

# Descubriendo conceptos básicos

---



Programación  
Cliente-Servidor

UNIVERSIDAD  
**SIGLO 21** | MIEMBRO DE LA RED  
**ILUMNO**

# » Introducción a las tecnologías Cliente/Servidor

## Historia y antecedentes

El mundo moderno ofrece una gran división en la clase de sistemas que se utilizan. Vamos a detallar a continuación las dos grandes divisiones: Por un lado, se encuentran las aplicaciones locales, que son aquellas que dependen solo de sí mismas para la resolución de un problema. Este tipo de aplicaciones trabaja de manera coordinada con su sistema operativo, debido a que las tareas que puedan o no realizar están directamente vinculadas con las restricciones que se otorguen tanto a los usuarios finales como a los procesos.

Una aplicación local en función del sistema operativo que se esté utilizando tiene acceso a los recursos locales conforme a las restricciones que se imponen a los usuarios o procesos. En la época moderna es corriente que a cada usuario se le brinden permisos de acceso a los medios, por ejemplo se definen cuotas de disco, procesador, memoria, ancho de banda para el acceso a la red y número de procesos en ejecución concurrente. Estas reglas si bien limitan la operación libre incrementan la seguridad de los sistemas. (Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.)

El otro tipo de aplicaciones, a diferencia de las anteriores, NO se valen de recursos propios, es decir, no dependen solo de sí mismas, sino que dependen de otros sistemas, y sin ellos la aplicación no puede funcionar. En este tipo de aplicaciones que trabajan de manera conjunta y cooperando entre sí, se vuelve una prioridad la utilización de las reglas de

negocios que determinan los sistemas operativos, en este caso, los sistemas operativos de red.

Este tipo de sistemas (los organizados en redes) era raramente utilizado hace unos años. Los usuarios preferían el trabajo local en sus terminales y con un determinado grupo de permisos. No obstante, su desarrollo fue propagándose a lo largo de los años, apoyado por el avance de la tecnología. Lo que más ha contribuido al desarrollo de estos sistemas es lo que se conoce como Internet o Web.

En el mundo moderno, no solo el capital humano es importante en toda empresa, sino que la información que se maneja dentro de la misma se ha puesto a la par del capital humano en lo que a importancia respecta.

Si se toma en cuenta sólo el sector bancario - financiero se ve claramente la transformación que ha sufrido esta industria; ya no manejan sólo dinero, sino también información, la cual debe ser administrada a través de redes y sistemas distribuidos. (Rogelio Niella, 2013, <https://goo.gl/fc7WHa>)

No solo se producen cambios culturales, sino también cambios de hábitos en los clientes bancarios-financieros. Por ejemplo, hoy la mayor parte de las operaciones bancarias puede realizarse vía una computadora en lugar de personalmente, contando, por supuesto, con las ventajas que la utilización de esta nueva metodología otorga.

Si bien esta metodología otorga grandes beneficios a sus usuarios, no es simple para un profesional de sistemas tomar decisiones, fundamentalmente las que atañen a la puesta en marcha.

Para los profesionales del área de sistemas, o las personas que de una u otra manera están relacionadas con ella, resulta difícil la toma de decisiones respecto de la implementación de dichas tecnologías. Qué utilizar frente a todo nuevo proyecto y si deben utilizarse tecnologías cliente/servidor es una de las difíciles tareas a la que se enfrenta todo profesional de sistemas.

Es por ello que en la presente lectura tú encontrarás un enfoque teórico en el que se van a desarrollar todos los conceptos de la arquitectura cliente/servidor que seguramente te serán de suma utilidad en tu vida profesional.

## Arquitectura

### Definición y clasificación de las arquitecturas Cliente/Servidor

Una arquitectura es un entramado de componentes funcionales que aprovechando diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios informáticos, de manera que pueden ser utilizados eficazmente dentro de la organización. (Arieto Gloria, 2009, <https://goo.gl/qVlbvs>)

Todas las organizaciones basan el concepto de sus sistemas de información en infraestructuras. Estas infraestructuras siempre han estado estrechamente vinculadas al tipo de empresa en cuestión; es decir, si la empresa presentaba una estructura de tipo centralizada, por departamentos y jerarquía, sus sistemas giraban en torno a ese tipo de problemática.

Los nuevos modelos (los de los sistemas actuales) giran en torno a otro tipo de organización, donde en lugar de departamentos se utilizan unidades operativas que trabajan en función de objetivos o tareas comunes.

No obstante, no todo son desventajas en referencia a los sistemas más viejos y centralizados; las empresas hoy están tendiendo a descentralizar algunas partes y centralizar otras.

Toda esta introducción nos ayuda a poder comprender el porqué de las dos temáticas que se tratarán a continuación: arquitecturas centralizadas y arquitecturas distribuidas.

Una arquitectura centralizada es aquella que presenta un servidor central en el que se encontrarán todos los datos.

Una arquitectura distribuida es aquella que presenta un tratamiento de tipo distribuido, es decir, en donde el trabajo no se centra solo en una máquina sino en varias, que son determinadas por el área correspondiente.

## Arquitectura centralizada

Tal como se mencionó con anterioridad, este tipo de arquitecturas son las que se conocen también como "tradicionales". Su principal característica es que son jerárquicas, centralizadas y divididas por departamentos.

Si se tiene en cuenta que cada departamento puede realizar sus propias tareas, sistemas con arquitecturas de este tipo no presentan una fácil interrelación con otros sistemas de otras áreas o departamentos.

El sistema aquí será único y los datos estarán alojados en un determinado servidor al que tendrán acceso los usuarios del departamento al que se haga referencia. Sus principales características son:

- Existirá una sola computadora que contendrá todos los datos, al presentarse este tipo de solución, esta computadora deberá ser provista de todos aquellos elementos de hardware que permitan el trabajo sin inconvenientes. Computadoras de este tipo deberán ser muy potentes.
- Existirá una sola computadora que controlará el acceso de las terminales (que no realizarán tareas más allá de ser terminales esclavos).
- Tanto las impresoras como las terminales se encontrarán con conexión a la máquina central.
- Existirá una sola computadora que será la responsable de la ejecución de los procesos.

Las principales ventajas y desventajas que se presentarán, estarán directamente vinculadas con sus características. La misma característica en una arquitectura podrá provocar tanto ventajas como desventajas. Se detallan a continuación:

### Ventajas

- Existirá una sola computadora que contendrá todos los datos, esto producirá alta disponibilidad y alto rendimiento en el procesamiento de las aplicaciones al ser la computadora tan potente.
- Existirá una sola computadora que controlará el acceso de las terminales. Esto facilitará y hará un mejor y más simple control de los usuarios y hará que se presente un alto nivel de seguridad.

- Tanto las impresoras como las terminales se encontrarán con conexión a la máquina central. Esto provocará una excelente compartición de recursos, aprovechando las máquinas que se puedan compartir.
- Existirá una sola computadora que será la responsable de la ejecución de los procesos. Esto provocará que se tenga que controlar una sola máquina que es la que alojará todos los datos y procesos a ejecutar.

### Desventajas

- Existirá una sola computadora que contendrá todos los datos. Si bien es esta ventaja importante, al requerirse que el trabajo se procese todo en esta máquina es que se demandarán características de hardware muy potentes para poder dar solución a lo que los usuarios soliciten.
- Existirá una sola computadora que controlará el acceso de las terminales (que no realizarán tareas más allá de ser terminales esclavos). Esto puede provocar, ante una caída del servidor, que las terminales queden sin poder trabajar hasta el re-establecimiento del mismo.
- Existirá una sola computadora que será la responsable de la ejecución de los procesos. Esto provocará una alta dependencia de las comunicaciones. En caso de caída de una línea, todos los puestos de trabajo dependientes de dicha línea quedarán inoperantes.

## Arquitectura distribuida

Este tipo de arquitecturas, a diferencia de la arquitectura anterior, presenta características de computadoras separadas, en donde cada máquina tendrá la capacidad suficiente para realizar tareas por sus propios medios.

El término arquitecturas distribuidas, hace referencia a una serie de procedimientos, políticas y requerimientos aplicados a la construcción de un sistema distribuido con objeto de unificar y simplificar el diseño, facilitando así la construcción y mantenimiento del mismo, y estandarizando su desarrollo, reduciendo costos y reutilizando componentes. ( Universidad Politécnica de Cartagena, 2013, <https://goo.gl/SXgCSZ>)

Enunciamos, a continuación, sus principales características:

- No existirá una sola computadora que contenga todos los datos, sino que cada computadora tendrá capacidad suficiente para realizar un trabajo autónomo.
- Al realizar cada computadora su trabajo de manera independiente, cobrará vital importancia la red de comunicación de datos.
- Al realizar cada computadora su trabajo de manera independiente, el usuario obtendrá mejores tiempos de respuesta.
- Si bien las terminales trabajarán de manera independiente, aquellos recursos que no estén disponibles se podrán tomar desde la red; radica en ello la importancia de la comunicación.
- No obstante, y si bien esta arquitectura se apoya en descentralizar aquellas tareas complejas o medianas complejas, algunas tareas serán delegadas al servidor central.

Al igual que se detalló en la arquitectura anterior, las principales ventajas y desventajas que se presentarán estarán directamente vinculadas con sus características. La misma característica en una arquitectura podrá provocar tanto ventajas como desventajas. Se detallan a continuación:

## Ventajas

- No existirá una sola computadora que contenga todos los datos, sino que cada computadora tendrá capacidad suficiente para realizar un trabajo autónomo. Esto provocará una mayor independencia en los terminales y hará que, ante la caída del servidor central, el trabajo pueda temporalmente seguir realizándose. También hará que los sistemas de información lleguen a todos los departamentos de la empresa.
- Al realizar cada computadora su trabajo de manera independiente, el usuario obtiene mejores tiempos de respuesta. Se entiende que un terminal que trabaje independiente responderá mucho que si dependiese de un servidor central, haciendo que su trabajo sea más flexible y potente.

## Desventajas

- Al realizar cada computadora su trabajo de manera independiente, cobra vital importancia la red de comunicación de datos. Esto presupone un intenso flujo de informaciones (muchas veces no útiles, como pantallas y datos incorrectos) dentro de la red, lo que puede elevar los costos de comunicaciones.
- No obstante, y si bien esta arquitectura se apoya en descentralizar aquellas tareas complejas o medianas complejas, algunas tareas serán delegadas al servidor central. Si bien se tiende a descentralizar, no se puede delegar en un terminal una tarea (como por ejemplo, alojar los datos) que siempre, independientemente de la estructura, estará directamente ligada al servidor.
- Supone una mayor complejidad.

## Situación actual

Como ya se ha dicho anteriormente, una arquitectura no reemplaza a la otra. Hoy la mayoría de las grandes empresas maneja una parte de cada una de ellas, lo que permite utilizar las ventajas de ambas.

Muchas veces una empresa encuentra diversas ventajas con la utilización de una arquitectura centralizada y otras veces no tanto. Es por ello que tú, como profesional de sistemas, deberás tener la suficiente capacidad de

analizar la problemática de la empresa y sugerir en qué casos aconsejarías utilizar una arquitectura y en qué casos la otra.

El objetivo a cubrir por toda organización es desarrollar una infraestructura de sus sistemas de información y comunicaciones, que permita construir sistemas de información que puedan evolucionar tan rápidamente como evolucionan las formas de negocio, y que se adecúen a las necesidades de nuevos servicios tan pronto como estas necesidades aparezcan. (Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.)

## Introducción a tecnologías Clientes/Servidor

Tomando el concepto descripto con anterioridad en referencia a las arquitecturas y sus diferentes tipos, podemos decir que cliente/servidor es una arquitectura de red en la que cada máquina o proceso en la red es cliente o servidor.

El modelo cliente/servidor tiene su base en la arquitectura distribuida, en la que las terminales -de ahora en adelante, los "clientes"- requerirán de un trabajo que realizarán otras computadoras centrales -de ahora en adelante, los "servidores"-. (Ver figura Modelo Cliente-Servidor en página 9).

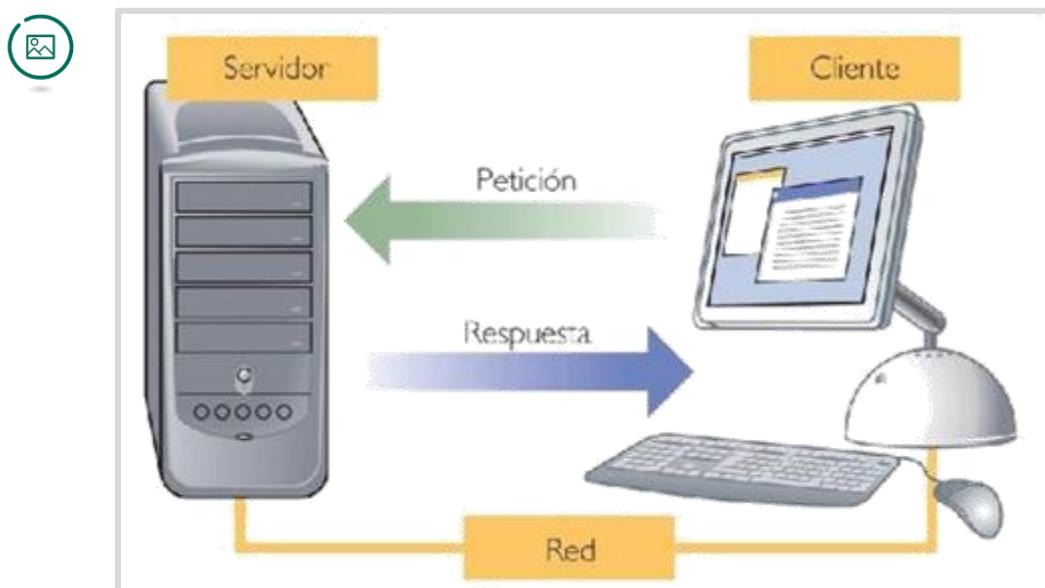
Si bien el concepto cliente/servidor es muy genérico, y puede ser entendido incluso en el ámbito de una sola máquina, donde unas aplicaciones pueden prestar servicio a otras, su significado desde el punto de vista informático suele presuponer la existencia de varias máquinas (al menos dos) unidas en una red. (Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.)

Por la característica del trabajo que realizarán, usualmente los servidores son máquinas grandes, potentes, mientras que los clientes no lo son tanto y usan los recursos que ofrecen los servidores.

Ya anteriormente se comentó la importancia en esta arquitectura de la comunicación, en el gráfico que se detalla a continuación se puede observar el porqué.

Sin esta comunicación, quien solicite servicios (el cliente) no podrá acceder a quien tiene que responder por ellos (el servidor).

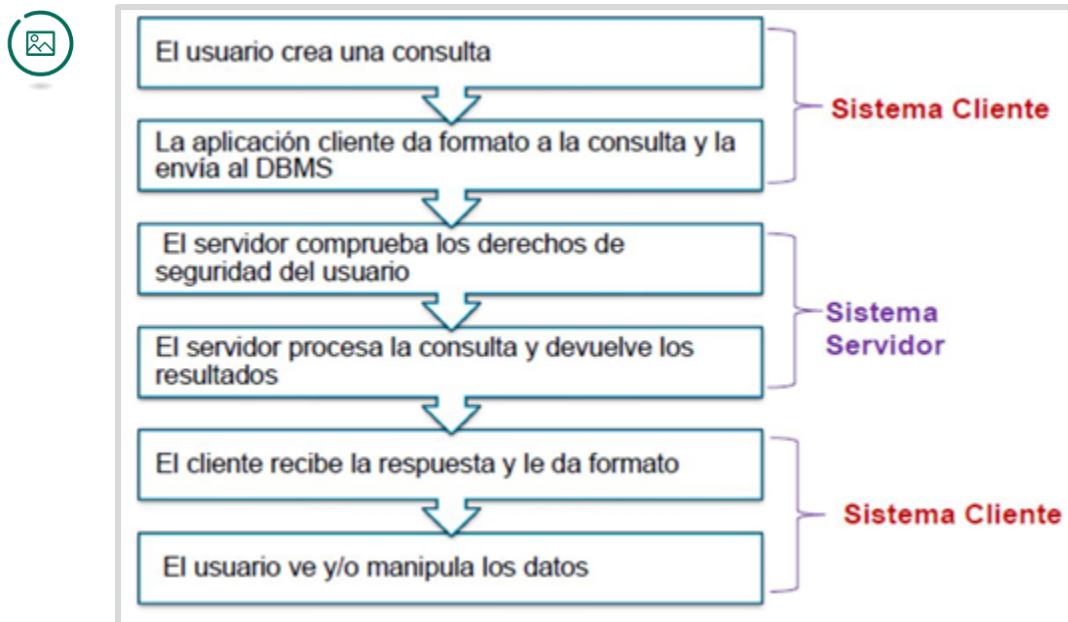
**Figura 1: Modelo cliente servidor**



Fuente: [Imagen sin título sobre Modelo Cliente Servidor]. (s. f.). Recuperado de:  
<http://goo.gl/tCBnHa>

Del gráfico anterior se desprende que los clientes serán los encargados de la interacción con el usuario, generalmente de manera gráfica, existiendo luego procesos que se encargarán de establecer la comunicación (enviando el pedido y recibiendo luego la respuesta), y que del otro lado existirá otra máquina que se encargue de procesar las consultas. Se resume a continuación todo lo expresado anteriormente mediante el siguiente gráfico.

Figura 2: Modelo cliente servidor



Fuente: [Imagen sin título sobre Modelo Cliente Servidor]. (s. f.). Fuente: <https://goo.gl/uuv1qP>

## Componentes esenciales de la infraestructura Cliente/Servidor

Todo modelo cliente/servidor tiene su base fundamental en la idea del servicio, en el que un cliente cumple un rol activo solicitándolo y un servidor cumple el rol pasivo entregándolo; unidos por protocolos que permiten que se produzca tal intercambio.

A continuación, se describen: clientes (con sus características y tipos), servidores (con sus características y tipos) y el middleware (con sus características y tipos).

### Cliente, características y tipos

Un cliente es todo proceso que reclama servicios de otro.  
Una definición un poco más elaborada podría ser la

siguiente: cliente es el proceso que permite al usuario formular los requerimientos y pasarlo al servidor. Es una aplicación informática que se utiliza para acceder a los servicios que ofrece un servidor, normalmente a través de una red de telecomunicaciones. Se lo conoce con el término front-end. (Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.)

Un cliente es quien toma contacto directo (a través de interfaces gráficas) con el usuario; es el responsable de iniciar las solicitudes, es quien este tipo de esquema desempeña un rol activo.

En sus comienzos, se utilizaba este término para máquinas que no eran capaces de ejecutar programas por sí mismas, sino que solamente se conectaban, realizaban la petición y mostraban los resultados. Su principal ventaja en este aspecto era su bajo costo. Hoy, además de en razón de su economía, se utiliza para obtener datos externos o para utilizar recursos de los que no se dispone en la máquina local.

Como se podrá comprender, el cliente más utilizado es el navegador web.

### **Características y funciones principales de un cliente**

- 1)** Interactúa con el usuario captando los datos necesarios para iniciar la solicitud, administrando en este paso lo que se conoce como la interfaz del usuario.
- 2)** Valida localmente los datos que vaya ingresando el usuario.
- 3)** Procesa la lógica de la aplicación en la captura de los datos.
- 4)** Algunos tipos almacenan datos.
- 5)** Genera los requerimientos al servidor
- 6)** Procesa las respuestas recibidas.
- 7)** Formatea los resultados.
- 8)** Muestra resultados al usuario.

Los clientes se dividirán según la complejidad que tengan que atender.

**Tabla 1: Tipos de clientes**



	Almacenamiento de datos local	Proceso de datos local
Cliente pesado	Si	Si
Cliente híbrido	No	Si
Cliente liviano	No	No

### **Cliente liviano o flaco**

Un cliente flaco se caracteriza por tener muy reducida su capacidad de procesamiento, simplificándose su tarea a la de tomar datos, enviarlos al servidor y mostrar los resultados que se obtengan de tal consulta.

Si se tienen en cuenta las características detalladas anteriormente, se reducirían a las siguientes:

- Interactúan con el usuario captando los datos necesarios para iniciar la solicitud, administrando en este paso lo que se conoce como la interfaz del usuario.
- Generan requerimientos al servidor.
- Procesan de las respuestas recibidas.

Ejemplo de estos clientes pueden ser los primeros navegadores web.

### **Cliente pesado o gordo**

Un cliente pesado o gordo se caracteriza por tener capacidad no solo para realizar solicitudes, sino también poder procesarlas y almacenarlas; no obstante, por sí solo no puede realizar todas las funciones que lo convertirían en autónomo.

Si se tienen en cuenta las características detalladas anteriormente, se puede decir que los clientes de este tipo realizan todas las tareas (desde la Nº 1 a la 8).

Ejemplo de estos pueden ser los clientes de correo, que si bien pueden recibir mensajes y redactar nuevos, siguen necesitando de un servidor para chequear los mismos.

### **Cliente híbrido**

Un cliente híbrido es una mezcla de los dos clientes anteriores; puede decirse que es muy similar a un cliente pesado, con la clave fundamental de que NO almacenan datos, sino que solamente los procesan.

Si se tienen en cuenta las características detalladas anteriormente, se puede decir que los clientes de este tipo realizan todas las tareas, a excepción de la tarea Nº 4, que es la de almacenamiento de los datos.

### **Servidor, características y tipos**

Un servidor es todo proceso que proporciona un servicio a otros. Es el proceso encargado de atender a múltiples clientes que hacen peticiones de algún recurso administrado por él.

Al proceso servidor se lo conoce con el término back-end. El servidor normalmente maneja todas las funciones relacionadas con la mayoría de las reglas del negocio y los recursos de datos. (Núñez Madrid, 2009, <https://goo.gl/vDJItZ>.)

Generalmente, los servidores están al "servicio" de otras aplicaciones (los clientes); su rol puede considerarse como pasivo, ya que NO son quienes inician una solicitud, sino quienes la contestan. Presentan, además, la particularidad de poder prestar o aceptar solicitudes simultáneamente de un sin número de clientes.

Usualmente se asocia la palabra servidor con una máquina potente, aunque no en todos los casos esto es lo que se presentará.

Siempre deberás analizar qué utilización se dará al servidor para luego recomendar la máquina que mejor se adapte a las necesidades.

#### **Características de un servidor:**

- 1)** Recepción de los requerimientos de los clientes
- 2)** Procesamiento de tales requerimientos

- 3) Realización de validaciones a nivel de base de datos, y no locales como realizan los clientes.
- 4) Formateo de datos para otorgar las respuestas a los clientes
- 5) Manejo de accesos concurrentes
- 6) Almacenamiento de la información

### Tipos de servidores

Los servidores se dividen en grandes grupos, dependiendo de la tarea a la que cada uno ha sido encomendado.

- Servidor de impresiones: maneja toda la información referente a las impresiones. Sus funciones principales son las de receptar los trabajos de impresión de los usuarios y otorgarles el correspondiente orden para que la impresión pueda ser realizada.
- Servidor de correo: es el responsable del manejo del correo dentro de una empresa. Se encarga de mover y almacenar el correo, así como también de enviar, recibir, enrutar y realizar otras operaciones relacionadas con e-mails para los clientes de la red.
- Servidor de archivos: almacena los archivos comunes dentro de una organización a los que luego cada usuario podrá acceder según los permisos que tenga sobre los mismos. Este tipo de servidores es ampliamente utilizado en las empresas para compartir archivos.
- Impresoras como servidores: las nuevas impresoras son muchas veces capaces por sí mismas de actuar como parte de una red, sin la necesidad de estar conectadas. Su poderoso software permite este tipo de configuración.
- Servidores FTP (siglas en inglés de File Transfer Protocol, 'Protocolo de Transferencia de Archivos'): es uno de los servicios que más se utiliza en Internet. Un servidor de este tipo nos ayuda a subir y bajar archivos. Existe en la actualidad una gran cantidad de programas que permiten conectarse a estos servidores y subir y bajar archivos sin ningún tipo de inconvenientes.
- Servidores de bases de datos: son los servidores por excelencia dentro de un entorno cliente/servidor, ya que son los que almacenan la información. Este tipo de servidores se caracterizan por ser máquinas potentes para poder contener la información y responder adecuadamente a los servicios que se le soliciten
- Servidor de telefonía: realiza funciones relacionadas con la telefonía.
- Servidor web: en la actualidad es uno de los tipos de servidores más populares. Es el responsable de recibir los pedidos de los clientes

mediante el protocolo HTTP (Del inglés, Hypertext Transfer Protocol, o “protocolo de transferencia de hipertexto” en español) y devolverlos a los clientes.



## Referencias

**Arieto Gloria, A. A.** (2009). *Información, Informática e Internet: del ordenador personal a la Empresa 2.0*. España: Visión Libros.

**Cartagena, U. P.** (s.f.). Arquitecturas Distribuidas. Recuperado de  
<http://ait.upct.es/asignaturas/ad/teoria/curso09/Tema-1-Intro-conceptos-basicos.pdf>

[Imagen sin título sobre Modelo Cliente Servidor]. (s. f.). Recuperado de::  
<https://goo.gl/uuv1qP>

[Imagen sin título sobre Modelo Cliente Servidor]. (s. f.). Recuperado de:  
<http://goo.gl/tCBnHa>

**Niella, R.** (s.f.). Resumen: Soluciones Transaccionales. Recuperado de:  
[http://www.oocities.org/ar/r\\_niella/Document/resumen.htm](http://www.oocities.org/ar/r_niella/Document/resumen.htm)

**Núñez, R.** (2009). Cliente-Servidor. Recuperado de:  
<http://es.scribd.com/doc/44110204/Cliente-Servidor-1>

# Concepto

---



Programación  
Cliente-Servidor

UNIVERSIDAD

**SIGLO 21**

MIEMBRO DE LA RED  
**ILUMNO**

# » **Middleware**

Recuerda, para que se pueda establecer el vínculo entre un cliente y un servidor es necesaria una infraestructura de comunicaciones.

En su definición más simple, middleware es la interfaz que provee la conectividad entre aplicaciones clientes y aplicaciones servidoras, y entre aplicaciones y bases de datos. Es una capa de software que protege a los desarrolladores de tener que manejar detalles de bajo nivel de diferentes protocolos de comunicación, sistemas operativos y arquitecturas de bases de datos. Este tipo de interfaces incluyen API's, PRC's, Pipes, mensajería de red y accesos a bases de datos. (Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.)

## Características de un Middleware

Las principales características de un middleware son las siguientes:

- Es el responsable directo de realizar el acceso a los datos; sus principales actividades deben concentrarse en aceptar las consultas (que fueron enviadas por el cliente y transmitidas a través de la red) y devolverlas con sus correspondientes códigos de error en caso de que así corresponda.
- Tiende a simplificar el desarrollo de las aplicaciones.
- Utilizar un middleware en una empresa, implica utilizar paquetes específicos de software para el desarrollo de estos módulos. Esto sujetará a la empresa a un suministrador y a su política de actualización del producto, que puede ser distinta que la de actualización de los sistemas operativos con los que se comunica el módulo middleware.

## Tipos de Middleware

Existen dos tipos de middleware:

- 1) Middleware general: este tipo permite la impresión de documentos remotos, manejos de transacciones, autenticación de usuarios, etc. Es el sustrato de la mayoría de las interconexiones de cliente/servidor. Incluye las pilas de comunicación, directorios distribuidos, servicios de autenticación, llamadas a procedimientos remotos y servicios en cola. Esta categoría incluye también las extensiones del sistema operativo redes, como los servicios distribuidos de archivos e impresión. Entre los productos que pertenecen a esta categoría tenemos: NetWare, Named Pipes, TCP/IP y NetBios.
- 2) Middleware de servicios específicos: generalmente, trabaja orientado a mensajes. Trabaja una sola transacción a la vez, es necesario para cumplir con tipos particulares de servicios. Pertenece a esta categoría: el middleware para base de datos como ODBC, el middleware para OLTP como ATMI y TxRPC.



## Referencias

Núñez, R. (2009). <http://es.scribd.com/doc/44110204/Cliente-Servidor-1>

# Ventajas e inconvenientes

---



Programación  
Cliente-Servidor

UNIVERSIDAD

**SIGLO 21**

MIEMBRO DE LA RED  
**ILUMNO**

# » Ventajas e inconvenientes de la aplicación Cliente / Servidor

## Ventajas

- Como las aplicaciones se distribuyen entre los clientes, en la mayoría de los casos el servidor suele ser una máquina no muy potente.
- Los clientes se comunican a través de todas interfaces gráficas y a su vez interactivas. Esto provoca una reducción de tiempo de aprendizaje.
- Brinda a los departamentos, soluciones locales pero totalmente integradas a nivel global.
- Proporcionan un mejor acceso a los datos.
- Al trabajar de manera local algunos procesos, eliminan la necesidad de mover grandes bloques de información por la red hacia los servidores.
- Existe independencia física y lógica entre clientes y servidores.
- Permiten centralizar el control de sistemas que estaban descentralizados, como por ejemplo la gestión de los computadores personales que antes estuvieran aislados.

## Inconvenientes

- Existe una complejidad importante dado que se tiene que integrar una gran variedad de productos.
- Se dificulta asegurar la seguridad ya que existirá una red de clientes y servidores con un sistema único centralizado. Se deben hacer verificaciones en el cliente y en el servidor.
- Problemas de congestión de red pueden provocar un bajo rendimiento en las aplicaciones de los clientes.
- Deben tenerse en cuenta aspectos que en aplicaciones locales no se tienen, como por ejemplo: clientes y servidores deben utilizar un mismo mecanismo (sockets o RPC).
- Deben existir estrategias para la consistencia de los datos.

## ¿Qué ventajas puede aportar el esquema cliente/servidor a las empresas?

Inicialmente podemos decir que estas arquitecturas tienden a la reducción de costos directos en la producción de software y que, por lo tanto, este es uno de sus principales aportes a las empresas.

El aporte a la reducción económica puede tomarse como un aporte directo, pues en toda nueva construcción pueden usarse los servidores que hay disponibles, reduciéndose el desarrollo solamente a la elaboración de los procesos del cliente, según los requerimientos deseados.

Asimismo permiten llevar información al lugar de la empresa donde se necesite, informatizando de esta forma áreas que pudiesen originalmente no tener sistemas.

Otro aspecto, no menor, es que favorece la adaptación tecnológica.



## Referencias

**De Luca, Damián** (2013). *APPs HTML5 para aplicaciones móviles*. Buenos Aires, Argentina: Alfa Omega.

**Mora, Sergio Luján** (2002). *Programación de Aplicaciones WEB: Historia, Principios Básicos y Clientes Web*. Alicante, España: Club Universitario.

# Clasificación y características

---



Programación  
Cliente-Servidor

UNIVERSIDAD  
**SIGLO 21** | MIEMBRO DE LA RED  
**ILUMNO**

## » Clasificación de modelos Cliente/Servidor

Es muy importante, para que tú como profesional puedas recomendar si conviene o no utilizar en una organización el modelo cliente/servidor, entender, además de sus componentes principales (que detallamos con anterioridad), cómo se dividen los mismos partiendo de diferentes aspectos.

A continuación, te presentamos las clasificaciones:

- 1) Por tamaño de sus componentes.
- 2) Por planos de software o de hardware.
- 3) Por la naturaleza del servicio.

### **Por tamaño de componentes, Fat Client (Thin Server), Fat Server (Thin Client)**

#### **Fat Client (Thin Server)**

En este tipo de clasificación, la mayor parte de la aplicación se corre del lado del cliente, por eso su nombre de "cliente gordo".

La principal función se relega al procesamiento y provisión de la base de datos.

Arquitecturas de estas características se recomiendan si tú tienes que dar soporte a sistemas de información gerencial.

#### **Fat Server (Thin Client)**

Contrariamente a lo que se detalló en la clasificación anterior, este tipo de arquitectura basa su nombre de "cliente flaco" en que el cliente sólo hace la interfaz con el usuario. Todo el proceso pesado se corre del lado del servidor.

Por otra parte, a diferencia del anterior, este tipo de arquitecturas es ampliamente recomendado en casos de sistemas transaccionales.

## Por planos o capas (Tier), planos a niveles de software, planos a niveles de hardware

A diferencia de la clasificación anterior, donde se detalló una categorización por tamaño de componentes, aquí se basará en lo que se conoce como planos o tiers.

Debido a su gran diferencia, los planos serán clasificados primero en planos de software y luego en planos de hardware.

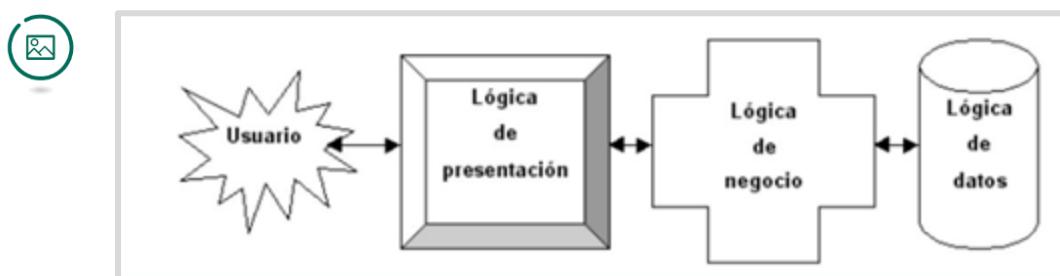
Antes de dar comienzo a esta explicación, es importante tener claros los conceptos de la distribución de funciones y qué significan las divisiones en dos y tres planos que se verán con frecuencia en todos los gráficos y explicaciones que se presentan en esta clasificación.

### Separación de funciones

La arquitectura cliente/servidor permite la separación de funciones en tres niveles:

- Lógica de presentación: sus principales actividades son las de obtener información del usuario, enviarla para su procesamiento y recibir luego los resultados de dicho trabajo.
- Lógica de negocio (o aplicación): su principal tarea es la de recibir la entrada del nivel de presentación e interactuar con la lógica de datos para ejecutar las reglas de negocio que tiene que cumplir la aplicación, y enviar el resultado del procesamiento al nivel de presentación.
- Lógica de datos: su principal tarea es almacenar los datos, recuperarlos, mantenerlos y asegurar la integridad de los mismos.

**Figura 1: Separación de funciones**



Fuente: Sergio Luján Mora (2002). Página 75

### Arquitectura de dos y tres planos

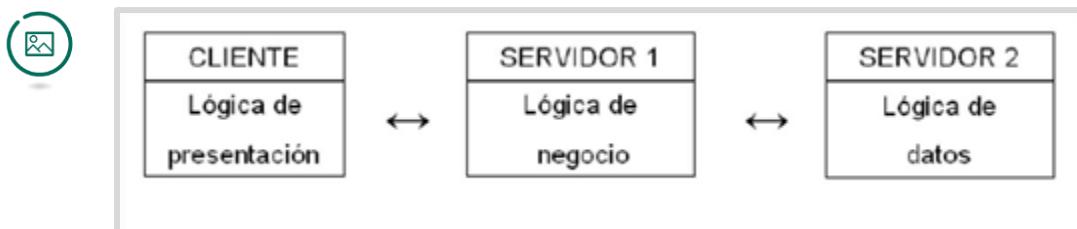
Sencillamente, se puede decir que la diferencia entre las arquitecturas de dos y tres planos radica en la forma en que se distribuyen las aplicaciones entre el cliente y el servidor.

Tomando en cuenta la explicación anterior (de la separación de funciones), una aplicación es de dos niveles cuando el cliente maneja la lógica de presentación, de negocio y de acceso a los datos, y el servidor únicamente gestiona los datos.

Siguiendo también la explicación anterior, en las arquitecturas de tres niveles la lógica de presentación, la lógica de negocio y la lógica de datos están separadas en distintos servidores, tal como se puede ver en el gráfico que se adjunta a continuación.

En esta arquitectura, la lógica de negocio y la de datos pueden estar repartidas entre distintos procesadores.

**Figura 2: Arquitectura de tres niveles**



Fuente: Sergio Luján Mora (2002). Pág.77.

### Plano a niveles de software

Este enfoque o clasificación es el más generalizado y el que más se ajusta a los enfoques modernos, dado que se fundamenta en los componentes lógicos de la estructura cliente/servidor y en la madurez y popularidad de la computación distribuida. Por ejemplo, esto permite hablar de servidores de aplicación distribuidos a lo largo de una red, y no tiene mucho sentido identificar a un equipo de hardware como servidor, sino más bien entenderlo como una plataforma física sobre la cual pueden operar uno o más servidores de aplicaciones. (Núñez Madrid, 2009, <https://goo.gl/vDJltZ>)

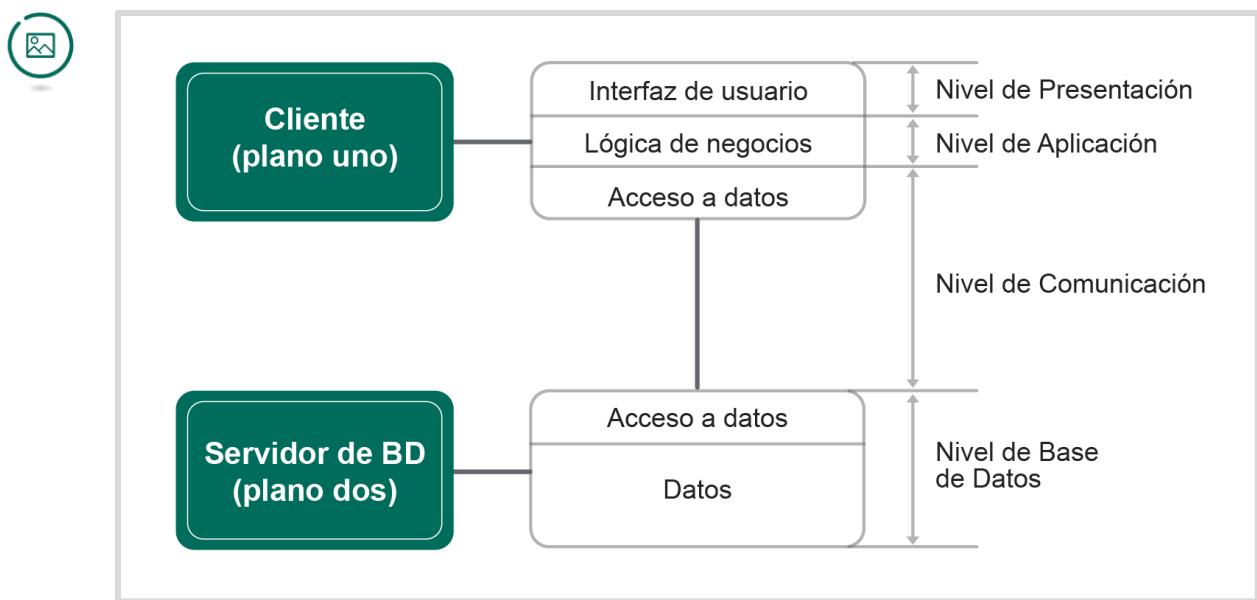
### Cliente/servidor dos planos

En este tipo de planes, como dijimos, el cliente maneja la lógica de presentación, de negocio y de acceso a los datos, y el servidor únicamente gestiona los datos.

Dependiendo de dónde se localice el grupo de tareas correspondientes a la lógica de negocios, se pueden tener a su vez dos tipos distintos dentro de esta misma categoría:

- 1) Implementado con SQL Remoto. Aquí el cliente (representado por el plano uno) envía las consultas/requerimientos al servidor de bases de datos (representado por el plano dos). El resultado vuelve a través de la red. La figura siguiente representa lo expresado anteriormente.

**Figura 3: Implementado con SQL Server**

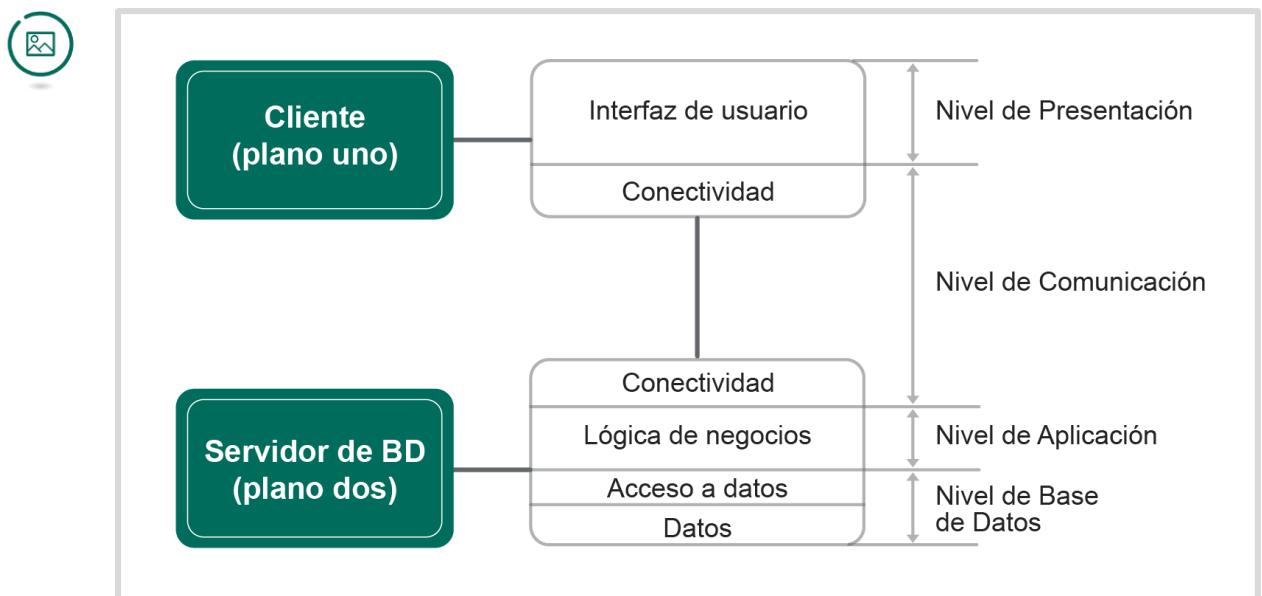


Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJltZ>

- 2) Implementado con procedimientos almacenados. Aquí también el cliente (plano uno) envía las comunicaciones al servidor, luego el nivel de comunicación la gestiona y el servidor la devuelve. La diferencia con el proceso anterior es que el cliente envía llamadas a funciones que residen en la base de datos, y es la base de datos la que resuelve y procesa la totalidad de las instrucciones SQL. Si bien esta metodología es ampliamente utilizada, muchas veces los desarrolladores centralizan todo el trabajo en un determinado motor de bases de datos, pero a la hora de migrar se presenta la dificultad de compatibilización, y es allí

donde el sistema se vuelve dependiente y resulta muy difícil su actualización.

**Figura 4: Implementado con procedimientos almacenados**



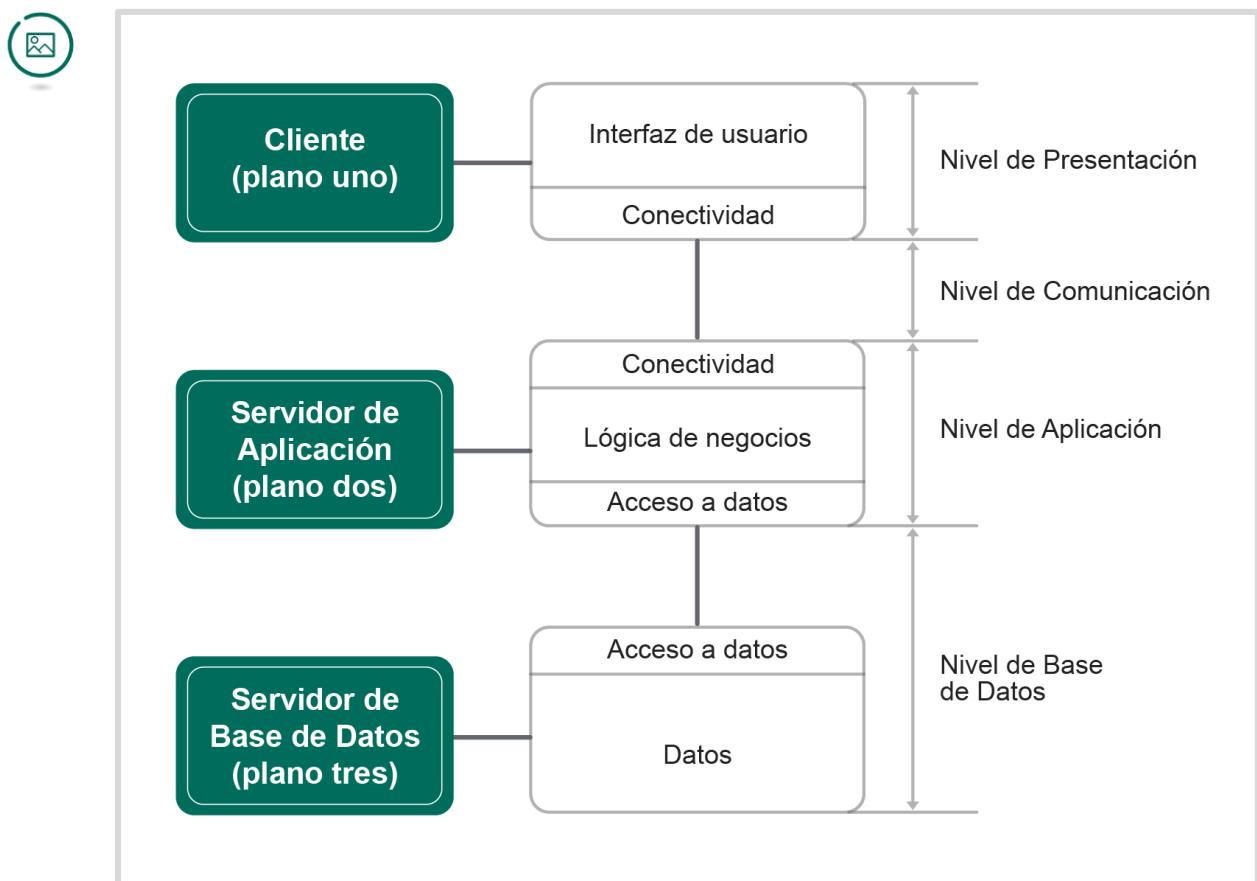
Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.

### Cliente/servidor tres planos

A diferencia de lo anterior, en esta estructura, además de las dos capas principales de software, se agrega otra capa más, que es la capa correspondiente al servidor de base de datos.

El cliente es quien inicia la petición, el servidor las recibe, administra y las responde. Dependiendo del tipo de solicitud, quien accede y se conecta con la base de datos es directamente el servidor.

Figura 5: Cliente/servidor tres planos



Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJItZ>.

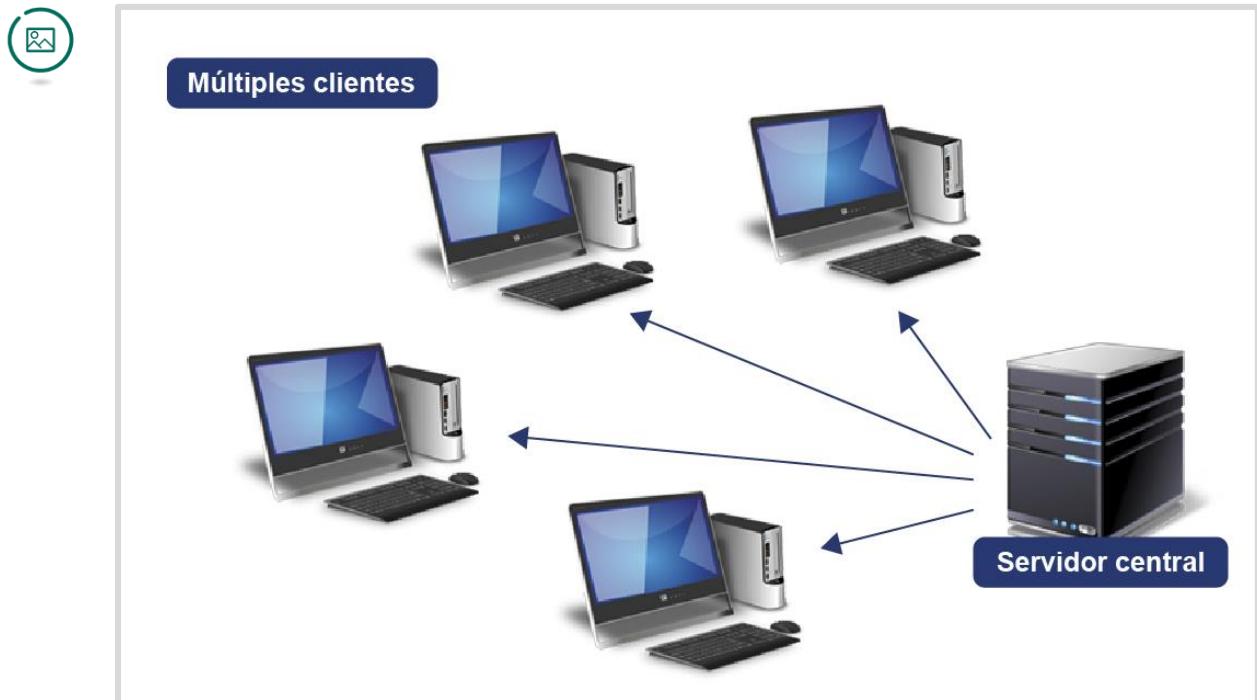
## Planos a niveles de hardware

Si bien se tendrá en cuenta qué tareas realizarán cliente y servidor, haremos aquí un enfoque especial en la parte física (hardware) de los dispositivos.

### Cliente/servidor dos planos

Sobre este tipo de arquitecturas se apoyan los sistemas de gestión y operativos donde los clientes se conectan a un servidor que es el encargado de recibir y contestar las solicitudes. Estos sistemas son los más conocidos.

Figura 6: Cliente/servidor dos planos

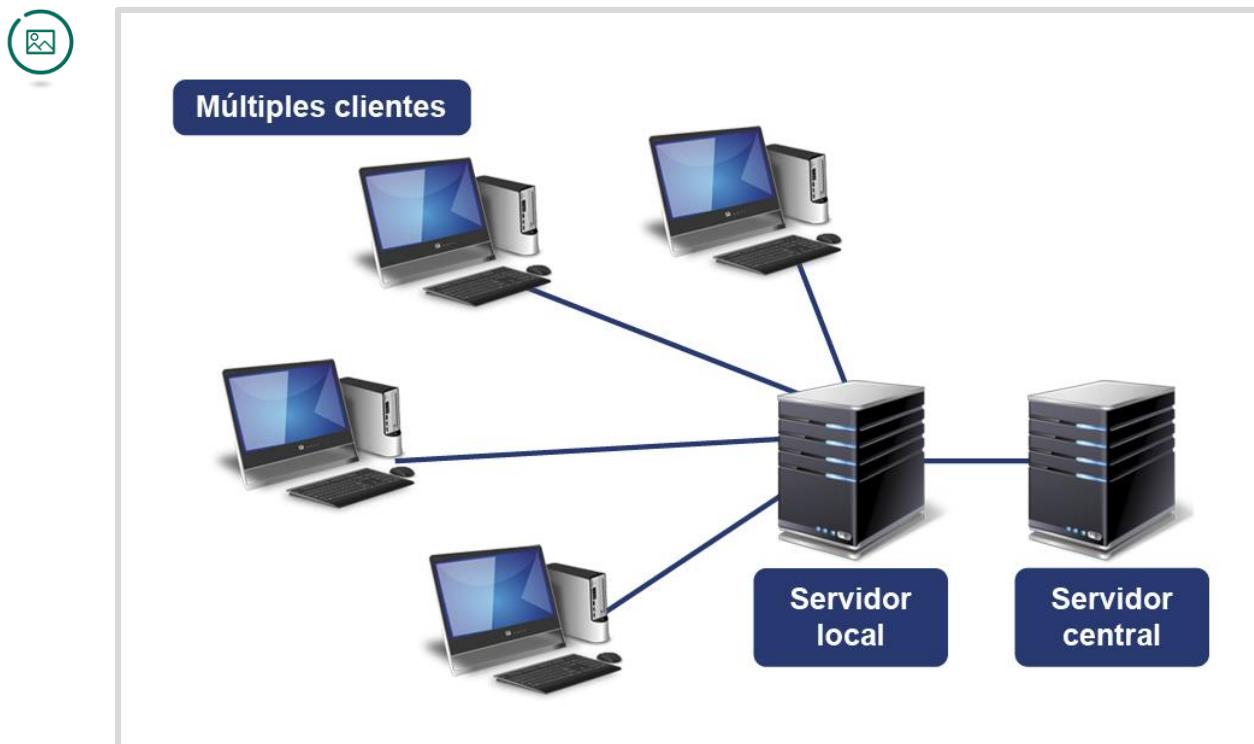


Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJtZ>.

#### Cliente/servidor tres planos

En esta arquitectura los clientes son conectados vía red a un servidor (generalmente local) y luego este servidor local se conecta a otro servidor (generalmente central).

Figura 7

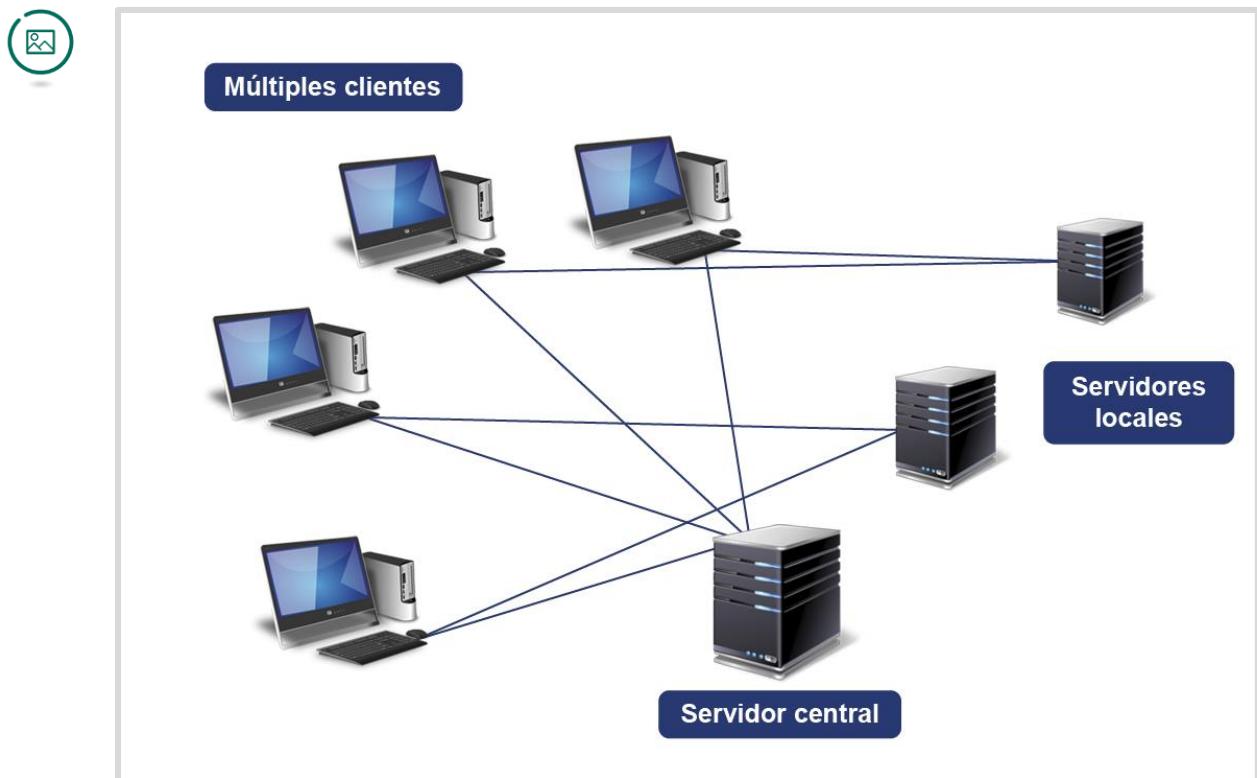


Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJtZ>.

#### Cliente/servidor múltiples planos

En este esquema se permite que las PCs cliente puedan conectarse directamente a los servidores centrales, evitando de esta manera el paso previo de conexión a un servidor local luego a otro remoto.

Figura 8: Cliente/servidor multi planos

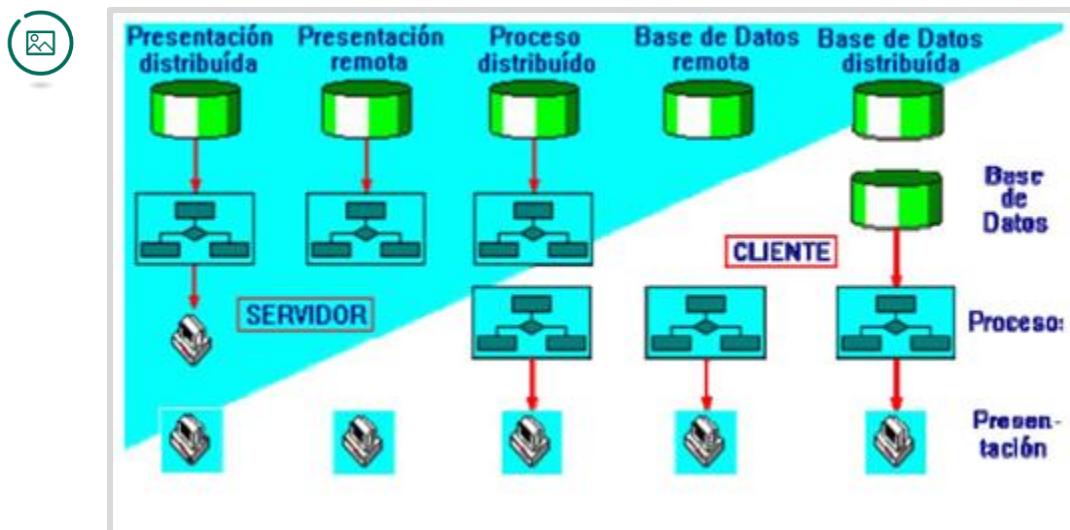


Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJtZ>.

## Características funcionales

Aquí la división se presentará cinco niveles, según las funciones que asumen el cliente y el servidor.

Figura 9: Características funcionales



Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.

- 1) **Presentación distribuida:** en este nivel el cliente es quien asume parte de las funciones de la presentación, no obstante el servidor sigue siendo quien realiza la mayor parte de las tareas, administrando y almacenando también todos los datos. Si bien el terminal físico no es un cliente flaco en su totalidad (como se observa en el gráfico que se adjunta), la base de datos, la lógica de la aplicación y la mayor parte de la interfaz de usuario se concentran en el servidor.

Figura 10: Presentación distribuida



Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.

- 2) Presentación remota: en este nivel ya se está trabajando con un cliente inteligente que tiene ya interfaz (entiéndase la interfaz base), por ejemplo: la captura de los datos, pequeñas validaciones y muestra de resultados de consultas.
- El servidor sigue teniendo aquí vital importancia ya que sigue almacenando los datos y conteniendo una interfaz ya más avanzada del usuario.

**Figura 11: Presentación remota**



Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.

- 3) Proceso distribuido: en este nivel, la mayor diferencia es donde se procesa la lógica de la aplicación.
- En los dos niveles anteriores la lógica se procesaba solamente en el servidor, mientras que en este nivel la lógica se procesa de igual manera en el cliente que en el servidor.

Como la lógica se divide en ambas partes, se lo conoce con el nombre de proceso distribuido o cooperativo.

**Figura 12: Base de datos remota**



Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.

- 4) Base de datos remota: en este nivel TODA la lógica de la aplicación se encuentra del lado del cliente, delegando solamente al servidor la parte del almacenamiento de la base de datos.

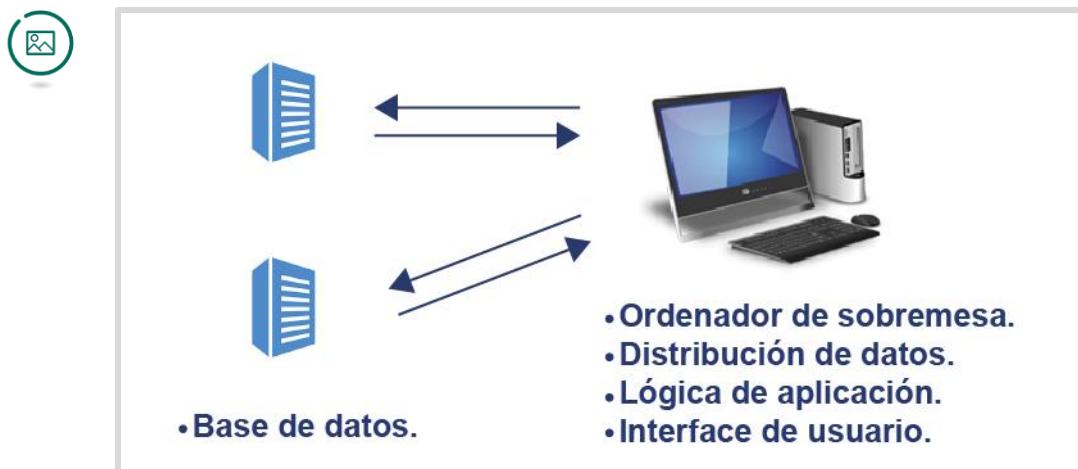
**Figura 13: Base de datos remota**



Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.

- 5) Base de datos distribuida: el quinto y último nivel es muy similar al nivel anterior, con la diferencia de que la gestión de la base de datos puede realizarse de manera compartida entre el cliente o varios servidores.

Figura 14: Base de datos distribuida



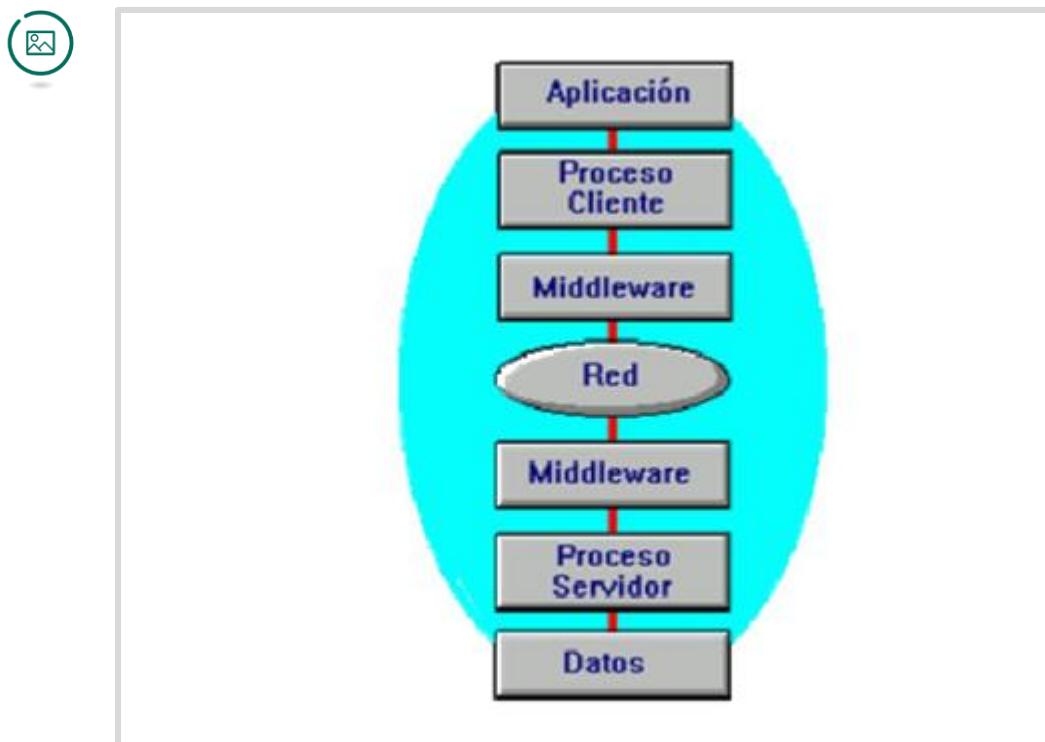
Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJltZ>.

### Características físicas

Se muestra a continuación un diagrama que refleja las características físicas de conexión entre las partes que componen una arquitectura cliente/servidor.

Como siempre, en lo que basa sus fundamentos esta tecnología es en aprovechar la potencia de las terminales para toda la parte de la lógica de presentación, para que de esta forma se pueda liberar al servidor para que realice las tareas que competen solamente a servidores.

Figura 15: Características físicas.



Fuente: Núñez Madrid, 2009, <https://goo.gl/vDJtZ>.

El proceso comienza -como siempre- en el cliente, el cual inicia la petición y es quien aloja a la aplicación.

Para la comunicación de todos estos elementos se utiliza un sistema de red que se encarga de transmitir la información entre clientes y servidores.

Se emplea aquí lo que anteriormente se describió como middleware. Este independiza los procesos de cliente y servidor.

Una vez establecida la conexión con el servidor, se toma de él lo que sea necesario, y por medio del middleware y del sistema operativo de red la información vuelve al usuario.

## Características lógicas

El principal aporte lógico que estas arquitecturas brindan es dotar al cliente de una interfaz gráfica capaz de tomar datos, editarlos, realizar pequeñas validaciones locales y luego mostrar los resultados.

Si bien los datos serán mostrados de manera transparente en el cliente, no se producirán duplicidades, ya que estarán almacenados en un mismo lugar: el servidor.

Todo lo expresado se realiza con la finalidad de que el usuario de un sistema de información, soportado por una arquitectura cliente/servidor, trabaje desde su estación de trabajo con distintos datos y aplicaciones, sin importarle dónde están o dónde se ejecuta cada uno de ellos.



## Referencias

Luján Mora, Sergio (2002). "Programación de Aplicaciones WEB. Historia. Principios Básicos y Clientes WEB." Editorial Club Universitario.

Núñez, R. (2009). <http://es.scribd.com/doc/44110204/Cliente-Servidor-1>

# Características clave

---



Programación  
Cliente-Servidor

UNIVERSIDAD  
**SIGLO 21** | MIEMBRO DE LA RED **ILUMNO**

# » Características clave de los sistemas distribuidos

## Compartición de recursos

Para entender lo que significa la compartición de recursos, en primer lugar deberemos dejar claro qué es lo que en informática se conoce con el nombre de recurso.

Un recurso puede definirse como toda aquella aplicación, herramienta, dispositivo o capacidad con que cuenta una computadora; es decir, los recursos se conocen como los componentes, ya sea de hardware o de software, que se necesitan para lograr un buen funcionamiento, tanto a nivel individual como a nivel colectivo.

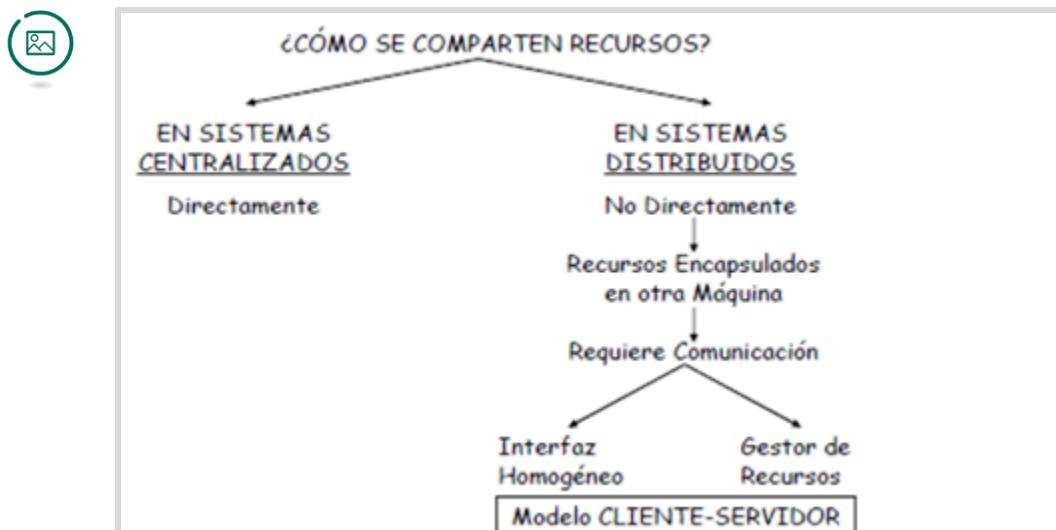
Por ejemplo, se identifican como recursos informáticos en una PC: la memoria, la capacidad de almacenamiento, e incluso la CPU (Unidad Central de Procesamiento) con la que cuenta una computadora.

Un recurso también puede ser un dispositivo. Por ejemplo, una computadora que cuenta con una impresora, cuenta con ese recurso.

Compartir recursos no es algo nuevo, ni aparece en el marco de los sistemas distribuidos; desde siempre los sistemas multiusuario han permitido la compartición de los recursos; no obstante, el aporte de los sistemas distribuidos consiste en que los recursos se encuentran físicamente en diferentes computadoras y solo pueden ser accedidos mediante las comunicaciones a través de la red.

Un claro ejemplo de lo detallado anteriormente puede ser compartir impresoras.

Figura 1



Fuente: [Imagen sin título]. (s. f.). Recuperado de: <https://goo.gl/y1Fk1l>, Pág. 12

## Apertura

Un sistema es abierto cuando puede ser extendido. La extensión puede ser con respecto al hardware o con respecto al software.

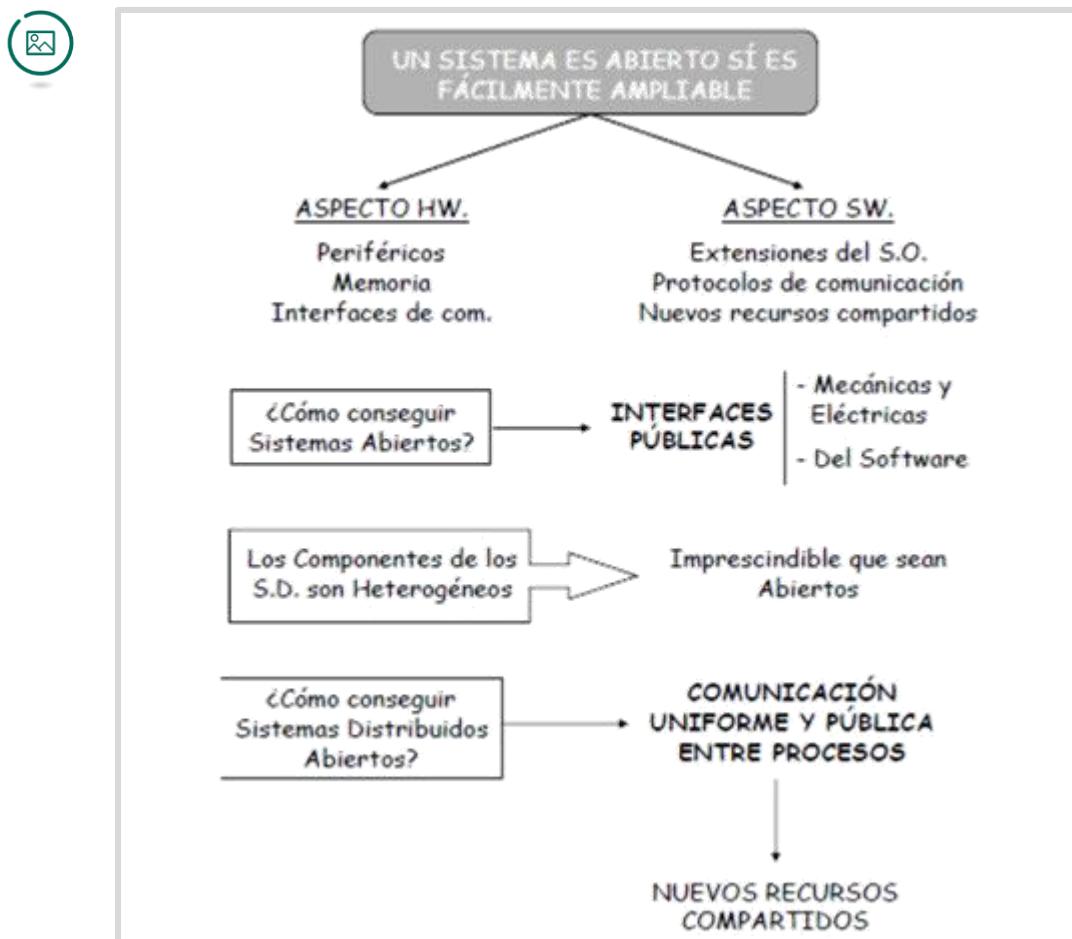
**Abierto con respecto al hardware:** se considera que es abierto cuando es posible el agregado de periféricos, memorias, interfaces de comunicación, sin importar el fabricante.

**Abierto con respecto al software:** se considera que es abierto cuando es posible el agregado de sistemas, también en este caso, sin importar el fabricante.

Esto se logra haciendo públicas las interfaces de comunicación tanto a nivel hardware como a nivel software.

Es de vital importancia lograr la apertura en un sistema distribuido, ya que uno de sus principales objetivos es que puedan construirse a partir de hardware y software heterogéneos, muy posiblemente provenientes de vendedores diferentes.

Figura 2: Apertura



Fuente: [Imagen sin título sobre Apertura.]. (s. f.). Recuperado de: <https://goo.gl/6GCG1d>, Página 14

## Transparencia

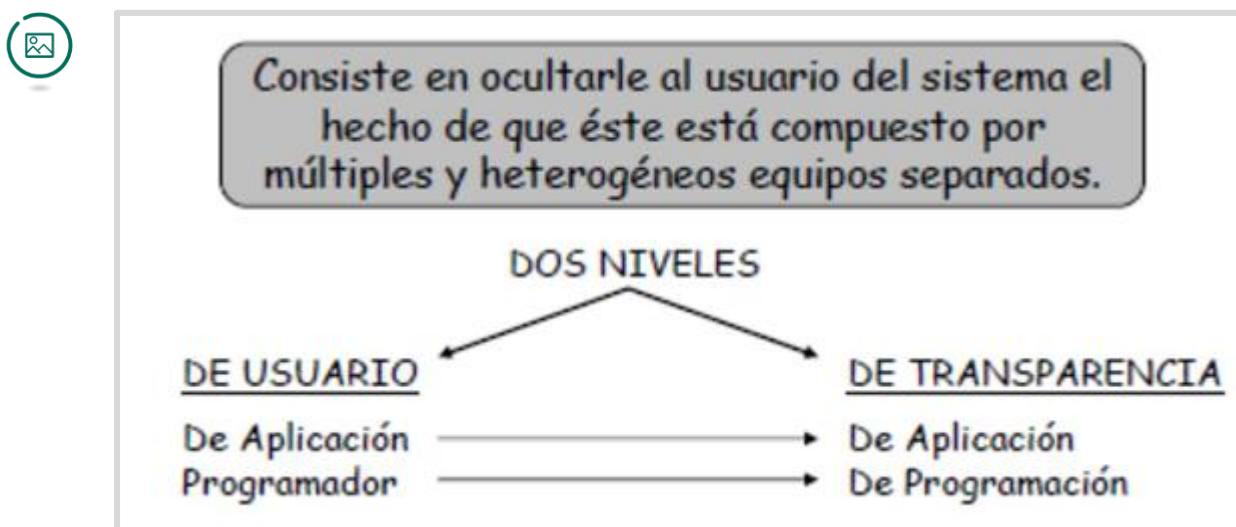
Transparencia consiste en ocultar al usuario y al programador cómo se componen físicamente los sistemas; es decir, se trata de mostrar el sistema como un todo en lugar de una colección de componentes.

Las transparencias más conocidas son:

- Transparencia de localización: se puede acceder a los objetos sin conocer su ubicación.
- Transparencia de concurrencia: se pueden ejecutar varios procesos a la vez.
- Transparencia de fallos: se pueden finalizar tareas (nos referimos a tareas de usuarios), aunque se produzca un fallo en el hardware o software.

- Transparencia de migración: se permite que se muevan objetos dentro de un sistema sin afectar a los usuarios.
- Transparencia de escalado: sin lugar a dudas, esta es una de las transparencias más importantes, ya que permite la expansión de los sistemas y de las aplicaciones sin necesidad de grandes cambios de estructura del sistema.

Figura 3: Transparencia



Fuente: [Imagen sin título sobre Transparencia]. (s. f.). Recuperado de: <https://goo.gl/6GCG1d>, Pág. 19

## Escalabilidad

Puede que una empresa en sus comienzos tenga solamente dos computadoras, conectadas mediante una red junto con un servidor y una impresora, formando esto un pequeño sistema distribuido.

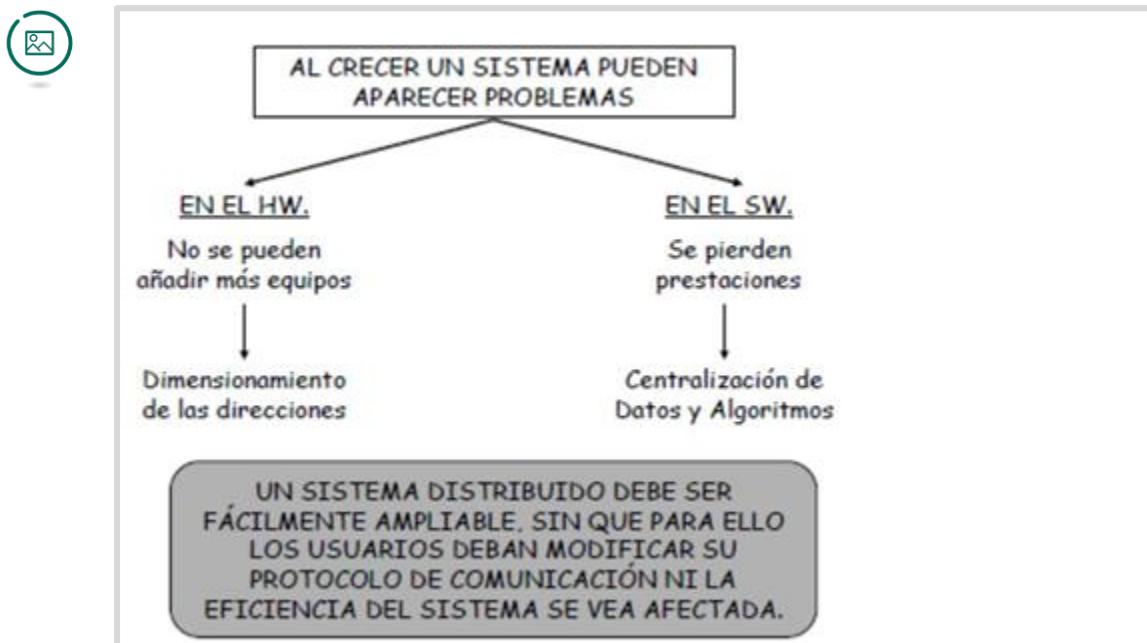
Si esta empresa crece y se necesita el agregado de computadoras y servidores, ni el software de un sistema ni el de la aplicación deberían cambiar al incrementarse la escala.

La necesidad de escalabilidad no necesariamente está ligada al hardware, sino que depende también del diseño del sistema, el cual deberá respetar los niveles de escalabilidad para que no se presenten luego limitaciones que inhabiliten la operación del sistema.

La demanda de escalabilidad en los sistemas distribuidos ha conducido a una filosofía de diseño en que cualquier recurso simple -hardware o software- puede extenderse para proporcionar servicio a tantos usuarios como se quiera.

Cuando el tamaño y complejidad de las redes de ordenadores crece, es un objetivo primordial diseñar software de sistema distribuido que seguirá siendo eficiente y útil con esas nuevas configuraciones de la red.  
 (Núñez Madrid, 2009, <https://goo.gl/vDJltZ>)

**Figura 4: Escalabilidad**



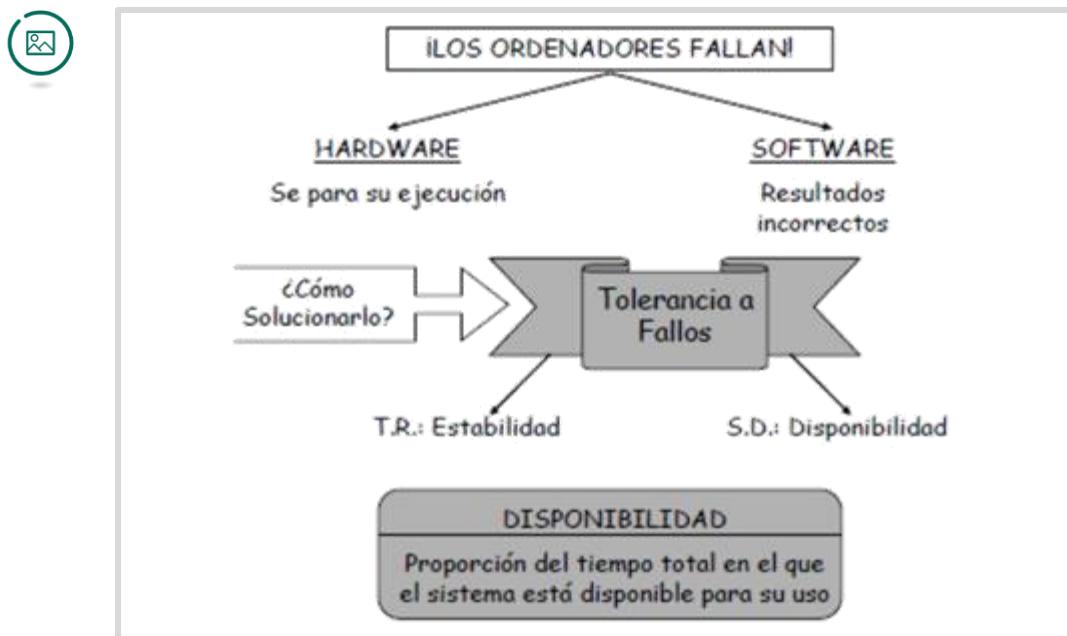
Fuente: [Imagen sin título sobre Escalabilidad.]. (s. f.). Recuperado de: <https://goo.gl/6GCG1d>, Pág., 15

## Tolerancia a fallos

Es de corriente conocimiento que muchas veces un sistema falla. Las fallas pueden afectar tanto al software como al hardware; esto llevará a que puedan producirse tanto resultados incorrectos como detenimientos del sistema antes de su finalización.

El diseño de un sistema distribuido tolerante a fallos deberá proveer elementos complementarios que permitan soportar tales inconvenientes, como la utilización de hardware redundante (por ejemplo: servidores de backups) o recuperación de software, es decir diseñar programas capaces de recuperarse de los fallos (por ejemplo: la utilización de transacciones).

Figura 5: Tolerancia a fallos



Fuente: [Imagen sin título sobre Tolerancia a fallos]. (s. f.). Recuperado de: <https://goo.gl/y1Fk1I>, Pág. 16



## Referencias

[Imagen sin título sobre]. (s. f.). Recuperado de:  
[http://www.dia.eui.upm.es/asignatu/sis\\_dis/Paco/Introduccion.pdf](http://www.dia.eui.upm.es/asignatu/sis_dis/Paco/Introduccion.pdf)

Núñez, R. (2009). <http://es.scribd.com/doc/44110204/Cliente-Servidor-1>