Two-Hour Introduction to Operating Systems

Dr. Mads Haahr
Department of Computer Science
Trinity College, Dublin

Some slides used here were adapted from notes by William Stallings & Patty Roy – thanks!

What is an Operating System?

- OS = Operating System
- A program that controls the execution of application programs
  - An interface between applications and hardware
  - Makes the computer more convenient to use
- Manages the resources of a computer and controls the way they are used
  - Allows resources to be used in an efficient manner
- Examples of OS's?

Where are OS's Used?

- In more and more places!
- Desktop and Server Computers
  - DOS + Windows 95/98/ME
  - Windows NT/2000/XP
  - Free Unix variants: Linux, FreeBSD, NetBSD, etc.
  - Commercial Unix variants: Solaris, HP-UX, AIX, etc.
  - MacOS
- Some Game Consoles
  - Xbox: Cut-down Windows 2000

Where are OS's Used?

- Personal Digital Assistants (PDAs)
  - PalmOS
  - Windows CE → Windows Mobile
  - Embedded Linux
- Mobile Phones
  - Symbian OS
  - Windows Mobile
- Cars (fancy ones)

Where are OS's Used?

- In the future also:
  - Digital Cameras (fancy ones)
  - MP3 Players (iPods, etc.)
  - Refrigerators!
- Others?

Layers of a Computer System

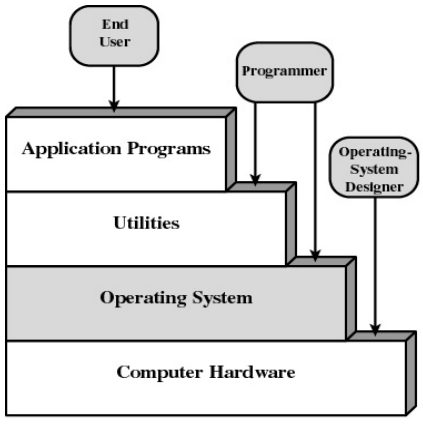


Figure 2.1 Layers and Views of a Computer System

Services Provided by the OS

- Program execution
- Access to Input/Output (I/O) devices
  - Disks, screens, keyboards, mice
  - Printers, cameras, speakers, etc.
- Controlled access to files
- System access
- Sometimes: Program development
  - Compilers, editors and debuggers

Services Provided by the OS

- Error detection and response
  - Internal and external hardware errors
    - Memory errors
    - Device failures
  - Software errors
    - Arithmetic overflow
    - Division by zero
    - Access to forbidden memory locations (why?)
  - Operating system cannot grant request of application

Services Provided by the OS

- Accounting
  - Collect statistics
  - Monitor performance
  - Used to anticipate future enhancements
  - Used for billing users

Operating System as Software

- The OS functions the same way as ordinary computer software
  - It is a program that is executed …
  - … but it has special privileges
- The OS relinquishes control of the processor to execute other programs

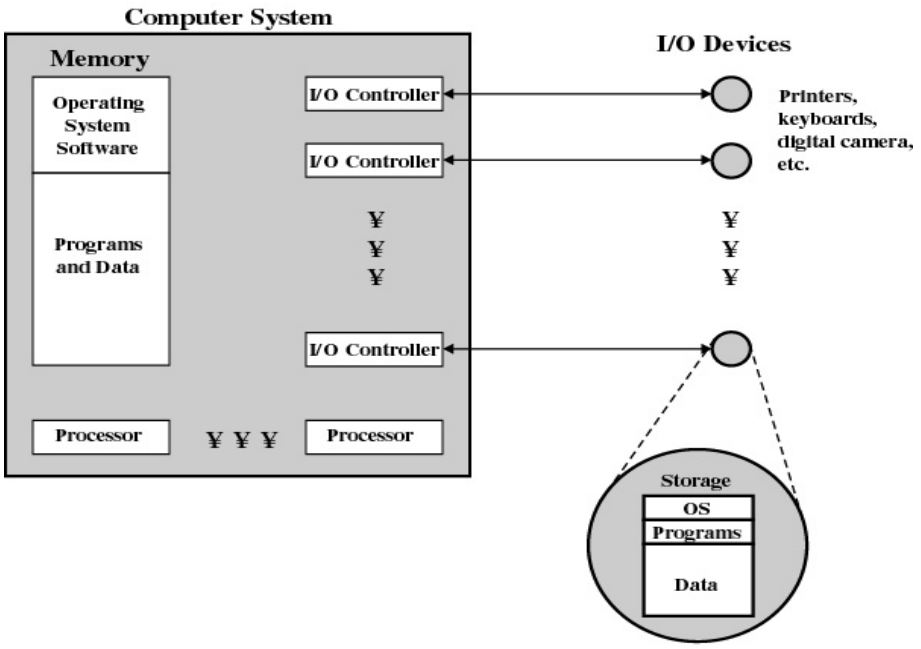


Figure 2.2 The Operating System as Resource Manager

Kernel

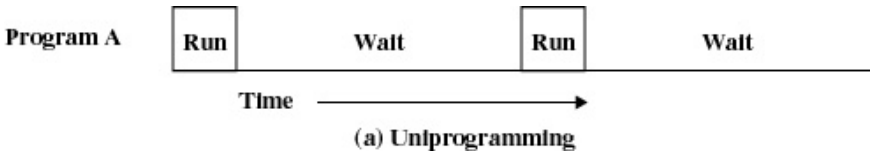
- Portion of operating system that is always in main memory
- Contains most-frequently used functions
- Also called the nucleus
- Good performance of the kernel is very important

Before Operating Systems…

- Serial Processing
  - No operating system
  - Machines run from a console with display lights and toggle switches, input device, and printer
  - Schedule time
  - Setup included loading the compiler, source program, saving compiled program, and loading and linking

Uniprogramming

- Processor must wait for I/O instruction to complete before preceding

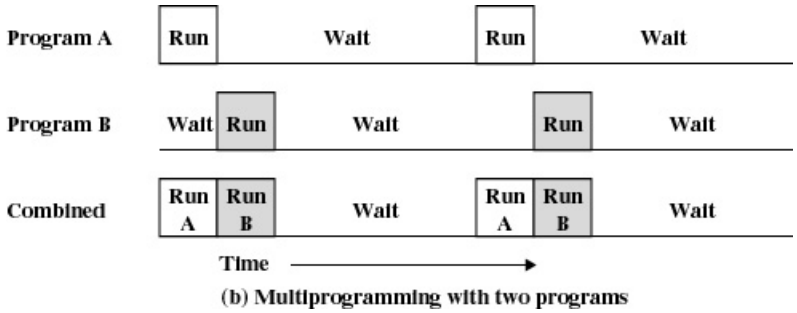


(a) Uniprogramming

- But! Processors are much faster than I/O devices… → inefficient use of CPU

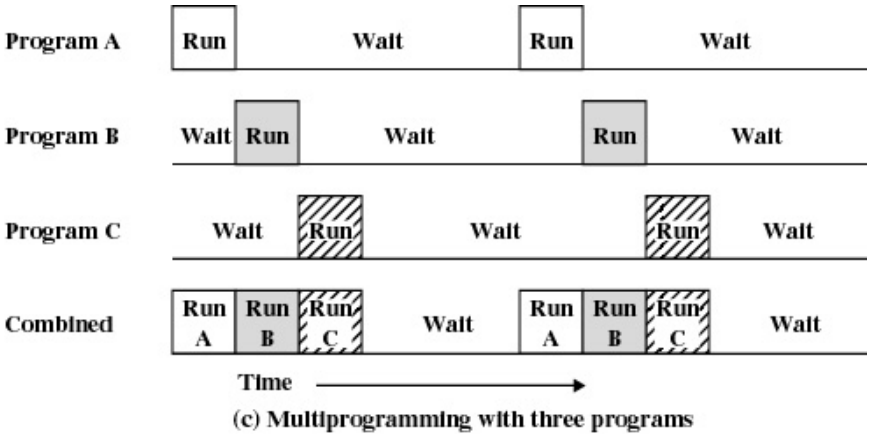
Multiprogramming

- When one job needs to wait for I/O, the processor can switch to the other job



(b) Multiprogramming with two programs

Multiprogramming



(c) Multiprogramming with three programs

A More Advanced Example

	Job 1	Job 2	Job 3
Job type	heavy compute	heavy I/O	heavy I/O
Duration	5 minutes	15 minutes	10 minutes
CPU req'd	80%	10%	10%
Memory req'd	50 MB	100 MB	80 MB
Need disk?	no	no	yes
Need terminal?	no	yes	no
Need printer?	no	no	yes

e.g., image analysis e.g., visualisation e.g., printing from disk

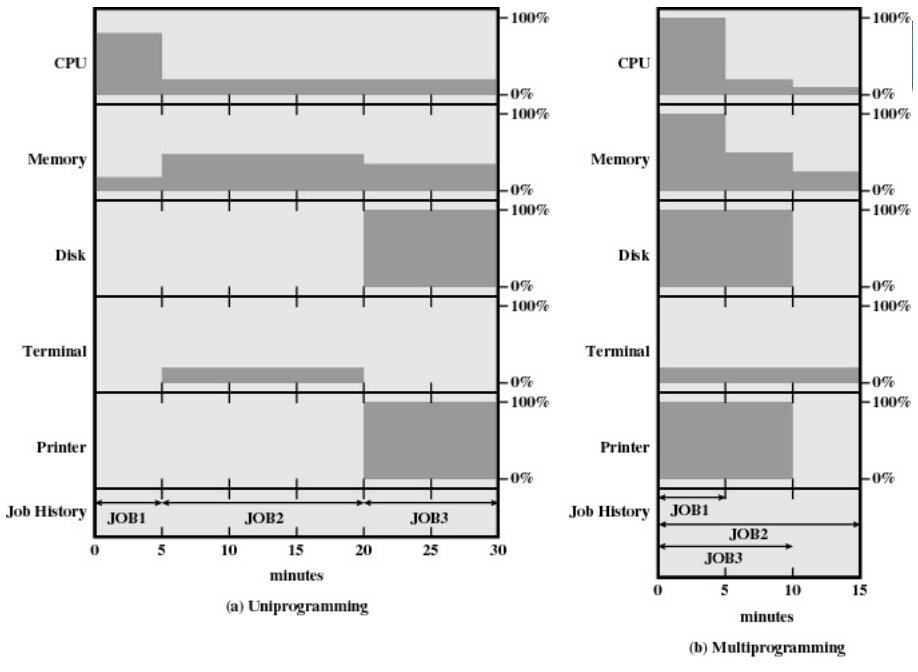


Figure 2.6 Utilization Histograms

Example Effects

	Uniprogramming	Multiprogramming
Processor use	22%	43%
Memory use	30%	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.

Achievements in OS Research

- Processes
- Memory management
- Information protection and security
- Scheduling and resource management
- System structure

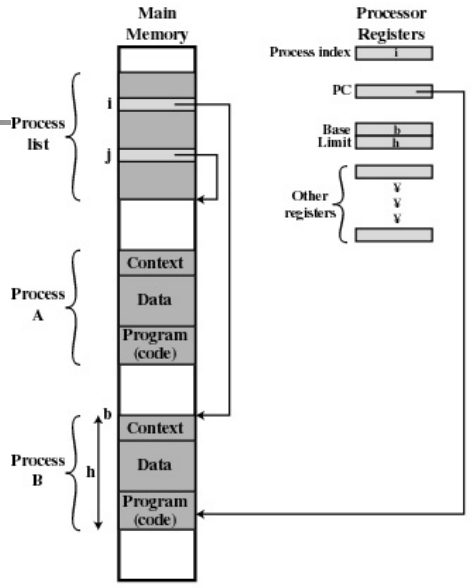
Processes

- A program in execution
- An instance of a program running on a computer
- The entity that can be assigned to and executed on a processor
- A unit of activity characterized by
  - a single sequential thread of execution
  - a current state
  - an associated set of system resources

Processes

- Consists of three components
  - An executable program, i.e., some code
  - Associated data needed by the program
  - Execution context of the program
    - All information the operating system needs to manage the process
    - e.g., who owns the process, which priority does it have, what resources does it currently 'own'

Processes



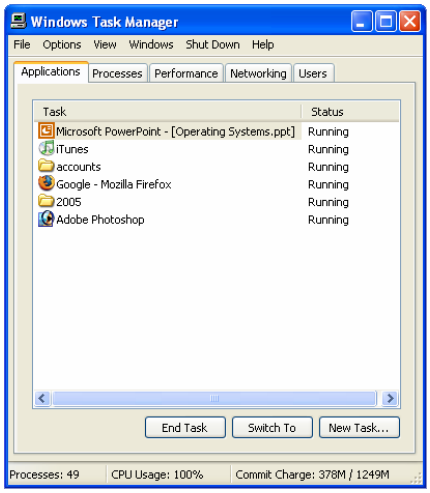
22 February 2005

Figure 2.8 Typical Process Implementation

OS Support for Processes

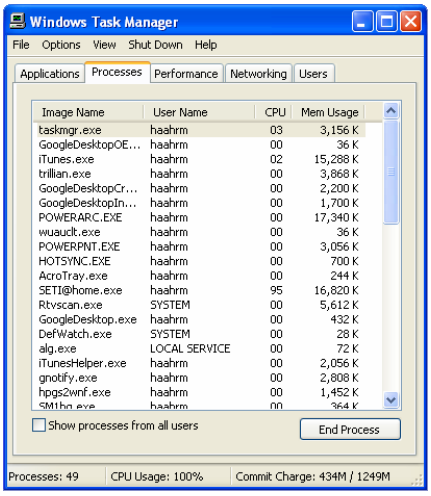
- Creating, destroying, starting, stopping, setting priorities, …
- Process synchronisation
  - Semaphores (wait/signal)
  - Monitors
- Deadlock detection
- Inter-process communication
  - Shared memory

Processes in Windows XP



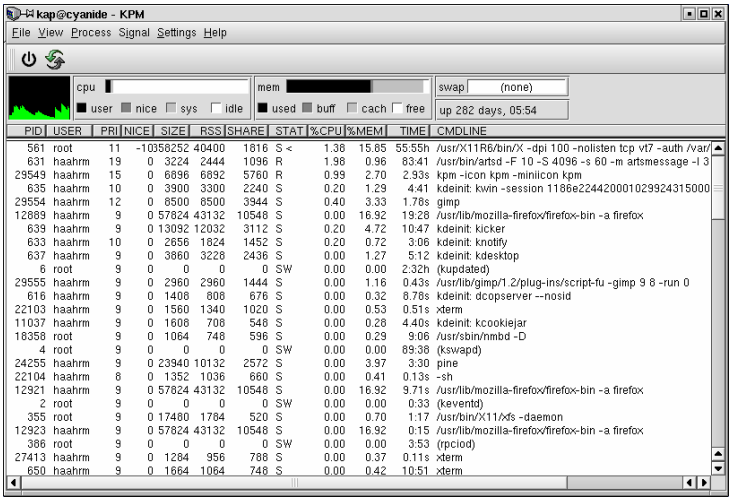
- Hit CTRL-ALT-DEL to start Task Manager
- Shows info about the PC's performance
- Info gathered by OS; updated continuously
- The Applications tab shows running applications and allows them to be ended

Processes in Windows XP



- The Processes tab shows the running processes
- CPU and memory usage
- Applications often consist of a number of processes
- Also, many system processes
- Individual processes can be ended → dangerous

Processes in Linux

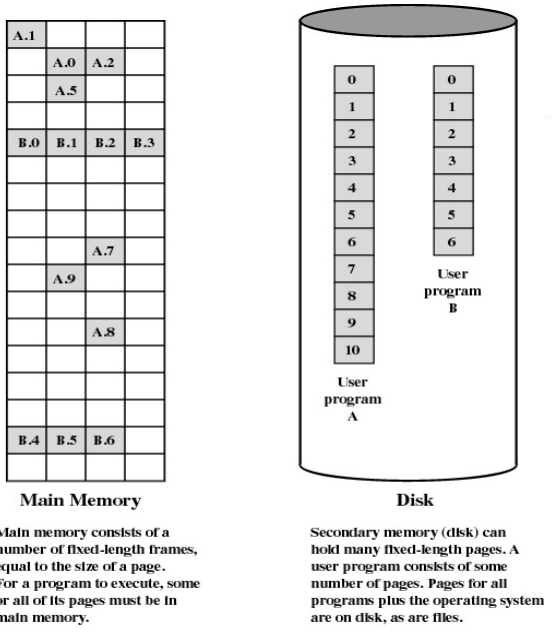


Memory Management

- Process isolation
  - Processes do not share memory
  - This prevents processes from affecting each other
- Protection and access control
- Automatic allocation and management

Virtual Memory and Paging

- Virtual Memory
  - Provides a logical rather than actual view of memory
  - Allows process to be comprised of a number of fixed-size blocks, called pages
- Virtual address
  - A page number and an offset within the page
  - Each page may be located anywhere in main memory
- Real address
  - The physical address in main memory



Main memory consists of a number of fixed-length frames, equal to the size of a page. For a program to execute, some or all of its pages must be in main memory.

Secondary memory (disk) can hold many fixed-length pages. A user program consists of some number of pages. Pages for all programs plus the operating system are on disk, as are files.

Figure 2.9 Virtual Memory Concepts

Virtual Memory Addressing

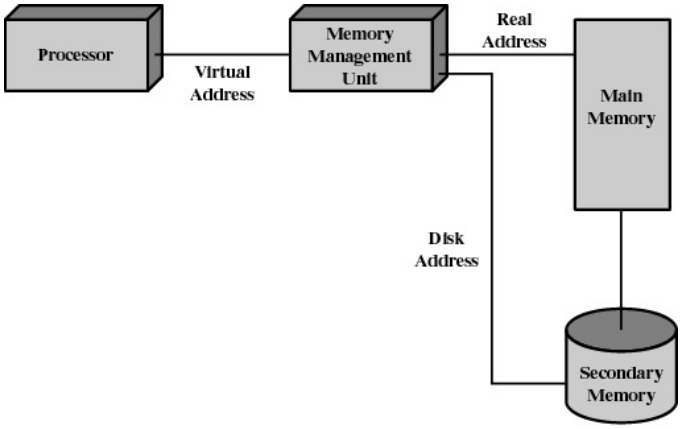
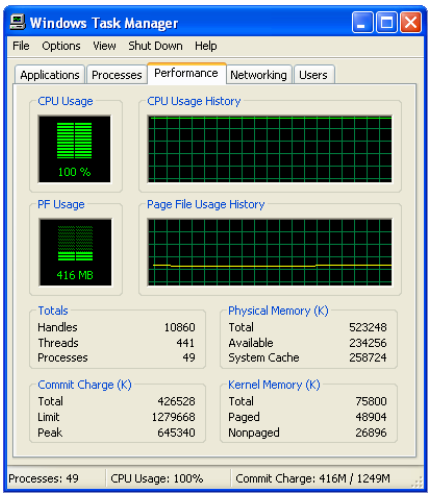


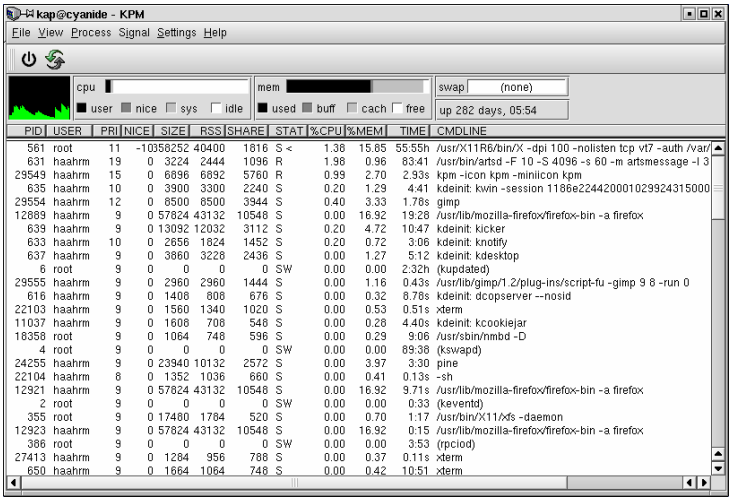
Figure 2.10 Virtual Memory Addressing

Memory in Windows XP



- The Performance tab shows CPU and virtual memory (page file) usage over time
- Memory statistics also shown
- SETI@home client used my spare CPU cycles in example
- Typically, workstations use very few CPU cycles

Memory in Linux



File System

- Implements long-term store
- Information stored in named objects called files
- OS typically offers the following file operations:
  - creating, deleting, renaming
  - reading, writing
  - locking, unlocking

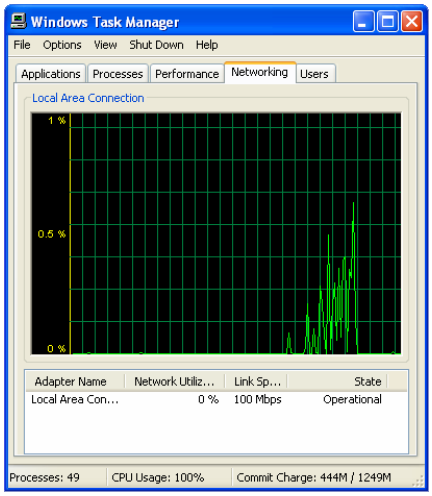
Information Protection and Security

- Access control
  - regulate user access to the system
- Information flow control
  - regulate flow of data within the system and its delivery to users
- Certification
  - proving that access and flow control perform according to specifications

Scheduling and Resource Management

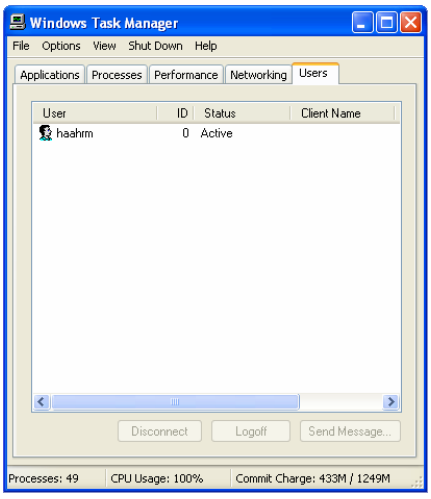
- Fairness
  - give equal and fair access to all processes
- Differential responsiveness
  - discriminate between different classes of jobs
  - e.g., long-running number crunching jobs vs high-priority interactive jobs
- Efficiency
  - maximize throughput, minimize response time, and accommodate as many uses as possible

Windows XP



- The Networking tab shows network bandwidth used over time
- Several network interfaces ('connections' here) are shown separately

Windows XP



- The Users tab shows the users currently logged in

Major OS Elements

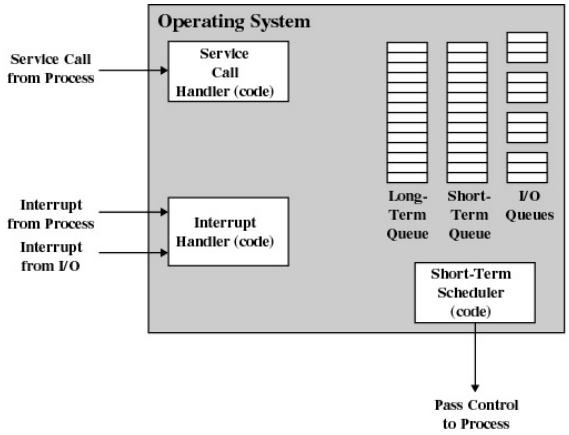


Figure 2.11 Key Elements of an Operating System for Multiprogramming

System Structure

- View the 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
- This decomposes a problem into a number of more manageable subproblems

Modern Operating Systems

- Multithreading
  - Each process is divided into threads that can run simultaneously
- Thread
  - Dispatchable unit of work
  - Executes sequentially and is interruptable
- A process is a collection of one or more threads

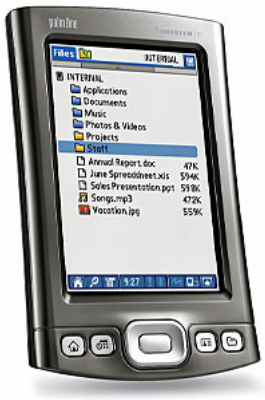
Modern Operating Systems

- Symmetric multiprocessing
  - There are multiple processors
  - These processors share same main memory and I/O facilities
  - All processors can perform the same functions

Modern Operating Systems

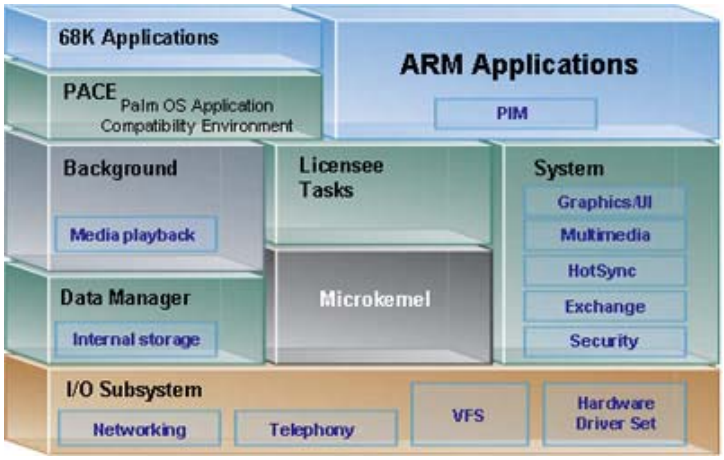
- Distributed operating systems
  - Provides the illusion of a single main memory and single secondary memory space
  - Distributed shared memory
  - Distributed file systems

Example OS: PalmOS



- Used for PalmPilot PDAs and successors
- Multitasking since PalmOS 5
- CPUs: Intel XScale, Texas Instruments OMAP, Motorola Dragonball MX
- Wireless: 802.11b, Bluetooth, GSM, CDMA
- 320×320+ displays
- Good battery utilisation

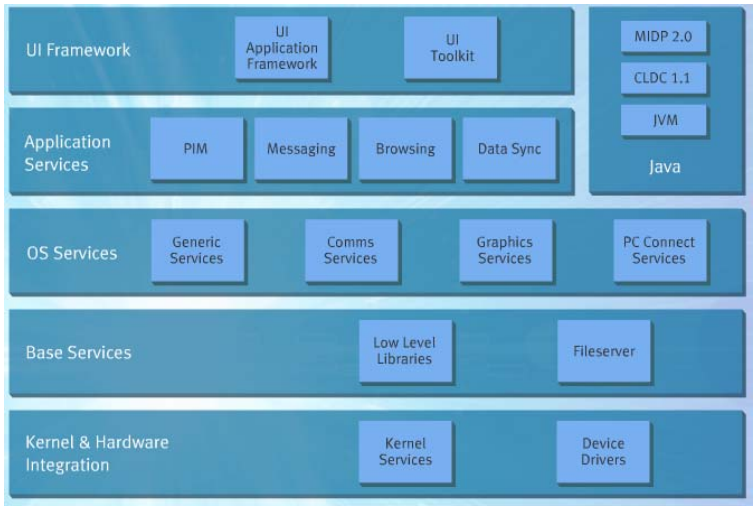
Example OS: PalmOS



Example OS: Symbian OS

- Designed for mobile phones
- Gives access to graphics, multimedia, networking, telephony, crypto, PC connectivity, etc.

Example OS: Symbian OS



The OS Wars

- Why is the OS such a big deal?
  - Windows vs Linux vs Mac
  - Symbian vs PalmOS vs Windows Mobile
- Incompatibilities
  - OS's have different interfaces
  - → programs must be written differently
  - → applications for one OS don't run on another
- Tendency to bloatware
  - Applications tend to move into the OS
  - Internet Explorer, Media Player, Search?

Security: Windows vs Linux

- Windows Family
  - Developed from DOS
  - Originally single-user machines
  - No network → few threats
- Linux
  - Developed from Unix
  - Originally multi-user networked servers
  - Designed to withstand with security threats from the beginning

Further Reading

- William Stallings. *Operating Systems: Internals and Design Principles, 4th edition.* Prentice Hall, 2001.
- Andrew Tanenbaum. *Modern Operating Systems, 2nd edition.* Prentice Hall, 2001.