Sistemas de Operação / Fundamentos de Sistemas Operativos

Introduction to operating systems

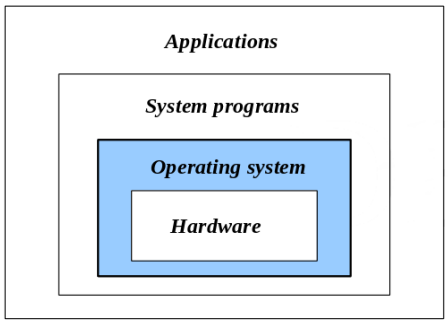
Artur Pereira <artur@ua.pt>

DETI / Universidade de Aveiro

Outline

Global view
Purpose of an Operating System

- Support (base) program executed by the computational system



- Gives life to the computer (hardware), providing an abstract interaction environment
- Acts as an interface between the machine and the application programs
- Dynamically allocates shared system resources to the executing programs
- Manages and schedules memory, processors, and other system devices

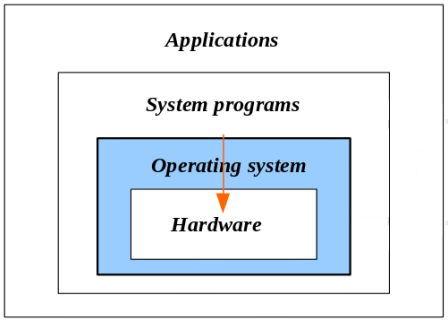
- Objectives of an Operating System:
  - Convenience in the way the computer is used
  - Efficiency in the use of computer resources
  - Ability to evolve as to permit the development of new system functions

---

- Two different perspectives: top-down and bottom-up

Global view
Operating system as an extended machine

- This is a outer-inner or user view



- The operating system provides an abstract view of the underlying computational system that frees programmers of the hardware details
  - A functional model of the computational system (a virtual machine), that is simple to understand and program
  - The interface with the hardware is a uniform programming environment, defined as a set of *system calls*, that allows the portability of applications among structurally different computational systems

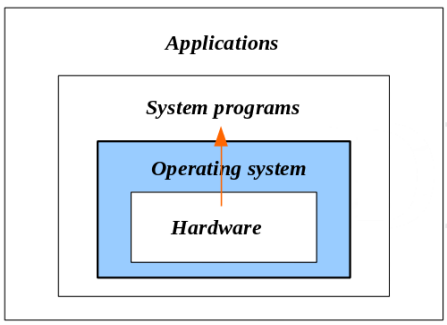
Global view
Operating system as an extended machine (2)

- Some typical functionalities of the operating system:
  - Establishment of a user interaction environment
  - Provision of facilities for the development, testing and validation of programs
  - Provision of mechanisms for the controlled execution of programs, including their intercommunication and synchronization
  - Dissociation of the program's address space from the constraints imposed by the size of the main memory
  - Control access to the system as a whole and to specific system resources, protecting against unauthorized access and resolving conflicts
  - Organization of secondary memory in file systems and control access to files
  - Definition of a general model for access to input/output devices, regardless of their specific characteristics
  - Detection of error situations and establishment of appropriate response

Global view
Operating system as a resource manager

- This is the inner-outer or computational system view

- A computational system is a system composed of a set of resources (processor(s), main memory, secondary memory, I/O device controllers) targeting the processing and storage of information



- The operating system is the program that manages the computer system, making the controlled and orderly allocation of its different resources to the programs that compete for them
  - resource usage is multiplexed in space and time
- aims at maximizing the performance of the computational system, ensuring the most efficient use of existing resources

Global view
Operating system as a resource manager (2)

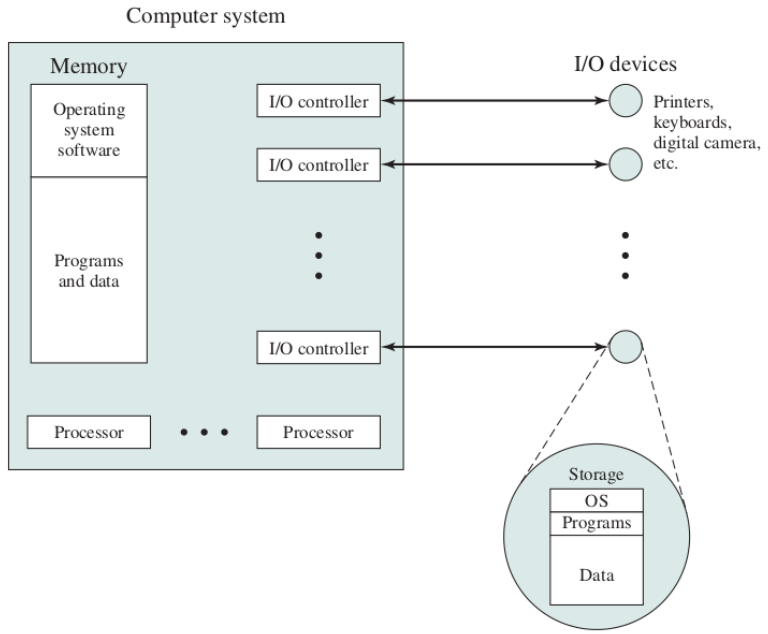
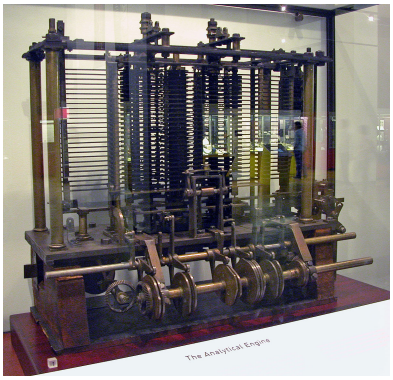


Figure 2.2 **The Operating System as Resource Manager**

Operating Systems: Internals and Design Principles, William Stallings

Evolution of computational systems
Prehistory

Period	Technology	Features
• Mid 19th century	• mechanical device	• no operating system • programmable via punched cards • conditional statement • never work properly



Analytical engine
Charles Babbage (& Ada Lovelace)

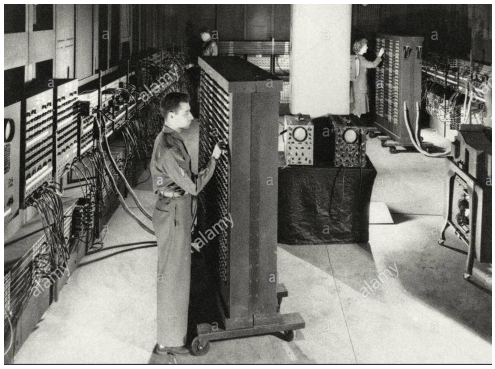
Evolution of computational systems
1st generation

Period
- 1945–1955

Predecessors
- Atanasoft-Berry (1937)
- Konrad Zuse (1941)
- Howard Aiken (1944)

Technology
- vacuum tubes
- electromechanical relays

Features
- programmed in machine language
- punched cards
- serial processing
- programmer has full control of the machine
- no operating system



ENIAC
(Electronic Numerical Integrator and Calculator)
J. Presper Eckert & John Mauchly
University of Pennsylvania

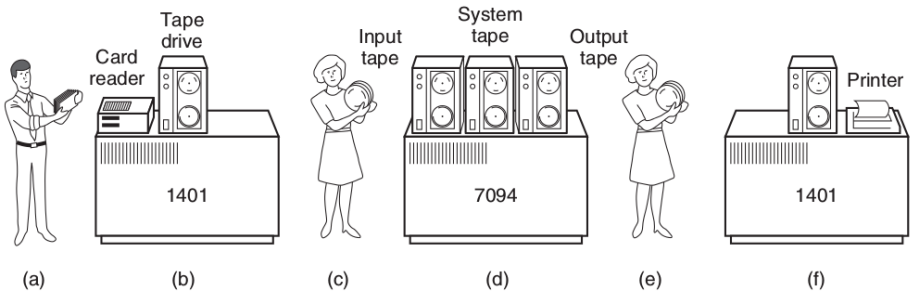
An Illustrated History of Computers, John Kopplin, 2002

Evolution of operating systems
Serial processing

- Serial processing – one user at a time
- User has direct access to the computer/processor
  - actually there is no operating system
- Scheduling and setup time issues:
  - computer must be previously reserved by users, possible causing waste of time or premature exit
  - a considerable amount of time was spent setting up the program to run
- Common software for all users (like linkers, libraries, . . .) appear
- Very poor processor utilization

Evolution of computational systems
2nd generation

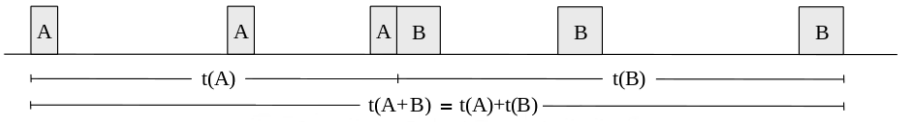
Period	Technology	Features
• 1955–1965	• transistor devices	• FORTRAN and assembly • simple batch operating system (monitor) • rudimentary command language (job control language)



Modern Operating Systems, Andrew Tanenbaum & Herbert Bos

Evolution of operating systems
Simple batch

- Simple batch – one job at a time
- user looses direct access to the processor
  - users submits jobs on cards or tapes to an operator
  - the operator joins jobs into a batch and places it in the computer input device
  - the monitor (batch OS) manages the execution of the batch's jobs in the processor
- better than serial processing
  - scheduling is done by the monitor, one job after the other
  - a job control language helps improving setup time
- requirements like memory protection and privileged instructions appear
  - (most of) the monitor and the user program must be in memory at the same time
- still poor processor utilization
  - processor is often idle



Evolution of computational systems
3rd generation

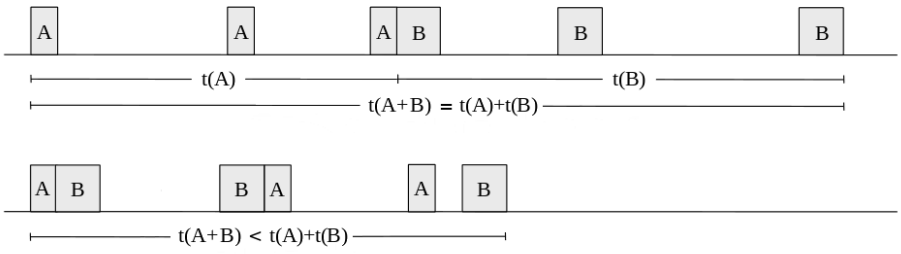
Period	Technology	Features
• 1965–1980	• integrated circuits (SSI/MSI) • family of computers (IBM S/360) • 16-bit minicomputers (DEC PDP-n) • 4-, 8-, 12- and 16-bit microprocessors	• multiprogrammed batch operating system • interative systems (time-sharing): CTSS (MIT); MULTICS; Unix • (proprietary) general usage operating systems • real-time systems

An IBM System/360 Model 20 CPU
with front panels removed,
with IBM 2560 MFCM (Multi-Function Card Machine)

wikipedia.com

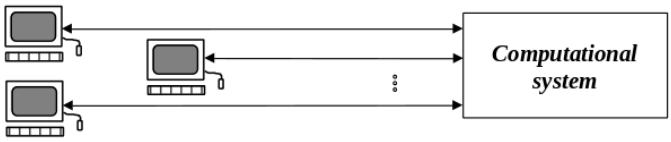
Evolution of operating systems
Multiprogrammed batch

- Multiprogrammed batch – while a job is waiting for the conclusion of an input/output operation, another job can use the processor
  - the approach is known as multiprogramming or multitasking
- scheduling, resource management and hardware support are required
  - monitor and all user programs (jobs) must be simultaneously in memory
  - interrupt and DMA I/O are needed to accomplish the I/O operation
  - what job to dispatch next?
- good processor utilization
  - but there is no direct user interaction with the computer



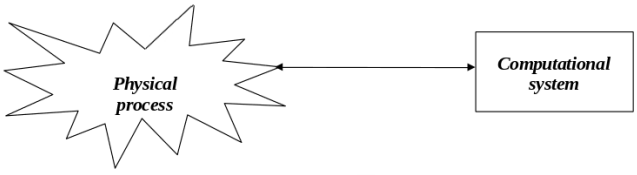
Evolution of operating systems
Time-sharing

- Time-sharing – keeping different users, each one in a different terminal, in direct and simultaneous connection with the system
  - when computers were big and costly, the option was to share them
- Using multiprogramming, the processor is successively assigned to the execution of the various programs present during very short time intervals (time quantum)
  - since a human response, when compared to a computer, is slow, the illusion that the system is entirely dedicated to every user is created
- Differences with multiprogrammed batch
  - try to minimize response time instead of maximize processor use
  - directives to OS given by user commands instead of job control language commands
- In addition to memory protection and privileged instructions, file access protection and resource (printers, . . .) contention are issues to be considered



Evolution of operating systems
Real-time operating systems

- Purpose – use the computer in the online monitoring and control of physical processes
- Method – variant of an interactive system that allows to impose maximum limits to the response times of the computational system to different classes of external requests



Evolution of computational systems
4th and 5th generations

4th generation

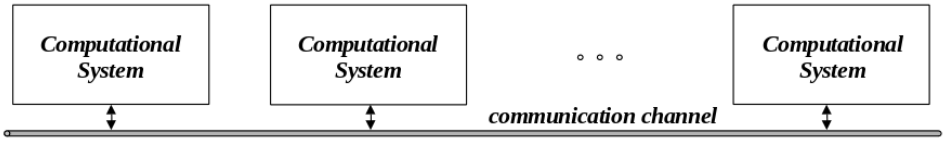
Period	Technology	Features
• 1980–present	• LSI/VLSI • personal computers (microcomputers) • network	• standard operating systems (MS-DOS, Macintoch, Windows, Unix) • network operating systems

5th generation

Period	Technology	Features
• 1990–present	• broadband, wireless • system on chip • smartphone	• mobile operating systems (Symbian, iOS, Android) • cloud computing • ubiquitous computing

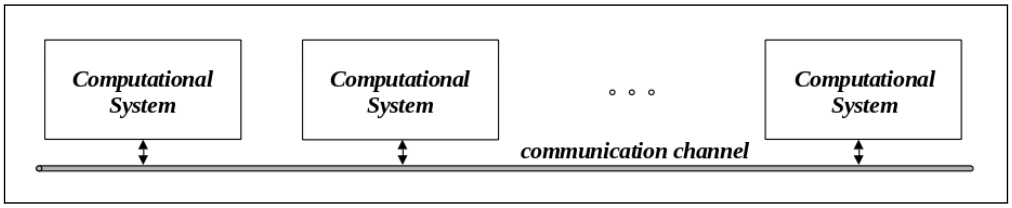
Evolution of operating systems
Network operating systems

- Purpose – take advantage of the interconnection facilities of computer systems (at the hardware level) to establish a set of services common to an entire community
- Services – file transfer (ftp); access to remote file system (NFS); resource sharing (printers, etc.); access to remote computational systems (telnet, remote login, ssh); e-mail; access to the world wide web (browsers)



Evolution of operating systems
Distributed operating systems

- Purpose – take advantage of the facilities for constructing multiprocessor computing systems, or for interconnecting different computational systems, to establish an integrated interaction environment where the user views the parallel computing system as a single entity
- Methodology – ensure a complete transparency in the access to processors or other resources of the distributed system, in order to allow
  - static/dynamic load balancing
  - increasing the speed of processing by incorporation of new processors or new computational systems
  - parallelization of applications
  - implementation of fault tolerance mechanisms



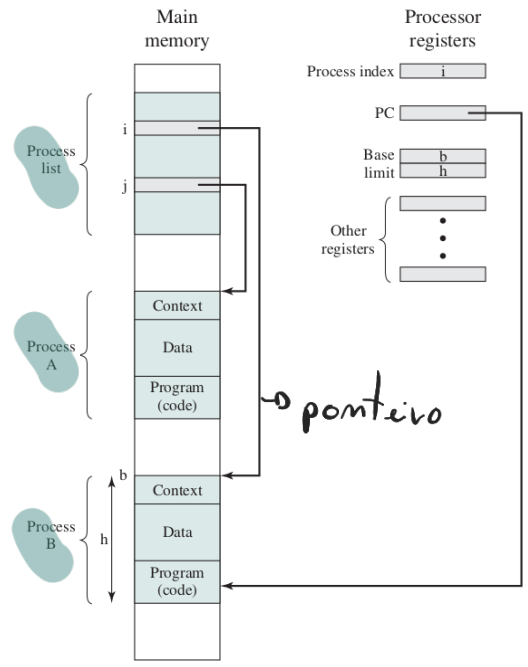
Key topics
Major advances

- Major theoretical advances in the development of operating systems:
  - The concept of process
  - Memory management
  - Resource scheduling and management
  - Information protection and security
- Each advance is characterized by principles, or abstractions, developed to meet difficult practical problems

Key topics
The process

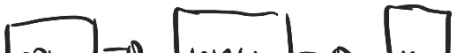
- It appears as a mean to control the activity of the various programs executing in a multiprogrammed system, possible belonging to different users
  - A process is not a program
- Program – set of instructions describing how a task is performed by a computer
  - In order for the task to be actually performed, the corresponding program has to be executed
- Process – an entity that represents a computer program being executed
  - it represents an activity of some kind
  - it is characterized by:
    - addressing space – code and data (actual values of the diferent variables) of the associated program
    - input and output channels and data (data that are being transfered from input devices and to output devices)
    - actual values of the processor internal registers
    - state of execution
    - other process specific data (process ID, ...)
- Different processes can be running the same program

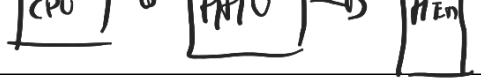
Key topics
A possible implementation of a process



Operating Systems: Internals and Design Principles, William Stallings

- each process occupies a block of memory
  - containing the program, the data, and part of the context
- each process occupies a record in a process list
  - with a pointer to the process' memory
  - and the other part of the context
- a process index identifies the running process
- the base and limit registers define the running process memory
- the program counter and all data references are checked against these register values, at every access





Key topics
Memory management

- Memory management is required in order to support a flexible use of data
- Some OS memory management funcionalities:
  - Process isolation – independent processes can not interfere with each other's memory
  - Automatic allocation and management – Programs should be dynamically allocated across the available memory
  - Dynamic growth of used memory – Memory used by programs should not be defined at start
  - Protection and access control – Sharing of memory should be possible, in a controlled way
  - Long-term storage – Many application programs need to store information for extended periods of time, after the computer has been powered down.
- This is accomplished with virtual memory and file systems
  - Virtual memory dissociates the memory see by a process and the real memory
  - File systems introduced the concept of file as the mean for long-term storage

Key topics
Resource scheduling and management

- There is a variety of resources shared among processes
  - main memory, I/O devices, processors, . . .
- The OS must:
  - manage these resources
  - schedule their use by the various active processes
- The resource allocation and scheduling policy must provide:
  - fairness – give approximately equal and fair access to resources
  - differential responsiveness – act considering the total set of requirements
  - efficiency – attempt to maximize throughput, minimize response time, and, in the case of time sharing systems, accommodate as many users as possible
    - may not be possible, as conflicts may exist

Key topics

Information protection and security

- In time sharing systems, short-term and long-term data may belong to different users
  - thus, protection and security is a requirement
  - meaning controlling access to computer systems and to the information stored in them
- Points covered by OS:
  - Availability – protect the system against interruption
  - Confidentiality – ensure users cannot access unauthorized data
  - Data integrity – protect data from unauthorized modification
  - Authenticity – verify the identity of users and the validity of messages or data