CH13 File-Sytem interface

File: address space logico contiguo.

- **Tipi**:
  - **Dati**
    - Numerico
    - Carattere
    - Binario
  - **Programma**
- Contenuto definito dal creatore del file:
  - Text file, source file, esecutable file

file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm, mp3, avi	binary file containing audio or A/V information

Attributi del file:

- Nome → only information kept in human-readable form
- Identificatore → unique tag (number) identifies file within file system
- Tipo → needed for systems that support different types
- Posizione → pointer to file location on device
- Dimensione → current file size
- Protezione → Controls who can do reading, writing, executing
- Tempo, data, identificazione tuente → data for protection, security, and usage monitoring

Operazioni su file:

- Creazione
- Cancellaione
- Scrivere alla posizione del write pointer
- Leggere alla posizione del read pointer
- Risposizionarsi nel file → seek
- Troncare
- Open(Fi) → cerca nella cartella sul disco una entry per Fi e sposta il contenuto della entry alla memoria
- Close(Fi) → sposta il contenuto della entry Fi dalla memroua alla directory sul disco

Open file: voglio lavorare sul file

- Quando un file è aperto, la sua entry è mantenuta in una tabella del S.O→ **open-file table**:
  - Tabella che mantiene la corrispondenza tra un numero e l'effettivo file
- Quando un file viene letto o scritto, si fa riferimento al **file pointer** o file displaichment:
  - Indica dove stai leggendo o scrivedo in un file
  - Solitamente si parte da un file vuoto e lo si scrive oppure si scrive in coda ad un file
    - In alcuni casi si modifica una parte di un file a caso
- **Contatore** di quanti hanno aperto quel file → **file-open count**: comodo che un file possa essere usato in concorrenza da più processi o più thread
- Dove sta il file sul disco, privilegi di accesso
- Quando i file sono aperti in concorrenza, servono delle forme di **lock**
  - Lock su un pezzo di file, non su tutto il file
    - In contemporanea si possono modificare diverse parti del file senza problemi
    - Due tipi di lock
      - **Shared** → simili di lock in lettura
        - Molti processi possono avere il lock contemporaneamente
      - **Esclusivo** → simili a lock in scrittura
        - Mandatory – access is denied depending on locks held and requested
        - Advisory – processes can find status of locks and decide what to do

Struttura del file:

- **Sequenza** di parole o byte
- Struttura record **semplice**
    - Linee
    - Lunghezza fissa (es: righe di 80 caratteri tutte)
    - Lunghezza variabile
- Strutture **complesse**
    - Documento formattato

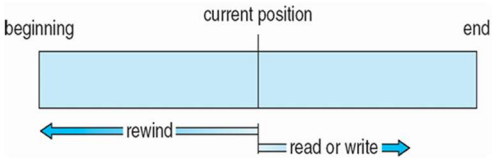
Nota: I file di testo, possono essere viste come due tipi di dati:

- Sequenza di parole, compresi gli a capo
- Record semplice a lunghezza variabile

ACCESSO **SEQUENZIALE**:

Si assume che stai leggendo/scrivendo dove sei

- Read next
- Write next
- Reset/rewind



ACCESSO **DIRETTO**:

Si inserisce una specie di indirizzo che mi dice dove devo andare
Due Tipologie diretto:

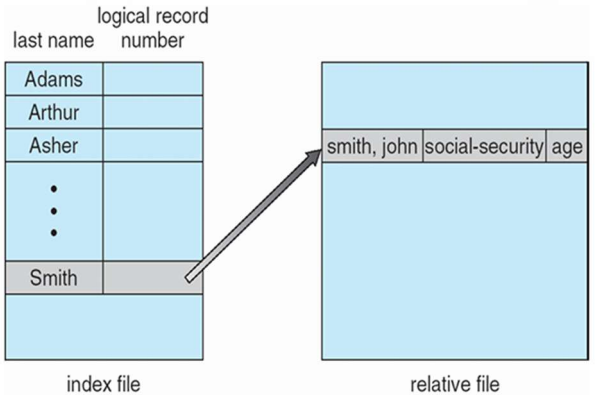
- **Vero**: Leggi o scrivi specificando il punto
    - **Read n**
    - **Write n**
    - **Rewrite n**
- **Simulato** usando sequenziale → Ti posiziono a n e poi ti faccio fare lettura/scrittura sequenziale
    - **Position to n**
        - **Read next**
        - **Write next**

n = relative block number

Ugualmente, si può avere il sequenziale usando il diretto.

sequential access	implementation for direct access
reset	*cp* = 0;
read next	*read cp*; *cp* = *cp* + 1;
write next	*write cp*; *cp* = *cp* + 1;

INDEX FILE: file organizzato con una parte che è strutturata come tabella di simboli e una parte ad accesso diretto

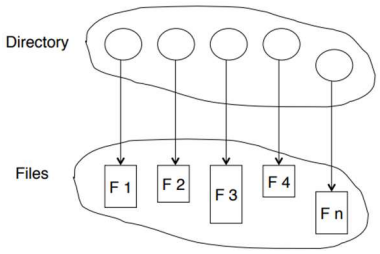


DIRETTORI

27/05/2024

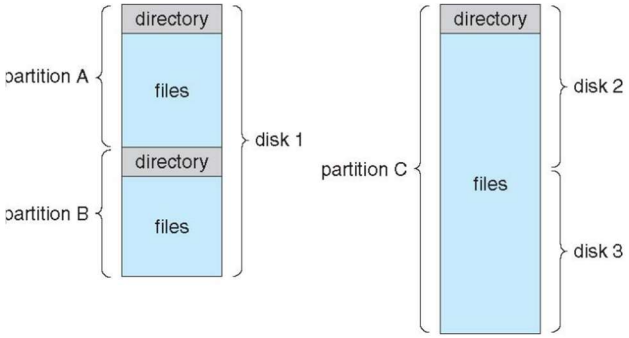
Collezione di nodi contenenti informazioni sui file

- Contenitore con puntatori a file



Struttura del disco:

- Disco può essere suddiviso **in partizioni**
  - Può essere in modalità raw
  - Può essere formattato con un file system
- A livello logico, ci sono i **VOLUMI**:
  - Unità in grado di contenere il file system:
    - Direttori
    - File



Nota: ogni filesystem ha la propria struttura su come divide spazio tra direttori e files

Tipi di file sistem:

- General purpose
- specializzati

Operazioni svolte sui direttori:

- ricerca di file
- creazione di file
- eliminazione di file
- lista dei direttori
- modifica del nome di un file
- attraversare il file system

Organizzazione:

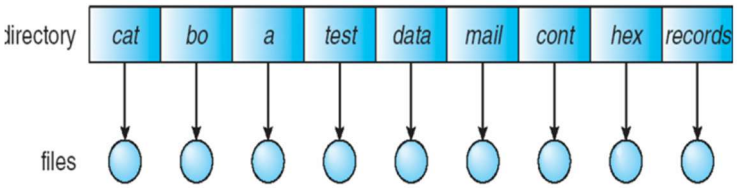
- efficiente nella ricerca
- strategie per i nomi
  - 2 utenti possono avere lo stesso nome su file diversi tra loro
  - Un file può essere raggiungibile con diversi nomi
- Grouping: raggruppamento logico di file in base alle proprietà (es: programmi java, giochi, ...)

Nota: ci possono essere più utenti su una macchina/elaboratore

TIPI di direttorio:

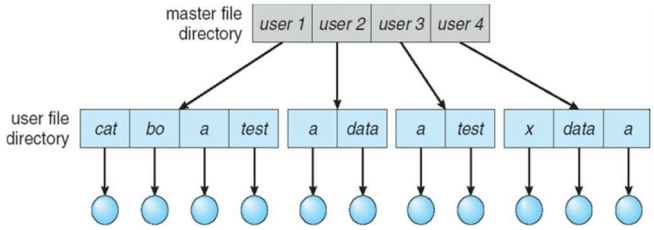
Direttorio di un solo livello:

- Singolo direttorio per tutti gli utenti
- Tanti file, nomi lunghi…
  - Non ci possono essere file con stesso nome



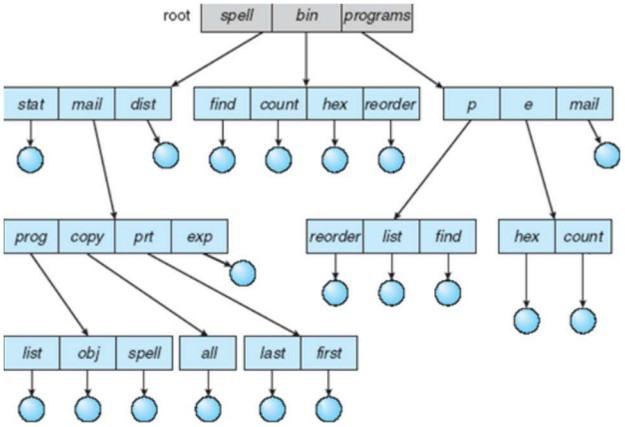
Direttorio per user (2 livelli):

- **Path name:** nome che contiene il cammino per raggiungere il file
  - Ricerca efficiente
- 2 user diversi possono avere file con lo stesso nome



Struttura ad albero:

- Ogni direttorio ha il proprio nome e il proprio significato
  - Nodi intermedi → direttori
  - Foglie → file
- Occorre realizzare una tabella di simboli
  - Gerarchica
  - Distribuita sui vari direttori



- **Absolute** or **relative** path name
- Creating a new file is done in current directory
- Delete a file

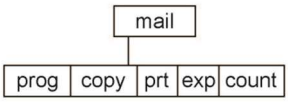
```
rm <file-name>
```

- Creating a new subdirectory is done in current directory

```
mkdir <dir-name>
```

Example: if in current directory `/mail`

```
mkdir count
```

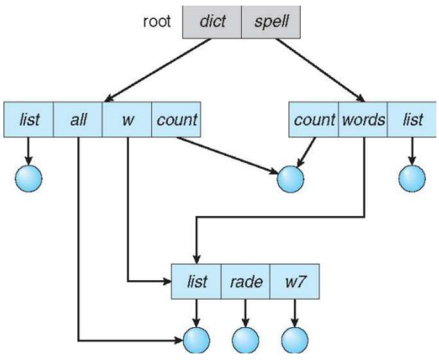


Deleting "mail" ⇒ deleting the entire subtree rooted by "mail"

Nota: *la cancellazione di un direttorio avviene solo quando è vuoto*

Può essere conveniente poter chiamare un **file con 2+ nomi diversi**

- Si arriva allo stesso file da due cartelle diverse
- **File condiviso**



In caso di **cancellazione di un file**, 2 possibilità**:**

- Elimino anche gli altri miei nomi (backpointers)
  - File deve avere la lista dei direttori (backpointers) in modo da cancellare i pointer pendenti
- Mantengo gli altri, eliminando solo il mio nome (entry-hold-count)
  - Usi un contatore di puntatore
    - Ogni cancellazione, decrementi puntatore
    - File cancellato solo quando l'ultimo che punto decide di cancellarlo.

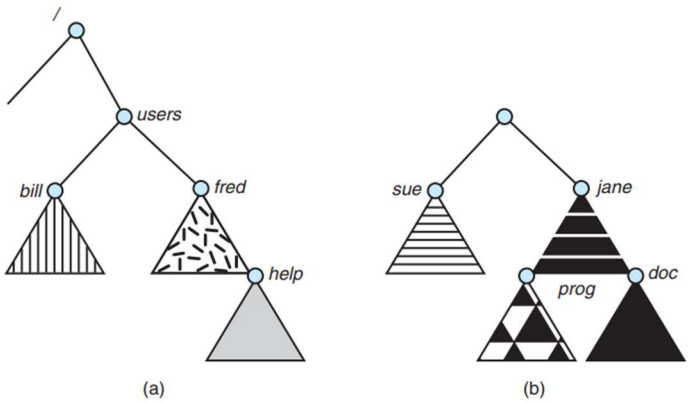
Nota: in un filesystem esistono cartelle e file condivisi
Nota: si possono condividisere sia le foglie che i direttori (list)

Creazione file:

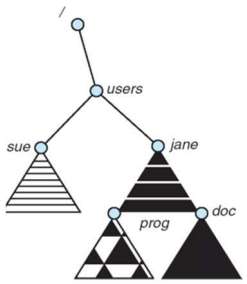
- Bisogna vedere se c'è già un file esistente con quel nome.
- Sì → si aggiunge
- No → si crea
- Bisogna evitare la creaione di cicli, 3 soluzioni:
    - Non condividere direttori ma solo i file
    - Garbage collection:
        - Si accettano cicli, ogni tanto si controlla tutto e si segnala l'errore
    - Check dei cicli ogni volta che viene creato un file , si controlla cose è raggiunigbile a partire della funzione che testo.

Direttori e file sono su disco ma c'è l'operazione mount che aggancia il file system che sta sul disco al sistema in esecuzione:

- Quando avviene mount, il file system dsa disco, diventa un sotto-albero che parte da radice /



a) Mounted
b) Non mounted, viene mounted ad un punto di mount



La condivisione di faile può essere fatta seguendo uno schema di protezione
l **sistema di protezione** si basa su:

- Utente → UserID
- Gruppo → GroupID

Inoltre si ha:

- Owner file/directory
- Gruppo di file/directori

Cidivisione di file remoti → condivisione tramite rete

- Esistono metodi manuali con cui esplicitamente si scarica/chiede accesso/modifica un file
- Ci sono anche stategie per gestire gli errrori

Nota: quando c'è file sharing biosogna fare attenzione alla consistenza dei file.

PROTEZIONE

Owner dovrebbe essere in grado didecide chi può fare cosa:

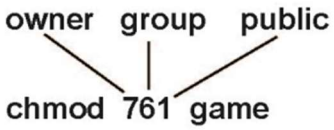
- Lettura
- Scriuttura
- Esecuzione
- Append
- Delete
- Lista

Schema usato per decidere se operazione puo rssrt fatta o no:ù- access ly

- Mode of access: read, write, execute
- Three classes of users on Unix / Linux

			RWX
a) **owner access**	7	⇒	1 1 1
b) **group access**	6	⇒	RWX 1 1 0
c) **public access**	1	⇒	RWX 0 0 1

- Ask manager to create a group (unique name), say G, and add some users to the group.
- For a particular file (say *game*) or subdirectory, define an appropriate access.



Attach a group to a file

```
chgrp      G      game
```

Decimo bit, indica se è un direttorio o un file (ultima slide)