Quesito

Discutere dell'importanza dei file, delle tipologie di file e della loro implementazione.

Risposta

In UNIX everything is a file! Un file è un astrazione del disco andando a risolvere il problema della memorizzazione, consentendo la persistenza, l' accesso multiplo e la gestione di grandi volumi di dati. Fornisce un metodo per scrivere e leggere informazioni dal disco. Ogni file ha un suo nome, la quale struttura può essere diversa in diversi file system. Per esempio in FAT 12/16/32 i nomi dei file non erano case sensitive, ovvero Maria era uguale a maRia. Oltre al nome, i file hanno anche un estensione, un'ulteriore informazione riguardo al file, per esempio file C hanno estensione .c e cosi via. In UNIX le estensioni sono puramente convenzionali, ovvero al sistema operativo non interessa sapere il contenuto di un file. Ma un compilatore C, può richiedere che il file abbia estensione .c. In Windows, invece, le estensioni hanno un ruolo ben preciso, poiche l'utente può assegnare ad un file un applicazione con cui aprire il file.

In UNIX e Windows i file sono una sequenza non struttura di byte, quindi è compito del'applicazione utente capire il contenuto all'interno di un file. I file possono essere binari o ASCII. I file ASCII sono file di testo normali mentre i file binari sono i file eseguibili o gli archivi. Ciascun file, al suo interno contiene un numero "magico" per specificare la tipologia di file, se binario o ASCII.

L'implementazione dei file può variare da file system a file system. In generale possiamo avere:

- Allocazione contigua: è la forma più semplice per implentare i file, poiche sono allocati in modo contiguo sul disco. E' facile da implementare, inoltre è efficente nella lettura in quanto non bisogna ruotare il braccio del disco, quindi facilita la lettura sequenziale ma non casuale, porta a grande frammentazione del disco.
- Allocazione a linked list: ciascun file è diviso in una lista concatenata, dove ciascun nodo è un blocco che contiene i dati del file. Quindi ciascun nodo è composto dal blocco che contiene i dati, la dimensione, un puntatore al prossimo blocco e nel caso si usasse una doppia linked list, anche un puntatore al blocco precedente. Rispetto all'allocazione contigua migliora la distribuzione sul disco, in quanto un file si puo trovare in parti diverse del disco, migliorando quindi anche la frammentazione. L'unica pecca è che i blocchi non possono essere di dimensione a potenze di 2, in quanto servono alcuni byte per memorizzare il puntatore, rendendo meno efficace la lettura. L'accesso casuale è molto lento, in quanto bisogna scorrere sempre tutta lista.
- Allocazione con FAT: E' esattamente l'allocazioene con linked list, l'unica differenza è che i puntatori sono tenuti in memoria principale in una FAT (File Allocation Table), migliorando cosi l'accesso causale e risolve lo svantaggio precedente, ovvero di non avere blocchi di dimensione a potenze di 2, in quanto adesso i puntatori sono tutti in RAM. Ad oggi viene solamente usato in dispositivi di archiviazione di piccole dimensioni come USB-drive o macchine fotografiche, in quanto per un TB di disco c'e bisogno di circa 3 GB di RAM.
- I-node: Sono una struttura dati, che contiene tutti gli attributi di un file, tranne nome e contenuto. Il contenuto è allocato in blocchi su disco, e i puntatori a questi blocchi si trovano all'interno dell'I-node. Quando viene creato un file, il suo nome punta all'I-node presente su disco, e quando viene aperto, in memoria viene caricato l'I-node. Quindi in memoria saranno presenti solo gli I-node dei file aperti, e questo è un grande vantaggio.