Universidad Carlos III de Madrid

Ingeniería Técnica en Informática de Gestión

Proyecto de Fin de Carrera

Estudio de la Evolución Web y Lenguajes Dinámicos.

Rincón Carrera, Juan Manuel

INDICE

1 INTRODUCCIÓN

El proyecto que se lleva a cabo trata de analizar las diferentes vertientes de tecnologías web poniendo en evidencia su parte evolutiva, sus funcionalidades, su futuro y su previsible lucha por la supervivencia o adaptación para una previsible unificación.

Trataremos el desarrollo web desde un punto de vista evolutivo, y abarcaremos el análisis tanto desde el punto de vista tecnológico con sus inicios estáticos hasta el punto de vista social con sus motivaciones humanas por trascender en el mundo virtual, pasando por los ineludibles intereses económicos.

Empezaremos, como medida de comprensión y en pos de prueba empírica, tratando diferentes lenguajes dinámicos y probando así las funcionalidades que ofrecen. Partiendo de una página en HTML[i] estático (HTML 4.01) comprobaremos qué funciones podemos explotar y qué requisitos cumple. Posteriormente, probaremos lenguajes dinámicos como PHP y PSP (tecnología basada en Python) y analizaremos las funciones ofrecidas y la dificultad y calidad de éstas.

Es de mencionar que CSS será incluido en cada una de las versiones con diferentes lenguajes para gestionar los estilos.

En función de lo estudiado trataremos de preveer la posible evolución de las tecnologías web y sus tendencias así como la cada vez más incipiente participación del factor humano, a nivel social y artístico.

El terreno en el que nos sumergimos es eminentemente cambiante y de momento cuasi caótico. Se han creado unos estándares con el objetivo de unificar las tecnologías existentes y que, cada vez, a niveles superiores son respetados y considerados como autoridad. La causa de esta "desorganización" inicial se debe, en un principio, por la libertad que ofrece internet y la guerra entre empresas proveedoras de navegadores como Microsoft con Internet Explorer y Google con Chrome o Mozilla con Firefox. Cada empresa ha intentado que prevalezca su estándar sobre los demás con el objetivo de posicionarse en el mercado y, poco a poco, van cediendo a estándares con mayor o menor velocidad. Estos estándares son W3C, EcmaScript (estandarización de JavaScript) e IETF. Se espera una nueva versión de HTML, la versión 5, del estándar W3C que ofrece nuevos etiquetas similares a las ya existentes en antiguas versiones de HTML que ofrecen una mayor organización semántica y quitan otras que se han quedado obsoletas sobretodo cediendo terreno a CSS y renovando el énfasis del scripting DOM (que es una manera de estructurar semánticamente la web para poder acceder de una manera más cómoda y rápida a sus diferentes etiquetas, ya sean HTML o XML).

2 OBJETIVOS

El proyecto se ha definido como una consultoría orientada al estudio y análisis del mejor entorno y lenguaje de definición para una empresa ficticia que requiere el diseño y desarrollo de su página web. Partiendo de esta premisa se dispone a analizar el entorno y contexto web global en la actualidad. Se realizará un estudio desde diferentes perspectivas que acabará confluyendo en el objetivo principal.

Se ha considerado interesante tener en cuenta factores sociales que contornan las tecnologías web para poder tener una perspectiva de hacia donde avanzan y porqué son movidos. Para llevar a cabo este entendimiento se ha analizado la historia, las tendencias sociales, artísticas y económicas de la web.

Una vez contextualizado el entorno se ha pasado a analizar la parte técnica del proyecto, su entorno de trabajo, las tecnologías a usar y las tecnologías a analizar.

Con el estudio hecho se ha procedido a desarrollar dos webs simples con los dos lenguajes candidatos (después del estudio de lenguajes y su posterior criba) para analizar en la práctica sus funcionalidades y así poder puntuarlos con el fin de elegir uno para desarrollar el proyecto requerido para la empresa ficticia.

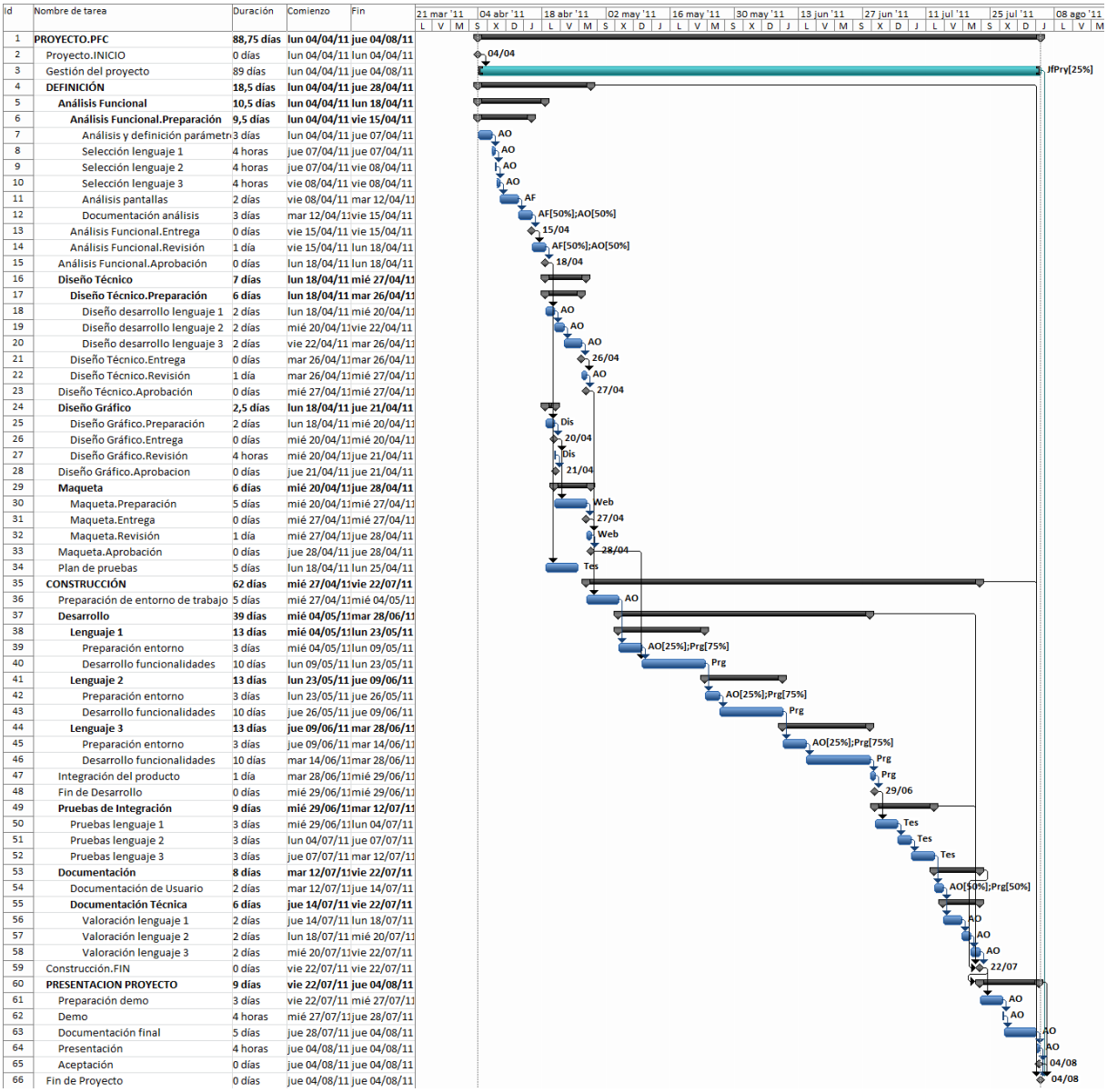
Debido a que para conseguir los objetivos no hay necesidad ninguna de una estructura de base de datos compleja, sino que con una trivial para ver cómo acceder basta no incluiré los Modelos de Datos.

2.1 PLANIFICACIÓN

A continuación, se adjunta una planificación estimada para la realización de este proyecto. Esta planificación ha sido desarrollada con Microsoft Project.

La tarea considerada más importante es el análisis funcional que es donde se establecen las bases del proyecto y que se detallarán más adelante:

- **Análisis y definición de parámetros.**
- **Preselección de lenguajes**.
- **Análisis de la funcionalidad del prototipo**.



Id	Nombre de tarea	Duración	Comienzo	Fin
1	**PROYECTO.PFC**	**88,75 días**	**lun 04/04/11**	**jue 04/08/11**
2	Proyecto.INICIO	0 días	lun 04/04/11	lun 04/04/11
3	Gestión del proyecto	89 días	lun 04/04/11	jue 04/08/11
4	**DEFINICIÓN**	**18,5 días**	**lun 04/04/11**	**jue 28/04/11**
5	**Análisis Funcional**	**10,5 días**	**lun 04/04/11**	**lun 18/04/11**
6	**Análisis Funcional.Preparación**	**9,5 días**	**lun 04/04/11**	**vie 15/04/11**
7	Análisis y definición parámetro	3 días	lun 04/04/11	jue 07/04/11
8	Selección lenguaje 1	4 horas	jue 07/04/11	jue 07/04/11
9	Selección lenguaje 2	4 horas	jue 07/04/11	vie 08/04/11
10	Selección lenguaje 3	4 horas	vie 08/04/11	vie 08/04/11
11	Análisis pantallas	2 días	vie 08/04/11	mar 12/04/11
12	Documentación análisis	3 días	mar 12/04/11	vie 15/04/11
13	Análisis Funcional.Entrega	0 días	vie 15/04/11	vie 15/04/11
14	Análisis Funcional.Revisión	1 día	vie 15/04/11	lun 18/04/11
15	Análisis Funcional.Aprobación	0 días	lun 18/04/11	lun 18/04/11
16	**Diseño Técnico**	**7 días**	**lun 18/04/11**	**mié 27/04/11**
17	**Diseño Técnico.Preparación**	**6 días**	**lun 18/04/11**	**mar 26/04/11**
18	Diseño desarrollo lenguaje 1	2 días	lun 18/04/11	mié 20/04/11
19	Diseño desarrollo lenguaje 2	2 días	mié 20/04/11	vie 22/04/11
20	Diseño desarrollo lenguaje 3	2 días	vie 22/04/11	mar 26/04/11
21	Diseño Técnico.Entrega	0 días	mar 26/04/11	mar 26/04/11
22	Diseño Técnico.Revisión	1 día	mar 26/04/11	mié 27/04/11
23	Diseño Técnico.Aprobación	0 días	mié 27/04/11	mié 27/04/11
24	**Diseño Gráfico**	**2,5 días**	**lun 18/04/11**	**jue 21/04/11**
25	Diseño Gráfico.Preparación	2 días	lun 18/04/11	mié 20/04/11
26	Diseño Gráfico.Entrega	0 días	mié 20/04/11	mié 20/04/11
27	Diseño Gráfico.Revisión	4 horas	mié 20/04/11	jue 21/04/11
28	Diseño Gráfico.Aprobacion	0 días	jue 21/04/11	jue 21/04/11
29	**Maqueta**	**6 días**	**mié 20/04/11**	**jue 28/04/11**
30	Maqueta.Preparación	5 días	mié 20/04/11	mié 27/04/11
31	Maqueta.Entrega	0 días	mié 27/04/11	mié 27/04/11
32	Maqueta.Revisión	1 día	mié 27/04/11	jue 28/04/11
33	Maqueta.Aprobación	0 días	jue 28/04/11	jue 28/04/11
34	Plan de pruebas	5 días	lun 18/04/11	lun 25/04/11
35	**CONSTRUCCIÓN**	**62 días**	**mié 27/04/11**	**vie 22/07/11**
36	Preparación de entorno de trabajo	5 días	mié 27/04/11	mié 04/05/11
37	**Desarrollo**	**39 días**	**mié 04/05/11**	**mar 28/06/11**
38	**Lenguaje 1**	**13 días**	**mié 04/05/11**	**lun 23/05/11**
39	Preparación entorno	3 días	mié 04/05/11	lun 09/05/11
40	Desarrollo funcionalidades	10 días	lun 09/05/11	lun 23/05/11
41	**Lenguaje 2**	**13 días**	**lun 23/05/11**	**jue 09/06/11**
42	Preparación entorno	3 días	lun 23/05/11	jue 26/05/11
43	Desarrollo funcionalidades	10 días	jue 26/05/11	jue 09/06/11
44	**Lenguaje 3**	**13 días**	**jue 09/06/11**	**mar 28/06/11**
45	Preparación entorno	3 días	jue 09/06/11	mar 14/06/11
46	Desarrollo funcionalidades	10 días	mar 14/06/11	mar 28/06/11
47	Integración del producto	1 día	mar 28/06/11	mié 29/06/11
48	Fin de Desarrollo	0 días	mié 29/06/11	mié 29/06/11
49	**Pruebas de Integración**	**9 días**	**mié 29/06/11**	**mar 12/07/11**
50	Pruebas lenguaje 1	3 días	mié 29/06/11	lun 04/07/11
51	Pruebas lenguaje 2	3 días	lun 04/07/11	jue 07/07/11
52	Pruebas lenguaje 3	3 días	jue 07/07/11	mar 12/07/11
53	**Documentación**	**8 días**	**mar 12/07/11**	**vie 22/07/11**
54	Documentación de Usuario	2 días	mar 12/07/11	jue 14/07/11
55	**Documentación Técnica**	**6 días**	**jue 14/07/11**	**vie 22/07/11**
56	Valoración lenguaje 1	2 días	jue 14/07/11	lun 18/07/11
57	Valoración lenguaje 2	2 días	lun 18/07/11	mié 20/07/11
58	Valoración lenguaje 3	2 días	mié 20/07/11	vie 22/07/11
59	Construcción.FIN	0 días	vie 22/07/11	vie 22/07/11
60	**PRESENTACION PROYECTO**	**9 días**	**vie 22/07/11**	**jue 04/08/11**
61	Preparación demo	3 días	vie 22/07/11	mié 27/07/11
62	Demo	4 horas	mié 27/07/11	jue 28/07/11
63	Documentación final	5 días	jue 28/07/11	jue 04/08/11
64	Presentación	4 horas	jue 04/08/11	jue 04/08/11
65	Aceptación	0 días	jue 04/08/11	jue 04/08/11
66	Fin de Proyecto	0 días	jue 04/08/11	jue 04/08/11

2.2 ESTIMACIÓN ECONÓMICA

De la planificación estimada detallada en el punto anterior, se obtienen las horas estimadas que los distintos perfiles dedicarán a las distintas tareas y fases del proyecto. A continuación se detallan estas horas:

Nombre del recurso	Descripción	Trabajo (horas)
JfPry	Jefe de Proyecto	178
Dis	Diseñador Web	20
Web	Maquetador Web	48
AF	Analista Funcional	32
AO	Analista Orgánico	294
Prg	Programador	310
Tes	Test	112
Cl	Cliente	0

Para el cálculo de los costes, se ha realizado una estimación de salarios de cada uno de los perfiles, añadiendo una estimación del coste para una empresa (coste Seguridad Social) y un coste fijo que representa el coste indirecto del perfil (equipo de trabajo, imputación de perfiles de staff). A continuación se detallan estos datos:

Nombre del recurso	Descripción	Salario (1)	Coste SS (2)	Coste fijo	Coste total (3)	Coste/Hora (4)
JfPry	Jefe de Proyecto	45.000 €	13.500 €	5.000 €	63.500 €	37,35 €
Dis	Diseñador Web	25.000 €	7.500 €	5.000 €	37.500 €	22,06 €
Web	Maquetador Web	20.000 €	6.000 €	5.000 €	31.000 €	18,24 €
AF	Analista Funcional	30.000 €	9.000 €	5.000 €	44.000 €	25,88 €
AO	Analista Orgánico	25.000 €	7.500 €	5.000 €	37.500 €	22,06 €
Prg	Programador	20.000 €	6.000 €	5.000 €	31.000 €	18,24 €
Tes	Test	25.000 €	7.500 €	5.000 €	37.500 €	22,06 €
Cl (5)	Cliente	0	0		0	0

(1) Salario bruto anual estimado

(2) Seguridad Social: 30% sobre salario

(3) Suma de salario, coste SS y coste fijo

(4) Coste total dividido por 1.700 horas de trabajo efectivo al año

(5) Se ha utilizado un perfil virtual, llamado cliente que representa las tareas a validar por su parte

Con el cómputo de las horas y el cálculo del coste hora para cada perfil, se calcula el coste total del proyecto, utilizando la siguiente tabla:

Nombre del recurso	Descripción	Trabajo (horas)	Coste/Hora (*)	Coste/Proyecto
JfPry	Jefe de Proyecto	178	37,35 €	6.649 €
Dis	Diseñador Web	20	22,06 €	441 €
Web	Maquetador Web	48	18,24 €	875 €
AF	Analista Funcional	32	25,88 €	828 €
AO	Analista Orgánico	294	22,06 €	6.485 €
Prg	Programador	310	18,24 €	5.653 €
Tes	Test	112	22,06 €	2.471 €
Cl	Cliente	0	0	0
				23.402 €

3 REQUISITOS

Antes de entrar a definir los requisitos específicos voy a comentar el contexto de trabajo en el que voy a desarrollar el proyecto. Después de analizar las diferentes opciones para poder desarrollar la parte técnica del proyecto ha destacado muy notablemente una opción: El entorno LAMP. Es un entorno gratuito, de los más extendidos en Internet para el desarrollo de portales y páginas web, tanto cada uno de sus componentes como la integración de esta cuenta con mucha y variada documentación y sus posibilidades de expansión son enormes. Por todo esto no he tenido ninguna duda al utilizar este conjunto de herramientas que a continuación me dispongo a analizar más detalladamente.

3.1 RESTRICCIONES GLOBALES

LAMP es un acrónimo de un conjunto de soluciones de software libre y código abierto, originalmente acuñado por las primeras letras de Linux (sistema operativo), servidor HTTP Apache, MySQL (software de base de datos) y Perl /PHP / Python, componentes principales para construir un servidor web funcional de propósito general.

La combinación exacta de software que se incluye en un paquete LAMP puede variar, especialmente en lo que respecta al software de scripting web, donde PHP puede ser sustituido o complementado por Perl o Python. Existen términos similares para esencialmente el mismo conjunto de software ('AMP') que se ejecuta en otros sistemas operativos, tales como Microsoft Windows (WAMP), Mac OS (MAMP), Solaris (SAMP), o OpenBSD (OAMP).

Aunque los autores originales de estos programas no los diseñaron específicamente para trabajar unos con otros, la filosofía de desarrollo y los conjunto de herramientas eran compartidas y habían sido desarrolladas en estrecha colaboración. La combinación de software se ha vuelto popular debido a que es gratuito, de código abierto, y por lo tanto fácilmente adaptable, así como a la omnipresencia de sus componentes, que se incluyen con la mayoría de las distribuciones de Linux actuales.

Cuando se usan juntos, forman un conjunto de soluciones tecnológicas que conforman un servidor de aplicaciones.

Una vez encuadrado el contexto de desarrollo, el cual podría considerarse un requisito técnico global los requisitos que a continuación voy a clasificar están sujetos a un punto de vista especial debido a la naturaleza del proyecto. Dado que el proyecto no es una aplicación entregable y con unos objetivos funcionales claros y determinados sino que es un proyecto de investigación cuyo objetivo final es la elección de un lenguaje mediante el conocimiento de las diferentes tecnologías existentes, su evolución, sus capacidades, su posible futuro y por último la comparación entre ellas y una conclusión general que pueda entrever un futuro de la tecnología web desde una posición firme de conocimientos actuales.

Dejada esta premisa clara los objetivos los dividiré entre requisitos funcionales y requisitos no funcionales:

- **Los requisitos funcionales** expresarán la necesidad de satisfacer una funcionalidad concreta, como puede ser la valoración numérica de la potencialidad de los diferentes lenguajes.
- **Los requisitos no funcionales**, en cambio, expresarán necesidades que no se pueden englobar en una funcionalidad concreta, como puede ser el estudio de los diferentes lenguajes.

3.2 Requisitos Funcionales

3.2.1 Análisis de los lenguajes

Se realizará un análisis de cada uno de los lenguajes en base al estudio realizado del mismo. Se analizará tanto su evolución y contexto como las funcionalidades que ofrece y su forma de implementación.

3.2.2 Definición de parámetros a medir

A partir del análisis de los lenguajes se obtendrán los parámetros medibles y por ende, comparables.

3.2.3 Evaluación numérica de los lenguajes

En base a un patrón de calificación se puntuará cada uno de los parámetros del lenguaje.

3.2.4 Comparativa de los lenguajes

Se creará una tabla comparativa de los lenguajes.

3.2.5 Posible evolución de cada lenguaje

Después de analizar cada lenguaje y el entorno web se intentará preveer la evolución de los diferentes lenguajes.

3.2.6 Conclusión sobre la incidencia social del entorno web

A partir del estudio realizado se concluirá cómo ha influido el entorno social en la evolución web y se intentará prever un futuro desarrollo.

3.2.7 Conclusión sobre la incidencia artística del entorno web

A partir del estudio realizado se concluirá cómo ha influido el entorno artístico en la evolución web y se intentará prever un futuro desarrollo.

3.2.8 Conclusión sobre la incidencia económica del entorno web

A partir del estudio realizado se concluirá cómo ha influido el entorno económico en la evolución web y se intentará prever un futuro desarrollo.

3.2.9 Conclusión general del entorno web

A partir de toda la información adquirida y de los estudios realizados se sacará una conclusión del entorno web.

3.3 Requisitos No Funcionales

3.3.1 Estudio tecnológico de la evolución web

Se realizará un estudio sobre la evolución de la tecnología web desde sus comienzos hasta el momento actual.

3.3.2 Instalación del entorno de trabajo (LAMP)

Se preparará un entorno de trabajo web, incluyendo la configuración de un servidor Apache, la configuración de los pertinentes módulos para el uso de lenguajes y la instalación de diferentes navegadores para probar los resultados.

3.3.3 Selección de los lenguajes a estudiar

Se buscará información sobre los diferentes lenguajes web o relacionados con el entorno web para seleccionar los lenguajes que serán estudiados en este proyecto.

3.3.4 Estudio de cada lenguaje

Se realizará un estudio detallado de cada lenguaje para poder comprender sus funcionalidades, su potencialidad, su facilidad de aprendizaje y su posible evolución.

3.3.5 Estudio social de la evolución web

Se realizará un estudio de la incidencia social en el entorno web y de su posible evolución.

3.3.6 Estudio artístico de la evolución web

Se realizará un estudio de la incidencia artística en el entorno web y de su posible evolución.

3.3.7 Estudio económico de la evolución web

Se realizará un estudio de la incidencia económica en el entorno web y de su posible evolución.

3.4 Requisitos técnicos

3.4.1 Servidor Apache

El **servidor HTTP Apache** es un servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que Behelendorf quería que tuviese la connotación de algo que es firme y enérgico pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la preocupación de su grupo era que llegasen las empresas y "civilizasen" el paisaje que habían creado los primeros ingenieros de internet. Además Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. Era, en inglés, *a patchy server* (un servidor "parcheado").

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en su cuota de mercado en los últimos años (Estadísticas históricas y de uso diario proporcionadas por Netcraft3).

La mayoría de las vulnerabilidades de seguridad descubiertas y resueltas tan sólo pueden ser aprovechadas por usuarios locales y no remotamente. Sin embargo, algunas se pueden accionar remotamente en ciertas situaciones, o explotar por los usuarios locales malévolos en las disposiciones de recibimiento compartidas que utilizan PHP como módulo de Apache.

La arquitectura del servidor Apache es muy modular. El servidor consta de una sección *core* y diversos módulos que aportan mucha de la funcionalidad que podría considerarse básica para un servidor web. En mi caso, he tenido que instalar dos módulos para que funcione con PHP y con PSP, lo cuales comentaré en la sección dedicada a cada uno de los lenguajes.

3.4.1.1 CONFIGURACIÓN DE APACHE (UBUNTU)

Apache se configura colocando directivas en archivos de configuración de texto plano. El archivo principal de configuración se llama apache2.conf. Además, se pueden añadir otros archivos de configuración mediante la directiva Include, y se pueden usar comodines para incluir muchos archivos de configuración. Todas las directivas deben colocarse en alguno de esos archivos de configuración. Apache2 sólo reconocerá los cambios realizados en los archivos principales de configuración cuando se inicie o se reinicie.

El servidor también lee un fichero que contiene los tipos mime de los documentos; el nombre de ese fichero lo establece la directiva TypesConfig, y es mime.types por omisión.

El archivo de configuración predeterminado de Apache2 es /etc/apache2/apache2.conf. Se puede editar este archivo para configurar el servidor Apache2. Podrá configurar el número de puerto, la raíz de documentos, los módulos, los archivos de registros, los hosts virtuales, etc.

3.4.1.2 OPCIONES BÁSICAS

Esta sección explica los parámetros de configuración esenciales para el servidor Apache2.

Apache2 trae una configuración predeterminada preparada para servidores virtuales. Viene configurado con un único servidor virtual predeterminado (usando la directiva VirtualHost) que se puede modificar, o dejarlo tal cual si sólo tiene un único sitio web, o usarlo como plantilla para servidores virtuales adicionales si tienes varios sitios web. Si se deja solo, el servidor virtual predeterminado funcionará como tu servidor predeterminado, o los usuarios del sitio web verán si la URL que introducen no concuerda con la directiva ServerName de cualquiera de tus sitios personalizados. Para modificar el servidor virtual predeterminado, edita el archivo /etc/apache2/sites-available/default. Si se desea configurar un nuevo servidor o sitio virtual, copia ese archivo dentro del mismo directorio con el nombre que haya elegido. Por ejemplo, sudo cp /etc/apache2/sites-available/default /etc/apache2/sites-available/minuevositio edita el nuevo archivo para configurar el nuevo sitio usando algunas de las directivas que se describen a continuación.

La directiva **ServerAdmin** especifica la dirección de correo del administrador del servidor. El valor por omisión es webmaster@localhost. Cambia esta dirección por alguna a la que le puedan llegar los mensajes que se envíen (si se es el administrador del servicio). Si el sitio web tiene algún problema, Apache2 mostrará un mensaje de error con en la que aparecerá esta dirección de correo para que la gente pueda enviar un informe del error. La directiva se encuentra en el fichero de configuración de su sitio en /etc/apache2/sites-available.

La directiva **Listen** especifica el puerto (y, opcionalmente, la dirección IP) por el que escuchará Apache2. Si no se especifica la dirección IP, Apache2 escuchará por todas las direcciones IP asignadas a la máquina en la que se ejecute. El valor predeterminado de la directiva Listen es 80. Cambiarlo a 127.0.0.1:80 provoca que Apache2 sólo escuche por su dispositivo loopback, de forma que no estará disponible para Internet. Se puede cambiar a 81 (por ejemplo) para cambiar el puerto por el que escucha, o dejarlo tal cual para que funcione normalmente. La directiva se puede encontrar y cambiar en su propio archivo de configuración, /etc/apache2/ports.conf.

La directiva **ServerName** es opcional, y especifica con cuál FQDN (Full Qualified Domain Name, Nombre de Dominio Totalmente Cualificado) responderá el sitio web. El servidor virtual predeterminado no especifica ninguna directiva ServerName, por lo que responderá a todas las peticiones que no se ajusten a ninguna directiva ServerName en otro servidor virtual. Si se acaba de adquirir el dominio ejemplo.com, y se desea asociar a él un servidor Ubuntu, el valor de la directiva ServerName en el archivo de configuración del servidor virtual debería ser ejemplo.com. Se añade esta directiva al nuevo archivo de configuración virtual que se creó previamente /etc/apache2/sites-available/minuevositio).

También se puede desear que el sitio responda a www.ejemplo.com, ya que muchos usuarios asumen que el prefijo www es apropiado. Para ello, se usa la directiva **ServerAlias**. Se puede usar comodines en la directiva ServerAlias. Por ejemplo, ServerAlias *.ejemplo.com hará que el sitio responda a cualquier solicitud de dominio que termine en .ejemplo.com.

La directiva DocumentRoot especifica dónde se debe buscar Apache los archivos que forman el sitio. El valor predeterminado es /var/www. No hay ningún sitio configurado allí, pero si descomentas la directiva **RedirectMatch** en /etc/apache2/apache2.conf, las peticiones se redirigirán a /var/www/apache2-default, que es donde reside el sitio predeterminado de Apache2. Se cambia este valor en el archivo de host virtual del sitio y se crea ese directorio si fuese necesario.

Apache2 no procesa el directorio /etc/apache2/sites-available. Los enlaces simbólicos en /etc/apache2/sites-enabled apuntan a los sitios «disponibles». Se usa la utilidad a2ensite (Apache2 Enable Site) para crear esos enlaces simbólicos, así:

```
sudo a2ensite minuevositio
```

Donde el archivo de configuración de su sitio es /etc/apache2/sites-available/minuevositio. Igualmente, se debe usar la utilidad a2dissite para deshabilitar sitios.

3.4.1.3 OPCIONES PREDETERMINADAS

Esta sección explica la configuración de las opciones predeterminadas del servidor Apache2. Por ejemplo, si se desea añadir un host virtual, las opciones que configuras para el host virtual tienen prioridad para ese host virtual. Para las directivas no definidas dentro de las opciones del host virtual, se usan los valores predeterminados.

El **DirectoryIndex** es la página servida por defecto por el servidor cuando un usuario solicita el índice de un directorio añadiendo la barra de división (/) al final del nombre del directorio.

Por ejemplo, cuando un usuario solicite la página http://www.ejemplo.com/este_directorio, se obtendrá la página **DirectoryIndex** si existe, un listado de directorio generado por el servidor si no

existe pero tiene especificada la opción Indexes, o una página Permiso Denegado si no se cumplen ninguna de las condiciones anteriores. El servidor intentará buscar uno de los archivos listados en la directiva DirectoryIndex y devolverá el primero que encuentre. Si no encuentra ninguno de esos archivos, y está establecida la opción Options Indexes para ese directorio, el servidor generará y devolverá una lista, en formato HTML, de los subdirectorios y archivos del directorio. El valor predeterminado, almacenado en /etc/apache2/apache2.conf, es «index.html index.cgi index.pl index.php index.xhtml». Por tanto, si Apache2 encuentra un archivo en un directorio solicitado que se ajusta a alguno de esos nombres, se mostrará el primero de todos.

La directiva **ErrorDocument** permite especificar un archivo que usará Apache2 para los eventos de error específicos. Por ejemplo, si un usuario solicita un recurso que no existe, se producirá un error 404, y (en base a la configuración predeterminada de Apache2), se mostrará el archivo /usr/share/apache2/error/HTTP_NOT_FOUND.html.var . Ese archivo no está en el **DocumentRoot** del servidor, sino que existe una directiva **Alias** en /etc/apache2/apache2.conf que redirige hacia /usr/share/apache2/error/ las solicitudes dirigidas al directorio /error. Para ver una lista de las directivas ErrorDocument predeterminadas, se usa la orden:

```
grep ErrorDocument /etc/apache2/apache2.conf
```

De forma predeterminada, el servidor escribe los registros de las transferencias en el archivo /var/log/apache2/access.log. Se puede cambiar esto sitio a sitio en los archivos de configuración de su servidor virtual con la directiva **CustomLog**, o también se puede omitirla para aceptar la opción predeterminada, especificada en /etc/apache2/apache2.conf. También se puede especificar el archivo en el que se registrarán los errores, por medio de la directiva ErrorLog, cuyo valor predeterminado es /var/log/apache2/error.log. Estos se mantienen separados de los registros de transferencias para ayudar en la resolución de problemas con su servidor Apache2. También se puede especificar la directiva **LogFormat** (consulta en /etc/apache2/apache2.conf su valor predeterminado).

Algunas opciones son especificadas por directorio en lugar de por servidor. Una de estas directivas es **Option**. Un parrafo Directory es encerrado entre etiquetas XML, como estas:

```
<Directory /var/www/mynewsite>

...

</Directory>
```

La directiva **Options** dentro del parrafo **Directory** acepta un o más de los siguientes valores (entre otros), separados por espacios:

ExecCGI - Permite la ejecución de scripts CGI. Los scripts CGI no serán ejecutados si esta opción no fue escogida.

Muchos archivos no deberían ser ejecutados como scripts CGI. Esto podría resultar muy peligroso. Los scripts CGI deberían mantenerse en un directorio separado fuera de su DocumentRoot, y dicho directorio debería ser el único que tuviese activada la opción ExecCGI. Así está establecido desde el principio, y la ubicación predeterminada para los scripts CGI es /usr/lib/cgi-bin.

Includes - Permite «server-side includes». Éstos, permiten a un fichero HTML incluir otros ficheros. No es una opción muy común, consulte el Apache2 SSI para más información.

IncludesNOEXEC - Permite «server-side includes», pero deshabilita los #exec y #include en los scripts CGI.

Indexes - Muestra una lista formateada del contenido de los directorios, si no existe el DirectoryIndex (como el index.html) en el directorio solicitado.

> * Por razones de seguridad, esto no debería establecerse, y desde luego no en el directorio indicado por DocumentRoot. Se debe habilitar esta opción con cuidado (y sólo para ciertos directorios) sólo si se esta seguro de querer que los usuarios vean todo el contenido del directorio.

Vistas múltiples - Soporte para vistas múltiples negociadas por contenido; esta opción está desactivada de forma predeterminada por motivos de seguridad. Consulte la documentación de Apache2 sobre esta opción.

SymLinksIfOwnerMatch - Solo seguirá enlaces simbólicos si el directorio de destino es del mismo usuario que el enlace.

3.4.1.4 CONFIGURACIÓN DE SERVIDORES VIRTUALES

Los servidores virtuales permiten ejecutar, en la misma máquina, diferentes servidores para diferentes direcciones IP, diferentes nombres de máquina o diferentes puertos. Por ejemplo, puedes tener los sitios web http://www.ejemplo.com y http://www.otroejemplo.com en el mismo servidor web usando servidores virtuales. Esta opción corresponde a la directiva **<VirtualHost>** para el servidor virtual predeterminado y para los servidores virtuales basados en IP. Corresponde a la directiva **<NameVirtualHost>** para un servidor virtual basado en el nombre.

El conjunto de directivas para un servidor virtual sólo se aplican a un servidor virtual particular. Si una directiva se establece a nivel del servidor completo y no se define dentro de las opciones del servidor virtual, entonces se usarán las opciones predeterminadas. Por ejemplo, se puede definir una dirección de correo electrónico para el webmaster y no definir direcciones de correo individuales para cada servidor virtual.

Se establece la directiva **DocumentRoot** apuntando al directorio que contenga el documento raíz (como el index.html) para el host virtual. El DocumentRoot por defecto es /var/www.

La directiva **ServerAdmin** dentro del apartado **VirtualHost** es la dirección de correo electrónico que se usa en el pie de las páginas de error si selecciona mostrar un pie de página con una dirección de correo en las páginas de error.

3.4.1.5 DIRECTIVAS DEL SERVIDOR

Esta sección explica como configurar básicamente un servidor.

- **LockFile** - La directiva LockFile establece la ruta al archivo de bloqueo usado cuando el servidor se compila con la opción USE_FCNTL_SERIALIZED_ACCEPT o USE_FLOCK_SERIALIZED_ACCEPT. Debe almacenarse en el disco local. Debe dejarse a su valor predeterminado, a menos que el directorio de registros esté ubicado en un directorio NFS compartido. Si es éste el caso, debería cambiarse el valor predeterminado a una

ubicación en el disco local y a un directorio en el que sólo tenga permisos de lectura el usuario «root» . **PidFile** - La directiva PidFile establece el archivo en el que el servidor registrará su identificador de proceso (pid). Sobre este archivo sólo debe tener permisos de lectura el usuario «root». En la mayoría de los casos, debería dejarse a su valor predeterminado.

- **User** - La directiva User establece el identificador de usuario utilizado por el servidor para responder a las peticiones. Esta opción determina el acceso de servidor. Todos los archivos inaccesibles para este usuario serán también inaccesibles para los visitantes de tu sitio web. El valor predeterminado para User es «www-data».
- La directiva **Group** es similar a la directiva **User**. Group establece el grupo sobre el que el servidor aceptará las peticiones. El grupo por defecto es también www-data.

3.4.1.6 MÓDULOS DE APACHE

Apache es un servidor modular. Esto supone que en el núcleo del servidor sólo está incluida la funcionalidad más básica. Las características extendidas están disponibles a través de módulos que se pueden cargar en Apache. De forma predeterminada, durante la compilación se incluye un juego básico de módulos en el servidor. Si el servidor se compila para que use módulos cargables dinámicamente, los módulos se podrán compilar por separado y se podrán añadir posteriormente usando la directiva **LoadModule**. En caso contrario, habrá que recompilar Apache para añadir o quitar módulos. Ubuntu compila Apache2 para que permita la carga dinámica de módulos. Las directivas de configuración ser pueden incluir condicionalmente en base a la presencia de un módulo en particular, encerrándolas en un bloque **<IfModule>**. Se puede instalar módulos adicionales de Apache2 y usarlos con su servidor web. Se puede instalar los módulos de Apache2 usando la orden apt-get. Por ejemplo, para instalar el módulo de Apache2 que proporciona autenticación por MySQL, puedes ejecutar lo siguiente en la línea de órdenes de una terminal:

```
sudo apt-get install libapache2-mod-auth-mysql
```

Una vez instalado el módulo, este estará disponible en el directorio /etc/apache2/mods-available. Se puedes utilizar el comando a2enmod para activar el módulo. Se puedes utilizar el comando a2dismod para desactivar el módulo. Una vez que se active el módulo, este estará disponible en el directorio /etc/apache2/mods-enabled.

3.4.1.7 CONFIGURACIÓN HTTPS

El módulo **mod_ssl** añade una importante característica al servidor Apache2 - la habilidad de encriptar las comunicaciones. De esta forma, cuando el navegador se esta comunicando utilizando la encriptación SSL, se utilizará el prefijo https:// al principio del Localizador de Recursos Uniformes (URL) en la barra de direcciones del navegador.

El módulo mod_ssl esta disponible en el paquete apache2-common. Si se tiene instalado este paquete, se podrá ejecutar el siguiente comando en un terminal para activar el módulo mod_ssl:

```
sudo a2enmod ssl
```

3.4.1.8 CERTIFICADOS Y SEGURIDAD

Para configurar un servidor seguro, se utiliza criptografía de clave pública para crear un par de claves pública y privada. En la mayoría de los casos, se envías la solicitud de certificado (incluyendo su clave pública), una prueba de la identidad de tu compañía, y el pago correspondiente a una Autoridad de Certificación (Certificate Authority, CA). La CA verifica la solicitud de certificado y tu identidad, y posteriormente envía un certificado para el servidor seguro.

También se puede crear su propio certificado auto-firmado. Se debe tener en cuenta, no obstante, que los certificados auto-firmados no deben usarse en la mayoría de los entornos de producción. Los certificados auto-firmados no son aceptados automáticamente por los navegadores de los usuarios. Los navegadores solicitarán al usuario que acepte el certificado para crear la conexión segura.

Cuando se tenga un certificado auto-firmado o un certificado firmado por una CA de su elección se necesitará instalarlo en el servidor seguro.

3.4.1.8.1 TIPOS DE CERTIFICADOS

Se necesita una clave y un certificado para trabajar con el servidor seguro, lo que significa que se deberá generar un certificado propio firmado por uno mismo, o comprar un certificado firmado por una CA. Un certificado firmado por una CA proporciona dos capacidades importantes para tu servidor:

Los navegadores (habitualmente) reconocen automáticamente el certificado y permiten establecer una conexión segura sin preguntar al usuario.

Cuando una CA envía un certificado firmado, está garantizando la identidad de la organización que está suministrando las páginas web al navegador.

Muchos navegadores web que soportan SSL tienen una lista de CAs cuyos certificados aceptan automáticamente. Si un navegador encuentra un certificado autorizado por una CA que no está en su lista, el navegador le preguntará al usuario si desea aceptar o denegar la conexión.

Se puedes generar un certificado firmado por uno mismo para el servidor seguro, pero se ha de tener en cuenta que un certificado auto-firmado no proporciona la misma funcionalidad que un certificado firmado por una CA. La mayoría de los navegadores web no reconocen automáticamente los certificados auto-firmados, y éstos además no proporcionan ninguna garantía acerca de la identidad de la organización que está proporcionando el sitio web. Un certificado firmado por una CA proporciona estas dos importantes características a un servidor seguro. El proceso para obtener un certificado de una CA es realmente fácil. A grandes rasgos, consta de:

- Crear dos llaves encriptadas pública y privada.
- Crear una solicitud de certificado basado en la clave pública. La solicitud de certificado contiene información sobre tu servidor y la compañía que lo aloja.

- Enviar la solicitud de certificado, junto con los documentos que prueban su identidad, a una CA. No podemos decir qué autoridad de certificación elegir. La decisión debe basarse en experiencias pasadas, o en las experiencias de amigos o colegas, o simplemente en factores económicos. Una vez se haya decidido por una CA, necesitas seguir las instrucciones que ésta proporcione para obtener un certificado proveniente de ella.
- Cuando la CA esté segura de que tiene todo lo que necesita, enviará un certificado digital.
- Instalar este certificado en el servidor seguro, y soportar transacciones seguras.

Cuando se obtenga un certificado de una CA, o se genere un propio certificado auto-firmado, el primer paso es generar una clave.

Generar una Petición de Firma de Certificado (Certificate Signing Request, CSR)

Para generar la Solicitud de Firma de Certificado (Certificate Signing Request, CSR), se deberá crear una propia clave. Para ello, se puede ejecutar la siguiente orden en la línea de órdenes de una terminal:

```
openssl genrsa -des3 -out server.key 1024

Generating RSA private key, 1024 bit long modulus

.............++++++ ↲

.........++++++ ↲

unable to write 'random state'

e is 65537 (0x10001)↲

Enter pass phrase for server.key:
```

Ahora se puede introducir la frase de paso. Para mayor seguridad, ésta debería contener, al menos, ocho caracteres. La longitud mínima al especificar -des3 es de cuatro caracteres. Se debes incluir números y/o signos de puntuación, y no debería ser una palabra que se pudiera encontrar en un diccionario. Además, se ha de recordar que la frase de paso distingue mayúsculas de minúsculas.

Se vuelve a escribir la frase de paso para verificarla. Cuando se haya vuelto a escribir correctamente, se generará la clave del servidor y se almacenará en el archivo server.key.

* También se puedes ejecutar el servidor web seguro sin una frase de paso. Esto puede ser conveniente porque así no se tendrá que introducir la frase de paso cada vez que se vaya a arrancar el servidor web seguro. Pero también resulta altamente inseguro y comprometer la clave significa también comprometer al servidor.

En todo caso, se puede escoger ejecutar el servidor web seguro sin frase de paso quitando la opción -des3 en la fase de generación, o ejecutando la siguiente orden en una terminal:

```
openssl rsa -in server.key -out server.key.insecure
```

Una vez se haya ejecutado la orden anterior, la clave insegura se almacenará en el archivo server.key.insecure. Se puedes usar este archivo para generar el CSR sin frase de paso.

Para crear el CSR, se ejecuta el siguiente comando en un terminal:

```
openssl req -new -key server.key -out server.csr
```

Se pedirá que se introduzca la frase de paso. Si se la introduce correctamente, se solicitará que se introduzca el nombre de la empresa, el nombre del sitio, la dirección de correo electrónico, etc. Cuando se hayan introducido todos esos detalles, se creará tu CSR y se almacenará en el archivo server.csr. Se puedes enviar ese archivo CSR a una AC para que lo procese. La AC usará ese archivo CSR y emitirá el certificado. Por otra parte, también se puede crear un certificado auto-firmado usando este CSR.

3.4.1.8.2 CREACIÓN DE UN CERTIFICADO AUTO-FIRMADO

Para crear un certificado auto-firmado, e ejecuta la siguiente orden en una terminal:

```
openssl·x509·-req·-days·365·-in·server.csr·-signkey·server.key·-out·server.crt
```

La orden anterior te solicitará que introduzcas la frase de paso. Cuando la hayas introducido, se creará tu certificado y se almacenará en el archivo server.crt.

Si tu servidor seguro se va a usar en un entorno de producción, probablemente necesitarás un certificado firmado por una CA. No se recomienda el uso de certificados auto-firmados.

3.4.1.8.3 INSTALAR EL CERTIFICADO

Se puede instalar el archivo de la clave server.key y el archivo del certificado server.crt o el archivo de certificado enviado por tu CA ejecutando las siguientes órdenes en la línea de órdenes de una terminal:

```
sudo cp server.crt /etc/ssl/certs

sudo cp server.key /etc/ssl/private
```

Se debería añadir las siguientes cuatro líneas al archivo /etc/apache2/sites-available/default o bien al archivo de configuración del servidor virtual seguro. Se debe colocarlas en la sección **VirtualHost**, bajo la línea **DocumentRoot**:

```
SSLEngine on

SSLOptions +FakeBasicAuth +ExportCertData +CompatEnvVars +StrictRequire

SSLCertificateFile /etc/ssl/certs/server.crt

SSLCertificateKeyFile /etc/ssl/private/server.key
```

El HTTPS suele escuchar en el puerto número 443. Se puede añadir la siguiente línea al archivo /etc/apache2/ports.conf:

```
Listen 443
```

3.4.1.8.4 ACCEDIENDO AL SERVIDOR

Una vez se haya instalado el certificado, se debería reiniciar el servidor web. Se puede ejecutar la siguiente orden en la línea de órdenes de una terminal para reiniciar el servidor web.

```
sudo /etc/init.d/apache2 restart
```

Se pedirá que se introduzca la frase de paso. Cuando se haya introducido la frase de paso correcta, se arrancará el servidor web seguro. Se puede acceder a las páginas del servidor seguro tecleando https://su_equipo/url en la barra de direcciones del navegador.

3.5 NAVEGADORES

A la hora de probar la ejecución de las diferentes webs he probado con los tres navegadores más utilizados y que hacen una idea de las diferencias que existen entre ellos. Éstos son los que he elegido:

3.5.1 FIREFOX

Mozilla Firefox es un navegador web libre y de código abierto descendiente de Mozilla Application Suite y desarrollado por la Fundación Mozilla.11 Es el segundo navegador más utilizado

de Internet con más de 450 millones de usuarios;12 posee una cuota de mercado que se sitúa aproximadamente entre el 19% y 30% para finales de junio de 2011, dependiendo de la fuente de medición global, con particular éxito en Alemania y Polonia, países donde es el más popular con un 60% y 47% de uso, respectivamente.

Para visualizar páginas web emplea el motor de renderizado Gecko, el cual implementa estándares web actuales además de otras funciones destinadas a anticipar probables adiciones a los estándares.

Sus características incluyen navegación por pestañas, corrector ortográfico, búsqueda progresiva, marcadores dinámicos, un administrador de descargas, navegación privada, navegación con georreferenciación, aceleración mediante GPU,21 e integración del motor de búsqueda que desee el usuario. Además se pueden añadir funciones a través de complementos desarrollados por terceros, entre los que hay una amplia selección, lo que según algunos estudios lo convierte en el navegador más personalizable y seguro del momento. Esto ha aumentado significativamente la comunidad de usuarios del navegador.

Es multiplataforma, estando disponible para varios sistemas operativos como Microsoft Windows, GNU/Linux, Mac OS X, FreeBSD,25 y en muchas otras plataformas. La última versión estable es la 6.0, publicada el 16 de agosto de 2011. Su código fuente es software libre, publicado bajo una triple licencia GNU GPL, GNU LGPL o Licencia Pública de Mozilla.26 Tiene como lema "Vuelve a descubrir la web".

3.5.1.1 CARACTERÍSTICAS

Las últimas características incluyen navegación por pestañas, corrector ortográfico, búsqueda progresiva, marcadores dinámicos, un administrador de descargas, navegación privada, navegación con georreferenciación y un sistema de búsqueda integrado que utiliza el motor de búsqueda que desee el usuario. Además se pueden añadir funciones a través de complementos desarrollados por terceros, entre los que hay una amplia selección, lo que según algunos estudios lo convierte en el navegador más personalizable y seguro del momento. Esto ha aumentado significativamente la comunidad de usuarios del navegador.

Los usuarios pueden personalizar Firefox con las extensiones y temas. Mozilla mantiene los repositorios de extensiones en addons.mozilla.org, con más de 6000 complementos a partir de junio de 2009.

Firefox proporciona un entorno para los desarrolladores web en el que se puede utilizar herramientas incorporadas, como la Consola de errores o el Inspector DOM, o extensiones como Firebug.

3.5.1.1.1 COMPATIBILIDAD CON ESTÁNDARES WEB

Resultado de la prueba de estándares web Acid3 en Firefox 4.0

Mozilla Firefox es compatible con varios estándares web, incluyendo HTML, XML, XHTML, SVG 1.1 (parcial), CSS 1, 2 y 3, ECMAScript (JavaScript), DOM, MathML, DTD, XSLT, XPath, e

imágenes PNG con transparencia alfa. Firefox también incorpora las normas propuestas por el WHATWG,91 92 y es compatible con el elemento HTML canvas.

En cuestión al cumplimiento de estándares web de Acid2 y Acid3, Firefox pasa satisfactoriamente la prueba de Acid2 a partir de la versión 3.0. Sin embargo, las ramas de versiones 3.x no pasan completamente la prueba de Acid3, pues obtienen un puntaje de 93/100 en Firefox 3.5, y un puntaje de 94/100 en la versión 3.6. En las versiones de desarrollo Beta, las 2 últimas RC y en la versión final, Mozilla Firefox 4.0 obtiene un puntaje de 97/100 en Acid3.

3.5.1.1.2 SEGURIDAD

Soporta el sistema SSL/TLS para proteger la comunicación con los servidores web, utilizando fuerte criptografía cuando se utiliza el protocolo https. También proporciona apoyo a las tarjetas inteligentes para fines de autenticación. Cuenta con una protección antiphishing, antimalware e integración con el antivirus. También y como medida prudencial que ha causado controversia, Firefox no incluye compatibilidad con los sistemas ActiveX, debido a la decisión de la Fundación Mozilla de no incluirlo por tener vulnerabilidades de seguridad.

3.5.1.1.3 LOCALIZACIONES

Es el navegador web más localizado hasta la fecha, cubriendo el 97% de la población con conexión a internet. El primer lanzamiento oficial en noviembre de 2011 fue distribuido en 28 diferentes lenguajes, incluyendo inglés británico/inglés estadounidense, español europeo/español argentino y chino en caracteres chinos tradicionales/caracteres chinos simplificados. La versión 3.6 está disponible para 86 localidades y 76 idiomas, y la 5.0 en 83 localidades y 75 idiomas.

3.5.2 CHROME

Google Chrome es un navegador web desarrollado por Google y compilado con base en componentes de código abierto como el motor de renderizado WebKit y su estructura de desarrollo de aplicaciones (framework). Google Chrome es el tercer navegador más utilizado en Internet con una cuota de mercado del 13,45% hasta finales de julio de 2011 según NetMarketShare, 22.14% según Statcounter, y en la conferencia Google I/O 2011 se anunció que posee más de 160 millones de usuarios. Está disponible gratuitamente bajo condiciones de servicio específicas. El nombre del navegador deriva del término usado para el marco de la interfaz gráfica de usuario («chrome»).

Por su parte, Chromium es el proyecto de software libre con el que se ha desarrollado Google Chrome y es de participación comunitaria (bajo el ámbito de Google Code) para fundamentar las bases del diseño y desarrollo del navegador Chrome (junto con la extensión Chrome Frame), además del sistema operativo Google Chrome OS. La porción realizada por Google está amparada por la licencia de uso BSD, con otras partes sujetas a una variedad de licencias de código abierto permisivas que incluyen MIT License, Ms-PL y la triple licencia MPL/GPL/LGPL.1 En esencia, los aportes hechos por el proyecto libre Chromium fundamentan el código fuente del navegador base sobre el que está construido Chrome y por tanto tendrá sus mismas características, pero con un logotipo ligeramente diferente y sin el apoyo comercial o técnico de la compañía Google. De acuerdo

a la documentación para desarrolladores, «“Chromium” es el nombre del proyecto, no del producto, y no debería aparecer nunca entre las variables del código, nombres de APIs, etc. Utilícese “chrome” en su lugar».

El 2 de septiembre de 2008 salió a la luz la primera versión al mercado, siendo ésta una versión beta. Finalmente, el 11 de diciembre de 2008 se lanzó una versión estable al público en general.10 Actualmente el navegador está disponible para la plataforma Microsoft Windows en más de 50 idiomas, y desde el 25 de mayo de 2010 para los sistemas Mac OS X y Linux11.

3.5.2.1 CARACTERÍSTICAS

3.5.2.1.1 SEGURIDAD Y ESTABILIDAD

Las metas primordiales al diseñar el navegador fueron mejorar la seguridad, velocidad y estabilidad que los navegadores existentes ofrecían. Se realizaron también cambios importantes a la interfaz de usuario. Chrome fue ensamblado partiendo de 26 diferentes bibliotecas de código de Google y otras de terceros tales como Netscape.72

3.5.2.1.2 CRÍTICAS

Ha sido criticado por lo que se podrían considerar problemas de seguridad y privacidad:

- RLZ identifier: Una cadena codificada enviada junto con todas las consultas a Google73 o cada 24 horas.
- Un identificador (ID) único («clientID») para identificar al usuario en los registros de accesos. Aunque parece que en las próximas versiones lo eliminarán.
- Una marca de tiempo de cuando fue instalado el navegador.
- Páginas de error alojadas en servidores de Google, cuando no se encuentra un servidor.
- Instalación automática de «Google Updater» (se puede desactivar).
- Precargado de DNS (ya se puede desactivar desde la barra de herramientas del navegador).
- Sugerencias automáticas de búsquedas en la barra de direcciones.
- Sistema de seguimiento de errores que envía información sobre cuelgues del navegador o errores.
- Todas estas funcionalidades han sido eliminadas en el navegador Iron.

3.5.2.1.3 LISTAS NEGRAS

Chrome descarga periódicamente actualizaciones de dos listas negras (para sitios de suplantación de identidad y para aquellos que contengan software malicioso) y advierte a los usuarios

cuando intenten visitar una página de contenido peligroso. Este servicio también está disponible para su uso por terceros a través de un API público y gratuito llamado «Google Safe Browsing API». En el proceso de mantenimiento de estas listas negras, Google también notifica a los propietarios de los sitios enumerados que pueden no ser conscientes de la presencia de los programas dañinos.

3.5.2.1.4 AISLAMIENTO DE PROCESOS (SANDBOXING)

Imagen de una pestaña triste al «colgarse» una pestaña del navegador, bajo Ubuntu.

El equipo de desarrollo a cargo de Gears estaba considerando la posibilidad de un navegador multiproceso (cabe señalar que un problema con las implementaciones actuales para navegadores web es que son inherentemente de un único proceso) y Chrome ha aplicado este concepto con arquitectura de multiprocesamiento similar al que utiliza Internet Explorer 8. Se asigna a cada tarea (por ejemplo, pestañas, plugins) un proceso separado. Esto evita que las tareas se interfieran unas a otras, lo que es bueno para la seguridad y la estabilidad; cada pestaña en Chrome se aísla (del inglés sandbox, textualmente caja de arena) para «impedir la instalación de software malicioso» o «impedir que lo que ocurre en una pestaña pueda afectar a lo que sucede en otra», en un modelo de asignación de procesos complejo. Si por algún motivo una mala programación de una página web o un plugin determinado ocasionan un cuelgue, sólo se perjudicará la pestaña actual dejando las demás intactas. Esa pestaña que falla pasará a ser una «pestaña triste», concepto similar a la pantalla de error de versiones anteriores del sistema operativo Mac OS, la sad Mac.

Siguiendo el principio de mínimo privilegio, cada proceso es despojado de sus derechos y con ello no puede escribir ni leer archivos en zonas sensibles (por ejemplo, documentos, escritorio). Este método es similar al «modo protegido» que utiliza Internet Explorer en Windows Vista. El equipo de desarrollo a cargo de «Sandbox» dice haber «convertido las actuales fronteras de los procesos en una cárcel», según sus propias palabras. Por ejemplo, el software malicioso que se ejecuta en una pestaña no puede robar números de tarjetas de crédito, interactuar con el ratón o decirle al sistema operativo que ejecute un programa al inicio del sistema y, además, este proceso desaparecerá cuando la pestaña se cierre. Esto no es más que la imposición de un simple modelo de seguridad informática según el cual hay dos niveles de seguridad multinivel: usuario y caja de arena. La caja de arena solo puede responder a solicitudes de comunicación iniciadas por el usuario.

3.5.2.1.5 ADMINISTRADOR DE TAREAS DE GOOGLE CHROME

Algunos plugins como el Adobe Flash Player no siguen algunos estándares de seguridad y por ello no pueden ser aislados como las pestañas. Estos a menudo necesitan ejecutarse en o por encima del nivel de seguridad del propio navegador. Para reducir la exposición a un ataque, los plugins se ejecutan en procesos separados que se comunican con el motor de renderizado, que a su vez opera con «muy bajos privilegios» en procesos dedicados para cada pestaña. Dichos complementos tendrán que ser modificados para operar dentro de esta arquitectura de software y seguir así los principios de mínimos privilegios.

Además, Chrome cuenta con una utilidad administradora de procesos, denominada Administrador de tareas que permitirá conocer el estado del navegador en su totalidad, ver individualmente estadísticas de sitios web o plugins (los cuales también van en procesos separados) sobre uso de memoria del sistema, ancho de banda usado (velocidad medida en bytes/s) y consumo

de tiempo de CPU, además de dar la opción de poder finalizar la ejecución de cada elemento individualmente.

3.5.2.1.6 MODO INCÓGNITO

El modo incógnito. Con un icono en la parte superior derecha que aparece en una sesión abierta de incógnito.

Chrome incluye un modo de navegar de Incógnito76 (similar a la Navegación privada de Safari y Firefox o el modo InPrivate de Internet Explorer 8) que permite navegar por Internet con total privacidad, ya que no registra ninguna actividad y borra de los archivos temporales las cookies utilizadas. Cuando se activa una de estas ventanas «nada de lo que ocurre en esa ventana saldrá de su computadora».

3.5.2.1.7 VELOCIDAD

La máquina virtual para JavaScript se consideró un proyecto suficientemente importante como para separarlo, tal como se hizo con Tamarin de Adobe/Mozilla, y fue desarrollado por un equipo situado en Dinamarca. Las actuales implementaciones se diseñaron «para los pequeños programas, donde el rendimiento y la interactividad del sistema no eran importantes». Sin embargo, aplicaciones Web como Gmail «están utilizando el navegador web al máximo cuando se trata de manipulaciones DOM y JavaScript». El resultante motor JavaScript V8, fue diseñado poniendo énfasis en la velocidad, e introduce nuevas características a este fin como transiciones de clase ocultas, generación dinámica de código, y recolección precisa de basura (refiriéndose a liberación de memoria). Pruebas de Google demuestran que V8 es aproximadamente dos veces más rápido que Firefox 3 y que la versión beta de Safari (navegador) 4.

Muchos sitios web han realizado pruebas de rendimiento usando la herramienta Benchmark SunSpider para JavaScript así como un conjunto de pruebas de cómputo intensivas propias de Google, las cuales incluyen ray tracing y la resolución de restricciones. De manera unánime reportaron que Chrome rindió mucho más rápido que todos sus competidores con los que había sido comparado, incluyendo Safari, Firefox 3, Internet Explorer 7 e Internet Explorer 8. Aunque el navegador web Opera no fue comparado con Chrome en esas pruebas, en puntuaciones anteriores ha demostrado ser un poco más lento que Firefox 3, el cual a su vez, era más lento que Chrome.

Chrome también guarda y utiliza una caché de direcciones DNS para acelerar la carga de los sitios web.

3.5.2.1.8 INTERFAZ

La interfaz de usuario incluye opciones para ir atrás, adelante, recargar página, ir y cancelar. Las opciones son similares a las del navegador Safari. El diseño de la ventana está basado en el tema nativo de Windows Vista.

Chrome incluye Google Gears, que añade características para desarrolladores que podrían o no convertirse en estándar web, y por lo general relacionadas con la construcción de aplicaciones web (incluyendo soporte para uso sin conexión).

La página de inicio del navegador se sustituye de manera predeterminada por la misma que aparece cada vez que se abre una nueva pestaña. Ésta muestra miniaturas de los nueve sitios web más visitados con las últimas búsquedas realizadas, los últimos marcadores accedidos y pestañas cerradas. Este concepto existía ya antes en el navegador Opera, conocido como «Discado rápido».

Una novedad introducida en Chrome es Omnibox, siendo ésta la barra de direcciones que aparece en la parte superior de cada pestaña, similar a la de Opera. Incluye funcionalidades de autocompletar texto (únicamente autocompletará URLs que se hayan introducido manualmente en lugar de todos los enlaces), sugerencias de búsqueda, páginas visitadas previamente, páginas populares (sin visitar) y búsqueda en el historial de navegación. Los motores de búsqueda también pueden ser capturados por el navegador cuando se utilicen su respectiva interfaz de usuario presionando la tecla Tab ⇆. Otra funcionalidad de la Omnibox es la modalidad de «Pegar y buscar», accesible con el menú contextual.

Chrome permite crear accesos directos en el escritorio que permitan lanzar aplicaciones web directamente en el navegador. Cuando el navegador se abre de esta manera, la ventana no muestra ningún control sino únicamente la barra de título. Esto limita la interfaz del navegador a «no interrumpir cualquier cosa que el usuario está tratando de hacer», lo que permite a las aplicaciones web ejecutarse junto a software local (similar a Mozilla Prism, Adobe AIR y Fluid).

Chrome utiliza el motor de renderizado WebKit como recomendación del equipo que desarrolló Android. Al igual que la mayoría de los navegadores, Chrome fue probado exhaustivamente antes de ser lanzado al público. Las nuevas compilaciones del navegador son probadas de manera automatizada en cientos de miles de sitios web comúnmente visitados, los cuales están en el índice de Google y son accedidos en menos de 20-30 minutos.

Chrome soporta la interfaz de programación de aplicaciones de Netscape (NPAPI), pero no soporta controles ActiveX. Además, Chrome no tiene un sistema de extensiones como el de Mozilla con la arquitectura XPInstall. El soporte para applets de Java se encuentra disponible en Chrome a partir de la actualización del entorno de ejecución de Java 6 actualización 10 o superior.

Para desarrolladores web, Chrome incluye un elemento de inspección similar al que se incluye en la extensión para Firefox, Firebug.

3.5.2.1.8.1 PESTAÑAS

Las pestañas son el principal componente de la interfaz de usuario de Chrome y, como tal, se han movido a la parte superior de la ventana en lugar de por debajo de los controles (similar al navegador Opera). Este es un sutil cambio, en contraste con el actual número de navegadores que se basan en ventanas que contienen pestañas. En Chrome, las pestañas son «flexibles» (incluyendo su estado) y puede ser transferidas sin problemas entre varias ventanas mediante arrastre. Cada pestaña tiene su propio conjunto de controles, entre ellos la barra de direcciones Omnibox. Las ventanas emergentes (en inglés pop-ups) «están confinadas a las pestañas de las que provienen» y no aparecen por encima de la pestaña actual sino como una pequeña barra de título en la parte inferior; podrán usarse cuando el usuario explícitamente las arrastre hacia afuera. Las ventanas emergentes no se ejecutan en su propio proceso.

De forma predeterminada, no hay barra de estado, a diferencia de otros navegadores que muestran una en la zona inferior de la ventana de navegación. Sin embargo, si el cursor del ratón se mueve sobre un enlace, la dirección de este se mostrará en la parte inferior izquierda de la pantalla. Al igual que otros navegadores web como Internet Explorer o Firefox, Chrome tiene un modo de pantalla completa, accesible mediante la tecla F11.

3.5.2.1.8.2 TEMAS VISUALES

A partir de la versión Chrome 3.0 beta92 para Windows, se agregó el soporte para la opción de agregar y cambiar temas visuales en la interfaz de Chrome. En la versión para desarrolladores de Mac OS y Linux también se encuentra funcionando esta opción. Google habilitó una galería de temas visuales93 para Chrome 3.0. La galería hasta el momento contiene 29 temas creados por Google, y 94 temas creados por variados artistas. No se requiere reiniciar el navegador para aplicar el tema, se instala automáticamente.

3.5.2.1.9 EXTENSIONES

El soporte de extensiones creadas por usuarios o la compañía, es soportado de manera predeterminada en Google Chrome desde la versión 4.0 para los sistemas operativos Windows, Linux y Mac OS X. Las extensiones en Chrome se encuentran disponibles en modo de galería para su rápida instalación,94 con más de 300 extensiones disponibles en su día de lanzamiento. Actualmente la galería de extensiones de Google Chrome posee 8.000 extensiones (Chrome Extensions). No se requiere reiniciar el navegador para aplicar alguna extensión, se instalan automáticamente. A partir de Google Chrome 5.0, se pueden utilizar las extensiones en modo incógnito y se pueden seleccionar individualmente para su uso en el modo incógnito.

3.5.2.1.10 SCRIPTS GREASEMONKEY

A partir del 1 de febrero de 2010, se habilitó el soporte nativo para scripts soportados por Greasemonkey, sin la necesidad de tener una tercera extensión dando soporte de ello. Con esta característica es posible utilizar los scripts creados en el sitio UserScript.org. Según el equipo de desarrolladores de Chromium,95 actualmente se encuentran disponibles 40.000 scripts en el sitio UserScript.

3.5.2.1.11 SINCRONIZACIÓN EN LA NUBE

En la versión actual y estable de Google Chrome, se pueden sincronizar los temas visuales, preferencias, marcadores, autocompletado de formularios, extensiones y aplicaciones, todo ello gracias a la computación en la nube (o computación en línea). Con ello se podrán utilizar los mismos temas visuales, preferencias, marcadores, autocompletado de formularios, extensiones y aplicaciones en cualquier computador, gracias a la implementación XMPP, la misma implementación que se utiliza

actualmente en el chat de Gmail. Solamente hay que poseer una cuenta de Google o una cuenta de correo de Gmail para tener acceso al servicio de sincronización en la nube.

3.5.2.1.12 TRADUCCIÓN DE SITIOS

A partir de Google Chrome 4.1 se agregó la característica de traducción sugerida y automática de sitios Web, todo esto gracias al Traductor de Google y los 52 idiomas que soporta actualmente. El sistema cuenta con una opción para desactivar esta característica por si resulta muy invasivo.

3.5.2.1.13 GEOLOCALIZACIÓN

En la última versión de Google Chrome 5.0, se encuentra disponible la geolocalización, aprovechando las habilidades que posee HTML5 para lograr ésta tarea. La geolocalización es útil en sitios sociales para mostrar dónde se encuentra el usuario, y/o compartir imágenes o videos de donde se encuentra el usuario. Esta característica ya funciona con Google Maps.

3.5.2.1.14 FLASH INTEGRADO

En la última versión estable de Google Chrome 5.0, se comenzó con la integración de la última versión del complemento Flash Player dentro del mismo navegador, recibiendo la colaboración y ayuda de Adobe Systems Incorporated (la empresa creadora de Flash). Esto facilita aún más el manejo de objetos incrustados en sitios Webs que utilicen Flash. Además, facilita la actualización del complemento directo desde el actualizador del navegador, y no por separado.

3.5.2.1.15 WEBM Y VP8

En la última versión estable de Google Chrome 6.0, se comenzó con la implementación del proyecto abierto WebM, el cual recolecta herramientas libres y abiertas para la reproducción de contenido multimedia en la Web. WebM está compuesto por el codec de video de código abierto VP8, el códec de audio libre Vorbis, y el contenedor multimedia de estándar abierto Matroska. YouTube en su sitio de pruebas para HTML5, ya está haciendo uso de WebM en sus videos, y cada vez más se pueden encontrar videos en YouTube usando WebM.

3.5.2.1.16 VISOR PDF

En la versión 6.0 de Google Chrome se implementó un visor de archivos PDF, en el que se puede aumentar o disminuir la página y buscar palabras del mismo modo que una página web común. Para protección del usuario, el visor está incluido dentro del «sandbox» de seguridad de

Google Chrome. Al igual que el complemento de Adobe Flash integrado, también recibe las últimas actualizaciones para soporte de archivos PDF.

3.5.2.1.17 APLICACIONES WEB

En la conferencia anual Google I/O 2010, se publicó un adelanto de lo que sería la tienda de aplicaciones Chrome Web Store, la cual sería la encargada de distribuir las aplicaciones web para el navegador Chrome. Eso, hasta que el día 7 de diciembre de 2010, se lanzó la tienda tras el lanzamiento de Chrome 8. Consiste en una tienda en línea que recolecta aplicaciones creadas en diferentes lenguajes de programación que son utilizados en la web, los cuales son: HTML, XHTML, JavaScript, CSS, Adobe Flash, Java, AJAX, HTML5 (video/audio incrustado), WebGL, y CSS3. Las aplicaciones se pueden utilizar conectado a Internet, o sin una conexión a Internet. Las aplicaciones de Chrome Web Store también se pueden usar en otros navegadores, siempre y cuando tengan tecnologías web actuales.

3.5.2.1.18 WEBGL 3D

Uno de los proyectos más ambiciosos del equipo de Chromium/Chrome, el cual brinda aceleración de gráficos tridimensionales vía hardware para juegos o videos para ser usados directos en el navegador. Todo este proyecto funcionará gracias a las herramientas que posee HTML5 y el proyecto WebGL. WebGL funciona sin problemas en Linux y Mac OS X, pero en Windows hay un pequeño percance, así que el equipo de Chromium ideó una forma de utilizar WebGL en Windows con el proyecto llamado Angle, el cual utilizará las bondades de DirectX para utilizar WebGL en Windows.

3.5.2.1.19 BÚSQUEDA INSTANTÁNEA

En septiembre de 2010 se añadió a Google Chrome la posibilidad de buscar instantáneamente en Google desde el propio navegador (Chrome Instant). Esto permite que al momento de escribir el nombre de algún sitio en la barra de direcciones (Omnibox), el sitio cargue mientras todavía se está tecleando, sin necesidad de pulsar la tecla ↵ Entrar.

3.5.2.1.20 OPCIONES OCULTAS

Google Chrome tiene algunas funciones escondidas, no accesibles dentro de la interfaz del usuario. Para acceder a ellas, se pueden teclear diversas órdenes en la barra de direcciones, con los siguientes resultados:

- about: - Aparece información sobre el navegador y su versión.
- about:version - Similar al anterior.
- about:sync - Muestra información y detalles de la sincronización de marcadores.

- about:plugins - Muestra información sobre los plugins instalados.
- about:memory - Los procesos del navegador en funcionamiento, como la memoria empleada por Chrome y otros navegadores abiertos simultáneamente.
- about:cache - Muestra el contenido de la caché.
- about:dns - Obtención de registros producidos por DNS.
- about:histograms - Una lista de los histogramas internos de Google Chrome.
- about:shorthang - La pestaña se cuelga durante un instante.
- about:crash - La pestaña se cuelga, mostrando el icono de la pestaña triste.
- about:credits - Créditos de las aplicaciones de código abierto que utiliza Google Chrome.
- about:terms - Condiciones de servicio de Google Chrome.
- about:flags - Experimenta con opciones que no han sido publicadas
- chrome://history/ - Historial de navegación.
- chrome://downloads/ - Descargas realizadas.
- chrome://extensions/ - Extensiones instaladas.
- chrome://favicon/<URL> - Muestra el favicon de la URL indica.
- chrome://thumb/<URL> - Muestra una vista previa de la URL indicada.
- chrome://inspector/inspector.html - Muestra una ventana vacía del inspector (herramienta para inspeccionar elementos de páginas web).
- chrome://newtab/ - Crea una nueva pestaña.
- chrome://settings/ - Cambia las opciones del navegador
- view-source:<URL> - Muestra el código fuente de la URL especificada.
- view-cache:<URL> - Muestra información en caché de la URL especificada.

3.5.2.1.21 OPCIONES EXPERIMENTALES

Opciones experimentales del navegador, quiere decir que en futuras actualizaciones se incorporarán por defecto, o por medio de alguna opción en la interfaz gráfica serán agregadas, ya que se encuentran en etapa de prueba. Para utilizarlas, hay que crear un acceso directo y agregar una de la opciones al final de la cadena de destino, en la opción «propiedades».

- --bookmark-menu - Agrega un botón de menú de marcadores al lado de la barra de direcciones. (Solo Windows)
- --enable-monitor-profile - Convierte las páginas Web de sRGB a tu actual perfil de monitor por defecto.
- --enable-webgl - Habilita WebGL, para gráficos 3D en Web. (Chrome 5)

- --no-sandbox - Deshabilita el Sandbox (aislamiento de procesos) de Chrome.
- --incognito - Habilita el modo incógnito en un acceso directo a Chrome.
- --pinned-tab-count=1 - Permite marcar pestañas al iniciar Chrome, y agregar el número y sitios que serán marcados.
- --omnibox-popup-count=10 - Aumenta el número de sugerencias para una búsqueda en el Omnibox.
- --user-agent="Agente_de_usuario" - Para cambiar el agente de usuario a elección.
- --enable-vertical-tabs - Habilita las pestañas al lado izquierdo de la ventana del navegador. (Windows, Chrome 5.0).

3.5.3 INTERNET EXPLORER

Windows Internet Explorer (anteriormente Microsoft Internet Explorer), conocido comúnmente como IE, es un navegador web desarrollado por Microsoft para el sistema operativo Microsoft Windows desde 1995. Ha sido el navegador web más utilizado de Internet desde 1999 hasta la actualidad, con un pico máximo de cuota de utilización del 95% entre el 2002 y 2003. Sin embargo, dicha cuota de mercado ha disminuido paulatinamente con los años debido a una renovada competencia por parte de otros navegadores, situándose aproximadamente entre el 36% y 60% para finales de junio de 2011, dependiendo de la fuente de medición global.

Su versión más reciente es la 9.0, publicada el 14 de marzo de 2011,5 y está disponible gratuitamente como actualización para Windows Vista SP2 o Windows Server 2008 SP2, además de Windows 7 y Windows Server 2008 R2. Los sistemas operativos Windows XP, Windows 2003 y anteriores no están soportados. Esta nueva versión de Internet Explorer incorpora considerables avances en la interpretación de estándares web respecto a sus precursores, como el soporte para CSS3, SVG, HTML5 (incluyendo las etiquetas <audio>, <video> y <canvas>), el formato de archivo tipográfico web "WOFF", además de incluir mejoras de rendimiento como la aceleración por hardware para el proceso de renderizado de páginas web y un nuevo motor de JavaScript denominado Chakra.

También se han producido compilaciones de Internet Explorer (algunas actualmente descontinuadas) para otros sistemas operativos, incluyendo Internet Explorer Mobile (Windows CE y Windows Mobile), Internet Explorer para Mac (Mac OS 7.01 a 10) e Internet Explorer para UNIX (Solaris y HP-UX).

3.5.3.1 CARACTERÍSTICAS

Internet Explorer ha sido diseñado para una amplia gama de páginas web y para proporcionar determinadas funciones dentro de los sistemas operativos, incluyendo Windows Update. Durante el apogeo de la guerra de navegadores, Internet Explorer sustituyó a Netscape cuando se encontraban a favor de apoyar las progresivas características tecnológicas de la época.

3.5.3.1.1 SOPORTE DE ESTÁNDARE

Internet Explorer, utilizando el motor de diseño Trident, soporta HTML 4.01, CSS 1.0, CSS 2.1 y XML 1, con pequeñas lagunas de contenido. El soporte para gran parte del borrador de estándar CSS3, así como HTML5 está en el proyectado para Internet Explorer 9.

Es totalmente compatible con XSLT 1.0, así como un dialecto de XSLT obsoleto creado por Microsoft al que se refiere a menudo como WD-XSL. Está proyectado soporte para XSLT 2.0 para versiones futuras de Internet Explorer, bloggers de Microsoft han indicado que el desarrollo está en marcha, pero las fechas no se han anunciado.

Internet Explorer ha sido objeto de críticas por su limitado soporte a estándares web abiertos y un objetivo de mayor importancia de Internet Explorer 9, es mejorar el soporte a las normas ya dichas.

3.5.3.1.2 NORMAS DE EXTENSIONES

Resultado de la prueba Acid2 que revisa la conformidad con la norma estándar CSS2 en Internet Explorer 8.

Acid3 en la versión preliminar de Internet Explorer 9. Acid3 prueba la compatibilidad con los lenguajes Document Object Model (DOM) y JavaScript, además de CSS3 y SVG 2.0 (aunque estos últimos no son aún estándares).

Internet Explorer ha introducido una serie de extensiones propietarias de muchas de las normas, incluyendo HTML, CSS y DOM. Esto ha dado lugar a una serie de páginas web que sólo se pueden ver correctamente con Internet Explorer.

Internet Explorer ha introducido una serie de prórrogas a JavaScript que han sido adoptadas por otros navegadores. Estas incluyen innerHTML, que devuelve la cadena de HTML dentro de un elemento, el XML HTTP Request, que permite el envío de la petición HTTP y la recepción de la respuesta HTTP. Algunas de estas funcionalidades no son posibles hasta la introducción de los métodos de DOM inducidos por W3C.

Otras normas que prevé Microsoft son: soporte vertical de texto, pero en una sintaxis diferente a la recomendación de la W3C; soporte para una variedad de efectos de imagen y apoyo al código de secuencia de comandos, en particular JScript Encode. También se prevé soporte a la incrustación de tipos de letra EOT en páginas web.

3.5.3.1.3 USABILIDAD Y ACCESIBILIDAD

Internet Explorer hace uso de la accesibilidad prevista en Windows. Internet Explorer también es una interfaz de usuario de FTP, con operaciones similares a las del Explorador de Windows (aunque ésta característica requiere una ventana que se abre en las últimas versiones del navegador, en lugar de forma nativa en el navegador). Las versiones recientes bloquean las ventanas emergentes e incluyen navegación por pestañas. La navegación con pestañas también puede ser añadida a las versiones anteriores mediante la instalación de las diferentes barras de herramientas, proporcionadas por los principales motores de búsqueda en internet.

3.5.3.1.4 CACHÉ

Internet Explorer guarda archivos temporales de Internet para permitir un acceso más rápido (o el acceso fuera de línea) a páginas visitadas anteriormente. El contenido está indexado en un archivo de base de datos, conocido como Index.dat. Los archivos múltiples que existen son diferentes índices de contenido, contenido visitado, RSS, Autocompletar, páginas web visitadas, las cookies, etc.

Antes de IE7, la limpieza de la caché se utilizaba para borrar el índice, pero los archivos no eran eliminados. Esta característica era un riesgo potencial para la seguridad tanto para los individuos como para las empresas. A partir de Internet Explorer 7, tanto el índice de entradas de los archivos como ellos mismos se eliminan de la memoria caché cuando se borra.

3.5.3.1.5 POLÍTICAS DE GRUPO

Internet Explorer es totalmente configurable mediante directiva de grupo. Los administradores de dominios Windows Server pueden aplicar y hacer cumplir una serie de ajustes que afectan a la interfaz de usuario (por ejemplo, deshabilitar elementos de menú y las opciones de configuración individual), así como las características de seguridad tales como la descarga de archivos, la configuración de la zona, por configuración del sitio, comportamiento de control ActiveX, y otros. La configuración puede ser establecida para cada usuario y para cada máquina. Internet Explorer también soporta autenticación integrada de Windows.

3.5.3.1.6 ARQUITECTURA

Internet Explorer utiliza una arquitectura componentizada en torno al "Modelo de objetos componentes" (COM). Se compone de cinco componentes principales, cada uno de los cuales están contenidos en archivos .dll distintos y exponen un conjunto de interfaces COM que les permite ser usados por el ejecutable principal de Internet Explorer, iexplore.exe:

3.5.3.1.6.1.1 WININET.DLL

Wininet.dll es el manejador de protocolo HTTP y FTP. Se ocupa de todas las comunicaciones de red para estos protocolos.

3.5.3.1.6.1.2 URLMON.DLL

Urlmon.dll es responsable de la manipulación de contenidos basadas en MIME y descarga de contenido web.

3.5.3.1.6.1.3 MSHTML.DLL

MSHTML.dll alberga el motor de renderizado Trident introducido en Internet Explorer 4, que se encarga de mostrar las páginas en la pantalla y el manejo de los DOM de las páginas web. MSHTML.dll analiza el HTML/CSS de los archivos y crea el interior de la representación DOM. También expone un conjunto de APIs para la inspección en tiempo de ejecución y modificación del árbol DOM. Internet Explorer no incluye la funcionalidad nativa de secuencias de comandos. Por el contrario MSHTML.dll expone a otro conjunto de APIs que permiten a cualquier entorno de programación ser conectado en el DOM.18

3.5.3.1.6.1.4 SHDOCVW.DLL

Shdocvw.dll proporciona la navegación, almacenamiento local y funcionalidades para el navegador.

3.5.3.1.6.1.5 BROWSEUI.DLL

Browseui.dll es responsable de la interfaz de usuario del navegador, incluyendo el marco de la interfaz gráfica de usuario (chrome), albergando todos los menús y barras de herramientas.

Internet Explorer 8 presenta algunos cambios arquitectónicos importantes, llamados Loosely Coupled IE, o LCIE). LCIE separa el proceso de la interfaz de usuario del proceso que alberga las diferentes aplicaciones web en diferentes pestañas (procesos por pestaña). Un proceso de la interfaz de usuario puede crear varios procesos a la vez, cada uno de los cuales puede ser de diferente nivel de integridad; cada pestaña puede alojar múltiples sitios web. Cada proceso de pestaña tiene su propia caché de cookies. Los dos procesos usan comunicación asíncrona entre procesos para sincronizarse entre sí. En general, habrá un único proceso por cada pestaña abierta con un sitio web. Sin embargo, en Windows Vista con modo protegido activado, la apertura de contenido privilegiado (como páginas HTML locales) crearán un nuevo proceso, para que no sea limitada por el modo de funcionamiento protegido.

3.5.3.1.7 EXTENSIBILIDAD

Internet Explorer expone también una serie de COMs) que permiten a otros componentes extender la funcionalidad del navegador. La extensibilidad se divide en dos tipos: «extensibilidad de navegador» y «extensibilidad de contenido». La extensibilidad de navegador puede ser utilizada para conectar componentes, añadir entradas de menú contextual, barras de herramientas, elementos de menú o objetos auxiliares del explorador. El contenido puede estar en términos de documentos activos (por ejemplo, SVG o MathML) o controles ActiveX. Los controles ActiveX son utilizados para

los contenidos manipuladores que hacen posible el uso de contenido empotrado dentro de una página HTML (por ejemplo, Adobe Flash o Microsoft Silverlight). Los objetos .doc se utilizan cuando el tipo de contenido no será incrustado en HTML (por ejemplo, Microsoft Word, PDF o XPS).

Los «Add-ons de Internet Explorer» se ejecutan con los mismos privilegios que el navegador mismo, a diferencia de los scripts que tienen un conjunto muy limitado de privilegios. Los add-ons pueden ser instalados de forma local, o directamente por un sitio web. Dado que los add-ons más tienen un acceso privilegiado al sistema, los add-ons pueden y han sido utilizados para comprometer la seguridad del sistema (add-ons maliciosos). Internet Explorer 6 con Service Pack 2 en adelante proporciona diversas herramientas en contra de los add-ons, incluye un Add-on Manager para el control de los controles ActiveX y los objetos auxiliares del explorador y un modo de operación No add-ons, así como mayores restricciones en los sitios web para instalar add-ons.

Internet Explorer puede tener hosting por otras aplicaciones a través de un conjunto de interfaces COM. Esto puede ser usado para incrustar el navegador dentro de la funcionalidad de la aplicación. Asimismo, la aplicación de alojamiento puede elegir sólo a MSHTML.dll, motor de renderizado, en lugar de todo el navegador.

3.5.3.1.8 SEGURIDAD

Internet Explorer utiliza una seguridad basada en zonas y grupos de sitios sobre determinadas condiciones, incluso si se trata de un Internet o intranet basada en web, así como un usuario en la lista blanca. Las restricciones de seguridad se aplican para cada zona; todos los sitios en una zona están sujetos a las restricciones.

Internet Explorer 6 SP2 y posteriores utilizan el Anexo de Ejecución del Servicio de Microsoft Windows para marcar los archivos ejecutables descargados de Internet como potencialmente peligrosos. Esto ayuda a la prevención de accidentes en la instalación de malware.

Internet Explorer 7 incluye un filtro contra suplantación de identidad (phishing), que restringe el acceso a sitios falsos a menos que el usuario anule la restricción. Internet Explorer 8, también bloquea el acceso a sitios conocidos por almacenar software malicioso. Las descargas también son analizadas para ver si son conocidas por estar infectadas.

En Windows Vista, Internet Explorer se ejecuta de manera predeterminada en lo que se denomina Modo protegido, donde los privilegios del navegador en sí están muy restringidos. Se puede, opcionalmente, navegar fuera de este modo, pero no es recomendable. Esto también limita la eficacia de los privilegios de los add-ons. Como resultado de ello, incluso si el navegador o cualquier add-on está en peligro, el daño que puede causar es limitado.

Se liberan periódicamente parches y actualizaciones para el navegador y están disponibles a través del servicio Windows Update, así como a través de Actualizaciones automáticas. Aunque los parches de seguridad siguen siendo lanzados periódicamente para una amplia gama de plataformas, las características más recientes y mejoras de seguridad son liberadas para sistemas basados en Windows XP SP2 y posteriores.

3.5.3.1.9 VULNERABILIDADES DE SEGURIDAD

Internet Explorer ha sido objeto de muchas vulnerabilidades de seguridad y preocupaciones: la mayor parte de spyware, adware, y virus informáticos se transmite través de Internet por la explotación de los fallos y defectos en la arquitectura de seguridad de Internet Explorer, a veces requieren nada más que la visualización de una página web maliciosa para instalar ellos mismos el virus.

Una amplia serie de fallos de seguridad que afectan a IE no se originan en el navegador en sí, sino en los ActiveX utilizados por éste. Debido a que los add-ons tienen los mismos privilegios que IE, los defectos pueden ser tan críticos como un defecto del navegador. Otros navegadores que utilizan NPAPI como su mecanismo de extensibilidad sufren los mismos problemas.

4 ALCANCE

4.1 DESCRIPCIÓN

El proyecto tratará de analizar la evolución web desde HTML estático hasta la adhesión de diferentes lenguajes dinámicos, comprobando que funciones ofrece cada uno de ellos, cual es su facilidad de aprendizaje y su posible evolución, en pos de intentar prever hacia donde se dirige la evolución web y cómo puede cambiar su paradigma en un futuro además de hacer una catalogación en función de su valoración.

La implementación del proyecto se efectuará bajo el conjunto LAMP, que incluye el Sistema Operativo Linux, el servidor Apache, el gestor de bases de datos MySQL y PHP, Python o Perl (este último no lo trataremos). Bajo este contexto y ejecutando las páginas en Firefox y Chrome se irán probando los diferentes lenguajes y valorándose en función de ciertos parámetros que más tarde serán comentados.

La evolución cronológica de éstos pasos parte del estudio de la web y el acercamiento al mismo. La recopilación de informacíon referente a los inicios y la reflexión sobre los orígenes. Una vez comprendido el sistema de funcionamiento y diferenciados los conceptos de lenguaje estático, lenguaje dinámico y hoja de estilo se procederá a la creación de una web sencilla pero que satisfaga ciertos requisitos funcionales que serán objeto de análisis. Como es puede ser la conexión con una base de datos, con el objetivo de un registro de usuarios y la comprobación de logueo, la capacidad de enviar mails o la manipulación de la web de forma dinámica por un usuario registrado.

A partir de la clarificación de estos objetivos se pasará a la implementación en HTML (utilizando una versión anterior a la 5, que se comentará más adelante) de la página comprobando los límites de este lenguaje base. Una vez claros éstos se procederá al aprendizaje de un lenguaje dinámico, PHP, y se modificarán las funciones en HTML que se puedan satisfacer con PHP en pos de abarcar todo lo posible con el lenguaje. Se estudiará la manera de configurar el servidor para utilizar el lenguaje, su forma de implementación, las funcionalidades que es capaz de satisfacer, la facilidad de aprendizaje y su potencionalidad.

Una vez terminada la página web con su parte estática y dinámica se entenderá que se han abarcado todas las funcionalidades requeridas y se procederá al estudio del siguiente lenguaje dinámico, PSP (Python Server Pages), otro lenguaje embebido en HTML basado en Python. Se intentará que la página quede idéntica a la anterior y de la misma manera se valorarán las funcionalidades, su facilidad de aprendizaje y su potencialidad.

4.2 DEFINICIÓN DE FUNCIONALIDADES

A la hora de probar los diferentes lenguajes dinámicos se implementarán una serie de funcionalidades concretas para probar las capacidades de cada uno que a su vez serán puntuados bajo una serie de parámetros que servirán para, más tarde, compararlos entre sí.

Las funcionalidades elegidas se especifican en los siguientes puntos.

4.2.1 CONEXIÓN BBDD

El lenguaje debe de ser capaz de conectarse a una base de datos, en concreto bajo MySQL. La base de datos existirá previamente y tan solo contendrá dos tablas básicas:

Tabla “Usuarios”:

Esta tabla será util a la hora de poder identificar un usuario para hacer el login y comprobar si su password es correcto.

Cuyos campos serán:

“id”,”nombre”,”apellidos”,”login”,”password”,”email”.

Tabla “Dioses”:

Esta tabla servirá para poder crear una de las páginas dinámicamente a partir de su contenido.

Cuyos campos serán:

”nombre”,”descripcion”.

4.2.2 Registro de clientes

Se podrán registrar clientes por medio de un formulario y quedarán almacenados en la base de datos en la tabla “Usuarios”.

4.2.3 Generación dinámica web

Una de las páginas será generada dinámicamente a partir de los datos almacenados en la tabla “Dioses”.

4.2.4 Navegabilidad

Una de las páginas tan sólo podrá ser accedida por usuarios que se hayan logueado correctamente. Para el resto de usuarios no se mostrará la información ni la opción de poder añadir nuevas tuplas a la tabla “Dioses”, que a su vez se mostrarán a partir de entonces en la página dinámica.

4.2.5 Diseño-Estilo

No hay que olvidar que las webs tienen como objetivo final usuarios “normales”, con ésto me refiero a que no sólo son expertos, y tanto para éstos y especialmente para los anteriores es importante que el diseño sea atractivo, llamativo. A la hora de analizar la evolución de la web a lo largo de su historia es notable el progreso que ha tenido especialmente en la vertiente artística y en la semantica. Antes existían miles de páginas de individuos con secciones diferentes, como unas pocas fotos, algo de información de ésto y de lo otro, algunas canciones favoritas y todo esto condimentado con cientos de gifs animados, sonidos y contrastes de colores que hoy en dia serían impensables. Se ha ido evolucionando a páginas web con un entorno semántico concreto y profundo y un diseño agradable y poco cargado, un ejemplo de ésto es posiblemente la página más visitada actualmente, el buscador de Google, un diseño sencillo, en blanco, agradable y con un potente motor de busqueda interno.

Los lenguajes deben poder ofrecer la posibilidad de adaptarse a los nuevos patrones de diseño y vertientes artísticas que sean demandadas por los usuarios y ésto será lo puntuado en esta sección.

4.2.6 Envío Formulario

El uso de formularios es una funcionalidad muy utilizada en las páginas web, ya no solo por la posibilidad de registrarse como usuario (lo cual compromete una base de datos) cuya funcionalidad ya hemos tenido en cuenta, sino con muchas otras alternativas, como por ejemplo mandar una sugerencia o una queja al webmaster. Esta parte intentará tener en cuenta la funcionalidad desde un punto de vista global.

4.2.7 Envío mails

Los lenguajes dinámicos suelen ofrecer funciones que son capaces de hacer funcionar el servidor Apache como un servidor SMTP y añaden así la posibilidad de mandar mails rellenando los campos necesarios.

4.3 Definición de parámetros a medir

En esta sección es de las más importantes del proyecto ya que en ella se van a definir los parámetros que serán puntuados para cada funcionalidad en los diferentes lenguajes de programación y a partir de ello se sacarán las conclusiones finales. Se ha reflexionado hondamente cuales pueden ser los parámetros adecuados y desde que perspectiva deben de ser medidos intentando abarcar de la manera más completa todas las posibles vertientes con las que afrontar un lenguaje desde el punto de vista de quién los desconoce y debe decidir cual aprender y usar dependiendo de los objetivos que tenga. Siendo pertinente evaluar desde la facilidad de aprendizaje para un usuario poco iniciado en la programación, la posible evolución o intento de predicción de la futura vida hasta la potencialidad funcional del lenguaje.

Estos parámetros serán puntuados posteriormente en un baremo del 1 al 10. Siendo el 1 la menor nota y el 10 la mejor.

Los parámetros y una breve explicación de los mismos se expone a continuación:

4.3.1 Curva de aprendizaje

Es importante desde el punto de vista del programador, y más aún si éste es novato, que el lenguaje de programación no sea excesivamente complejo en pos de empezar a ver resultados y poder invertir el tiempo en completar los objetivos propuestos. La pérdida de tiempo en el aprendizaje de una sintaxis compleja o unas funciones poco intuitivas a su vez predestinan a un corto ciclo de vida del lenguaje, ya que la evolución de éstos ha demostrado que los lenguajes complejos no benefician a la hora de poder afrontar un problema y acaban quedando obsoletos.

4.3.2 Potencialidad

Este parámetro se refiere a la eficacia con la que el lenguaje es capaz de llevar a cabo una funcionalidad concreta. Si para llevarla a cabo se necesitan más o menos lineas de código y si se debe recurrir a más o menos librerías. También influyen en éste parámetro la cantidad de librerías existentes a las que el programador puede recurrir.

4.3.3 Evolución

La evolución que ha tenido el lenguaje es importante a la hora de poder estimar los fallos de seguridad que tiene actualmente, la gente que actualmente lo usa (lo cual influye directamente en su obsolescencia), las funcionalidades que se le han ido añadiendo y su adaptación a los cambios (de otros lenguajes, bases de datos...), lo que a su vez pronostica una mayor flexibilidad.

4.3.4 Flexibilidad

Es importante para los lenguajes de progración y más concretamente para los lenguajes de programación web que dispongan de una gran flexibilidad para ser capaces de adaptarse a las nuevas tendencias del mercado. Un lenguaje que sólo funciona con una base de datos se una cierta compañía, pongamos por caso, quedará completamente obsoleto si ésta quiebra, por eso los lenguajes buscan tener una buena abstracción en pos de poderse adaptar a los cambios.

4.4 Identificación inicial de lenguajes

En esta sección expongo un análisis de los lenguajes de programación web más populares. Se dará una visión global del lenguaje, un poco de su historia, su paradigma y sus cualidades. En la siguiente sección elegiré los lenguajes sobre los cuales será objeto este estudio.

4.4.1 HTML

HTML, siglas de HyperText Markup Language (Lenguaje de Marcado de Hipertexto), es el lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un script (por ejemplo Javascript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

HTML también es usado para referirse al contenido del tipo de MIME text/html o todavía más ampliamente como un término genérico para el HTML, ya sea en forma descendida del XML (como XHTML 1.0 y posteriores) o en forma descendida directamente de SGML (como HTML 4.01 y anteriores).

4.4.1.1 MARCADO HTML

HTML consta de varios componentes vitales, incluyendo elementos y sus atributos, tipos de data, y la declaración de tipo de documento.

4.4.1.1.1 ELEMENTOS

Los elementos son la estructura básica de HTML. Los elementos tienen dos propiedades básicas: atributos y contenido. Cada atributo y contenido tiene ciertas restricciones para que se considere válido al documento HTML. Un elemento generalmente tiene una etiqueta de inicio (p.ej. <nombre-de-elemento>) y una etiqueta de cierre (p.ej. </nombre-de-elemento>). Los atributos del elemento están contenidos en la etiqueta de inicio y el contenido está ubicado entre las dos etiquetas (p.ej. <nombre-de-elemento atributo="valor">Contenido</nombre-de-elemento>). Algunos elementos, tales como
, no tienen contenido ni llevan una etiqueta de cierre. Debajo se listan varios tipos de elementos de marcado usados en HTML.

El marcado estructural describe el propósito del texto. Por ejemplo, <h2>Golf</h2> establece «Golf» como un encabezamiento de segundo nivel, el cual se mostraría en un navegador de una manera similar al título «Marcado HTML» al principio de esta sección. El marcado estructural no define cómo se verá el elemento, pero la mayoría de los navegadores web han estandarizado el formato de los elementos. Un formato específico puede ser aplicado al texto por medio de hojas de estilo en cascada.

El marcado presentacional describe la apariencia del texto, sin importar su función. Por ejemplo, negrita indica que los navegadores web visuales deben mostrar el texto en negrita, pero no indica qué deben hacer los navegadores web que muestran el contenido de otra manera (por ejemplo, los que leen el texto en voz alta). En el caso de negrita e itálica, existen elementos que se ven de la misma manera pero tienen una naturaleza más semántica: enfásis fuerte y énfasis. Es fácil ver cómo un lector de pantalla debería interpretar estos dos elementos. Sin embargo, son equivalentes a sus correspondientes elementos

presentacionales: un lector de pantalla no debería decir más fuerte el nombre de un libro, aunque éste esté en itálicas en una pantalla. La mayoría del marcado presentacional ha sido desechada con HTML 4.0, en favor de hojas de estilo en cascada.

El marcado hipertextual se utiliza para enlazar partes del documento con otros documentos o con otras partes del mismo documento. Para crear un enlace es necesario utilizar la etiqueta de ancla <a> junto con el atributo href, que establecerá la dirección URL a la que apunta el enlace. Por ejemplo, un enlace a Pagina.com sería de la forma Wikipedia. También se pueden crear enlaces sobre otros objetos, tales como imágenes .

4.4.1.1.2 ATRIBUTOS

La mayoría de los atributos de un elemento son pares nombre-valor, separados por un signo de igual «=» y escritos en la etiqueta de comienzo de un elemento, después del nombre de éste. El valor puede estar rodeado por comillas dobles o simples, aunque ciertos tipos de valores pueden estar sin comillas en HTML (pero no en XHTML). De todas maneras, dejar los valores sin comillas es considerado poco seguro. En contraste con los pares nombre-elemento, hay algunos atributos que afectan al elemento simplemente por su presencia (tal como el atributo ismap para el elemento img).

4.4.1.1.3 CÓDIGOS HTML BÁSICOS

<html>: define el inicio del documento HTML, le indica al navegador que lo que viene a continuación debe ser interpretado como código HTML. Esto es así de facto, ya que en teoría lo que define el tipo de documento es el DOCTYPE, significando la palabra justo tras DOCTYPE el tag de raíz, por ejemplo:

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Strict//EN"
"http://www.w3.org/TR/html4/strict.dtd">

<script>: incrusta un script en una web, o se llama a uno mediante src="url del script". Se recomienda incluir el tipo MIME en el atributo type, en el caso de JavaScript text/javascript.

<head>: define la cabecera del documento HTML; esta cabecera suele contener información sobre el documento que no se muestra directamente al usuario. Como por ejemplo el título de la ventana del navegador. Dentro de la cabecera <head> podemos encontrar:

- **<title>**: define el título de la página. Por lo general, el título aparece en la barra de título encima de la ventana.
- **<link>**: para vincular el sitio a hojas de estilo o iconos. Por ejemplo:<link rel="stylesheet" href="/style.css" type="text/css">.
- **<style>**: para colocar el estilo interno de la página; ya sea usando CSS, u otros lenguajes similares. No es necesario colocarlo si se va a vincular a un archivo externo usando la etiqueta <link>.

- **<meta>**: para metadatos como la autoría o la licencia, incluso para indicar parámetros http (mediante http-equiv="") cuando no se pueden modificar por no estar disponible la configuración o por dificultades con server-side scripting.

<body>: define el contenido principal o cuerpo del documento. Esta es la parte del documento html que se muestra en el navegador; dentro de esta etiqueta pueden definirse propiedades comunes a toda la página, como color de fondo y márgenes. Dentro del cuerpo <body> podemos encontrar numerosas etiquetas. A continuación se indican algunas a modo de ejemplo:

- **<h1> a <h6>**: encabezados o títulos del documento con diferente relevancia.
- **<table>**: define una tabla.
- **<tr>**: fila de una tabla.
- **<td>**: celda de una tabla (debe estar dentro de una fila).
- **<a>**: hipervínculo o enlace, dentro o fuera del sitio web. Debe definirse el parámetro de pasada por medio del atributo href. Por ejemplo: Wikipedia se representa como Wikipedia).
- ** **: división de la página. Se recomienda, junto con css, en vez de <table> cuando se desea alinear contenido.
- **<img>**: imagen. Requiere del atributo src, que indica la ruta en la que se encuentra la imagen. Por ejemplo: . Es conveniente, por accesibilidad, poner un atributo alt="texto alternativo".
- **
**: etiquetas para listas.
- **<b>**: texto en negrita (etiqueta desaprobada. Se recomienda usar la etiqueta).
- **<i>**: texto en cursiva (etiqueta desaprobada. Se recomienda usar la etiqueta).
- **<s>**: texto tachado (etiqueta desaprobada. Se recomienda usar la etiqueta <del>).
- **<u>**: texto subrayado.

La mayoría de etiquetas deben cerrarse como se abren, pero con una barra («/») tal como se muestra en los siguientes ejemplos:

- <table><tr><td>Contenido de una celda</td></tr></table>.
- <script>Código de un [[script]] integrado en la página</script>.

4.4.1.2 NOCIONES BÁSICAS DE HTML

El lenguaje HTML puede ser creado y editado con cualquier editor de textos básico, como puede ser Gedit en Linux, el Bloc de notas de Windows, o cualquier otro editor que admita texto sin formato como GNU Emacs, Microsoft Wordpad, TextPad, Vim, Notepad++, entre otros.

Existen además, otros editores para la realización de sitios web con características WYSIWYG (What You See Is What You Get, o en español: «lo que ves es lo que obtienes»). Estos editores permiten ver el resultado de lo que se está editando en tiempo real, a medida que se va desarrollando el documento. Ahora bien, esto no significa una manera distinta de realizar sitios web, sino que una forma un tanto más simple ya que estos programas, además de tener la opción de trabajar con la vista preliminar, tiene su propia sección HTML la cual va generando todo el código a medida que se va trabajando. Algunos ejemplos de editores WYSIWYG son KompoZer, Microsoft FrontPage, o Macromedia Dreamweaver.

Combinar estos dos métodos resulta muy interesante, ya que de alguna manera se ayudan entre sí. Por ejemplo; si se edita todo en HTML y de pronto se olvida algún código o etiqueta, simplemente me dirijo al editor visual o WYSIWYG y se continúa ahí la edición, o viceversa, ya que hay casos en que sale más rápido y fácil escribir directamente el código de alguna característica que queramos adherirle al sitio, que buscar la opción en el programa mismo.

Existe otro tipo de editores HTML llamados WYSIWYM que dan más importancia al contenido y al significado que a la apariencia visual. Entre los objetivos que tienen estos editores es la separación del contenido y la presentación, fundamental en el diseño web.

HTML utiliza etiquetas o marcas, que consisten en breves instrucciones de comienzo y final, mediante las cuales se determinan la forma en la que debe aparecer en su navegador el texto, así como también las imágenes y los demás elementos, en la pantalla del ordenador.

Toda etiqueta se identifica porque está encerrada entre los signos menor que y mayor que (<>), y algunas tienen atributos que pueden tomar algún valor. En general las etiquetas se aplicarán de dos formas especiales:

- Se abren y se cierran, como por ejemplo: negrita que se vería en su navegador web como negrita.
- No pueden abrirse y cerrarse, como <hr /> que se vería en su navegador web como una línea horizontal.
- Otras que pueden abrirse y cerrarse, como por ejemplo .

Las etiquetas básicas o mínimas son:

- <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
- <html lang="es">
- <head>
- <title>Ejemplo</title>
- </head>
- <body>
- ejemplo
- </body>
- </html>

4.4.1.3 ACCESIBILIDAD WEB

El diseño en HTML aparte de cumplir con las especificaciones propias del lenguaje debe respetar unos criterios de accesibilidad web, siguiendo unas pautas, o las normativas y leyes vigentes en los países donde se regule dicho concepto. Se encuentra disponible y desarrollado por el W3C a través de las Pautas de Accesibilidad al Contenido Web 1.0 WCAG (actualizadas recientemente con la especificación 2.012), aunque muchos países tienen especificaciones propias, como es el caso de España con la Norma UNE 139803.13

4.4.1.4 HISTORIA DE HTML

4.4.1.4.1 PRIMERAS ESPECIFICACIONES

La primera descripción de HTML disponible públicamente fue un documento llamado HTML Tags (Etiquetas HTML), publicado por primera vez en Internet por Tim Berners-Lee en 1991. Describe 22 elementos comprendiendo el diseño inicial y relativamente simple de HTML. Trece de estos elementos todavía existen en HTML 4.

Berners-Lee consideraba a HTML una ampliación de SGML, pero no fue formalmente reconocida como tal hasta la publicación de mediados de 1993, por la IETF, de una primera proposición para una especificación de HTML: el boceto Hypertext Markup Language de Berners-Lee y Dan Connolly, el cual incluía una Definición de Tipo de Documento SGML para definir la gramática. El boceto expiró luego de seis meses, pero fue notable por su reconocimiento de la etiqueta propia del navegador Mosaic usada para insertar imágenes sin cambio de línea, reflejando la filosofía del IETF de basar estándares en prototipos con éxito. Similarmente, el boceto competidor de Dave Raggett HTML+ (Hypertext Markup Format) (Formato de marcaje de hipertexto), de 1993 tardío, sugería, estandarizar características ya implementadas tales como tablas.

4.4.2 HTML 5

4.4.2.1 LAS NOVEDADES DE HTML5

HTML5 está llamada a ser el reemplazo del actual (X)HTML, una de las patas de la web desde su nacimiento. Precisamente en un momento en el que la web está lo suficientemente madura, este estándar aprende de los errores cometidos e intenta solucionar la mayoría de problemas con los que un desarrollador web se encuentra. Como muchas de sus novedades son interesantes y afectan directamente a la futura web, desde AnexoM te vamos a comentar en varios artículos los cambios más importantes, empezando por este artículo donde comentaremos los nuevos elementos.

Antes de seguir habría que aclarar que HTML5 sigue en borrador y lo seguirá estando durante algunos años más. El enfoque general ha cambiado bastante respecto a versiones anteriores de HTML, añadiendo semántica y accesibilidad implícitas, especificando cada detalle y borrando cualquier ambigüedad. También se tiene en cuenta que muchas páginas web actuales son dinámicas, pareciéndose más a aplicaciones que a documentos. Algo básico es que HTML5 está definido en

base al DOM (la representación interna de una web con la que trabaja un navegador), dejando de lado la representación “real”, definiendo a la vez un estándar HTML y XHTML.

4.4.2.2 MEJOR ESTRUCTURA

Hoy en día se abusa bastante del elemento div, que nos permite estructurar una web en bloques. En HTML5 hay varios elementos que sirven para estructurar mejor una página web, estableciendo qué es cada sección, y reemplazando en muchas ocasiones a div. Con este extra de semántica, será mucho más coherente y fácil de entender por otras personas. Y lo que es más importante, será trivial de entender para una máquina, dándole más importancia a unas secciones y pudiendo jugar con esos datos automáticamente. Concretamente, la tarea de un buscador será mucho más fácil, pero cualquier aplicación que “lea” páginas web se beneficiará. Estos son los elementos:

- **section**: representa una sección “general” dentro de un documento o aplicación, como un capítulo de un libro. Puede contener subsecciones y si lo acompañamos de h1-h6 podemos estructurar mejor toda la página.
- **article**: representa un contenido independiente en un documento, el caso más claro son las entradas de un blog o las noticias de un periódico online. Así, dentro de la portada podremos tener varios artículos demarcados semánticamente, por lo que una herramienta puede extraerlos fácilmente.
- **aside**: representa un contenido que está muy poco relacionado con el resto de la página, como una barra lateral. Esencial para delimitar el contenido “importante” del contenido “de apoyo”, haciendo más caso al primero que al segundo.
- **header**: representa la cabecera de una sección, y es de suponer que se le dé más importancia que al resto, sobre todo si la sección es un artículo.
- **footer**: representa el pié de una sección, con información acerca de la página/sección que poco tiene que ver con el contenido de la página, como el autor, el copyright o el año.
- **nav**: representa una sección dedicada a la navegación entre el sitio, como la típica barra superior de los periódicos.

4.4.2.3 HTML5 ESTRUCTURADO

En la anterior imagen vemos un ejemplo de cómo cambiaría un documento escrito en HTML normal a HTML5 con estos elementos.

4.4.2.4 MEJORES FORMULARIOS

El elemento input ha sido ampliado y ahora permite todos estos tipos de datos:

- datetime, datetime-local, date, month, week, time, para que indicar una fecha/hora.

- **number** para que el usuario indique un número.
- **range** para indicar un rango entre dos números.
- **email** para indicar un correo electrónico.
- **url** para indicar una dirección web.
- **search** para indicar una búsqueda.
- **color** para indicar un color.

Lo más interesante de esto es que los navegadores podrán implementar interfaces específicas para cada tipo de dato, por ejemplo una fecha o un color se podrán indicar de manera directa e intuitiva. Otro ejemplo sería el teclado del iPhone, que muestra unos símbolos u otros dependiendo de si es un texto normal, un email (añade @ y el punto) o una url (añade la barra y el punto com), y que por tanto gana mucho con este estándar.

4.4.2.5 OTROS ELEMENTOS IMPORTANTES

- **audio y video** sirven para incrustar un contenido multimedia de sonido o de vídeo, respectivamente. Sin duda uno de los añadidos más interesantes, ya que permite reproducir/controlar vídeos y audios sin necesidad de plugins como el de Flash. Se tratan de manera totalmente nativa como cualquier otro elemento, por ejemplo se pueden incluir enlaces o imágenes dentro de un vídeo. Aunque las implementaciones actuales son un tanto ineficientes, se espera que en un futuro próximo se optimicen. Portales de vídeo como Youtube o Dailymotion ya están empezando a mostrar que un futuro sin Flash es posible (¡y necesario!).
- **embed** sirve para contenido incrustado pero no nativo, sino ejecutado por plugins como el de Flash. Aunque embed está soportado por casi todos los navegadores desde hace tiempo, es ahora cuando entra parte del estándar y evita el infierno/pelea entre object y embed.
- **canvas** es un elemento complejo que permite generar gráficos, dibujando elementos dentro de él. Aunque nunca hayas oído hablar de él, seguro que lo has usado alguna vez, por ejemplo de Google Maps. Es un elemento muy potente que dará bastante que hablar en el futuro, y que será el culpable de aplicaciones web espectaculares.

4.4.2.6 MÁS ELEMENTOS

- **dialog** se plantea para escribir conversaciones, por ejemplo para transcripciones de chat.
- **figure** se plantea para asociar un contenido multimedia (una foto, un vídeo, etc) a un título o leyenda.
- **mark** representa un texto resaltado, por ejemplo para resaltar una búsqueda.
- **meter** representa una medida, como el número de KB.

- **progress** representa el estado de una tarea, y se puede usar por ejemplo al subir un documento o al realizar varias tareas pesadas. Esto permitirá barras de tareas personalizadas y potentes.
- **time** representa una fecha o una hora.
- **command** representa un comando que el usuario puede ejecutar en su navegador.
- **output** representa una salida de un programa, probablemente ejecutado directamente en el navegador, como una calculadora.
- **datagrid** representa datos de manera interactiva y permite trabajar dinámicamente con información y cambiar la página respecto a esa información. Será útil sobre todo si se quiere trabajar con aplicaciones que necesiten de bastantes datos a la vez en el lado del cliente.

4.4.2.7 NUEVOS ATRIBUTOS

Los elementos ya existentes añaden nuevos atributos que permiten hacer cosas muy interesantes. Los más importantes o novedosos son:

- **Atributo ping** para el elemento a. Este atributo contiene una lista de URLs, las cuáles serán llamadas cuando un usuario haga click en ese enlace. Por ejemplo, un uso práctico sería para estadísticas.
- **Atributo autofocus** para los elementos de un formulario. Indica qué elemento de un formulario debe ganar el foco al cargar una página. Por ejemplo, en la página principal de Google la cajita de texto gana el foco automáticamente para ayudarnos a escribir sin usar el ratón.
- **Atributo form** para los elementos de un formulario. Indica a qué formulario pertenece este elemento, y permite poner un elemento de un formulario en cualquier parte de una página. Tal y como está ahora, todos los elementos deben ir dentro de <form>.
- **Atributo required** para los elementos de un formulario. Indica que es obligatorio rellenar un valor para enviar ese formulario, algo que hoy en día se comprueba con Javascript o en el servidor.
- **Atributos autocomplete, min, max, multiple, pattern y step** para los elementos **input**. Estos atributos indican que el valor del input debe cumplir ciertos requisitos, por ejemplo seguir cierto patrón.
- **Atributo novalidate** para el elemento form. Esto deshabilitará la validación de sus elementos, algo útil si se usa dinámicamente (es decir, con Javascript, activando y desactivándolo).
- **Atributos formaction, formenctype, formmethod, formnovalidate, y formtarget** para los elementos de un formulario. Estos atributos modifican los atributos del formulario si el elemento es usado (por ejemplo un botón es pulsado o un input es rellenado). Esto permite que dependiendo qué botón usemos podemos tratar el formulario en diferentes páginas usando solo un formulario, algo complicado en HTML4.
- **Atributo scoped** en el elemento **style.** Esto permite aplicar estilos solo a cierto subárbol del documento. Por ejemplo, imagina que tenemos un elemento con id igual a “cabecera”; si

añadimos el atributo scoped a un style, los estilos contenidos en él solamente se aplicarán a ese elemento y a sus hijos.

- **Atributo async** en el elemento **script.** Con este atributo especificamos que el código interno se puede ejecutar en cualquier momento de la página, mejorando la velocidad de carga.
- **Atributo manifest** en el elemento **html**. Esto permite indicar nuevos elementos, sobre todo será útil en aplicaciones web.
- **Atributo reversed** en el elemento **ol**. De esta forma la lista será numerada en orden descendiente (3, 2, 1…).

4.4.2.8 NUEVOS ATRIBUTOS GLOBALES

Además de los anteriores tenemos otros atributos que pueden ser aplicados a todos los elementos de un documento. Esto lo hacen especialmente conveniente si vamos a usar Javascript para modificarlos dinámicamente, ya que no tenemos que comprobar el tipo de elemento para usar los atributos comunes.

- **Atributos id, class, style, title, lang y tabindex**, ya existentes pero ahora permitidos en todos los elementos. Pues eso, los atributos más usados se vuelven universales.
- **Atributo content editable,** para indicar que el elemento es editable por el usuario. Muy interesante, ya que ahora podremos editar cualquier cosa de manera trivial para el desarrollador, no solamente los elementos de un formulario.
- A**tributo contextmenu**, para indicar un menú contextual sobre un elemento. Como veis, está todo muy pensado para las aplicaciones web.
- **Atributos data-***, donde el asterisco puede ser cualquier nombre. Esto permite atributos personalizados, que luego podremos obtener con Javascript.
- **Atributo draggable**, para indicar que un elemento se puede arrastrar.
- **Atributo hidden**, para ocultar un elemento que ya no interesa.
- **Atributo spellcheck**, para indicar si el contenido de un elemento debe ser pasado por el corrector ortográfico.

4.4.2.9 NUEVAS APIS

No está claro que todas las APIs siguientes se vayan a incluir en el estándar HTML5 propiamente, de hecho seguro que alguna de ellas se separará creando un estándar propio dedicado. De cualquier forma, estas son las nuevas APIs que nacen o se desarrollan en HTML5:

Una API para dibujar en 2D, que se podrá usar junto al nuevo elemento canvas. Básicamente, se pueden pintar elementos sobre un lienzo, de manera similar a lenguajes como Java.

Una API para controlar los nuevos elementos multimedia, video y audio. Así podremos controlar la reproducción con código Javascript, algo interesante pero que puede dar más de sí.

Una API para guardar datos localmente, utilísimo para que las aplicaciones web puedan trabajar sin necesitar conexión a Internet. Ese DOM storage está ya implementado en las últimas versiones de los grandes navegadores, así que dentro de poco podremos disfrutar de esta clase de aplicaciones sin necesidad de extensiones como Google Gears.

Una API para que las aplicaciones web puedan enlazarse a protocolos o tipos de archivos MIME, otro añadido extremadamente útil. Esto permitiría abrir las fotos de tu disco duro directamente en una aplicación de retoque online, o un archivo mp3 en una biblioteca online, etc…

Una API para editar los campos que sean editables. Esto permite controlar los elementos HTML que son editables por el usuario, tipo Word. Por ejemplo, Google Docs o editar HTML en emails será muy sencillo, y con esta API es trivial cambiar ese contenido con Javascript.

Una API para controlar las acciones Drag & Drop sobre los elementos que se puedan arrastrar. Esto se puede conseguir actualmente con algunas librerías, pero con esta API el navegador lo permite de manera nativa y más poderosa.

Una API para controlar el historial desde una aplicación web. Esto permitirá a las aplicaciones web que se muevan con Javascript añadir páginas al historial para que los botones Atrás-Adelante funcionen siempre.

Una API para habilitar la comunicación entre varias “páginas web”. Es decir, si tenemos varios iframes externos en una web, podemos comunicarnos con ellos y compartir información de manera segura, por ejemplo con gadgets de Facebook o similares.

4.4.2.10 NOVEDADES EN EL DOM

Se han añadido a los elementos del DOM nuevas funciones y atributos que facilitan su uso y permiten realizar acciones muy usadas. Aquí comentaré los más interesantes, que trabajan sobre el documento (HTMLDocument) o sobre cualquier elemento (HTMLElement).

- La función getElementsByClassName() se añade a todos los elementos y al documento. Su funcionamiento es similar a getElementsById(), pero en este caso selecciona todos los elementos del documento o de cierto subárbol del documento que contengan esa clase. Su definición es tan amplia que incluso elementos que contenga SVG o MathML pueden ser encontrados.
- El atributo classList está disponible para cualquier elemento, y contiene una lista con todas las clases que tiene ese elemento. Además contiene varios métodos que facilitan trabajar con esa lista: buscar, modificar, borrar, etc. Muy útil para trabajar con elementos que puedan tener más de una clase, es muy sencillo y muy conveniente de usar. Los enlaces también tienen un elemento similar adicional llamado relList que permite lo mismo pero con el atributo rel (como el famoso rel=”nofollow”).
- El atributo innerHTML se añade, por fin, al estándar. Prácticamente usado por todas las aplicaciones web y soportado por todos los navegadores, creo que todos los desarrolladores web lo conocen de sobra. Para aquellos que no lo conozcan, este atributo permite trabajar con el contenido de un elemento en texto plano, incluso cambiando elementos HTML que pueda haber. Igualmente, se añade a todos los elementos y al propio documento, pudiendo cambiar TODO el contenido de una web.

- Los atributos activeElement y hasFocus están disponibles sobre un documento, y permiten conocer qué elemento está activo y si el documento tiene el foco, respectivamente.
- La función getSelection() se aplica también al documento y devuelve un objeto con el texto y elementos que están seleccionados.
- El atributo designMode es otra novedad sobre el documento e indica/modifica que el documento pueda o no ser editado.
- La función execCommand() se aplica sobre el documento y permite ejecutar acciones o "comandos" típicos de edición de documentos. Por ejemplo, con este método se llamaría a los útiles copiar/pegar, pero también a otros típicos como crear un enlace o cambiar el color de un elemento. Como el anterior, la mayoría de estos comandos trabajan sobre elementos editables.

4.4.2.11 ELEMENTOS ELIMINADOS

Como decía, estos son los elementos eliminados y las razones de por qué son prohibidos:

- Los siguientes elementos (muy usados hace pocos años) se quitan de HTML5 porque son **puramente presentacionales** (no tienen semántica) y todo el tema estético se debe tratar con CSS:
  - -basefont
  - -big
  - -center
  - -font
  - -s
  - -strike
  - -tt
  - -u
- Los elementos para **trabajar con frames** (frame, frameset y noframes) se quitan del estándar por razones obvias: afectan negativamente a la usabilidad y accesibilidad de la web. Además, prácticamente rompen la web, y si se necesita algo similar se puede acudir a los iframe, más potentes y mejor pensados.
- El elemento acronym se elimina simplemente porque crea confusión sobre su uso, y los desarrolladores no entienden demasiado bien para qué usarlo. Las abreviaciones y acrónimos se pueden marcar con abbr, que sí se mantiene en el estándar.
- El elemento applet se ha declarado obsoleto y hoy en día no se utiliza. El elemento object reemplaza sus funciones y es lo común hoy en día.

- El elemento isindex se quita definitivamente. En la era de las cavernas se utilizaba para mandar información al servidor, pero con la llegada de los formularios su uso es arcaico y poco útil.
- El elemento dir también se declara obsoleto (ya lo era en HTML4), y simplemente se recomienda usar listas normales con ul.
- El elemento noscript se mantiene en HTML pero no en XML/XHTML, ya que su contenido está en HTML. No estoy muy de acuerdo con este movimiento, pero así será.

4.4.2.12 ATRIBUTOS ELIMINADOS

Para empezar, todos los atributos referentes a la presentación han sido eliminados, por la misma razón de antes: CSS sirve mejor ese propósito. Recuerdo que el atributo style (que contiene CSS) es ahora universal y puede ser aplicado a todos los elementos, así que si queremos indicar su presentación sin añadir una hoja de estilos aparte, tendremos que usar este atributo. Atención a la lista porque esto sí que es importante, ya que algunos de estos elementos son muy usados, aunque otros están muy obsoletos:

- **Atributo align** en todos los elementos.
- **Atributos alink, link, text y vlink** en el elemento body.
- **Atributo background** en el elemento body.
- **Atributo bgcolor** en los elementos table, tr, td, th y body.
- **Atributo border** en todos los elementos.
- **Atributos cellpadding y cellspacing** en el elemento table.
- **Atributos char y charoff** en los elementos col, colgroup, tbody, td, tfoot, th, thead y tr.
- **Atributo clear** en el elemento br.
- **Atributo compact** en los elementos dl, menu, ol y ul.
- **Atributo frame** en el elemento table.
- **Atributo frameborder** en el elemento iframe.
- **Atributo height** en los elementos td y th.
- **Atributos hspace y vspace** en los elementos img y object.
- **Atributos marginheight y marginwidth** en el elemento iframe.
- **Atributo noshade** en el elemento hr.
- **Atributo nowrap** en los elementos td y th.
- **Atributo rules** en el elemento table.
- **Atributo scrolling** en el elemento iframe.
- **Atributo size** en el elemento hr.

- **Atributo type** en los elementos li, ol y ul.
- **Atributo valign** en los elementos col, colgroup, tbody, td, tfoot, th, thead y tr.
- **Atributo width** en los elementos hr, table, td, th, col, colgroup y pre.

Como veis, algunos de estos atributos sí que se mantienen para ciertos elementos, como la anchura y altura en las imágenes. Sin embargo estos no son los únicos atributos que se eliminan, también hay otros que se quitan por redundancia, por evitar confusiones, por su bajo uso o porque simplemente se han quedado obsoletos.

- **Atributo accesskey** en los elementos a, area, button, input, label, legend y textarea.
- **Atributos rev y charset** en los elementos link y a.
- **Atributos shape y coords** en el elemento a.
- **Atributo longdesc** en los elementos img y iframe.
- **Atributo target** en el elemento link.
- **Atributo nohref** en el elemento area.
- **Atributo profile** en el elemento head.
- **Atributo version** en el elemento html.
- **Atributo name** en los elementos img y a. Para obtener un comportamiento similar se recomienda usar id.
- **Atributo scheme** en el elemento meta.
- **Atributos archive, classid, codebase, codetype, declare y standby** en el elemento object.
- **Atributos valuetype y type** en el elemento param.
- **Atributo language** en el elemento script.
- **Atributo summary** en el elemento table.
- **Atributos axis y abbr** en los elementos td y th.
- **Atributo scope** en el elemento td.

4.4.2.13 OTROS CAMBIOS

Hay diversos elementos que no se eliminan por su extendida fama, pero que siendo un tanto ortodoxos deberían eliminarse. Para evitar esto los que están escribiendo el estándar han tenido que redefinir su definición, de tal forma que se tratan de manera similar pero semánticamente son diferentes.

Un ejemplo muy claro es u e i, muy usados pero que progresivamente pierden importancia frente a strong y em. Estos dos elementos, que indicaban negrita y cursiva respectivamente, pasan a definirse de una manera muy vaga para indicar un texto diferente en alguna manera al texto normal. Otros elementos se redefinen, particularmente me resulta curioso que se mantenga small mientras

big se elimina. De cualquier manera, no es demasiado relevante para los desarrolladores web, en el sentido de que podrán seguir usándolos como ahora.

4.4.2.14 CONCLUSIONES

Tal y cómo habéis ido viendo, las novedades de HTML5 se centran en facilitar la implementación de aplicaciones web, avanzar hacia la web semántica y limpiar un poco toda la basura heredada de las anteriores versiones de HTML. Aunque todo eso parezca lejano, lo cierto es que muchos navegadores ya implementan algunas partes sueltas de HTML5, y ya existen varias páginas experimentales que juegan con estos elementos.

4.4.3 CSS

El nombre hojas de estilo en cascada viene del inglés Cascading Style Sheets, del que toma sus siglas. CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C (World Wide Web Consortium) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.

La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación.

Por ejemplo, el elemento de HTML <H1> indica que un bloque de texto es un encabezamiento y que es más importante que un bloque etiquetado como <H2>. Versiones más antiguas de HTML permitían atributos extra dentro de la etiqueta abierta para darle formato (como el color o el tamaño de fuente). No obstante, cada etiqueta <H1> debía disponer de la información si se deseaba un diseño consistente para una página y, además, una persona que leía esa página con un navegador perdía totalmente el control sobre la visualización del texto.

Cuando se utiliza CSS, la etiqueta <H1> no debería proporcionar información sobre cómo será visualizado, solamente marca la estructura del documento. La información de estilo, separada en una hoja de estilo, especifica cómo se ha de mostrar <H1>: color, fuente, alineación del texto, tamaño y otras características no visuales, como definir el volumen de un sintetizador de voz, por ejemplo.

La información de estilo puede ser adjuntada como un documento separado o en el mismo documento HTML. En este último caso podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo "style".

4.4.3.1 LOS TRES TIPOS DE ESTILOS

CSS proporciona tres caminos diferentes para aplicar las reglas de estilo a una página Web:

- Una hoja de estilo externa, es una hoja de estilo que está almacenada en un archivo diferente al archivo donde se almacena el código HTML de la página Web. Esta es la manera de programar más potente, porque separa completamente las reglas de formateo para la página HTML de la estructura básica de la página:

- Una hoja de estilo interna, que es una hoja de estilo que está incrustada dentro de un documento HTML. (Va a la derecha dentro del elemento <head>.) De esta manera se obtiene el beneficio de separar la información del estilo del código HTML propiamente dicho. Se puede optar por copiar la hoja de estilo incrustada de una página a otra (esta posibilidad es difícil de ejecutar si se desea para guardar las copias sincronizadas). En general, la única vez que se usa una hoja de estilo interna, es cuando se quiere proporcionar alguna característica a una página Web en un simple fichero, por ejemplo, si se está enviando algo a la página Web.
- Un estilo en línea (inline) es un método para insertar el lenguaje de estilo de página directamente dentro de una etiqueta HTML. Esta manera de proceder no es totalmente adecuada. El incrustar la descripción del formateo dentro del documento de la página Web, a nivel de código, se convierte en una manera larga, tediosa y poco elegante de resolver el problema de la programación de la página. Este modo de trabajo se podría usar de manera ocasional si se pretende aplicar un formateo con prisa, al vuelo. No es todo lo claro o estructurado que debería ser, pero funciona. Éste es el método recomendado para maquetar correos electrónicos en HTML.

4.4.3.2 VENTAJAS DE USAR LAS HOJAS DE ESTILO

Las ventajas de utilizar CSS (u otro lenguaje de estilo) son:

- Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
- Los navegadores permiten a los usuarios especificar su propia hoja de estilo local, que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o, incluso, a elección del usuario. Por ejemplo, para ser impresa, mostrada en un dispositivo móvil o ser "leída" por un sintetizador de voz.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño (siempre y cuando no se utilice estilo en línea).

4.4.3.3 DIAGRAMADO DE PÁGINA EN CSS

Antes de que estuviera disponible CSS, la única forma de componer espacialmente una página era el uso de tablas. Aunque es una técnica cómoda y versátil, se está usando un elemento con una semántica particular, que es la de expresar información tabular, solamente por su efecto en la presentación.

La introducción de CSS ha permitido en muchos casos reemplazar el uso de tablas. Sin embargo, CSS no permite aún la versatilidad que ofrecían las tablas, lograr un diagramado de una página compleja suele ser una tarea difícil en CSS y las diferencias entre navegadores dificultan aún

más la tarea. Se espera que futuros desarrollos en CSS3 resuelvan esta deficiencia y hagan de CSS un lenguaje más apto para describir la estructura espacial de una página.

4.4.4 PHP

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Se usa principalmente para la interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

PHP es un acrónimo recursivo que significa PHP Hypertext Pre-processor (inicialmente PHP Tools, o, Personal Home Page Tools). Fue creado originalmente por Rasmus Lerdorf en 1994; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la Free Software Foundation considera esta licencia como software libre.

Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. El lenguaje PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores, el número de sitios en PHP ha compartido algo de su preponderante sitio con otros nuevos lenguajes no tan poderosos desde agosto de 2005. Es también el módulo Apache más popular entre las computadoras que utilizan Apache como servidor web.

El gran parecido que posee PHP con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy corta. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones.

Aunque todo en su diseño está orientado a facilitar la creación de sitios webs, es posible crear aplicaciones con una interfaz gráfica para el usuario, utilizando la extensión PHP-Qt o PHP-GTK. También puede ser usado desde la línea de órdenes, de la misma manera como Perl o Python pueden hacerlo; a esta versión de PHP se la llama PHP-CLI (Command Line Interface).

Cuando el cliente hace una petición al servidor para que le envíe una página web, el servidor ejecuta el intérprete de PHP. Éste procesa el script solicitado que generará el contenido de manera dinámica (por ejemplo obteniendo información de una base de datos). El resultado es enviado por el intérprete al servidor, quien a su vez se lo envía al cliente. Mediante extensiones es también posible la generación de archivos PDF, Flash, así como imágenes en diferentes formatos.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite.

XAMPP es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor Web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl. El programa está liberado bajo la licencia GNU y actúa como un servidor Web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP esta disponible para Microsoft Windows, GNU/Linux, Solaris, y MacOS X.

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como Unix (y de ese tipo, como Linux o Mac OS X) y Microsoft Windows, y puede interactuar

con los servidores de web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

PHP es una alternativa a las tecnologías de Microsoft ASP y ASP.NET (que utiliza C# y Visual Basic .NET como lenguajes), a ColdFusion de la empresa Adobe, a JSP/Java y a CGI/Perl. Aunque su creación y desarrollo se da en el ámbito de los sistemas libres, bajo la licencia GNU, existe además un entorno de desarrollo integrado comercial llamado Zend Studio. Recientemente, CodeGear (la división de lenguajes de programación de Borland) ha sacado al mercado un entorno de desarrollo integrado para PHP, denominado 'Delphi for PHP. También existen al menos un par de módulos para Eclipse, uno de los entornos más populares.

4.4.4.1 CARACTERÍSTICAS DE PHP

4.4.4.1.1 VENTAJAS

- Es un lenguaje multiplataforma.
- Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- El código fuente escrito en PHP es invisible al navegador web y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Capacidad de expandir su potencial utilizando módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su sitio web oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Tiene manejo de excepciones (desde PHP5).
- Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar (muchos otros lenguajes tampoco lo hacen), aun haciéndolo, el programador puede aplicar en su trabajo cualquier técnica de programación o de desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del patrón de diseño Modelo Vista Controlador (MVC), que permiten separar el tratamiento y acceso a los datos, la lógica de control y la interfaz de usuario en tres componentes independientes.

4.4.4.1.2 INCONVENIENTES

Como es un lenguaje que se interpreta en ejecución, para ciertos usos puede resultar un inconveniente que el código fuente no pueda ser ocultado. La ofuscación es una técnica que puede dificultar la lectura del código pero no la impide y, en ciertos casos, representa un costo en tiempos de ejecución.

4.4.5 PYTHON

Python es un lenguaje de programación de alto nivel cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible.

Se trata de un lenguaje de programación multiparadigma ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico, es fuertemente tipado y multiplataforma.

Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License, que es compatible con la Licencia pública general de GNU a partir de la versión 2.1.1, e incompatible en ciertas versiones anteriores.

4.4.5.1 CARACTERÍSTICAS Y PARADIGMAS

Python es un lenguaje de programación multiparadigma. Esto significa que más que forzar a los programadores a adoptar un estilo particular de programación, permite varios estilos: programación orientada a objetos, programación imperativa y programación funcional. Otros paradigmas están soportados mediante el uso de extensiones.

Python usa tipado dinámico y conteo de referencias para la administración de memoria.

Una característica importante de Python es la resolución dinámica de nombres; es decir, lo que enlaza un método y un nombre de variable durante la ejecución del programa (también llamado ligadura dinámica de métodos).

Otro objetivo del diseño del lenguaje es la facilidad de extensión. Se pueden escribir nuevos módulos fácilmente en C o C++. Python puede incluirse en aplicaciones que necesitan una interfaz programable.

Aunque la programación en Python podría considerarse en algunas situaciones hostil a la programación funcional tradicional del Lisp, existen bastantes analogías entre Python y los lenguajes minimalistas de la familia Lisp como puede ser Scheme.

4.4.5.2 FILOSOFÍA

Los usuarios de Python se refieren a menudo a la Filosofía Python que es bastante análoga a la filosofía de Unix. El código que sigue los principios de Python de legibilidad y transparencia se dice que es "pythonico". Contrariamente, el código opaco u ofuscado es bautizado como "no pythonico" ("unpythonic" en inglés). Estos principios fueron famosamente descritos por el desarrollador de Python Tim Peters en El Zen de Python:

- Bello es mejor que feo.
- Explícito es mejor que implícito.
- Simple es mejor que complejo.
- Complejo es mejor que complicado.
- Plano es mejor que anidado.
- Disperso es mejor que denso.
- La legibilidad cuenta.
- Los casos especiales no son tan especiales como para quebrantar las reglas.
- Aunque lo práctico gana a la pureza.
- Los errores nunca deberían dejarse pasar silenciosamente.
- A menos que hayan sido silenciados explícitamente.
- Frente a la ambigüedad, rechaza la tentación de adivinar.
- Debería haber una -y preferiblemente sólo una- manera obvia de hacerlo.
- Aunque esa manera puede no ser obvia al principio a menos que usted sea holandés.
- Ahora es mejor que nunca.
- Aunque nunca es a menudo mejor que ya mismo.
- Si la implementación es difícil de explicar, es una mala idea.
- Si la implementación es fácil de explicar, puede que sea una buena idea.
- Los espacios de nombres (namespaces) son una gran idea ¡Hagamos más de esas cosas!

Desde la versión 2.1.2, Python incluye estos puntos (en su versión original en inglés) como un huevo de pascua que se muestra al ejecutar import this.

4.4.5.3 MODO INTERACTIVO

El intérprete de Python estándar incluye un modo interactivo en el cual se escriben las instrucciones en una especie de intérprete de comandos: las expresiones pueden ser introducidas una a una, pudiendo verse el resultado de su evaluación inmediatamente, lo que da la posibilidad de probar porciones de código en el modo interactivo antes de integrarlo como parte de un programa.

Esto resulta útil tanto para las personas que se están familiarizando con el lenguaje como para los programadores más avanzados.

Existen otros programas, tales como IDLE, bpython o IPython, que añaden funcionalidades extra al modo interactivo, como el autocompletado de código y el coloreado de la sintaxis del lenguaje.

4.4.5.4 PROGRAMACIÓN EN ENTORNOS WEB

4.4.5.4.1 ZOPE

Zope es un servidor de aplicaciones web de código abierto escrito en el lenguaje de programación Python. Para las funciones de edición de contenidos, así como personalizaciones básicas, puede ser usado mediante un navegador web. La programación avanzada así como el desarrollo de nuevas funcionalidades requiere la edición de componentes en «file system».

Un sitio web de Zope está compuesto de objetos en lugar de archivos, como es usual con la mayoría de los otros sistemas de servidores web. Las ventajas de usar objetos en lugar de archivos son:

- Combinan el comportamiento y los datos en una forma más natural que los archivos de texto plano.
- Alientan el uso de componentes estándares que se ocupan de una parte particular de las que forman una aplicación Web, permitiendo flexibilidad y buena descomposición.
- Posibilitan procesos automáticos de gestión de información.

Lo más característico de Zope es su base de datos orientada a objetos, llamada ZODB o Zope Object Database. Esta base de datos almacena objetos ordenados en un sistema similar a un sistema de ficheros, pero cada objeto tiene propiedades, métodos u otros objetos. Esta aproximación es muy diferente de las base de datos relacionales habituales. Sin embargo, Zope dispone de múltiples conectores para las diferentes bases de datos relacionales y ofrece sistemas básicos de conexión y consulta abstrayéndolos como objetos.

Actualmente existen dos ramas principales, zope2 y zope3. Este último es una reimplementación del servidor zope, donde se ha tratado de volcar toda la experiencia adquirida en zope2. Zope3 no trae compatibilidad hacia atrás, por lo que los componentes hechos para zope2 no funcionan. Aún se está en un proceso de adaptación hacia este nuevo zope, para lo cual está usando un componente llamado five, con el cual desde zope2 pueden tener la facilidad de zope3.

Algunos ejemplos de sitios que usan Zope son Launchpad y schooltool.

4.4.5.5 PSP

Otra de la grandes virtudes que presenta python es la cantidad de alternativas que presenta para el desarrollo de aplicaciones web, PSP es una de ellas. PSP se presenta como una alternativa

para poder generar páginas dinámicas de una manera no tan complicada, su sintaxis esta inspsirada en JSP.

PSP es una implementación de codigo python dentro de codigo html, parecido a la que se hace en PHP o JSP. La estructura de una página PSP es la misma que se hace en JSP, por ejemplo utilizando los delimitadores <% 'codigo python' %>.

Los archivos se guardan con extensión .psp y para poder ejecutarla tenemos varias alternativas; lo podemos hacer implementando un modulo en Apache o tambien usar un Servidor Aplicaciones como por ejemplo Webware.

4.4.5.5.1 MOD_PYTHON

Mod_Python es un módulo para Apache diseñado por Gregory Trubetskoy, que nos permite conectar Python directamente con Apache. El diseño de Apache se compone de una especie de cadena de montaje en la que existen puntos de control a los que podemos conectarnos y responder a eventos producidos en ellos usando Python.

Tenemos que configurar Apache para que haga uso de Mod_Python, aunque normalmente al instalar el módulo se realizan cambios en el fichero de configuración de Apache para activarlo. Aún así es necesario comunicar a Apache en qué directorio queremos que actúe Mod_Python y de qué manera:

```
<Directory>
/usr/local/www/ data/prueba>
AddHandler mod_python .py
PythonHandler index
PythonAuthenHandler index
PythonDebug On
AuthType Basic
AuthName ”Area Restringida”
require valid-user
</Directory>
```

Tenemos que incorporar un código como el anterior a nuestro fichero de configuración para Apache (el nombre del fichero y su localización puede cambiar con la distribución que empleemos). Es importante que adaptemos la ruta del directorio a la que usemos en nuestro sistema. Lo primero que haremos será proteger nuestra aplicación contra usua rios externos mediante una contraseña, de forma que durante el desarrollo no sea posible acceder a ella. La manera más sencilla de hacerlo es

mediante autenticación. El protocolo AUTH de HTTP es un estándar que soportan todos los navegadores.

En Mod_Python existen una serie de funciones que Apache llamará en el caso de que existan. El nombre de estas funciones se relaciona directamente con las etapas de procesado de peticiones de HTTP de Apache.

Todos estas funciones, que llamaremos handlers a partir de ahora, aceptan un solo parámetro: un objeto de la clase Request. Este objeto encapsula la petición y todos los datos que trae consigo.

Ahora nos concentraremonos en el authenhandler. Su código es muy sencillo. Cuando Apache encuentra un handler de autenticación como éste, lo primero que hace es presentar al usuario de la aplicación web una ventana en la que le pedirá tanto su nombre de usuario como su clave.

Esta acción se realiza de forma automática, nosotros sólo recogeremos el fruto de esa interacción: el nombre y la clave del usuario. Para ello accedemos a la variable req.user del objeto Request que aceptamos como parámetro. Para recoger la clave es preciso invocar get_basico_auth_pw() antes de comprobar el nombre del usuario, puesto que es esta función la que se encarga de traer ambos datos de forma segura.

El handler se encarga de la autenticación sustituyendo a Apache en el proceso. Comprueba que el nombre del usuario y la clave exista en la base de datos, y si se da el caso, se devuelves el valor apache.OK; en caso contrario, apache.HTTP_UNATHORIZED.

Estos valores son constantes que define Apache, en base a las cuales realizará ciertas acciones.

Pasemos a la función handler. Cuando el objeto Request se pasa a handler ya se ha recorrido la mayor parte del proceso de procesado de peticiones. handler es el núcleo de la aplicación.

El esquema de trabajo de las distintas funciones es más o menos el mismo. Se comprueba la presencia de una cookie o de unos argumentos esperados, y en base a ambas variables se toman unas acciones, que generalmente consistirán en redirigir la petición a otra función, realizar alguna interacción con la base de datos o pintar una pantalla por defecto.

4.4.5.5.2 PROCESADO DE PARÁMETROS

Los parámetros GET se pueden recoger gracias a la variable req.args. Pero no todo está solucionado, porque req.args devuelve la cadena de texto con los parámetros, no los propios parámetros. Se puede emplear la función cgi.parse_qs() del módulo cgi. Esta función rompe la cadena de parámetros GET en variables y genera un diccionario con el nombre de los parámetros y sus valores.

Los parámetros POST no tienen mayor complicación. Simplemente se comprueba el método con el que se están enviando los parámetros, y en el caso de que sea POST, se lee usando el método req.read(), pasando a procesarlos como con los parámetros GET.

4.4.5.5.3 LAS COOKIES

El procesado de las Cookies se realiza gracias a la clase Cookie. El trabajo con ella es bastante simple. Posee dos métodos: get_cookies() y add_cookies(), que permiten recoger y añadir información a una cookie respectivamente. Ambas funciones operan sobre el objeto req. Como medida es aconsejable cifrar la información presente en las cookies, para lo cual se emplea la función Cookie.MarshalCookie, que permite serializar (convertir en un array de bytes) cualquier objeto que se preste a ello, eliminando el problema de decidir cómo representarlo en la cookie, y de paso lo firma empleando md5 para asegurar la integridad de los datos. Para ello se le pasa un tercer parámetro tanto a get_cookies() como a add_cookies() llamado secret con la clave que emplearemos en la firma digital. El método get_cookies() devuelve un diccionario con los datos presentes en la cookie.

4.4.6 JAVASCRIPT

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

JavaScript se diseñó con una sintaxis similar al C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo Java y JavaScript no están relacionados y tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

Tradicionalmente se venía utilizando en páginas web HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. JavaScript se interpreta en el agente de usuario, al mismo tiempo que las sentencias van descargándose junto con el código HTML.

4.4.7 VBSCRIPT

VBScript (abreviatura de Visual Basic Script Edition) es un lenguaje interpretado por el Windows Scripting Host de Microsoft. Su sintaxis refleja su origen como variación del lenguaje de programación Visual Basic. Ha logrado un apoyo significativo por parte de los administradores de Windows como herramienta de automatización, ya que, conjunta y paralelamente a las mejoras introducidas en los sistemas operativos windows donde opera fundamentalmente, permite más margen de actuación y flexibilidad que el lenguaje batch (o de proceso por lotes) desarrollado a finales de los años 1970 para el MS-DOS.

El crecimiento del uso de las tecnologías de Internet ha supuesto un significativo avance para este lenguaje, dado que es parte fundamental de la ejecución de aplicaciones de servidor programadas en ASP (Active Server Pages), las cuales están en auge en el período 1997-2003,

declinando actualmente en favor de tecnologías de código gestionado y máquinas virtuales, más seguras en la ejecución de procesos, y por tanto, más adaptadas para ejecuciones en entornos públicamente accesibles y distribuidos. Microsoft ha intentado competir mediante esta tecnología también en entornos de cliente, donde el lenguaje más utilizado es Javascript o su versión estandarizada ECMAScript, sin éxito. Actualmente microsoft no ha puesto a disposición pública nuevas versiones del lenguaje, en favor de la tecnología .NET en la que se incluye el lenguaje hermano Visual Basic, dentro del entorno de ejecución de la plataforma .NET (CLR, o Common Language Runtime). Sin embargo sigue siendo muy útil en gestión de estaciones de trabajo y servidores en windows.

4.4.7.1 INTERPRETACIÓN

VBScript es interpretado por el motor de scripting vbscript.dll, que puede ser invocado por el motor ASP asp.dll en un entorno web, por wscript.exe en un entorno Windows de interfaz gráfica, por cscript.exe en un entorno de línea de comandos y por iexplore.exe cuando se trata de scripts a nivel de cliente (similar al javascript). Cuando el código fuente VBScript se guarda en ficheros independientes, éstos tienen típicamente la extensión .vbs.

Cuando se emplea en Internet Explorer, VBScript funciona de forma muy similar a JavaScript, procesando código contenido en el documento HTML. VBScript también puede usarse para crear aplicaciones HTML independientes (extensión .hta), que necesitan Internet Explorer 5.0 o superior para poder ser ejecutados. Los desarrolladores de aplicaciones en web suelen preferir JavaScript debido a su mayor compatibilidad con otros navegadores de Internet, ya que VBScript sólo está disponible para el navegador de Microsoft Internet Explorer y no en otros como Firefox, Google Chrome u Opera.

4.4.7.2 SEGURIDAD

VBScript es el lenguaje usado para escribir algunos famosos gusanos de red, como I Love You. Esto se debe a varias razones. Primero, el icono parecido a un pergamino azul que representa a los ficheros .vbs puede llevar a pensar a los usuarios inexpertos que se trata de un fichero de texto. Segundo, es fácil escribir un gusano informático en VBScript que se propague por correo electrónico (se necesitan pocas líneas de código). Microsoft ha solucionado los agujeros de seguridad explotados por dichos programas maliciosos. Este solucionado no significa erradicado, solo ha complicado el proceso; pues si por ejemplo el I Love You se propagaba a través del Outlook, ahora si se utiliza el mismo método sale un mensaje de advertencia, por lo que se suele emplear métodos como el envío mediante un servidor smtp (bastante más complejo de programar que por el otro método).

4.4.8 ASP

Active Server Pages (ASP), también conocido como ASP clásico, es una tecnología de Microsoft del tipo "lado del servidor" para páginas web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Services (IIS).

La tecnología ASP está estrechamente relacionada con el modelo tecnológico y de negocio de su fabricante. Intenta ser solución para un modelo de programación rápida ya que "programar en ASP es como programar en Visual Basic y C#", por supuesto con muchas limitaciones y algunas ventajas específicas en entornos web.

Lo interesante de este modelo tecnológico es poder utilizar diversos componentes ya desarrollados como algunos controles ActiveX así como componentes del lado del servidor, tales como CDONTS, por ejemplo, que permite la interacción de los scripts con el servidor SMTP que integra IIS.

Se facilita la programación de sitios web mediante varios objetos integrados, como por ejemplo un objeto de sesión basada en cookies, que mantiene las variables mientras se pasa de página a página.

Es limitado a solo funcionar con IIS, por lo que su uso es cuestionado por la mayoría de los programadores web quienes prefieren otros lenguajes de programación del lado del servidor como por ejemplo PHP, Perl, Java Etc.

4.4.9 JSP

JavaServer Pages (JSP) es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.

Esta tecnología es un desarrollo de la compañía Sun Microsystems. La Especificación JSP 1.2 fue la primera que se liberó y en la actualidad está disponible la Especificación JSP 2.1.

Las JSP's permiten la utilización de código Java mediante scripts. Además, es posible utilizar algunas acciones JSP predefinidas mediante etiquetas. Estas etiquetas pueden ser enriquecidas mediante la utilización de Bibliotecas de Etiquetas (TagLibs o Tag Libraries) externas e incluso personalizadas.

4.4.9.1 ARQUITECTURA

JSP puede considerarse como una manera alternativa, y simplificada, de construir servlets. Es por ello que una página JSP puede hacer todo lo que un servlet puede hacer, y viceversa. Cada versión de la especificación de JSP está fuertemente vinculada a una versión en particular de la especificación de servlets.

El funcionamiento general de la tecnología JSP es que el Servidor de Aplicaciones interpreta el código contenido en la página JSP para construir el código Java del servlet a generar. Este servlet será el que genere el documento (típicamente HTML) que se presentará en la pantalla del Navegador del usuario.

JSP -> Servidor Aplicaciones (Servlets) -> Cliente (Navegador)

Es posible enriquecer el lenguaje de etiquetas utilizado por JSP. Para ello debemos extender la capa de alto nivel JSP mediante la implementación de Bibliotecas de Etiquetas (Tags Libraries). Un ejemplo de estas bibliotecas son las propocionadas por Sun bajo la denominación de JSTL o las distribuidas por Apache junto con el Framework de Struts.

TagLibs -> JSP -> Servidor Aplicaciones (Servlets) -> Cliente (Navegador)

El rendimiento de una página JSP es el mismo que tendría el servidor equivalente, ya que el código es compilado como cualquier otra clase Java. A su vez, la máquina virtual compilará dinámicamente a código de máquina las partes de la aplicación que lo requieran. Esto hace que JSP tenga un buen desempeño y sea más eficiente que otras tecnologías web que ejecutan el código de una manera puramente interpretada.

La principal ventaja de JSP frente a otros lenguajes es que el lenguaje Java es un lenguaje de propósito general que excede el mundo web y que es apto para crear clases que manejen lógica de negocio y acceso a datos de una manera prolija. Esto permite separar en niveles las aplicaciones web, dejando la parte encargada de generar el documento HTML en el archivo JSP.

Otra ventaja es que JSP hereda la portabilidad de Java, y es posible ejecutar las aplicaciones en múltiples plataformas sin cambios. Es común incluso que los desarrolladores trabajen en una plataforma y que la aplicación termine siendo ejecutada en otra.

Los servlets y Java Server Pages (JSPs) son dos métodos de creación de páginas web dinámicas en servidor usando el lenguaje Java. En ese sentido son similares a otros métodos o lenguajes tales como el PHP, ASP o los CGIs, programas que generan páginas web en el servidor. Sin embargo, se diferencian de ellos en otras cosas.

Para empezar, los JSPs y servlets se ejecutan en una máquina virtual Java, lo cual permite que, en principio, se puedan usar en cualquier tipo de ordenador, siempre que exista una máquina virtual Java para él. Cada servlet (o JSP, a partir de ahora lo usaremos de forma indistinta) se ejecuta en su propia hebra, es decir, en su propio contexto; pero no se comienza a ejecutar cada vez que recibe una petición, sino que persiste de una petición a la siguiente, de forma que no se pierde tiempo en invocarlo (cargar programa + intérprete). Su persistencia le permite también hacer una serie de cosas de forma más eficiente: conexión a bases de datos y manejo de sesiones, por ejemplo.

Los JSPs son en realidad servlets: un JSP se compila a un programa en Java la primera vez que se invoca, y del programa en Java se crea una clase que se empieza a ejecutar en el servidor como un servlet. La principal diferencia entre los servlets y los JSPs es el enfoque de la programación: un JSP es una página Web con etiquetas especiales y código Java incrustado, mientras que un servlet es un programa Java puro que recibe peticiones y genera a partir de ellas una página web.

4.4.10 PERL

Perl es un lenguaje de programación diseñado por Larry Wall en 1987. Perl toma características del lenguaje C, del lenguaje interpretado shell (sh), AWK, sed, Lisp y, en un grado inferior, de muchos otros lenguajes de programación.

Estructuralmente, Perl está basado en un estilo de bloques como los del C o AWK, y fue ampliamente adoptado por su destreza en el procesado de texto y no tener ninguna de las limitaciones de los otros lenguajes de script.

4.4.10.1 CARACTERÍSTICAS

La estructura completa de Perl deriva ampliamente del lenguaje C. Perl es un lenguaje imperativo, con variables, expresiones, asignaciones, bloques de código delimitados por llaves, estructuras de control y subrutinas.

Perl también toma características de la programación shell. Todas las variables son marcadas con un sigilo precedente (sigil). Los sigilos identifican inequívocamente los nombres de las variables, permitiendo a Perl tener una rica sintaxis. Notablemente, los sigilos permiten interpolar variables directamente dentro de las cadenas de caracteres (strings). Como en los shell, Perl tiene muchas funciones integradas para tareas comunes y para acceder a los recursos del sistema.

Perl toma las listas del Lisp, hash (memoria asociativa) del AWK y expresiones regulares del sed. Todo esto simplifica y facilita todas las formas del análisis sintáctico, manejo de texto y tareas de gestión de datos.

En Perl 5, se añadieron características para soportar estructuras de datos complejas, funciones de primer orden (p. e. clausuras como valores) y un modelo de programación orientada a objetos. Éstos incluyen referencias, paquetes y una ejecución de métodos basada en clases y la introducción de variables de ámbito léxico, que hizo más fácil escribir código robusto (junto con el pragma strict). Una característica principal introducida en Perl 5 fue la habilidad de empaquetar código reutilizable como módulos. Larry Wall indicó más adelante que "la intención del sistema de módulos de Perl 5 era apoyar el crecimiento de la cultura Perl en vez del núcleo de Perl".

Todas las versiones de Perl hacen el tipificado automático de datos y la gestión de memoria. El intérprete conoce el tipo y requerimientos de almacenamiento de cada objeto en el programa; reserva y libera espacio para ellos según sea necesario. Las conversiones legales de tipo se hacen de forma automática en tiempo de ejecución; las conversiones ilegales son consideradas errores fatales.

4.4.10.2 DISEÑO

El diseño de Perl puede ser entendido como una respuesta a tres amplias tendencias de la industria informática: rebaja de los costes en el hardware, aumento de los costes laborales y las mejoras en la tecnología de compiladores. Anteriormente, muchos lenguajes de ordenador como el Fortran y C, fueron diseñados para hacer un uso eficiente de un hardware caro. En contraste, Perl es diseñado para hacer un uso eficiente de los costosos programadores de ordenador.

Perl tiene muchas características que facilitan la tarea del programador a costa de unos requerimientos de CPU y memoria mayores. Éstas incluyen gestión de memoria automática; tipo de dato dinámico; strings, listas y hashes; expresiones regulares; introspección y una función eval().

Larry Wall fue adiestrado como lingüista y el diseño de Perl ha sido muy aleccionado con principios lingüísticos. Ejemplos incluyen la Codificación Huffman (las construcciones más comunes deben ser las más cortas), buena distribución (la información importante debe ir primero) y una larga colección de primitivas del lenguaje. Perl favorece las construcciones del lenguaje, tan naturales, como para los humanos son la lectura y la escritura, incluso si eso hace más complicado al intérprete Perl.

La sintaxis de Perl refleja la idea de que "cosas que son diferentes deben parecer diferentes". Por ejemplo, escalares, arrays y hashes tienen diferente sigilo. Índices de array y claves hash usan diferentes clases de paréntesis. Strings y expresiones regulares tienen diferentes delimitadores

estándar. Esta aproximación puede contrastarse con lenguajes como Lisp, donde la misma construcción S-expresión y sintaxis básica se usa para muchos y variados propósitos.

Perl tiene características que soportan una variedad de paradigmas de programación, como la imperativa, funcional y la orientada a objetos. Al mismo tiempo, Perl no obliga a seguir ningún paradigma en particular, ni obliga al programador a elegir alguna de ellas.

Hay un amplio sentido de lo práctico, tanto en el lenguaje Perl como en la comunidad y la cultura que lo rodean. El prefacio de Programming Perl comienza con, "Perl es un lenguaje para tener tu trabajo terminado". Una consecuencia de esto es que Perl no es un lenguaje ordenado. Incluye características si la gente las usa, tolera excepciones a las reglas y emplea la heurística para resolver ambigüedades sintácticas. Debido a la naturaleza indulgente del compilador, a veces los errores pueden ser difíciles de encontrar. Hablando del variado comportamiento de las funciones internas en los contextos de lista y escalar, la página de manual de perlfunc(1) dice "En general, hacen lo que tu quieras, siempre que quieras la coherencia."

Perl tiene varios lemas que transmiten aspectos de su diseño y uso. Uno es There's more than one way to do it (Hay más de una forma de hacerlo) (TMTOWTDI, usualmente pronunciado 'Tim Toady'). Otros son "Perl: la motosierra del ejército Suizo de los lenguajes de programación" y "Límites imprecisos". Una meta prefijada de Perl es hacer las cosas fáciles de forma fácil y las tareas difíciles, posibles. A Perl también se le ha llamado "El esparadrapo de Internet".

4.4.10.3 APLICACIONES

Perl tiene muchas y variadas aplicaciones, gracias a la disponibilidad de muchos módulos estándares y de terceras partes.

Se ha usado desde los primeros días del Web para escribir guiones (scripts) CGI. Es una de las "tres Pes" (Perl, Python y PHP), que son los lenguajes más populares para la creación de aplicaciones Web, y es un componente integral de la popular solución LAMP para el desarrollo web. Grandes proyectos escritos en Perl son Slash, IMDb4 y UseModWiki, un motor de Wiki. Muchos sitios web con alto tráfico, como Amazon.com y Ticketmaster.com usan Perl extensamente.

Perl se usa a menudo como un "lenguaje pegamento", ligando sistemas e interfaces que no fueron diseñados específicamente para interoperar; y para el "escarbado de datos", convirtiendo o procesando grandes cantidades de datos para tareas como por ejemplo crear informes. De hecho, estas fortalezas están íntimamente unidas. Su combinación hace a Perl una popular herramienta de propósito general para los administradores de sistemas, especialmente en programas pequeños que pueden ser escritos y ejecutados en una sola línea de comandos.

Perl es también ampliamente usado en finanzas y bioinformática, donde es apreciado por su desarrollo rápido, tanto de aplicaciones como de despliegue, así como la habilidad de manejar grandes volúmenes de datos.

4.4.10.4 DISPONIBILIDAD

Perl es software libre y está licenciado bajo la Licencia Artística y la GNU General Public License. Existen distribuciones disponibles para la mayoría de sistemas operativos. Está

especialmente extendido en Unix y en sistemas similares, pero ha sido portado a las plataformas más modernas (y otras más obsoletas). Con sólo seis excepciones confirmadas, puede ser compilado desde el código fuente en todos los Unix, compatibles POSIX o cualquier otra plataforma Unix compatible. Sin embargo, esto no es normalmente necesario, porque Perl está incluido por defecto en la instalación de los sistemas operativos más populares.

Debido a los cambios especiales necesarios para soportar al Mac OS Classic, existe una adaptación especial llamada MacPerl.

4.4.10.4.1 GNU/LINUX

Perl está instalado por defecto en las distribuciones más populares de GNU/Linux incluyendo Gentoo, Slackware, Mandriva, Debian, RedHat y SUSE.

4.4.10.4.2 WINDOWS

Los usuarios de Microsoft Windows normalmente instalan una distribución binaria de Perl. Compilar Perl desde el código fuente bajo Windows es posible, pero la mayoría de las instalaciones no disponen del necesario compilador de C.

La capa de emulación Cygwin proporciona otra forma de correr Perl bajo Windows. Cygwin proporciona en entorno parecido al Unix en Windows que incluye gcc, por lo que compilar Perl desde el código es una opción accesible para los usuarios que prefieren esta opción.

En junio de 2006, win32.perl.org fue lanzado por Adam Kennedy en nombre de la Fundación Perl. Es una comunidad web "para todo lo relacionado con Windows y Perl".

4.4.11 COLDFUSION

En computación, Coldfusion (Adobe ColdFusion) es un servidor de aplicaciones y un lenguaje de programación usado para desarrollar aplicaciones de Internet, generalmente sitios web generados dinámicamente. En este aspecto, es un producto similar a ASP, JSP o PHP.

ColdFusion es una herramienta que corre en forma concurrente con la mayoría de los servidores web de Windows, Mac OS X, Linux y Solaris (también en servidores web personales en Windows 98 y puede ser usado para intranets). El servidor de aplicaciones web de ColdFusion trabaja con el servidor HTTP para procesar peticiones de páginas web. Cada vez que se solicita una página de ColdFusion, el servidor de aplicaciones ColdFusion ejecuta el guion o programa contenido en la página.

ColdFusion es un lenguaje de programación, puede crear y modificar variables igual que en otros lenguajes de programación que nos son familiares. Posee control de flujo de programas, como IF, Case, ciclo, etc. Tiene muchas funciones built-in para realizar tareas más complicadas, por ejemplo: para averiguar qué día de la semana será el 3 de agosto del 2027 :

```
DayOfWeekAsString(DayOfWeek('2027/08/03'))
```

No es un lenguaje de bases de datos, pero interactúa de manera simple con bases de datos (Sybase, Oracle, MySQL, SQL Server, o Access). Usando SQL estándar, las páginas y aplicaciones web pueden fácilmente recuperar, guardar, formatear y presentar información dinámicamente.

Muchas de las funciones poderosas de ColdFusion, como leer desde y escribir en discos duros del servidor, son basadas en tags. Así como el tag puede tener argumentos como 'width'o 'align', el tag <CFFILE>tiene argumentos que especifican 'action=read/write/copy/delete', path=' etc. El tag <CFFORM>construye automáticamente todo el código JavaScript para verificar los campos requeridos antes de hacer el formulario. ColdFusion también tiene tags para COM, Corbay Appletsy Servletsde Java. ColdFusion fue diseñado para desarrollar sitios complejos y de alto tráfico. ColdFusion está diseñado para correr en máquinas multi-procesador, y permite construir sitios que pueden correr en clusters de servidores. Es un lenguaje que se ejecuta en el servidor. A diferencia de JavaScript y Applets Java, que se ejecuta en el cliente, ColdFusion se ejecuta en el servidor web. Esto significa que los guiones escritos en ColdFusion correrán de la misma manera en cualquier navegador web. Coldfusion tiene problemas de inestabilidad y es capaz de soportar poca carga[cita requerida]. Este problema sólo puede solucionarse poniendo gran cantidad de servidores web balanceados entre sí.

4.4.12 RUBY

Ruby es un lenguaje de programación interpretado, reflexivo y orientado a objetos, creado por el programador japonés Yukihiro "Matz" Matsumoto, quien comenzó a trabajar en Ruby en 1993, y lo presentó públicamente en 1995. Combina una sintaxis inspirada en Python y Perl con características de programación orientada a objetos similares a Smalltalk. Comparte también funcionalidad con otros lenguajes de programación como Lisp, Lua, Dylan y CLU. Ruby es un lenguaje de programación interpretado en una sola pasada y su implementación oficial es distribuida bajo una licencia de software libre.

4.4.12.1 SEMÁNTICA

Ruby es orientado a objetos: todos los tipos de datos son un objeto, incluidas las clases y tipos que otros lenguajes definen como primitivas, (como enteros, booleanos, y "nil"). Toda función es un método. Las variables siempre son referencias a objetos, no los objetos mismos. Ruby soporta herencia con enlace dinámico, mixins y métodos singleton (pertenecientes y definidos por un sola instancia más que definidos por la clase). A pesar de que Ruby no soporta herencia múltiple, la clases pueden importar módulos como mixins. La sintaxis procedural está soportada, pero todos los métodos definidos fuera del ámbito de un objeto son realmente métodos de la clase Object. Como esta clase es padre de todas las demás, los cambios son visibles para todas las clases y objetos.

Ruby ha sido descrito como un lenguaje de programación multiparadigma: permite programación procedural (definiendo funciones y variables fuera de las clases haciéndolas parte del objeto raíz Object), con orientación a objetos, (todo es un objeto) o funcionalmente (tiene funciones anónimas, clausuras o closures, y continuations; todas las sentencias tiene valores, y las funciones devuelven la última evaluación). Soporta introspección, reflexión y metaprogramación, además de soporte para hilos de ejecución gestionados por el intérprete. Ruby tiene tipado dinámico, y soporta polimorfismo de tipos (permite tratar a subclases utilizando la interfaz de la clase padre). Ruby no

requiere de polimorfismo de funciones al no ser fuertemente tipado (los parámetros pasados a un método pueden ser de distinta clase en cada llamada a dicho método).

De acuerdo con las preguntas frecuentes de Ruby, "Si te gusta Perl, te gustará Ruby y su sintaxis. Si te gusta Smalltalk, te gustará Ruby y su semántica. Si te gusta Python, la enorme diferencia de diseño entre Python y Ruby/Perl puede que te convenza o puede que no."

4.4.12.2 CARACTERÍSTICAS

- Orientado a objetos.
- Cuatro niveles de ámbito de variable: global, clase, instancia y local.
- Manejo de excepciones.
- Iteradores y clausuras o closures (pasando bloques de código).
- Expresiones regulares nativas similares a las de Perl a nivel del lenguaje.
- Posibilidad de redefinir los operadores (sobrecarga de operadores).
- Recolección de basura automática.
- Altamente portable.
- Hilos de ejecución simultáneos en todas las plataformas usando green threads.
- Carga dinámica de DLL/bibliotecas compartidas en la mayoría de las plataformas.
- Introspección, reflexión y metaprogramación.
- Amplia librería estándar.
- Soporta inyección de dependencias.
- Soporta alteración de objetos en tiempo de ejecución.
- Continuaciones y generadores.

Ruby actualmente no tiene soporte completo de Unicode, a pesar de tener soporte parcial para UTF-8.

4.4.13 CGI

Interfaz de entrada común (en inglés Common Gateway Interface, abreviado CGI) es una importante tecnología de la World Wide Web que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor web y una aplicación externa cuyo resultado final de la ejecución son objetos MIME. Las aplicaciones que se ejecutan en el servidor reciben el nombre de CGIs.

Las aplicaciones CGI fueron una de las primeras prácticas de crear contenido dinámico para las páginas web. En una aplicación CGI, el servidor web pasa las solicitudes del cliente a un programa externo. Este programa puede estar escrito en cualquier lenguaje que soporte el servidor, aunque por razones de portabilidad se suelen usar lenguajes de script. La salida de dicho programa es enviada al cliente en lugar del archivo estático tradicional.

CGI ha hecho posible la implementación de funciones nuevas y variadas en las páginas web, de tal manera que esta interfaz rápidamente se volvió un estándar, siendo implementada en todo tipo de servidores web.

4.4.13.1 FORMA DE ACTUACIÓN DE CGI

A continuación se describe la forma de actuación de un CGI de forma esquemática:

- En primera instancia, el servidor recibe una petición (el cliente ha activado un URL que contiene el CGI), y comprueba si se trata de una invocación de un CGI.
- Posteriormente, el servidor prepara el entorno para ejecutar la aplicación. Esta información procede mayoritariamente del cliente.
- Seguidamente, el servidor ejecuta la aplicación, capturando su salida estándar.
- A continuación, la aplicación realiza su función: como consecuencia de su actividad se va generando un objeto MIME que la aplicación escribe en su salida estándar.
- Finalmente, cuando la aplicación finaliza, el servidor envía la información producida, junto con información propia, al cliente, que se encontraba en estado de espera. Es responsabilidad de la aplicación anunciar el tipo de objeto MIME que se genera (campo CONTENT_TYPE).

4.4.13.2 PROGRAMACIÓN DE UN CGI

Un programa CGI puede ser escrito en cualquier lenguaje de programación que produzca un fichero ejecutable. Entre los lenguajes más habituales se encuentran: C, C++, Perl, Java, Visual Basic... No obstante, debido a que el CGI recibe los parámetros en forma de texto será útil un lenguaje que permita realizar manipulaciones de las cadenas de caracteres de una forma sencilla, como por ejemplo Perl. Perl es un lenguaje interpretado que permite manipulaciones sencillas de ficheros y textos, así como la extracción y manipulación de cadenas de caracteres, unidas a unas búsquedas rápidas y fáciles.

4.4.13.3 TIPOS HABITUALES DE CGIS

- **Contador de accesos:** Cuenta el número de veces que se ha solicitado una página determinada. Se guarda el valor en un fichero. Cada vez que se invoca se incrementa, para su posterior visualización.

- **Buscador:** Localiza páginas que contengan los términos especificados. Utiliza una tabla que enumera las palabras y para cada una especifica las páginas dónde se encuentra.
- **Correo:** Obtiene información estructurada del usuario.
- **Contribuciones:** Permite añadir enlaces o anotaciones a una página, indicando la procedencia de la adición.
- **Estadísticas de uso:** Presenta información sobre los acontecimientos producidos en el servidor de WWW. El servidor mantiene un registro (log) de los acontecimientos que se han producido.
- **Administración remota del servidor:** Permite interactuar con el servidor desde WWW. Invoca los programas que controlan o modifican el comportamiento del servidor.

4.5 POPULARIDAD

A la hora de analizar los lenguajes es importante tener en cuenta su popularidad, ya que es un factor crucial para el futuro desarrollo de nuevas versiones, nuevas funcionalidades y la adaptación a nuevas tecnologías. Dada la importancia de éste factor analizaremos la tabla de los lenguajes de programación más populares y extraeremos los lenguajes web que nos interesan para este estudio. En la siguiente tabla vemos los 20 lenguajes de programación más populares en Agosto de 2011 y cuál era su posición el año anterior.

Position Aug 2011	Position Aug 2010	Delta in Position	Programming Language	Ratings Aug 2011	Delta Aug 2010
1	1	=	Java	19.409%	+1.42%
2	2	=	C	17.390%	-0.48%
3	3	=	C++	8.433%	-1.23%
4	4	=	PHP	6.134%	-3.05%
5	6	↑	C#	6.042%	+1.06%
6	9	↑↑↑	Objective-C	5.494%	+2.34%
7	5	↓↓	(Visual) Basic	5.013%	-0.40%
8	7	↓	Python	3.415%	-0.81%
9	8	↓	Perl	2.315%	-1.11%
10	11	↑	JavaScript	1.557%	-0.84%
11	23	↑↑↑↑↑↑↑↑↑↑	Lua	1.362%	+0.83%
12	12	=	Ruby	1.329%	-0.65%
13	10	↓↓↓	Delphi/Object Pascal	1.076%	-1.35%
14	16	↑↑	Lisp	0.905%	+0.28%
15	22	↑↑↑↑↑↑↑	Transact-SQL	0.823%	+0.27%
16	28	↑↑↑↑↑↑↑↑↑↑	Ada	0.699%	+0.30%
17	19	↑↑	RPG (OS/400)	0.660%	+0.05%
18	17	↓	Pascal	0.659%	+0.04%
19	46	↑↑↑↑↑↑↑↑↑	F#	0.604%	+0.37%
20	-	=	Assembly*	0.599%	-

De ésta tabla podemos inferir que en lo que concierne a tecnologías web los puestos más altos los ocupan PHP y la tecnología Python. Aunque algunos de los lenguajes que están alrededor

de las primeras posiciones tienen un alto porcentaje de uso podemos presuponer que son para aplicaciones no ligadas a la web, como es el ejemplo de Java, que es líder en el sector empresarial y aunque tenga su parte y en muchos casos se implemente la interfaz con lenguajes HTML y CSS con Java embebido, la potencialidad buscada en este lenguaje radica en la capa lógica y no en la de vista, que es en la que esta interesado este estudio.

4.6 SELECCIÓN PREVIA DE LENGUAJES

A continuación voy a presentar los lenguajes que serán objeto de estudio directo e implementación y uno por uno describiré las razones que han llevado a esta selección. Serán vitales varios factores para esta selección, tales como su popularidad actual, la posibilidad de que exista un ciclo de vida largo para los lenguajes elegidos, una evolución previsible de cara a la adaptación de nuevas tecnologías y unas bases sólidas de confianza y seguridad para embarcarse en proyectos de cierta envergadura y seriedad.

4.6.1 HTML

HTML es el pilar de la programación web, supone toda base y estructura la página, por ende era un lenguaje que no se puede dejar fuera, de hecho, será analizado en primer lugar como ejemplo de lenguaje estático y se comentarán las posibilidades que ofrece la nueva versión y sus posibles repercusiones.

Según las tendencias que se han sido siguiendo de la web parece evidente que HTML se seguirá usando por mucho tiempo como base de las páginas web y se deducen esfuerzos por actualizarlo y ofrecer nuevas funcionalidades como el caso de la versión 5 que se comentó en la sección anterior.

Parece indubitable la perdurabilidad de este lenguaje y de su complemento para el diseño CSS que será comentado a continuación.

4.6.2 CSS

Al igual que HTML es el pilar estructural de la página web, CSS es el pilar de su diseño. Es un lenguaje especializado en ofrecer un amplio abanico de configuraciones para el estilo de las páginas, los botones, los menús... De hecho, se usa también para el diseño de otras intefaces no relacionadas con la web. Por ésto, se puede tener la certidumbre de que ni CSS ni HTML serán sustituidos, al menos en un plazo de tiempo corto o medio.

Estos dos lenguajes serán los primeros en ser analizados juntos y darán base estructural y de diseño para los siguientes dos análisis de lenguajes dinámicos que a su vez aportarán funcionalidades que desde el punto de vista estático son imposibles de implementar.

4.6.3 PHP

PHP es un lenguaje que simplemente por popularidad se merecía ser analizado, lleva en el ranking de los 5 lenguajes más populares y a su vez, es el representante web de este ranking. Es muy utilizado y combina las ventajas de ser un lenguaje no demasiado reciente pero sobretodo muy utilizado, lo que conlleva unas grandes librerías y buenas adaptaciones a diferentes tecnologías y métodos de programación.

4.6.4 PSP

El siguiente lenguaje a analizar es Python, en concreto su parte para desarrollo web, que será implementada usando un módulo para el servidor Apache llamado mod_python y servirá para poder embeber código python en HTML como hace PHP. Este lenguaje web es el siguiente en popularidad y fue lenguaje del año 2010.

Parece que es un lenguaje que a nivel empresarial va cobrando más importancia y con una filosofía muy especial que fue comentada en la apartado anterior.

4.7 ESTUDIO DE LA WEB

En esta sección se analizará la evolución que ha tenido la web, primeramente se verá su cronología, después un poco de su historia y por último las diferentes versiones que se han ido desarrollando:

4.7.1 CRONOLOGÍA DE INTERNET

1969: DARPA comienza a planificar la creación de una red que conecte computadores en caso de una eventual guerra atómica que incomunique a los humanos sobre la tierra, con fines principalmente de defensa.

1972: se realizó la Primera demostración pública de ARPANET, una nueva Red de comunicaciones financiada por la DARPA que funcionaba de forma distribuida sobre la red telefónica conmutada.

1986: la NSF comenzó el desarrollo de NSFNET que se convirtió en la principal Red en árbol de Internet, complementada después con las redes NSINET y ESNET, todas ellas en Estados Unidos. Paralelamente, otras redes troncales en Europa, tanto públicas como comerciales, junto con las americanas formaban el esqueleto básico (\"backbone\") de Internet.

1989: En el CERN de Ginebra, un grupo de Físicos encabezado por Tim Berners-Lee, crearon el lenguaje HTML, basado en el SGML.

En **1990** el mismo equipo construyó el primer cliente Web, llamado WorldWideWeb (WWW), y el primer servidor web. La intención original era hacer más fácil el compartir textos de investigación entre científicos y permitir al lector revisar las referencias de un artículo mientras lo fuera leyendo.

2004: Nace el concepto Web 2.0. Dale Dougherty de O'Reilly Media utilizó este término en una conferencia en la que compartió una lluvia de ideas junto a Craig Cline de MediaLive en la que hablaba del renacimiento y evolución de la web.

2007: Comienza la transformación hacia la Web 3.0.

4.7.2 HISTORIA

La idea subyacente de la Web se remonta a la propuesta de Vannevar Bush en los años 40 sobre un sistema similar: un entramado de información distribuido con un interface operativo que permite el acceso tanto a la misma como a otros artículos relevantes 2 determinados por claves. Este proyecto nunca fue materializado, quedando en el plano teórico bajo el nombre de MEMEX. Es en los años 50 cuando Ted Nelson realiza la primera referencia a un sistema de hipertexto, donde la información es enlazada de forma libre. Pero no es hasta 1980, con un soporte operativo tecnológico para la distribución de información en redes informáticas, cuando Tim Berners-Lee propone ENQUIRE al CERN (refiriéndose a Enquire Within Upon Everything, en castellano Preguntando de Todo Sobre Todo), donde se materializa la realización práctica de este concepto de incipientes nociones de la Web. En marzo de 1989, Tim Berners Lee, ya como personal de la división DD del CERN, redacta la propuesta, que referenciaba a ENQUIRE y describía un sistema de gestión de información más elaborado. El World Wide Web ya había nacido. Con la ayuda de Robert Cailliau, se publicó una propuesta más formal para la World Wide web el 12 de noviembre de 1990. Berners-Lee usó un NeXTcube como el primer servidor web del mundo y también escribió el primer navegador web, WorldWideWeb en 1990. También había creado todas las herramientas necesarias para que una web funcionase: el primer navegador web (el cual también era un editor web), el primer servidor web y las primeras páginas web que al mismo tiempo describían el proyecto.

El 6 de agosto de 1991, envió un pequeño resumen del proyecto World Wide Web al newsgroup alt.hypertext. Esta fecha también señala el debut de la web como un servicio disponible públicamente en Internet. El gran avance de Berners-Lee fue unir hipertexto e Internet. En el proceso, desarrolló un sistema de identificadores únicos globales para los recursos web y también: el Uniform Resource Identifier (URI). A diferencia de sus predecesores, como HyperCard, World Wide Web era no-propietario, haciendo posible desarrollar servidores y clientes independientemente y añadir extensiones sin restricciones de licencia. El 30 de abril de 1993, el CERN anunció que la web sería gratuita para todos. ViolaWWW fue un navegador bastante popular en los comienzos de la web que estaba basado en el concepto de la herramienta hipertextual de software de Mac denominada HyperCard.

4.7.2.1 WEB 1.0

La Web 1.0 empezó en los años 60's de la forma más básica que existe, con navegadores de sólo texto bastante rápidos como ELISA. Después surgió el HTML (**H**yper **T**ext **M**arkup **L**anguage) que hizo las páginas web más agradables a la vista, así como los primeros navegadores visuales tales como IE, Netscape, etc.

La Web 1.0 es de sólo lectura. El usuario no puede interactuar con el contenido de la página (nada de comentarios, respuestas, citas, etc.) estando totalmente limitado a lo que el Webmaster sube a ésta. Web 1.0 es una frase que se refiere a un estado de la World Wide Web, y cualquier página Web diseñada con un estilo anterior del fenómeno de la Web 2.0. Es en general un término que ha sido creado para describir la Web antes del impacto de la fiebre punto com en el 2001, que es visto por muchos como el momento en que el Internet dio un giro. Es la forma más fácil en el sentido del término Web 1.0 cuando es usada en relación a término Web 2.0.

4.7.2.1.1 CARACTERÍSTICAS

Terry Flew, en la tercera edición de *New Media* describe cual cree que son las diferencias que caracterizan a la Web 1.0 y a la Web 2.0.

> "move from personal websites to blogs and blog site aggregation, from publishing to participation, from web content as the outcome of large up-front investment to an ongoing and interactive process, and from content management systems to links based on tagging (folksonomy)"

Sintetizando, el concepto original de la Web (en este contexto, llamada Web 1.0) era páginas estáticas HTML que no eran actualizadas frecuentemente. El éxito de éstas dependía de webs más dinámicas (a veces llamadas Web 1.5) donde los CMS servían páginas HTML dinámicas creadas desde una actualizada base de datos. En ambos sentidos, el conseguir visitas y la estética visual eran considerados factores muy importantes.

La información se movía unidireccionalmente: de las webs a las personas. Apenas cambiaba, aunque adoptaba diferentes formas: webs personales, de empresa, etc. Los foros de opinión se convirtieron muy pronto en una excepción a esto. Este planteamiento termina con el pinchazo tecnológico de 2001, y una lección: lo importante no es el contenido, sino los ingresos. El modelo 'Web 1.0′ funcionará siempre para determinadas necesidades.

4.7.2.2 WEB 2.0

Se refiere a una segunda generación de Web basada en comunidades de usuarios y una gama especial de servicios, como las redes sociales, los blogs, los wikis o las folcsonomías, que fomentan la colaboración y el intercambio ágil de información entre los usuarios, donde el estaticismo de las páginas pasa a transformarse en una matriz de interacción del usuario con la Red pudiendo él mismo incluir su propia información en el sistema, creando o no webs interactivas y visuales. Es decir, los sitios Web 2.0 actúan más como puntos de encuentro, o webs dependientes de usuarios, que como webs tradicionales.

Los teóricos de la aproximación a la Web 2.0 creen que el uso de la Web está orientado a la interacción y redes sociales, que pueden servir contenido que explota los efectos de las redes, creando o no webs interactivas y visuales. Es decir, los sitios Web 2.0 actúan más como puntos de encuentro, o webs dependientes de usuarios, que como webs tradicionales.

La Web 2.0 es la transición que se ha dado de aplicaciones tradicionales hacia aplicaciones que funcionan a través del Web enfocadas al usuario final. Se trata de aplicaciones que generen colaboración y de servicios que reemplacen las aplicaciones de escritorio. Es una etapa que ha definido nuevos proyectos en Internet y está preocupándose por brindar mejores soluciones para el usuario final. Muchos aseguran que hemos reinventado lo que era el Internet, otros hablan de burbujas e inversiones, pero la realidad es que la evolución natural del medio realmente ha propuesto cosas más interesantes como lo analizamos diariamente en las notas de actualidad.

En general, cuando mencionamos el término Web 2.0 nos referimos a una serie de aplicaciones y páginas de Internet que utilizan la inteligencia colectiva para proporcionar servicios interactivos en red dando al usuario el control de sus datos.

Así, podemos entender como 2.0 -"todas aquellas utilidades y servicios de Internet que se sustentan en una base de datos, la cual puede ser modificada por los usuarios del servicio, ya sea en su contenido (añadiendo, cambiando o borrando información o asociando datos a la información existente), pues bien en la forma de presentarlos, o en contenido y forma simultáneamente."- (Ribes, 2007).

4.7.2.2.1 CARACTERÍSTICAS

Los protocolos de mensajes bidireccionales son uno de los elementos clave de la infraestructura de la Web 2.0. Los dos tipos más importantes son los métodos RESTful y SOAP. REST indican un tipo de llamada a un servicio Web donde el cliente transfiere el estado de todas las transacciones. SOAP y otros métodos similares dependen del servidor para retener la información de estado. En ambos casos, el servicio es llamado desde un API. A veces este API está personalizado en función de las necesidades específicas del sitio Web, pero los APIs de los servicios Web estándares (como por ejemplo escribir en un blog) están también muy extendidos. Generalmente el lenguaje común de estos servicios Web es el XML, si bien puede haber excepciones.

Recientemente, una forma híbrida conocida como Ajax ha evolucionado para mejorar la experiencia del usuario en las aplicaciones Web basadas en el navegador. Esto puede ser usado en webs propietarias (como en Google Maps) o en formas abiertas utilizando un API de servicios Web.

La funcionalidad de Web 2.0 se basa en la arquitectura existente de servidor Web pero con un énfasis mayor en el software dorsal. La redifusión sólo se diferencia nominalmente de los métodos de publicación de la gestión dinámica de contenido, pero los servicios Web requieren normalmente un soporte de bases de datos y flujo de trabajo mucho más robusto y llegan a parecerse mucho a la funcionalidad de Internet tradicional de un servidor de aplicaciones. El enfoque empleado hasta ahora por los fabricantes suele ser bien un enfoque de servidor universal, el cual agrupa la mayor parte de la funcionalidad necesaria en una única plataforma de servidor, o bien un enfoque plugin de servidor Web con herramientas de publicación tradicionales mejoradas con interfaces API y otras herramientas. Independientemente del enfoque elegido, no se espera que el camino evolutivo hacia la Web 2.0 se vea alterado de forma importante por estas opciones.

4.7.2.3 WEB 3.0

En ocasiones se ha relacionado el término *Web 2.0* con el de Web semántica. Sin embargo ambos conceptos, corresponden más bien a estados evolutivos de la Web, y la Web semántica correspondería en realidad a una evolución posterior, a la Web 3.0 o Web inteligente. La combinación de sistemas de redes sociales como Facebook, Twitter, FOAF y XFN, con el desarrollo de etiquetas (o *tags*), que en su uso social derivan en folcsonomías, así como el plasmado de todas estas tendencias a través de blogs y wikis, confieren a la Web 2.0 un aire semántico sin serlo realmente. Sin embargo, en el sentido más estricto para hablar de Web semántica, se requiere el uso de estándares de metadatos como Dublin Core y en su forma más elaborada de ontologías y no de

folcsonomías. De momento, el uso de ontologías como mecanismo para estructurar la información en los programas de blogs es anecdótico [5] y sólo se aprecia de manera incipiente en algunos wikis.

Por tanto podemos identificar la Web semántica como una forma de Web 3.0. Existe una diferencia fundamental entre ambas versiones de Web (2.0 y semántica) y es el tipo de participante y las herramientas que se utilizan. La 2.0 tiene como principal protagonista al usuario humano que escribe artículos en su blog o colabora en un wiki. El requisito es que además de publicar en HTML emita parte de sus aportaciones en diversos formatos para compartir esta información como son los RSS, ATOM, etc. mediante la utilización de lenguajes estándares como el XML.

Con esta nueva transformación se permitirá la actualización y el dinamismo perpetuo a través de una interacción constructivista y organizativa de contenidos y estructuras por parte del usuario. El término Web 3.0 es asociado por la prensa generalista al concepto de Web semántica que se está desarrollando bajo la tutela de Tim Berners-Lee, el mismo que inventó la Web a principios de los 90. Las características que diferencian esta etapa de las anteriores se podrían resumir en:

1. Transformación de la estructura Web actual en la de Web semántica.

2. Utilización de Inteligencia Artificial en combinación con la nueva estructura.

3. Prevalencia del usuario en la creación, organización y rendimiento del contenido a través de un modelo de cooperación globalizada: Joost, Mechanical Turk Amazon, Google Image Labeler.

4. Potenciación de nuevas formas de ocio y comunicación entre usuarios: Skype, World of Warcraft.

4.7.2.3.1 CARACTERÍSTICAS

El primer paso hacia la "Web 3.0" es el nacimiento de la "Data Web", ya que los formatos en que se publica la información en Internet son dispares, como XML, RDF y microformatos; el reciente crecimiento de la tecnología SPARQL, permite un lenguaje estandarizado y API para la búsqueda a través de bases de datos en la red. La "Data Web" permite un nuevo nivel de integración de datos y aplicación inter-operable, haciendo los datos tan accesibles y enlazables como las páginas Web. La "Data Web" es el primer paso hacia la completa “Web Semántica”. En la fase “Data Web”, el objetivo es principalmente, hacer que los datos estructurados sean accesibles utilizando RDF. El escenario de la "Web Semántica" ampliará su alcance en tanto que los datos estructurados e incluso, lo que tradicionalmente se ha denominado contenido semi-estructurado (como páginas Web, documentos, etc.), estén disponible en los formatos semánticos de RDF y OWL.

Suma de la inteligencia artificial, utilizada para describir el camino evolutivo de la red que la conduce. Algunos escépticos lo ven como una visión inalcanzable. Sin embargo, compañías como IBM y Google están implementando nuevas tecnologías que cosechan información sorprendente, como el hecho de hacer predicciones de canciones que serán un éxito, tomando como base información de las webs de música de la Universidad. Existe también un debate sobre si la fuerza conductora tras Web 3.0 serán los sistemas inteligentes, o si la inteligencia vendrá de una forma más orgánica, es decir, de sistemas de inteligencia humana, a través de servicios colaborativos como del.icio.us, Flickr y Digg, que extraen el sentido y el orden de la red existente y cómo la gente interactúa con ella.

4.7.2.4 WEB 4.0

Este término motiva a pensar qué será ese tipo de Web, por ahora algunos señalan que el resultado de Web 3D + Web 3.0 (web semántica) + Inteligencia Artificial + Voz como vehículo de intercomunicación= Web 4.0 (web total) es decir que una vez se establezca esta web semántica (entre el año 2010 y el 2020) será el turno de avanzar hacia la web 4.0 en la que el sistema operativo establecido en la web cobre protagonismo, hacia una Web Ubicua, donde el objetivo primordial será el de unir las inteligencias donde tanto las personas como las cosas se comuniquen entre sí para generar la toma de decisiones. Para el 2020 se espera que haya agentes en la Web que conozcan, aprendan y razonen como lo hacemos las personas.

4.7.3 TENDENCIAS SOCIALES DE LA WEB

Es curioso cómo el ser humano utiliza la tecnología, la ciencia. La evolución de ésta siempre ha estado marcada por la parte más humana y social, casi instintiva. Se inventa la televisión, un medio de difusión de información de masas, un gran invento y dentro de ciertos límites bien usado en un primer momento, y hoy por hoy ya no sólo como su temido reverso, como medio manipulativo, sino para ofrecer entretenimiento social, como periodismo rosa o reality-shows. Porque el ser humano no deja de ser un ser social, y eso le marca profundamente, le mueve subconscientemente. Antes de saber cómo funciona un tubo de rayos catódicos prefirió saber con quién se casaba tal persona o como interactuaban otras personas en ciertas circunstancias.

Los inventos se crean con un cierto objetivo, o quizás, como tantas veces a sucedido son producto del más disparatado azar, aunque en éstos casos el inventor, trata de darle un uso racionala su invento, cosa que el mercado y las corrientes de la masa de la sociedad acepta gustosamente pero adorna. Y lo adorna a veces tanto que se acaba olvidando el objetivo primario, o quizás sólo lo desplaza, o en el mejor de los casos lo adorna.

Internet surge como un medio potentísimo de transmisión de información, en un principio entre universidades y con fines militares y a posteriori para uso público. Hasta llegar al punto de hoy en día que es cuando la mayor tecnología de información que ha creado el hombre es sobretodo y con más diferencia usada para las redes sociales. Es cierto que no es lo único y es muy usada en el entorno laboral, en la búsqueda de información y para el ocio. Pero es curioso como la vertiente social es la que más ha llegado a llamarle la atención al ser humano, se mete en un mundo virtual cuasi infinito de información pero lo que más le interesa es saber cómo está su compañero al que podrá ver al día siguiente, o que ha comido su pareja, con la que dormirá en unas horas.

Vemos dos líneas que convergen, por un lado la tendencia humana a la sociabilización, a demostrar que es un animal social y que lo necesita, y por otro lado la tecnología que ha creado internet y las redes sociales. Esta convergencia resulta inexorablemente en grandes cambios en las vidas humanas. Ya no puedes pensar que es una lástima que se haya perdido el contacto con tal persona a la que hace mucho tiempo que no ves y dibujar una sonrisa torcida pensando que cuantas vueltas da el mundo porque la tienes entre tus contactos y en el momento que quieras puedes quedar con ella. Eres capaz de contactar con cientos de amigos en cuestión de pocos minutos… es más, ahora dispones de cientos de amigos, porque aunque tu círculo social cercano va cambiando como ha pasado desde todos los tiempos, ahora esas personas se van acumulando en tu red social… Hay personas a las que quieres tener cerca, saber de ellas y que sepan de ti; otras que son amigos de un amigo que quizás sólo has visto durante un par de horas y las cuales tienen la misma posibilidad de

acceder a tu información que las más cercanas. El concepto “amigo” está cambiando, lo están cambiando las redes sociales.

A la hora de buscar pareja, de enamorarse, lo primero que se hace es recurrir al perfil de la persona elegida, estudiar sus fotos, cómo se divierte, que gustos y preferencias tiene, si ya tiene pareja, etc. Se puede pseudo-conocer a una persona sin conocerla personalmente. Esa persona apenas puede cambiar drásticamente de personalidad porque uno es en gran medida lo que los demás esperan que sea y los demás cada vez tienen una idea más detallada de cómo eres, dejando cada vez menos margen a la mutabilidad natural de la personalidad. Quedará como cosa del pasado el poder mudarte de ciudad y decidir ser otra persona, dejar atrás los errores del pasado. Esta escapatoria quedará para las películas de ciencia ficción en las que al final se acabará encontrando el perfil original de la persona y obligándola a seguir con su viejo yo.

La red está influyendo decisivamente en las vidas de las personas, social y psicológicamente, lo cual es lógico, ya que si pensamos desde un punto de vista estrictamente biológicos, somos entidades cerradas que dejando la alimentación y el crecimiento que conlleva, lo demás lo percibimos por medio de los sentidos, que al fin y al cabo se traduce en información. Y la red es una revolución en la información. La influencia era predecible e ineludible.

Además la expansión y proliferación web es increíblemente rápida llegando a ser, hace unos años, en el 2006, 600.000 millones de páginas, accedidas o bien mediantes URL o bien mediante otra página. Quedando como resultado 6 páginas por persona viva en el planeta.

4.7.4 TENDENCIAS ARTÍSTICAS DE LA WEB

Como cambio profundo en la sociedad por medio de internet y de la web en concreto, el ser humano ha empezado a desarrollar actividades sociales, psicológicas y expresiones artísticas. Referenciando a éste último aspecto han ido surgiendo nuevas tendencias englobadas en lo que se conoce como Net Art.

Englobados en éste término existen cuatro corrientes principales:

4.7.4.1 BROWSER ART

Los browsers se utilizan como navegadores creativos, herramientas alternativas generadas por artistas y programadores que permiten nuevas formas de visualizar y estructurar las páginas web, dejando a la vista las estructuras compositivas ocultas de la información. Es un tipo de arte diseñado exclusivamente para la red. Utiliza interfaces que rechazan los parámetros establecidos de los navegadores comerciales como son Internet Explorer o Netscape.

4.7.4.2 TRABAJOS DE BROWSER ART

ZNC browser (Peter Luining / 2002/ http://znc.ctrlaltdel.org)

Este browser traduce el código Html en código numérico ASCII y este código es utilizado para generar tonos y crear colores. Cada carácter (letra o signo) de una página Html se traduce a un número ASCII, y cada número ASCII corresponde a un tono y color

determinado. De esta manera se generan estados sonoros y visuales únicos para cada página.

Webtracer (Tom Betts / 2001/ www.nullpointer.co.uk/-/werbtracer2.htm)

Permite obtener información acerca de la estructura del sitio analizando las intenciones y principios utilizados en la construcción y diseño del mismo. Con este browser se visualiza la estructura de la red como un diagrama molecular en 3D, mostrando las páginas como nodos o átomos, y los links como líneas interconectadas con los nodos. La estructura puede ser examinada desde cualquier ángulo y distancia, y los nodos pueden ser seleccionados para acceder a más información acerca de su contenido. Esta aplicación refleja claramente la secuencia de sus hiperlinks entre las páginas y la estructura interna del servidor. De esta manera deja al desnudo las tendencias del diseñador de la información.

Wrong Browser (Grupo JODI / 2001/ http://wrongbrowser.com)

Este navegador interpreta los sitios (urls) de una manera diferente a los navegadores convencionales. Desafían el uso del navegador común interlinkeando aleatoriamente los diferentes dominios, pero restringiendo la navegación a dominios de tres letras (llama al azar direcciones del tipo hal, 3ds, zkm, etc.).

Al entrar a la página da la opción de elegir que extensión de dominio se desea utilizar. Hay cinco posibilidades: .com, .co.kr, .co.jp, .nl y .org, y también da la opción de bajar el browser para Macintosh o de Windows. Al ejecutar el programa los dominios son presentados en crudo (Html) y simultáneamente superpuestos entre sí de forma caótica, con distintos colores y transparencias, formando un collage interactivo. Se puede interactuar coloreando o arrastrando el texto y moviendo los distintos urls mediante unos nodos.

4.7.4.3 SOFTWARE ART

En el software art el artista es el programador. El software no es utilizado sólo como un instrumento funcional sino que se considera como una creación artística, en la que al material estético es el código generado, y la forma expresiva es la programación del software. Deja en transparencia su funcionamiento y programación, para cuestionarlo y redefinirlo como un artefacto cultural que actúa dentro de la cultura modificando nuestra manera de ver la sociedad, y a su vez modificándose a sí mismo en comunicación con el entorno.

4.7.4.4 GAME ART

El Game Art se basa en las modificaciones o alteraciones de los juegos de ordenador ya existentes (gráficos, arquitectura, sonido, diseño de personajes, etc.).
Su objetivo es modificar el carácter original del juego, con finalidad artística. No se trata de enganchar al mayor número de usuarios, ni de conseguir una mayor identificación con el personaje, ni de ser el más competitivo del mercado, ni de ganar. Se trata de subvertir y parodiar éticas y estéticas preconcebidas; se trata de socializar induciendo a la reflexión.

4.7.4.5 ART VIRUS

Es una subtendencia del net art, en la que el virus tiene la finalidad de provocar un objeto artístico. No sólo resulta un género artístico, sino un experimento sobre el poder latente del universo virtual, al problematizar los temas de la legalidad y de la propiedad intelectual.

4.7.5 TENDENCIAS ECONÓMICAS DE LA WEB

Es un hecho evidente cómo a nivel empresarial Internet se ha convertido no sólo en una herramienta sino en un elemento indispensable. Ya sea para ofrecer servicios, vender productos o simplemente como factor de confianza, ya que cada vez es más difícil que tenga credibilidad una empresa sin representación en la red.

La red va ganando terreno en el campo de las ventas a la compra “in-situ” y maximiza algunas alternativas a niveles no sospechados como son las subastas, inventadas posiblemente desde el principio del propio comercio y elevadas de una manera asombrosa gracias a las posibilidades de mostrar una cantidad de productos a un número de usuarios a un mismo que tiempo que de otra manera es inimaginable.

Para ciertos productos, es cierto que hay ciertos subconjuntos que no y que posiblemente jamás pertenecerán a los siguientes, que se ha llegado a automatizar su compra, como demuestran los avances en domótica con ciertos frigoríficos que son capaces de encargar la compra cuando detectan que los productos están en bajas cantidades.

A nivel de servicios puede suplir los relacionados con la información, y de hecho, cada vez más, los suple. Cualquier tipo de información es referenciada con mayor asiduidad en Internet, es más cómodo para el proveedor de información, que sólo tiene que colgarla una vez y no necesita conocerla de memoria ni buscarla en un momento dado, y para el receptor que puede proveerse de ella en cualquier momento, las veces que quiera y con la velocidad deseada.

Y luego nos encontramos con lo más relacionado con nuestro tema, el Software. El Software está perdiendo la poca entidad física que tenía y cada vez se apunta más a dejar de comercializarlo por medio de CDs y a empezar a ser la red la proveedora, ya no sólo para descargarlo desde ella, que podríamos decir que es el paso intermedio, sino para ejecutarlo directamente en ella, que al principio era soportado en mayor o menor medida por aplicaciones web y que ahora todos los ojos se dirigen a lo conocido por “The Cloud”.

4.7.6 CONCLUSIÓN

Internet, las tecnologías web, se han convertido en algo imprescindible en gran parte de las facetas de la vida de las personas a muchos niveles. Es algo inconcebible el que se retroceda en este camino hacia la comunicación, ya que la información es poder y si hay algo característico del ser humano es la búsqueda de éste. La ciencia nunca ha retrocedido a lo largo de la historia, ha podido aletargarse o ser obstaculizada, como sucedió en la Edad Media pero nunca ha retrocedido. De estos dos axiomas podemos deducir que solo queda un camino para la comunicación, para Internet y las tecnologías web. Ir hacia delante.

Al estar tan sumidas las tecnologías en tres de las corrientes más importantes para el ser humano, como anteriormente comentamos, a saber, la economía, el ámbito social y el arte su impulso es tremendo, y su velocidad, vertiginosa. Psicológicamente apenas somos capaces de asimilar los cambios a los que somos sometidos por ella, las bifurcaciones en diferentes puntas de investigación son centenares, sin saber cual será la próxima punta que revolucionará la sociedad.

En cierta manera da vértigo pensar que pueda haber cambios tan radicales desde dentro del ser humano hasta fuera, sociales. La globalización por ejemplo, tan sólo ha podido ser posible gracias a unas comunicaciones muy avanzadas las cuales se han aplicado a los sistemas de producción y han creado una revolución económica de la cual el mundo entero se ha resentido, ya no sólo cambiando el paradigma económico, sino social, al distribuir el trabajo de una forma diferente, dejando al tercer mundo como mano de obra, al primero y al segundo con una inmensa mayoría dedicada a los servicios y dando toda libertad a las empresas para ser más poderosas que los estados. Evidentemente este cambio trasciende las posibilidades de comunicación cotidianas de una persona media e influyen decisivamente en la forma de vivir global. A nivel político, los estados pierden poder y surgen organizaciones como el FMI, el BM y el OCM que son capaces de gestionar a nivel global. La política pierde poder en pos de la economía, la economía dispara su actividad al agilizarse sus operaciones por medio de las comunicaciones cada vez más veloces y eficaces, surgen nuevos productos financieros tan sólo posibles por las altas velocidades en las comunicaciones, como los derivados de 24h. Todo esto desemboca en una crisis mundial.

Cuesta asimilar que todo esto derive de Internet, de las comunicaciones tan ágiles como no se podía imaginar antaño, cuesta creer que han sido ingenieros quienes han ofrecido este poder, quienes lo han desarrollado y quienes siguen impulsándolo, a modo de cómo le pasó a Einstein con su relatividad que desembocó en la bomba atómica.

No obstante, no todo es negativo, de hecho, la ciencia siempre ha sido y seguirá siendo totalmente amoral, cómo toda herramienta. También se han organizado revoluciones contra dictadores como en los países árabes, es más fácil denunciar injusticias y que estas denuncias lleguen a buen puerto, se está creando una concienciación de grupo a nivel mundial y se empiezan a hacer tentativas como el movimiento altermundista, el Foro Social Mundial, o incluso el 15-M.

5 PROYECTO

El proyecto se ha decidido abordar en una serie de fases con el fin de recopilar los resultados de una manera ordenada. Éstas han sido las fases seguidas:

- Primero ha habido una toma de requisitos y un análisis de las posibles funcionalidades principales de la futura web.
- Seguidamente se han analizado y definido los criterios que se utilizarán para medir los lenguajes y así, poderlos someter a análisis.
- Lo siguiente ha sido realizar el estudio de los lenguajes y a su fin, llevar a cabo la preselección.
- Una vez determinados los lenguajes a estudiar más profundamente se han desarrollado sus funcionalidades.
- Al final de cada desarrollo de cada lenguaje se ha sacado una reflexión del lenguaje.
- Con toda la información obtenida en el desarrollo de los lenguajes se han contrastado los criterios y se han evaluado contrastándolos entre sí.
- Por último se ha reflexionado una conclusión final.

A continuacion se muestran los entregables resultantes de las etapas anteriores.

5.1 ENTREGABLES

5.1.1 CÓDIGO

Mirar en: ANEXO I:CODIGO.

5.1.2 EVALUACIÓN DE PARÁMETROS POR LENGUAJE

5.1.2.1 HTML/CSS

Realmente la conjunción de HTML y CSS forma la estructura y la maqueta de cualquier página web, ésto es, como van a ser repartidos los contenidos y cuál va a ser su aspecto. Según ha ido evolucionando la web se ha ido descartando darle más funcionalidades a los dos lenguajes y en cambio si se ha querido que se especifiquen y concreten sus funciones, como en el caso de HTML que antes podía tomar funciones de maquetación pero que se han ido reservado como competencia exclusiva de CSS (de momento es admitido aún estas opciones pero en un futuro está pensado que queden como prohibidas). Esto repercute en que se han dejado aparte ciertas funcionalidades para los lenguajes dinámicos:

5.1.2.1.1 GENERACIÓN DINÁMICA.

No disponible.

5.1.2.1.2 CONEXIÓN BD.

No disponible.

5.1.2.1.3 REGISTRO CLIENTES.

No disponible.

5.1.2.1.4 ENVIO MAILS.

No disponible.

5.1.2.1.5 NAVEGABILIDAD.

No disponible.

5.1.2.1.6 DISEÑO-ESTILO

Esta parte realmente compete casi exclusivamente al código CSS. De hecho, con los principios de CSS se crean también interfaces de otro tipo de aplicaciones diferentes de las web. Se ha convertido en uno de los estándares más potentes y conocidos y hoy por hoy se puede encontrar una cantidad impresionante de material reutilizable en la red.

5.1.2.1.6.1 CURVA APRENDIZAJE

Los principios de CSS son sencillos y se basan en estructuras jerárquicas muy simples e intuitivas, incluso para pequeños proyectos se puede usar sin ningún tipo de jerarquía. Ésto consigue que se pueda empezar a trabajar con CSS con muy pocos conocimientos de una manera eficaz. No obstante puede llegar a complicarse extremadamente ya que es una tecnología muy potente y se pueden crear diseños realmente complejos. Por lo que cada vez se puede encontrar más código reutilizable por la red.

5.1.2.1.6.2 POTENCIALIDAD

Realmente se pueden considerar como dos de los lenguajes más potentes, por un lado HTML se basa en XML, que es un metalenguaje de etiquetas que cada vez cobra más importancia a la hora de las transmisiones de datos y es una filosofía cada vez más usada, por su facilidad y claridad.

Por otro lado, CSS ha logrado aislar la capa de diseño completamente y ofrece posibilidades casi infinitas a la hora de cualquier tipo de maquetación. Al apoyarse en jerarquías de clases se evita la redundancia y se compacta el código. Esto es gracias a la propiedad de herencia.

5.1.2.1.6.3 EVOLUCIÓN

La evolución de estos dos lenguajes ha tendido a su especialización y concreción del ámbito. En vez de haber conseguido cada vez más funcionalidades y haberse expandido lo que han hecho ha sido consolidarse y fortalecerse haciéndose completamente imprescindibles a la hora de tratar la web.

5.1.2.1.6.4 FLEXIBILIDAD

La flexibilidad queda patente en la diversidad de lenguajes dinámicos que son embebidos en HTML, al ser la estructura de la página web no hay lenguaje que o bien esté embebido en HTML o bien a la hora de generar resultados, genere código HTML. Se pueden combinar técnicas y lenguajes al igual que se pueden combinar funciones que vayan generando código CSS diferente según pase el tiempo o según se sucedan eventos.

Su fortaleza reside en la rigidez al haberse consagrado cada uno en su ámbito y en la moldeabilidad al poder ser manipulados por casi cualquier lenguaje desde dentro (embebido) o desde fuera (que el lenguaje genere el código).

5.1.2.1.7 FORMULARIO

La estructura de cualquier formulario es llevada a cabo por HTML, por medio de unas etiquetas y un método que normalmente suele ser GET o POST, no obstante ahí acaba la funcionalidad que HTML aporta a los formularios, es su base y es imprescindible pero no aporta ningún tipo de control a la hora de usar los campos, para eso se le añaden las funcionalidades de los lenguajes dinámicos.

5.1.2.1.7.1 CURVA APRENDIZAJE

Ofrece un sistema muy sencillo de plantear los formularios por lo que es muy sencillo de aprender.

5.1.2.1.7.2 POTENCIALIDAD

Al ser la parte estructural se puede considerar que es muy potente, ya que todas las demás aportaciones son “extras”.

5.1.2.1.7.3 EVOLUCIÓN

No ha tenido evolución propia alguna, siempre ha estado así y los que realmente si que han ido evolucionando son los lenguajes dinámicos que se le añaden para aumentar sus funcionalidades.

5.1.2.1.7.4 FLEXIBILIDAD

Es flexible tanto en cuanto se le puede ir añadiendo todo tipo de funcionalidades que ofrezcan los lenguajes dinámicos.

5.1.2.2 PHP

Este si es el primer lenguaje dinámico del cual vamos a poder analizar bien las funcionalidades propuestas. Es uno de los lenguajes más usados en la actualidad y más potentes. Como hemos mencionado anteriormente, éste y el resto de los lenguajes dinámicos se utilizan sobre la base de HTML y la maquetación de CSS.

5.1.2.2.1 CONEXIÓN BD

La conexión es sencilla de realizar con una línea. Es un estándar en el cual se debe de especificar el servidor, el nombre de la base de datos, el usuario y la contraseña, y con eso se puede ejecutar la conexión.

5.1.2.2.1.1 CURVA APRENDIZAJE

Es muy sencillo de aprender y de hecho, los diferentes lenguajes tienen una manera muy similar de conectarse, con casi la misma función y los mismos parámetros.

5.1.2.2.1.2 POTENCIALIDAD

Es potente y en la mayoría de los casos se puede controlar si existe algún tipo de error a la hora de crear la conexión.

5.1.2.2.1.3 EVOLUCIÓN

No ha tenido apenas evolución a lo largo del tiempo ya que no le ha hecho falta, es sencillo y eficaz.

5.1.2.2.1.4 FLEXIBILIDAD

Es flexible tanto en cuánto es capaz de devolver diferentes errores y por lo tanto dar la posibilidad al programador de enfrentarse y resolver un amplio abanico de errores o contratiempos.

5.1.2.2.2 REGISTRO CLIENTES

Esta sección involucra no sólo la conexión con la BD sino también la búsqueda en ésta, la comprobación de la no existencia del usuario, la recogida de datos por medio del formulario y la recogida de los datos y guardado por parte de la BD.

5.1.2.2.2.1 CURVA APRENDIZAJE

Tiene una curva de aprendizaje un poco más alta que las funciones anteriores pero hay que concretar que viene dada por la lógica de la programación que es un poco más compleja que las anteriores funcionalidades, ya que hay que hay que implementar los 3 puntos anteriormente nombrados.

5.1.2.2.2.2 POTENCIALIDAD

Es potente en la medida que los son sus partes, ya que en este caso no se puede decir que el lenguaje dinámico sea el verdadero protagonista de la funcionalidad, pero es cierto que es capaz de conjugar la tecnología de la BD (MySQL) con la recogida de los datos del formulario (HTML) muy eficazmente.

5.1.2.2.2.3 EVOLUCIÓN

Más que evolución se puede hablar de adaptación a las nuevas tecnologías de las BBDD ya que la recogida de datos del formulario no ha cambiado desde el principio pero por la simple razón de que no lo ha necesitado.

5.1.2.2.2.4 FLEXIBILIDAD

Es flexible en la medida de que cómo capa lógica entre los dos lenguajes es capaz de generar un campo casi infinito de posibilidades entre los datos del formulario y los que al final quedarán grabados en la BD.

5.1.2.2.3 GENERACIÓN DINÁMICA

Esta funcionalidad consiste en obtener información de la BD y a partir de ella generar la página. En este caso la tabla contiene tuplas con un nombre y una descripción que a su vez podrá ser añadida por cualquier usuario registrado y logueado.

5.1.2.2.3.1 CURVA APRENDIZAJE

Esta funcionalidad es fácil de aprender una vez que se sabe conectar a la BD y obtener información. Con un simple “while” hasta que no haya más información se pueden ir escribiendo las filas que se van obteniendo en HTML.

5.1.2.2.3.2 POTENCIALIDAD

Es muy potente ya que una vez obtenida la información se puede presentar de tantas maneras como se le ocurran al programador tratando la información antes de presentarla si se terciase.

5.1.2.2.3.3 EVOLUCIÓN

Debido a que se puede desglosar esta acción en dos, conectar a la BD que ya hemos comentado y luego el uso del lenguaje de programación en sí y la capacidad de embeber en él (o viceversa) código HTML, no existe una evolución que pueda ser tenida en cuenta seriamente, casi todas las posibilidades (casi infinitas) ya estaban ahí desde un principio.

5.1.2.2.3.4 FLEXIBILIDAD

La flexibilidad es enorme dado que se puede tratar la información de muchísimas maneras diferentes: obtenerla de diferentes BBDD, mezclarla, tratarla, usarla de maneras diferentes dependiendo de la capa lógica o de la información en sí, etc…

5.1.2.2.4 NAVEGABILIDAD

Esta funcionalidad se concreta en la posibilidad de entrar en una zona privada solo visible para usuarios logueados. El mecanismo a seguir es que cuando uno se loguea se crea una variable global llamada "$_SESSION" que es una matriz en la que se puede almacenar la información que se desee, para ésto en la primera posición he creado un campo llamado "access" que toma valores booleanos y cuando se realiza un logueo se pone a "true" y es precisamente lo que se comprueba a la hora de mostrar la información confidencial.

5.1.2.2.4.1 CURVA APRENDIZAJE

Esta funcionalidad es extremadamente sencilla una vez que se ha concretado el método de conocer si la sesión está iniciada. Con un simple "if" se puede controlar todo.

5.1.2.2.4.2 POTENCIALIDAD

Como en la funcionalidad anterior, es muy potente ya que deja a la imaginación del programador todas las posibilidades de implementación.

5.1.2.2.4.3 EVOLUCIÓN

Me remito a la funcionalidad anterior para remarcar que en éste caso desde un principio el abanico que queda abierto es algo menos amplio aún así increíblemente extenso como para no haber necesitado evolución.

5.1.2.2.4.4 FLEXIBILIDAD

No es excesivamente flexible ya que esta contextualizada en el marco de dejar acceder o no, no obstante si se quisiera abrir el marco a más acciones, con la posibilidad de guardar la información que se desee en el array "$_SESSION" se podrían hacer infinidad de cosas.

5.1.2.2.5 DISEÑO-ESTILO

Dejando claro que el peso central del diseño está a cargo de CSS es cierto que los lenguajes dinámicos dejan abierto un camino muy interesante a nivel de capa lógica. Sabiendo que el diseño de los objetos que existen en la programación está definido por CSS puede parecer que esta definición es estática, y es cierto sin los lenguajes dinámicos, pero con ellos se es capaz de elegir diferentes definiciones para un mismo objeto dependiendo de las condiciones que se quieran poner, incluso temporales para que vayan cambiando, o dependiendo del perfil elegido por el usuario.

5.1.2.2.5.1 CURVA APRENDIZAJE

La curva de aprendizaje es un poco más alta que las anteriores, ya que el concepto aunque sea sencillo a la hora de la implementación uno se enfrenta a diversos problemas de carga de la página, definiciones CSS incompatibles con los elementos de alrededor, incompatibilidades con el navegador, etc. No obstante con un cierto orden es una funcionalidad muy vistosa y asequible.

5.1.2.2.5.2 POTENCIALIDAD

Es muy potente ya que combina el inmenso potencial de CSS con las posibilidades exponenciales de combinarlo por medio del flujo de datos de un lenguaje dinámico.

5.1.2.2.5.3 EVOLUCIÓN

En un principio no se tuvo en cuenta esta posibilidad y ha ido evolucionando desde la nada hasta poder tener un control absoluto de los estilos. A lo largo del tiempo han ido surgiendo ideas entre los programadores de cómo poder usarlo y cada vez el marco de actuación es más amplio y potente.

5.1.2.2.5.4 FLEXIBILIDAD

Es enormemente flexible, ya que precisamente se ésta su cualidad más importante y útil. Las restricciones reales son las impuestas por los navegadores y los objetos colindantes al modificado. Y las posibilidades casi infinitas.

5.1.2.2.6 FORMULARIO

Esta funcionalidad está más enfocada a las acciones que se pueden hacer respecto al formulario básico HTML, como controlar los campos, recibir y procesar la información…

5.1.2.2.6.1 CURVA APRENDIZAJE

Realmente la manera de tratar los formularios está muy explotada y se conocen muy bien los métodos, es sencilla la idea y las funciones a las que se les llama quizás tengan algo más de dificultad para un programador novato ya que dependen del tratamiento de cadenas de caracteres y además hay mucha documentación y código por internet con lo que aún siendo un poco más complejo que otras funcionalidades, no debería entrañar demasiada dificultad de aprendizaje.

5.1.2.2.6.2 POTENCIALIDAD

Es potente en la medida en que cumple con eficacia su tarea y no deja lugar a fallos.

5.1.2.2.6.3 EVOLUCIÓN

No ha sufrido apenas evolución ya que desde un primer momento se tenía clara su función.

5.1.2.2.6.4 FLEXIBILIDAD

Por lo mismo que comento en el apartado anterior no se puede decir que globalmente tenga una gran flexibilidad, pero sí localmente, me explico, las condiciones para tratar los formularios aún dentro de un marco específico, son totalmente libres para poder ser creadas por el programador.

5.1.2.2.7 ENVÍO MAIL

Esta funcionalidad es muy sencilla y está incluida en todos los lenguajes dinámicos, y hace que el servidor actúe como servidor SMTP.

5.1.2.2.7.1 CURVA APRENDIZAJE

Casi inexistente, es una función sencillísima de usar.

5.1.2.2.7.2 POTENCIALIDAD

Es potente ya que cumple perfectamente con su función perfectamente y además tiene la posibilidad de añadir todos los parámetros de control posibles en un mail.

5.1.2.2.7.3 EVOLUCIÓN

La evolución ha sido nula ya que desde el principio se contaba con todas las posibilidades.

5.1.2.2.7.4 FLEXIBILIDAD

La que aporta las posibilidades de un mail.

5.1.2.3 PSP

Este es el segundo lenguaje dinámico que se va a estudiar. Es un lenguaje relativamente nuevo en su aplicación a la web y por esto mismo no existe demasiada documentación ni librerías excesivamente extensas, como pasaba en PHP. Existen dos maneras de usarlo en el ámbito web (aparte de la mezcla de ambas maneras), por un lado se puede embeber en HTML, como PHP y por otra se puede crear una aplicación que genere el código HTML. Se ha usado la primera manera con el objetivo de poder comparar los lenguajes y no las diferentes metodologías con las que afrontar el problema.

5.1.2.3.1 CONEXIÓN BD

Como en PHP, la conexión es sencilla de realizar con una línea. Es un estándar en el cual se debe de especificar el servidor, el nombre de la base de datos, el usuario y la contraseña, y con eso se puede ejecutar la conexión.

5.1.2.3.1.1 CURVA APRENDIZAJE

Es un poco más complejo de aprender que PHP, en éste caso se deben de incluir las librerías que nos aportarán las funciones de conexión. Con una subclase denominada “cursor” se tendrá acceso a la información de la BD. Sobre ese objeto se irán aplicando funciones para extraer lo relevante.

5.1.2.3.1.2 POTENCIALIDAD

Es muy potente ya que aunque sea algo más difícil de aprender, una vez que se tienen los conocimientos adquiridos se tiene acceso a muchas más opciones gracias a lo extensas que son las librerías ya existentes y necesarias para su uso.

5.1.2.3.1.3 EVOLUCIÓN

Al ser un módulo aparte al que se le llama ha tenido la posibilidad de ser más completo que el caso de PHP, lo que se ha ido consiguiendo por medio de una evolución mayor.

5.1.2.3.1.4 FLEXIBILIDAD

Gracias a su modularidad también se ha conseguido una mayor flexibilidad a la hora de importar diferentes módulos y combinarlos con todas las funciones disponibles que ello acarrea.

5.1.2.3.2 REGISTRO CLIENTES

Esta sección está estrechamente ligada a la anterior en el caso de PSP ya que al ser modular no se puede conectar sin la BD sin aprender a usar el módulo, lo que implica que casi todo el trabajo de aprendizaje, el conocimiento de su potencialidad y flexibilidad y la evolución que ha podido tener el módulo son adquiridos de la misma manera. Por lo que no se comentará y se le asignarán los mismos valores anteriores.

5.1.2.3.3 GENERACIÓN DINÁMICA

Esta funcionalidad consiste en obtener información de la BD y a partir de ella generar la página. En el caso de PSP, una vez aprendido como se usa el módulo de la BD es increíblemente sencillo exponer los datos, y como la dificultad de saber usar el módulo ya ha sido tenida en cuenta, nos centraremos es la exposición de éstos datos.

5.1.2.3.3.1 CURVA APRENDIZAJE

Cómo el caso anterior, se usará un simple “while” hasta que no quede más información que mostrar. Se usarán dos variables para ir almacenando el nombre y la descripción de cada dios e irlas mostrando a cada vuelta del “while”.

5.1.2.3.3.2 POTENCIALIDAD

Se podría decir que es igual de potente que PHP pero tiene un punto a su favor que no tiene PHP y es que cómo el módulo da muchas opciones diferentes de recoger la información facilita también la manera de mostrarla.

5.1.2.3.3.3 EVOLUCIÓN

Al igual que en el caso PHP tampoco existe una evolución que pueda ser tenida en cuenta cómo tal para esta sección en concreto.

5.1.2.3.3.4 FLEXIBILIDAD

Adquiere muchísima flexibilidad gracias a las funciones ofrecidas en los módulos, la combinación de éstos y la moldeabilidad del lenguaje en sí.

5.1.2.3.4 NAVEGABILIDAD

Esta funcionalidad es muy fácil de usar en PSP, se comprueba si el usuario esta logueado por medio de una variable “session” muy parecida a la de PHP, que también es un array en el que la primera posición he llamado “abierta”, si es así, se muestra la información privada.

5.1.2.3.4.1 CURVA APRENDIZAJE

Es muy fácil de aprender, como en PHP, con un simple “if” se controla.

5.1.2.3.4.2 POTENCIALIDAD

Es eficaz en sí misma y tan potente como el programador sea capaz de imaginar.

5.1.2.3.4.3 EVOLUCIÓN

Debido a que lo que ésta en tela de juicio no es la conexión con la BD sino la propia lógica de programación o en todo caso esa variable “session”, no se puede juzgar una verdadera evolución.

5.1.2.3.4.4 FLEXIBILIDAD

Deja espacio a la flexibilidad en el momento que “session” es un array rellenable por el programador, pudiendo así hacer tan compleja como desee con los parámetros que le sean necesarios.

5.1.2.3.5 DISEÑO-ESTILO

Al igual que en el caso de PHP, con PSP se puede jugar con el código CSS, haciéndolo dinámico según la imaginación del programador. Tan sólo hay que linkar como hoja estilo un archivo “.psp” el cual contendrá las funciones pertinentes para que lo que devuelva sea código CSS.

5.1.2.3.5.1 CURVA APRENDIZAJE

La curva de aprendizaje es más alta que la de PHP sobretodo porque no hay a día de hoy ejemplos en internet asique hay que guiarse por lo que se sabe del lenguaje e ir probando. Es un poco más costoso que funcione con PSP pero se ahorran líneas de código.

5.1.2.3.5.2 POTENCIALIDAD

Una vez que se sabe el mecanismo de funcionamiento (en especial lo que hay devolver) es una funcionalidad muy potente con la que se pueden hacer infinidad de variaciones.

5.1.2.3.5.3 EVOLUCIÓN

Viendo la documentación que hay en internet al respecto casi se podría decir que está en plena eclosión y descubrimiento de las nuevas funcionalidades que puede ofrecer PSP frente a PHP, supongo, basadas en la modularidad tan estricta y completa. No me extrañaría que de aquí a uno o dos años saliera un módulo completo para controlar CSS.

5.1.2.3.5.4 FLEXIBILIDAD

Al tratarlo desde la capa lógica, el resultado es enormemente flexible, combina todas las propiedades visuales de CSS con las combinaciones de la capa lógica.

5.1.2.3.6 FORMULARIO

Como en el caso de PHP, y de hecho casi de la misma manera PSP trata por medio de funciones el control de los campos del formulario y el contenido de la información.

5.1.2.3.6.1 CURVA APRENDIZAJE

He aquí la verdadera diferencia entre ambos lenguajes, ya que aunque los dos sean parecidos en el nivel de aprendizaje, en este caso, lo que destaca de PHP son las librerías ya escritas y la de documentación que se puede encontrar por internet.

5.1.2.3.6.2 POTENCIALIDAD

La potencialidad es inherente a la capacidad del programador y su dedicación a cada campo del formulario o a lo que se quiera hacer con la información recogida.

5.1.2.3.6.3 EVOLUCIÓN

No se puede decir que tenga una evolución concreta como tal, simplemente con la lógica del lenguaje se puede trabajar dependiendo de cuales sean los objetivos marcados por el programador.

5.1.2.3.6.4 FLEXIBILIDAD

La flexibilidad es enorme ya que es el programador quién realmente pone la forntera dónde sus objetivos se establezcan o hasta dónde su imaginación llegue. Igualmente a la hora de tratar los datos.

5.1.2.3.7 ENVÍO MAIL

Como en el caso de PHP, esta funcionalidad es muy sencilla y se hace con una simple función que convierte el servidor en un servidor SMTP.

5.1.2.3.7.1 CURVA APRENDIZAJE

Casi inexistente, es una función muy fácil de usar.

5.1.2.3.7.2 POTENCIALIDAD

Es potente ya que cumple perfectamente con su función perfectamente y además tiene la posibilidad de añadir todos los parámetros de control posibles en un mail.

5.1.2.3.7.3 EVOLUCIÓN

No ha habido evolución.

5.1.2.3.7.4 FLEXIBILIDAD

La que aporta las posibilidades de un mail.

5.2 CUADRO RESUMEN

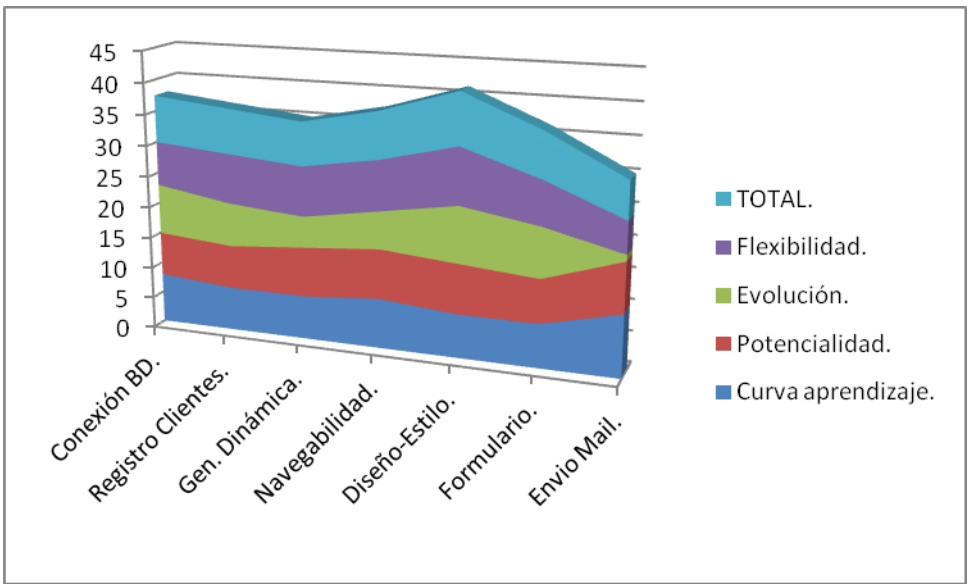
A continuación voy a mostrar los cuadros que resumen y puntuan los estudios anteriores referentes a los lenguajes:

HTML/CSS	Curva aprendizaje	Potencialidad	Evolución	Flexibilidad	TOTAL.
Conexión BD	--	--	--	--	--
Registro Clientes	--	--	--	--	--
Gen. Dinámica	--	--	--	--	--
Navegabilidad	--	--	--	--	--
Diseño-Estilo	6	10	10	10	9,5
Formulario	9	2	10	9	8,25
Envio Mail	4	6	3	3	3,625

Aún llevado acabo el análisis voy a obviar el resultado final ya que no tiene comparativa porque no existe opción alternativa en desarrollo web.

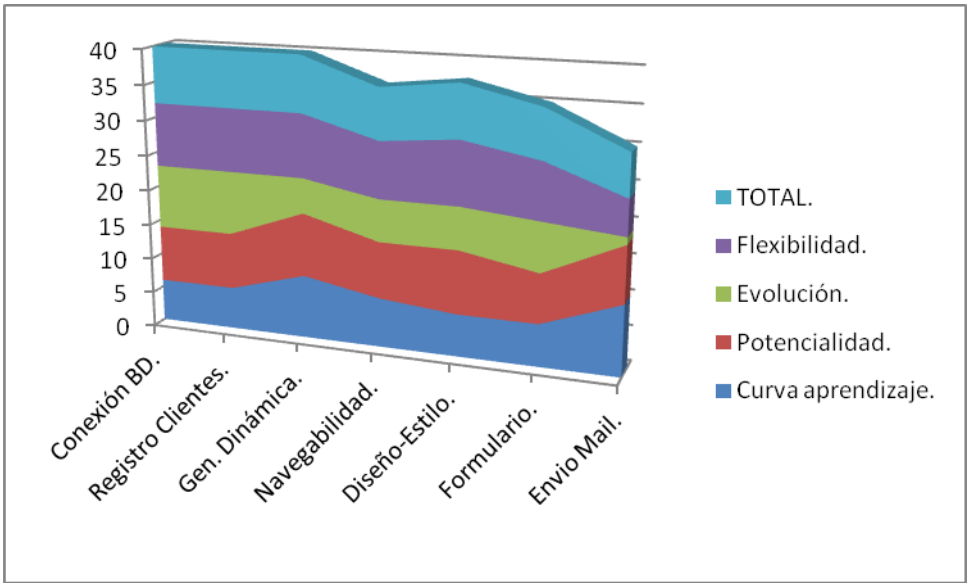
PHP	Curva aprendizaje.	Potencialidad.	Evolución.	Flexibilidad.	TOTAL.
Conexión BD.	8	7	8	7	7,5
Registro Clientes.	7	7	7	8	7,25
Gen. Dinámica.	7	8	5	8	7

Navegabilidad.	8	8	6	8	7,5
Diseño-Estilo.	7	8	9	9	8,25
Formulario.	7	7	8	7	7,25
Envio Mail.	10	8	1	5	6



RESULTADO FINAL: 7,25.

PSP	Curva aprendizaje.	Potencialidad.	Evolución.	Flexibilidad.	TOTAL.
Conexión BD.	6	8	9	9	8
Registro Clientes.	6	8	9	9	8
Gen. Dinámica.	9	9	5	9	8
Navegabilidad.	7	8	6	8	7,25
Diseño-Estilo.	6	9	6	9	7,5
Formulario.	6	7	7	8	7
Envio Mail.	10	8	1	5	6



RESULTADO FINAL: 7,3928.

5.3 SELECCIÓN FINAL

Después de todo el análisis se puede apreciar que el lenguaje mejor puntuado es PSP con 7,39 frente a los 7,25 de PHP. La diferencia entre ambos lenguajes es mínima así que antes de tomar la decisión será muy recomendable buscar nuevos criterios ajenos para asegurarse de que la decisión será la correcta.

Dadas las circunstancias de la petición se deberá valorar el punto de vista de la empresa a la hora de la toma de decisión. Serán puntos cruciales saber con que tipos de desarrolladores cuenta la empresa para usarlos en el proyecto de la página web. Cuál será la posible ampliación de funcionalidades de ésta. En el caso de no contar con los desarrolladores y de que las posibles ampliaciones funcionales sean mínimas se analizará el sueldo de los desarrolladores de ambos lenguajes para decidir cuál será más económico para la empresa.

Estos puntos con casi toda probabilidad favorecen a PHP por ser un lenguaje más asentado en el diseño web y teniendo en cuenta que el caso concreto a analizar no se trata de una aplicación web con una capa lógica excesivamente amplia o cambiante parece que la resolución de nuestra elección se inclinará hacia PHP.

No obstante, me parece digno de mención que en el caso de que no se tratara de una web "simple", sino de una aplicación web, una web con una amplia o cambiante capa lógica, una web experimental en la cual se aproveche el modo interactivo (uso de consola y generación inmediata de resultados) de Python, una web con tecnología de alta innovación o en definitiva cualquier tipo de web, de aplicación web o de aplicación que no tenga un contexto hermético o que tenga la posibilidad de cambios o ampliaciones poco conocidas a priori o relacionadas con una alta innovación que requiera una gran flexibilidad la elección sin duda será Python, y como tecnología para tratar HTML, PSP.

6 CONCLUSIONES

A la vista de este proyecto se ha podido conocer la actualidad y pasado de la web, saber en que contexto se mueve y cuales son sus herramientas principales, básicas, emergentes y consolidadas.

Se ha podido aprender desde la parte humana hasta la tecnológica, pasando por el entorno que lo soporta, la creación de un servidor, sus componentes y su configuración.

Se han analizado los actuales lenguajes de programación y conocido sus principales características además de aprendido a programar en los dos candidatos más prometedores.

No sólo se ha aprendido a programar en ellos sino que se ha aprendido a compararlos, a poner en evidencia sus ventajas y defectos, su historia, su comportamiento ante diferentes funcionalidades y su curva de aprendizaje.

El resultado final ha sido poco concluyente ya que dependía en gran medida de la empresa pero si ha sido muy concluyente el patrón para elegir que lenguaje es el más adecuado en cada caso, que dado que la empresa y la posible web son ficticias es mucho más importante que una conclusión final rigurosa y concreta.

7 PRÓXIMOS PASOS

Desde mi punto de vista y a la vista de los resultados dependientes en gran medida de la asiduidad de uso de los lenguajes de programación para su elección recomendaría el estudio de las vertientes de Java para la web.

Java es el lenguaje más utilizado a nivel empresarial y ésto le confiere muchas ventajas, como es la seriedad en la creación de patrones de desarrollo y la gran cantidad de frameworks existentes concienzudamente probados.

A la tecnología web basada en Java, cómo anteriormente se ha comentado brevemente se le llama JSP.

A continuación paso a describir un poco sus características más básicas como modo de introducción para poder empezar su estudio:

7.1 JSP

Un JSP ("Java Server Page") es uno de los componentes más básicos empleados para aplicaciones de Servidor en Java. Su composición consta de dos grandes partes: HTML y lenguaje Java.

Mediante HTML se especifica el contenido estático de despliegue y es mediante fragmentos del lenguaje Java que se genera contenido dinámico en efecto cumpliendo la definición de aplicación de Servidor. Los elementos que se encuentran entre <% y %> son elementos Java , mientras el resto es considerado HTML puro. La ejecución y/o transformación de elementos Java es trabajo de un "Servlet Engine" como podria ser Tomcat.

Secuencia de Ejecución ("Lifecycle")

La secuencia de ejecución para un JSP es la siguiente (partiendo de la requisición de página):

- Compilar JSP a Servlet.
- Cargar/Compilar el JSP(Servlet).
- Inicializar el JSP(Servlet).
- Ejecutar el JSP(Servlet).
- Limpiar el JSP(Servlet).

El primer paso puede ser sorprendente, pero efectivamente todo JSP es convertido a un Servlet , la razón de esto es que el surgimiento de JSP's se debió a la necesidad de facilitar un formato programático sencillo para personal no familiarizado con Java, esto es, el uso de JSP's

permite eliminar visualmente funciones de arranque, ejecución y terminación , la inclusión de éstas se lleva acabo en la conversión de JSP a Servlet.

7.2 SERVLET

A diferencia de un JSP un Servlet es un componente escrito puramente en Java, en términos Java esto equivale a una Clase y como cualquier otra Clase Java ésta se encuentra compuesta por diversos métodos/funciones.Los tres métodos que debe implementar un Servlet son:

- service (Obligatorio): Este método es la parte medular de todo Servlet ya que dentro de él se incluyen las tareas principales de ejecución.
- init (Opcional) : Es un método ejecutado antes del método service, su labor principal es adquirir/inicializar algún recurso que será empleado por service, estos recursos típicamente son conexiones hacia Bases de Datos.
- destroy (Opcional) : Ejecutado una vez que ha terminado el método service, su labor es liberar los recursos utilizados/adquiridos en el proceso de ejecución los cuales generalmente son aquellos reservados por init.

7.2.1 HTTPSERVLET Y EL MÉTODO SERVICE()

Para generar un Servlet es necesario que la Clase en cuestión herede ("inherit") el comportamiento de la clase llamada HttpServlet, a través de dicha Clase se tiene acceso a los métodos antes mencionados. También vale mencionar que es posible heredar ("inherit") de la clase GenericServlet , sin embargo, para efectos de este curso y generalmente todo proyecto en ambientes Web, se opta por emplear HttpServlet ya que es una clase más completa e idónea para ambientes HTTP (el protocolo utilizado en Internet), inclusive si observa la documentación de dichas Clases notará que HttpServlet desciende (hereda) precisamente de GenericServlet.

7.2.2 WAR'S ("WEB ARCHIVES")

Para facilitar la distribución e interacción entre JSP's/Servlets estos son agrupados en una estructura denominada WAR ("Web-Archive") descrita a continuación :

- / *.html *.jsp *.css : Este directorio base contiene los elementos típicamente empleados para un sitio web: Documentos HTML, CSS ("Cascading Style Sheets"), JavaScript y Graficas; además en este directorio residen los JSP's a utilizarse en el WAR, aquí no residen los Servlets estos deben ser colocados en otra parte del WAR ya que son Clases Java puras.
- /WEB-INF/web.xml : Este archivo contiene elementos de configuración del WAR como : Página de Inicio, Ubicación ("Mapeo") de Servlets, parámetros para componentes adicionales tales como "Struts" y otros elementos como manejo de errores.

- /WEB-INF/classes/ : Este directorio contiene las clases Java utilizadas dentro del WAR, es dentro de este directorio que generalmente residen los Servlets diseñados para el WAR.
- /WEB-INF/lib/ : Este directorio contiene los archivos JAR que serán utilizados por la aplicación, estos generalmente corresponden a las clases (JAR's) utilizadas para conectarse a Bases de Datos o aquellas utilizadas por librerías de JSP's.

Este tipo de estructura logra una interoperabilidad para las diversas aplicaciones Java, ya que los diversos "Servlet Engines"/"Application Servers" emplean esta misma estructura para ejecutar componentes.

7.2.3 WEB.XML

web.xml es un archivo escrito en XML que describe diversas características del archivo WAR y contiene varios elementos:

- El primer elemento <web-app> indica el inicio de la aplicación, es dentro de este elemento que se definen todos los elementos restantes.
- El elemento <servlet> define las características de un Servlet y a su vez esta compuesto por los elementos <servlet-name> y <servlet-class> que indican un nombre corto para el Servlet así como el nombre de la Clase Java que contiene el Servlet respectivamente.
- Posteriormente se define el elemento <servlet-mapping> el cual define la ubicación en términos de directorios de un sitio (URL).
- Finalmente el elemento <welcome-file-list> indica que cuando se solicite cualquier directorio sin indicar un archivo en especifico se envíe el archivo llamado: index.jsp, index.html o index.htm, donde el primero tiene preferencia en caso de existir más de un archivo en la lista.

7.2.4 MVC

La estructura MVC ("Model-View-Controller") es un paradigma utilizado en diversos desarrollos de software, a través de este "Framework" se logra una división de las diferentes partes que conforman una aplicación, siendo su principal razón de ser: manutención del código fuente.

Conforme incrementan las necesidades de cualquier aplicación, la modificación a código existente se hace inminente y si no existe una clara división de uso , el código no solo se torna indescifrable sino en ocasiones impredecible debido a la mezcla de funcionalidades que pueden surgir.

A través de MVC se realiza la siguiente división :

- Model: Concentra las funcionalidades relacionadas con el Modelo de datos, esto es, el acceso y manipulación de depósitos informativos como Bases de Datos y Archivos.
- View: Se basa en el aspecto visual/gráfico que será empleado por la aplicación en cuestión.

- Controller: Empleado como un mediador entre el medio gráfico ("View") y el modelo ("Model"), coordina las acciones que son llevadas acabo entre ambos.

7.2.5 Con JSP's y Servlets

El uso de "MVC" en ambientes Web para JSP's y Servlets ha empezado a generar gran interés, debido a que una vez diseñada una aplicación para ambiente Web es raro que ésta permanezca sin cambios, el uso de MVC permite realizar diseños con JSP's/Servlets que logran verdaderas soluciones a escala.

Finalmente, vale mencionar que hoy en día existen diversas implementaciones para utilizar un "Framework MVC" en ambientes de JSP's/Servlets, entre ellas se encuentran :

- Struts (http://jakarta.apache.org/struts)
- Spring (http://www.springframework.org/)
- Maverick (http://mav.sourceforge.net/)

7.2.6 Java Beans

Un Java Bean es una manera de modularizar el uso datos en una aplicación con JSP's/Servlets a través de una Clase, su característica primordial es el uso de los métodos get y set los cuales permiten el acceso a los valores del Bean, el diseño de un Java Bean es relativamente sencillo ya que no posee código extenso.

A través de Java Beans es posible encapsular conceptos de diferentes tipos, en el proceso permitiendo la manipulación de valores a través de una sola Clase, de esta manera facilitando el manejo de información y a su vez el desarrollo de Software.

7.2.7 Filtros

Otro paradigma empleado en diseños modulares de JSP's y Servlets es el de Filtros, como su nombre lo implica, un filtro permite que determinada requisición o respuesta sea analizada en circunstancias especificas. En este caso la requisición y/o respuesta corresponden a los principales elementos de un Servlet (por ende JSP también) que son los objetos ServletRequest y ServletResponse respectivamente.

Una de las principales características de un filtro es que puede ser colocado en cualquier punto de una secuencia de actividades ("Work-Flow"), ya sea entre una solicitud de JSP a JSP, Servlet a Servlet, o inclusive Servlet/JSP a HTML, la única condición que debe cumplirse es que debe existir un objeto de entrada (ServletRequest) y otro de salida (ServletResponse), condición que obviamente cumple todo JSP y Servlet.

Ahora bien, en lo que concierne el proceso de filtrado este puede ser de cualquier tipo imaginable, aunque los candidatos naturales para filtros son procesos de registro ("Logs"), pruebas de integridad ("Debugging") o simplemente la modificación de una respuesta previo a su envío.

7.2.8 EVENTOS ("LISTENERS")

A través de eventos se ofrece el mecanismo para realizar determinadas acciones al momento de ocurrir un acontecimiento en un JSP o Servlet, en el sentido estricto de la palabra este mecanismo no es conocido como evento en J2EE sino como oyente o "listener", pero en principio es el mismo concepto.

8 ANEXO I: CÓDIGO

8.1 PÁGINA EN HTML Y CSS

8.1.1 AREA.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
<div align="center">

<h3> Area restringida. </h3>


<img src='./files/homero.jpg' border='0' alt='Homero' align='center'>";

<br><br>
Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.
<br><br>
Ius id vidit volumus mandamus, vide veritus democritum te nec, ei eos debet libris consulatu. No mei ferri graeco dicunt, ad cum veri accommodare. Sed at malis omnesque delicata, usu et iusto zzril meliore. Dicunt maiorum eloquentiam cum cu, sit summo dolor essent te. Ne quodsi nusquam legendos has, ea dicit voluptua eloquentiam pro, ad sit quas qualisque. Eos vocibus deserunt quaestio ei.
<br><br>
Blandit incorrupte quaerendum in quo, nibh impedit id vis, vel no nullam semper audiam. Ei populo graeci consulatu mei, has ea stet modus phaedrum. Inani oblique ne has, duo et veritus detraxit. Tota ludus oratio ea mel, offendit persequeris ei vim. Eos dicat oratio partem ut, id cum ignota senserit intellegat. Sit inani ubique graecis ad, quando graecis liberavisse et cum, dicit option eruditi at duo. Homero salutatus suscipiantur eum id, tamquam voluptaria expetendis ad sed, nobis feugiat similique usu ex.
<br><br>
Eum hinc argumentum te, no sit percipit adversarium, ne qui puto feugiat persecuti. Odio omnes scripserit ad est, ut vidit lorem maiestatis his, putent mandamus gloriatur ne pro. Oratio iriure rationibus ne his, ad est corrumpit splendide. Ad duo appareat moderatius, ei falli tollit denique eos. Dicant evertitur mei in, ne his deserunt perpetua sententiae, ea sea omnes similique vituperatoribus. Ex mel errem intellegebat comprehensam, vel ad tantas antiopam delicatissimi, tota ferri affert eu nec. Legere expetenda pertinacia ne pro, et pro impetus persius assueverit.
<br><br>
Ea mei nullam facete, omnis oratio offendit ius cu. Doming takimata repudiandae usu an, mei dicant takimata id, pri eleifend inimicus euripidis at. His vero singulis ea, quem euripidis abhorreant mei ut, et populo iriure vix. Usu ludus affert voluptaria ei, vix ea error definitiones, movet fastidii signiferumque in qui.
<br><br>
Vis prodesset adolescens adipiscing te, usu mazim perfecto recteque at, assum putant erroribus mea in. Vel facete imperdiet id, cum an libris luptatum perfecto, vel fabellas inciderint ut. Veri facete debitis ea vis, ut eos oratio erroribus. Sint facete perfecto no vel, vim id omnium insolens. Vel dolores perfecto pertinacia ut, te mel meis ullum dicam, eos assum facilis corpora in.
<br><br>
Mea te unum viderer dolores, nostrum detracto nec in, vis no partem definiebas constituam. Dicant utinam philosophia has cu, hendrerit prodesset at nam, eos an bonorum dissentiet. Has ad placerat intellegam consectetuer, no adipisci mandamus senserit pro, torquatos similique percipitur est ex. Pro ex putant deleniti repudiare, vel an aperiam sensibus suavitate. Ad vel epicurei convenire, ea soluta aliquid deserunt ius, pri in errem putant feugiat.
<br><br>
Sed iusto nihil populo an, ex pro novum homero cotidieque. Te utamur civibus eleifend qui, nam ei brute doming concludaturque, modo aliquam facilisi nec no. Vidisse maiestatis constituam eu his, esse pertinacia intellegam ius cu. Eos ei odio veniam, eu sumo altera adipisci eam, mea audiam prodesset persequeris ea. Ad vitae dictas vituperata sed, eum posse labore postulant id. Te eligendi principes dignissim sit, te vel dicant officiis repudiandae.
<br><br>
Id vel sensibus honestatis omittantur, vel cu nobis commune patrioque. In accusata definiebas qui, id tale malorum dolorem sed, solum clita phaedrum ne his. Eos mutat ullum forensibus ex, wisi perfecto urbanitas cu eam, no vis dicunt impetus. Assum novum in pri, vix an suavitate moderatius, id has reformidans referrentur. Elit inciderint omittantur duo ut, dicit democritum signiferumque eu est, ad suscipit delectus mandamus duo. An harum equidem maiestatis nec.
At has veri feugait placerat, in semper offendit praesent his. Omnium impetus facilis sed at, ex viris tincidunt ius. Unum eirmod dignissim id quo. Sit te atomorum quaerendum neglegentur, his primis tamquam et. Eu quo quot veri alienum, ea eos nullam luptatum accusamus. Ea mel causae phaedrum reprimique, at vidisse dolores ocurreret nam.
<br><br><br><br>



<br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center"> Autor: Juan Manuel Rincon.</p></FONT><BR>

</div>
</body>
```

```
</html>
```

8.1.2 CABECERA.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

  <img src="http://localhost/HtmlProject/files/LLII.jpg" alt="Cabecera" style="position: absolute; top: 0; left: 0; width: 100%; height: 100%" />

</body>

</html>

CONTACTO

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
<div align="center">

<H1>Contacta con nosotros:</H1><br>

      Escribe un mail a: juanmanuel.rincon.carrera@gmail.com.
<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center"> Autor: Juan Manuel Rincon.</p></FONT><BR>

</div>
</body>

<html>
```

8.1.3 DIOSES.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

<div align="center">
   <h2> Dioses. </h2>
</div>

<div align="center">

  <img src="http://localhost/HtmlProject/files/202.jpg" />

</div>

<br><br>

<br><h3>Hera</h3>
Hera (en griego antiguo Ἥρα Hēra, o equivalentemente Ἥρη Hērē en jónico y griego homérico) legítima esposa de Zeus y una de las tres hermanas de Zeus en el panteón olímpico de la mitología griega clásica. Su principal función era como diosa de las mujeres y el matrimonio. Su equivalente en la mitología romana era Juno. Se le sacrificaban la vaca y más tarde el pavo real. Su madre era Rea y su padre Crono.
<br>_____<br>


<br><h3>Hades</h3>
En la mitología griega Hades (en griego antiguo Ἅδης Hadēs, originalmente Ἀιδης Haidēs o Ἀϊδης Aïdēs —dórico Ἀϊδας Aidas—, 'el invisible') alude tanto al antiguo inframundo griego como al dios de éste. La palabra hacía referencia en Homero solo al dios; siendo el genitivo Ἅιδού Haidou un elisión para designar ubicación: '[la casa/dominio] de Hades'. Finalmente también el nominativo llegó a designar la morada de los muertos.
<br>_____<br>
```

```
<br><h3>Zeus</h3>
En la mitología griega Zeus (en griego antiguo Ζεύς Zeús) es el «padre de los dioses y los hombres», que gobernaba a los dioses del monte Olimpo como un padre a una familia, de forma que incluso los humanos que no eran sus hijos naturales se dirigían a él como tal. Era el Rey de los Dioses que supervisaba el universo. Era el dios del cielo y el trueno. Sus atributos incluyen el rayo, el águila, el toro y el roble. Además de su herencia indoeuropea, el clásico Zeus «recolector de nubes» también obtuvo ciertos rasgos iconográficos de culturas del antiguo Oriente Próximo, como el cetro. Zeus fue frecuentemente representado por los artistas griegos en dos poses: de pie, avanzando con un rayo levantado en su mano derecha, y sentado majestuosamente.
<br>_____<br>


<br><h3>Prometeo</h3>
En la mitología griega, Prometeo (en griego antiguoΠρομηθεύς, ‘previsión’, ‘prospección’) es el Titán amigo de los mortales,  honrado principalmente por robar el fuego de los dioses en el tallo de una cañaheja, darlo a los humanos para su uso y ser castigado por este motivo.
<br>_____<br>


<br><br><br>
Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.


<br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center"> Autor: Juan Manuel Rincon.</p></FONT><BR>


</body>

</html>
```

8.1.4 INDEX.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<html>

   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>HTML</title>
   </head>

   <frameset rows="20%,*">
      <frame name="Cabecera" src="Cabecera.html" marginwidth="10" marginheight="10" scrolling="auto" >

         <frameset cols="15%,*">

            <frame name="Indice" src="Indice.html" marginwidth="10" marginheight="10" scrolling="auto" >
               <frame name="Cuerpo" src="Principal.html" marginwidth="10" marginheight="10" scrolling="auto" >

               </frame>
            </frame>

         </frameset>
         <noframes>
            <body> <p> Esta p&aacute;gina utiliza frames. Es posible que si utiliza un navegador antiguo no se est&eacute; mostrando adecuadamente.</p>
            </body>
         </noframes>
   </frameset>

   <body>



   </body>
</html>

INDICE

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body class="indice">
```

```
   <A HREF="Principal.html" target="Cuerpo">Inicio</A><BR><BR>
   <A HREF="Contacto.html" target="Cuerpo">Contacta</A><BR><BR>
   <A HREF="Quienes.html" target="Cuerpo">Quienes somos</A><BR><BR>
   <A HREF="Dioses.html" target="Cuerpo">Dioses</A><BR><BR>
   <A HREF="Area.html" target="Cuerpo">Area Restringida</A><BR><BR>


</body>
</html>
```

8.1.5 PRINCIPAL.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>
<body>

   <TABLE>
      <div align="center">
         <img src="files/Ciceron.jpg" width="504" height="360" border="0" alt="Ciceron">
      </div>
     <p><br><br><br><br>Est enim, iudices, haec non scripta, sed nata lex, quam non didicimus, accepimus, legimus, verum ex natura ipsa arripuimus, hausimus, expressimus, ad quam non docti, sed facti, non instituti, sed imbuti sumus.<br><br>Marco Tulio CICER&Oacute;N<br><br><br><br>Existe, de hecho, jueces, una ley no escrita, sino innata, la cual no hemos aprendido, heredado, le&iacute;do, sino que de la misma naturaleza la hemos agarrado, exprimido, apurado, ley para la que no hemos sido educados, sino hechos; y en la que no hemos sido instruidos, sino empapados. </p>

   </TABLE>
   <br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center"> Autor: Juan Manuel Rincon.</p></FONT><BR>


</body>

</html>
```

8.1.6 QUIENES.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
<div align="center">
<h3> Quienes somos. </h3>
<br><br>

Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.
<br><br>
Ius id vidit volumus mandamus, vide veritus democritum te nec, ei eos debet libris consulatu. No mei ferri graeco dicunt, ad cum veri accommodare. Sed at malis omnesque delicata, usu et iusto zzril meliore. Dicunt maiorum eloquentiam cum cu, sit summo dolor essent te. Ne quodsi nusquam legendos has, ea dicit voluptua eloquentiam pro, ad sit quas qualisque. Eos vocibus deserunt quaestio ei.
<br><br>
Blandit incorrupte quaerendum in quo, nibh impedit id vis, vel no nullam semper audiam. Ei populo graeci consulatu mei, has ea stet modus phaedrum. Inani oblique ne has, duo et veritus detraxit. Tota ludus oratio ea mel, offendit persequeris ei vim. Eos dicat oratio partem ut, id cum ignota senserit intellegat. Sit inani ubique graecis ad, quando graecis liberavisse et cum, dicit option eruditi at duo. Homero salutatus suscipiantur eum id, tamquam voluptaria expetendis ad sed, nobis feugiat similique usu ex.
<br><br>
Eum hinc argumentum te, no sit percipit adversarium, ne qui puto feugiat persecuti. Odio omnes scripserit ad est, ut vidit lorem maiestatis his, putent mandamus gloriatur ne pro. Oratio iriure rationibus ne his, ad est corrumpit splendide. Ad duo appareat moderatius, ei falli tollit denique eos. Dicant evertitur mei in, ne his deserunt perpetua sententiae, ea sea omnes similique vituperatoribus. Ex mel errem intellegebat comprehensam, vel ad tantas antiopam delicatissimi, tota ferri affert eu nec. Legere expetenda pertinacia ne pro, et pro impetus persius assueverit.
<br><br>
```

Ea mei nullam facete, omnis oratio offendit ius cu. Doming takimata repudiandae usu an, mei dicant takimata id, pri eleifend inimicus euripidis at. His vero singulis ea, quem euripidis abhorreant mei ut, et populo iriure vix. Usu ludus affert voluptaria ei, vix ea error definitiones, movet fastidii signiferumque in qui.
**

**
Vis prodesset adolescens adipiscing te, usu mazim perfecto recteque at, assum putant erroribus mea in. Vel facete imperdiet id, cum an libris luptatum perfecto, vel fabellas inciderint ut. Veri facete debitis ea vis, ut eos oratio erroribus. Sint facete perfecto no vel, vim id omnium insolens. Vel dolores perfecto pertinacia ut, te mel meis ullum dicam, eos assum facilis corpora in.
**

**
Mea te unum viderer dolores, nostrum detracto nec in, vis no partem definiebas constituam. Dicant utinam philosophia has cu, hendrerit prodesset at nam, eos an bonorum dissentiet. Has ad placerat intellegam consectetuer, no adipisci mandamus senserit pro, torquatos similique percipitur est ex. Pro ex putant deleniti repudiare, vel an aperiam sensibus suavitate. Ad vel epicurei convenire, ea soluta aliquid deserunt ius, pri in errem putant feugiat.
**

**
Sed iusto nihil populo an, ex pro novum homero cotidieque. Te utamur civibus eleifend qui, nam ei brute doming concludaturque, modo aliquam facilisi nec no. Vidisse maiestatis constituam eu his, esse pertinacia intellegam ius cu. Eos ei odio veniam, eu sumo altera adipisci eam, mea audiam prodesset persequeris ea. Ad vitae dictas vituperata sed, eum posse labore postulant id. Te eligendi principes dignissim sit, te vel dicant officiis repudiandae.
**

**
Id vel sensibus honestatis omittantur, vel cu nobis commune patrioque. In accusata definiebas qui, id tale malorum dolorem sed, solum clita phaedrum ne his. Eos mutat ullum forensibus ex, wisi perfecto urbanitas cu eam, no vis dicunt impetus. Assum novum in pri, vix an suavitate moderatius, id has reformidans referrentur. Elit inciderint omittantur duo ut, dicit democritum signiferumque eu est, ad suscipit delectus mandamus duo. An harum equidem maiestatis nec.
At has veri feugait placerat, in semper offendit praesent his. Omnium impetus facilis sed at, ex viris tincidunt ius. Unum eirmod dignissim id quo. Sit te atomorum quaerendum neglegentur, his primis tamquam et. Eu quo quot veri alienum, ea eos nullam luptatum accusamus. Ea mel causae phaedrum reprimique, at vidisse dolores ocurreret nam.
**

**

**

**
**<hr noshade size="5" width="50%" align="center">
**
**<FONT SIZE="-1"> Autor: Juan Manuel Rincon. </FONT>
**

** **

</body>

</html>

8.1.7 STYLE.CSS

```
body {background-color:  #DEB887;
      text-indent: 1cm
            margin: 0 auto;

}

body.indice {background-color: #F4A460;
          vertical-align: middle;
          text-align: justify;
}

b {
   font-size: xx-small;
   font-weight: bold;

}
```

8.2 PÁGINA EN PHP

8.2.1 AREA.PHTML

AREA

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

```
<body>
<div align="center">

<h3> Area restringida. </h3>

<?php
include("libreria.phtml");
session_start();
if($_SESSION["access"]==false) {
   echo "Debes logearte o registrarte para tener acceso a esta secci&oacute;n.";
?>
<br><br><br><br><br><br><br><br><br><br><br><br><br><br>
<?php
PiePagina();

}else {
echo "<img src='./files/homero.jpg' border='0' alt='Homero' align='center'>";
echo"
<br><br>
Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.
<br><br>
Ius id vidit volumus mandamus, vide veritus democritum te nec, ei eos debet libris consulatu. No mei ferri graeco dicunt, ad cum veri accommodare. Sed at malis omnesque delicata, usu et iusto zzril meliore. Dicunt maiorum eloquentiam cum cu, sit summo dolor essent te. Ne quodsi nusquam legendos has, ea dicit voluptua eloquentiam pro, ad sit quas qualisque. Eos vocibus deserunt quaestio ei.
<br><br>
Blandit incorrupte quaerendum in quo, nibh impedit id vis, vel no nullam semper audiam. Ei populo graeci consulatu mei, has ea stet modus phaedrum. Inani oblique ne has, duo et veritus detraxit. Tota ludus oratio ea mel, offendit persequeris ei vim. Eos dicat oratio partem ut, id cum ignota senserit intellegat. Sit inani ubique graecis ad, quando graecis liberavisse et cum, dicit option eruditi at duo. Homero salutatus suscipiantur eum id, tamquam voluptaria expetendis ad sed, nobis feugiat similique usu ex.
<br><br>
Eum hinc argumentum te, no sit percipit adversarium, ne qui puto feugiat persecuti. Odio omnes scripserit ad est, ut vidit lorem maiestatis his, putent mandamus gloriatur ne pro. Oratio iriure rationibus ne his, ad est corrumpit splendide. Ad duo appareat moderatius, ei falli tollit denique eos. Dicant evertitur mei in, ne his deserunt perpetua sententiae, ea sea omnes similique vituperatoribus. Ex mel errem intellegebat comprehensam, vel ad tantas antiopam delicatissimi, tota ferri affert eu nec. Legere expetenda pertinacia ne pro, et pro impetus persius assueverit.
<br><br>
Ea mei nullam facete, omnis oratio offendit ius cu. Doming takimata repudiandae usu an, mei dicant takimata id, pri eleifend inimicus euripidis at. His vero singulis ea, quem euripidis abhorreant mei ut, et populo iriure vix. Usu ludus affert voluptaria ei, vix ea error definitiones, movet fastidii signiferumque in qui.
<br><br>
Vis prodesset adolescens adipiscing te, usu mazim perfecto recteque at, assum putant erroribus mea in. Vel facete imperdiet id, cum an libris luptatum perfecto, vel fabellas inciderint ut. Veri facete debitis ea vis, ut eos oratio erroribus. Sint facete perfecto no vel, vim id omnium insolens. Vel dolores perfecto pertinacia ut, te mel meis ullum dicam, eos assum facilis corpora in.
<br><br>
Mea te unum viderer dolores, nostrum detracto nec in, vis no partem definiebas constituam. Dicant utinam philosophia has cu, hendrerit prodesset at nam, eos an bonorum dissentiet. Has ad placerat intellegam consectetuer, no adipisci mandamus senserit pro, torquatos similique percipitur est ex. Pro ex putant deleniti repudiare, vel an aperiam sensibus suavitate. Ad vel epicurei convenire, ea soluta aliquid deserunt ius, pri in errem putant feugiat.
<br><br>
Sed iusto nihil populo an, ex pro novum homero cotidieque. Te utamur civibus eleifend qui, nam ei brute doming concludaturque, modo aliquam facilisi nec no. Vidisse maiestatis constituam eu his, esse pertinacia intellegam ius cu. Eos ei odio veniam, eu sumo altera adipisci eam, mea audiam prodesset persequeris ea. Ad vitae dictas vituperata sed, eum posse labore postulant id. Te eligendi principes dignissim sit, te vel dicant officiis repudiandae.
<br><br>
Id vel sensibus honestatis omittantur, vel cu nobis commune patrioque. In accusata definiebas qui, id tale malorum dolorem sed, solum clita phaedrum ne his. Eos mutat ullum forensibus ex, wisi perfecto urbanitas cu eam, no vis dicunt impetus. Assum novum in pri, vix an suavitate moderatius, id has reformidans referrentur. Elit inciderint omittantur duo ut, dicit democritum signiferumque eu est, ad suscipit delectus mandamus duo. An harum equidem maiestatis nec.
At has veri feugait placerat, in semper offendit praesent his. Omnium impetus facilis sed at, ex viris tincidunt ius. Unum eirmod dignissim id quo. Sit te atomorum quaerendum neglegentur, his primis tamquam et. Eu quo quot veri alienum, ea eos nullam luptatum accusamus. Ea mel causae phaedrum reprimique, at vidisse dolores ocurreret nam.
<br><br><br><br>
<h3>Añadir un nuevo Dios:</h3>";

?>
<form action="crea_dios.php" method="POST"><br>
   Nombre:<br> <input type="text" name="nombre"><br><br>
   Descripcion:<br> <textarea name="descripcion" cols=50 rows=8></textarea><br><br>
 <br>
   <input type="submit" name="Crear" value="Enviar">
</form>



<?php

PiePagina();
}

?>
</div>
</body>
```

```
</html>
```

8.2.2 CABECERA.PHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

  <img src="http://localhost/HtmlProject/files/LLII.jpg" alt="Cabecera" style="position: absolute; top: 0; left: 0; width: 100%; height: 100%" /

</body>

</html>
```

8.2.3 COMPRUEBA.PHP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

   <?php
   session_start();
   $_SESSION["access"] = false;

   $server="localhost";
   $database="tesis"; /* Nuestra base de datos */
   $dbpass="samsung"; /*Nuestro password mysql */
   $dbuser="root"; /* Nuestro user mysql */

   $login1=($_POST['login']);
   $pass1=$_POST['pass'];
   $query="SELECT * FROM Usuario WHERE (nombre='$login1')";



   $link=mysql_connect($server,$dbuser,$dbpass);


   if (!$link) {

      die('Could not connect: ' . mysql_error());
   }



   $result=mysql_db_query($database,$query,$link);


   if(mysql_numrows($result)==0) {



      header ("Location: IndiceX.phtml");
   } else {


      $array=mysql_fetch_array($result);
      if($array["password"]==$pass1 ) {
         //crypt($pass1,"semilla") )
         /* Comprobamos que el password encriptado en la BD coincide con el password que nos han dado al encriptarlo. Recuerda usar la misma semilla para encriptar los dos passwords. */

         $_SESSION["nombre"]=$array["nombre"];
         $_SESSION["apellidos"]=$array["apellidos"];
         $_SESSION["login"]=$array["login"];
         $login=$array["login"];
         $_SESSION["access"] = true;
```

```
        session_register("SESSION");


        header("location: Indice.phtml");


      } else {
        header ("Location: IndiceX.phtml");

      } /* Cerramos este ultimo else */
   } /* Cerramos el else que corresponde a la comprobación de que el login existe */

   ?>

</body>

</html>
```

8.2.4 CONTACTO.PHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
<div align="center">

<H1>Contacta con nosotros:</H1><br>

      <FORM ACTION="mail.phtml" METHOD="GET">
      <br>
      Introduzca su direccion de email:<br>
      <INPUT TYPE="text" NAME="direccion"><BR><BR>

      Introduzca el asunto del mail:<br>
      <INPUT TYPE="text" NAME="asunto"><BR><BR>

      Introduzca el contenido del mail:<br>
      <textarea name="cuerpo" cols=32 rows=6></textarea><br><br>

       <INPUT TYPE="submit" VALUE="Enviar">
      </FORM>

   <?php
      include 'libreria.phtml';
      PiePagina();
   ?>
</div>
</body>

<html>
```

8.2.5 CREA_DIOS.PHP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

<?php
session_start();
$server="localhost";
$database="tesis"; /* Nuestra base de datos */
$dbpass="samsung"; /*Nuestro password mysql */
$dbuser="root"; /* Nuestro user mysql */

/* Primero comprobamos que no existe un usuario con el mismo login ya registrado */

$nombre=($_POST['nombre']);
$descripcion=($_POST['descripcion']);
```

```
$query="SELECT * FROM Noticias WHERE titulo='$nombre'";
$link=mysql_connect($server,$dbuser,$dbpass);
$result=mysql_db_query($database,$query,$link);
if(mysql_num_rows($result)) {
   echo "El dios ya existe en la Base de Datos.";
} else {
  mysql_free_result($result);
      $query="INSERT INTO Dioses ( nombre, descripcion) VALUES ('$nombre','$descripcion')";

      $result=mysql_db_query($database,$query,$link);

      if (!$link) {
         die('Could not connect: ' . mysql_error());
      }


       header("location: Clientes.phtml");



} /* Cierre del else que corresponde a if(mysql_num_rows...) */
include("libreria.phtml");
PiePagina();

?>

</body>

</html>
```

8.2.6 CREA_USER.PHP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

<?
session_start();
$server="localhost";
$database="tesis"; /* Nuestra base de datos */
$dbpass="samsung"; /*Nuestro password mysql */
$dbuser="root"; /* Nuestro user mysql */

/* Primero comprobamos que no existe un usuario con el mismo login ya registrado */

$login=($_POST['login']);
$pass1=($_POST['pass1']);
$pass2=($_POST['pass2']);
$nombre=($_POST['nombre']);
$apellidos=($_POST['apellidos']);
$email=($_POST['email']);

$query="SELECT * FROM Usuario WHERE login='$login'";
$link=mysql_connect($server,$dbuser,$dbpass);
$result=mysql_db_query($database,$query,$link);
if(mysql_num_rows($result)) {
   echo "El usuario ya existe en la Base de Datos.";
} else {
   mysql_free_result($result);
   /* Ahora comprobamos que los dos pass coinciden */
   if($pass1!=$pass2) {
      echo "Los passwords deben coincidir.<br>";

   } else {
//$pass1=crypt($pass2, "semilla");

      /* Encripatmos el password, con la clave "semilla" que debeis sustituirpor la que mas os guste. Hay otros metodos de
encriptacion, mirad en php.net si quereis conocerlos. */

      $query="INSERT INTO Usuario ( nombre, apellidos, login, password, email) VALUES
('$nombre','$apellidos','$login','$pass1','$email')";

      $result=mysql_db_query($database,$query,$link);
```

```
      if (!$link) {
         die('Could not connect: ' . mysql_error());
      }

      if(mysql_affected_rows($link)) {
         echo "<br><br>Usuario introducido correctamente.";
echo "<br><br><br><a href='Formulario.phtml'>Volver a Formulario</a><br><br><br>";

      } else {
         echo "<br><br>Error al introducir el usuario.";
echo "<br><br><br><a href='Formulario.phtml'>Volver a Formulario</a><br><br><br>";

      } /* Cierre del else */
   } /* Cierre del else que corresponde a if(mysql_affected_rows.....) */
} /* Cierre del else que corresponde a if(mysql_num_rows...) */
include("libreria.phtml");
echo "<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>";
PiePagina();

?>

</body>

</html>
```

8.2.7 DIOSES.PHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

<div align="center">
   <h2> Dioses. </h2>
</div>

<div align="center">
   <?php
   include 'libreria.phtml';
   echo "<img src='./files/202.jpg' border='0' alt='hola?' align='center'>";
   ?>
</div>

<br><br>

<?php
session_start();
$server="localhost";
   $database="tesis"; /* Nuestra base de datos */
   $dbpass="samsung"; /*Nuestro password mysql */
   $dbuser="root"; /* Nuestro user mysql */


   $query="SELECT * FROM Dioses";

$link=mysql_connect($server,$dbuser,$dbpass);

if (!$link) {
      die('Could not connect: ' . mysql_error());
  }
$result=mysql_db_query($database,$query,$link);

while($row = mysql_fetch_array($result))
{
echo "<br><h3>";
echo $row['nombre'];
echo "</h1>";
echo $row['descripcion'];
echo "<br>_____<br>";
}

mysql_close($link);


?>
```

```
<br><br><br>
Para poder añadir más dioses debes estar registrado.


<br><br><br>
Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.


<?php


PiePagina();
?>

</body>

</html>
```

8.2.8 FORMULARIO.PHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
<div align="center">
<br>
Complete los datos para registrarse en nuestra base de datos:
<br>

<form action="crea_user.php" method="POST"><br>
   Login: <input type="text" name="login"><br>
   Password: <input type="password" name="pass1"><br>
   Repite Password: <input type="password" name="pass2"><br><br>
   Nombre: <input type="text" name="nombre"><br>
   Apellidos: <input type="text" name="apellidos"><br>
   E-mail: <input type="text" name="email"><br>
 <br>
   <input type="submit" name="Crear" value="Enviar">
</form>

<br><br><br><br><br>
<?php
include("libreria.phtml");
PiePagina();
?>
</div>
</body>

</html>


INDEX.PHP

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<html>

   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>PHP</title>
   </head>

   <frameset rows="20%,*">
      <frame name="Cabecera" src="Cabecera.phtml" marginwidth="10" marginheight="10" scrolling="auto" >

         <frameset cols="15%,*">

            <frame name="Indice" src="Indice.phtml" marginwidth="10" marginheight="10" scrolling="auto" >
               <frame name="Cuerpo" src="Principal.phtml" marginwidth="10" marginheight="10" scrolling="auto" >

               </frame>
            </frame>

         </frameset>
```

```
        <noframes>
          <body> <p> Esta p&aacute;gina utiliza frames. Es posible que si utiliza un navegador antiguo no se est&eacute; mostrando
adecuadamente.</p>
          </body>
        </noframes>
   </frameset>

   <body>



   </body>
</html>
```

8.2.9 INDICE.PHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body class="indice">


   <?

   session_start();
   if($_SESSION["access"]==true) {

      $login1=($_SESSION["login"]);

      echo "<b><font size=1>";
      echo "<br>Bienvenido ".$login1;


      echo " <br><a href='logout.php'>logout</a><br><br><br>";
      echo "</font></b>";

   } else {

      /* Cerramos la parte de codigo PHP porque vamos a escribir bastante HTML y nos será mas cómodo así que metiendo echo's */
      ?>


   <form action="comprueba.php" method="POST" target="Indice"><br>
      Login: <input type="text" name="login"><br>
      Password: <input type="password" name="pass"><br>
      <input type="submit" value="Entrar">
   </form>
   <br>
      <?
   } /* Y cerramos el else */
   ?>



   <A HREF="Principal.phtml" target="Cuerpo">Inicio</A><BR><BR>
   <A HREF="Formulario.phtml" target="Cuerpo">Registrarse</A><BR><BR>
   <A HREF="Contacto.phtml" target="Cuerpo">Contacta</A><BR><BR>
   <A HREF="Quienes.phtml" target="Cuerpo">Quienes somos</A><BR><BR>
   <A HREF="Dioses.phtml" target="Cuerpo">Dioses</A><BR><BR>
   <A HREF="Area.phtml" target="Cuerpo">Area Restringida</A><BR><BR>


</body>
</html>
```

8.2.10 INDICEX.PHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body class="indice">
```

```
<?

session_start();
if($_SESSION["access"]==true) {

   $login1=($_SESSION["login"]);

   echo "<b><font size=1>";
   echo "<br>Bienvenido ".$login1;


   echo " <br><a href='logout.php'>logout</a><br><br><br>";
   echo "</font></b>";

} else {

   /* Cerramos la parte de codigo PHP porque vamos a escribir bastante HTML y nos será mas cómodo así que metiendo echo's */
   ?>


<form action="comprueba.php" method="POST" target="Indice"><br>
   Login: <input type="text" name="login"><br>
   Password: <input type="password" name="pass"><br>
   <input type="submit" value="Entrar">
</form>
<br>
<font size="-2"><b>Error al hacer login.</b></font>
<br><br>
   <?
} /* Y cerramos el else */
?>



<A HREF="Principal.phtml" target="Cuerpo">Inicio</A><BR><BR>
<A HREF="Formulario.phtml" target="Cuerpo">Registrarse</A><BR><BR>
<A HREF="Contacto.phtml" target="Cuerpo">Contacta</A><BR><BR>
<A HREF="Quienes.phtml" target="Cuerpo">Quienes somos</A><BR><BR>
<A HREF="Dioses.phtml" target="Cuerpo">Dioses</A><BR><BR>
<A HREF="Area.phtml" target="Cuerpo">Area Restringida</A><BR><BR>


</body>
</html>
```

8.2.11 LIBRERIA.PHP

```
<!-- Libreria PHP -->

<?php


function PiePagina() {
   ?><br><br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center">Autor: Juan Manuel Rincon.</p></FONT><BR>
   <?
}
?>
```

8.2.12 LOGOUT.PHP

```
<?php
session_start();
if(!isset($_SESSION)) {
   header("location: Indice.phtml");
} else {
   session_unset();
   session_destroy();
   header("location: Indice.phtml");
}
?>
```

8.2.13 MAIL.PHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

   <body>

   <H1>Enviado!</H1>

   <?php
   //$direccion=$_GET['direccion'];

   $cuerpo=$_GET['cuerpo'];
   $asunto=$_GET['asunto'];
   //$from="web@andurin.com";

 $from=$_GET['direccion'];
   $cabeceras= "From: $from";

   if ($from=="") {

      mail("juanmanuel.rincon.carrera@gmail.com", $asunto, $cuerpo, $cabeceras);

      echo "El mail se ha enviado correctamente.";
   }else{
mail("juanmanuel.rincon.carrera@gmail.com", $asunto, $cuerpo, $cabeceras);
   echo "Se ha enviado un email desde la direccion: ",$from,".";
   }

   echo "<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>";
   include 'libreria.phtml';
   PiePagina(); ?>
</body>
</html>

PRINCIPAL

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

   <?php
   include("libreria.phtml");?>
<TABLE>
      <div align="center">
        <?php
         echo "<img src='./files/Ciceron.jpg' border='0' alt='hola?' align='center'>";
         ?>
      </div>

  </TABLE>

  <?php
      echo "<br><br><br><br>Est enim, iudices, haec non scripta, sed nata lex, quam non didicimus, accepimus, legimus, verum ex natura ipsa arripuimus, hausimus, expressimus, ad quam non docti, sed facti, non instituti, sed imbuti sumus.<br><br>Marco Tulio CICER&Oacute;N<br><br><br><br>Existe, de hecho, jueces, una ley no escrita, sino innata, la cual no hemos aprendido, heredado, leído, sino que de la misma naturaleza la hemos agarrado, exprimido, apurado, ley para la que no hemos sido educados, sino hechos; y en la que no hemos sido instruidos, sino empapados." ;
  PiePagina();
  ?>
</body>

</html>
```

8.2.14 QUIENES.PHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
<div align="center">
<h3> Quienes somos. </h3>
<br><br>
```

Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.
**

**
Ius id vidit volumus mandamus, vide veritus democritum te nec, ei eos debet libris consulatu. No mei ferri graeco dicunt, ad cum veri accommodare. Sed at malis omnesque delicata, usu et iusto zzril meliore. Dicunt maiorum eloquentiam cum cu, sit summo dolor essent te. Ne quodsi nusquam legendos has, ea dicit voluptua eloquentiam pro, ad sit quas qualisque. Eos vocibus deserunt quaestio ei.
**

**
Blandit incorrupte quaerendum in quo, nibh impedit id vis, vel no nullam semper audiam. Ei populo graeci consulatu mei, has ea stet modus phaedrum. Inani oblique ne has, duo et veritus detraxit. Tota ludus oratio ea mel, offendit persequeris ei vim. Eos dicat oratio partem ut, id cum ignota senserit intellegat. Sit inani ubique graecis ad, quando graecis liberavisse et cum, dicit option eruditi at duo. Homero salutatus suscipiantur eum id, tamquam voluptaria expetendis ad sed, nobis feugiat similique usu ex.
**

**
Eum hinc argumentum te, no sit percipit adversarium, ne qui puto feugiat persecuti. Odio omnes scripserit ad est, ut vidit lorem maiestatis his, putent mandamus gloriatur ne pro. Oratio iriure rationibus ne his, ad est corrumpit splendide. Ad duo appareat moderatius, ei falli tollit denique eos. Dicant evertitur mei in, ne his deserunt perpetua sententiae, ea sea omnes similique vituperatoribus. Ex mel errem intellegebat comprehensam, vel ad tantas antiopam delicatissimi, tota ferri affert eu nec. Legere expetenda pertinacia ne pro, et pro impetus persius assueverit.
**

**
Ea mei nullam facete, omnis oratio offendit ius cu. Doming takimata repudiandae usu an, mei dicant takimata id, pri eleifend inimicus euripidis at. His vero singulis ea, quem euripidis abhorreant mei ut, et populo iriure vix. Usu ludus affert voluptaria ei, vix ea error definitiones, movet fastidii signiferumque in qui.
**

**
Vis prodesset adolescens adipiscing te, usu mazim perfecto recteque at, assum putant erroribus mea in. Vel facete imperdiet id, cum an libris luptatum perfecto, vel fabellas inciderint ut. Veri facete debitis ea vis, ut eos oratio erroribus. Sint facete perfecto no vel, vim id omnium insolens. Vel dolores perfecto pertinacia ut, te mel meis ullum dicam, eos assum facilis corpora in.
**

**
Mea te unum viderer dolores, nostrum detracto nec in, vis no partem definiebas constituam. Dicant utinam philosophia has cu, hendrerit prodesset at nam, eos an bonorum dissentiet. Has ad placerat intellegam consectetuer, no adipisci mandamus senserit pro, torquatos similique percipitur est ex. Pro ex putant deleniti repudiare, vel an aperiam sensibus suavitate. Ad vel epicurei convenire, ea soluta aliquid deserunt ius, pri in errem putant feugiat.
**

**
Sed iusto nihil populo an, ex pro novum homero cotidieque. Te utamur civibus eleifend qui, nam ei brute doming concludaturque, modo aliquam facilisi nec no. Vidisse maiestatis constituam eu his, esse pertinacia intellegam ius cu. Eos ei odio veniam, eu sumo altera adipisci eam, mea audiam prodesset persequeris ea. Ad vitae dictas vituperata sed, eum posse labore postulant id. Te eligendi principes dignissim sit, te vel dicant officiis repudiandae.
**

**
Id vel sensibus honestatis omittantur, vel cu nobis commune patrioque. In accusata definiebas qui, id tale malorum dolorem sed, solum clita phaedrum ne his. Eos mutat ullum forensibus ex, wisi perfecto urbanitas cu eam, no vis dicunt impetus. Assum novum in pri, vix an suavitate moderatius, id has reformidans referrentur. Elit inciderint omittantur duo ut, dicit democritum signiferumque eu est, ad suscipit delectus mandamus duo. An harum equidem maiestatis nec.
At has veri feugait placerat, in semper offendit praesent his. Omnium impetus facilis sed at, ex viris tincidunt ius. Unum eirmod dignissim id quo. Sit te atomorum quaerendum neglegentur, his primis tamquam et. Eu quo quot veri alienum, ea eos nullam luptatum accusamus. Ea mel causae phaedrum reprimique, at vidisse dolores ocurreret nam.
**

**

```
<?php
include 'libreria.phtml';


PiePagina();
?>

</div>

</body>

</html>
```

8.2.15 STYLE.CSS

```
body {background-color:  #DEB887;
     text-indent: 1cm

   margin: 0 auto;

}


body.indice {background-color: #F4A460;
          vertical-align: middle;
          text-align: justify;
}


b {
   font-size: xx-small;
   font-weight: bold;
```

8.3 PÁGINA EN PSP

8.3.1 AREA.PSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>


<div align="center">
<%

import cgi


if not session:
req.write("<body><h3> Area restringida. </h3>Debes logearte o registrarte para tener acceso a esta secci&oacute;n.")
%>
<br><br><br><br><br><br><br><br><br><br><br><br><br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center"> Autor: Juan Manuel Rincon.</p></FONT><BR>
<%
else:
kk=0
%>
<h3> Area restringida. </h3>
<img src='./files/homero.jpg' border='0' alt='Homero' align='center'>
<br><br>
Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.
<br><br>
Ius id vidit volumus mandamus, vide veritus democritum te nec, ei eos debet libris consulatu. No mei ferri graeco dicunt, ad cum veri accommodare. Sed at malis omnesque delicata, usu et iusto zzril meliore. Dicunt maiorum eloquentiam cum cu, sit summo dolor essent te. Ne quodsi nusquam legendos has, ea dicit voluptua eloquentiam pro, ad sit quas qualisque. Eos vocibus deserunt quaestio ei.
<br><br>
Blandit incorrupte quaerendum in quo, nibh impedit id vis, vel no nullam semper audiam. Ei populo graeci consulatu mei, has ea stet modus phaedrum. Inani oblique ne has, duo et veritus detraxit. Tota ludus oratio ea mel, offendit persequeris ei vim. Eos dicat oratio partem ut, id cum ignota senserit intellegat. Sit inani ubique graecis ad, quando graecis liberavisse et cum, dicit option eruditi at duo. Homero salutatus suscipiantur eum id, tamquam voluptaria expetendis ad sed, nobis feugiat similique usu ex.
<br><br>
Eum hinc argumentum te, no sit percipit adversarium, ne qui puto feugiat persecuti. Odio omnes scripserit ad est, ut vidit lorem maiestatis his, putent mandamus gloriatur ne pro. Oratio iriure rationibus ne his, ad est corrumpit splendide. Ad duo appareat moderatius, ei falli tollit denique eos. Dicant evertitur mei in, ne his deserunt perpetua sententiae, ea sea omnes similique vituperatoribus. Ex mel errem intellegebat comprehensam, vel ad tantas antiopam delicatissimi, tota ferri affert eu nec. Legere expetenda pertinacia ne pro, et pro impetus persius assueverit.
<br><br>
Ea mei nullam facete, omnis oratio offendit ius cu. Doming takimata repudiandae usu an, mei dicant takimata id, pri eleifend inimicus euripidis at. His vero singulis ea, quem euripidis abhorreant mei ut, et populo iriure vix. Usu ludus affert voluptaria ei, vix ea error definitiones, movet fastidii signiferumque in qui.
<br><br>
Vis prodesset adolescens adipiscing te, usu mazim perfecto recteque at, assum putant erroribus mea in. Vel facete imperdiet id, cum an libris luptatum perfecto, vel fabellas inciderint ut. Veri facete debitis ea vis, ut eos oratio erroribus. Sint facete perfecto no vel, vim id omnium insolens. Vel dolores perfecto pertinacia ut, te mel meis ullum dicam, eos assum facilis corpora in.
<br><br>
Mea te unum viderer dolores, nostrum detracto nec in, vis no partem definiebas constituam. Dicant utinam philosophia has cu, hendrerit prodesset at nam, eos an bonorum dissentiet. Has ad placerat intellegam consectetuer, no adipisci mandamus senserit pro, torquatos similique percipitur est ex. Pro ex putant deleniti repudiare, vel an aperiam sensibus suavitate. Ad vel epicurei convenire, ea soluta aliquid deserunt ius, pri in errem putant feugiat.
<br><br>
Sed iusto nihil populo an, ex pro novum homero cotidieque. Te utamur civibus eleifend qui, nam ei brute doming concludaturque, modo aliquam facilisi nec no. Vidisse maiestatis constituam eu his, esse pertinacia intellegam ius cu. Eos ei odio veniam, eu sumo altera adipisci eam, mea audiam prodesset persequeris ea. Ad vitae dictas vituperata sed, eum posse labore postulant id. Te eligendi principes dignissim sit, te vel dicant officiis repudiandae.
<br><br>
Id vel sensibus honestatis omittantur, vel cu nobis commune patrioque. In accusata definiebas qui, id tale malorum dolorem sed, solum clita phaedrum ne his. Eos mutat ullum forensibus ex, wisi perfecto urbanitas cu eam, no vis dicunt impetus. Assum novum in pri, vix an suavitate moderatius, id has reformidans referrentur. Elit inciderint omittantur duo ut, dicit democritum signiferumque eu est, ad suscipit delectus mandamus duo. An harum equidem maiestatis nec.
At has veri feugait placerat, in semper offendit praesent his. Omnium impetus facilis sed at, ex viris tincidunt ius. Unum eirmod dignissim id quo. Sit te atomorum quaerendum neglegentur, his primis tamquam et. Eu quo quot veri alienum, ea eos nullam luptatum accusamus. Ea mel causae phaedrum reprimique, at vidisse dolores ocurreret nam.
<br><br><br><br>
```

```
<h3>Añadir un nuevo Dios:</h3>

<form action="crea_dios.psp" method="POST"><br>
Nombre:<br> <input type="text" name="nombre"><br><br>
Descripcion:<br> <textarea name="descripcion" cols=50 rows=8></textarea><br><br>
<br>
<input type="submit" name="Crear" value="Enviar">
</form>

<br><br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center"> Autor: Juan Manuel Rincon.</p></FONT><BR>

</div>

</body>

</html>
```

8.3.2 CABECERA.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

<img src="http://localhost/PythonProject/files/LLII.jpg" alt="Cabecera" style="position: absolute; top: 0; left: 0; width: 100%; height:
100%" />


</body>

</html>
```

8.3.3 COMPRUEBA.PSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body class="indice">
<%
import cgi,MySQLdb,string,_mysql,_mysql_exceptions

a=form['login']
b=form['pass1']

db=MySQLdb.connect(host="localhost",user="root",passwd="samsung",db="tesis")

cursor=db.cursor()

cod=cursor.execute("SELECT count(*) FROM Usuario WHERE login = '" + a + "'")

lines = cursor.fetchone()


result=str(lines[0])


if result=='1':
cursor1=db.cursor()
cursor1.execute("SELECT count(*) FROM Usuario WHERE login = '" + a + "' and password ='" + b + "'")
lines1=cursor1.fetchone()
result1=str(lines1[0])
if not result1=='1':
%>

<b><font size=1>
      <br>
<%
```

```
req.write('Password incorrecto.')

%>
<br>
     <a href='logout.psp'>volver</a><br><br><br>
<%

else:
session['abierta']= a
%>

<b><font size=1>
       <br>
    <% req.write("Bienvenido " + a) %>
       <br>
       <a href='logout.psp'>logout</a><br><br><br>

<%


else:
%>
<b><font size=1>
       <br>
<%
req.write('No existe el login introducido.')
%>
<br>
  <a href='logout.psp'>volver</a><br><br><br>
<%
p=1
%>
</b></font><br>




<br>
<A HREF="Principal.psp" target="Cuerpo">Inicio</A><BR><BR>
<A HREF="Formulario.psp" target="Cuerpo">Registrarse</A><BR><BR>
<A HREF="Contacto.html" target="Cuerpo">Contacta</A><BR><BR>
<A HREF="Quienes.html" target="Cuerpo">Quienes somos</A><BR><BR>
<A HREF="Dioses.psp" target="Cuerpo">Dioses</A><BR><BR>
<A HREF="Area.psp" target="Cuerpo">Area Restringida</A><BR><BR>


</body>

</html>
```

8.3.4 CONTACTO.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
<div align="center">
<H1>Contacta con nosotros:</H1><br>

      <FORM ACTION="mail.psp" METHOD="GET">
      <br>
      Introduzca su direccion de email:<br>
      <INPUT TYPE="text" NAME="direccion"><BR><BR>

      Introduzca el asunto del mail:<br>
      <INPUT TYPE="text" NAME="asunto"><BR><BR>

      Introduzca el contenido del mail:<br>
      <textarea name="cuerpo" cols=32 rows=6></textarea><br><br>

       <INPUT TYPE="submit" VALUE="Enviar">
      </FORM>

  <br><br>
  <p id="Pie">
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center" position:absolute; bottom:0>Autor: Juan Manuel Rincon.</p></FONT><BR>
```

```
   </div>
</body>

</html>
```

8.3.5 CREA_DIOS.PSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

<%

import cgi,MySQLdb,string,_mysql,_mysql_exceptions

control=0

nombre=form['nombre']
descripcion=form['descripcion']


if not nombre or not descripcion:
control=1
req.write("<br><br>Debes rellenar el nombre y la descripcion.<br><br>")
req.write('<br> <a href="Area.psp">Recargar formulario.</a><br>')


db=MySQLdb.connect(host="localhost",user="root",passwd="samsung",db="tesis")

cursor=db.cursor()

cod=cursor.execute("SELECT * FROM Dioses WHERE nombre = '" + nombre + "'")



lines = cursor.fetchone()

if lines != None:
line1 = str(lines[0])




req.write ('<br><br>El dios '+line1+' ya existe en la Base de Datos.<br>')
req.write('<br> <a href="Area.psp">Recargar formulario.</a><br>')
else:


cursor1=db.cursor()
cursor1.execute("INSERT INTO Dioses ( nombre, descripcion) VALUES (%s,%s)",(nombre,descripcion))

req.write("""<br><br>Se ha introducido el dios """ + nombre + """ correctamente.""")
req.write("""<br><br><br><br><br><br> <a href="Area.psp">Recargar formulario.</a><br><br><br>""")

%>

</body>

</html>
```

8.3.6 CREA_USER.PSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
```

```
<%

import cgi,MySQLdb,string,_mysql,_mysql_exceptions

control=0

login=form['login']
pass1=form['pass1']
pass2=form['pass2']
nombre=form['nombre']
apellidos=form['apellidos']
email=form['email']

if not login or not pass1 or not pass2 or not nombre or not apellidos or not email:
control=1
req.write("<br><br>Debes rellenar todos los campos.<br><br>")
req.write('<br> <a href="Formulario.psp">Volver a Formulario.</a><br>')


db=MySQLdb.connect(host="localhost",user="root",passwd="samsung",db="tesis")

cursor=db.cursor()

cod=cursor.execute("SELECT count(*) FROM Usuario WHERE login = '" + login + "'")



lines = cursor.fetchall()

result=''

for line in lines:
for column in line:
if column == 'None':
result= result + 'null'
else:
result=result+str(column)+"


if result=='1' and not control==1:
req.write ('<br><br>El login ya existe en la Base de Datos.<br>')
req.write('<br> <a href="Formulario.psp">Volver a Formulario.</a><br>')
else:
if not pass1==pass2 and not control==1:
req.write('<br>Los passwords deben conincidir.<br><br>')
req.write('<br> <a href="Formulario.psp">Volver a Formulario.</a><br>')

elif not control==1:
cursor1=db.cursor()
cursor1.execute("INSERT INTO Usuario ( nombre, apellidos, login, password, email) VALUES
(%s,%s,%s,%s,%s)",(nombre,apellidos,login,pass1,email))

req.write("""<br><br>Usuario introducido correctamente""")
req.write("""<br><br><br><br><br><br> <a href="Formulario.psp">Volver a Formulario.</a><br><br><br>""")

%>

</body>

</html>
```

8.3.7 DIOSES.PSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

<div align="center">
   <h2> Dioses. </h2>
</div>

<div align="center">
        <img src="files/202.jpg">
     </div>
```

```
<br><br>

<%
import cgi,MySQLdb,string,_mysql,_mysql_exceptions

db = MySQLdb.connect(host="localhost",user="root",passwd="samsung",db="tesis")

cursor = db.cursor()

cod = cursor.execute("SELECT * FROM Dioses")

while (1):
lines=cursor.fetchone()
if lines == None: break
lines1 = str(lines[0])
lines2 = str(lines[1])
req.write('<br><h3>' + lines1 + '</h3>' + lines2 + '<br>_____<br>')



%>



<br><br><br>
Para poder añadir más dioses debes estar registrado.


<br><br><br>
Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.

<br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center"> Autor: Juan Manuel Rincon.</p></FONT><BR>



</body>

</html>
```

8.3.8 FORMULARIO.PSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>
<div align="center">
<br>
Complete los datos para registrarse en nuestra base de datos:
<br>
<form action="crea_user.psp" method="POST"><br>
   Login: <input type="text" name="login"><br>
   Password: <input type="password" name="pass1"><br>
   Repite Password: <input type="password" name="pass2"><br><br>
   Nombre: <input type="text" name="nombre"><br>
   Apellidos: <input type="text" name="apellidos"><br>
   E-mail: <input type="text" name="email"><br>
   <input type="submit" name="Crear">
</form>

<br><br><br><br><br><br><br>
<p id="Pie">
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center" position:absolute; bottom:0>Autor: Juan Manuel Rincon.</p></FONT><BR>

</p>

</div>
</body>

</html>
```

8.3.9 INDEX.PSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>

   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Prueba PSP</title>
   </head>
   <frameset rows="20%,*">
      <frame name="Cabecera" src="Cabecera.html" marginwidth="10" marginheight="10" scrolling="auto" >

         <frameset cols="15%,*">

            <frame name="Indice" src="Indice.psp" marginwidth="10" marginheight="10" scrolling="auto" >
               <frame name="Cuerpo" src="Principal.psp" marginwidth="10" marginheight="10" scrolling="auto" >

               </frame>
            </frame>

         </frameset>
         <noframes>
            <body> <p> Esta p&aacute;gina utiliza frames. Es posible que si utiliza un navegador antiguo no se est&eacute; mostrando
adecuadamente.</p>
            </body>
         </noframes>
   </frameset>

   <body>



   </body>
</html>
```

8.3.10 INDICE.PSP

```
<link rel="stylesheet" type="text/css" href="style.css" />
<body class="indice">


<%

import cgi

if session.has_key('abierto'):
a=form['login']
%>

<html><b><font size=1>
      <br>
      <% req.write("Bienvenido " + a) %>
      <br>
      <a href='logout.psp'>logout</a><br><br><br>
      </font></b>
<%
else:
%>


   <form action="comprueba.psp" method="POST" target="Indice"><br>
      Login: <input type="text" name="login"><br>
      Password: <input type="password" name="pass1"><br>
      <input type="submit" value="Entrar">
   </form>
   <br>


<A HREF="Principal.psp" target="Cuerpo">Inicio</A><BR><BR>
```

```
<A HREF="Formulario.psp" target="Cuerpo">Registrarse</A><BR><BR>
<A HREF="Contacto.html" target="Cuerpo">Contacta</A><BR><BR>
<A HREF="Quienes.html" target="Cuerpo">Quienes somos</A><BR><BR>
<A HREF="Dioses.psp" target="Cuerpo">Dioses</A><BR><BR>
<A HREF="Area.psp" target="Cuerpo">Area Restringida</A><BR><BR>


</body>
</html>
```

8.3.11 LOGOUT.PSP

```
<%
session.delete()

psp.redirect('/PythonProject/Indice.psp')
%>
```

8.3.12 MAIL.PSP

```
<html>
   <head>
      <title>Ejemplo de PSP</title>
   </head>
   <body>
   <link rel="stylesheet" type="text/css" href="style.css" />

   <H1>Enviado!</H1>

<%
import smtplib

def email(req, direccion, asunto, cuerpo):
   if not (direccion and asunto and cuerpo):
      req.write("Falta un campo del mail.")


   msg = """\
From: %s
Subject: %s
To: webmaster

I have the following comment:

%s

Thank You,



""" % (direccion,asunto, cuerpo)




   conn = smtplib.SMTP("localhost")
   conn.sendmail(email, ["juanmanuel.rincon.carrera@gmail.com"], msg)
   conn.quit()

%>


</body>
</html>
```

8.3.13 PRINCIPAL.PSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />
```

```
<html>
<body>

   <TABLE>
      <div align="center">
         <img src="files/Ciceron.jpg" width="504" height="360" border="0" alt="Ciceron">
      </div>
     <p><br><br><br><br>Est enim, iudices, haec non scripta, sed nata lex, quam non didicimus, accepimus, legimus, verum ex natura ipsa arripuimus, hausimus, expressimus, ad quam non docti, sed facti, non instituti, sed imbuti sumus.<br><br>Marco Tulio CICER&Oacute;N<br><br><br><br>Existe, de hecho, jueces, una ley no escrita, sino innata, la cual no hemos aprendido, heredado, le&iacute;do, sino que de la misma naturaleza la hemos agarrado, exprimido, apurado, ley para la que no hemos sido educados, sino hechos; y en la que no hemos sido instruidos, sino empapados. </p>

   </TABLE>
   <br><br>
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center"> Autor: Juan Manuel Rincon.</p></FONT><BR>


</body>

</html>
```

8.3.14 QUIENES.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<link rel="stylesheet" type="text/css" href="style.css" />

<html>

<body>

<div align="center">

<h3> Quienes somos. </h3>
<br><br>
Lorem ipsum ad his scripta blandit partiendo, eum fastidii accumsan euripidis in, eum liber hendrerit an. Qui ut wisi vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetuer signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.
<br><br>
Ius id vidit volumus mandamus, vide veritus democritum te nec, ei eos debet libris consulatu. No mei ferri graeco dicunt, ad cum veri accommodare. Sed at malis omnesque delicata, usu et iusto zzril meliore. Dicunt maiorum eloquentiam cum cu, sit summo dolor essent te. Ne quodsi nusquam legendos has, ea dicit voluptua eloquentiam pro, ad sit quas qualisque. Eos vocibus deserunt quaestio ei.
<br><br>
Blandit incorrupte quaerendum in quo, nibh impedit id vis, vel no nullam semper audiam. Ei populo graeci consulatu mei, has ea stet modus phaedrum. Inani oblique ne has, duo et veritus detraxit. Tota ludus oratio ea mel, offendit persequeris ei vim. Eos dicat oratio partem ut, id cum ignota senserit intellegat. Sit inani ubique graecis ad, quando graecis liberavisse et cum, dicit option eruditi at duo. Homero salutatus suscipiantur eum id, tamquam voluptaria expetendis ad sed, nobis feugiat similique usu ex.
<br><br>
Eum hinc argumentum te, no sit percipit adversarium, ne qui puto feugiat persecuti. Odio omnes scripserit ad est, ut vidit lorem maiestatis his, putent mandamus gloriatur ne pro. Oratio iriure rationibus ne his, ad est corrumpit splendide. Ad duo appareat moderatius, ei falli tollit denique eos. Dicant evertitur mei in, ne his deserunt perpetua sententiae, ea sea omnes similique vituperatoribus. Ex mel errem intellegebat comprehensam, vel ad tantas antiopam delicatissimi, tota ferri affert eu nec. Legere expetenda pertinacia ne pro, et pro impetus persius assueverit.
<br><br>
Ea mei nullam facete, omnis oratio offendit ius cu. Doming takimata repudiandae usu an, mei dicant takimata id, pri eleifend inimicus euripidis at. His vero singulis ea, quem euripidis abhorreant mei ut, et populo iriure vix. Usu ludus affert voluptaria ei, vix ea error definitiones, movet fastidii signiferumque in qui.
<br><br>
Vis prodesset adolescens adipiscing te, usu mazim perfecto recteque at, assum putant erroribus mea in. Vel facete imperdiet id, cum an libris luptatum perfecto, vel fabellas inciderint ut. Veri facete debitis ea vis, ut eos oratio erroribus. Sint facete perfecto no vel, vim id omnium insolens. Vel dolores perfecto pertinacia ut, te mel meis ullum dicam, eos assum facilis corpora in.
<br><br>
Mea te unum viderer dolores, nostrum detracto nec in, vis no partem definiebas constituam. Dicant utinam philosophia has cu, hendrerit prodesset at nam, eos an bonorum dissentiet. Has ad placerat intellegam consectetuer, no adipisci mandamus senserit pro, torquatos similique percipitur est ex. Pro ex putant deleniti repudiare, vel an aperiam sensibus suavitate. Ad vel epicurei convenire, ea soluta aliquid deserunt ius, pri in errem putant feugiat.
<br><br>
Sed iusto nihil populo an, ex pro novum homero cotidieque. Te utamur civibus eleifend qui, nam ei brute doming concludaturque, modo aliquam facilisi nec no. Vidisse maiestatis constituam eu his, esse pertinacia intellegam ius cu. Eos ei odio veniam, eu sumo altera adipisci eam, mea audiam prodesset persequeris ea. Ad vitae dictas vituperata sed, eum posse labore postulant id. Te eligendi principes dignissim sit, te vel dicant officiis repudiandae.
<br><br>
Id vel sensibus honestatis omittantur, vel cu nobis commune patrioque. In accusata definiebas qui, id tale malorum dolorem sed, solum clita phaedrum ne his. Eos mutat ullum forensibus ex, wisi perfecto urbanitas cu eam, no vis dicunt impetus. Assum novum in pri, vix an suavitate moderatius, id has reformidans referrentur. Elit inciderint omittantur duo ut, dicit democritum signiferumque eu est, ad suscipit delectus mandamus duo. An harum equidem maiestatis nec.
At has veri feugait placerat, in semper offendit praesent his. Omnium impetus facilis sed at, ex viris tincidunt ius. Unum eirmod dignissim id quo. Sit te atomorum quaerendum neglegentur, his primis tamquam et. Eu quo quot veri alienum, ea eos nullam luptatum accusamus. Ea mel causae phaedrum reprimique, at vidisse dolores ocurreret nam.
```

```
<br><br>

<br><br>
  <p id="Pie">
<hr noshade size="5" width="50%" align="center"><br>
<FONT SIZE="-1"><p align="center" position:absolute; bottom:0>Autor: Juan Manuel Rincon.</p></FONT><BR>

</div>

</body>

</html>
```

8.3.15 STYLE.CSS

```
body {background-color:  #DEB887;
     text-indent: 1cm

   margin: 0 auto;



}


body.indice {background-color: #F4A460;
          vertical-align: middle;
          text-align: justify;
}


b {
   font-size: xx-small;
   font-weight: bold;

}
```

9 ANEXO II: REFERENCIAS Y BIBLIOGRAFÍA

http://doc.ubuntu-es.org/HTTPD_Servidor_web_Apache2

http://www.wikipedia.es

http://www.linux-magazine.es

http://www.tiobe.com

http://www.anexom.es/tecnologia/diseno-web/las-novedades-de-html5-i/

Trabajo sobre tendencias artísticas de: Ana Mariela Leibovich, Daniela Sciascio, Eleonora Superville.

http://david.grajal.net/web/cw1_hojas_de_estilo_dinamicas.html

http://javaweb.osmosislatina.com/

http://www.gestiopolis.com/canales/economia/articulos/37/tendensocint1.htm

http://gerenciaactual.blogspot.com/2007/12/tendencias-sociales-en-internet-y-su.html

10 ANEXO III: DEFINICIONES Y ACRÓNIMOS

ANSI: American National Standards Institute

Apache Software Foundation: Organización no lucrativa creada para dar soporte a los proyectos de software bajo la denominación *Apache.*

API: Application Programming Interface

APK: Android Package

BBDD: Bases de Datos

CSS: Cascading Style Sheets

DBMS: Database Management System

DDL: Data Definition Language

DML: Data Manipulation Language

DOM: Document Object Model

DTD: Document Type Definition

ESN: External Social Network

HTML: HyperText Markup Language

HTTP: HyperText Transfer Protocol

IDE: Integrated Development Environment

IETF: Internet Engineering Task Force

ISN: Internal Social Network

ISO: International Organization for Standardization

JVM: Java Virtual Machine

JNI: Java Native Interface

LAMP: Linux, Apache, MySQL, PHP/Python

PHP: PHP Hypertext Preprocessor

S.O.: Sistema Operativo

SAX: Simple API for XML

SGML: Standard Generalized Markup Language

SQL: Structured Query Language

TCP: Transmission Control Protocol

Ubuntu: Ubuntu es un sistema operativo mantenido por Canonical y la comunidad de desarrolladores. Utiliza un núcleo Linux, y su origen está basado en Debian.

UI: User Interface

URI: Uniform Resource Identifier

W3C: World Wide Web Consortium

XML: eXtensible Markup Language

XMPP: eXtensible Messaging and Presence Protocol