Operating System
Context and Motivation

• To understand what is a software system, it is mandatory to understand two very relevant elements
  - The computer as a basic building block for applications
  - The operating system as a mediator between the two layers
• More in detail, the goals and missions of an operating system that impact on the application
Contents

• What is an operating system (O.S.)?
• Elements managed by an O.S.
• Functionality in a O.S.
• Types of O.Ss
• Boot-loading an O.S.
• Parts of an O.S.
  1. Interrupts management
  2. Process management
  3. Memory management
  4. Filesystem management
So, what is an operating system?

Informally:

- A “program” acting as an intermediate between the computer (“infomatic system”) and the hardware
- Goals of an O.S.:
  - Run programs
  - Store users’ data
  - Reduce complexities on the O.S.
- It uses hardware efficiently
Structure of a computer system

Four main components:

1. **Hardware**: CPU, memory, I/O (Input/output) devices

2. **Operating systems**: Controls and coordinates the use of hardware between several applications (users)

3. **Application programs**:
   - The use an operating system to solve users’ problems
   - Applications’ examples: Text processors, compilers, web navigators, data bases, games, etc

4. **Users**
   - People and other machines
Four components in an computer system
Operating System: definition

• There is not an universally accepted definition
  - In fact, each operating system has its own characteristics

• Two main features:

  *The O.S. is a resource manager*
  - Managing all hardware resources included in a computer
  - Managing user demands and addressing conflicts by using fair resource sharing

  *The O.S. is a “program” that controls programs*
  - Involved in applications’ lifecycle
  - That controls programs’ execution and shares the available CPU time among them
O.S. missions

- **CPU sharing:**
  - Scheduling different activities

- **Memory control:**
  - Assigning different memory blocks among programs

- **Input / Output control**
  - Managing data from and towards peripherals
  - Managing interrupts of these devices
  - Managing interrupts from these devices

- **User Interface:**
  - OSs simplify its usage

- **File-system managing:**
  - Organising hard disk information
Types of OSs

Some elements that enable its classification:

- **By the number of users:**
  - Mono user vs. multi-user

- **By the number of tasks:**
  - Mono-tasking vs. Multitasking

- **By the number of processors:**
  - Uniprocessor, multiprocessor (symmetric, asymmetric).
Booting an O.S.

• To boot up a computer, it is followed an initialisation process (more or less similar in all OSs):
  - **Code is read from** the non-volatile storage (ROM)
    “This code detects the boot-up device and reads (from the boot-up sector) the code required to detect the operating system position”
  - **The kernel** of the O.S is loaded
    “Always active functionality such as facilities to run processes”
  - **The O.S takes control starting the O.S services** (programs and system daemons) and booting the user interface (to interact with it)
O.S. internals

- A computer system has
  - One or more CPUs, main memory and I/O devices (all interconnected by buses)
  - Programs that execute concurrently competing for resources
I/O management: Some ideas

• Each I/O device has a HW controller able to run commands concurrently with the CPU
• Each I/O controller has buffers
• The CPU reads and writes in the I/O buffers of the controllers
• I/O Controllers may “interrupt” the CPU by mean of hardware interrupts, to inform that something happened:
  - A new key has been pressed
  - A write operation in the disc has finished
Interrupts

• Operating system has a pre-configured table with information (pointer to a routine) on how to handle each of these hardware interrupts.

• As the hardware detects one interrupt, the control is yield to the routine, method or function included in the interrupt table.

• These routine processes the interrupt interacting with the O.S.

• At the end of the processing of an interrupt, the previous interrupted process resumes.
Interrupts in a PC

CPU

Real Time Clock

Keyboard

Serial
sound
floppy

SCSI

ide0
ide1
Interrupt management

• The O.S. keeps the execution state of the CPU (CPU registers, including the program counter: PC).
• It determines the type of interruption that happened:
  - Polling or asking to each device
  - The use of interrupt vectors
• It runs the routine associated to the interrupt
Chronogram to handle an interrupt
O.S. task queues associated to I/O devices

<table>
<thead>
<tr>
<th>Device</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>card reader 1</td>
<td>idle</td>
</tr>
<tr>
<td>line printer 3</td>
<td>busy</td>
</tr>
<tr>
<td>disk unit 1</td>
<td>idle</td>
</tr>
<tr>
<td>disk unit 2</td>
<td>idle</td>
</tr>
<tr>
<td>disk unit 3</td>
<td>busy</td>
</tr>
</tbody>
</table>

- **request for line printer**
  - address: 38546
  - length: 1372

- **request for disk unit 3**
  - file: xxx
  - operation: read
  - address: 43046
  - length: 20000

- **request for disk unit 3**
  - file: yyy
  - operation: write
  - address: 03458
  - length: 500
Direct Memory Access (DMA)

• The transference of huge amounts of data among an I/O device (for instance a disk) and memory, will be slow if it is done with a CPU

• The device controller reads or writes directly on main memory
  - At the end of the transference, the system is notified by a CPU interrupt
Storage infrastructures

- Internal data from the CPU (very close)
  - Registers
  - Data caches
  - Instructions caches
  - Addresses translation tables

- Main memory– storing the working space of the CPU; i.e., code and data for running programs and the O.S itself

- Secondary storage – System Disks
  - Persistent storage of users’ data and the system

- Tertiary storage – External systems
  - USB disks, USB pens, diskettes ...
Storage hierarchy

Why is it necessary to have different storage hierarchies?

I. Speed
II. Cost
III. Volatility

Caches – increasing efficiency in the system storing data that will be required in a short-time
# Storage hierarchy

<table>
<thead>
<tr>
<th>Level</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>registers</td>
<td>cache</td>
<td>main memory</td>
<td>disk storage</td>
</tr>
<tr>
<td>Typical size</td>
<td>&lt; 1 KB</td>
<td>&gt; 16 MB</td>
<td>&gt; 16 GB</td>
<td>&gt; 100 GB</td>
</tr>
<tr>
<td>Implementation technology</td>
<td>custom memory with multiple ports, CMOS</td>
<td>on-chip or off-chip CMOS SRAM</td>
<td>CMOS DRAM</td>
<td>magnetic disk</td>
</tr>
<tr>
<td>Access time (ns)</td>
<td>0.25 – 0.5</td>
<td>0.5 – 25</td>
<td>80 – 250</td>
<td>5,000,000</td>
</tr>
<tr>
<td>Bandwidth (MB/sec)</td>
<td>20,000 – 100,000</td>
<td>5000 – 10,000</td>
<td>1000 – 5000</td>
<td>20 – 150</td>
</tr>
<tr>
<td>Managed by</td>
<td>compiler</td>
<td>hardware</td>
<td>operating system</td>
<td>operating system</td>
</tr>
<tr>
<td>Backed by</td>
<td>cache</td>
<td>main memory</td>
<td>disk</td>
<td>CD or tape</td>
</tr>
</tbody>
</table>
Multiprogramming managing

• **Multiprogramming**– required for efficiency
  - The CPU many be empty in mono-programming systems (e.g. waiting for I/O)
  - Multi-programming organizes tasks to maximize CPU’s usage
  - These tasks should share the main memory too
  - Processor management is performed by the scheduler
  - If a process is to wait for an I/O action, the OS changes the task

• **Temporal sharing** is a logic sharing in which the CPU changes tasks so quickly that user may interact with tasks without perceiving delays (**interactivity**)
  - Response time should be < 1 second
  - Each user should have at least one task running process
  - If there is not enough main memory, then **swapping**
  - **Virtual memory** combines main memory with **swap memory**
Memory management

![Diagram showing memory management with operating system, job 1, job 2, job 3, job 4, and swap sections.]
Applications’ Interface: System Calls

- An application may use service from the O.S. performing call systems
- These calls run code inside the O.S.
- The user code runs in user’s mode
  - Less priority
  - Less rights for reading in memory
  - Interruptible from the O.S.
- The code of the O.S. runs in “kernel” mode
Application interface: Dual Mode

Dual mode

**User mode** and **kernel mode**

The hardware has a bit in the state registry to indicate the mode of the CPU

- Some instructions of the CPU may only run on **kernel mode** (privileged)
- Each time a call to the system is called, the system changes to kernel mode (as in a hardware interrupt).
From user to kernel mode transition

- Several causes:
  - Timer from the scheduler
  - HW interrupt
  - System call from the application
    - I/O procedures
    - Application switching
    - Accessing to OS data (e.g. date command)
  - One process has not work to perform
Process management

- A process is a running program. It a work-unit in the system. A program or application is an passive *entity*, a process is a *active entity*.
- The process needs resources to complete the task
  - CPU, memory, I/O, files, and communication to other processes
- After the end of a process, it should return all resources to the system
- A process may have threads
  - Each thread is executed within a process
  - Each thread has a different program counter
  - That may be optimized to enhance switching-time among processes
- A modern system will execute concurrently several processes, each one with several threads
Memory management

• To be executed on a CPU, the code of an application should be loaded in memory

• The memory management is in charge of determining what is loaded in memory
  - It tries to optimize the system in cases where there is not enough memory to load all applications

• Memory management related memory
  - To determine what processes use certain memory parts
  - To decide in which memory block is set each process
  - To compact free memory
Process Management

• The operating system is responsible for process management:
  – Creation and process finishing
  – Process suspend and resume
  – Process synchronization
  – Communication processes among processes
  – Avoid starvation and process inter-blockage
Memory management

• To be executed on a CPU, the code of an application should be loaded in memory

• The memory management is in charge of determining what is loaded in memory
  - It tries to optimize the system in cases where there is not enough memory to load all applications

• Memory management related memory
  - To determine what processes use certain memory parts
  - To decide in which memory block is set each process
  - To compact free memory
Files management

• The O.S. is in charge of organizing the information store in disk by using file systems
  - Disk abstraction- **files and directories**
  - The O.S will know how to access the information stored in the disk

• File system management
  - Files stored in directories
  - Each file will have access rights
  - The O.S.:
    - Creates, removes and renames files and directories in disk
    - Offers primitives to read and write files
    - Manages the low-level issues of discs with controllers
    - It performs backups
Memory management

• To be executed on a CPU, the code of an application should be loaded in memory

• The memory management is in charge of determining what is loaded in memory
  - It tries to optimize the system in cases where there is not enough memory to load all applications

• Memory management related memory
  - To determine what processes use certain memory parts
  - To decide in which memory block is set each process
  - To compact free memory
Disk management

- Disk format tools
  - Offering a structure to store a file system
- Reading operations cache
- Writing operations buffer
- Free space management and compact tools
- Operation optimization to improve performance
  - Common zone issue require less time than if they are in other spaces of the disc
I/O Subsystem

• The I/O subsystem is in charge of:
  - Managing buffers that adapt CPU speed to the device (more slow)
  - Caches that improve readings
  - Concurrency management
  - Similar access interfaces (although devices are different)
Thanks
Protection and security

• Protection – access control on the resources of the computer system (e.g. to the file system)

• Security – defense against system attacks, DoS, worms, virus, …

• The system should identify the user and determine its rights:
  - User identity (userIDs)
  - Each resource has a control access zone with each UID and its reading, writing and running associated rights
  - Group identification (groupId) that allows to group users with similar rights