File System

Sistemi Operativi

Antonino Staiano
Email: antonino.staiano@uniparthenope.it

Introduzione

- Panoramica elaborazione dei file
- File e Operazioni su File
- Organizzazione dei file e Metodi di accesso
- Directory
- Protezione dei file
- Allocazione di Spazio su Disco
- Affidabilità del FS
- Journaling
- Casi di studio

Panoramica elaborazione dei file

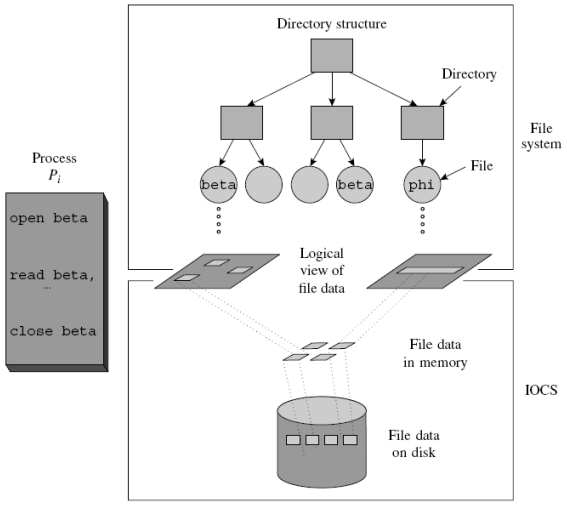


Figure 13.1 File system and the IOCS.

File system e IOCS

- Il file system vede un file come una collezione di dati di proprietà dell'utente, condiviso da un insieme di utenti autorizzati e memorizzati in modo affidabile per lungo tempo
- L'IOCS vede il file come un archivio di dati acceduto in modo veloce e memorizzato su un dispositivo di I/O usato in modo efficiente

Table 13.1 Facilities Provided by the File System and the Input-Output Control System

File System
• Directory structures for convenient grouping of files
• Protection of files against illegal accesses
• File sharing semantics for concurrent accesses to a file
• Reliable storage of files
Input-Output Control System (IOCS)
• Efficient operation of I/O devices
• Efficient access to data in a file

- Due tipi di dati: dati nel file e dati di controllo (metadati)

Elaborazione dei file in un programma

- Livello di linguaggio di programmazione
  - File: oggetto con attributi che descrive l'organizzazione dei suoi dati e i metodi per accedere ai dati

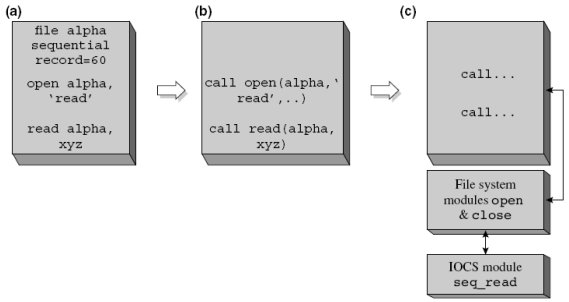


Figure 13.2 Implementing a file processing activity: (a) program containing file declaration statements; (b) compiled program showing calls on file system modules; (c) process invoking file system and IOCS modules during operation.

File e Operazioni su file

- I tipi di file possono essere raggruppati in due classi
  - File strutturati: collezione di record
    - Record: collezione di campi
    - Campo: contiene un singolo dato
    - Ogni record si assume contenga un campo chiave unico
  - Stream di byte: sequenza di byte ''piatta''
- Un file ha attributi memorizzati nella sua entrata nella directory

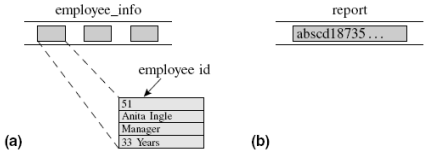


Figure 13.3 Logical views of (a) a structured file `employee_info`; (b) a byte stream file `report`.

File e Operazioni su file (cont.)

Operation	Description
Opening a file	The file system finds the directory entry of the file and checks whether the user whose process is trying to open the file has the necessary access privileges for the file. It then performs some housekeeping actions to initiate processing of the file.
Reading or writing a record	The file system considers the organization of the file (see Section 13.3) and implements the read/write operation in an appropriate manner.
Closing a file	The file size information in the file's directory entry is updated.
Making a copy of a file	A copy of the file is made, a new directory entry is created for the copy and its name, size, location, and protection information is recorded in the entry.
File deletion	The directory entry of the file is deleted and the disk area occupied by it is freed.
File renaming	The new name is recorded in the directory entry of the file.
Specifying access privileges	The protection information in the file's directory entry is updated.

Organizzazione dei file e metodi di accesso

- Modelli di accesso ai record
  - Sequenziale
  - Casuale
- L'organizzazione dei file è una combinazione di due caratteristiche
  - Metodo con cui di dispongono i record in un file
  - Procedura per accedere ai record
- Le caratteristiche di una periferica di I/O determinano l'efficacia per uno specifico pattern di accesso
  - Un nastro è adatto per un accesso sequenziale
  - Un disco implementa in modo efficiente sia l'accesso sequenziale che casuale
- Gli accessi ai file sono regolati da una specifica organizzazione e implementati da un modulo IOCS chiamato metodo di accesso
  - Organizzazione sequenziale
  - Organizzazione diretta
  - Organizzazione indicizzata

Organizzazione sequenziale dei file

- I record sono memorizzati in sequenza ascendente o discendente sulla base del campo chiave
- Due tipi di operazioni:
  - Lettura del prossimo (o precedente) record
  - Saltare il prossimo (o precedente) record
- Utilizzi:
  - Quando i dati possono essere preordinati in modo conveniente in modo ascendente o discendente
  - Per i file stream di byte

Organizzazione diretta dei file

- Fornisce convenienza/efficienza di elaborazione quando i record sono acceduti in ordine casuale
- I file sono chiamati file ad accesso diretto
- Un comando read/write indica il valore del campo chiave
  - Il valore chiave è usato per generare l'indirizzo di un record sul dispositivo fisico
- Svantaggi
  - Il calcolo dell'indirizzo dei record consuma tempo di CPU
  - Spreco di spazio su disco
  - Presenza di record fittizi per valori chiave che non si usano

Esempio: file ad accesso sequenziale e diretto

- Gli impiegati con numeri 3, 5-9, 11 hanno lasciato l'azienda
  - I file ad accesso diretto usano record fittizi per tali record

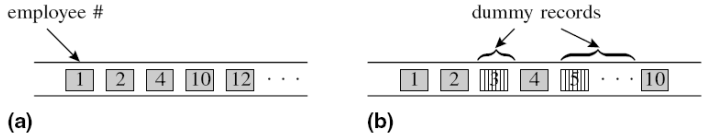


Figure 13.4 Records in (a) sequential file; (b) direct-access file.

Organizzazione indicizzata dei file

- Un indice aiuta a determinare la posizione di un record dal valore chiave
  - Organizzazione indicizzata pura: (chiave, indirizzo disco)
  - L'organizzazione *indicizzata sequenziale* usa un indice per identificare la sezione del disco che può contenere il record
    - I record della sezione sono ricercati in modo sequenziale

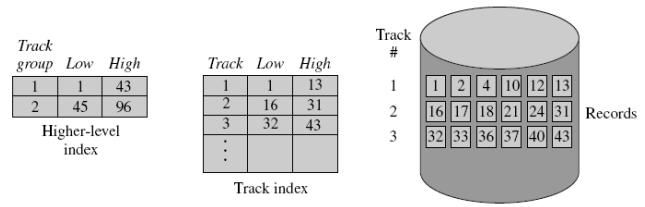


Figure 13.5 Track index and higher-level index in an index sequential file.

Metodi di Accesso

- Metodo di accesso: il modulo IOCS che implementa gli accessi ad una classe di file usando una specifica organizzazione del file
  - Procedura determinata dall'organizzazione dei file
  - Sono usate tecniche avanzate di I/O per l'efficienza:
    - Buffering dei record
      - I record di un file di input sono letti prima del momento in cui sono necessari ad un processo
    - Blocchi di dati
      - Un grande blocco di dati, di dimensione maggiore di un record nel file, viene letto o scritto sul dispositivo di I/O

Directory

File name	*Type and size*	*Location info*	*Protection info*	*Open count*	*Lock*	*Flags*	*Misc info*

Field	Description
File name	Name of the file. If this field has a fixed size, long file names beyond a certain length will be truncated.
Type and size	The file's type and size. In many file systems, the type of file is implicit in its extension; e.g., a file with extension .c is a byte stream file containing a C program, and a file with extension .obj is an object program file, which is often a structured file.
Location info	Information about the file's location on a disk. This information is typically in the form of a table or a linked list containing addresses of disk blocks allocated to a file.
Protection info	Information about which users are permitted to access this file, and in what manner.
Open count	Number of processes currently accessing the file.
Lock	Indicates whether a process is currently accessing the file in an exclusive manner.
Flags	Information about the nature of the file—whether the file is a directory, a link, or a mounted file system.
Misc info	Miscellaneous information like id of owner, date and time of creation, last use, and last modification.

Figure 13.6 Fields in a typical directory entry.

Directory (cont.)

- Il file system deve concedere agli utenti:
  - Libertà di assegnare i nomi ai file
  - Condivisione dei file
- Il file system crea tante directory
  - Usa una struttura per organizzarle
    - Fornisce un modo per assegnare nomi e condivisione

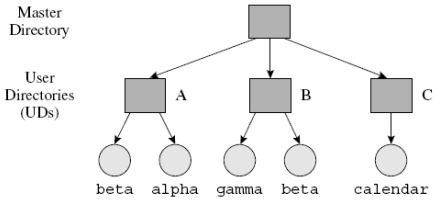


Figure 13.7 A directory structure composed of master and user directories.

Alberi delle directory

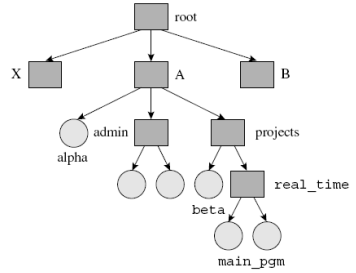


Figure 13.8 Directory trees of the file system and of user A.

- Alcuni concetti: home directory, directory corrente
- Nomi dei percorsi usati per identificare in modo univoco i file
  - Pathname relativi
  - Pathname assoluti

Grafi delle directory

- La struttura ad albero porta ad una asimmetria nel modo in cui utenti diversi possono accedere ai file condivisi
  - Soluzione: usare una struttura di grafo aciclico
    - Un collegamento (link) è una connessione orientata tra due file esistenti nella struttura della directory

```
( ~C, ~C/software/web_server, quest )
```

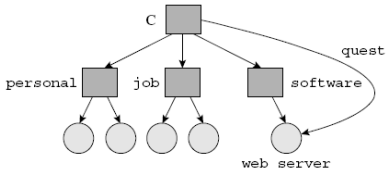


Figure 13.9 A link in the directory structure.

Protezione dei file

- Gli utenti hanno bisogno di una condivisione controllata dei file
  - Il campo *protection info* dell'entrata nella directory del file è usato per controllare l'accesso al file
- Solitamente, *protection info* è memorizzato nella lista di controllo accessi
  - Lista di (*<user_name>*,*<list_of_access_privileges>*)
    - I gruppi utente possono essere usati per ridurre la dimensione della lista
- In molti file system, i privilegi sono di tre tipi:
  - Read
  - Write
  - Execute

Allocazione di spazio su disco

- In un FS possono convivere più SO
  - ogni FS è creato su un **disco logico** ovvero su una ***partizione*** di un disco
- L'allocazione di spazio sul disco è eseguita dal file system
- **Prima ->** modello di allocazione di memoria contigua
  - Comportava frammentazione esterna
- **Ora ->** modello di allocazione non contigua
  - Problemi:
    - Gestire lo spazio libero su disco
      - Uso: free list o disk status map (DSM)
    - Evitare troppi movimenti della testina del disco
      - Uso: estensione (blocchi di disco consecutivi, detto anche cluster) o gruppi di cilindri (cilindri consecutivi su disco)
    - Accedere ai dati nel file
      - Gestione di info sullo spazio allocato per l'accesso ai dati
      - Dipende dall'approccio: concatenato o indicizzato

Allocazione di spazio su disco (cont.)

- Il DSM ha un'entrata per ogni blocco del disco
  - L'entrata indica se il blocco è libero o allocato ad un file
  - L'informazione può essere mantenuta in un singolo bit
    - DSM è anche chiamato bit map
- Il DSM è consultato ogni volta deve essere allocato ad un file un nuovo blocco del disco

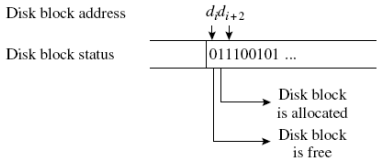
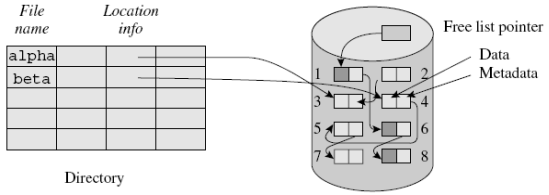


Figure 13.12 Disk status map (DSM).

Allocazione concatenata



- Ogni blocco ha dati, indirizzo del prossimo blocco
  - Semplice da implementare
  - Basso overhead di allocazione/ de allocazione
- Supporta file sequenziali in modo abbastanza efficiente
- I file con organizzazione non sequenziale non possono essere acceduti in modo efficiente
- Scarsa affidabilità (corruzione dei metadati)

Allocazione concatenata (cont.)

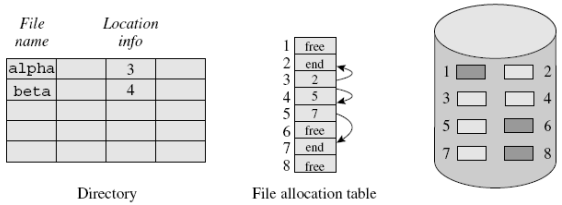
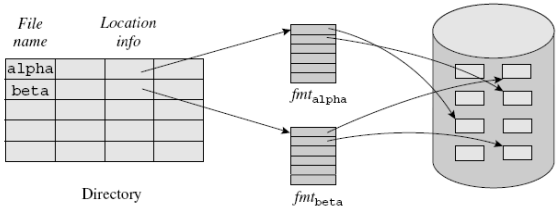


Figure 13.14 File Allocation Table (FAT).

- MS-DOS usa una variante di allocazione concatenata che memorizza i metadati separatamente dai dati nel file
- FAT ha un elemento corrispondente per ogni blocco del disco
  - Problema: è necessario accedere alla FAT per ottenere l'indirizzo del prossimo blocco su disco
    - Soluzione: FAT tenuta in memoria durante l'elaborazione del file

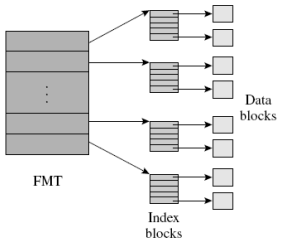
Allocazione indicizzata



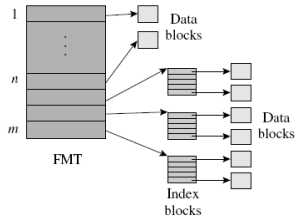
- E' gestito un indice (file map table (FMT)) per annotare gli indirizzi dei blocchi del disco allocati ad un file
  - Forma più semplice: FMT può essere un array di indirizzi di blocchi del disco

Allocazione indicizzata (cont.)

- Altre varianti:
  - Organizzazione FMT a due livelli: compatta, ma l'accesso ai blocchi di dati è più lento
  - Organizzazione FMT ibrida: piccoli file di *n* o meno blocchi di dati accessibili in modo efficiente



A two-level FMT organization.



A hybrid organization of FMT.

Affidabilità del file system

- Grado a cui un file system funzionerà correttamente anche quando si verifica un guasto
  - Ad esempio, corruzione di dati nei blocchi del disco, crash del sistema dovuto a mancanza di corrente
- Due aspetti principali:
  - Assicurare la correttezza della creazione dei file, cancellazione e aggiornamenti
  - Prevenire perdita di dati nel file
- Guasto: difetto in qualche parte del sistema
  - Il verificarsi di un guasto causa un malfunzionamento del sistema
- Malfunzionamento: comportamento erroneo del sistema
  - O che differisce dal comportamento atteso

Perdita di consistenza del file system

- Implica la correttezza dei metadati e delle operazioni del file system
- Un guasto può causare i seguenti malfunzionamenti:
  - Qualche dato di un file aperto può essere perso
  - Parte di un file aperto può diventare inaccessibile
  - I contenuti di due file possono essere scambiati
- Per esempio (primo caso), consideriamo l'aggiunta di un blocco del disco ad un file e un guasto al passo 3:
  1. *d_j.next := d_1.next;*
  2. *d_1.next := address (d_j);*
  3. *Write d_1 to disk.*
  4. *Write d_j to disk.*

// aggiunge il blocco dj tra d1 e d2

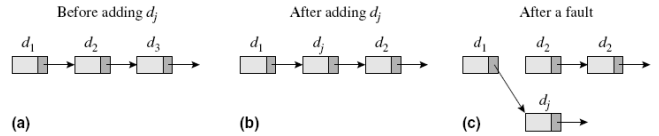


Figure 13.24 Inconsistencies in metadata due to faults: (a)–(b) before and after adding d_j during normal operation; (c) after a fault.

Perdita di consistenza del file system (cont.)

- I contenuti di due file possono essere scambiati se i metadati sono salvati solo dopo una close del file
  - P_1 cancella il blocco d_k da *beta*
  - P_2 aggiunge un nuovo record ad *alpha*
    - Il FS alloca un nuovo blocco d_j e lo mette prima del blocco d_2 in *alpha*
  - Supponiamo che d_j=d_k e che si verificano:
    - *alpha* viene chiuso
    - Il FS aggiorna la copia su disco di *alpha* (aggiunge d_j ad *alpha*)
    - C'è un calo di tensione

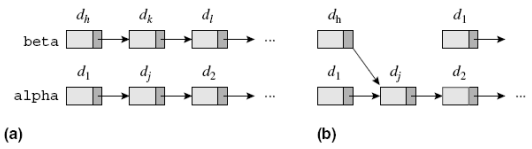


Figure 13.25 Files `alpha` and `beta`: (a) after adding d_j during normal operation; (b) if $d_j = d_k$, `alpha` is closed and a power outage occurs.

Approcci per l'affidabilità del FS

Approach	Description
Recovery	Restore data and metadata of the file system to some previous consistent state.
Fault tolerance	Guard against loss of consistency of data and metadata due to faults, so that system operation is correct at all times, i.e., failures do not occur.

- Il recupero (recovery) è un approccio classico che è attivato quando si osserva un malfunzionamento
- La tolleranza ai guasti fornisce sempre operazioni corretti nel FS

Tecniche di recovery

- *Stato del file system* al tempo t_i: insieme di tutti i dati e metadati nel FS al tempo t_i
- Un backup è la registrazione dello stato del FS
  - Overhead creazione backup
    - Quando è usata l'allocazione di spazio indicizzata, è possibile creare un backup su disco di un file con tecniche che richiamano il copy-on-write della memoria virtuale
  - Overhead della rielaborazione
    - Le operazioni eseguite dopo il backup devono essere rielaborate
  - Soluzione: usare una combinazione di backup e backup incrementali
    - Backup incrementale: copia di file o blocchi del disco modificati dopo l'ultimo backup o backup incrementale
      - Creati ad intervalli più brevi dei backup e rimossi alla creazione del nuovo backup

Tecniche di recovery (cont.)

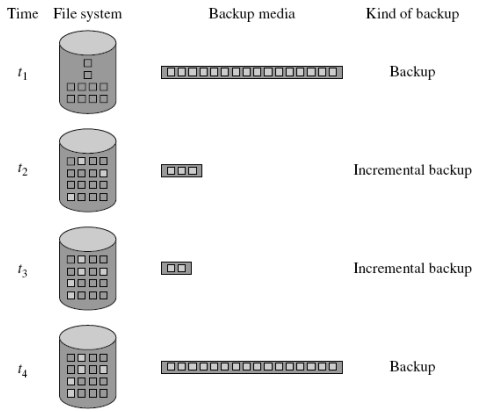


Figure 13.26 Backups and incremental backups in a file system.

Creazione dei backup

- Il perno centrale nella creazione dei backup è la consistenza dei metadati
- Consideriamo lo scenario
  1. Le strutture dati per la *free list* sono scritte nel backup
  2. Un record è aggiunto al file *phi* (richiede un nuovo blocco da allocare a *phi* dalla free list)
  3. *phi* scritto nel backup
- Inconsistenza scrittura della *free list* e del file *phi* nel backup
- Inconsistenze dei metadati
  - Si congelano tutte le attività durante la creazione del backup
  - Normale backup ma con semplificazioni (free list non scritta nel backup)
    - Al ripristino il FS può scandire il disco e costruire la free list

Tecniche di recupero (cont.)

- Per ridurre l'overhead per la creazione di backup (quando è usata l'allocazione indicizzata) sono copiati solo l'FMT ed il blocco del disco i cui contenuti sono aggiornati dopo aver creato il backup
  - Risparmia sia spazio su disco che tempo

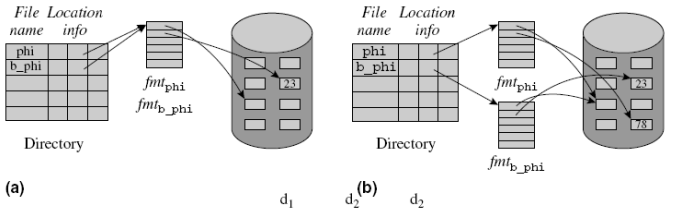


Figure 13.27 Creating a backup: (a) after backing up file `phi`; (b) when `phi` is modified.

Tecniche di tolleranza ai guasti

- L'affidabilità del FS può essere migliorata ricorrendo a due precauzioni
  - Prevenendo la perdita dei dati o metadati a causa di malfunzionamenti del dispositivo di I/O
    - Approccio: usare dispositivi stabili
  - Prevenendo inconsistenza dei metadati dovute ai guasti
    - Approccio: usare azioni atomiche

Memorizzazione stabile

- Mantiene due copie dei dati
  - Può tollerare un guasto nella memorizzazione di un dato
  - Incorre in elevato overhead di spazio e tempo
  - Non può indicare se una copia è vecchia o nuova

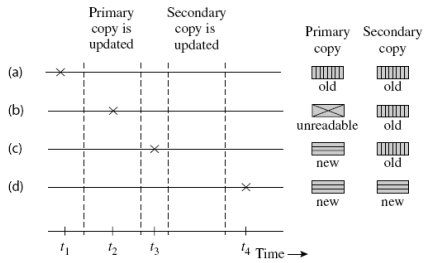


Figure 13.28 Fault tolerance using the stable storage technique.

Azioni atomiche

Azione atomica: un'azione che consiste di un insieme di sotto-azioni la cui esecuzione ha la proprietà che

1. Gli effetti di tutte le sue sotto-azioni si compiono, oppure
2. Gli effetti di nessuna delle sue sotto-azioni si compiono

begin atomic action *add_a_block;*
  $d_j.next := d_1.next;$
  $d_1.next := address(d_j);$
  write d_1;
  write d_j;
end atomic action *add_a_block;*

- Per l'implementazione sono usate due strutture dati (gestite in dispositivi stabili)
  - *intention list* (entrate del tipo *<disk block id>, <nuovo contenuto>*)
    Le entrate sono create ad ogni modifica di dati o metadati
  - *commit flag* (due campi transaction id e valore)
    - Creato con **begin atomic action** (Ai, NC)
    - NC diventa C in corrispondenza di **end atomic action**
    - Cancellato quando tutti le modifiche della *intention list* sono eseguite

Azioni atomiche (cont.)

Algorithm 13.2 *Implementation of an Atomic Action*

1. *Execution of an atomic action* A_i*:*
   a. When the statement **begin atomic action** is executed, create a *commit flag* and an *intentions list* in stable storage, and initialize them as follows:
   commit flag := (A_i, "not committed");
   intentions list := "empty";
   b. For every file update made by a subaction, add a pair (d, v) to the intentions list, where d is a disk block id and v is its new content.
   c. When the statement **end atomic action** is executed, set the value of A_i's *commit flag* to "committed" and perform Step 2.
2. *Commit processing:*
   a. For every pair (d, v) in the intentions list, write v in the disk block with the id d.
   b. Erase the commit flag and the intentions list.
3. *On recovering after a failure:*
   If the commit flag for atomic action A_i exists,
   a. If the value in commit flag is "not committed": Erase the commit flag and the intentions list. Reexecute atomic action A_i.
   b. Perform Step 2 if the value in commit flag is "committed."

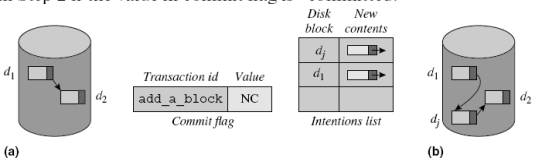


Figure 13.30 (a) Before and (b) after commit processing. (*Note:* NC means *not committed*.)

Journaling file system

- Uno shutdown non pulito può portare alla perdita dei dati
  - Approccio tradizionale: tecniche di recupero
  - Approccio moderno: usare tecniche di tolleranza ai guasti in modo che il sistema può ripristinare velocemente le operazioni dopo lo shutdown
    - Un journaling FS implementa la tolleranza ai guasti gestendo un diario quotidiano (journal)

Table 13.6 Journaling Modes

Mode	Description
Write behind	Protects only metadata. Does not provide any protection to file data.
Ordered data	Protects metadata. Limited protection is offered for file data as well—it is written to disk before metadata concerning it is written.
Full data	Journals both file data and metadata.

Casi di studio

- Unix File system
  - Berkeley Fast File system
- Linux File System
- Windows File System

Unix File System

- Strutture dati del FS
  - Un'entrata della directory contiene solo il nome del file
  - L'Inode del file contiene la dimensione del file, id proprietario, permessi di accesso, e info sull'allocazione dei blocchi su disco
  - Una struttura di file contiene le informazioni su un file aperto
    - Contiene la posizione corrente nel file (offset) e il puntatore al suo inode
  - Un descrittore di file punta ad una struttura di file
  - Allocazione indicizzata del disco con tre livelli di indirezione
- Semantica condivisione file di Unix
  - Il risultato di una write eseguita da un processo è immediatamente visibile a tutti gli altri processi che correntemente accedono al file

Unix File System (cont.)

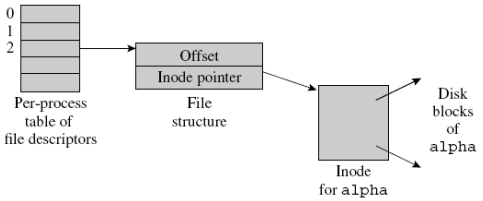


Figure 13.32 Unix file system data structures.

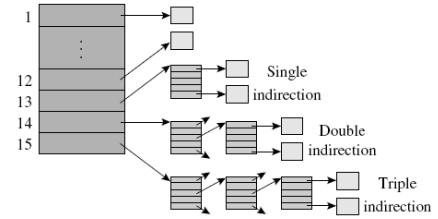


Figure 13.33 Unix file allocation table.

Berkeley Fast File System

- FFS sviluppato per affrontare le limitazioni del FS s5fs
- Supporta alcuni miglioramenti come nomi di file lunghi e l'uso di link simbolici
- Inclide numerose innovazioni riguardanti l'allocazione di blocchi del disco e l'accesso al disco
  - Permette l'uso di grandi blocchi di disco (fino a 8KB)
  - Usa gruppi di cilindri per ridurre il movimento della testina del disco
  - Cerca di minimizzare la latenza di rotazione durante la lettura di file sequenziali

Linux File System

- Linux fornisce un file system virtuale (VFS)
  - Supporta un comune modello di file che somiglia al modello di file di Unix
- Il file system standard è ext2
  - Varietà di lock per la sincronizzazione dei processi
  - Usa la nozione di gruppo di blocchi
  - Ext3 incorpora il journaling

Windows File System

- NTFS è progettato per server e workstation
  - Caratteristiche chiave: recuperabilità del FS
- Nozione di partizione e volumi
  - I volumi hanno una master file table (MFT)
- Directory organizzate come B+ tree
- Link hard e simbolici (chiamati giunzioni)
- Tecniche speciali per file sparsi e compressione dati
- Le modifiche ai metadati sono transazioni atomiche
- Capacità *write behind* del journaling FS