Visione d’insieme

I file(s)

I files sono l’elemento principale per la maggior parte delle applicazioni, nella maggior parte dei casi infatti un file è proprio l’input dell’applicazione, ma altrettanto spesso anche l’output è un file.
In particolar modo i file sono importanti in quanto questi sopravvivono anche dopo la morte dei processi (a differenza della RAM del processo che invece viene sovrascritta)

Dunque si può dire che il file system è una delle parti del sistema operativo che sono più importanti per l’utente

Queste sono le proprietà che sono richieste per rendere i file più usufruibili possibile dall’utente:

- esistenza a lungo termine
- condivisibilità con altri processi (tramite nomi simbolici)
- strutturabilità (directory gerarchiche)

Gestione dei file

I file sono gestiti da un insieme di programmi e librerie di utilità all’interno del kernel. I programmi sono comunemente conosciuti come il *File (Management) System* e ovviamente vengono eseguiti in kernel mode. Le librerie, invece, vengono invocate come system call (sempre in kernel mode)

Questi programmi e librerie hanno a che fare con la memoria secondaria (dischi, chiavi USB, …). Particolarità sta nel fatto che in Linux, selezionate porzioni di RAM, possono essere gestite come file. Forniscono infatti un’astrazione sotto forma di operazioni tipiche (creazione, modifica, etc.) e per ogni file vengono mantenuti degli attributi (o metadati), come proprietario, data creazione, etc.

Operazioni tipiche sui file

Le operazioni standard eseguibili su un file sono:

- Creazione (con annessa scelta del nome, al momento della creazione il file è tipicamente vuoto)

- Cancellazione
- Apertura → necessaria per poter leggere e scrivere
- Lettura → solo sui file aperti (e non chiusi nel frattempo)
- Scrittura → solo sui file aperti (e non chiusi nel frattempo)
- Chiusura → necessaria per le performance

Terminologia

Introduciamo delle terminologie comuni (seppur datata), introduciamo quindi:

- **Campo** (field)
- **Record**
- **File**
- **Database**

Campi e Record

Campi

I campi sono i valori di base, contengono infatti dei valori singoli. Sono caratterizzati da una lunghezza e dal tipo di dato (con demarcazioni). Esempio tipico: carattere ASCII

Record

Mettendo insieme campi diversi (ma correlati) ottengo un record ed ognuno di essi viene trattato come un'unità. Esempio tipico: un impiegato è caratterizzato dal record nome, cognome, matricola, stipendio

File e Database

File

Mettendo insieme record ottengo un file (nei SO generici moderni, ogni record è un solo campo con un byte, stream di byte). Ognuno è trattato come un'unità con nome proprio e possono implementare meccanismi di controllo dell'accesso (alcuni utenti possono accedere ad alcuni file, altri ad altri)

Database

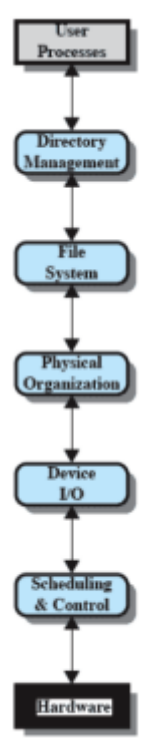
Collezioni di dati correlati formano un database. Questi mantengono anche relazioni tra gli elementi memorizzati e sono realizzati con uno o più file. Sono inoltre gestiti dai DBMS, che sono tipicamente processi di un sistema operativo

---

Sistemi per la gestione dei file

File system

Riguarda i dispositivi di archiviazione (es. HDD, SSD, CD, DVD, floppy disk, USB, …)



Qui il logical I/O è tipicamente diviso in tre parti
Directory Management → fornisce tutte le operazioni atte a gestire i file (creare, spostare, cancellare, …); da nomi di file a identificatori di file
File system → struttura logica ed operazioni (apri, chiudi, leggi, scrivi, …)
Organizzazione fisica → si occupa di allocare e deallocare spazio su disco (quando ad esempio si chiede di creare un file); da identificatori di file a indirizzi fisici su disco
Scheduling & Control → ovviamente è qui che ci sono i vari `SCAN`

In questa parte ci occuperemo di analizzare questi tre blocchi.
I programmi del **File Management System** forniscono servizi agli utenti e alle applicazioni per l'uso di file e definiscono anche il modo in questi sono utilizzati, sollevando i programmatori dal dover scrivere codice per gestire i file

---

Obiettivi per i File Management System

Come sempre ci sono degli obiettivi per il SO da dover raggiungere, che sono:

- rispondere alle necessità degli utenti riguardo la gestione dei dati (creazione etc.)
- garantire che i dati nei file sono validi (a mano che non si rompa il supporto hardware)
- ottimizzare le prestazioni (sia dal punto di vista del SO, *throughput*, che dall'utente, tempo di risposta)
- fornire supporto per diversi dispositivi di memoria secondaria (dischi magnetici, chiavi USB, etc.)
- minimizzare i dati persi o distrutti
- fornire un insieme di interfacce standard per i processi utente
- fornire supporto per l'I/O effettuato da più utenti in contemporanea

---

Requisiti per i File Management System

1. Ogni utente dev'essere in grado di creare, cancellare, leggere, scrivere e modificare un file
2. Ogni utente deve poter accedere, in modo controllato, ai file di un altro utente
3. Ogni utente deve poter leggere modificare i permessi di accesso ai propri file
4. Ogni utente deve poter ristrutturare i propri file in modo attinente al problema affrontato
5. Ogni utente deve poter muovere dati da un file ad un altro
6. Ogni utente deve poter mantenere una copia di backup dei propri file (in caso di danno)
7. Ogni utente deve poter accedere ai propri file tramite nomi sinbolici