LEZIONE 12

Realizzazione dei file system

Struttura del file system

* Struttura del file
* Unità di memorizzazione logica
* Raccolta di informazioni correlate
* **Il file system risiede permanentemente nella memoria secondaria (permanente perché anche in mancanza di alimentazione continua ad avere la propria validità)**
* **Il file system ha una struttura stratificata**
* **Blocco di controllo dei file**

(**FCB,** file control block), struttura di memorizzazione che contiene informazioni sui file, come la **proprietà, i permessi e la posizione del contenuto**)

File system stratificato

Immagine che contiene screenshot

Descrizione generata automaticamente**spiegazione**

ci sono questi livelli che possono definire vere e proprie visioni diverse, ognuna di esse ha una caratteristica. Partendo dall’alto verso il basso, si passa da una visione di alto livello (molto compatta), a visioni man mano sempre meno compatte. Infatti man mano che si scende si troveranno delle primitive in grado di gestire il file system, con una metodica però sempre più bassa, si faranno delle funzioni e dei formalismi sempre più semplici, che sono più vicini a quelle che sono i criteri richiesti per il loro trattamento.

* Questa stratificazione è molto importante, perché **permette sia agli sviluppatori di intervenire livello per livello, e permette una versatilità migliore, dato che per ogni livello c’è la possibilità di operare sul livello sottostante, senza dover conoscere il modo con cui il livello sottostante realizza delle funzioni, ma semplicemente richiamando delle funzioni, e questa cosa comporta un grande livello di versatilità.**

Tipico descrittore di file

**Immagine che contiene screenshot

Descrizione generata automaticamente Spiegazione**

Questo blocco informativo contiene tutte le informazioni del file a cui fa riferimento.

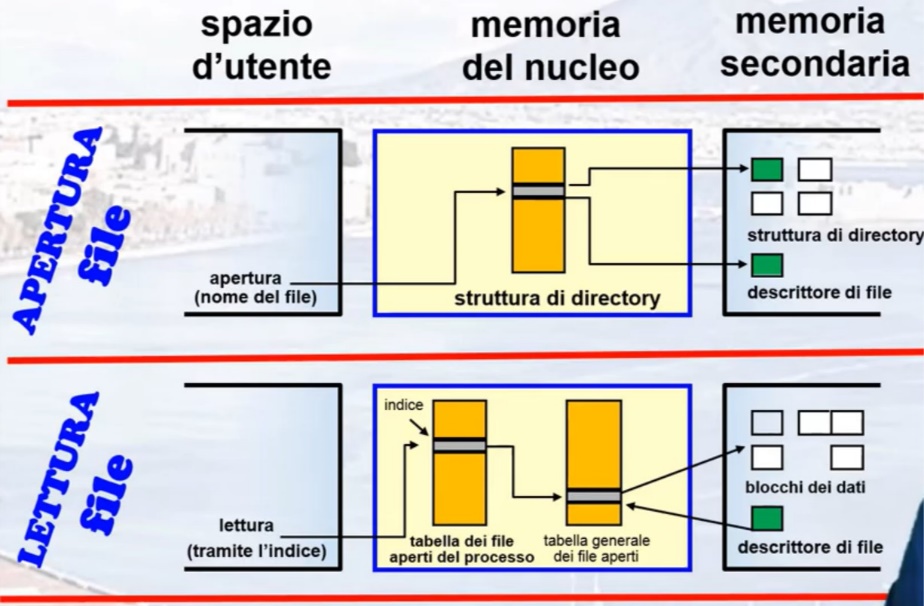
ACL (sta nella riga proprietario) significa access control list.

Struttura del file system che si mantengono nella memoria

Sono una parte minima rispetto tutto il file. In realtà il file viene salvato nella memoria secondaria, ma quando bisogna accedere a esso, bisogna valutare strategie che caricano nella memoria principale solo alcune parti di file (quelle che servono), **soprattutto i file indice**.

**File indice:** sono dei file che racchiudono e indicano dove si trovano i dati di un file (ragionamento simile alla tabella delle pagine)

Anche in questa circostanza si ha una stratificazione:

**spiegazione**

A sx c’è lo spazio utente che vede semplicemente la necessità di effettuare l’apertura di un file, oppure di fare una lettura di un file. Queste operazioni vengono poi gestite dalla memoria del nucleo, la quale si interfaccia anche con la memoria secondaria, dalla quale recupera la struttura delle cartelle e il descrittore del file. Nel momento in cui l’utente vuole leggere (figura sotto), vengono recuperate dal descrittore le informazioni che permettono di puntare ai blocchi dei dati del file stesso.

File system virtuali

Sono file system che permettono la visione di un file system che è molto di alto livello; nel senso che non devono costringere l’utente a conoscere su quale tipo di device per esempio i file risiedono.

* Il **file system virtuale** (VFS, virtual file system) fornisce una tecnica object-oriented per la realizzazione del file system.
* Separa le operazioni generiche del file system dalla loro realizzazione definendo un’interfaccia VFS uniforme.
* Il VFS è basato su una struttura di rappresentazione dei file detta **vnode** che contiene un indicatore numerico unico per tutta la rete di ciascun file.

**Immagine che contiene segnale, orologio

Descrizione generata automaticamenteSchema di un file system virtuale**

**Spiegazione**

Ad un livello molto elevato, c’è un’interfaccia file system che comunica con i file system veri e propri (sono i 3 rettangoli sotto) grazie al VFS. Inoltre all’interfaccia del file system non interessa sapere questi oggetti dove risiedono (infatti 2 sono su disco e una su rete); sarà la VFS che permette di unificare le 3 interfacce diverse, in modo tale da semplificare l’accesso da parte degli utenti (interfaccia del file system).

Realizzazione delle directory

Il metodo più facile è quello di sviluppare una lista lineare.

* **Lista lineare** contiene i nomi dei file con puntatori ai blocchi di dati.
* Metodo di facile programmazione
* Esecuzione onerosa in termini di tempo
* **Tabella hash: lista lineare con struttura di dati hash.**
* diminuisce notevolmente il tempo di ricerca nella directory
* collisioni: situazione in cui da due nomi si file si ottiene un riferimento alla stessa locazione
* dimensione fissa

Metodi di assegnazione

Un metodo di assegnazione indica il modo in cui i blocchi di disco vengono assegnati ai file.

Ci sono 3 metodi:

1. **Assegnazione contigua**

* Ciascun file deve occupare un **insieme di blocchi contigui** nel disco. Questo metodo è molto semplice dato che basta conoscere solo l’indirizzo del primo blocco (inteso come numero di blocco), e quanti blocchi servono per contenere l’intero file.
* Questo metodo però ha un grosso limite/**svantaggio**, ovvero, **l’impiego dello spazio,** perché siccome la contiguità richiede che i blocchi siano uno dopo l’altro, se tale condizione non c’è in memoria, non si può inserire tale file.
* **il vantaggio** è: nel momento in cui c’è la contiguità per inserire il file, l’accesso (gestione) è molto rapida dato che è compatto e quindi non bisogna andare a cercare i vari pezzi sparsi in memoria.

**Immagine che contiene orologio

Descrizione generata automaticamenteEsempio**

**Spiegazione**

Ci sta un disco e una cartella. Nella cartella ci sono le informazioni dei file. Il primo file inizia a 0 e ha grandezza , quindi sul disco si prende 0-1 e cosi poi anche gli altri.

1. **Assegnazione concatenata**

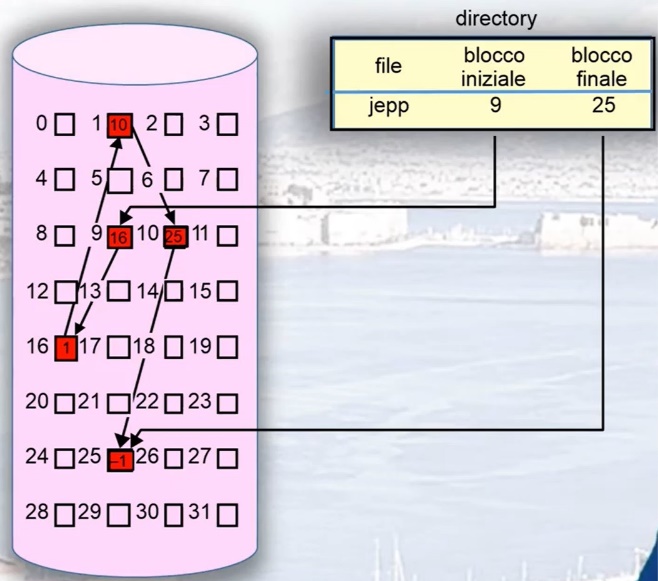
Immagine che contiene screenshot

Descrizione generata automaticamenteSi ha la possibilità di separare il contenuto di un file su tutta la memoria, perché si utilizza una sorta di linked list (lista concatenata), che è rappresentata dai blocchi dei dati del file, i quali sono contenuti laddove c’è spazio nella memoria. Quindi da un blocco si punta ad un altro blocco fino a quando non si è recuperato tutto il file. Infatti come si vede il blocco non ha solo l’area dati, ma contiene anche un’informazione per il puntamento del blocco successivo.

**Vantaggio:** con questo metodo si ha la possibilità di gestire in maniera semplice il discorso riguardante lo spazio appartenente ad un file, dato che si ha bisogno solo di un indirizzo di partenza. **Quindi a differenza del metodo precedente, qui non c’è uno spreco di spazio.**

**Svantaggio:** metodo più complesso, dato che bisogna fare una serie di “salti” per recuperare tutte le informazioni relative al file. **Inoltre non vi è un accesso casuale,** cioè non si può accedere a punti specifici, se no seguendo il percorso dei vari puntatori nei blocchi.

* Una variante di importante del metodo di assegnazione concatenata consiste nell’uso della tabella di assegnazione del file **(FAT, file allocation table).** Tale metodo di assegnazione, semplice ma efficiente, dello spazio dei dischi è usato nei sistemi operativi MS-DOS-OS/2.

**Esempio**

**Spiegazione**

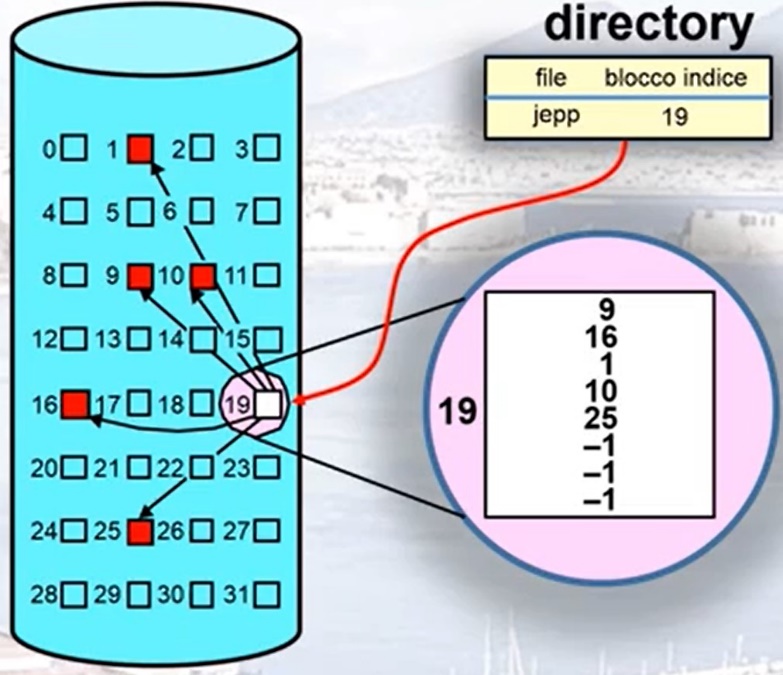
C’è il file che comincia al blocco 9, quindi si inizia da quel blocco, al cui interno ci sono alcune informazioni e il puntatore al blocco successivo, e così via fin quando non si recuperano tutti i blocchi, e di conseguenza ricavato tutto il file. Si può anche non conoscere il blocco finale dato che si capisce di aver finito quando in un blocco si trova ad esempio -1 (null), però così può capitare di non accorgersi di un errore (se non ci sta il blocco finale segnato).

1. **Assegnazione indicizzata**

Immagine che contiene pensile, computer, luce

Descrizione generata automaticamenteSi ha l’accesso random ai pezzi del file. Infatti si ha una tabella che raggruppa tutti i puntatori ai vari blocchi di dati.

**Esempio**

**spiegazione**

C’è una cartella con blocco indice 19. Questo blocco non contiene informazioni sul file, ma contiene tutti i puntatori ai blocchi che costituiscono l’intero file. **Questi blocchi indice sono tutti la stessa grandezza, cioè sono grandezza fissa** (infatti qui ci sono al suo interno tre -1, perché i puntatori sono pochi).

**Vantaggio:**

* **accesso casuale,** cioè permette un accesso rapido a punti diversi di file.
* **accesso dinamico** senza frammentazione esterna.
* Ogni file deve avere un blocco indice, quindi è auspicabile che questo sia quanto più piccolo possibile; ma **se il blocco indice è troppo piccolo, non può contenere un numero di puntatori sufficienti per un file di grandi dimensioni,** (stesso discorso vale pure se è troppo grande), quindi è necessario disporre di un meccanismo per la gestione di questa situazione:

1. **Schema concatenato**
2. **Indice a più livelli**
3. **Schema combinato**

**Esempio Schema concatenato UNIX**

**Immagine che contiene screenshot

Descrizione generata automaticamentespiegazione**

Questo blocco grande rappresenta l’informazione del file. Dove c’è scritto blocchi diretti è un blocco di indirizzi, e sono blocchi diretti, quindi se il file è piccolo, si può accedere subito ai blocchi dati. Se invece il file è troppo grande, allora si utilizza un indiretto singolo/ doppio e cosi via a seconda di quando è grande il file. L’indiretto singolo non ha un accesso diretto ai dati, ma accede ad un’altra tabella che contiene i puntatori ai dati. Indiretto doppio invece punta ad un blocco che a sua volta punta ad un altro blocco i cui indirizzi puntano ai dati, e cosi via con gli indiretti tripli ecc.

**Vantaggi:** molto versatile in quanto in un unico contesto sta utilizzando diverse strategie.

Gestione dello spazio libero

Il modo più semplice di gestire spazi liberi è quello di **immagazzinare un vettore di bit**, dove ogni bit rappresenta un blocco. **Se il bit vale 1 allora il blocco è libero, se vale 0 viceversa.**

**Esempio**

Un disco che ha liberi i seguenti blocchi: 2, 3, 4, 8, 9, 10, 11, 12, 13, 17, 18, 25, 26, 27

E gli altri sono occupati. La mappa di bit dello spazio libero è: 00**1111**00**111111**000**11**000000**111**0000

In realtà quando serve uno spazio libero si vuole sapere subito dov’è. Quindi bisogna sapere dov’è il primo 1 nella mappa. Tale bit lo si trova con questa formula:

**(numero di bit per parola) x (numero di parole di valore 0) + scostamento del primo bit 1**

**Vantaggi:** sono la sua relativa semplicità ed efficienza nel trovare il primo blocco libero o **n** blocchi liberi consecutivi nel disco.

**Svantaggi:** questo vettore di bit per poter essere gestito deve essere completamente contenuto nella memoria centrale, quindi bisogna anche essere capaci d’ipotizzare un altro metodo, ovvero, **la lista concatenata dei blocchi liberi.** Tale lista funziona così: si comincia con un puntatore a un primo blocco libero e oltre a fornire spazio, fornisce anche il link al prossimo blocco e cosi via fino alla fine.

Efficienza e prestazioni

* **L’efficienza di pende da:**
* Algoritmi usati per l’assegnazione del disco
* Gestione delle directory
* **Prestazioni:**
* **Cache del disco:** sezione separata della memoria centrale dove tenere i blocchi in prtevisione di un loro utilizzo entro breve tempo.
* **Rilascio indietro** (free-behind) e **lettura anticipata** (read-ahead)(si intende principio di località): tecniche di ottimizzazione degli accessi sequenziali.
* Per migliorare le prestazioni del PC si riserva e si gestisce una sezione della memoria come **un disco virtuale o un disco RAM.**

**Diverse locazioni di cache per i dischi**

**Immagine che contiene orologio, segnale, metro

Descrizione generata automaticamenteRICORDA BENE:**

**disco RAM**: si può definire una porzione di memoria RAM come un disco (perché magari non serve tutta), il vantaggio è enorme, in quanto la RAM è molto veloce rispetto ad un disco, quindi si può ad esempio decidere di istallare all’interno di questo disco RAM un programma che si vuole altamente performante.

Immagine che contiene testo, segnale, persone, uomo

Descrizione generata automaticamenteI/O SENZA una buffer cache unificata

**Spiegazione**

Ci sono operazione I/O write e read che passano per il buffer cache, e poi va a scrivere su file system. Oppure si può avere una cache delle pagine, la quale è associata alla memoria, che per poter scrivere dei dati su file system, deve anch’essa transitare sul buffer cache.

Per poter evitare questo doppio passaggio si può usare una buffer cache unificata.

I/O CON una buffer cache unificata

Immagine che contiene segnale

Descrizione generata automaticamente**il vantaggio è notevole, in quanto per prima cosa utilizza dei formalismi più omogenei**

Ripristino

Serve per:

* **Verifica della coerenza:** si confrontano i dati delle directory con quelli contenuti nel blocchi dei dischi, tentando di correggere ogni incoerenza.
* Si utilizzano i **programmi di sistema che consentono di fare delle copie di riserva (backup)** dei dati residenti nei dischi su altri dispositivi di registrazione dati (dischetti, nastri magnetici o dischi ottici).
* Il ripristino della situazione antecedente la perdita di un singolo file o di un intero disco richiederà il recupero **(restore)** dei dati dalle copie di riserva.

NFS

* **NFS** (network file system) è sia una realizzazione sia una definizione di un sistema per l’accesso a file remoti attraverso LAN (o anche WAN).
* Ne contesto dell’NFS si considera un **insieme di stazioni di lavoro interconnesse come un insieme di calcolatori indipendenti con file system indipendenti.** Lo scopo è quello di permettere un certo grado di condivisione tra questi file system, su richiesta esplicita, in modo trasparente.

**Esempio**

Immagine che contiene oggetto, sedendo, orologio, segnale

Descrizione generata automaticamente**spiegazione**

l’utente che va a sfogliare l’albero U non deve sapere se questa user local si trova su un file system o un altro file system. Questo grazie all’NFS che riesce a rendere trasparente la costruzione del file system sottostante e farlo vedere uniforme.

Protocollo di montaggio

* Stabilisce la connessione logica iniziale tra un server e un client.
* Comprende il nome della directory remota da montare e il nome del calcolatore server in cui tale directory è memorizzata.
* La richiesta di montaggi si associa alla RPC corrispondente e s’invia al server di montaggio in esecuzione nello specifico calcolatore server.
* Lista di esportazione: specifica i file system locali esportati per il montaggio e i nomi dei calcolatori ai quali è permessa tale operazione.
* Quando il server riceve una richiesta di montaggio conforme alla propria lista di esportazione, riporta al client una maniglia di file (file handle) da usare come chiave per ulteriori accessi al file.
* La maniglia di file contiene tutte le informazioni di cui ha bisogno il server per gestire un proprio file.

Protocollo NFS

* **Offre un insieme di RPC per operazioni su file remoti che svolgono le seguenti operazioni:**
* **Ricerca** di un file in una directory
* **Lettura** di un insieme di elementi e di directory
* **Manipolazione** di collegamenti e di directory
* **Accesso ad attributi di file**
* **Lettura e scrittura di file**
* Una caratteristica importante dei server NFS è l’assenza dell’informazione di stato: ogni richiesta deve fornire un insieme completo di argomenti.
* I dati modificati si devono riscrivere nei dischi del server prima che i risultati siano riportati al client.
* Il protocollo NFS non fornisce meccanismi per il controllo della concorrenza.

Architettura NFS

* Interfaccia del filesystem UNIX (basate sulle chiamate oper, read, write e close e sui descrittori di file)
* **File system virtuali (**VFS, virtual file system)**:** identifica i file locali da quelli remoti e invoca l’appropriata operazione del file system.
* **Un vantaggio di questa architettura è che il client e il server sono identici; così un calcolatore può essere un client, un server o entrambi.**

**Schema architettura NFS**

**Immagine che contiene orologio

Descrizione generata automaticamente**

Traduzione dei nomi di percorso

* La traduzione dei nomi di percorso si compie suddividendo il percorso stesso in nomi componenti ed eseguendo una chiamata lookup nell’NFS separata per coppia formata da un nome componente e un vnode di directory.
* Una cache per la ricerca dei nomi delle directory nel sito del client, conserva i vnode per i nomi delle directory remote; in questo modo si accelerano i riferimenti ai file con lo stesso nome di percorso iniziale.

Funzionamento remoto

* Tra le normali chiamate del sistema dello UNIX per operazioni su file e le RPC del protocollo NFS esiste una corrispondenza quasi da uno a uno.
* Dal punto di vista concettuale, l’NFS aderisce al paradigma del servizio remoto, ma in pratica si usano tecniche di memorizzazione transitoria e cache per migliorare le prestazioni.
* Non c’è corrispondenza diretta tra un’operazione remota e una RPC, le RPC prelevano blocchi e attributi dei file che memorizzano localmente nella cache. Le successive operazioni remote usano i dati nella cache, soggetti a vincoli.
* Esistono due cache: la cache degli attributi dei file (informazioni degli inode) e la cache dei blocchi di file.
* I client non liberano i blocchi di scrittura differita finche il server non ha confermato che i dati sono stati scritti sui dischi.