



Types Of Operating Systems

Date 10/01/2004



Brief history of OS design

In the beginning

- ◆ OSes were runtime libraries
 - The OS was just code you linked with your program and loaded into the computer.
 - First computer interface was switches and lights, then punched tape and cards.
- ◆ Next was Main frame systems which includes
 - Batch Systems
 - OS was permanently loaded in Primary memory
 - Input devices were card readers and tape drives
 - Output devices were printers, tape drives, card punches
 - User \neq Operator (hire an operator)
 - No direct user interactions



- User prepares a job consist of program,data and some control information about the nature of the job
- Output consist of result of the program as well as the dump of the final memory and register contents for debugging.
- Reduce setup time by batching similar jobs
- Automatic job sequencing – automatically transfers control from one job to another.
- Main task of the OS was to transfer control from one job to another.
- Problems
 1. How does the monitor know about the nature of the job (e.g., Fortran versus Assembly) or which program to execute?
 2. How does the monitor distinguish
 - (a) job from job?
 - (b) data from program?
- Solution
 - Introduce control cards

Memory Layout for a Simple Batch System



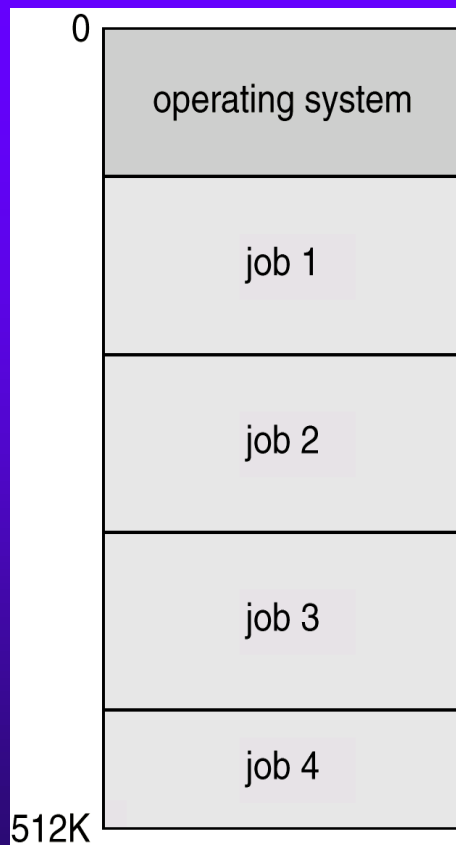


- Parts of OS
 - Control card interpreter – responsible for reading and carrying out instructions on the cards.
 - Loader – loads systems programs and applications programs into memory.
 - Device drivers – know special characteristics and properties for each of the system's I/O devices.
- Problem: Slow Performance – I/O and CPU could not overlap ; card reader very slow.
- Solution: Off-line operation – speed up computation by loading jobs into memory from tapes and card reading and line printing done off-line.
- Spooling Overlap I/O of one job with computation of another job. While executing one job, the OS reads next job from card reader into a storage area on the disk (job queue). Outputs printout of previous job from disk to printer.
- *Job pool* – data structure that allows the OS to select which job to run next in order to increase CPU utilization

– Multi Programmed Systems

Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.

High CPU utilization compared to batch processing.





Multi programming

- Keeps multiple runnable jobs loaded in memory
- Overlaps I/O processing of a job with computation of another.
- Benefits from I/O devices that can operate asynchronously
- Requires the use of interrupts and DMA
- Optimizes system throughput (number of jobs finished in a given amount of time) at the cost of response time.
- **OS Features Needed for Multiprogramming**
 - I/O routine supplied by the system.
 - Memory management – the system must allocate the memory to several jobs.
 - CPU scheduling – the system must choose among several jobs ready to run.
 - Allocation of devices.

– Time Sharing (Multi tasking) Systems

- Logical extension of multiprogramming system
- The CPU is multiplexed among several jobs that are kept in memory and on disk (the CPU is allocated to a job only if the job is in memory).
- It allows many users to share the computer simultaneously.
- Gives illusion that each user has his own machine.
- Based on time slicing dividing CPU time among the users.
- Introduce new class of application – interactive
- Users interact(through mouse,keyboard etc..) with computers (editors,debuggers etc..)

– Desktop Systems

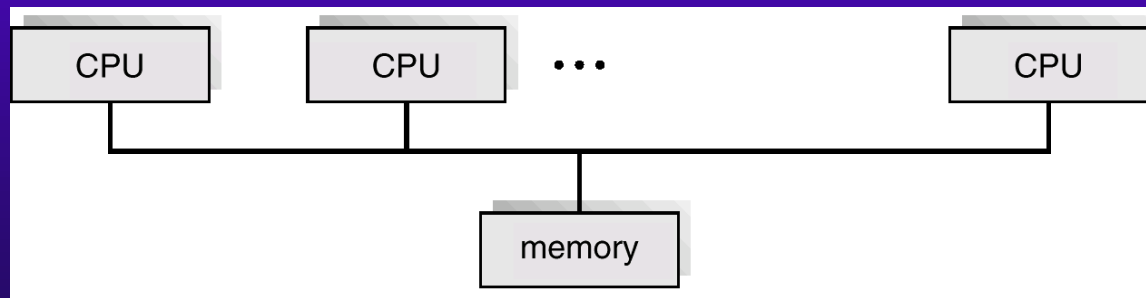
- computer system dedicated to a single user.
- Preferences -- User convenience and responsiveness
- Main OS in use – Windows,UNIX,LINUX,MAC..
- Can adopt technology developed for larger operating system' often individuals have sole use of computer and do not need advanced CPU utilization of protection features.

– Multi processor Systems

- More than one processor in close communication, sharing the computer bus, the clock and sometimes memory and peripheral devices.
- Advantages of parallel system:
 - Increased *throughput*
 - Economical
 - Increased reliability
 - graceful degradation, fail-soft systems



- *Symmetric multiprocessing (SMP)*
 - Each processor runs an identical copy of the OS.
 - Many processes can run at once without performance deterioration.
 - Most modern operating systems support SMP
 - *Tightly coupled system* – processors share memory and a clock; communication usually takes place through the shared memory.
- *Asymmetric multiprocessing*
 - Each processor is assigned a specific task; master processor schedules and allocates work to slave processors.
 - More common in extremely large systems



– Distributed Systems

- Distribute the computation among several physical processors.
- *Loosely coupled system* – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or telephone lines.
- Enables Parallelism but speed up is not the goal.
- Advantages of distributed systems.
 - Resources Sharing
 - Computation speed up – load sharing
 - Reliability
 - Communications
- Network Operating System
 - provides file sharing
 - provides communication scheme
 - runs independently from other computers on the network



- Distributed Operating System
 - less autonomy between computers
 - gives the impression there is a single operating system controlling the network.
- Types of Distributed Systems
 - Client – Server Systems
 - Compute server system, File server system
 - Peer to Peer Systems

– Clustered Systems

- Usually performed to provide high availability.
- In Asymmetric clustering one machine will be in hot stand by mode while other is running the application.
- In Symmetric clustering 2 or more hosts are running applications and they are monitoring each other. This mode is more efficient.
- Parallel clusters allow multiple hosts to access the same data on the shared storage.

– Real-Time Systems

- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.
- Well-defined fixed-time constraints.
- *Hard real-time system.*
 - Secondary storage limited or absent, data stored in short-term memory, or read-only memory (ROM)
 - Conflicts with time-sharing systems, not supported by general-purpose operating systems.
- *Soft real-time system*
 - Limited utility in industrial control or robotics
 - Useful in applications (multimedia, virtual reality) requiring advanced operating-system features.
- Examples QNX,RTLINUX.



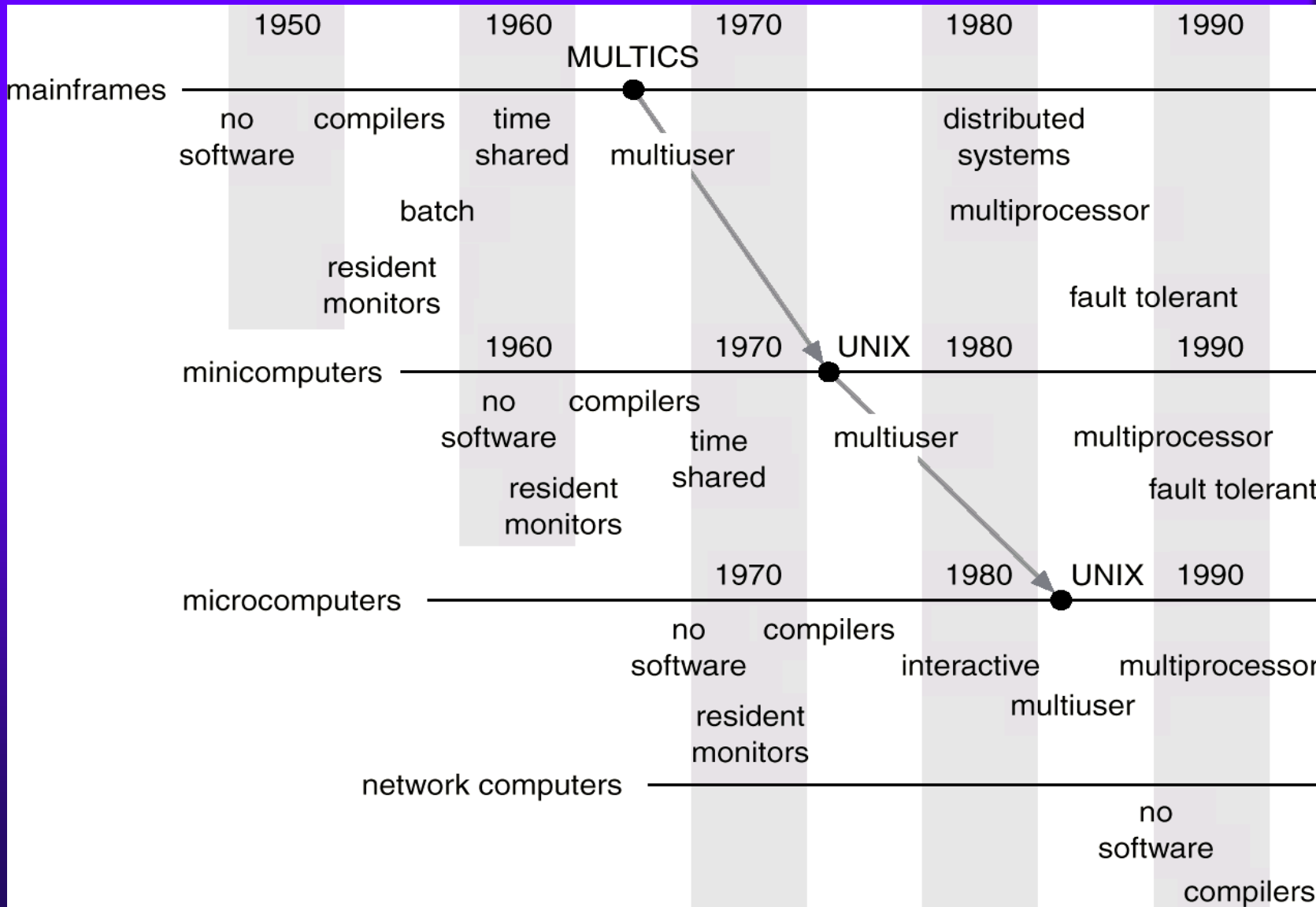
– Embedded Operating Systems

- OS embed on the System itself
- Fast, Application specific
- Examples Processor in modern washing machines, Cell phones, Control systems etc...

– Handheld Systems

- Power consumption and weight must be low
- Memory ranges from 512KB to 8MB.
- Speed of the processor can not be very high because of the power consumption.

Migration of Operating-System Concepts and Features





Next Class

- ◆ Design Approaches for an operating System
 - System components
 - Operating system services
 - System calls
 - System programs
 - System structure
 - Virtual machines

Read Chapter 3.