

Operating Systems Structure Chapter 2

1

Operating System

- Is a control program that controls the execution of application programs
 - OS must relinquish control to user programs and regain it safely and efficiently
 - Tells the CPU when to execute other pgms
- Is an interface between the user and hardware
- Masks the details of the hardware to application programs
 - Hence OS must deal with hardware details

2

Services Provided by the OS

- Facilities for Program creation
 - editors, compilers, linkers, and debuggers
- Program execution
 - loading in memory, I/O and file initialization
- Access to I/O and files
 - deals with the specifics of I/O and file formats
- System access
 - Protection in access to resources and data
 - Resolves conflicts for resource contention

3

Evolution of an Operating System

- Must adapt to hardware upgrades and new types of hardware. Examples:
 - Character vs graphic terminals
 - Introduction of paging hardware
- Must offer new services, eg: internet support
- The need to change the OS on regular basis place requirements on it's design
 - modular construction with clean interfaces
 - object oriented methodology

4

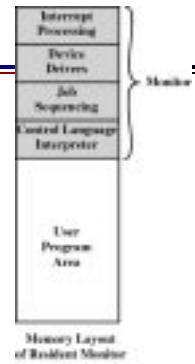
Simple Batch Systems

- Are the first operating systems (mid-50s)
- The user submit a job (written on card or tape) to a computer operator
- The computer operator place a **batch** of several jobs on a input device
- A special program, the **monitor**, manages the execution of each program in the batch
- **Resident monitor** is in main memory and available for execution
- Monitor utilities are loaded when needed

5

The Monitor

- Monitor reads jobs one at a time from the input device
- Monitor places a job in the user program area
- A monitor instruction branches to the start of the user program
- Execution of user pgm continues until:
 - end-of-pgm occurs
 - error occurs
- Causes the CPU to fetch its next instruction from Monitor



6

Job Control Language (JCL)

- Is the language to provide instructions to the monitor
 - what compiler to use
 - what data to use
- Example of job format: ----->>
- \$FTN loads the compiler and transfers control to it
- \$LOAD loads the object code (in place of compiler)
- \$RUN transfers control to user program

```

$JOB
$FTN
...
FORTRAN
program
...
$LOAD
$RUN
...
Data
...
$END
    
```

7

Job Control Language (JCL)

- Each read instruction (in user pgm) causes one line of input to be read
- Causes (OS) input routine to be invoke
 - checks for not reading a JCL line
 - skip to the next JCL line at completion of user program

8

Batch OS

- Alternates execution between user program and the monitor program
- Relies on available hardware to effectively alternate execution from various parts of memory

9

Desirable Hardware Features

- Memory protection
 - do not allow the memory area containing the monitor to be altered by user programs
- Timer
 - prevents a job from monopolizing the system
 - an interrupt occurs when time expires

10

Desirable Hardware Features

- Privileged instructions
 - can be executed only by the monitor
 - an interrupt occurs if a program tries these instructions
- Interrupts
 - provides flexibility for relinquishing control to and regaining control from user programs

11

Multiprogrammed Batch Systems

- I/O operations are exceedingly slow (compared to instruction execution)
- A program containing even a very small number of I/O ops, will spend most of its time waiting for them
- Hence: poor CPU usage when only one program is present in memory



12

Multiprogrammed Batch Systems

- If memory can hold several programs, then CPU can switch to another one whenever a program is awaiting for an I/O to complete
- This is **multitasking (multiprogramming)**



13

Requirements for Multiprogramming

- Hardware support:
 - I/O interrupts and (possibly) DMA
 - in order to execute instructions while I/O device is busy
 - Memory management
 - several ready-to-run jobs must be kept in memory
 - Memory protection (data and programs)
- Software support from the OS:
 - Scheduling (which program is to be run next)
 - To manage resource contention

14

Example: three jobs are submitted

	JOB1	JOB2	JOB3
Type of job	Heavy compute	Heavy I/O	Heavy I/O
Duration	5 min.	15 min.	10 min.
Memory req.	50K	100 K	80 K
Need disk?	No	No	Yes
Need terminal	No	Yes	No
Need printer?	No	No	Yes

- Almost no contention for resources
- All 3 can run in minimum time in a multitasking environment (assuming JOB2/3 have enough CPU time to keep their I/O operations active)

15

Advantages of Multiprogramming

	Uniprogramming	Multiprogramming
Processor use	17%	33%
Memory use	33%	67%
Disk use	33%	67%
Printer use	33%	67%
Elapsed time	30 min.	15 min.
Throughput rate	6 jobs/hr	12 jobs/hr
Mean response time	18 min.	10 min.

16

Time Sharing Systems (TSS)

- Batch multiprogramming does not support interaction with users
- TSS extends multiprogramming to handle multiple interactive jobs
- Processor's time is shared among multiple users
- Multiple users simultaneously access the system through terminals

17

Time Sharing Systems (TSS)

- Because of slow human reaction time, a typical user needs 2 sec of processing time per minute
- Then (about) 30 users should be able to share the same system without noticeable delay in the computer reaction time
- The file system must be protected (multiple users...)

18

System Structure

- Because of its enormous complexity, we view the OS system as a series of levels
- Each level performs a related subset of functions
- Each level relies on the next lower level to perform more primitive functions
- Well defined interfaces: one level can be modified without affecting other levels
- This decomposes a problem into a number of more manageable sub problems

19

Monolithic systems

- Monolithic/layered system
- All services provided by the OS
- OS functionality provided by the OS/kernel
- Unix, VMS, Linux
- Features
 - Easy to understand
 - Simplifies construction
 - Careful interface
 - Very rigid

20

Characteristics of Modern Operating Systems

- New design elements were introduced recently
- In response to new hardware development
 - multiprocessor machines
 - high-speed networks
 - faster processors and larger memory
- In response to new software needs
 - multimedia applications
 - Internet and Web access
 - Client/Server applications

21

Microkernel architecture

- Support a minimal set of essential functions in the kernel
 - primitive memory management (address space)
 - Interprocess communication (IPC)
 - basic scheduling
- Other OS services are provided by processes running in user mode (servers)
 - device drivers, file system, virtual memory...
- More flexibility, extensibility, portability...
- A performance penalty by replacing service calls with message exchanges between process...

22

Advantages of kernelized design

- Supports heterogeneity
- Supports flexibility
- Supports extensibility
- Challenge- performance/security

23