

CHAPTER 1: OPERATING SYSTEM FUNDAMENTALS

What is an operating system?

- A collection of software modules to assist programmers in enhancing system *efficiency, flexibility, and robustness*
- An *Extended Machine* from the users' viewpoint
- A *Resource Manager* from the system's viewpoint

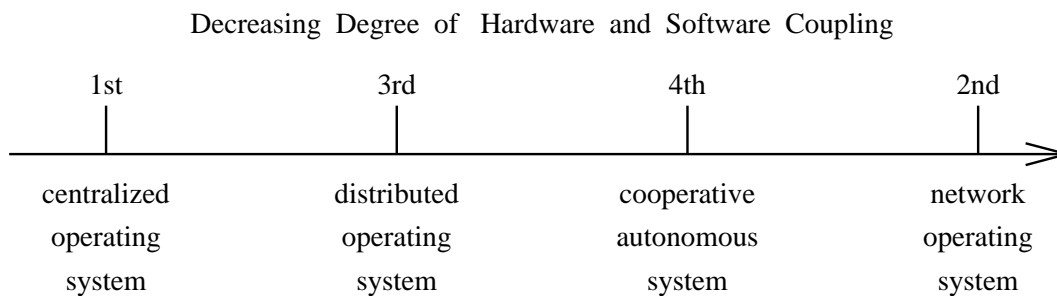
What are the primary functions of an operating system?

- multiplexing the processor(s)
- scheduling processes
- coordinating interaction among processes, interprocess communication and synchronization
- managing system resources (I/O, memory, data files)
- enforcing access control and protection
- maintaining system integrity and performing error recovery
- providing an interface to the users

Evolution of modern operating systems

1. *Centralized operating system*: resource management and extended machine to support *Virtuality*
2. *Network operating system*: resource sharing to achieve *Interoperability*
3. *Distributed operating system*: a single computer view of a multiple-computer system for *Transparency*
4. *Cooperative autonomous system*: cooperative work with *Autonomicity*

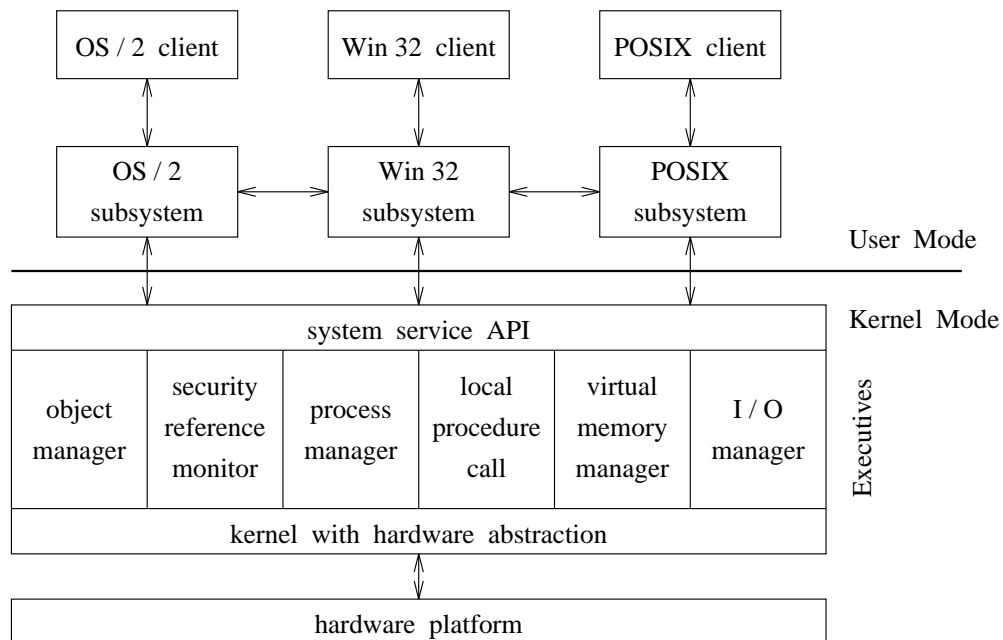
A spectrum of modern operating systems



Operating system structuring methods

- modularization
- vertical partitioning (layered one-in-one-out structure)
- horizontal partitioning
- client/server model
- minimal (or micro) kernel
- subsystem with API and SPI

Windows NT: an example of operating system structure



Overview of centralized operating systems:

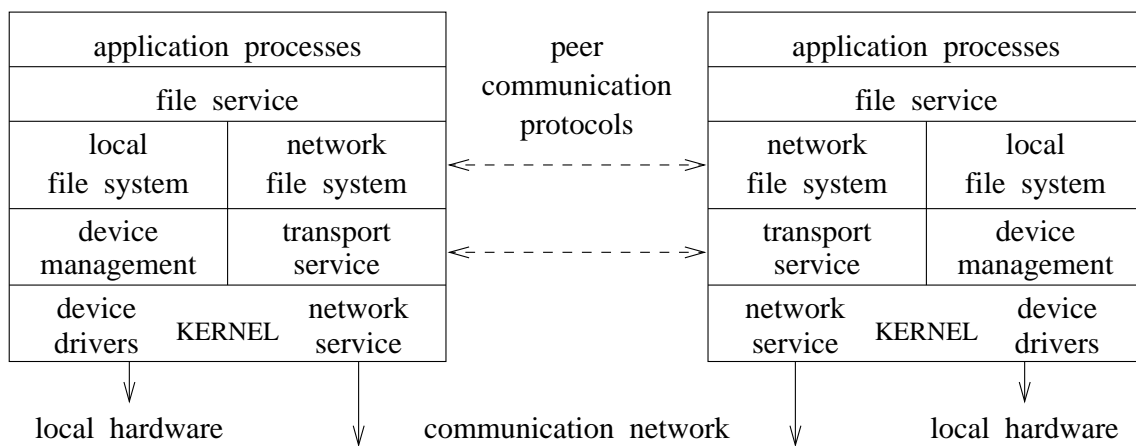
Resource Manager

- *Process management*
 - interprocess communication
 - process synchronization
 - process scheduling
- *Memory management*
 - memory allocation and deallocation
 - logical to physical address mapping
 - virtual memory support: segmentation and paging
 - protection
- *Device management*
 - device driver
 - buffering
 - spooling
- *Data management*
 - file access
 - file sharing
 - concurrency control
 - data replication

Network operating system

- *interoperability*: ability of information exchange among heterogeneous systems
- supported by network communication protocols
- *transport service*: the primary interface between operating system and computer network
- characterized by common network applications (servers)
 - remote login
 - file transfer
 - messaging
 - browsing
 - remote execution

A network file system example



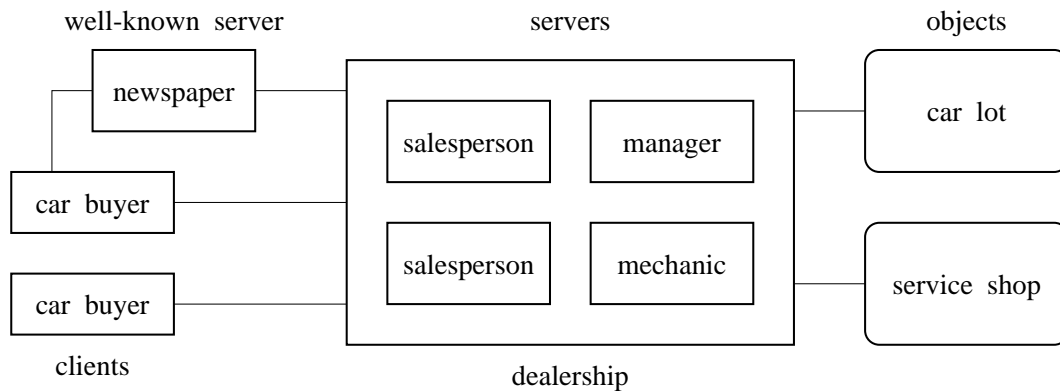
Distributed operating system

- transparency
- servers for supporting resource sharing and distributed processing
- algorithms to implement transparencies
- details in latter chapters

Cooperative autonomous system

- client/server model
- object model
- software bus (middleware, broker, or trader)
- CORBA and ODP
- Peer-to-Peer (P2P) systems

An example of cooperative autonomous system



Why do we need distributed control algorithms?

An algorithm is sometimes called protocol if it specifies coordination more than computation.

- algorithm changes due to message passing
- need for consensus algorithms due to lack of global information
- concurrency control algorithms to avoid interference in resource sharing
- coherency control algorithms to maintain consistency for data replication
- protocols for group communication in distributed applications
- fault-tolerance algorithms for handling failure and recovery
- real-time and distributed scheduling algorithms