- **Movilidad**. Este es un concepto relativo y dependiente de la implantación concreta. Si el software está ubicado en un servidor web en Internet o bien disponemos de una intranet externalizada (extranet), cualquier usuario con un portátil y una conexión a Internet móvil podría acceder a la aplicación.
- **Reducción de costes en los puestos cliente** (**mayor longevidad**). Debido a que las páginas se ofrecen desde el servidor web (donde se suelen ejecutar la mayoría de los procesos y la lógica de negocio), el equipo cliente queda relegado a mostrar los resultados y formularios, para lo cual no es necesario un hardware potente en los puestos de trabajo, lo que se traduce en reducción de costes y una mayor longevidad en el uso de los mismos (no hay que cambiar el hardware de los puestos porque ahora se requieran operaciones más complejas).

Sin embargo no todo son ventajas. Debemos recordar que en el mundo real de los requisitos y restricciones no existe la solución perfecta, sino la más o menos adecuada al caso planteado. Por ello, una solución web también tiene sus inconvenientes, unos derivados del modelo web y otros como consecuencia de cómo se implante.

1.3 ARQUITECTURA CLIENTE SERVIDOR. ELEMENTOS

La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados **servidores**, y los demandantes, llamados **clientes**. Un cliente realiza peticiones a otro programa, el servidor, que le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un solo programa. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma.

Una disposición muy común son los sistemas multicapa en los que el servidor se descompone en diferentes programas que pueden ser ejecutados por diferentes computadoras aumentando así el grado de distribución del sistema. La arquitectura cliente-servidor sustituye a la arquitectura monolítica en la que no hay distribución, tanto a nivel físico como a nivel lógico.

La red cliente-servidor es aquella red de comunicaciones en la que todos los clientes están conectados a un servidor, en el que se centralizan los diversos recursos y aplicaciones con los que se cuentan y que se ponen a disposición de los clientes cada vez que estos son solicitados. Esto significa que todas las gestiones que se realizan se concentran en el servidor, de manera que en él se disponen los requerimientos provenientes de los clientes que tienen prioridad, los archivos que son de uso público y los que son de uso restringido, los archivos que son de solo lectura y los que, por el contrario, pueden ser modificados, etc. Este tipo de red puede utilizarse conjuntamente en caso de que se esté utilizando en una red mixta.

Características

En la arquitectura C/S el remitente de una solicitud es conocido como cliente. Sus características son:

- ✓ Es el que inicia solicitudes o peticiones. Tiene, por tanto, un papel activo en la comunicación (dispositivo maestro o amo).
- ✓ Espera y recibe las respuestas del servidor.
- ✓ Por lo general, puede conectarse a varios servidores a la vez.
- ✓ Normalmente, interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.
- ✓ Al contratar un servicio de red, se debe de tener encuenta la velocidad de conexión que se le otorga al cliente y el tipo de cable que utiliza.

Al receptor de la solicitud enviada por el cliente se conoce como servidor. Sus características son:

- ✓ Al iniciarse espera a que le lleguen las solicitudes de los clientes. Desempeñan entonces un papel pasivo en la comunicación (dispositivo esclavo).
- ✓ Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.
- ✓ Por lo general, aceptan conexiones desde un gran número de clientes (en ciertos casos el número máximo de peticiones puede estar limitado).
- ✓ No es frecuente que interactúen directamente con los usuarios finales.

Ventajas

- ✓ **Centralización del control**: los accesos, recursos y la integridad de los datos son controlados por el servidor, de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema. Esta centralización también facilita la tarea de poner al día datos u otros recursos (mejor que en las redes P2P).
- ✓ **Escalabilidad**: se puede aumentar la capacidad de clientes y servidores por separado. Cualquier elemento puede ser aumentado (o mejorado) en cualquier momento, o se pueden añadir nuevos nodos a la red (clientes y/o servidores).
- ✓ **Fácil mantenimiento**: al estar distribuidas las funciones y responsabilidades entre varios ordenadores independientes, es posible reemplazar, reparar, actualizar o incluso trasladar un servidor, mientras que sus clientes no se verán afectados por ese cambio (o se afectarán mínimamente). Esta independencia de los cambios también se conoce como encapsulación.
- ✓ **Tecnologías**: existen algunas suficientemente desarrolladas, diseñadas para el paradigma de C/S, que aseguran la seguridad en las transacciones, la amigabilidad de la interfaz y la facilidad de empleo.

Desventajas

- ✓ La congestión del tráfico ha sido siempre un problema en el paradigma de C/S. Cuando una gran cantidad de clientes envían peticiones simultaneas al mismo servidor, puede ser que cause muchos problemas para éste (a mayor número de clientes, más problemas para el servidor). Al contrario, en las redes P2P, como cada nodo en la red hace también de servidor, cuantos más nodos hay, mejor es el ancho de banda que se tiene.

- ✓ El paradigma de C/S clásico no tiene la robustez de una red P2P. Cuando un servidor está caído las peticiones de los clientes no pueden ser satisfechas. En la mayor parte de redes P2P, los recursos están generalmente distribuidos en varios nodos de la red. Aunque algunos salgan o abandonen la descarga; otros pueden todavía acabar de descargar consiguiendo datos del resto de los nodos en la red.
- ✓ El software y el hardware de un servidor son generalmente muy determinantes. Un hardware regular de un ordenador personal puede no poder servir a cierta cantidad de clientes. Normalmente, se necesita software y hardware específicos, sobre todo en el lado del servidor para satisfacer el trabajo. Por supuesto, esto aumentará el coste.
- ✓ El cliente no dispone de los recursos que puedan existir en el servidor. Por ejemplo, si la aplicación es una Web no podemos escribir en el disco duro del cliente o imprimir directamente sobre las impresoras sin sacar antes la ventana previa de impresión de los navegadores.

Ejemplos. La mayoría de los servicios de Internet son tipo de cliente-servidor. La acción de visitar un sitio web requiere una arquitectura cliente-servidor, ya que el servidor web sirve las páginas web al navegador (al cliente). Al leer este artículo en Wikipedia, la computadora y el navegador web del usuario serían considerados un cliente; y las computadoras, las bases de datos y los usos que componen Wikipedia serían considerados el servidor. Cuando el navegador web del usuario solicita un artículo particular de Wikipedia, el servidor de Wikipedia recopila toda la información a mostrar en su base de datos, la articula en una página web y la envía de nuevo al navegador web del cliente.

Otro ejemplo podría ser el funcionamiento de un juego *on line*. Si existen dos servidores de juego, cuando un usuario lo descarga y lo instala en su computadora pasa a ser un cliente. Si tres personas juegan en un solo computador existirían dos servidores, un cliente y tres usuarios. Si cada usuario instala el juego en su propio ordenador existirían dos servidores, tres clientes y tres usuarios.

1.4 ARQUITECTURA DE TRES NIVELES

En la arquitectura en tres niveles existe un nivel intermedio. Esto significa que la arquitectura generalmente está compartida por:

- Un **cliente**, es decir, el equipo que solicita los recursos, equipado con una interfaz de usuario (generalmente un navegador web) para la presentación.
- El **servidor de aplicaciones** (también denominado software intermedio), cuya tarea es proporcionar los recursos solicitados, pero que requiere de otro servidor para hacerlo.
- El **servidor de datos**, que proporciona al servidor de aplicaciones los datos que éste le solicitó.

El uso masivo del término arquitectura en tres niveles también denota las siguientes arquitecturas:

- Aplicación compartida entre un cliente, un software intermedio y un servidor empresarial.
- Aplicación compartida entre un cliente, un servidor de aplicaciones y un servidor de base de datos empresarial.

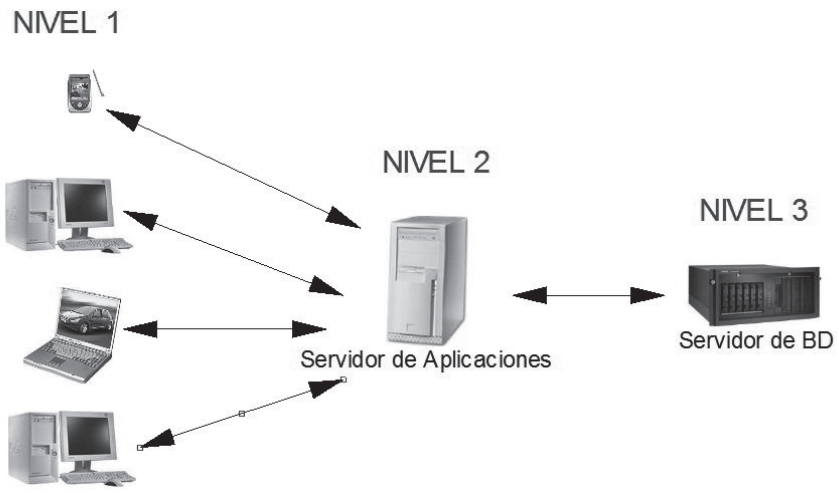

Figura 1.1. *Arquitectura de 3 niveles*

La arquitectura en dos niveles es, por tanto, una arquitectura cliente-servidor en la que el servidor es polivalente, es decir, puede responder directamente a todas las solicitudes de recursos del cliente.

Sin embargo, en la arquitectura en tres niveles las aplicaciones al nivel del servidor son descentralizadas de uno a otro, es decir, cada servidor se especializa en una determinada tarea, (por ejemplo, servidor web/servidor de bases de datos). La arquitectura en tres niveles permite:

- Un mayor grado de flexibilidad.
- Mayor seguridad, ya que la seguridad se puede definir independientemente para cada servicio y en cada nivel.
- Mejor rendimiento, ya que las tareas se comparten entre servidores.

Hay que tener en cuenta que los distintos niveles representados como equipos físicos distintos podrían ser llevados a cabo por programas "servidores" dentro de un mismo servidor físico. Es decir, un software que funcionase como servidor de aplicaciones y otro software como servidor de bases de datos.

1.5 PROTOCOLOS DE APLICACIÓN MÁS UTILIZADOS

Un **protocolo** es un método estándar que permite la comunicación entre procesos (que potencialmente se ejecutan en diferentes equipos), es decir, es un conjunto de reglas y procedimientos que deben respetarse para el envío y la recepción de datos a través de una red. Existen diversos protocolos de acuerdo a cómo se espera que sea la comunicación. Algunos protocolos, por ejemplo, se especializarán en el intercambio de archivos, como el FTP (*File Transfer Protocol*, Protocolo de transferencia de ficheros); otros pueden utilizarse simplemente para administrar el estado de la transmisión y los errores (como es el caso de ICMP), etc.

En Internet, los protocolos utilizados pertenecen a una sucesión de protocolos o a un conjunto de protocolos relacionados entre sí. Este conjunto de protocolos se denomina TCP/IP. Entre otros, contiene los siguientes protocolos:

1.5.1 EL PROTOCOLO HTTP

Desde 1990, el protocolo HTTP (*Hiper Text Transfer Protocol*, Protocolo de transferencia de hipertexto) es el protocolo más utilizado en Internet. La versión 0.9 solo tenía la finalidad de transferir los datos a través de Internet (en particular páginas web escritas en HTML). La versión 1.0 del protocolo (la más utilizada) permite la transferencia de mensajes con encabezados que describen el contenido de los mensajes mediante la codificación MIME.

El propósito del protocolo HTTP es permitir la transferencia de archivos (principalmente, en formato HTML) entre un navegador (el cliente) y un servidor web localizado mediante una cadena de caracteres denominada dirección URL (*Uniform Resource Locator*, localizador uniforme de recursos).

La comunicación entre el navegador y el servidor se lleva a cabo en dos etapas:

- El navegador realiza una solicitud HTTP.
- El servidor procesa la solicitud y después envía una respuesta HTTP.

1.5.2 EL PROTOCOLO HTTPS

El protocolo seguro de Transferencia de hipertexto (HTTPS, *Hiper Text Transfer Protocol Secure*) es la versión segura del protocolo HTTP. La diferencia es que HTTPS permite realizar transacciones de forma segura. Por lo tanto, podremos desarrollar actividades de tipo *e-commerce*, acceso a cuentas bancarias *on line*, tramites con la administración pública, etc.

En los navegadores comunes como Firefox, Explorer o Chrome, cuando estamos empleando un protocolo HTTPS podemos ver el icono de un candado que aparece en la barra principal de nuestro navegador. Además, en la barra de direcciones podremos ver que "http://" será sustituido por "https://".

Y, ¿cómo funciona la conexión exactamente? ¿Por qué es más segura? Básicamente, lo que ocurre es que la página web codifica la sesión con certificado digital. De este modo, el usuario tiene ciertas garantías de que la información que envíe desde dicha página no podrá ser interceptada y utilizada por terceros.

Estos certificados de seguridad son conocidos como SSL. Cuando estos estás instalados en la página web veremos el candado del que hablábamos anteriormente. Por otro lado, si hay instalados Certificados de Validación Extendida, además del candado, los usuarios podremos ver que la barra de URL del navegador toma un fondo verdoso.

1.5.3 EL PROTOCOLO FTP

El protocolo FTP (*File Transfer Protocol*, Protocolo de transferencia de archivos) es, como su propio nombre indica, un protocolo para transferir archivos.

La implementación del FTP se remonta a 1971, cuando se desarrolló un sistema de transferencia de archivos (descrito en RFC141) entre equipos del Instituto Tecnológico de Massachusetts (MIT, Massachusetts Institute of Technology). Desde entonces, diversos documentos de RFC han mejorado el protocolo básico, pero las innovaciones más importantes se llevaron a cabo en julio de 1973.

El protocolo FTP define la manera en que los datos deben ser transferidos a través de una red TCP/IP. El objetivo del protocolo FTP es:

- Permitir que equipos remotos puedan compartir archivos.
- Permitir la independencia entre los sistemas de archivo del equipo del cliente y del equipo del servidor.
- Permitir una transferencia de datos eficaz.