

# Le Directory

Le directory sono **file che tengono traccia degli altri file** all'interno di un file system.

## Sistemi di directory a livello singolo

Forma più semplice di sistema di directory, abbiamo una **sola directory principale (root)**, con all'interno tutti i file. Era comune nei primi PC e nel supercomputer CDC 6600.


Il vantaggio principale era nella **semplicità** e nella **rapidità nel localizzare** dei file.

### **Evoluzione dei file system**

Non dobbiamo pensare a questi primi file system come un qualcosa di obsoleto che non viene più usato.

Infatti la complessità di un file system dipende anche dalla tecnologia su cui viene applicato.

Per esempio in alcuni sistemi Embedded o su tecnologie RFID questo tipo di sistema potrebbe funzionare perfettamente.

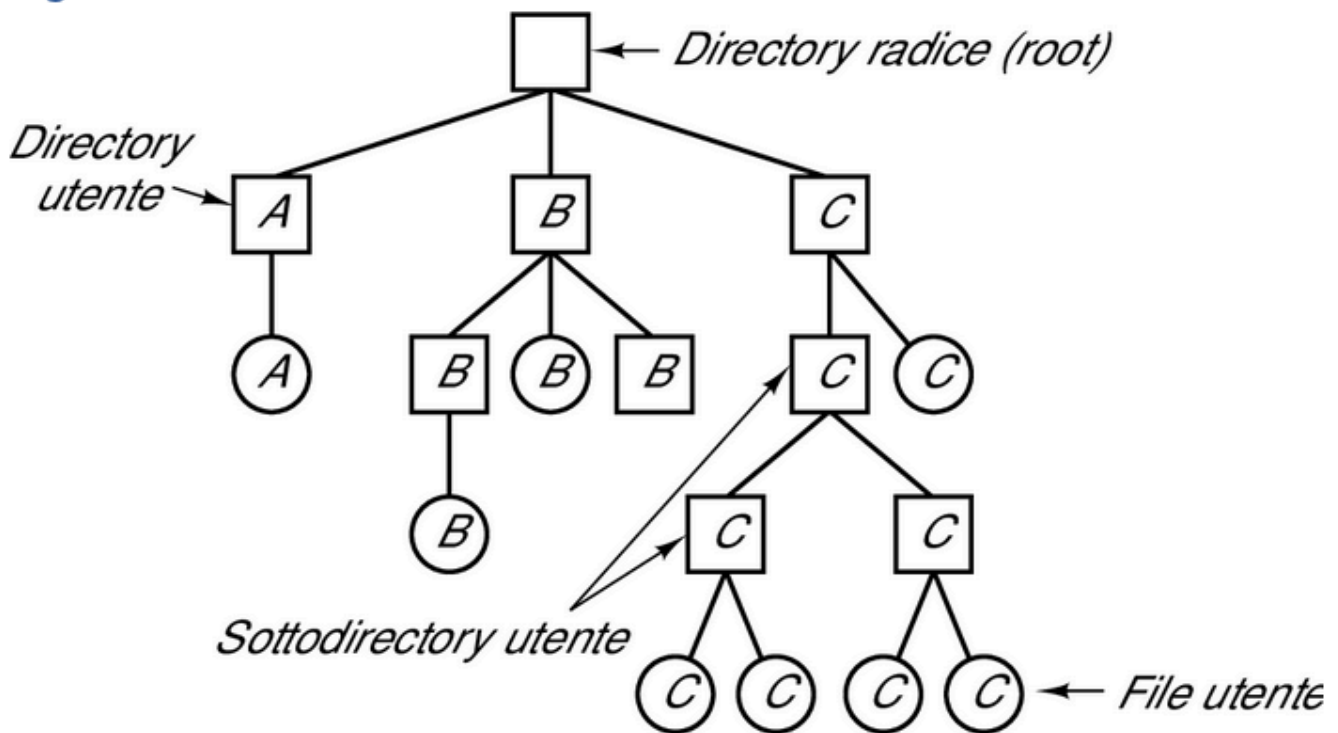
 **Idee apparentemente obsolete possono essere rilevanti in contesti moderni e dispositivi a basso costo.**

## Sistemi di directory gerarchici

Singolo livello -> **Semplici** applicazioni, ma non funziona se ci sono migliaia di file.

Serve creare una gerarchia nel quale ogni utente ha una directory principale privata.

**Figura 4.8**



Un sistema di directory gerarchico.

In questo esempio, le directory A, B e C appartengono a utenti differenti, i quali hanno creato delle **subdirectory** usate per organizzare i loro lavori. I file system moderni sono strutturati in questo modo.

### Un pò di storia

Questo modello di file system è una delle tante idee sperimentate da MULTICS negli anni '60.

## Nomi di percorso

Per trovare i file all'interno del file system dobbiamo poter assegnare un nome ad essi.

Esistono due metodi che vengono usati:

- **Nome di Percorso Assoluto:** Composto dal percorso che comincia la directory principale ed arriva al file.
- **Nome di Percorso Relativo:** Utilizza il concetto di **directory di lavoro**, in questo caso tutti i nomi di percorso che non cominciano con la directory principale sono considerati relativi a quella di lavoro.

Ogni processo ha la sua **working directory**, quando la cambia e in seguito esce, nessun altro processo è influenzato e nel file system non rimangono tracce del cambio.

Se una procedura di libreria cambia la directory di lavoro e non **ripristina la situazione** una volta terminata, il resto del programma potrebbe smettere di funzionare perché non si trova più dove crede di essere.

Esistono delle voci speciali in tutti i SO che operano su gerarchia:

- punto (.) -> Rappresenta la **directory corrente**.
- puntopunto(..) -> Rappresenta la **directory genitore**.

Vengono usati per muoversi all'interno dell'albero.

## Operazioni sulle directory

### Operazioni di Base:

- *Create* -> Creazione di una directory vuota.
- *Delete* -> Eliminazione di una directory possibilmente vuota.
- *Opendir* -> Apertura di una directory per la lettura del suo contenuto.
- *Closerdir* -> Chiusura di una directory dopo la lettura per liberare risorse.

### Lettura e Modifica:

- *readdir* -> Restituisce la prossima voce in una directory aperta senza esporre la struttura interna.
- *rename* -> Rinomina una directory, simile al rinomio di un file.

### Linking e Unlinking:

- *link* -> Crea un hard link, collegando un file esistente a un nuovo processo condividendone l'**l-node**.
- *unlink* -> Rimuove una voce di directory cancellando il file se è l'unico link.

### Link simbolici:

- Varianti dei hard link che possono puntare a file su dischi o computer diversi. Rappresentano un file tramite un riferimento indiretto che il file system risolve all'uso. Offrono **maggiore flessibilità**, ma possono essere meno efficienti rispetto agli hard link.