Chapter 3: Operating Systems

Computer Science: An Overview
Tenth Edition

by
J. Glenn Brookshear

Review

- What is an Instruction?
- What is stored program?
- What is a machine cycle?
- How does a machine cycle implement stored program concept?
- ALU and CU
- Two special-purpose registers

Do you know them?

- Android
- iOS
- DOS
- BSD
- Unix
- Solaris
- Chrome OS

OS for Personal Computers

- Linux
- Microsoft Windows
- Apple macOS
- BSD
- Unix

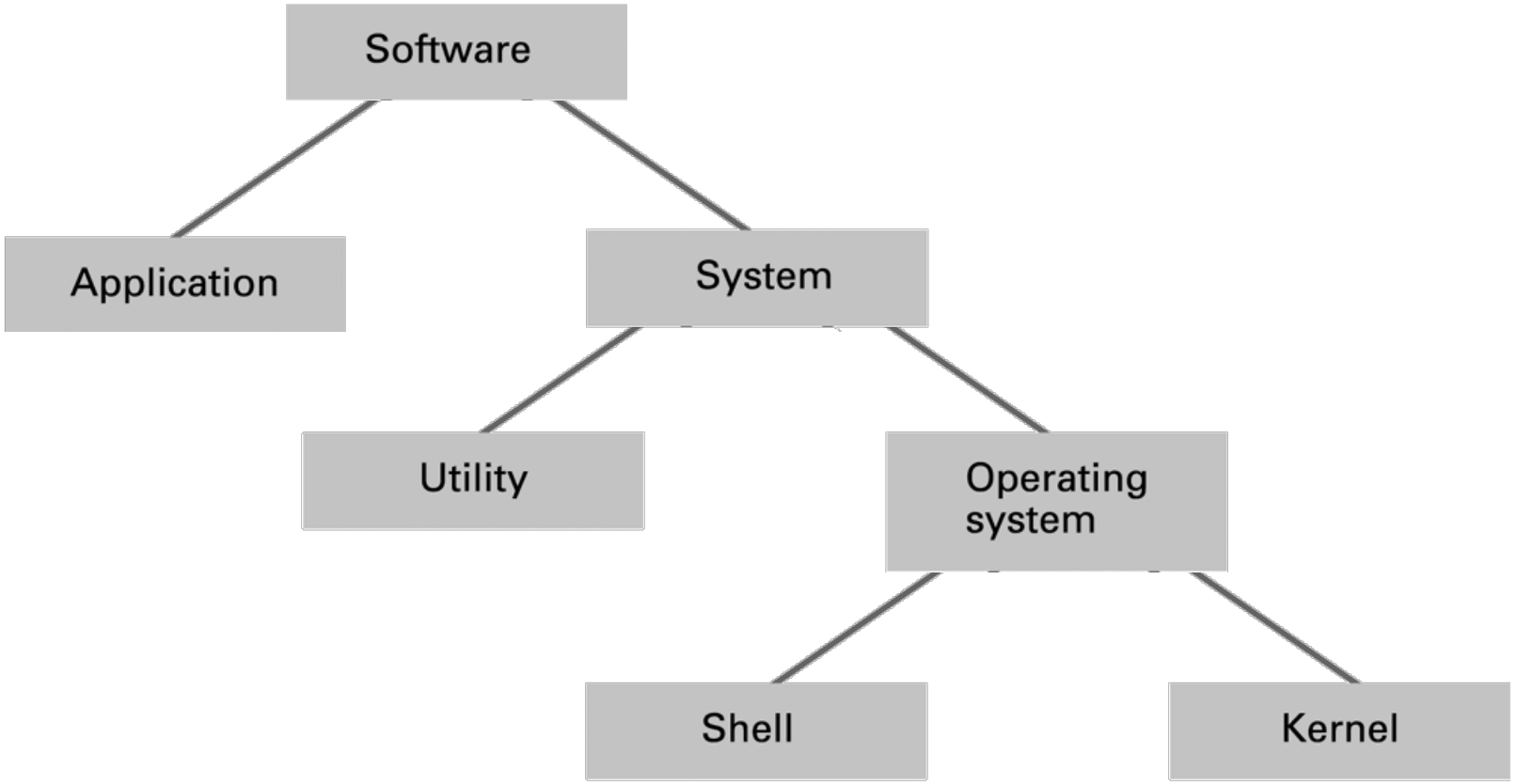
Embedded OS

- Linux
- iOS
- Android
- WindowsCE

OS for Small Devices

- Embedded systems, PDA, mp3 player, cell phone, GPS,…
  - Limited storage, limited power,
  - Usually has real time requirement
- Turnkey system: store all programs and data in a persistent memory
  - No BIOS and program loader

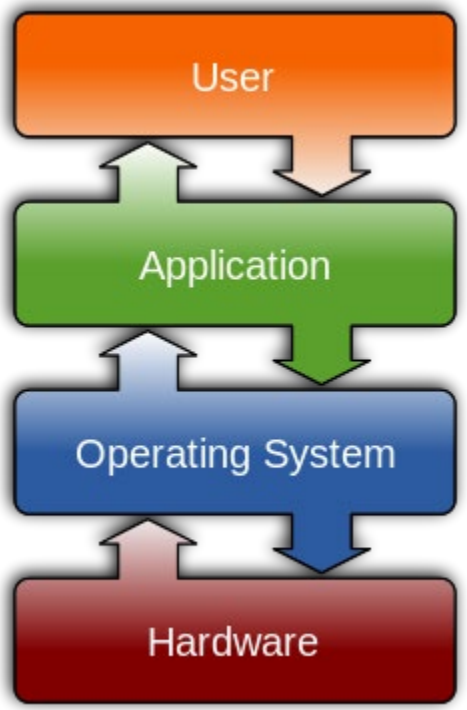
Software Classification



- Operating system is one kind of software

Operating System Architecture

- An operating system (OS) is software that manages computer hardware and software resources and provides common services for computer programs.

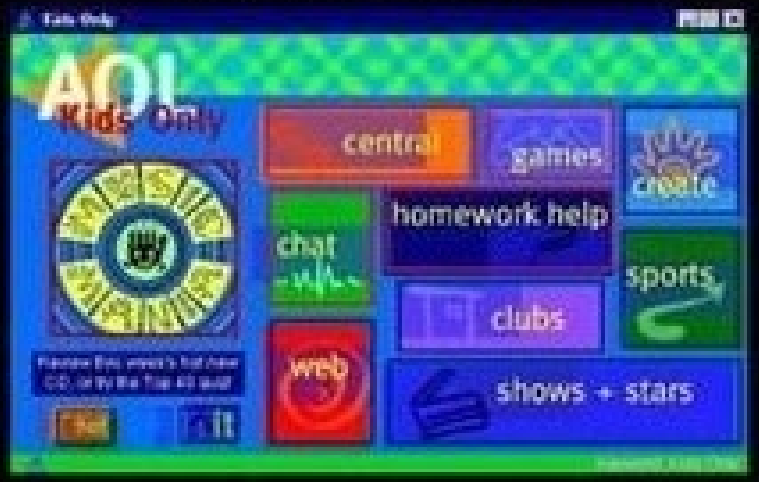


Functions of Operating Systems

- Oversee operation of computer
- Store and retrieve files
- Schedule programs for execution
- Coordinate the execution of programs

Windows (from 1985-present)

AOL 1996



Windows 8 2012

HISTORY

Those that fail to learn from history, are doomed to repeat it.

Mac OS (from 1984-present)

Linux

- First released on 5 October 1991 by Linus Torvalds

Some facts of Linux

- As of June 2013, more than 95% of the world's 500 fastest supercomputers run some variant of Linux
- During the second quarter of 2013, 79.3% of smartphones sold worldwide used Android
- 60% of Web servers ran Linux versus 40% that ran Windows Server (in the year of 2008)
- The first major film produced on Linux servers was 1997's Titanic.
- The estimated market share of Linux on desktop is 1.61%

GNU

- a free software, mass collaboration project, announced on 27 September 1983, by Richard Stallman at MIT.

UNIX

- Dates back to the mid-1960s
- Developed at AT&T's Bell Labs research center
  - The Unix Operating System, AT&T Tech Channel

How does a computer start executing?

Simple Answer

- The program counter is initiated with a particular address in a special memory when the computer is powered on
  - That address is start of a (special) program → to bring up other programs and the system
  - But DRAM is volatile!
  - So, we use read-only memory (ROM)

Getting it Started (Bootstrapping)

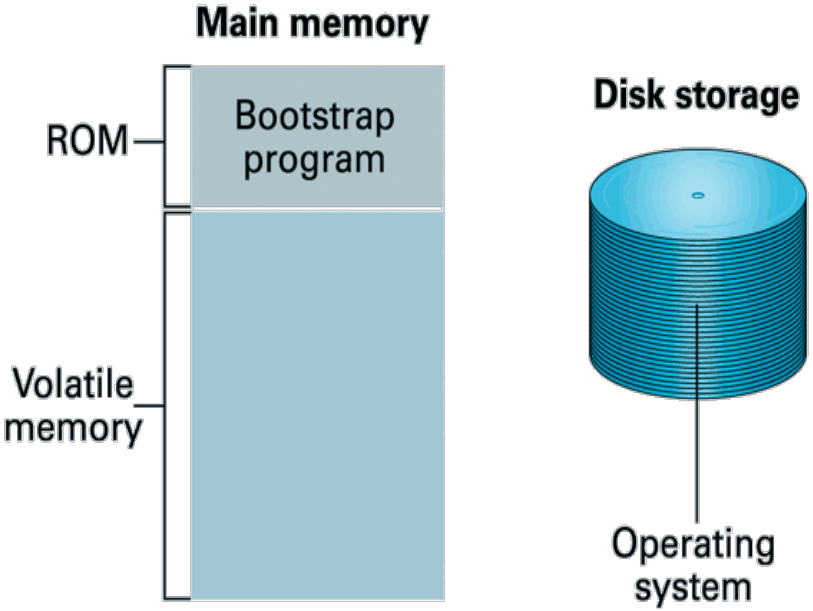
- **Bootstrap:** Program in ROM (example of firmware)
  - Run by the CPU when power is turned on
  - Transfers operating system from mass storage to main memory
  - Executes jump to operating system

The Booting Process

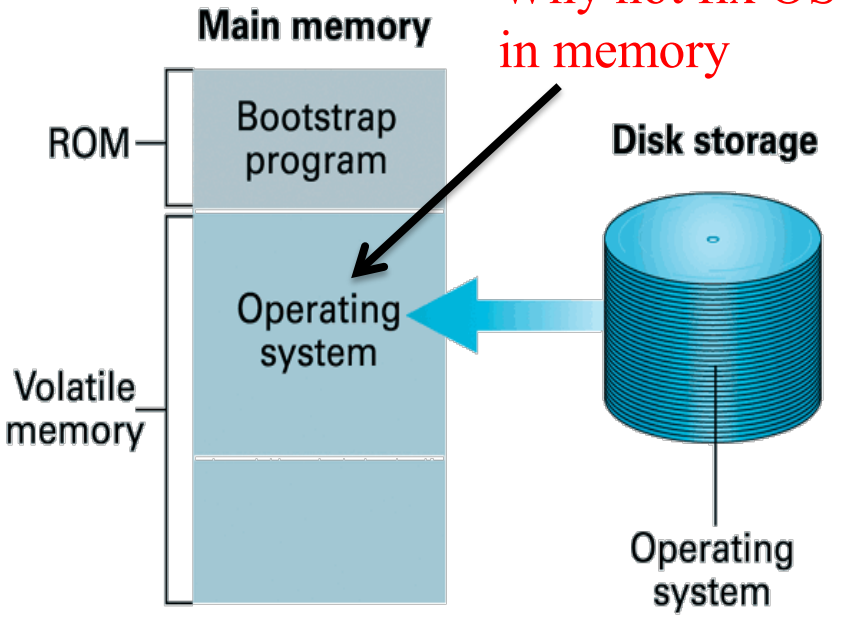
Turn key system

The program counter is initiated with a particular address in ROM where the bootstrap is stored

Why not fix OS in memory



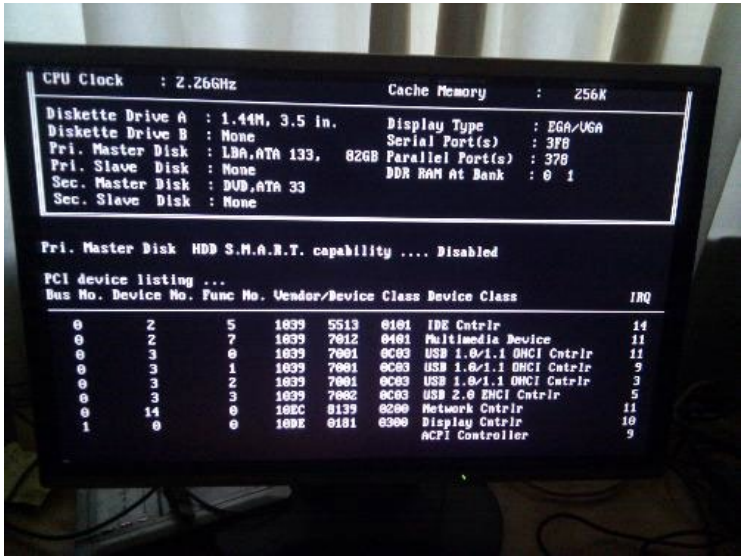
Step 1: Machine starts by executing the bootstrap program already in memory. Operating system is stored in mass storage.

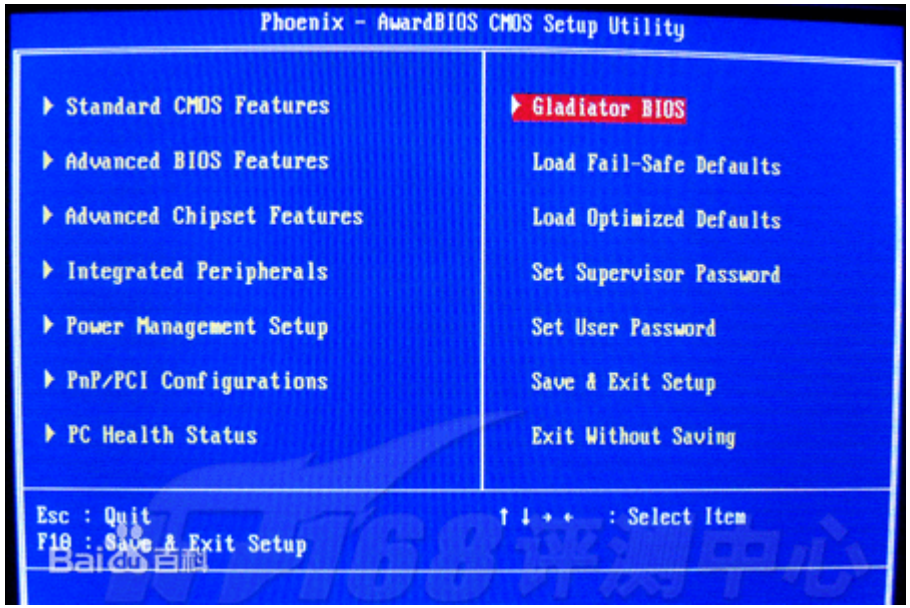


Step 2: Bootstrap program directs the transfer of the operating system into main memory and then transfers control to it.

BIOS

BIOS是英文"Basic Input Output System"的缩略语，直译过来后中文名称就是"基本输入输出系统"。其实，它是一组固化到计算机内主板上一个ROM芯片上的程序，它保存着计算机最重要的基本输入输出的程序、系统设置信息、开机后自检程序和系统自启动程序。 其主要功能是为计算机提供最底层的、最直接的硬件设置和控制





http://www.fitzenreiter.de/ata/bios-post_eng.html

- Questions?

Why do we need an operating system?

Suppose the computer runs only one program …

Full control of everything, e.g. CPU, memory, …

Manage everything

Suppose the computer runs many programs …

- How do they get executed?
- How do they get the most important resource?

Grabbing the CPU

- In a single-processor computer, only one CPU to be shared by all programs

- Who can get the microphone?

We Need a Chairperson!

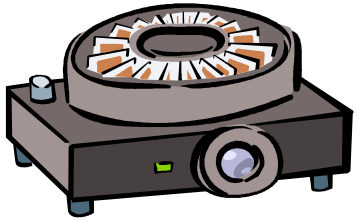
- The chairperson decides who gets the microphone to speak next
- Two ways to schedule:
  - Let each speaker talk until he/she finishes
  - “Interrupts” the speaker to get back the microphone and turn to another speaker

Chairperson Can Do More

- Which portion of blackboard a speaker can write?

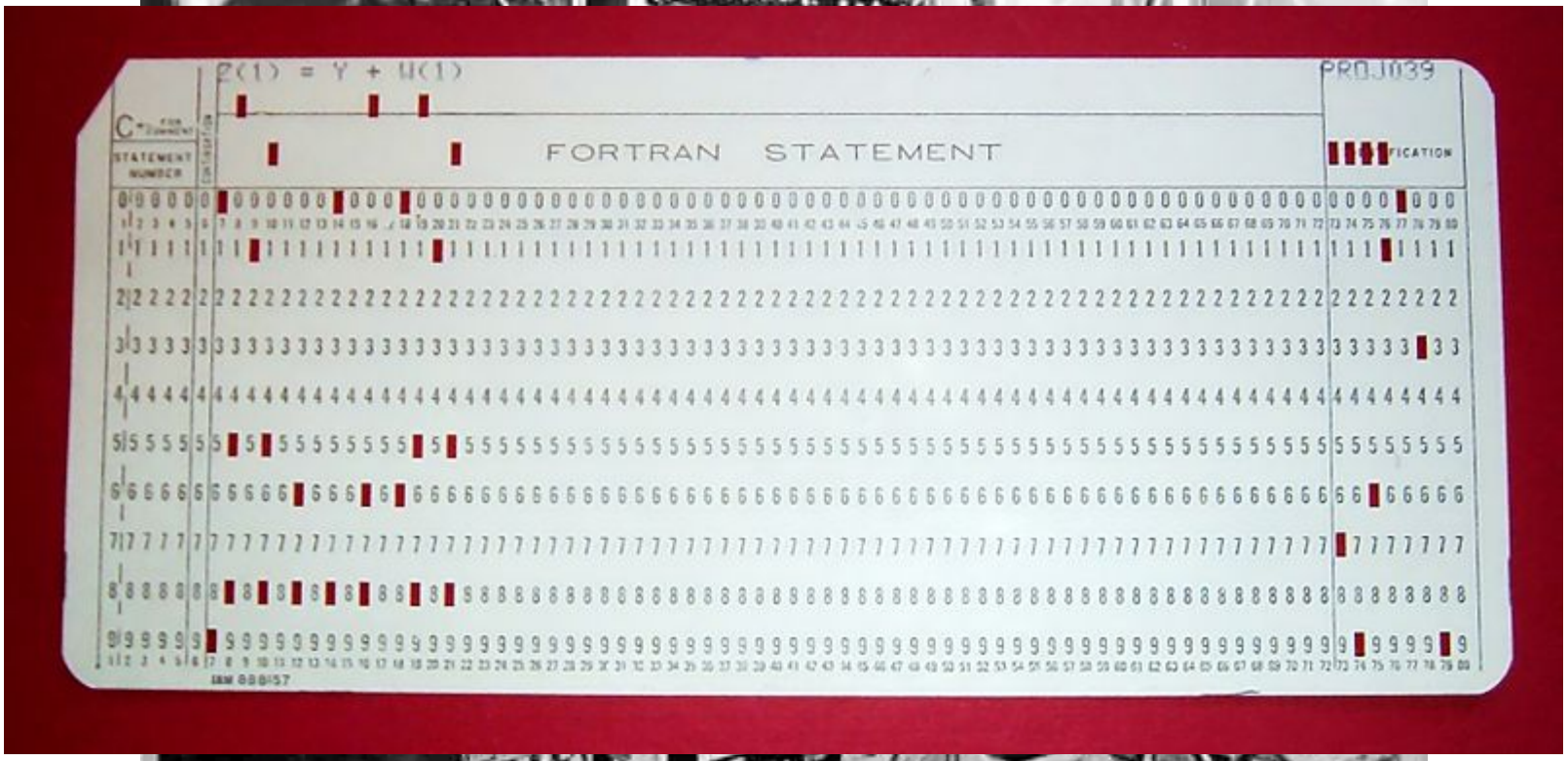
Chairperson is OS

- Who can use projector ?
  - Device management



The birth of OS

- Before OS was invented



Wikipedia

Jobs

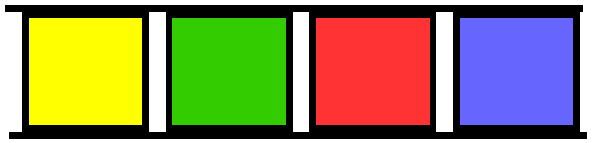
- Jobs queue

http://getlevelten.com/blog/ian-whitcomb/whats-wrong-project-application-queue

Queue

- FIFO

IN



OUT

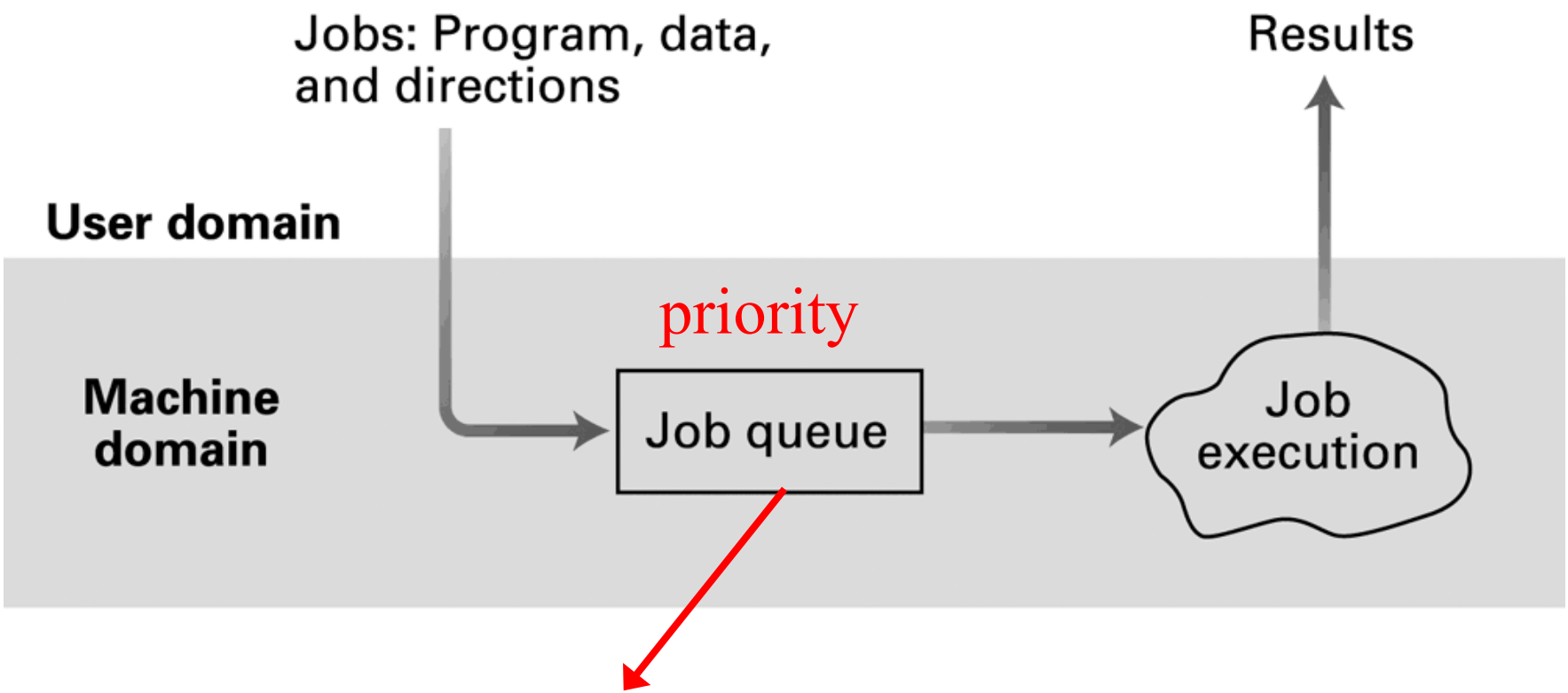
Jobs

- Jobs queue

Significant set-up time to run programs -> Need for Batch systems

http://getlevelten.com/blog/ian-whitcomb/whats-wrong-project-application-queue

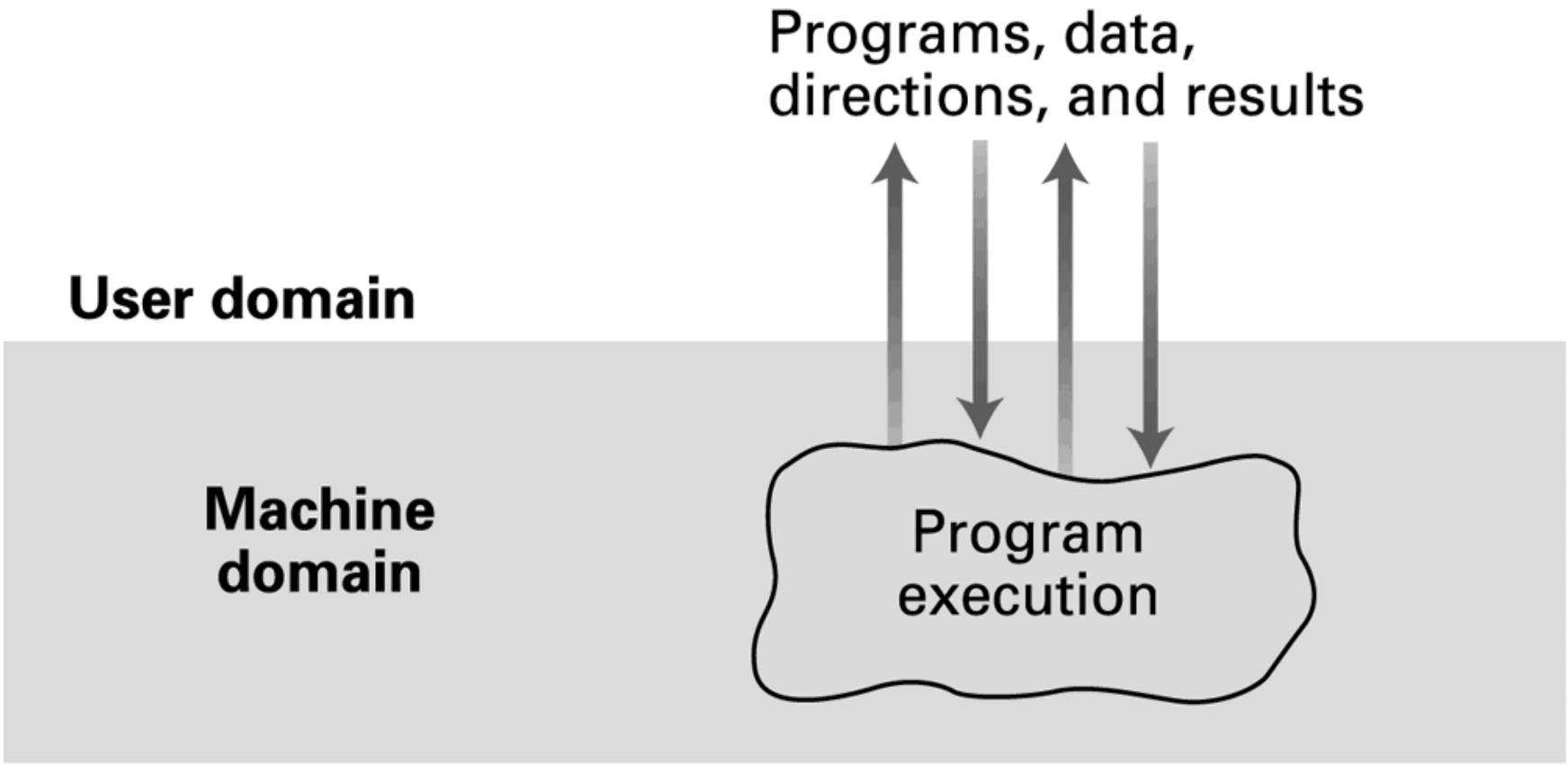
Batch processing



Resident Monitor

- Ancestor of OS (1950s)
- Remained resident in memory
  - Transfer control to the program
  - When the program ends, resident monitor resumes
  - Then transfer control to the next program
  - Still function is many embedded systems nowadays

Interactive processing



Text editing, music/movie playing, …

Computer terminal

Wikipedia

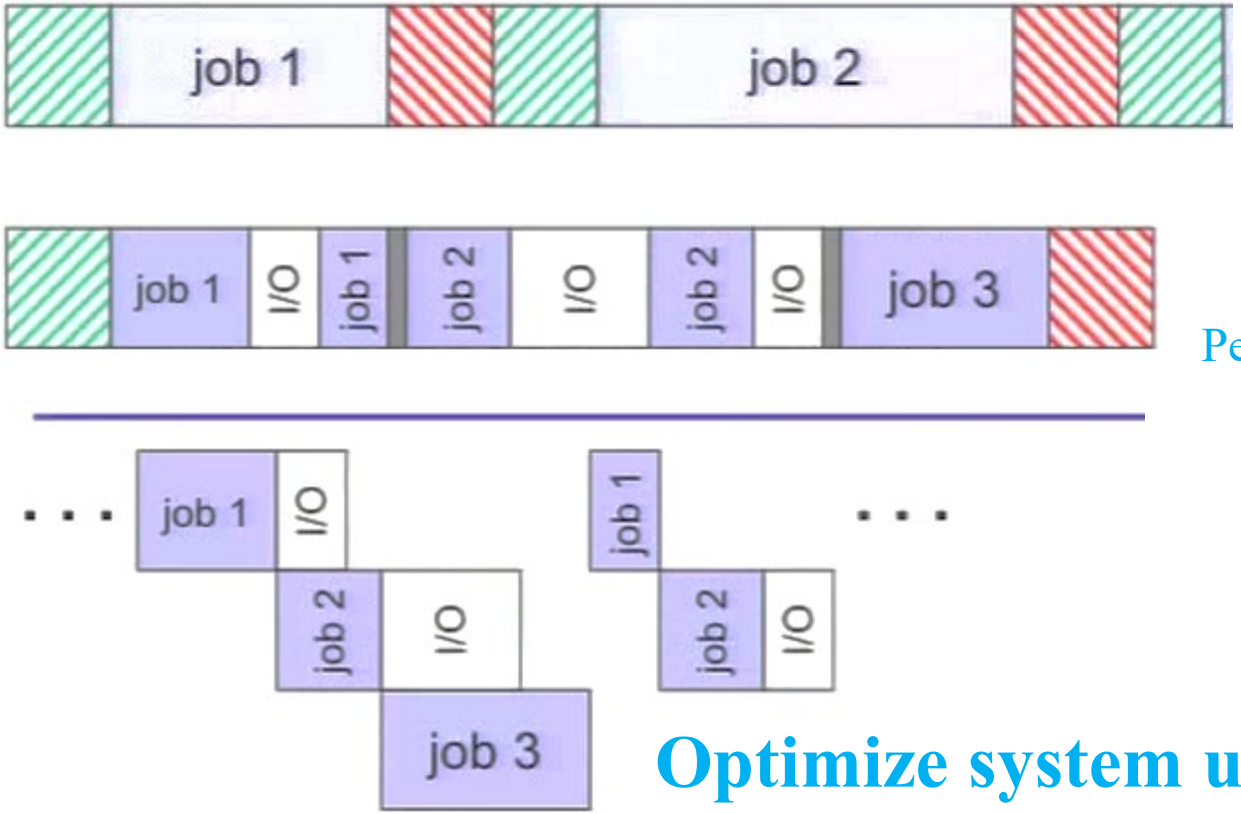
Real-time processing

Time-sharing system

- 1960s – 1970s
- Although batch systems allow system resources to be used more efficiently, the computer can only execute a program at a time and the user cannot interact with the job while it is executing

Multiprogramming (多道程序)

- The first example of a sophisticated OS (1960s)
- Early mode of multitasking

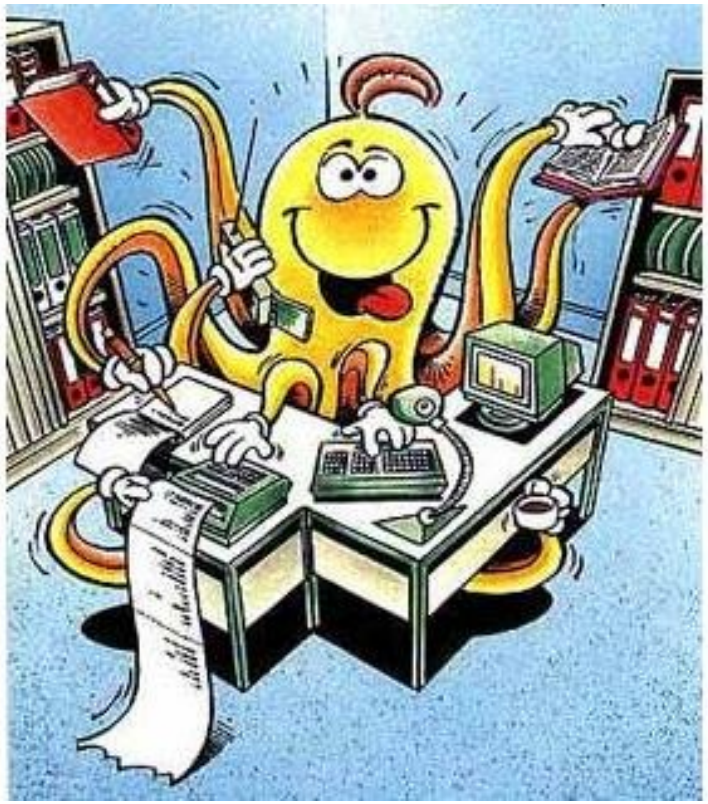


Peter Tröger

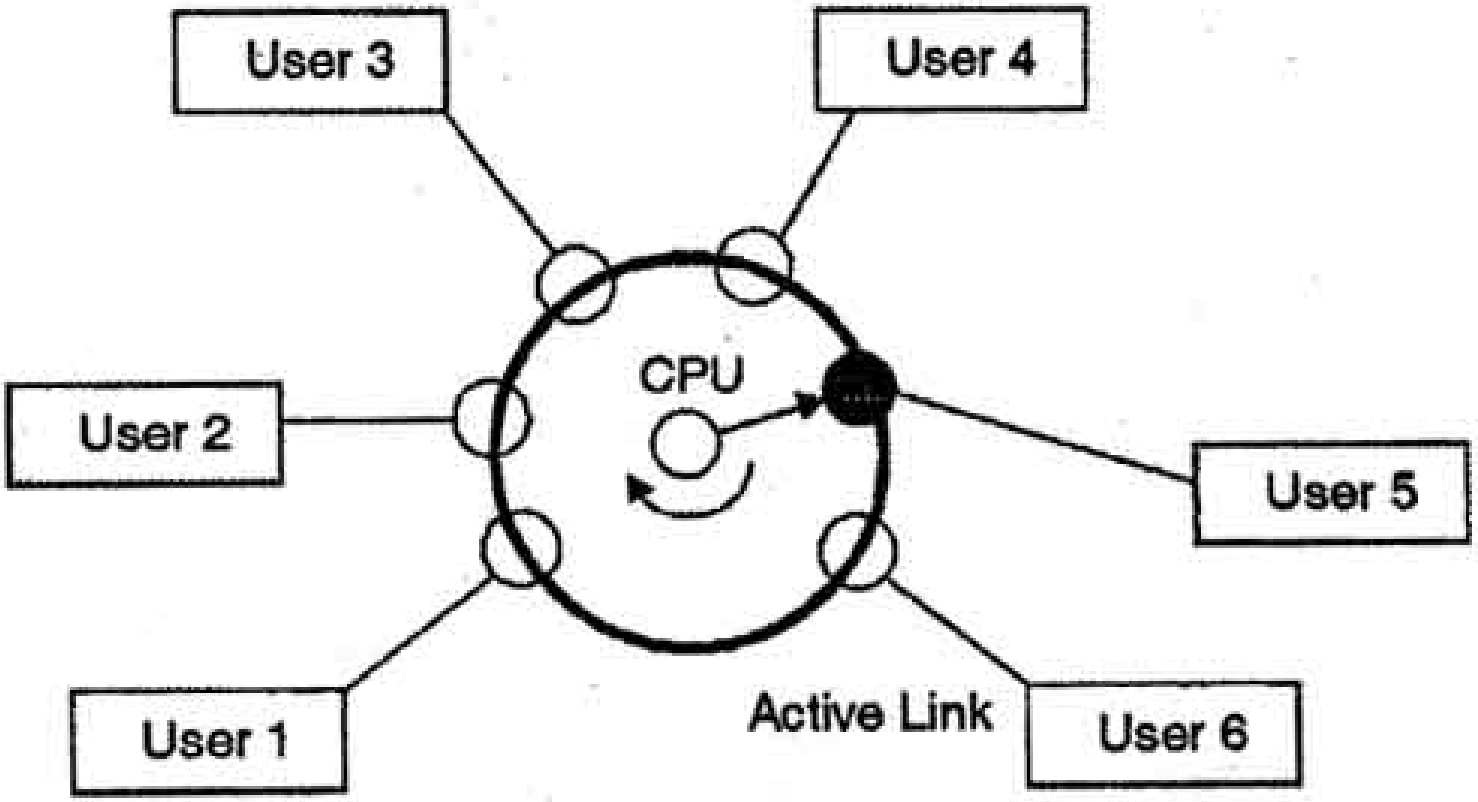
Optimize system utilization

Multi-tasking

- Multitasking: CPU switches between jobs frequently to allow user interaction



Optimize real time interaction



DINESH THAKUR

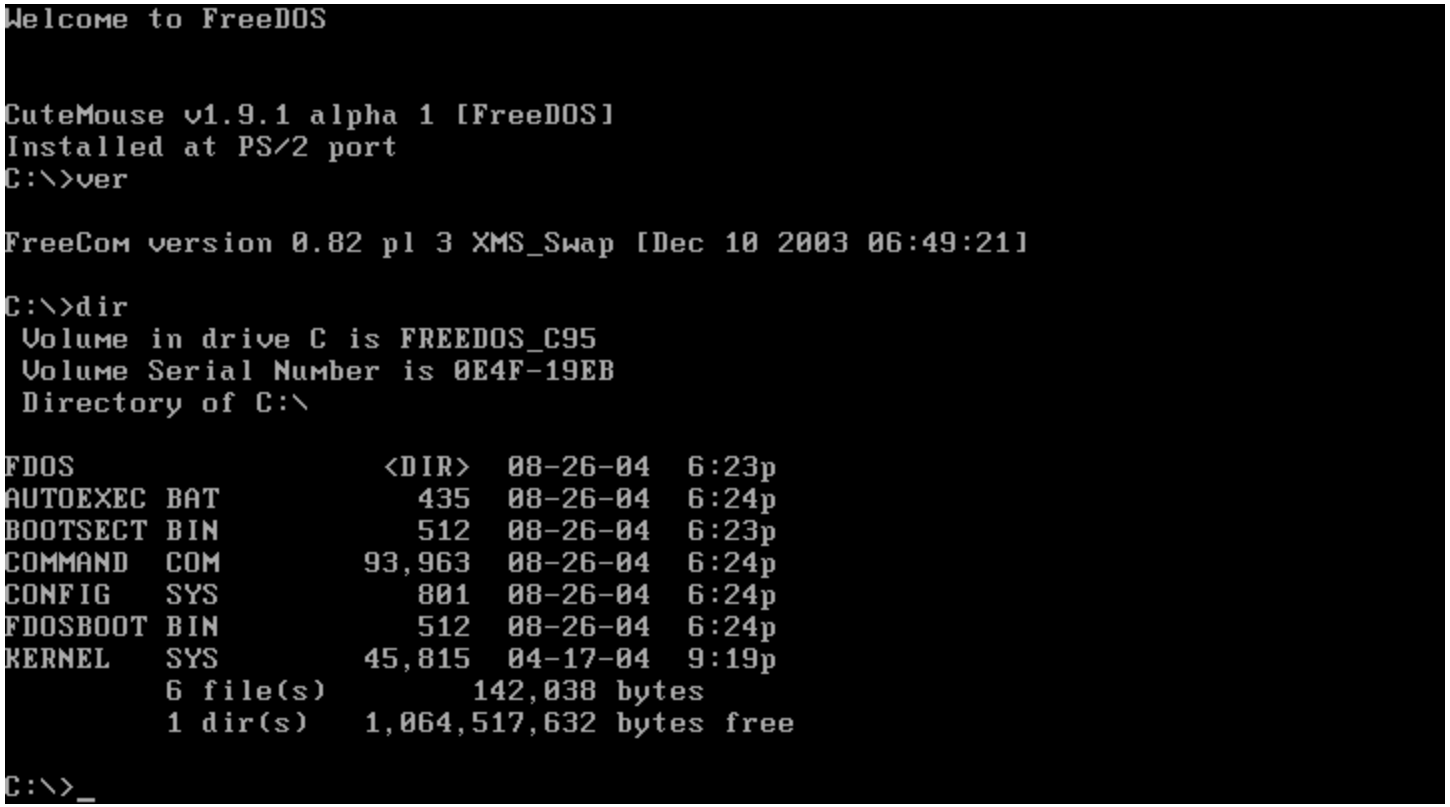
Evolution of Shared Computing

- Batch processing
- Interactive processing
  - Requires real-time processing
- Time-sharing
  - Multiprogramming
    - Execute a part of a program once
  - Multitasking
- Multiprocessor machines
- Grid/Cloud computing

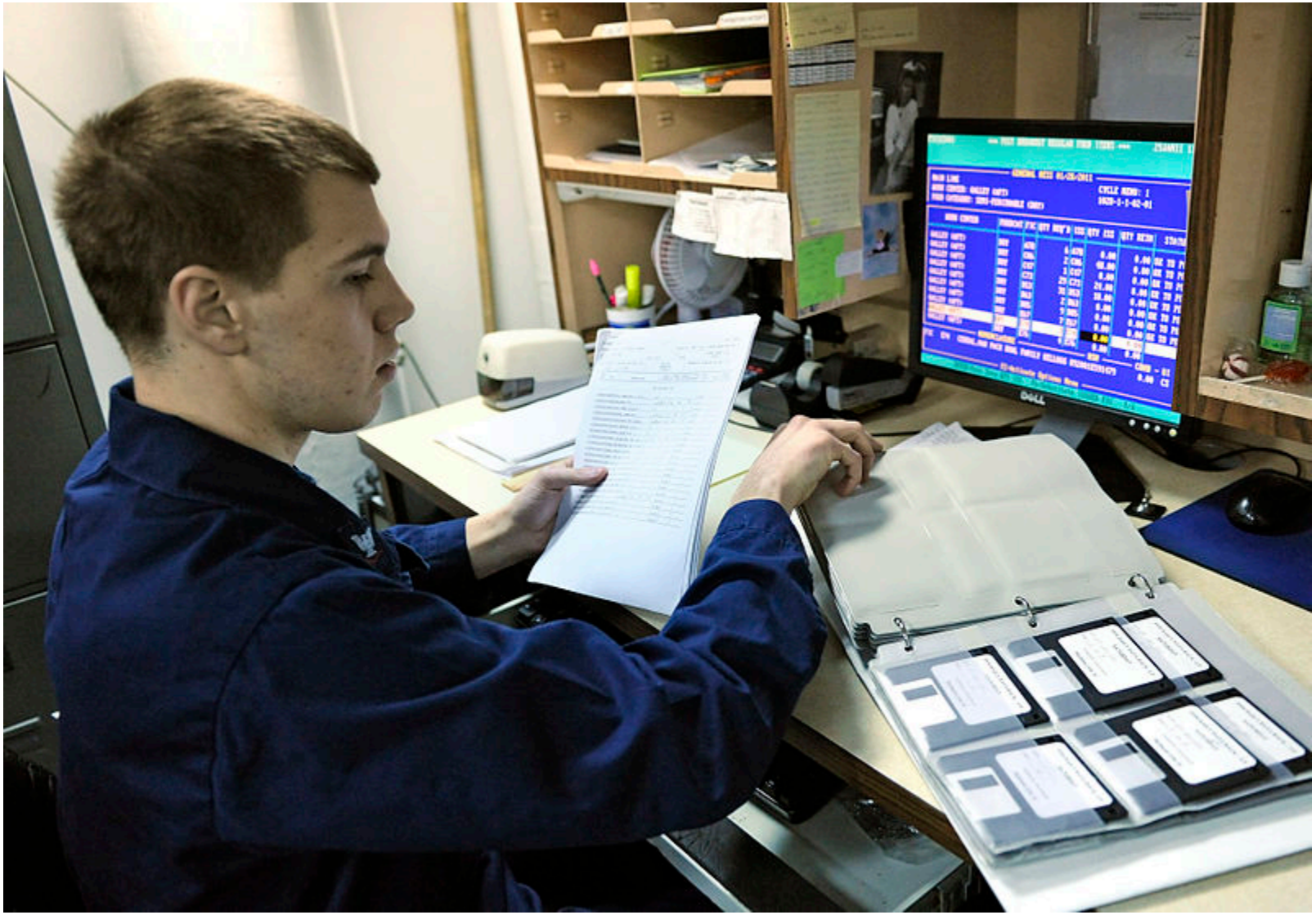
DOS (Disk Operation System)

- MS-DOS
- PC DOS

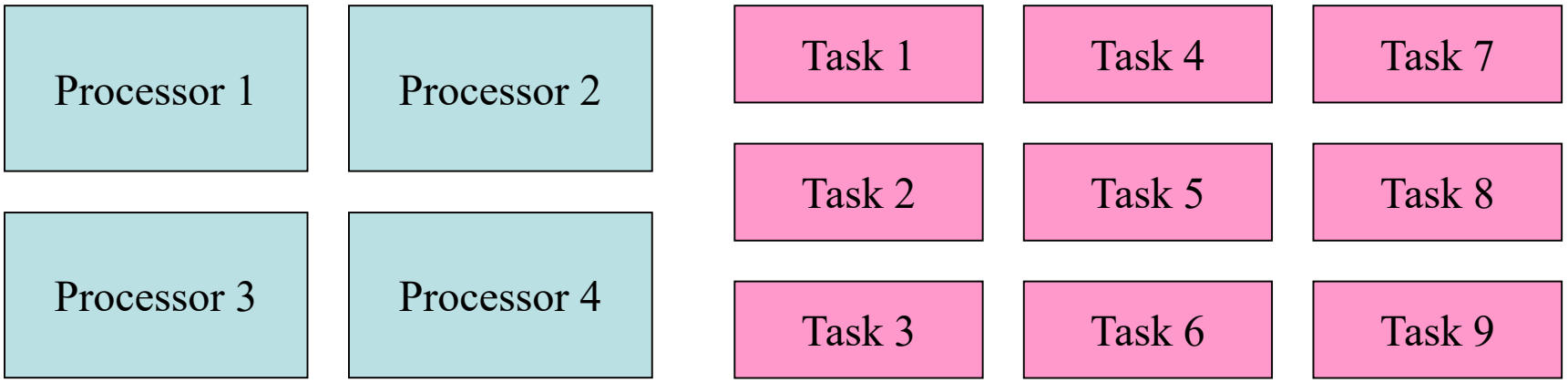
Mostly single-user single-task



US Navy 110129-N-7676W-152 Culinary Specialist 3rd Class John Smith uses the existing DOS-based food service management system aboard the aircraft (2011)



Multiprocessor Machines



- How to assign tasks to processors?
  - Load balance problem
- How to use processors to handle one task?
  - Parallelization, scaling problem

Side effects of user interaction

- .bat
  - @ECHO off
  - ECHO Hello World!
  - PAUSE
- BASH
- PowerShell
- Macro

- Questions?

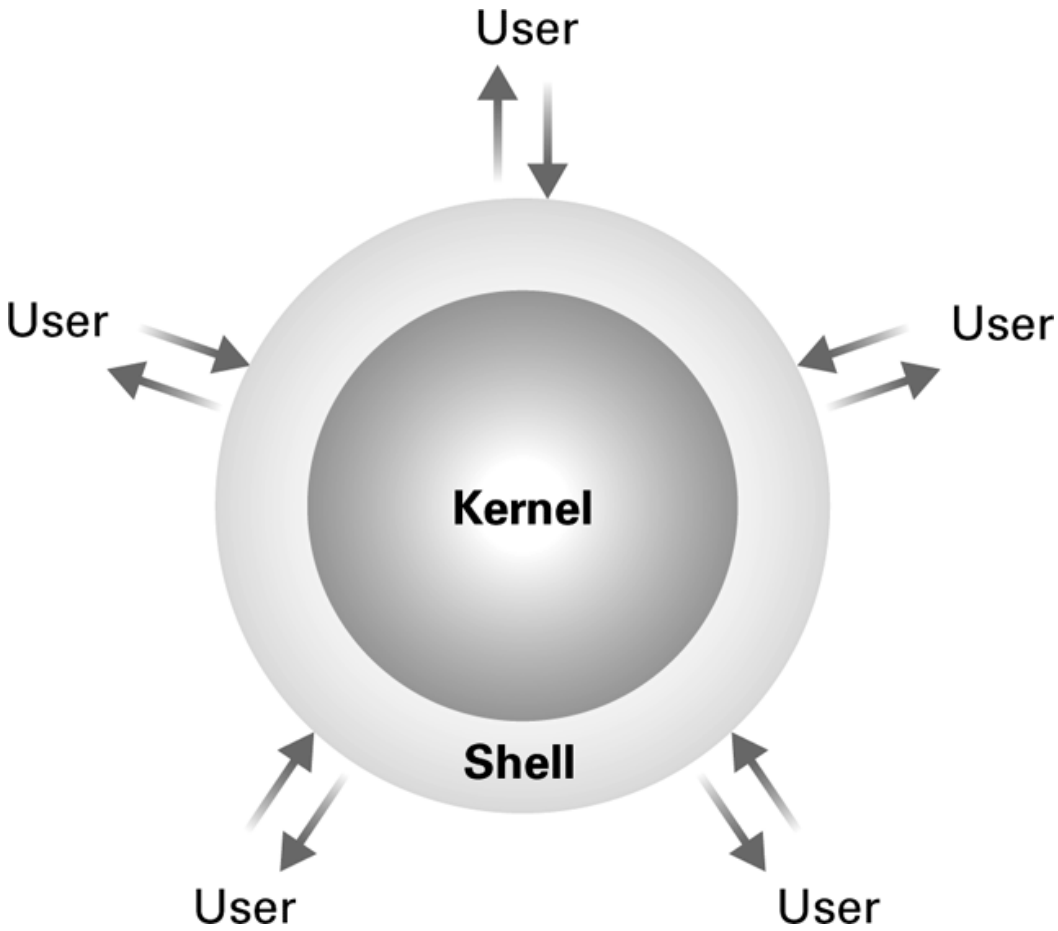
Components of OS

- For **user**: shell, privilege control (security)
- For **data**: file manager
- For **hardware**: device manager, memory manager, and boot manager
- For **software**:
  - Where to store: file manger, registry
  - How to execute: scheduler, process manager

Operating System Components

- **Shell:** Communicates with users
  - Command line interface (CLI)
  - Graphical user interface (GUI)
- **Kernel:** Performs basic required functions
  - File manager
  - Device drivers
  - Memory manager
  - Scheduler and dispatcher

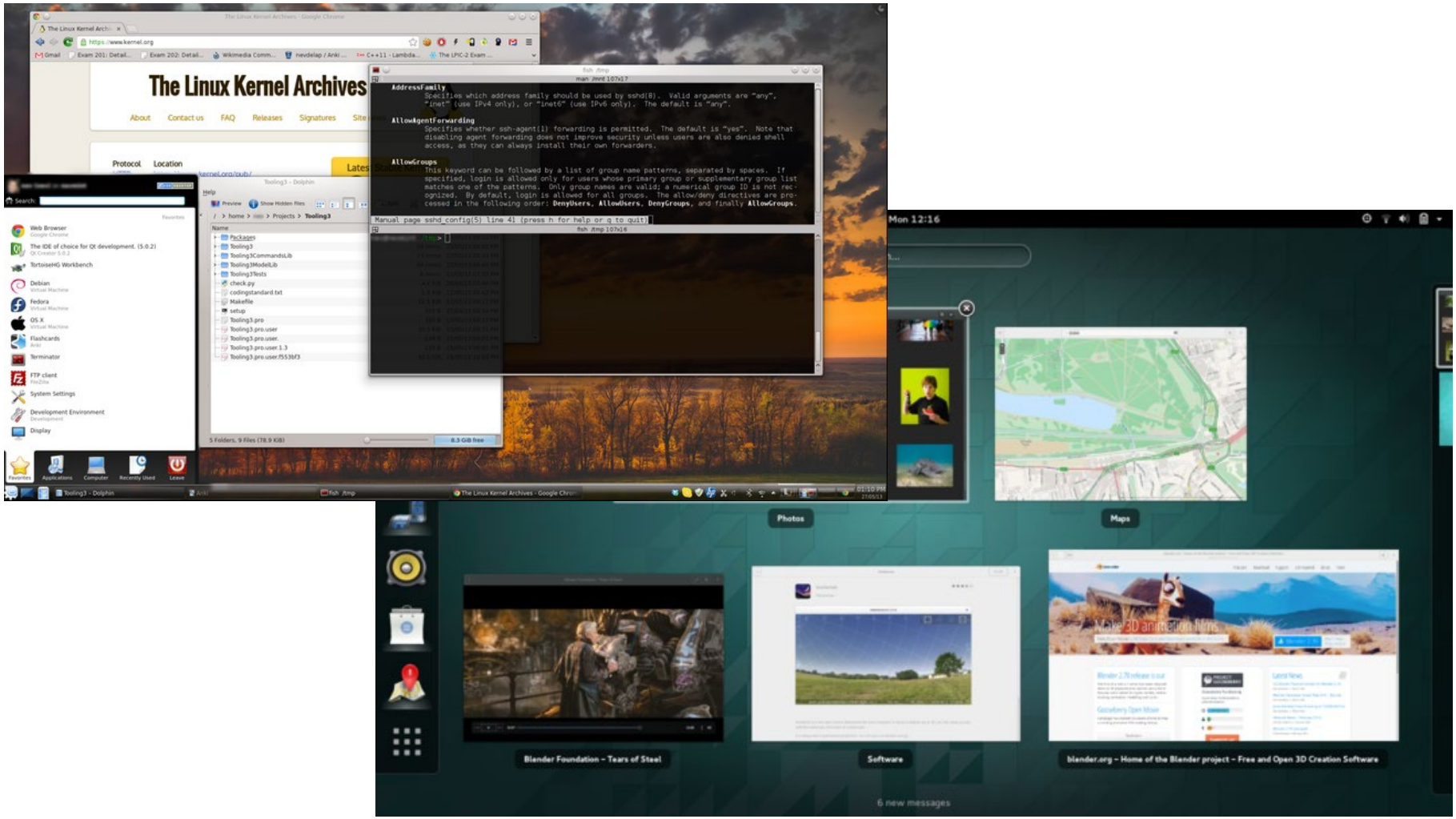
Figure 3.4 The shell as an interface between users and the operating system



CLI and GUI

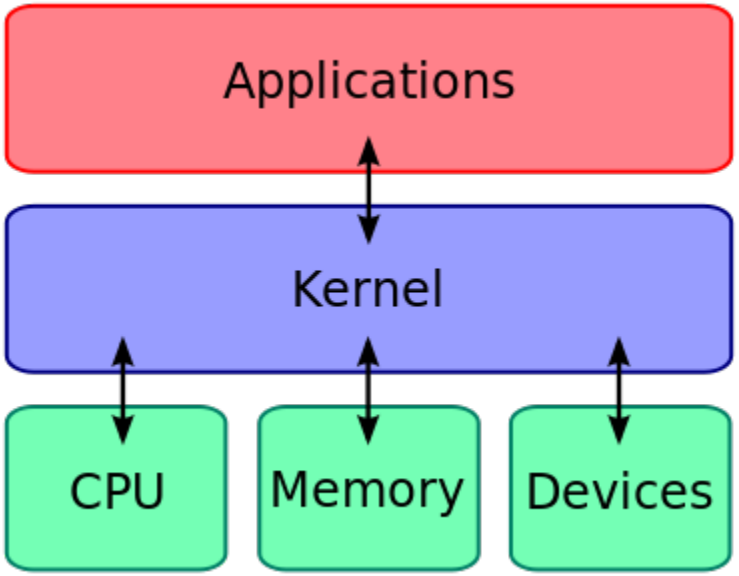
```
chealer@vinci:/usr/share/doc/bash$ export LC_ALL=C
chealer@vinci:/usr/share/doc/bash$ cd ~chealer/
chealer@vinci:~$ ls
Cloutier  Ido         Musique  logs      skolo  sources
Desktop   Mes images  boston   ncix.png  smb4k  vieux
chealer@vinci:~$ #Why is there color when calling ls without arguments?
chealer@vinci:~$ which ls
/bin/ls
chealer@vinci:~$ $(!!)
$(which ls)
Cloutier  Ido         Musique  logs      skolo  sources
Desktop   Mes images  boston   ncix.png  smb4k  vieux
chealer@vinci:~$ type ls #"ls" doesn't just run /bin/ls
ls is aliased to `ls --color=auto'
chealer@vinci:~$ echo $PS1
${debian_chroot:+($debian_chroot)}\u@\h:\w\$
chealer@vinci:~$ sh
sh-3.1$ echo $PS1
\s-\v\$
sh-3.1$ echo $BASH_VERSION
3.1.17(1)-release
sh-3.1$ ls
Cloutier  Ido         Musique  logs      skolo  sources
Desktop   Mes images  boston   ncix.png  smb4k  vieux
sh-3.1$ echo $SHELLOPTS # ls isn't an alias in POSIX mode
braceexpand:emacs:hashall:histexpand:history:interactive-comments:monitor:posix
sh-3.1$ kill
kill: usage: kill [-s sigspec | -n signum | -sigspec] pid | jobspec ... or kill
-l [sigspec]
sh-3.1$ /bin/kill &> killerror # collect stdout and stderr of $ /bin/kill; in ki
llerror
sh-3.1$ wc -l !$
wc -l killerror
7 killerror
sh-3.1$ type kill # kill doesn't just run /bin/kill, even in POSIX mode.
kill is a shell builtin
sh-3.1$ !$ -n 9 $$ # OK, kill self
kill -n 9 $$ # OK, kill self
Killed
chealer@vinci:~$
```

e.g. KDE and GNOME



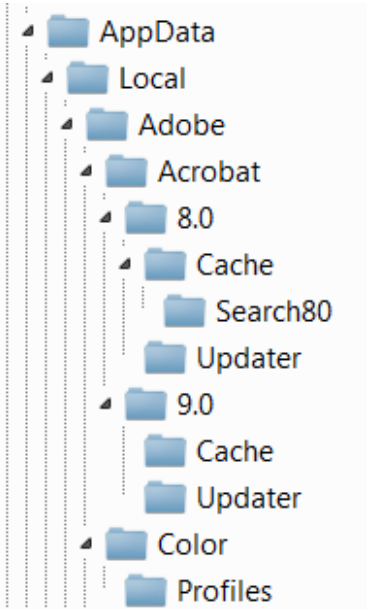
Kernel

- manages input/output requests from software, and translates them into data processing instructions for the central processing unit and other electronic components of a computer.
- The kernel is a fundamental part of a modern computer's operating system



File Manager

- **Directory** (or **Folder**): A user-created bundle of files and other directories (subdirectories)
- **Directory Path:** A sequence of directories within directories



Device Driver

File management system

- FAT FAT32 NTFS
- EXT3 EXT4 ZFS Btrfs

Memory Manager

- Suppose computer runs many programs at the same time
  - What if programs are larger than the memory?
  - Which program uses which part of the memory?
  - How to protect them from each other?

Memory Manager

- Allocates space in main memory
- May create the illusion that the machine has more memory than it actually does (**virtual memory**) by playing a “shell game” in which blocks of data (**pages**) are shifted back and forth between main memory and mass storage
- How?
  - Store only needed portion in memory and the remaining in disks
  - Shuffle portions between memory and disks
  - Program uses virtual address, while OS does the mapping
- ***Paging***: memory is grouped into *pages* to facilitate the mapping and shuffling

- Questions?

Review

- Which of the following facts about Operating System are true?
  - A servant that provides computing facilities and environment
  - A boss that controls all the hardware and software
  - Without OS, the computer will not function
  - OS is much more complex than any other software
  - OS takes most of the CPU resources

Review

- Batch processing vs Time-sharing
  - Enable interaction
  - Less interactive
  - Suitable for processing many similar jobs
  - Suitable for concurrent different jobs

Review

- Multiprogramming vs Multi-tasking
  - Maximum the CPU usage
  - Maximum the interaction
  - Time-sharing system

Review

- Shell vs Kernel
  - Enable program execution
  - Enable user interaction with OS

Fundamentals of OS

- Assuming you are allocated one and only one task
- What will you do?

Fundamentals of OS

- Assuming you are allocated a number of tasks
- What will you do?
  - None of them is urgent
  - Some are urgent while others are not

轻重缓急！

Fundamentals of OS

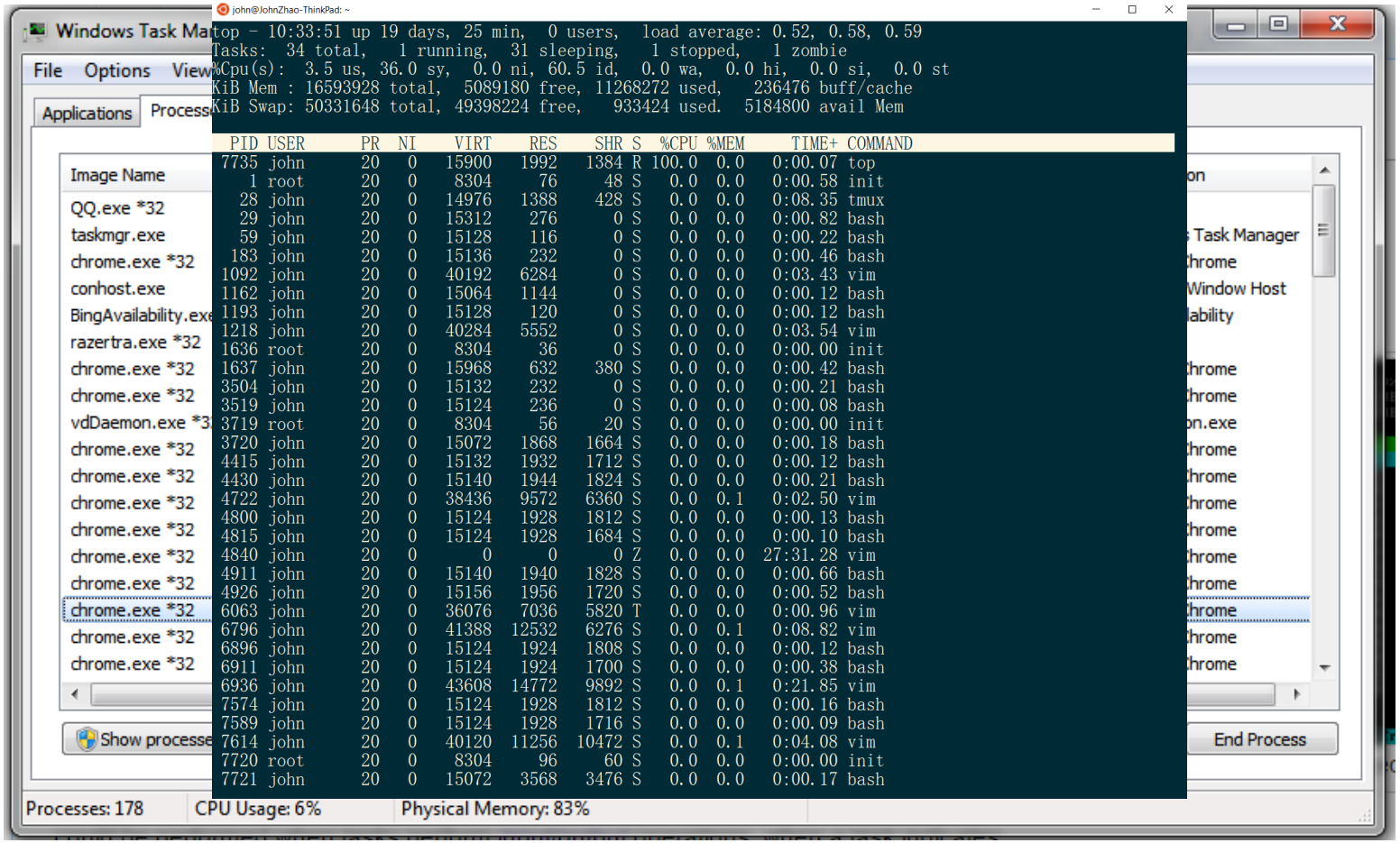
- Conclusion
  - Single task -> Uniprogramming
  - A batch of tasks -> Batch processing
    - How to optimize? Multiprogramming
    - 轻重缓急 -> Priority
- Finally, if all of the tasks are urgent and need timely response
- What will you do?

Fundamentals of OS

- The problem of concurrent execution of multi-tasks
  - Time-sharing system
  - 宏观上并行，微观上串行

Process

- What is a process



Process

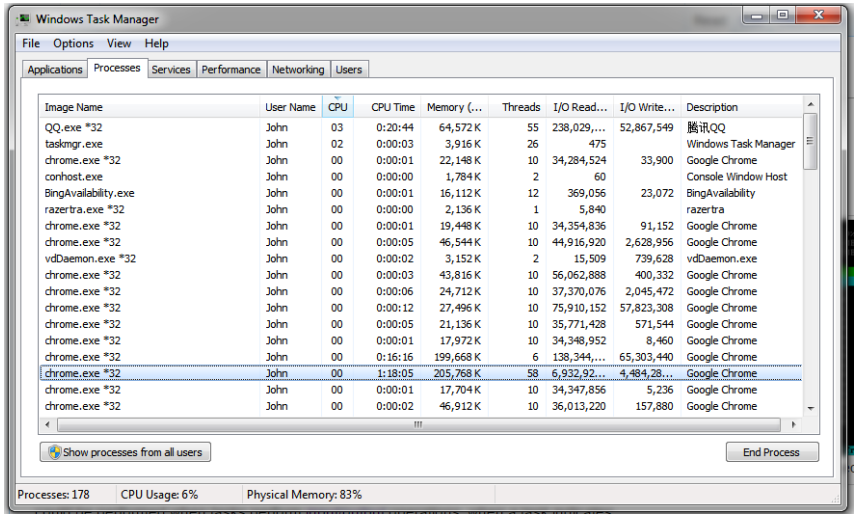
- Each process represents a task
- Jobs vs Tasks
  - A collection of tasks that is used to perform a computation is known as a *job* (MSDN)
- Is a process the same as a program?

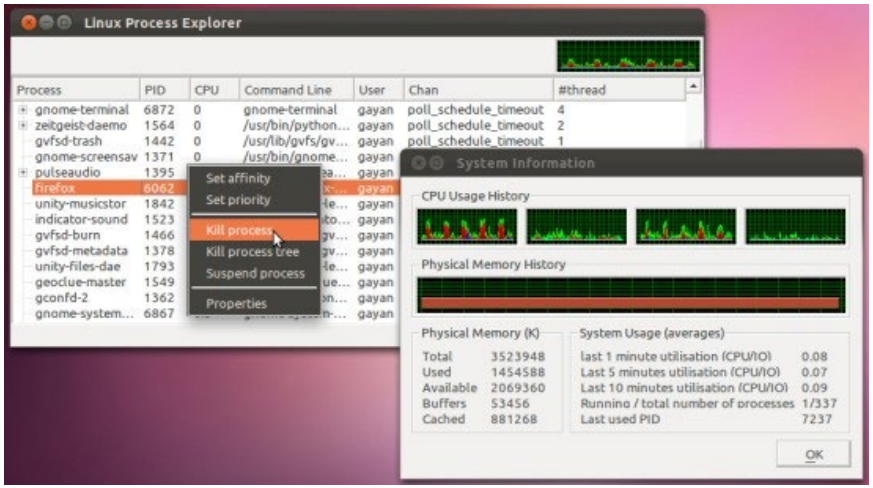
A Program vs. a Process

- Program: a set of instructions, e.g., notepad.c, notepad.exe
- Process: activity of executing a program
- A program can be run multiple times, each instance/activity called a process

Process table

- A table of all the processes maintained by the operating system.
- Each entry is a process and its descriptions (PCB).





Process control block (PCB)

- "the manifestation of a process in an operating system"
  - Process identification
  - Processor state data (the status of a process, saved registers, program counter etc.)
  - Process control data (scheduling state, privileges etc.)

The secret of concurrent execution

- What do you have to do when switching from one ongoing task to another?
  - Simply stop the current task and turn to another
  - Record the status of the current task and suspend it and turn to another
- Context switch

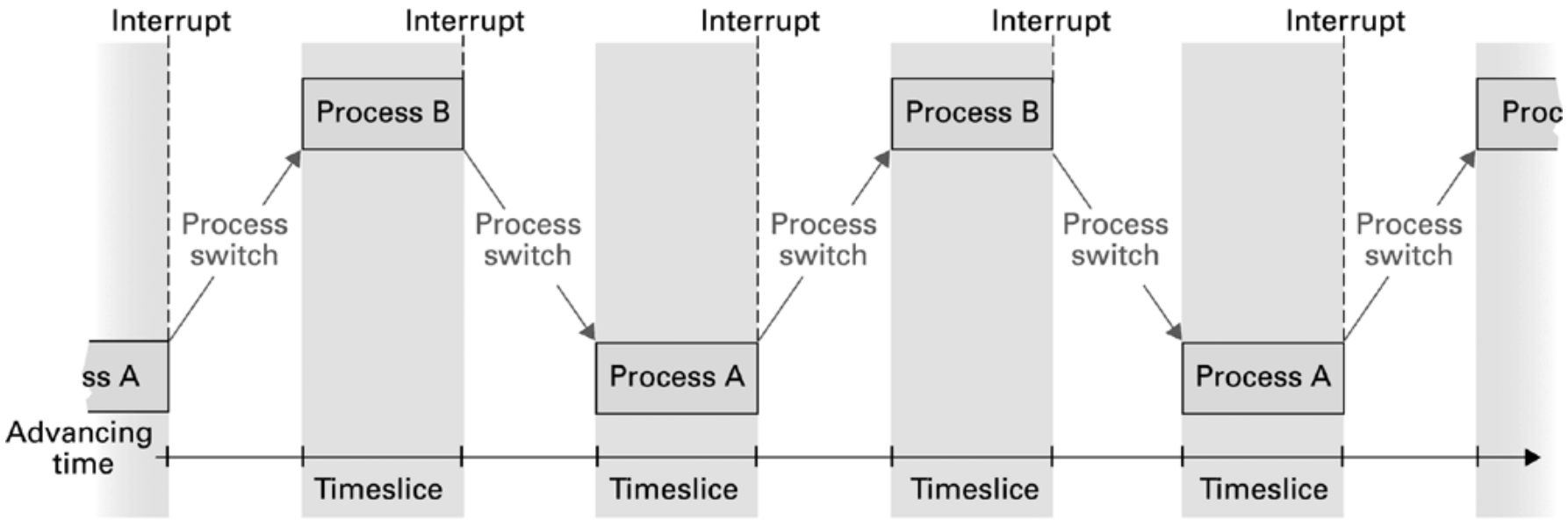
What is a context?

- Snapshot of current status of a process (PCB)
  - A process identifier, or PID
  - Register values, Program Counter value
  - The memory space, I/O, files for the process
  - Can be saved and resumed as if the process is not interrupted
- Another meaning: execution *state* of the process
  - Ready: ready for execution
  - Waiting: waiting for some I/O
  - Complete: finished process

Context switch

- The process of storing and restoring the state (context) of a process so that execution can be resumed from the same point at a later time.
- This enables multiple processes to share a single CPU and is an essential feature of a multitasking operating system.

Figure 3.6 Time-sharing between process A and process B



Who is responsible for context switching?

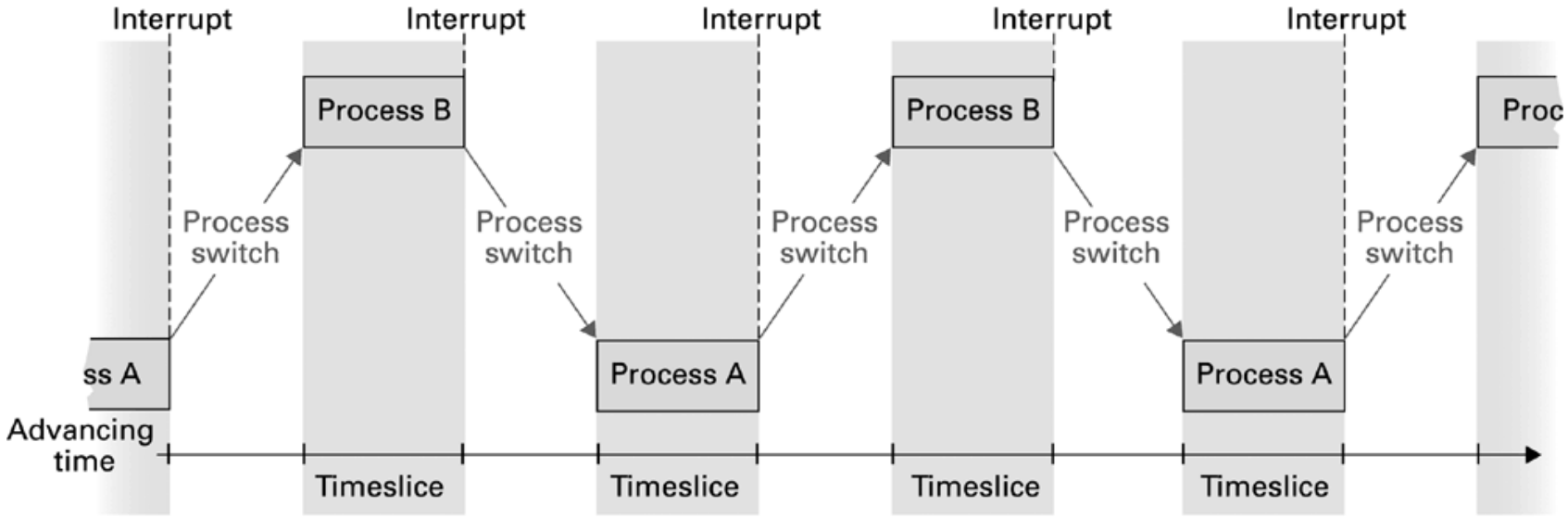
- **Process management**
  - **Scheduler (调度):** Adds new processes to the process table and removes completed processes from the process table
  - **Dispatcher (分派):** Controls the allocation of time slices to the processes in the process table
  - The OS's main responsibility
    - Scheduler provides the policy
    - Dispatcher provides the mechanism

Scheduler

- Determines which processes should be considered for execution based on some priorities or concerns
  - Using process table for administration
- Process table
  - Process state
  - Priority
  - Non-scheduling information: memory pages, etc.

Dispatcher

- Gives time slices to a process that is ready
- Executes a context switch when the running process's time slice is over
  - *Time slice*: a time segment for each execution

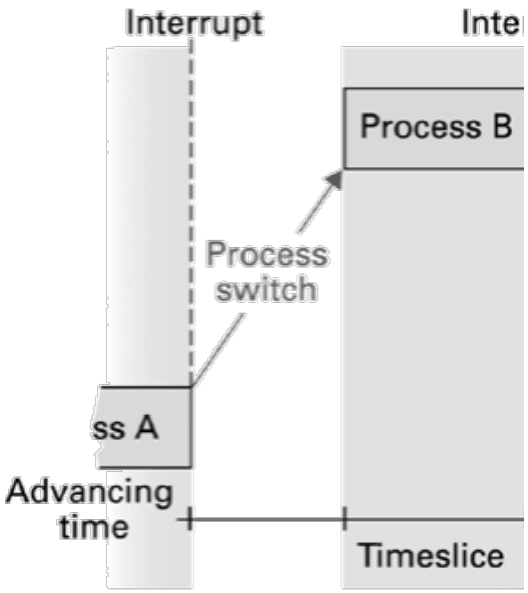


When to switch?

- **Interrupt** a signal to the processor emitted by hardware or software indicating an event that needs immediate attention.
- The *interrupt handler* (part of dispatcher) starts after the interrupt to perform context switch
- Modern architectures are interrupt driven
  - Software interrupt (I/O)
  - Hardware interrupt (press a key)

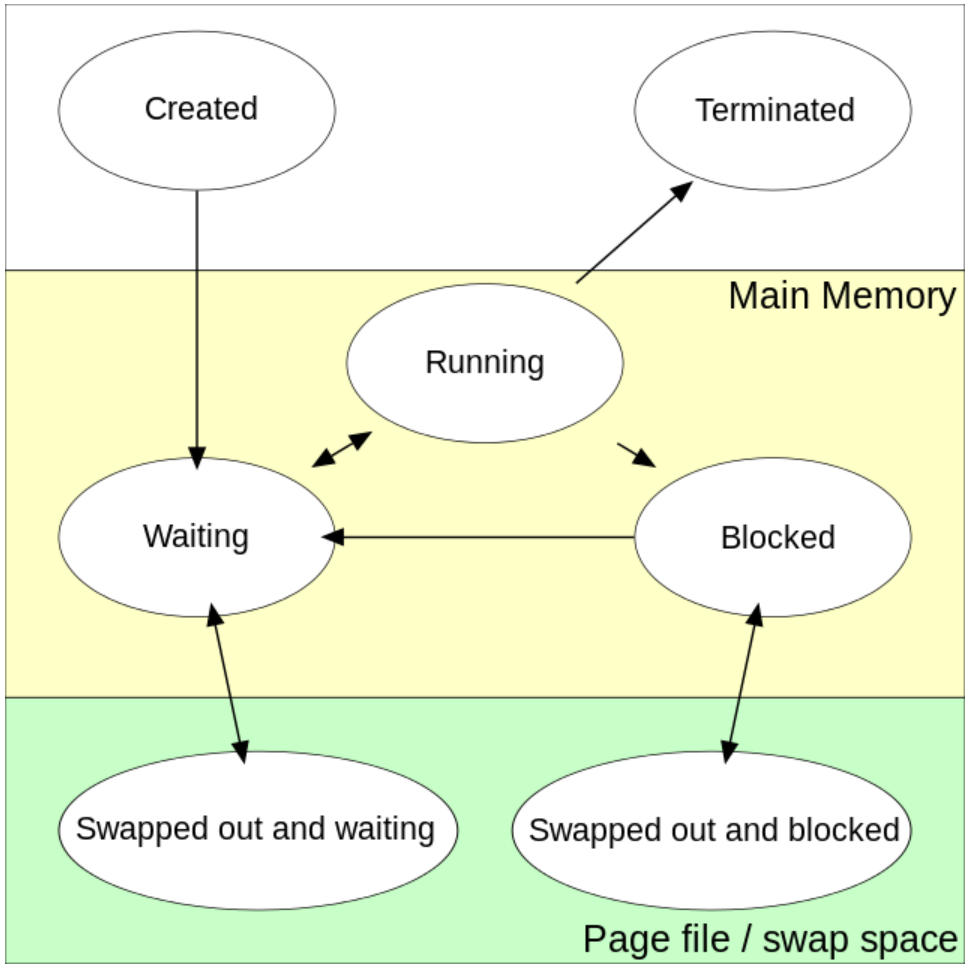
Context Switch (process switch)

1. Get an interrupt (from timer)
2. Go to the interrupt handler
   a. Save the context of process A
   b. Find a process ready to run (Assume that is process B)
   c. Load the context of process B
3. Start (continue) process B



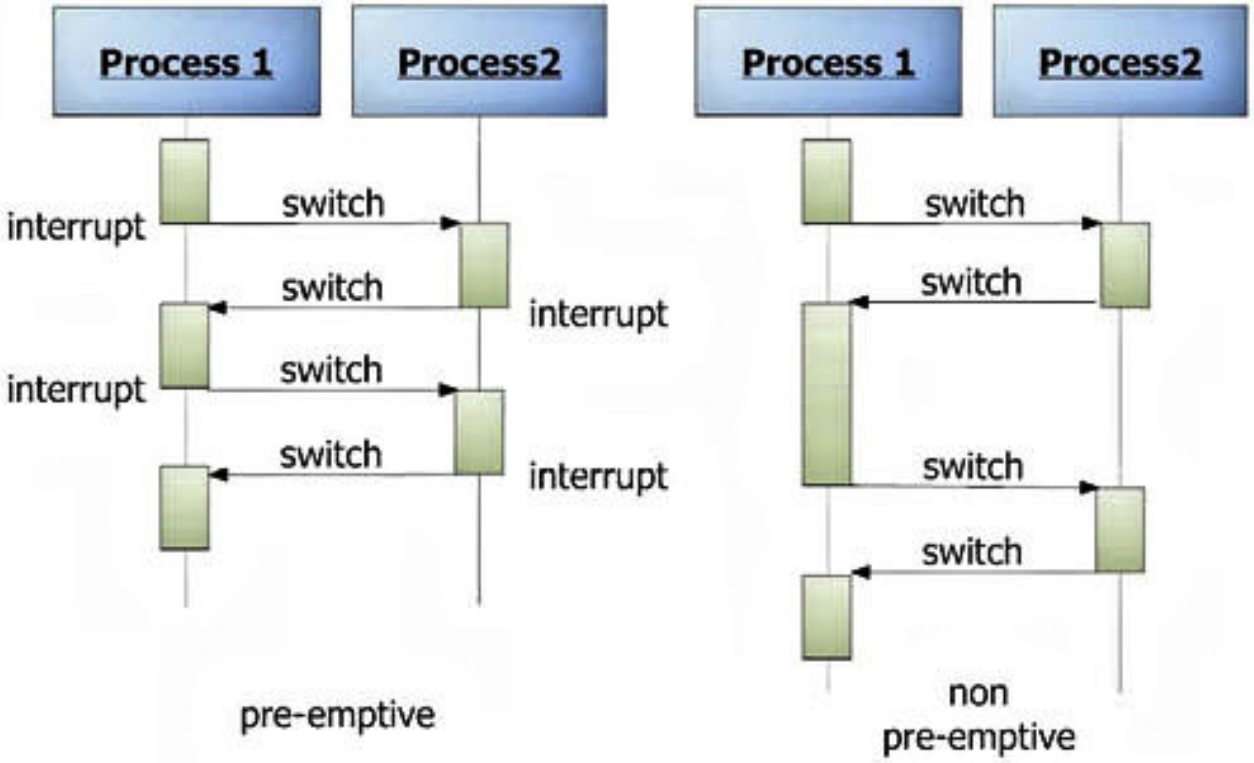
The process state

- Don’t be confused with PCB
- Running
- Blocked
- Ready/waiting



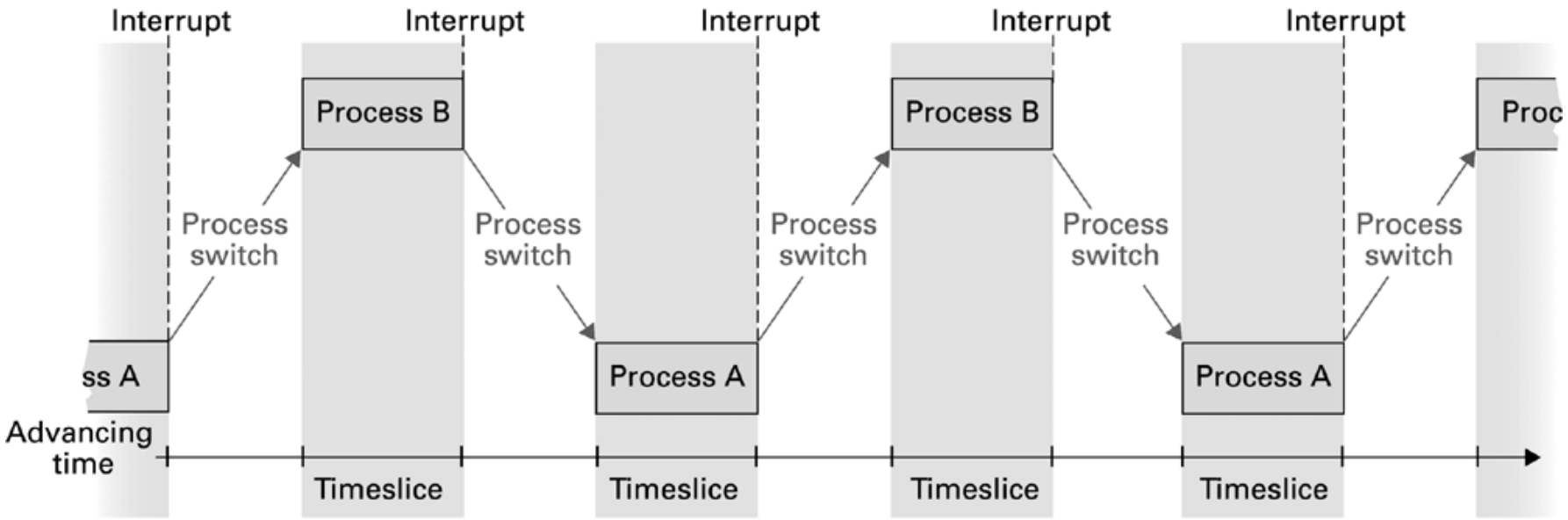
(Wikipedia)

Multi-tasking: Preemptive vs Cooperative



http://sqljunkieshare.files.wordpress.com/2012/01/bergerrtlinuxfig1.jpg

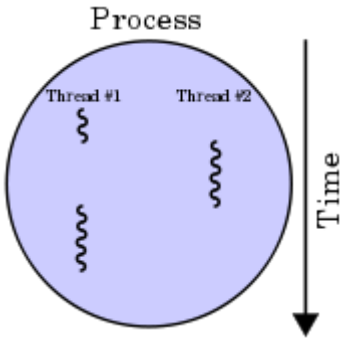
Figure 3.6 Time-sharing between process A and process B



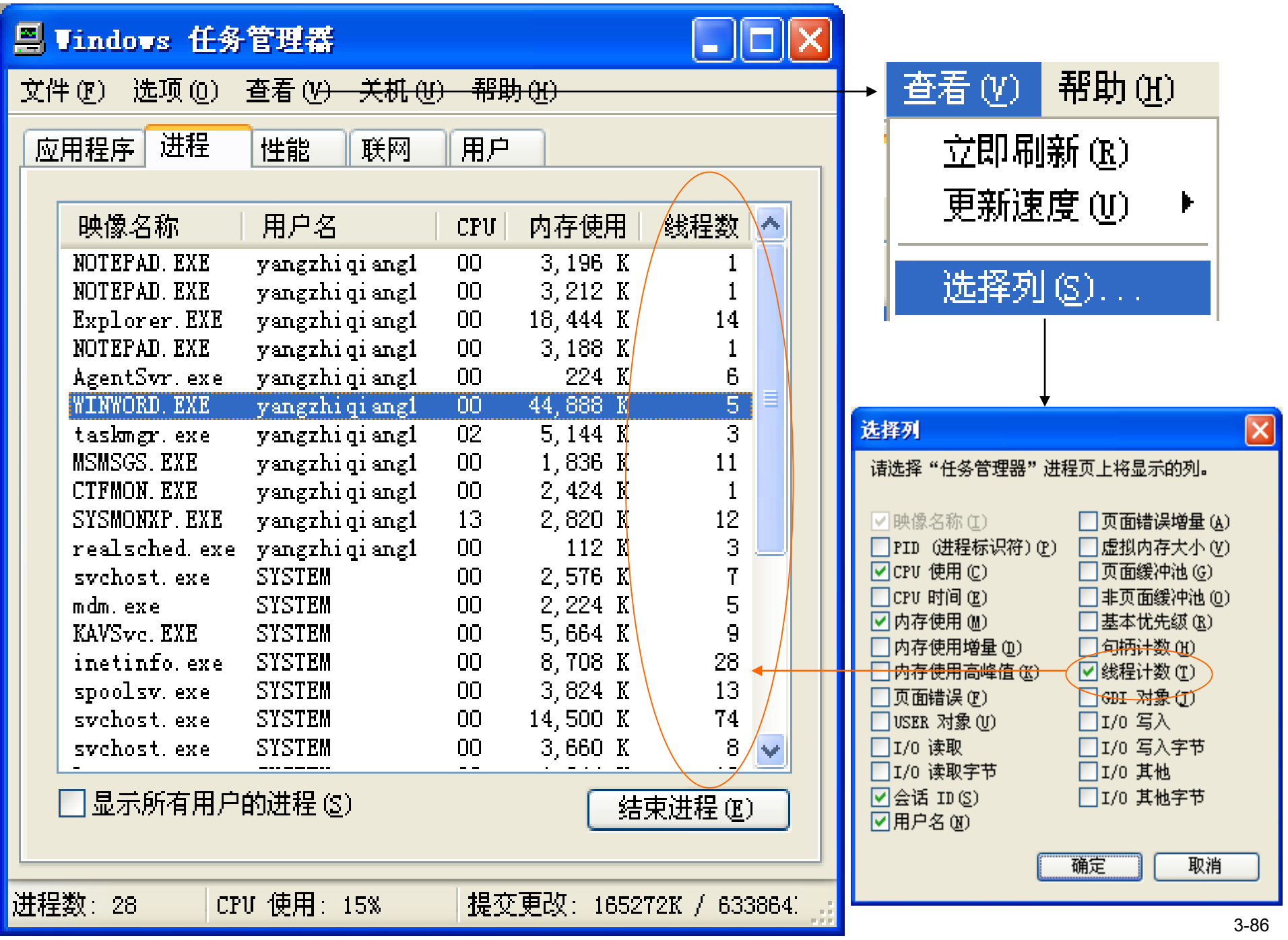
Exercises

- Suppose an OS allocates time slices in 10 millisecond units and the time required for a context switch is negligible. How many processes can obtain a time slice in one second?

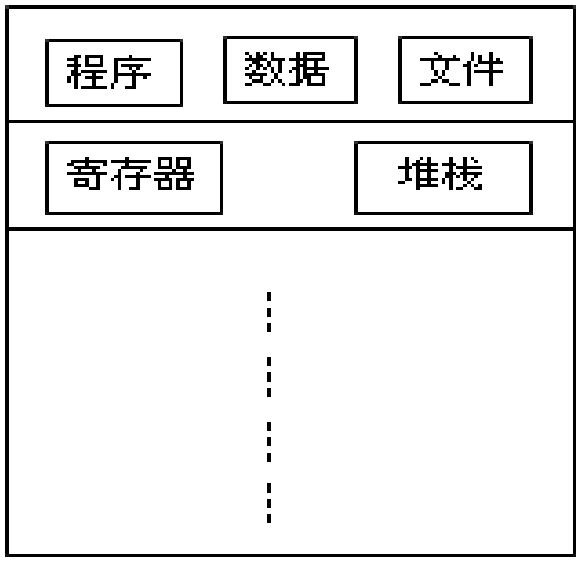
Thread



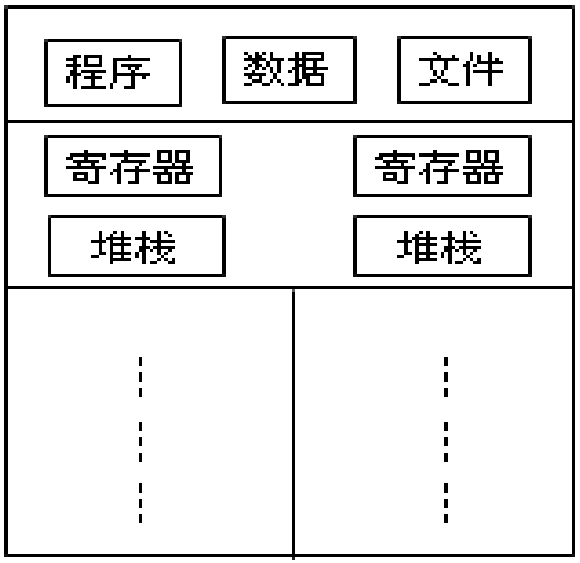
- A subprocess existing within a process that allows multiple independent instances to be executed concurrently
  - Multiple threads share resources such as memory, program code, …
  - Each thread has its own program counter, registers, and stack (local memory)
- The context switch of threads is much faster than that of processes


Windows 任务管理器
文件(F) 选项(O) 查看(V) 关机(U) 帮助(H)
应用程序 进程 性能 联网 用户
映像名称 用户名 CPU 内存使用 线程数
NOTEPAD.EXE yangzhiqiang1 00 3,196 K 1
NOTEPAD.EXE yangzhiqiang1 00 3,212 K 1
Explorer.EXE yangzhiqiang1 00 18,444 K 14
NOTEPAD.EXE yangzhiqiang1 00 3,188 K 1
AgentSvr.exe yangzhiqiang1 00 224 K 6
WINWORD.EXE yangzhiqiang1 00 44,888 K 5
taskmgr.exe yangzhiqiang1 02 5,144 K 3
MSMSGS.EXE yangzhiqiang1 00 1,836 K 11
CTFMON.EXE yangzhiqiang1 00 2,424 K 1
SYSMONXP.EXE yangzhiqiang1 13 2,820 K 12
realsched.exe yangzhiqiang1 00 112 K 3
svchost.exe SYSTEM 00 2,576 K 7
mdm.exe SYSTEM 00 2,224 K 5
KAVSvc.EXE SYSTEM 00 5,664 K 9
inetinfo.exe SYSTEM 00 8,708 K 28
spoolsv.exe SYSTEM 00 3,824 K 13
svchost.exe SYSTEM 00 14,500 K 74
svchost.exe SYSTEM 00 3,660 K 8
显示所有用户的进程(S)
结束进程(E)
进程数: 28
CPU 使用: 15%
提交更改: 165272K / 633864
查看(V) 帮助(H)
立即刷新(R)
更新速度(U)
选择列(S)...
选择列
请选择“任务管理器”进程页上将显示的列。
映像名称(I)
PID (进程标识符)(P)
CPU 使用(C)
CPU 时间(E)
内存使用(M)
内存使用增量(D)
内存使用高峰值(K)
页面错误(F)
USER 对象(U)
I/O 读取
I/O 读取字节
会话 ID(S)
用户名(N)
页面错误增量(A)
虚拟内存大小(V)
页面缓冲池(G)
非页面缓冲池(O)
基本优先级(B)
句柄计数(H)
线程计数(T)
GDI 对象(J)
I/O 写入
I/O 写入字节
I/O 其他
I/O 其他字节
确定
取消

A Thread vs. a Process



单线程



多线程

Share more resources

High Concurrence

Fast Switch

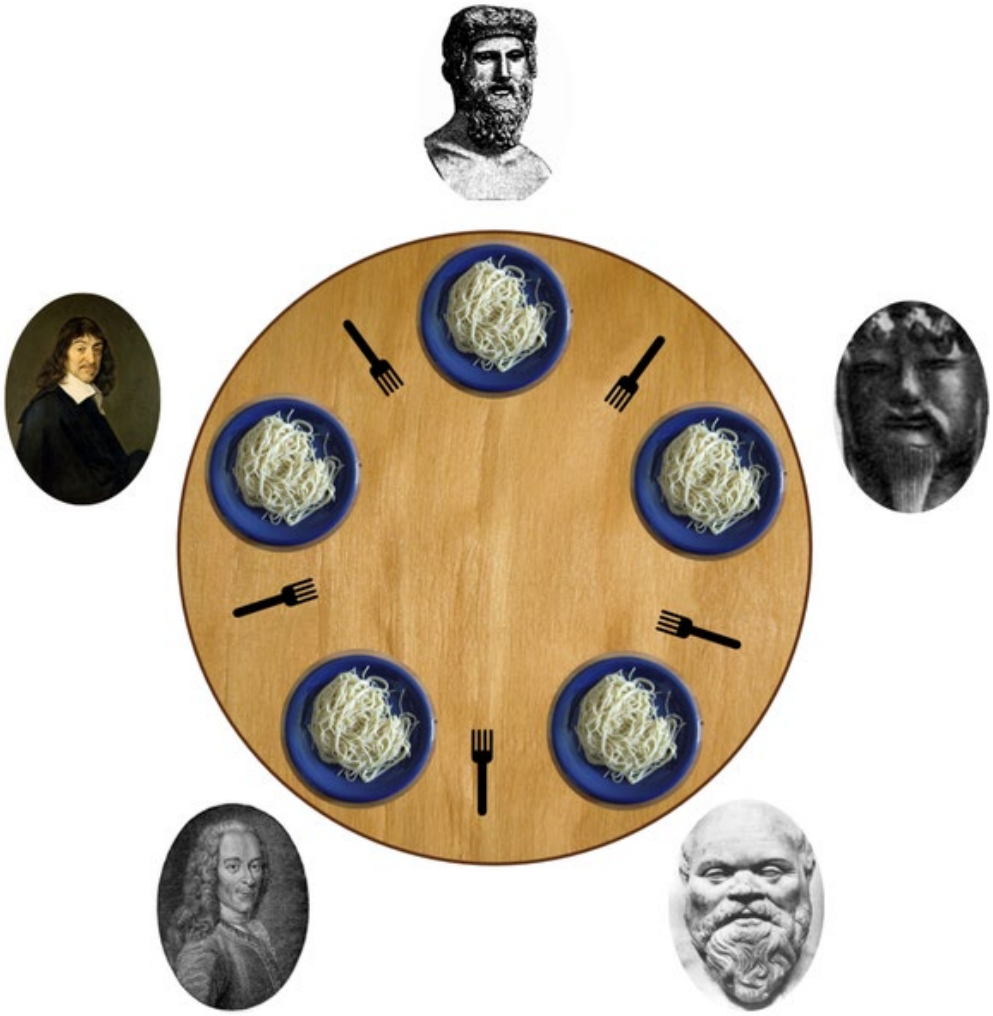
- Questions?

Competition for Resources

- What are *resources*?
  - CPU, memory, files, peripheral devices, …
- In a multitasking system, resources are shared by processes
  - Some resources should not be employed by more than one process simultaneously
  - E.g., printer

Dining philosophers problem

- Each philosopher must alternately think and eat
- A philosopher can only eat spaghetti when he has both left and right forks
- They do not talk to each other
- **What may happen?**



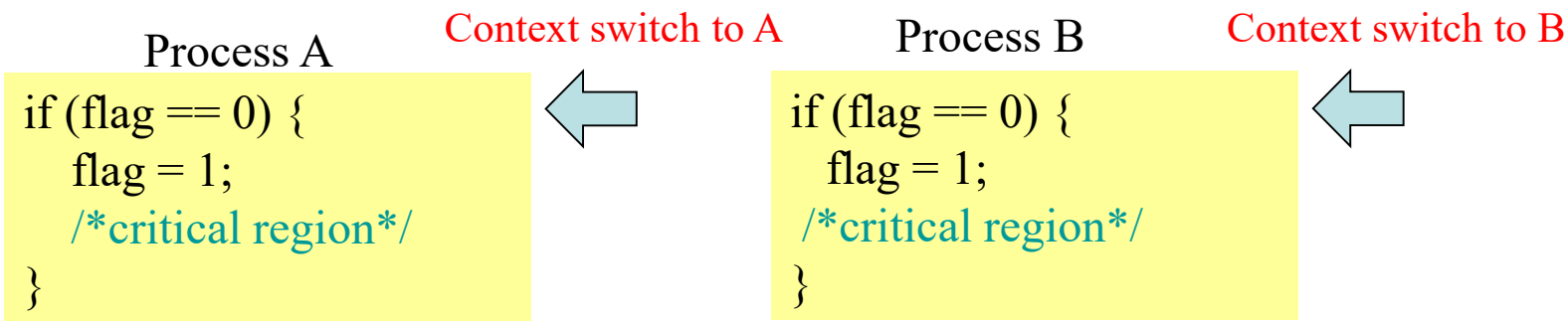
(Edsger Dijkstra, 1965)

Handling Competition for Resources

- **Semaphore:** A “control flag”
- **Critical Region:** A group of instructions that should be executed by only one process at a time
- **Mutual exclusion:** ensuring that no two concurrent processes are in their critical sections at the same time.

First Algorithm

- Use a flag (a global memory address)
  - flag = 1: the critical region is occupied
  - flag = 0: no process is in the critical region
- Problem:



  - Can both processes get into the critical region?

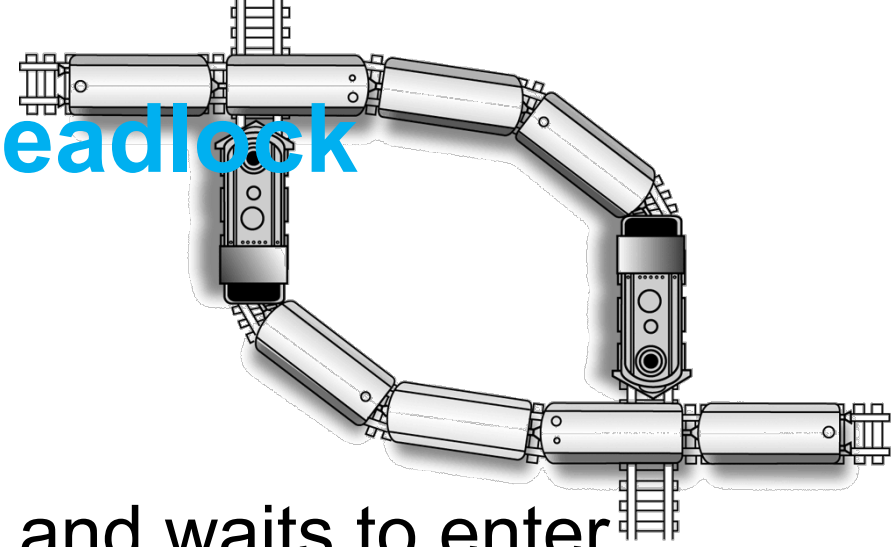
Solutions

- Testing & setting the flag must be completed without interruption
  1. Use disable_Interrupt() to prevent context switch during the flag test and set process.

```
Disable_Interrupt();
if (flag == 0) {
    flag = 1;
    Enable_Interrupt();
    / *critical region*/
}
Enable_Interrupt();
```

  2. A machine instruction called “test-and-set” which cannot be interrupted
- Semaphore: a properly implemented flag

Another Problem: Deadlock



- Example:
  - A is in critical region 1, and waits to enter critical region 2
  - B is in critical region 2, and waits to enter critical region 1

Process A

```
if (test_set(flag1)) {
    /*critical region 1*/
   while(!test_set(flag2));
   /*critical region 2*/
 }
```

Context switch to A

Process B

```
if (test_set(flag2)) {
    /*critical region 2*/
   while (!test_set(flag1));
     /*critical region 1*/
   }
```

Context switch to B

Deadlock

- Processes block each other from continuing
- Conditions required for deadlock
  - Competition for non-sharable resources
  - Resources requested on a partial basis
  - An allocated resource can not be forcibly retrieved
  - Circular wait

Remove any one of the conditions can resolve the deadlock.

Exercises

- There is a bridge that only allows one car to pass. When two cars meet in the middle, it causes “deadlock”. The following solutions remove which conditions?
  - Do not let a car onto the bridge until the bridge is empty.
  - If cars meet, make one of them back up.
  - Add a second lane to the bridge.

Solutions

- Kill one of the process
- Processes need to request all the required resources at one time
- Spooling
  - For example, stores the data to be printed and waits the printer available
- Divide a file into pieces so that it can be altered by different processes

- Questions

Security

- Attacks from outside
  - Problems
    - Insecure passwords
    - Sniffing software
  - Counter measures
    - Auditing software

Security (continued)

- Attacks from within
  - Problem: Unruly processes
  - Counter measures: Control process activities via privileged modes and privileged instructions

Security

- Attacks
  - Malware
  - Spyware and phishing
  - Adware and spam
  - Abnormal behaviors
- Defenses
  - User management
    - Privilege control
  - Protections
    - Antivirus software
    - Auditing software
    - Firewall, spam filter
  - Encryption

User Management

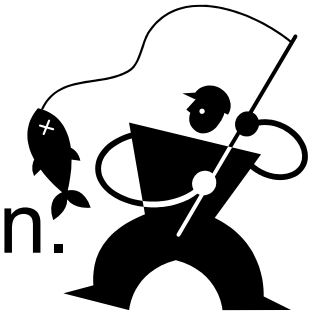
- To protect the computer's resource from access by unauthorized personnel.
- User authentication process:
  - Username, password, fingerprint, …
- Super user / administrator / root
  - A kind of user having higher privilege to control machines and operating system.

Privilege Control

- To prevent malicious programs to execute dangerous instructions.
- Privilege levels:
  - Nonprivilege mode: only “safe” instructions
    - For example, to access some part of memory.
  - Privilege mode: all kinds of instructions
    - Those instructions that can be only executed in the privilege mode are called **privilege instructions**.

Spyware and Phishing

- Spyware: collects information about users without their knowledge.
  - Keylogger: log the keys struck on a keyboard
  - Login sniffing: simulates the login process to get valid user name and password.
  - Network sniffing: intercept network messages
- Phishing: acquires information by masquerading as a trustworthy entity in an electronic communication.



Adware and Spam

- Adware: automatically plays, displays, or downloads advertisements to a computer after the software is installed on it or while the application is being used.
- Spam: sends unsolicited bulk messages indiscriminately.
  - Email spam

Abnormal Behaviors

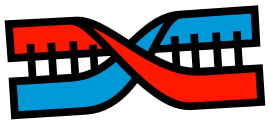
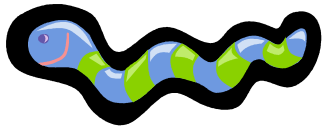
- Dictionary attack: trying passwords derived from a list of words in a dictionary.
- Denial of service attack: overloading a computer (server) with messages to make a computer resource unavailable to its intended users.
- Spoofing attack: masquerading as a party other than one's self

Protections

- **Antivirus software**: detecting and removing the presence of known viruses and other infections.
- **Auditing software**: detecting and preventing abnormal situations
- **Firewall**: filtering messages passing through computers.
  - **Spam filter**: firewall for email spam

Malware 恶意软件

- Infect programs/computers, erase data, slowdown performance…
- Types


  - Virus: attached to an existing program
  - Worm: a stand alone program
  - Trojan horse: disguised as valid files or programs

Review

- Which of the following is a process or a program?
  - A bulk of instructions on the disk
  - A bulk of instructions in the main memory
  - An instance of a program that is being executing

Review

- Which of the following are the facts about processes or threads?
  - They are active instances of programs
  - They represent “sub-tasks” within a task
  - They share the same memory
  - They have a slower switching speed
  - They can communicate with each other