Chapter 1

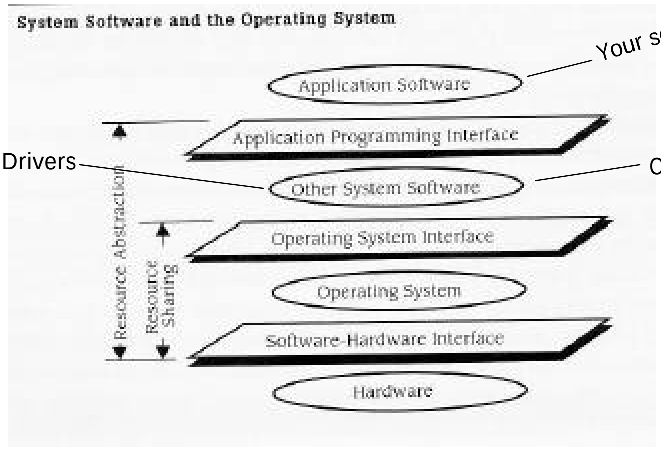
Introduction

What is an Operating System (OS) ?

- Definition 1:
  - An OS is the *interface* between the hardware and the software environment
- Definition 2:
  - An OS is a *resource manager* – provides "resource abstraction"

- In fact, it achieves 1 through 2.
- Therefore, both definitions are applicable at some times.

System Software and the OS interface


System Software and the Operating System
Application Software
Application Programming Interface
Other System Software
Operating System Interface
Operating System
Software-Hardware Interface
Hardware
Resource Abstraction
Resource Sharing

Your software

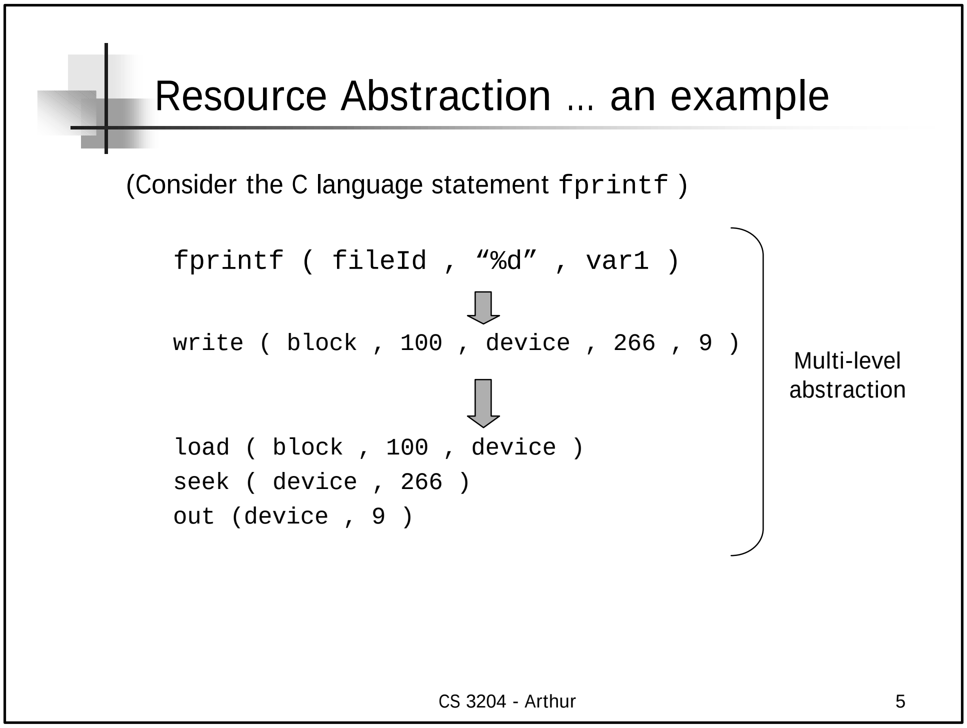
Drivers

Editors
Compilers
Loaders

from the textbook

Resource Abstraction

- How does the OS "manage resources"?
  - By providing *Resource Abstraction* to the other system software and applications
- What is Abstraction ?
  - Abstraction hides the details
- *Resource Abstraction*
  - hides the "nitty-gritty" details of the underlying resource



Resource Abstraction … an example

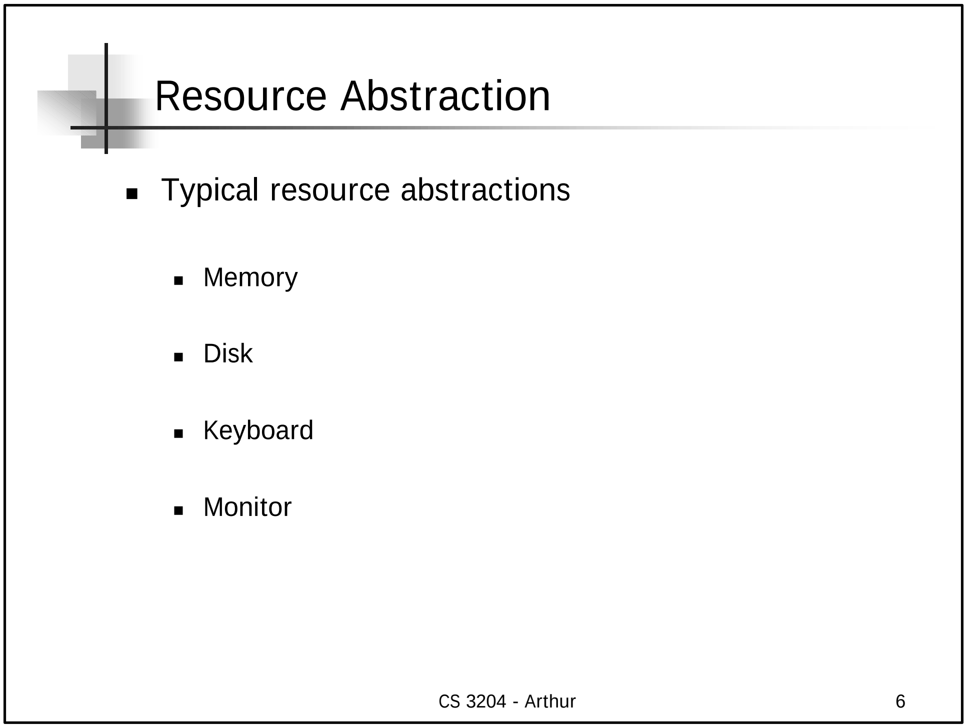
(Consider the C language statement `fprintf`)

```
fprintf ( fileId , “%d” , var1 )

write ( block , 100 , device , 266 , 9 )

load ( block , 100 , device )
seek ( device , 266 )
out (device , 9 )
```

Multi-level
abstraction



Resource Abstraction

- Typical resource abstractions
  - Memory
  - Disk
  - Keyboard
  - Monitor

Resource Sharing

- Managing resources through abstractions implies the ability to *‘share resources’*
- Types of Sharing:
  - Space Multiplexed
    - Divided into 2 or more distinct units of resource
    - Example: disk, memory
  - Time multiplexed
    - Exclusive control for a short period of time
    - Example: processor

Resource Sharing

- Multiple processors accessing *__same__* resource concurrently
- Isolation: only one processor has access at any given time

Terminology

- Concurrency
  - The simultaneous execution of different programs
  - Types of Concurrency
    - **Physical** – multiple processors
      - Example: CPU, I/O
    - **Logical** – interleaved execution
      - Example: processes
- Multiprogramming
  - The concurrent execution of multiple programs on a single processor
  - Could be space-multiplexed into memory and time-multiplexed in processors

Problems:

- Simultaneous access to memory
- Lost updates

OS Strategies

- Batch
- Time share
- PCs and Workstations
- Process Control & Real-time systems
- Networked

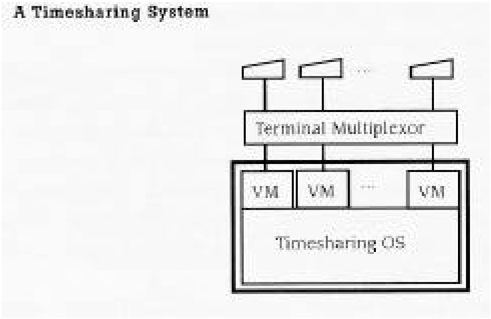
Batch processing systems

- Sequentially loaded set of jobs
- Supported multiprogramming
- Jobs compete for Resources
  - 1st: memory
  - 2nd: processor
  - 3rd: ???
- No "real time" interaction between user and computer

Time share (1970s)

- Multiprogramming environment
- Multiple interactive users
- Why time-share (TS) ?
  - To spread the cost of large machine
  - To fully utilize computing power
- TS provides each user with his/her own Virtual Machine

Time share system…



from the textbook

Time share… ctd.

- TS eventually supported multitasking
  - Multitasking:
    - A time share system that support <u>multiple processes</u> per user, where.
    - A process is a “program in execution

- TS elevated the importance of
  - Need for barriers and safeguards among users and there processes - User/User & Process/Process
    - Memory protection
    - File Protection

Personal Computers (PCs) & Workstations

- Originally
  - Single User
  - Single Processor
- Now
  - Single or Multiple Users
  - Multiprogrammed

PCs Workstations… Evolution

- Earlier machines
  - Too large, too expensive, and too fast for <u>one</u> person

↘

- Mini-computers
  - Smaller versions (like DEC PDP), yet they too grew in size

↘

- Micro-computer
  - Single chip processor

↘

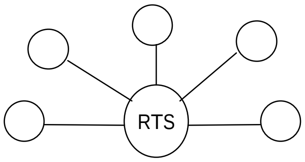
- Workstation
  - Multiple user
  - Multiprogrammed
  - Multitasking

PCs & Workstations… Contribution

- Contributed to the growth of
  - Networking
    - Email
    - File server
  - Point and click interface
    - Like that in Mac and Windows

Process Control & Real time Systems

- Process Control Systems (PCS)
  - Single application monitoring one process
  - Example: System to monitor the heat of a liquid
- Real Time Systems (RTS)
  - Tied together Process Control Systems



Real Time Systems… type

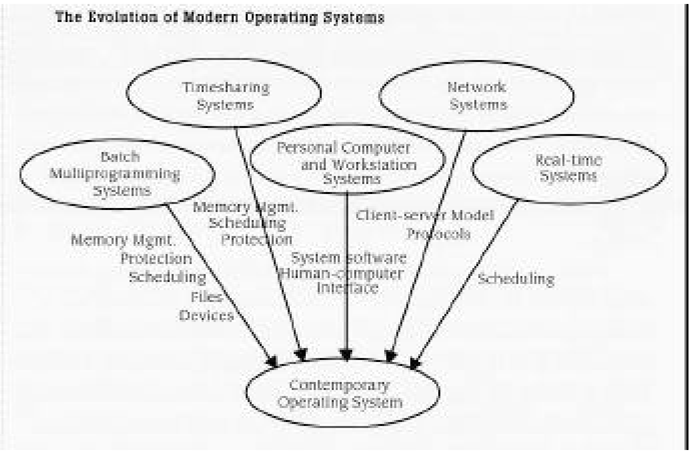
- Hard RTS
  - Had timing constraints that COULD NOT be missed
  - Example: Chemical processes, Nuclear power plants, Defense systems
- Soft RTS
  - Make best effort to accommodate time constraints
  - Example: Transaction processing (ATM)

RTS: Tradeoff of generality of operations/functionality
to ensure that deadlines can be made

Networks of Computers

- Problem is too large
  - Partition it among machines
- Communication exchange
  - Email
  - File transfers
- Servers
  - File
  - Printer
  - Database
- Provide access to non-local resources
  - LAN, WAN
  - Client / Server

Summary



The Evolution of Modern Operating Systems

Timesharing Systems
Network Systems
Batch Multiprogramming Systems
Personal Computer and Workstation Systems
Real-time Systems
Memory Mgmt. Scheduling Protection
Client-server Model Protocols
Memory Mgmt. Protection Scheduling Files Devices
System software Human-computer Interface
Scheduling
Contemporary Operating System

from the text book