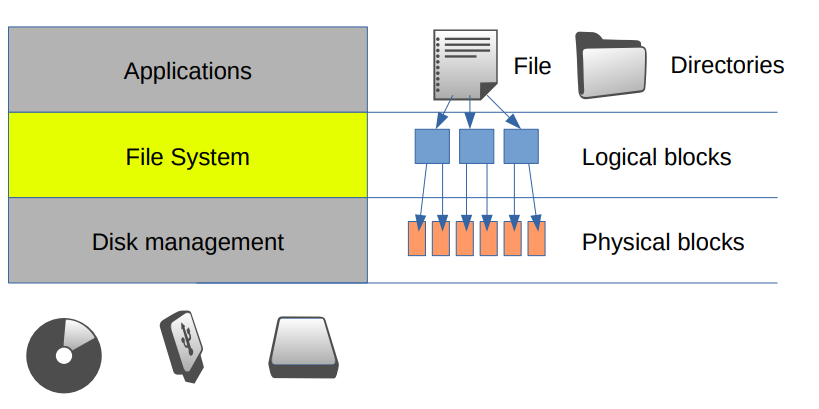
**Filesystem**



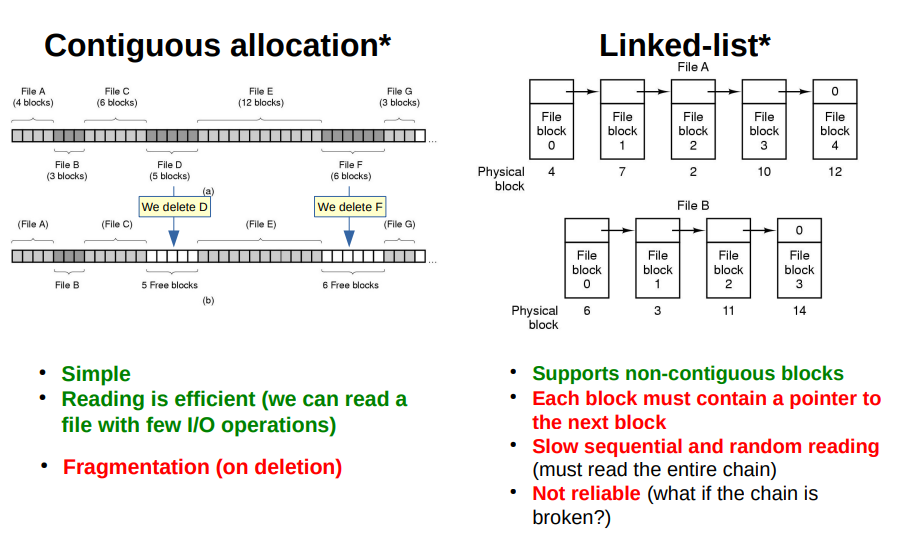
Il filesystem è l’astrazione della strutture ad albero, ma anche il componente kernel che consente di interagire con il disco.

Il sistema operativo si occupa di:

* Gestire lo spazio utilizzato
  + Allocare spazio
  + Liberare spazio
* Fornire fornire l’accesso ai dati
* Fornire uno storage affidabile e veloce.

La gestione dello spazio disponibile avviene tramite una lista o una bitmap

**Allocazione contigua**



L'allocazione soffre del problema della frammentazione quando avvengono cancellazioni

Utile per i sistemi WORM (write once read many) CD, DVD ...

TOC (table of contents) tabella degli indici.

Devo sapere quanto è grande il file che vado a leggere

**Allocazione a catena non contigua**

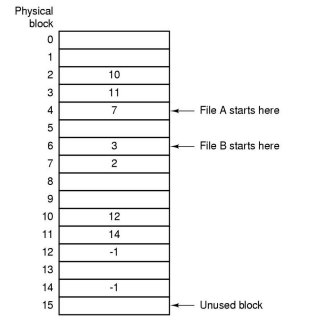
Si usa una lista linkata, una parte di blocco continene i dati e l’altra il riferimento al blocco successivo.

È lento perché bisogna attraversare la lista per raggiungere un determinato blocco.

Problema di affidabilità, se si rompe un blocco si perdono tutte le informazioni che seguono.

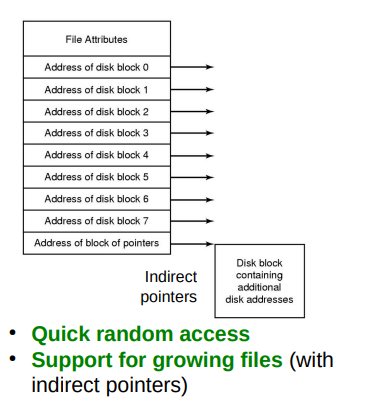
**File allocation table (FAT)**

La struttura dei dati viene caricata in ram, viene caricata la tabella dei puntatori

La dimensione ha dimensione pari al numero di blocchi da gestire, problema la tabella può eccedere la dimensione del disco.

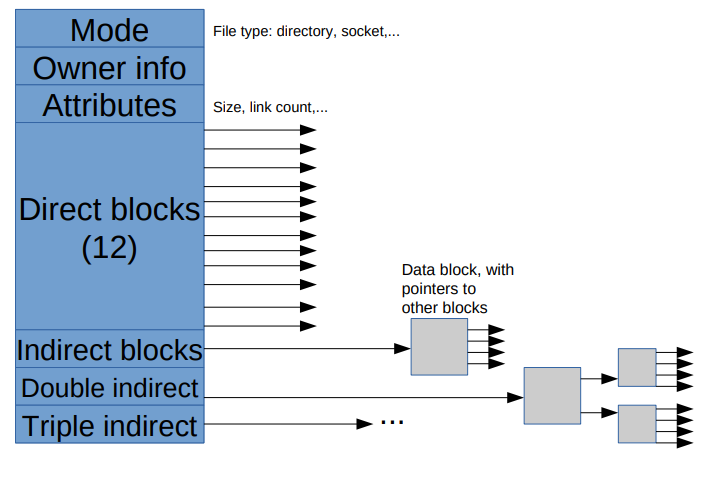
La dimensione del FAT si differenzia per la dimensione della tabella o del cluster.

**Strutture INODE**



Si compone di strutture dati che contengono informazioni dei blocchi, la dimensione degli INODE deve essere fissa.

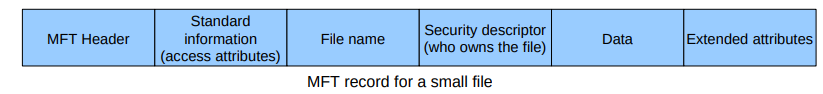
Sono usati dai sistemi UNIX



Se il file è più grande di un blocco diretto si ha un blocco indiretto che a sua volta punta alla continuazione della tabella.

**NTFS**

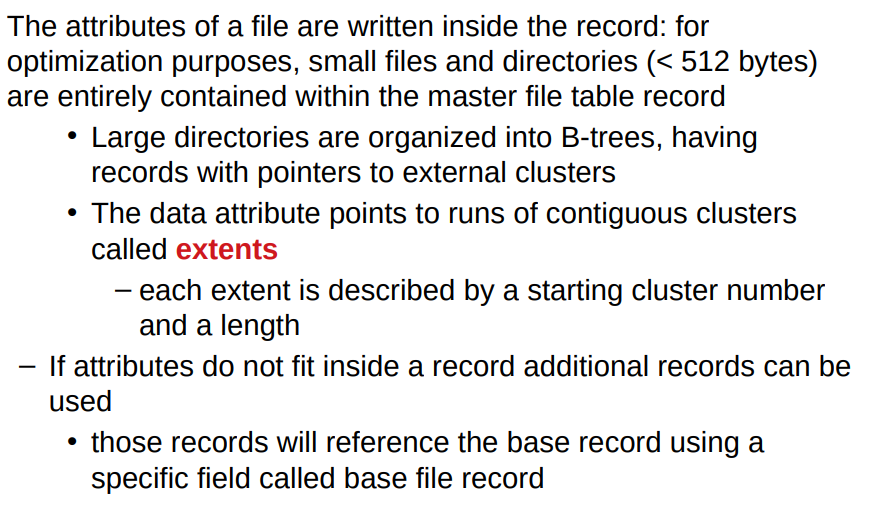
Ha alla base una tabella MFT (master file table)



MFT è come un database, MFT è un file

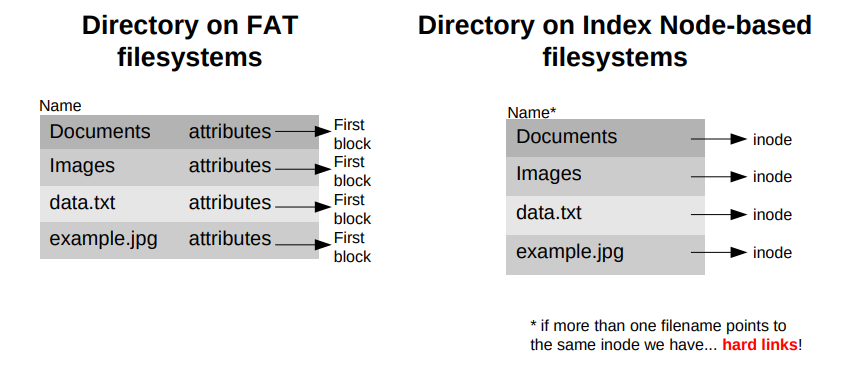
Ogni file è un record della tabella e contiene una chiave e degli attributi.

I primi 16 record sono riservati a file speciali



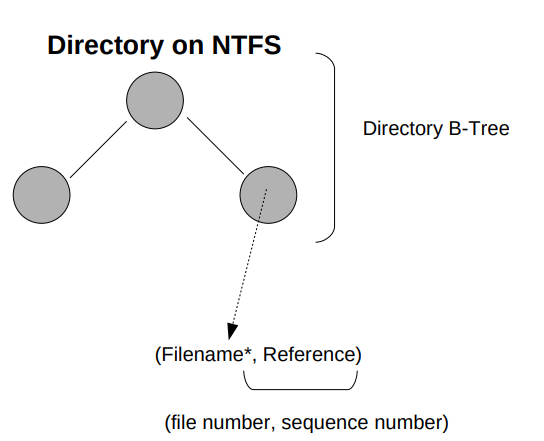
**Directory**

Nei sistemi FAT una directory è una tabella che lega un nome a degli attributi



Negli Inode è un collegamento tra un nome e un inode

In ntfs è un b-tree e ogni file ha un sequence number che consente operazioni di controllo.



**Formattazione del disco**

Oltre a ripulire il disco dai dati si ricrea la struttura del filesystem

Normalmente i filesystem sono implmentati lato kernel

FUSE (filesystems in userspace)

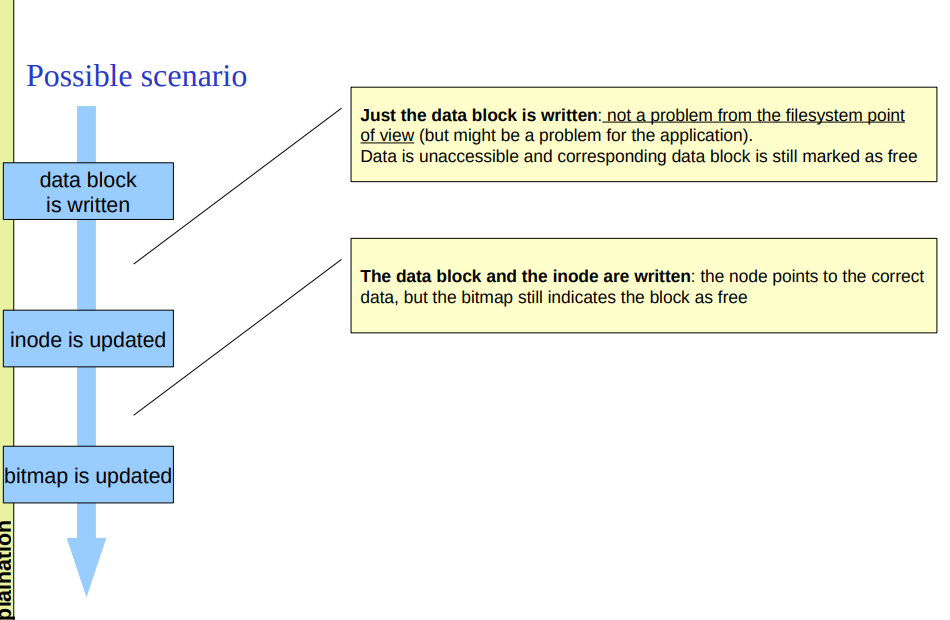
Consente di scrivere filesystem con qualunque linguaggio

FUSED consente di scrivere driver generici

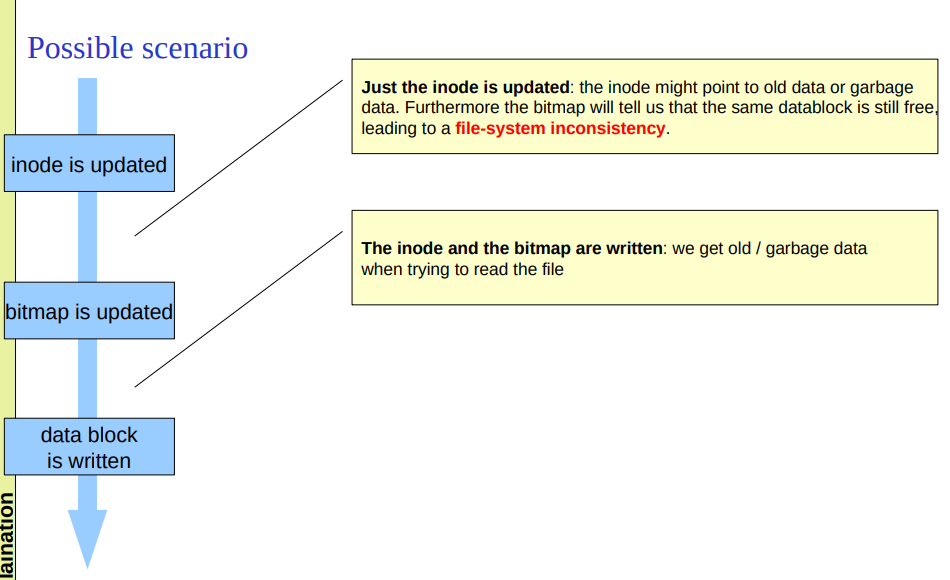
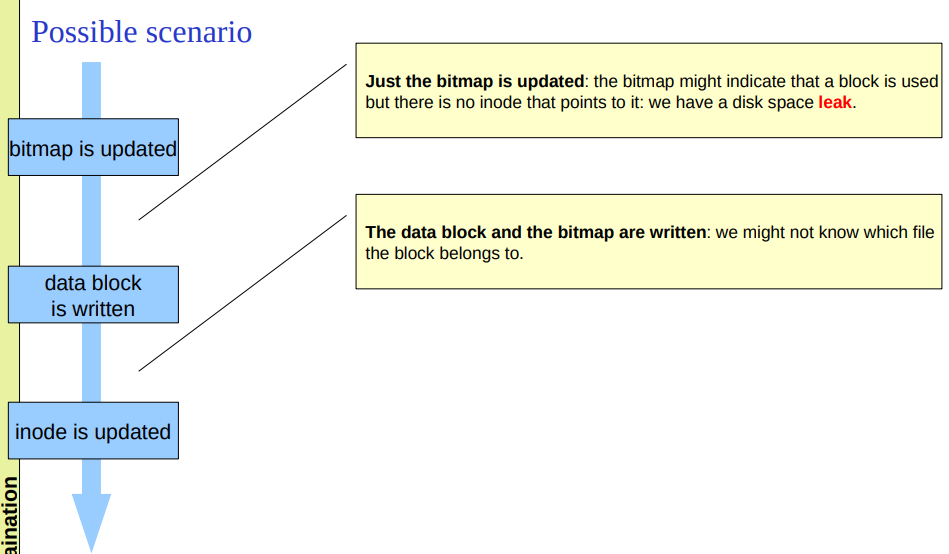
**Prevenire perdita dei dati o inconsistenze**

Il filesystem modifica la bitmap al crescere del file, aggiorna l’inode scrive un nuovo blocco di dati.

Le operazioni sono 3 scritture indipendenti, non viene fatto in modo atomico, ciò può causare inconsistenze.



L'ordine delle operazioni può essere influenzato dallo scheduler del disco



Controlli che effetua il FS

Controlla il flag suuperblock

Contorlla i blocchi allocati e contorlla ogni inode e tiene traccia dei blocchi liberi

Controlla lo stato degli inode

Controlla che gli inode puntino agli stessi blocchi

Controlla la struttura delle directory

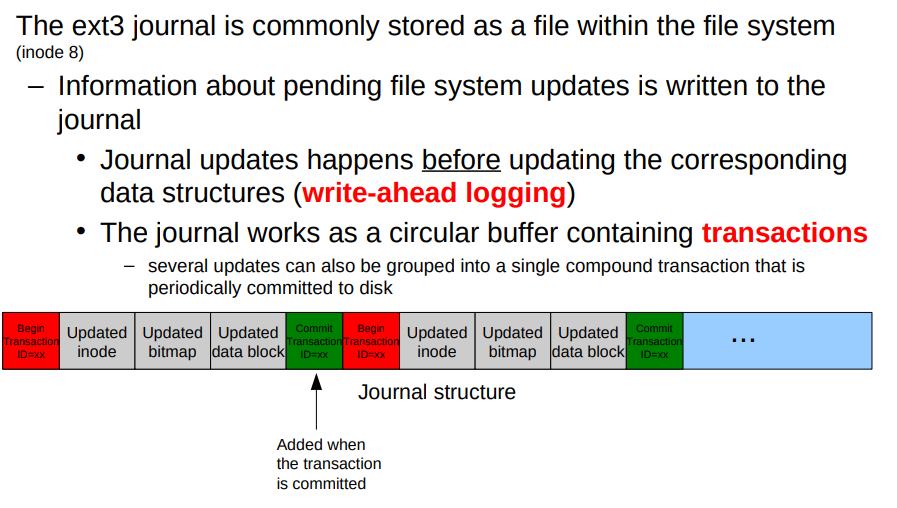
**Journaling**

Usando un meccanismo di journaling il file system mantiene un log di tutti i cambiamenti ogni volta che si fanno delle modifiche, se il sistema crasha si può esaminare il log e rieseguirlo per ripristinare i dati.

Ci sono 2 tipi di journaling

* Physical: ogni operazione di aggiornamento sul disco viene loggata
* Logical: solo i cambiamenti dei metadati sono loggati (modifiche degli inode bitmap ecc..)

Ext3 usa un meccanismo a buffer circolare



Si scrive sul log e poi si applicano sul disco.

Come i database si usa un sistema di transazioni, una volta effettuato il commit si effettuano le transazioni.

Se fallisce le transazioni non si effettuano le modifiche.

Su Ext3 ci sono 3 tipi di journaling:

* **Writeback**: solo i metadati sono loggati e i blocchi sono scritti direttamente su disco (consistenza minore, maggiori performance)
* **Full data journaling:** insieme ai metadati i dati sono loggati nel journal, in caso di crash si possono rispristinare i dati (consistenza maggiore)
* **Ordered:** solo i metadati vengono aggiornati nel log, ma il FS aggiorna i dati prima dell’aggiornamento dei metadati

In caso di crash bisogna effettuare operazioni di recupro prima del montaggio del filesystem

For each transaction in the journal:

– If the crash happened before the transaction has been committed the pending update is ignored

– If the crash happened between the commit and the checkpoint the transaction is replayed (changes on disk are redone)

Con revoke si dice di non sovrascrivere il blocco

Per evitare il suddetto problema, ext3 inserisce un revoke registrare nel journal quando un blocco di metadati viene liberato – Quando si riproduce il log, il sistema operativo guarda i record ed evita di riprodurre gli associati dati