[1.1] Introducción

A finales de los años 80 se desarrolló el lenguaje de marcas SGML. Al final de esa década **Tim Bernes Lee**, científico británico que trabajaba en el **CERN**[1], utilizó SGML para definir un nuevo lenguaje de etiquetas que llamó **Hypertext Markup Language** (lenguaje de marcado de hipertexto) para crear documentos transportables a través de Internet en los que fuera posible el **hipertexto**. Además de HTML, el equipo de Tim Bernes Lee definió las bases del protocolo **http** de Internet, encargado de trasportar los documentos HTML.

A pesar de tardar en ser aceptado, HTML fue un éxito rotundo y la causa indudable del éxito de Internet. Hoy en día casi todo en Internet se ve a través de documentos HTML que, popularmente, se denominan **páginas web**.

Inicialmente estos documentos se veían con ayuda de intérpretes de texto (como el antiguamente famoso **Lynx** de **Unix**) que reconocían el código HTML. Después aparecieron los llamados **navegadores** con capacidad de interpretar el código HTML de forma más visual y cambiar el tipo de letra, colores, incorporar imágenes, sonido, etc. Es decir, realmente los documentos HTML se convirtieron en páginas web.

En la actualidad HTML sigue siendo el lenguaje fundamental de las páginas web, pero ahora Internet es la web; es decir, prácticamente todo en Internet se ve a través de una página web. Por eso hoy en día HTML es la capa superficial bajo la que se agolpan tecnologías muy diversas y muy distintas de HTML.

[1.2]historia de HTML

La historia del lenguaje HTML es muy extensa, pero se puede resumir analizando sus orígenes en la era de los ochenta y repasando cada una de las distintas versiones que fueron surgiendo.

1989 Inicio de su desarrollo por parte de Tim Bernes Lee

1991 Lanzamiento de la web (se basa en 3 conceptos: http, html y url)

1992 Lanzamiento oficial del HTML

1994 Creación de la W3C (consorcio de empresas que definen los estándares de la web)

1998 Lanzamiento del HTML 4 (versión que mas duró)

1999 XHTML (HTML 4.1 actulización del estándar HTML 4)

2004 Creación de la WHATWG

2008 Lanzamiento HTML 5 (Lanzado por la WHATWG en forma independiente)

2014 estándar HTML 4 (lanzado por W3C de forma oficial)

2016 se publicó la recomendación HTML 5.1.

2017 se publicó la recomendación HTML 5.2.

[2] funcionamiento de las aplicaciones web

[2.1] HTML, CSS y JavaScript

Una aplicación web, es una aplicación creada usando como base el lenguaje HTML. Por lo tanto, se trata de una aplicación que se ejecuta en un navegador de Internet. Esencialmente las aplicaciones web actuales utilizan:

- **HTML**. Para dar significado a los contenidos de la aplicación web. Permite indicar qué textos son títulos, cuáles son párrafos normales, cuáles son celdas de una tabla, cuáles son imágenes, etc.
- **CSS**. Lenguaje que permite dar formato y maquetación a los contenidos. Color, tamaño de letra, posición, etc.
- **JavaScript**. Permite diseñar la interactividad de la página. Permite que las acciones del usuario se puedan capturar y que la página reaccione a ellas.

[2.2] Protocolo http

La transmisión de páginas web (que en definitiva son documentos HTML) se realiza mediante el protocolo **http**, que es parte de la pila de protocolos **TCP/IP**. Se trata de un protocolo basado en una comunicación **petición-respuesta**; de modo que un **cliente** realiza una petición de recurso indicando su dirección en Internet. La petición llegará a un **servidor http** (también llamado **servidor web**), el cual responde a dicha petición, bien transmitiendo al cliente el recurso solicitado o bien indicando un mensaje de error si el recurso no está disponible.

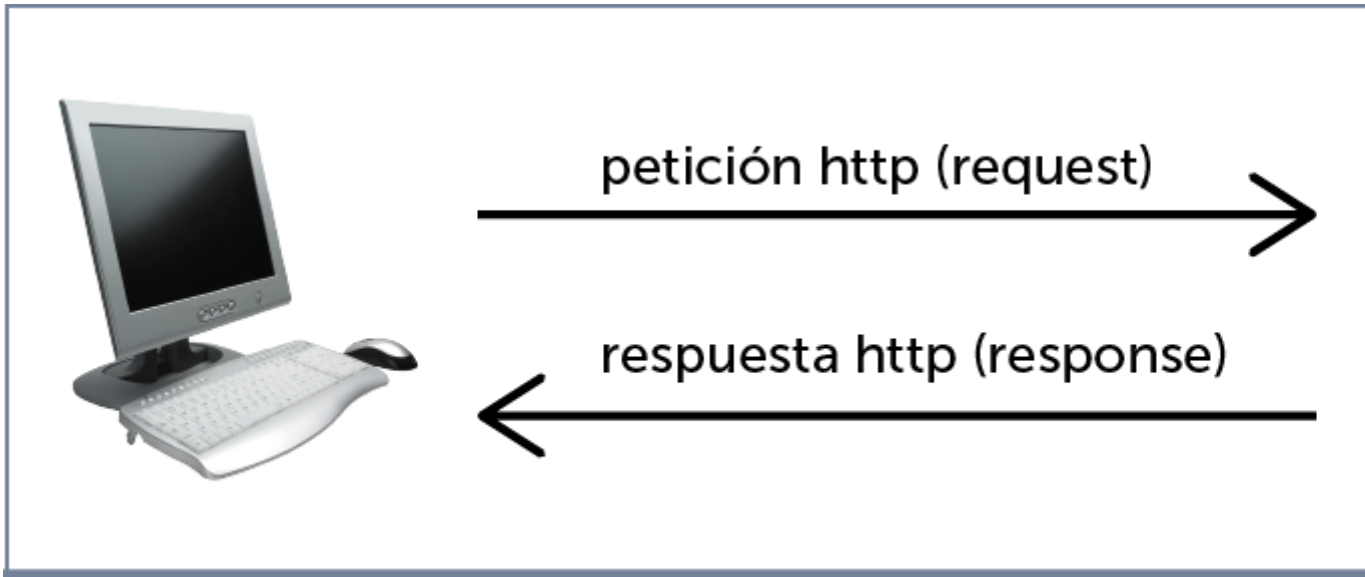

Ilustración 2-1. Esquema simple de funcionamiento del protocolo http

Los paquetes http que se intercambian entre el cliente y el servidor contienen una cabecera con información de control y luego el cuerpo. Este cuerpo es el código HTML en el caso de una página web, o la imagen, audio, etc. que se está transmitiendo.

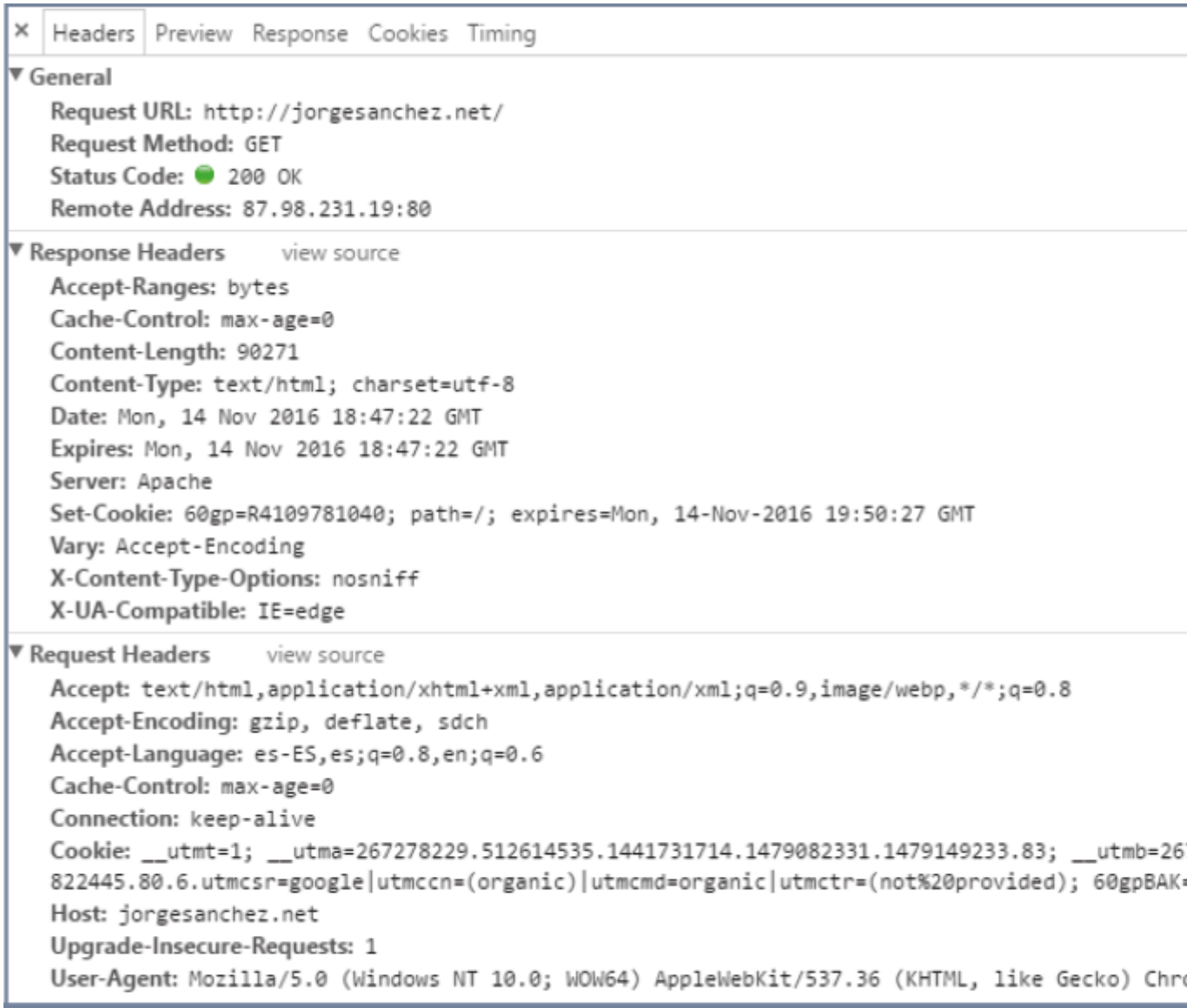

Ilustración 2-2. Cabecera que devuelve el paquete de respuesta http portada del sitio web jorgesanchez.net

En la Ilustración 2, se observa el paquete devuelto por el servidor web cuando recibe una petición http (de tipo GET) a la URL http://jorgesanchez.net. Se indica que la respuesta fue correcta (código 200, OK), la IP del servidor que responde, el tamaño del paquete, la fecha en la que expira, etc. Además se anexa la cabecera de la petición original (*request headers*) en la que se ve el navegador del usuario que hizo la petición, el lenguaje de su equipo, etc.

Además de la cabecera, el contenido de los paquetes http posee un cuerpo con el contenido del recurso que se solicitaba. Por

ejemplo en el caso de haber solicitado una página web, el cuerpo contiene código HTML que el navegador podrá traducir.

El protocolo http sirve para transportar todo tipo de contenidos. De hecho, el código HTML de una página web hace referencia a otros archivos (imágenes, archivos CSS, etc.). Cuando el navegador detecta estas referencias, solicita al servidor web estos recursos. El navegador responderá con nuevos paquetes que contendrán el contenido solicitado.

El cliente (normalmente un navegador) será el encargado de mostrar el recurso.

Entre los recursos transmitidos por http que un navegador es capaz de reconocer están:

- Código HTML, CSS y JavaScript
- Imágenes JPEG, PNG, SVG y GIF
- Vídeos MP4 y otros formatos
- Audio MP3 y otros formatos

[2.3] front-end y back-end

En este apartado se explica brevemente el funcionamiento de una aplicación web compleja. El único propósito es indicar que las aplicaciones web utilizan numerosas tecnologías.

En la mayoría de aplicaciones web complejas, cuando un navegador requiere de dicha aplicación, el servidor necesita ejecutar acciones como: interpretar código del lado del servidor (por ejemplo en **PHP** o **ASP.Net**), pedir recursos a otros servidores (bases de datos, mapas, streaming de vídeo, etc.) o almacenar datos de sesión.

Estas acciones son totalmente opacas al usuario. Tras su ejecución, el navegador del usuario recibe código que sí es capaz de interpretar (normalmente HTML, CSS y JavaScript).

Esto implica que una aplicación web puede realizar tareas en el llamado **lado del servidor**. Es decir, tareas que se ejecutan en el servidor y cuyo código no podemos obtener. Por lo tanto, cuando se crea una aplicación web hay personas dedicadas a programar en código traducible en el lado del servidor y código en el lado del cliente,

El lado del servidor se asocia al desarrollo **back-end**, sus programadores son denominados programadores back-end. Mientras que los profesionales encargados de la apariencia en el navegador son llamados programadores front-end.

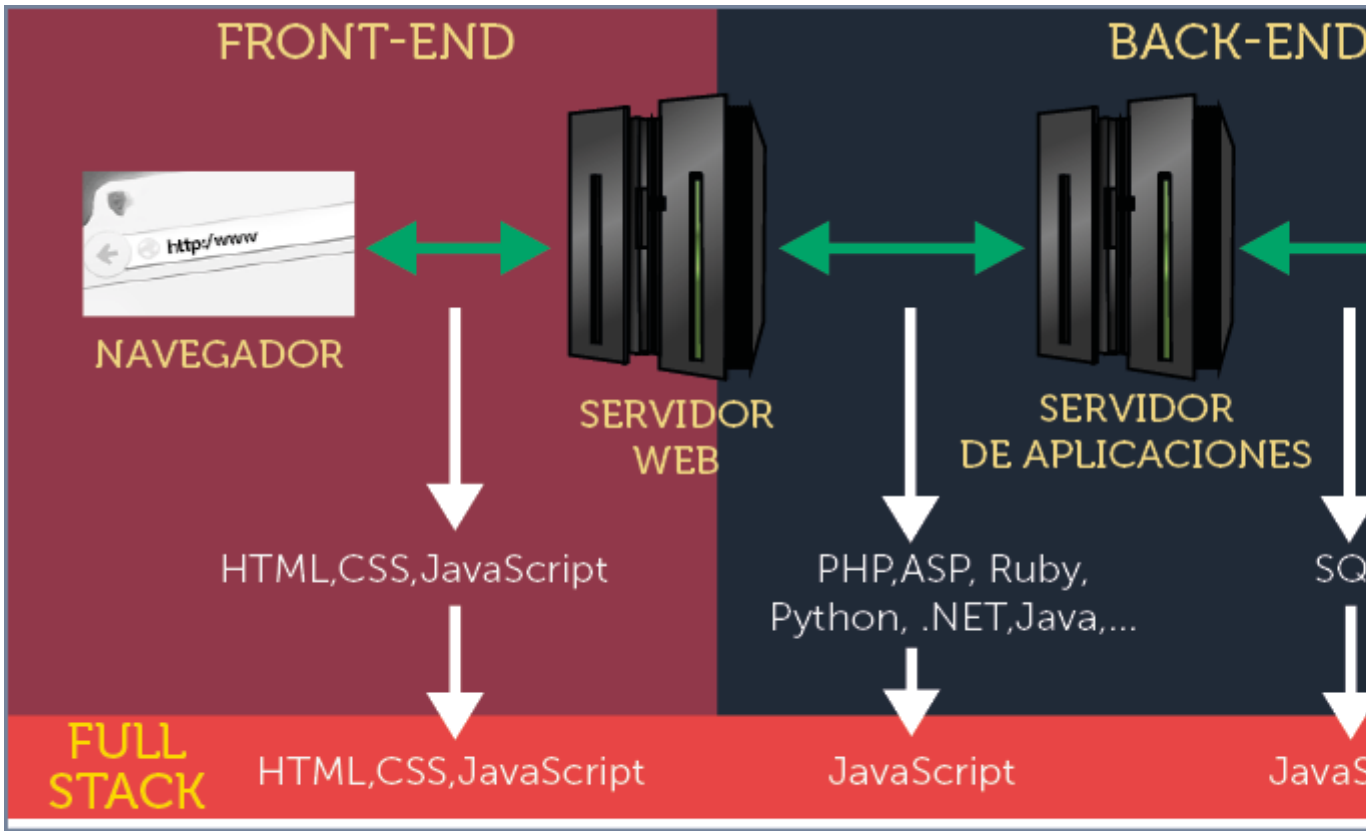

Ilustración 2-3. Esquema completo de tecnologías habituales en fro destaca la técnica Full Stack que usa JavaScript en todas las capas.

[2.3] Herramientas para crear aplicaciones en HTML

Para escribir código HTML bastaría con un editor de texto plano como el **bloc de notas** de Windows o **vi** de Linux/Unix. No

obstante la escritura de HTML en este tipo de editores es incómoda ya que no proporcionan ayuda ni aceleran la escritura.

Además, los resultados se deben mostrar en un navegador o, aún mejor, probar en varios navegadores para comprobar problemas de compatibilidad (ya que hay elementos HTML que no son compatibles con todos los navegadores).

Así la lista de navegadores a instalar para probar nuestro código sería: **Microsoft Internet Explorer**, **Google Chrome**, **Mozilla Firefox**, **Apple Safari** y **Opera**. Por otro lado es interesante probar en dispositivos móviles o bien añadir plugins a nuestros navegadores que simulen la visualización de un móvil.

Para escribir el código lo ideal es trabajar mediante editores de código capaces de entender el lenguaje y colorear de diferente manera las etiquetas HTML para distinguirlas del texto normal y así trabajar mejor y que además incorporen herramientas que faciliten la edición.

Así tenemos las siguientes opciones:

- **Editores de texto multipropósito (editores de código)**. Son programas que permiten editar texto, pero que están pensados para escribir código en cualquier lenguaje de programación. Su ventaja está en que colorean la sintaxis dependiendo del lenguaje en el que estamos escribiendo, porque son editores *políglotas*.

  Cuando escribimos en HTML, colorean el texto resaltando el código de forma que sea más legible. Además nos ayudan a escribir código HTML de forma más eficaz, porque incluso detectan los fallos en el mismo. Entre los más conocidos están **Notepad++**, **TextMate, Atom, Visual Studio Code** o **Sublime Text.**

- **Editores de texto HTML**. Son editores especializados en escribir código HTML. La diferencia respectos a los anteriores, es que, con estos, no podremos escribir código en lenguajes como C o Java. A cambio, tendremos más funciones para escribir en HTML. Por ejemplo, podremos

previsualizar el resultado del código HTML (adelantándonos al navegador), coloreado más ajustado a los lenguajes de la web, reconocimiento de librerías y plantillas de uso habitual de diseñadores,…

Algunos son: **Coda**, **Komodo Edit**, **WebStorm** o **Brackets.** Los editores **Atom y Visual Studio Code** hay quienes les colocan en este tipo porque, aunque son editores políglotas, están muy especializados en aplicaciones web.

- **Editores XML**. Pensados para la gente que trabaja creando documentos y elementos relacionados con XML. Entienden HTML, pero no son tan potentes como los anteriores, aunque pueden servir para trabajos que no requieren mucha especialización en HTML. Los editores XML más famosos son **Oxygen** y **XMLSpy**.
- **Entornos completos de programación**. Son los llamados **IDE** (*Entornos de Desarrollo Integrado*) que son entornos de trabajo para programar en lenguajes avanzados como **Java** o **C++,** pero que suelen trabajar perfectamente para crear proyectos para la web. Destacan **Eclipse**, **Aptana** (versión de Eclipse para lenguajes en la web), **NetBeans, IntelliJ IDEA** y **Microsoft Visual Studio**.
- **Editores online.** Se trata de aplicaciones web que nos permiten escribir y probar nuestro HTML. Por ejemplo, el editor de la w3schools:

  http://www.w3schools.com/html/tryit.asp?filename=tryhtml_basic

  O bien:

  http://htmledit.squarefree.com/

  Para pequeños fragmentos se usa https://codepen.io

- **Editores visuales**. Permiten trabajar de forma que podamos construir páginas web sin escribir código, de forma visual. Se basan en el software **WYSIWYG** (**What**

You See Is What You Get), lo que se ven en pantalla es el resultado que se obtendrá. Referido a que trabajas viendo el trabajo tal cual quedará finalmente. En realidad es imposible ya que el aspecto final dependerá del dispositivo y software que se utilice al ver la página y eso variará enormemente de un usuario a otro; pero al menos permiten hacerse una idea muy visual del resultado.

Adobe Dreamweaver sería el más famoso de este tipo y el más completo, ya que también permite trabajar de forma cómodo con el código. Otras opciones son **Adobe Muse**, **Tumult Hype**, **Google Web Designer**, **Kompozer**, **WYSIWYG Web Builder** o **Xara Web Designer**.

[4] normalización y estándares.

[4.1] Estandarización

Las incompatibilidades existentes en los navegadores han supuesto un problema desde el inicio de la web. La solución pasó por intentar estandarizar el lenguaje. Por ello, el propio Tim Bernes Lee fundó la **World Wide Web Consortium** (abreviado **W3C**) como organismo de estandarización del lenguaje HTML ante la industria. Este organismo se encarga de proponer estándares para el lenguaje HTML con la esperanza de que sea aceptado por los fabricantes de navegadores.

Con esta finalidad W3C trabaja con recomendaciones, a las que pone un número. Así ha habido HTML 2, HTML 3, HTML 4, etc. La W3C también se encarga de estandarizar otras tecnologías y lenguajes relacionados con HTML como CSS, JavaScript, SVG o el propio XML.

Las directrices de W3C son seguidas por la mayoría de navegadores aunque no al 100%, lo que sigue generando problemas a los creadores de páginas web. Por si fuera poco, este organismo ha perdido peso al apostar por XHTML, que es un HTML que usa las normas de XML y que ha perdido la batalla

de ser el estándar definitivo en favor de HTML 5, que, inicialmente, no fue apoyado por este organismo.

[4.2]validar el código HTML

Los navegadores no son estrictos con las normas HTML; es decir, aunque tengamos algunos fallos, van a ignorarlos en aras de mostrar la mejor versión del código que hagamos. A los navegadores les interesa hacer funcionar el código, sea válido o no.

Sin embargo, como creadores de páginas web, debemos cumplir las normas del lenguaje aprendiendo así realmente el lenguaje y además impidiendo que el navegador se equivoque al interpretar nuestros fallos. Además, hace que otros creadores de páginas comprendan el código perfectamente, facilitando el trabajo colaborativo.

El validador oficial de la W3C (http://validator.w3.org/) permite validar todas las versiones de HTML, incluida la versión 5.

[4.3] Compatibilidad entre navegadores

La tecnología dominante actualmente para crear páginas web es la que surge de HTML5 y que incluye el uso del lenguaje CSS (versión 3) y de JavaScript.

Aunque hoy en día casi todos los navegadores soportan HTML5, muchos (sobre todo si no están actualizados) no soportan todas las características. De hecho, como HTML5 está vivo y sigue introduciendo nuevas posibilidades, no hay ningún navegador que incluya todas las opciones.

[4.4] Referencias de ayuda

Ante tantos cambios e incompatibilidades, la situación es difícil para aprender. Por ello hay direcciones en Internet que nos ayudan a utilizar los distintos elementos HTML5 (incluso nos dicen su compatibilidad). Las más importantes son :

- **w3schools.** Contiene información al día sobre HTML, CSS, JavaScript y otras tecnologías web. Está en inglés pero es sencilla y bien estructurada. La ayuda de las etiquetas de HTML está en la dirección:

  http://www.w3schools.com/tags/

- **Mozilla Developer Network**. La página oficial de los desarrolladores del navegador Mozilla Firefox, está incluso mejor organizada que la anterior. Contiene información de todos los elementos HTML a medida que aparecen, aunque no estén del todo estandarizados, indicando su compatibilidad con los navegadores y si ya es estándar o no.

  Eso sí, hace más hincapié en los elementos que el navegador Mozilla Firefox es capaz de reproducir.

  https://developer.mozilla.org/en-US/docs/Web/HTML/Element

- **World Wide Web Consortium (W3C)**. Referencia oficial de la documentación de todos los estándares web. Se estructura de manera muy formal, pero tiene la ventaja de que esta documentación es la que se refiere al estándar. Sobre HTML podemos acudir a la dirección:

  https://www.w3.org/TR/html5/

[4.5] Publicación de páginas web

Se denomina **sitio web** al conjunto de páginas web y recursos de las mismas que contienen toda la información asociada a una determinada dirección de inicio en Internet.

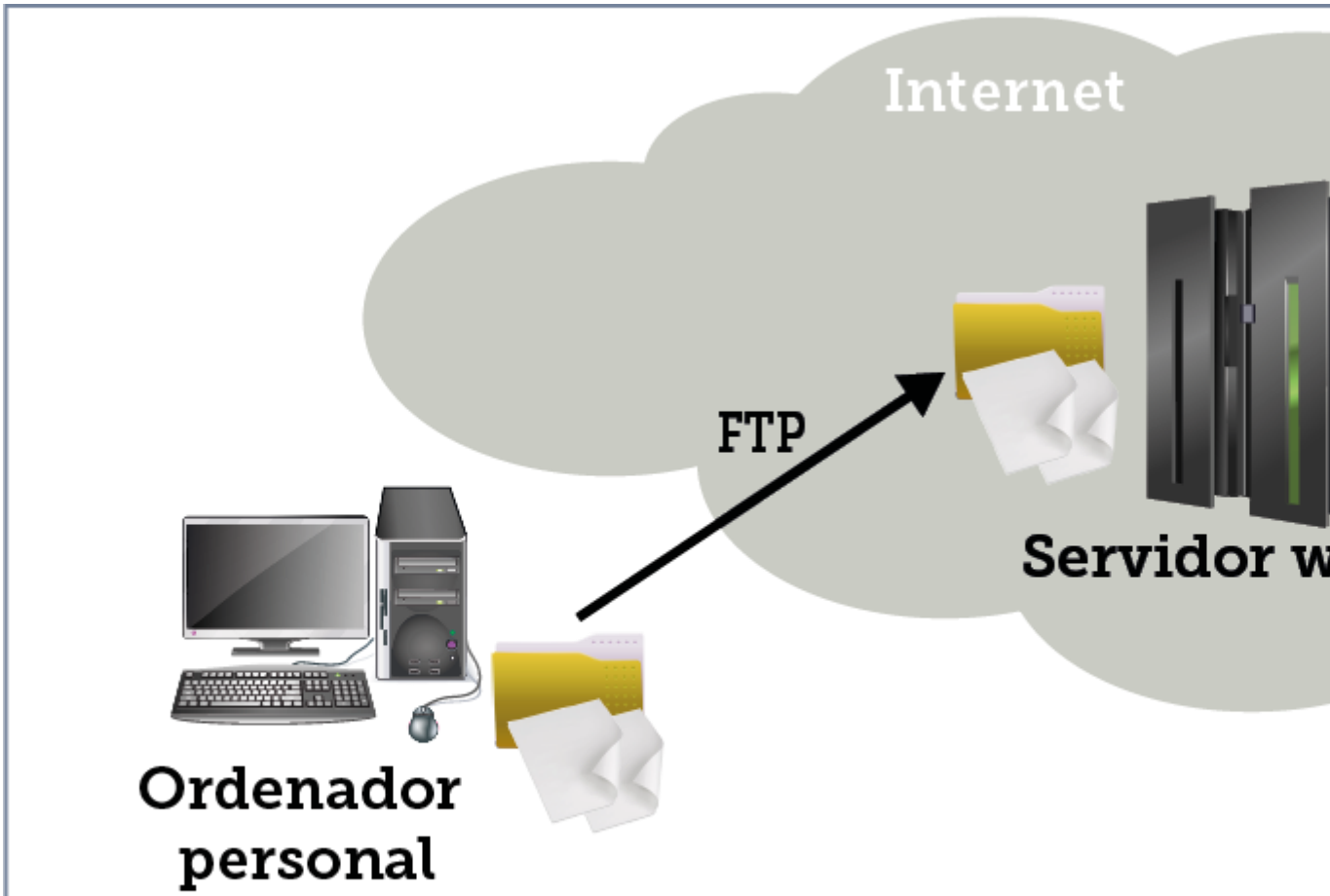

Ilustración 2-6. Idea de la publicación de los ficheros en el servidor mediante el protocolo FTP

Cuando una persona desea crear un nuevo sitio web, inicialmente le crea en su ordenador de trabajo y para ello debe crear una carpeta y en ella almacenar todas las páginas y recursos necesarios (imágenes, sonidos, vídeos, archivos auxiliares,…). Esa carpeta se deberá enviar al servidor web que hayamos contratado o del que dispongamos para publicar nuestra página en Internet.

Para ello normalmente se utiliza el protocolo de transmisión de ficheros conocido como **FTP**, aunque es posible que esa transmisión se haga con otros protocolos como **WebDAV** o **RDS**. Con copiar la carpeta en el sitio adecuado de nuestro servidor, la página estará publicada. Normalmente para ello se nos pide un usuario y contraseña que verifica que realmente somos los propietarios del espacio.

Otra opción, más habitual hoy en día para los desarrolladores, es utilizar sistemas de control de versiones en la nube (el más conocido es **GitHub**) y descargar la última versión del código directamente al servidor web. Es decir, desde nuestro ordenador subimos la última versión de nuestro código al repositorio en la nube y luego actualizamos el código en el servidor web.

Ilustración 2-7. Publicación del código de la página web usando siste
versiones en la nube