Operating Systems Architecture

Operating System (OS)

- The **operating system** is program that acts as an intermediary between a user of a computer and the computer hardware
  - In other words, it's the interface between the hardware and the user.
- The primary roles of the operating system are to:
  - Coordinate executing of programs
  - Make the computer system convenient to use
  - Utilize the computer hardware in an efficient manner
- The OS provides an abstraction – making it easier for applications to work with each other without causing conflicts.

What Operating Systems Do

- For workstations, provides users convenience, **ease of use** and **good performance**
- For shared computers such as **mainframes** or **servers** it must keep all users happy
- Users of dedicate systems such as **workstations** have dedicated resources but frequently use shared resources from **servers**
- Handheld computers are resource poor, so optimized for usability and battery life
- Some computers have little or no user interface, such as embedded computers in devices and automobiles

What is an OS?

- It's a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- It's a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer
- "The one program running at all times on a computer" is the **kernel**.
- Everything else is either
  - a system program (ships with the operating system) , or
  - an application program.

Operating System Architecture

The operating system is an abstraction between the computer hardware and the us

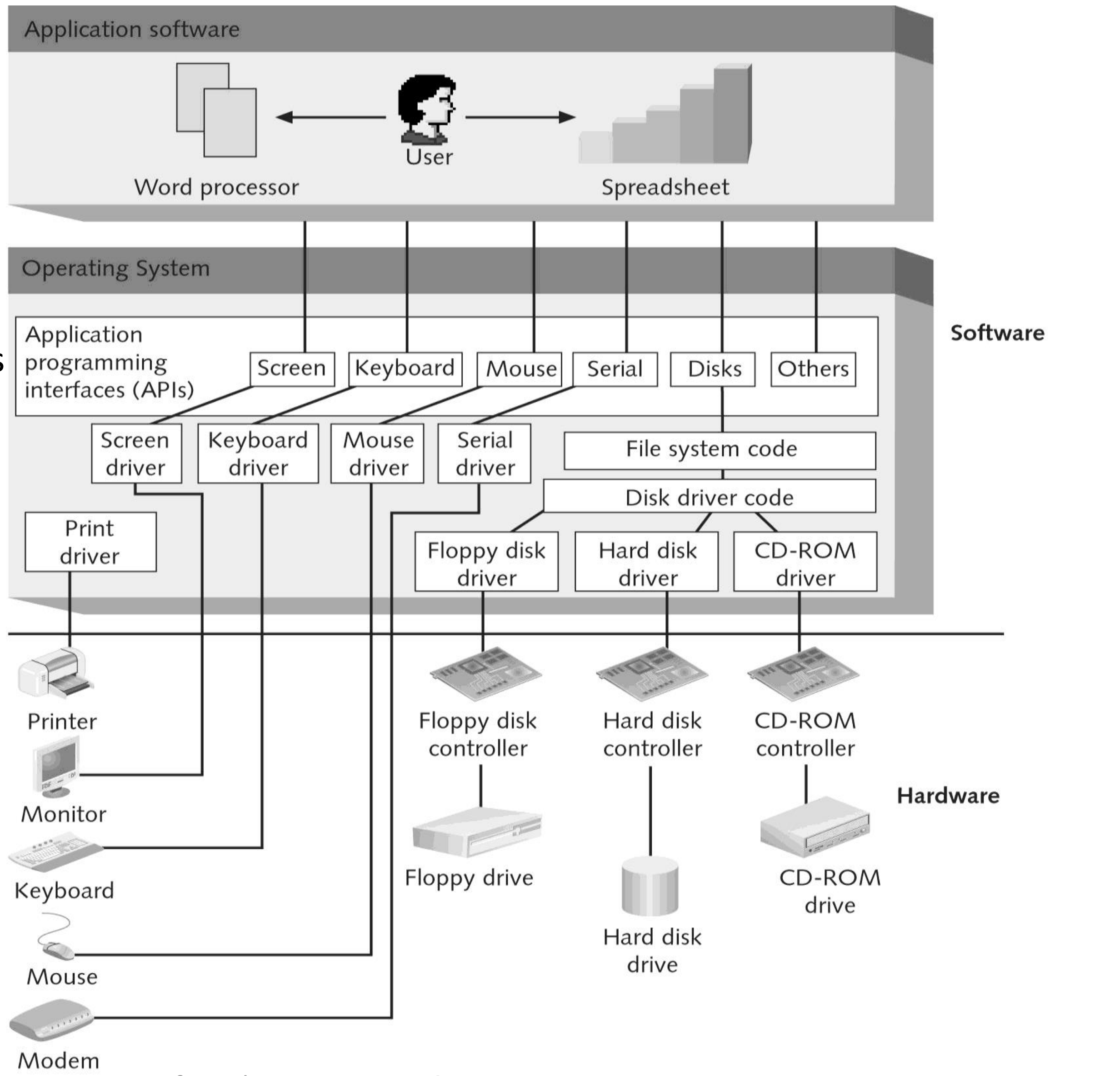


Figure 1-3 Application programs communicate with hardware through the operating system

Popular Components Of Modern Operating Systems

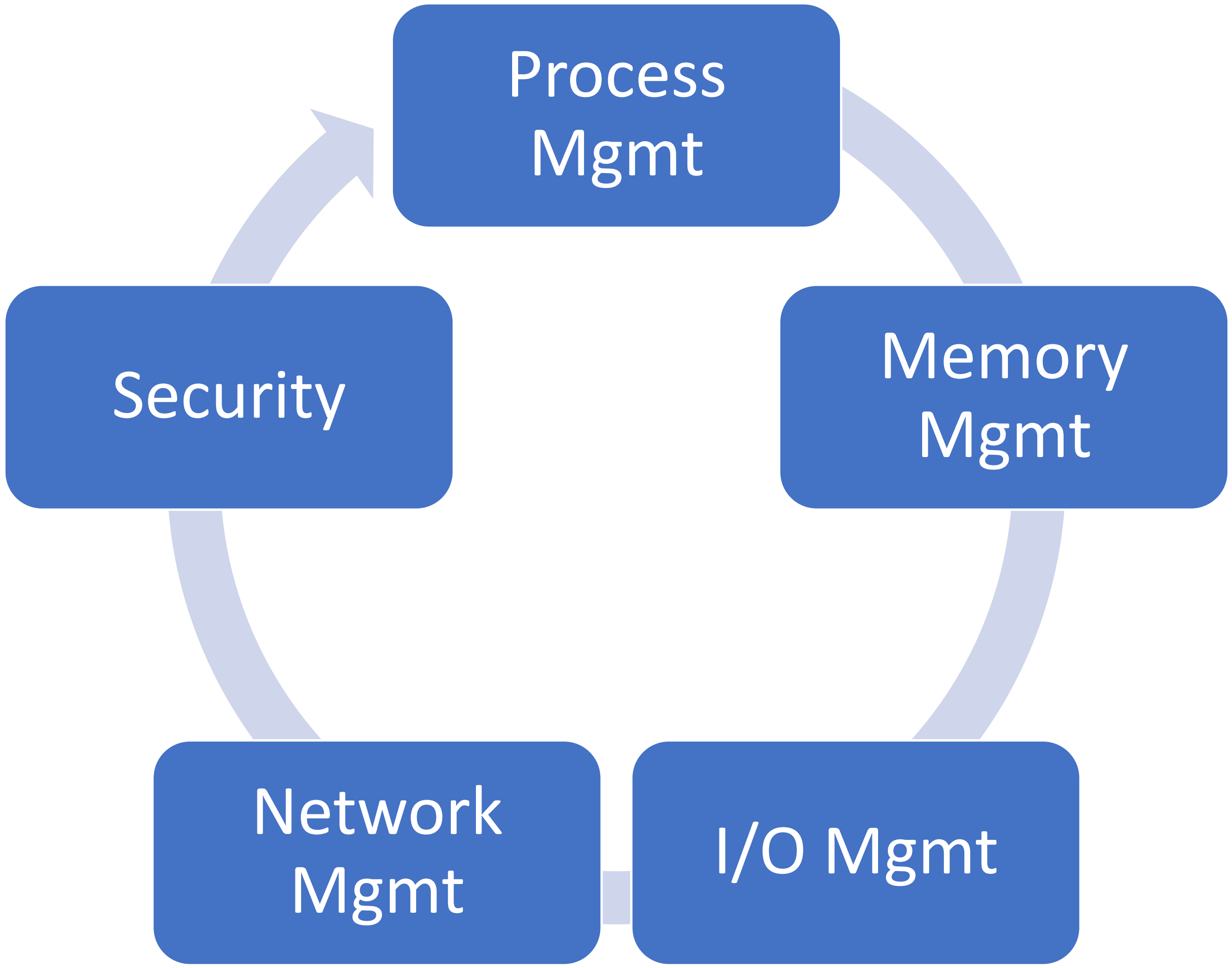
- **Boot loader** – a simple program to load the OS from volatile storage at startup.
- **Kernel** – the heart of the OS. Responsible for processes, memory, protection and devices
- **File System** – an abstraction for the storage of data on disk.
- **Network Stack** – an implementation of network protocols for computer to computer communication
- **Services** – background process to support basic operation.
- **UI** – allows users to interact with the OS. There are command shells like bash or cmd.exe as well as graphical user interface shells, like explorer or finder.

Startup

- A **bootstrap program** is loaded at power-up or restart
  - Typically stored in ROM or EPROM, generally known as **firmware or BIOS**
  - Initializes all aspects of system (hardware)
  - Loads operating system kernel and starts execution

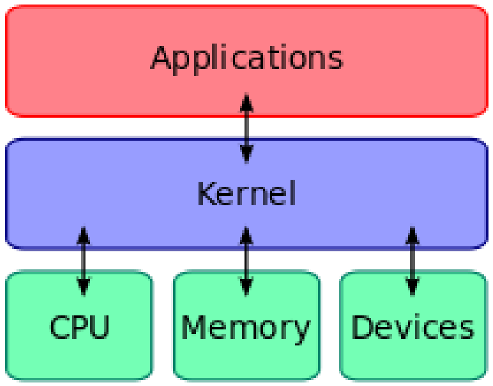
- The kernel takes over and loads the rest of the Operation System.
  - The kernel continues to provide a connection between the application software and the hardware fro this point forward.

The Key OS Kernel Responsibilities



Operating Systems Operations

- **Dual-mode** operations allows an OS to protect itself and other system components
  - Two modes are **User mode** and **kernel mode**
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user



Process Management

- A process is a program in execution. It is a unit of work within the system. A program is a ***passive entity***, a process is an ***active entity***.
- A process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Handling programs in initialization, execution, and termination
  - Task scheduling
  - Program Execution / Termination
- Process termination reclaims any reusable resources
- Single-threaded processes have one **program counter** specifying location of next instruction to execute
  - Process executes instructions sequentially, one at a time, until completion
- Multi-threaded processes have one program counter per thread

Multitasking / Multiprocessing

- **Multitasking** – each application runs as a separate task
- **Multithreading** – parts of the same application can run as separate tasks (video and image processing)
- **Multiprocessing** – when a computer has multiple CPUs / Cores, a single running application can thread across cores
- **Multiprocessor** systems continue to grow in use and importance.
  - Advantages include:
    1. Increased throughput
    2. Economy of scale
    3. Increased reliability (fault tolerance)
  - Two types:
    1. **Asymmetric Multiprocessing** – each processor is assigned a specific task.
    2. **Symmetric Multiprocessing** – each processor performs all tasks

Memory Management

- To execute a program all (or part) of the instructions must be in memory
- All (or part) of the data that is needed by the program must be in memory.
- Memory management determines what is in memory and when
  - Optimizing CPU utilization and computer response to users
- Memory management activities
  - Keeps track of which parts of memory are currently being used and by what processes
  - Deciding which processes and data to move into and out of memory
  - Allocating and de-allocating memory space as needed

Memory management (cont.)

- **Virtual memory** – the presentation of more memory to applications than is available by using disk.
- **Paging** – a block of continuous memory of a predetermined size
- **Page faults** – occurs when an application accesses a page that has been virtualized to disk and therefore needs to be re-loaded back into memory

Storage Management

- The OS provides a uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - files
  - Each medium is controlled by device (i.e., disk drive, tape drive)
- File-System management
  - Organizing of files, usually into directories
  - Access control to determine who can access what files
  - OS activities include:
    - Creating and deleting files and directories
    - Manipulating files and directories
    - Mapping files onto secondary storage
    - Backup files (locally or to separate, non-volatile storage media)

I/O and Network Management

- **Disk management functions** such as free space management, storage allocation, de-fragmentation.
- **Hardware Devices** – Device drivers provide an interface for interaction between the device and the operating system.
- **Networking** – implementation of network protocols for computer to computer communication

- One purpose of an OS is to obfuscate hardware devices from the user
- I/O subsystem is responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (user IDs, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (group ID) allows set of users to be defined and controls managed
  - **Privilege escalation** allows user to change to effective ID with more rights

User Interfaces

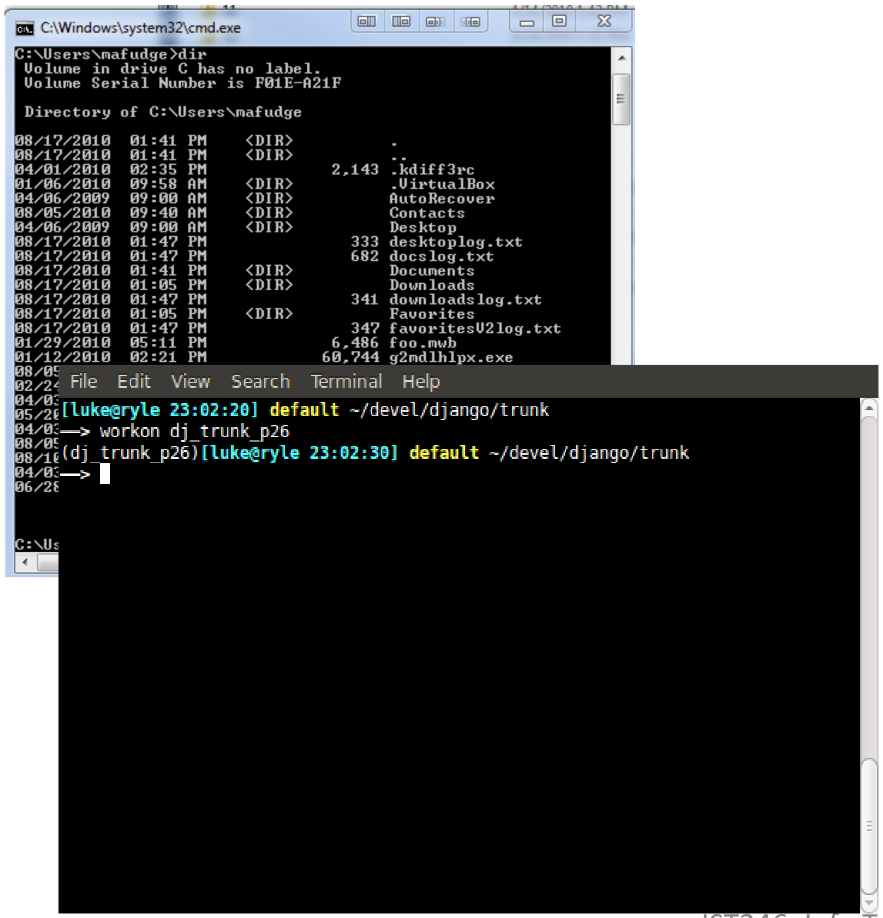
- “The **user interface** is the space where interactions between humans and machines occur.”
- The goal is to allow effective operation and control of the machine from the human end, while the machine feeds back information that aids the users decision making process.
- Typically the goal is to produce a UI which makes it **easy** (self explanatory), **efficient** (fast), and **enjoyable** (user friendly) to operate a machine.
- The term user interface is generally assumed to mean the **graphical user interface** (GUI), but **command-line** (CLI) interfaces also exist for a deeper interaction with the machine and purposes of automation.

Command-line Interfaces

- A **command-line interface (CLI)** is a means of interacting with a program where the user issues commands to the program in the form of successive lines of text.
- The CLI was once the primary means of interaction with most computer systems.
- The interface is usually implemented with a command line shell, which is a program that accepts commands as text input and converts commands to appropriate operating system functions.
- Command-line interfaces are less widely used by casual computer users, but typically preferred by more advanced computer users, as they often provide a more concise and powerful means to control a program or operating system.
- Programs with command-line interfaces are generally easier to **automate** via **scripting**.

User Interfaces

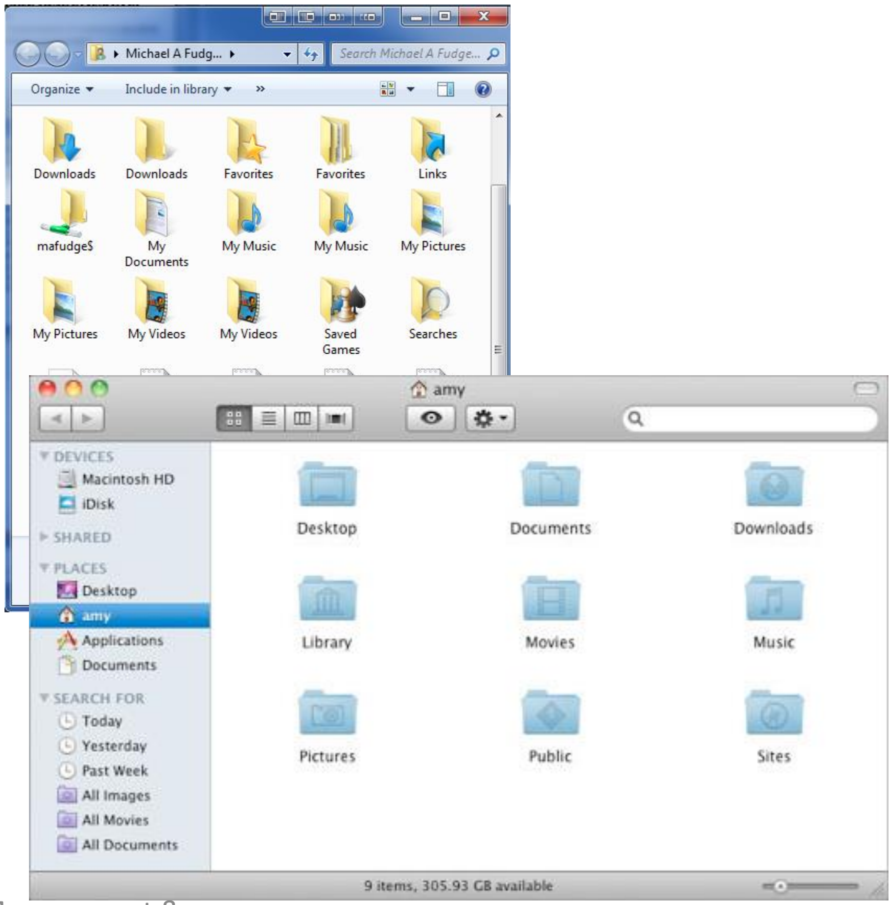
CLI



```
C:\Users\mafudge>dir
 Volume in drive C has no label.
 Volume Serial Number is F01E-A21F

 Directory of C:\Users\mafudge

08/17/2010  01:41 PM    <DIR>          .
08/17/2010  01:41 PM    <DIR>          ..
04/01/2010  02:35 PM             2,143 .kdiff3rc
01/06/2010  09:58 AM    <DIR>          .VirtualBox
04/06/2009  09:00 AM    <DIR>          AutoRecover
08/05/2010  09:40 AM    <DIR>          Contacts
04/06/2009  09:00 AM    <DIR>          Desktop
08/17/2010  01:47 PM               333 desktoplog.txt
08/17/2010  01:47 PM               682 docslog.txt
08/17/2010  01:41 PM    <DIR>          Documents
08/17/2010  01:05 PM    <DIR>          Downloads
08/17/2010  01:47 PM               341 downloadslog.txt
08/17/2010  01:05 PM    <DIR>          Favorites
08/17/2010  01:47 PM               347 favoritesV2log.txt
01/29/2010  05:11 PM             6,486 foo.mwb
01/12/2010  02:21 PM            60,744 g2mdlhlpx.exe
08/05
02/24
04/03
05/20
04/03
08/05
08/10
04/03
06/28


C:\Us
```

```
[luke@ryle 23:02:20] default ~/devel/django/trunk
—> workon dj_trunk_p26
(dj_trunk_p26)[luke@ryle 23:02:30] default ~/devel/django/trunk
—> █
```

GUI


Michael A Fudg...
Search Michael A Fudge...
Organize
Include in library
Downloads
Downloads
Favorites
Favorites
Links
mafudge$
My Documents
My Music
My Music
My Pictures
My Pictures
My Videos
My Videos
Saved Games
Searches
amy
DEVICES
Macintosh HD
iDisk
SHARED
PLACES
Desktop
amy
Applications
Documents
SEARCH FOR
Today
Yesterday
Past Week
All Images
All Movies
All Documents
Desktop
Documents
Downloads
Library
Movies
Music
Pictures
Public
Sites
9 items, 305.93 GB available