Operating Systems: INTRODUCTION

Charles Abzug, Ph.D.
Department of Computer Science
James Madison University
Harrisonburg, VA 22807

Voice Phone: ***540-568-8746***; Cell Phone: ***443-956-9424***
E-mail: ***abzugcx@JMU.edu*** OR ***CharlesAbzug@ACM.org***
Home Page:
https://users.cs.jmu.edu/abzugcx/public/CharlieAbzugHomePage.pdf

Overview of Operating Systems

Tanenbaum, Andrew S. (2008). ***Modern Operating Systems. Third Edition.*** **Upper Saddle River, NJ: Prentice-Hall. ISBN: 0-13-031358-0.**

CHAPTER 1: Introduction

Sobell, Mark G. (2005). ***A Practical Guide to Linux Commands, Editors, and Shell Programming.*** **Upper Saddle River, NJ: Prentice-Hall Professional Technical Reference. ISBN: 0-13-147823-0 (alk. paper).**

CHAPTER 1: Welcome to *LINUX*
CHAPTER 2: Getting Started

DEFINITION

Operating System: the program that manages the computer hardware and that arbitrates the sharing of hardware resources by the applications software.

Principal Functions of the Operating System (OS)

1. Provides abstractions of the hardware resources of the computer, and of some of its own software resources as well, to facilitate the tasks of:

   i. the Applications Programmer: the service of a virtual machine
      a) Hardware services
      b) Software services

   ii. the User: provides an interface between the User and:
      a) the Hardware
      b) the System Software

2. Coordinates in real-time the sharing and use of all the hardware components of the computer among:

   a. multiple Users (some machines)
   b. multiple Processes (*i.e.,* Application Programs)

Sharing of Resources

1. Multiplexing: the sharing of a resource
   a) Space-Multiplexing
      i. Main Memory
      ii. Disk
   b) Time-Multiplexing
      i. CPU
      ii. Printer

2. Concurrency *vs.* Simultaneity.

Major Design Goals

1. Convenience
2. Efficiency
3. Fairness
4. Speed
5. Promptness

Position and Role of the Operating System (OS) in the PC

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-1.

Size of a Modern General-Purpose OS

$\geq$ 5 million lines of source code (SLOC)

Computer Environment:

Principal Determinant of the Relative Importance of the Various Design Goals

Different operating environments may impose radically different design goals.

Computer Environment:

The Principal Determinant of the Relative Importance of Various Design Goals

- Single-User PC: Ease of Use

Computer Environment:

The Principal Determinant of the Relative Importance of Various Design Goals

- Single-User PC: Ease of Use
- Minicomputer/Mainframe: FIRST: Maximum Utilization of Resources
  NEXT: Fairness among Users

Computer Environment:

The Principal Determinant of the Relative Importance of Various Design Goals

- Single-User PC: Ease of Use
- Minicomputer/Mainframe: FIRST: Maximum Utilization of Resources
  NEXT: Fairness among Users
- Workstation: Compromise between Individual Usability and Efficient Resource Utilization

Computer Environment:

The Principal Determinant of the Relative Importance of Various Design Goals

- Single-User PC: Ease of Use
- Minicomputer/Mainframe: FIRST: Maximum Utilization of Resources
  NEXT: Fairness among Users
- Workstation: Compromise between Individual Usability and Efficient Resource Utilization
- Handheld: FIRST in importance: Usability
  SECOND in importance: Economy of Battery Usage

OS Terminology

- Process: In brief, an executing program; *i.e.,* a program that is loaded, in whole or in part, into memory, and that may be running, either continuously or intermittently, on a processor.
  - i. Text segment (Main Memory: User Address Space): executable code
  - ii. Data segment (Main Memory: User Address Space)
  - iii. Stack segment (Main Memory: User Address Space)
  - iv. Process Control Block (PCB: maintained by the kernel in Kernel Address Space)
- Job: another name for a Process when operating in batch mode.
- Job Pool: all jobs that are "in the system," i.e., that have been submitted either by or on behalf of users.
- Job Scheduling: deciding to which jobs or processes is memory to be allocated.
  NOTE: Jobs are otherwise waiting in secondary storage until memory is made available for them.
- CPU Scheduling: deciding which job runs on the CPU.

OS Terminology (continued)

- Kernel: The portion of the OS that needs to remain in Main Memory continuously from the completion of Boot-Up until Shut-Down
- Resource: Anything needed for a process to run, e.g.:
  - i. Memory
  - ii. Space on a disk
  - iii. The CPU
  - iv. Buffer for Input or for Output
- Application Program: designed to perform useful work for an end user
- Utility Program: a relatively small application program that can be called upon either directly by a User or by a process acting on behalf of a user to carry out a relatively small or simple task, *e.g.*:
  - Sort
  - Find
  - Search
  - Display the contents of a directory

Brief History of Control of Computers (OS)

1. Single job at a time: Manual Control
2. Concatenation of several jobs: Batch Mode (also one job running at a time)
3. Change of submission mode: Punched Cards supplanted first by Magnetic TAPE, subsequently by Magnetic Disk
4. Multiprogramming: more efficient use of resources

   Whenever the current job becomes unable to execute (due to I/O or other system service request), another job is dispatched so that the CPU continues to be usefully occupied.
5. Time-Sharing: the rapid interleaving of several jobs. Each job appears to be exclusively resident in the system because of rapid response time.

   Another name for Time-Sharing: Multi-Tasking

An Early Batch System

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-3.

IBM's Job Control Language (JCL): Several OS's

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-4.

Multiprogramming: SEVERAL JOBS MEMORY-RESIDENT

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-5.

Brief History of Control of Computers (OS)

6. Evolution from Mainframe/Minicomputer → Personal Computer (PC):

    CPU utilization no longer the critical issue

7. Evolution from Standalone PCs → Networked PCs:

    Re-Emergence of issues from large multi-user system environment:

        Protection of data
        Isolation of users from each other
        Protection from malicious users

8. Importance of Graphical User Interface (GUI):

    Apple/MacIntosh family: MacOS (GUI over Mach/Free BSD kernel)

    WIntel family: MS-DOS & IBM-DOS evolving to OS/2 and "Windows"

Brief History of Control of Computers (OS)

9. Multiprocessors: tightly-controlled systems

Certain critical resources usually shared:
- System Clock
- System Bus
- Main Memory
- Peripheral Devices
- Power Supply

Brief History of Control of Computers (OS)

9. Multiprocessors: tightly-controlled systems

Certain critical resources usually shared:
- System Clock
- System Bus
- Main Memory
- Peripheral Devices
- Power Supply

Advantages:
- Economies of scale achieved through sharing
- Increased Reliability: Fault Tolerance OR High Availability
- Increased Throughput: ALWAYS less than proportional to number of processors because of:
  - (i) Overhead
  - (ii) Contention for Resources

Two Principal Forms of Multiprocessor Systems:
- (i) Symmetric Multiprocessing (SMP): All processors run OS, and intercommunicate as necessary.
- (ii) Asymmetric Multiprocessing: One master processor runs OS; remainder are slaves.

NOTE: Asymmetric is simpler;
Symmetric is more efficient, more robust.

Brief History of Control of Computers (OS)

10. Back-End (Slave) Processors: processors specialized to handle particular tasks

Examples:
- Channel Processor for I/O (IBM)
- Disk Controller with built-in processor
- Keyboard Controller

11. Distributed Systems:

Loosely Coupled
Separate Independent Autonomous Peer Entities

- LAN: Local-Area Network
- MAN: Metropolitan-Area Network
- WAN: Wide-Area Network

Network Operating System:
- (i) Sharing of files across the network
- (ii) Sharing of devices across the network
- (iii) Cross-Network Inter-Process Communication

Brief History of Control of Computers (OS)

12. Clustered Systems: shared storage

Two types:
- (i) Symmetric Clustering
- (ii) Asymmetric Clustering (hot standby mode)

Importance of Distributed Lock Manager (DLM)

High-Availability YES; Fault-Tolerance NO

13. Handheld Systems (Personal Digital Assistants, PDAs)

Three major issues affecting OS:
- (i) Limitation of amount of memory; consequently, Real Memory only (no Virtual Memory)
- (ii) Slowness of processor (for economy of battery drain)
- (iii) Smallness of size of display screen

Connection via:
- Wireless
- Infra-Red
- Temporary Cable

Brief History of Control of Computers (OS)

14. Real-Time Systems:

Two types: (i) Soft Real-Time: Time constraints are important, but occasional failure to meet them is tolerable.
(ii) Hard Real-Time: Time constraints are ABSOLUTE; failure to meet them can NEVER be allowed to occur.

Brief History of Control of Computers (OS)

14. Real-Time Systems:

Two types: (i) Soft Real-Time: Time constraints are important, but occasional failure to meet them is tolerable.
(ii) Hard Real-Time: Time constraints are ABSOLUTE; failure to meet them can NEVER be allowed to occur.

Soft Real-Time capability is currently built into many OSs;
(a) Real-Time and Non-Real-Time tasks are present simultaneously on the system
(b) Preference in execution is given to Real-Time.

Hard Real-Time:
(a) not miscible with Virtual Memory
(b) limited capability to mix with Secondary Storage.

∴ Hard Real-Time is never built into a general-purpose OS.

Memory Layout for a Simple Batch System

- **The entire user program area is available for use by the one and only job that is running at the current time.**
- **No concurrency**

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 3-1a.

Memory Layout for Multi-Programmed Batch Systems

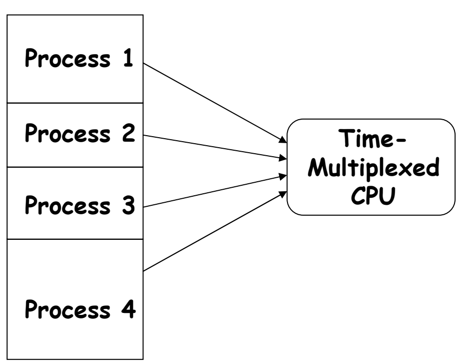
- **Several jobs are kept in memory at the same time.**
- **The CPU is multiplexed among them.**

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-5.

Multiprogramming

- Technique for sharing the CPU among runnable processes
  - Process may be blocked on I/O.
  - Process may be blocked waiting for other resource, including the CPU.
- While one process is blocked, another might be able to run.
- Multiprogramming OS accomplishes CPU sharing “automatically”.
  - scheduling
- Reduces time to run all processes, when taken together, although a particular process may take longer to run.

How Multiprogramming Works



Space-Multiplexed Memory

Timesharing System

- Uses multiprogramming.
- Supports interactive computing model (*i.e.,* the illusion of multiple consoles).
- Different scheduling & memory allocation strategies than batch.
- Considerable attention given to resource isolation (security & protection).
- Intended to optimize response time.

Personal Computers

- CPU sharing among one person's processes.
- Power of computing for personal tasks
  - Graphics
  - Multimedia
- Original trend was towards a very small OS.
- OS focus on resource abstraction.
- Rapid evolution to "personal multitasking" systems.

Process Control & Real-Time

- Computer is dedicated to a single purpose.
- Classic embedded system.
- Must respond to external stimuli in fixed time.
- Continuous media popularizing real-time techniques.
- An area of growing interest.

Networks

- LAN (Local Area Network) evolution
- 3Mbps (1975) → 10 Mbps (1980) → 100 Mbps (1990) → 1 Gbps (2000)
- High speed communication means new way to do computing.
  - Shared files
  - Shared memory
  - Shared procedures/objects

Symmetric Multi-Processing (SMP) Architecture

Critical Defining Feature for SYMMETRIC MULTI-PROCESSING: Each processor can run application programs, or it can run the Operating System (OS).

All processors are equal.

However, at some particular instant there may be no need for the OS to run, in which case all processors can be kept busy running application programs, i.e., doing useful work, as opposed to system overhead. ADVANTAGE: System efficiency.

DISADVANTAGE: Requirement for synchronization and resolution of conflicts among different processors, which may be simultaneously attempting to read and write to data structures in the OS.

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 8-9.

Asymmetric Multi-Processing Architecture

Critical Defining Feature for ASYMMETRIC MULTI-PROCESSING: One processor runs the Operating System exclusively, and the other processors run the application programs. The processor running the OS is permanently dedicated to overhead use, and never performs any useful work.

All processors are NOT equal.

Advantage: Control is simple, and there is no need for conflict resolution.

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 8-8.

Software Layers and their Relationship to the Hardware

Tanenbaum (2008). *Modern Operating Systems. 2d Edition.*
Figure 1-1.
© 2008 Pearson Education

"SPOOL" = Simultaneous Peripheral Operation On-Line

EXAMPLE: a print SPOOL

- **User process sends a print job to the "SPOOLer", *i.e.,* it writes the print job to the SPOOLer's directory.**
- **The SPOOLer is an independent, system-owned process.**
- **The SPOOLer picks up the print jobs one by one, and oversees their printing.**
- **ADVANTAGE I: Possibility of several equivalent printers (printing can be either concurrent or simultaneous).**
- **ADVANTAGE II: User neither knows nor cares which printer prints his/her job.**
- **ADVANTAGE III: All users protected against others' malfunction.**

Historically Important Operating Systems

1. FMS: FORTRAN Monitor System (IBM)
2. IBSYS (for the IBM 704)
3. OS/360 (for the IBM System 360)
4. TS/360: Time Sharing OS for the IBM System 360
5. VM/370: Virtual Machine OS for the IBM System 370, follow-on to the 360)
6. CTSS: Compatible Time Sharing System (Fernando Corbató *et al.* at MIT on an IBM 7094)
7. MULTICS: MULTiplexed Information and Computing Service (General Electric Hardware)
8. UNICS (later changed to UNIX): developed by two refugees from the MULTICS project (originally developed for and ran on a Digital Equipment Corporation PDP-7 minicomputer).
   a) AT&T System V Release 4
   b) ULTRIX (Digital Equipment Corporation runs on VAX hardware)
   c) BSD: Berkeley Standard Distribution (U of California at Berkeley)
   d) Solaris (Sun Microcomputer Corporation for SPARC & other arch.)
   e) AIX (IBM for the PowerPC):
   f) HP/UX
   g) FreeBSD:
   h) LINUX

Historically Important Operating Systems (continued)

9. VMS: Virtual Memory System (Digital Equipment Corporation (DEC), runs on VAX and on DEC Alpha; David Cutler)
10. TOPS-10, RT-11, RSTS (DEC for the PDP-11)
11. CP/M: Control Program for Microcomputers (Digital Research; Gary Kildall)
12. DOS: Disk Operating System (Seattle Computer Products)
13. PC-DOS (Microsoft for the IBM PC) & MS-DOS (for clones)
14. Windoze

Review of Hardware Concepts

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-6.

Simultaneity WITHIN the CPU (I)

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-7a.

Simultaneity WITHIN the CPU (II)

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-7b.

Hierarchical Memory Organization

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-9.

Magnetic Disk Unit

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-10.

Independence from the Process of the I/O Operation

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-11.

Price for Increased Efficiency: Increased Complexity

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-12.

Eight Buses: Cache bus, Local bus, Memory bus, PCI, SCSI, USB, IDE, ISA

Parent and Child Processes

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-13.

Possibility of Deadlock *vs.* Actual Deadlock

NOTE: This figure did not come from our course text.

Example of a Hierarchical File System

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-14.
© 2008 Pearson Education

Attachment of a Device to the Root FileSystem

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-15.

Communication from One Process to Another via a Pipe

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-16.

Development of Hardware Features to Support the OS

NOTE: This figure did not come from our course text.

UNIX System Call to Read Data from a File

```
count = read(fd, &buffer, nbytes);

bytesActuallyRead = read(fileDescriptor, &buffer, numberOfBytesToRead);
```

Sequence of Operations in Implementing a System Call

count = read(fd, &buffer, nbytes);

1,2,3: Push parameters onto the stack.
4: Call the library procedure for file read.
5: Place the sequence number for "file read" into a system-specific register.
6: TRAP (*i.e.,* switch to execution of kernel) in kernel mode.
7: Obtain the address of the handler for the "file read" system call.
8: Execute "file read".

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-17.

Sequence of Operations in Implementing a System Call

count = read(fd, buffer, nbytes);

9: Restore USER-mode and return from the kernel to the library routine.
10: Return from the library routing to the User program.
11: Adjust the Stack Pointer.
12: Continue executing the user's program.

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-17.

Some *POSIX* System Calls

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-18.

System Calls in the "Win32" Application Programming Interface (API)

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-23.

Example of a Simple Shell

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-19.

The Three Major Segments of a Typical Process

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-20.

A Hard Link to a *UNIX* File

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-21.

Mounting of a "Floppy" Disk in a Root File System: BEFORE

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-22a.

Mounting of a "Floppy" Disk in a Root File System: AFTER

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-22b.

Structural Model of an OS of Monolithic Design

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-24.

The "THE" Operating System

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-25.

The "VM370" Operating System with "CMS"

"VM370" ≡ "Virtual Machine 370"

"CMS" ≡ "Conversational Monitor System"

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-28.

Microkernel Architecture (*e.g.,* MACH): Client-Server Model

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 13-3.

Distributed System: Client-Server Model

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-27.

Metric Prefixes

Tanenbaum (2008). *Modern Operating Systems. 3rd Edition.*
Figure 1-31.

END