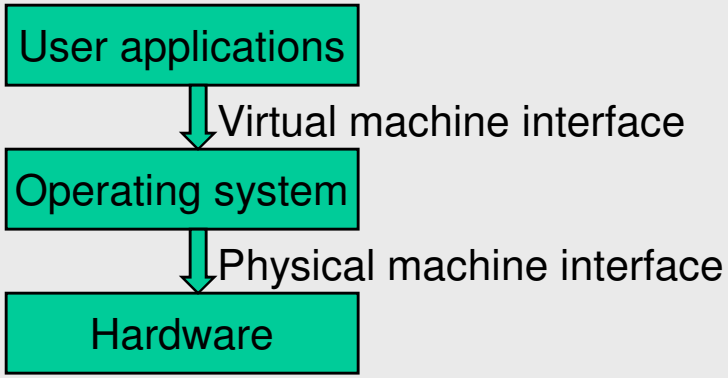
Software Concepts

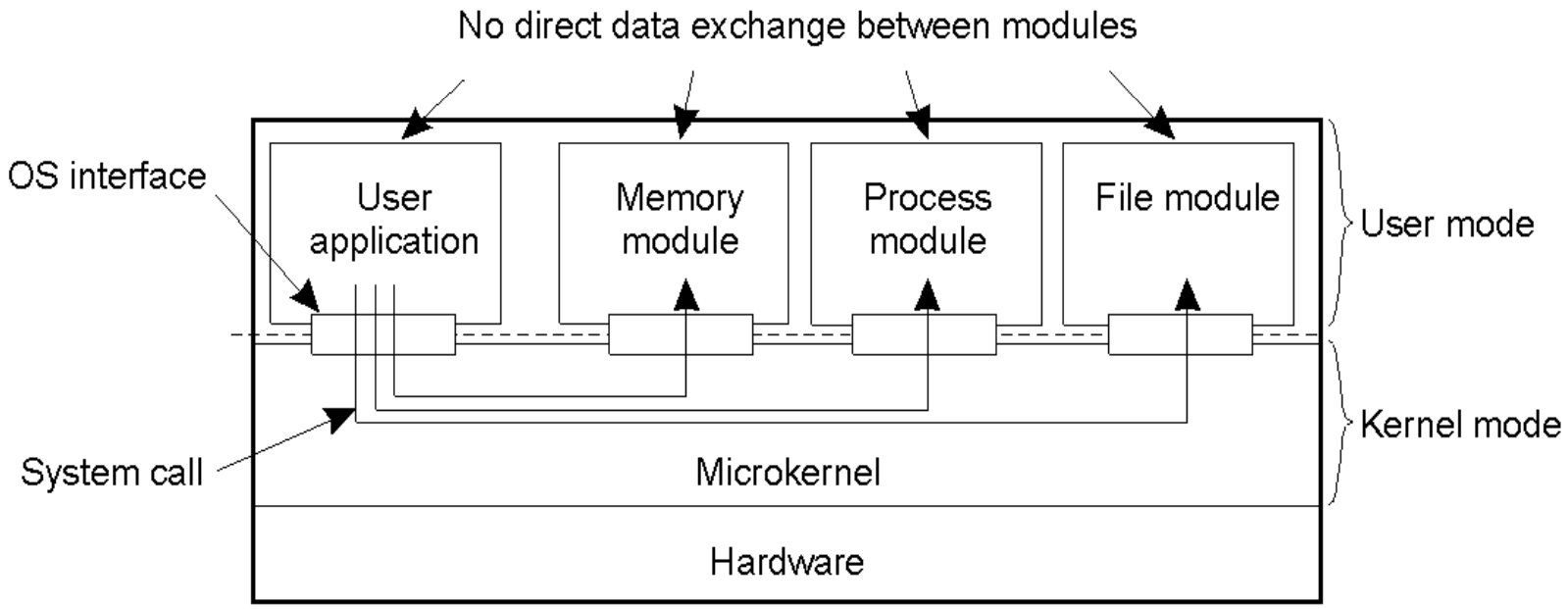


- Operating system:
  - Interface between users and hardware
  - Implements a virtual machine that is easier to program than raw hardware
- Primary functions:
  - Services: file system, virtual memory, networking, CPU scheduling, …
  - Coordination: concurrency, memory protection, security, networking,…

Uniprocessor Operating Systems

Microkernel architecture

- Small kernel
- user-level servers implement additional functionality

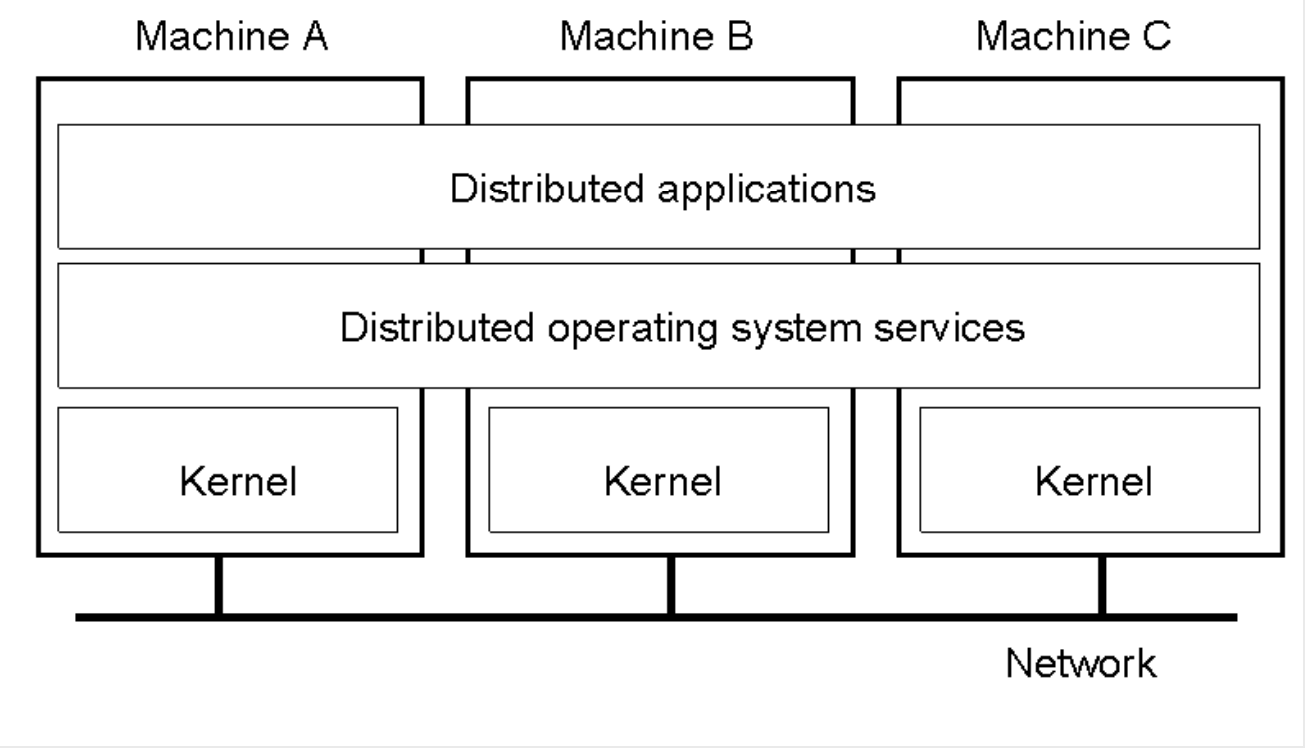


Multiprocessor Operating Systems

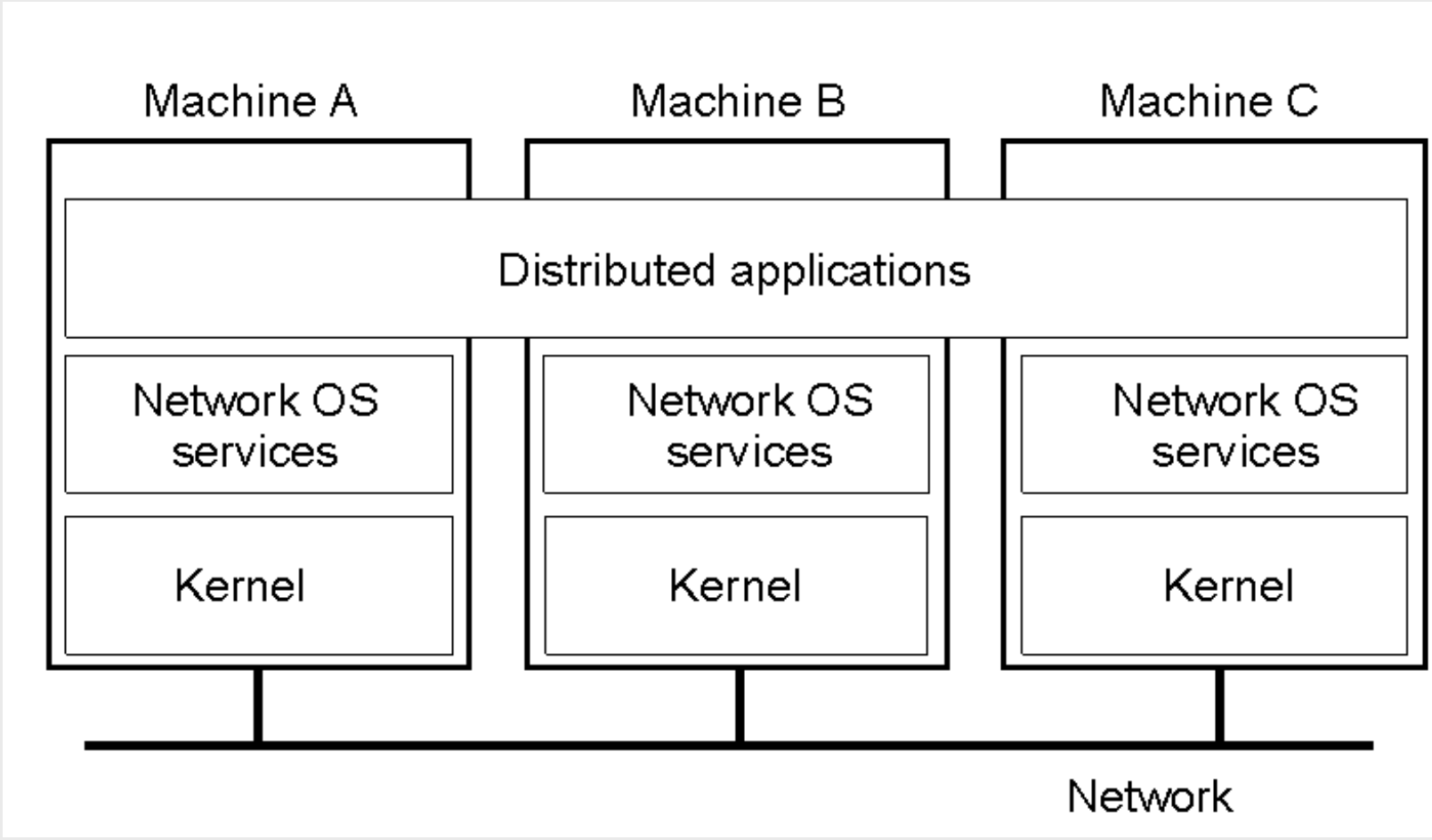
- Like a uniprocessor operating system
- Manage multiple CPUs transparently to the user
- Each processor has its own hardware cache
  - Maintain consistency of cached data
  - Scalability issues
- Shared variable versus message passing

Multicomputer Operating Systems

- More complex than multiprocessor OS
  - Because communication has to be through explicit message passing

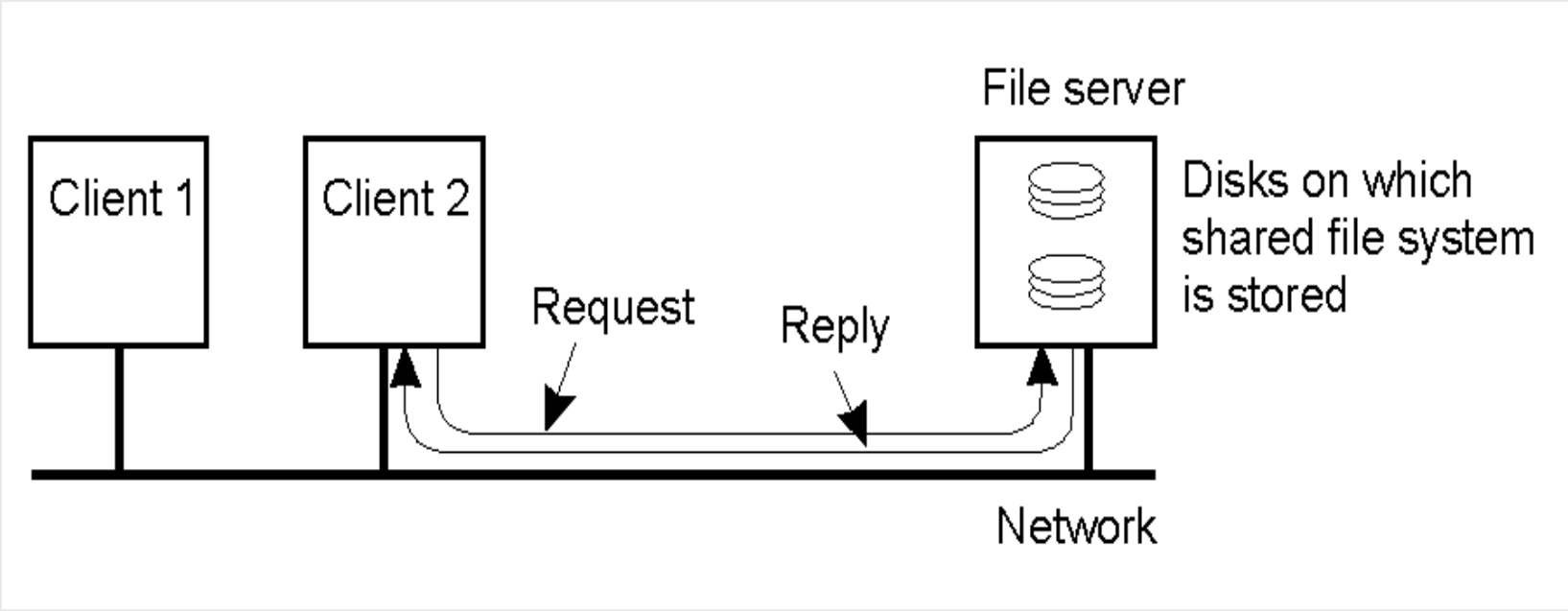


Network Operating System



Network Operating System

- Employs a client-server model
  - Minimal OS kernel
  - Additional functionality as user processes



Network-Operating Systems

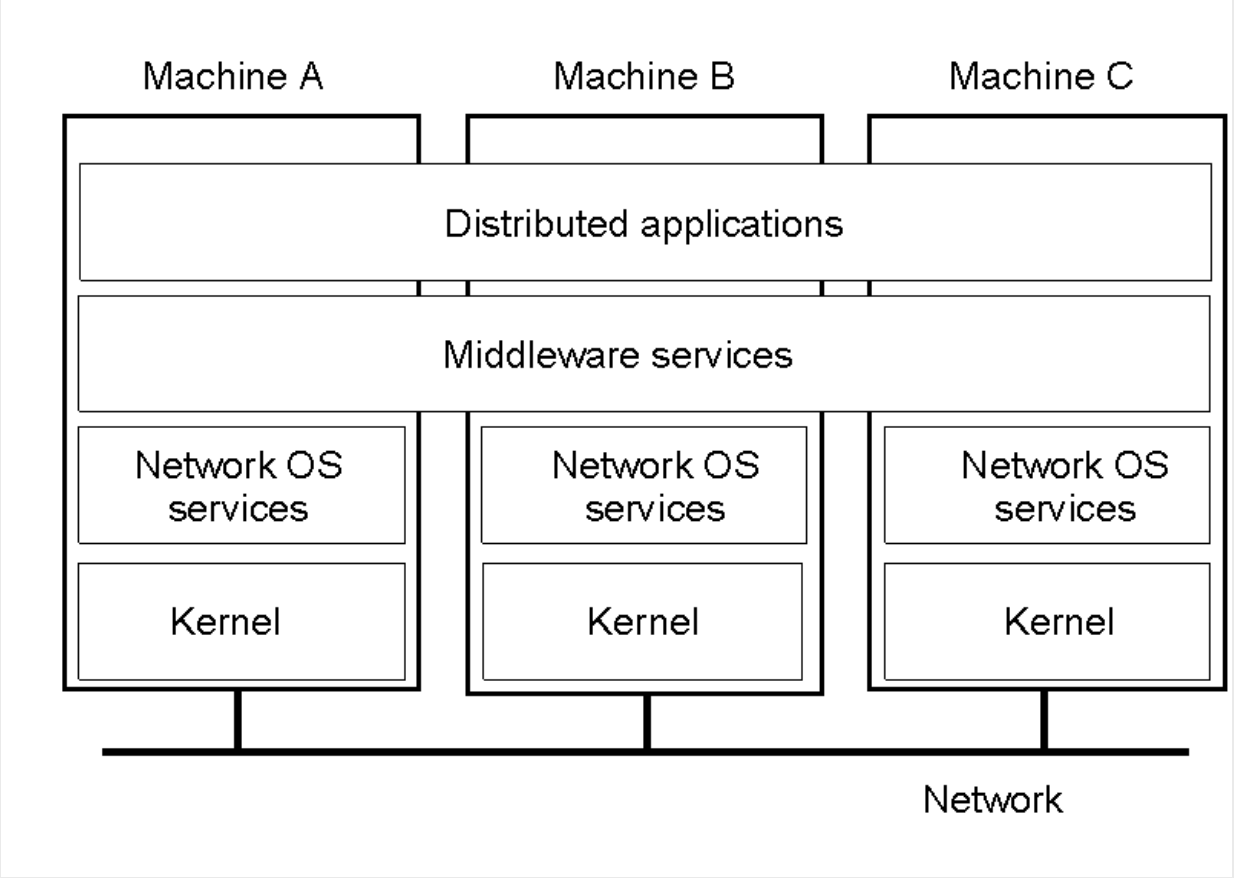
- Users are aware of multiplicity of machines. Access to resources of various machines is done explicitly by
  - Remote logging into the appropriate remote machine.
  - Transferring data from remote machines to local machines, via the File Transfer Protocol (FTP) mechanism.

Distributed Operating System

- Users not aware of multiplicity of machines.
- Manages resources in a distributed system
  - Seamlessly and transparently to the user
- Looks to the user like a centralized OS
  - But operates on multiple independent CPUs
- Provides transparency
  - Location, migration, concurrency, replication,…
- Presents users with a virtual uniprocessor

Middleware-based Systems

- General structure of a distributed system as middleware.



What is Middleware

- Software above the operating system but below the application program
- Middleware refers to the software that is common to multiple applications and builds on the network transport services to enable ready development of new applications and network services
- CORBA, DCOM, Java RMI, Grid

Middleware Layer

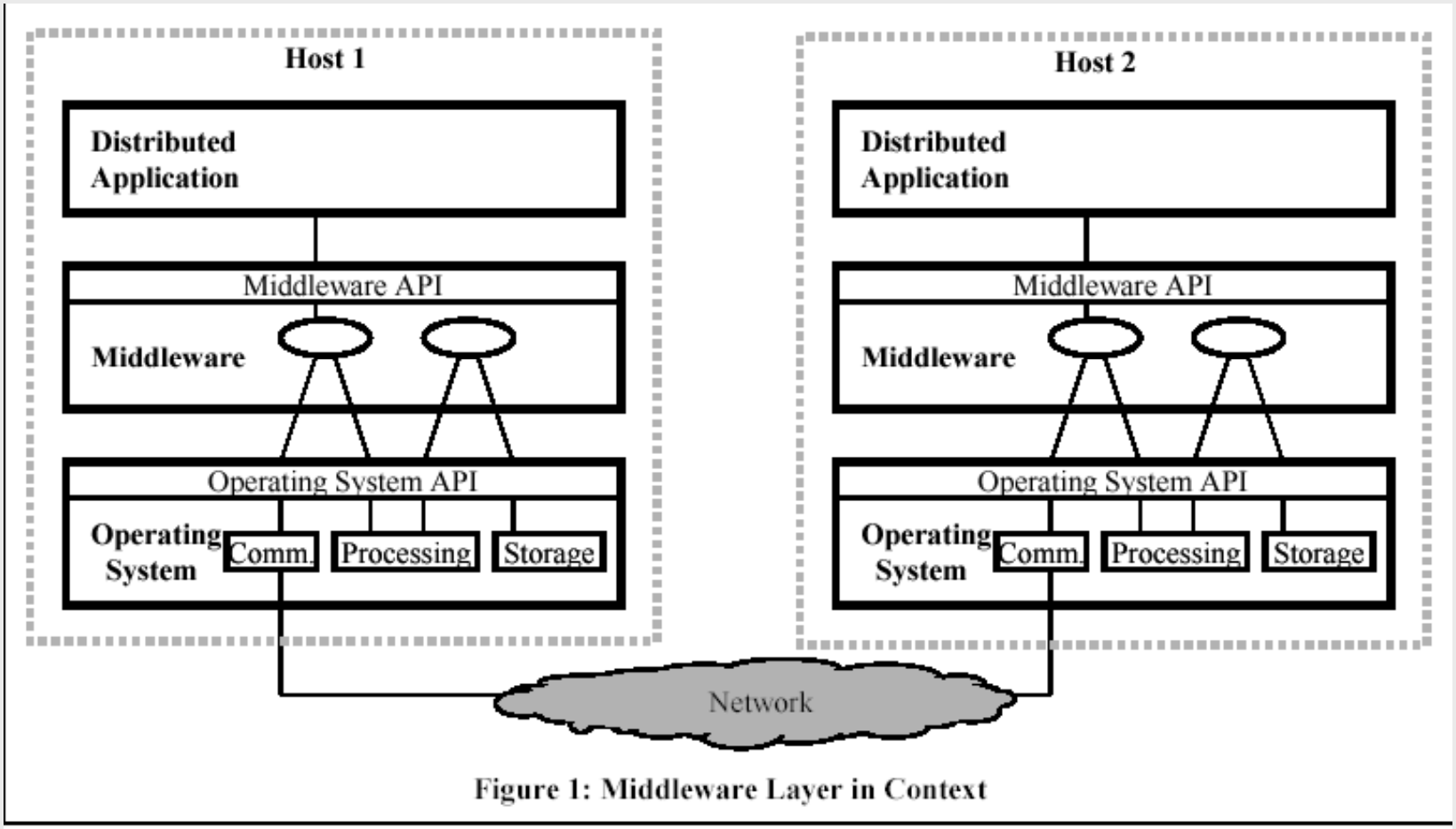


Figure 1: Middleware Layer in Context

Technical Challenges

- **The changing environment**
  - Computing world has changed and middleware has to adapt to this ever changing environment
- **Architecture**
  - New technological advances impose changes in established middleware architecture
- **Dynamic configuration**
  - Dynamic changes in system configuration will be inherent characteristics of future computing environments.

Client/Server Organization

- Server: a sw module manages a set of resources of a particular type using certain policies and methods.
  - Servers may be run in different machines
    - Mail server, http server
  - A machine can maintain more than one server
- Client: a sw module requests services from servers.
- Centralized server versus by distributed servers
  - centralized server: e.g. printer and mail
  - distributed servers: e.g. file servers
- Proxy server and caches: middleman between origin server and clients

Peer-to-Peer Organization

- All processes play similar roles, interacting cooperatively as peers to perform a distributed activity or computation without any distinction between clients and servers.
- Fully distributed and parallel
- For Example
  - Remote memory access
  - Process migration
  - P2P file exchange

Mobile Code Organization

- Mobile codes
  - Programs that function as they are transferred from one host to the other. Instead of sending requests associated with input data to a server for processing, the mobile code approach uploads codes to the server for execution
  - E.g. Javascript code, Java Applets
- Mobile agent
  - has the ability to travel from host to host autononmously, carrying their code as well as running state.
  - Itinerary mobility (proactive mobility)
  - Security in mobile agents
    - Server protection
    - Agent protection

Any Questions?

Questions?

- What is the difference between operating system and (software) system?
- What is the difference between network OS and Distributed OS?
- What is the difference between Distributed OS and Distributed (software) system?
- What is middleware?
- What is the difference between middleware and Distributed (software) system?

Distributed Systems

- What is a distributed (computing) system?

"A collection of independent computers that appears to its users as a single coherent system"

-A. Tanenbaum

Examples

- Some examples of distributed systems
  - Department computing cluster
  - Corporate systems
- Application examples
  - Email
  - News
  - Multimedia information systems- video conferencing
  - Airline reservation system
  - Banking system
- What is the most used distributed system?
  - World Wide Web

Distributed vs. Single Systems

- Data sharing
  - Multiple users can access common database, data files,…
- Device/resource sharing
  - Printers,servers,CPUs,….
- Communication
  - Communication with other machines…
- Flexibility
  - Spread workload to different & most appropriate machines
- Extensibility
  - Add resources and software as needed

Distributed vs. Centralized Systems

- Economics
  - Microprocessors have better price/performance than mainframes
- Speed
  - Collective power of large number of systems
- Geographic and responsibility distribution
- Reliability
  - One machine's failure need not bring down the system
- Extensibility
  - Computers and software can be added incrementally

Disadvantages of Distributed Systems

- Software
  - Little software exists compared to central processing
  - Complexity of the system: coordination of processes
- Networking
  - Still slow and can cause other problems (e.g. when disconnected)
- Security
  - Data may be accessed by unauthorized users
  - Authentication, in addition to access and flow control

Key Characteristics

- Support for resource sharing
- Openness
- Concurrency
- Scalability
- Fault tolerance (reliability)
- Transparence

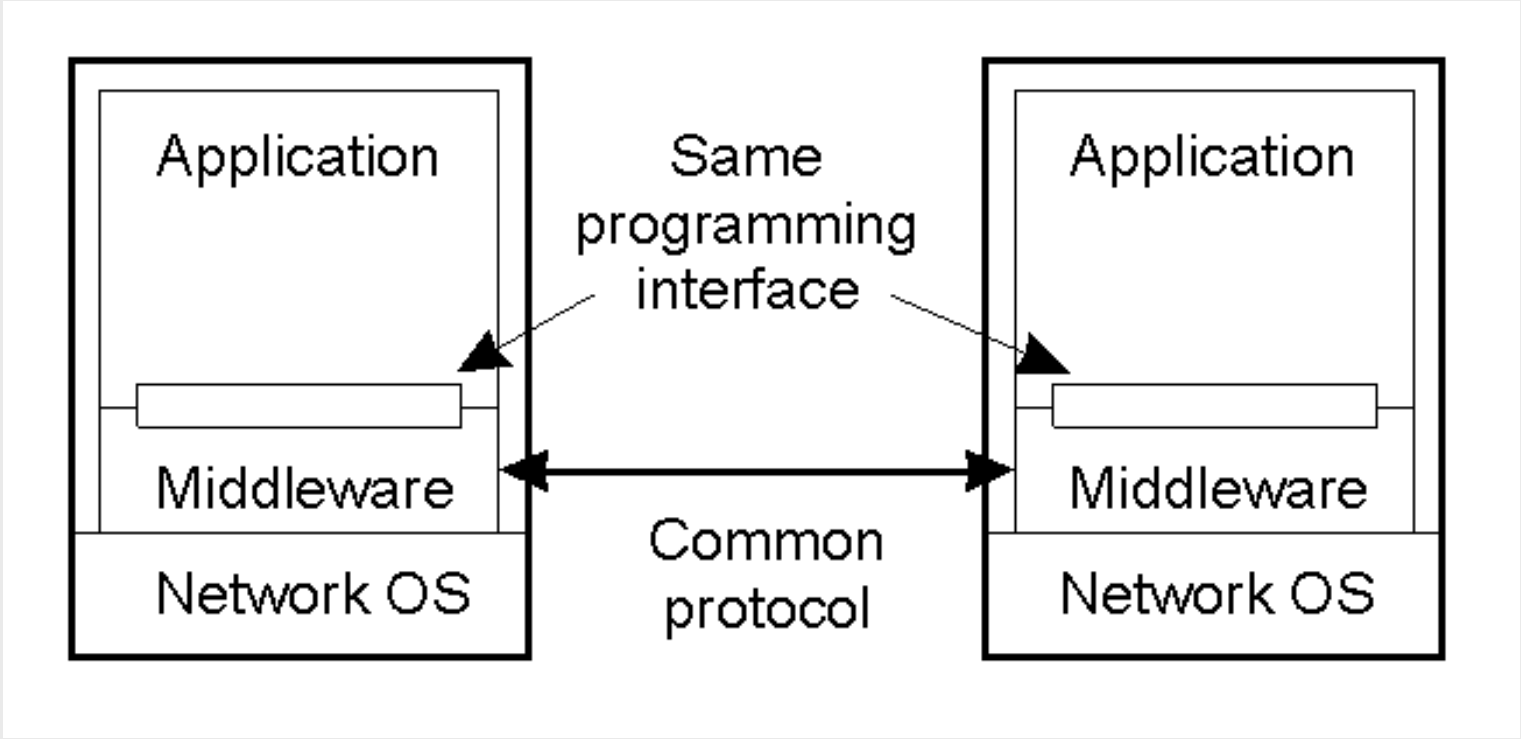
Resource Sharing

- Share hardware,software,data and information
- Hardware devices
  - Printers,disks,memory,…
- Software sharing
  - Compilers,libraries,toolkits,…
- Data
  - Databases,files,…

Openness

- Determines whether the system can be extended in various ways without disrupting existing system and services
- Hardware extensions
  - Adding peripherals,memory,communication interfaces…
- Software extensions
  - Operating systems features
  - Communication protocols
- Standard rules and protocols

Middleware and Openness



- In an open middleware-based distributed system, the protocols used by each middleware layer should be the same, as well as the interfaces they offer to applications.

Concurrency

- In a single system several processes are interleaved
- In distributed systems: there are many systems with one or more processors
  - Many users simultaneously invoke commands or applications
  - Many servers processes run concurrently, each responding to different client request
  - What is the difference between parallel and concurrent processing?

Scalability

- Scale of system
  - Few PCs servers ->dept level systems->local area networks->internetworked systems->wide area network…
  - Ideally, system and application software should not change as systems scales
- Scalability depends on all aspects
  - Hardware
  - Software
  - Networks
  - Application

Fault Tolerance

- Ability to operate under failures: possibly at a degraded performance level
- Two approaches:
  - Hardware redundancy: use of redundant components
  - Software recovery: design of programs to recover, Checkpointing/migration
- In distributed systems:
  - Servers can be replicated
  - Databases may be replicated
  - Software recovery involves the design so that state of permanent data can be recovered
- Fault detection

Transparency in a Distributed System

Transparency	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may be moved to another location while in use
Replication	Hide that a resource may be shared by several competitive users
Concurrency	Hide that a resource may be shared by several competitive users
Failure	Hide the failure and recovery of a resource
Persistence	Hide whether a (software) resource is in memory or on disk

Distributed Operating System

- Manages resources in a distributed system
  - Seamlessly and transparently to the user
- Looks to the user like a centralized OS
  - But operates on multiple independent CPUs
- Provides transparency
  - Location, migration, concurrency, replication,…
- Presents users with a virtual uniprocessor

Types of Existing Distributed OSs

System	Description	Main Goal
DOS	Tightly-coupled operating system for multi-processors and homogeneous multicomputers	Hide and manage hardware resources
NOS	Loosely-coupled operating system for heterogeneous multicomputers (LAN and WAN)	Offer local services to remote clients
Middleware	Additional layer atop of NOS implementing general-purpose services	Provide distribution transparency

An overview of

- DOS (Distributed Operating Systems)
- NOS (Network Operating Systems)
- Middleware

Comparison between Systems

Item	Distributed OS		Network OS	Middleware-based OS
	Multiproc.	Multicomp.		
Degree of transparency	Very High	High	Low	High
Same OS on all nodes	Yes	Yes	No	No
Number of copies of OS	1	N	N	N
Basis for communication	Shared memory	Messages	Files	Model specific
Resource management	Global, central	Global, distributed	Per node	Per node
Scalability	No	Moderately	Yes	Varies
Openness	Closed	Closed	Open	Open

Summary

- Key issues of distributed systems
- Hardware concepts
  - Multiprocessors
  - Multicomputers
  - Distributed systems
- Software concepts
  - Uniprocessor OS
  - Distributed OS
  - Network OS
  - Middleware
- Readings
  - Review Central OS, Chapter 1 of the text