Diseño de Sitios Web

Amilcar Meneses Viveros

Departamento de Aplicación de Microcomputadoras
Instituto de Ciencias
Universidad Autónoma de Puebla

Julio 2001

Resumen

El crecimiento acelerado del uso la tecnología de Internet, a ocasionado que muchas personas, que no se consideran profesionales en el terreno de la computación, puedan utilizar esta tecnología para desplegar información en el Web. Esta aceleración a propiciado que el Web tenga una gran cantidad de información a la que cualquier persona, que tenga una computadora conectada a Internet, puede accesar. Sin embargo, el mal uso de está tecnología y la mala organización de la información, en las diferentes páginas que forman un sitio Web, ocasionan que la localización de la información que se busca sea difícil o se presente de manera confusa.

No basta con conocer los aspectos técnicos —por ejemplo creación de documentos HTML, manejo de CGI's, Scrips, Java, etc. —para desarrollar un sitio Web, también hace falta que los diseñadores sepan organizar la información de manera adecuada. En este documento se sintetizan muchas ideas de diseñadores y desarrolladores de sitios Web para crear y organizar un sitio de manera adecuada, estas ideas son tanto en técnicas de presentar la información, como en aspectos técnicos que se deben considerar y conocer.

Capítulo 1

Introducción

El crecimiento acelerado del uso la tecnología de Internet, a ocasionado que muchas personas, que no se consideran profesionales en el terreno de la computación, puedan utilizar esta tecnología para desplegar información en el Web.

Internet se desarrolló a partir de un proyecto militar a finales de la década de los sesenta, cuyo objetivo era crear una red en la cual, las diferentes computadoras que tenía el Departamento de Defenza de los Estados Unidos, pudieran compartir información. Debido a que este tipo de red podría considerarse un objetivo militar, uno de sus requerimientos era que no debería estar centralizada, tal que si uno de los puntos de la red fallaba, esta debería de ser capaz de utilizar los demás puntos de la red para seguir compartiendo su información. Esta red se le conoció como ARPAnet[1]

Años después, la Fundación Nacional de Ciencias de Estados Unidos quiso crear una red para conectar a los institutos de investigación, optó por el protocolo que utilizaba ARPAnet (TCP/IP) e inició Internet.

Al inicio de la decada de los noventa, la Fundación Nacional de Ciencias permitió la entrada de la iniciativa privada para el uso comercial de Internet y, en 1993 despegó su popularidad.

Sin embargo, el manejo de las herramientas Internet presentaba una barrera tecnológica para los nuevos usuarios, ya que casi toda la información que se compartía era, básicamente, texto ASCII. Afortunadamente para los

[1]ARPAnet fue nombrada por la Agencia de Proyectos de INvestigación Avanzada del Departamendo de Defenza (ARPA).

nuevos usuarios, casi al mismo tiempo, físicos del CERN[2], liberaron HTML (lenguaje para marcación de hipertexto), lenguaje cuyo objetivo era compartir documentos electronicos multimedia a través de Internet.

La popularidad de HTML comenzó cuando estudiantes y profesores del NCSA[3], de la Universidad de Illinois, en Urbana Champaign desarrollaron un Navegador denominado Mosaic. Este navegador fue desarrollado originalmente sobre la plataforma de NeXTSTEP. Los desarrolladores de Mosaic se dieron cuenta del potencial de esta herramienta y de las tecnologías de los servidores Web, y abandonaron NCSA para fundar Netscape Communications, para producir su versión de Mosaic, llamada Navigator, y programas servidores web.

La edición de documentos HTML es la forma en que la mayoría de los usuarios de Internet publican su información. Sin también se deben conocer algunos otros aspectos técnicos al respecto.

Toda computadora conectada a Internet (o a una red con protocolo TCP/IP) posee una dirección única que la diferencía de las demás. Esta dirección está formada por cuatro números enteros (cada número está en el rango $[0, 255]$) unidos por un punto. Por ejemplo: 148.255.1.3 o 122.1.0.155. A esta dirección se le llama dirección IP.

Para evitar que los usuarios utilicen números, es frecuente que las computadoras tengan un nombre asociado. Para evitar las repeticiones de nombres, se aprovecha el echo que Internet es una red formada por redes, así que se hace una división de grupos llamados dominios, y a que a su vez tienen una división de varios subdominios. Así, el nombre de una computadora en Internet se compone del nombre de la computadora (definida en su red local), concatenada con punto con los diferentes dominios. Por ejemplo:

```
delta.cs.cinvestav.mx
delta.icivil.unam.mx
delta.cs.mit.edu
www.jornada.unam.mx
www.cs.cmu.edu
www.peanuts.org
www.apple.com
```

[2]CREN es el Laboratorio Europeo de Física de Partículas

[3]National Center for Supercomputing Applications

Existen diferentes extensiones en el nombre que identifican a la rama a la que pertenece la máquina. La rama *edu*, indica que se trata de una institución educativa, la rama *com* indica que se trata de una rama comercial. la rama *org* indica que se trata de una rama no lucrativa, *gov* para gubernamentales[4]. Cuando las computadoras están fuera de Estados Unidos se les agregan extensiones que hacen referencia a sus respectivos paises: *mx* para México, *arg* para Argentina, y *ca* para Canada, por mencionar algunos.

Para poder utilizar los nombres en la red, se utilizan computadoras que sirven como servidores de nombres, las cuales mantienen las tablas de nombres con sus direcciones IP.

Las máquinas que se encargan de almacenar y suministrar los documentos HTML que un usuario requiera se denominan servidores (servidores Web). Las máquinas de los usuarios que despliegan las páginas de los servidores desde los navegadores se denominan clientes. Las operaciones que ocurren en el servidor se denomina que es en el *lado del servidor*, estas operaciones incluyen la administración de las páginas HTML, ajecuciones de CGI, etc. Las operaciones que se ejecutan en la máquina que tiene el navegador, se denominan operaciones *de lado del cliente*, estas incluyen la solucitud de páginas, ejecución de Applets, etc.

Cuando un cliente solicita la entrada a un sitio de internet, se envía una solicitud al servidor para que envíe la página principal. Esta solicitud se apoya en el uso de servidores de nombres para localizar la dirección IP correspondiente del servidor. Cuando un servidor de Internet recibe una solicitud, verifica que el cliente que hace la solicitud esté autorizado, y en caso de ser así, busca la página y la envía, en caso de encontrarlo. Si no se tiene el permiso o si no se tiene el documento que se solicita se envía un mensaje de error. También es posible que los navegadores soliciten a los servidores otro tipo de recursos, ademas de documentos HTML, como son: Applets, archivos de sonido, o archivos de imágenes, por mencionar algunos.

No es necesario estar conectado a Internet para escribir documentos HTML, basta con tener un editor de texto y un navegador para visualizarlos. Además se puede contar con otras herramientas de software que se encargan de editar estos documentos de manera gráfica, sin que el usuario se entere que está utilizando HTML.

HTML es un lenguaje que está enfocado al manejo de hipertexto. Tiene

[4] *gov* es en Estados Unidos, y en México es *gob*

un conjunto de estándares definidos, sin embargo las grandes compañias que se encargan de desarrollar los navegadores (como Microsoft y Netscape) crean sus propias extensiones al lenguaje e intentar definirlas como nuevos estándares. Recomendamos a los desarrolladores de sitios Web centrarse en estos cuatro puntos para que sus documentos sean visibles por cualquier navegador:

1. Concentrarse en el contenido y no en la apariencia. Recuerde que HTML se encarga sólo de dar un formato u organización básica al documento y que muchos detalles de visualización se pueden definir desde el navegador.

2. Evitar el uso de etiquetas que no son parte del estandar de HTML. Para conocer más del estandar recomendamos ver la página

   `http://www.w3.org/`

3. Evitar el diseño orientado a un navegador. Ya que como se mecionó en el punto anterior, muchos elementos visuales se definen en sus preferencias, y además, cada uno presenta el documento HTML de manera diferente (aunque el documento utilice sólo HTML estándar).

4. Evitar el uso de extensiones y uso de fallas. Muchos desarrolladores de sitios prefieren utilizar las extensiones que han desarrollado Netscape y Microsoft, para obtener efectos especiales en la presentación de sus documentos. O aprovechan las fallas de algunos navegadores para desplegar código HTML para obtener efectos (en realidad son defectos) especiales. Sin embargo los usuarios de Internet pueden utilizar distintos navegadores a parte de Explorer y Navigator, como OmniWeb, Mosaic o Lynx.

Aparte de estas recomendaciones técnias los derarrolladores deben atender a un conjunto de reglas de diseño para facilitar la búsqueda de información en el sitio que estén desarrollando. Se deben considerar no sólo los aspectos técnicos para hacer esto eficiente, sino apectos de diseño gráfico, y de redacción de texto principalmente.

Este documento contiene cuatro partes princiapales: primero se discuten las características principales de HTML, después se realizan algunas discertaciones y recomendaciones de la arquitectura tecnológica que se puede

utilizar. En el tercer capítulo se discute la arquitectura de la información, y finalmente, se discute el funcionamiento de los CGI's.

Capítulo 2

El lenguaje HTML

HTML es el lenguaje *"natural"* para publicar hipertexto en el World Wide Web. Es un formato no propietario basado en SGML, y se puede crear y procesar por una gran cantidad de herramientas, desde simples editores de texto hasta sofisticadas herramientas de autoría WYSIWYG[1], como DreamWeaver. HTML utiliza etiquetas (o *"tags"* como se le conoce en alguna literatura) como `<h1>` y `</h1>` para estructurar textos en encabezados, párrafos, listas y ligas de hipertexto, entre otros elementos de una página.

2.1 Iniciando con HTML

Esta es una pequeña introducción para escribir HTML. Se pueden escribir HTML desde herramientas sencillas como editores. Una buena forma de aprender HTML es viendo como otras personas editan sus documentos HTML. Para esto se utilizan las opciones *View-Source* del navegador que se esté utilizando.

En páginas sencillas se puede ver como iniciar con un título, agregar encabezados y párrafos, enfatizar texto, agregar imágenes, ligar a otras páginas y utilizar distintos tipos de listas.

[1]WYSWYG: de las siglas en inglés *"What you see, what you get"*, es decir, lo que vez es lo que obtienes

2.1.1 Elementos principales de una página HTML

Una página HTML comienza y finaliza con las étiquetas `<html>` y `</html>`, para indicarle al navegador que se presenta información con formato HTML. Sin embargo algunos navegadores pueden omitir estas étiquetas, pues casi toda la información que presentan es HTML y la consideran redundante. Apesar de esta consideración siempre es recomendable colocarla, pues forma parte del estandar de HTML y algunos navegadores viejos si la consideran.

Una página HTML contiene dos elementos principales: un encabezado y un cuerpo. El el encabezado contiene información general de la página, como su título, el tipo de *font* que se utilizará, los colores que se quieran predeterminar, y algunos scripts[2]. Aunque se puede omitir esta parte de la página siempre resulta conveniente considerarla. El cuerpo contiene la información que se muestra en el área de despliegue del navegador.

La estructura principal de una página HTML se ve de esta forma:

```
<html>
<head>
<!-- Aqui se definen los par\'ametros globales de la p\'agina -->
</head>
<body>
<!-- Aqui se definen el cuerpo de la pagina, es decir la informacion
     que se va a desplegar -->
</body>
</html>
```

En este ejemplo, Se puede observar que el inicio y fin de la página llevan las etiquetas: `<html>` y `</html>`. Además las etiquetas `<head>` y `<body>` necesitan sus respectivas etiquetas de cierre `</head>` y `</body>`. De hecho, la mayoría de las etiquetas HTML tiene una etiqueta de cierre. Finalmente, la etiqueta `<!-- ... -->` se utiliza para colocar comentarios, los cuales no despliega el navegador, y son para propósito de documentación de la página.

2.1.2 Iniciando una página con un Título

Todos los documentos HTML necesitan un título. Para este propósito se escribe la siguiente declaración en la parte de encabezado del archivo HTML.

[2]Un script es un pequeño programa que interpreta el navegador

```
<html>
<head>
<title>Mi primer documento HTML</title>
</head>
<body>
<!-- Aqui se definen el cuerpo de la pagina, es decir la informacion
     que se va a desplegar -->
</body>
</html>
```

El texto *"Mi primer documento HTML"* se cambia por el que se necesite. La etiqueta `<title>` precede al texto del título y termina con la etiqueta de finalización `</title>`. El título debe ponerse al inicio del documento y se despliega en la barra de encabezado de la ventana del navegador.

2.1.3 Agregando encabezados y párrafos

Es bien conocido que los diferentes encabezados le dan a un texto una importancia distinta como son capítulo, sección, subsección, etc. En HTML existen seis niveles de encabezados que van desde `H1` hasta `H6`, donde `H1` es el de mayor importancia y `H6` el de menor importancia.

En este punto es de vital importancia que el lector recuerde que HTML es un lenguaje para dar cierto formato u orden a documentos, lo que se ve en un navegador, muchas veces diferen de lo que se ve en el resto de navegadores, así que se sugiere que siempre se concentre más en la información que se va a desplegar que en la distribución en la página.

Los encabezados forman parte del cuerpo del documento, esto es, se deben colocar entre las etiquetas `<body>` y `</body>`.

La forma de escribir un encabezado importante es:

```
<h1>Un encabezado importante</h1>
```

y la forma de escribir un encabezado un poco menos importante es:

```
<h2>Un encabezado un poco menos importante</h2>
```

Cada párrafo se debe escribir entre las etiquetas `<p>` y `</p>` (la etiqueta `</p>` puede ser opcional). Por ejemplo:

```
<p>Este es el primer p&aacute;rrafo.</p>

<p>Este es el segundo p&aacute;rrafo.</p>
```

En este ejemplo se notan el uso de las llamadas entidades de carácter: `&aacute;`. Una entidad de carácter es un elemento de HTML que se utiliza para desplegar carácteres especiales, como letras con acentos y símbolos. La sintaxis de las entidades de carácter es:

$$\&\{nombre\_entidad\};$$

2.1.4 División de áreas de trabajo con ayuda de rayas y plecas

Se utiliza el elemento o etiqueta $< hr >$ para presentar rayas horizontales que se pueden utilizar como plecas o divisiones del área de texto.

Por ejemplo si deseamos una raya horizontal que atraviese toda el aŕea de despliegue del navegador, se colocaría:

```
<H1> Programaci&oacute;n Orentada a Objetos<H1>
<hr>
La Programaci&oacute;n Orientada a Objetos permite que los
programadores puedan reutilizar c&oacute;digo de formas
optimizadas y transparente. <p>
```

Si se desa colocar una pleca en un porcentaje de la página y alineada a un posición se egrega:

```
<hr width="40%" align="center">
<hr width="10%" align="right">
```

2.1.5 Etiquetas de estilo

En ocasiones es necesario presentar información con fonts especiales que le indiquen al lector que la frase o palabra es importante, o que hace referencia a un tipo de contenido. Dependiendo del navegador que se utilice se mostrarán como letras cursivas, negritas, subrayadas, o monoespaciado, por mencionar algunos estilos. Las etiquetas que tiene HTML para este propósito son[3]:

[3]Todas estas etiquetas requieren de su etiqueta de cierre correspondiente

`<em>` Se utiliza para enfatizar texto, la mayoría de los navegadores la despliegan como cursivas. Por ejemplo:

```
Realmente este es un t&oacute;pico <em>interesante</em>!
```

`<cite>` Esta etiqueta se utiliza para indicar que el texto delimitado indica una referencia bibliográfica, como el título de un libro o revista. Por ejemplo:

```
Gabriel Garc&iacute;a M&aacute;rquez obtuvo el premio
N&oacute;bel de literatura en 1982, con el libro
<cite>Cien a&ntilde;os de soledad</cite>.
```

`<code>` El navegador presenta el texto para que el lector sepa que se trata de una parte del código fuente de un programa. Por ejemplo:

```
La clase <code>Window</code> define las propiedades
y comportamiento de las ventanas en el ambiente
gr&aacute;fico de NeXTSTEP.
```

`<dfn>` Se utiliza esta etiqueta para definir términos o frases especiales. Por ejemplo:

```
Utilizamos la <dfn>Programaci&oacute; Orientada a Objetos</dfn>
como una alternativa para desarrollar programas con
c&oacute;digo reutilizable.
```

`<i>` Se uriliza para definir letras cursivas o itálicas. Por ejemplo:

```
La palabra <i>It&aacute;lica</i> debe aparecen en letras
cursivas.
```

`<kbd>` Se utiliza cuando se desea que el usuario tecleé algo desde su terminal. Por ejemplo:

```
Utilice el comando <kbd>ls -l</kbd>, si desea ver el listado
de su directorio con formato de lista.
```

`<strong>` Sirve para destacar texto, como $< em >$, pero con mas énfasis. Por ejemplo:

```
El siguiente texto es <strong>mucho muy importante</strong>.
```

`<b>` Coloca con negritas el texto que delimita. Por ejemplo:

```
Al <b>KKK</b> le molesta que lo escriban con negritas.
```

2.1.6 Manejo de tipos de letras

En realidad loa tipos de letras se manejan con las etiquetas que se mensionaron en el punto anterior. Sin embargo, la etiqueta `<FONT>` permite definir el tamaño y color que se puede utilizar. Se pueden utilizar 7 tamaños diferentes de *"font's"*. Por ejemplo,

```
<FONT SIZE="1" COLOR="red">Tama&ntilde;o 1</FONT><BR>
<FONT SIZE="2" COLOR="blue">Tama&ntilde;o 2</FONT><BR>
<FONT SIZE="3" COLOR="#000000">Tama&ntilde;o 3</FONT><BR>
<FONT SIZE="4">Tama&ntilde;o 4</FONT><BR>
<FONT SIZE="5">Tama&ntilde;o 5</FONT><BR>
<FONT SIZE="6">Tama&ntilde;o 6</FONT><BR>
<FONT SIZE="7">Tama&ntilde;o 7</FONT><BR>
```

El color en HTML no sólo es para el texto, también se puede definir para el fondo del navegador, y para las ligas (visitadas y no visitadas). En todos los casos el color se maneja con tipos predefinidos como: *"red", "blue", "green", "black"*, etc. Sin embargo, el usuario puede manipular este atributo utilizando la forma $\#rrggbb$, donde rr representa el valor hexadecimal (entre 00 y FF) del rojo, gg para el verde, y bb para el azul.

2.1.7 Manejando la alineación de elementos de un documento

Se puede alinear alementos de un documento HTML para obtener mejores resultados en su presentación.

Se puede utilizar la etiqueta `<center>` para centrar cualquier elemento que desee. Por ejemplo:

```
<center>
Este texto va centrado
</center>
```

Pára alinear texto se debe contemplar que se alinea a partir de párrafos, con un atributo en la etiqueta `<p>`. Por ejemplo:

```
<p align="left">
Este texto se alinea a la izquierda
<br>
Este texto tambi&eacute;n...
<p align="right">
Este texto se orienta a la derecha
<br>
Este texto tambi&eacute;n...
<p align="center">
Este texto va centrado
<br>
Este texto tambi&eacute;n...
```

En realidad las alineaciones de elementos se manejan como un atributo en las etiquetas que los manejan. Estas etiquetas incluyen: `<applet>`, `<caption>`, `<col>`, `<div>`, `<embed>`, `<h$>`, `<ifrmae>`, `<img>`, `<marquee>`, `<object>`, `<spacer>`, `<table>`, `<tr>`, `<th>` y `<td>`, para mayor información consultar [2].

2.1.8 Agregando imágenes

Las imágenes ayudan a que las páginas sean distintivas y además sirven de guía de los servicios que se ofrecen. La forma mas sencilla de agregar una imagen es utilizando la etiqueta `<img>`. Supongamos que se tiene una imagen llamada "gloriatrevi.jpeg" en el mismo directorio del archivo HTML, y tiene 200 pixeles de ancho por 150 pixeles de alto, entonces para desplegarla, se colocarí la siguiente línea de código:

```
<img src="gloriatrevi.jpeg" width="200" height="150">
```

Los atributos `width` y `height` no son estrictamente necesarios, pero ayudan a desplegar rápidamente la página Web. También es deseable que una imagen

tenga un título para saber a que se refiere en caso de que algún usuario no pueda visualizarla.

```
<img src="gloriatrevi.jpeg" width="200" height="150"
     alt="La chica del pelo suelto">
```

El atributo `alt` se utiliza para dar una breve descripción de la imagen. Para imágenes complejas también será necesario una mayor descripción. Supongamos que se ha escrito en el archivo "gloria.html", entonces se tendría:

```
<img src="gloriatrevi.jpeg" width="200" height="150"
     alt="La chica del pelo suelto" longdesc="gloria.html">
```

Se puede crear imágenes en muchas formas, por ejemplo con una cámara digital, digitalizando una imagen, o dibujando una. La mayoría de los navegadores entienden los formatos de imágenes GIF y JPEG, y algunos navegadores nuevos entienden el formato de imagen PNG. Hay que evitar el uso de imágenes de gran tamaño para no retardar la presentación de la página.

2.1.9 Definiendo colores y el fondo del navegador

Se puede definir el color de fondo, color de texto, o de ligas desde la etiqueta `<body>`. Por ejemplo si deseamos colocar un diseño o textura en el fondo de la imagen, se utiliza el atributo `background`.

```
<body background="fondo.jpg">
...
</body>
```

Si se desea poner un color como fondo se utiliza el atributo `bgcolor`, y para el color del texto el atributo `text`. Por ejemplo, para definir el color de fondo azul, y el texto en rojo:

```
<body bgcolor="blue" text="red">
...
</body>
```

Para controlar el color del hipertexto, se utilizan los atributos `link`, `vlink`, y `alink`. El atributo `link` hace referencia al color de las ligas que el usuario aún no visita. El atributo `vlink` hace referencia a las ligas que el usuario ya visitó, y el atributo `alink` hace referencia al hipertexto que el usuario tenga seleccionado con el ratón. Por ejemplo:

```
<body link="red" vlink="purple" alink="fuchsia">
</body>
```

2.1.10 Agregando ligas a otras páginas

La razón de que el Web sea efectiva, es su habilidad de definir ligas de una página a otra, y seguir las ligas presionando un botón del ratón.

Las ligas se definen con la etiqueta `<a>`. Por ejemplo, definiremos una liga a la página definida por el archivo *"gloria.html"*.

```
Esta es una liga a la <a href="gloria.html">p&aacute;gina
de Gloria Trevi</a>
```

El texto que se encuentra entre `<a>` y `</a>` se utiliza como la entrada a la liga. Es común que esta entrada esté en color azul y con texto subrayado.

Para realizar una liga a con otro sitio Web, se necesita tener la dirección WEB completa (comnmente llamada URL). Por ejemplo hacer una liga a la página *www.jornada.unam.mx*, se debe escribir:

```
Esta es una liga a <a href="http://www.jornada.unam.mx/">
La Jornada Virtual</a>.
```

En lugar de tener la dirección URL con el nombre, es posible tener la dirección IP, del sitio al se desea hacer referencia. Por ejemplo:

```
Para mayor informaci\'on, vea la p\'agina de la
<a href="http://148.247.2.176/ver2001/everano">
XXIV Escuela de Verano, <em>Computaci&oacute;n 2001</a>
```

También se puede poner una imagen en una liga de hipertexto, por ejemplo: las siguientes líneas permiten que se presione el botón en el escudo de la UAP para entrar a su página principal.

```
<a href="/"><img src="uap.gif" alt="UAP"></a>
```

2.1.11 Tres tipos de listas

HTML soporta tres tipos de listas. El primer tipo se le denomina lista no numerada. Esta lista utiliza las etiquetas `<ul>`, para indicar el inicio de la lista, y `<li>`, para indicar el inicio de cada elemento de la lista. Por ejemplo:

```
<ul>
  <li>Primer elemento en la lista</li>

  <li>Segundo elemento en la lista</li>

  <li>Tercer elemento en la lista</li>
</ul>
```

Note siempre se necesita finalizar la lista con la etiqueta `</ul>`, pero la etiqueta `</li>` es opcional.

El segundo tipo de lista es la lista numerada o lista ordenada. Esta lista utiliza las etiquetas `<ol>` y `<li>` (para indicar el inicio de la lista, y el inicio de cada elemento de la lista, respectivamente). Por ejemplo:

```
<ol>
  <li>Primer elemento en la lista</li>

  <li>Segundo elemento en la lista</li>

  <li>Tercer elemento en la lista</li>
</ol>
```

Al igual que la lista no numerada, siempre se debe terminar con la etiqueta de finalizacion `</ol>`.

La tercera lista es la lista de definicin. Esta lista te permite tener una lista de términos y sus definiciones. Este tipo de lista inicia con la etiqueta `<dl>` y termina con la etiqueta `</dl>`. Por ejemplo:

```
<dl>
  <dt>El primer termino</dt>
  <dd>Esta es su definicion</dd>

  <dt>El segundo termino</dt>
  <dd>Esta es su definicion</dd>

  <dt>El tercer termino</dt>
  <dd>Esta es su definicion</dd>
</dl>
```

Las etiquetas de finalización `</dt>` y `</dd>` son opcionales y se pueden omitir. Las listas pueden estar anidadas. Por ejemplo:

```
<ol>
  <li>El primer elemento de la lista</li>

  <li>
    El primer elemento de la lista
    <ul>
      <li>Primer elemento anidado</li>
      <li>Segundo elemento anidado</li>
    </ul>
  </li>

  <li>El tercer elemento de la lista</li>
</ol>
```

Además se pueden utilizar párrafos y encabezados para elementos de listas mayores.

2.2 Características avanzadas

Una vez comentado los temas bśicos, es hora de discutir características más avanzadas. En esta sección aprederemos a forzar el rompimiento de líneas, a introducir espacios que no se corten, hacer ligas a regiones dentro de la página, utilizar texto preformateado, a colocar flujo de texto alrededor de las imágenes, definir regiones con imágenes que se activen presionando el botón del ratón y crear tablas.

2.2.1 Como forzar el rompimiento de líneas

Ocacionalmente sera deseable forzar el rompimiento de líneas. Para hacer esto se utiliza el elemento `br`, por ejemplo si quieres incluir una dirección postal:

```
<p>Intech<br>
Av. 49 Poniente 1402<br>
```

```
Col. Prados Agua Azul<br>
Puebla, Pue. C.P. 72000</p>
```

El elemento `br` nunca necesita una etiqueta de finalización. En general los elementos que no tienen una etiqueta de finalización se les conoce como elementos vacíos.

2.2.2 Introducir espacios que no se corten

Los navegadores ordenan automáticamente el texto según sus márgenes. Los rompimientos de líneas se pueden colocar donde aparezcan espacios en blanco. En ocaciones es deseable prevenir que el navegador separe un conjunto de palabras. Ejemplos son nombres de productos y de personajes, por ejemplo "Gloria Trevi". El truco es utilizar ` ` en el lugar del carácter de espacio, por ejemplo:

```
Existen personajes ex&oacute;ticos, como
Gloria Trevi, que hacen hasta lo
imposible por llamar la atenci&oacute;n.
```

En general es una mala práctica utilizar en forma repetida el elemento ` ` para ordenar un texto. En vez de esto, se puede intentar hacer el ordenamiento a través de reglas de estilo.

2.2.3 Ligas a regiones dentro de la página

Es posible que se escriba una gran página Web que tenga un índice cerca del inicio, ahora aprenderemos a realizar las ligas a las secciones cortrespondientes dentro de la página.

Supongamos que cada sección inicia con un encabezado, por ejemplo:

```
<h2>Aplicaciones de Intranet</h2>
```

Entonces lo que se debe hacer es que el encabezado sea el destino de la liga del hipertexto. Para este propósito se debe encerrar su contenido con `<a name=identificador> ... </a>`

```
<h2><a name="aplicaciones-intranet">Aplicaciones de Intranet</a></h2>
```

El atributo nombre especifica el nombre que se utilizará para identificar la liga destino, en este caso *"aplicaciones-intranets"*. Ahora el índice puede incluir una liga de hipertexto que utilice este nombre, por ejemplo:

```
<ul>
  ...
  <li><a href="#aplicaciones-intranets">Aplicaciones de
      Intranet</a></li>
  ...
</ul>
```

Es necesario utilizar el carácter # antes del nombre destino. Si el destino se encuentra en un documento diferente, entonces se deberá poner la direccin web del documento antes del carácter #. Por ejemplo, si el documento está en `http://www.unam.mx/`, entonces la liga sería:

```
<a href="http://www.unam.mx/#aplicaciones-intranets">
   Aplicaciones de Intranets</a>
```

Poco a poco el lector será capaz de definir ligas destino sin utilizar el elemento `<a>`. Este nuevo método es mucho más fácil, todo lo que se necesita hacer es agregar un atributo `id` al encabezado, por ejemplo:

```
<h2 id="aplicaciones-intranets">Aplicaciones de Intranets </h2>
```

Este método no funciona para navegadores de cuarta generación o anteriores, se debe tener cuidado si se utilizan estos navegadores.

2.2.4 Texto Preformateado

Una de las ventajas del Web es que el texto se ajusta automaticamente en líneas dentro del área definida por el tamaño de la ventana. Sin embargo, en ocasiones es deseable deshabilitar esta propiedad. Un ejemplo es cuando se incluye código de un programa de ejemplo. Para esto se debe utilizar el elemento `pre`. Por ejemplo:

```
<pre>
    void Node::Remove()
    {
```

```
        if (prev)
            prev->next = next;
        else if (parent)
            parent->SetContent(null);

        if (next)
            next->prev = prev;

        parent = null;
    }
</pre>
```

Lo cual se muestra como:

```
    void Node::Remove()
    {
        if (prev)
            prev->next = next;
        else if (parent)
            parent->SetContent(null);

        if (next)
            next->prev = prev;

        parent = null;
    }
```

El texto y los colores de fondo fueron asignados a través de la página de estilos. Note que en todos los cambios de líneas y espacios son idénticos a los que aparecen en el archivo HTML. La excepción es una línea nueva exáctamente después de la etiqueta inicial `<pre>` e inmediatamente antes de la etiqueta final `</pre>` la cual se desecha. Esto significa que los siguientes dos ejemplos son tabularmente idénticos:

```
<pre>preformatted text</pre>

<pre>
preformatted text
</pre>
```

El texto con formato predefinido, generalmente se tabula utilizando un font uniespaciado donde cada carácter tiene el mismo ancho. Si se define una regla de estilo para el elemento `pre`, algunos navegadores olvidan utilizar el fondo de uniespacio, y necesitan utilizar la propiedad de la familia de fonts. Por ejemplo si deseas dibujar todos los elementos `pre` en verde, se puede definir la regla de stilo:

```
<style type="text/css">
  pre { color: green; font-family: monspace }
</style>
```

2.2.5 Flujo de texto alrededor de imágenes

Con HTML se puede escojer cualquier imagen se maneje como parte de una línea de texto, o que sea flotante a los márgenes derecho e izquierdo. Se puede controlar por este medio el atributo de alineación. Si este atributo se selecciona a la izquierda, la imagen se va hacia el margen izquiero, y si se selecciona a la derecha, la imagen se va hacia la derecha. Por ejemplo

```
 <p><img src="apple.jpg" alt="apple graphic"
width="32" height="21" align="left"> Este texto se ir&aacute;
al lado derecho de la gr&aacute;fica.</p>
```

El siguiente ejemplo utiliza `align="right"`

```
<p><img src="apple.jpg" alt="apple graphic"
width="32" height="21" align="right"> Este texto se ir&aacute;
al lado izquierdo de la grafica.</p>
```

Para forzar que esta interpretación continúe bajo la imagen, se debe utilizar el elemento `<br clear=all>`, por ejemplo:

```
<p><img src="apple.jpg" alt="apple graphic"
width="32" height="21" align="right"> Este texto se ir&aacute;
al lado izquierdo de la gr&aacute;fica.<br clear="all">
Esto inica una nueva l&iacute;nea bajo la imagen.</p>
```

2.2.6 Creando tablas

Las tablas consitutyen el elemto principal que muchos diseñadores de páginas (y programas) utilizan para organizar la presentación de un documento HTML. El manejo de tablas involucra el uso de varias etiquetas, como son: `<table>`, `<tr>`, `<td>`, y `<th>`.

La etiqueta `<table>` le inidica al navegador que inicia el formato de una tabla (hasta que se encuentre su respectiva etiqueta de cierra `</table>`). Esta etiqueta tiene los atributos `border`, `cellspacing`, `cellpadding` y `width` para permitir colocar bordes en la tabla, definir el espacio entre las celdas de la tabla, definir el espacio entre el borde de la tabla y su contenido, y especificar el ancho de la tabla con pixeles o con un porcentaje respecto al ancho de la página. Por ejemplo:

```
<table width="90%" cellpadding="3" cellspacing="4" border>
<!-- Aqui se definen los elementos de la tabla: renglones y celdas -->
<table>
```

Las tablas se componen de renglones de celdas, así, la etiqueta `<tr>` se utiliza para indicar el inicio y fin de un renglón de celdas. La etiqueta `<tr>` tiene los atributos `align`, `nowrap` y `valign`, para definir la alineación (izquierda, derecha o al centro) de las celdas que componen el rengón, detener el ajuste automático del navegador, y definir la alineación vertical .Por ejemplo:

```
<table border>
<tr nowrap valign="center">
  ...
</tr>
<tr align="right" valign="bottom">
 ...
</tr>
</table>
```

Para definir las celdas en los renglones se utilizan las etiquetas `<th>` y `<td>`. Ambas operan en forma similar y la única diferencia es que `<th>` se utiliza para definir encabezados de una tabla (el texto que contienen se muestra resaltado en el navegador). Un renglón puede tener cualquier número de celdas. Las celdas de las tablas pueden contener cualquier etiqueta HTML de formato. Las etiquetas para definir celdas cuentan con varios atributos, entre

los que destacan: `align, height, width, valign` y `nowrap`, para definir la alineación del contenido de la celda, su altura, su longitud, su alineación vertical, y si deshabilita la alineación automática, Por ejemplo:

```
<table border>
 <tr>
  <th>Alineada</th>
  <th>Arriba</th>
  <th>Linea Base</th>
  <th>Centro</th>
  <th>Abajo</th>
 </tr>
 <tr align="Center">
  <th><h1>Linea base del texto...<br>Linea 2</h1></th>
  <td align="top">Texto</td>
  <td align="baseline">Texto</td>
  <td align="center">Texto</td>
  <td align="bottom">Texto</td>
 </tr>
</table>
```

Se pueden crear tablas con celdas que abarquen 2 o mas celdas de renglones o culmnas de la tabla con los atributos `rowspan` y `colspan`. Por ejemplo:

```
<table border>
<caption>Muestra del atributo colspan</caption>
 <tr>
    <td>celda 1</td>
    <td colspan="2">celda 2</td>
 </tr>
 <tr>
    <td colspan="2">celda 4</td>
    <td>celda 6</td>
 </tr>
 <tr>
    <td>celda 7</td>
    <td>celda 8</td>
```

```
    <td>celda 9</td>
 </tr>
</table>

<table border>
<caption>Muestra del atributo rowspan</caption>
 <tr>
    <td>celda 1</td>
    <td rowspan="2">celda 2</td>
    <td>celda 3</td>
 </tr>
 <tr>
    <td>celda 4 <br>celda 4</td>
    <td>celda 6</td>
 </tr>
 <tr>
    <th>celda 7</th>
    <th>celda 8</th>
    <th>celda 9</th>
 </tr>
</table>
```

Finalmente, como se muestra en estos últimos ejemplos, se puede utilizar la etiqueta `<caption>` para asignarle un título a la tabla.

2.3 Ultimas consideraciones

Evidentemente faltan muchas etiquetas por menscionar aqui, sin embargo se han seleccionado las más representativas para el desarrollo de los documentos HTML. Sin embargo, en el último capítulo abordaremos el manejo de la etiqueta `<form>` y otras técnicas para proporcinar dinamismo a los documentos HTML.

Capítulo 3

Arquitectura de Información

Cuando se desarrollan sitios Web, el personal involucrado (programadores, editores, diseñadores gráficos, etc.) se enfrenta a una aplia gama de problemas a resolver, desde los técnicos, hasta los de edición de texto y el formato de las páginas. A grandes razgos, los desarrolladores se enfrentan a problemas de tres tipos:

1. Problemas técnicos: como son ancho de banda[1], conocimiento de HTML, scripts, CGI's, y bases de datos, por menciar algunos.

2. Problemas de diseño gráfico. Como son composiciones deficientes, uso inapropiado de colores, mala redacción en el contenido, hetereogenidad en el diseño de las diferentes páginas y/o documentos que forman el sitio Web.

3. Problemas adicionales. Que incluyen principalmente la navegación dentro del sitio.

Recuerde el lector que hay diferentes aspectos que desagradan de un sitio Web (y que son la principal causa de su fracaso). Cuando diseñe y desarrolle su sitio siempre tenga presente estos aspectos molestos:

- No se puede o es difícil encontrar la información que se desea.

- El diseño gráfico y la composición de las páginas son deficientes.

[1]Denominamos ancho de banda a la velocidad en que tarda en mostrarse una página solicitada

- Hay un uso excesivo de efectos especiales de animación y sónido.
- No se habla el lenguaje del usuario del sitio.
- Diseño centralizado en el Webmaster (¡ no sea egocéntrico!!!).
- Uso de páginas con letreros *"En construcción"*.
- Falta de atención a los detalles del uso de HTML y de la verificación del funcionamiento de las diferentes ligas que están en las páginas.

Por otro lado existen los aspecto pósitivos que se deben buscar (y que lamentablemente nadie repara en estos). Estos puntos son:

- Estética.
- Grandes ideas en la asociatividad de elementos gráficos y metafóricos (encontrar páginas que logran este punto es muy raro).
- La información que se presenta en el sitio es de *utilidad* para el usuario.
- Hay facilidad de búsqueda debido a la organización del sitio y a las herramientas de búsqueda que incluya.
- Darle al usuario la posibilidad de personalizar el sitio (esto se logra con diseãndo el sitio Web como una aplicación Web).

Una herramienta que puede se útil para evitar los problemas en los sitios Web, y lograr los aspéctos positivos es utilizar la arquitectura de la información —en realidad puede sonar un poco excesivo este térnimo, pero es un concepto muy antiguo que se bada en la clasificación y organización de las cosas y conceptos.

Específicamente, la arquitectura de la información se utiliza en el diseño de sitios Web en los siguientes aspectos:

- Clarificar la misión y visión del sitio.
- Determinar el contenido y funcionalidad del sitio.
- Determinar el modo en que los usuarios encontraran la información en el sitio (sistemas de organización, navegación, rotulado y búsqueda).
- Proyectar el crecimiento y/o modificación a futuro.

3.1 Organización de la información

Un problema que siempre se presenta es la manera de organizar la información. Debido a que siempre se organiza siguiendo las reglas del lenguaje y es bien sabido que el lenguaje es ambiguo (es decir existen palabras o términos que tienen varios significados). Por ejemplo, supongamos que deseamos clasificar *"medicina alternativa"*, en ¿qué modalidad la incluimos?, pudiera ser: filosofía, religión, charlatanería, o salud y medicina. Esto nos pone en un verdadero dilema y tal vez la solución mas sensata sea darle la vuelta y dejar que otro lo haga... o ¡mejor! revisar el tipo de usuarios que tendrá el sitio, y sobre este estudio, determinar la solución.

Otro aspecto que se debe cuidar es evitar la hetereogeneidad en el sitio Web que se busca, esto es, darle un formato uniforme a los diferentes documentos que componen el sitio, y dar el mismo grado de granularidad al acceso de la información que se presenta. Lamentablemente el 8% de los sitios que hay en el Web contienen información heterogéenea. A fin de cuentas, hay que buscar que el sitio sea homogéneo.

Otro aspecto importante, recuerde que pueden existir muchos puntos de vista válidos para el sitio, por experiencie se puede recomendar utilizar la *"lluvia de ideas"* para captar los detallesque se nos pueden escapar y evitar que la visión del sitio resulte limitada.

3.2 Organización de un sitio Web

Para organizar un sitio Web, primero debemos considerar las diferencias entre sitio Web, aplicación Web, documento HTML, y página HTML.

Una página HTML es un archivo que contiene código HTML y la información que se desea presentar. El usuario obtiene esta información mediante un navegador, con el formato que indiquen las etiquetas HTML que el archivo tenga. Un documento HTML es un conjutno de varias páginas HTML (en ocasiones sólamente es una, pero ya son raros estos documentos) que contienen información relacionada.

Un sitio Web es un conjunto de documentos HTML, y puede incorporar el uso de aplicaciones Web. Una aplicación Web es un programa distribuido que corre en el Web con el de esquema servidor centralizado. Algunos autores manejan que una aplicación Web es más general que un sitio, ya que el sitio

puede considerarse como una aplicación con elementos estáticos.

Los sitios Web —a diferencia de los documentos que se preparan para revistas, libros u otro material impreso— tienen la característica de poder conectar fácilmente diferentes páginas; tal que, la forma de organizar los documentos y las relaciones que existen entre los elementos de sus partes intervienen en la organización total del sitio.

La organización de un sitio Web de bebe basar en un sistema de organización. Un sistema de organización se compone de los esquemas de organización, y de las estructuras de la organización. Los esquemas de organización tratan de la forma en la que se organiza la información, crearndo grupos y subgrupos de ésta. La estrctura de la organización, trata de la relación de los elementos de un grupo con los diferentes grupos y subgrupos que se han formado.

3.2.1 Esquemas de organización

Un buen esquema de organización puede ser el inicio de un buen sitio Web. Podemos distinguir tyres tipos de esquemas de organización: exáctos, ambiguos e híbridos.

Esquemas de organización exactos

Estos sistemas de organización son los más sencillo de implantar, ya que "dividen la información en secciones bien definidas y excluyentes entre sí"[2]. En este tipo de esquema sobresalen los siguientes:

Alfabético Este esquema de organización tiene como base un índice.

Cronológico Este esquema divide la información por fechas.

Geográfico Este esquema divide la información por áreas geográficas.

Esquemas de organización ambiguos

Estos esquemas de organización dividen la información en categorías. La definición de elementos en categorías puede facilitar su búsqueda a cierto tipo de usuarios. Sin embargo, el pelo en la sopa es la ambigüedad implícita en el lenguaje, lo que ocasiona que estos esquemas sean difíciles de definir y,

en ocasiones, de utilizar. Los esquemas de organización más notables de este tipo son:

Temático La información se organiza por temas o materias.

Funcional La organización está enfocada a grupos de acciones que de alguna forma tienen relación.

Público específico La información está dirigida a un sector de los usuarios de manera explícita.

Metafórico La organización de la información se presenta de manera implícita al relacionar información complejas con ideas simples para el usuario.

Esquemas de organización híbridos

En este esquema se mezclan esquemas de organizión exactos y ambiguos. Un sistema de organización hínrido no es una buena idea, ya que si de por si el lenguaje es ambiguo, este esquema aumenta el grado de ambigüedad. En general se considera un mal esquema de organización si las diferentes divisiones se muestran al usuario con el mismo nivel de importancia, y por consiguiente un sitio con este esquema dará muchos problemas a sus usuarios.

Sin embargo, si es necesario utilizar este esquema de organización una buena solución es dividir en la página secciones donde aparescan agrupados los elementos por esquemas.

3.2.2 Estructuras de organización

La estrctura de organización es un elemento intangible en un sitio Web, sin embargo define gran parte del esquema de navegación que se utilizará. Existen tres tipos básicos de estructuras de organización que se utilizan en un sitio Web: estructura jerárquica, estructurade hipertexto y estructura de bases de datos relacionales.

Estructura jerárquica

Se realizan divisiones mutuamente excluyentes hacia abajo en una estructura de árbol invertido. La estructura jerárquica es el complemento de los esque-

mas de organización exacta. Esta estructura es un buen punto de inicio para el diseño de un sitio Web.

Existen algunas reglas que se deben considerar al utilizar estas estructuras. Se debe buscar un equilibrio entre amplitud y profundidad en este tipo de estructuras. La amplitud es el número de opociones que se tiene en cada nivel de la jerarquía. La profundidad, es el número de niveles que tenga la jerarquía. Generalmente podemos distinguir entre dos tipos de jeráquias problemáticas: *"amplia y poco profunda"*, y *"angosta y profunda"*.

Una jerarquía amplia y poco profunda, presenta el problema de saturación de opciones al usuario. Mientras la jerarquía amplia y poco profunda, presenta el problema de que el usuario debe desplegar varias páginas antes de encontrar la información que busca.

Considere la regla de "siete mas menos dos" para equilibrar el número de opciones que tenga un menú o la lista que defina la amplitud de la jerarquía.

Estructura de hipertexto

La estructura de hipertexto permite combinar elementos de una página con elementos que existan en otras páginas. Esto, permite mezclar información entre los diferentes documentos que conforman un sitio Web, ya que cuando se trabaja con esquemas de organización ambiguos, la información no es excluyente. Se debe utilizar con mucho cuidado para evitar que el usuario se pierda entre estas ligas de comunicación, de hecho se considera buena práctica utilizar la estructura de hipertexto como un complemento de las otras dos estructuras.

Estrcutura de base de datos relacional

El esquema de base de datos se utiliza para definir estructuras similares para diferentes tipos de usuarios, o distintas eventos que ocurren en el tiempo. Es decir, se apoyan de una base de datos para generar el contenido de las páginas de manera automática, sin tener que generar distintas versiones de páginas con estrusturas similares.

3.3 Sistemas de navegación

Los sistemas de navegación permite que el usuario no se pierda en el sitio, mientras busca la información que necesita. El esquema jerárquico evita que el usuario se extravíe, a la vez que busca de manera congruente su información. Sin embargo muchas veces se requieren de sistemas de navegación alternativos para brindar contexto y proporcionar una mayor facilidad de búsqueda y movimiento en el sitio.

Un sitio Web contempla distintos sistemas de navegación. Los sistemas de navegación más comunes son: sistema de navegación jerárquicos, globales, local, *ad hoc*

Jerárquicos Los sistemas de bavegacón jerárquicos establecen ligas entre los documentos como si se tratase de un árbol invertido. Se excluyen las relaciones con las demás ramas, amenos que se retroceda en la jerarquía.

Globales Estos sistemas de navegación están dirigidos como complemento de la navegación jerárquica, haciendo énfasis en la navegación lateral (de los elementos que se encuentran en el mismo nivel de la jerarquía) y latera (hacia elementos significativos que se encuentran en un nivel más bajo o alto de la jerarquía, por ejemplo hacia una rama que inicie una jerarquía).

Estos sistemas de navegación generalmente comprenden las ligas que se colocan en la parte inferior de una página. En cada nivel se deben cambiar.

Locales Cuando se trabaja con un sitio Web grande que involucre la integración de subsitios, es posible que se tenga un sistema de navegación para el subsitio, sin embargo, este subsitio debe tener referencias hacia la página principal y/o hacia otras páginas de interés del sitio global.

ad hoc Este sistema de navegación son las ligas que no estan colocadas en una región de la página de una manera determinada, sin embargo se encuentran insertadas en el texto. Esto permite mayor flexibilidad para manejar esquemas de organización muy ambiguos. Estas ligas van incluidas en el texto. Se debe tener cuidado de no saturar el texto con este tipo de navegación. Además es muy probable que el usuario no repare en la existencia de las ligas integradas.

Capítulo 4

Arquitecturas de sitios Web

El desarrollo de aplicaciones a nivel distribuido se ha incrementado notablemente. Este crecimiento se debe en gran medida al éxito que ha tenido Internet. Diferentes aplicaciones se han adecuado y desarrollado en esta plataforma de trabajo, principalmente de aplicaciones que trabajan con bases de datos o que intentar dar un contenido dinámico a los diferentes documentos HTML.

4.1 Arquitectura de trabajo

Cuando se trabajan con sitios de Internet grandes, a menudo es prefereible trabajar con una base de datos para mantener la información que se desa publicar de manera ordenada. Y después hacer gestiones a la base de datos para generar páginas HTML de manera dinámica. Esta es la razón por la hay que realizar mucha programación del lado del servidor, de hecho los CGI's resuelven este problema. Estrategias para CGI's hay muchas, entre las que destacan: ASP, PHP, Perl, ShellScript, CGI con C y C++, JavaServer Pages, Servlets y WebObjects. Sin embargo, en todos los casos, la parte fuerte se realiza cuando interactúan con una base de datos.

El trabajo con la base de datos, por otro lado, es otro factor a considerar, ya que ésta debe mostrar su rendimiento para que el tyrabajo con los CGI's sea rápido. El rendimiento de la base de datos depende tanto de su diseño, como de la configuración y administración del manejador de base de datos que se utilice (llámese ORACLE, Sybase o Informix, por mencionar unos

cuantos).

Cuando se diseña una base de datos se toman en consideración varios factores, entre los que destacan el tipo de información que se requiere almacenar la plataforma de trabajo y las herramientas de diseñoo y desarrollo que se tienen o se pueden adquirir. Afortunadamente existen diferentes métodos para almacenar y recuperar información.

La tarea más difícil y tediosa en el dieño de la base de datos es decidir la manera en la cual se almacenará y se recuperará la información. Se tiene que cuidar muchos aspectos en esta etapa, por ejemplo: darle a la base de datos la posibilidad de mejorar y/o crecer.

Una solución a este problema es trabajar con los recursos que ofrece Internet. En este esquema algún usuario hace una solicitud desde una página HTML que necesita información de una base de datos. Para responder a estas solicitudes se ejecutan varios pasos. El servidor Web recibe la solicitud del visitante, entonces envía la información necesaria a un programa CGI. El programa CGI actúa como puente de enlace entre dos sistemas completamente diferentes para que trabajen juntos. El programa CGI ejecuta la consulta que solicita el visitante, recibe el resultado de la base de datos, genera una respuesta adecuada y la envía al servidor Web, el cual, a su vez, la envía al visitante.

Se observa que forzosamente se tienen dos servidores interactuando, el servidor Web (que es el que recibe y responde las solicitudes del usuario) y el servidor de base de datos (que es el encargado de proporcionar los datos que el usuario necesita). Existen diversos pasos involucrados en este proceso. El objetivo de los Webmasters es juntarlos para que trabajen de forma transparente para los usuarios. Muchos temas se encuentran involucrados para poder desarrollar este tipo de aplicaciones; entre las que destacan, el manejo de HTML para la creación de páginas Web , el uso del lenguaje de programación sobre el cual se escriba el CGI, el manejo del manejador de base de datos, el uso de CGI's con los Servidores Web, e implícitamente, el manejo de procesos en el sistema operativo que se trabaje.

4.1.1 Formas de trabajo de las aplicaciones de Internet

Las aplicaciones que se desarrollan en el Web trabajan bajo el esquema cliente/servidor. En este esquema una parte del sistema se dedica a realizar solicitudes (cliente) a otro módulo que se encarga de dar respuesta a dichas solicitudes (servidor). La idea principal de las computaciones cliente/servidor es que se cuenta con un depósito central de información (generalmente es una base de datos) que se desea distribuir sobre demanda a algún conjunto de personas o máquinas. Una parte fundamental de este esquema es que el depósito de información está centralizado, de tal forma que las modificaciones realizadas a la información son visibles a todos los usuarios. Le llamaremos servidor al depósito de información, al software que se dedica a distribuirla y a las máquinas donde la información y el software residen. Le llamaremos cliente al software que radica, generalmente, en otra máquina y que se comunica con el servidor para buscar información, procesarla y mostrarla.

Actualmente el Web es un enorme sistema cliente/servidor, debido a que todos los clientes y servidores coexisten bajo la misma red.

Inicialmente las aplicaciones en el Web eran sencillas, las solicitudes de los clientes eran procesadas de una sola forma. Cuando el servidor recibía una solicitud, regresaba al cliente un archivo que tena información que el browser del cliente podía interpretar, formateándolo y mostrándolo en la máquina local. Sin embargo, esta forma de trabajo sólo muestra información y no puede actuar sobre ella, esto es, no se puede modificar o agregar datos a la información almacenada en el servidor.

A grandes razgos, las aplicaciones Web pueden trabajan de tres formas.

1. Estática.- Esta es la forma de trabajo má sencilla y la primera que se utilizó en la Web. En esta forma el servidor responde a cada solicitud del cliente mediante un archivo con información que el navegador del cliente puede interpretar, formateándola y mostrándola en la máquina local. Esta forma de trabajo sólo le permite al cliente visualizar información almacenada en el servidor y no puede, de ninguna manera, actuar sobre ella.

2. Programación del lado del servidor. Esta forma de trabajo permite una mayor interacción por parte del usuario. Ahora se le permitirá al cliente realizar, por ejemplo, transacciones a una base de datos que se

encuentre en el servidor. En esta forma el servidor realiza labores más complejas para responder a las solicitudes hechas por el cliente. Por ejemplo, suponga que el usuario desea registrar su nombre o alguna otra información en una base de datos. Esta solicitud a la base de datos se debe atender en el lado del servidor (recuerde que la información que se agrega o modifica es potencialmente disponible a todos los clientes). Entonces el servidor ejecutará una serie de acciones (a través de un programa) con los datos que le envíe el cliente. Las acciones que realizará dependerán del tipo de solicitud e información que le envíe el cliente. El servidor regresará una página HTML. Este tipo de programación se puede realizar mediante el uso de scripts CGI o de servlets.

3. Programación del lado del cliente. Esta forma de trabajo permite que el cliente ejecute programas que le envía el servidor, como respuesta a una solicitud. Es decir, el servidor envía, en lugar de regresar una o unas páginas HTML, un programa para que el cliente ejecute y genere las páginas que el usuario espera ver. Esto tiene una razón específica: evitar el tráfico en la red y así aumentar la velocidad de respuesta de la aplicación. Sin embargo este método sólo se utiliza con operaciones que no afectan la integridad de los datos almacenados en el servidor. Es decir, cuando el cliente solicita operaciones de lectura sobre los datos del servidor y que transformará a un nuevo dato local. Esta operación de transformación se le delegará al cliente. Por ejemplo, suponga que el cliente solicita una operación que involucra una animación, una animación se muestra como una secuencua de imágenes, y de ninguna forma afectan la integridad de los datos que almacena el servidor. Entonces, el servidor regresa un programa para que el cliente genere la secuencia de cuadros de animación, en lugar de que el servidor los genere y envíe, cada uno, en una página HTML.

Cada una de estas formas de trabajo tiene sus ventajas y desventajas. Sin embargo, reiteramos que la programación más extensa y la que tiene mayor impacto en los sitios Web, es la programación del lado del cliente.

Capítulo 5

Manejo de CGI

La programación del lado del servidor es fundamental para que muchas aplicacioines puedan generar sus páginas HTML de manera dinámica. La programación del lado del servidor se realiza a través de programas llamados CGI. Un CGI es un programa que ejecuta el servidor Web ante una solicitud de un cliente. El CGI tiene la cualidad de mantener una comunicación con el servidsor de Internet, tanto para recibir y mandarle información. En este capítulo mostraremos la forma de trabajo de los CGI's, tanto en la manera de invocarlos desde una página HTML, como en su funcionamiento en el lado del servidor.

5.1 La etiqueta <FORM> de HTML

HTML permite diseñar estructuras que funcionen como interfaces gráficas de usuario (GUI), permitiendo que un usuario accese los servicios que se ofrecen en una página Web. Sin embargo, recordemos que las páginas HTML son estáticas, esto es, no procesan ninguna computación que no sea acomodar texto. Para proporcionar la capacidad de poder ejecutar programas a través de una página HTML se incorpora el manejo de CGI.

Para tener una cooperación entre elementos HTML y de los CGI's se cuenta con los formularios de HTML. El programa CGI se encargará del procesamiento de datos y se usa HTML para representar el formulario a los lectores en sus navegadores.

En HTML, los formularios se producen mediante la etiqueta <FORM>. La

sintaxis de esta etiqueta es como sigue:

```
<FORM ACTION={url}  METHOD={method}>
```

Cualquier código HTML válido, excepto otra etiqueta `<FORM>`, puede ir aqui, hasta la etiqueta de cierre `</FORM>`.

```
<INPUT [TYPE="text"] NAME={name} [SIZE={number}] [VALUE={default}]
[MAXLENGHT={number}]>

<INPUT TYPE="password" NAME={name} [SIZE={number}] [VALUE={default}]
[MAXLENGHT={number}]>

<INPUT TYPE="checkbox" NAME={name} [VALUE={default}] [CHECKED] >

<INPUT TYPE="radio" NAME={name} VALUE={default} [CHECKED]>

<INPUT TYPE="hidden" NAME={name} VALUE={default}>

<INPUT TYPE="submit" [NAME={name}] [VALUE={label}>

<INPUT TYPE="reset" [VALUE={label}]>

<INPUT TYPE="image" NAME={name} SRC={url} [ALIGN={aligment}]>

<SELECT NAME={name} [SIZE={number}] [MULTIPLE]>
  <OPTION [SELECTED] [VALUE={value}]>{text}
 [<OPTION [SELECTED] [VALUE={value}]>{text}...]
</SELECT>

<TEXTAREA NAME={name} ROWS={number} COLS={number}>
  El texto por omision se escribe aqui
</TEXTAREA>
```

La etiqueta `<FORM>` indica que lo que viene a continuación es la definición de un formulario. En el parámetro `ACTION` se pone el URL del programa CGI que se desee. En el parámetro `METHOD` se especifica el método que se utilizará para pasarle los parámetros al programa CGI. Estos métodos pueden ser `GET` o `POST`.

5.2 Trabajando con CGI

Se han desarrollado una gran cantidad de lenguajes de programación, podríamos hacer una taxonomía de estos lenguajes agrupándolos por su uso o propósito en dos grandes conjuntos.

1. Lenguajes de propósito general (C, Pascal, Basic, ...)
2. Lenguajes de propósito específico
    - Simbólico (LISP, ALGOL, REC, PROLOG)
    - Administrativo (COBOL)
    - Formato de texto e imágenes (TeX, PostScript, HTML, ..)
    - Manejo de transacciones de bases de datos (SQL)

Los lenguajes de propósito general se pueden utilizar para el desarrollo de, prácticamente, cualquier tipo de aplicación (actualmente el lenguaje de propósito general más utlizado es C). Estos lenguajes se caracterizan por transformarse a código ejecutable de máquina donde se ejecutan cuando se compilan, lo cual los hace muy rápidos en ejecución —de hecho la velocidad de ejecución de los programas escritos en estos lenguajes depende de la velocidad del procesador y del S.O en el cual trabajen—, contienen instrucciones para el control de flujo (como condicionales y ciclos), dan la facilidad de trabajar y definir varios tipos de datos, permiten separar los programa en funciones y módulos, y tienen la facilidad de hacer llamados al Sistema Operativo (de manera implícita o explícita) a través de instrucciones propias del lenguaje o de su biblioteca de funciones.

Sin embargo el problema con estos lenguajes es que la programación es muy complicada, y por esto cara en desarrollo y mantenimiento, para resolver algunos problemas muy específicos. Para resolver este problema se han desarrollado los lenguajes de proósito específico, estos lenguajes mantienen una programación fácil y rápida para los problemas en los cuales se han desarrollado. El problema con estos lenguajes es que cuando se desea atacar un problema que esta fuera de su dominio, esto es cuando el problema no es del tipo de problemas para el cual se han desarrollado, la resolución resulta ser muy compleja o imposible.

HTML es un lenguaje de comandos simple y limitado para dar formato en pantalla a un texto, hacer ligas, y responder a solicitudes de multimedia. Los comandos de HTML no contienen funciones de control ni puede hacer llamdos al sistema, cosa que si tienen y hacen los lenguajes de propósito general, sin embargo tiene la gran ventaja de de accesar a varios servicios de Internet World Wide Web. A diferencia de HTML los lenguajes convencionales no tienen forma de accesar a los servicios que ofrece Internet. Y no se pueden utilizar como parte de los servicios de Internet debido a que, por el hecho de ser de propósito general, resulta muy complicado, y caro, que realicen las funciones que desempeña HTML.

Sin embargo en muchos casos es deseable incrementar las capacidades de una página Web permitiendo la ejecución de programas. Para resolver las carencias que HTML y los lenguajes de propósito general tienen para este propósito los servidoresde Internet incluyen un espacio en disco, denominado cgi-bin, para establecer un intercambio de información entre el navegador y un programa de propósito general o script .

La Interfaz Común de Puerta de Enlace (CGI) es, como ya mencionamos, un área o espacio de disco, un conjunto de variables y convenciones, que utiliza el servidor de Internet para establecer un intercambio de información entre el navegador y el programa que se ejecuta en el servidor. A los programas que se encuentran en esta área se les denomina programas CGI.

Fundamentalmente, los programas CGI's ejecutan 3 pasos básicos de entrada y salida:

1. Obtener los datos de entrada del servidor: Estos datos se obtienen en forma de variables estandarizadas (de ambiente), datos de formularios y datos de consulta.

2. Proporcionar datos al servidor y navegador cliente para el manejo de información de salida, esta información se envía en el encabezado MIME.

3. Proporciona datos de salida al cliente (browser Web). Esta información se envía como una secuencia de comandos HTML.

5.2.1 Localización de los programas CGI

Normalmente los programas CGI deben almacenarse en el directorio cgi-bin o en algún subdirectorio dentro de éste. Esta área se habilita como parte

de la instalación del servidor de Internet. Actualmente, muchos servidores permiten al administrador especificar el nombre del directorio para los programas CGI y generalmente soportan múltiples directorios para múltiples servidores virtuales (esto es, un servidor físico que actúa como si fuera otros, cada uno con su propio árbol de directorios). La localización exacta la debe proporcionar el administrador del sistema.

Por ejemplo, suponga que tenemos un servidor Web instalado en la trayectoria `/usr/bin/http/Webroot`, entonces la ruta del directorio cgi-bin estaría localizada en `/usr/bin/http/Webroot/cgi-bin`.

5.2.2 Estructura de un programa CGI

Ahora veremos un pequeño ejemplo de un CGI, y discutiremos su comportamiento.

```
/*
 * Programa CGI: hola.c
 * Proposito:    Envia el mensaje HOLA
 */

#include <stdio.h>
#include <stdlib.h>

main()
{
   printf("Content-type:text/html\n\n");
   printf("<HTML>\n");
   printf("<BODY>\n");
   printf("<H1>HOLA MUNDO</H1>");
   printf("</BODY>\n");
   printf("</HTML>\n");
}
```

Para compilarlo utilice escriba la siguiente línea:

```
% cc -o hola hola.c
```

Por lo general los programas CGI toman su entrada de las variables de ambiente y mandan su salida a la salida estandar. La salida del programa

primero debe decirle al browser de que tipo de datos envía, y después manda su resultado, el cual debe estar en un formato que el browser pueda desplegar. Por lo general esta salida se encuentra en formato HTML.

La manera de indicarle al browser el tipo de dato que se le envía es utilizando la siguiente directiva:

```
Content-type: <MIME-type>
```

Los valores que toma `<MIME-type>` por lo general son: `text/html`, `text/plain`, `image/gif`, `image/jpeg`, `audio/basic` y `video/mpeg`. Aunque en realidad, depende de las capacidades del navegador.

La última parte del programa construye una página html.

5.2.3 Ejecución de un programa CGI

Para ejecutar un programa CGI se debe tener en el servidor. El área donde se almacenan los programas CGI se dan de alta en el servidor, generalmente en un directorio denominado `/cgi-bin/`, la mayoría de las veces este es un alias del directorio real de almacenamiento, pero se define cuando se configura el servidor de Web que se este utilizando. Por otro lado algunos servidores requieren que los programas CGI tengan la extensión "`.cgi`", aunque no es general para todos los servidores.

En la máquina servidor, el usuario cliente no es quien ejecuta el programa CGI, en su lugar, es el usuario propietario del servidor de Internet, el que se encarga de ejecutarlo (generalmente, se crea un usuario en el sistema operativo del servidor, que hara las veces de usuario ante el sistema operativo de las acciones que realice el servidor Web).

El programa CGI debe tener habilitados los servicios de lectura y ejecución para que el usuario del servidor Web pueda ejecutarlo.

Una vez que se tenga almacenado el programa y activos sus permisos de lectura y escritura, se procede a ejecutarlo desde un navegador. Esto se realiza desde su campo de diálogo, indicando el URL, directorio CGI y el nombre del programa CGI. Por ejemplo:

```
http://www.toallasfox.com.mx/cgi-bin/hola.cgi
http://148.263.138.116/cgi-bin/hola.cgi
```

5.2.4 Paso de datos a un programa CGI

Un programa CGI debe ser capaz de accesar la información del servidor HTTP, del browser y del usuario, para que funcione adecuadamente. De igual forma, debe ser capaz de regresar información al usuario, en forma de texto u objetos gráficos.

Fundamentalmente, los programas CGI's ejecutan 3 pasos básicos de entrada y salida:

1. Obtener los datos de entrada del servidor: Estos datos se obtienen en forma de variables estandarizadas (de ambiente), datos de formularios y datos de consulta.

2. Proporcionar datos al servidor y browser cliente para el manejo de información de salida, esta información se envía en el encabezado MIME.

3. Proporciona datos de salida al cliente (browser Web). Esta información se envía como una secuencia de comandos HTML.

Los programas CGI se ejecutan en un ambiente muy diferente del de otros programas. Particularmente, no obtienen su entrada de la entrada estandar, por lo que no la manejan de la forma acostumbrada. Los programas CGI obtienen la mayoría de su entrada de las variables de ambiente establecidas por el servidor *http* (las variables de ambiente son parte de una memoria especial nombrada en la mayoría de los sistemas operativos, que se utilizan como paso argumentos entre un proceso padre y un proceso hijo). Algunas de las variables son impuestas por el estándar CGI, otras por el servidor de Web y por los browsers. Sin embargo, en ocaciones es posible pasarle datos al proceso hijo a través de la línea de comandos y de la entrada estándar.

Algunas variables de ambiente contienen información que hace referencia al usuario, estas variable son

HTTP_USER_AGENT Nombre y versión del browser.

HTTP_ACCEPT Formatos MIME que acepta el browser.

REMOTE_HOST Contiene el nombre en texto del host.

REMOTE_ADDR Contiene la direccion IP del host.

Otras variables tienen información referente al servidor y al hardware que éste ejecuta: .

SERVER_SOFT_NAME Nombre y versión del servidor de Web.

SERVER_NAME Nombre del servidor host.

GATEWAY_INTERFACE Revisiones de la especificación CGI que se utiliza en el servidor de Internet

Tambien existen variables que son específicas de la solicitud:

QUERY_STRING Variable que contiene la información que se desea pasar al CGI. Posiblemente sea la variable de ambiente más importante.

SCRIPT_NAME Nombre del programa CGI.

SERVER_PROTOCOL Nombre y número de revisión del cual viene la solicitud.

SERVER_PORT Número de puerto al que llego la solicitud.

PATH_INFO y bf PATH_TRANSLATE Indica la forma de pasarle información a un programa CGI.

CONTENT_TYPE Información del método que se seleccionó para pasarle valores al los CGI.

CONTENT_LENGTH Número de bytes de los datos de entrada.

AUTH_TYPE Especifica la autentificación del usuario.

REQUEST_METHOD Método para manejar la solicitud.

La forma de leer estas variables de ambiente en un programa C es:

```
char * cp;
char * empty = "<empty>";

#define safenv(a) ((cp = getenv(a)) ? cp : empty)
...
printf("HTTP_USER_AGENT = %s\n", safenv("HTTP_USER_AGENT"));
```

Existen dos formas de pasarle argumentos a un CGI, estos se indican desde la opción `METHOD` en la etiqueta `<FORM>` de una página HTML. Los métodos son:

GET Indica al CGI que los valores de entrada se pasana atrvés de la variable de ambiente `QUERY_STRING`.

POST Pasa los mismos datos (y con el mismo formato) de entrada hacia el CGI, pero desde la entrada estándar.

Bibliografía

[1] Chuck Musciano y Bill Kennedy; *HTML la guía completa*; McGraw Hill-O'Relly; 1999.

[2] Louis Rosenfeld y Peter Morville; *Arquitectura de Información para WWW*; McGraw Hill-O'Relly; 2000.

[3] Bruce Eckel; *Thinking in Java*. Second. Edition, 1999.

[4] Marty Hall; *Servlets y JavaServer Pages, Guía Práctica*; Sunmicrosystems-Prentice Hall; 2000.

[5] Kernighan & R. Pike; *The UNIX Programming Environment*; Prentice Hall, Inc; 1984.

[6] Eric Ladd & Jim O'Donnel,et. al.; *Using HTML 3.2, Java 1.1 and CGI*, Platinum Edition, QUE Corporation. 1996.

[7] Mark Felton; *CGI Programming with C++ and C*; Lucent Technologies Inc.; Prentice Hall, 1997.

[8] William E. Weinman; *The CGI Book*; Prentice Hall, 1996.

[9] Marc J.Rochkind, *Advanced UNIX Proamming*, Advanced Programming Institute, Inc, Colorado, Prentice Hall, 1985.