Chapter 1

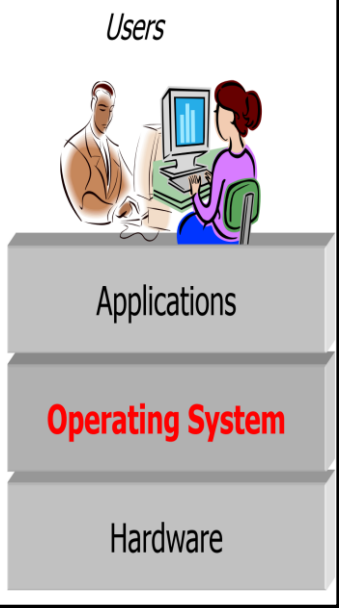
INTRODUCTION TO OPERATING SYSTEMS

Introduction

- What is an operating system?
- Operating Systems History
  - ✓ Simple Batch Systems
  - ✓ Multiprogrammed Batch Systems
  - ✓ Time-sharing Systems
  - ✓ Personal Computer Systems
- Parallel and Distributed Systems
- Real-time Systems

What is an Operating System?

- An OS is a program that acts an intermediary between the user of a computer and computer hardware.
- Major cost of general purpose computing is software.
  - ✓ OS simplifies and manages the complexity of running application programs efficiently.



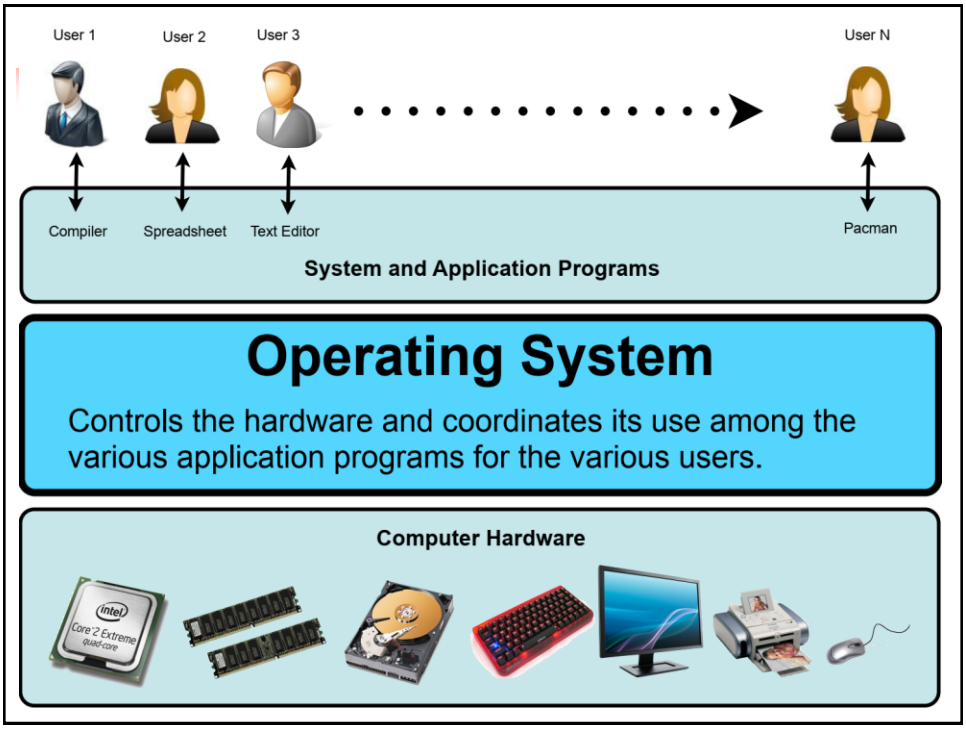
Đinh Công Đoan

Main functionalities of OS

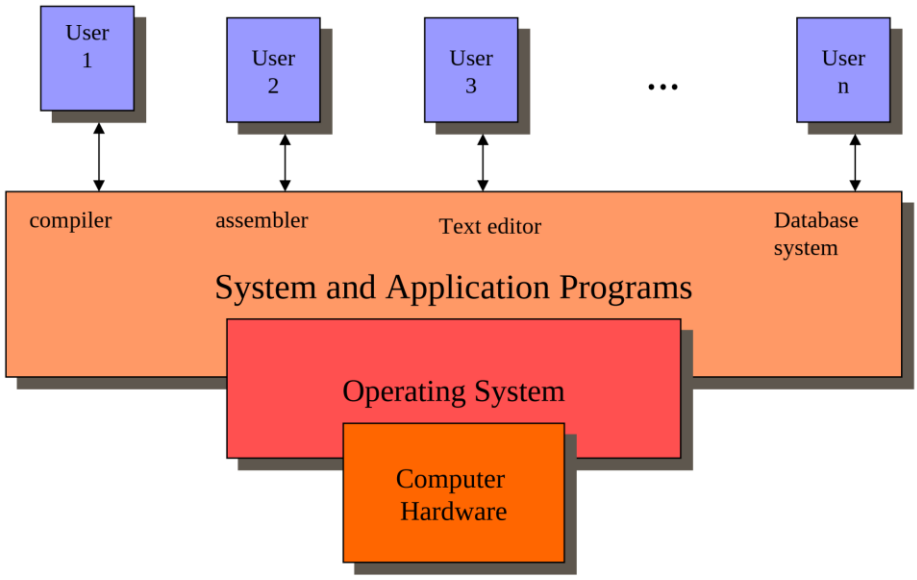
- Process Management
- Memory Management
- Storage Management
- Mass-Storage Management
- To provide effective and user friendly services

Computer System Components

- Hardware
  - ✓ Provides basic computing resources (CPU, memory, I/O devices).
- Operating System
  - ✓ Controls and coordinates the use of hardware among application programs.
- Application Programs
  - ✓ Solve computing problems of users (compilers, database systems,video games, business programs such as banking software).
- Users
  - ✓ People, machines, other computers



Abstract View of System



Operating System Views

- OS is a Resource allocator
  - ✓ Manages all resources
  - ✓ Decides between conflicting requests for efficient and fair resource use
- OS is a Control program
  - ✓ Controls execution of user programs and operation of I/O devices.
- Kernel
  - ✓ The program that executes forever (everything else is an application with respect to the kernel)

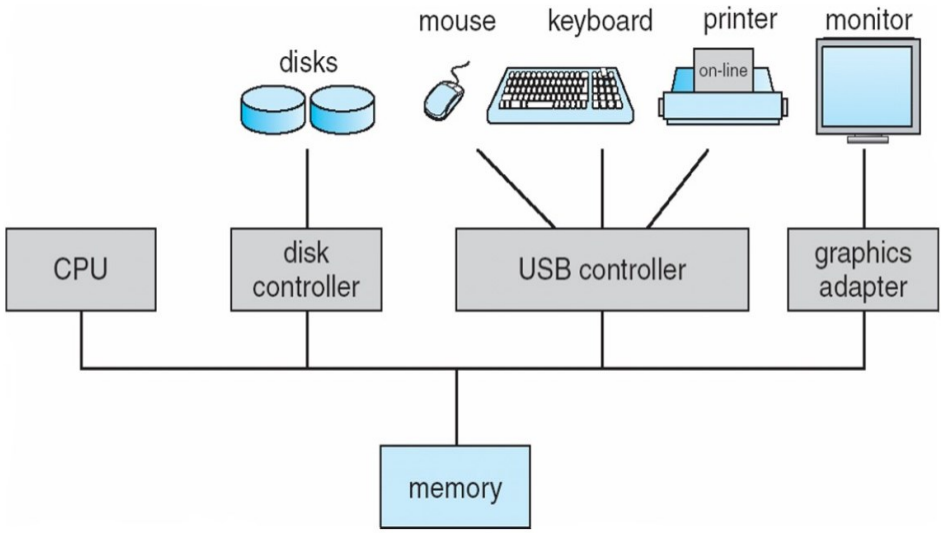
Goals of an Operating System

- Simplify the execution of user programs and make solving user problems easier.
- Use computer hardware efficiently.
  - ✓ Allow sharing of hardware and software resources.
- Make application software portable and versatile.
- Provide isolation, security and protection among user programs.
- Improve overall system reliability
  - ✓ Error confinement, fault tolerance, reconfiguration

Why should I study Operating Systems?

- Need to understand interaction between the hardware and applications
  - ✓ New applications, new hardware..
  - ✓ Inherent aspect of society today
- Need to understand basic principles in the design of computer systems
  - ✓ efficient resource management, security, flexibility
- Increasing need for specialized operating systems
  - ✓ e.g. embedded operating systems for devices – cell phones, sensors and controllers
  - ✓ real-time operating systems - aircraft control, multimedia services

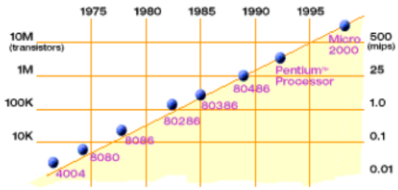
Computer System Organization

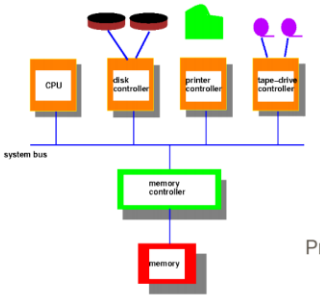


Systems Today

Hardware Complexity Increases

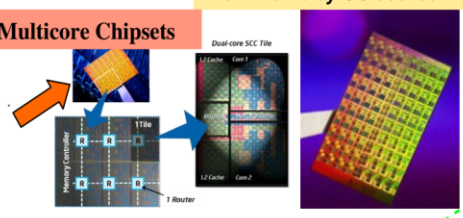
Moore's Law: 2X transistors/Chip Every 1.5 years

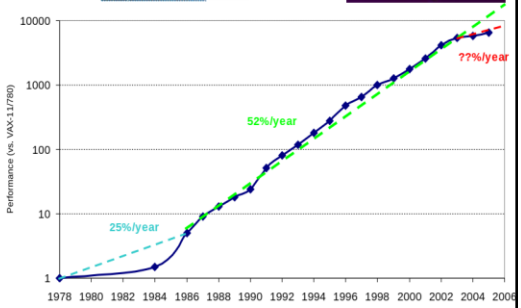




From Berkeley OS course

Intel Multicore Chipsets





Principles of Operating Systems -

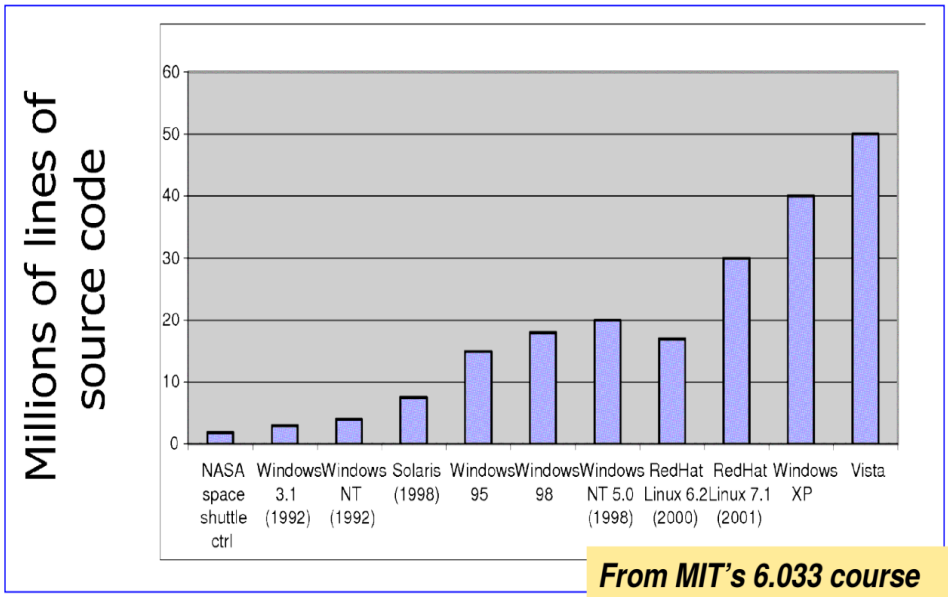
From Hennessy and Patterson, *Computer Architecture: A Quantitative Approach*, 4th edition, Sept. 15, 2006

OS needs to keep pace with hardware improvements

	1981	1997	2014	Factor (2014/1981)
Uniprocessor speed (MIPS)	1	200	2500	2.5K
CPUs per computer	1	1	10+	10+
$/Processor MIPS	$100K	$25	$0.20	500K
DRAM Capacity (MiB)/$	0.002	2	1K	500K
Disk Capacity (GiB)/$	0.003	7	25K	10M
Home Internet	300 bps	256 Kbps	20 Mbps	100K
Machine room network	10 Mbps (shared)	100 Mbps (switched)	10 Gbps (switched)	1000
Ratio of users to computers	100:1	1:1	1:several	100+

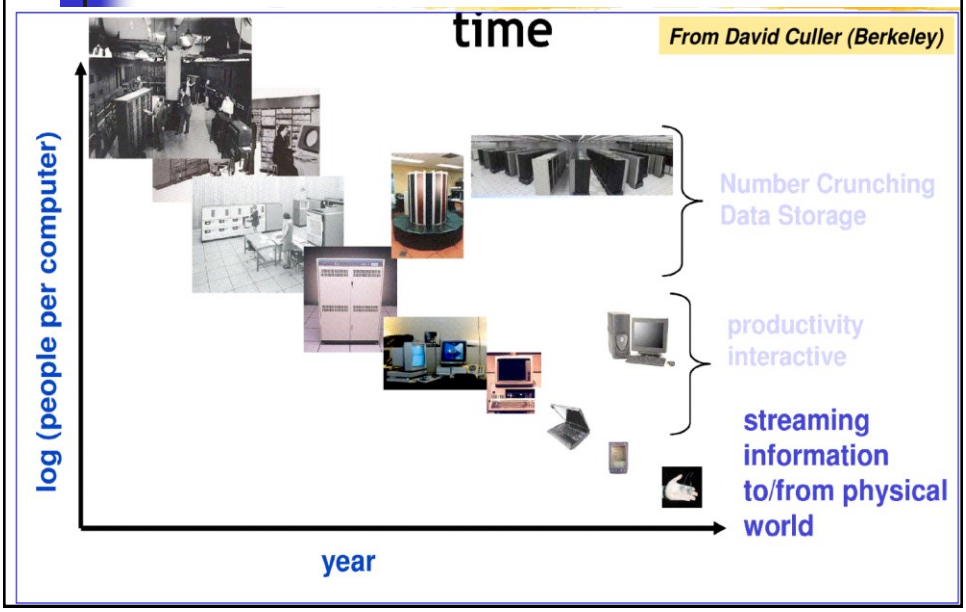
22

Software Complexity Increases



From MIT's 6.033 course

People-to-Computer Ratio Over Time



Operating System Spectrum

- Monitors and Small Kernels
  - ✓ special purpose and embedded systems, real-time systems
- Batch and multiprogramming
- Timesharing
  - ✓ workstations, servers, minicomputers,timeframes
- Transaction systems
- Personal Computing Systems
- Mobile Platforms, devices (of all sizes)

Early Systems - Bare Machine (1950s)

- **Hardware** – ***expensive*** ; **Human** – ***cheap***
- Structure
  - ✓ Large machines run from console
  - ✓ Single user system
    - Programmer/User as operator
  - ✓ Paper tape or punched cards
- Early software
  - ✓ Assemblers, compilers, linkers, loaders, device drivers, libraries of common subroutines.
- Secure execution
- Inefficient use of expensive resources
  - ✓ Low CPU utilization, high setup time.

Simple Batch Systems (1960's)

- Reduce setup time by batching jobs with similar requirements.
- Add a card reader, Hire an operator
  - ✓ User is NOT the operator
  - ✓ Automatic job sequencing
    - Forms a rudimentary OS.
  - ✓ Resident Monitor
    - Holds initial control, control transfers to job and then back to monitor.
  - ✓ Problem
    - Need to distinguish job from job and data from program

Supervisor/Operator Control

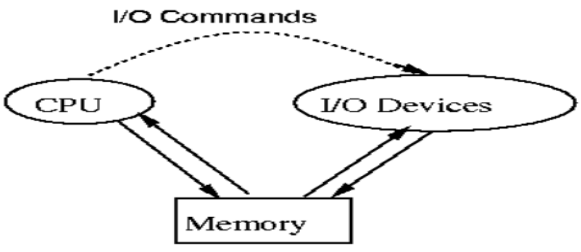
- Secure monitor that controls job processing
  - Special cards indicate what to do.
  - User program prevented from performing I/O
- Separate user from computer
  - User submits card deck
  - cards put on tape
  - tape processed by operator
  - output written to tape
  - tape printed on printer
- Problems
  - Long turnaround time - up to 2 DAYS!!!
  - Low CPU utilization
    - I/O and CPU could not overlap; slow mechanical devices

Batch Systems - Issues

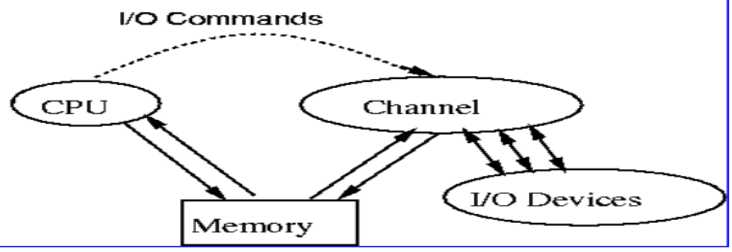
- Solutions to speed up I/O:
- Offline Processing
  - load jobs into memory from tapes, card reading and line printing are done offline.
- Spooling
  - Use disk (random access device) as large storage for reading as many input files as possible and storing output files until output devices are ready to accept them.
  - Allows overlap - I/O of one job with computation of another.
  - Introduces notion of a job pool that allows OS choose next job to run so as to increase CPU utilization.

Speeding up I/O

- Direct Memory Access (DMA)



- Channels



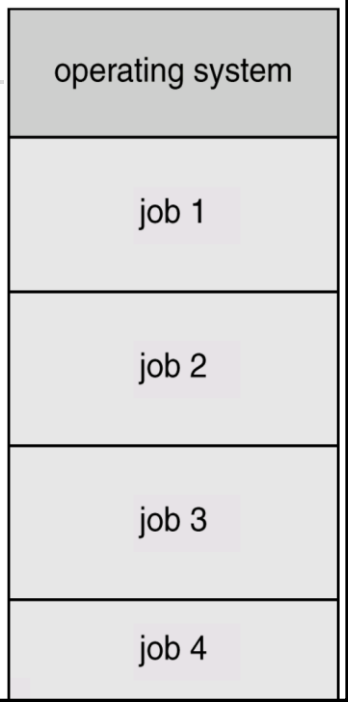
Batch Systems - I/O completion

- How do we know that I/O is complete?
  - ✓ Polling:
    - Device sets a flag when it is busy.
    - Program tests the flag in a loop waiting for completion of I/O.
  - ✓ Interrupts:
    - On completion of I/O, device forces CPU to jump to a specific instruction address that contains the interrupt service routine.
    - After the interrupt has been processed, CPU returns to code it was executing prior to servicing the interrupt

How are programs run ?

- Multiprogrammed systems
  - ✓ Many programs (Jobs) are loaded into main memory.
  - ✓ CPU exercution is share between programs (jobs)
  - ✓ To get the best of CPU utilization
- The roles of OS:
  - ✓ Jobs scheduling: to choose the proper one fromm jobs pool (on the mass storage disk) to bee loaded into memory for exercution.
  - ✓ Memory management
  - ✓ CPU scheduling
  - ✓ Resources allocation
  - ✓ Protection

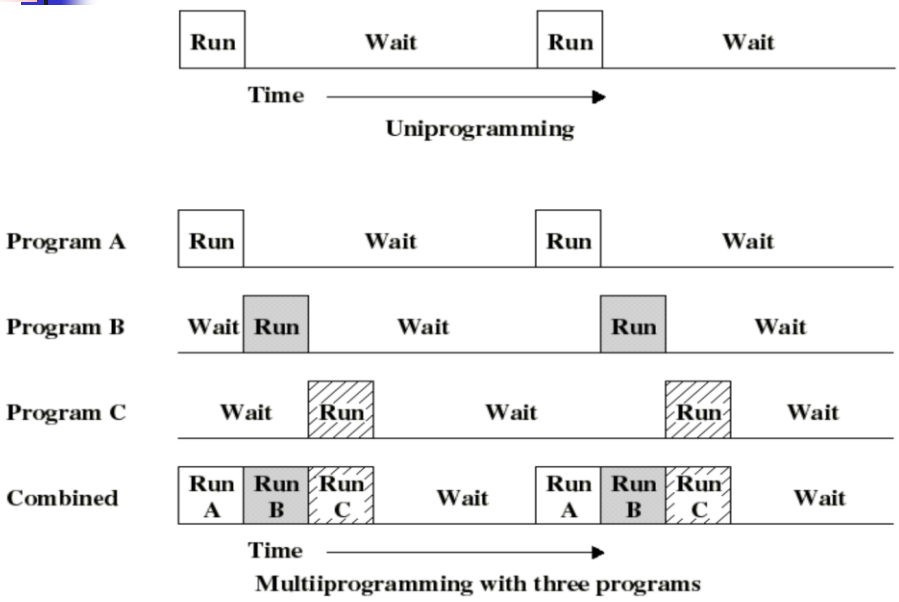
Đinh Công Đoan



Multiprogramming

- Use interrupts to run multiple programs simultaneously
  - ✓ When a program performs I/O, instead of polling, execute another program till interrupt is received.
- Requires secure memory, I/O for each program.
- Requires intervention if program loops indefinitely.
- Requires CPU scheduling to choose the next job to run

Multiple jobs



- Time-sharing systems
  - ✓ Multiprogrammed systems does not work well with users in term of interaction
  - ✓ CPU used interchangable between jobs
    - time slice, quantum time for each job when gain the CPU
    - Interactive interface user with higher response time (about 1 sesond)
  - ✓ Job can obtains CPU only when located in main memory.
  - ✓ Whenever required a job can be swapped out, moving from main memory to the storage device, providing memory space for another jobs

Timesharing

- **Hardware** – ***getting cheaper***; **Human** – ***getting expensive***
- Programs queued for execution in FIFO order.
- Like multiprogramming, but timer device interrupts after a quantum (timeslice).
  - ✓ Interrupted program is returned to end of FIFO
  - ✓ Next program is taken from head of FIFO
- Control card interpreter replaced by command language interpreter.

Timesharing (cont.)

- Interactive (action/response)
  - ✓ when OS finishes execution of one command, it seeks the next control statement from user.
- File systems
  - ✓ online filesystem is required for users to access data and code.
- Virtual memory
  - ✓ Job is swapped in and out of memory to disk

Personal Computing Systems

- **Hardware** – ***cheap*** ; **Human** – ***expensive***
- Single user systems, portable.
- I/O devices - keyboards, mice, display screens, small printers.
- Laptops and palmtops, Smart cards, Wireless devices.
- Single user systems may not need advanced CPU utilization or protection features.
- Advantages:
  - ✓ user convenience, responsiveness, ubiquitous

Paralell computers

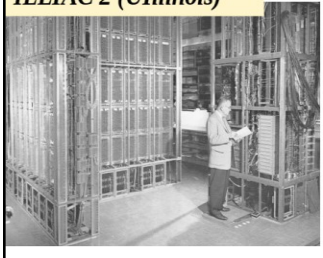
- Parallel systems (parallel, multiprocessor, or tightly-coupled system)
  - ✓ More than one CPU (Multi-Cores)
  - ✓ Share the common computer bus, clock
  - ✓ Advantages:
    - System throughput: more CPU → Faster proceesing
    - Effective use of system resources by sharing such as disks, I/O devices
    - Reliability: high

Parallel Systems

- Multiprocessor systems with more than one CPU in close communication.
- Improved Throughput, economical, increased reliability.
- Kinds:
  - ✓ Vector and pipelined
  - ✓ Symmetric and asymmetric multiprocessing
  - ✓ Distributed memory vs. shared memory
- Programming models:
  - ✓ Tightly coupled vs. loosely coupled ,message-based vs. shared variable

Parallel Computing Systems

ILLIAC 2 (UIllinois)



Climate modeling, earthquake simulations, genome analysis, protein folding, nuclear fusion research, …..

K-computer(Japan)

Connection Machine (MIT)

Tianhe-1(China)

IBM Blue Gene

Distributed Systems

- **Hardware** – ***very cheap*** ; **Human – *very expensive***
- Distribute computation among many processors.
- Loosely coupled –
  - ✓ no shared memory, various communication lines
- client/server architectures
- Advantages:
  - ✓ resource sharing
  - ✓ computation speed-up
  - ✓ Reliability
  - ✓ communication - e.g. email
- Applications - digital libraries, digital multimedia

Distributed Computing Systems

Globus Grid Computing Toolkit

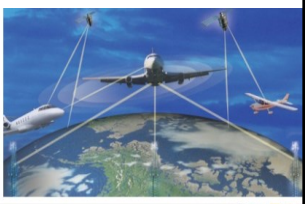
Cloud Computing Offerings

PlanetLab

Gnutella P2P Network

Real-time systems

- Correct system function depends on timeliness
- Feedback/control loops
- Sensors and actuators
- Hard real-time systems –
  - ✓ Failure if response time too long.
  - ✓ Secondary storage is limited
- Soft real-time systems –
  - ✓ Less accurate if response time is too long.
  - ✓ Useful in applications such as multimedia, virtual reality



A personal computer today

- Super AMOLED display
- Capacitive touchscreen (multitouch)
- Audio (speaker, microphone)
- Vibration
- S pen

- 4G LTE
- NFC
- WiFi
- Bluetooth
- Infrared

- 64 GB internal storage (extended by microSD)
- Adreno 330 GPU
- Hexagon DSP
- Multimedia processor

- 13 MP front camera
- 2 MP back camera
- Accelerometer
- Gyroscope
- Proximity sensor
- Compass
- Barometer
- Temperature sensor
- Humidity sensor
- Gesture Sensor
- GPS

41

Operating systems are everywhere

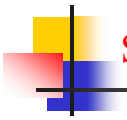



Operating systems are everywhere









Summary of lecture

- What is an operating system?
- Early Operating Systems
- Simple Batch Systems
- Multiprogrammed Batch Systems
- Time-sharing Systems
- Personal Computer Systems
- Parallel and Distributed Systems
- Real-time Systems