

```
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
```

```
#define MAXPAROLA 30
#define MAXRIGA 80
```

```
int main(int argc, char *argv[])
```

```
{
    int freq[MAXPAROLA]; /* vettore di contatori
delle frequenze delle lunghezze delle parole */
    char riga[MAXRIGA];
    int i, inizio, lunghezza;
    FILE * f;
```

```
    for(i=0; i<MAXPAROLA; i++)
        freq[i]=0;
```

```
    if(argc != 2)
```

```
    {
        fprintf(stderr, "ERRORE: serve un parametro con il nome del file\n");
        exit(1);
    }
```

```
    f = fopen(argv[1], "r");
    if(f==NULL)
```

```
    {
        fprintf(stderr, "ERRORE: impossibile aprire il file %s\n", argv[1]);
        exit(1);
    }
```

```
    while( fgets( riga, MAXRIGA, f ) != NULL )
```

Linux File System

Linux File System

Stefano Quer - Pietro Laface

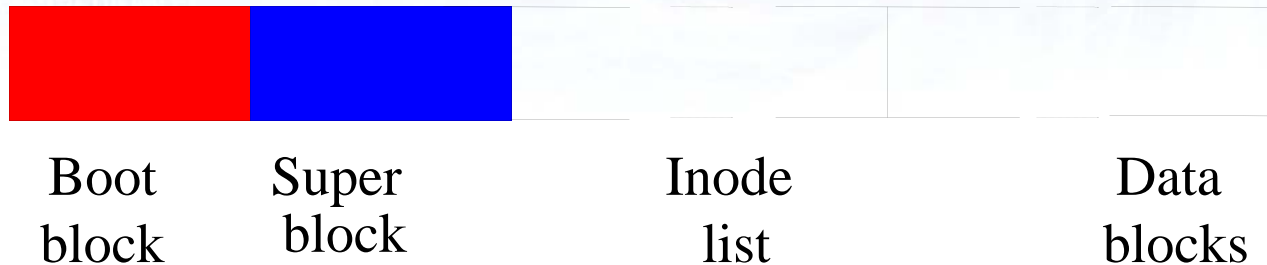
Dipartimento di Automatica e Informatica

Politecnico di Torino

File System

- ❖ The file-system is one of the most visible aspects of an operating system
- ❖ It provides mechanisms for permanent storing of data
- ❖ It includes the management of
 - File
 - Directories
 - Disks and disk partitions

File System structure



- ❖ The first block is the **boot** block. It contains the bootstrap code that is loaded and executed at system power on.
- ❖ The **superblock** describes the file system state:
 - size
 - how many files it can contain
 - free list (map) of Inodes and data blocks
 - other information

Inode = integer
numbered structure

Inode

- ❖ The **inode** is the **file descriptor**, which includes all information related to a file, excluding its filename
 - Owner
 - File type: *regular, directory, special* ,...
 - Access rights
 - Access times
 - Link number
 - File size
 - Table of the data block addresses on disk

Directory

- ❖ A **directory** is a special file that contains a list of filenames, each associated to the corresponding **Inode**

Filename	Inode
.	1234
..	75
a.c	21000

Inode example

Owner:	user1
Group:	group1
Type:	regular file
Access rights:	rwxr-xr-x
Access:	Oct 5 2016 h: 8:15
Modified:	Oct 5 2016 h: 10:30
Inode:	Oct 5 2016 h: 13:30
Size:	3050 bytes
Disk addresses (pointers)	

Fila data modified, not
its inode

Inode modified, not the
file data

Opening a file

```
FILE * fp1; int fd1;  
fp1=fopen("a.c", "r");  
fd1=open("a.c", O_RDONLY);
```

Notice the differences between
fopen and **open**

- ❖ To use a file you have to "open" it
 - Open is a system call
 - The request to open a file makes the kernel
 - Copy the inode of the file in **kernel memory**
 - Return to the caller an integer number, that is the file descriptor
 - The file descriptor is the handle of all other operations on the file (write, read, etc)

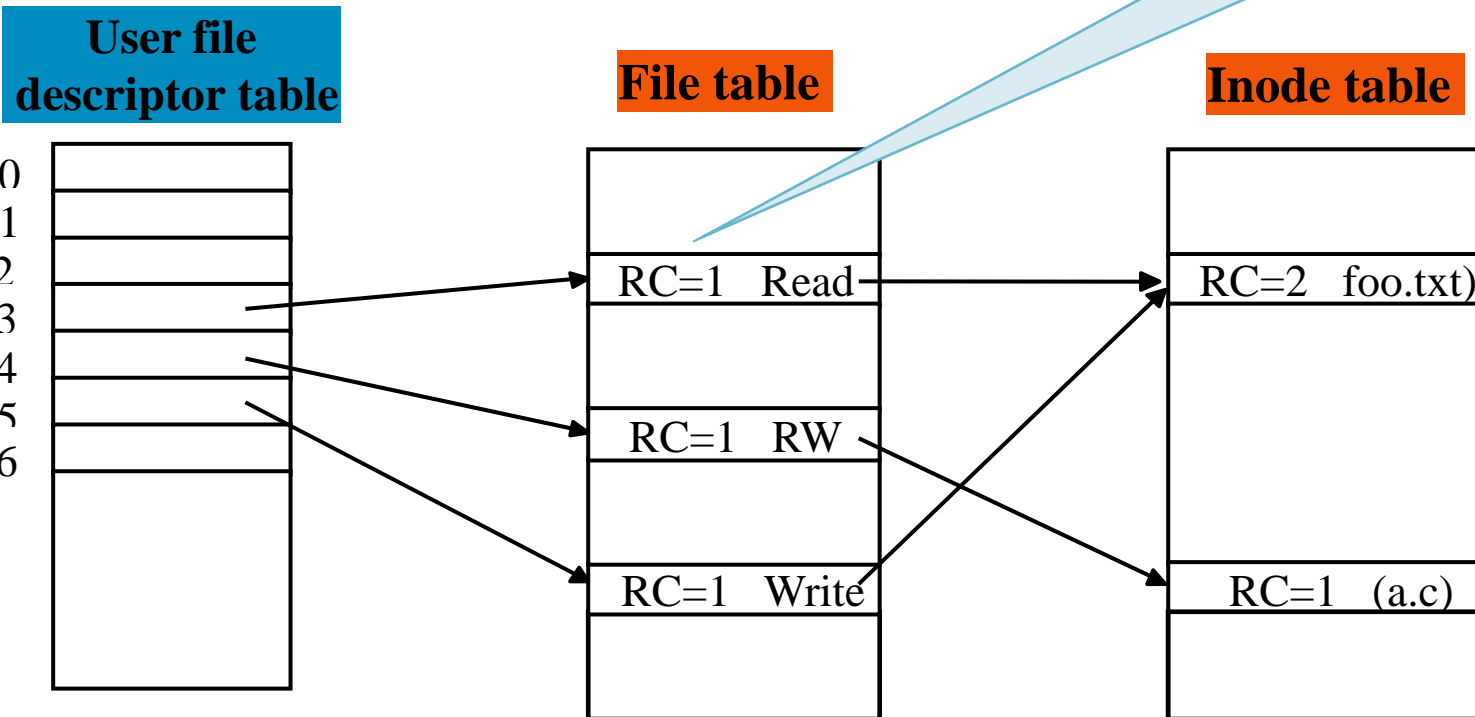
File System structures

```
fd1=open("foo.txt", O_RDONLY);
```

```
fd2=open("a.c", O_RDWR);
```

```
fd3=open("foo.txt", O_WRONLY);
```

RC = Reference Count
RW permissions



Resolving pathname ../a/b

