeda dicha pagina, donde un programa que se ha instalado en dicho servidor gestionara esa información y generara una respuesta que será enviada hacia el lector, cerrando de esta forma el ciclo de la comunicación establecida.

Esquemáticamente un script CGI funciona como se muestra en el siguiente diagrama.

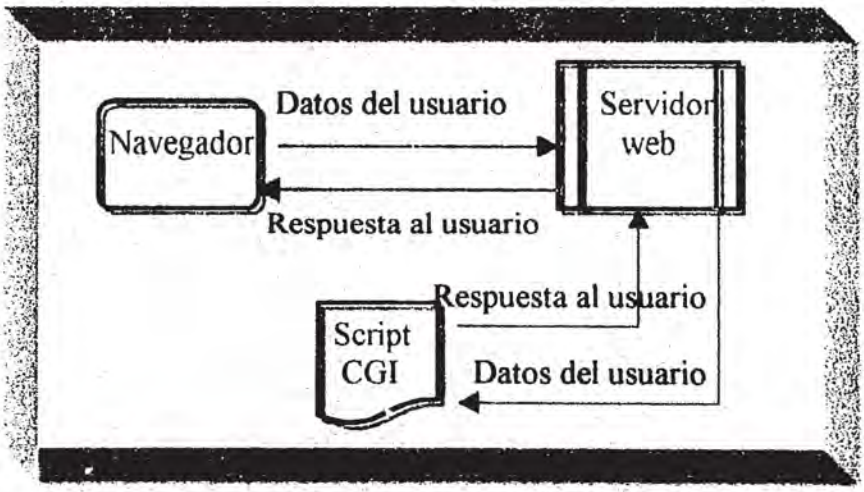


Figura 3.1. Diagrama del funcionamiento del script CGI.

Una forma de concretar y comprender que es la interfaz CGI, es definiendo cada una de las palabras del acrónimo Interfaz de Pasarela Común.

- ✓ **Interfaz** – utiliza un método bien definido para interactuar con un servidor Web.
- ✓ **Pasarela** – proporciona a los usuarios una forma de tener acceso a diferentes programas que proporcionan bases de datos, imágenes, archivos, etc.
- ✓ **Común** – interactúa con diferentes sistemas operativos.

Con la interfaz CGI se define un método para que el servidor Web a través del empleo de un script acceda a aplicaciones externas desde el contexto de un documento Web interactivo.

El funcionamiento de un script o programa CGI implica un proceso a través del cual se logran paginas Web interactivas.

El proceso de interacción inicia cuando el lector de una pagina Web envía información al sitio Web, propiamente al servidor que hospeda dicha pagina, donde un programa que se ha instalado en dicho servidor gestionara esa información y generara una respuesta que será enviada hacia el lector, cerrando de esta forma el ciclo de la comunicación establecida.

Esquemáticamente un script CGI funciona como se muestra en el siguiente diagrama.

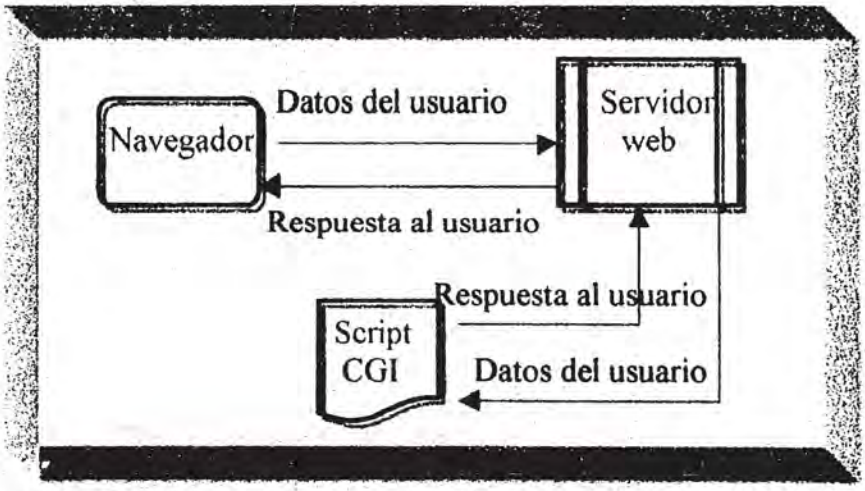


Figura 3.1. Diagrama del funcionamiento del script CGI.

Los script CGI podrán ser escritos en cualquier lenguaje de programación que no requiera un entorno especial para ejecutarse. Algunos de estos lenguajes son:

C / C++
Perl
Visual Basic
AppleScript
Cualquier shell de Unix

El CGI no existe de forma aislada, depende de los estándares de HTML y HTTP. HTML es el estándar que permite a los navegadores Web comprender el contenido de documentos HTML y HTTP es el protocolo de comunicaciones que entre otras cosas, permiten a los servidores Web conversar con los navegadores Web.

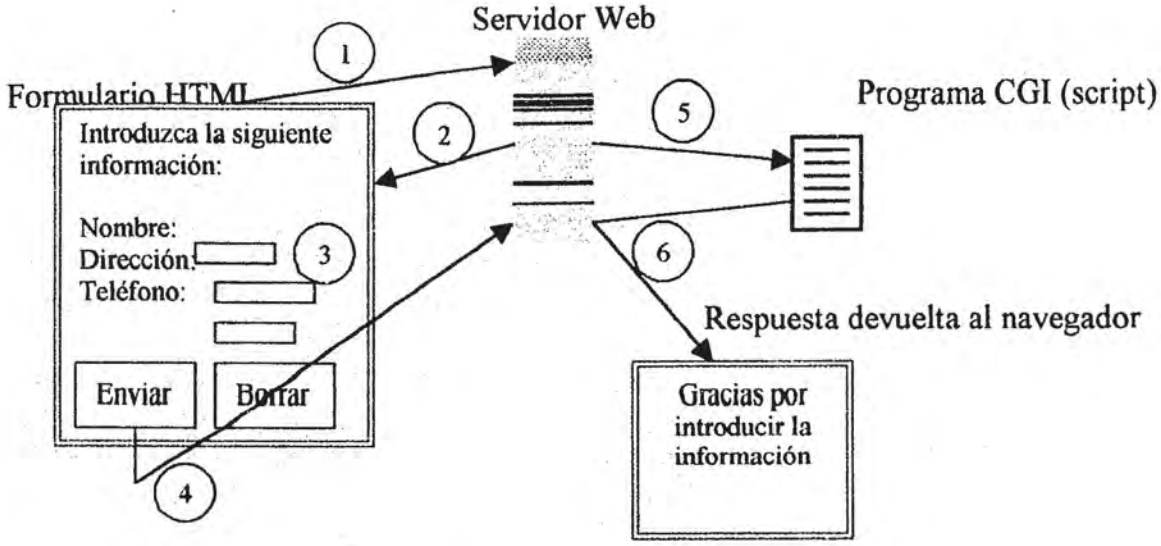
Existen diferentes alternativas para el desarrollo de documentos Web interactivos, puede ser a través del uso de los Formularios HTML, o bien con directivas HTML o SSI (Server Side Include). Con cualquiera de estas alternativas el usuario podrá intercambiar información con. el servidor Web.

2. El Formulario HTML

Un método alterno para crear paginas Web interactivas consiste en el uso de los Formularios HTML. Un formulario HTML permite manipular mayor cantidad de información para darle usos muy variados, desde consultas y actualizaciones a bases de datos, hasta realizar compra/ventas de diversos artículos o hacer alguna reservación para viajar.

Un formulario HTML se diseña dentro de una pagina Web y se compone de un conjunto de cuadros de entrada o campos que admiten texto y/o números, de igual manera, se dispondrá de determinadas opciones o valores para que el usuario elija entre ellas la información que desea enviar al servidor Web.

En la siguiente figura se muestra el proceso de intercambio de información esquemáticamente:



1. El navegador solicita un documento HTML del servidor Web.
2. El servidor Web envía el documento, que incluye un formulario.
3. El usuario introduce la información requerida en el formulario.
4. Cuando el usuario pulsa el botón enviar, el navegador envía la información de los campos del formulario y el nombre del programa CGI a ejecutar.
5. El servidor ejecuta el programa CGI con los datos del formulario.
6. El programa CGI procesa los datos y genera una respuesta de salida que se enviara al usuario.

Una de las ventajas que se observa con el empleo de los formularios es que es bastante fácil la introducción de datos y las respuestas generadas se devuelven como cualquier solicitud de documento Web.

Elementos que integran un formulario HTML

A continuación se muestran los elementos o etiquetas HTML con los que se diseña un formulario.

La etiqueta con la que se define un formulario es: **<FORM>**, y su terminador es **</FORM>**, dentro de éstas sé contendrán los campos que el usuario habrá de llenar, además, podrá contener etiquetas HTML como cualquier documento HTML.

Una pagina Web podrá contener tantos formularios como se desee, pero cada formulario deberá apuntar a un programa CGI con el que se procesara los datos provenientes de los formularios.

La etiqueta FORM acepta los siguientes atributos:

- ACTION.- Con este atributo se asocia un formulario con un programa CGI.

  El parámetro para ACTION es simplemente un URL que referencia al programa CGI, es decir, señala el nombre y la dirección del programa al que el servidor deberá redirigir los datos para su procesamiento.

- METHOD.- El parámetro de este atributo le dice al navegador que método se emplea para el envío de los datos del formulario hacia el servidor Web.

  Los dos métodos posibles son: GET y POST, los cuales se analizaran más adelante.

- ENCTYPE.- A través de este atributo se le indica al navegador el tipo de contenido MIME[*] que se utilizara para enviar los datos del formulario al servidor.

De igual manera, en que los programas CGI comunican el tipo de contenido MIME cuando envían una respuesta al navegador, los navegadores deberán de informar del contenido MIME al enviar los datos al servidor.
Así el tipo de contenido MIME que utiliza el navegador para enviar los campos del formulario es: "aplicación/x- www –form-urlcodificado". Dado que éste es el tipo implícito, se puede omitir este atributo dentro de la etiqueta FORM sin ningún problema.

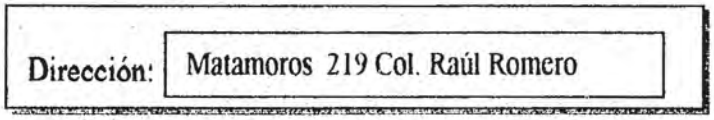
[*] Abreviatura de Multipurpose Internet Mail Extensions (Extensiones Multipropósito de Correo Internet). Aunque los mensajes de Internet sólo pueden contener texto en ASCII, la extensión MIME permite que un mensaje de correo electrónico posea archivos que no están en ASCII.

Los tipos de campos que se pueden emplear para el diseño de un formulario son:

Cuadro de texto.- El cuadro de texto es un editor de texto básico que puede aceptar casi cualquier tipo de datos, incluyendo caracteres de letras, palabras, espacios, números y signos de puntuación.

La etiqueta de cuadro de texto es:

<INPUT TYPE ="text" NAME ="direccion">



Para las etiquetas de campos es imprescindible el uso del atributo NAME para identificar el campo, ya que el nombre asignado a NAME pasa al servidor cuando el usuario presenta el formulario. Por lo tanto la asignación del nombre es decisión libre del diseñador.

Otros atributos de utilidad son:

*VALUE, el cual se utiliza para asignar un valor predefinido al campo.

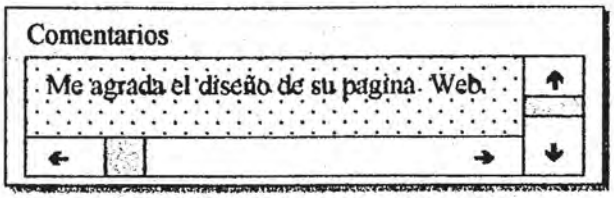
*SIZE, puede especificar la extensión del cuadro de texto.

*MAXLENGTH, especifica el número máximo de caracteres que se pueden introducir en un campo.

Cuadros de texto desplegable.- Un cuadro de texto desplegable es un cuadro que puede aceptar múltiples líneas de texto. Incluye barras de desplazamiento y controles para moverse a lo largo de la ventana del cuadro de texto.

La etiqueta para un cuadro desplegable es:

<TEXTAREA NAME = "comentario">



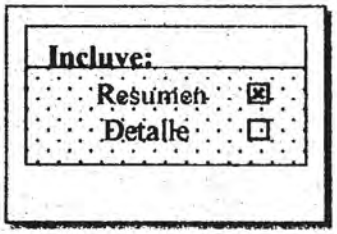
Algunos atributos que se emplean para este tipo de campos son ROWS y COLS, que permiten definir el tamaño visible del cuadro de texto desplegable sin que limite la cantidad de texto que se puede escribir en él.

<TEXTAREA NAME ="comentario" ROWS =3 COLS = 30>

Cuadro de comprobación.- Un cuadro de comprobación muestra opciones para activar o desactivar junto a un pequeño cuadro que muestra una X cuando se pulsa sobre él.

La etiqueta para el cuadro de comprobación es:

<INPUT TYPE ="checkbox" NAME = "opción">



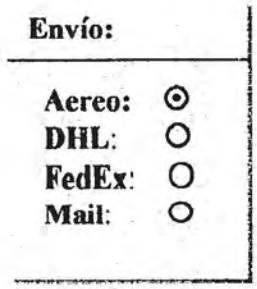
Un atributo útil para el cuadro de comprobación es CHECKED (activado) que se emplea para definir el estado inicial del cuadro, por lo general estos se manejan como desactivados.

El atributo CHECKED no toma ningún parámetro y para utilizarlo basta con incluirla dentro de la sentencia de la etiqueta del cuadro de comprobación.

Botones de radio.- Los botones de radio representan un campo que puede tomar valores de entre diferentes opciones.

La etiqueta para un botón de radio es:

```
<INPUT TYPE = "radio"  NAME ="envío"  VALUE ="aéreo">
<INPUT TYPE = "radio"  NAME ="envío"  VALUE ="dhl">
<INPUT TYPE = "radio"  NAME ="envío"  VALUE ="fedex">
<INPUT TYPE = "radio"  NAME ="envío"  VALUE ="mail">
```

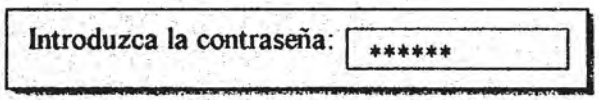


Como se observa los elementos o campos que forman parte de un mismo grupo o contexto se les asigna el mismo nombre aunque con distintos valores.

Campos de contraseña.- Un campo de contraseña es bastante similar a un cuadro de texto, pero con la particularidad de que en él no se podrán ver los caracteres que se escriben, puesto que estos se observan como asteriscos o cualquier otro símbolo.

La etiqueta para el campo de contraseña es:

```
<INPUT TYPE ="password"  NAME ="clave">
```



Campos ocultos.- Los campos ocultos como su nombre lo indica estos no aparecen en el formulario, y más que beneficiar al usuario ayuda al desarrollador del Web en el rastreo de información.

Por ejemplo, cuando en un sitio Web con seguridad que requiere una contraseña de acceso, no querrá que los usuarios autorizados tengan que introducir su contraseña cada vez que intenten cargar una nueva pagina Web, en vez de eso puede hacer que introduzca la contraseña una vez, entonces el servidor devuelve un código secreto que se cargara en el campo oculto con el atributo VALUE.

De esta forma cuando se presente el formulario, el código secreto volverá a pasar al servidor, permitiendo que el programa CGI identifique al usuario como autorizado.

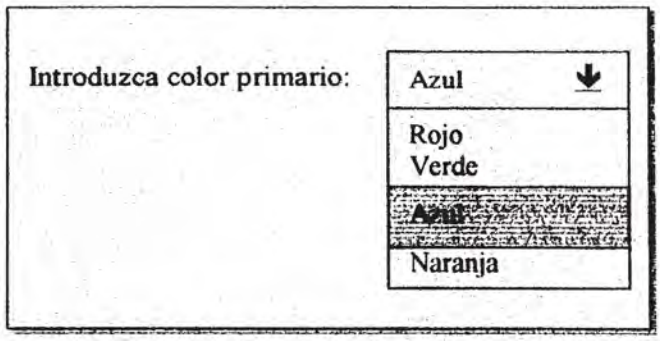
La etiqueta para el campo oculto es:

```
<INPUT TYPE ="hidden"  NAME ="oculto">
```

Cuadros de lista de opción única.- Un cuadro de lista permite que el usuario haga una sola selección entre distintas opciones. Funcionalmente, el cuadro de lista es muy parecido al botón de radio pero es más compacto.

Un cuadro de lista se define de la siguiente forma:

```
<SELECT  NAME ="selección">
      <OPTION>rojo
      <OPTION>verde
      <OPTION>azul
</SELECT>
```

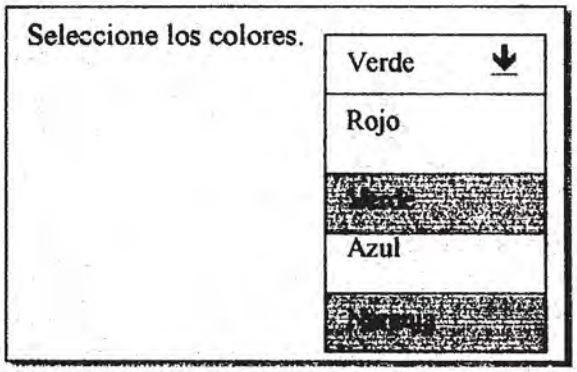


En un cuadro de lista las opciones se marcan con la etiqueta <OPTION>, después de la ultima etiqueta <option> se debe terminar con el elemento </SELECT>.

Cuadros de lista de opción múltiple.- Un cuadro de lista de opción múltiple es básicamente igual aun cuadro de lista, pero permite que el usuario haga múltiples selecciones de las diferentes opciones que muestra la lista.

Para tener un cuadro de lista múltiple bastara con incluir el atributo MULTIPLE en la etiqueta <SELECT>.

```
<SELECT MULTIPLE NAME ="selec_mul">
    <OPTION>rojo
    <OPTION>verde
    <OPTION>azul
    <OPTION>naranja
</SELECT>
```



Botones.- En los formularios se manejan dos tipos de botones: *Submit y Reset.*

Botón Submit.- Cuando se pulsa sobre el botón Submit, el explorador envía el contenido de los campos del formulario al servidor Web, quien a su vez se los pasa al programa CGI especificado en el formulario.

La etiqueta para el botón submit es:

```
<INPUT TYPE ="submit">
```

Botón Reset.- Al pulsar sobre el botón Reset este inicializara o limpiara los campos del formulario sin necesidad de volver a cargar la pagina.

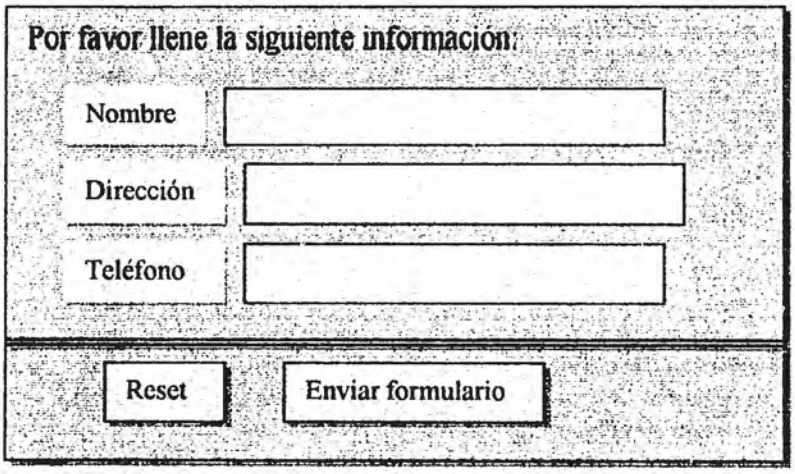
La etiqueta para el botón Reset es:

<INPUT TYPE ="reset">

Con el botón submit aparece por defaul sobre el botón el texto "submit query", y para el botón reset, aparece el texto "reset". Sin embargo, si se desea modificar los textos que muestran ambos botones, se debe incluir el atributo VALUE.

Por ejemplo:

<INPUT TYPE ="submit" VALUE = "Enviar formulario">



3. La Interacción del servidor Web con el script CGI

El servidor Web es un elemento protagónico en la ejecución de los script CGI, a través de éste se establece el canal de comunicación entre el formulario HTML y el script encargado de procesar la información del formulario, razón por la cual, es necesario conocer aspectos importantes del servidor Web que aloja tanto al formulario como al script.

Para ejecutar un servidor Web se necesita de una computadora de características especiales que ofrezca gran rapidez y que cuente con un sistema operativo multitarea y multiusuario. Hoy en día, la mayoría de los servidores Web se ejecutan bajo sistemas operativos UNIX de gran capacidad.

El servidor Web es un programa cuyo propósito será recibir las peticiones de los navegadores Web que solicitan documentos HTML, de esta forma cuando recibe una petición busca el documento HTML y lo envía al navegador que lo solicito.
El servidor Web poseen una conexión física única a Internet, sin embargo también utiliza una serie de conexiones lógicas. Estas conexiones lógicas no son más que conductos separados a través de los cuales se producen distintos intercambios de datos o servicios, estos conductos se nombran como puertos y se identifican por medio de un número. Por omisión, los servidores Web y los navegadores Web utilizan el puerto 80.

Un servidor Web ofrece cuatro funciones principales:

- Sirve paginas Web
- Ejecuta programas o script CGI y devuelve su salida.
- Controla el acceso al servidor
- Supervisa y registra las estadísticas de acceso al servidor.

El modo en que se implementan estas funciones varía de un servidor a otro.

Todos los servidores Web contienen un conjunto de archivos de registro que incluye normalmente:

Un registro de acceso
Un registro de agente
Un registro de errores

El servidor registra los accesos en el *registro de acceso.* El formato para el registro de acceso contiene: el nombre o numero IP del sistema que hace la petición, la fecha y hora, la ruta y nombre del documento solicitado, el código de éxito o fracaso resultante, y el número de byte transferidos.

El *registro de agente* contiene el nombre y la versión del agente de contacto (navegador Web). Los administradores Web pueden utilizar el registro de agente para ver que navegador utilizan los usuarios para visualizar sus paginas.

El servidor registra cualquier error que se produzca en el *registro de errores.* Los errores incluyen peticiones de paginas que no existen, peticiones de paginas para las que no hay un permiso adecuado, intentos de ejecutar script CGI que no existen y errores de verificación de usuario. El registro de errores incluye la fecha y hora, el nombre del host y de dominio del sistema solicitante, la naturaleza del error y el motivo del error.

Actualmente, existen servidores Web que se ejecutan tanto en plataformas UNIX como en Windows y hasta en LINUX. Por ejemplo, la organización NCSA desarrolladora de Mosaic, posee un servidor robusto y versátil para los sistemas Unix, llamado httpd (demonio http).
Otra alternativa es Website de O `Reilly & Associates, servidor Web nuevo y potente para Windows NT y Windows 95/98.

Lo primero que se debe considerar para la ejecución de una aplicación CGI en Internet, es contar con un servidor Web en el que sea posible grabar el programa o script y la pagina o documento Web activo. Una alternativa es recurrir a las paginas personales que conceden las empresas proveedoras de acceso a Internet a sus clientes, o a través de una empresa especializada en el hospedaje de Webs.

El siguiente paso será guardar la aplicación CGI en el directorio adecuado para que el servidor Web pueda acceder a ésta.
Cuando se ejecuta por primera vez el servidor Web, lee los archivos de configuración que le indican donde encontrar sus archivos y directorios, es decir, para ubicar el directorio raiz del Web. A modo de ejemplificar como se ubican y distribuyen los archivos y directorios en un servidor Web bajo la plataforma Unix, observemos el siguiente esquema.

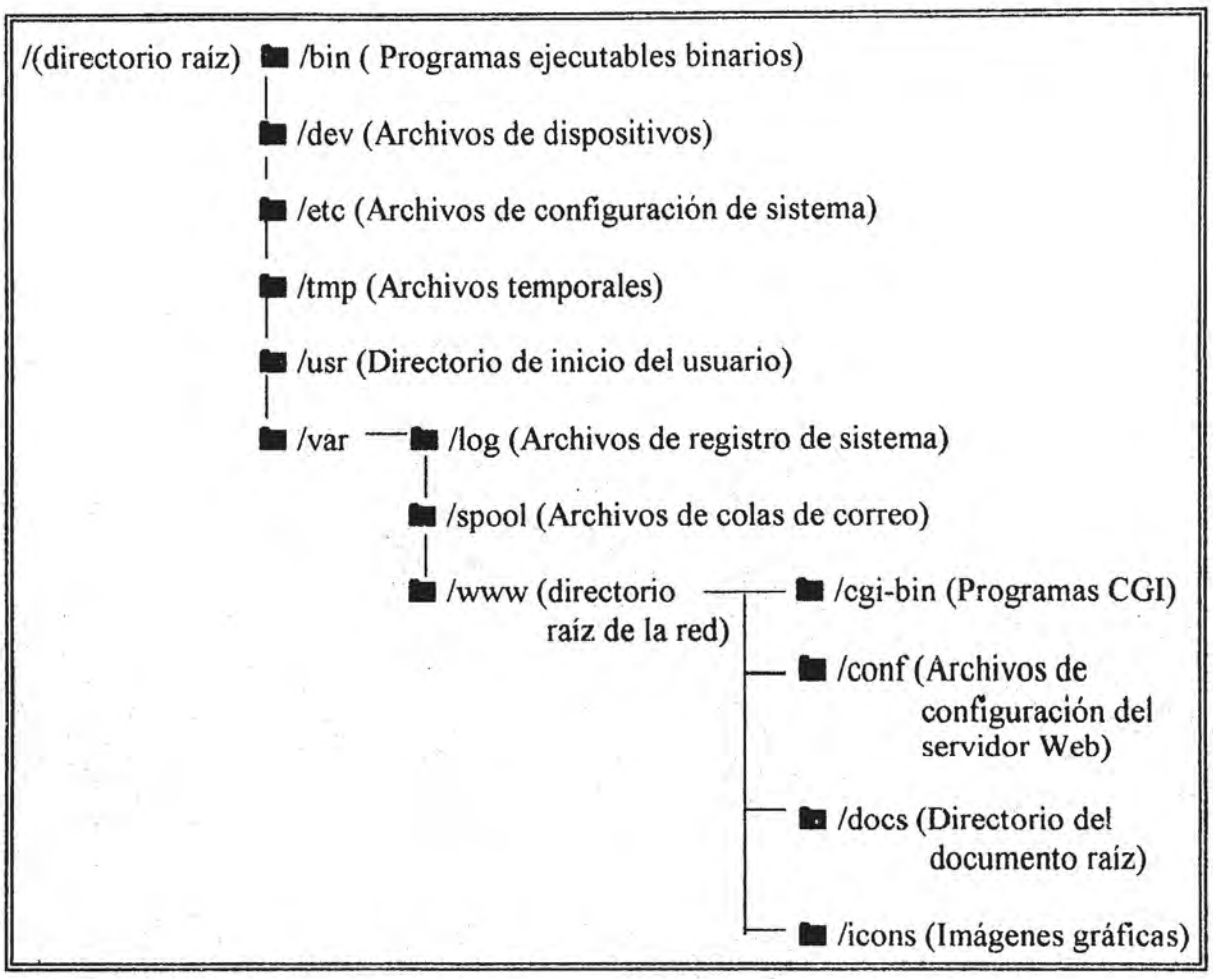


Figura 3.3. Ejemplo Arbol de directorios de un servidor Web

Por lo regular, los servidores Web se configuran de manera que todas las aplicaciones CGI se coloquen en un directorio cgi –bin. No obstante, el servidor Web podría tener un alias de modo que existieran "directorios virtuales". Cada usuario podría tener su propio directorio cgi – bin. La ubicación de este directorio está bajo el control del administrador del sitio Web.

Un aspecto importante antes de colocar los programas en un directorio determinado, es considerar las normas estrictas sobre los permisos que se les deben asignar a los archivos, dígase programas CGI o archivos de datos, sobre todo si se trata de un sistema multiusuario como Unix.

En un sistema operativo multiusuario son muchos los usuarios que pueden acceder a él, y cada uno de ellos tiene sus propios archivos y directorios. En Unix, como en otros sistemas operativos, es posible tener archivos y directorios privados que no pueden ser leídos por otros usuarios, o bien pueden ser leídos pero no modificados. Por esta razón no es de extrañarse la implantación de los **permisos para los archivos** en los servidores Web.

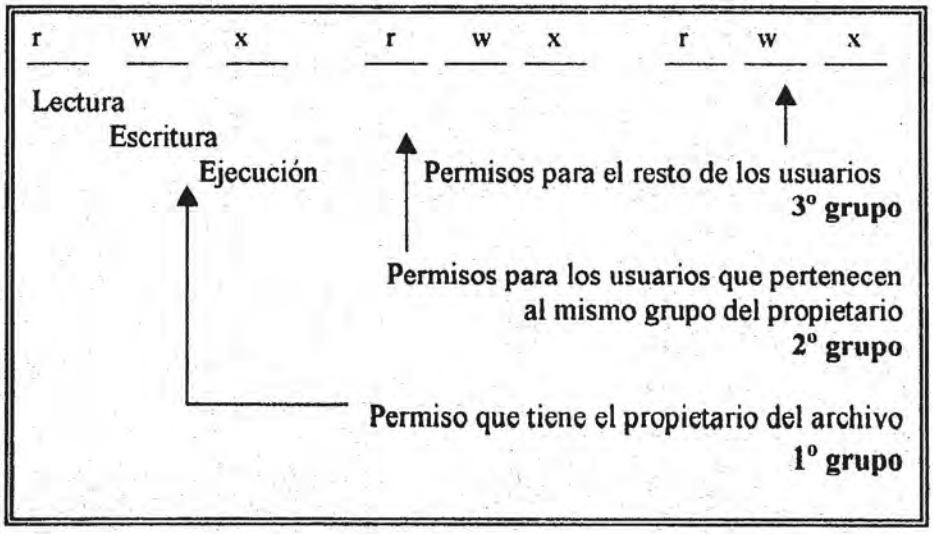
El comando *dir* (en Unix *ls-l*) muestra el siguiente resultado

```
drwxr -xr -x  5 luis DP3725 97 jun 16   14:33 cgi-bin
-rwxr --r --  1 luis DP3725 97 jun 10   9: 10 index.html
```

La 1ra columna indica si se trata de un directorio o un archivo, así como los permisos que sobre él tienen los demás usuarios para ejecutarlo, leer y escribir. En esta columna se observan letras y guiones que en realidad son tres grupos de tres letras cada uno. En las siguientes columnas muestra el nombre del propietario del archivo, fecha de modificación y nombre del archivo.

En el siguiente esquema se muestra como se agrupan los permisos para un archivo

Significado de la columna de permisos



Significado de los caracteres que aparecen en un grupo.

Permiso de lectura: si es una **r** (read) indica que se puede leer, si es guión no se tiene permiso para leer.

Permiso de escritura: la letra **w** (write) indica permiso para escribir o modificar el archivo o directorio.

Permiso de ejecución: la letra **x** (execute) permite ejecutar el archivo o archivos contenidos en un directorio.

Los permisos también se pueden expresar a través de un código de números que utiliza tres dígitos. Cada uno de estos dígitos es el resultado de una suma compuesta de tres dígitos:

DIGITO	PERMISOS
4	Lectura
2	Escritura
1	Ejecución

Así, el dígito 7 representa la suma de los tres permisos, donde el 5 autoriza a leer y ejecutar, el 6 deja leer y escribir pero no ejecutar.

Los script CGI deben tener el código 755, que permiten un acceso completo al propietario y autoriza a los demás para ejecutarlo y leerlo.

La cuestión de los permisos en los archivos o directorios de un servidor Web son de vital importancia para el correcto funcionamiento del programa CGI, especialmente si estos programas generan archivos para contener datos.

Cuando un usuario llama a través de un navegador a un programa CGI o a una pagina Web, el servidor que responde a esta solicitud recibe a este usuario con un nombre genérico que tiene definido para todos los navegadores que solicitan paginas o programas. Cuando el programa que se ha llamado crea un archivo, el propietario de ese archivo no será el programador que lo hizo, sino el usuario genérico con cuyo nombre los navegadores acceden al sistema.
Una alternativa para que el propio programador sea el propietario y pueda manipular los archivos creados por el script, es crear previamente un archivo en blanco en el servidor. Así sólo se añadirán nuevas líneas en el archivo existente, de esta forma el usuario no será propietario de los archivos creados por el programa que se está ejecutando. La otra opción, es dejar esta tarea de manipulación de los archivos creados al administrador del servidor[2].

[2] Juan palacio, perl paginas Web interactivas, editorial Ra – ma, Madrid 1998. P. 113.

Los servidores de paginas Web dispone de una serie de mecanismos de seguridad que permiten especificar desde que directorios se pueden ejecutar script, qué usuarios pueden acceder a ellos y contraseñas. Algunos administradores configuran su servidor para permitir a los usuarios instalar programas en cualquier directorio; en otros casos, es necesario grabar en el directorio raíz del usuario y en el subdirectorio que vaya contener los script un archivo con el nombre "*.htaccess*". Éste es un archivo especial para Unix. En él se definen los permisos y contraseñas que el usuario establece para trabajar con los archivos contenidos en el directorio.

Un aspecto que es importante mencionar, es sobre el formato en que se graba el archivo script dentro del servidor. El uso de un tipo de archivo ya sea binario o texto para guardar el script implica diferencias en su procesamiento dependiendo del sistema operativo encargado de su ejecución. Comúnmente, cuando se graba un archivo script en modo binario, los saltos de línea, que en DOS están formados por dos caracteres y en Unix por uno sólo generan diferencias al momento de ser leído por el sistema operativos que emplea el servidor Web. Preferentemente los archivos script deben grabarse como un archivo de texto, de modo que puedan ser leídos correctamente por el servidor que los procesa, sobre todo si se piensa en trabajar bajo sistemas operativos Unix.

Mientras el servidor es el canal de comunicación entre el formulario y el script, la ejecución propia del script será responsabilidad del lenguaje interprete o compilador usado en su diseño y creación, el cual obviamente se ubica dentro de algún directorio del servidor Web. Dentro de los lenguajes que se emplean están: C, C++, Perl, o algún Shell de Unix.
Es de vital importancia especificar dentro de la primera línea del script la ruta del directorio dónde se ubica el lenguaje interprete o compilador que lo ejecutara. Previo a la creación del script CGI, se debe consultar con el administrador del servidor Web la ubicación del lenguaje interprete o compilador con el que se puede contar para el desarrollo y programación del script.

Para ejecutar un programa CGI basta con teclear el URL (Local Recurso Universal) de éste en el navegador. El servidor Web deberá reconocer que se está solicitando un programa CGI que deberá ejecutarlo.

Un ejemplo del URL sería de la siguiente forma:

http://localhost/cgi-bin/prueba.pl

El servidor Web ejecutara el script CGI y el navegador Exhibirá cualquier salida que éste genere.

El URL del programa CGI es una trayectoria "virtual". La ubicación real del Script o programa en el servidor dependerá de la configuración del servidor y del tipo de servidor que se utilice. Por ejemplo, si se está ejecutando bajo el sistema operativo Linux y el servidor Web NCSA en una configuración " estándar", entonces la trayectoria virtual de su ubicación se traducirá en /usr/local/etc/httpd/cgi-bin/prueba.pl . Si se estuviera ejecutando en el servidor WebSite bajo Windows 95, la trayectoria traducida será: /website/cgi-shl/prueba.pl.

Existen otras formas de invocar programas CGI además de utilizar una URL, una opción puede ser a través de un vínculo de hipertexto.

 haga click aquí para ejecutar el script CGI.

O bien, utilizando el botón "enviar" sobre un formulario HTML, y en otros casos con el uso de las directivas SSI (server-side include), esta ultima alternativa se verá más adelante.

Cuando el servidor Web ejecuta un programa CGI abre de manera automática los siguientes identificadores de archivo de entrada/salida estándar: *STDIN, STDOUT, STDERR,* y prepara ciertas variables para la recepción y almacenamiento de los datos enviados desde el navegador.

- STDIN.- Este archivo representa la entrada estándar del programa CGI que contiene información generada por el formulario HTML. Propiamente, se refiere al canal de entrada de datos conectado al servidor de Web..

- STDOUT.- Es la salida estándar de un programa CGI que se vincula al archivo STDIN del navegador. Esto quiere decir que, cuando se imprime información usando alguna instrucción o función como print () dentro del programa CGI, se está escribiendo directamente en la ventana del navegador. En general se refiere a la pantalla o al canal de salida del servidor que invoca un script.

- STDERR.- Es el archivo de salida estándar de un programa CGI que se vincula a la bitácora de errores del servidor. Por ejemplo, la salida de la función die (), muestra un mensaje indicando que un archivo no se puede abrir y se colocara en la bitácora del servidor. STDERR permite guardar mensajes de estado que no desea que el usuario vea. Sin embargo, hay que considerar que no todos los sistemas operativos ni todo el software del servidor conectan STDERR al archivo de bitácora.

El uso de estos archivos identificadores pueden tener variantes con relación al sistema operativo y al software del servidor Web que se utilice, razón por la cual, es conveniente hacer uso de los manuales para conocer como actúan los archivos estándares de entrada / salida del servidor

Los servidores Web también ponen a disposición de los script una serie de variables con información que puede ser útil; éstas son las llamadas "**variables de entorno**".
Las variables de entorno no son exclusivas de los servidores Web, sino que en una computadora personal también hay una serie de variables definidas en el entorno operativo, pero al tratarse de entornos distintos las variables disponibles son por lo tanto distintas.

Al iniciarse la ejecución de un programa CGI, el servidor Web inicializa diversas variables de entorno a las cuales podrá tener acceso el script. En el siguiente cuadro se muestran una descripción de las variables de entorno más utilizadas.

Tabla 1 variables de entorno que puede consultarse de un servidor Web.

Variable	*Descripción*
AUTH_TYPE	Proporciona, en forma opcional, el protocolo o tipo de autentificación de acceso a un script.
CONTENT_LENGTH	Muestra la longitud en bytes de la cadena de parámetros enviada al script a través del identificador de archivo STDIN. Se usa en particular con el método POST en el procesamiento de formularios.
HTTP_cadena	Señala información sobre el cliente.
PATH	Contiene información sobre la trayectoria de la solicitud http que invocó al script.
QUERY_STRING	Contiene la cadena de parámetros enviados por el método GET cuando se procesa un formulario.
REMOTE_ADDR	Contiene la dirección IP del usuario que hace la petición.
REMOTE_HOST	Proporciona el nombre del dominio del sitio desde el que se ha conectado el usuario.
REQUEST_METHOD	Método mediante el cual se pondrá a disponibilidad del script información sobre el envío de parámetros (GET o POST).

SERVER_SOFTWARE	Contiene el nombre y versión del software del servidor Web. (por ejemplo Website/1.1e.)
DOCUMENT_NAME	Proporciona el nombre de la pagina que ha llamado al script
DOCUMENT_URL	Proporciona el nombre de la pagina que ha llamado al script, incluyendo su trayectoria.
GATEWAY_INTERFACE	Proporciona la versión del CGI que maneja el ervidor Web local. (por ejemplo CGI/1.1.)
HTTP_ACCEPT	Proporciona una lista separada por comas de los tipos MIME que acepta el software del navegador. Se revisa esta variable para ver si el cliente acepta cierto tipo de archivo gráfico.
HTTP_FORM	Muestra la dirección e-mail del usuario. No todos los servidores Web darán esta información.
HTTP_USER_AGENT	Muestra el tipo y versión del navegador Web del usuario.
PATH_TRANSLATED	Mapea la trayectoria virtual del script (desde la raíz del directorio del servidor) a la trayectoria física empleada para invocar al script.
REMOTE_USER	Proporciona, opcionalmente, el nombre utilizado por el usuario para accesar su script protegido.
SERVER_NAME	Contiene el nombre de host configurado para el servidor.
SERVER_PORT	Contiene el numero de puerto sobre el que el servidor Web local está escuchando. El numero estándar de puerto es 80.
SERVER_PROTOCOLO	Contiene la versión del protocolo Web que usa este servidor. (por ejemplo HTTP/1.0.)

Existen dos métodos por los cuales se envían los datos del formulario hacia el script, estos son GET o POST. Normalmente, GET será el método predeterminado.
Dependiendo del método que se trate, el servidor alista ciertas variables para colocar en ellas los datos recibidos y de esta forma los datos sean accesibles al script que los procesara.

El Method = " GET"

Supóngase que se enviara la siguiente información con el método GET.

Código de cliente: 125
Fecha de viaje: 15-11-1999
Destino: Cancun

Al pulsar sobre el botón "enviar" del formulario, el navegador realiza las siguientes acciones:

*Examina la etiqueta ACTION para ubicar la dirección del script al que enviara los datos.

http://www.servidor.com/cgi-bin/reservas.cgi

*Busca la etiqueta METHOD para determinar la forma de envío de los datos. Para el caso del método GET, el navegador crea una cadena codificada que contiene el URL (dirección del script) junto con los datos que se enviaran. Esta cadena se vería como sigue:

http://www.servidor.com/cgi-bin/reservas.cgi?COD=125&FECHA=15-11-1999&DEST=Cancun

Como se observa el envío GET conforma una cadena con los siguientes elementos

Dirección del script: http://www.servidor.com/cgi-bin/reservas.cgi
Signo de interrogación (?)
Parejas clave = valor separadas por el signo (&): COD=125&FECHA=15-11-1999&DEST=Cancun

Una vez enviada la información desde el navegador hacia el script, lo primero que debe hacer el script es identificar a través de que método fueron enviados los parámetros.
Para ello consulta la variable de entorno **REQUEST_METHOD**, cuyo contenido es: GET.

Con el método GET la cadena enviada se almacena en la variable de entorno **QUERY_STRING,** guardando todo lo que el navegador puso después del signo de interrogación (?).

La cadena almacenada es:

COD=125&FECHA=15-11-1999&DEST=Cancun

El Method = " POST"

Retomando el ejemplo anterior, supóngase que se enviara la siguiente información con el método POST.

Código de cliente: 125
Fecha de viaje: 15-11-1999
Destino: Cancun

Cuando el usuario oprime el botón "enviar" del formulario, el navegador ubica la dirección del script al que enviara los datos, enseguida, compone una cadena codificada con los pares CLAVE=VALOR unidos por el signo (&).

La cadena codifica es:

COD=125&FECHA=15-11-1999&DEST=Cancun

A diferencia del método GET, la cadena codificada no se envía pospuesta a la dirección del script, sino que llegara hasta el servidor como una corriente de datos independiente.

El script deberá identificar el método que se uso para realizar el envío de parámetros, consultando la variable de entorno **REQUEST_METHOD**. En este caso el contenido de la variable de entorno es: POST

Los datos enviados mediante el método POST llegan hasta el script a través de la entrada estándar, conocida como STDIN. La longitud de la cadena de datos que llega se almacena en la variable de entorno **CONTENT_LENGHT**.

En el siguiente cuadro se resume las variables de entorno que interviene en los métodos de envío de datos al script CGI.

Figura 3.4. Esquema de las variables de entorno que consulta el script CGI.

METODO	IDENTIFICA EL METODO	¿DÓNDE SE ALMACENAN LOS DATOS?
POST	REQUEST_METHOD	ENTRADA ESTANDAR Longitud = CONTENT_LENGHT
GET		QUERY_STRING

Las organizaciones Web han establecido condiciones o estándares sobre el uso del protocolo HTTP, de esta forma obliga a que el contenido de los comandos, respuestas y datos que se transmitan entre el cliente navegador y el servidor deben estar claramente definidos. Normalmente, junto con el URL se envían datos o parámetros que deberán ser evaluados por un programa, dicha información se transforma en una cadena codificada bajo un formato especifico.

En ocasiones, resulta difícil saber simplemente por el contexto si la información lleva un carácter de espacio delimitando un campo o un carácter de espacio real entre dos palabras. Para aclarar la ambigüedad, se creo el "**Esquema de Codificación URL**". Con este esquema se evitan ambigüedades semánticas, así, todos los espacios se convierten en un signo de más (+) y los caracteres especiales se convierten a su equivalente en hexadecimal anteponiéndoles un signo de porcentaje (%).[3]

Por ejemplo, la siguiente cadena:

Jorge García<dj@planet.net>

Se codifica como. Jorge+Garc%A1a+%3Cdj@planet.net%3E.

[3] Medinets, David. Perl 5 a través de ejemplos; editorial Prentice Hall; España; 1989. P. 418.

Si se observa, el carácter "<" se convirtió a %3C y el carácter ">" se convirtió en %3E. Otro carácter que se considera especial es la vocal acentuada "í" que al codificarse se convierte en %A1

Para que el script CGI procese la cadena de información que recibe del formulario debe decodificar los caracteres especiales que aparecen en la cadena trasmitida y separar los campos pares CLAVE = VALOR, de esta forma, los datos serán entendibles para los procesos que le siguen.

Cuando se ejecuta un script CGI, los resultados del procesamiento de los datos se envían a una salida estándar que no es la pantalla, sino un canal que se encarga de dirigir los resultados hacia el servidor, quien a su vez los transmitirá por la red hacia el navegador que los ha solicitado.

Por consecuencia es importante considerar los siguientes puntos:

- Los programas navegadores pueden recibir información en varios formatos: texto, HTML, imágenes GIF o JPG, sonido MIDI, etc.
- Cuando un servidor envía un archivo, sabe por la extensión del mismo el tipo de datos que contiene.
- Los scripts que envían datos de respuesta hacia el navegador antes de generar la respuesta deben mandar primero una línea con la cláusula: "*Content-type*" informando al servidor del tipo de datos que deberá remitir al cliente.

Por esta razón, el navegador debe disponer en cada caso de la información necesaria sobre el tipo de documento que ha de proveer y la extensión que se le abra de especificar.

El sistema con que se estandarizo el significado de estas extensiones se conoce como: Multipurpose Internet Mail Extensions (MIME), o también, Extensiones de Multipropósito de Correo Internet.
Las especificaciones MIME hacen posible que un tipo determinado de archivo quede especificado indicando dos palabras clave separadas por una "/", donde la primera palabra indica el tipo de contenido y hace corresponder el archivo con uno de los siete grupos predefinidos que se muestran en la tabla siguiente.

En el siguiente cuadro se muestran algunos ejemplos de extensiones MIME.

Tabla 2. Cuadro de extensiones MIME.

Tipo MIME	Extensión
Text/html	**.html, .htm, shtml**
Audio/wav	**.wav**
Audio/mid	**.mid**
Image/gif	**.gif**
Image/jpeg	**.jpeg, .jpg**
Video/mpg	**.mpg**

Actualmente, los servidores Web integran una característica llamada Server Side Includes o directivas SSI, que ofrece una alternativa más para crear documentos HTML dinámicos. Una directiva SSI es un conjunto de funciones integradas dentro de los servidores Web que dan a los desarrolladores de las paginas HTML la capacidad de insertar datos en documentos HTML por medio de directivas especiales.

Una diferencia única, entre un programa o script CGI y una directiva SSI, es que los script CGI deben enviar a la salida el encabezado "content-type" como su primera línea de salida.

Cuando el servidor Web recibe una solicitud de envío de pagina desde el navegador, revisa el contenido del archivo HTML que envía por sí en él apareciera alguna etiqueta con directivas que deberá ejecutar y cambiar por el resultado obtenido. El servidor lee línea por línea y si localiza los caracteres <! --# que tienen un significado especial para él, sabrá que todo lo contenido hasta -- > no deberá ser enviado al navegador, sino procesarlo y mandar en su lugar la respuesta obtenida.

Un ejemplo común para saber como funciona una directiva SSI, es cuando al navegar se entra a una pagina que muestran su número de visitantes. Donde el documento HTML de dicha pagina, incluye la directiva a través de la cual ejecuta el programa CGI que calcula el número de visitantes. Esta directiva se vería de la siguiente forma.

```
<p align ="center"><font size ="5">Esta pagina ha sido
visitada: <!--#exec cgi = "cgi-bin/contador.cgi" -- > veces.</font>
```

En la siguiente figura se observa una pagina Web que incluye una directiva SSI para mostrar a los usuarios el número de visitantes al sitio Web.

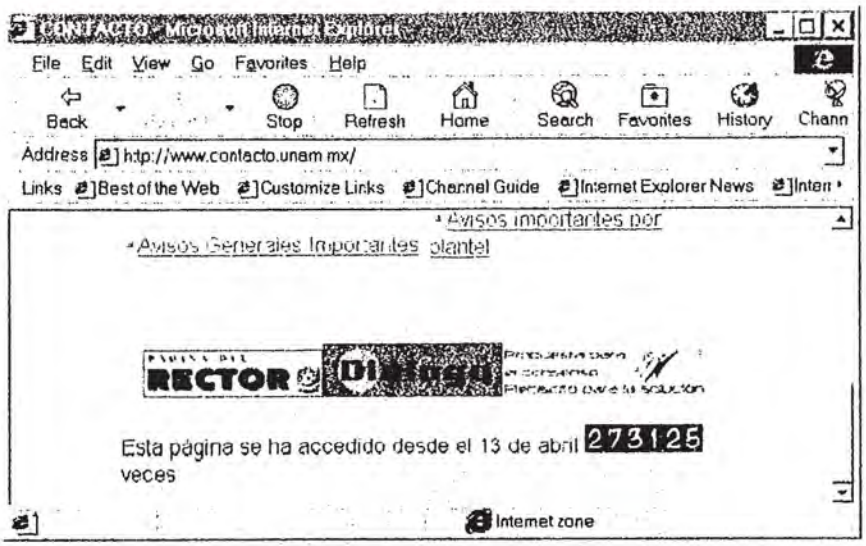


Figura 3.5 Pagina Web que emplea directivas SSI para llamar al script CGI que devuelve el numero de visitantes de esta pagina.

A continuación se muestra un cuadro con algunas directivas SSI útiles para el diseño de paginas interactivas.

Directiva	Acción
Include	<!--#INCLUDE FILE ="fichero.html" --> Incrusta el fichero indicado. Las referencias de directorio que se incluyen son relativas al directorio actual. <!--#INCLUDE VIRTUAL ="/ficheros/fichero.html" --> Incrusta el fichero indicado. Las referencias de directorio que se incluyen son absolutas.
Echo	<!--#ECHO VAR ="DATE_LOCAL" --> Incrusta la fecha y hora locales <!--#ECHO VAR ="LAST_MODIFIED" --> Incrusta la fecha de la última modificación del documento HTML que incluye esta directiva. <!--#ECHO VAR ="DOCUMENT_URL" --> Incrusta la ruta y nombre del documento HTML que incluye esta directiva <!--#ECHO VAR ="DOCUMENT_NAME" --> incrusta el nombre del documento HTML que incluye esta directiva

Exec	<!--#EXEC CMD ="pwd.../graphics" --> Ejecuta la sentencia UNIX encerrada entre comillas como si hubiera sido tecleada desde la línea de comandos por el usuario. <!--#EXEC CGI ="usr/local/bin/contador.pl" --> Ejecuta el script cgi referido en las comillas.

Todas las directivas SSI se ven como comentarios HTML dentro de un documento. De esta forma, las directivas simplemente serán ignoradas en los servidores Web que no las manejen. Algunos servidores Web sólo buscan directivas SSI en las paginas grabadas con la extensión shtml.

4. Aplicaciones de los CGI

Las aplicaciones CGI pueden realizar una gran variedad de tareas. Por ejemplo, se puede crear paginas Web sobre la marcha, accesar a bases de datos, mantener sesiones de telnet, generar gráficas y reunir estadísticas.

Actualmente, con la gran cantidad de usuarios que visitan sitios Web, donde se llevan acabo transferencias de datos o de información, el tema de la seguridad es de vital importancia. Hoy en día, los sitios Web ofrecen espacio en su servidor Web para aquellos usuarios que desean participar incluyendo su propia pagina Web o para aquellos usuarios que interactuan con los visitantes de su pagina Web y que deseen ejecutar sus propios script. Por esta razón, los administradores de los servidores Web recurren a aplicaciones especificas del CGI, conocidas como utilerías, para proteger y resguardar su información y sistema.

Una de estas utilerías es **CGIwrap**. CGIwrap es una utilería basada en UNIX escrita por Nathan Neulinger, que permite a los usuarios generales ejecutar script CGI sin necesidad de accesar el directorio raiz cgi-bin del servidor. CGIwrap realiza varias revisiones de seguridad sobre los script antes de cambiar el ID (nombre identificador de usuario) para que corresponda con el del propietario del script Los CGI que se ejecuten mediante CGIwrap se almacenaran en un directorio cgi-bin bajo el directorio publico de cada usuario. Los programas o script se ejecutan empleando el ID del usuario, de modo que el daño se limite a su directorio base.

CGIwrap funciona con NCSA, apache y probablemente en cualquier otro servidor Web UNIX. Puede encontrar información adicional en el sitio Web:
http://www.umr.edu/-cgiwrap/htdocs/install.html

La mayoría de los Webmasters desean rastrear el progreso de un usuario de pagina a pagina según va haciendo clic en el sitio. Por desgracia, HTTP es un protocolo sin estado. Los protocolos sin estado no tienen memoria; sólo comprenden el comando actual. Un usuario podría visitar un sitio, salir y regresar después, posiblemente con una dirección IP diferente. Quien mantiene el sitio no tiene forma de saber si es o no el mismo navegador.

Como respuesta a esta situación se ha recurrido al uso de **cookies** en los script CGI. Un cookie es una pequeña porción de datos almacenada por el servidor Web en el disco duro local del visitante. Se puede emplear para rastrear su trayectoria a través de un sitio Web y desarrollar un perfil con fines de comercialización o de información. También pueden usarse los cookies para contener información con números de cuenta y de decisión de compras a fin de poder crear aplicaciones de compras.

Precisamente, una de las aplicaciones CGI más populares es la "lista de compras" o también, llamada "**tienda virtual**". Esta aplicación permite a las compañías poner inventarios en línea, en donde los clientes y usuarios pueden revisar los artículos del inventario a través del CGI.

Una lista de compras es una de las aplicaciones con mayor demanda para los administradores, porque ésta integra muchas rutinas CGI en un sistema. Por ejemplo, la orden de cada cliente para mantener una única lista de compras, requiere que la aplicación mantenga una trayectoria de cada cliente y de su lista correspondiente. Para guardar la trayectoria de las actividades del cliente de pagina a pagina, la aplicación CGI tendrá que encontrar la forma de "recordar" estas transacciones, porque el servidor no puede.
Además, la lista misma es un archivo de base de datos que se construye y modifica sobre la marcha basada en las necesidades del cliente. El cliente es capaz de agregar y borrar artículos, tal como, un cambio de cantidades de los artículos ordenados. Este tipo de aplicación demanda un administrador de base de datos completo.
Los script hacen mas que sólo desplegar y modificar los artículos de la lista de compras que están contenidas en el archivo de la base de datos, estos incluso, manipulan los campos de la base de datos para ejecutar cálculos de precios, tal como un subtotal y generar totales.
Esta aplicación permite al cliente visitar y conocer cada objeto dentro de la *tienda o almacén virtual* de una forma completamente interactiva y cómoda.

Otra aplicación bastante útil, es el uso del "**form-mail**", término inglés que se emplea para designar a los script encargados de recibir los datos de un formulario HTML y enviarlos a una dirección de correo.

Este tipo de script resulta muy beneficioso, porque permite recoger cualquier tipo de información de los visitantes y recibirla en cualquier dirección de correo de la gran red. Esta aplicación es de bastante utilidad sobretodo para el Webmaster, quien se dedica a atender las inquietudes de los usuarios que visitan el sitio Web.

CAPITULO IV
FUNDAMENTOS DEL LENGUAJE PERL

En este capitulo se describe de forma general los elementos de programación de Perl, con la finalidad de familiarizarse con la sintaxis de este lenguaje, puesto que a través de este se construirán los script o programas CGI de la aplicación que se pretende desarrollar.

La razón principal por la que se ha acudido al uso del lenguaje Perl para la realización de los script CGI, se debe primordialmente a las características y facilidades que este ofrece en el desarrollo de programas, que entre otras cosas, se destinan a la creación de espacios interactivos dentro del Web.

1. Características del lenguaje Perl.

Perl es un lenguaje de programación desarrollada por Larry Wall, que estructuralmente presenta una mezcla de elementos del lenguaje C y de sentencias del entorno UNIX. Es un lenguaje especialmente útil para llevar a cabo procesos con archivos y cadenas de texto, tareas de mantenimiento en sistemas UNIX y programación de script CGI.

Perl es un lenguaje de uso y distribución gratuita, y dispone de intérpretes para la práctica en diversos sistemas operativos (DOS, Windows 3.x / 95 / NT,UNIX, Mac, etc).Todos ellos se encuentran recopilados en el CPAN, y también están disponibles en numerosos sitios de la red. CPAN (Comprehensive Perl Archive Network) es el resultado de las aportaciones que altruistamente hacen programadores de todo el mundo, poniendo al alcance documentos, referencias, utilidades, librerías, intérpretes, etc.
En la red se puede encontrar la referencia original de perl escrita por Larry Wall, y otras muchas que con mayor o menor rigor explican diversos aspectos del lenguaje, aunque por supuesto, la mayoría están escritas en inglés. La siguiente página de enlaces es un buen punto de partida para localizarlas.

http://www.trcs.com/perl/direcciones.html

Perl es un lenguaje interpretado, esto quiere decir que necesita un programa intérprete para traducir el programa perl, que propiamente es un programa en código fuente, a un programa en código objeto o binario directamente ejecutable.
El intérprete de Perl es un archivo ejecutable que se llama "perl.exe" y que debe estar grabado en el disco de la computadora para poder ejecutar los programas escritos en Perl.

La siguiente instrucción le indica al sistema operativo que llame al intérprete perl.exe pasándole como parámetro el nombre del programa perl a ejecutar.

```
perl saludo.cgi ó perl saludo.pl
```

Para la creación de un programa en Perl basta con escribir las instrucciones que lo componen en un editor de texto y grabarlo como archivo de texto "ASCII" (texto simple, sin ningún tipo de formato)

La razón que ha llevado a elegir a Perl como lenguaje de programación para el script CGI de la aplicación que diseñara tiene que ver principalmente con las siguientes características:

Perl es un lenguaje optimo para:

Recibir y enviar información en forma de texto.
Leer y escribir en archivos de texto.
Trabajar con cadenas.

Además, el hecho de que la mayoría de los servidores Web que trabajan bajo el sistema UNIX incluyen un interprete de perl, explica el porqué los diseñadores de espacios Web ven en este lenguaje la herramienta ideal y accesible para la programación de script CGI.

2. Elementos básicos de la programación en Perl.

En esta sección se describe de manera general el uso de los principales elementos que integran al lenguaje de programación de Perl. Este lenguaje de programación como otros lenguajes, posee una sintaxis particular que emplea elementos básicos de la programación que son: variables, operadores, funciones, instrucciones y operaciones con cadenas y listas. A continuación se dan a conocer los elementos mencionados.

A) variables y operadores

Variables.- Las variables son elementos del lenguaje que permiten la manipulación de datos o información mediante la asignación de valores que pueden cambiar durante la ejecución del programa. Una variable se identifica por el atributo *nombre* que se le asigna como identificador de variable y el *tipo* que describe el tipo de datos que va a contener la variable.

En Perl se distinguen tres tipos de variables: escalares, listas y listas asociativas.

- **Escalares**

Las variables escalares se identifican por el signo "$" seguido del nombre de la variable y pueden almacenar cualquier tipo de valor.

Por ejemplo, variables escalares para almacenar valores numéricos.

```
$fresa = 3;
$piña = 7;
$total_frutas = $fresa + $piña;
print "el número de frutas es: $total_frutas";
```

también puede contener caracteres:

```
$nombre = "Rosa";
$saludo = "hola";
print "$nombre $saludo";
```

Las comillas en Perl no son simples delimitadores de cadenas de caracteres, sino que actúan como una función que devuelve un valor. Las comillas dobles analizan su contenido antes de facilitar el valor, mientras que las comillas sencillas lo devuelven sin aplicarle ningún análisis previo.

- **Listas**

Las listas se identifican por el signo "@" seguido del nombre de la lista y puede contener una o más variables escalares.

Vea el siguiente ejemplo:

```
@conjunto = ("aire", "tierra", "fuego", 7 );
print "@conjunto";
print "\n";
print "$conjunto[3] $conjunto[0]";
```

La primera línea crea una lista de nombre *conjunto* y le asigna cuatro valores.

@conjunto = ("aire", "tierra", "fuego", 7);

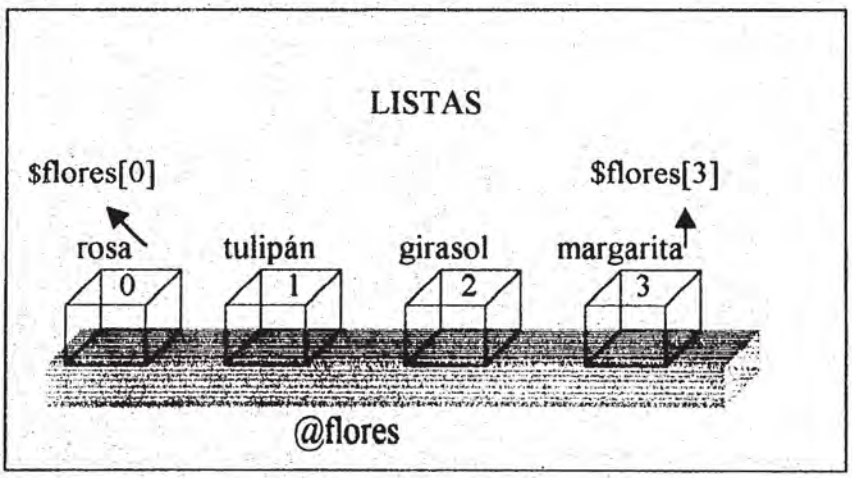
En una lista es posible almacenar más de un tipo de valores que pueden ser alfanuméricos (aire, tierra, fuego) o enteros (7) tal como se muestra.

La línea: *print "@conjunto";* contiene la instrucción print, que se ocupa de enviar datos hacia la salida estándar (pantalla) a menos que se indique otra cosa. Aquí, print muestra la lista completa. Como se dijo anteriormente las comillas dobles no se limitan a encerrar a la variable, sino que interpretan su contenido. El uso de las comillas dobles en las listas devuelve todos los elementos separados por un espacio.

La tercera línea: *print "\n";* emplea los caracteres \n que como en otros lenguajes se interpreta como un salto de línea.

En la última línea se hace referencia a los elementos por separado contenidos en la lista a través de una variable escalar, la variable escalar debe tener el mismo nombre de la lista acompañada de un subíndice o numero de orden del elemento ($conjunto[3] $conjunto[0]).

Dentro de la lista cada elemento se identifica por la asignación de un subíndice numérico entre corchetes ([ ]), que se enumera a partir del valor cero de forma consecutiva. Basta hacer referencia al elemento de la lista mediante un número de subíndice para sustraer el elemento deseado. Observe el esquema.



- **Listas asociativas**

Son listas de variables escalares que asignan una clave a cada uno de los valores que contienen. Se representan por el signo "%" seguido del nombre del nombre de la variable.

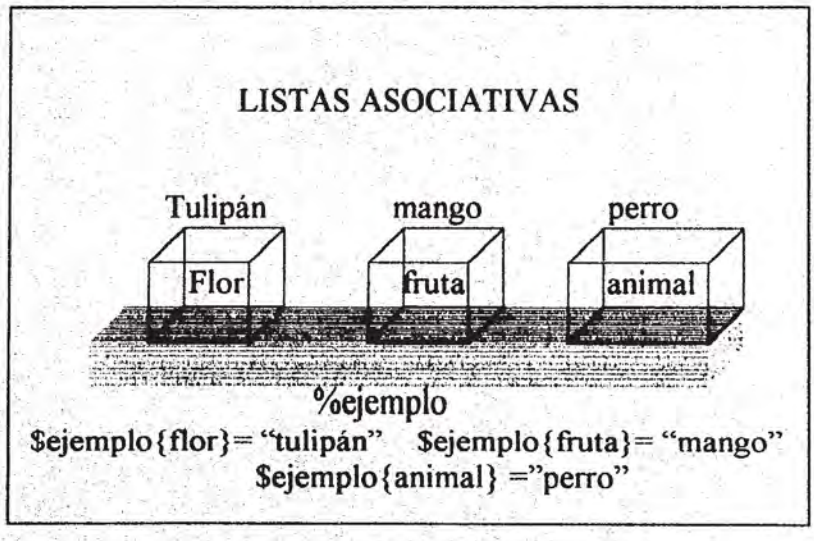
Para ver como funcionan analicemos el siguiente programa.

```
# LISTAS ASOCIATIVAS

%ejemplo = ("flor", "tulipán", "fruta", "mango", "animal", "perro" );
$maceta = $ejemplo{flor};
$alimento  = $ejemplo{fruta};
$mascota  = $ejemplo{animal};
$ejemplo{mineral} = "cuarzo";
print "$maceta $alimento $mascota \n";
@claves =    keys %ejemplo;
print "@claves";
```

La primera línea crea y asigna valores a la lista asociativa *%ejemplo*. En apariencia se asignan 6 variables escalares, aunque en realidad sólo son 3 con sus respectivas claves.

%ejemplo = ("flor", "tulipán", "fruta", "mango", "animal", "perro");

Dentro de las listas los elementos contenidos tienen asignado un lugar de orden que se emplea para referirse a éstos de forma individual. De esta forma Perl considera que el primer valor que lee es la clave de la primera variable, el siguiente, su contenido, y así sucesivamente.



Para hacer referencia a los elementos de la lista asociativa por separado, se asigna una variable escalar con el mismo nombre de la lista asociativa indicando entre llaves el valor clave asociado al elemento deseado.

$maceta = $ejemplo{flor};
$alimento = $ejemplo{fruta};
$mascota = $ejemplo{animal};

También existe la posibilidad de agregar nuevos elementos a la lista a través de la siguiente línea.

$ejemplo{mineral} = "cuarzo";

Aquí, se inserta un nuevo elemento a la lista *%ejemplo*, de valor "cuarzo" y clave mineral.

Esta instrucción ayuda a crear la lista en caso de que no exista, puesto que en perl no es necesario realizar una declaración de variables previa. En el momento que se trabaja con una variable del tipo que sea perl la creará de forma automática.

La línea: *print "$maceta $alimento $mascota \n";* tan sólo imprime en pantalla los valores asignados a las variables escalares ahí señaladas.

En la línea: *@claves = keys %ejemplo;*
aparece la función **Keys**, cuyo objetivo es devolver todas las claves existentes en una lista asociativa

En @claves se crea una lista y se le asignan como elementos las claves de la lista asociativa (flor, fruta, animal, mineral). Y finalmente se imprime el contenido de @claves con la instrucción: *print "@claves";*

la sintaxis de la función es:

keys LISTA ASOCIATIVA

Variables reservadas.-

El lenguaje Perl utiliza variables reservadas para fines específicos, estas variables son de gran utilidad para el almacenamiento de datos que permiten realizar tareas determinadas y optimizar procesos de un programa.

En la siguiente tabla se muestran las variables reservadas de uso común en el lenguaje perl.

También existe la posibilidad de agregar nuevos elementos a la lista a través de la siguiente línea.

$ejemplo{mineral} = "cuarzo";

Aquí, se inserta un nuevo elemento a la lista *%ejemplo*, de valor "cuarzo" y clave mineral.

Esta instrucción ayuda a crear la lista en caso de que no exista, puesto que en perl no es necesario realizar una declaración de variables previa. En el momento que se trabaja con una variable del tipo que sea perl la creará de forma automática.

La línea: *print "$maceta $alimento $mascota \n";* tan sólo imprime en pantalla los valores asignados a las variables escalares ahí señaladas.

En la línea: *@claves = keys %ejemplo;*
aparece la función **Keys**, cuyo objetivo es devolver todas las claves existentes en una lista asociativa

En @claves se crea una lista y se le asignan como elementos las claves de la lista asociativa (flor, fruta, animal, mineral). Y finalmente se imprime el contenido de @claves con la instrucción: *print "@claves";*

la sintaxis de la función es:

keys LISTA ASOCIATIVA

Variables reservadas.-

El lenguaje Perl utiliza variables reservadas para fines específicos, estas variables son de gran utilidad para el almacenamiento de datos que permiten realizar tareas determinadas y optimizar procesos de un programa.

En la siguiente tabla se muestran las variables reservadas de uso común en el lenguaje perl.

Tabla 1. Variables reservadas del lenguaje de programación de perl.

Variable reservada	Como funciona

$_	Almacena la última línea leída. Es una variable de lectura/escritura, es decir, que además de leer su contenido, también es posible asignárselo.
	$archivo = 'citas.txt';
	open (CITA, "$archivo");
	while (<CITA>) {
	print $_;
	}
	close CITA;

$&	Guarda la última cadena encontrada en una comparación o sustitución. Es una variable de sólo lectura. $_ = "Y tu como te llamas?"; /p[a-z]*s/; print $&; #imprimirá "llamas"
$`	Guarda el trozo de cadena que predecía al último encuentro en una comparación o sustitución. Es una variable de sólo lectura. $_ = "Y tu como te llamas?"; /p[a-z]*s/; print $`; #imprimirá "Y tu como te "
$'	Guarda el trozo de cadena que seguía al último encuentro en una comparación o sustitución. Es una variable de lectura. $_ = "Y tu como te llamas?"; /p[a-z]*s/; print $'; #imprimirá "?"

$.	Guarda el número de orden que ocupa la última línea leída.
	
	$archivo = 'citas.txt';
	open (CITA, "$archivo");
	while (<CITA>) {
	print $. $_;
	}
	close CITA;

@ARGV	Guarda la lista de parámetros que se han pasado al programa en la línea de comandos. Todo lo que se teclea detrás del nombre del programa se considera una lista de parámetros separados por un espacio se guardan en @ARGV. Perl programa.cgi parame1 parame2 parame3  $ARGV[0] $ARGV[1] $ARGV[2]

@_	Guarda la lista de parámetros que se han pasado a una subrutina.
	$total = &multiplica (3,2);
	print $total;
	exit;
	
	sub multiplica {
	($a, $b) = @_;
	return ($a*$b);
	}

$_[n]	Guarda el elemento n de la lista almacenada en @_.
	$total = &multiplica (3,2);
	print $total;
	exit;
	
	sub multiplica {
	return ($_[0]*$_[1]);
	}

%ENV	Guarda la lista asociativa cuyas claves son los nombres de las variables de entorno definidas en el sistema operativo para nuestro programa; y en cuyos valores se almacena el contenido de las mismas.

OPERADORES.- Los operadores son elementos del lenguaje que permiten realizar operaciones con los valores de las variables. De acuerdo al tipo de datos (numéricos, lógicos, carácter) a manipular le corresponde un operador especifico.

- **OPERADOR DE ASIGNACIÓN:**

Asigna un valor a la variable que se encuentra a su izquierda. Es único para números y cadenas. Se representa con el signo (=).

```
$a = 3;
@b = ("uno", "dos", "tres");
```

O bien, hacer asignaciones en cadena: $a = $b = $c = 3;

- **OPERADORES ARITMÉTICOS PARA NÚMEROS:** Realizan operaciones aritméticas con los valores implicados

EXPRESIÓN	OPERADOR	EJEMPLO
+	Suma	$total = 5 + 3;
-	Resta	$total = 5 - 3;
*	Multiplicación	$total = 5 * 3;
/	División	$total = 5 / 3;
**	Exponenciación	$total = 5 ** 3;
%	Modulo de división (resta)	$total = 5 % 3;
+=	Suma y asignación	$x += 5; equivale a $x = $x + 5;
-=	Resta y asignación	$x -= 5; equivale a $x = $x - 5;
**=	Expon. Y asignación	$x ** 2; equivale a $x = $x **2;
/	División y asignación	$x /= 2; equivale a $x = $x / 2;
%=	Modulo y asignación	$x %= 8; equivale a $x = $x %8;
++	Autoincremento	$x ++; equivale a $x = $x + 1;
--	Autodecremento	$x -- ; equivale a $x = $x - 1;

- **OPERADORES ARITMÉTICOS PARA CADENAS:** Sirven para unir cadenas de caracteres.

EXPRESIÓN	OPERADOR	EJEMPLO
.	Suma	$z = "ho" . "la";
.=	Suma y asignación	$z = "ho"; $z .= "la";
X=n	Concatenación	Hace n copias dentro de la variable original. $z = "ab"; $z X=3; #$z ="ababab"

- **OPERADORES PARA COMPARAR NÚMEROS**: Estos permiten comparar dos valores numéricos. Esta comparación devuelve siempre el valor de uno (cierto) cuando es verdad lo que comparan, y cero (falso) cuando no lo es.

EXPRESIÓN	OPERADOR	EJEMPLO
>	Mayor que	if ($a > $b) {print "$a es mayor\n"; }
<	Menor que	if ($a < $b) {print "$b es mayor\n"; }
==	Igual	if ($a == $b) {print "son iguales\n"; }
>=	Mayor o igual	if ($a >= $b) {print "$b no es mayor\n"; }
<=	Menor o igual	if ($a <= $b) {print "$a no es mayor\n"; }
¡=	Distinto	if ($a ¡= $b) {print "no son iguales\n"; }
<=>	Comparación	Devuelve –1, 0 o 1 si la expresión izquierda es menor, igual o mayor que la de la derecha respectivamente.

- **OPERADORES PARA COMPARAR CADENAS:** Permiten comparar dos cadenas, estableciendo un valor distinto de 0 cuando es cierta, y 0 cuando es falsa. Para determinar cual es mayor o menor se analizan los caracteres que las forman como si se tratara de ordenarlas alfabéticamente.

EXPRESIÓN	OPERADOR	EJEMPLO
Gt	Mayor que	if ($a gt $b) {print "$a es mayor\n"; }
Lt	Menor que	if ($a lt $b) {print "$b es mayor\n"; }
Eq	Igual	if ($a eq $b) {print "son iguales\n"; }
Ge	Mayor o igual	if ($a ge $b) {print "$b no es mayor\n"; }
Le	Menor o igual	if ($a le $b) {print "$a no es mayor\n"; }
Ne	Distinto	if ($a ne $b) {print "no son iguales\n"; }
Cmp	Comparación	Devuelve –1, 0 o 1 si la expresión izquierda es menor, igual o mayor que la de la derecha respectivamente.

- **OPERADORES LÓGICOS:** Los operadores lógicos combinan o modifican expresiones lógicas. Una expresión podrá ser cierta o falsa. Una expresión es cierta si tiene un valor conocido y éste es distinto de 0. Por el contrario es falsa si su valor es cero o no definido.

EXPRESIÓN	OPERADOR	EJEMPLO
&&	Y (AND)	if ($a > $b) $$ ($a > 0) {...}
||	O (OR)	if ($a > $b) \|\| ($a > 0) {...}
¡	NO (NOT)	if ($a > $b) {...}

B) Funciones, instrucciones y subrutinas

En la programación de perl el uso de las estructuras de control (por ejemplo, las estructura if – else, while, for) como en otros lenguajes, ayudan a dirigir el flujo de acciones a ejecutarse sobre los datos que maneja un programa determinado a través de instrucciones especificas. De igual forma, la existencia de funciones de perl permite ahorrar recursos de programación, puesto que definen un proceso bien definido para el tratamiento de los datos devolviendo un valor que ayudara en la ejecución de un programa.
Para ejemplificar se analizara este programa.

```
# contando

$fichero = 'contador.dat';
open (CONTADOR, ">$fichero") || die "Error: no se puede abrir el archivo";
for ($cuenta = 1; $cuenta < 21; $cuenta++) {
        print CONTADOR "$cuenta\n";
}
close (CONTADOR);
open (CONTADOR, "$fichero" ) || die "Error: no se puede abrir el archivo";
while ($linea = <CONTADOR>) {
        print "$linea";
}
close (CONTADOR);
```

Este programa devuelve como resultado una lista de números del 1 al 20 en la pantalla.

En la primer línea se asigna a la variable fichero el nombre del archivo a manipularse.
$fichero = 'contador.dat';

Dentro del programa se observa la función **open**, con ella se establece comunicación entre el programa y un archivo externo permitiendo leer sus datos o grabar información nueva.
La sintaxis de open es:

Open (MANIPULADOR, EXPRESIÓN)
MANIPULADOR: Es la palabra, escrita en mayúsculas, que utiliza el programa para hacer referencia al archivo abierto. EXPRESIÓN: Corresponde al nombre del archivo. Puede ir precedida por los siguientes signos: ">" , ">>" , "<" , que indican si el archivo se abre para escribir, añadir o leer datos, respectivamente.

La línea: *open (CONTADOR, ">$fichero") || die "Error: no se puede abrir el archivo";* se encarga de abrir el archivo: contador.dat y de asociarlo al manipulador CONTADOR. Al ejecutarse devuelve un valor de uno si consigue abrir el archivo o de lo contrario devuelve un cero. La instrucción (die "Error: no se puede abrir el archivo";) se ejecuta cuando ocurre algún error en la función open, deteniendo la ejecución del programa y enviando el mensaje hacia la salida de error estándar.

El signo ">" colocado antes de la variable que contiene el nombre del archivo indica que la apertura se realiza en modo de escritura.

Los tres modos posibles para abrir un archivo son:

Lectura.

Permite leer los datos de un archivo, marcara error si se intenta escribir en él. Si no se indica nada delante del nombre del archivo abre en modo de lectura

```
Open ( DATOS, "salida.dat");
```

Otra forma de indicarlo es antepomiendo el signo "<".
Si el archivo no existe se produce un error.

Inserción.

Permite leer y escribir sobre el archivo. Los datos que se escriban en el archivo se añadirán a continuación de los que ya existen. Si el archivo no existe previamente, se creará.

Para ello se emplea el signo ">>".

```
open (DATOS, ">>salida.dat");
```

Escritura.

Permite leer y escribir sobre el archivo. Los datos que hubieran previamente se perderán. Si el archivo no existe se creará. Utilizando el signo ">".

```
Open (DATOS, ">salida.dat");
```

Particularmente, la sentencia print en perl a parte de mostrar los datos en pantalla, también puede enviar los datos hacia otros dispositivos (archivos, impresora, pantalla).

En la línea: *print CONTADOR "$cuenta\n";* print envía los datos de la variable *$cuenta* hacia el archivo referido en el manipulador CONTADOR.

La sintaxis de print es:

Print MANIPULADOR ARGUMENTOS;
MANIPULADOR. Representa al dispositivo que recibirá los datos. Si se omite este parametro perl entiende que Debe enviarlo a la salida estandar (pantalla). ARGUMENTOS. Es una cadena, o lista de cadenas de caracteres separadas por una coma, que se enviarán al dispositivo de salida especificado.

Otra instrucción es: *close (CONTADOR);*
Con **close** se cierra el archivo referido por el manipulador CONTADOR que previamente se ha abierto con la instrucción open.
Si hay datos pendientes de leer o escribir almacenados en la memoria (buffer)se enviaran inmediatamente.

La sintaxis de close es:

Close MANIPULADOR;

La siguiente instrucción

```
while ($linea = <CONTADOR>) {
        print "$linea";
}
```

En la condición del while: *($linea = <CONTADOR>)*
se observa la expresión <CONTADOR>, la cual encierra el nombre del manipulador entre < >. Con esta expresión es posible leer una línea del archivo asociado a CONTADOR.
La sentencia de la condición *while* devuelve una línea (hasta el carácter \n) y la almacena en la variable *$linea*, repitiendo el ciclo hasta alcanzar el fin de archivo Esta condición tomara el valor 1 o cierto si ha podido leer la línea, y 0 o falso en el caso contrario terminando la ejecución del ciclo *while*.

A gracias a que perl mantiene para cada archivo un puntero que indica cual es la última línea leída, nunca se lee la misma línea del archivo.

Con *print "$linea";* se imprimen una a una todas las líneas del archivo.

SUBRUTINAS.- Una subrutina es propiamente una función, en un sentido estricto, una función es un proceso con entradas y salidas bien definidas. Su implantación debe ir encaminada a la realización de bloques de programas bien definidos en relación con el proceso que realicen.

Una función es un programa que recibe una serie de datos a través de una lista y devuelve determinada información de un tipo específico. A la lista de datos que la función recibe se le conoce con el nombre de lista de parámetros.[1]

El manejo de subrutinas dentro de un programa Perl es relativamente sencillo. Para definir una subrutina basta con escribir la palabra "sub" seguida del nombre que se le asignara a la subrutina y a continuación encerrar entre llaves "{ }" las instrucciones que la conforman.

```
Sub nombre_subrutina { .. . instrucciones. . . }
```

Las subrutinas pueden aparecer en cualquier parte del listado de un programa, porque sólo se ejecutan cuando se les llama de forma explícita.

Como ya se menciono toda subrutina devuelve datos, y pueden ser llamadas sólo con su nombre precedido del signo "&", o igualándolo a una variable del tipo de datos esperado.

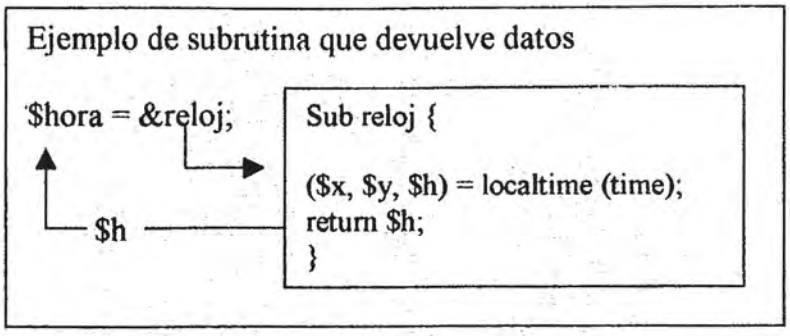
Esto es, una subrutina podrá ser llamada de las siguientes formas:

```
&nombre_subrutina;
$datos = &nombre_subrutina;
@datos = &nombre_subrutina;
%datos = &nombre_subrutina;
```

Para lograr que una subrutina devuelva datos al programa principal basta con incluir la siguiente sentencia dentro de la subrutina: **return datos;**

[1] Peñaloza R. Ernesto, fundamentos de programación, ed apuntes UNAM, p.128.

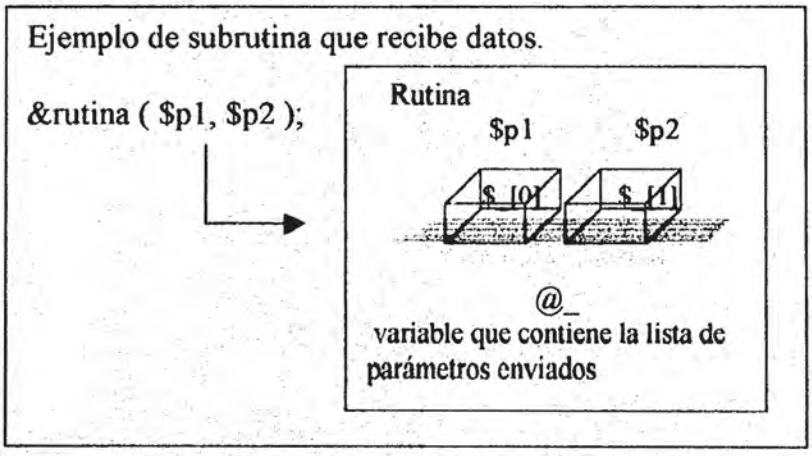
Con return finaliza la ejecución de la subrutina y devuelve el tipo de datos especificado en la variable asociada cuando se llama a ésta.



Así como es posible recibir datos desde una subrutina, también se puede enviarle datos cuando se invoca. Para enviar datos o parámetros a una subrutina se escribe a continuación del nombre de ésta los parámetros a enviar

```
&nom_subrutina ($entrada );
```

Con esta línea se llama a una subrutina y se le pasa el contenido de la variable $entrada.



C) Operaciones con cadenas y con listas.

Operaciones con cadenas.- Una de las ventajas importantes de perl es que permite llevar acabo procesos con cadenas de texto en una forma fácil y sencilla. Dentro de las operaciones que se pueden realizar con las cadenas son:

Búsqueda de modelos.
Búsqueda y sustitución.
Traslación de caracteres

♦ Búsqueda de modelos

Una forma de trabajar cadenas es a través de la búsqueda de modelos, para comprender mejor las ventajas de aplicar esta operación se utilizara el siguiente ejemplo.

Supóngase que se tiene el archivo citas.txt con las siguientes líneas y se requiere mostrar en pantalla las líneas que contengan la cadena: "la "

Archivo: citas.txt

A los veinte años, reina la voluntad; a los treinta, el ingenio; a los cuarenta el juicio. Ben.F
Todo el mundo aspira a la vida dichosa, pero nadie sabe en qué consiste. Séneca
Nunca somos tan felices ni tan desdichados como nosotros creemos. La Rochefoucault.
La e ulación es más peligrosa que el odio. Gracian
Todas las obras de arte deben empezar por el final. E. Allan Poe
La ociosidad es madre de todos los vicios. Proverbio latino.
Nuestras vidas son abreviadas por nuestra ignorancia. Spencer

Para buscar la cadena "la" se ocupara el siguiente programa.

```
#busca la subcadena "la".

$archivo = 'citas.txt';
open (CITA, "$archivo");
while ($cita = <CITA> ) {
    if ($cita =~ /la/ ){
            print "$cita";
    }
}
close CITA;
```

Las primeras líneas se encargan de abrir el archivo citas.txt en modo lectura.
A continuación el ciclo while lee una línea en cada vuelta y la almacena en la variable $cita.

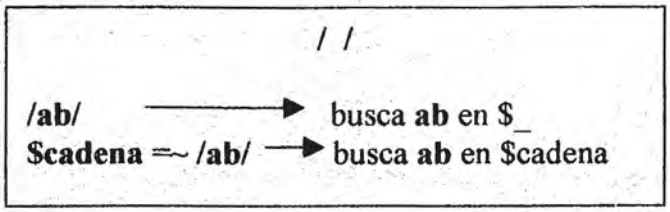
la parte nueva son las siguientes líneas:

```
if ($cita =~ /la/ ){
        print "$cita";
```

como se observa *print* imprimirá la línea leída sí se cumple la condición delimitada entre paréntesis ($cita =~ /la/).

En donde la pareja de operadores / / delimita una subcadena para ser localizada dentro de una cadena.

El operador =~ une la cadena en la que se va a buscar con la función de búsqueda / /.



En la siguiente tabla se muestra el alcance de los intervalos [ ] para llevar acabo una búsqueda dentro de una cadena.

[0-9]	Cualquier dígito.
[A-Z]	Cualquier letra mayúscula.
[A-Z][a-z]	Cualquier combinación de mayúscula y minúscula
[A-Za-z0-9]	Cualquier letra o dígito.
[^A-Z]	Cualquier carácter, excepto las letras mayúsculas.
[A-ZÑ]	Cualquier mayúscula, incluida la Ñ.

- **Búsqueda y sustitución**

El operador **s/ / /** sirve para reemplazar una subcadena por otra. De este modo será posible cambiar la subcadena "la" por "##", en el archivo citas.txt del ejemplo anterior

```
#modificando cadenas.

$archivo = 'citas.txt';
open (CITA, "$archivo");
while ( <CITA> ) {
        s/la/##/;
        print;
   }
close CITA;
```

El operador *s/ / /* busca en la variable reservada **$_** la subcadena (la) delimitada por la primera pareja de barras, cuando la encuentra la sustituye por la subcadena (##) de las dos últimas barras.

Aunque se emplea la variable reservada $_ también es posible sustituir el contenido de cualquier variable.

S/ / /

S/ab/ba/ ⟶ cambia **ab** por **ba** en **$_**

$cadena =~ s/ab/ba/ cambia **ab** por **ba** en **$cadena**

De momento no se ha especificado sobre que hacer cuando encuentre una subcadena en minúscula o en mayúscula, para ello existen modificadores como: **i, g, e**.

El modificador "i" hará que se ignoren las diferencias entre mayúsculas y minúsculas. Mientras que el **modificador "g"** sustituirá todas las ocurrencias que encuentre.

Así, la línea *s/la/##/*, podrá cambiar por esta otra línea *s/la/##/gi;*

Con los modificadores la capacidad para manipular cadenas crece enormemente, generando diversas posibilidades.
Por ejemplo, sustituir la aparición de dos vocales seguidas por "&&".

```
S/[aeiou][aeiou]/&&/g;
```

O bien, cambiar cualquier dígito por una "x".

```
S/[0-9]/x/g;        o lo que es lo mismo   s/\d/x/g;
```

"\d", representa un carácter reservado al que perl le da un significado especial. Siempre que se emplea un carácter reservado con la función específica que perl le atribuye se debe anteponer una barra inclinada (\) para indicar que se usa con el significado particular que tiene y no como el carácter que es.

\r	Carácter de salto de línea (código ASCII 13)
\n	Nueva línea (en Unix ASCII 10, en DOS ASCII 13+ ASCII 10)
\t	Tabulación
\f	Carácter de salto de línea (ASCII 12)
\d	Cualquier dígito [0-9]
\D	Cualquier carácter que no sea dígito (igual a [^0-9]
\w	Cualquier letra, número o subrayado (igual a [a-zA-Z0-9]
\W	Todos los que no sean \w
\s	Cualquier espacio (espacio, tabulador, salto de línea o salto de pagina).
\S	Todos los caracteres que no sean \s.

.	Un carácter cualquiera, excepto salto de línea
?	Uno o más como el carácter anterior
+	Ninguno o uno como el carácter anterior
*****	Ninguno o más como el carácter anterior

La expresión s/ / / podrá definir un intervalo de caracteres en las primeras barras pero, en la segunda debe contener inequívocamente los caracteres que servirán de reemplazo.

Una instrucción como esta seria incorrecta *s/[0-9]/[0-9]/g;*

La alternativa que ofrece perl es trabajar con variables reservadas que se generan de forma automática **($1, $2, ... $9)** donde es posible almacenar las últimas nueve subexpresiones encontradas durante la búsqueda en una cadena..
Así, la línea **s/ ([A-Z]) / ($1) /g;** define una subexpresión en el primer parámetro */ ([A-Z]) /* encerrando entre paréntesis lo que se desea forme parte de la subexpresión, en este caso es cualquier mayúscula. Cada vez que perl encuentra el primer parámetro almacena la subexpresión en la variable reservada **$1.**
En el segundo parámetro */ ($1) /* se indica que se va a cambiar lo encontrado en la subexpresión almacenada en la variable $1, por la mayúscula encontrada entre paréntesis

También es de gran utilidad el **modificador e**, puesto que realiza una evaluación del segundo parámetro del operador s/ / / como si se tratará de cualquier sentencia perl. Tal como se observa en el programa ejemplo.

```
$_ = '3+2 = total';
s/ total / 3+2/e;
print;
```

La salida del programa devuelve el siguiente resultado 3+2 = 5.

♦ Traslación de caracteres

La operación de traslación de caracteres utiliza la instrucción **tr/ / /**, cuya función es similar a la instrucción s/ / /. Ambas se emplean para reemplazar elementos dentro de una cadena. Pero propiamente, tr reemplaza caracteres y adicionalmente puede devolver el número de caracteres reemplazados, mientras que s/ / / es capaz de sustituir subcadenas de mayor longitud.

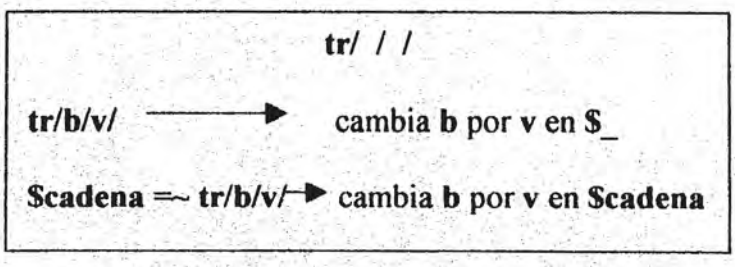
Para tr, el primer parámetro se considera una lista de caracteres que debe reemplazar y no como una subcadena. El segundo parámetro igualmente es otra lista que contienen los elementos por los que sustituirá de forma correlativa a los encontrados.

Véase los siguientes programas para analizar como opera tr.

```
#traslación de caracteres.
$archivo = 'citas.txt';
open (CITA, "$archivo");
while ( <CITA> ) {
        tr/ba /va_/;
        print;
  }
close CITA;
```

Este programa cambia las "b" por "v", las "a" por "A" y los espacios en blanco por "_".

```
#numero de caracteres cambiados.

$_ = 'tigre de hijo pintito';
$num_cambios = tr/ /_/;
print "$_\n$num_cambios";
```



Pack. Es una instrucción que resulta conveniente para transformar datos de un tipo a otro. La sintaxis es:

```
$resultado = pack (plantilla, argumentos);
```

En la plantilla se define el tipo de conversión que debe realizar. Esta plantilla se presenta como una cadena de caracteres que varían de acuerdo al tipo de conversión. La cadena que forma la plantilla contienen tantos caracteres como valores tenga en la lista de argumentos. Los argumentos son una lista de valores decimales o alfanuméricos que representan los datos a convertir.
Cada carácter empleado en la plantilla define un tipo de conversión que devolverá como salida. Véase la siguiente tabla.

Carácter	Tipo de dato de salida
A	Cadena de caracteres. Los huecos no asignados se emplean con espacios.
a	Cadena de caracteres. Los huecos no asignados se emplean con caracteres nulos (ASCII 0).
b	Cadena de bits en orden ascendente
B	Cadena de bits en orden descendente
c	No de código ASCII con signo (rango –127 a 127)
C	No de código ASCII sin signo (rango 0 a 255)
s	Número corto con signo
S	Número corto sin signo
i	Número entero con signo
I	Número entero sin signo
I	Número largo con signo
L	Número largo sin signo
f	Número de coma flotante
F	Número de coma flotante, doble precisión.
p	Puntero a una cadena terminada en carácter nulo.
P	Puntero a una estructura.

Algunos ejemplos son:

```
$a =pack ("A20" , hola );      #$a contiene una cadena con la palabra hola y 16 espacios
$a =pack ("CCCC" , 104, 111, 108, 97 );  #$a devuelve hola
$dec = hex (41);      #hex convierte a decimal el valor hexadeximal encerrado (41).
$letra = pack ("C", $dec);      # $letra contiene "a" (ascii 65 decimal ).
```

OPERACIONES CON LISTAS.-

En perl se distinguen dos tipos de listas: listas de variables escalares y listas asociativas. Existen instrucciones de bastante utilidad para el trabajo con listas, las cuales se verán a continuación

Foreach.- Es una instrucción interesante que realiza una ciclo desde el primero hasta el último elemento de una lista.

La sintaxis es:

```
Foreach $elemento_leido (nombre_de_la_lista ){
        Sentencias perl
    }
```

$elemento_leido

variable que contiene, en cada vuelta, el elemento leído desde la lista. Es un argumento opcional, al omitirlo el elemento leído se almacena en $_.

Nombre_de_la_lista

Nombre de la lista que se desea recorrer

Véase como ejemplo el siguiente programa.

```
#El uso de foreach

@colores = ("rojo", "azul", "verde", "morado", "gris");
foreach (@colores){
    print;
}
```

Otra función es **keys**, la cual devuelve las claves de una lista asociativa.

La sintaxis es:

```
@lista_claves = keys(%lista_asociativa)
```

Ejemplo:

```
#El uso de foreach y keys

%formula = ("sal", "ClNa", "agua", "H2O", "oxigeno", "O2" );
@claves = keys(%formula);
foreach (@claves){
    print;
}
```

En este programa se combina el uso de foreach y keys, que permite leer todas las claves de la lista asociativa *%formula*, para después imprimirlas.

Con los recursos de perl, fácilmente las dos instrucciones vistas (foreach y keys)se pueden integrar en una sola construcción sintáctica obteniendo la siguiente línea.

```
Foreach $clave (keys (%formula) ){
        Print "clave: $clave – valor: $formula{$clave}\n";
}
```

spl:.- Otra instrucción interesante es split, cuya función consiste en trocear una cadena en subcadenas basándose en un divisor y devolver datos distintos de acuerdo al contexto escalar o de lista a la que se integra. Si se asocia split a una variable escalar devolverá el número de subcadenas que se generan al trocear la cadena.

```
$trozos = split ( /divisor /, $cadena);
```

Cuando se asocia a una lista devuelve una lista con las subcadenas generadas al dividir la cadena.

```
@trozos = split ( /divisor /, $cadena);
```

Para agregar elementos a una lista asociativa, que como ya se ha visto asocia claves a sus elementos y no números de orden como las lista escalares, sólo se necesita especificar tal cual los elementos a agregar.

```
#añade elementos a una lista asociativa.

%formula = ("sal", "ClNa", "oxigeno", "O2" );
$formula {hidrogeno} = "H";
```

delete.- Es la instrucción que permite borrar elementos de una lista asociativa.

La sintaxis es:

$borrado = delete expresión;
$borrado. Valor del elemento eliminado. Esta asignación se puede omitir. Expresión. Define al elemento que se desea eliminar. La clave puede ser explícita o referirse con una variable.

En el siguiente programa se define una lista asociativa con 2 elementos, se añade un nuevo elemento y, finalmente se borran todos.

```
%formula = ("sal", "ClNa", "oxigeno", "O2" );
$formula {hidrogeno} = "H";
foreach $clave (keys (%formula)) {
        delete $formula {$clave};
}
```

read.- Sirve para leer datos en trozos de un tamaño específico. Con esta sentencia no es necesario leer hasta encontrar un salto de línea.

La sintaxis es:

Read (MANIPULADOR, $variable, tamaño);
MANIPULADOR. Palabra asociada al archivo a través de una instrucción open. $variable. Variable escalar que guardará el fragmento de datos leídos desde el archivo asociado a MANIPULADOR. Tamaño. Número de caracteres (bytes) que se leerá desde el archivo asociado a MANIPULADOR.

CAPITULO V
LA APLICACIÓN: AUTOMATIZACIÓN DEL PROCESO DE EVALUACIÓN ACADÉMICA DE LA ASIGNATURA DE GRAFICACIÓN POR COMPUTADORA.

Este capitulo presenta una herramienta de trabajo dirigida a la docencia de la asignatura de Graficación por computadora que se imparte en la ENEP Aragón. El uso de esta herramienta permite automatizar la evaluación del aprendizaje de esta asignatura, la cual es posible hacer llegar hasta ellos gracias al recurso del CGI que provee la tecnología del Web.
Considerando que la evaluación del aprendizaje forma parte del proceso de la evaluación académica; entonces esta herramienta permite al docente realizar el diseño y calificado de los exámenes de manera optimizada. Los resultados de esta evaluación conforman un elemento más de valoración que se integrara al proceso de la evaluación académica, al determinar por medio de exámenes el grado de conocimientos adquiridos por el alumno, es decir de su aprendizaje.

1. La problemática de la evaluación del aprendizaje y la alternativa del CGI.

La universidad resulta un lugar idóneo para que hombre y tecnología convivan. Cuando el hombre hace uso de las distintas herramientas que le provee la tecnología, consigue realizar su trabajo de una manera mucho más fácil y concentrarse en alcanzar resultados eficientes.
Este es el caso que se presenta en la comunidad estudiantil de la ENEP Aragón dentro del grupo donde se imparte la asignatura de Graficación por Computadora, cuya actividad escolar originó la necesidad de buscar opciones que involucren el uso de los recursos tecnológicos actuales que mejoren específicamente, las actividades de la evaluación del aprendizaje y de la enseñanza, que son parte del proceso para la evaluación académica[1] de la asignatura de Graficación.

[1] La evaluación es un proceso dinámico que comprende la investigación continua de la enseñanza y del aprendizaje. La enseñanza se evalúa cuando se investiga las aptitudes de los alumnos, las destrezas, las habilidades, las adquisiciones de conocimientos, .. etc. También se realiza una evaluación para determinar los objetivos educacionales, los planes y programas de la organización para juzgar y valorar todas las fases de la tarea escolar. (Spencer Guidice).

La situación actual para el diseño y evaluación de los exámenes de la asignatura involucra un proceso en el que intervienen las siguientes actividades:

Diseñar un examen de opciones
Aplicar el examen al grupo de alumnos.
Calificar todos y cada uno de los exámenes aplicados.
Registrar los resultados de los exámenes calificados
Informar a los alumnos las calificaciones que obtuvieron en el examen.

Para la realización de estas actividades es necesario disponer del tiempo adecuado para la ejecución de cada una, donde el profesor se encarga de la mayoría de las actividades haciendo un gasto de energía, trabajo y sobretodo tiempo. Por otro lado, los alumnos que son parte principal en este proceso, se ven obligados por estos factores a formar parte de una dinámica que muchas veces resulta bastante lenta.

Es evidente que los factores de trabajo y tiempo es necesario disminuirlos al mínimo para generar una proceso dinámico, donde ambas partes profesor y alumno, se beneficien renovando y mejorando el desempeño de la enseñanza y aprendizaje.

Así, el propósito final de esta investigación, es proponer el uso de los programas CGI para minimizar o eliminar todos los factores inherentes al proceso de evaluación de exámenes que se presenta en la asignatura de graficación. Como se ha visto a lo largo de la investigación, el CGI es un recurso tecnológico que a través del Web, medio de comunicación versátil que proporciona Internet, representa una alternativa de mejora en la practica de exámenes y evaluaciones de cualquier asignatura.

Por lo tanto, la problemática a solucionar se resume en automatizar la evaluación de exámenes de la asignatura de Graficación por Computadora con la ayuda de los script CGI.

2. Definición del proceso automatizado para la evaluación de exámenes de Graficación por Computadora.

Una vez identificadas claramente cada una de las actividades que involucran un proceso de evaluación de exámenes, es posible establecer un sistema automatizado de este proceso. Este sistema constituye una nueva herramienta que se ha nombrado: "SISTEMA AUTO-EVALUACION DE GRAFICACION POR COMPUTADORA".

Los principales objetivos de este sistema son los siguientes: generar el examen correspondiente de Graficación para ser aplicado vía Internet, proporcionar automáticamente los resultados de la evaluación de dicho examen y llevar un registro de los resultados de la evaluación del examen.

Con este sistema se pretende que el profesor elabore de una manera fácil y dinámica cada unos de los exámenes que se habrán de aplicar. Para ello, el primer paso a considerar es el diseño y elaboración de las preguntas o cuestionarios referentes a los temas que se presentan en el temario de la asignatura. Estas preguntas integraran el examen correspondiente al tema sobre el que se evaluaran los conocimientos de los alumnos.

Con la ayuda de un documento Web interactivo será posible crear las preguntas que el profesor convenga para formar un catálogo de preguntas, que estará disponible cada vez que se genere un nuevo examen. El catálogo será un documento Web que a partir del cual el profesor podrá elegir las posibles preguntas ha incluirse en un examen.
Una vez elegidas las preguntas, se generará un nuevo examen que estará listo para aplicarse vía Internet. El examen es otro documento Web interactivo por medio del cual los alumnos responderán a las preguntas mostradas y obtendrán de forma inmediata su calificación.

Los formularios HTML son una alternativa que permite generar documentos Web interactivos. Un formulario representa una buena opción para la interacción y para la manipulación de información a través de los cuadros de texto, botones de radio, botones simples y lista desplegable entre otros; éstos facilitan la realización de documentos de selección, además, permite que el diseñador realice su propio diseño como cualquier documento HTML. Por ello, el uso de formularios HTML proporciona claras ventajas para el diseño de un examen de selección múltiple, de relación de columnas, hasta de preguntas abiertas, tal como una hoja normal de examen de clase, o para el diseño de un catálogo que opera bajo el mismo principio de selección.

Finalmente lo que logra una interacción completa, es la existencia de scripts o programas CGI que generen en la marcha los documentos Web con los que interactuarán tanto profesor como alumnos.

Según lo expuesto, los puntos a desarrollar son las siguientes:

1. Diseño del formulario HTML que permita ingresar nuevas preguntas al catálogo de GPC (Graficación Por Computadora).

2. Desarrollo del programa CGI que permita la creación y actualización del catálogo de GPC

3. Desarrollo del programa CGI que permita la modificación del catálogo ya existente, eliminado preguntas que ya no son necesarias.

4. Desarrollo del programa CGI que permita visualizar el catálogo de preguntas para realizar la selección de las preguntas que integraran un nuevo examen.

5. Desarrollo del programa CGI que sea capaz de generar un nuevo examen a partir de las preguntas elegidas del catálogo.

6. Desarrollo del script o programa CGI que califique el examen generado que habrán de contestar los alumnos enviando como salida la calificación resultante y realice un registro de los resultados de cada alumno.

Cada uno de estos puntos representan los elementos que ayudaran a conformar e integrar al nuevo Sistema Auto-Evaluación de Graficación Por Computadora.

Los requerimientos para el desarrollo de cada uno de los programas listados son:

1. Editor de textos.
2. Lenguaje de programación (interprete perl para Windows)
3. Editor de paginas Web
4. Sistema operativo (Windows 95 / 98).
5. Conexión con Internet.
6. Espacio en el disco del servidor donde alojar los documentos Web y programas CGI.
7. Programa FTP
8. Navegador (Internet Explorer)
9. programa para la edición de imágenes y conversión de formatos gráficos.

En la siguiente sección de este capitulo se desarrollaran cada uno de los puntos que integraran el Sistema Auto – Evaluación de Graficación por Computadora.

3. Diseño y desarrollo del Sistema Auto - Evaluación de Graficación por Computadora.

- Diseño del formulario HTML que permita ingresar nuevas preguntas al catálogo de GPC (Graficación Por Computadora).

En este punto se diseña un documento Web con un formulario HTML que le permita al profesor componer y colocar las preguntas elaboradas dentro de un catálogo. Siempre que se pretenda ingresar nuevas preguntas se hará uso de este documento de manera que se actualice el catálogo de preguntas. Las preguntas ya elaboradas se podrán consultar en el catálogo actualizado.

En el formulario de este documento será posible introducir las preguntas y las posibles respuestas a estas, incluyendo la respuesta correcta que se especificara en un recuadro, esto con el fin de utilizarse en procesos posteriores que tendrán que ver con el calificado de los exámenes.

Para el diseño del documento Web que incluirá el formulario se utilizara el editor de paginas Web: WEBSCOPE97, el navegador Internet Explorer 4.0 y el editor de imágenes: PaintShop 3.11.

El formulario HTML se diseña dentro de las líneas del documento Web: **crea_catalgo.** La etiqueta correspondiente para la creación del formulario es:

<form method="POST"
action="http://tigre.aragon.unam.mx/cgi-bin/exagra/crea_cata.pl">

A través de esta etiqueta se establece el vinculo de este formulario con el programa CGI con el que interactuara y los respectivos parámetros o cláusulas que permiten llamar al programa que procesara los datos que envíe el formulario.

La cláusula: method=POST, define la forma en la que se enviara los datos de los campos del formulario hacia el servidor, estos datos se almacenan en la variable de entorno CONTENT_LENGHT del servidor.
El valor POST de la cláusula method, se guarda en la variable de entorno REQUEST_METHOD del servidor, así cuando el programa consulta esta variable, conocerá la forma del envío de los datos y buscara en las variables correspondientes los datos enviados.

La cláusula action, define el nombre y la ubicación del programa al que deben enviarse los datos del formulario.

Otras etiquetas importantes que ayudan a definir los campos que integraran el formulario son.

Elabore la pregunta No. 1<input type="Text" name="p_1" size="60">

Esta es una etiqueta de campo que define un cuadro de texto, que obviamente permite introducir texto, y con el atributo name es posible identificar el campo y la información asociada a este.
Como se recordara la información que se captura a través del formulario viaja hacia el servidor conformada en pares de clave =valor, donde la clave corresponde al valor asignado a name (p_1) y el valor será el texto introducido en el campo.
Mientras que el atributo size, especifica la extensión del cuadro de texto.

En este campo el profesor introduce la pregunta que desea ingresar al catalogo.

Los siguientes campos también son cuadros de texto donde se colocaran las cinco posibles respuestas a la pregunta elaborada.

Escriba las posibles respuestas a la pregunta elaborada
a)<input type="text" name="rap1" size="55">
b)<input type="text" name="rbp1" size="55">
c)<input type="text" name="rcp1" size="55">
d)<input type="text" name="rdp1" size="55">
e)<input type="text" name="rep1" size="55">

Un campo de particular interés es el botón de radio, puesto que con este es posible realizar una selección de entre uno o varios datos, asignando al grupo de datos un mismo valor clave para name. Mientras que con el atributo value, se asignan los distintos valores a seleccionar

Tal es el caso del campo donde el profesor elige de un grupo de opciones la respuesta correcta a la pregunta elaborada.

Marca la Respuesta
correcta
a)<input type="radio" name="solp1" value="sa">
b)<input type="radio" name="solp1" value="sb">
c)<input type="radio" name="solp1" value="sc">
d)<input type="radio" name="solp1" value="sd">
e)<input type="radio" name="solp1" value="se">

Otros botones que siempre están presentes en el diseño de un formulario son: submit y reset.

<input type="Submit" value="Enviar"><input type="Reset">

El botón submit permite el envío de los campos del formulario hacia el servidor Web, para que estén disponibles al programa encargado de su procesamiento.
El botón reset inicializa o limpia los valores de los campos del formulario, para ser llenados nuevamente en caso de requerirlo.

Finalmente toda etiqueta tiene su respectivo terminador y <form ...> no es la excepción, así su terminador al final del diseño del formulario es </form>.

En la siguiente figura se observa como se vería en el navegador Explorer el documento Web crea_catalgo.htm, que incluye el formulario HTML.

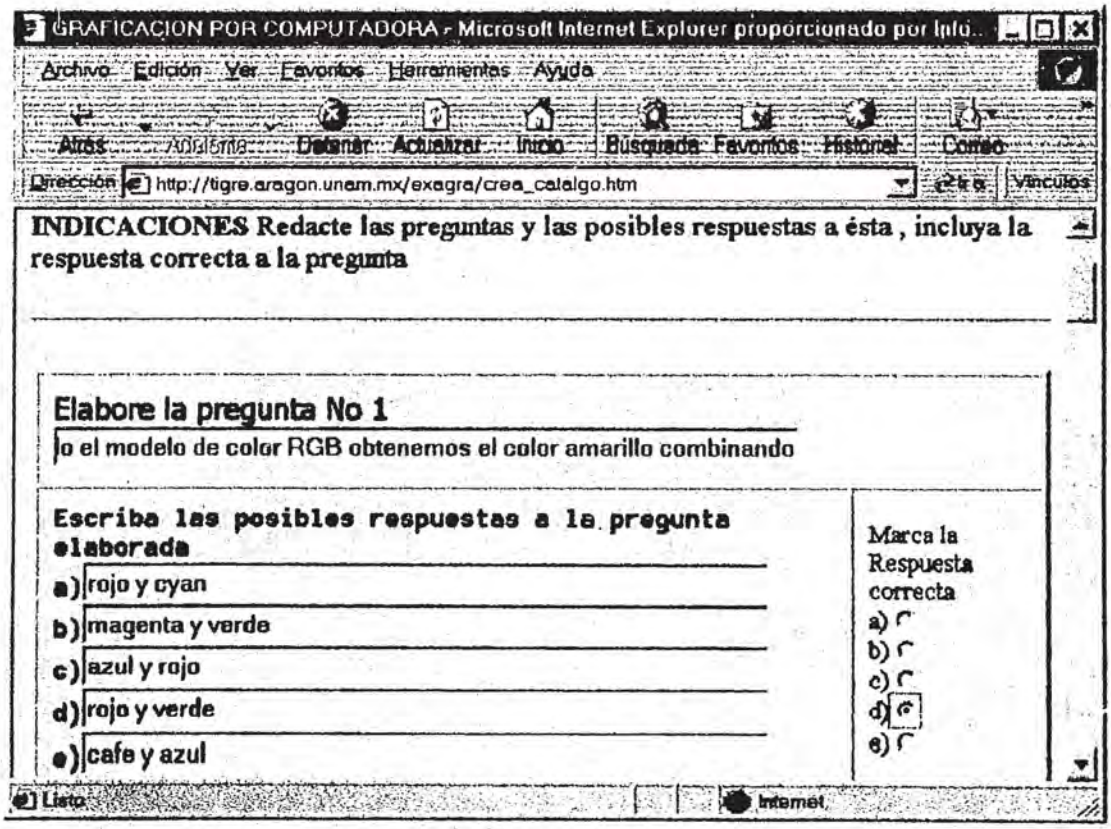


Figura 5.1. Documento Web que contiene el formulario HTML para agregar nuevas preguntas al catalogo de GPC.

- Desarrollo del programa CGI que permita la creación y actualización del catálogo de GPC

Para el desarrollo de los programas CGI que se realizaran en adelante se empleara el intérprete de perl V. 5 para Windows, y el editor de texto WordPad.

El código de cada un de los programas CGI se escribirán en el Wordpad tal como si fuera un simple archivo de texto, considerando que siempre se guardaran únicamente como archivos de textos.

El código principal del programa encargado de la creación y actualización del catálogo se escribió bajo el nombre de: **crea_cata.**

Este programa se encarga de procesar los datos enviados desde el formulario del documento Web (crea_catalgo.html), para ingresar nuevas preguntas al catalogo de Graficación, de tal modo que se actualice el archivo que contiene el registro de las preguntas que se han elaborado.

La primera línea del código especifica la ruta de ubicación dentro del servidor del interprete de perl encargado de ejecutar el programa, seguido de los signos "#!".

```
#! /usr/bin/perl
```

Este comentario especial es la instrucción que permite ejecutar al interprete Perl sobre el script o programa que contiene esta instrucción.

Las siguientes líneas recuperan información de las variables de entorno y almacenan en la variable $entrada los parámetros y datos que fueron enviados desde el formulario hacia el servidor.

```
if ($ENV{'REQUEST_METHOD'} eq "GET"){
        $entrada = $ENV{'QUERY_STRING'};
}elsif ($ENV{'REQUEST_METHOD'} eq "POST"){
   read(STDIN, $entrada, $ENV{'CONTENT_LENGTH'});
  }else {
     $entrada = $ARGV[0];
  }
```

La estructura if examina el contenido de la variable de entorno $ENV{'REQUEST_METHOD'}, para identificar el método utilizado para el envío de los parámetros. Si es por el método POST, los datos son leídos de la entrada estándar (STDIN) y la longitud de la cadena leída se almacena en $ENV{'CONTENT_LENGHT'}.

Si es por el método GET, la cadena se almacena en la variable $ENV{'QUERY_STRING'}.

El envío de parámetros desde la línea de comandos es una opción alterna al método POST o GET, y es útil especialmente al realizar pruebas desde una computadora personal a través de la línea de comandos de la ventana DOS. Los parámetros enviados se almacenan en la variable especial "$ARGV".

Una vez que se tiene la cadena de parámetros en la variable $entrada, se procede a decodificar dicha cadena, puesto que al viajar del navegador hacia el servidor los datos se codifican en una cadena bajo el esquema de codificación URL.
A través de este esquema se conforma una cadena a la que se le agregan símbolos como "&" para separar los pares clave – valor, que corresponden al nombre del campo y al valor o datos asociados a éste, respectivamente. Otros símbolos como "=", agrupan o relacionan las claves con sus valores respectivos. Por otro lado, se agrega el símbolo "%" junto con notaciones en números hexadecimales en aquellas palabras que llevan letras acentuadas, también, es común encontrar signos de "+" para marcar los espacios entre palabras.

Una línea de información codificada que viaja hasta el servidor Web se vería de la siguiente forma:

Nombre=fernanda&direccion=matamoros+no+219&telefon=55655906
&P_1=significado+de+las+siglas+FPS

Para decodificar la cadena recibida se hace uso del siguiente código.

```
foreach (split(/\&/,$entrada)){
    ($clave, $valor) = split(/=/,$_);
     $clave =&deco($clave);
     $valor =&deco($valor);
     $arreglo{$clave} = $valor;
    }

sub deco{
  $_ = shift;
  s/\+/ /g;
  s/%(..)/pack("C",hex($1))/ge;
  return($_);
}
```

Mediante el uso del bucle foreach combinada con la función split, permite separar la información que viaja en la cadena que se almaceno en la variable $entrada, extrayendo los datos que se introdujeron en los campos del formulario, guardando en las variables $clave y $valor los valores respectivos de la clave (nombre asignado al campo) y el valor (datos introducidos por el usuario).

Una vez asignados los datos a las variables se les aplica la función &deco para decodificar los datos en caso de que contengan símbolos "%" con números hexadecimales o signos "+". De esta manera la información o datos quedan entendibles para ser procesados posteriormente.

función decodificadora

```
sub deco{
  $_ = shift;
  s/\+/ /g;
  s/%(..)/pack("C",hex($1))/ge;
  return($_);
}
```

Dentro de la función sub deco se emplean operaciones de búsqueda y sustitución de cadenas, para realizar conversiones de ocurrencias de caracteres en código hexadecimal a código ASCII.
En la primer operación de búsqueda y sustitución se encuentran todos aquellos símbolos de + y se sustituyen por espacios.
En la segunda operación se localiza el carácter "%" seguido de dos cualesquiera para formar una subexpresión o subcadena: /%(..), que habrá de sustituirse por el carácter que tiene asignado en el código ASCII.

Hecho esto, la función devuelve los valores ya decodificados hacia las variables de donde se llama a la función.

```
$clave =&deco($clave);
$valor =&deco($valor);
```

El siguiente paso es crear una lista asociativa con los nuevos datos de las variables $clave y $valor.

```
$arreglo{$clave} = $valor;
```

De esta forma con la ayuda de la lista asociativa "$arreglo" es posible asociar la clave del campo con su respectivo valor, con el objeto de tener un fácil acceso a la información del formulario.

Lo siguiente es disponer de un archivo dat donde se guardaran las preguntas, junto con las posibles respuestas y la solución correcta a la pregunta. Esta información esta ordenada y disponible en la lista asociativa $arreglo. El archivo destinado para dicho fin es: preres.dat.
El nombre de este archivo se asigna a la variable: $fichero y para referirse a él se le asocia un manipulador con el nombre de "SALIDA", haciendo uso de la siguiente instrucción:

```
open(SALIDA, ">>$fichero") || die "error al abrir archivo";
```

Open es una instrucción que permite abrir el archivo para escribir en él, si existe un archivo previo las nuevas líneas se agregan al final de las ya existentes. Si no existe se creará el archivo.
El manejo del archivo en modo actualización permite ir agregando las nuevas preguntas elaboradas para conformar un catálogo con n preguntas.

Lo siguiente es realizar un recorrido por la lista asociativa para preparar los datos que serán ingresados al archivo preres.dat

```
$cuenta=10;
$con=0;

 while($con < $cuenta){
 $con=$con+1;
 $pre="p_$con";
  $checa= $arreglo{$pre};

 if( $checa =~ /[a-zA-Z0-9]/){

 $pre="p_$con";
 $resa="rap$con";
 $resb="rbp$con";
 $resc="rcp$con";
 $resd="rdp$con";
 $rese="rep$con";
 $solu="solp$con";

print SALIDA
"$arreglo{$pre}__$arreglo{$resa}_$arreglo{$resb}_$arreglo{$resc}_$arreglo{$resd}_
$arreglo{$rese}_\#$arreglo{$solu}\n";
}
}
close SALIDA;
```

La instrucción print, envía lo escrito en ésta hacia la salida estándar (monitor del navegador), pero si se especifica un manipulador de archivo lo escrito se envía hacia el archivo al que hace referencia.

La siguiente línea de print utiliza el manipulador SALIDA para hacer llegar la información hasta el archivo especificado. Los elementos de la lista asociativa que se marcan entre comillas, se envían agregando símbolos de separación para cada información proveniente de los campos del formulario.

Cuando ya se ha vaciado la información de los formularios en el archivo preres.dat que contendrá las preguntas que conforman el catálogo de preguntas de Graficación, será posible realizar la selección de las preguntas a incluirse en el nuevo examen a través del catalogo.

Enseguida, se envía un reporte de salida del proceso que realizo el programa CGI (crea_cata) hacia el navegador, para informar que se realizo satisfactoriamente el ingreso de preguntas al catálogo.
Cuando se ejecuta el programa en una computadora personal la salida estándar será la pantalla de la computadora en cuestión, pero cuando se ejecuta a través de un servidor Web, se emplea como salida estándar un canal que envía los datos hacia el servidor, quien a su vez los transmitirá por la red hacia la pantalla del navegador solicitante.

Cualquier instrucción print dentro del programa sin asociarle algún manipulador enviara el contenido especificado en ésta hasta el servidor Web. El servidor examina las directivas dirigidas a él y el resto lo lanza como respuesta al navegador.
Los programas que envían datos de salida hacia el navegador deben de iniciar la respuesta de salida con la línea: *content-type* informando al servidor el tipo de datos que debe remitir al navegador solicitante.

La primera línea que envía print como salida es: print "content-type: text/html \n\n" if

Donde, content-type es una cláusula dirigida hacia el servidor que le indica claramente del tipo de información que se enviara al navegador solicitante, cuando el tipo es text/html se genera una salida en formato HTML. Esta línea no se refleja en la pantalla del navegador, puesto que es una cláusula de uso único para el servidor.
Cuando ya se ha definido el tipo de datos a enviar al navegador, se debe enviar dos salto de línea: \n \n enseguidos para que la información que reciba a continuación no la trate como parte de esta cláusula y pueda enviar el resto de los contenidos de print tal cual hacia la pantalla del navegador.

Las líneas print que construyen la respuesta de salida incluyen etiquetas HTML para definir una salida que pueda ser visualizada en la pantalla del navegador.

```
print "<html>\n";
print "<head>\n";
print "<title>Catalogo de Graficación por Computadora</title>\n";
print "</head>\n";
print "<body bgcolor\=\"\#ffefd5\">\n";
print "<h2 align\=\"center\"><font face\=\"Comic Sans MS\">Resultados del ingreso de
preguntas al catalogo de Graficación por Computadora</font></h2>\n";
print "<hr>\n";
print "<br>\n";
..
..

print "<th bgcolor\=\"\#e9967a\"><font face\=\"Tahoma \">Las
preguntas  ingresaron al catálogo  satisfactoriamente</font></th>\n";
print "</tr>\n";
print "<tr>\n";
print "<td bgcolor\=\"\#ffdab9\" align\=\"center\"><font face\=\"lucida Sans\">Consulte
el catálogo de Graficación por Computadora para la
elaboración del examen correspondiente en el botón del menu principal<br><a
href\=\"/exagra/presenta.htm\">menu principal </a></font></td>\n";
print "</tr>\n";
print "</table>\n";
print "</center>\n";
print "</body>\n";
print "</html>\n";
```

La información que se reporta al navegador solicitante es el ingreso satisfactorio de las preguntas elaboradas al catálogo y de la dirección donde lo puede consultar el profesor para realizar la selección de preguntas que desea incluir en un nuevo examen.

En la siguiente figura se muestra como se observa en el navegador la respuesta de salida que genera el script (crea_cata.pl) en el proceso de actualización de catálogo de GPC.

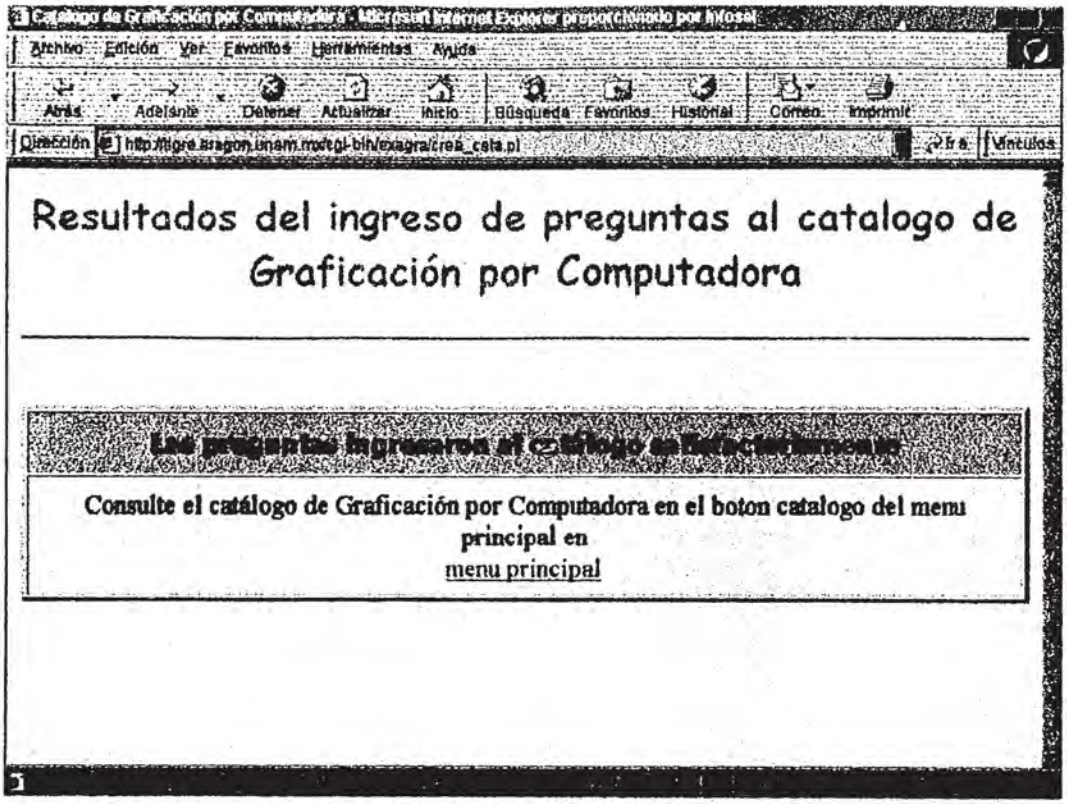


Figura 5.2. Pantalla de salida generada por el programa crea_cata que actualiza el catalogo de preguntas de GPC.

- Desarrollo del programa CGI que permita la modificación del catalogo ya existente, eliminado preguntas que ya no son necesarias.

Con este programa CGI se pretende eliminar preguntas del catalogo de GPC que ya no se incluirán en el diseño de un nuevo examen.

El programa muestra el catálogo de preguntas para elegir de éste aquellas preguntas que serán borradas, el programa encargado de esta tarea se nombra: **modif_cata.** Cuando ya han seleccionado las preguntas ha eliminar se envían a un programa CGI que se encargara de proceder a la actualización del archivo, borrando las preguntas elegidas.

La tarea principal de modif_cat es abrir el archivo preres.dat para sustraer todas las preguntas que actualmente existen en el archivo, para crear un documento Web en la marcha que muestra todas las preguntas en un formulario de selección de preguntas que se desean borrar.

Se abre el archivo preres.dat en modo de lectura para recorrer línea a línea el contenido de éste e identificar las pregunta y sus posibles respuestas. Para este caso resulta bastante practico que cada elemento de información del archivo contenga signos de separación para identificarlos claramente. Cada línea contenida en el archivo se conforma de la siguiente forma:
Pregunta__respuesta a_ respuesta b_ respuesta c_ respuesta d_ respuesta e_#solucion

El siguiente código extrae del archivo todas las preguntas que se encuentran registradas en el catálogo.

```
open(SALE, "<$fichero") || die "error al abrir el archivo";

while($linea = <SALE>){

 ($pgunta, $rpuesta) = split(/=/,$linea);
 $num=$.;
...
...
close SALE;
```

El bucle while devuelve cada línea del archivo y la almacena en la variable $linea, si Perl mantiene para cada archivo abierto un puntero que indica cual es la última línea leída, esto ocurre de manera automática y cada vez que se ejecuta la expresión:
$linea = < SALE> no se lee la misma línea, sino la siguiente a la última leída.

Concluido el proceso de extracción de las preguntas del catalogo se cierra el archivo dat que contiene las preguntas empleando la instrucción close. A través de close se rompe la relación manipulador ⇔ archivo que estableció open.

Todas las preguntas extraídas se muestra en un formulario. Para que se visualice este formulario en la pantalla del navegador se emplea la instrucción print, sin especificar ningún manipulador. A través de print se escribirán las etiquetas HTML que construirán el documento Web que contiene el formulario para la selección de preguntas a eliminar. Este código se vería de la siguiente manera

```
print "<body bgcolor\=\#ffefd5 text\=\#000000>\n";
print "<h2 align\=\"center\"><font

.
.....
print "<p><b>INDICACIONES</b> De este catalogo elija las preguntas que desea eliminar, seleccione las preguntas de la columna izquierda, \n";
print "<br><br>\n";
print "<hr>\n";
....
print "<form method\=\"POST\"
action\=\"http://tigre.aragon.unam.mx/cgi-bin/exagra/mod_catalo.pl\">\n";
print "<center>\n";
.
print "<td><i>ELIJA LAS PREGUNTAS A ELIMINAR</i></td><td><i>ESTAS SON LAS POSIBLES RESPUESTAS, </i></td>\n";
print "</tr>\n";
```

proceso de extracción de preguntas del archivo.

```
print "<input type\=\"Submit\" value\=\"Enviar cambios \"><input type\=\"Reset\">\n";
print "</form>\n";
```

Dentro del código que construye documento Web a través de print se emplean caracteres que deben tratarse con cierto cuidado y que son comunes de encontrar en las líneas de código HTML. Una forma de anular el significado especial que pudieran tener en la sintaxis de perl es anteponiendo una barra invertida (\) en aquellos caracteres que tienen un significado especial y que deben ser enviados como tales. Los caracteres que tienen un significado especial son: " , #, =. Tal como se observa en las siguientes líneas.

En la ejecución de este mismo programa se genera un archivo dat temporal que contiene una copia de las preguntas del archivo original (preres.dat) para utilizarse en el siguiente programa que actualizara del archivo de preguntas de Graficación.

La siguiente figura muestra el documento HTML que genera como salida el programa modif_cata, para la selección de preguntas a eliminar.

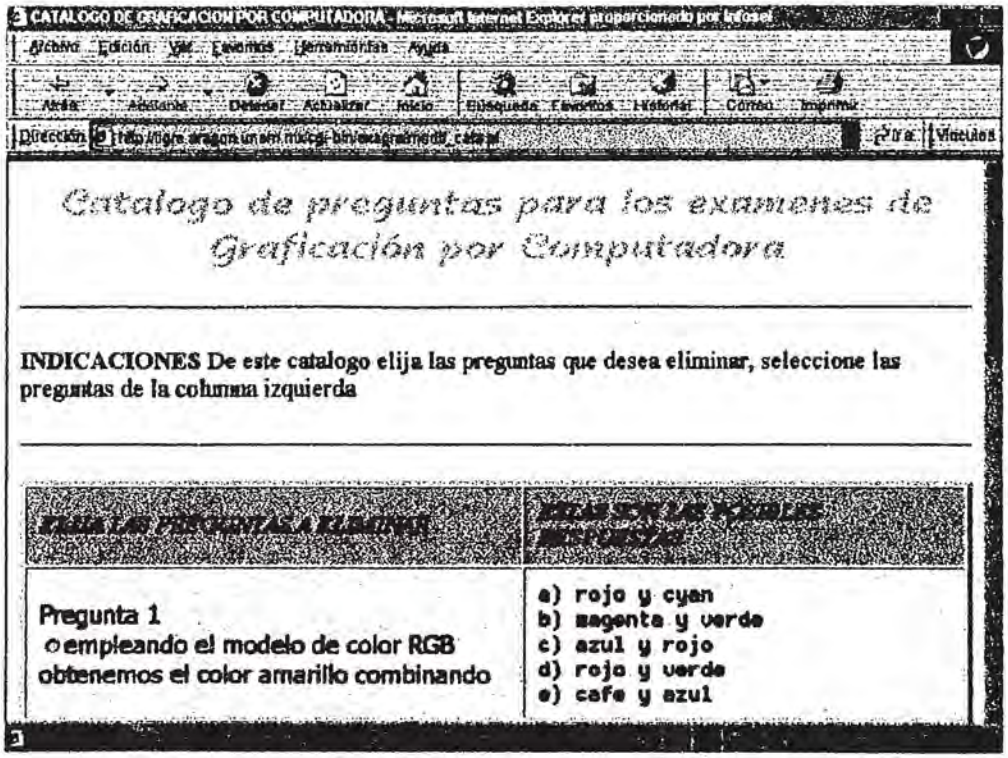


Figura 5.3. Documento Web que incluye un formulario generado por el programa modif_cata para eliminar preguntas del catalogo de GPC.

El formulario del documento Web que genera el programa modif_cata envía los datos que en éste se introducen y se ejecuta otro programa CGI cuyo nombre es: **mod_catalo.** Este programa se encarga específicamente de realizar la actualización del archivo que contiene las preguntas, eliminando todas aquellas que se seleccionaron en el formulario para ser borradas de forma definitiva del catalogo.

El código principal de este programa procesa la información proveniente de los campos del formulario, donde se señalo aquellas preguntas que se habrán de borrar.
La información de los campos del formulario comprende una clave donde se señala la pregunta a eliminar y el número de registro asociada a ésta, (pregunta-registro).

Esta clave se guarda en una lista asociativa para su manipulación. Con esto es posible identificar claramente dentro del archivo que guarda el registro de las preguntas del catalogo aquellas a borrarse.

En el archivo temporal (tempo.dat)que guarda una replica de las preguntas del catalogo(preres.dat) se identifica la pregunta-registro a borrar y se elimina, posteriormente, realiza una copia tal cual de este archivo temporal ya modificado y reemplaza el archivo preres.dat, que originalmente guarda la relación de preguntas del Catalogo de Graficación. Este proceso se realiza en las líneas siguientes.

```
open(SALE, ">$fiche") || die "error al abrir el archivo";
 open(ENTRA, "<$ficha") || die "error al abrir el archivo";
   while($linea = <ENTRA>){
    foreach $clves (keys(%arreg)){
     ($fijo, $numpr) = split(/_/,$clves);

        if ($. == $numpr){
        $linea=~ s/(.{25,})\n/-/;
          chop $linea;
           }
      }
          print SALE "$linea";
}
 close ENTRA;
 close SALE;
```

El programa mod_catalo genera como salida en el navegador información del proceso que ha realizado. En la sig. Figura se muestra como se visualizara en la pantalla del navegador.

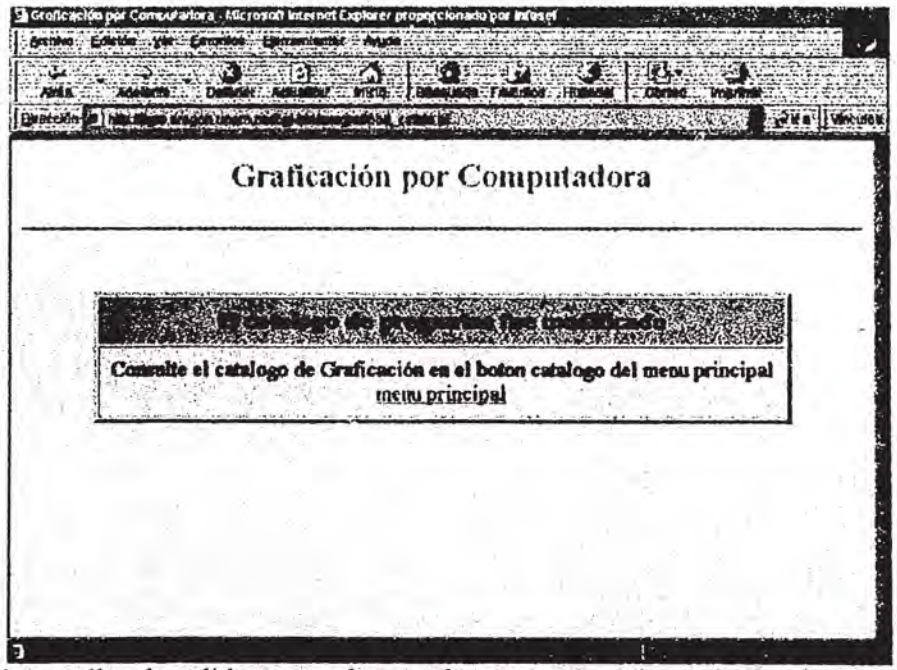


Figura 5.4 pantalla de salida generada por el programa mod_cata que actualiza el catálogo de preguntas de GPC.

- Desarrollo del programa CGI que permita visualizar el catálogo de preguntas para realizar la selección de las preguntas que integraran un nuevo examen.

Este programa se nombra: **ctalogo_gpc**, el código empleado ya se ha utilizado en procesos de programas anteriormente visto.

Específicamente este programa tiene la tarea de presentar en la pantalla del navegador el catálogo de Graficación con todas las preguntas que han sido diseñadas por el profesor.

Para ello se hace uso del archivo (preres.dat) que contiene las preguntas registradas. Este archivo se abre en modo de lectura para recorrerlo línea a línea y sustraer las preguntas; con las preguntas extraídas se construirá un formulario para la selección de preguntas que empleara el profesor para crear un nuevo examen.

El código principal es:

```
Print "<form method\=\"POST\"action\=\"http://tigre.aragon.unam.mx/cgi-bin/exagra/crea_exam.pl\">\n";
print "<center>\n";
print "<table bgcolor\=\#ffdab9 border\=3 cellpadding\=\"10\">\n";
print "<tr>\n";
print "<td><i>ELIJA LAS PREGUNTAS</i></td><td><i>ESTAS SON LAS POSIBLES RESPUESTAS</i></td>\n";
print "</tr>\n";

**proceso de extracción de las preguntas del archivo preres.dat**

print "<tr>\n";
print "<td><font size\=3><font face\=\"Tahoma\">Pregunta $num<br>\n";
print"<input type\=\"radio\" name\=\"pre_$num\" value\=\"pg$num\">$preg</font></font></td>\n";

  ($repa,$repb,$repc,$repd,$repe)= split(/_/,$resp);

  print "<td>\n";
  print "<font size\=3><font face\=\"Fixedsys\">\n";
  print "a) $repa<br>\n";
  print "b) $repb<br>\n";
  print "c) $repc<br>\n";
  print "d) $repd<br>\n";
  print "e) $repe<br>\n";
  print "</font></font></td>\n";
  print "</tr>\n";
```

```
print "<input type\=\"Submit\" value\=\"Enviar al nuevo examen\"><input type\=\"Reset\">\n";
print "</center>\n";
print "</form>\n";
```

A continuación se muestra en la figura como se visualizara el formulario del catalogo de preguntas en el navegador.

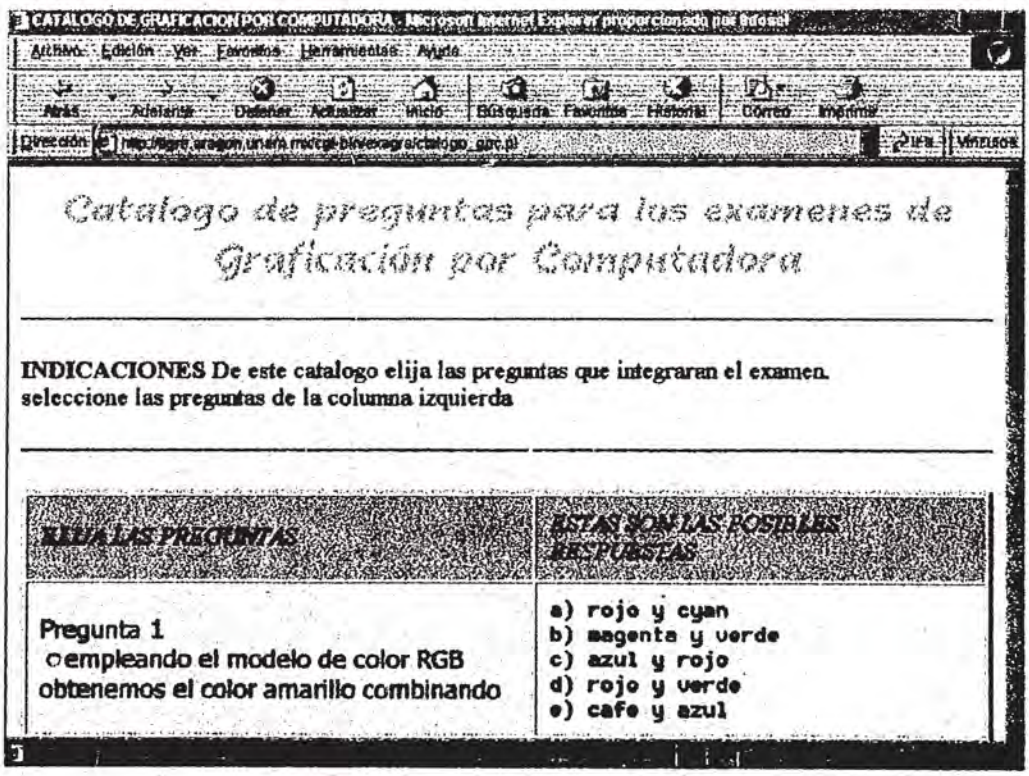


Figura 5.5. Salida generada por el programa ctalogo_gpc que muestra el catálogo de preguntas de GPC.

- Desarrollo del programa CGI que sea capaz de generar un nuevo examen a partir de las preguntas elegidas del catálogo.

Este programa se encarga de generar el nuevo examen de Graficación, al que se nombro: **crea_exam**.

Para crear el nuevo examen se baso en un diseño de examen de opción múltiple, por qué resulta más fácil una evaluación de preguntas cerradas, que permite una evaluación sin ambigüedad, que de preguntas abiertas, donde el criterio de evaluación no siempre se puede estandarizar para aplicarse en un proceso sistematizado.

El objetivo de este programa es recoger las preguntas seleccionadas del formulario del catalogo de preguntas de Graficación para conformar y generar el documento examen de Graficación.

El proceso para la obtención y depuración de los datos provenientes de los campos del formulario del catálogo de preguntas, se trabaja de la misma forma como se ha hecho en los programas anteriores. Este proceso se realiza en las primeras líneas del código del programa dejando listos los datos para ser utilizados en los procesos siguientes.

La primera línea informa al servidor la ubicación del interprete de perl que ejecutara el programa crea_exam. Esta instrucción siempre deberá colocarse al inicio de todo programa CGI sin excepción.

```
#! /usr/bin/perl
```

El siguiente proceso se encarga de generar el documento HTML (examen_gpc.html) que fungirá como el examen de Graficación que habrán de contestar los alumnos.
Previo a este paso, se establece la ruta o dirección que ocupara el nuevo archivo html, considerando que se pueda guardar tanto en la computadora personal para pruebas posteriores, como en el servidor Web donde se colocara finalmente.

```
$hubica = $ENV{REMOTE_ADDR};
$archdoc = 'examen_gpc.html';

if ($hubica){
  $ruta = '../../html/exagra/';
}else {
  $ruta = '..\pagina\\';
}
$exam = "$ruta$archdoc";
```

Con la instrucción open se abre el archivo (examen_gpc.html) a crearse para construir el nuevo documento HTML, el archivo se abre en modo de escritura y se le asocia el manipulador SALDOC, cuya ruta esta guardada en la variable $exam .

```
open(SALDOC, ">$exam") || die "error al abrir archivo";
```

Todo lo que se escriba dentro del archivo se hará mediante la instrucción print SALEDOC, donde se especifica entre comillas dobles, las etiquetas HTML que permitirán construir el nuevo examen de Graficación.

```
print SALDOC "<html>\n";
print SALDOC "<head>\n";
print SALDOC "<title>EXAMEN PARCIAL DE GRAFICACIÓN POR COMPUTADORA</title>\n";
print SALDOC "</head>\n";
print SALDOC "<body bgcolor\=\#ffefd5 text\=\#000000>\n";
print SALDOC "<h2 align\=\"center\"><font face\=\"Nadianne\"><font
color\=\#e9967a>Examen de Graficación por
Computadora</font></font></h2>\n"; print SALDOC "<hr><br>\n";
print SALDOC "<p><b>INDICACIONES</b> Inserte sus datos correctamente. Lea cuidadosamente las preguntas y marque su respuesta correcta\n";
print SALDOC "<br><hr>\n";
.....
```

Dentro del documento se define el formulario donde se habrán de introducir los datos del alumno como: nombre del alumno, numero de cuenta, además de las respectivas preguntas que habrán de ser contestas por los alumnos.

```
print SALDOC "<form method\=\"POST\"
action\=\"http://tigre.aragon.unam.mx/cgi-bin/exagra/califica.pl\">\n";
....
...
print SALDOC "<td>Inserte sus datos como se indica </td>\n";
....
print SALDOC "<td>Nombre: <input type\=\"Text\" name\=\"nombre\"
size\=\"30\"><br><br>\n";
print SALDOC "No.de cuenta: <input type\=\"Text\" name\=\"nocta\"
size\=\"10\"></td>\n";
.....
print SALDOC "<td>Lea las preguntas y seleccione la respuesta correcta </td>\n";
print SALDOC "</tr>\n";
...
```

Como ya se sabe, los formularios envían información codificada a través de pares de clave –valor, durante el proceso de obtención estos datos se decodificaron y se almacenaron para su fácil manipulación en una lista tipo asociativa que se nombro %arregl, donde se almacena la relación de números de registro de preguntas que el profesor desea incluir en el examen

La instrucción foreach permite recorrer cada uno de los elementos (clave-valor)que integrar la lista . Con la instrucción keys dentro de foreach es posible obtener los elementos claves, las cuales se asignan a la variable $pclav, para realizar un análisis de la información contenida. La función split analiza la cadena de información de $pclav para extraer el valor o número de registro asociado a la pregunta que se debe escribir en el documento examen.

```
.foreach $pclav (keys(%arregl)){

($fijo, $nupre) = split(/_/,$pclav);
.....
.....
```

El bucle while recorre las líneas del archivo para substraer las preguntas que correspondan con el número de registro asociado de la pregunta que se ha de mostrar en el nuevo examen.

```
open(SALE, "<$fiche") || die "error al abrir el archivo";
 while($linepre = <SALE>){
    last if $. == $nupre;
}
close SALE;
...
```

Una vez obtenida la línea que corresponde a la pregunta buscada, se almacena la información correspondiente a la pregunta en $prgun, y las posibles respuestas en $respus, para colocarse en el formulario del documento examen (examen_gpc.htm)

```
($prgun,$respus) = split(/=/,$linepre);
```

Cuando se concluye el proceso de elaboración del formulario para el examen, el programa envía a la pantalla del navegador una respuesta informando que se finalizo el proceso de elaboración de examen. Sin olvidar de informar previamente al servidor Web del contenido MIME con el que se enviara la salida al navegador.

```
print "content-type: text/html \n\n" if
    $ENV{REMOTE_ADDR};
```

Enseguida se procede a escribir las etiquetas html que construirán la respuesta de salida que incluye información referente a la creación del nuevo examen y de su ubicación, además, se incluye una liga al documento (examen_gpc.html) para un acceso rápido.

```
.....
print "<th bgcolor\=\"\#e9967a\"><font face\=\"lucida sans unicode\">El
examen de graficación por computadora esta listo!</font></th>\n";
print "</tr>\n";
.....

print "<tr>\n";
print "<td bgcolor\=\"\#ffdab9\" align\=\"center\"><font face\=\"lucida
console\">Consulte el examen de Graficación por Computadora en <br><a
href\=\"/exagra/examen_gpc.html\">examen
listo</a></font></td>\n";
```

En la siguiente figura se muestra en el navegador la respuesta de salida generada por el programa crea_exam, que informa sobre la creación del nuevo examen.

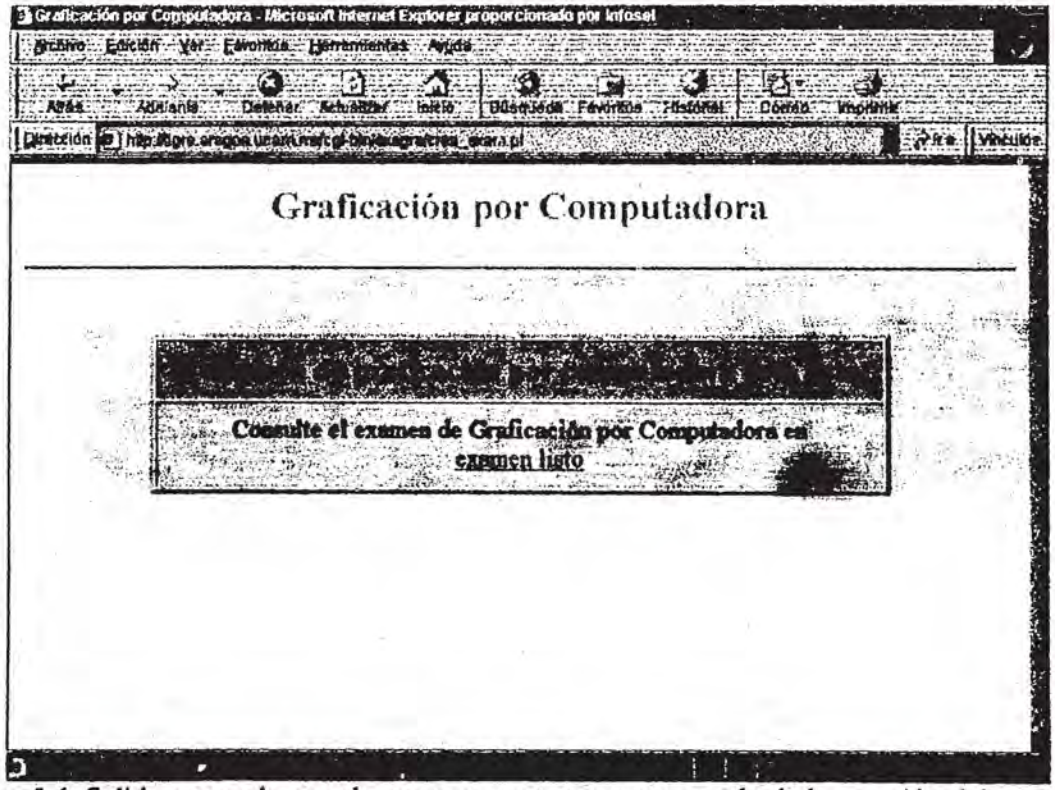


Figura 5.6. Salida generada por el programa crea_exam encargado de la creación del examen de GPC.

A continuación se muestra en la siguiente figura como luce el nuevo documento examen (examen_gpc.html), generado durante la ejecución del script crea_exam.

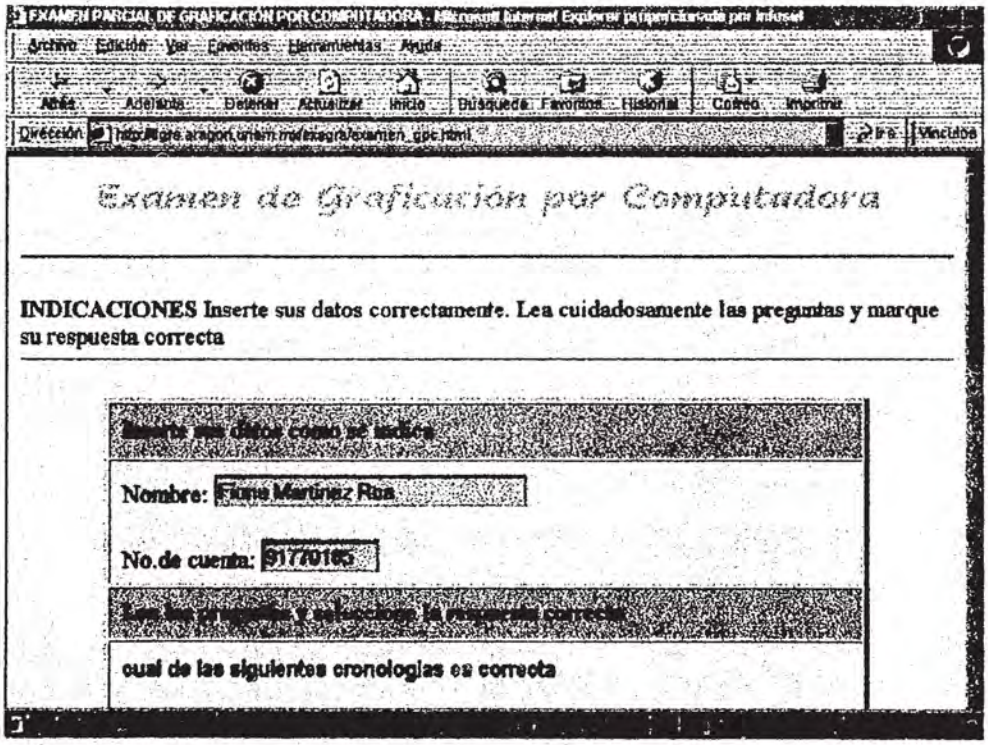


Figura 5.7. Pantalla que muestra el documento examen de GPC.

- Desarrollo del script o programa CGI que califique el examen generado que habrán de contestar los alumnos, enviando como salida la calificación resultante y realice un registro de los resultados de cada alumno.

El objetivo del siguiente programa es capturar y procesar las respuestas del alumno que ingresaran en el documento examen (examen_gpc.html) de manera, que se genere como respuesta de salida la calificación obtenida. De esta forma el alumno conocerá de inmediato la calificación que obtuvo.

El otro objetivo es llevar un registro de los alumnos que contesten el examen; el registro de cada alumno ingresara en un archivo dat e incluirá el nombre del alumno, su numero de cuenta y la calificación obtenida. El profesor dispondrá de este archivo en el que podrá consultar los resultados de la evaluación del examen aplicado.

A modo de exploración, se mostrara en la salida de resultados una relación en forma de lista de aquellas preguntas que se contestaron equivocadamente. La finalidad de esto es proporcionar información al alumno sobre los temas o tópicos sobre los cuales deberá poner especial atención para corregirlos.

A continuación se explica el programa encargado de este proceso al que se le nombró **califica**.

Se inicializarán las variables que se emplearán en el proceso de conteo para la evaluación de las respuestas enviadas por los alumnos a través del formulario. Además, se asigna a la variable $fichero el nombre del archivo dat donde se registraran los datos del alumno (nombre, número de cuenta, calificación obtenida).

```
$fichero = 'registro.dat';
$cali = 0;
$prom = 0;
```

Se extrae los datos depositados en las variables de entorno del servidor Web, que contienen las respuestas dadas por los alumnos en el formulario del examen (examen_gpc.html). Los datos obtenidos se guardan en una lista tipo asociativa para procesos posteriores, enseguida se les aplica la función &deco encargada de la decodificación de dicha información.

```
if ($ENV{'REQUEST_METHOD'} eq "GET"){
   $entrada = $ENV{'QUERY_STRING'};
  }elsif ($ENV{'REQUEST_METHOD'} eq "POST"){
   read(STDIN, $entrada, $ENV{'CONTENT_LENGTH'});
  }else {
     $entrada = $ARGV[0];
  }
```

```
foreach (split(/\&/,$entrada)){
    ($clave, $valor) = split(/=/,$_);
     $clave =&deco($clave);
     $valor =&deco($valor);
     $arreglo{$clave} = $valor;
    }

sub deco{
.
.

}
```

Cuando ya sean preparado y depurado los datos enviados desde el formulario, es posible identificar las respuestas de las preguntas que contestaron cada alumno.

Para entregar al alumno la calificación obtenida, se comparan las soluciones de las preguntas registradas en el catálogo, contra las respuestas que selecciono el alumno en el formulario del examen. El registro de preguntas se encuentra en el archivo preres.dat, el registro de cada pregunta contiene: la pregunta, las posibles respuestas y la *solución correcta*.

El siguiente código permite hacer el recorrido del archivo dat hasta encontrar el número de registro que corresponda a la pregunta del examen que se desea verificar, para extraer su solución correcta y guardarla en la lista $vector.
Con esta lista se podrá comparar las soluciones que proporciona el alumno.

```
open(SALE, "<$ficha") || die "error al abrir fichero";

  while($line = <SALE>){
     last if $. == $numer;
  }
 close SALE;

   ($pregus, $respes) = split(/\#/,$line);

   $nclav ="sol$numer";
    $vector{$nclav} = $respes;
}
}
```

Mediante las listas asociativas: $arreglo, que contiene las respuestas del alumno y $vector, que contiene las soluciones correctas, se realizara la comparación de los valores contenidos en ambas listas.

La comparación de respuesta – solución se lleva a cabo pregunta a pregunta comparando las cadenas contenidas en los campos valor de las listas. Como ya se ha mencionado, una lista asociativa se compone por pares clave – valor, donde la clave, corresponde al registro de la pregunta, y el valor, equivale a la respuesta asociada a ésta.

```
$sol=$vector{$provi};
$envia=$arreglo{$prov};
```

Por cada respuesta dada por el alumno que coincida con la solución correcta se establece una variable bandera para indicar el acierto, en caso de no coincidir se establece un falso. Esto es para llevar un conteo de respuestas correctas y determinar una calificación.

```
if($ban == $band){

    $cali = $cali+1;
   }
else{
   $premal{$prov}=$envia;
   }
}
}
```

Cuando se realiza la comparación de respuestas se agrega un acierto a la bandera, pero cuando se agrega un falso, significa que se identifico una respuesta errónea. Como otro de los objetivos es proporcionar información al alumno de las preguntas en las que se ha fallado, se almacena en una lista asociativa $premal, las claves de las preguntas mal contestadas.

A través de una regla de tres simple se obtiene la calificación final y se almacena como nuevo elemento $prom en la lista asociativa %arreglo.

```
$prom = (($cali*10)/$cuenta);
$arreglo{'promedio'} = $prom;
```

En este momento, ya se cuenta con la información necesaria para registrar en el archivo (registro.dat) la relación de alumnos que contestaron el examen. Los datos a registrarse son: nombre del alumno, numero de cuenta, fecha de registro y promedio.
Mediante print SALIDA, se escribirán los datos de cada alumno a registrar en el archivo (registro.dat).

```
open(SALIDA, ">>$fichero") || die "error al abrir fichero";
print SALIDA "$arreglo{'nombre'}_";
print SALIDA "$arreglo{'nocta'}_";
print SALIDA "$arreglo{'promedio'}_";
print SALIDA "$arreglo{'fecha'}_";
print SALIDA "\n";
close SALIDA;
```

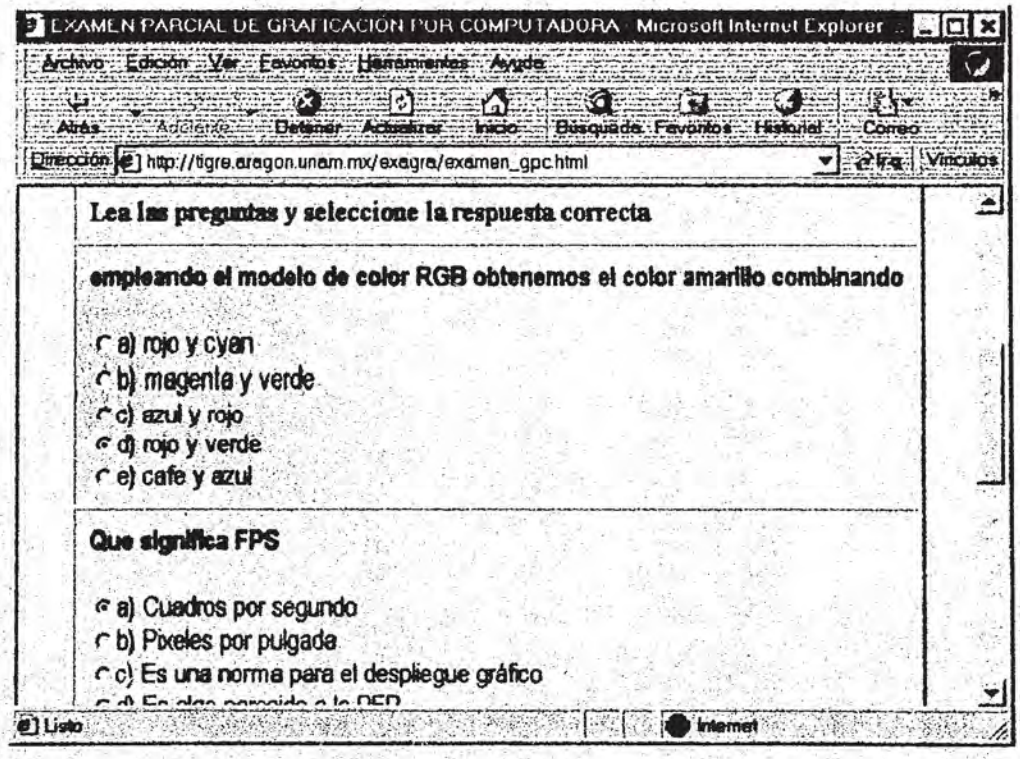
Desde la pantalla del menú principal aparece un icono que al pulsarlo presenta en pantalla la relación de alumnos que integran el archivo registro.dat con los resultados del examen aplicado.

La salida del programa enviara a la pantalla del navegador los resultados de la evaluación examen al alumno. Además, se incluye una lista de aquellas preguntas que se contestaron mal.

```
print "content-type: text/html \n\n" if
    $ENV{REMOTE_ADDR};
.....
print "<title>Resultados de examen</title>\n";
..
print "<td bgcolor\=\"\#ffdab9\" align\=\"center\"><font face\=\"Tahoma\">Tu
calificación <u>$arreglo{'nombre'}</u> es:$arreglo{'promedio'}</font></td>\n";
print "</tr>\n";

print "<font face\=\"Comic Sans MS\"><font color\=\#00008b>A continuación se
muestran las preguntas contestadas con respuestas equivocadas</font></font></h5>\n";
print "<br>\n";
.....
#(este código coloca las preguntas mal contestadas, recorriendo la relación guardada en
la lista %premal).

foreach $clas (keys(%premal)){
   ($fij, $nmero) = split(/_/, $clas);

  open(SAL, "<$ficha") || die "error al abrir fichero";
  while($lin = <SAL>){
     last if $. == $nmero;
  }
 close SAL;

   ($preg, $resp) = split(/__/,$lin);
   ($rpa,$rpb,$rpc, $rpd,$rpe) =split(/_/,$resp);
   $checa=$premal{$clas};
```

```
    if($checa =~ /a/){
     $pone=$rpa;
    }
    if($checa =~ /b/){
     $pone=$rpb;
    }
    if($checa =~ /c/){
     $pone=$rpc;
    }
   if($checa =~ /d/){
     $pone=$rpd;
    }
   if($checa =~ /e/){
     $pone=$rpe;
    }
.print " <tr>\n";
print "<td>$preg</td><td>$pone</td>\n";
print " </tr>\n";
}
.....
print "</body>\n";
print "</html>\n";
```

Figura 5.8. Pantalla que se muestra en el navegador con el documento examen de Graficación que el alumno deberá responder.



Enseguida se muestra la pantalla de resultado que genera como salida hacia el navegador el script califica, encargado de la evaluación del examen de GPC.

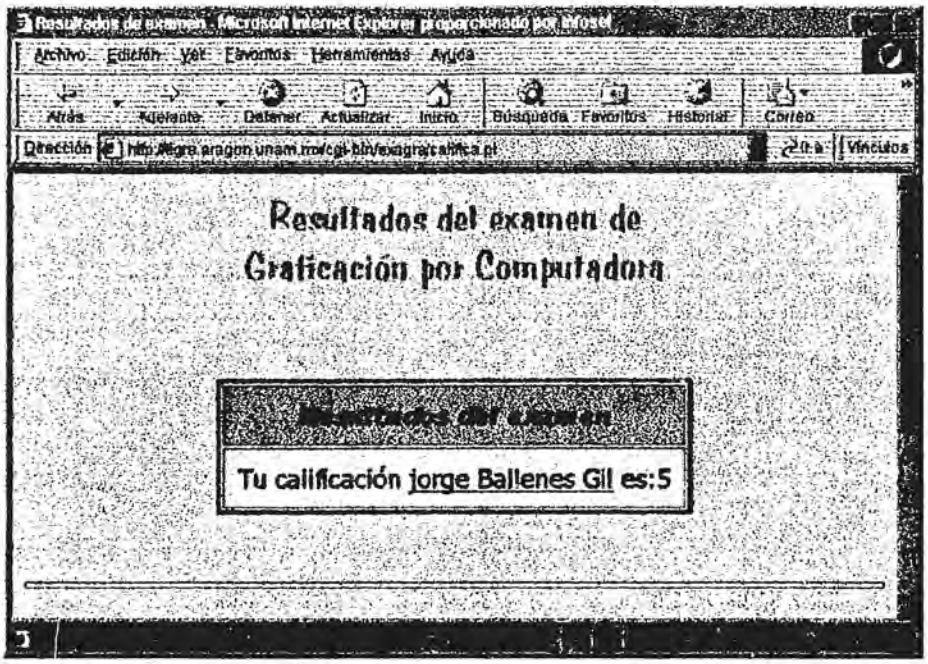
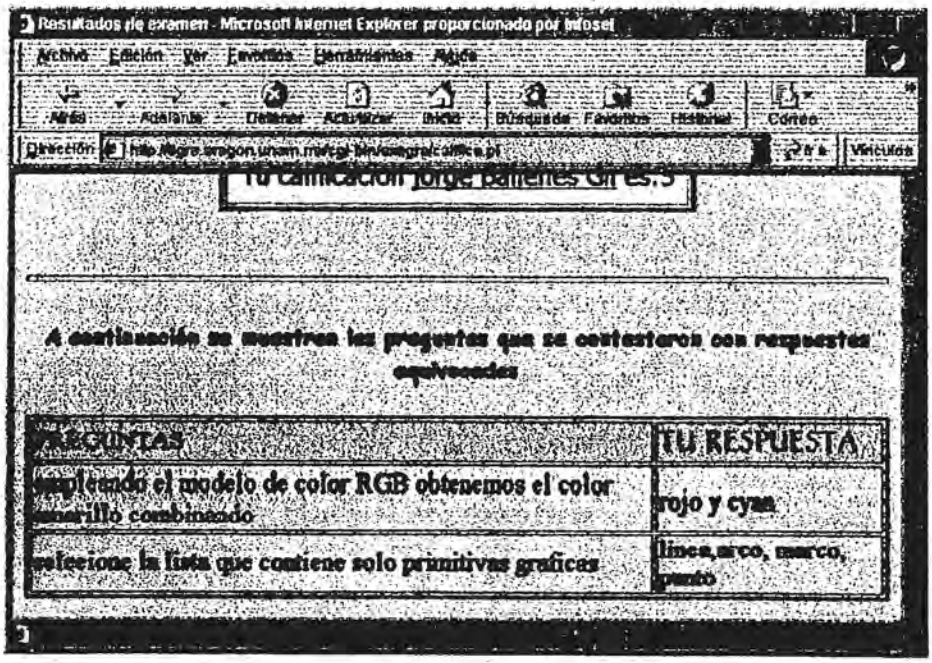


Figura 5.9. Pantalla de resultados que envía el programa califica.



Hasta este punto se ha concluido el desarrollo y diseño de los elementos integrales del sistema que incluyen los formularios, los programas o script CGI y los archivos dat.

Para entrar al sistema se ha diseñado la pagina Web principal de esta aplicación, que incluye una pagina de autenticación, donde el profesor habrá de teclear su clave de acceso para poder entrar al sistema. El código html que genera esta pagina se ha nombrado: **paswd.**

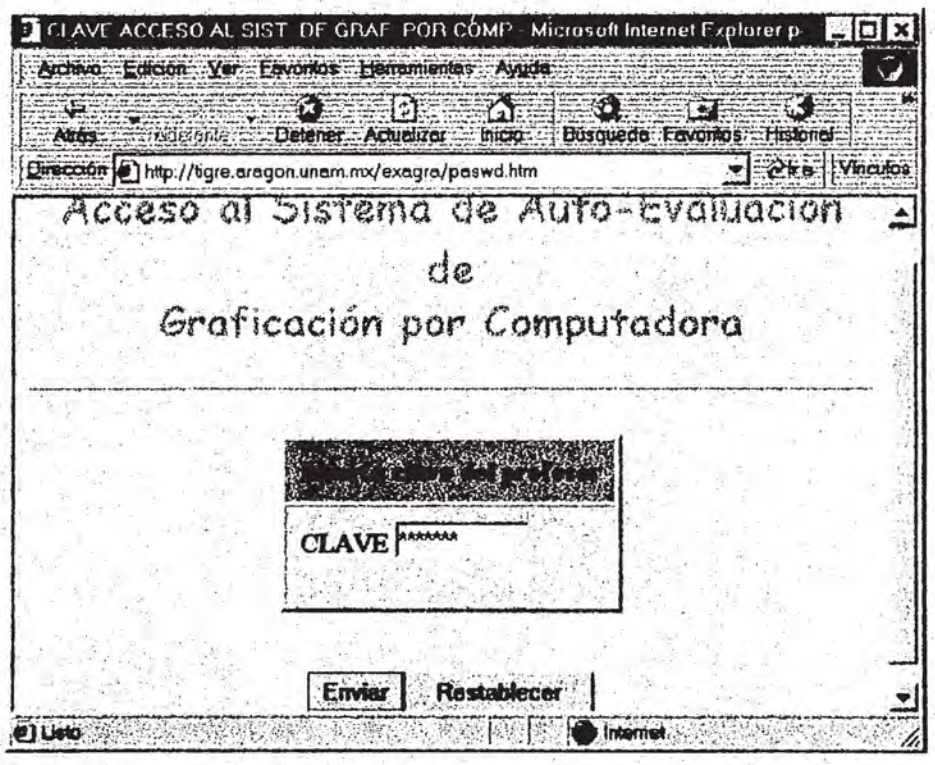


figura 5.10. Pantalla de la pagina de identificación visualizada en el navegador que permite el acceso al sistema

El programa CGI (**clave**) encargado de recibir la clave e identificarla, determina si se permite o no tener acceso al sistema.

Entrando al sistema aparece la pagina principal desde la cual se tiene acceso a un menú de opciones a través de botones que ejecutan cada una de las operaciones que realiza el sistema. Las opciones posibles son las siguientes:

- Un botón para la actualización del catálogo de preguntas de Graficación, desde el cual el profesor introducirá las preguntas a incluirse en el catálogo.

- Un botón para la modificación del catalogo de preguntas, que permite al profesor eliminar aquellas preguntas que considere innecesarias.

- Un botón para presentar el catálogo de preguntas de graficación actualizado, en donde le permite al profesor elegir aquellas preguntas que desea colocar en un nuevo examen.

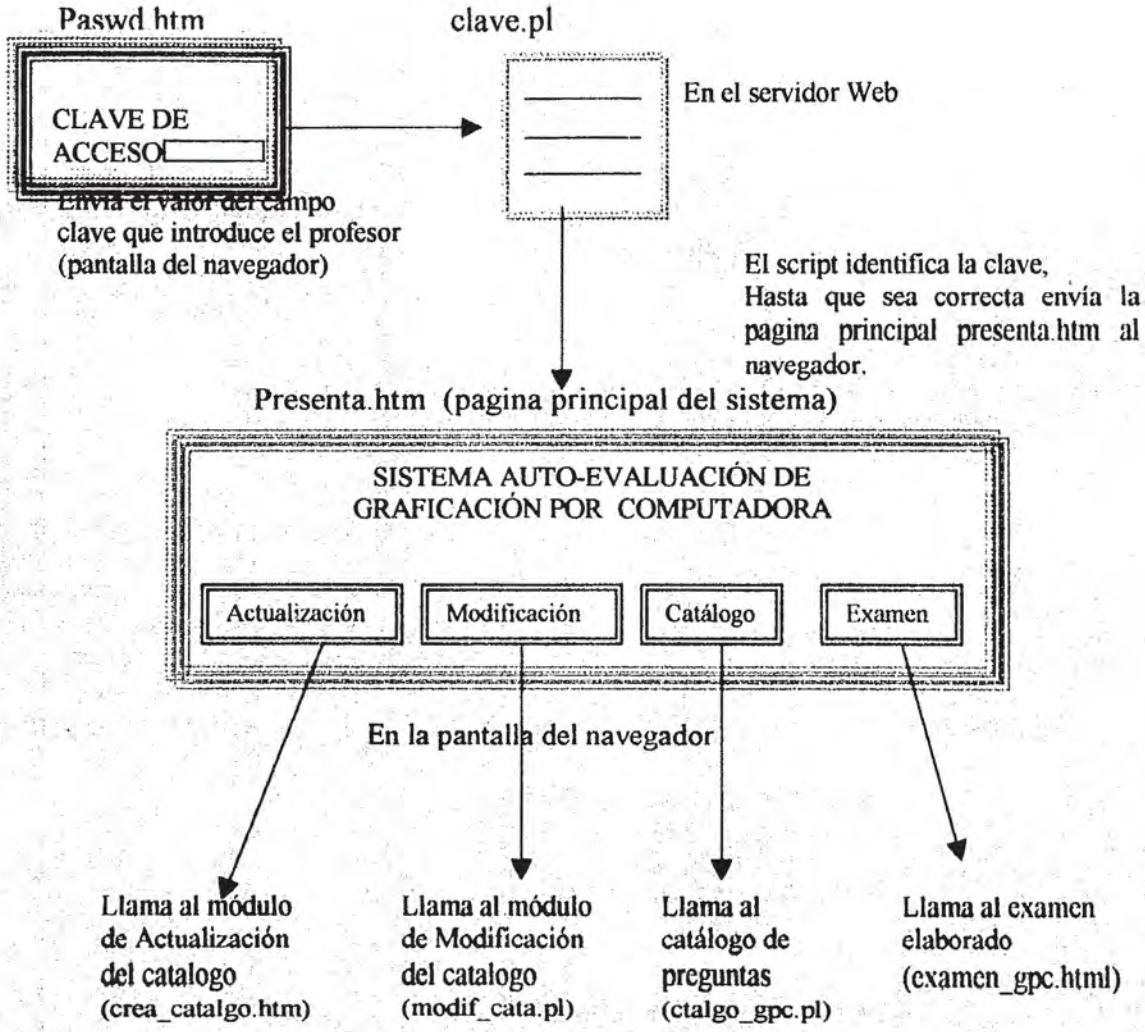
- Un botón que presenta el examen generado, el cual estará disponible para que los alumnos lo contesten

En el diseño de esta pagina principal se hace uso del programa PaintShop para la edición y conversión de formatos de imágenes, que ayudaran a darle una mejor presentación.

La pagina principal del sistema Auto – Evaluación de Graficación por Computadora se nombra: **presenta.**

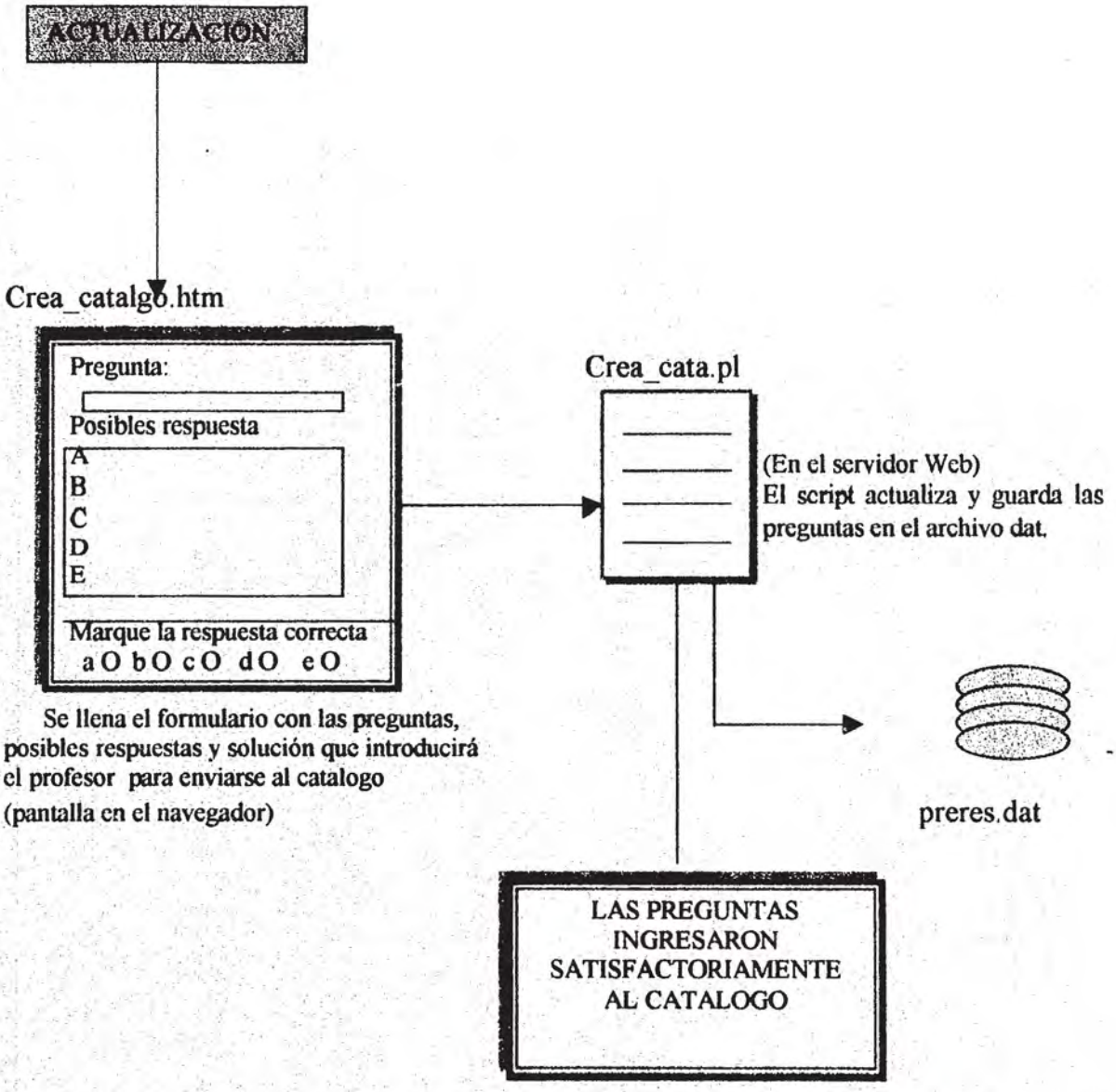
Figura 5.11. Pantalla principal del sistema Auto – evaluación de GPC.

En el siguiente esquema se podrá observar como interactuan cada uno de los programas que integran el sistema, que conjuntamente tiene como objetivo automatizar la evaluación de los exámenes de la asignatura de Graficación por computadora.



A continuación se observa a detalle cada uno de los módulos del sistema.

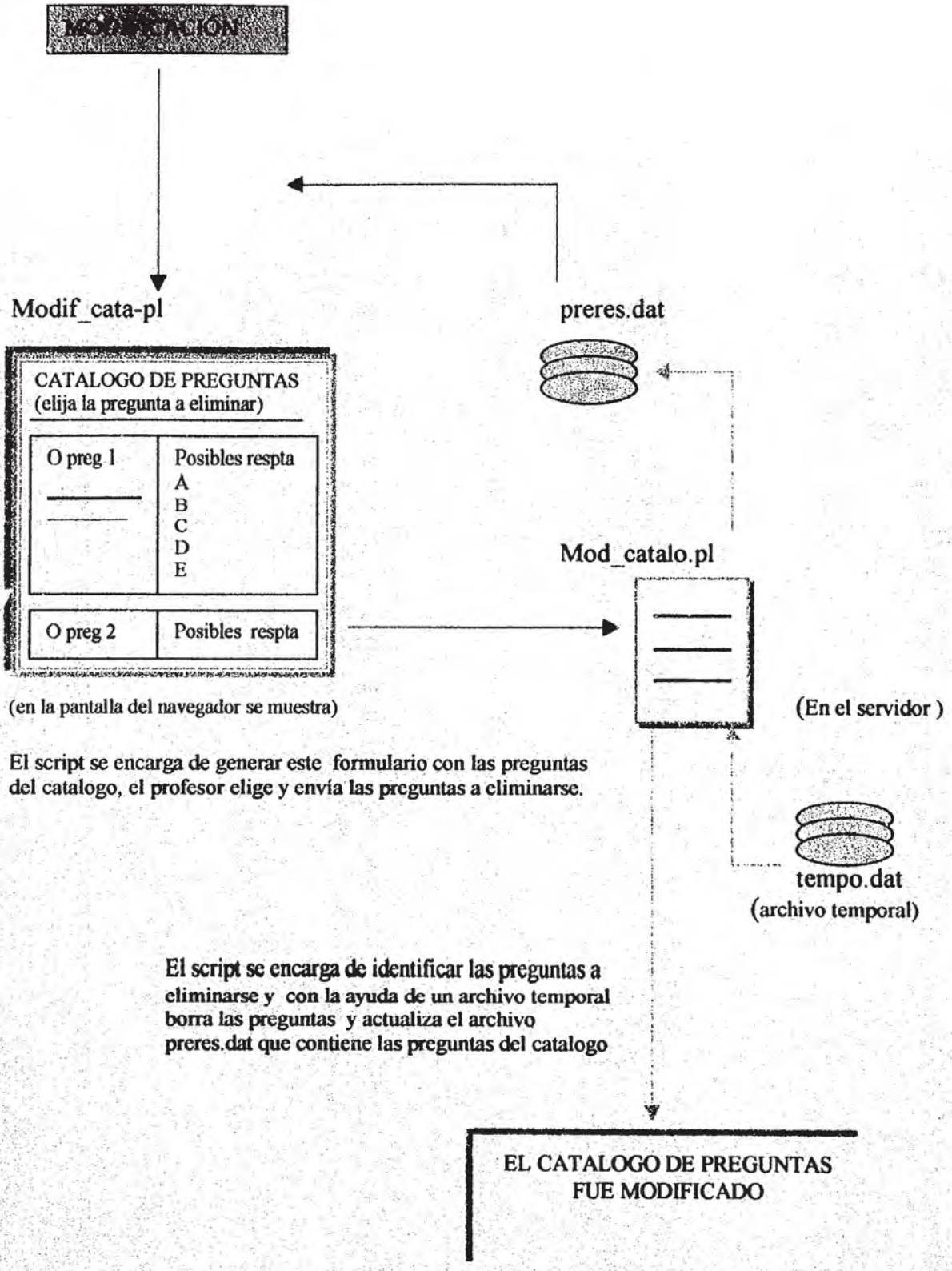
Modulo de actualización.-



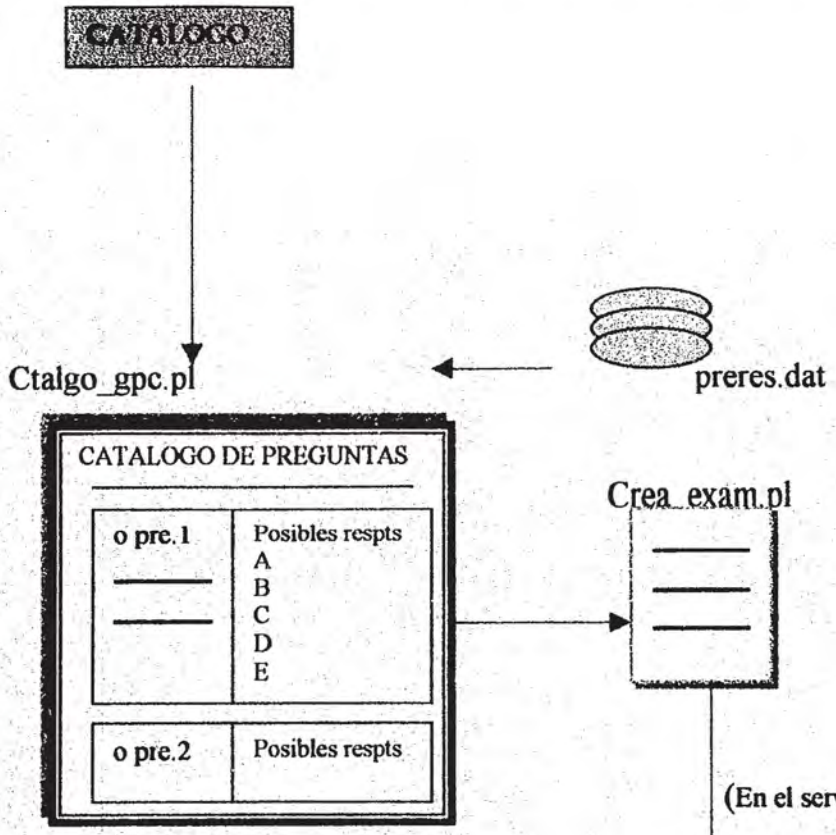
Se llena el formulario con las preguntas, posibles respuestas y solución que introducirá el profesor para enviarse al catalogo (pantalla en el navegador)

Respuesta que genera como salida el programa, que se envía a la pantalla del navegador

Modulo de modificación del catalogo de preguntas (eliminación de preguntas)



Modulo de selección de preguntas del catalogo



(En la pantalla del navegador muestra)

El script genera este catálogo con el formulario
De donde se eligen las preguntas que integraran el examen de Graficación. El Profesor las selecciona y envía

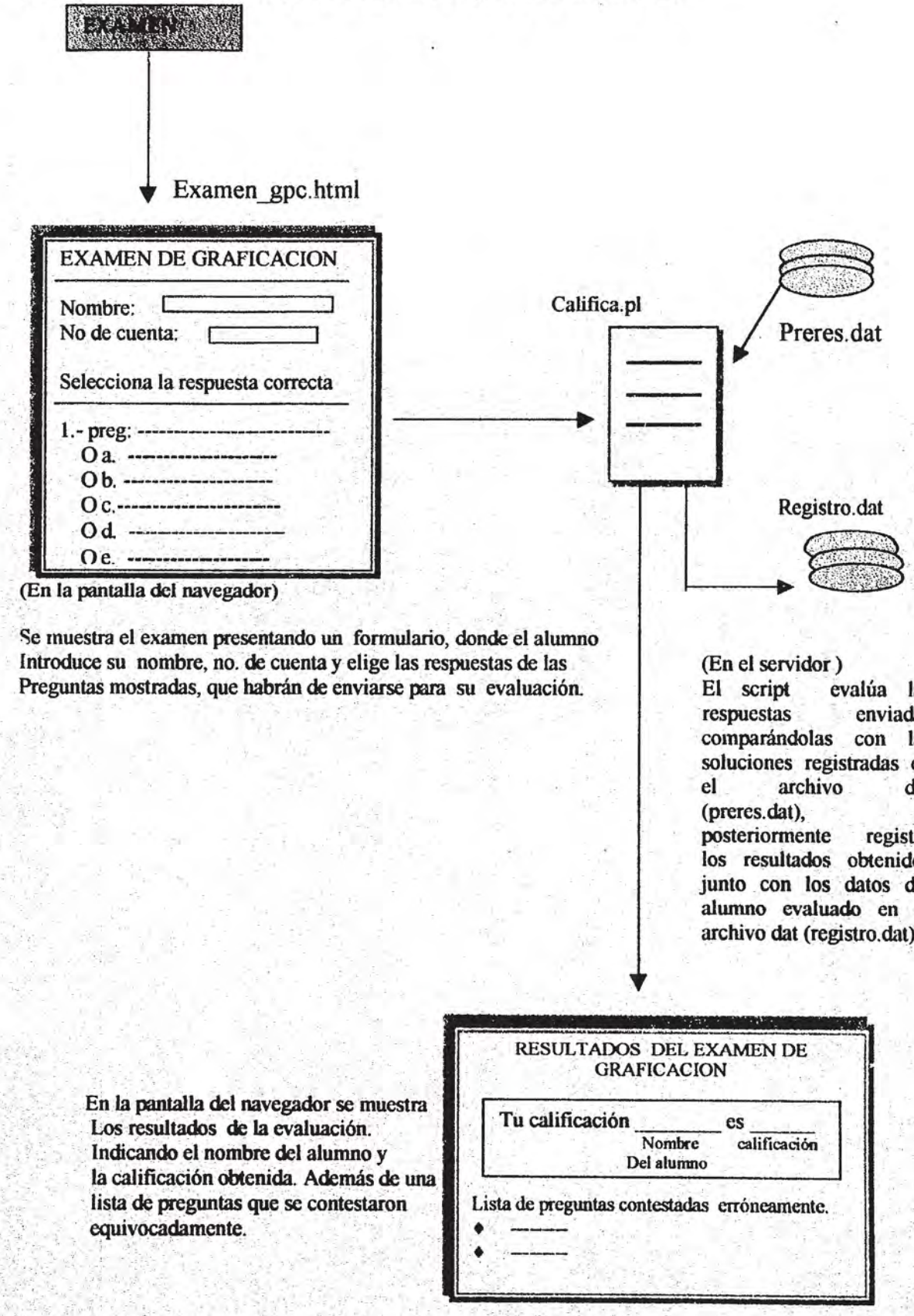
(En el servidor)

el script construye la pagina del examen examen_gpc.htm con las preguntas enviadas.

Examen_gpc.html

La salida del programa muestra
En la pantalla del navegador una liga de enlace al nuevo examen de Graficación por computadora listo.

Modulo de aplicación del examen y de evaluación.



4. Pruebas y puesta en marcha del Sistema de Auto - Evaluación de Graficación por Computadora.

En este apartado se consideraran aquellos elementos que intervendrán en el funcionamiento el sistema de auto- evaluación diseñado. Es conveniente realizar una serie de pruebas piloto a fin de observar los resultados y posibles errores que pudieran generarse, para realizar las modificaciones pertinentes antes de montar la aplicación en el servidor Web y no llevarse chascos.

Cada uno de los programas se han diseñado bajo condiciones que contemplan su ejecución en una computadora personal (PC).

Los requerimientos mínimos con los que debe contar la PC para ejecutar y probar los programas CGI son: S.O Windows 95, navegador Internet Explorer, procesador Pentium, memoria RAM (32 bits) o superior, 133 Mherz de velocidad o superior, disponibilidad de espacio en disco para la instalación del interprete de perl.

El primer paso consiste en instalar el intérprete de perl, es posible conseguir una versión de perl desde el sitio Web: http://www.tres.com/perl/direccines.html, que ofrece diversas ligas o enlaces a otras paginas que contienen tutoriales, el CPAN, utilidades y ejemplos de scripts CGI, además de llevar hacia la pagina ActiveWare de donde es posible bajar la revisión más reciente del interprete de perl para Windows 95/NT sin costo alguno. A través del sitio Activeware se realiza la distribución de perl para Windows 32 bits, poniendo a disposición las ultimas revisiones de la versión 5 del interprete de perl.

Cuando se baja el intérprete de perl a la PC, se copia a disco el archivo pw321315.exe que ocupa algo más de 1.5 MB que al ejecutarse realiza de forma automática la instalación en plataformas Windows 95/ 98/NT.
Durante la instalación del intérprete de perl se crea el directorio Perl en el directorio raíz de C, donde se ubicara el intérprete.

Enseguida de la instalación de perl, se preparan las condiciones para realizar las pruebas dentro de la PC, para ello, se crea un directorio con el nombre de *pruebas*, en este se colocan todos los programas CGI que habrán de ejecutarse. Los documentos html que se generaran durante la ejecución de los programas CGI se colocaran dentro del directorio *paginas*, ambos directorios se ubicaran dentro del directorio raíz de C.

Para dar inicio a las pruebas de los scritp o programas CGI se abre la ventana de comandos (ventana de MS-DOS), desde la cual se llama al intérprete de perl para probar el código de cada uno de los programas.

Dentro de la ventana de MS- DOS se teclea lo siguiente:

Primero se cambia al directorio pruebas donde están guardados los programas CGI

C:\>cd pruebas

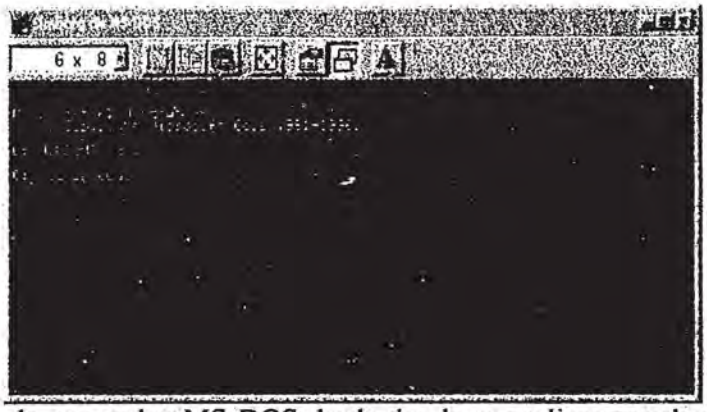


Figura 12. Ventana de comandos MS DOS desde donde se realizan pruebas de los programas CGI.

Ya ubicados en el directorio Pruebas, se invoca al interprete de perl tecleando la ruta completa de su ubicación, y junto a él el nombre del programa a ejecutar separado por un espacio.

C:\pruebas>c:\perl\bin\perl programa.cgi

Antes de proseguir, es necesario considerar que los programas CGI trabajan bajo el entorno cliente - servidor, de esta forma, estos programas habrá de procesar los datos que lleguen al servidor Web desde el navegador, para posteriormente generar información resultante que tendrá que regresar al navegador solicitante.
Entendido esto, se debe considerar que los datos que viajan del navegador hacia el servidor se conforman bajo ciertas normas de codificación. Por tanto, con la finalidad de realizar pruebas de cómo operarían de forma real los programas se debe emular dicha información con las mismas normas de codificación.

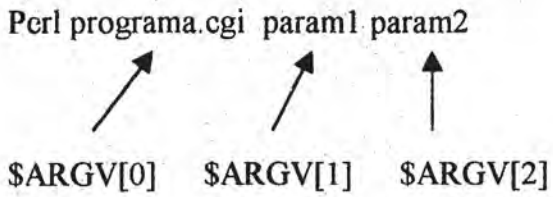
En el momento en que se oprime el botón enviar del formulario donde se introdujo la información, estos datos que viajan se codifican en pares claves=valor, separando cada campo de información del formulario por un "&".

De igual manera se debe seguir este formato de codificación para realizar las pruebas dentro de la PC, apegados a la forma real de cómo viaja la información.

La información que llega al servidor se almacena en las variables de entorno que estarán a disposición del programa CGI para su procesamiento, sin embargo, al ejecutarse bajo la PC las condiciones cambian, por lo que se debe hacer llegar esta información hasta el programa como una línea de parámetros que se almacenara en la variable reservada @ARGV.

Todo aquello que se teclee detrás del nombre del programa se considera como una lista de parámetros, que se separan por un carácter espacio.

PARAMETROS DESDE LA LINEA DE COMANDOS



Otro aspecto a considerar, es el hecho de que todo programa CGI genera información de salida que se transmite a la salida estándar a través de un canal que envía los datos hacia el servidor, quien a su vez los transmitirá por la red presentándola al navegador solicitante. El programa debe especificar el tipo de contenido MIME al servidor mediante la cláusula content-type, indicando de esta forma el formato en el se presentaran los datos de salida. Por lo tanto se debe emplear el formato HTML para que se visualice la información en el navegador.

Cuando se realizan pruebas de un programa CGI en la PC y genera como salida un documento html para ser visualizado en un navegador, las condiciones cambian, puesto que la salida estándar será la misma pantalla de la PC y debido al formato de los datos de salida estos no se podrán visualizar como tal en la pantalla, por esta razón, se redirigue la información de salida a un archivo tipo html, que podrá visualizarse posteriormente abriendo este archivo ***html*** en el navegador instalado (Internet Explorer).

La forma de enviar la salida html a un archivo es indicando en la línea de comandos de la ventana de MS-DOS el carácter de redirección ">" seguido de un nombre de archivo

```
Perl programa.cgi >salida.html
```

Cuando se ejecuta esta línea el intérprete no envía los datos de salida hacia la pantalla, sino a un archivo al que llamará salida.html. Cuando se disponga abrir este archivo se verá el aspecto real que tendrá el resultado generado, de esta forma se observa la presentación del resultado en el navegador y los posibles errores que pudieran existir.

Establecidas las condiciones que se deben considerar para ejecutar y probar los programas CGI dentro de la PC, es posible realizar las pruebas convenientes con cada uno de los programas diseñados, antes de colocarse en el servidor Web.

A través de la ventana de comandos de MS-DOS se llamara al intérprete de perl para pasarle como parámetros el nombre del programa a ejecutar y la línea de datos que se procesara.

- Pruebas para el primer programa: **crea_cata**.

Los datos que debe procesar este programa se enviarán desde el formulario del documento crea_catalgo.htm. La información que recibe son las preguntas elaboradas por el profesor. Para simular este flujo de información en la PC se escribe a continuación del nombre del programa la línea de datos bajo la codificación con la que llegaría hasta el servidor.

La línea codificada de datos corresponde a la pregunta, posibles respuestas y la solución de la misma:

p_1=seleccione+la+lista+que+contiene+
graficas+primitivas&rap1=linea+mapa+de+bits+arco+circulo&rbp1=punto+texto+vect
ores+arco&rcp1=linea+texto+punto+arco+circulo&solp1=sa

En la ventana de MS-DOS, dentro del directorio *pruebas* se teclea la línea:

C:\pruebas>c:\perl\bin\perl crea_cata.txt p_1=seleccione+la+lista+que+contiene+
graficas+primitivas&rap1=linea+mapa+de+bits+arco+circulo&rbp1=punto+texto+vect
ores+arco&rcp1=linea+texto+punto+arco+circulo&solp1=sa >salida.html [enter]

En esta línea se identifica entre cada espacio: la ruta del interprete perl, el nombre del programa, la línea de información que enviarían los campos del formulario y el nombre del archivo al que se redirecciona la respuesta de salida que genera el programa.

- Pruebas para el programa: **modif_cata**

Este programa solo recibe como parámetro el nombre del programa a ejecutarse y redirecciona la salida de este hacia el archivo salida.html.

En la ventana de MS-DOS, dentro del directorio *Pruebas*, se introduce la siguiente línea:

C:\pruebas>c:\perl\bin\perl modif_cata.txt >salida.html [enter]

La salida de este programa genera un formulario que enviara información hasta el programa mod_catalo, sobre las preguntas que se desean eliminar.

- Pruebas para el programa: **mod_catalo**

Este programa se encarga de recibir y procesar la información proveniente del formulario, que envía la relación de preguntas a eliminar del catalogo. Así, la línea de parámetros que se debe introducir en la ventana MS-DOS para simular esta información es:

C:\pruebas>c:\perl\bin\perl mod_catalo.txt pre_1=pg1&pre_2=pg2&pre_3=pg3&
Pre_4=pg4 >sale.html [enter]

- Pruebas para el programa **ctalogo_gpc.**

Este programa se ejecuta sin introducir más parámetro que el nombre mismo del programa y redireccionando su salida.

En la ventana de comandos se escribe la línea de parámetros

C:\pruebas>c:\perl\bin\perl ctalogo_gpc.txt >sali.html [enter]

La salida de este programa genera un formulario. Este formulario nuestra el catalogo de preguntas, a partir de éste se enviara información al programa crea_exam.

- Pruebas para el programa: **crea_exam**.

Este programa procesa la información que proviene del formulario del programa anterior, específicamente, recibe aquellas preguntas elegidas del catalogo para ser incluidas en un nuevo examen.

Así la línea que se debe introducir en la ventana MS-DOS es:

C:\pruebas>c:\perl\bin\perl crea_exam.txt pre_1=pg1&pre_2=pg2&pre_3=pg3&
Pre_4=pg4 >sale.html

Se identifica entre cada espacio: la ruta para llamar al interprete, el nombre del programa, el flujo de datos enviados que llegara desde el formulario y el nombre del archivo al que se redirecciona la respuesta de salida que genera el programa.

La salida de este programa crea un documento html que se coloca en el directorio *paginas*, el archivo html creado es examen_gpc.html y contendrá las preguntas que conformara el nuevo examen.

- Pruebas para el programa **califica**.

Este programa recibe los datos del alumno, así como, las respuestas de las preguntas contestadas a través del formulario del documento examen. Durante la ejecución de este se generara y actualizara el archivo *registro.dat*, que contendrá los registros de los alumnos incluyendo: nombre del alumno, número de cuenta y calificación obtenida. Obviamente se realiza previamente un proceso de evaluación que genera una respuesta de salida informando la calificación obtenida al alumno.

Para las pruebas de este programa se introduce en la ventana de comandos de DOS lo siguiente:

C:\pruebas>c:\perl\bin\perl califica.txt nombre=Fionne+Ramirez&nocta=91770185&
Prg_1=sa&prg_2=sc&prg_3=sd&prg_4=se&prg_5=sa >sale.html

Entre cada espacio se observa: la ruta de interprete perl, el nombre del programa a ejecutar, la línea de datos enviados que viajaran desde el formulario, y el nombre del archivo al que se redireccionará la salida del programa.

Al realizar las pruebas dentro de la PC con cada uno de los programas CGI diseñados, emulando las condiciones y la forma en la que viajarían los datos desde el navegador, es más fácil identificar los posibles errores en los que se incurriría.
Una vez que se han probado, observado su comportamiento y se han obtenido los resultados esperados, se procede al siguiente paso, el cual consiste en colocar dichos programas en el servidor Web que administra la página Web de Graficación, cuya dirección es: tigre.aragon.unam.mx.

Para la instalación del sistema Auto-evaluación de Graficación por Computadora, es necesario ubicar en el servidor Web los documentos html y cada unos de los programas CGI en los directorios correspondientes.
Para este paso el administrador del servidor ha de proporcionar previamente una cuenta de usuario y un password que permite tener acceso al servidor Web con permisos para alojar archivos como propietario de estos y obviamente con la posibilidad de modificarse.

Una alternativa para colocar estos archivos dentro del servidor es emplear un programa FTP, esta herramienta resulta bastante practica para el envío de archivos de forma remota desde una PC. Con el programa FTP para Windows, llamado Cute FTP en su versión shareware, es posible enviar y recibir archivos entre la PC y el servidor bajo una conexión de Internet.

Al arrancar el programa FTP aparece la ventana que permite agregar " nuevos sitios" para conectar. Ver figura.

Lo que sigue es configurar el nuevo sitio al que se enviaran los scripts o programas. Con el botón "add site", se abre una ventana donde permite editar el nuevo sitio.

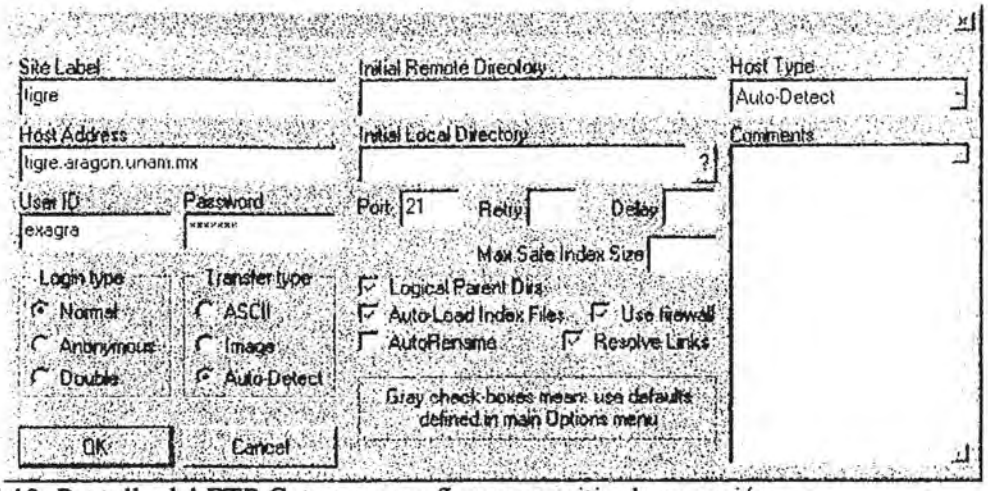


Figura 5.13. Pantalla del FTP Cute para configurar un sitio de conexión.

En esta ventana se introduce el host address (dirección del servidor)como: tigre.aragon.unam.mx, el user ID (nombre o cuenta de usuario) que se dispone: exagra y el password (clave de acceso).

Esta operación de configuración se realiza una solo vez y queda guardada en la ventana principal para elegir posteriormente el sitio al que se desea conectar, para ello basta con marcar el sitio y pulsar "connect".

Cuando se abre una sesión de FTP conectándose al sitio indicado se abre la ventana de control principal donde se observan dos paneles, uno del lado izquierdo, que muestra el directorio de la PC, y otro panel de lado derecho que muestra el archivo asignado a la cuenta dentro del servidor.

Para el envío y recepción de archivos:
Siempre que se envíe o se reciba un archivo se debe activar previamente la función de transferencia en "auto- detección", de manera que, automáticamente reconozca el tipo de archivo que se está enviando o recibiendo sin especificar si se trata de un archivo tipo ASCII, imagen, o binario.

En entre otras funciones que ofrece el CuteFTP es la opción de cambiar los permisos de ejecución, lectura o escritura de los archivos y directorios ubicados en el servidor, para ello se selecciona el archivo al que se le cambiaran los permisos en el panel derecho, y a continuación se abre el menú "commands" del menú de la ventana y de la opción "custom commands" se selecciona "change files acces mask" , donde se presenta una ventana en la que se puede especificar el código del permiso que se pretende dar al archivo o directorio seleccionado.

Dentro del servidor Web se dispone de los siguiente directorios donde se habrán de colocar los archivos que se envíen vía FTP.

Bajo el directorio raíz de usuario asignado a exagra: home/exagra/ se colocaran todos los archivos del sistema de auto –evaluación.

Cada uno de estos archivos se distribuirá en los siguientes directorios del servidor Web, bajo el directorio asignado a la cuenta de usuario exagra.

Una forma de verlo esquemáticamente es de la siguiente forma:

En el directorio cgi–bin se colocaran todo los programas o script CGI.
En el directorio html, se colocaran los documentos html.
Y en el directorio icons se guardan iconos o imágenes.

Home/httpd

/cgi-bin/exagra/	crea_cata.pl
	modif_cata.pl
	mod_catalo.pl
	ctalogo_gpc.pl
	crea_exam.pl
	califica.pl
	clave.pl
	alumn_reg.pl
	borregis.pl
	preres.dat
	registro.dat
	solucion.dat
/html/exagra/	crea_catalgo.htm
	paswd.htm
	presenta.htm
	(aquí se guardaran los documentos generados durante la ejecución del script)
	examen_gpc.html

Cuando ya se han enviado los archivos al servidor es fundamental establecer los permisos adecuados a cada uno estos, de manera que al ejecutarse los scripts no se generen errores ocasionados por los permisos inadecuados, puesto que resulta común que se presenten errores generados por esta razón.

Se tiene dos opciones para cambiar los permisos de los archivos, una es a través del programa FTP (CuteFTP) desde el menú de la ventana principal, o también dentro de una sesión de Telnet, herramienta con la que cuenta Windows 95/98, que permite realizar una conexión remota con el servidor Web e interactuar con el sistema operativo de éste como si estuviera en la propia PC.

Para abrir una sesión telnet y conectarse al servidor, se teclea en ejecutar del menú de inicio de Windows el comando "telnet", el cual abre una ventana y en la barra de menú se debe establecer conectar con "tigre.aragon.unam.mx". Cuando se conecta al servidor, este pide el nombre de la cuenta del usuario (exagra) y la clave de acceso asignada para poder entrar al sistema y colocarse en el directorio raíz del usuario.

Bajo la sesión de telnet es posible introducir los comandos para copiar archivos, abrir un editor de línea en la pantalla para editar archivos; renombrar archivos, cambiar o asignar permisos a los archivos, buscar rutas de archivos, etc, para esto se requiere tener un mínimo de conocimientos de comandos en Unix.
Los comandos se introducen a partir de la línea de comandos donde indica el cursor.

Una pantalla de sesión de telnet se puede observar como muestra la siguiente figura.

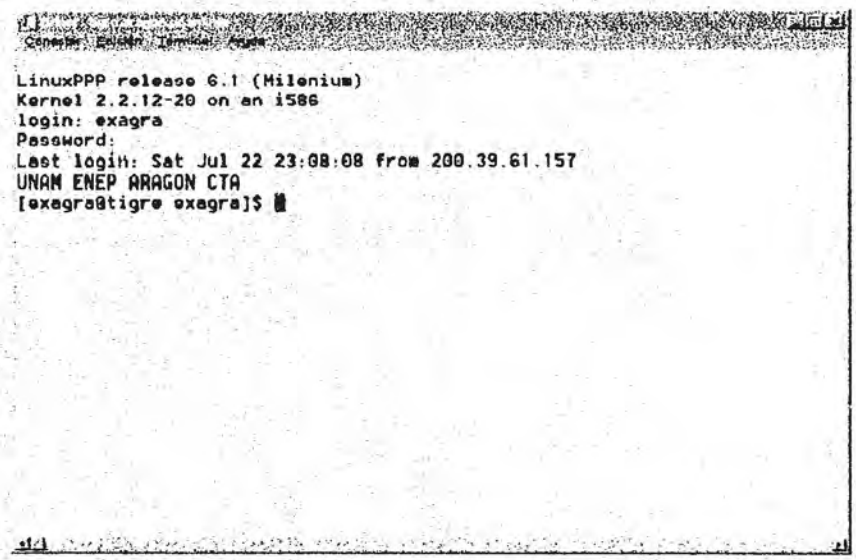


Figura 5.15. Pantalla que muestra una sesión de telnet.

Al entrar en una sesión de telnet siempre pedirá el user name (login) y el password asignado. Una vez comprobados entrara al directorio raíz del usuario.

En la sesión telnet será posible ubicar a través del uso del comando which la ruta del intérprete de perl que se encuentra en el servidor, dicha ubicación se habrá de referenciar en la primer línea de los programas CGI, para llevar a cabo su ejecución.

Este comando se usa de la siguiente forma:

[exagra@tigre exagra]$which perl [enter].

Con este comando se visualiza la ruta absoluta del archivo perl. La ruta que proporciona este comando es: /usr/bin/perl. (esta ruta de ubicación se debe escribir en la primer línea de los programas CGI)

Para abrir una sesión telnet y conectarse al servidor, se teclea en ejecutar del menú de inicio de Windows el comando "telnet", el cual abre una ventana y en la barra de menú se debe establecer conectar con "tigre.aragon.unam.mx". Cuando se conecta al servidor, este pide el nombre de la cuenta del usuario (exagra) y la clave de acceso asignada para poder entrar al sistema y colocarse en el directorio raíz del usuario.

Bajo la sesión de telnet es posible introducir los comandos para copiar archivos, abrir un editor de línea en la pantalla para editar archivos; renombrar archivos, cambiar o asignar permisos a los archivos, buscar rutas de archivos, etc, para esto se requiere tener un mínimo de conocimientos de comandos en Unix.
Los comandos se introducen a partir de la línea de comandos donde indica el cursor.

Una pantalla de sesión de telnet se puede observar como muestra la siguiente figura.

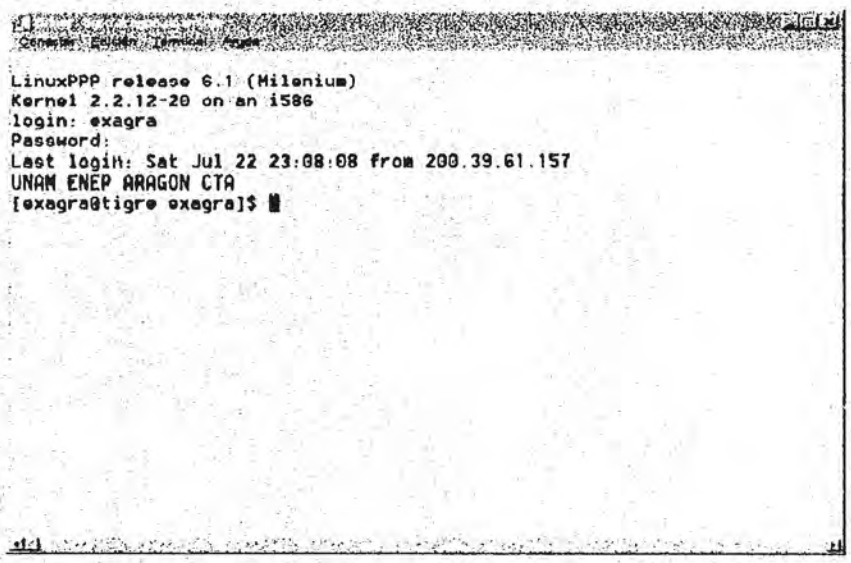


Figura 5.15. Pantalla que muestra una sesión de telnet.

Al entrar en una sesión de telnet siempre pedirá el user name (login) y el password asignado. Una vez comprobados entrara al directorio raíz del usuario.

En la sesión telnet será posible ubicar a través del uso del comando which la ruta del intérprete de perl que se encuentra en el servidor, dicha ubicación se habrá de referenciar en la primer línea de los programas CGI, para llevar a cabo su ejecución.

Este comando se usa de la siguiente forma:

[exagra@tigre exagra]$which perl [enter].

Con este comando se visualiza la ruta absoluta del archivo perl. La ruta que proporciona este comando es: /usr/bin/perl. (esta ruta de ubicación se debe escribir en la primer línea de los programas CGI)

Se debe renombrar los programas CGI que se enviaron como archivos tipo texto, a nombres con terminación .pl , de manera que el interprete de perl los reconozca como scripts perl.

De esta forma los siguientes archivos se renombraran como sigue:

Crea_cata.txt	a	crea_cata.pl
Modif_cata.txt	a	modif_cata.pl
Mod_catalo.txt	a	mod_catalo.pl
Ctalogo_gpc.txt	a	ctalogo_gpc.pl
Crea_exam.txt	a	crea_exam.pl
Clave.txt	a	clave.pl
Califica.txt	a	califica.pl
Alumn_reg.txt	a	alumn_reg.pl
Borregis.txt	a	borregis.pl

Ya sea a través del programa FTP o mediante la sesión de telnet, se puede cambiar o renombrar estos archivos.

Con una sesión telnet se haría uso del siguiente comando:

[exagra@tigre exagra]$mv crea_cata.txt crea_cata.pl [enter]

El siguiente paso es establecer los permisos de acceso a cada uno de los archivos del sistema de auto-evaluación.

Para asignar los permisos correspondientes a cada uno de los archivos, se puede hacer uso del programa FTP como se menciono en párrafos anteriores o con una sesión telnet a través del comando: chmod.

En una sesión telnet el comando chmod se usa de la siguiente forma:

[exagra@tigre exagra]$chmod 755 crea_cata.pl [enter]

El número después de chmod hace referencia al permiso que se asigna al archivo que se menciona enseguida.

Para cada uno de los archivos que conforman el sistema de Auto-evaluación se debe asignar un permiso en particular. Los permisos correspondientes para cada tipo de archivo se mencionan a continuación.

A los programas CGI o scripts se les asigna el código de acceso 755, que permite acceso completo al propietario (7), autoriza a los demás usuarios de su grupo (5) y a otros (5), permisos de ejecución.

A los archivos tipo html no generados en la ejecución del scrpts se les asigna por default el código de acceso 644, que proporciona al propietario permiso de lectura y de escritura (6), a los demás usuarios de su grupo (4) y a los otros (4), permisos de lectura.

A los archivos tipo html, gif generados o solicitados en la ejecución de los scripts se les asigna el código de acceso 666, que da al propietario permiso de lectura y escritura (6), a los demás usuario de su grupo (6) y a los otros (6), permisos de lectura y escritura.

A los archivos tipo dat se les asigna el código de acceso 766, que permite al propietario acceso completo (7), a los demás usuarios de su grupo (6) y a los otros (6), permisos de lectura y escritura. Esto es para que permita tanto al profesor como a los alumnos permiso para agregar información a los archivos dat que contendrán información, tal como, la relación de preguntas elaboradas (archivo de catálogo de preguntas) o bien, la relación de registro de alumnos (nombre, número de cuenta, calificación) de los alumnos que efectúen el examen de Graficación.

Con el comando ls -l, similar al dir de windows, se lista los archivos de un directorio a detalle con los permisos asignados, los propietarios, tamaño y fecha de creación.

En caso de requerirse, un comando útil es vi o pico, para editar archivos que requieran de modificarse en alguna línea, la forma de ejecutarlo será:

[exagra@tigre exagra]$pico crea_cata.pl [enter]

Una vez conformado la configuración del entorno bajo el cual se trabajara el sistema, que involucra la correcta ubicación de los archivos en el servidor, así como, la designación de los permisos correspondientes, se está en condiciones para llevar acabo la ejecución y utilización del sistema de auto-evaluación de Graficación por Computadora. Desde cualquier computadora conectada a Internet el profesor podrá tener acceso al sistema de auto evaluación, sólo bastara con indicar la dirección correspondiente en el navegador:

http://tigre.aragon.unam.mx/exagra/clave.htm

A través de esta dirección el profesor entra al sistema en la pagina de identificación, donde introducirá su clave de acceso que le permita operar el sistema en cualquier parte donde se encuentre, sin estar exactamente dentro de la aula de clases.
O bien, cuando el alumno este en cualquier lugar con una computadora conectada a Internet podrá realizar su examen en la siguiente dirección y obtener sus resultados de forma inmediata.

http://tigre.aragon.unam.mx/exagra/examen_gpc.htm

Aunque, también se puede disponer de una sesión de examen bajo la observación del profesor, que bien puede ser en el centro de computo, donde se tiene acceso a Internet. Mediante las facilidades que provee la utilización del Internet y la acción de los script CGI para optimizar la evaluación de exámenes, será mínimo el tiempo y esfuerzos dedicados para esta labor.

Se debe renombrar los programas CGI que se enviaron como archivos tipo texto, a nombres con terminación .pl , de manera que el interprete de perl los reconozca como scripts perl.

De esta forma los siguientes archivos se renombraran como sigue:

Crea_cata.txt	a	crea_cata.pl
Modif_cata.txt	a	modif_cata.pl
Mod_catalo.txt	a	mod_catalo.pl
Ctalogo_gpc.txt	a	ctalogo_gpc.pl
Crea_exam.txt	a	crea_exam.pl
Clave.txt	a	clave.pl
Califica.txt	a	califica.pl
Alumn_reg.txt	a	alumn_reg.pl
Borregis.txt	a	borregis.pl

Ya sea a través del programa FTP o mediante la sesión de telnet, se puede cambiar o renombrar estos archivos.

Con una sesión telnet se haría uso del siguiente comando:

[exagra@tigre exagra]$mv crea_cata.txt crea_cata.pl [enter]

El siguiente paso es establecer los permisos de acceso a cada uno de los archivos del sistema de auto-evaluación.

Para asignar los permisos correspondientes a cada uno de los archivos, se puede hacer uso del programa FTP como se menciono en párrafos anteriores o con una sesión telnet a través del comando: chmod.

En una sesión telnet el comando chmod se usa de la siguiente forma:

[exagra@tigre exagra]$chmod 755 crea_cata.pl [enter]

El número después de chmod hace referencia al permiso que se asigna al archivo que se menciona enseguida.

Para cada uno de los archivos que conforman el sistema de Auto-evaluación se debe asignar un permiso en particular. Los permisos correspondientes para cada tipo de archivo se mencionan a continuación.

A los programas CGI o scripts se les asigna el código de acceso 755, que permite acceso completo al propietario (7), autoriza a los demás usuarios de su grupo (5) y a otros (5), permisos de ejecución.

A los archivos tipo html no generados en la ejecución del scrpts se les asigna por default el código de acceso 644, que proporciona al propietario permiso de lectura y de escritura (6), a los demás usuarios de su grupo (4) y a los otros (4), permisos de lectura.

A los archivos tipo html, gif generados o solicitados en la ejecución de los scripts se les asigna el código de acceso 666, que da al propietario permiso de lectura y escritura (6), a los demás usuario de su grupo (6) y a los otros (6), permisos de lectura y escritura.

A los archivos tipo dat se les asigna el código de acceso 766, que permite al propietario acceso completo (7), a los demás usuarios de su grupo (6) y a los otros (6), permisos de lectura y escritura. Esto es para que permita tanto al profesor como a los alumnos permiso para agregar información a los archivos dat que contendrán información, tal como, la relación de preguntas elaboradas (archivo de catálogo de preguntas) o bien, la relación de registro de alumnos (nombre, número de cuenta, calificación) de los alumnos que efectúen el examen de Graficación.

Con el comando ls -l, similar al dir de windows, se lista los archivos de un directorio a detalle con los permisos asignados, los propietarios, tamaño y fecha de creación.

En caso de requerirse, un comando útil es vi o pico, para editar archivos que requieran de modificarse en alguna línea, la forma de ejecutarlo será:

[exagra@tigre exagra]$pico crea_cata.pl [enter]

Una vez conformado la configuración del entorno bajo el cual se trabajara el sistema, que involucra la correcta ubicación de los archivos en el servidor, así como, la designación de los permisos correspondientes, se está en condiciones para llevar acabo la ejecución y utilización del sistema de auto-evaluación de Graficación por Computadora. Desde cualquier computadora conectada a Internet el profesor podrá tener acceso al sistema de auto evaluación, sólo bastara con indicar la dirección correspondiente en el navegador:

http://tigre.aragon.unam.mx/exagra/clave.htm

A través de esta dirección el profesor entra al sistema en la pagina de identificación, donde introducirá su clave de acceso que le permita operar el sistema en cualquier parte donde se encuentre, sin estar exactamente dentro de la aula de clases.
O bien, cuando el alumno este en cualquier lugar con una computadora conectada a Internet podrá realizar su examen en la siguiente dirección y obtener sus resultados de forma inmediata.

http://tigre.aragon.unam.mx/exagra/examen_gpc.htm

Aunque, también se puede disponer de una sesión de examen bajo la observación del profesor, que bien puede ser en el centro de computo, donde se tiene acceso a Internet. Mediante las facilidades que provee la utilización del Internet y la acción de los script CGI para optimizar la evaluación de exámenes, será mínimo el tiempo y esfuerzos dedicados para esta labor.

CONCLUSIONES

Este trabajo de tesis representa la culminación de un proceso de maduración, no sólo el de unos meses a la fecha, sino de cinco años de preparación durante la carrera de Ingeniería en computación.

El desarrollo de esta investigación permitió concretar conceptos, ideas y conocimientos aplicados a la solución de una problemática especifica dentro del área de la docencia.
Los cometidos y objetivos planteados al inicio de esta investigación, que eran renovar el proceso en la evaluación de exámenes para la asignatura de Graficación por Computadora, se alcanzan al crear un sistema de auto - evaluación de exámenes.

Este sistema representa una herramienta que se construye a través de los script CGI, recurso que se trae gracias a las facilidades que se obtienen con Internet, de manera que se logra sintetizar tanto la elaboración de exámenes, como el calificado de estos mismos.

A través del terreno que han ganado los script CGI dentro del Internet, resultan una alternativa idónea para enfrentar las necesidades de nuestros tiempos, donde la facilidad y rapidez para la obtención de información apoyaran la actividad diaria del docente.
Gracias a que el Internet conforma ahora una nueva forma de comunicación poderosa y accesible, se puede llegar a cualquier parte del mundo, así la utilización de esta herramienta permite estar a la par de la tecnología poniéndose al servicio de las actividades de la enseñanza y del aprendizaje en cualquier lugar donde se demande, no sólo para el caso de la asignatura de Graficación por Computadora, sino para otras tantas asignaturas que podrán ser beneficiadas.

Esta herramienta como tal libera trabajo y tiempo invertido para dar más espacio tanto al profesor como al alumno y se enfoquen en conseguir las metas de la enseñanza, que tienen que ver con la transmisión y apropiamiento del conocimiento, así como de la culturización del alumno, que permite formar profesionistas de calidad dentro de nuestra universidad.

Con este trabajo se coloca el primer escalón de muchos que pueden construirse para beneficiar al grupo de personas que tienen a su cargo la labor de la enseñanza de las generaciones que están por formarse en el ámbito universitario.

Mientras se tenga al alcance los elementos técnicos y tecnológicos que permitan desarrollar nuevas herramientas que ayuden a facilitar el desempeño de las actividades del hombre en cualquier área que este se encuentre, será siempre un nuevo reto para el ingeniero en computación aplicar positivamente la tecnología a favor de nuestra sociedad

BIBLIOGRAFIA

Becker, Data. *Todo sobre Internet*. Marcombo, Barcelona 1995, 400p.

Breedlove, Bod. *Web programming unleashed.* ed. Sams Net, Indiana 1996, 1010p.

Comer E, Douglas. *El libro de Internet*. Prentice Hall, México, D.F. 1995, 312p.

Ferreyra, Gonzalo. *Internet paso a paso hacia la autopista de la información*. Computec Alfaomega, México, D.F. 1996, 423p.

Garcia, Fco Javier y Tramullas, Jesus. *Internet y ciberespacio World Wide Web.* Ra –Ma, Madrid 1995, 197p.

Medinets, David. *Perl 5 a través de ejemplos.* Prentice Hall, México, D.F. 1997, 658p.

Palacio, Juan. *Perl paginas Web interactivas*. Computec Ra- Ma, México, D.F. 1999, 380p.

Stout, Rick. *World Wide Web Manual de referencia*. Mc graw Hill, México, D.F. 1996, 525p.

Soria, Ramón. *HTML diseño y creación de paginas Web*. Computec Ra - Ma, México, D.F. 1998, 153p.

Sol, selena y Berznieks, Gunther. *Instant Web Scripts with CGI/perl*. M & T Books, New York 1996, 809p.

Tittel, ed y Gaither, Mark, et al. *Foundations of World Wide Web programming with HTML & CGI*. Programmers press, California, E.U. 1995, 648p.

"Lenguaje de programacion perl", dirección en Internet: http://www.yahoo.com/Computers_and_internet/programming_languajes/perl/, fecha de consulta 25 de junio de 2000.

" Perl" , dirección en Internet: http://perl.com /info/documentation.html. fecha de consulta 18 de agosto de 2000.

"Código perl ", dirección en Internet: http://www. Tres.com/perl. Fecha de consulta 24 de mayo de 2000.

Hermoso Najera, Salvador, Ciencia de la educación, Nva. Biblioteca Pedagógica, ed. Oasis, México, 1982, p260.