1

CONCEPTOS ESENCIALES

Es muy posible que, a lo largo de nuestra vida, nos tengamos que enfrentar en algún momento a la mirada de unos ojillos inquietos, llenos de curiosidad que nos preguntarán: «Abuelo, ¿es verdad que cuando eras pequeño no había Internet?». Y es que aquellos que han nacido en la era digital, los *Homo digitalis* (hoy retoños humanos), no concebirán el mundo sin la Red, a la que verán como nosotros, los nacidos en la era pre-digital, vemos el agua corriente, la luz o el gas.

Estamos en pleno siglo XXI y la Red forma parte de nuestras vidas desde hace tiempo, es un elemento más que hemos integrado en nuestra existencia de forma irreversible. Si por algún motivo dejase de existir, todos los esfuerzos de la Humanidad se centrarían en reconstruirla o en crear una solución similar. Internet ha venido para quedarse, la única cuestión es cómo y hacia donde evolucionará. Hoy, de un modo u otro, todo aquello que concierne al ser humano está orientado hacia la Red o bien tiene una traducción en esta. La revolución tecnológica nos ha calado como una lluvia fina, nos hemos adaptado a ella progresivamente con una gran naturalidad. A pesar de la celeridad de su devenir, incorporamos y asimilamos el uso de una nueva aplicación, un nuevo servicio o incluso una nueva tecnología con la misma facilidad con la que probamos el nuevo sabor de un producto conocido.

La llegada de los dispositivos móviles táctiles ha supuesto un salto cualitativo por su naturaleza itinerante y su intuitiva interfaz de usuario, ofreciendo una versatilidad y una facilidad de uso sin precedentes. Estos pequeños terminales se han convertido prácticamente en prótesis mentales, en extensiones corpóreas del usuario, en asistentes personales que facilitan el acceso a la información y al conocimiento, que permiten la realización de gestiones y la comunicación a todos los niveles. Los más jóvenes se inician en el manejo de estos dispositivos guiados única y exclusivamente por su intuición, sin ningún tipo de formación o mera introducción, explorando e interactuando al tiempo que adquieren destrezas y configuran su red

neuronal en base a la experiencia de uso. Estas tecnologías no solo nos permiten llevar Internet en el bolsillo, sino que, en función de nuestra ubicación, nos brindan información específica y personalizada muy valiosa. Los teléfonos móviles han dado paso definitivamente a estas computadoras de bolsillo, terminales personales ya imaginados por el físico y novelista Isaac Asimov a finales de los años cincuenta del siglo pasado, casi una década antes de la aparición del primer ordenador personal, más de dos respecto del lanzamiento del primer ordenador portátil y casi cuatro antes de la creación del primer y elemental teléfono inteligente.

Hoy tenemos la posibilidad de acceder a cualquier tipo de información en tiempo real o en diferido, podemos crear y compartir contenido multimedia, organizar un viaje, realizar operaciones con las administraciones públicas, escuchar música o realizar compras, entre otras muchas opciones. Y todo ello podemos hacerlo en cualquier sitio y en cualquier momento. Quien más quien menos, todos nos movemos con cierta intensidad a diario por la Red. Es posible que en algunas ocasiones no seamos plenamente conscientes de ello, sin embargo, cuando guardamos un archivo en DropBox, cuando se actualiza nuestro antivirus o cuando visualizamos la predicción meteorológica local en nuestro móvil, obviamente, estamos haciendo uso de Internet. Si bien es cierto que no precisamos conocer la tecnología que hace posible la Red para hacer uso de la misma, no obstante, resulta esencial si vamos a poner en marcha un proyecto pues potenciamos nuestras capacidades y ampliamos nuestros horizontes.

1.1 QUÉ ES INTERNET

Internet es un sistema descentralizado de redes conectadas entre sí y distribuidas a lo largo del planeta que ofrece servicios de comunicación de datos. Su origen se remonta a finales de los años sesenta, cuando el departamento de Defensa de Estados Unidos creó un sistema de cómputo que tenía el propósito de facilitar la colaboración científica para la investigación de carácter militar. Aquel programa, denominado Arpanet, tenía como característica fundamental la posibilidad de mantener las comunicaciones tras un hipotético ataque enemigo. Para ello se desarrolló una red entre iguales o, lo que es lo mismo, descentralizada. Era necesario que existiesen múltiples rutas entre los nodos que integraban la red, de modo que, si una vía de conexión era destruida se pudiesen utilizar otras alternativas de forma inmediata. Bajo esta misma filosofía se fueron conectando grupos de redes entre sí, es por lo que también se conoce a Internet como «la red de redes». Las primeras en integrarse a esta incipiente red fueron algunas universidades norteamericanas, transformando a lo largo de los años setenta este proyecto militar en una plataforma de índole académico. La esencia de aquella red primigenia perdura en nuestros días a pesar de los enormes y numerosos

cambios experimentados desde entonces. Aquella concepción de red descentralizada con que se creó Arpanet constituye la base de lo que hoy es Internet. De hecho, cuando nos conectamos a la Red, el dispositivo con el que lo hacemos se convierte en un nodo emisor-receptor capaz de conectarse con cualquier otro sin la necesidad de que exista un núcleo central que gestione dicha comunicación.

Internet es, en parte, la estructura física que empleamos hoy para comunicarnos y acceder a diversos tipos de contenidos y servicios. Sin embargo, esta estructura necesaria, por sí misma, sería insuficiente. Internet es algo más. Cuando realizamos una videoconferencia, navegamos por la web o simplemente contestamos a un correo, aunque no nos demos cuenta, estamos haciendo uso de los diferentes protocolos vinculados a la Red, es decir, existen una serie de normas esenciales que todas las aplicaciones deben cumplir para que la comunicación entre redes heterogéneas sea posible. Gracias a estos métodos estándar podemos realizar conexiones entre dispositivos tan diferentes como un PC y un iPhone. La estructura física y su familia de protocolos asociados forman una unidad indivisible en la que reside, en buena medida, una de las claves de su éxito.

1.1.1 La tecnología de la Red

Cuando nos conectamos a Internet, desde cualquier dispositivo, en realidad nos estamos asomando a un universo tecnológico de enorme complejidad, un universo que podemos simplificar dividiéndolo en dos partes: el lado físico y el intangible o inmaterial. Resulta imprescindible contar con la existencia de una red física de magnitud planetaria que permita el tráfico de datos, pero es igualmente necesario contar con un conjunto de estándares que establezcan unas normas y procedimientos comunes que posibiliten las comunicaciones entre los distintos puntos de la red.

Este conjunto de normas y procedimientos, denominados protocolos de Internet, son en la práctica un *software* que encontramos implementado en múltiples aplicaciones tales como los navegadores, los clientes FTP, los gestores de correo electrónico o el propio sistema operativo de nuestro equipo. Por otro lado, si queremos realizar una conexión necesitaremos enlazar con la estructura física de la Red y para ello tendremos que contratar los servicios de una compañía de telecomunicaciones que opere como ISP (*Internet Service Provider*). Esta compañía establecerá una conexión desde nuestro dispositivo hasta su nodo más cercado y desde este tendremos acceso a la Red. Los medios físicos que permiten el acceso a Internet pueden ser muy diversos, aunque los más comunes son el hilo de cobre, la fibra óptica, los satélites y las ondas de radio. Dependiendo de la ubicación geográfica en que nos encontremos tanto nosotros como el destino de nuestra comunicación y dependiendo también del tipo de tecnología que emplee nuestro operador, nuestros datos viajarán haciendo uso de unos medios u otros, aunque casi con toda probabilidad lo harán sobre un conjunto de ellos.

Pongamos un supuesto. Si visitamos, por ejemplo, una página web alojada en un servidor de Buenos Aires y lo hacemos desde Pamplona con un teléfono inteligente, nuestra petición será fragmentada y enviada a través de ondas de radio (UMTS o superior) hasta el repetidor más cercano y de este hasta el nodo más próximo, donde nuestros paquetes, a pesar de tener un mismo destino, podrán tomar diferentes caminos siendo conducidos muy probablemente hasta Madrid y de ahí hasta Lisboa, donde se sumergirán en las profundidades a través del cableado submarino rumbo a las Islas Canarias, desde donde atravesarán el Atlántico bajo las aguas hasta llegar a las costas brasileñas, que serán bordeadas con dirección sur hasta llegar a Buenos Aires por Rio de La Plata; una vez allí, recorrerán a través de fibra óptica la distancia que exista hasta el centro de datos o centro de cómputo que alberge al servidor que contenga la página web solicitada. Tras recibir la petición, el servidor procederá a satisfacer la demanda iniciando el camino de vuelta con nuevos datos hasta nuestro móvil ubicado en la capital de Navarra. La realidad es más compleja aún y las rutas de nuestros paquetes pueden seguir caminos muy diversos en función de las condiciones existentes en cada momento. Si podemos emitir y recibir de forma inteligible es gracias a la unión de las infraestructuras y de los protocolos que gobiernan la Red, juntos constituyen un nutrido conjunto de tecnologías que hacen posible este pequeño milagro que llamamos Internet.

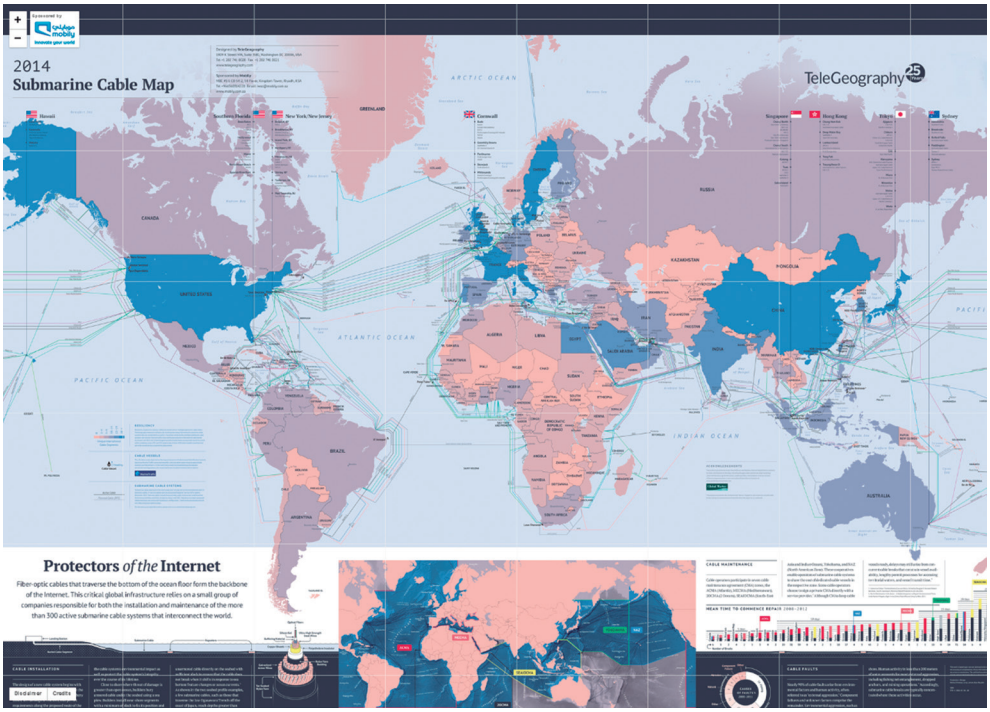

Figura 1.1. En la página web de TeleGeography podemos encontrar un mapa que muestra la red actual de cable submarino

1.1.2 Protocolos de Internet

Mientras a lo largo de la década de los setenta, algunas universidades norteamericanas transformaban Arpanet en una red con fines académicos, el Pentágono desarrollaba *Internetting*. Este proyecto tenía el propósito de permitir que redes preexistentes pudieran unirse a Arpanet, para lo que diseñaron un protocolo de comunicaciones que se convertiría en el origen de lo que hoy se conoce como TCP/IP o protocolo de Internet. Un protocolo es un conjunto de normas para el intercambio de información entre dos o más dispositivos. TCP/IP (*Transmission Control Protocol/ Internet Protocol*) es, en realidad, una familia de protocolos que permiten el tráfico de datos entre diferentes equipos independientemente de su arquitectura física y de su sistema operativo. Antes de que TCP/IP viera la luz sobre Arpanet, las redes solo eran viables entre equipos que compartían *software* y *hardware*, lo que les confería un carácter cerrado, siendo consideradas intranets.

En virtud de este protocolo, cada equipo conectado a Internet es registrado con un número único denominado dirección IP. Este identificador se compone de cuatro números comprendidos entre el 0 y el 255 y separados por puntos. Cuando accedemos a una página web realizamos previamente una solicitud al servidor que la alberga, tras esta, el protocolo TCP del servidor divide en paquetes el fichero requerido, numera cada uno de ellos y se los pasa al protocolo IP que se encarga de enviar dichos paquetes de un equipo a otro, etiquetando todos ellos con una serie de datos para garantizar que se encaminen hacia su destino correctamente y por la vía más rápida, que no será necesariamente la más corta. Una vez recibidos los paquetes en nuestro equipo, el protocolo TCP local, se encargará de recomponer todos los paquetes para reconstruir el fichero enviado, de modo que podamos visualizarlo en nuestra pantalla.

TCP e IP son los dos protocolos más importantes, pero la familia TCP/IP es muy numerosa y cada uno de ellos está diseñado para realizar una tarea específica. Así, tenemos el protocolo HTTP que nos permite navegar por la Red, FTP que establece las normas de transferencia de ficheros, SMTP encargado de conducir el correo electrónico desde el cliente al servidor, POP que realiza la tarea inversa descargando el correo desde el servidor hasta la bandeja de entrada de nuestro cliente de correo, IMAP de propósito similar a POP pero con características añadidas, RTP administrador de la entrega de audio o vídeo en las transmisiones de voz sobre IP y las aplicaciones de chat entre otras, y por concluir, Telnet y SSH protocolos empleados para realizar conexiones remotas interactivas en modo terminal. Naturalmente existen muchos más, tan solo hemos hecho mención de los más representativos.

El modelo TCP/IP trabaja haciendo uso de una pila de capas de carácter lógico, entre las que distribuye los protocolos necesarios para realizar una entrega

o una recepción de datos. Estas capas representan los diferentes pasos por los que han de fluir los datos para su correcto procesamiento. Existe una correlación entre el modelo de capas de Internet y el estándar de red ISO, si bien el primero agrupa algunos niveles respecto del segundo, contando con cuatro y siete capas respectivamente.

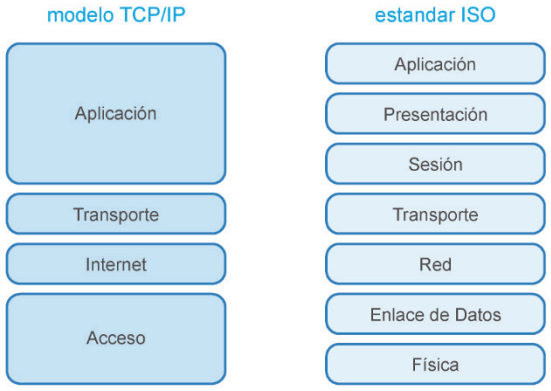

Figura 1.2. Correlación entre el modelo de capas de Internet y el estándar de red ISO

Cuando iniciamos una transferencia de datos, el proceso arranca en la capa superior, conocida como capa de Aplicación, donde se definen las aplicaciones de red y los servicios de Internet que están disponibles para el usuario (capa equivalente a la unión de las capas de sesión, presentación y aplicación del modelo OSI). A continuación, se encuentra la capa de Transporte, responsable del envío y recepción de datos de los elementos de la capa superior, garantizando la correcta fragmentación y recomposición de la información (equivalente a la capa del mismo nombre del modelo OSI). Después nos encontramos con la capa de Internet (equivalente a la de red del modelo OSI), donde se reciben los paquetes de la capa superior y se introducen una serie de datos entre los que se encuentran las direcciones IP tanto de origen como de destino para su correcto enrutamiento. Por último, encontramos la capa de Acceso al medio (equivalente a la unión de la capa de enlace de datos y la física del modelo OSI), donde se lleva a cabo un control de errores sobre el flujo, se describen los estándares de *hardware* y finalmente los datos son enviados. Cuando el equipo de destino recibe la transmisión, esta será procesada de forma inversa, es decir, los datos entrarán por la capa de Acceso al medio para terminar en la de Aplicaciones. Este sofisticado proceso se lleva a cabo con una velocidad pasmosa y es el que permite la comunicación entre dispositivos tan dispares como un Mac, una tableta con Android o un servidor Linux.

1.1.3 Qué es y cómo funciona la Web

Como en el caso de otros grandes inventos, detrás del origen de la Web está el intento de un hombre por optimizar su metodología de trabajo. En este caso fue el científico británico Berners-Lee del CERN, un laboratorio europeo de física nuclear, quien ideó una forma de distribuir e intercambiar información acerca de sus investigaciones de una forma más eficiente. Él y su grupo crearon en la década de los 80 algo conceptualmente muy sencillo, pero que sentaría las bases de la navegación web. Se trataba de una herramienta de *software* no secuencial que permitía saltar de un documento a otro mediante la creación de enlaces asociativos. Obviamente estamos hablando del hipertexto, aunque, este como concepto, es anterior. Fue Ted Nelson, filósofo e informático estadounidense, el primero en acuñar el término al desarrollar el proyecto Xanadú en 1965. A finales de la década de los 80, el CERN, se había convertido en el nodo de Internet más potente de Europa y Berners-Lee consideró la posibilidad de integrar el hipertexto en la Red. Tras una primera propuesta que fue desestimada, presentó en 1990 junto con el científico belga Robert Cailliau un modelo revisado que sí fue aprobado. Desarrollaron el primer navegador, al que bautizaron como WorldWideWeb, y el primer servidor web, al que dieron el nombre de HTTPD (*HyperText Transfer Protocol Daemon*). Berners-Lee terminó la primera página web en diciembre de ese mismo año, el modelo acababa de germinar. Es difícil imaginar hasta qué punto fueron conscientes, en aquel preciso instante los científicos del CERN, del enorme impacto que alcanzaría aquel proyecto.

La Web es un sistema de distribución y recuperación de documentos basados en hipertexto. Es el servicio que nos permite navegar entre las innumerables webs que existen, siguiendo los enlaces a golpe de ratón. Nos movemos usando el protocolo HTTP (*HyperText Transfer Protocol*) y el lenguaje de marcado HTML (*HyperText Markup Language*), que es el que permite, en última instancia, componer las páginas web en nuestros navegadores. Es frecuente cometer el error de confundir Internet con la Web y viceversa, sin embargo, es sencillo diferenciar ambos conceptos si consideramos que la Web, con toda su relevancia, no es otra cosa que un servicio más de Internet, como lo puede ser el correo electrónico, la transferencia de archivos o las conexiones remotas.

1.1.3.1 CÓDIGO HTML

HTML no es un lenguaje de programación propiamente dicho, sino más bien un sencillo código de etiquetado que permite definir la estructura básica que vertebra los diferentes contenidos de una página web, lo que no le impide ser reconocido como el lenguaje de la Web. Desde que vio la luz en 1990, este código ha recorrido un largo camino no exento de dificultades y, desde luego, de grandes cambios. Hasta 1995, año en que fue liberada la versión 2.0, pasó casi inadvertido a pesar de que

ya era considerado un estándar. La versión 3.0, lanzada en 1997, fue prácticamente ignorada por las compañías propietarias de los navegadores que, en plena guerra por la dominancia, desarrollaron sus propias especificaciones. La versión 4.0 encontró un marco más favorable, pero hasta 2004 no se inició el proyecto que daría lugar a la versión 5.0 que hoy conocemos como HTML5, una edición que goza del reconocimiento de toda la comunidad.

Existen numerosas herramientas que nos permiten crear páginas web sin tener que introducir una sola línea de código, por lo que no es necesario conocer HTML. Sin embargo, sí puede resultar muy conveniente contar con algunas nociones, pues nos brindará una perspectiva más precisa del terreno que pisamos, y como hemos apuntado, se trata de un código realmente sencillo y muy fácil de aprender. La estructura de un documento HTML viene fijada por la ubicación de una serie de etiquetas predefinidas. Además, estas etiquetas sirven para dar significado a los diferentes elementos que componen la página web, unos elementos que son referenciados por el código del documento y nunca incrustados en el mismo, de modo que por un lado tendremos el archivo HTML y por otro los diferentes objetos asociados, tales como imágenes, vídeos, normas de diseño o librerías JavaScript.

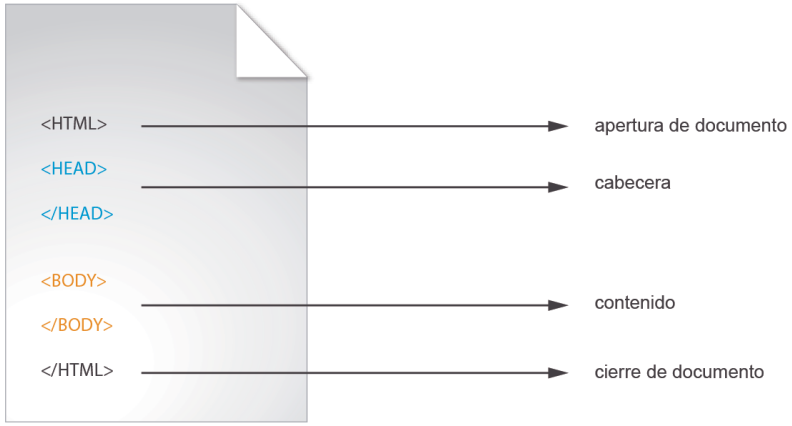

Figura 1.3. Estructura básica de un documento HTML

En líneas generales, las etiquetas HTML se presentan por pares, representando la apertura y el cierre de un espacio cuyo contenido poseerá unas características específicas definidas por dichas etiquetas. De este modo podemos encontrar una etiqueta que establece el inicio del documento y otra de cierre al final del archivo. Entre ambas, las etiquetas «Head» limitan el área que alberga todos aquellos datos que, siendo relevantes, no serán visualizados en la página web. Aquí podemos encontrar enlaces a recursos externos como el vínculo a una o varias hojas de estilos −aquellas

que fijan la apariencia visual del documento– y, también, enlaces a bibliotecas de JavaScript locales o remotas, entre otras muchas posibilidades. El mensaje se encuentra acotado por las etiquetas «Body» entre las cuales se ubicará todo aquello que deba ser visualizado en el navegador, es decir, la página web propiamente dicha. Como es natural, existen muchas otras etiquetas con diferentes características y propósitos. La versatilidad de HTML admite además embeber código de lenguajes avanzados como JavaScript, PHP o ASP, lo que permite la creación de aplicaciones web de gran complejidad y sofisticación.

1.1.3.2 HOJA DE ESTILOS

Una hoja de estilos es un conjunto de instrucciones que define la apariencia visual de una página web. A pesar de que el concepto no es nuevo y tampoco exclusivo de la Web, lo cierto es que desde 1999 y tras la publicación de los primeros apuntes de su última especificación, que dio en llamarse CSS3, se ha impuesto progresivamente como modelo de diseño web a todos los niveles. Ya en 2011 y tras el lanzamiento del grueso de esta última versión, se empezó a considerar como una capa más en el desarrollo de aplicaciones web.

Cuando hablamos de CSS nos referimos a un lenguaje cuya razón de ser es separar la apariencia y el contenido de un sitio web. Las reglas CSS definidas gobiernan el formato en que será mostrado el contenido al usuario, es decir, estamos hablando de la distribución de imágenes, vídeos, texto, fuentes tipográficas, tamaños, alineaciones, colores, fondos, comportamientos interactivos y un largo etc.; aplicándose de este modo un diseño previamente definido sobre los contenidos de la web. A pesar de que las reglas de estilo pueden ser definidas en el propio archivo HTML al que hacen referencia, lo más común es implementarlas en un documento independiente con la extensión CSS, de modo que pueda contener la información de estilo de todos los documentos que componen el sitio web.

Se denominan hojas de estilos en cascada porque una regla definida en el documento CSS principal puede ser sobrescrita, haciéndola más específica, dentro de un archivo en particular o incluso concretando un parámetro en un punto concreto, heredando no obstante todos aquellos valores que no han sido explícitamente añadidos o modificados. Este comportamiento nos permite fijar las reglas generales para todo el sitio web y al mismo tiempo ser precisos definiendo valores personalizados para casos concretos.

CSS posee una sintaxis muy sencilla. Si abrimos un documento encontraremos una sucesión de reglas, cada una de las cuales se compone de dos partes diferenciadas: el selector y la declaración. El primero indica al navegador qué elemento o etiqueta ha de recibir el estilo definido precisamente en la declaración.

Ésta última se encuentra delimitada por una llave de apertura y otra de cierre, y entre ellas puede haber una o un conjunto de reglas, cada una de las cuales estará constituida por un par: propiedad, valor.

Uno de los principales beneficios que aporta la hoja de estilos al diseño web es el carácter modular que le confiere mediante la separación de contenido y apariencia visual, como anteriormente apuntábamos. Tenemos control sobre la estructura del sitio web a través de los archivos HTML y sobre la presentación a través de los documentos CSS, de manera que podemos realizar cambios de forma independiente y, al mismo tiempo, de ámbito global. Es posible definir una nueva etiqueta que automáticamente será aplicada a todos los contenidos de los diferentes archivos que compongan nuestra web sin necesidad de editar éstos, es decir, definimos una única vez, pero se aplica en múltiples puntos. Sin embargo, con todo, una de las principales ventajas es poder definir el diseño que se aplicará a contenidos aún no presentes. Pensemos en sitios avanzados que poseen un gestor de contenidos, y más concretamente en una aplicación de comercio electrónico, cuando introducimos un nuevo artículo deseamos que se apliquen sobre él las mismas normas gráficas que poseen el resto de elementos y en ningún caso nos planteamos tener que incorporarlas cada vez que se produce un alta.

1.1.3.3 PROTOCOLO HTTP

Cuando introducimos una dirección en nuestro navegador se inicia un proceso que guarda cierta similitud con el que se produce cuando marcamos un número de teléfono. Nuestro equipo realiza una conexión con el servidor que está vinculado con la dirección requerida y este recibe nuestra solicitud, que no es otra cosa que una petición de acceso al contenido del sitio web que alberga. El servidor atiende nuestra solicitud enviándonos el código que, una vez interpretado por nuestro navegador, nos presentará en pantalla en forma de página web. Es un gesto que resulta sencillo y rápido a pesar de que encierra procesos de enorme complejidad de los que el protocolo HTTP es el principal responsable.

HTTP es el protocolo estándar para la transferencia de recursos en la Web, se ocupa de la transmisión de HiperTexto articulando los intercambios de información entre los navegadores y los servidores web, haciendo posible la navegación mediante la introducción de direcciones web y el seguimiento de enlaces. Cuando tecleamos una URL en la barra de dirección de nuestro navegador, aparece de forma automática el prefijo «http://». Se trata del identificador del protocolo que estamos empleando y es que, técnicamente hablando, nuestro navegador es esencialmente una aplicación cliente HTTP que permite realizar conexiones y transferir datos con servidores que poseen aplicaciones HTTP servidor.

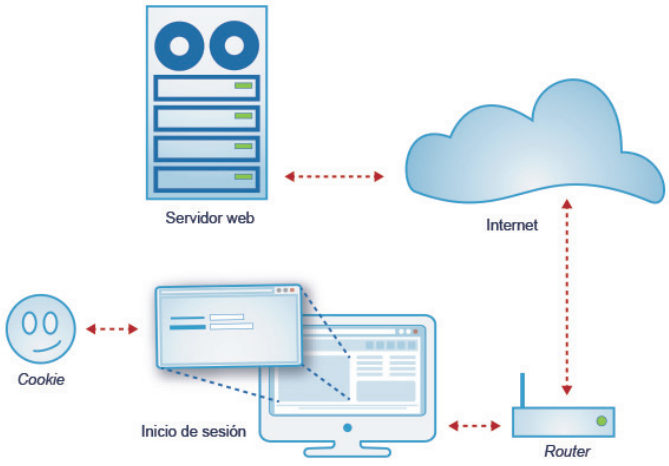

Figura 1.4. HTTP es un protocolo sin estado, por lo que no permite crear conexiones duraderas. Una de las vías para salvar esta aparente inconveniencia es la creación de cookies

HTTP es un protocolo sin estado, es decir, no conserva ningún tipo de registro acerca de las conexiones realizadas, dicho de otro modo, tanto el cliente como el servidor «olvidan» que ha existido contacto entre ambos. Este punto puede resultar un tanto chocante, puesto que es muy común que los sitios de cierta complejidad requieran mantener el estado. Por ejemplo, cuando iniciamos sesión en una web o cuando introducimos productos en el carrito de la compra, tenemos la sensación de haber establecido una conexión que perdura en el tiempo, sin embargo, esto no es así, ya que HTTP no lo permite. Si permanecemos autentificados en una web, generalmente es gracias al uso de las famosas *cookies*. Una *cookie* es un pequeño archivo que contiene información que un servidor almacena en nuestro sistema. Este archivo puede permitir, entre otras cosas, que mantengamos de forma virtual la conexión y que, de este modo, exista el concepto de sesión en la navegación web. No obstante, es posible dar a las *cookies* otros usos menos bondadosos para con los visitantes, como por ejemplo el rastreo de usuarios. Las *cookies* pueden contener información muy diversa y permanecer en nuestros equipos por tiempo indefinido, algo que puede representar una quiebra en la privacidad del usuario. Existen otras vías alternativas que permiten el almacenamiento de datos persistentes, como pueden ser el uso de lenguajes de *scripting* tales como JavaScript, el uso de variables de servidor o incluso la gestión de sesiones basadas en servidor. Podría parecer a simple vista que el hecho de no poseer estado es un defecto o una carencia de HTTP, sin embargo, esta apreciación está muy lejos de la realidad, pues es precisamente esta particularidad la que hace que este protocolo sea sencillo y eficiente. Si los servidores web tuviesen que mantener la conexión con cada uno de los cientos de miles o millones de clientes que potencialmente pueden

acceder a sus recursos, obviamente, quedarían colapsados más pronto que tarde independientemente de sus características.

Es muy probable que en más de una ocasión hayamos advertido cómo el prefijo «http://», que aparece en la barra de dirección de nuestro navegador, cambia por «https://». Se trata de una variante del protocolo que añade una capa de seguridad para proteger la comunicación frente a un posible ataque pirata. Si nuestra comunicación es interceptada no se podrá acceder a la información. Este sistema está basado en el protocolo criptográfico SSL (*Secure Socket Layer*). Es frecuente encontrarlo en comercios electrónicos y bancos, pero su uso ya se está extendiendo a algunas redes sociales y en general a sitios web que manejan información sensible.

1.1.3.4 ANATOMÍA DE UNA URL

Mientras navegamos, generalmente hacemos un uso intenso de las URL. Seamos o no conscientes de ello y tanto si las introducimos directamente como si llegamos a ellas desde diferentes enlaces, nuestro navegador nos habrá mostrado un nutrido conjunto de direcciones al cabo del día. Sin embargo, es muy posible que nunca nos hayamos parado a preguntarnos por qué son como son, a qué obedecen sus divisiones y qué utilidad tiene cada una de ellas.

URL es el acrónimo inglés de Localizador Uniforme de Recursos (*Uniform Resource Locator*). Su sintaxis permite localizar el servidor que contiene la web y además indica la ruta donde se encuentra el recurso dentro del disco duro del servidor. Por este y otros motivos, encontraremos direcciones concisas y comprensibles, y otras en cambio, enormemente largas, complejas y casi indescifrables. Estas últimas, por lo general, contienen información adicional que es transmitida al servidor. Analicemos una URL sencilla, como la de la figura 1.5.

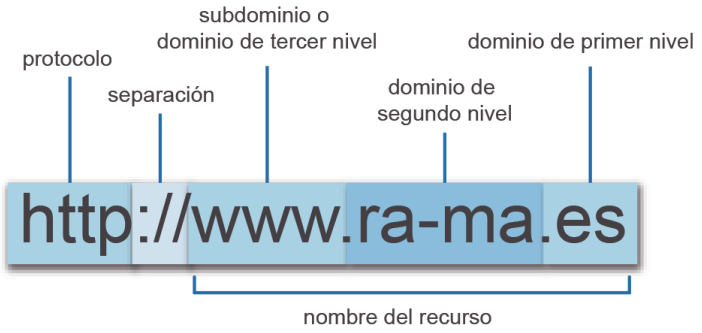

Figura 1.5. Secciones de una URL

La primera parte de la URL es «http» que, como vimos anteriormente, hace referencia al protocolo que estamos utilizando. La segunda parte, el nombre del recurso, se encuentra separada de la primera por los caracteres «://» que no poseen ningún significado especial, más allá de marcar una división entre el nombre del protocolo y la dirección. Esta última se encuentra dividida a su vez en diferentes etiquetas separadas por puntos. El primer bloque, «www», contiene el nombre del subdominio, que en este caso es el principal. A continuación, encontramos «rama», que es conocido como dominio de segundo nivel y que está estrechamente relacionado con la siguiente división, «es», denominada como dominio de primer nivel. Estas dos últimas partes conforman el dominio en sí mismo y es el nombre que utilizará nuestro navegador para localizar al servidor. Generalmente no es preciso especificar el subdominio cuando se trata del principal, ya que será el propio servidor el que nos redirija hacia el subdominio que figure por defecto. Si introducimos en la barra de direcciones de nuestro navegador la URL *ra-ma.es* seremos reconducidos de forma automática a *www.ra-ma.es*. Solo en algunos casos, y por cuestiones de seguridad, tendremos que introducir la URL completa.

Cuando adquirimos un dominio estamos comprando el derecho a utilizarlo. Si lo que queremos es crear una web, necesitaremos contratar algún plan de hospedaje y vincularlo. Una vez tengamos ambas cosas, técnicamente podremos crear tantos subdominios como deseemos, otra cosa es que las condiciones del plan nos lo permitan. Al acceder a nuestro espacio web, por lo general, encontramos una carpeta con el nombre «www», que albergará nuestra web principal, posiblemente la única que vayamos a necesitar. Un subdominio a nivel físico no es más que una carpeta al mismo nivel que la principal, y a nivel lógico un parámetro DNS. Tener subdominios puede tener mucho sentido, veámoslo en un supuesto. Una compañía que realiza instalaciones desea tener tres webs diferenciadas por cuestiones de marketing. La principal, cuya URL podría ser «www.mi_empresa.com» contiene una web corporativa donde el usuario puede encontrar toda la información sobre dicha sociedad. Además, desean tener un blog de actualidad de su sector que les permita destacar su marca y, al mismo tiempo, captar visitas hacia su web principal, cuya URL podría ser «blog.mi_empresa.com». Y, por último, han pensado en tener un foro de soporte técnico donde solventar las cuestiones de sus clientes, este foro podría tener la dirección «soporte.mi_empresa.com». Las tres webs son claramente identificables por el dominio único y su identidad corporativa (línea de diseño), pero físicamente están ubicadas en diferentes carpetas por lo que ganamos en orden y seguridad. Por otro lado, será más fácil que el usuario localice en la Web el recurso que busca. Otro elemento que tampoco necesitamos introducir al teclear una URL es el nombre del archivo de inicio. Cuando nuestro navegador realiza la petición, el servidor muestra la página por defecto, que en la mayoría de los casos es «index.html» o «default.html».

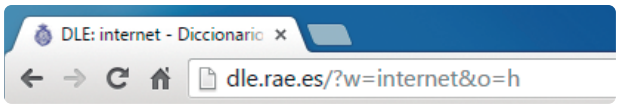

Figura 1.6. Detalle de la barra de dirección del navegador Google Chrome

Si accedemos a una web desde un buscador, o bien estamos navegando por ella, es muy posible que la URL se presente larga y compleja. Generalmente esto se debe a dos factores. Por un lado, a medida que nos movemos por la web vamos entrando en las diferentes carpetas en que esta ha sido dispuesta, revelándonos en la URL la ruta hasta el punto en que nos hayamos separando el nombre de cada carpeta con una barra inclinada «/» de forma muy similar a como lo hace nuestro sistema operativo. Por otro lado, cuando realizamos interacciones en la web, tales como filtrar el listado de los productos de un comercio electrónico en base a una marca, modelo o color, esos valores, a menudo son enviados al servidor a través de la URL. Veamos este último caso a través de un ejemplo. Si entramos en la web de la Real Academia Española *rae.es* y buscamos el significado de la palabra «Internet» a fecha de hoy, la URL que mostrará nuestro navegador será *http://dle.rae.es/?w=internet&o=h*. La primera parte, con el texto «dle», nos está indicando que hemos sido redirigidos a un subdominio secundario con dicho nombre y, por tanto, a una carpeta diferente de la principal. La segunda y la tercera parte las conforma el dominio *rae.es*. A continuación, encontramos una barra inclinada que, en este caso, hace de cierre y tras la cual aparece un símbolo de interrogación «?» que es utilizado en las URL para separar la ruta de los parámetros que pasamos al servidor. De hecho, es fácil adivinar que «w=internet» está constituido por la asignación del valor «internet» a la variable «w». A pesar de que la información que pasamos al servidor no ha de ir necesariamente vía URL y que en todo caso puede ir oculta o codificada, lo cierto es que es frecuente encontrar variables concatenadas con el símbolo «&», como podemos observar en este caso en la parte final, «o=h».

1.1.4 Los navegadores

Como hemos visto con anterioridad, el primer navegador fue desarrollado en 1990 por Berners-Lee, considerado como «padre» de la Web. Desde entonces hasta hoy hemos asistido a grandes transformaciones e incluso a polémicos enfrentamientos entre algunas compañías desarrolladoras. Tras el de Berners-Lee, el primer navegador que alcanzó cierta repercusión social fue Mosaic, cuya primera versión fue liberada en 1993. En un principio solo corría sobre UNIX (sistema operativo cuyo núcleo fue clonado por Linus Torvalds en 1991 para proponer Linux), pero pronto aparecieron versiones para Windows y Macintosh. En 1994 entró en escena Netscape Navigator quien, en apenas un año, superó a Mosaic en prestaciones y popularidad. Microsoft decidió entonces comprar Mosaic, darle un nuevo impulso y lanzarlo como la primera

versión de Internet Explorer, acompañando a Windows 95. A partir de ese preciso instante se inició una carrera entre Microsoft y Netscape Communications por ofrecer un navegador con mejores características que el del competidor, independientemente de si interpretaba el código HTML conforme a los estándares o no. Obviamente no se trataba de construir la Web, sino más bien de dominar el mercado. Paradójicamente ambos navegadores eran gratuitos, y es que el negocio no estaba tanto en su distribución como en el control futuro de la navegación. Este período fue tan encarnizado que dio en llamarse la guerra de los navegadores o *Browser Wars*. Finalmente, Microsoft puso en marcha una agresiva política empresarial que terminó con la derrota de Netscape en 1997. Ese mismo año, Netscape liberó el grueso del código bajo licencia libre, dando lugar al proyecto Mozilla. A pesar de los esfuerzos invertidos en el proyecto, Mozilla hubo de empezar de nuevo y recorrer un largo camino hasta que en 2002 viera la luz la primera versión de lo que dieron en llamar Mozilla Firefox. Un año después aparecía Safari, el navegador de Apple, que se hizo inmediatamente con el mercado de los Macintosh y con un reducido sector de Windows. Dos años más tarde, fruto de una rama de Mozilla, se presentó Firefox con nombre propio y con toda la intención de comerle terreno a iExplorer, objetivo que alcanzó con gran éxito llegando incluso a superarle. Sin embargo, el privilegio de vencer de una forma contundente al navegador de Microsoft estaría reservado a Google Chrome, cuya primera versión vio la luz a finales de 2008. Desde entonces hasta ahora, el pulso entre los navegadores más destacados no ha cesado.

En 2015 y tras encajar el golpe, Microsoft vuelve a la carga con un nuevo navegador: Edge. Algunos autores consideran que nos encontramos ante una segunda guerra de navegadores, aunque lo cierto es que el fragor de la batalla no es comparable al que protagonizaron en el pasado, por lo que tal vez sea más adecuado hablar de libre competencia. La llegada de los dispositivos móviles ha diversificado los frentes y ha dado cierto protagonismo a firmas de popularidad residual como es el caso de la noruega Opera *software*.

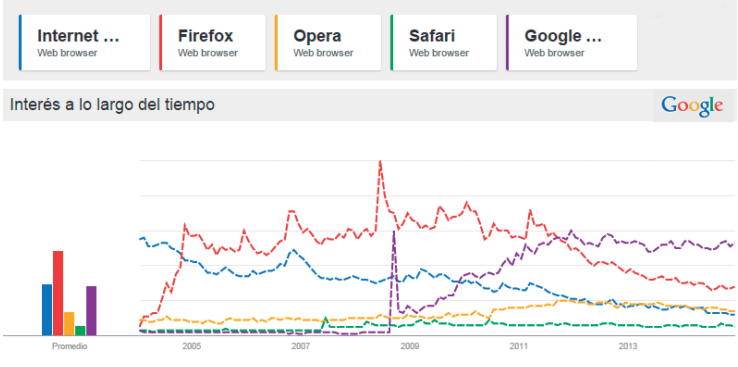

Figura 1.7. Popularidad de los navegadores desde 2004 en función del interés suscitado como tema de búsqueda, según Google Trends

A pesar de las diferencias, los navegadores son muy similares entre sí. Esencialmente, son aplicaciones cliente que permiten la comunicación con servidores web mediante el protocolo HTTP, cuya misión principal reside en componer, interpretar y visualizar documentos HTML junto a sus recursos vinculados. Todos ellos comparten elementos tales como la interfaz de usuario, el gestor de comunicaciones, el motor de renderizado, el sistema de almacenamiento de datos o el intérprete JavaScript.

La inmensa mayoría de los navegadores actuales respetan los estándares web, algo ciertamente muy positivo puesto que el objetivo de diseñadores y desarrolladores consiste en lograr que sus creaciones se visualicen adecuadamente en los distintos navegadores y dispositivos existentes. A pesar de esto, persisten pequeñas diferencias de ejecución, lo que se traduce en la necesidad de realizar ajustes en los proyectos web para lograr los mejores resultados. Uno de los factores que más influyen en esta disparidad de criterios en la interpretación de código es el mencionado motor de renderizado. Los cuatro motores más extendidos son Blink, Gecko, WebKit y Trident, a los que habría que añadir EdgeHTML empleado por Microsoft Edge.

Figura 1.8. Los cuatro motores de renderizado más extendidos en la actualidad

- Blink está basado en un proyecto anterior, WebKit, aunque hoy está siendo desarrollado de forma independiente por parte de Google y es utilizado por los navegadores Chrome y Opera, entre otros.
- Gecko es un motor de código abierto que fue desarrollado inicialmente por Netscape, hoy lo podemos encontrar en Firefox.
- WebKit es otro motor de código abierto, fue desarrollado por Apple, Google y Nokia junto a un nutrido conjunto de compañías, Safari es el navegador más destacado que lo incorpora.
- Trident fue creado por Microsoft, aunque es utilizado por numerosos navegadores entre los cuales están, obviamente, Internet Explorer y Windows Mobile.

Existe una cantidad enorme de navegadores y para hacernos una idea basta con realizar una sencilla búsqueda en la Web. Si nos acercamos a un conjunto más o menos representativo, veremos cómo la inmensa mayoría presenta una interfaz de usuario muy similar: barra de direcciones y botones de avance, retroceso, inicio, cancelación de la carga y recarga. Curiosamente, estos elementos no obedecen a ninguna especificación sino a la experiencia de los usuarios en la navegación.

La rapidez y la eficiencia son factores muy valorados en un navegador, aunque hoy en día exigimos además una serie de prestaciones que resultan casi indispensables, tales como el soporte de múltiples pestañas, la restauración de sesión, la gestión de marcadores e historial, la protección *anti-phising*, la limpieza de información privada, la posibilidad de incorporar extensiones o el cambio del aspecto visual mediante la elección de «temas». La forma en que se contemplen todos estos aspectos en un navegador, influyen a la hora de decantarnos por el uso de uno u otro. En todo caso, y puesto que tenemos en mente llevar a cabo un proyecto web, es fundamental que nos aseguremos de que nuestro trabajo se visualiza de la mejor manera posible en el mayor número de navegadores y dispositivos diferentes. Para verificarlo, la opción más eficaz es utilizar un emulador de pantalla como Screenfly de Quirktools.com o Device Mode de Chrome.

Por otro lado, si somos desarrolladores o editamos código de terceros y queremos que este cumpla con las especificaciones oficiales, debemos visitar la página web del W3C (*World Wide Web Consortium*) donde encontraremos documentación precisa acerca de los estándares HTML y CSS entre otros. El W3C nació en 1994 con el propósito de estandarizar los protocolos web mediante la publicación de recomendaciones. Hoy, este organismo internacional no solo cuenta con el reconocimiento de toda la comunidad, sino que, sin su existencia, la Web que todos conocemos sería muy difícil de imaginar.

1.2 ARQUITECTURA DE LA WEB

Internet es por definición una red de igual a igual, a pesar de lo cual, la Web, uno de sus principales servicios, se basa en el paradigma cliente-servidor. Puede resultar un tanto paradójico, sin embargo, si realizamos una pequeña aproximación a su arquitectura, comprobaremos que este modelo es el más idóneo para la Web. Cuando accedemos al contenido de una página web, como hemos visto anteriormente, realizamos una conexión con el servidor que la alberga y es este quien nos la envía. Por tanto, establecemos una relación cliente-servidor donde nosotros desarrollamos el rol de cliente. Conocer el funcionamiento de este modelo es de capital importancia para poder entender el reparto de procesos que existe entre el cliente y el servidor, y en resumidas cuentas, para poder entender el funcionamiento de la Web.

1.2.1 Servidores web

Un servidor es un nodo que brinda servicios a otros nodos dentro de una red informática. Aunque generalmente cuando nos referimos a este modelo solemos estar pensando en ordenadores de diferente naturaleza, en realidad, lo que convierte a un equipo en cliente o en servidor es el *software* que está instalado en él. Este es el motivo de que nos refiramos a los navegadores como clientes web, de que hablemos de clientes FTP o de clientes de correo y, por otro lado, de servidores web, servidores FTP o servidores de correo, por citar algunos de los servicios más comunes. En función de la naturaleza de los servicios que presta un servidor, este es denominado de una u otra forma, aunque lo más común es que un servidor preste diferentes servicios al mismo tiempo.

Cuando hablamos de servidores solemos imaginarnos enormes máquinas ubicadas en algún lugar desconocido del planeta, bajo enormes medidas de seguridad. Aunque lo cierto es que esta imagen no es descabellada, en la práctica, todo ordenador que se conecta a Internet es susceptible de convertirse en un servidor, solo precisa del *software* adecuado. Si quisiéramos, podríamos convertir nuestro equipo doméstico en cuestión de minutos en un servidor al que potencialmente se pudiera conectar cualquier persona desde cualquier parte del mundo. Entonces nuestro equipo haría el papel de servidor y quienes se conectasen harían el de cliente. Si instalamos *software* servidor web, tendremos que ubicar en nuestro disco duro al menos una página web, y los usuarios que se conecten, deberán hacerlo mediante el uso de una aplicación cliente web, es decir, un navegador. Pero también podríamos ejecutar en nuestro equipo, por ejemplo, una aplicación servidor FTP de modo que todos aquellos usuarios que contasen con un cliente FTP, como Filezilla, podrían acceder a nuestro disco duro y descargarse los archivos que a tal propósito hubiésemos ubicado.

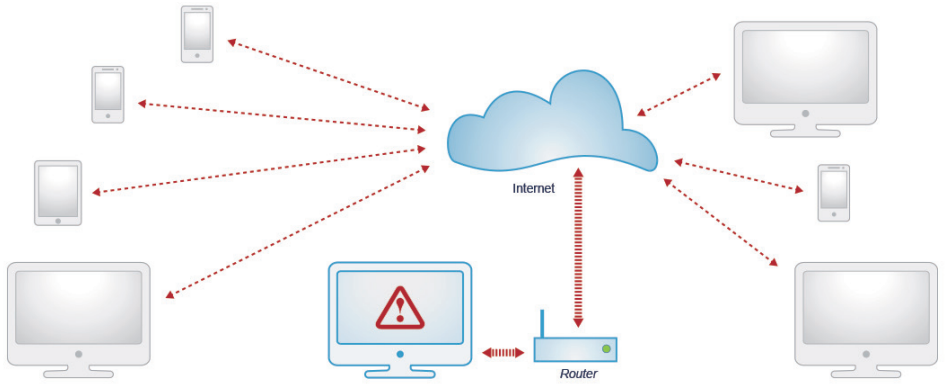

Figura 1.9. La ejecución de software cliente en un equipo doméstico puede producir un cuello de botella

Existe *software* servidor de carácter gratuito y es muy sencillo ponerlo en marcha, sin embargo, vamos a citar tres motivos esenciales por los que no sería una buena idea poner nuestro equipo a disposición de la comunidad. En primer lugar, nuestro ordenador, por muy potente que sea, posee unas características que no están enfocadas a soportar un número elevado de conexiones simultáneas y, menos aún, a que éstas hagan un uso concurrente de sus recursos. Aunque tan solo se recibiesen unas pocas conexiones, nuestro equipo quedaría seriamente comprometido, pudiendo caerse o prestar un servicio ineficiente. En segundo lugar, en nuestro domicilio contamos con un ancho de banda muy limitado, de modo que, si desde otros equipos comunicasen con el nuestro a un mismo tiempo, muy pronto congestionarían nuestra conexión a Internet, causando una acusada ralentización o incluso bloqueándola. Por si los dos puntos anteriores no nos han disuadido suficientemente, tenemos un tercero. Pensemos que, para prestar cualquier tipo de servicio en la Red, uno de los requisitos esenciales es que este permanezca disponible en todo momento. Esto significa que nuestro equipo ha de estar funcionando, conectado y con la misma IP de forma permanente.

Por estos y otros motivos recurrimos al alquiler de servidores para alojar nuestras páginas web. Estos equipos no son necesariamente más rápidos que los nuestros, pero están diseñados específicamente para desarrollar el rol de servidor y, por eso, están optimizados para soportar accesos masivos y poseen un ancho de banda infinitamente superior al que podamos tener en nuestras casas. Si queremos verlo a través de una metáfora, podríamos comparar un coche con un autobús: el primero tal vez corra más, pero desde luego nunca será capaz de transportar a más gente que el segundo. Además, los equipos concebidos para desarrollar el rol de servidor web poseen ciertas características que les permiten estar en funcionamiento de forma ininterrumpida incluso en el caso de que falle la fuente de alimentación o el disco duro. Estos equipos no son tan grandes como quizás podamos imaginar, sin embargo y puesto que generalmente se disponen en *rack*, es decir, apilados formando pequeñas torres, su visión en conjunto sí es ciertamente impresionante.

1.2.1.1 PARADIGMA CLIENTE-SERVIDOR

Puede parecer una perogrullada, pero, hacer que dos ordenadores conectados en red se pongan en contacto no es nada fácil. Imaginemos que tenemos dos equipos entre los que deseamos establecer una conexión a través de una aplicación. Si la ejecutamos en el primer equipo, esta enviará una señal de conexión que no encontrará ninguna respuesta y, por tanto, concluirá su proceso. Si después la ponemos en marcha en el segundo equipo, ocurrirá lo mismo, puesto que el primero no responderá. Esto es lo que se conoce como el dilema del *rendezvous* o rendibú. Podemos pensar que la solución pasa por hacer que la aplicación no desfallezca tras el primer intento y que, en su lugar, reitere de forma sistemática el envío de señales. Sin embargo, aunque ambos equipos

se envíen mutuamente señales continuamente, las probabilidades de que ambas lleguen y que no colisionen, serían insuficientes. La solución a nivel lógico es muy sencilla en realidad: basta con que uno de los dos permanezca a la escucha indefinidamente, de este modo, cuando el otro envíe una señal de conexión, encontrará la respuesta que busca. Esta solución es, en esencia, el paradigma cliente-servidor.

Figura 1.10. La solicitud de un cliente será atendida solo si el servidor permanece a la escucha

En realidad, este modelo nos es familiar, puesto que se percibe a través de nuestra propia experiencia como usuarios de la Web. Según esta visión podríamos concebir la Web como un sistema donde los equipos que se encuentran conectados han de pertenecer a una de estas dos categorías: servidores o usuarios. Los segundos accederían a los primeros, que serían los que asumirían la carga del trabajo, mientras que los usuarios se limitarían a recibir la información solicitada. Las cosas no son exactamente así. El paradigma cliente-servidor, sobre el que se apoya la Web, es un modelo distribuido en el que la carga de los procesos se reparte entre ambos roles. No olvidemos que se trata de una solución que permite la comunicación entre diferentes dispositivos donde todos juegan un papel relevante. Como individuos somos usuarios de la Red, pero desde el momento en que abrimos un navegador, nos convertimos en un nodo cliente con capacidad de procesar datos. Sabemos que los roles se definen a nivel de *software*, independientemente del *hardware*. Esto tal vez nos sugiera una cuestión: ¿podría nuestro ordenador interpretar ambos papeles al mismo tiempo? La respuesta es sí, de forma taxativa. Es algo que puede parecer carente de sentido pero que, como veremos más adelante, puede resultar de gran utilidad para ciertos propósitos.

Las aplicaciones que permiten jugar un rol determinado están basadas en los diferentes estándares que conforman la familia de protocolos TCP/IP. Estos protocolos fijan las normas que gobiernan el intercambio de paquetes de bits y, por tanto, describen cómo es localizado el equipo de destino y por qué puerto ha de entrar, entre otros muchos aspectos. Un puerto lógico es un canal de comunicación que puede estar abierto o cerrado. Para que la comunicación sea efectiva es necesario que esté abierto y que haya una aplicación escuchando en el puerto por donde llegan los datos, de otro modo serán ignorados. Aunque los puertos existentes pueden ser utilizados generalmente por diferentes aplicaciones de forma arbitraria, existen ciertos servicios que poseen un puerto específico asignado por defecto. Es el caso del protocolo HTTP asociado al puerto 80, el protocolo FTP al 21, el SMTP al 25, el POP3 al 110 y el Telnet al 23, entre otros muchos.

Podría parecer que esta arquitectura hace muy vulnerable a la Web convirtiéndola en un gigante con pies de barro, pues basta con que se caiga un servidor para que todos sus contenidos sean inaccesibles. No obstante, debemos tener en cuenta que éstos, por regla general, se encuentran en centros de proceso de datos (*Data Center*) donde existe una monitorización constante en el marco de una alta seguridad que contempla las diferentes eventualidades que pueden surgir. Además, existen centros de respaldo con información redundante ideados para reemplazar a un centro asociado en caso de emergencia. Como sistema, la Web no es inexpugnable, pero es más robusta de lo que podamos pensar.

1.2.1.2 WEBS ESTÁTICAS Y WEBS DINÁMICAS

Cuando introducimos una URL en la barra de direcciones de nuestro navegador estamos haciendo una petición implícita a un servidor DNS que se encargará de atenderla, traduciendo la URL a una dirección IP, que será utilizada por el navegador para localizar el servidor que alberga el recurso que buscamos. Una vez que nos hemos comunicado con el servidor que hospeda la web que buscamos, y en función de las tecnologías con que esta se haya desarrollado, se llevarán a cabo diferentes procedimientos. En base a estos últimos podemos dividir las páginas web en dos categorías: estáticas y dinámicas.

Las primeras están desarrolladas en HTML casi exclusivamente, pudiendo incorporar algún código de *script* y tener vinculado todo tipo de contenido. En este caso, tras nuestra solicitud, el servidor transfiere el documento al navegador, así como los contenidos asociados, tales como hojas de estilos CSS, imágenes, audio o vídeo. Se harán las solicitudes pertinentes hasta recibir todo el material (lo que puede implicar peticiones a servidores de terceros), tras lo cual, nuestro navegador procederá a interpretar el código HTML y a ejecutar en el mismo instante cualquier código que, como JavaScript, pueda contener. En la medida en que se vaya componiendo la página, esta se irá montando en nuestra pantalla.

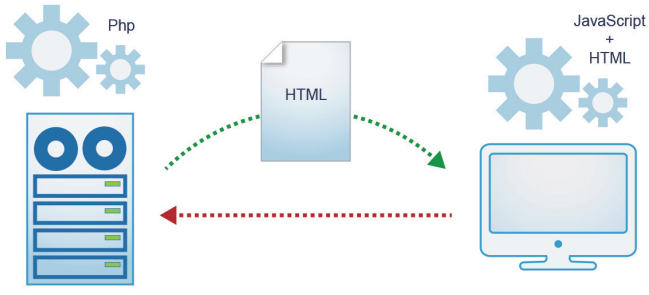

Figura 1.11. Las páginas web dinámicas son generadas y enviadas al cliente en tiempo real

Las páginas web dinámicas son aplicaciones más complejas en su desarrollo y en su proceso de ejecución, pero aportan un potencial enorme, sin el cual, la Web sería muy pobre y no podría brindar muchos de los servicios que hoy demandamos. Para conocer la mecánica de esta tecnología, pondremos un supuesto. Tenemos la intención de realizar una reserva de hotel. Para ello, introducimos la URL en el navegador. Un servidor DNS devuelve la IP y, de este modo, conectamos con el servidor que alberga la web del hotel. Hasta aquí no se aprecia ninguna diferencia destacable, es en el momento de hacer la reserva cuando todo cambia. La web no nos puede indicar si hay habitaciones libres en la fecha seleccionada sin comprobar previamente en su base de datos la disponibilidad en tiempo real. Para ello, ha de ejecutar un código en el servidor que establecerá una conexión con la base de datos y realizará la consulta. Este código estará escrito en algún lenguaje de alto nivel, como PHP, ASP o Ruby. Hecha la consulta, se traducirá el código de respuesta a HTML para finalmente, hacerlo llegar hasta nuestro equipo junto con todos los elementos que compongan la web. Esa traducción supone, en realidad, la creación de páginas HTML de forma dinámica, puesto que no existían previamente y son generadas para dar respuesta a una solicitud específica. Una vez recibida la página, el navegador se encargará de procesar los archivos para mostrárnosla en pantalla. Como es natural, el proceso no terminaría aquí. Si queremos hacer una reserva, tras fijar la fecha, tendremos que indicar algunos datos más y, por último, seremos conducidos a la pasarela de pago.

Lo interesante de todo esto es que, además de hacernos una idea de cómo funciona el reparto de procesos, hemos identificado a los actores principales de este sistema: *software* servidor, lenguaje de alto nivel y base de datos.

1.2.1.3 PARADIGMA AMP

La programación del lado del servidor consiste en el procesamiento de la petición de un nodo cliente mediante la interpretación de un *script* en el servidor web, lo que permite generar páginas HTML de forma dinámica, con información específica y en tiempo real. Todo *script* que se ejecuta en un servidor web se conoce como *server-side scripting* o secuencia de comandos del servidor. Esta tecnología ha evolucionado de forma muy notable con el devenir de la Red, ofreciendo hoy diferentes alternativas entre las que destaca el paradigma AMP, por su enorme popularidad y potencial.

Las siglas AMP se corresponden con las iniciales de tres proyectos de *software* que, aunque fueron concebidos de forma independiente, trabajan juntos como un sistema capaz de servir contenido web. Estos tres pilares de naturaleza libre y gratuita, han servido de plataforma para el desarrollo de múltiples proyectos de código abierto de gran éxito, como WordPress, Joomla o Drupal. Estamos hablando

de Apache, MySQL y PHP. Apache es una aplicación servidor HTTP multiplataforma y es capaz de convertir a un equipo cualquiera en un servidor web preparado para dar respuesta a las solicitudes que le sean realizadas desde un navegador. MySQL es un sistema gestor de bases de datos, relacional, multiusuario y multihilo, que almacena todo aquello que nos es mostrado mediante el lenguaje PHP. Éste último, es un lenguaje de programación de alto nivel, uso generalista, muy flexible y potente, que se ejecuta en el servidor donde genera páginas HTML de forma dinámica. Este modelo también es empleado en menor medida con lenguajes como Python o Perl en sustitución de PHP, manteniendo su nombre, pues el acrónimo no cambia.

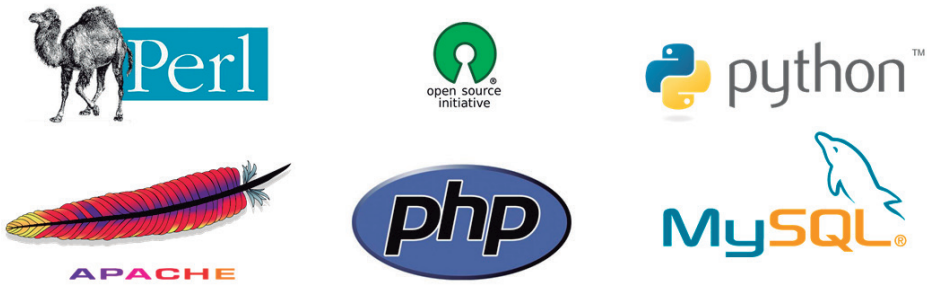

Figura 1.12. Logotipos de los principales proyectos que conforman el modelo AMP

Es posible que encontremos documentación sobre este modelo bajo el nombre de LAMP. La ele inicial hace referencia a Linux, el sistema operativo sobre el que corrían estos tres proyectos inicialmente. De hecho, estas aplicaciones vienen preinstaladas en muchas de las distribuciones Linux. Sin embargo, hoy podemos ejecutarlas sobre Windows o sobre Mac, por lo que también encontraremos proyectos tales como WAMP o MAMP, e incluso algunos que cuentan con versiones para las tres plataformas, como XAMP o Ampps. En todo caso, el paradigma AMP es común en todos ellos.

1.2.2 Servidores DNS

Si realizamos la configuración de nuestro *router* de forma manual, tendremos que introducir dos números que nos habrá facilitado nuestro proveedor de Internet (o ISP). Se trata de dos direcciones IP, dos valores numéricos divididos en cuatro

bloques y separados por puntos que sirven para identificar a dos servidores DNS. Si nos conectamos a una wifi y no tenemos que especificar ningún valor, es porque el protocolo de configuración dinámica (DHCP) nos ha asignado de forma automática una IP y dos direcciones de sendos servidores DNS. Como usuarios no percibimos su funcionamiento, a pesar de ser una pieza clave del sistema. Los servidores DNS (*Domain Name System*) son los encargados de traducir la URL que introducimos en nuestro navegador a la IP asociada, de modo que sea posible localizar el servidor web que alberga los recursos a los que deseamos acceder.

1.2.2.1 SISTEMA DE NOMBRES DE DOMINIO

Como sabemos, todos los dispositivos que se conectan a Internet han de estar identificados por una IP única que se comporta como un número de teléfono: si se marca, la llamada será conducida a través de la red telefónica hasta el terminal correspondiente. Sin embargo, nosotros empleamos las URL para navegar y no las direcciones IP. El motivo es obvio, resulta mucho más sencillo memorizar el dominio *rae.es* que su IP: «192.145.222.100». Además, debemos tener en cuenta que a lo largo del día podemos llegar a visitar un conjunto de webs bastante importante y resultaría poco práctico tener que manejar colecciones numéricas en lugar de texto. Pero hay otra razón de igual o mayor peso. Si, por ejemplo, la Real Academia Española, por cualquier motivo desease cambiar su web de servidor, lógicamente cambiaría también su IP, pero no su URL, que seguiría siendo la misma. De lo que extraemos que, navegar mediante el uso de URL no solo hace más cómoda y eficaz a la Web, también la hace más flexible y estable.

Hemos mencionado la existencia de servidores DNS en Internet al describir los procesos que tienen lugar cuando solicitamos la visualización de una web. Los hemos identificado como intermediarios entre la URL y la dirección IP del servidor que aloja el recurso al cual deseamos acceder, pero, ¿qué son realmente?

El DNS o sistema de nombres de dominio, es un sistema de nomenclatura jerárquica para los diferentes recursos que se encuentran conectados a Internet. Podemos concebirlo como un conjunto de tablas de equivalencia entre los dominios y las direcciones IP, como si se tratase de una agenda de teléfonos donde recogemos de forma asociativa el nombre y el teléfono de cada registro. Técnicamente hablando, dichas tablas, son en realidad una inmensa base de datos distribuida que se encuentra replicada en cada uno de los nodos de una red de servidores ubicados a lo largo del planeta y que se actualizan entre sí de forma ininterrumpida mediante conexiones programadas. Gracias al sistema DNS obtenemos la IP del dominio que hemos introducido de forma casi inmediata e imperceptible para nosotros.

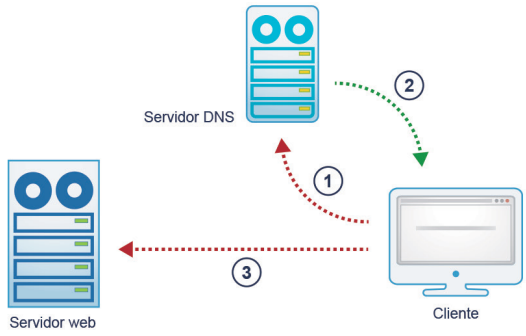

Figura 1.13. Los servidores DNS juegan el papel de intermediarios entre las URL y los servidores que albergan los recursos. Al introducir una URL en el navegador, este realiza una conexión automática con un servidor DNS [1], del que obtiene la IP del servidor correspondiente [2] para, finalmente, establecer una conexión con este último [3]

Cuando registramos un dominio y lo vinculamos a un servicio de hospedaje, se nos informa de que este no estará disponible generalmente hasta pasadas 24h o 48h, que es el tiempo estimado para la propagación de las DNS. Durante esa espera, nuestro dominio se encontrará ilocalizable, perdido en el limbo de Internet. Esto nos puede llevar a preguntarnos si existen recursos en Internet cuyas direcciones no se encuentren registradas y que, por tanto, permanezcan aparentemente inaccesibles. La respuesta es sí, por supuesto. Existen varios proyectos de DNS alternativos, entre los que destaca OpenNIC, ofreciendo acceso a una Red independiente al margen de la oficial.

1.2.2.2 REGISTRO DE DOMINIOS

Antes incluso de que Internet se conociese como tal, ya se había considerado la necesidad de establecer algún tipo de regulación sobre la adquisición de dominios. Esta responsabilidad que fue asignada en primera instancia al Network Information Centre (NIC) dependiente del departamento de Defensa del gobierno estadounidense, hoy es gestionada por la Internet Corporation for Assigned Names and Numbers (ICANN) dependiente del departamento de comercio del mismo gobierno. La ICANN es una compleja organización constituida como corporación sin ánimo de lucro y considerada de utilidad pública que, entre otros cometidos, tiene la potestad de acreditar a aquellas empresas que deseen explotar la venta de dominios y cumplan con los requisitos establecidos por esta entidad. Huelga decir que este sistema no cuenta con la aprobación unánime de la comunidad internacional de usuarios de Internet, a pesar de lo cual, hoy por hoy es una solución difícil de eludir. Las compañías registradoras acreditadas pujan entre ellas ofreciendo un mismo servicio en un mercado abierto, motivo por el cual, lo más frecuente es encontrar paquetes

que incluyen otros servicios relacionados, tales como espacio web o buzones de correo. A estas empresas se les unen aquellas que se dedican a la reventa, una actividad que lejos de ser ilícita está perfectamente consentida, a pesar de que éstas últimas no precisen cumplir condición alguna, de hecho, cualquier persona o entidad puede hacerse con el control de cualquier dominio que esté libre para ofrecerlo posteriormente al precio que mejor considere, sin que exista límite legal alguno ni penalización de ningún tipo por practicar una especulación que en ningún modo beneficia al crecimiento de la Web. En esta enorme lonja virtual hay sitio para todos, también para los que hacen de la picaresca su *modus vivendi*, por lo que es clave tener muy claro qué es exactamente lo que estamos adquiriendo antes de introducir los datos de nuestra tarjeta de crédito, y para ello, resulta necesario conocer algunos aspectos básicos sobre la naturaleza de los dominios.

Un dominio es un identificador registrado que permite la localización inequívoca de un recurso en Internet y que está compuesto de dos partes separadas por un punto: el nombre de dominio o dominio de segundo nivel y su extensión o dominio de primer nivel. En el caso del dominio *rae.es* diríamos que su nombre o dominio de segundo nivel es «rae» y su extensión o dominio de primer nivel es «es». También podemos encontrar los a menudo mal denominados dominios de segundo nivel, pues son en realidad de tercer nivel: son aquellos que ofrecen direcciones del tipo «nombre.com.es». Por otro lado, es posible encontrar definido como dominio de tercer nivel aquello que, en realidad, es un subdominio: «subdominio.nombre.es».

Existe cierta disparidad de criterios al referirse a estos conceptos, lo que en ocasiones nos puede confundir en cierta medida, por ello, insistimos en la importancia de saber qué es exactamente lo que buscamos, pues solo de este modo podremos hacer una comparativa objetiva respecto de las diferentes ofertas que vamos a encontrar en la Red. Un dominio propio es una pieza fundamental para la creación de nuestra identidad en Internet y no debemos esperar a tener nuestra web terminada para registrarlo, es esencial que lo hagamos cuanto antes. Existen empresas que, como hemos comentado, se dedican a ocupar dominios que quedan libres o que puedan resultar de interés para, más tarde, revenderlos a precios estratosféricos. Por esto es preciso que nos hagamos con la titularidad del dominio que deseamos junto a las extensiones que consideremos de mayor interés, como, por ejemplo: «nombredominio.com», «nombredominio.es», «nombredominio.eu» y «nombredominio.biz», poco importa cuál de ellas usemos después como principal. Por otro lado, debemos pensar en un plan de hospedaje que se ajuste a las necesidades futuras de nuestra web, algo que podemos contratar más adelante y de forma independiente a los dominios o conjuntamente como parte de un pack. En todo caso, el registro de un dominio, sin prestaciones añadidas, es siempre más económico que su adquisición mediante un paquete comercial, aunque la publicidad de algunas compañías nos pueda sugerir lo contrario.

1.2.3 Sistemas de gestión de contenidos (CMS)

Un sistema de gestión de contenidos o CMS es un programa informático que permite la creación y gestión de contenidos a través de una interfaz web. Puede parecer una aproximación un tanto vaga e imprecisa, pero nos permite hacernos una primera idea.

A principios de los años noventa el concepto de sistemas de gestión de contenidos aún no existía como tal, aunque algunas de sus principales funciones empezaban a implementarse como aplicaciones independientes entre sí, sin llegar a constituir un sistema. En 1995 el sitio de noticias tecnológicas CNET lanzó un sistema de administración de documentos y publicaciones que permitía la creación de nuevas páginas a partir de plantillas predefinidas. Un año después, y tras la unión con el proyecto Vignette, nacía StoryServer, la primera herramienta apoyada sobre una base de datos relacional que podemos considerar como gestor de contenidos. Tras este hito, la proliferación de este tipo de productos fue verdaderamente intensa, a pesar de que su alto coste limitaba su ámbito exclusivamente al área empresarial. Habría que esperar hasta el año 2000 para ver los primeros proyectos CMS de código abierto, que en un principio solo estaban disponibles para servidores Linux. Hoy contamos con soluciones de código abierto y de pago, de carácter generalista o específico, emergentes o de largo recorrido, grandes y poderosas o pequeñas y ligeras, podría decirse que existe un amplio abanico de posibilidades para cada uno de los proyectos que tengamos en mente, sean del tipo que sean. Sin duda, profundizar en el concepto nos ayudará a entenderlo mejor y, de este modo, sabremos qué podemos esperar de estos desarrollos.

Toda aplicación orientada al usuario ha de contar necesariamente con una interfaz que permita a este interactuar con ella, independientemente del propósito de la misma. Esta interfaz, generalmente visual, permitirá al usuario realizar aquellas tareas que se hayan contemplado en su desarrollo. Desde el sencillo juego buscaminas hasta el complejo panel de un controlador aéreo, ambos habrán sido creados pensando en la interacción con el futuro usuario. Obviamente esto no convierte a toda aplicación en un CMS, por tanto, la existencia de una interfaz será un requisito necesario, pero nunca suficiente. Para hablar de un gestor de contenidos propiamente dicho han de existir, al menos, dos interfaces orientadas hacia dos perfiles diferentes: usuario y administrador; por eso hablamos de contenidos creados por los administradores a los cuales acceden los usuarios. Es muy probable que el acrónimo CMS ya nos resultase familiar y que lo asociemos inmediatamente a nombres como WordPress, Joomla o Drupal, pero ¿sabemos exactamente lo que es?

1.2.3.1 QUÉ ES UN CMS

Un gestor de contenidos es una parte, o módulo, de una aplicación a través de la cual un usuario acreditado y sin necesidad de poseer conocimientos técnicos, puede crear y modificar los contenidos de la parte complementaria de la misma. En otras palabras, es posible desarrollar cualquier tipo de *software* y dotarlo de una interfaz web para su administración, entonces, nos referiremos a esta interfaz como el gestor de contenidos de dicha aplicación. Sin embargo, es más frecuente hablar de esta capacidad de una forma global, de modo que podríamos entender por CMS toda aplicación web diseñada para llevar a cabo la creación, gestión, publicación y archivado del contenido de un sitio web. Es decir, a efectos de categorización, la capacidad de gestionar contenidos web trasciende por encima de todas las demás a pesar de que existen multitud de proyectos CMS muy diferentes y con las más diversas características y finalidades.

Normalmente, todo CMS está vinculado a una o más bases de datos que registran y almacenan los diferentes componentes que conforman los contenidos, de modo que estos son accesibles, modificables y reutilizables. El flujo de trabajo suele ser configurable, de forma que es posible adaptarlo a nuestras necesidades. Ahora bien, si contemplamos este proceso de forma esquemática, podríamos definirlo de la siguiente manera: un usuario autorizado accede a la interfaz web de administración desde donde modifica los contenidos existentes, tras lo cual, éstos son actualizados en tiempo real quedando accesibles para el resto de usuarios desde la interfaz de acceso a la información.

Figura 1.14. Representación del flujo de trabajo esencial en un CMS: un usuario autorizado accede a la administración a través de una interfaz web, pudiendo actualizar los contenidos en tiempo real

Aunque quizás no sea muy preciso catalogar a un CMS como *framework* o entorno de trabajo, es muy probable que así lo encontremos en más de una ocasión y lo cierto es que, en todo caso, este modo de contemplarlo tiene su sentido. Todas estas aplicaciones nacen con el objeto de simplificar en mayor o menor medida la creación y el mantenimiento de sitios web de cierta complejidad, lo que no quiere

decir que baste instalarlas para hacerlas funcionar. Se trata más bien de un punto de partida desde el cual podemos desarrollar una estructura de contenidos, una apariencia visual desvinculada de éstos y establecer una jerarquía de perfiles de usuario. Esta labor de desarrollo no está exenta de dificultades y, por regla general, son precisos ciertos conocimientos. Un CMS, por tanto, está más cerca de ser una plataforma que una aplicación. De hecho, y a pesar de la facilidad y rapidez con la que se puede llevar a cabo la instalación de algunos proyectos, tras esta, suele quedar mucho trabajo por delante. Esta somera descripción no debe inducir al desánimo en ningún caso, puesto que se trata tan solo de la instantánea generalista de un paisaje lleno de contrastes. A lo largo de estas páginas iremos desentrañando cada uno de los aspectos que debemos conocer para superar con éxito las diferentes etapas que encontraremos a nuestro paso.

Es importante entender que los gestores de contenidos no son la panacea y que, por lo tanto, no serán siempre y en toda circunstancia la mejor opción. Sin embargo, cuando hablamos de aplicaciones web de la complejidad de un comercio electrónico, encontraremos que las facilidades de que provee un CMS superarán con creces a las dificultades que este pueda presentar. Veremos cómo su naturaleza nos sirve de trampolín para llegar de forma amable, rápida y poderosa hasta donde, de otro modo, no alcanzaríamos sino tras meses de duro desarrollo.

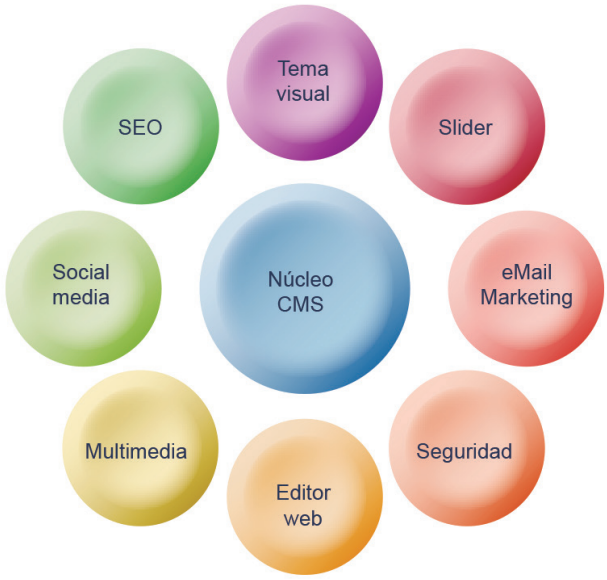

Figura 1.15. Un CMS amplía sus capacidades mediante la adición de nuevos módulos funcionales ya existentes, modificados o desarrollados específicamente para cubrir demandas concretas

A pesar de que las características de un gestor de contenidos pueden variar mucho en función del proyecto en cuestión, a grandes rasgos, sus bondades son inherentes al concepto:

- Permiten la creación de sitios web complejos de una forma mucho más eficiente y asequible. El hecho de contar con una estructura ya existente que posee unas funcionalidades básicas hace que resulte viable la implementación de aplicaciones web de cierta envergadura por parte de compañías medianas y pequeñas, e incluso en algunos casos, por particulares.

- Los tiempos y, por tanto, los costes de desarrollo se reducen drásticamente, por lo que el cliente recibe un *software* más potente, con mejores y mayores características al mismo precio por el que antes adquiría un sitio web básico, estático y no administrable.

- Los productos web finales están basados en una estructura conocida, lo que brinda al cliente la libertad de encargar modificaciones o posteriores ampliaciones, ya sean funcionales o gráficas, a diferentes compañías desarrolladoras que competirán en beneficio del cliente.

- Por lo general, contemplan la gestión de usuarios y permisos de forma dinámica, lo que facilita la creación y administración de diferentes perfiles y su asignación a los usuarios en alta, permitiendo implícitamente definir un flujo de trabajo específico sobre la creación y manipulación de contenidos.

- La edición de contenidos se realiza a través de un editor web, facilitando así a las personas sin especialización técnica la actualización del contenido web, lo que se traduce en una reducción de costes y una gestión de contenidos más adecuada y eficaz. No obstante, en la mayoría de los casos es preciso ofrecer una pequeña formación a los futuros administradores.

- La apariencia visual se encuentra separada del contenido, de forma que es posible introducir nuevos elementos sin realizar maquetación alguna, ya que éstos adoptarán de forma inmediata y automática la línea de diseño establecida en el sitio y, de forma análoga, se puede actualizar el diseño del sitio sin que esto afecte a los contenidos.

- Todos los archivos y contenidos se encuentran vinculados a la base de datos, por lo que realizar copias de seguridad, actualizaciones, cambios en la estructura o en el diseño, resulta más eficiente y seguro.

- La mayoría de los CMS cuentan con una amplia oferta en extensiones que añaden nuevos comportamientos y potentes características a los sitios web.

Estas ventajas son comunes, generalmente, a todo CMS que se encuentre alojado en un servidor que nos brinde acceso a nuestros archivos y datos. Si optamos por las conocidas como *soluciones alojadas* es muy posible que veamos cercenadas algunas de sus características y que tengamos ciertas dificultades para migrar nuestro proyecto a otro servidor.

1.2.3.2 TIPOS Y CARACTERÍSTICAS COMUNES

Sería verdaderamente complicado hacer una relación clasificada mínimamente rigurosa de los diferentes proyectos CMS existentes dado que, las funcionalidades de los diferentes paquetes suelen evolucionar internándose en otras áreas anexas y, además, la instalación de extensiones puede modificar en gran medida su enfoque; por otro lado, el enorme número de proyectos hace improbable semejante tarea. Tampoco sería muy preciso hacer una enumeración de tipos o categorías, pues no se ajustaría a la realidad existente, puesto que en la práctica las líneas divisorias son difusas o inexistentes.

Sin embargo, hecha esta salvedad, sí podemos citar algunos de los tipos más mencionados en Internet, aunque como apuntamos, es muy frecuente encontrar aplicaciones CMS cuyas características no obedecen a dicha clasificación.

- **WCM** (*Web Content Management*). Gestores de contenido para sitios web de propósito general. Proveen de las facilidades propias de un CMS a fin de permitir la gestión de contenidos vinculados a una base de datos a través de una interfaz web.

- **ERP** (*Enterprise Resource Planning*). Se trata de complejos sistemas de planificación de recursos empresariales. Esta opción ha permitido a muchas empresas abaratar costes en su gestión y eliminar de su estructura informática la figura del servidor local, menos seguro y versátil. Además, y al igual que cualquier otro tipo de CMS, es accesible desde cualquier dispositivo con conexión a Internet, lo que multiplica su potencial y facilita su mantenimiento.

- **CRM** (*Customer Relationship Managment*). Son potentes herramientas de marketing relacional. Representan un pilar fundamental sobre el que se puede sustentar toda la estrategia de negocio de una compañía que tenga como objetivo establecer un contacto más cercano con sus clientes.

- **Blogs** (*web logs*). Estos están orientados a la publicación de artículos siguiendo un formato de apilamiento cronológico, donde la última entrada está siempre sobre las demás. Naturalmente admite contenidos

de todo tipo e incluso existen categorías derivadas como los *vídeo-blogs* o *vlogs* donde el medio comunicativo principal es el vídeo.

- **LCMS** (*Learning Content Management System*). Permiten la creación de comunidades de aprendizaje, existiendo un amplio margen para definir diferentes enfoques y metodologías muy diversas. Puede contemplarse como un complemento a las clases presenciales o directamente como plataforma de formación virtual. Entre sus características más destacadas se encuentra la gestión de recursos, el seguimiento de la actividad de los alumnos y la administración de las áreas comunes como foros o chats.

- **Wikis** («rápido» en hawaiano). Facilitan la creación de un espacio abierto y compartido cuyos contenidos son desarrollados por diversos usuarios. Este entorno posibilita una actualización rápida y sencilla de contenidos.

- **Redes sociales**. Existe un universo lleno de posibilidades más allá de Facebook, Twitter o LinkedIn. Diferentes enfoques, funcionalidades o temáticas más específicas, han facilitado la proliferación de todo tipo de redes. Éstos CMS permiten crear comunidades a la carta añadiendo características mediante la instalación de módulos.

- **Comercio electrónico**. Los CMS se han convertido en una alternativa real frente a otros desarrollos gracias al altísimo nivel alcanzado durante los últimos años, cubriendo con eficacia y robustez aspectos tales como la gestión de almacén, los pedidos, los envíos, la facturación o los modos de pago. El grado de flexibilidad y escalabilidad que ofrecen, suele ser suficiente como para adaptarse a un enorme abanico de modelos de negocio diferentes permitiendo, además, cualquier adaptación visual o funcional para adecuarse al devenir del comercio. En algunos casos, un único módulo puede convertir a un CMS de carácter generalista en un comercio orientado a la web.

- **Proyectos**. Representan una herramienta casi indispensable para la planificación y el desarrollo de cualquier proyecto por parte de un equipo, ya sea de modo local o remoto. Proveen de útiles herramientas que facilitan la comunicación, el acceso a la documentación, la programación de tareas, y la monitorización de la evolución con estadísticas e informes.

- **Foros**. Son posiblemente el formato de intercambio de información más antiguo y conocido de la Red, un sistema sencillo y eficaz que ha preservado intacta su popularidad a lo largo del tiempo. Aunque existen diversas alternativas para su implementación, si el foro es la razón de ser del sitio, un CMS puede ser la opción que ofrezca más posibilidades.

Contamos con una oferta tan grande como diversa, cuya magnitud puede incluso resultar un tanto abrumadora en un principio, a pesar de lo cual, debemos tener siempre presente que este escenario juega muy claramente a nuestro favor. A pesar de las grandes diferencias que podemos encontrar entre los CMS, existen ciertas características que comparten la gran mayoría de ellos. Independientemente de si han sido desarrollados por una comunidad altruista o por una compañía con ánimo de lucro, casi con toda seguridad, cumplirán los siguientes puntos:

- Su desarrollo se ha llevado a cabo en un **lenguaje de alto nivel** que será interpretado en el lado del servidor, generando páginas HTML de forma dinámica que, a su vez, serán ejecutadas por el navegador del cliente.
- Se encuentra estrechamente vinculado a una **base de datos** que registra los contenidos, los diferentes usuarios y la propia estructura del sistema.
- Cuenta con herramientas que permiten al desarrollador dotar de navegabilidad al sitio mediante la creación de una **estructura de menús** que facilitarán al usuario el acceso a los diferentes apartados y contenidos.
- La existencia de **secciones y categorías** facilitan la creación de una estructura sobre la cual se han de articular los diferentes contenidos de forma coherente y ordenada, lo que posibilita su rápida localización y su correcta indexación por parte de los robots de búsqueda.
- Está dotado de un sistema de «temas» que contienen el código que otorga al sitio web una determinada **apariencia visual**. Estos «temas» o plantillas pueden ser de tipo comercial o gratuito, pueden personalizarse o activarse tal cual, aunque los mejores resultados se obtienen cuando el tema es desarrollado desde cero de acuerdo a unas necesidades específicas.
- Posee dos interfaces de usuario claramente diferenciadas: la **parte pública** o *front end* donde se encuentran los contenidos y la **parte de administración** o *back end* desde donde se crean y gestionan estos.

A pesar de que existen CMS de *software* propietario, lo más común es encontrar estas aplicaciones liberadas bajo diferentes licencias de código abierto, lo que favorece su crecimiento y abre las puertas al desarrollo de nuevos módulos por parte de terceros, lo que a su vez proporciona una notable escalada en su potencial.

Sea como fuere la fórmula elegida para el desarrollo de un proyecto CMS, la comunidad que se crea en torno a él es sin duda la piel a través de la cual transpira, la que lo hace amable y accesible para los usuarios, la que, en última instancia, determinará el éxito o el fracaso de un determinado *software*.

1.2.3.3 CÓDIGO ABIERTO

Open source o código abierto es una expresión que nació con la pretendida intención de sustituir con carácter nominativo al movimiento conocido como *software* libre o *free software*. En principio tan solo se trataba de acabar con el error al que históricamente ha inducido el doble significado que la palabra *free*, entendiendo muchos, en este caso, como *software* gratuito lo que en realidad quería decir *software* libre. Algunas comunidades no aceptaron la nueva designación, por considerar que se prescindía de una palabra clave para describir la filosofía que encierra este pensamiento y encontrando, además, ciertos matices diferenciadores entre abierto y libre. A pesar de que a efectos prácticos ambas corrientes comparten objetivos esenciales, tipos de licencia e incluso proyectos, lo cierto es que persisten ciertas discrepancias de carácter conceptual que impiden una fusión.

El código abierto, quizás más pragmático y menos profundo, pone el acento en el carácter accesible del código, no obstante, para que un proyecto pueda ser licenciado como tal ha de cumplir ciertos requisitos indispensables que podríamos resumir en los siguientes puntos:

- El código fuente ha de estar disponible **libre y gratuitamente** para cualquier persona o entidad sin discriminación de ningún tipo.
- La licencia debe permitir la **manipulación del código fuente**, pudiéndose comercializar o no el resultado final.
- Si el *software* liberado como código abierto forma parte de un paquete mayor, el resto del código no tendrá que ser abierto necesariamente, pudiendo poseer **cualquier otro tipo de licencia**.

El *software* propietario por regla general no es comercializado como tal, es decir, no se adquiere el *software* en sí mismo sino el derecho a utilizarlo en el contexto establecido en su licencia. De este modo, si un desarrollador pretende realizar alguna modificación sobre el código adquirido verá frustradas sus aspiraciones pues, o bien no tendrá acceso al código fuente, o bien no tendrá permiso del propietario para ello, aunque es habitual que se den ambas circunstancias a la vez. En este sentido, el código abierto se presenta no solo como un bien para aquellos usuarios o entidades que acceden a un recurso de calidad, de forma libre y gratuita, sino que, además, como corriente, supone un empuje determinante para el desarrollo del *software* en general y de la Web en particular, haciendo de esta un espacio más rico, abierto y creativo, más potente y colaborativo. Al contrario de lo que se podría pensar, el código abierto ha sido todo un revulsivo para la industria del *software* a todos los niveles, beneficiándose de él compañías grandes y pequeñas, particulares y organizaciones.

Cuando el *software* es de tipo propietario, el usuario depende de la compañía desarrolladora, que solo implementará aquello que resulte económicamente rentable, tratando de maximizar sus ganancias. Por el contrario, cuando el código es abierto no existe esa dependencia y por tanto se puede realizar cualquier modificación o añadido que resulte de utilidad independientemente de su rentabilidad económica, como por ejemplo la traducción a un idioma minoritario.

Sin embargo, hay que señalar que generalmente el código abierto posee ciertos inconvenientes que es preciso asumir, como la no existencia de un número de teléfono desde el que nos den soporte técnico ni de una garantía por parte del distribuidor.

1.2.3.4 FRONT END Y BACK END

Estos dos términos son los nombres que más comúnmente reciben las dos caras de una misma moneda, las dos interfaces de una aplicación CMS. El *front end* también denominado en ocasiones como *front office* es la interfaz con la que interactúa el usuario, ya sea un visitante o un cliente. En el caso de un comercio electrónico, el usuario accede a los productos, a sus categorías, al carrito, y es desde aquí desde donde puede registrarse como cliente y realiza sus compras. Se trata por tanto de la cara visible de nuestro sistema, la parte pública.

El *back end* o *back office* es la parte privada de la aplicación, la interfaz que permite la administración del sitio y la gestión de sus contenidos y usuarios. Siguiendo con el ejemplo anterior, sería desde este acceso desde donde se podría gestionar el almacén, dar de alta los nuevos productos, ordenar sus categorías, establecer los precios, definir posibles promociones, etc.

Figura 1.16. Los usuarios acceden a los contenidos que se encuentran en el front end. Dichos contenidos son creados y administrados por parte de los administradores desde el back end

En otro contexto, es posible encontrar estos nombres para designar al conocido como lado del cliente y lado del servidor respectivamente. Estos conceptos, que nada tienen que ver con los anteriormente descritos, sirven para identificar el lugar donde se ejecuta un determinado código. Si un *script* se ha desarrollado en

algún lenguaje como PHP o ASP se ejecutarán en el servidor y si lo está en HTML o JavaScript, lo hará en el lado del cliente, es decir, en el navegador del usuario. Hay diseñadores y desarrolladores que se definen como del *front end* o del *back end*, para especificar su área de especialización.

1.2.4 Datos, metadatos e información

Es muy frecuente encontrar alusiones ambiguas a estos tres términos que, a pesar de guardar relación entre sí, obedecen en realidad a conceptos diferentes. Un dato es la unidad mínima de información, aunque por sí misma no puede ser considerada como tal. Al igual que el átomo de un árbol no es un árbol, un dato no constituye información en sí mismo. Un dato es un valor o referente que recibe el computador a través diversos medios y que será transformado por medio de algún *software* específicamente desarrollado para tal fin. Los datos en conjunto representan el fluido que atravesará la red lógica que el programador ha tejido para la construcción de una solución o en el desarrollo de un algoritmo.

Únicamente cuando un conjunto de datos es ordenado e interpretado, podemos hablar de información, es decir, sin la acción transformadora de la interpretación, los datos son solo valores carentes de significado. Cuando un conjunto de datos se examina conjuntamente a la luz de un enfoque, hipótesis o teoría se puede apreciar la información contenida en dichos datos.

Si tuviésemos que definir qué son los metadatos, podríamos decir que se trata de datos que describen otros datos. Puede parecer una entelequia, pero tiene todo su sentido como vamos a ver en el siguiente ejemplo: si tuviésemos que localizar una aguja en un pajar, lo más eficaz sería seguramente utilizar un potente imán. Esto que puede parecer una tonta obviedad, sin dejar de serlo, nos está dando una importante pista acerca del significado del concepto de metadato. Si hemos logrado encontrar la aguja, ha sido porque esta posee una propiedad que hemos sabido utilizar. Si dotamos a los datos de distintivos que recojan sus propiedades, podremos realizar búsquedas, ordenaciones y obtendremos información útil sobre la naturaleza de cada uno de ellos.

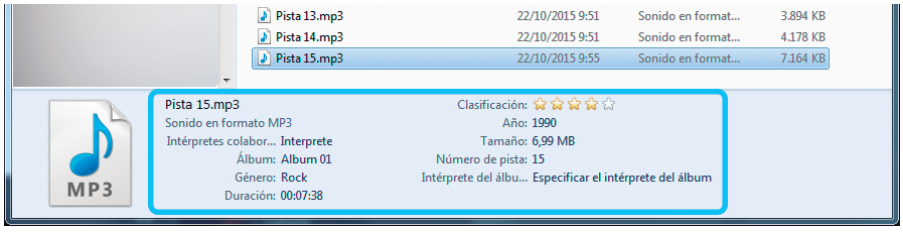

Figura 1.17. Metadatos de un archivo de audio

Un archivo de audio posee datos que si bien no constituyen la información esencial que compone el fichero, sí pertenecen a él y hacen referencia a algunas de sus características. Generalmente los metadatos son un conjunto de datos dispuestos por pares atributo-valor, que nos aportan información acerca de las características de un objeto. Sirven para categorizar e identificar la información.

1.2.5 Taxonomías

La taxonomía es la ciencia que estudia los principios, métodos y fines de la clasificación y, en el terreno de la informática, representa una vía esencial para la organización de la información de acuerdo a un orden jerárquico específico. La utilización de taxonomías nos permite crear sistemas coherentes de organización de contenidos, haciendo posible su gestión y localización de una forma eficiente. La unidad mínima está constituida por el taxón y representa un tema o concepto. Obviamente, el primer punto que debemos abordar en la elaboración de taxonomías pasa por identificar estas unidades temáticas.

Las taxonomías deben gobernar sobre todo aquello que haga referencia de una u otra forma a los taxones que hayamos definido, ya se trate de un contenido como tal o de un componente. Cualquier elemento debe estar sujeto a la estructura que hayamos determinado, de otro modo, este carecerá de coherencia y eficacia.

Supongamos que se ha desarrollado un sitio web para la venta de material de montaña y que se cuenta con una lista de temas o taxones que van a ser tratados en sus contenidos: moda, ropa de montaña, ropa de nieve, calzado, complementos, material, escalada, nieve, mochilas, mosquetones, guantes, cuerdas, piolets, crampones, camisas, sudaderas, pantalones, chaquetas, botas, zapatillas, gafas, cascos, hombre, mujer, niños y complementos. Ahora debemos crear un orden válido a partir de los diferentes temas que han sido contemplados:

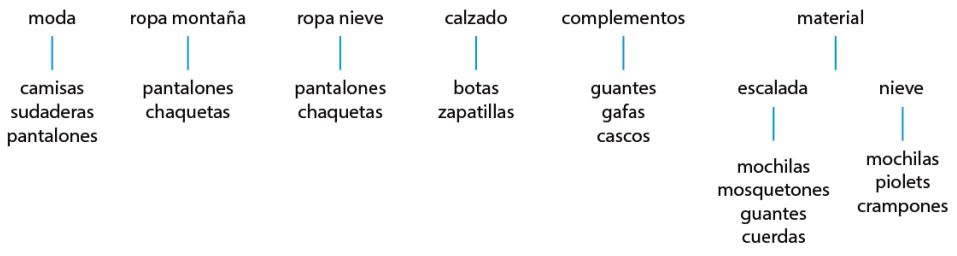

Figura 1.18. Representación jerárquica de los diferentes taxones contemplados en el ejemplo

Naturalmente, con el tiempo habrá cambios y esta es una de las razones por las que resulta tan importante realizar una estructura lo suficientemente robusta. Si en un futuro se han de introducir nuevos taxones, la estructura existente ha de permitir su inserción sin requerir modificaciones importantes. Es importante señalar que hablamos de un orden temático y nunca de la disposición de las diferentes opciones de un menú. El uso de taxonomías en la gestión de contenidos aporta enormes ventajas, entre las que cabría destacar las siguientes:

- La **búsqueda interna de contenidos** del sitio web mejora sustancialmente. El contenido adecuadamente etiquetado resulta más fácil y más rápido de localizar y, además, los resultados obtenidos son de una mayor calidad puesto que obedecen a una estructura coherente.

- Al haber definido una estructura interna, las búsquedas de **contenido relacionado** resultan más eficaces y útiles. Podemos estar hablando de productos en una tienda de comercio electrónico o simplemente de entradas de un blog, resulta igualmente conveniente poder ofrecer elementos bien relacionados entre sí.

- Se mejora sensiblemente la **navegabilidad** del sitio web. Si construimos nuestras secciones y categorías en base a la estructura temática que hemos creado, los menús y demás enlaces internos que creemos resultarán más intuitivos y naturales para nuestros visitantes.

- Las taxonomías también facilitan la **reutilización de contenido**. Esto resulta muy útil para las actualizaciones de contenido, para introducir elementos de un contenido anterior en uno actual o incluso para relanzar una campaña de marketing.

- Se pueden crear **sinónimos de los taxones** para acercar los contenidos adecuados a los usuarios potenciales. Si un visitante emplea un estilo de lenguaje distinto al nuestro y no lo hemos contemplado, no encontrará lo que busca y lo perderemos. Esto, siendo importante en sí mismo, quizá lo es aún más cuando se trata de sitios de contenido técnico donde se maneja una jerga que el usuario común desconoce o no acostumbra a utilizar. De este modo salvamos esta barrera léxica y ampliamos el espectro de posibles visitantes.

Estos puntos ponen de manifiesto las ventajas que aporta un buen desarrollo de taxonomías. Invertir el tiempo necesario para construir taxonomías coherentes y bien estructuradas puede resultar árido, pero una vez hayamos implantado el sistema y empecemos el proceso de gestión de contenidos, nos alegraremos enormemente de haberlo hecho.

En la Red podemos encontrar taxonomías sectoriales que nos podrán ayudar a ahorrar tiempo y esfuerzo si lo que vamos a crear es un sitio web de cierta envergadura o un comercio electrónico con gran variedad de productos.

1.2.5.1 SECCIONES Y CATEGORÍAS

En el ejemplo de la figura 1.18 hemos omitido deliberadamente los taxones hombre, mujer y niños. Obviamente podrían incluirse de diversas formas en la estructura existente, bastaría con hacer alguna pequeña modificación, sin embargo, en este caso se ha optado por tratarlos como categorías.

Desde el punto de vista filosófico, una **categoría** es un concepto muy amplio que tiene como razón de ser la agrupación de elementos semejantes e incluso, su clasificación bajo un orden jerárquico. Dentro del campo de los CMS, posee un significado más específico y un uso muy concreto, de modo que no existe posibilidad alguna de confusión con las taxonomías. Las categorías se establecen en base a los taxones existentes y a la estructura de taxonomías desarrollada. Cada uno de los elementos que componen los diferentes contenidos es asociado a una o varias categorías a modo de etiquetas identificadoras. De esta forma, trasladamos implícitamente las taxonomías a los contenidos.

Si volvemos por un instante al ejemplo de la figura 1.18, veremos que podemos asociar un ramal de la clasificación mediante la vinculación de los diferentes taxones que se encuentran en la ruta a través de su asociación como categorías y subcategorías. Si, por ejemplo, introducimos un elemento relacionado con un modelo de botas, asociaremos este a los taxones «calzado» y «botas»; pero también a «hombre», «mujer» o «niño» (según proceda). Como podemos observar, se trata de flexibilizar nuestra estructura de taxonomías, haciéndola más versátil. La opción elegida será buena o mala en función de cada caso.

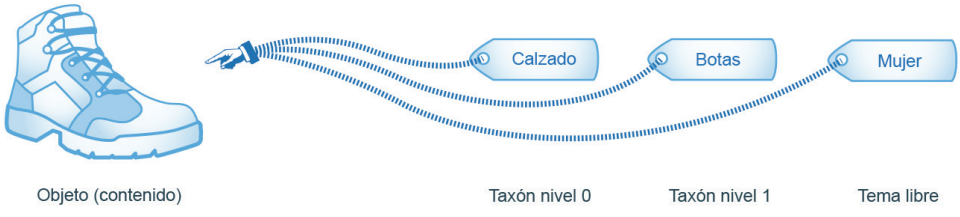

Figura 1.19. A un nuevo contenido le asociaremos, a modo de categorías, los taxones que se encuentren en su ramal temático, así como aquellos temas que no hayan sido contemplados dentro de la estructura de taxonomías

Una **sección** es una parte de un todo, es decir, una división que se lleva a cabo generalmente con algún propósito. En el terreno de los contenidos, las secciones representan una vía para regular la forma en que se muestran éstos a los usuarios, obedeciendo a una visión organizativa que puede ser diferente al de las taxonomías, por lo que no ha de guardar necesariamente ninguna relación. Podemos contemplar las secciones como opciones de menú o como páginas contenedoras de diferente tipo de información.

Así, y volviendo una vez más al ejemplo de la figura 1.18, si establecemos las secciones Novedades, Tienda, Liquidaciones, Foro, Contacto y Enlaces externos, encontraríamos que las tres primeras, así como sus posibles subopciones, deberían estar sujetas a las taxonomías, sin embargo, las tres últimas podrían perfectamente no guardar ninguna relación.

1.2.6 Base de datos

Una base de datos es un conjunto de datos relacionados almacenados de forma estructurada para facilitar su posterior recuperación. El sistema de fichas que se utilizaba en las bibliotecas de finales de siglo es un buen ejemplo de este concepto, y es que la relación entre base de datos e informática es relativamente reciente.

En una base de datos digital, por regla general, los datos se recogen en tablas que pueden estar referenciadas entre sí, diferenciando cada una de las entradas mediante un índice primario único o una combinación de secundarios. El modelo relacional, nacido a principios de los años setenta, es el más extendido hoy en día. Este paradigma permite la asociación de registros de diferentes tablas manteniendo la integridad de los datos. Pensemos, por ejemplo, en la base de datos de una biblioteca. Supongamos que localizamos un libro por su título. Lo natural es que podamos acceder al listado de libros del autor, a su ficha, a otros libros de temática similar o incluso a otros títulos de la misma editorial. Si ahora hacemos el mismo ejercicio con una tienda donde podemos encontrar otras muchas variables como la marca, la categoría, el tipo de producto, el precio, el tamaño o el color, entenderemos rápidamente que todos esos datos no pueden almacenarse en una sola tabla y tampoco en varias aisladas entre sí. Es preciso crear una estructura de datos, a menudo compleja, que permita establecer estas relaciones de forma estable. Para facilitar esta tarea nacen los sistemas gestores de bases de datos.

1.2.6.1 SISTEMA GESTOR DE BASES DE DATOS

Un sistema gestor de bases de datos o SGBD es una aplicación modular que provee de una serie de herramientas para crear bases de datos, definir su estructura y administrar los datos que residen en éstas, permitiendo añadir, borrar, modificar y analizar dichos datos. Además, un SGBD cuenta con un conjunto de características

orientadas a proporcionar fiabilidad y eficacia en el uso y la administración de los datos. Entre éstas cabe destacar las siguientes:

- Todos los datos se encuentran **centralizados** en una única ubicación compartida por todos los usuarios del sistema, de este modo se evitan redundancias e inconsistencias al tiempo que se facilitan la administración, el control y las copias de seguridad.
- Existe **independencia** de los datos respecto de las aplicaciones que acceden a éstos, tanto a nivel lógico como a nivel físico. Esta es una fortaleza que permite diseñar libremente las estructuras de datos más adecuadas al margen del *software* que será utilizado por los usuarios que, por otro lado, serán ajenos al modo en que se almacenan los datos y al lugar en que residen. Además, esta independencia admite futuras actualizaciones de las aplicaciones sin afectar a los datos.
- Posee mecanismos para preservar la **integridad** de las bases de datos frente a inconsistencias tales como la redundancia, la inserción de datos inadecuados o la eliminación directa de entradas con dependencias en terceras tablas.
- Cuenta con un sistema de **seguridad** que protege los datos frente a usuarios no autorizados, estableciendo generalmente diversos niveles de permisos.
- Se encarga de la gestión del acceso concurrente que permite el **uso compartido** de las bases de datos y preserva su fiabilidad mediante mecanismos de recuperación frente a fallos.

Tres lenguajes proveen de las diferentes herramientas: el lenguaje de **definición de datos**, el lenguaje de **manejo de datos** y el lenguaje de **control de datos**. Estos lenguajes, generalmente unificados bajo una misma sintaxis, son utilizados desde otros lenguajes de alto nivel, lo que facilita en gran medida el trabajo de desarrollo. El lenguaje más extendido a día de hoy es conocido como lenguaje de consulta estructurado o SQL, un estándar para los SGBD relacionales. Se trata de un sencillo lenguaje declarativo, es decir, no hay que especificar cómo se han de hacer las cosas, basta con describir qué es lo que se ha de hacer.

1.2.6.2 MYSQL

MySQL es un sistema gestor de bases de datos relacional de código abierto que integra el lenguaje SQL. Este sistema, multiusuario y multihilo, se ha ido imponiendo progresivamente a lo largo del tiempo por todo el planeta, y no solo dentro de la Red. Su eficiencia, robustez, escalabilidad y su gran portabilidad lo han catapultado hasta la primera posición de una clasificación donde se encuentran

SGBD de la talla de Oracle o DB2. Si bien es cierto que su gratuidad es un factor fundamental que juega a su favor, la clave de su éxito reside en ser uno de los tres pilares esenciales del paradigma AMP, un modelo sobre el que se apoyan cientos de proyectos CMS.

La arquitectura de MySQL se encuentra dividida en tres capas a nivel lógico. La primera, la capa de **conexión**, contempla la comunicación con otros lenguajes. En segundo lugar, la capa de **lógica**, se ocupa del funcionamiento de las consultas a las bases de datos. Por último, la capa de **almacenamiento**, es la que posee las directrices que determinan el modo en que se guardan los datos, permitiendo la utilización de cuatro motores de almacenamiento diferentes: MyISAM (opción por defecto), InnoDB, Memory y NDB.

MySQL admite el acceso remoto, es decir, una web y su base de datos pueden encontrarse en servidores diferentes, ya sea por motivos de seguridad o por optimización de recursos. Además, se pueden crear réplicas en tiempo real mediante el modelo maestro-esclavo, lo que facilita la creación de copias de respaldo y la distribución de la carga entre varios servidores, entre otros aspectos.

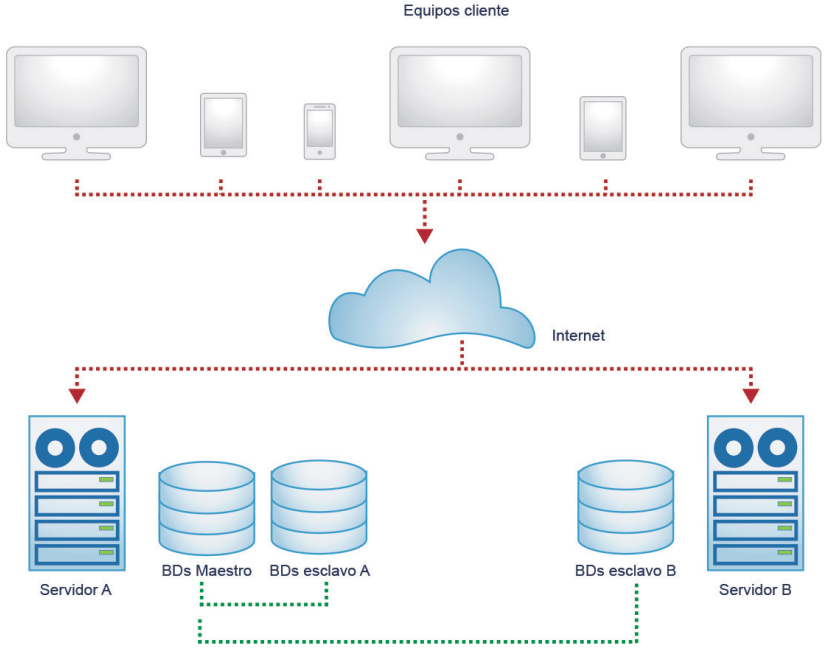

Figura 1.20. En esta configuración MySQL de ejemplo encontramos la base de datos principal (BDs Maestro) alojada en el servidor A, junto a una réplica (BDs esclavo A) que juega el papel de copia de seguridad. Por otro lado, en el servidor B contamos con otra réplica (BDs esclavo B) que se emplearía para balancear la carga de accesos concurrentes