Sistemas Operativos II

Módulo IV

SISTEMAS de MÚLTIPLES PROCESADORES

1

Temas del Módulo IV

- **Introducción.**
- **Multiprocesadores.**
  - Hardware de multiprocesador.
  - Tipos de sistemas operativos multiprocesador.
- **Multicomputadoras.**
  - Hardware de una multicomputadora.
  - Software de comunicación de bajo nivel y a nivel de usuario.
  - Llamada a procedimiento remoto.
  - Memoria compartida distribuida.
- **Sistemas distribuidos.**
  - Hardware de red.
  - Servicios y protocolos de red.
  - Middleware.

2

Introducción

- Las primeras computadoras, como la ENIAC de 1946, podían ejecutar 300 operaciones por segundo, unas 1000 veces más que las calculadoras mecánicas, que sólo hacían operaciones básicas como sumas y restas.
- Inicialmente las computadoras fueron creadas para fines científicos, lo que llevó a la necesidad de hacer cálculos más complejos, y para eso, hace falta más velocidad.
- Al principio fueron monoprocesador, y durante años su evolución estaba marcada por el aumento de la velocidad del reloj. Pero eso tiene sus limitaciones en cuanto al tamaño que puede tener el procesador y la disipación de calor.
- Una manera de aumentar el poder de cálculo es utilizar más procesadores, esto es, varias CPU operando a velocidad “normal” pero que en conjunto tienen mucho más poder de cálculo que una sola.

3

Introducción

- Así fueron evolucionando, implementándose de distintas maneras un aumento de la cantidad de procesadores, hasta los procesadores modernos, que en un solo encapsulado tienen varios núcleos e hilos de procesamiento.
- En este módulo veremos los distintos modelos de sistemas de múltiples procesadores, mencionados en el Módulo I:
  - Multiprocesadores.
  - Multicomputadoras.
  - Sistemas distribuidos.

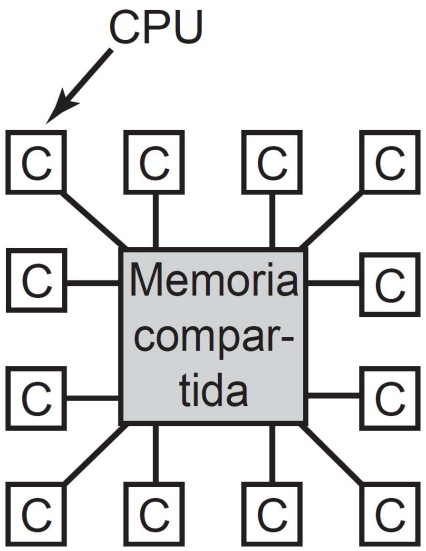
4

Multiprocesadores

- Un **multiprocesador con memoria compartida** (multiprocesador, de aquí en adelante, abreviado **MP**) es un sistema de cómputo en el que dos o más CPUs comparten todo el acceso a una RAM común.
- Un programa que se ejecuta en cualquiera de las CPUs ve un espacio normal de direcciones virtuales (por lo general paginadas).
- La única propiedad inusual que tiene este sistema es que la CPU puede escribir cierto valor en una palabra de memoria y después puede volver a leer esa palabra y obtener un valor distinto (tal vez porque otra CPU lo cambió).
- Si se organiza en forma correcta, esta propiedad forma la base de la comunicación entre procesadores: una CPU escribe ciertos datos en la memoria y otra lee esos datos.

5

Multiprocesadores



6

Multiprocesadores

- En su mayor parte, los sistemas operativos multiprocesadores son sólo sistemas operativos regulares: manejan las system calls, administran la memoria, proveen un sistema de archivos y administran los dispositivos de E/S.
- Sin embargo, hay áreas en las que tienen características únicas: la sincronización de procesos, la administración de recursos y la programación de tareas.
- A continuación mencionamos el hardware de los multiprocesadores (se ven en detalle en Arquitecturas II) y después pasaremos a ver las cuestiones relacionadas con estos sistemas operativos.

7

Multiprocesadores

Hardware de multiprocesador.

- Todos los multiprocesadores tienen la propiedad de que cada CPU puede direccionar toda la memoria. Sin embargo, la velocidad a la que la acceden los clasifica en dos grandes grupos:
  - Multiprocesadores **UMA** (Uniform Memory Access, acceso uniforme a memoria): cada palabra de memoria se puede leer con la misma velocidad que cualquier otra palabra de memoria.
  - Multiprocesadores **NUMA** (Non-Uniform Memory Access, acceso no uniforme a la memoria): la velocidad de acceso a memoria depende de la distancia relativa a la que se encuentra cada CPU.

8

Multiprocesadores

Hardware de multiprocesador. *(cont.)*

- Dentro de los multiprocesadores UMA tenemos tres arquitecturas:
  - Multiprocesadores UMA con arquitectura basada en bus.
  - Multiprocesadores UMA con interruptor de barras cruzadas (crossbar switch).
  - Multiprocesadores UMA que utilizan redes de conmutación multietapa.
- Todas estas arquitecturas se ven en detalle en Arquitecturas II, por lo que ahondaremos más acerca de los SO's de multiprocesadores.

9

Multiprocesadores

Tipos de sistemas operativos multiprocesador.

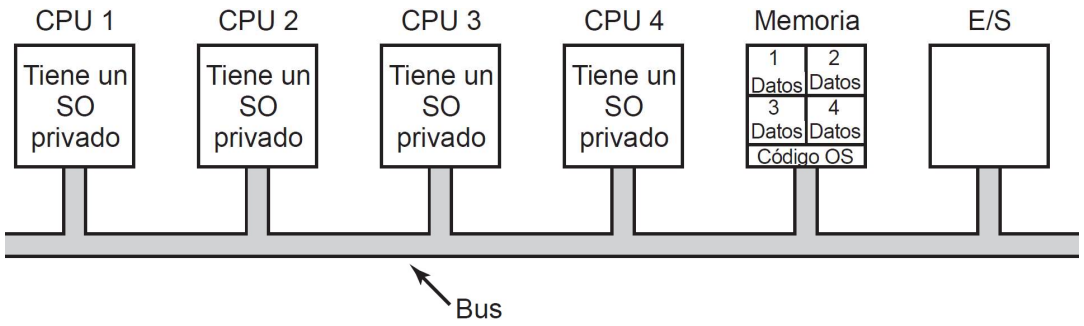
- Existen varios métodos de diseño de SO para MP, de los cuales analizaremos sólo tres. Todos ellos puede aplicarse a cualquiera de las plataformas de hardware que mencionamos.
- **Cada CPU tiene su propio sistema operativo.**
- La manera más simple de organizar un SO para MP es dividiendo estáticamente la memoria y su propia copia privada del sistema operativo.
- De esta manera, las *n* CPU operan entonces como *n* computadoras independientes.
- Una optimización obvia es permitir que todas las CPU compartan el código del SO y obtengan copias privadas sólo de las estructuras de datos del SO.

10

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

- **Cada CPU tiene su propio sistema operativo.** *(cont.)*



11

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

- **Cada CPU tiene su propio sistema operativo.** *(cont.)*
- Hay cuatro aspectos de diseño que caracterizan esta metodología:
  - Cuando un proceso invoca una system call, ésta se atrapa y maneja en su propia CPU, usando las estructuras de datos de ese SO.
  - Como cada SO tiene sus propias tablas, también tiene su propio conjunto de procesos. No hay compartición de procesos. Si un usuario se conecta a la CPU 1, todos sus procesos se ejecutan en ella. Puede ocurrir que la CPU 1 esté inactiva mientras la CPU 2 está cargada de trabajo.
  - Tampoco hay compartición de páginas. La CPU 1 puede tener páginas de sobra, y la CPU 2 estar paginando continuamente. La CPU 2 no puede pedir páginas a la CPU 1, ya que la asignación de memoria es fija.
  - Si cada SO mantiene un buffer de bloques del disco, puede pasar que haya un bloque "sucio" en varios buffers, lo que pude producir inconsistencia de datos. Y si no se utilizan esos buffers, el rendimiento se deteriora considerablemente.

12

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

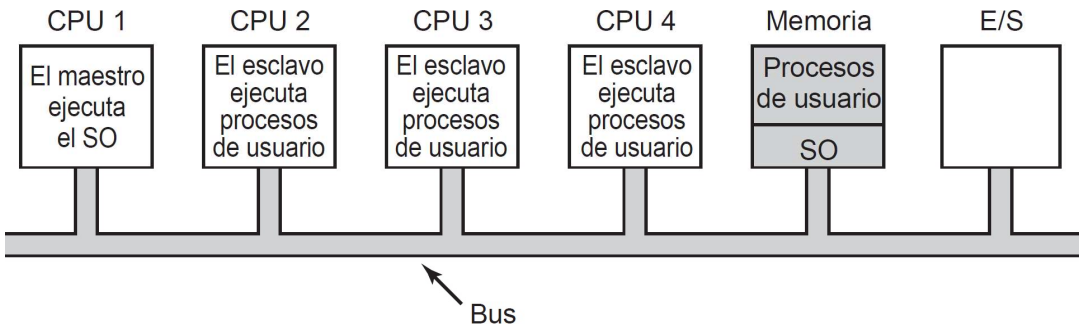
- **Multiprocesadores maestro-esclavo.**
- En este modelo existe una copia del SO y sus tablas en una única CPU. Todas las llamadas al sistema se redirigen a esa CPU para procesarlas ahí. También puede ejecutar proceso de usuario si tiene tiempo de sobra.
- A este modelo se le conoce como maestro-esclavo, ya que la CPU con el SO es el maestro y todas los demás son los esclavos. Con este modelo se resuelven la mayoría de los problemas del modelo anterior.
- Hay una sola estructura de datos (una lista o un conjunto de listas con prioridades) que lleva la cuenta de los procesos listos.

13

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

- **Multiprocesadores maestro-esclavo.** *(cont.)*



14

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

- **Multiprocesadores maestro-esclavo.** *(cont.)*
- Cuando una CPU está inactiva, pide al SO en la CPU maestro un proceso para ejecutar. Por ende, nunca ocurre que una CPU esté inactiva mientras que otra esté sobrecargada.
- También es posible asignar las páginas entre todos los procesos en forma dinámica y sólo hay una caché de buffer, por lo que nunca ocurren inconsistencias.
- El problema con este modelo es que con muchas CPUs, el maestro se convertirá en un cuello de botella.
- Si, por ejemplo, el 10% del tiempo está manejando llamadas al sistema, entonces con 10 CPUs el maestro casi se saturará y con 20 CPUs estará completamente sobrecargado.

15

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

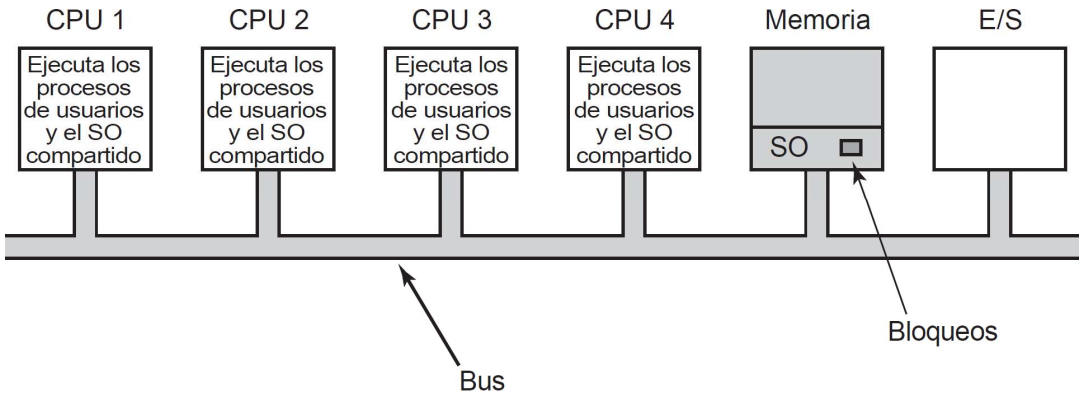
- **Multiprocesadores simétricos.**
- El **SMP** (Symmetric Multi-Processing, multiprocesador simétrico), elimina la asimetría del modelo anterior.
- Hay una copia del SO en memoria, pero cualquier CPU puede ejecutarlo.
- Cuando se hace una llamada al sistema, la CPU en la que se hizo la atrapa para el kernel y procesa esa llamada al sistema.
- Este modelo equilibra los procesos y la memoria en forma dinámica, ya que sólo hay un conjunto de tablas del SO.
- También elimina el cuello de botella de la CPU, ya que no hay un maestro.

16

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

- **Multiprocesadores simétricos.** *(cont.)*



17

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

- **Multiprocesadores simétricos.** *(cont.)*
- Este modelo presenta un inconveniente importante cuando dos o más CPUs ejecutan el SO al mismo tiempo.
- Por ejemplo, si dos CPUs elijen el mismo proceso para ejecutar, o si reclaman la misma página libre en memoria.
- La solución más simple para estos problemas es asociar un **mutex** con el SO, convirtiéndolo en una gran región crítica.
- Cuando una CPU desea ejecutar código del SO, debe adquirir primero el mutex. Si el mutex está bloqueado, sólo espera.
- De esta forma, cualquier CPU puede ejecutar el SO, pero sólo una a la vez.

18

Multiprocesadores

Tipos de sistemas operativos multiprocesador. *(cont.)*

- **Multiprocesadores simétricos.** *(cont.)*
- Esto funciona, pero casi tan mal como el modelo maestro-esclavo, pero es fácil de mejorar.
- Muchas partes del SO son independientes unas de otras, por lo que se pueden crear varias regiones críticas independientes que no interactúan entre sí, cada una con su propio mutex.
- Algunas tablas, como la tabla de procesos, pueden ser utilizadas por varias regiones críticas, por lo que también tendrán su propio mutex.
- Así, cada región crítica se puede ejecutar sólo por una CPU a la vez, y cada tabla crítica se puede utilizar sólo por una CPU a la vez.
- La mayoría de los multiprocesadores modernos utilizan esta organización.

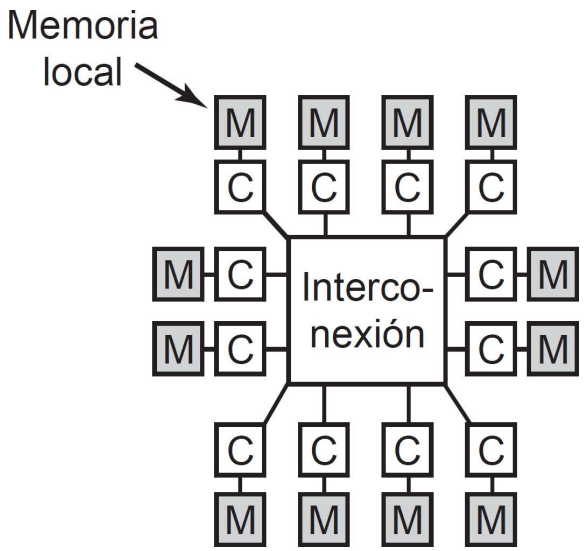
19

Multicomputadoras

- Los multiprocesadores son populares y atractivos debido a que ofrecen un modelo de comunicación simple: todas las CPUs comparten una memoria común.
- Los procesos pueden escribir mensajes en la memoria, para que otros procesos los lean. La sincronización se puede realizar con mutexes, semáforos, monitores y otras técnicas más.
- El único contratiempo es la dificultad de construir multiprocesadores grandes, además de que es un proceso costoso.
- Para resolver esto se ha investigado sobre las **multicomputadoras** (en adelante, **MC**): CPUs con acoplamiento fuerte que no comparten memoria, sino que cada una tiene su propia memoria.
- Estos sistemas son conocidos por una variedad de nombres, tales como **clúster de computadoras** y **COWS** (Clusters of Workstations, clústeres de estaciones de trabajo).

20

Multicomputadoras



21

Multicomputadoras

- Es fácil construir las multicomputadoras, ya que el componente básico es una PC que tiene sólo los componentes esenciales, además de una placa de red de alto rendimiento.
- El secreto para obtener un alto rendimiento es diseñar de manera inteligente la red de interconexión y la placa de red. Este problema es completamente análogo al de construir la memoria compartida en un multiprocesador.
- Sin embargo, el objetivo es enviar mensajes en una escala de tiempo en microsegundos, en vez de acceder a la memoria en una escala de tiempo de nanosegundos, por lo que es más simple, económico y fácil de lograr.
- Analizaremos brevemente el hardware de las multicomputadoras, luego el software de comunicación. Por último, veremos de qué manera se puede compartir la memoria entre nodos una MC.

22

Multicomputadoras

Hardware de una multicomputadora.

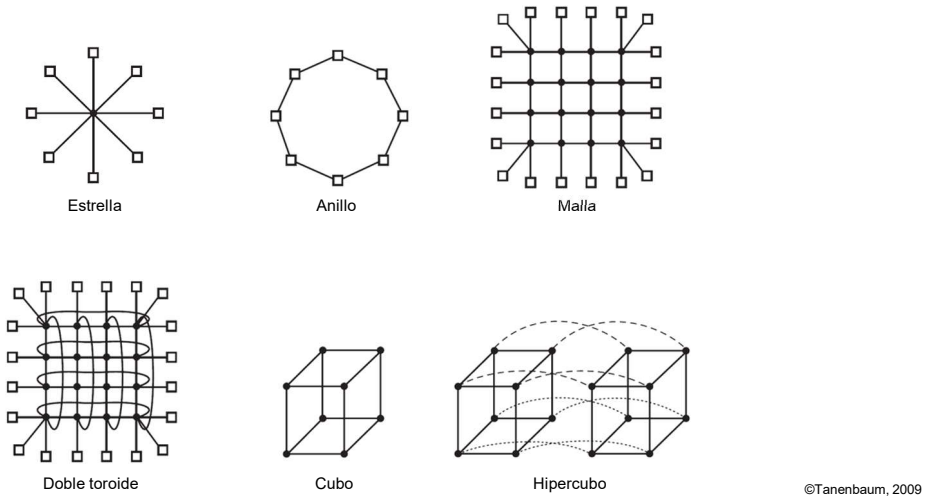
- **Tecnología de interconexión.**
- Cada nodo tiene una placa de red, de la cual salen uno o dos cables (o fibras). Estos cables se conectan a otros nodos o a switches. Las topologías de interconexión utilizadas son las siguientes:
  - En un sistema pequeño, puede haber un switch al que están conectados todos los nodos en la topología de **estrella**, muy común en redes Ethernet.
  - Otra alternativa, sin switches, es la tipo **anillo**. De cada placa de red salen dos cables, uno al nodo de la derecha y otro al de la izquierda.
  - La **malla** es un diseño bidimensional, muy utilizado en sistemas comerciales. Es muy regular y tiene la facilidad de poder escalar a tamaños mayores.
  - Una variante de la malla es el **doble toroide**, que es una malla con sus bordes interconectados. Es más tolerante a fallos y los caminos entre nodos son mucho más cortos.
  - El **cubo** es un arreglo tridimensional, que se puede ampliar a **hipercubos**, de hasta *n* dimensiones.

23

Multicomputadoras

Hardware de una multicomputadora. *(cont.)*

- **Tecnología de interconexión.** *(cont.)*



24

Multicomputadoras

Hardware de una multicomputadora. *(cont.)*

- **Interfaces de red.**
- La forma en que están construidas estas las placas de red de una MC tiene consecuencias considerables en el SO.
- En casi todas las MC, las placas de red tienen bastante RAM, para contener los paquetes entrantes y salientes.
- Esto es necesario para mantener la fluidez de la comunicación entre nodos; si los paquetes están en la RAM principal, esto no se puede garantizar.
- La velocidad de transmisión de estas placas suele ser constante y en extremo alta (superior a 20Gbps), por lo que deber ser capaces de almacenar los paquetes entrantes en tiempo real.
- Las placas suelen conectarse en los puertos PCIe para conectarse con la RAM principal, por lo que compiten con el disco por el bus.

25

Multicomputadoras

Hardware de una multicomputadora. *(cont.)*

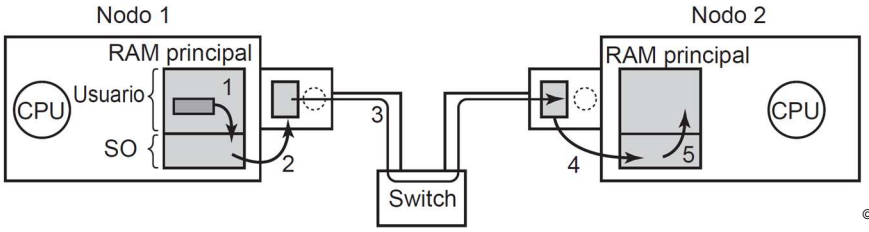
- **Interfaces de red.** *(cont.)*
- Para evitar esa competencia, las placas de red pueden tener uno o más canales DMA. De esta manera se logran conexiones de alta velocidad con la RAM principal, en bloques de datos más grandes.
- Además, las placas pueden tener una CPU completa, llamadas **procesadores de red**.
- Los procesadores de red permiten delegar parte del trabajo del CPU principal, realizando tareas de:
  - Manejo de transmisiones confiables.
  - Multitransmisión (envío de paquetes a más de un nodo).
  - Compresión/descompresión de datos.
  - Cifrado de datos.
- Esto requiere sincronizar las CPU, sobrecargando de trabajo al SO.

26

Multicomputadoras

Software de comunicación de bajo nivel.

- El mayor problema de la comunicación de alto rendimiento es el copiado excesivo de paquetes.
- En el mejor caso, habrá tres copias: de la RAM a la placa de red de origen, otra a la placa de red de destino, y otra a la RAM de destino.
- Pero si la placa de red está en el espacio de direcciones del kernel, habrá dos copia extra: del espacio de usuario al del kernel mediante una llamada al sistema, tanto en el origen como en el destino.



27

Multicomputadoras

Software de comunicación de bajo nivel. *(cont.)*

- Una solución posible es asignar la placa de red al directamente al espacio de usuario. Sin embargo esto ocasiona otros problemas:
  - Se producen competencias entre los procesos por el uso de la placa de red, lo que se soluciona utilizando mutexes.
  - El kernel también puede necesitar la placa de red; si se comparte el espacio con los usuarios, pueden ocasionare problemas de seguridad.
- Por lo tanto, el diseño más simple es utilizar dos placas de red, una asignada al espacio de usuario para el tráfico de las aplicaciones, y otra asignada al espacio del kernel para uso del SO.
- La mayoría de las MC utilizan este diseño.

28

Multicomputadoras

Software de comunicación a nivel de usuario.

- Habíamos mencionado en el Módulo I que la concurrencia en los entornos sin memoria compartida se realizaba con mediante el envío de mensajes entre los nodos.
- Para ello, se planteaban dos mecanismos:
  - Llamadas ***send*** y ***receive***.
  - Llamada a procedimiento remoto (**RPC**).
- Veremos a continuación ambos mecanismos.

29

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

- **Enviar y recibir (*send* y *receive*).**
- Es la forma más simple. El SO proporciona la forma de enviar y recibir mensajes, y los procedimientos de biblioteca hacen que estas llamadas subyacentes estén disponibles para los procesos de usuario.
- En forma genérica, la llamadas para enviar un mensaje tienen la forma:
  - send(dest, &mptr);
- Se envía el mensaje al que apunta ***mptr*** a un proceso identificado por ***dest***, y hace que el proceso que hizo la llamada se bloquee hasta que se haya enviado el mensaje.

30

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

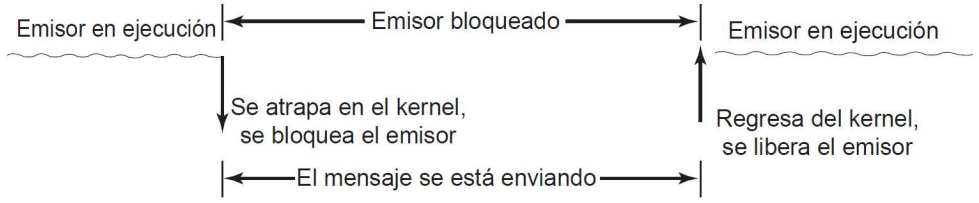
- **Enviar y recibir (*send* y *receive*).** *(cont.)*
- La llamada para recibir un mensaje puede tener la forma:
  - receive(direc, &mptr);
- El proceso que hizo la llamada se bloquea hasta que llegue un mensaje. Cuando llega, el mensaje se copia al búfer al que apunta ***mptr*** y el proceso se desbloquea.
- El parámetro ***direc*** especifica la dirección en la que el receptor está escuchando.
- Tal como se plantearon *send* y *receive*, se denominan **llamadas con bloqueo**.

31

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

- **Enviar y recibir (*send* y *receive*).** *(cont.)*
- Cuando un proceso llama a *send*, especifica un destino y un búfer para enviarlo a ese destino. Mientras se está enviando el mensaje, el proceso emisor se bloquea.
- La instrucción que sigue a la llamada a *send* no se ejecuta sino hasta que el mensaje se haya enviado por completo.



32

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

- **Enviar y recibir (*send* y *receive*).** *(cont.)*
- De manera similar, una llamada a *receive* no devuelve el control sino hasta que se haya recibido un mensaje y se haya colocado en el búfer al que apunta el parámetro.
- El proceso permanece suspendido en *receive* hasta que llega un mensaje, aunque tarde horas en llegar.
- En algunos sistemas, el receptor puede especificar de qué proceso desea recibir, en cuyo caso permanece bloqueado hasta que llega un mensaje de ese emisor.

33

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

- **Enviar y recibir (*send* y *receive*).** *(cont.)*
- Si bien existe la alternativa de llamada sin bloqueo, éstas son muy poco utilizadas en la práctica. De hecho, la mayoría de los sistemas implementa sólo una de estas formas: con bloqueo.
- La razón es simplemente porque tienen el mejor rendimiento, por un lado, y por otro lado, la implementación es mucho menos compleja.
- Además, el bloqueo facilita su utilización en sistemas multihilos, ya que mientras el hilo que hizo la llamada se bloquea, el resto de los hilos puede continuar con su ejecución.

34

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

- **Llamada a procedimiento remoto (RPC).**
- Un paso más allá de las llamadas a *send* y *receive* son las **llamadas a procedimientos remotos (Remote Procedure Call)**.
- Estas permiten que los procesos llamen a procedimientos que se encuentran en otras CPU, mientras que *send* y *receive* se limitan al envío de datos únicamente.
- Las RPC fueron planteadas por Birrel y Nelson (1984), y la idea es hacer que una llamada a un procedimiento remoto sea lo más parecida a una llamada local.
- Entonces, la RPC oculta el paso de mensajes y las operaciones de E/S.

35

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

- **Llamada a procedimiento remoto (RPC).** *(cont.)*
- Cuando un proceso en la máquina 1 llama a un procedimiento en la máquina 2, el proceso que hizo la llamada en la máquina 1 se suspende, y se lleva a cabo la ejecución del procedimiento al que llamó en la máquina 2.
- La información se puede transportar del proceso que hizo la llamada al procedimiento que llamó mediante los parámetros, y puede devolverse en el resultado del procedimiento.
- Por tradición, el procedimiento que hace la llamada se denomina cliente y el procedimiento al que se llamó se denomina servidor.

36

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

- **Llamada a procedimiento remoto (RPC).** *(cont.)*
- En su forma más simple, para llamar a un procedimiento remoto, el programa cliente se debe enlazar con un pequeño procedimiento de biblioteca conocido como resguardo (stub) del cliente, el cual representa al procedimiento del servidor en el espacio de direcciones del cliente.
- De manera similar, el servidor se enlaza con un procedimiento conocido como resguardo del servidor.
- Estos procedimientos ocultan el hecho de que la llamada al procedimiento desde el cliente al servidor no es local.

37

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

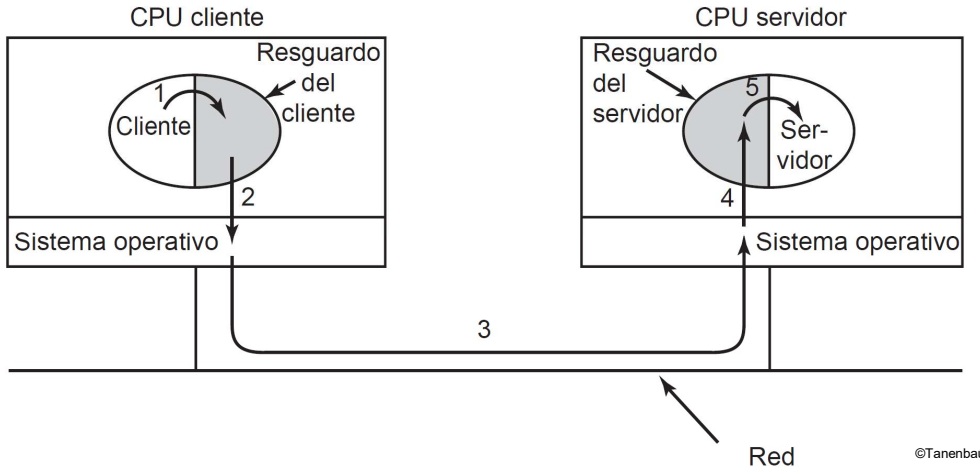
- **Llamada a procedimiento remoto (RPC).** *(cont.)*
- Los pasos para llevar a cabo un RPC son:
  1. El cliente llama al resguardo del cliente. Esta llamada es una llamada a un procedimiento local, donde los parámetros se meten en la pila de la manera usual.
  2. El resguardo del cliente empaqueta los parámetros en un mensaje y realiza una llamada al sistema para enviarlo.
  3. El kernel envía el mensaje de la máquina cliente a la máquina servidor.
  4. El kernel pasa el paquete entrante al resguardo del servidor.
  5. El resguardo del servidor llama al procedimiento del servidor.
- La respuesta sigue la misma ruta en dirección opuesta.

38

Multicomputadoras

Software de comunicación a nivel de usuario. *(cont.)*

- **Llamada a procedimiento remoto (RPC).** *(cont.)*



39

Multicomputadoras

Memoria compartida distribuida.

- Aunque RPC tiene sus atractivos, muchos programadores prefieren un modelo de memoria compartida y les gustaría utilizarlo también en una multicomputadora.
- Es posible preservar muy bien la ilusión de la memoria compartida (aún cuando en realidad no existe) mediante el uso de una técnica llamada **DSM** (Distributed Shared Memory, memoria compartida distribuida).
- A continuación, veremos su funcionamiento.

40

Multicomputadoras

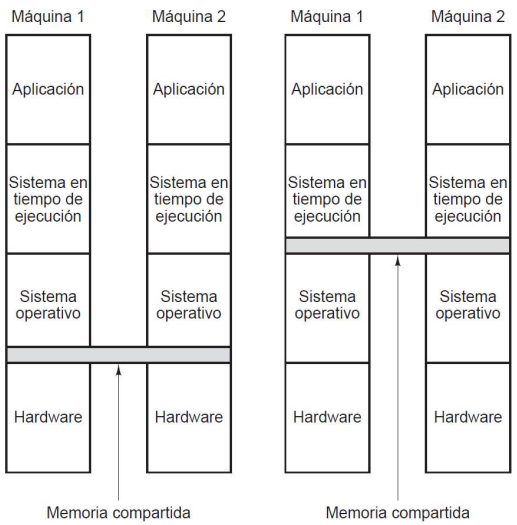
Memoria compartida distribuida. *(cont.)*

- Cada máquina tiene su propia memoria virtual y sus propias tablas de páginas.
- Cuando una CPU realiza una operación *LOAD* o *STORE* en una página que no tiene, se produce una interrupción que pasa al SO.
- Después, el SO localiza la página y pide a la CPU que la contiene que la desasigne y la envíe a través de la red de interconexión.
- Al llegar, la página se asigna y se reinicia la instrucción que fracasó.
- En efecto, el SO sólo está dando servicio a los fallos de página desde la RAM remota, en vez de hacerlo desde el disco local.
- Para el usuario, parece como si la máquina tuviera memoria compartida.

41

Multicomputadoras

Memoria compartida distribuida. *(cont.)*



42

Multicomputadoras

Memoria compartida distribuida. *(cont.)*

- Una manera de optimizar la DSM es duplicando las páginas de sólo lectura, típicamente el código de un programa. De esa manera se reducen considerablemente los fallos de página.
- Un problema de DSM es la **compartición falsa**. Ésta se produce cuando dos variables, A y B, son independientes entre sí, pero se encuentran en una misma página.
- Si el CPU1 usa intensivamente A, y el CPU2 hace lo mismo con B, esa página estará viajando constantemente entre ambas máquinas, bajando el rendimiento de las aplicaciones que las utilizan.
- Una solución es reduciendo el tamaño de la página, con lo que se reduce la probabilidad de que se de esa situación.
- También se resuelve mediante compiladores inteligentes, que eviten que variables independientes compartan la misma página.

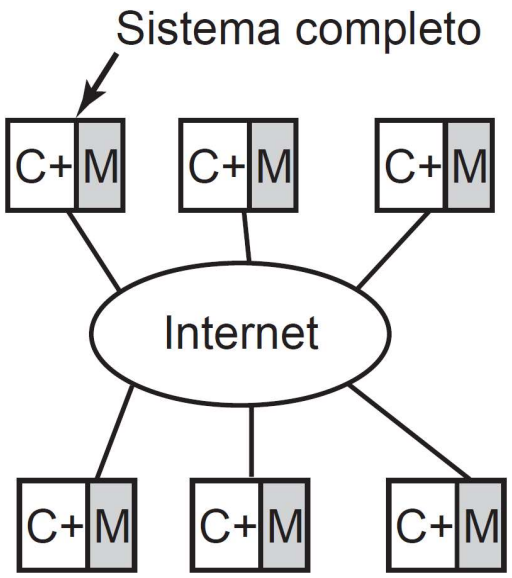
43

Sistemas distribuidos

- Como mencionamos en el Módulo I, el tercer tipo de sistema de sistema de múltiples procesadores donde se da la concurrencia son los **sistemas distribuidos** (en adelante, **SD**).
- Son similares a las MC en cuanto a que cada nodo tiene su propia memoria privada, sin memoria física compartida en el sistema. Sin embargo, los SD tienen un acoplamiento aún más débil que las MC.
- Las diferencias fundamentales de los nodos de un SD con los de una MC son:
  - Cada nodo en un SD es **una computadora completa**, incluyendo periféricos. En una MC, sólo tienen CPU, RAM, placa de red, y a veces un disco.
  - Los nodos pueden estar **esparcidos por todo el mundo**, mientras que en una MC está en una misma habitación.
  - En un SD los nodos pueden tener un **SO diferente**, con su propio sistema de archivos y bajo su propia administración. En una MC, los nodos comparten el SO, el sistema de archivos y la administración.

44

Sistemas distribuidos



45

Sistemas distribuidos

- Mencionamos que los nodos de un SD tienen un acoplamiento aún más débil que las MC.
- Eso significa que el retardo en la comunicación es mayor, en el orden de los milisegundos, esto es, tres órdenes de magnitud más grande que una MC.
- Por otro lado, la heterogeneidad de hardware y SO en los nodos también hacen al bajo acoplamiento de los nodos.
- Esto lleva a la necesidad de crear un paradigma común que ofrezca una manera uniforme de ver todo el sistema.
- Una de las formas de obtener cierta uniformidad es tener un nivel de software encima del SO.
- Este nivel es llamado **middleware**, y lo veremos con un poco más de detalle, más adelante en esta sección.

46

Sistemas distribuidos

- En la siguiente tabla, tenemos un resumen comparativo de los tres tipos de sistemas de múltiples procesadores:

Elemento	Multiprocesador	Multicomputadora	Sistema distribuido
Configuración de nodo	CPU	CPU, RAM, interfaz de red	Computadora completa
Periféricos de nodo	Todos compartidos	Compartidos, excepto tal vez el disco	Conjunto completo por nodo
Ubicación	Mismo bastidor	Mismo cuarto	Posiblemente a nivel mundial
Comunicación entre nodos	RAM compartida	Interconexión dedicada	Red tradicional
Sistemas operativos	Uno, compartido	Varios, igual	Posiblemente todos distintos
Sistemas de archivos	Uno, compartido	Uno, compartido	Cada nodo tiene el suyo
Administración	Una organización	Una organización	Muchas organizaciones

47

Sistemas distribuidos

Hardware de red.

- Los SD se construyen sobre redes de computadoras, básicamente en dos tipos:
  - **LAN** (Local Area Network, red de área local), abarcando desde un edificio hasta un campus.
  - **WAN** (Wide Area Network, red de área extendida), abarcando una ciudad, un país, incluso hasta a nivel mundial.
- Ejemplo de una red LAN es típicamente una red Ethernet, basada en el estándar IEEE 802.3 y los demás estándares afines.
- Si bien no es una red, sino una federación de miles de redes interconectadas, un ejemplo de red WAN es Internet.
- Los detalles de estos tipos de redes se estudian en profundidad en las asignaturas Comunicaciones I y II.

48

Sistemas distribuidos

Servicios y protocolos de red.

- Todas las redes de computadoras ofrecen distintos **servicios** de comunicación a los usuarios, los que se implementan siguiendo ciertas reglas (**protocolos**).
- **Servicios de red.**
- Se utilizan básicamente dos:
  - Servicio orientado a conexión: similar al sistema telefónico; el usuario primero establece una conexión, la utiliza y después la libera. También es llamado *conmutación de circuitos*.
  - Servicio orientado a no conexión: parecido al sistema postal. Cada mensaje se envía de manera independiente, por un camino distinto. Si a un mismo destino se envían dos mensajes, no necesariamente llegarán "en orden". Se conoce también como *conmutación de paquetes*.
- Estos servicios se estudian exhaustivamente en Comunicaciones II.

49

Sistemas distribuidos

Servicios y protocolos de red. *(cont.)*

- **Protocolos de red.**
- Existen muchos protocolos, incluyendo los de enrutador a enrutador, los de host a host y varios más. Las redes modernas utilizan una **pila** (stack) para distribuir los distintos protocolos en niveles, uno encima de otro.
- Los protocolos de base de los SD son los dos protocolos principales de Internet.
  - IP (Internet Protocol, protocolo de internet): está basado en datagramas que viajan de manera independiente (*conmutación de paquetes*).
  - TCP (Transmission Control Protocol, protocolo de control de transmisión): se encarga de proveer una comunicación confiable entre nodos.
- Estos protocolos se estudian en detalle en Comunicaciones I y II.

50

Sistemas distribuidos

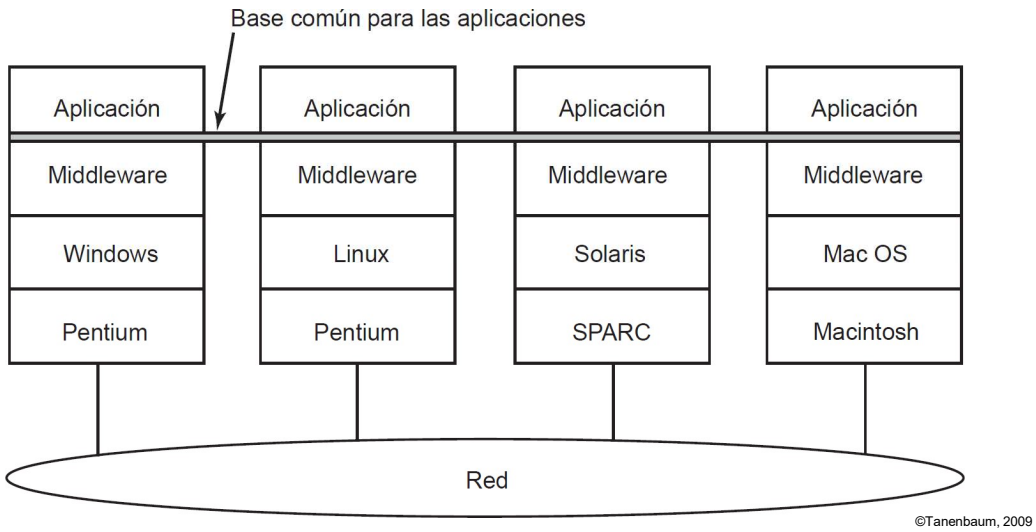
Middleware.

- Habíamos mencionado que el middleware es una capa de software por encima del SO, que permite obtener cierta uniformidad frente a los distintos SO y hardware subyacente en un SD.
- Esta capa provee ciertas estructuras de datos y operaciones que permiten a los procesos y usuarios que están en máquinas remotas interoperar de una manera consistente.
- En cierto sentido, el middleware es como si fuera el SO de un SD, aunque en realidad no lo es.
- Veremos a continuación, un par de capas o niveles de middleware, que permiten producir un paradigma constante para las aplicaciones y los usuarios.

51

Sistemas distribuidos

Middleware. *(cont.)*



52

Sistemas distribuidos

Middleware. *(cont.)*

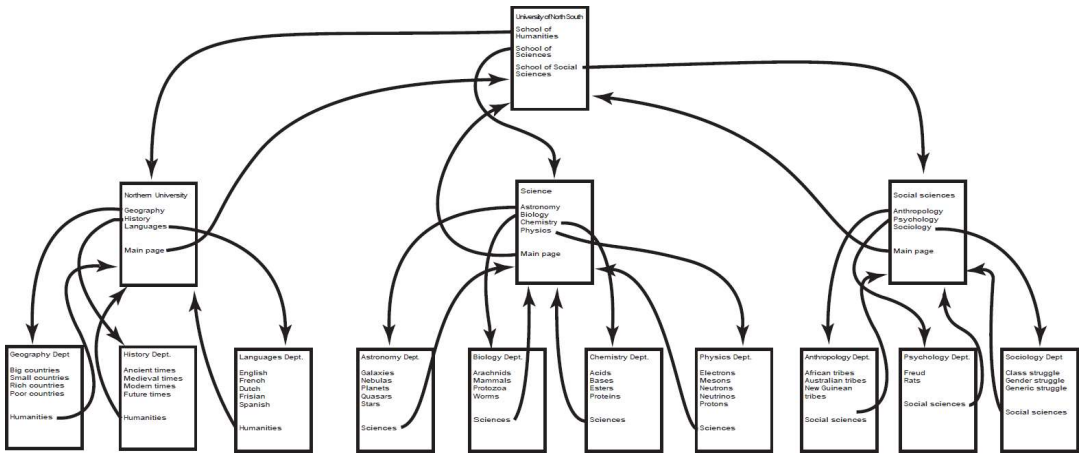
- **Middleware basado en documentos.**
- El ejemplo más simple y conocido es la WWW (World Wide Web).
- El paradigma original detrás de WWW es muy simple:
  - Cada computadora puede contener uno o más documentos, llamados páginas web. Cada página web contiene texto, imágenes, audios, videos, etc.
  - Además, las páginas tienen hipervínculos, que son punteros a otras páginas.
  - El usuario solicita una página web mediante un software llamado navegador web (web browser). Al hacer clic en un hipervínculo, la página actual se sustituye por la apuntada por el hipervínculo.
- La Web es, en esencia, un gran **grafo dirigido de documentos** que pueden apuntar a otros documentos.

53

Sistemas distribuidos

Middleware. *(cont.)*

- **Middleware basado en documentos.** *(cont.)*



54

Sistemas distribuidos

Middleware. *(cont.)*

- **Middleware basado en documentos.** *(cont.)*
- Cada página web tiene una dirección única, llamada **URL** (Uniform Resource Locator, localizador uniforme de recursos).
- Una URL tiene la forma *protocolo://nombre-DNS/nombre-archivo*. Los protocolos más comunes son el *http* y *https* (Hyper-Text Transfer Protocol, protocolo de transferencia de hipertexto), siendo el segundo una versión del primero con una capa de seguridad.
- La computadora que contiene las páginas web y provee la capa de middleware para su acceso se llama **servidor web**.
- El navegador web hace las veces de **cliente web**, ya que tiene la capacidad de acceder e interpretar la capa de middleware.

55

Sistemas distribuidos

Middleware. *(cont.)*

- **Middleware basado en sistemas de archivos.**
- La Web hace que un SD tenga la apariencia de una gran colección de documentos. Otro paradigma es que el SD tenga la apariencia de un **gran sistema de archivos**.
- Utilizar un modelo de sistema de archivos para un SD significa que hay un solo sistema de archivos global, en el que usuarios de todo el mundo tienen autorización de leer y escribir en archivos.
- Para lograr la comunicación, un proceso tiene que escribir datos en un archivo y otros procesos tienen que leer los datos de vuelta.
- Aquí surgen muchos de los problemas estándar con los sistemas de archivos, pero también aparecen nuevos problemas relacionados con la distribución.

56

Sistemas distribuidos

Middleware. *(cont.)*

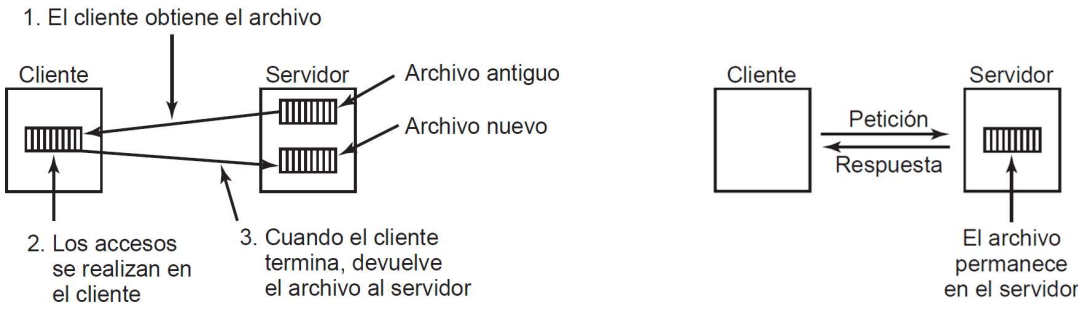
- **Middleware basado en sistemas de archivos.** *(cont.)*
- **Modelo de transferencia.**
- Existen dos modelos de transferencia:
  - Envío/descarga: el proceso descarga el archivo al que quiere acceder. Tanto la lectura como la escritura se hacen localmente. Cuando el proceso termina de usar el archivo, lo envía actualizado en el servidor.
  - Acceso remoto: el archivo permanece en el servidor y el cliente envía distintos comandos para realizar el trabajo ahí.
- El modelo envío/descarga tiene la ventaja de ser muy simple, y es muy eficiente en el manejo de los archivos.
- Sin embargo, tiene como desventaja que el cliente tiene que tener espacio suficiente para almacenar el archivo, además del riesgo de inconsistencia cuando se accede concurrentemente a un archivo.

57

Sistemas distribuidos

Middleware. *(cont.)*

- **Middleware basado en sistemas de archivos.** *(cont.)*
- **Modelo de transferencia.** *(cont.)*



58

Sistemas distribuidos

Middleware. *(cont.)*

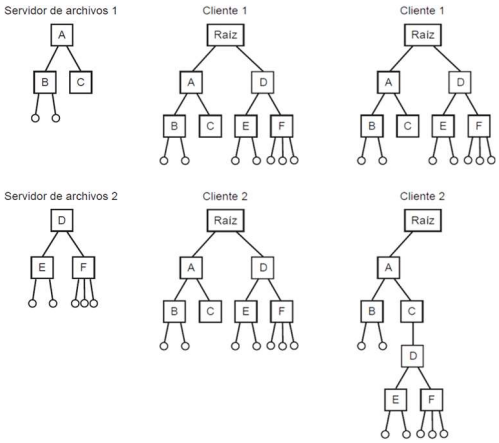
- **Middleware basado en sistemas de archivos.** *(cont.)*
- **Jerarquía de directorios.**
- Un problema de diseño es si todos los clientes tiene la misma vista de la jerarquía de directorios.
- Suponiendo dos servidores de archivos, una posibilidad es que todos los clientes tengan la misma vista del árbol de directorios.
- Esto facilita la programación y comprensión del sistema.
- Pero en sistemas que administran varios sistemas de archivos con montaje remoto, la vista de los clientes puede llegar a ser distinta.
- Este modo tiene cierta flexibilidad, pero dificulta su lectura con clientes que son heterogéneos.

59

Sistemas distribuidos

Middleware. *(cont.)*

- **Middleware basado en sistemas de archivos.** *(cont.)*
- **Jerarquía de directorios.** *(cont.)*



©Tanenbaum, 2009

60

Sistemas distribuidos

Middleware. *(cont.)*

- **Middleware basado en sistemas de archivos.** *(cont.)*
- **Transparencia de nomenclatura.**
- Es deseable que el sistema tenga dos características importantes:
  - Transparencia de localización: supongamos el nombre de ruta de un archivo, */servidor1/dir1/dir2/x*. Se indica donde está el archivo *x*, pero no dónde está ese servidor. Se puede mover el servidor a cualquier parte de la red sin tener que cambiar el nombre de la ruta.
  - Independencia de ubicación: es cuando en un sistema se pueden mover los archivos sin cambiar sus nombres. Por ejemplo, si el archivo *x* es tan grande que es necesario moverlo a otro servidor, pero sin que los clientes "noten" el cambio.
- Esto último es difícil de lograr, requiere un diseño cuidadoso, pero facilita la vida de los programadores y usuarios.

61

Sistemas distribuidos

Middleware. *(cont.)*

- **Middleware basado en sistemas de archivos.** *(cont.)*
- **Semántica de compartición de archivos.**
- Para que los sistemas de archivos sean accesibles de manera concurrente, es necesario establecer ciertas reglas que garanticen la consistencia de los datos:
  - Consistencia secuencial: funciona a la perfección en monoprocesadores. Cuando se hace una lectura seguida de una o más escrituras, el valor leído corresponde siempre al último escrito. Es ineficiente con varios procesadores.
  - Semántica de sesión: se basa en el modelo de transferencia envío/descarga. Cuando un cliente cierra un archivo, envía de inmediato una copia al servidor; de esa manera se garantiza la consistencia de datos en las lecturas siguientes.
- Como alternativa al último método, se puede bloquear un archivo que ha sido descargado, tal como funcionan los mutexes.

62

Fin del Módulo IV

63