GOBIERNO DE ESPAÑA
MINISTERIO DE INDUSTRIA, TURISMO Y COMERCIO
plan avanza2
INTECO
Instituto Nacional de Tecnologías de la Comunicación

GUÍA DE ESTÁNDARES WEB

Centro de Referencia en Accesibilidad y Estándares Web

Abril 2010

ÍNDICE

1. OBJETIVO DE LA GUÍA

La consecución de la Universalidad se establece como uno de los principales objetivos de la Web actual, resultando fundamental para ello garantizar el cumplimiento de los Estándares Web definidos por los organismos oficiales.

Un sitio Web basado en Estándares, entre otras cosas, facilitará su acceso por parte de cualquier usuario y a través de un amplio número de dispositivos, tendrá un mejor rendimiento al contar con un código más limpio, posibilitará un mantenimiento más sencillo y mejorará la búsqueda de sus contenidos gracias a un mayor aporte semántico.

El objeto de la presente guía es el de ofrecer a los desarrolladores una visión general de los principales **Estándares** existentes en el entorno Web, analizando las características y el funcionamiento de cada uno de ellos, las ventajas que se derivan de su aplicación, su evolución histórica y su tendencia.

2. ESTÁNDARES WEB

2.1. ¿QUÉ SON?

Un estándar puede definirse como un conjunto de reglas normalizadas que indican los requisitos a cumplir por todo producto, proceso o servicio, con el fin garantizar la compatibilidad entre los distintos elementos que lo utilicen.

Así, el *World Wide Web Consortium* (*W3C*) desarrolla **Estándares Web** o **Recomendaciones** que tienen por finalidad conseguir que las tecnologías que conforman la Web sean interoperables, eficientes, confiables, accesibles y fáciles de usar, lo que a su vez repercutirá en el desarrollo de aplicaciones cada vez más robustas.

Estas recomendaciones son el fruto de un proceso neutro, transparente y consensuado en el que toman parte los miembros del W3C (más de 400 organizaciones en la actualidad), su equipo de trabajo, expertos y aquellos usuarios de la Web que deseen colaborar.

Los Estándares Web han surgido de la necesidad de evitar la fragmentación de la Web así como de mejorar la organización de la información ofrecida en ella, y muchos de ellos han ido sentando las bases de su desarrollo y fomentando su éxito.

Algunos de los estándares Web más conocidos y ampliamente utilizados son el lenguaje de etiquetado para hacer páginas Web **HTML** (*HyperText Markup Language*), el lenguaje para crear estructuras de documentos **XML** (*eXtensible Markup Language*), y el lenguaje de hojas de estilos **CSS** (*Cascading Style Sheets*), que permiten controlar la presentación de los documentos (X)HTML.

2.2. ¿CÓMO FUNCIONAN?

El primer paso a la hora de crear un Estándar Web es llevar cabo un proceso inicial controlado en el que intervienen todos los usuarios de las tecnologías, con el fin de aportar conocimientos y opiniones que contribuyan a la mejora de los documentos.

A continuación, se obtienen unos estándares de calidad, los cuales pueden emplearse de forma libre en la comunidad Web al estar sujetos a la Política de Patentes del W3C, mientras que las especificaciones sufren un proceso de refinamiento exhaustivo antes de que se consideren *Recomendaciones*. Al utilizar las mismas tecnologías, las máquinas se entienden entre sí y cualquier usuario puede interactuar con el resto.

Para aquellos usuarios que, utilizando las *Recomendaciones* del W3C, deseen conocer si están aplicando adecuadamente o no las especificaciones, el propio Consorcio ofrece

herramientas (manuales de directivas o buenas prácticas, validadores sintácticos de lenguajes, etc)

2.3. ¿CUÁLES SON LAS VENTAJAS DE SU EMPLEO?

A continuación se indican las principales ventajas que conlleva la **aplicación de Estándares en el desarrollo de un sitio Web**:

- **Código más sencillo**: Un código limpio, válido, modular y semánticamente correcto facilita su comprensión y reutilización por parte de cualquier desarrollador, ayudando asimismo a que las aplicaciones puedan convertirlo de forma sencilla a otro formato.

- **Compatibilidad**: Los Estándares Web garantizan la compatibilidad del código independientemente del navegador o plataforma empleado. Además, se consigue una mayor estabilidad del sitio Web de cara al futuro y a la aparición de nuevas herramientas.

- **Mejora de la accesibilidad**: Los Estándares Web ayudan a hacer el contenido de un sitio Web accesible a un mayor número de usuarios, independientemente del idioma, localización geográfica, cultura, limitación técnica, física, psíquica o sensorial de éstos, cumpliéndose las directrices y sin que se sacrifique el aspecto visual o el rendimiento del mismo.

- **Mejora del posicionamiento**: Los sitios Web desarrollados en base a Estándares tendrán una mejor posición en los motores de búsqueda. En el caso de que se emplee un código complejo, los robots de búsqueda localizarán e indexarán los contenidos con más dificultad.

- **Mejor adaptación al dispositivo final**: El empleo de Estándares permite que la información sea interpretada por diferentes tipos de dispositivo (navegadores visuales y sólo textos, lectores de pantalla, lectores Braille, dispositivos móviles, etc).

- **Mejor adaptación al usuario**: El usuario puede ajustar la presentación del sitio según sus preferencias o necesidades.

- **Mejora en la impresión**: A través de los Estándares se proporciona de una forma sencilla versiones para imprimir de todas las páginas Web.

- **Mejora del mantenimiento**: La separación de contenido y presentación mediante el empleo de hojas de etilo CSS facilita futuros cambios. Así, resulta más sencillo efectuar

modificaciones en un único documento (CSS) que en todas las páginas (documentos (X)HTML) en las que se hayan incluidos estilos.

- **Ahorro de ancho de banda y carga de páginas más rápida**: Los sitios basados en Estándares hacen uso de un menor ancho de banda, lo cual implica a su vez un ahorro en los gastos de alojamiento Web. Por otra parte, la adecuación gramatical de las páginas de un sitio, contribuye a que se muestren más rápido a los usuarios, lo que mejora la experiencia de éstos.

- **Mayor confianza en la Web**: La Web es un medio colaborativo, donde los usuarios interactúan y se relacionan, siendo necesaria la confianza entre sí. Para ello, se han desarrollado tecnologías como las firmas digitales de documentos, la encriptación de datos confidenciales o las políticas de privacidad de datos de los sitios Web.

- **Mayor carga semántica**: Se proporcionan mecanismos para añadir significado a los recursos, haciendo posible que una máquina pueda interpretar los datos de la Web de forma análoga a como lo hacen los seres humanos. De este modo, también se consigue una mejora del rendimiento y eficiencia de la Web, beneficiando a los usuarios a través de una mayor precisión en sus búsquedas y operaciones.

- **Competitividad**: La aplicación de Estándares aporta una mayor ventaja competitiva en el mercado.

3. CATEGORIZACIÓN DE LOS ESTÁNDARES WEB

3.1. XML

3.1.1. ¿Qué es?

XML es una especificación de carácter genérico derivada del Estándar SGML (*Standard Generalized Markup Language*) que permite definir lenguajes de marcado. Es lo que se denomina un metalenguaje: no se usa directamente, sino que sirve para definir otros lenguajes. Su importancia reside en su capacidad para expresar el significado de un contenido con independencia del formato de documento final que se presente al usuario gracias a una serie de etiquetas.

Un documento XML puede ser procesado por un sistema automático o transformado en un formato adaptado al usuario. No se ve limitado por las características o capacidades del usuario ni la forma de presentación. Por su flexibilidad, XML es aplicable a una gran diversidad de campos como pueden ser el intercambio de mensajes de datos entre diferentes sistemas, dibujos vectoriales, correo por voz, subtítulos para multimedia, fórmulas matemáticas, partituras de música, y páginas Web.

Por ser un formato estandarizado existe un gran conocimiento y mucha experiencia en su uso. Existen numerosas herramientas para el procesamiento y transformación de XML desde editores de etiquetas hasta aplicaciones especializadas para el dominio de una aplicación concreta como podría ser un programa de dibujo. Todos trabajan con el mismo formato subyacente.

XML se establece como una tecnología que se rodea de un conjunto de tecnologías paralelas que la complementan, entre las cuales caben destacar las siguientes:

- **XSL**: Familia de lenguajes basados en el estándar XML (*XSLT*, *XSL-FO* y *XPath*) que permite definir una presentación o formato para un documento XML.
    - **XSLT**: lenguaje empleado para transformar la información en el formato final más apropiado para el usuario.
    - **XSL-FO**: lenguaje que permite describir la forma en que se presentan los componentes de un documento XML.
    - **XPath**: Lenguaje que permite identificar de forma inequívoca cualquier elemento o atributo de un documento XML.
- **XLink**: lenguaje creado para poder definir de forma estándar hipervínculos en archivos XML.
- **XPointer** y **XFragments**: lenguajes para apuntar a partes de un archivo XML.

- **XQuery:** lenguaje de consulta similar a SQL para colecciones de datos XML.
- **XSchema**: lenguaje de esquema empleado para describir la estructura y contenido adecuados de los elementos incluidos en los documentos XML.
- **CSS**: lenguaje de hojas de estilos que permite controlar la presentación de documentos (X)HTML y XML.

A continuación se detallan las principales características del lenguaje XML:

- XML permite guardar la información en un formato independiente del documento final que recibe el usuario.
- Posee una estructura sencilla que facilita su comprensión, aprendizaje y empleo.
- Posee una arquitectura abierta y extensible a través de la definición de nuevas etiquetas, lo que garantiza su correcto funcionamiento bajo cualquier tipo de navegador (antiguo, presente o futuro).
- XML se establece como Estándar para el intercambio de información estructurada entre diferentes aplicaciones y plataformas de un modo sencillo, seguro y fiable.
- Se trata de un lenguaje flexible que agrupa un amplio abanico de aplicaciones (páginas Web, bases de datos, etc).
- XML marca la semántica o significado de cada elemento: sea una persona, un código de barras, o un círculo, describiendo las relaciones entre los elementos.
- Se encuentra estructurado, lo que permite el modelado de datos de diferentes niveles de complejidad y facilita su procesamiento.
- Los documentos XML pueden ser validados contra una DTD.
- Mediante el empleo de XML se obtiene un comportamiento más estable y actualizable de las aplicaciones Web.
- Los documentos XML proporcionan metainformación sobre sí mismos, lo que repercutirá en búsquedas más precisas.
- El análisis de un documento XML es un proceso estandarizado, lo que permite utilizar cualquier analizador, evitando de este modo errores y optimizando el desarrollo de aplicaciones.

- Es independiente del medio y plataforma empleados, lo que permite la publicación de contenidos en diversos formatos así como la utilización de cualquier herramienta estándar.

- Como toda tecnología W3C, XML es gratuito.

3.1.2. Principales diferencias entre XML y HTML

XML y HTML son lenguajes diferentes. Así, ambos derivan del SGML, estándar internacional para la definición de la estructura y el contenido de diferentes tipos de documentos electrónicos, si bien, **HTML** es un subconjunto de SGML creado para **mostrar información y dar formato legible** a los contenidos de las páginas Web, mientras que **XML** deriva igualmente del lenguaje SGML pero está orientado a **describir, almacenar e intercambiar datos** en la Web.

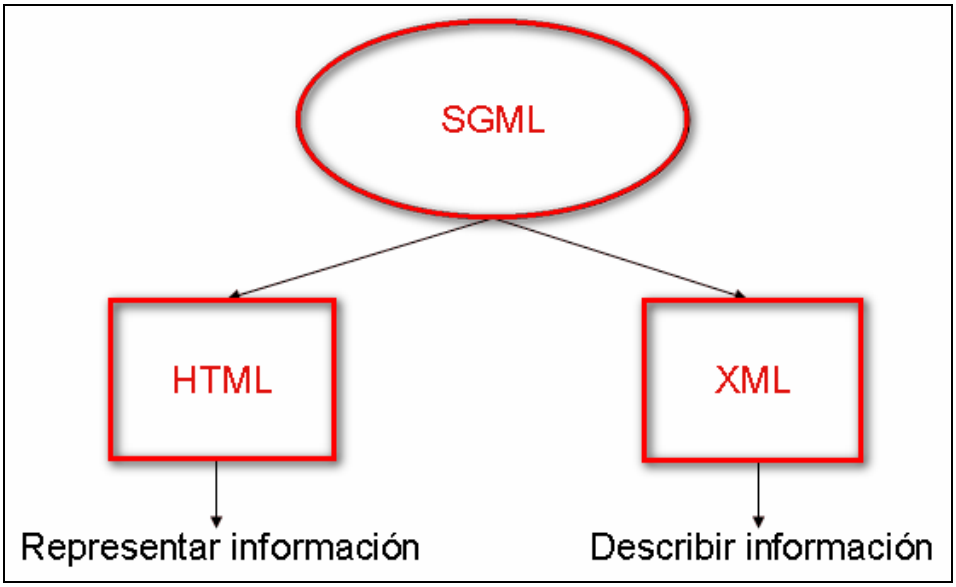


Figura 1. Diferencias entre HTML y XML

XML es un metalenguaje. Mientras el estándar HTML define un conjunto fijo de elementos, el estándar XML no define nombres específicos y permite una gran flexibilidad en la elección de los mismos. Sin embargo mientras HTML es tolerante frente a incorrecciones del lenguaje, XML es muy estricto, incluyendo reglas sobre el anidamiento de los elementos, el cierre de las etiquetas, el uso de mayúsculas y minúsculas, etc. Un documento que cumple con estas reglas decimos que está bien formado. Mientras que en el caso del HTML un navegador intentará procesar un documento con codificación HTML aunque esté mal formado, un documento XML será rechazado si no está formado correctamente. Este rigor ha permitido la creación de sistemas capaces de procesar cualquier documento XML incluso sin conocer su definición de antemano.

Aunque XML permite la definición de elementos para satisfacer las necesidades de cualquier campo de aplicación, en la práctica un sistema reconocerá un vocabulario que agrupe elementos vinculados a su propio ámbito. Cada vocabulario se convierte por tanto en una aplicación de XML.

A diferencia de HTML, es imprescindible que la estructura del documento XML sea correcta. Así, si el documento contiene errores, una aplicación debe rechazarlo. Para comprobar el archivo es suficiente abrirlo en un navegador Web reciente, de forma que si el documento contiene errores se visualizará un mensaje de error, mientras que si el documento tiene una estructura correcta se mostrará el documento original.

3.1.3. VoiceXML

VoiceXML es un lenguaje para definir **aplicaciones de voz**. De la misma forma que un navegador Web convencional presenta los documentos HTML de forma visual, un interprete VoiceXML presenta los documentos VoiceXML de forma auditiva. El interprete VoiceXML suele ser algo parecido a un navegador.

El lenguaje HTML está orientado a la creación de documentos estáticos que posteriormente pueden hacerse dinámicos e interactivos con formularios. VoiceXML ha sido diseñado en primer lugar para la creación de diálogos interactivos mediante voz sintetizada, sonido digitalizado, reconocimiento del habla del usuario y reconocimiento de teclas mediante sus tonos, grabación de la información hablada, y telefonía. Su objetivo es proporcionar a las aplicaciones interactivas de respuesta vocal las ventajas del paradigma aplicado en el entorno Web.

De la misma forma que se hace con los documentos HTML, se pueden localizar los documentos VoiceXML mediante URIs y éstos pueden ser alojados en cualquier servidor Web. A diferencia de un navegador Web convencional, que se ejecuta en el ordenador del usuario, el interpretador VoiceXML se suele ubicar en un **servidor remoto** (aunque existen sistemas de navegación visual o de navegación GPS con interacción por voz).

3.2. LENGUAJES ESTRUCTURALES

3.2.1. Los orígenes del HTML

HTML (*HyperText Markup Language*) es el lenguaje de marcado empleado universalmente para **crear páginas Web**. Se trata de un lenguaje de hipertexto constituido por un conjunto de **etiquetas** que marcan la apertura y el cierre de cada elemento, mediante el cual es posible incluir de forma **estructurada** textos, imágenes, objetos programados y scripts.

El hecho de que HTML sea un Estándar del W3C, permite que cualquier página Web creada a través de dicho lenguaje pueda ser visualizada de forma homogénea, con independencia del navegador o plataforma empleados (siempre que estos sean fieles a los estándares).

Así, los orígenes del HTML se remontan a 1980, año en el que Tim Berners-Lee, trabajador del CERN (*European Laboratory for Particle Physics*), comienza a elaborar un sistema de hipertexto para Internet, no siendo hasta el 1990 cuando definiera el lenguaje HTML como un subconjunto del poderoso lenguaje de etiquetado SGML. En 1991, Tim Berners-Lee publica la primera descripción formal de HTML, conocida como **HTML Tags**, en la que se recogen los 22 primeros elementos del lenguaje.

En 1993, el organismo IETF (*Intenet Engineering Task Force*) elabora una propuesta para estandarizar **HTML**. Si bien, no se llega a establecer como Estándar ninguna de las dos propuestas existentes en el momento (HTML y HTML+). Será el 22 de Septiembre del año 1995 cuando el IETF logre publicar el Estándar **HTML 2.0** como primer Estándar oficial de HTML, creado con fines divulgativos y académicos, y donde prevalecía el contenido por encima del diseño.

Con todo, HTML 2.0 no permitía controlar el diseño de las páginas ni añadir elementos multimedia, a lo que la empresa **Netscape** responde definiendo nuevas etiquetas en el Estándar. Por otro lado, el consorcio internacional W3C, creado en Marzo de 1995, comenzó a desarrollar un borrador para la versión **HTML 3.0**, no siendo bien acogido debido al elevado número de elementos y atributos que se definieron en él, lo que le hacia muy complejo para poder desarrollarse mediante la tecnología del momento y finalmente fue abandonado.

El 14 de Enero de 1997 es la fecha elegida por el W3C para publicar **HTML 3.2**, que es oficialmente su primera recomendación. En ella se abandonan muchas de las características de HTML 3.0 y se incluyen los últimos avances desarrollados por los navegadores *Internet Explorer* y *Netscape Navigator*, como por ejemplo los applets de Java o el texto flotado.

Sin embargo, el avance más notorio se observa en **HTML 4.0**, recomendación publicada por el W3C el 18 de Diciembre de 1997 y revisada el 24 de Abril de 1998. Mediante esta versión de HTML se pretende dar soporte a marcos, hojas de estilo CSS, scripts y tablas complejas. También se introducen mejoras en los formularios y en la accesibilidad general de las páginas, así como a nivel de código, especificándose un conjunto de elementos desaprobados y obsoletos. Posteriormente, esta versión sufre una revisión que da lugar a **HTML 4.01**, la última especificación oficial de HTML, publicada el 24 de Diciembre de 1999 y que es muy similar a su antecesora.

A partir de este momento, el W3C deja aparcado el desarrollo del Estándar HTML para centrarse en una nueva vía, el **XHTML** (*eXtensible HyperText Markup Language*), una versión más estricta y limpia de HTML preparada para su uso con herramientas basadas en XML. Este cambio de rumbo motiva, de la mano de integrantes de **Mozilla Foundation**, **Opera Software** y **Apple**, la creación en el año 2004 de la asociación **WHATWG** (*Web Hypertext Application Technology Working Group*), cuyo objetivo es implementar el nuevo Estándar **HTML 5**, del que ya existe un borrador desde el 22 de Enero de 2008.

Este nuevo contexto hace que el W3C retome el desarrollo de HTML en Marzo del 2007, si bien se trabaja de forma simultánea en la implementación de XHTML, publicándose su

primera recomendación, el **XHTML 1.0** el 26 de Enero de 2000, y su segunda recomendación, el XHTML 1.1 el 31 de Mayo de 2001, que es una versión modularizada de XHTML 1.0. Por último, cabe destacar que también existe un borrador de la novedosa especificación **XHTML 2.0**, la cual data del 26 de Julio del 2006.

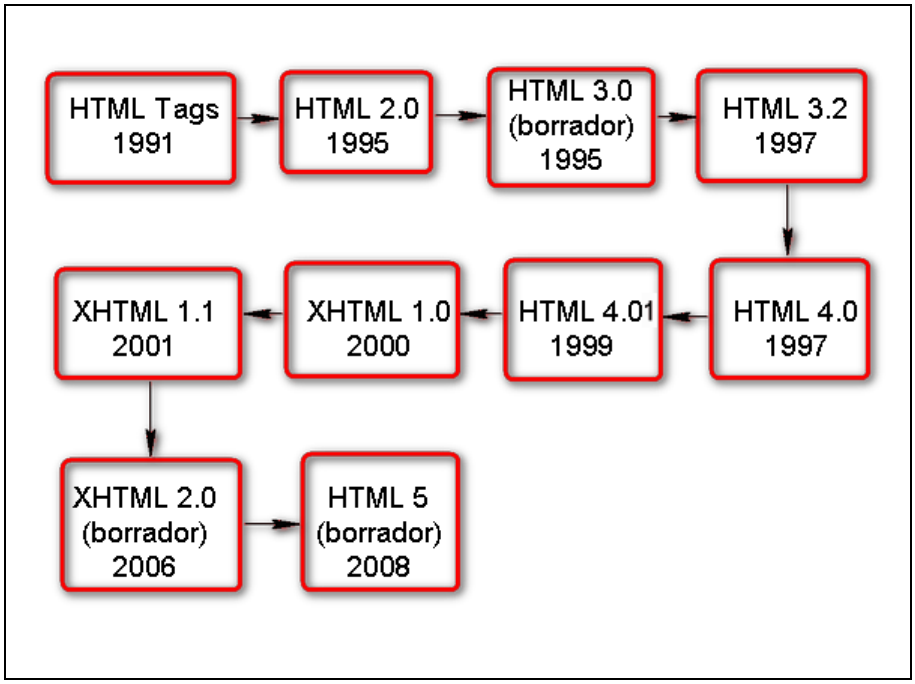


Figura 2. Recorrido histórico del Estándar HTML

3.2.2. HTML 4.01

HTML 4.01 constituye la última especificación de HTML publicada oficialmente por el W3C. Se trata de una revisión de HTML 4.0 con la que se pretende introducir mejoras respecto a versiones anteriores del lenguaje, dando para ello soporte a scripts, hojas de estilo, capacidades de impresión y opciones multimedia, mejorando asimismo la accesibilidad y e internacionalización de los documentos.

Una página HTML se compone principalmente de dos secciones: **cabecera** y **cuerpo**.

La cabecera, delimitada por las etiquetas `<HEAD>` y `</HEAD>`, contiene información sobre la propia página, como son el título, metadatos (palabras clave, descripción, autor…), referencias a hojas de estilo externas y a feeds de RSS, etc. Esta información no será visible para el usuario, salvo el título, el cual se muestra en la parte superior izquierda de las ventanas de los navegadores.

Por otro lado, el cuerpo encierra el contenido de la página que será visualizado por el usuario (imágenes, textos, formularios, tablas, objetos programados, etc), encontrándose definido entre las etiquetas `<BODY>` y `</BODY>`.

Dichas secciones se engloban dentro de las etiquetas `<HTML>` y `</HTML>`, destinadas a identificar respectivamente el principio y el fin de toda página HTML. Notar que no puede ser colocada ninguna otra etiqueta antes o después de dichas etiquetas ya que serán ignoradas, con la excepción de la **declaración de tipo DOCTYPE** que es el primer elemento que debe constar en el código de la página.

Ejemplo de código:

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

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

<HTML>

<HEAD>

<TITLE>Título de la página</TITLE>

<!-- Otra datos de la página -->

</HEAD>

<BODY>

<!-- Contenido de la página -->

</BODY>

</HTML>
```

Figura 3. Estructura de un documento HTML

Los componentes estructurales sobre los que se fundamenta HTML son los **elementos** y los **atributos**. Así, los elementos de HTML representan **estructuras de información variadas**: imágenes, párrafos de texto, listas, tablas, formularios, enlaces, objetos programados, etc. Es importante diferenciar un elemento de una etiqueta, ya que son conceptos distintos y resulta habitual su confusión. Así, un elemento HTML consta de dos etiquetas, una de apertura (`<ELEMENTO>`) y otra de cierre (`</ELEMENTO>`), además del propio contenido del elemento (`<ELEMENTO>contenido</ELEMENTO>`).

En HTML 4.01 existen elementos para los que es posible **prescindir de la etiqueta de cierre**, como por ejemplo, los elementos `P` y `LI`, así como elementos que permiten **omitir la etiqueta de apertura** (`HEAD` y `BODY`). También se incluyen elementos que carecen de contenido, conocidos comúnmente como **elementos vacíos**, lo cuales no tienen etiquetas finales.

Para un elemento de HTML es posible definir **propiedades asociadas (atributos)** dentro de su etiqueta de apertura. Se pueden especificar varios atributos para un mismo elemento, de forma que a cada atributo se le asigna un valor que pude ser indicado entre comillas dobles o comillas simples.

Se ha de tener en cuenta que la especificación HTML 4.01 no es sensible al empleo de mayúsculas y minúsculas, por lo que resulta completamente válido indicar el nombre de los elementos y los atributos en mayúscula (lo cual no es permitido en XHTML 1.0 como se puede apreciar en la sección *Diferencias entre XHTML 1.0 y HTML 4.01* de la presente guía).

Asimismo, en HTML 4.01 se establece un conjunto de **elementos** y **atributos desaconsejados** (aquellos que han quedado desactualizados):

- **Elementos**: `APPLET`, `BASEFONT`, `CENTER`, `DIR`, `FONT`, `ISINDEX`, `MENU`, `S`, `STRIKE`, `U`.
- **Atributos**: *`start`*, *`value`*, *`align`*, *`valign`*, *`clear`*, *`nowrap`*, *`hspace`*, *`vspace`*, *`compact`*, *`face`*, *`size`*, *`background`*, *`bgcolor`*, *`color`*, *`text`*, *`link`*, *`alink`*, *`vlink`*, *`noshade`*.

(Puede obtener más información sobre la declaración DOCTYPE en la guía de comprobación de accesibilidad relativa a Legibilidad y Metainformación disponible en la sección Accesibilidad > Guías de Comprobación de la página http://www.inteco.es)

3.2.3. XHTML 1.0

XHTML (*Extensible Hypertext Markup Language*) es un conjunto de documentos basados en XML que nace como **extensión** del HTML 4.0 y que permite la **interoperabilidad** entre aplicaciones basadas en XML. Se trata de una paso adelante en el campo del desarrollo Web, al aprovecharse las ventajas del XML, manteniendo la compatibilidad hacia atrás y hacia delante con las distintas aplicaciones de usuario.

La primera especificación formal de XHTML es **XHTML 1.0**, que se establece como una reformulación de HTML 4.0 conforme a XML y compatible con aplicaciones de usuario desarrolladas para HTML 4.0.

Su empleo proporciona una serie de ventajas:

- Basado en XML, lo que permite un sencillo manejo mediante herramientas XML Estándar.
- Capacidad de interacción creciente entre distintos contextos XHTML.
- Compatibilidad con aplicaciones de usuario conformes a HTML 4.0 y XHTML 4.0, obteniéndose un comportamiento idéntico en ambos tipos de aplicación.

- Posibilidad de crear contenidos Web y aplicaciones de usuario mediante técnicas de desarrollo modular. En este sentido, XHTML 1.0 se extiende y acota a través de un mecanismo que permite definir módulos (conjuntos de elementos) y combinarlos entre sí en función de las necesidades de cada ámbito.

- Mejora de la transformación del contenido.

3.2.4. Diferencias entre XHTML 1.0 y HTML 4.01

XHTML 1.0 es un lenguaje basado en XML, mientras que HTML 4.01 proviene del SGML, lo que provoca que existan **diferencias** entre ambos. A continuación se detallan algunas de las reglas de XHTML 1.0 que lo diferencian de HTML 4.01:

- Los **elementos** deben estar **adecuadamente anidados**, con el objeto de garantizar la corrección gramatical de los documentos.

  Ejemplo de código correcto:

```
<p>Párrafo de <em>prueba</em>.</p>
```

  Ejemplo de código incorrecto:

```
<p>Párrafo de <em>prueba</p>.</em>
```

- XHTML es sensible al empleo de minúsculas y mayúsculas, por lo que los nombres de **elementos** y **atributos** deben escribirse **en minúscula**:

  Ejemplo de código correcto:

```
<p>Párrafo de prueba</p>
```

  Ejemplo de código incorrecto:

```
<P>Párrafo de prueba</P>
```

- Los **elementos no vacíos** deben tener **etiqueta de cierre**.

  Ejemplo de código correcto:

```
<p>Párrafo de prueba 1</p><p>Párrafo de prueba 2</p>
```

  Ejemplo de código incorrecto:

```
<p>Párrafo de prueba 1<p>Párrafo de prueba 2</p>
```

- El **valor** de cualquier **atributo** debe ir **entrecomillado**.

  Ejemplo de código correcto:

```
<div id=”cabecera”>
…
</div>
```

  Ejemplo de código incorrecto:

```
<div id=cabecera>
…
</div>
```

- Para garantizar la compatibilidad de XHTML servido como HTML, los **elementos vacíos** se deben cerrar incluyendo un **espacio** y el **carácter “/”** al final de su declaración.

  Ejemplo de código correcto:

```
<input id=”prueba” name=”prueba” type=”text” />
```

  Ejemplo de código incorrecto:

```
<input id=” prueba” name=” prueba” type=”text”>
```

- No se permite definir **pares atributo-valor** de forma minimizada.

  Ejemplo de código correcto:

```
<input type=”tipo” checked=”checked” />
```

  Ejemplo de código incorrecto:

```
<input type=”tipo” checked />
```

- Se debe englobar el contenido de los elementos `SCRIPT` y `STYLE` dentro de una **sección CDATA**, con el fin de ignorar los caracteres que incluyen para evitar problemas con entidades como < y &. Otra posibilidad es la de incluir el código de estos elementos en **archivos externos** y enlazarlos desde la propia página.

Ejemplo de código:

```
<script>
 <![CDATA[
  …
 ]]>
 </script>
```

- No se permiten las **exclusiones de SGML**, es decir, no es posible prohibir el anidamiento de unos elementos específicos en otros.

- Con el fin de obtener documentos bien estructurados, **se prohíbe el empleo** del atributo *name* para identificar fragmentos de información (con la excepción de su uso en controles de formulario), debiéndose utilizar en su lugar el atributo *id*.

3.3. LENGUAJES DE PRESENTACIÓN

3.3.1. Perspectiva histórica

Las hojas de estilo nacen alrededor del año1970, como respuesta a la necesidad de definir un mecanismo que permitiera dotar de presentación a los documentos electrónicos. El auge de las hojas de estilo se produce con la llegada de Internet y la adopción de lenguaje HTML para el desarrollo de páginas Web.

El W3C planteó la creación de un lenguaje de hojas de estilo específico para el lenguaje HTML, presentándose nueve propuestas, de las cuales sólo dos se tuvieron finalmente en cuenta: la **CHSS** (*Cascading HTML Style Sheets*) de Hakon Wium Lie y la **SSP** (*Stream-based Style Sheet Proposal*) de Bert Bos. Lie y Bos se unen entre 1994 y 1995 para definir un nuevo lenguaje que tomaba lo mejor de cada propuesta, llamándolo CSS (*Cascading Style Sheets*).

En 1995, el W3C estandariza el lenguaje CSS y lo añade a su grupo de trabajo de HTML, siendo en Diciembre de 1996 cuando aprobara la primera recomendación oficial del mismo, conocida como **CSS nivel 1**.

El grupo de trabajo de CSS del W3C publica la segunda recomendación el 12 de Mayo de 1998, conocida como **CSS nivel 2**, que es compatible con CSS nivel 1 e incluye novedades como el posicionamiento y los estilos en tablas, permitiendo asimismo el empleo de hojas de estilo adaptadas al medio de presentación.

En el momento de elaboración de la presente guía, la recomendación de CSS que se emplea es la **CSS 2.1**, publicada inicialmente en el año 2008 y que aún se está revisando (en Septiembre del año 2009 se publicó una versión candidata a recomendación que aún no está aprobada). CSS 2.1 es una evolución de la recomendación CSS nivel 2, con la que se

corrigen algunos errores anteriores y se incluyen sólo aquellos puntos que se han tenido en cuenta en los navegadores.

La siguiente recomendación será **CSS nivel 3**, planteada en el año 1998 y que aún sigue en desarrollo y no se encuentra oficialmente aprobada, habiendo visto la luz varios borradores de la misma. En este caso, no se trata de una única recomendación, sino se han desarrollado varias recomendaciones para los diferentes aspectos o elementos que intervienen en ella.

3.3.2. Hojas de estilo CSS

Una hoja de estilo CSS (*Cascading Style Sheets*) es un lenguaje creado por el W3C que permite a los desarrolladores controlar los aspectos de estilo y formato de múltiples páginas (X)HTML simultáneamente, permitiendo separar contenido de presentación en todo momento.

Las hojas de estilo CSS están formadas por un conjunto de reglas que se aplican a páginas (X)HTML, donde cada regla se compone de un **selector** y de una **declaración**, ésta última con dos partes a su vez: la **propiedad** y el **valor asignado** a la misma.

Ejemplo de regla CSS:

```
p {font-size: 1.2em;}
```

Donde:

- `p` es el selector
- `font-size` es la propiedad
- `1.2em` es el valor de la propiedad

El efecto de presentación se define a través de la declaración, mientras que el selector será el componente de la regla que especifique el elemento al que afectará dicha declaración.

Existen diversas formas de proporcionar estilos a una página Web:

- **Hoja de estilo externa**: Hoja CSS que se enlaza a una página (X)HTML desde la cabecera de ésta (elemento `HEAD`) mediante el elemento `LINK`. Con esta técnica se consigue una perfecta separación de la presentación y la estructura de la página.
- **Estilos en línea**: Aplicando estilos en el contenido de la propia página (X)HTML sobre aquellos elementos que lo permitan, por medio del atributo *`style`*. El empleo de este método conlleva la mezcla de presentación y contenido.

- **Estilos embebidos**: Incluyendo en la página Web el código de presentación CSS deseado, utilizando para ello el elemento `STYLE`. Generalmente, estos estilos se definen en la cabecera (elemento `HEAD`).

Hay que tener en cuenta que la definición de estilos visuales en el interior de las páginas de un sitio Web dificulta su mantenimiento y la limpieza del código.

El empleo de hojas de estilo CSS es una práctica que comporta una serie de beneficios como son:

- Aumento de la legibilidad y reducción del peso de las páginas Web.
- Mejora del mantenimiento y actualización de los sitios Web.
- Mejora de la accesibilidad: se pueden definir hojas de estilo locales en función de las necesidades o preferencias del usuario.
- Versatilidad: Se ofrecen diferentes hojas de estilo para los diferentes tipos de medio existentes (sintetizadores de voz, dispositivos braille, dispositivos de mano, pantallas de computador a color, televisión, etc).

3.3.3. Soporte

Las hojas de estilo CSS han sufrido un largo proceso de adaptación en los distintos navegadores Web del mercado.

Así, el navegador *Internet Explorer 5* en su versión para Mac OS fue el primer navegador con soporte completo de CSS 1. A partir de su séptima versión, *Internet Explorer* ofrece soporte para CSS 2.1, abarcando todas las características de esta especificación en su octava versión.

Por su lado, *Opera*, *Google Chrome* y *Safari* ofrecen en la actualidad un amplio soporte para CSS 2.1, incluyendo además muchos elementos de CSS3, mientras que el navegador *Mozilla Firefox*, sin tener un soporte de CSS3 tan depurado, está recortando distancias con éstos en sus últimas versiones.

3.3.4. CSS Versión 2.1

CSS 2.1 es la especificación oficial del lenguaje de hojas de estilo que **se emplea en la actualidad** para dotar de presentación a las páginas (X)HTML y está destinada a sustituir a CSS2 (basada a su vez en CSS 1).

En la especificación CSS 2.1 se corrigen algunos errores de CSS2 y se añade una serie de características que ya han sido ampliamente aplicadas. Algunas características de CSS 2 no han sido modificadas en CSS 2.1, mientras que otras sí han experimentado cambios e incluso han sido eliminadas.

CSS 2.1 cuenta con soporte de hojas de estilo para medios específicos, con el fin de que los desarrolladores puedan adaptar la presentación de sus documentos a diferentes tipos de dispositivo (navegadores visuales, impresoras, dispositivos auditivos, dispositivos braille, dispositivos de mano, etc). También permite posicionar contenidos, diseñar tablas, incluir características de internacionalización y propiedades relacionadas con la interfaz de usuario.

Se puede decir que CSS 2.1 representa globalmente el empleo de todas las propiedades de CSS que se han desarrollado desde la fecha de publicación de la recomendación, superando la centena y agrupándose en diferentes tipos: de cajas, de texto y tipografía, de colores y fondo, de posicionamiento y visualización, de tablas, de listas, de contenido generado, de medios y otras.

Al igual que sus versiones anteriores, se basa en una serie de **principios de diseño**, que son los siguientes:

- **Accesibilidad**: En CSS 2.1 se incluyen propiedades que permitirán que las páginas Web sean más accesibles a usuarios con discapacidad.

- **Compatibilidad**: Las aplicaciones de usuario CSS 2.1 podrán interpretar hojas de estilo CSS 1, mientras que aplicaciones de usuario CSS 1 podrán leer las hojas de estilo CSS 2.1 y descartar aquellas partes que no comprendan.

- **Complementariedad**: Las hojas de estilo complementan los documentos estructurados proveyendo información de estilo del texto marcado, resultando sencillo cambiar la hoja de estilo con un bajo impacto en el marcado del documento.

- **Interoperabilidad**: Las propiedades de CSS 2.1 pueden ser empleadas conjuntamente con otros lenguajes.

- **Flexibilidad**: Las hojas de estilo pueden ser aplicadas al contenido de varias maneras (hojas CSS externas, estilos embebidos y estilos en línea).

- **Independencia**: Las hojas de estilo permiten que los documentos a los que se aplican sean independientes de plataforma y dispositivo, mientras que las propias hojas de estilo son independientes de plataforma pero en CSS 2.1 pueden ir dirigidas a diferentes grupos de dispositivos.

- **Mantenimiento**: Mediante el empleo de una hoja de estilos externa vinculada desde los documentos, se obtiene un estilo consistente y se simplifica el mantenimiento.

- **Rendimiento**: Las hojas de estilo CSS disminuyen el tamaño del contenido en la mayoría de las ocasiones, lo que se traduce en un mejor rendimiento de la red.

- **Riqueza**: CSS 2.1 proporciona a los desarrolladores un amplio abanico de efectos de presentación que pueden ser aplicados en las páginas Web, de forma independiente del dispositivo de acceso utilizado.

- **Simplicidad**: CSS es un lenguaje simple que puede ser leído y escrito por un humano.

3.4. ESPECIFICACIONES W3C EN BORRADOR

3.4.1. XHTML 2.0

XHTML 2.0 es un lenguaje de marcado para aplicaciones Web que está formado por un conjunto actualizado de módulos XHTML. Se trata de un lenguaje que permite generar contenidos para múltiples propósitos y dotarlos de una mayor carga semántica al conferir un mayor peso a la estructura que a la presentación. Así, los efectos de presentación sólo podrán ser definidos en las páginas Web mediante hojas de estilo, lo que permite obtener asimismo una mayor flexibilidad, accesibilidad e independencia de dispositivo.

A la hora de desarrollar XHTML 2.0 se tuvieron en cuenta los siguientes aspectos:

- Empleo de un código XML lo más genérico posible.

- Empleo de hojas de estilo para presentación, dando un mayor peso a la estructura que a la presentación de las páginas.

- Mayor nivel de accesibilidad.

- Mayor grado de internacionalización.

- Mayor independencia de dispositivo. Se trata de buscar un diseño que evite tener que desarrollar una versión diferente de una página para cada tipo de dispositivo en el que se visualizará (teléfonos, PDAs, tabletas, televisores, etc)

- Posibilidad de procesar páginas XHTML 2.0 mediante herramientas de Web semántica.

XHTML 2.0 ha sido diseñado de forma que resulta sencilla su interpretación para desarrolladores de HTML y XHTML 1.0, corrigiéndose errores de anteriores versiones e

introduciendo mejoras. Se trata de un lenguaje que es compatible con los navegadores antiguos, y salvo ciertas características como XForms y XML Events, también funciona con los nuevos navegadores.

A continuación se indican los **cambios más notables** que tienen lugar en XHTML 2.0:

- Se rompe la compatibilidad hacia atrás
- Validación XML obligatoria
- Desaparecen los elementos de presentación.
- Se incluyen nuevos elementos:
    - Listas de navegación: `NL` y `NAME`.
    - Secciones y encabezados: `SECTION` y `H`.
    - Saltos de línea: `LINE` en lugar de `BR`.
    - Líneas de separación: `SEPARATOR` en lugar de `HR`.
    - Citas: `QUOTE` en lugar de `Q`.
    - Formularios: `XFORMS` en lugar de `FORM`.
    - Marcos: `XFRAMES` en lugar de `FRAME`.
- Cambio de funcionalidad en algunos elemento y atributos:
    - El elemento `OBJECT` agrupa a los anteriores elementos `IMG` y `APPLET`
    - El atributo *`href`* puede ser definido en cualquier elemento.

Pese a que XHTML 2.0 posee un borrador formal que data del 26 de Julio de 2006, **no se ha llegado a consolidar como Estándar**, debido a que el grupo de trabajo de XHTML del W3C cesó su actividad en este lenguaje el 31 de diciembre de 2009, con el fin de acelerar el proceso de implementación de HTML 5.0 como Estándar definitivo.

3.4.2. HTML 5

HTML 5 es la quinta revisión del lenguaje de marcado de hipertexto del W3C. Se trata de una especificación que permite mejorar la interoperabilidad entre aplicaciones Web, ofreciendo nuevos elementos, fruto del estudio de las prácticas más habituales de los

desarrolladores Web, así como unos criterios de conformidad claros para los agentes de usuario.

Sus inicios se remontan al año 2004, cuando el WHATWG (*Web Hypertext Application Technology Working Group*) empezó a desarrollar el pliego de condiciones bajo el nombre de "*Aplicaciones Web 1.0*" y que finalmente se hizo público el 22 de Enero de 2008. El último borrador ha sido publicado recientemente, el 4 de Marzo de 2010, sin embargo, la posibilidad de que HTML 5 se convierta en recomendación oficial del W3C no se contempla a corto plazo.

HTML 5 facilita el trabajo a los desarrolladores mediante la definición de **dos tipos de sintaxis paralelas**: HTML 5, servida como *text/html* y XHTML 5, servida como *application/xhtml+xml*.

Además, presenta una serie de ventajas como son:

- Posibilidad de guardar aplicaciones Web y ejecutarlas localmente y de forma independientemente al sistema operativo empleado, cuando no se disponga de conexión a Internet.
- Posibilidad de definir el propio lenguaje con independencia de la sintaxis, al encontrarse especificado en términos del Modelo de Objetos del Documento.
- Ejecución más eficiente de aplicaciones y sitios Web con una carga elevada de código.
- Capacidad de determinar la localización geográfica de los usuarios.

3.4.2.1. Diferencias entre HTML 5, HTML 4.01 y XHTML 1.0

Una de las principales diferencias que presenta HTML 5 respecto a sus anteriores versiones es la existencia de un único tipo de declaración DOCTYPE: `<!doctype html>`.

Por otro lado, en HTML5 se incluyen **nuevos elementos** que permiten estructurar de forma más óptima una página Web, aportando un valor semántico a cada sección, lo cual repercute positivamente en la coherencia, facilita su interpretación. A continuación se indican dichos elementos de estructura:

- `ASIDE`: Contenido no relacionado directamente con el resto de la página.
- `ARTICLE`: Contenido independiente en una página.
- `FOOTER`: Pié de una sección.
- `HEADER`: Cabecera de una sección.

- `NAV`: Sección con elementos de navegación.
- `SECTION`: Sección general de una página.

También se han implementado **mejoras relativas a los formularios**, de forma que el elemento `INPUT` permite un mayor número de tipos como son: *color*, *date*, *email*, *number*, *range*, *search* o *url* entre otros.

Otros elementos novedosos que se proporcionan en HTML 5 son los siguientes:

- `CANVAS`: empleado conjuntamente con JavaScript, permite generar dinámicamente imágenes, gráficos, animaciones y transformaciones, lo que dota de gran interactividad en su diseño a las páginas Web.
- `COMMAND`: Se emplea para representar comandos que pueden ser ejecutados en un navegador Web.
- `DATAGRID`: Permite mostrar información tabular.
- `DATALIST`: Se emplea en formularios para crear controles de selección.
- `DETAILS`: Se utiliza para proporcionar información adicional sobre otro elemento.
- `DIALOG`: Permite representar conversaciones entre varias personas.
- `EMBED`: Se emplea en sustitución del elemento `OBJECT` para incrustar contenido ejecutado por plugins externos.
- `FIGURE`: Se emplea para identificar imágenes en el contenido.
- `KEYGEN`: Permite representar un control para la generación de un par de claves (una pública y otra privada).
- `MARK`: Se utiliza para crear un efecto de texto resaltado.
- `MENU:` Utilizado para ofrecer listas de comandos.
- `METER`: Se emplea para definir medidas.
- `OUTPUT`: Permite representar la salida de un programa.
- `PROGRESS`: Utilizado para representar el progreso, porcentaje o estado de un proceso.

- `RUBY`: Se emplea para especificar anotaciones Ruby.

- `TIME`: Permite representar una hora o fecha.

HTML 5 también contiene **nuevos atributos** como son: *`autofocus`*, *`inputmode`*, *`max`*, *`media`*, *`min`*, *`pattern`* o *`ping`*.

Por otra parte, **desaparecen** diversos **elementos** y **atributos** empleados en versiones anteriores:

- **Elementos**: `ACRONYM`, `APPLET`, `BASEFONT`, `BIG`, `CENTER`, `DIR`, `FONT`, `FRAME`, `FRAMESET`, `ISINDEX`, `NOFRAMES`, `S`, `STRIKE`, `TT`, `U`.

- **Atributos**: *`axis`*, *`abbr`* y *`header`* en `TH` y `TD`, *`archive`*, *`classid`*, *`codetype`* y *`standby`* en `OBJECT`, *`charset`* en `SCRIPT`, *`rev`* y *`charset`* en `A` y `LINK`, *`name`* en `MAP`, *`nohref`* en area, *`profile`* en `HEAD`, *`scheme`* en `META`, *`target`* en `LINK`, *`valuetype`* en `PARAM`, *`version`* en `HTML` y *`summary`*.

3.4.2.2. Multimedia

Un aspecto fundamental en HTML 5 es el tratamiento de elementos multimedia. En este sentido, HTML 5 permite **reproducir** e **incrustar** contenido multimedia de sonido y vídeo mediante los elementos `AUDIO` y `VIDEO` respectivamente, **sin la necesidad de instalar** ninguna aplicación, plugin o códec externos, lo cual supone un importante avance y un gran beneficio no sólo para los usuarios de Internet, sino también para los usuarios de dispositivos móviles, reduciendo la carga de sus sistemas operativos y eliminando cualquier incompatibilidad derivada del empleo de plugins (por ejemplo Flash).

La mayoría de los navegadores actuales (*Opera*, *Google Chrome*, *Mozilla Firefox* y *Safari*, entre otros) ya implementan el elemento `VIDEO`, permitiendo mostrar el contenido audiovisual. No obstante se plantea un problema, **la elección del formato de vídeo**. Cuando el W3C elaboró el borrador de HTML 5, estableció un formato de vídeo con *Ogg Theora*, un códec de vídeo libre, pero tras las protestas de algunas de sus organizaciones, interesadas en emplear códecs propietarios, no se concretó ningún codec definitivo para el elemento `VIDEO`.

Así, navegadores como *Mozilla Firefox* y *Opera* apuestan por el empleo de un códec libre (*Ogg Theora*), mientras que otros como *Google Chrome* o *Safari*, además de utilizar Ogg Theora,tambien emplean códecs de pago (*H.264/MPEG-4*). Por ejemplo, portales multimedia como *YouTube* o *Vimeo* emplean *H.264/MPEG-4* como códec, por lo que no será posible reproducir o incrustar sus contenidos multimedia en caso de emplear como navegador *Mozilla Firefox* u *Opera*. Sin embargo, el portal audiovisual *DailyMotion* hace uso del códec libre Ogg Theora.

3.4.2.3. Soporte

HTML 5 está suscitando un enorme interés en el ámbito Web, lo que está favoreciendo un esfuerzo por parte de los desarrolladores de navegadores para adaptarse a sus características. Así, las últimas versiones de navegadores como *Safari*, *Google Chrome*, *Opera* o *Mozilla Firefox*, son compatibles con HTML 5, mientras que el navegador *Internet Explorer* de Microsoft en su última versión aún no reconoce HTML 5, aunque se prevee que para su novena versión lo haga.

3.4.3. CSS Versión 3

La tercera versión del Estándar de hojas de estilo CSS nace con el objetivo de otorgar a los desarrolladores un **mayor control** sobre los elementos de las páginas Web, proporcionando nuevas características necesarias para conseguir aquellos efectos de presentación que no podían obtenerse mediante las versiones anteriores y que en ocasiones implicaban modificaciones en el propio contenido, rompiéndose la máxima de separar el contenido y presentación.

CSS 3 conserva muchas de las propiedades de CSS 2.1, si bien, contará con nuevos selectores de atributo y nuevas pseudo clases. Además, incorpora características renovadas en el apartado gráfico que permiten posicionar de forma más sencilla los elementos en la página.

Tampoco descuida el apartado auditivo contando con un módulo que permite crear hojas de estilo para definir el modo de reproducción de voz de un documento (X)HTML mediante un sintetizador, así como un módulo de audio para agregar sonidos de fondo o efectos de transición, controlar la posición del sonido que se está reproduciendo, etc.

Por otro lado, CSS 3 se estructura en un conjunto de **módulos**, lo cual se debe a la dificultad de manejo de CSS 2 a medida que éste ha ido creciendo. De este modo, el W3C puede implementar cada módulo por separado y a un ritmo independiente hasta alcanzar el estado de recomendación.

3.4.3.1. Novedades

CSS3 ofrece nuevas características que permite a los desarrolladores la aplicación de efectos estilísticos muy avanzados. A continuación se detallan las más relevantes:

- **Nuevos formatos de color**: Tales como *CMYK*, *HSL*, *HSLA* y *RGBA*. Notar que con los formatos *HSLA* y *RGBA* nace un nuevo concepto, las transparencias. Así, el parámetro A (o canal alpha) es utilizado para aplicar un grado de transparencia al color especificado para los elementos de las páginas.

- **Efectos en bordes**: CSS 3 hace posible el empleo de bordes con esquinas redondas, con un efecto degradado o con la incorporación de imágenes. Para ello se proporcionan respectivamente las propiedades `border-radius`, `border-color` y `border-image`.

- **Sombras**: A través de la propiedad `box-shadow` se puede definir un efecto de sombra a los elementos de la página, a través de los valores de desplazamiento horizontal y vertical de la sombra, difuminado y color de la sombra. Mientras que para aplicar sombra a los textos de las páginas se incluye la propiedad `text-shadow`.

- **Opacidad**: En CSS 3 se puede definir el grado de transparencia de cajas, textos e imágenes por medio de la propiedad `opacity`.

- **Múltiples imágenes de fondo**: La propiedad `background` de CSS 3 permite definir varios fondos de imagen para un elemento de la página.

- **Múltiples columnas**: Es posible estructurar textos demasiado extensos en varias columnas mediante las propiedades `colum-width` (ancho de las columnas a crear), `column-gap` (espacio en blanco entre columnas) y `column-rule (`línea divisoria entre las columnas).

- **Nuevo modelo de cajas**: La propiedad `box-sizing` de CSS 3 hace posible especificar el comportamiento del navegador, calculando el ancho de un elemento y controlando si una caja se puede redimensionar.

- **Web Fonts**: CSS 3 permite la representación dinámica de fuentes de texto en las páginas Web, sin necesidad de que se encuentren instaladas de forma local en el equipo del desarrollador. Para que la fuente pueda ser descargada y visualizada, se hace necesaria la declaración de la regla `@font-face`, que establece el nombre y la ubicación de la misma.

3.4.3.2. Soporte

En el momento de publicación de la presente guía, los navegadores Web que cuentan con un mayor soporte de CSS 3 son los que utilizan el motor "*WebKit*" como *Safari* o *Google Chrome*, mientras que *Mozilla Firefox* y *Opera* no soportan algunos de los módulos CSS 3, situándose un paso por detrás de los primeros. Por su parte, *Internet Explorer* en su versión actual (la octava) ofrece un soporte para CSS 3 muy reducido, si bien, Microsoft ha anunciado que dará soporte a todos los selectores de CSS3 en su futura novena versión.

3.5. MODELO DE OBJETOS DEL DOCUMENTO (DOM)

El modelo de objetos de un documento Web (DOM) no es más que un **modelo abstracto de objetos** generado por un navegador cuando, para presentar la página, analiza el código que recibe del servidor y crea una representación o estructura interna. Técnicamente, el DOM es una API independiente del lenguaje de programación, que permite al desarrollador modificar el contenido y la visualización de los documentos (X)HTML como si se tratase de documentos XML.

El DOM es una **estructura compleja** cuyo proceso de definición empezó con un subconjunto modesto, el nivel 1. Posteriormente se ha ido ampliando y completando con otros niveles que los fabricantes han adoptado progresivamente. Hoy en día es posible programar adecuadamente sólo contra el DOM W3C para lograr prácticamente la plena interoperabilidad y la posibilidad de programar una sola vez para ejecutar en cualquier navegador.

DOM permite manipular los contenidos de una página mediante JavaScript. Para ello, el navegador realiza de forma rápida y automática una transformación interna de la página original en una estructura más manejable basada en nodos interrelacionados que representan los contenidos de la página Web y sus relaciones, comúnmente denominada "***Árbol de nodos***".

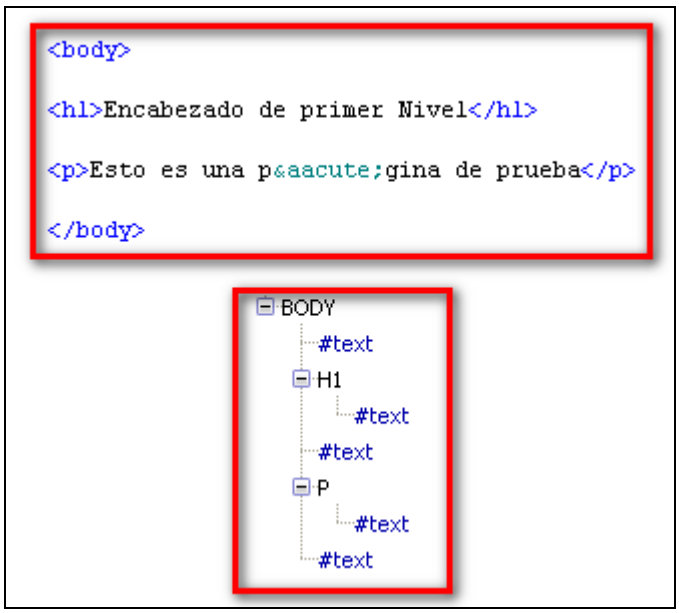


Figura 4. Código HTML de ejemplo y su correspondencia en el árbol DOM

En este proceso de transformación automática, cada etiqueta (X)HTML se transforma en un primer nodo, correspondiente a la propia etiqueta, y un segundo nodo (hijo del primero) en el que se incluye el texto encerrado por dicha etiqueta. En caso de que una etiqueta sea definida dentro de otra, el procedimiento es idéntico, teniendo en cuenta que los nodos resultantes son nodos hijo de su etiqueta padre.

Los objetos del DOM tienen propiedades que corresponden a los atributos del lenguaje de marcado, pero además tienen métodos que les permite realizar acciones. Algunas de estas acciones consisten en agregar nuevos contenidos, modificarlos o quitar contenidos de un documento Web. Se ha de tener en cuenta que los nombres de los objetos son muy similares a sus respectivas etiquetas (X)HTML, lo que en muchas ocasiones provoca que el desarrollador trabaje con ellos sin diferenciarlos de éstas.

Es imprescindible que el documento (X)HTML se haya cargado por completo ya que el árbol del DOM no se genera hasta que el navegador haya obtenido todo el documento (X)HTML, por lo que las diversas funciones que proporciona la API del DOM no se activan hasta que el árbol de nodos es completamente generado.

Cabe destacar que el navegador *Mozilla Firefox* cuenta con una extensión muy útil llamada "***DOM Inspector***", que además de mostrar visualmente el árbol de nodos generado por el DOM, permite acceder a los contenidos y propiedades de cada nodo.

3.6. OTROS LENGUAJES

3.6.1. Expresiones matemáticas: MathML

Las expresiones matemáticas poseen una **carga semántica** muy alta, por lo que han de ser inequívocas. Para su representación, se emplea un completo lenguaje visual con gran número de símbolos y estilos de escritura.

El lenguaje matemático hace posible una comunicación rápida y fiable entre personas, si bien, hay que contemplar dos grandes problemas en su empleo**: la entrada y la salida de información**. Por un lado, es necesario encontrar el modo de convertir un proceso matemático en un lenguaje completamente visual. Mientras que por otra parte, también se plantea la problemática en cuanto a la interpretación de la información matemática a través de un lenguaje visual por parte de los sistemas informáticos y las personas que cuentan con una discapacidad visual o cognitiva.

Habitualmente, se han empleado imágenes de mapa de bits para resolver los inconvenientes del lenguaje matemático, lo cual conlleva una serie de desventajas:

- Imposibilidad de modificar la presentación.
- Información no comprensible para sistemas automatizados o programas de matemáticas.
- Pérdida de indexación y búsqueda.
- Dificultad de mantenimiento.

MathML es un lenguaje que permite **presentar**, **manipular** y **publicar** expresiones matemáticas en la Web. Si se desarrolla correctamente una expresión en MathML, ésta podrá ser evaluada mediante un sistema automático, presentada en un navegador, leída en voz alta por un lector de pantalla y ser manipulada en un procesador de textos, además de poder imprimirse.

En MathML existen dos posibilidades a la hora de presentar las expresiones matemáticas:

- Describir la notación escrita (Codificación de contenido).

- Explicar los procesos, las reglas y su aplicación (Codificación de presentación).

$$P_L(d|X;\theta,\lambda) = \frac{1}{Z_L}\exp\left(-\sum_{i=1}^{M}\frac{|d_i(1)-x_i^T\theta_r|}{\lambda_{1r}} - \sum_{s=1}^{3}\sum_{i=1}^{M}\sum_{j\in N_s(i)}\frac{|d_i(s)-d_j(s)|}{\lambda_{2rs}}\right)$$

Hoy en día, los navegadores Web están empezando a implementar MathML de forma nativa. No obstante, existe falta de homogeneidad, ya que cada navegador implementa un subconjunto de MathML diferente, a lo que se ha de añadir la imposibilidad de garantizar en cualquier caso una correcta visualización de los elementos de las expresiones matemáticas.

Estos problemas de compatibilidad pueden resolverse mediante el empleo de MathPlayer, un **plugin** para *Microsoft Internet Explorer* mediante el cual es posible visualizar de forma completa y fiable las expresiones matemáticas en MathML.

Entre otras funcionalidades, MathPlayer permite ampliar imágenes, leer en voz alta ecuaciones o presentarlas en Braille, copiar código XML para su uso en otras aplicaciones compatibles con MathML. Otro aspecto muy importante, es que MathPlayer también proporciona a los productos de apoyo síntesis de voz de forma directa con matices de ritmo, énfasis y entonación, para su adecuada comprensión.

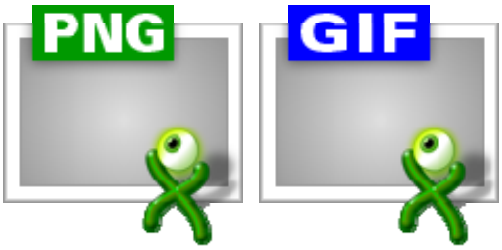
Los lectores de pantalla no interactúan adecuadamente con los modos nativos estándar de los navegadores. Para que una expresión matemática elaborada mediante MathML pueda ser utilizada a través de lectores de pantalla, y debido a que éstos interactúan con el sistema de síntesis de voz de MathPlayer, es aconsejable crear una página preparada para la integración del plugin MathPlayer.

3.6.2. Formatos gráficos: PNG vs GIF

PNG (Portable Network Graphics) se define como un **formato de archivo de gráfico** creado por el W3C (año 1995) que permite **almacenar imágenes sin pérdidas** y con un **ratio de compresión correcto**, lo que le ha situado como el sustituto gratuito al GIF (Graphics Interchange Format), siendo éste un formato patentado cuyos derechos pertenecen a

Unisys, propietario del algoritmo de compresión LZW, y que por lo tanto necesita licencia para su empleo.

PNG es una recomendación del W3C desde el 10 de noviembre de 2003 y actualmente también se considera un estándar internacional (ISO/IEC 15948:2003).



A continuación se detalla el conjunto de características que hace de PNG un formato superior al formato GIF:

- El formato PNG permite generar imágenes de mapas de bits sin pérdida de información, con un ratio de comprensión de un 5% a un 25% mejor que el ratio de compresión del formato GIF.
- PNG ofrece colores más ricos y precisos, al soportar una profundidad de color de hasta 16,7 millones de colores, mientras que GIF admite sólo un máximo de 256 colores distintos en la misma imagen.
- PNG soporta canales alfa y puede definir 256 niveles de transparencia, mientas que GIF sólo permite la definición como transparente de un único color de la paleta.
- Posibilita una corrección de gama.
- Los archivos gráficos en formato PNG pueden ser indexados por los motores de búsqueda, debido a la inclusión de metainformación.
- PNG ofrece un modo de compresión progresivo (entrelazado de dos dimensiones) que facilita el reconocimiento de la imagen en el inicio de su descarga, lo cual resulta muy útil cuando tiene un tamaño muy grande o la velocidad de conexión es lenta.
- Se trata de una especificación de libre uso que no requiere pagar licencia.

Es importante indicar que PNG también presenta algunos aspectos negativos respecto a otros formatos como GIF y JPEG, entre los cuales se citan los siguientes:

- Al contrario que GIF, PNG no soporta animaciones. No obstante, se ha de tener en cuenta que existe un formato alternativo del W3C que sin ser un estándar oficial, sí soporta animaciones, conocido como MNG (*Multiple-image Network Graphics*).

- Al tratarse de un formato sin pérdida de calidad, los archivos resultantes a través de PNG pueden ser demasiado grandes (formato orientado a imágenes gráficas), en comparación por ejemplo con el formato JPEG (*Joint Photographic Experts Group*), el cual tiene un ratio de compresión muy alto pero en detrimento de la calidad de la imagen (formato orientado a imágenes muy grandes y fotografías digitales).

- No tiene soporte en algunos navegadores antiguos.

3.6.3. Sindicación de contenidos: RDF / RSS

RDF (*Resource Description Framework*) es un lenguaje desarrollado por el W3C y convertido en recomendación desde Febrero de 1999, que permite la **descripción**, **especificación** e **intercambio** de **metadatos** en la Web.

Bajo una sintaxis XML, RDF se concibe como un lenguaje orientado a facilitar el procesamiento de información entre aplicaciones que intercambian información comprensible para las páginas Web.

RDF constituye una tecnología fundamental dentro de la **Web semántica**, cuyo funcionamiento se resume en una conversión de recursos de la Web en expresiones compuestas que se estructuran en tres partes:

- El recurso, es decir, lo que se describe.

- La propiedad del recurso que se desea definir.

- El valor de la propiedad o el otro recurso con el que se define la relación.

RDF presenta varios campos de aplicación como son las bases de conocimiento, los motores de búsqueda o las aplicaciones catalogadoras.

A partir del lenguaje RDF surge el RSS (*Real Simple Syndication*), un vocabulario RDF basado en XML que permite describir y catalogar información de tal manera que sea posible encontrar información precisa adaptada a las preferencias de los usuarios y que ésta pueda ser reutilizada.

RSS es un formato estándar y abierto diseñado para la **distribución automatizada de titulares de noticias y contenidos de sitios Web**. La sindicación de contenidos consiste

en mostrar un índice con un conjunto de titulares publicados en un sitio Web a cuyos contenidos se puede acceder sin necesidad de acceder al propio sitio.

La función de un archivo RSS, también conocido como *feed RSS* o *canal RSS*, es la de notificar de forma automática cualquier cambio que se realice en los recursos seleccionados por los usuarios, sin necesidad de que éstos accedan a la página Web para comprobarlo. Este archivo se estructura en "*ítems*", de forma que para cada uno de ellos se indica un título, un resumen y un enlace específico que conduce al contenido o texto completo del mismo.

Un archivo RSS, a diferencia de un archivo (X)HTML, **no puede ser interpretado por un navegador Web**, únicamente se visualiza el código XML que lo compone. Para poder interpretar el contenido de un canal RSS y visualizarlo adecuadamente, resulta necesario el empleo de un programa denominado "*agregador*" o "*lector de feeds*". Así, al iniciar un lector de feeds, el usuario recibe las noticias de forma automática sin necesidad de recorrer diferentes sitios Web en su búsqueda, lo cual es un proceso beneficioso en cuanto a la comodidad y ahorro de tiempo que supone.

En la actualidad se pueden encontrar diversos lectores de feeds para sistemas Windows (*Feedreader*, *Feedemon*, *Newsgator*, *Rssreader*…), Linux (*Lifearea*, *Straw*, *Rsstail*, *NRSS*, *Yarssr*…), Macintosh (*Netwire*) y Web (*Bloglines* o *NewsMonster*).

RSS posee varias **especificaciones**, las cuales se detallan a continuación:

- **RSS 1.0**: Se trata de una especificación basada por completo en RDF, concebida como extensión de ésta y con el fin de poderse emplear conjuntamente con otros documentos RDF.

- **RSS 0.91, 0.92 y 2.0**: Se adopta la especificación XML 1.0, abandonándose la especificación original de documentos RDF. Asimismo, se cambia el significado de las siglas RSS *Rich Site Summary* por *Really Simply Syndication*.

- **ATOM**: Especificación desarrollada por IBM, Google y otras empresas de posting, sin basarse en ninguna versión de RSS, tiene un formato similar e igualmente su objetivo es la sindicación de contenidos, con la ventaja de que un documento Atom puede contener más información y es más consistente que un documento RSS.

3.6.4. Presentaciones multimedia: SMIL

En la Web actual está creciendo la inclusión de contenidos multimedia, por lo que resulta muy importante **proporcionar alternativas** a los mismos en forma de texto o voz. Así, para personas que no pueden oír el sonido se ofrecerá un texto alternativo, mientras que para

personas que no puedan ver la imagen pero sí oír, se aportará una descripción de la acción que transcurre en la pantalla. Para que estas alternativas resulten efectivas, se hace necesario **sincronizarlas** con el vídeo y el sonido, así como leer los subtítulos en pantalla en cada momento y que se oigan las descripciones.

El amplio abanico existente de formatos multimedia creados por diferentes entidades privadas, constituye un problema a la hora de proporcionar alternativas sincronizadas en la Web. Así, cada uno de estos formatos cuenta con su propio sistema de generación de subtítulos y audio-descripciones, no resultando adecuados para cumplir las necesidades de accesibilidad para este tipo de contenidos.

Para solucionar este problema nace SMIL (*Synchronized Multimedia Integration Language*), un lenguaje establecido como norma, desarrollado por el grupo *Synchronized Multimedia Activity* del W3C que es independiente de los formatos privados y que permite sincronizar texto, audio y video en diferentes combinaciones. Se trata de un potente y flexible metalenguaje en XML que permite tener un **control absoluto sobre presentaciones multimedia**, describiendo el contenido que debe aparecer, en qué momento y en qué parte de la pantalla debe hacerlo.

Las principales ventajas que presenta SMIL son las siguientes:

- Facilita a los autores definir presentaciones multimedia interactivas.
- Se puede integrar con otros lenguajes de la familia XML.
- Permite modificar el comportamiento de una presentación multimedia en relación al tiempo trascurrido de la misma.
- Posibilita la inclusión de vínculos en presentaciones multimedia.
- Permite definir la posición de los elementos de la presentación en pantalla.
- Es un lenguaje independiente de plataforma.

Existen diferentes especificaciones de SMIL, siendo la versión 3.0 del 1 de Diciembre de 2008 la última de ellas a fecha de publicación de la presente guía. En la actualidad, la **compatibilidad de SMIL** está creciendo de forma paulatina, de modo que la mayoría de reproductores multimedia ya cuentan con soporte para éste. Igualmente, ya existen navegadores que sin necesidad de instalar ningún plugin son capaces de interpretar y ejecutar código SMIL.

3.6.5. Gráficos vectoriales: SVG

Las imágenes vectoriales consisten en definiciones curvas, puntos, polígonos y textos con sus dimensiones, rellenos, colores y tramas, llevadas a cabo a través de instrucciones fácilmente modificables.

Los ejemplos más conocidos de dibujos vectoriales son:

- Animaciones en Adobe Flash.
- Dibujos realizados en programas como *Adobe Illustrator*, procesadores de texto, diapositivas y las gráficas de las hojas de cálculo.
- Lenguaje de definición de página *PostScript* para impresión de alta calidad.
- Sistemas de dibujo de ingeniería (*CAD*) donde son importantes las dimensiones precisas y la posibilidad de gran ampliación.

Estos formatos tienen las desventajas de ser binarios y comprimidos lo que dificulta su tratamiento por un navegador para extraer texto, y su código patentado que impide la creación de diversas aplicaciones de usuario.

Con SVG la información se escribe en forma de texto, y tiene la ventaja de ser **código abierto**, **transportable** e **interoperable**, de la misma forma que HTML. Además, al ser un lenguaje XML, SVG se puede mezclar directamente con otros lenguajes XML como pueden ser una fórmula matemática en MathML o un comentario en SMIL. Por ser una especificación pública del W3C, numerosos programas de dibujo permiten la creación de dibujos SVG (OpenOffice o Adobe Illustrator entre otros).

Además, SVG permite **separar el contenido de la presentación** y es integrable con otros lenguajes, pudiéndose transformar mediante herramientas basadas en XML. Usado con imaginación, SVG constituye una tecnología muy potente para mejorar la **usabilidad** y **accesibilidad** en la Web.

3.6.6. Transformaciones: XSLT

XSLT es un potente lenguaje basado en XML que permite definir transformaciones de documentos XML válidos en otros.

Para efectuar una transformación XSLT se requiere un **procesador XSLT**. La función de éste consiste en buscar en el documento de datos correspondencias entre los nodos y el patrón, de modo que en el caso de encontrarlas, combinará el fragmento correspondiente del árbol con la plantilla. Un procesador XSLT de uso independiente y libre distribución muy conocido es Xalan.

La transformación se puede llevar a cabo tanto en cliente (navegador Web con soporte para XSLT) como en servidor. Generalmente, XSLT se utiliza en servidor, si bien, con el aumento de comunicaciones en XML mediante XmlHttpRequest y XForms, puede llegar a ser usado más ampliamente en cliente.

En caso de que una transformación XSLT sea llevada a cabo desde cliente, resulta necesario definir la **hoja de transformaciones XSLT** a aplicar, en la cual se recogerán diversas **plantillas** (fragmentos de XML que se utilizarán en el documento final) y un **patrón XPath**.

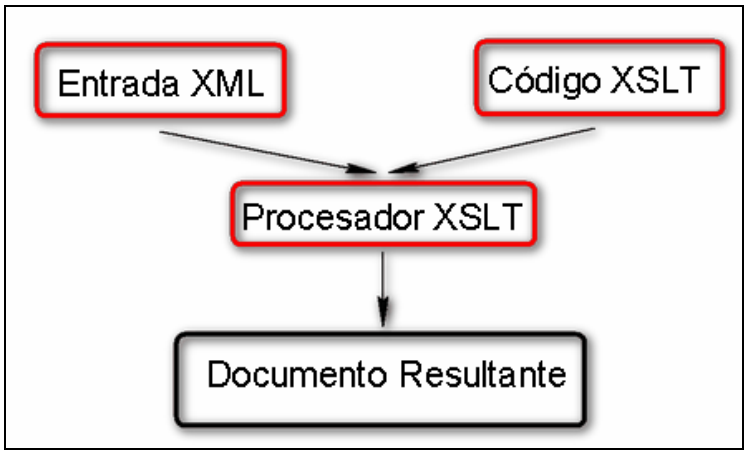


Figura 5. Transformaciones XSLT

Entre las ventajas que aporta XSLT, cabe citar las siguientes:

- Permite guardar los datos de forma independiente del documento final.
- Permite transformar la información en un formato adecuado a las necesidades especiales del usuario, como puede ser SVG o XHTML.
- Mediante XMLHttpRequest se pueden recuperar datos en XML del servidor y transformarlos al lenguaje de la página dentro de ésta (procedimiento habitual en AJAX).
- Permite transformaciones variadas tales como cálculos, selección dinámica y bloques repetidos.
- Permite al diseñador Web con conocimientos XML realizar tareas que antes requerían la intervención de un programador.

ANEXO I: TECNOLOGÍAS DE SCRIPT

I.1 JAVASCRIPT

En sus inicios, no todas las implementaciones de JavaScript eran compatibles con los diferentes navegadores existentes en el mercado. Para superar este problema, se presentó JavaScript ante el organismo de normalización ECMA (*European Computer Manufacturer's Association*) para crear una base de lenguaje consensuada y en el año 1996 se aprobó como la **norma ECMA-262**.

El lenguaje normalizado se llama ECMAScript y desde entonces ha pasado por varias versiones más. A pesar de que tanto JavaScript como sus variantes tienen extensiones y alcances más amplios que ECMAScript, siguen siendo compatibles con el núcleo que es ECMAScript. En la práctica JavaScript sigue siendo el nombre usado para referenciar al lenguaje.

JavaScript es una herramienta que **permite al navegador alterar los contenidos presentes en una página Web**, como por ejemplo agregar o eliminar contenidos, modificar estilos visuales o modificar textos del contenido. Se trata de un lenguaje altamente flexible que proporciona distintas formas de implementación para soluciones relativas al diseño y modificación de contenidos, siendo tarea del desarrollador utilizar el método más apropiado y accesible.

El uso de JavaScript es beneficioso para la mayoría de los usuarios, incluidos los usuarios con discapacidad, siempre y cuando se apliquen unas pautas sencillas de diseño y se tengan en cuenta las necesidades de una diversidad de usuarios y contextos de uso de forma **no intrusiva**.

Un proceso de diseño óptimo para una página Web comienza por la definición del diseño y del contenido sin utilizar scripts. En caso de se incluyan los scripts después de que la página Web haya sido diseñada, se evitarán problemas de accesibilidad derivados del uso inapropiado de JavaScript.

I.2 AJAX

AJAX (*Asynchronous JavaScript And XML*) es una técnica de desarrollo Web que permite crear aplicaciones interactivas. Estas aplicaciones se ejecutan desde cliente, estableciéndose una comunicación asíncrona con el servidor, de forma que el contenido de las páginas se actualiza sin necesidad de volver a cargarlas. Además de resultar una técnica cómoda para el usuario, AJAX permite aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

AJAX no se establece no como una única tecnología, sino como un conjunto de varias tecnologías que se pueden aplicar de formas muy variadas:

- **XHTML** y **CSS**: Presentación basada en estándares.
- **DOM:** Interacción y manipulación dinámica de la presentación.
- **XML**, **XSLT** y **JSON:** Intercambio y manipulación de información.
- **XMLHttpRequest**: Intercambio asíncrono de información
- **JavaScript**: Unión del resto de tecnologías.

En las aplicaciones Web tradicionales, las acciones del usuario en la página provocan llamadas al servidor, de forma que una vez que la petición del usuario ha sido procesada, el servidor devuelve una nueva página (X)HTML al navegador del usuario. Esta técnica puede resultar algo molesta para el usuario, dado que en caso de realizarse peticiones al servidor de forma continuada, el usuario deberá esperar que se recargue por completo con los cambios solicitados.

Sin embargo, en AJAX el cliente hace por medio del objeto **XMLHttpRequest** una petición al servidor, éste la procesa y le devuelve una respuesta en XML en lugar de una página (X)HTML. Posteriormente, el propio objeto XMLHttpRequest procesa dicha respuesta y actualiza únicamente las secciones necesarias de la página, evitando tener que recargarla por completo.

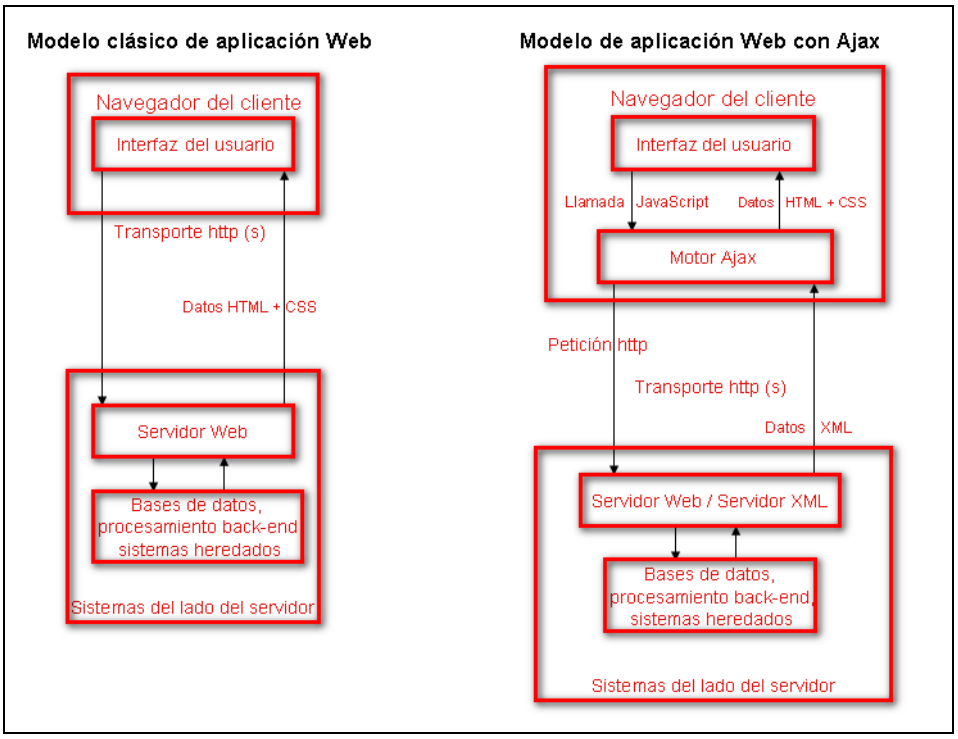


Figura 6. Diferencias entre el modelo clásico de aplicación Web y el modelo de aplicación Web con AJAX.

AJAX ofrece una serie de ventajas como son las siguientes:

- Mejora de la interacción del usuario con la aplicación, al evitarse las recargas constantes de páginas.
- Mejora de la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la respuesta del servidor.
- Al basarse en estándares abiertos, AJAX es multiplataforma, siendo posible su uso en diversos sistemas operativos y navegadores.

No obstante, también existen aspectos negativos en la utilización de AJAX como por ejemplo:

- Dependencia de JavaScript, al ser necesario que el navegador soporte y tenga habilitado JavaScript.
- Falta de integración con el botón "*History Back*" de los navegadores, que puede desorientar a los usuarios.
- El rendimiento de la máquina puede resentirse, al tener que ejecutar más código en cliente.
- El desarrollo de aplicaciones con AJAX es más complejo.
- Se pierde la referencia de la URL, por ejemplo a la hora de recomendar una página o de regresar a ella.

En la actualidad existen múltiples aplicaciones basadas en AJAX, entre las que se pueden encontrar gestores de correo electrónico como *Gmail*, *Windows Live Mail* o *Yahoo Mail*, aplicaciones de cartografía como *Google Maps* o *Yahoo Maps*, aplicaciones web de oficina como *Google Docs*, u otras como *Flickr*, *Digg* y *Netvibes*.

También se contempla la posibilidad de que el empleo de AJAX pueda llegar a sustituir a técnicas como Flash e incluso a aplicaciones de escritorio.

ANEXO II: REFERENCIAS OFICIALES

- Especificación oficial del lenguaje **HTML 3.2** (Recomendación del 14 de Enero de 1997): http://www.w3.org/TR/REC-html32.html
- Especificación oficial del lenguaje **HTML 4.01** (Recomendación del 24 de Diciembre de 1999): http://www.w3.org/TR/html401/
- Especificación del lenguaje **HTML 5** (Borrador del 4 de Marzo de 2010): http://www.w3.org/TR/html5/
- Especificación oficial del lenguaje **XHTML 1.0** (Recomendación del 26 de Enero de 2000)
- Especificación oficial del lenguaje **XHTML 2.0** (Borrador del 26 de Julio del 2006): http://www.w3.org/TR/xhtml2/
- Especificación oficial del lenguaje **CSS Nivel 1** (Recomendación del 17 de Diciembre de 1996): http://www.w3.org/TR/REC-CSS1/
- Especificación oficial del lenguaje **CSS Nivel 2.1** (Candidatura a Recomendación del 8 de Septiembre de 2009): http://www.w3.org/TR/CSS2/
- Página oficial del lenguaje **CSS 3** (Desarrollo actual): http://www.w3.org/Style/CSS/current-work#CSS3
- Especificación oficial del lenguaje **XML 1.0** (Recomendación del 26 de Noviembre de 2008): http://www.w3.org/TR/REC-xml/
- Especificación oficial del lenguaje **XSLT Versión 1** (Recomendación del 16 de Noviembre de 1999): http://www.w3.org/TR/xslt
- Especificación oficial del lenguaje **MathML** Versión 2 (Recomendación del 21 de Octubre de 2003): http://www.w3.org/TR/MathML2/
- Especificación oficial del lenguaje **SVG 1.1** (Recomendación del 14 de Enero de 2003): http://www.w3.org/TR/SVG11/
- Especificación oficial del formato gráfico **PNG** (Recomendación del 10 de Noviembre de 2003): http://www.w3.org/TR/PNG/

- Especificación oficial del lenguaje **SMIL 3.0** (Recomendación del 1 de Diciembre de 2008): http://www.w3.org/TR/SMIL3/

- Especificación oficial del lenguaje **VoiceXML** (Recomendación del 16 de Marzo de 2004): http://www.w3.org/TR/voicexml20/

- Especificación oficial del lenguaje **RDF** (Recomendación del 120 de Febrero de 2004): http://www.w3.org/TR/rdf-syntax-grammar/

- Especificación oficial del objeto **XMLHttpRequest** (Borrador del 19 de Noviembre de 2009): http://www.w3.org/TR/XMLHttpRequest/

- Especificación oficial del lenguaje **ECMAScript** (ECMA-262): http://www.ecma-international.org/publications/standards/Ecma-262.htm