

Programación en Internet: clientes web

Sergio Luján Mora

Prefacio

En pocos años, Internet ha invadido casi todos los aspectos de la vida. Podemos comunicarnos a través de Internet de distintas formas (correo electrónico, radio y televisión *online*, telefonía IP). Podemos comprar diversos productos en Internet (libros, discos, entradas de cine). Podemos conocer gente a través de Internet (*chat*, foros de discusión). Para que todo ello funcione, hacen falta profesionales especializados en “programación en Internet”.

En este libro se trata una pequeña porción de la “programación en Internet”: la programación de aplicaciones web. Las aplicaciones web se encuadran dentro de las arquitecturas cliente/servidor. En concreto, en este libro se estudia como programar la parte cliente, los clientes web. Las tecnologías que se contemplan son HTML y JavaScript. Existen otras tecnologías, como ActiveX o *applets*, pero no están estandarizadas como HTML y JavaScript.

Este libro se complementa con otro de próxima aparición que tratará la programación de aplicaciones web desde el lado del servidor. En él se mostrarán las tecnologías que se emplean para programar los servidores web: CGI, ASP, JSP, etc.

Evidentemente, existen programas que permiten crear páginas HTML sin tener ni idea de HTML. Pero igual que se puede conducir un coche sin tener ni idea de mecánica o pilotar un avión sin tener ni idea de aerodinámica o de motores a reacción, conviene conocer “lo que se tiene entre manos” para poder obtener el máximo partido. La programación de aplicaciones web no consiste sólo en crear páginas web muy bonitas, con muchas imágenes y colores; hay que validar la entrada del usuario, acceder a bases de datos, etc.

Este no es un libro sobre programación básica, donde se expliquen conceptos como variable, bucle de repetición, expresión lógica o recursividad. Es necesario poseer un nivel mínimo de programación para poder abordar los temas que tratan sobre JavaScript. Sin embargo, cualquiera que haya programado en

algún lenguaje no tendrá problemas en asimilarlo.

El libro está estructurado en seis capítulos. Los dos primeros capítulos son introductorios: se repasan las arquitecturas cliente/servidor en general y se presenta un tipo concreto, las aplicaciones web. El tercer capítulo está dedicado al lenguaje HTML. En el cuarto capítulo se explican los lenguajes de *script* en general y en el siguiente capítulo se trata un lenguaje concreto: JavaScript. El libro finaliza con el modelo de objetos de documento, que permite acceder a los elementos de una página web desde un lenguaje de *script*. Por último, existen tres apéndices donde se resumen las etiquetas de HTML y se explica como trabajar con los colores en HTML y como depurar errores de JavaScript.

Aunque no figuran como autores (y no tienen “ni idea” de HTML o JavaScript), sin la participación de mis padres este libro no existiría, ya que su apoyo a lo largo de bastantes años me ha permitido llegar a escribir este libro.

Me gustaría agradecer a Marisa la paciencia (¿infinita o ínfima?) que ha tenido todas las veces que he llegado tarde debido a este libro.

Por último, mando un saludo a mis antiguos compañeros del Laboratorio Multimedia (mmlab), donde me introduje en el mundo de Internet, y a mis actuales compañeros del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Alicante.

Alicante, 8 de octubre de 2001

Sergio Luján Mora

Índice general

Prefacio	III
Índice general	V
Índice de cuadros	XI
Índice de figuras	XIII
Índice de acrónimos	XV
1. Arquitecturas cliente/servidor	1
1.1. Introducción	2
1.2. Separación de funciones	3
1.3. Modelos de distribución en aplicaciones cliente/servidor	3
1.3.1. Presentación distribuida	4
1.3.2. Aplicación distribuida	4
1.3.3. Datos distribuidos	4
1.4. Arquitecturas de dos y tres niveles	5
1.5. Descripción de un sistema cliente/servidor	5
2. Qué es una aplicación web	7
2.1. Introducción	8
2.1.1. El cliente	8
2.1.2. El servidor	9
2.2. Transferencia de páginas web	10
2.3. Entornos web	11
2.3.1. Internet	11

2.3.2. Intranet	12
2.3.3. Extranet	12
2.4. Ventajas y desventajas	13
3. HTML	15
3.1. Introducción	17
3.2. Evolución de HTML	17
3.3. Clasificación de las páginas	19
3.4. Qué necesito para usar HTML	21
3.5. Conceptos básicos de HTML	22
3.5.1. Estructura de una página	23
3.5.2. Caracteres especiales y secuencias de escape	25
3.6. Metadatos	27
3.7. Etiquetas HTML	28
3.8. Formato del texto	29
3.8.1. Encabezados de secciones	30
3.8.2. Formatos de caracteres	32
3.8.3. La etiqueta 	32
3.8.4. Alineamiento del texto	36
3.9. Listas	41
3.9.1. Listas de definición	41
3.9.2. Listas ordenadas	42
3.9.3. Listas no ordenadas	45
3.10. Colores	47
3.10.1. Color de fondo de una página	47
3.10.2. Color del texto	47
3.11. Enlaces	48
3.11.1. Enlace a un punto del mismo documento	48
3.11.2. Enlace a otro documento	50
3.11.3. Enlace a un punto de otro documento	51
3.12. Tablas	53
3.12.1. Tablas invisibles	57
3.12.2. Tablas como marcos	58
3.13. Imágenes	58
3.13.1. Etiqueta 	60
3.13.2. Imágenes como fondo de una página	61
3.14. Formularios	61

3.14.1. Controles de un formulario	63
3.14.2. Campos de verificación	67
3.14.3. Campos excluyentes	67
3.14.4. Campos de texto	67
3.14.5. Listas de selección	67
3.14.6. Áreas de texto	69
3.15. Marcos	70
3.16. Guía de estilo	75
3.16.1. Organizar el código HTML	76
3.16.2. Cuidado con los colores	76
3.16.3. Cuidado con los tipos de letra	76
3.16.4. Sacar partido al hipertexto	76
3.16.5. Usar las capacidades multimedia	77
3.16.6. Identidad corporativa	77
3.16.7. Permitir que los usuarios se comuniquen	77
3.16.8. Facilitar las búsquedas	77
3.16.9. Revisar las páginas periódicamente	78
3.16.10. Los enlaces	78
4. Lenguajes de script	79
4.1. Introducción	79
4.2. Diferencias entre VBScript y JavaScript	80
4.3. Para qué sirven	80
4.4. Como se usa un lenguaje de script en un navegador	81
5. JavaScript	87
5.1. Introducción	88
5.1.1. Aplicaciones	89
5.1.2. Qué necesito para programar en JavaScript	90
5.1.3. JavaScript y Java	90
5.1.4. Versiones	91
5.1.5. JavaScript y ECMA	93
5.1.6. JScript	95
5.1.7. Diferencias entre JavaScript y JScript	95
5.2. El lenguaje	96
5.2.1. Características básicas	96
5.2.2. Comentarios	98

5.2.3.	Declaración de variables	98
5.2.4.	Ámbito de las variables	100
5.2.5.	Caracteres especiales	102
5.2.6.	Operadores	103
5.2.7.	Palabras reservadas	105
5.3.	Sentencias	105
5.3.1.	Condicionales	105
5.3.2.	De repetición	108
5.3.3.	De manipulación de objetos	112
5.4.	Funciones	114
5.4.1.	Declaración de funciones	114
5.4.2.	Funciones predefinidas	116
5.5.	Objetos	120
5.5.1.	Creación de objetos	121
5.5.2.	Métodos de un objeto	122
5.5.3.	Eliminación de objetos	123
5.6.	Tratamiento de cadenas	123
5.7.	Operaciones matemáticas	128
5.8.	Validación de formularios	132
5.8.1.	Validación campo nulo	133
5.8.2.	Validación alfabética	135
5.8.3.	Validación numérica	138
6.	Modelo de objetos de documento	145
6.1.	Introducción	146
6.2.	Modelo de objetos en Netscape Communicator	146
6.2.1.	Objeto document	148
6.2.2.	Cómo acceder a los controles de un formulario	152
6.2.3.	Objeto history	157
6.2.4.	Objeto location	157
6.2.5.	Objeto navigator	160
6.2.6.	Objeto window	161
6.3.	Modelo de objetos en Microsoft Internet Explorer	170
A.	Resumen etiquetas HTML	173
A.1.	Introducción	174
A.2.	Etiquetas que definen la estructura del documento	175

A.3. Etiquetas que pueden ir en la cabecera	176
A.4. Etiquetas que definen bloques de texto	177
A.5. Etiquetas de listas	177
A.6. Etiquetas de características del texto	178
A.7. Etiquetas de anclas y enlaces	179
A.8. Etiquetas de imágenes y mapas de imágenes	180
A.9. Etiquetas de tablas	181
A.10. Etiquetas de formularios	183
A.11. Etiquetas de marcos	186
A.12. Etiquetas de situación de contenidos	187
A.13. Etiquetas de script	189
A.14. Etiquetas de applets y plug-ins	189
A.15. Etiquetas de ajuste del texto	191
A.16. Atributos universales	192
B. Colores en HTML	193
B.1. Como trabajar con las componentes RGB	193
B.1.1. Obtener las componentes del color deseado en decimal	194
B.1.2. Transformar las componentes de decimal a hexadecimal	194
B.2. Tabla de colores	197
C. Depuración de errores de JavaScript	201
C.1. Introducción	202
C.2. Depuración en cualquier navegador	202
C.3. Netscape Communicator	202
C.3.1. Modificar las preferencias	204
C.3.2. Evaluación de expresiones con la consola	207
C.3.3. Netscape JavaScript Debugger	207
C.4. Microsoft Internet Explorer	211
Bibliografía	217
Índice alfabético	219

Índice de cuadros

3.1. Versiones de HTML	18
3.2. Caracteres con un significado especial en HTML	26
3.3. Caracteres especiales	26
3.4. Diferencias entre GIF y JPEG	59
5.1. JavaScript frente a Java	91
5.2. Relación entre las versiones de JavaScript y de Netscape Navigator	92
5.3. Relación entre las versiones de JavaScript y de ECMA	94
5.4. Relación entre las versiones de JScript y los productos de Microsoft	95
5.5. Caracteres especiales	102
5.6. Precedencia de los operadores de JavaScript	104
5.7. Palabras reservadas de JavaScript	105
5.8. Propiedades del objeto Math	128
B.1. Equivalencias para pasar del sistema decimal al hexadecimal . .	195
B.2. Nombres de algunos colores en HTML	199

Índice de figuras

1.1. Separación de funciones	3
1.2. Presentación distribuida	4
1.3. Aplicación distribuida	4
1.4. Datos distribuidos	5
2.1. Esquema básico de una aplicación web	8
3.1. Primera página HTML	25
3.2. Ejemplo de encabezados	31
3.3. Formatos físicos y lógicos	33
3.4. Distintos tipos de letra con la etiqueta 	35
3.5. Distintos tamaños de letra con la etiqueta 	37
3.6. Alineamiento de párrafos: izquierda, derecha, centrado y justificado	39
3.7. Bloques de texto con distinta sangría	40
3.8. Listas de definición	42
3.9. Listas ordenadas	44
3.10. Listas no ordenadas	46
3.11. Enlace a un destino interno	49
3.12. Destino del enlace interno	50
3.13. Página con enlace a otra página	52
3.14. Página con enlace a otra página	52
3.15. Página con dos enlaces a otra página	54
3.16. Página destino de los enlaces	54
3.17. Tabla sencilla	56
3.18. Tablas como marcos	59
3.19. Imágenes con distinto alineamiento del texto	62
3.20. Formulario con distintos controles	66

3.21. Distintas listas de selección	69
3.22. Áreas de texto de distinto tamaño	71
3.23. Página con dos marcos verticales	75
4.1. Código JavaScript en un enlace	85
5.1. Ejemplo de caracteres especiales	103
5.2. Validación campo nulo	133
5.3. Validación alfabética	138
5.4. Validación numérica	143
6.1. Modelo de objetos en Netscape Communicator	147
6.2. Propiedades del objeto document	153
6.3. Propiedades del objeto location	159
6.4. Propiedades del objeto navigator	162
6.5. Interacción entre varias ventanas a través del objeto window . .	169
6.6. Modelo de objetos en Microsoft Internet Explorer	171
B.1. Ventana para modificar colores en Microsoft Paint	195
B.2. Ventana para definir colores personalizados en Microsoft Paint .	196
B.3. Calculadora en modo científico	198
C.1. Consola JavaScript de Netscape Communicator	203
C.2. Consola JavaScript con mensajes de error	205
C.3. Evaluación de expresiones	208
C.4. Netscape JavaScript Debugger	209
C.5. SmartUpdate en Netscape Communicator	210
C.6. Mensaje de alerta de Microsoft Internet Explorer	211
C.7. Mensaje de alerta en la barra de estado de Microsoft Internet Explorer	212
C.8. Opciones de Microsoft Internet Explorer	213
C.9. Mensaje de error en Microsoft Internet Explorer	214
C.10. Mensaje de error en Microsoft Internet Explorer	214
C.11. Nuevas opciones de Microsoft Internet Explorer	214
C.12. Depurador de Microsoft	215

Índice de acrónimos

ASP *Active Server Pages*

Tecnología de Microsoft que permite crear páginas web dinámicas en el servidor. Se puede decir que las páginas **ASP** son similares a los programas **CGI**. Las páginas **ASP** suelen estar programados en *VBScript*, aunque también se pueden programar en otros lenguajes.

ASCII *American Standard Code for Information Interchange*

Código binario utilizado para representar letras, números, símbolos, etc. A cada carácter se le asigna un número del 0 al 127 (7 bits). Por ejemplo, el código **ASCII** para la A mayúscula es 65. Existen códigos **ASCII** extendidos de 256 caracteres (8 bits), que permiten representar caracteres no ingleses como las vocales acentuadas o la ñe. Los caracteres de la parte superior (128 a 255) de estos códigos **ASCII** extendidos varían de uno a otro. Por ejemplo, uno de los más extendidos es ISO Latin-1 (oficialmente ISO-8859-1).

CGI *Common Gateway Interface*

Estándar que permite el intercambio de información entre un servidor y un programa externo al servidor. Un programa **CGI** es un programa preparado para recibir y enviar datos desde y hacia un servidor web según este estándar. Normalmente se programan en *C* o en *Perl*, aunque se puede usar cualquier lenguaje de propósito general.

DHTML *Dynamic HTML*

Conjunto de extensiones a **HTML** que permiten modificar el contenido de una página web en el cliente sin necesidad de establecer una nueva comunicación con el servidor. Se basa en el uso de **DOM** para acceder al contenido de la página.

DLL *Dynamic Link Library*

Fichero que almacena funciones ejecutables o datos que pueden ser usados por una aplicación en **Microsoft Windows**. Una **DLL** puede ser usada por varios programas a la vez y se carga en tiempo de ejecución (no en tiempo de compilación).

DOM *Document Object Model*

Especificación que define como se puede acceder a los objetos de un documento **HTML** (ventanas, imágenes, formularios) a través de un lenguaje de *script*. Básicamente define un jerarquía de objetos. **DOM** se encuentra en proceso de estandarización por **W3C**. **DHTML** depende de **DOM** para cambiar dinámicamente el contenido de una página web. Desgraciadamente, los dos navegadores mayoritarios poseen distintos modelos de objetos.

ECMA *European Computer Manufacturers Association*

ECMA es una asociación internacional que establece estándares relacionados con sistemas de comunicación y de información.

GIF *Graphics Interchange Format*

Formato gráfico de mapas de bit desarrollado por **COMPUSERVE**. Incorpora compresión de datos, transparencias y animaciones. Existen dos versiones de este estándar gráfico: 87a y 89a.

HTML *HyperText Markup Language*

Lenguaje compuesto de una serie de etiquetas o marcas que permiten definir el contenido y la apariencia de las páginas web. Aunque se basa en **SGML**, no se puede considerar que sea un subconjunto. Existen cientos de etiquetas con diferentes atributos. **W3C** se encarga de su estandarización. El futuro sustituto de **HTML** es **XHTML**.

HTTP *HyperText Transfer Protocol*

Es el protocolo que emplea la **WWW**. Define cómo se tienen que crear y enviar los mensajes y qué acciones debe tomar el servidor y el navegador en respuesta a un comando. Es un protocolo *stateless* (sin estado), porque cada comando se ejecuta independientemente de los anteriores o de los posteriores. Actualmente, la mayoría de los servidores soportan **HTTP 1.1** (**RFC 2616** de junio de 1999). Una de las principales ventajas de esta versión es que soporta conexiones persistentes: una vez que el navegador

se conecta al servidor, puede recibir múltiples ficheros a través de la misma conexión, lo que aumenta el rendimiento de la transmisión hasta en un 20 %.

ISAPI *Internet Server Application Program Interface*

Un API para el servidor Microsoft Internet Information Server. Permite programar aplicaciones web.

ISO *International Organization for Standards*

Organización fundada en 1946, cuyos miembros son las organizaciones nacionales de normalización (estandarización) correspondientes a los países miembros. Entre sus miembros se incluyen a la ANSI (Estados Unidos), BSI (Gran Bretaña), AFNOR (Francia), DIN (Alemania) y UNE (España).

JPEG *Joint Photographic Experts Group*

Formato gráfico de mapas de bit. Incorpora compresión de datos con pérdidas y permite trabajar con 24 bits de color.

JSP *Java Server Pages*

Tecnología de SUN MICROSYSTEMS que permite crear páginas dinámicas en el servidor. Equivale a la tecnología **ASP** de Microsoft. Se programan en *Java*.

MIME *Multipurpose Internet Mail Extensions*

Se usa en el correo electrónico desde 1992 para enviar y recibir ficheros de distinto tipo. Se puede consultar el estándar en **RFC** 1341, **RFC** 1521 y **RFC** 1522.

PNG *Portable Network Graphics*

Formato gráfico de mapas de bit similar a **GIF**. **W3C** ha decidido sustituir **GIF** por **PNG** debido a que el primero emplea un algoritmo que está patentado, mientras que **PNG** es totalmente gratuito. Tanto Microsoft Internet Explorer como Netscape Communicator aceptan este formato.

RFC *Request for Comments*

Medio de publicar propuestas sobre Internet. Cada **RFC** recibe un número. Algunos se convierten en un estándar de Internet.

RGB *Red Green Blue*

Notación de los colores en la que cada color se representa como una combinación de los tres colores básicos (primarios) rojo (*red*), verde (*green*) y azul (*blue*). Se trata de un modelo aditivo (se parte del negro). Mediante la combinación adecuada de los tres colores básicos se consigue todo el espectro de colores. Además de **RGB** existen otras formas de representar los colores. Otra de las más corrientes es CMYK (*cyan, magenta, yellow, black*), que se trata de un modelo sustractivo.

SGML *Standard Generalized Markup Language*

Lenguaje que permite organizar y etiquetar los distintos elementos que componen un documento. Se emplea para manejar grandes documentos que sufren constantes revisiones y se imprimen en distintos formato. Desarrollado y estandarizado por **ISO** en 1986.

TCP/IP *Transmission Control Protocol/Internet Protocol*

Familia de protocolos que se emplean en las comunicaciones de Internet.

URL *Universal Resource Locator*

También conocido como *Uniform Resource Locator*. Sistema de direccionamiento de máquinas y recursos en Internet. Es decir, se trata de una dirección que permite localizar cualquier máquina o documento que se encuentre accesible a través de Internet.

W3C *World Wide Web Consortium*

Consortio internacional de compañías involucradas en el desarrollo de Internet y en especial de la **WWW**. Su propósito es desarrollar estándares y “poner orden” en Internet.

WWW *World Wide Web*

Sistema de servidores web conectados a Internet (no todos los ordenadores conectados a Internet forman parte de la **WWW**). Su protocolo de comunicación es **HTTP**, su lenguaje de creación de documentos **HTML** y sus sistema de direccionamiento de los recursos **URL**. Los navegadores web (*browsers*) permiten navegar por la web.

XHTML *Extensible HyperText Markup Language*

HTML escrito según las normas que marca **XML**. Por tanto, se trata de una aplicación concreta de **XML** y no tienen que confundirse entre sí.

XML *Extensible Markup Language*

Metalinguaje de etiquetado basado en **SGML**. Diseñado específicamente para la **WWW** por **W3C**. Permite que un usuario diseñe sus propias etiquetas, con sus atributos y las reglas de construcción de documentos (sintaxis).

Capítulo 1

Arquitecturas cliente/servidor

Las aplicaciones web son un tipo especial de aplicaciones cliente/servidor. Antes de aprender a programar aplicaciones web conviene conocer las características básicas de las arquitecturas cliente/servidor.

Índice General

1.1. Introducción	2
1.2. Separación de funciones	3
1.3. Modelos de distribución en aplicaciones cliente/- servidor	3
1.3.1. Presentación distribuida	4
1.3.2. Aplicación distribuida	4
1.3.3. Datos distribuidos	4
1.4. Arquitecturas de dos y tres niveles	5
1.5. Descripción de un sistema cliente/servidor	5

1.1. Introducción

Cliente/servidor es una arquitectura de red¹ en la que cada ordenador o proceso en la red es **cliente** o **servidor**. Normalmente, los servidores son ordenadores potentes dedicados a gestionar unidades de disco (servidor de ficheros), impresoras (servidor de impresoras), tráfico de red (servidor de red), datos (servidor de bases de datos) o incluso aplicaciones (servidor de aplicaciones), mientras que los clientes son máquinas menos potentes y usan los recursos que ofrecen los servidores.

Esta arquitectura implica la existencia de una relación entre procesos que solicitan servicios (**clientes**) y procesos que responden a estos servicios (**servidores**). Estos dos tipos de procesos pueden ejecutarse en el mismo procesador o en distintos.

La arquitectura cliente/servidor implica la realización de aplicaciones distribuidas. La principal ventaja de esta arquitectura es que permite separar las funciones según su servicio, permitiendo situar cada función en la plataforma más adecuada para su ejecución. Además, también presenta las siguientes ventajas:

- Las redes de ordenadores permiten que múltiples procesadores puedan ejecutar partes distribuidas de una misma aplicación, logrando concurrencia de procesos.
- Existe la posibilidad de migrar aplicaciones de un procesador a otro con modificaciones mínimas en los programas.
- Se obtiene una escalabilidad de la aplicación. Permite la ampliación horizontal o vertical de las aplicaciones. La **escalabilidad horizontal** se refiere a la capacidad de añadir o suprimir estaciones de trabajo que hagan uso de la aplicación (clientes), sin que afecte sustancialmente al rendimiento general. La **escalabilidad vertical** permite la migración hacia servidores de mayor o menor capacidad y velocidad o de un tipo diferente.

¹Otro tipo de arquitectura de red es *peer-to-peer* (entre pares o de igual a igual), en la que cada ordenador de la red posee responsabilidades equivalentes.

- Posibilita el acceso a los datos independientemente de donde se encuentre el usuario.

1.2. Separación de funciones

La arquitectura cliente/servidor nos permite la separación de funciones en tres niveles, tal como se muestra en la Figura 1.1:

- **Lógica de presentación.** La presentación de los datos es una función independiente del resto.
- **Lógica de negocio (o aplicación).** Los flujos de trabajo pueden cambiarse según las necesidades existentes de un procesador a otro.
- **Lógica de datos.** La gestión de los datos debe ser independiente para poder ser distribuida según las necesidades de la empresa en cada momento.

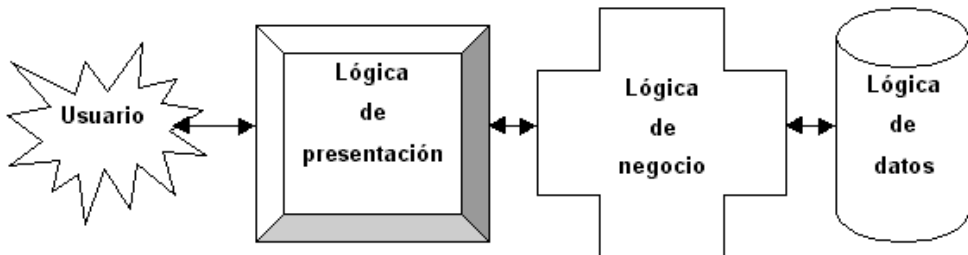


Figura 1.1: Separación de funciones

Si un sistema distribuido se diseña correctamente, los tres niveles anteriores pueden distribuirse y redistribuirse independientemente sin afectar al funcionamiento de la aplicación.

1.3. Modelos de distribución en aplicaciones cliente/-servidor

Según como se distribuyan las tres funciones básicas de una aplicación (presentación, negocio y datos) entre el cliente y el servidor, podemos contemplar

tres modelos: presentación distribuida, aplicación distribuida y datos distribuidos.

1.3.1. Presentación distribuida

El cliente sólo mantiene la presentación, el resto de la aplicación se ejecuta remotamente (Figura 1.2). La presentación distribuida, en su forma más simple, es una interfaz gráfica de usuario a la que se le pueden acoplar controles de validación de datos, para evitar la validación de los mismos en el servidor.

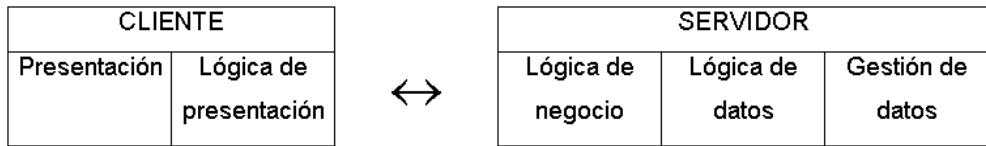


Figura 1.2: Presentación distribuida

1.3.2. Aplicación distribuida

Es el modelo que proporciona máxima flexibilidad, puesto que permite tanto a servidor como a cliente mantener la lógica de negocio realizando cada uno las funciones que le sean más propias, bien por organización, o bien por mejora en el rendimiento del sistema (Figura 1.3).

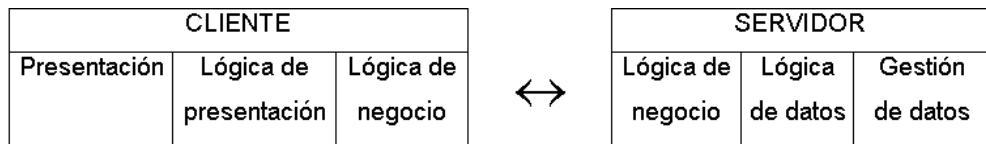


Figura 1.3: Aplicación distribuida

1.3.3. Datos distribuidos

Los datos son los que se distribuyen, por lo que la lógica de datos es lo que queda separada del resto de la aplicación (Figura 1.4). Se puede dar de dos formas: ficheros distribuidos o bases de datos distribuidas.

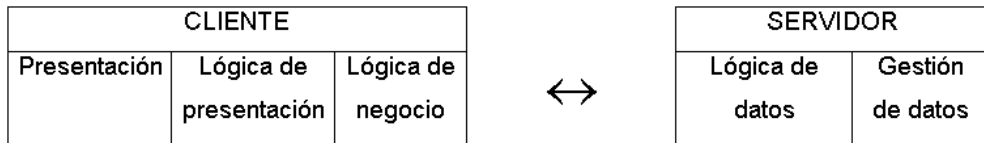


Figura 1.4: Datos distribuidos

1.4. Arquitecturas de dos y tres niveles

La diferencia entre las aplicaciones de dos y tres niveles estriba en la forma de distribución de la aplicación entre el cliente y el servidor.

Una arquitectura de dos niveles está basada en un sistema gestor de bases de datos donde el cliente mantiene la lógica de la presentación, negocio, y de acceso a los datos, y el servidor únicamente gestiona los datos. Suelen ser aplicaciones cerradas que supeditan la lógica de los procesos cliente al gestor de base de datos que se está usando.

En las arquitecturas de tres niveles, la lógica de presentación, la lógica de negocio y la lógica de datos están separadas, de tal forma que mientras la lógica de presentación se ejecutará normalmente en la estación cliente, la lógica de negocio y la de datos pueden estar repartidas entre distintos procesadores.

El objetivo de aumentar el número de niveles en una aplicación distribuida es lograr una mayor independencia entre un nivel y otro, lo que facilita la portabilidad en entornos heterogéneos.

1.5. Descripción de un sistema cliente/servidor

Un sistema cliente/servidor suele presentar las siguientes características:

1. Una combinación de la parte cliente (también llamada *front-end*) que interactúa con el usuario (hace de interfaz entre el usuario y el resto de la aplicación) y la parte servidor (o *back-end*) que interactúa con los recursos compartidos (bases de datos, impresoras, módems).
2. La parte cliente y servidor tienen diferentes necesidades de recursos a la hora de ejecutarse: velocidad de procesador, memoria, velocidad y capacidad de los discos duros, dispositivos de entrada/salida, etc.

3. El entorno suele ser heterogéneo y multivendedor. El hardware y sistema operativo del cliente y el servidor suelen diferir. El cliente y el servidor se suelen comunicar a través de unas API² y RPC³ conocidas (por ejemplo, ODBC⁴ para acceder a bases de datos).
4. Normalmente la parte cliente se implementa haciendo uso de una interfaz gráfica de usuario, que permite la introducción de datos a través de teclado, ratón, lápiz óptico, etc.

² *Application Program Interface*, interfaz de programación de aplicaciones.

³ *Remote Procedure Call*, llamada a procedimiento remoto.

⁴ *Open Database Connectivity*, conectividad de bases de datos abierta.