

IMPLEMENTAZIONE DEL FILE SYSTEM

Implementazione del File System

- Struttura del File System
- Implementazione
- Implementazione delle Directory
- Metodi di Allocazione
- Gestione dello spazio libero
- Efficienza e Performance
- Recovery

Struttura del File System

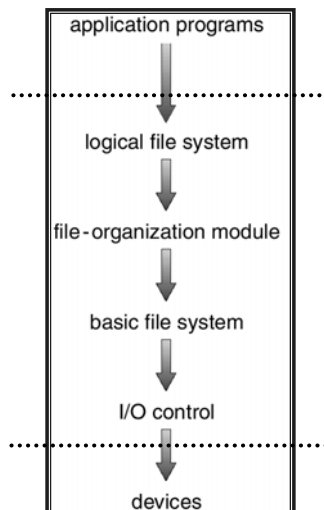
- Struttura dei File
 - Unità Logica di memoria
 - Collezione di informazioni

- Un file è memorizzato e trasferito a blocchi.

- Il File system risiede su memoria secondaria (dischi).

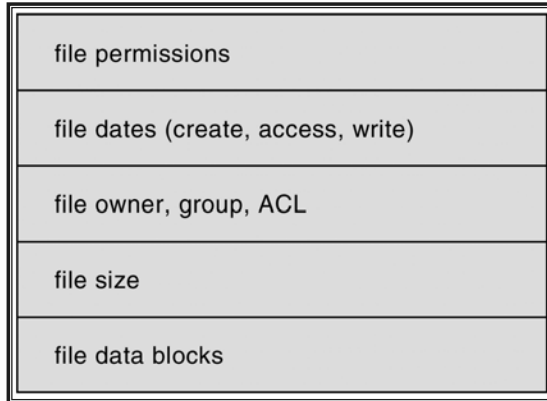
- Il File system organizzato a livelli funzionali.

Livelli di un File System



Un File Control Block

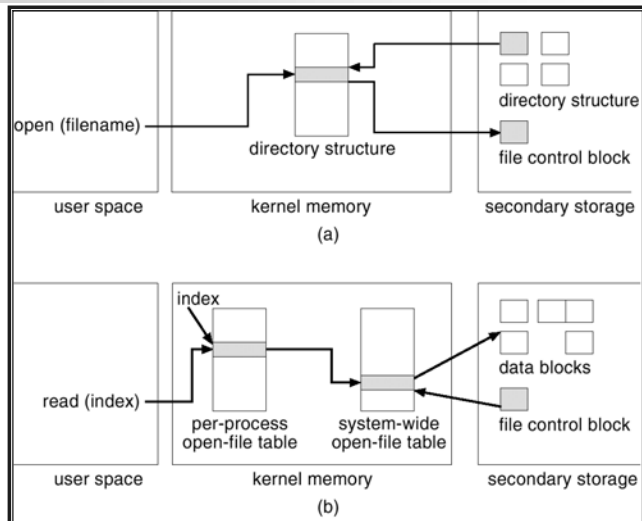
File control block – struttura di memoria che consiste delle informazioni riguardanti un file.



Strutture del File System in Memoria

- Le due figure seguenti illustrano le strutture necessarie al file system fornite dal sistema operativo.
- Figura (a) descrive l'operazione di apertura di un file (*open*).
- Figura (b) descrive l'operazione di lettura (*read*) di un file (*open*).

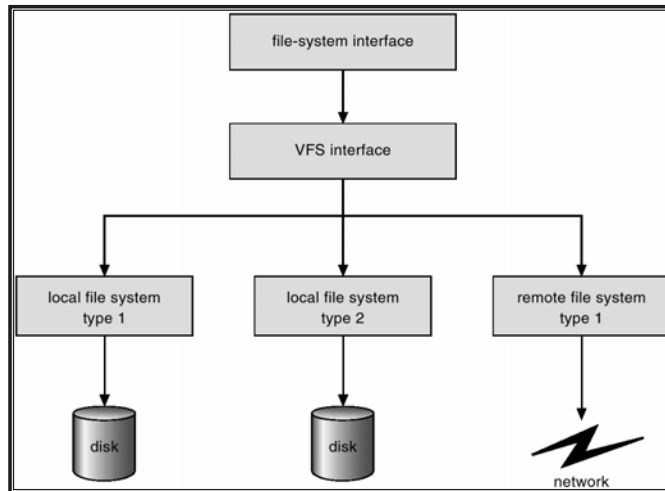
Strutture del File System in Memoria



File System Virtuali

- Alcuni file system, come UNIX, permettono di gestire in maniera integrata diversi tipi di file system.
- Questo è fatto tramite un File System Virtual (VFS) che fornisce una rappresentazione object-oriented del file system
- Un VFS permette di usare una stessa interfaccia (API) per differenti tipi di file system.
- L'interfaccia opera verso il VFS che "nasconde" i diversi tipi di file system sottostanti.

Vista Schematica di un File System Virtuale



Implementazione delle Directory

- La scelta dei metodi di allocazione e gestione delle directory ha un impatto sull'efficienza e l'affidabilità del file system. Due metodi principali:
- **Lista lineare** dei nomi dei file con i puntatori ai blocchi dei dati (contenuto dei file)
 - semplice da programmare
 - non molto efficiente nella ricerca (lineare)
- **Tabella hash** – lista lineare con struttura hash per la ricerca.
 - Diminuisce il tempo di ricerca
 - *collisioni* – situazioni dove due nomi di file portano alla stessa locazione.
 - Dimensione fissata legata alla funzione hash.

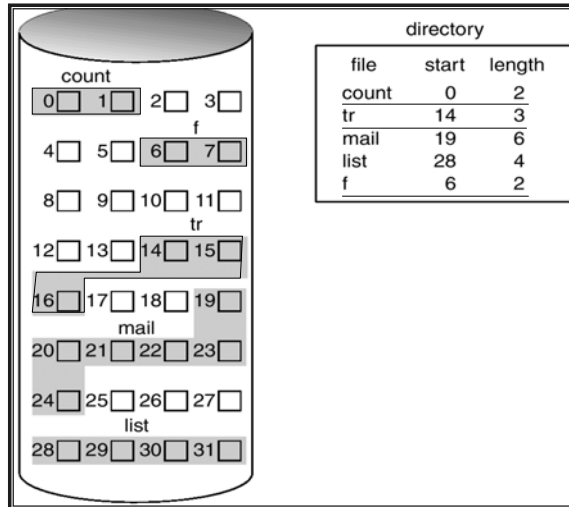
Metodi di Allocazione

- Un metodo di allocazione si occupa di come allocare sulla memoria secondaria i blocchi di un file.
- Tre metodi principali:
- **Allocazione Contigua**
- **Allocazione concatenata** (*linked*)
- **Allocazione indicizzata**

Allocazione Contigua

- Ogni file occupa un insieme contiguo di blocchi sul disco.
- Semplice : è necessario conoscere la locazione di partenza (indirizzo del primo blocco) e la lunghezza (numero di blocchi).
- Accesso casuale.
- Spreco di spazio.
- Occorre trovare lo spazio sufficiente (problema di allocazione dinamica di memoria).
- I file in certi casi non possono crescere di dimensione.

Allocazione Contigua

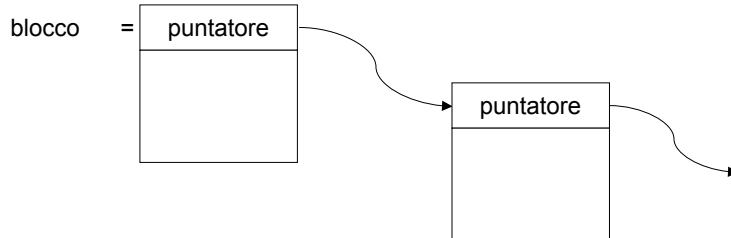


Allocazione Contigua modificata

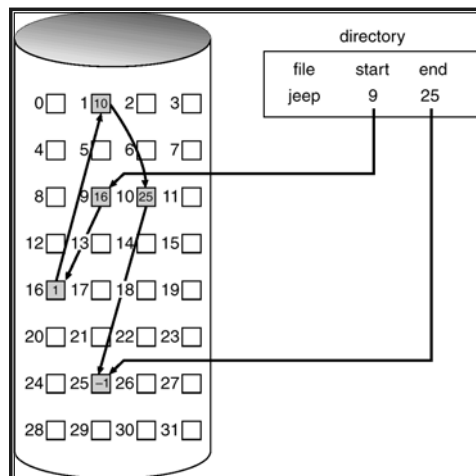
- Alcuni nuovi sistemi operativi moderni usano uno schema modificato dell'allocazione contigua
- Se al file non basta lo spazio di memoria contigua allocata si alloca un ulteriore spazio contiguo (**extent**) su una parte libera del disco.
- Un file quindi può essere composta da due o più *extent*.
- E' necessario avere un contatore degli extent e l'indirizzo del primo *extent*.

Allocazione Concatenata

- Ogni file è gestito tramite una lista concatenata di blocchi di disco: i blocchi non devono essere necessariamente contigui e quindi possono stare in punti diversi del disco.



Allocazione Concatenata



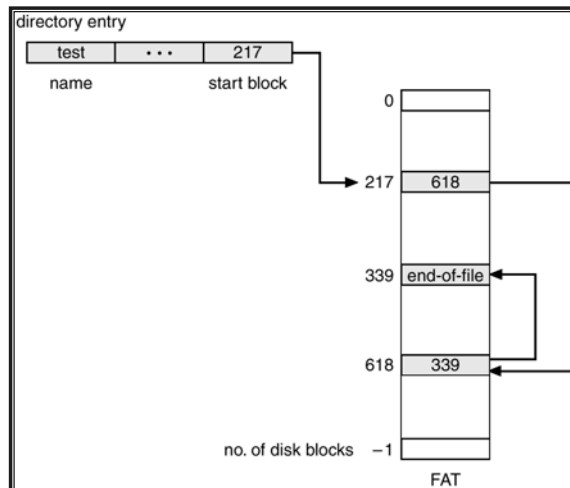
Allocazione Concatenata

- Semplice – serve solo l'indirizzo di partenza.
- Sistema di gestione dello spazio libero – non c'è spreco.
- Accesso sequenziale (quello diretto è inefficiente).
- I file possono crescere.

Una variante:

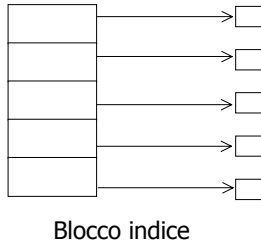
- **File allocation table (FAT)**
 - Tabella con tanti elementi per quanti sono i blocchi sul disco.
 - Ogni elemento contiene l'indice del prossimo blocco.
- Usata in MS-DOS e OS/2.

File Allocation Table

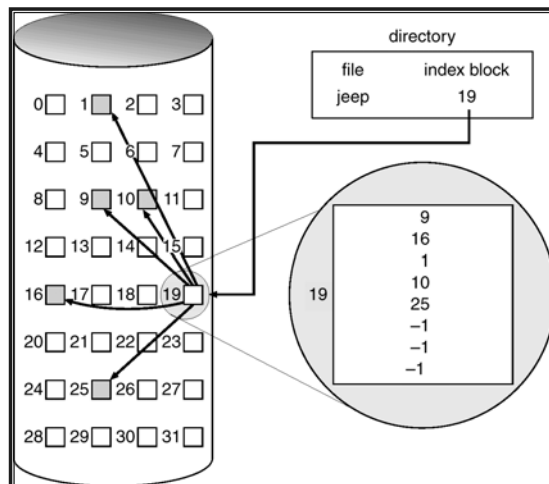


Allocazione Indicizzata

- Mantiene tutti i puntatori ai blocchi dei file in un'unica struttura: il **blocco indice**.
- Vista logica:



Esempio di Allocazione Indicizzata



Allocazione Indicizzata

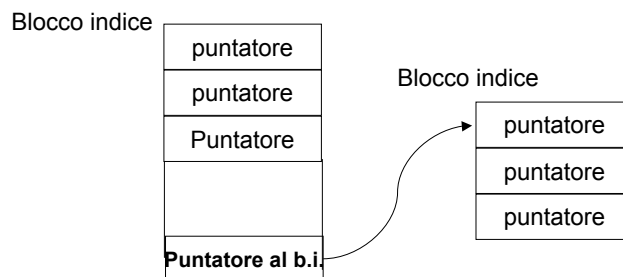
- Blocco indice occupa spazio.
- Accesso diretto
- Non c'è frammentazione esterna.

- Per file composti da un numero massimo di 256K word con blocco indice di 512 word è sufficiente un blocco indice.

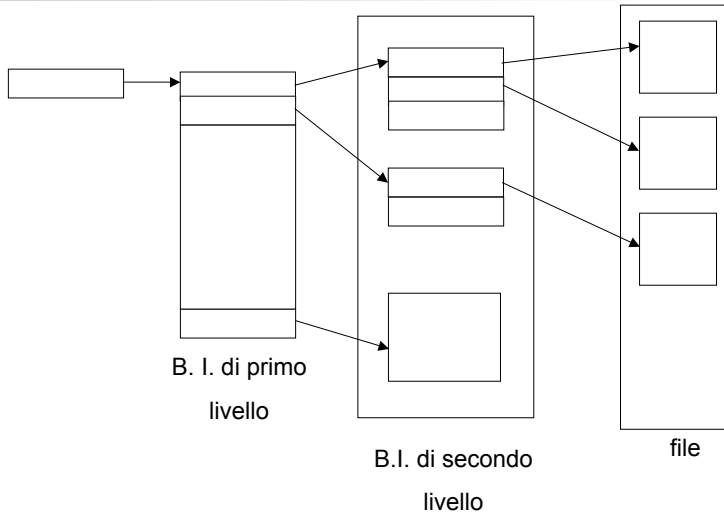
- Se un blocco non è sufficiente a contenere gli indici dei blocchi di un file:
 - schema concatenato
 - schema multilivello
 - schema combinato.

Allocazione Indicizzata: schema concatenato

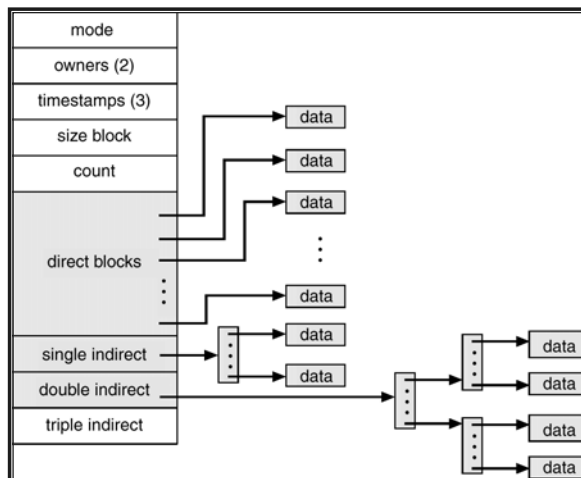
- L'ultima parte di un blocco indice contiene un puntatore ad un altro blocco indice (se il file è molto grande).



Allocazione Indicizzata: schema multilivello

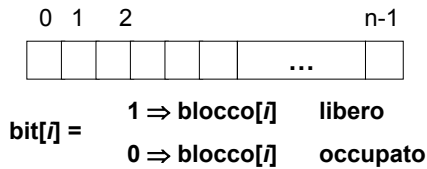


Schema Combinato : UNIX (4K byte per blocco)



Gestione dei blocchi liberi

- Per memorizzare i blocchi liberi si usa il vettore di Bit (n bit per n blocchi)



Calcolo del numero del primo blocco libero usando le word:

(numero di bit per word) * (numero di word con valore 0) + offset del primo bit 1

Gestione dei blocchi liberi

- La bit map richiede uno spazio che potrebbe non essere tutto in memoria. Esempio:

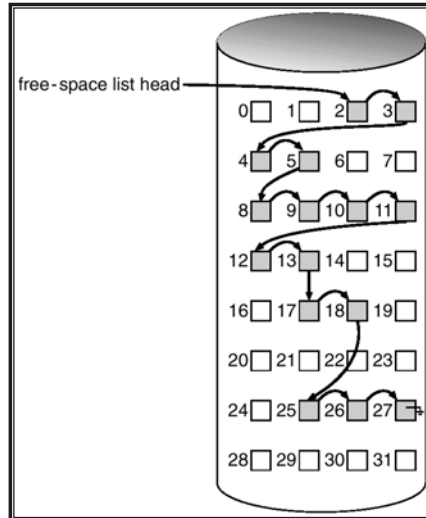
dim. blocco = 2^{12} byte

dim disco = 2^{30} byte (1 gigabyte)

$n = 2^{30}/2^{12} = 2^{18}$ bit (or 32K byte)

- Soluzione alternativa
- Lista concatenata (lista blocchi liberi)
 - Bassa efficienza nella ricerca
 - Non c'è spreco di spazio
- *Grouping* di blocchi per migliorare le prestazioni.

Lista concatenata dei blocchi liberi



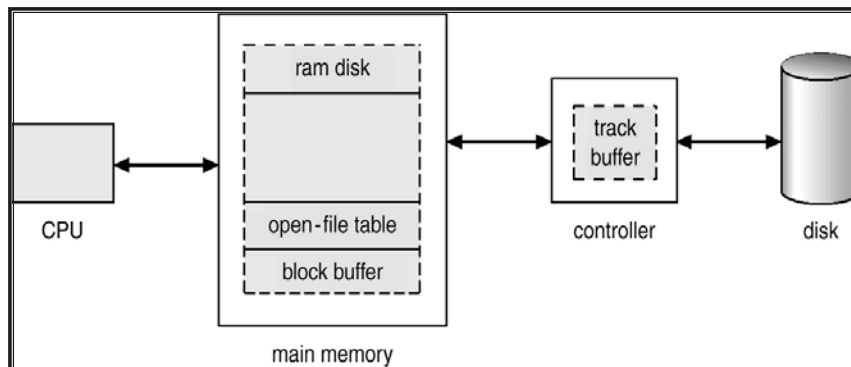
Gestione dei blocchi liberi

- Occorre proteggere i dati critici:
 - Puntatore alla lista dei blocchi liberi.
 - Bit map
 - ❖ Deve essere tenuta sul disco e in memoria
 - ❖ Per un blocco i il bit[i] non può essere = 1 in memoria
bit[i] = 0 sul disco.
 - Soluzione:
 - ❖ Si assegna bit[i] = 1 sul disco.
 - ❖ Si alloca il blocco[i]
 - ❖ Si assegna bit[i] = 1 in memoria.

Efficienza e Performance

- Efficienza della memoria secondaria dipende da:
 - algoritmi di allocazione e algoritmi di gestione delle directory
 - tipi di dati memorizzati nelle entry delle directory.
- Performance
 - *cache del disco* – sezione separata della memoria centrale per i blocchi usati più frequentemente.
 - *Rilascio indietro e lettura anticipata* – tecniche per ottimizzare l'accesso sequenziale.
 - Uso di una sezione della memoria centrale come *disco virtuale* o *disco RAM*.

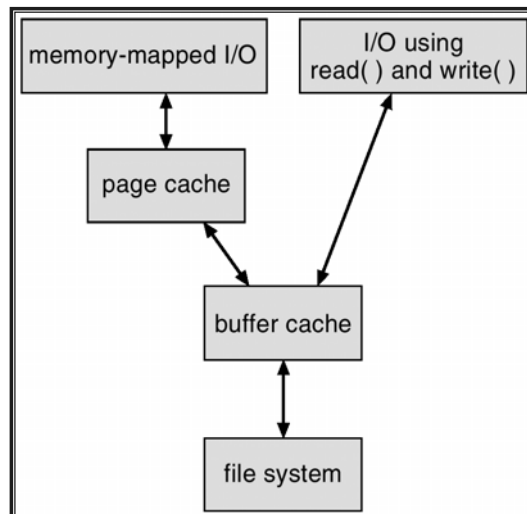
Diverse locazioni di caching sul disco



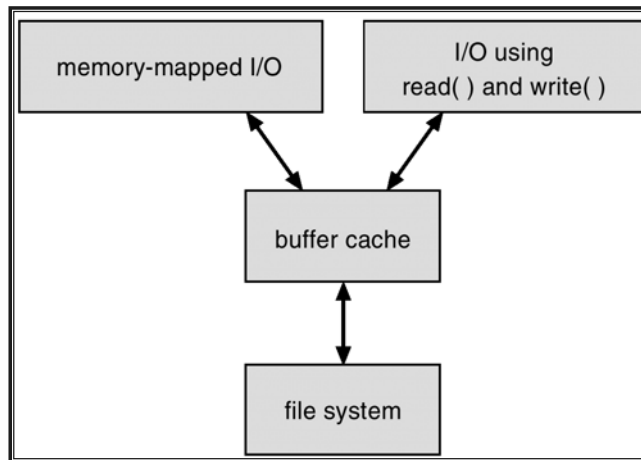
Page Cache

- Molti sistemi come Linux, Solaris e Windows NT usano una **page cache** che contiene pagine piuttosto che blocchi fisici di disco usando tecniche di memoria virtuale.
- Solaris usa un sistema di memoria virtuale unificato che usa la page cache per contenere sia pagine dei processi che pagine di dati dei file.
- Alcune versioni di UNIX implementano una **buffer cache unificata** che velocizza e ottimizza la lettura di pagine dal disco.

I/O senza Buffer Cache unificata



I/O con Buffer Cache unificata



Ripristino (Recovery)

- **Controllo di consistenza:** confronta i dati nella struttura delle directory con i blocchi che sono sul disco e tenta di riparare le inconsistenze.
- **Backup:** Uso di programmi di sistema per realizzare copie di *back up* dei dati su dispositivi diversi (floppy disk, nastri). *Incrementale o totale*.
- Ripristino dei file persi ripristinando i dati dalla copia di backup.
- ***fsck*** in UNIX e ***chkdsk*** in Windows.

File System basati su Log

- Alcuni file system (NTFS, Solaris) usano un sistema di recovery basato su **log** per gestire tutti gli aggiornamenti sui file.
- Tutte le operazioni sono registrate su un **log** come *transazioni*. Dopo aver registrato l'operazione sul log si modifica il file.
- Quando il file system viene modificato, la transazione viene rimossa dal log.
- Se si verifica un crash del file system, tutte le transazioni memorizzate nel log dovranno essere eseguite al suo ripristino.

Domande

- Cosa contiene un file control block e come viene usato dal file system ?
- Discutere i benefici dell'allocazione concatenata rispetto all'allocazione contigua.
- Spiegare come risolvere i problemi di dimensionamento del blocco indice nell'allocazione indicizzata.
- A cosa serve e come è organizzata la FAT ?
- Spiegare come usando i log si può effettuare il ripristino di un file system dopo un crash.