Sistemas de Operação / Fundamentos de Sistemas Operativos

File systems in a nutshell

Artur Pereira <artur@ua.pt>

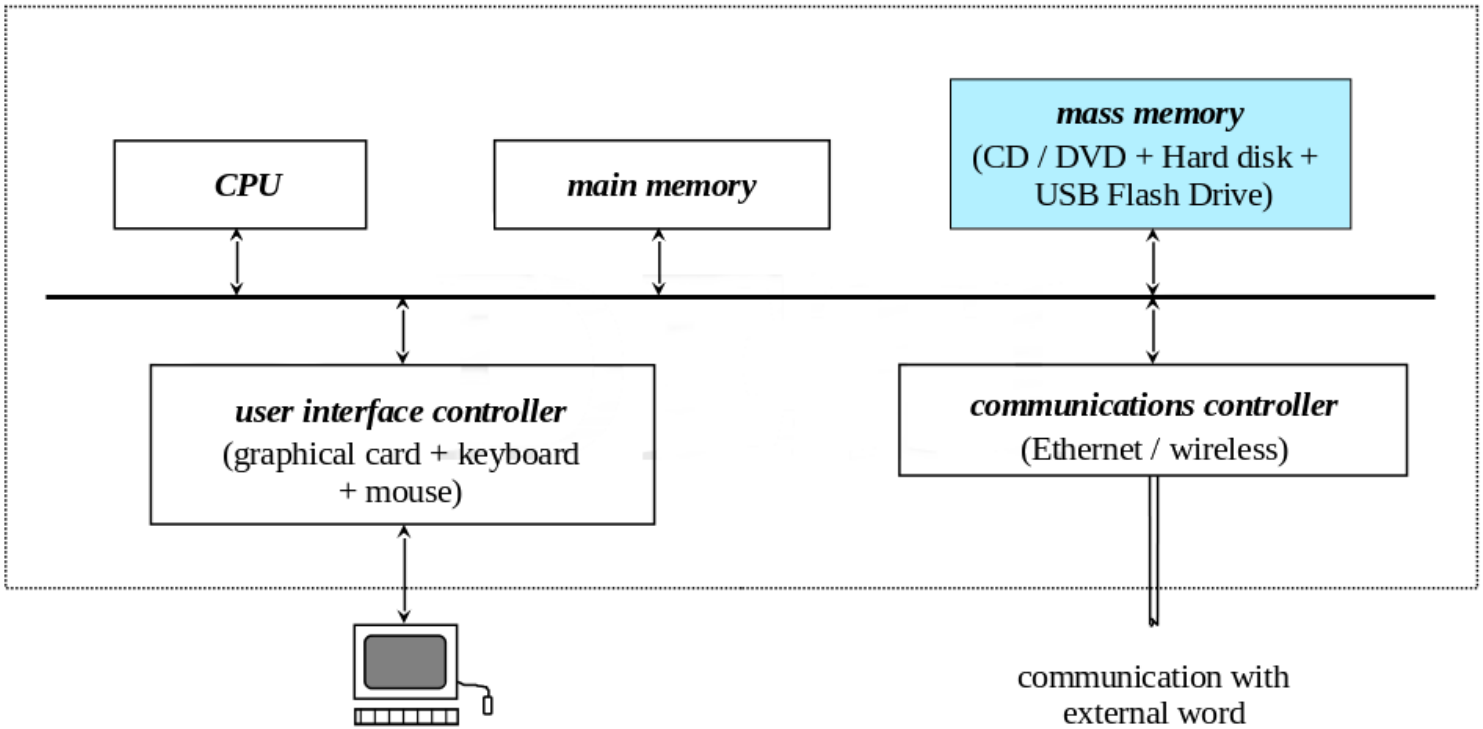
DETI / Universidade de Aveiro

Outline

1. Overview
2. Mass storage
3. The file concept
4. Inodes

Overview

- Simple view of a computational system, highlighting the mass memory component:
- File system is the part of the operating system responsible to manage access to mass storage



Mass storage
Types of mass storage devices

Type	Technology	Capacity (Gbytes)	Type of use	Transfer rate (Mbytes/s)
CD-ROM	mechanical / optical	0.7	read	0.5
DVD	mechanical / optical	4–8	read	0.7
HDD	mechanical / magnetical	250–4000	read / write	480
USB FLASH	semiconductor	2–256	read / write	60(r) / 30(w)
SSD	semiconductor	64–512	read / write	500

Mass storage
Operational abstraction of mass storage

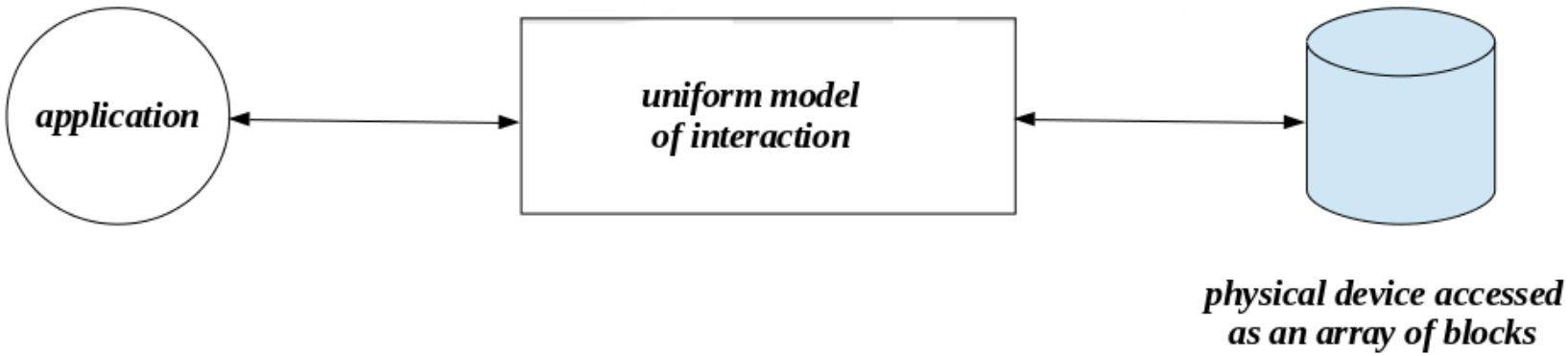
- Mass storage can be seen in operational terms as a very simple model
  - each device is represented by an array of NTBK storage blocks, each one consisting of BKSZ bytes (typically BKSZ ranges between 256 and 32K)
  - access to each block for reading or writing can be done in a random manner
- This is called Logical Block Addressing – LBA
  - Blocks are located by an integer index (0, 1, ...)
  - The ATA Standard included 22-bit LBA, 28-bit LBA, and 48-bit LBA

- Note that:
  - a block is the only unit of interaction
  - thus, a single byte **can not** be accessed directly

- What to do to change a byte of a block?

Mass storage
User abstraction of mass memory

- Some considerations:
  - Despite creating a uniform model, LBA is not an appropriate way for a user to access mass memory data
  - Direct manipulation of the information contained in the physical device can not be left entirely to the responsibility of the application programmer
  - Access must be guided by quality criteria, in termos of efficacy, efficiency, integrity and sharing
- Thus, a uniform model of interaction is required



- Solution: the **file concept**

File concept
What is a file?

- file is the logical unit of storage in mass memory
  - meaning that reading and writing information is always done within the strict scope of a file
  - Remember that physically the unit of interaction is the block

- Basic elements of a file:
  - identity name/path – the (generic) way of referring to the information
  - identity card – meta-data (owner, size, permissions, times, ...)
  - contents – the information itself, organized as a sequence of bits, bytes, lines or registers, whose precise format is defined by the creator of the file and which has to be known by whoever accesses it

- From the point of view of the application programmer, a file is understood as an abstract data type, characterized by a set of attributes and a set of operations

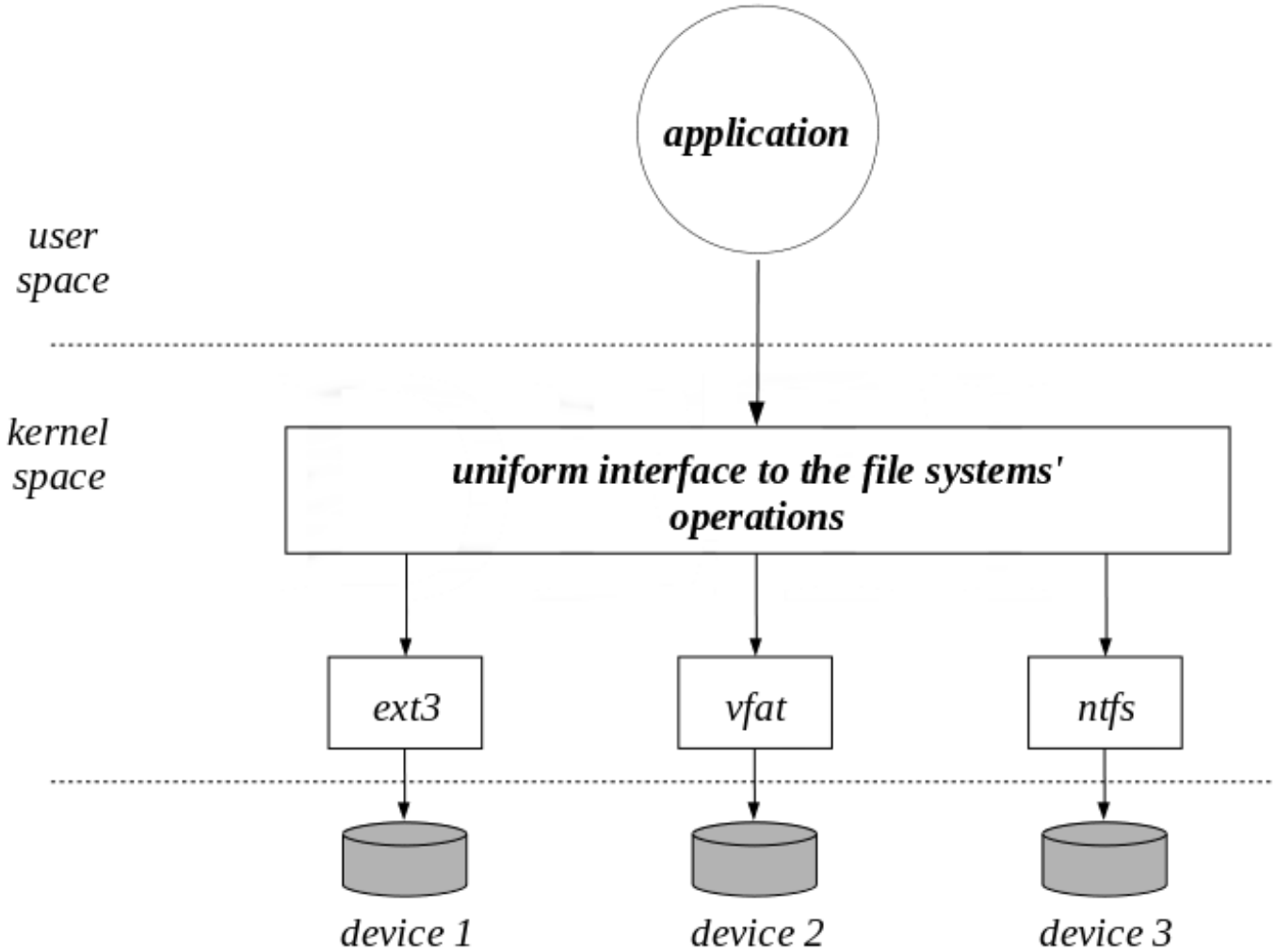
File concept
Role of operating system

- A role of the operating system is to implement this data type, providing a set of operations (system calls) which establishes a simple and secure communication interface for accessing the mass memory

- The file system is part of the operating system dedicated to this task
- Different implementations of the file data type lead to different types of file systems
  - Ex: ext3, FAT, NTFS, APFS, ISO 9660, ...

- Nowadays, a single operating system implements different types of file systems, associated with different physical devices, or even with the same
  - This feature facilitates interoperability, establishing a common means of information sharing among heterogeneous computational systems

File concept
Virtual file system



File concept
Types of files

- From the operating system point of view, there are different types of files:
  - ordinary/regular file – file whose contents is of the user responsability
  - directory – file used to track, organize and locate other files and directories
  - shortcut (symbolic link) – file that contains a reference to another file (of any type) in the form of an absolute or relative path
  - character device – file representing a device handled in bytes
  - block device – file representing a device handled in blocks
  - socket – file used for inter-process and inter-machine communication
  - named pipe – file used for inter-process communication

- Note that text files, image files, video files, application files, etc., are all regular files

File concept
Attributes of files

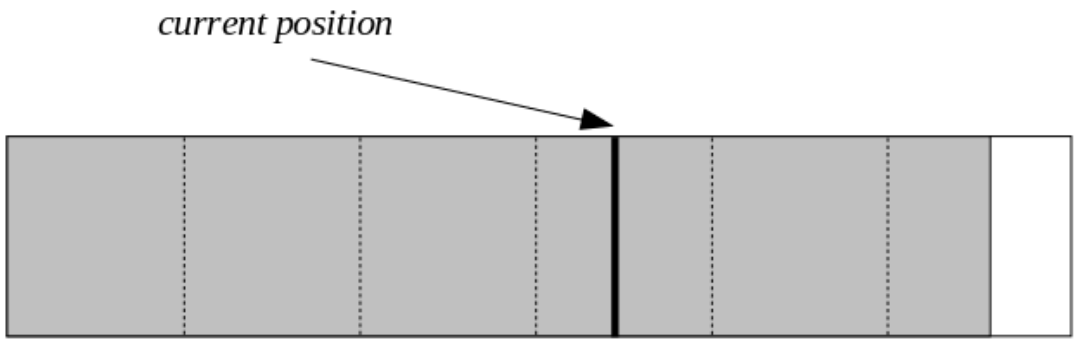
- Common attibutes of a file
  - type – one of the referred above
  - name/path – the way users usually refer to the file
  - internal identification – the way the file is known internally
  - size(s) – size in bytes of information; space occupied on disk
  - ownership – who the file belongs to
  - permissions – who can access the file and how
  - access and modification times – when the file was last accessed or modified
  - location of data in disk – ordered set of blocks/clusters of the disk where the file contents is stored

---

- Remember that a disk is a set of numbered blocks

File concept
Operations on files (1)

- Common operations on regular files
  - creation, deletion
  - opening, closing – direct access is not allowed
  - reading, writing, resizing
  - positioning – in order to allow random access



File concept
Operations on files (2)

- Common operations on directories
  - creation, deletion (if empty)
  - opening (only for reading), closing
  - reading (directory entries)
    - A directory can be seen as a set/sequence of (directory) entries, each one representing a file (of any valid type)

- Common operations on shortcuts (symbolic links)
  - creation, deletion
  - reading (the value of the symbolic link)

- Common operations on files of any type
  - get attributes (access and modification times, ownership, permissions)
  - change attributes (access and modification times, permissions)
  - change ownership (only root or admin)

File concept
Typical file operations on Unix

- As referred to before, the operations are based on system calls
- system calls common to any type of file
  - close, mknod, chmod, chown, stat, utimes, ...
- system calls on regular files
  - creat, open, link, unlink, read, write, truncate, lseek, ...
- system calls on directories
  - mkdir, rmdir, getdents, ...
- system calls on symbolic links
  - readlink, symlink, ...

- On a terminal execute `man 2 ≪syscall≫` to see a description

Inodes
What is an inode?

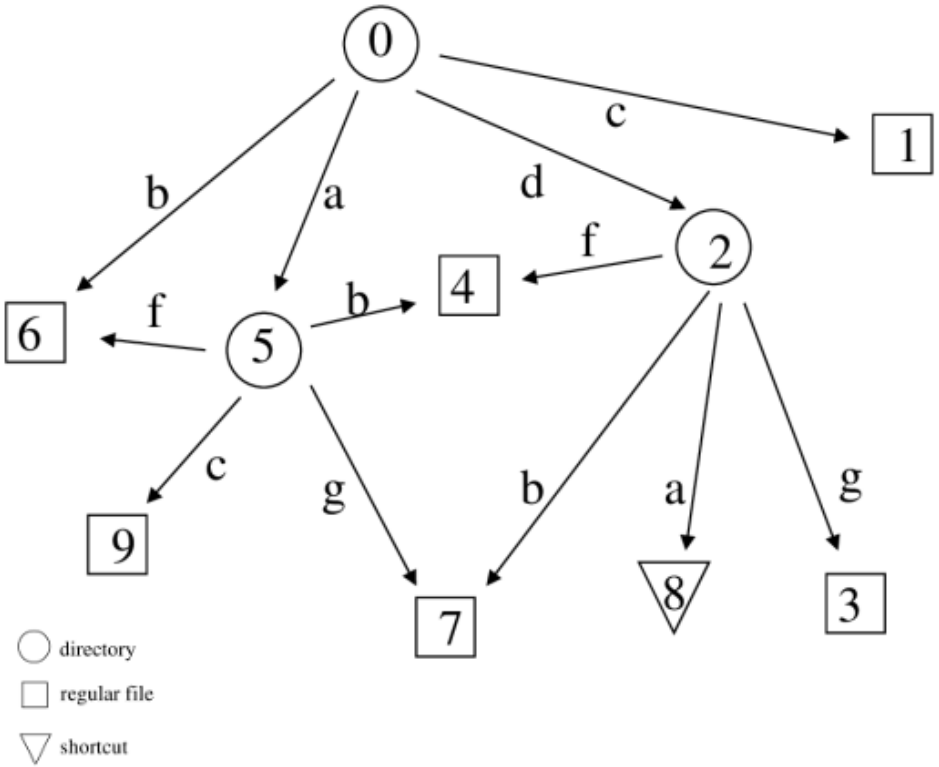
- In Unix, the inode (identification node) plays a central role in the implementation of the file data type
  - An inode is typically identified by an integer number

- It corresponds to the identity card of a file and contains:
  - file type
  - owner information
  - file access permissions
  - access times
  - file size (in bytes and blocks)
  - sequence of disk bocks with the file contents
- The name/path is not in the inode
  - it is in the directory entry

- disk inodes vs. in-core inodes

Inodes
Hierarchy of files

- Every file uses one and only one inode
- Same inode can have different pathnames
- Hierarchy of files may not be a tree



- The contents of a disk can be seen as a graph, where
  - Nodes are the files (directories, regular files, shortcuts, . . .), each one having an associated inode
  - Arrows define the hierarchy
- What is a directory?
- What is a link?
- What is a shortcut (symlink)?