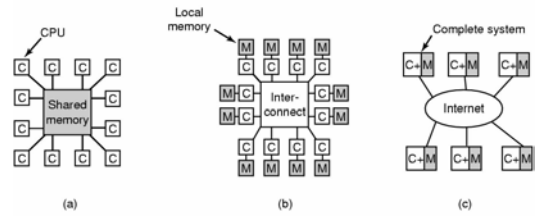
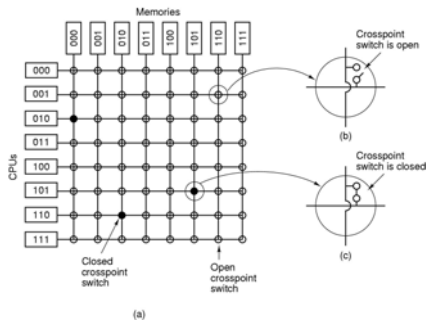


Multiple Processor and Distributed Systems

Multiprocessor Systems



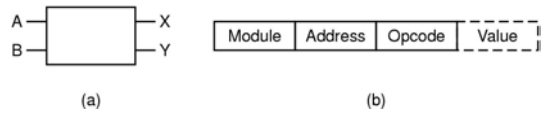
Multiprocessor Hardware (2)



◆ UMA Multiprocessor using a crossbar switch

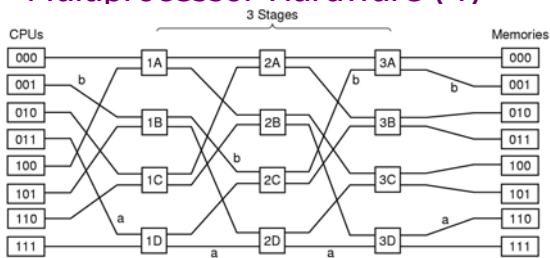
Multiprocessor Hardware (3)

◆ Multiprocessors using multistage switching networks can be built from 2x2 switches



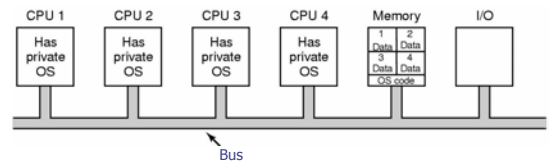
(a) 2x2 switch (b) Message format

Multiprocessor Hardware (4)



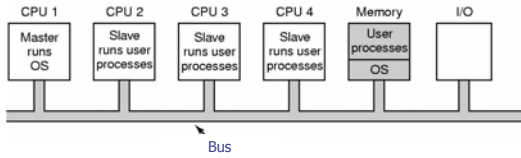
◆ Omega Switching Network

Multiprocessor OS Types (1)



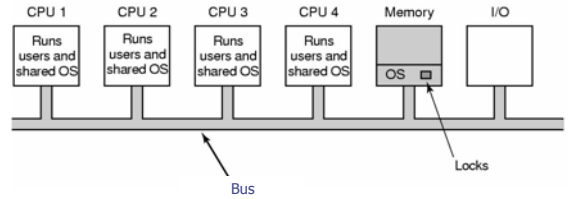
Each CPU has its own operating system

Multiprocessor OS Types (2)



Master-Slave multiprocessors

Multiprocessor OS Types (3)



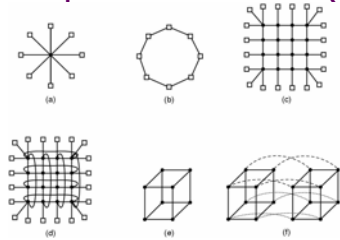
- ◆ Symmetric Multiprocessors
 - SMP multiprocessor model

Multicomputers

- ◆ Definition:
 - Tightly-coupled CPUs that do not share memory*

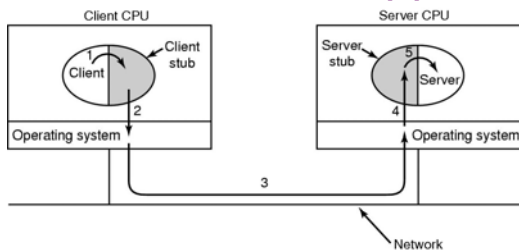
- ◆ Also known as
 - cluster computers
 - clusters of workstations (COWs)

Multicomputer Hardware (1)



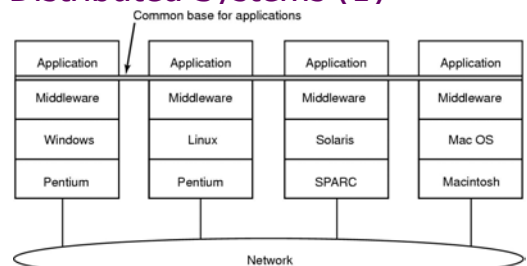
- ◆ Interconnection topologies
 - (a) single switch
 - (b) ring
 - (c) grid
 - (d) double torus
 - (e) cube
 - (f) hypercube

Remote Procedure Call (1)



- ◆ Steps in making a remote procedure call
 - the stubs are shaded gray

Distributed Systems (1)



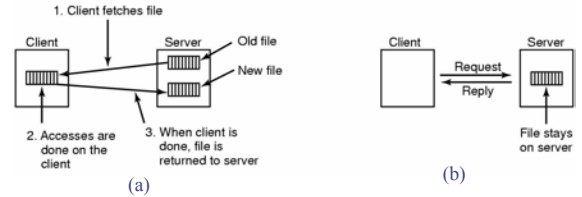
Achieving uniformity with middleware

Document-Based Middleware

How the browser gets a page

1. Asks DNS for IP address
2. DNS replies with IP address
3. Browser makes connection
4. Sends request for specified page
5. Server sends file
6. TCP connection released
7. Browser displays text
8. Browser fetches, displays images

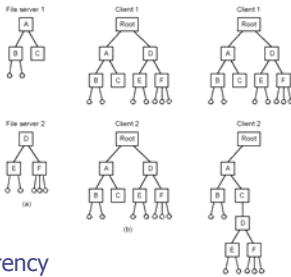
File System-Based Middleware (1)



Transfer Models

- (a) upload/download model
- (b) remote access model

File System-Based Middleware (2)



Naming Transparency

- (b) Clients have same view of file system
- (c) Alternatively, clients with different view

Network File System

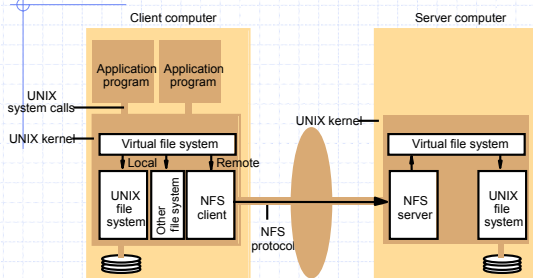
- The Network File System (NFS) was developed to allow machines to mount a disk partition on a remote machine as if it were on a local hard drive. This allows for fast, seamless sharing of files across a network.
- Three aspects of NFS are of interest: the architecture, the protocol, and the implementation.

11/28/2005

BR

16

NFS architecture



11/28/2005

BR

17

NFS Architecture (contd.)

- Allows an arbitrary collection of clients and servers to share a common file system.
- In many cases all servers and clients are on the same LAN but this is not required.
- NFS allows every machine to be a client and server at the same time.
- Each NFS server exports one or more directories for access by remote clients.

11/28/2005

BR

18

NFS Protocol

- ◆ One of the goals of NFS is to support a heterogeneous system, with clients and servers running different operating systems on different hardware. It is essential the interface between clients and server be well defined.
- ◆ NFS accomplishes this goal by defining two client-server protocols: one for handling mounting and another for directory and file access.
- ◆ Protocol defines requests by clients and responses by servers.

11/28/2005

BR

19

Mounting

- ◆ Client requests a directory structure to be mounted, if the path is legal the server returns file handle to the client.
- ◆ Or the mounting can be automatic by placing the directories to be mounted in the `/etc/rc`: automounting.

11/28/2005

BR

20

File Access

- ◆ NFS supports most unix operations except open and close. This is to satisfy the "statelessness" on the server end. Server need not keep a list of open connections. See the operations listed in earlier slides.
- ◆ (On the other hand consider your database connection... you create an object, connection is opened etc.)

11/28/2005

BR

21

Implementation

- ◆ After the usual system call layer, NFS specific layer Virtual File System (VFS) maintains an entry per file called vnode (virtual I-node) for every open file.
- ◆ Vnode indicate whether a file is local or remote.
 - For remote files extra info is provided.
 - For local file, file system and I-node are specified.
 - Lets see how to use v-nodes using a mount, open, read system calls from a client application.

11/28/2005

BR

22

Vnode use

- ◆ To mount a remote file system, the sys admin (or `/etc/rc`) calls the mount program specifying the remote directory, local directory in which to be mounted, and other info.
- ◆ If the remote directory exist and is available for mounting, mount system call is made.
- ◆ Kernel constructs vnode for the remote directory and asks the NFS-client code to create a r-node (remote I-node) in its internal tables. V-node in the client VFS will point to local I-node or this r-node.

11/28/2005

BR

23

Remote File Access

- ◆ When a remote file is opened by the client, it locates the r-node.
- ◆ It then asks NFS Client to open the file. NFS file looks up the path in the remote file system and return the file handle to VFS tables.
- ◆ The caller (application) is given a file descriptor for the remote file. No table entries are made on the server side.
- ◆ Subsequent reads will invoke the remote file, and for efficiency sake the transfers are usually in large chunks (8K).

11/28/2005

BR

24