



UNIVERSITÀ DEGLI STUDI DI NAPOLI
PARTHENOPE

Sistemi Operativi

File System

LEZIONE 23

prof. Antonino Staiano

Corso di Laurea in Informatica – Università di Napoli Parthenope

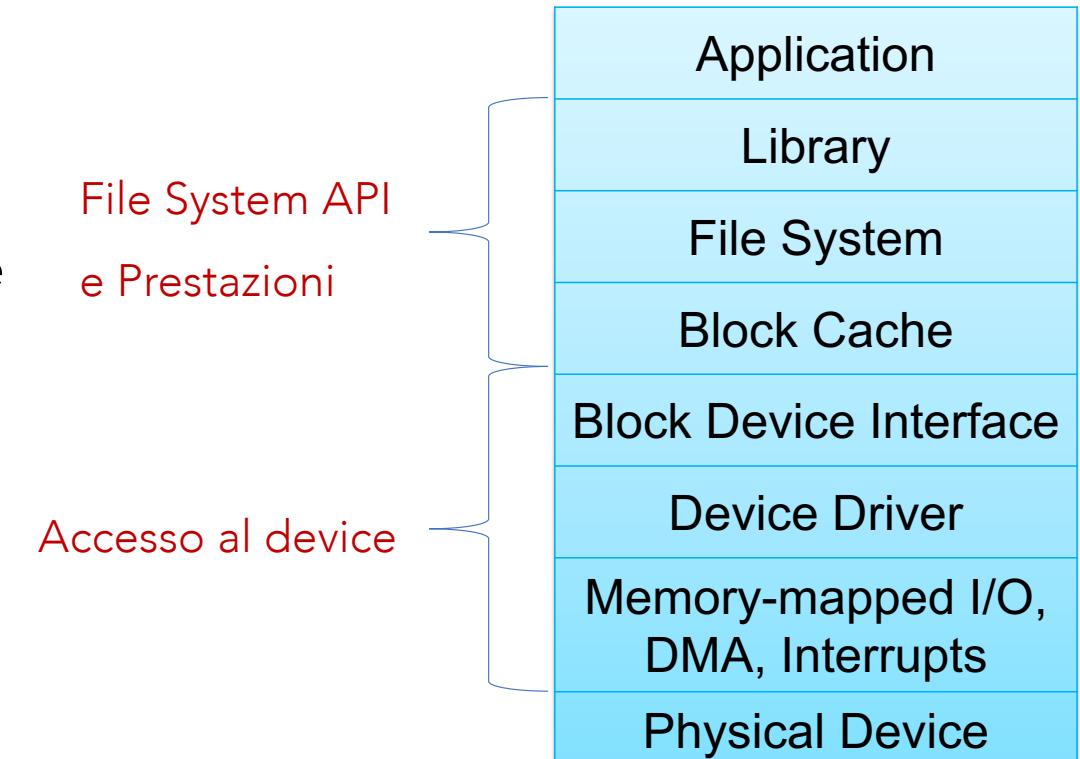
antonino.staiano@uniparthenope.it

File System

- Ha a che fare con la memorizzazione a lungo termine dei dati
 - Grandi quantità di informazioni
 - Informazioni che permangono oltre il ciclo di vita di un processo
 - Necessario accesso concorrente a processi multipli
- E' necessario garantire
 - Accesso conveniente e veloce ai file
 - Memorizzazione affidabile
 - Condivisione di file tra collaboratori

Introduzione

- Per accedere e memorizzare i file si usano i **dispositivi di I/O**
- I device di I/O sono acceduti mediante una serie di astrazioni organizzate in strati
- La soluzione che possiamo adottare
 - File System
 - Input-Output Control System (IOCS)
 - Per separare gli aspetti a livello di file da aspetti legati alla memorizzazione e all'accesso efficiente



Panoramica elaborazione dei file

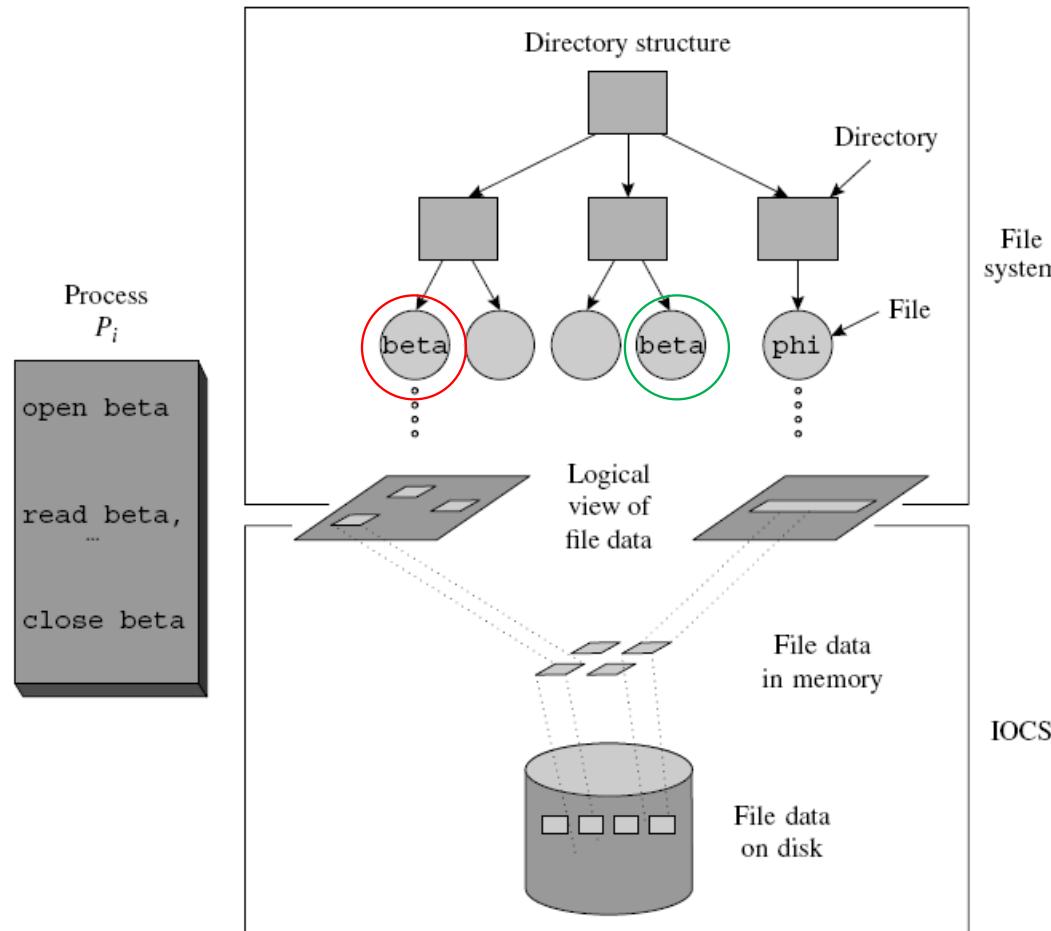


Figure 13.1 File system and the IOCS.

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 memorizzato 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
<ul style="list-style-type: none">• 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)
<ul style="list-style-type: none">• Efficient operation of I/O devices• Efficient access to data in a file

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

File

- E' una collezione di dati caratterizzata da un nome
 - Dati
 - Ciò che un utente o un'applicazione mette all'interno del file
 - Un array di byte privi di tipo
 - Metadati
 - Informazioni aggiunte e gestite dal SO
 - Dimensione, proprietario, info di sicurezza, istante di modifica

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

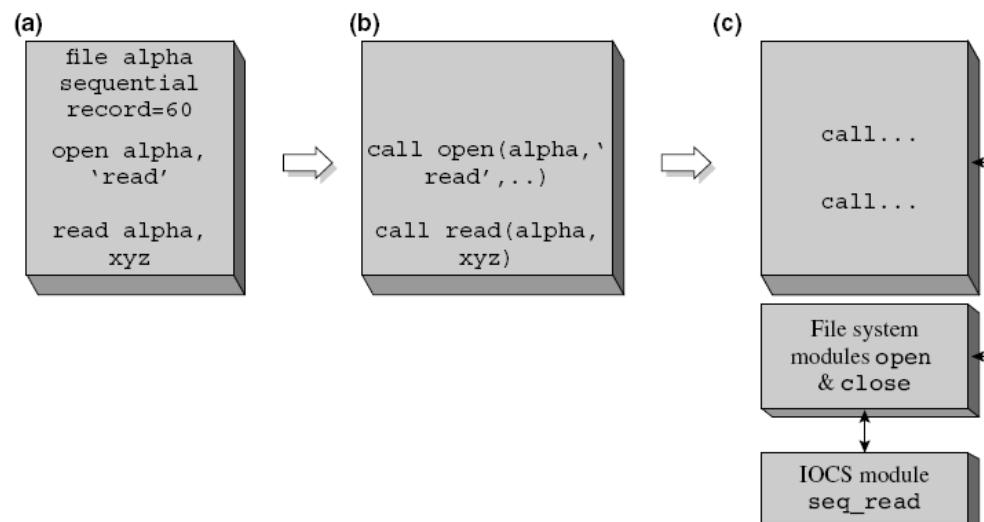


Figure 13.2 Implementing a file processing activity: (a) program containing file declaration statements; (b) compiled program showing calls on file system modules; (c) process invoking file system and IOCS modules during operation.



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"

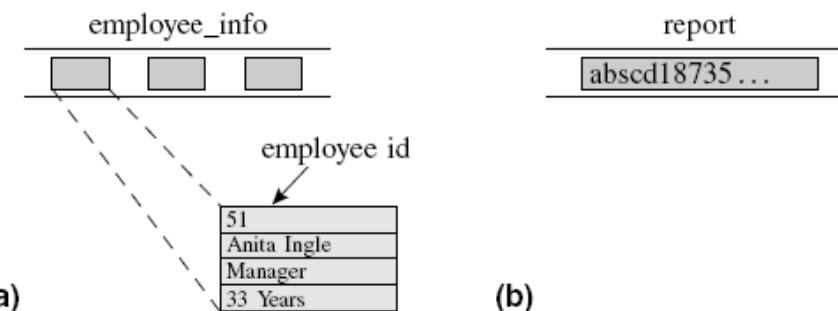


Figure 13.3 Logical views of (a) a structured file `employee_info`; (b) a byte stream file `report`.



File e Operazioni su file (cont.)

Table 13.2 Operations on Files

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 per un processo
 - Sequentiale
 - Casuale
- L'organizzazione dei file è una combinazione di due caratteristiche
 - Metodo con cui si 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
- Un FS supporta diverse organizzazioni, le tre fondamentali sono:
 - 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 fintizi 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 fintizi per tali record

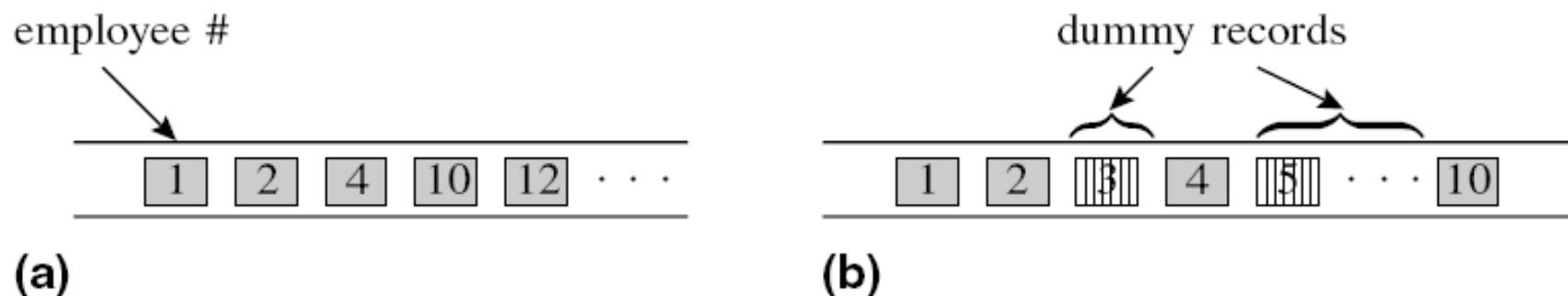


Figure 13.4 Records in (a) sequential file; (b) direct-access file.

Organizzazione indicizzata dei file

- Un indice aiuta a determinare la posizione di un record dal valore della 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

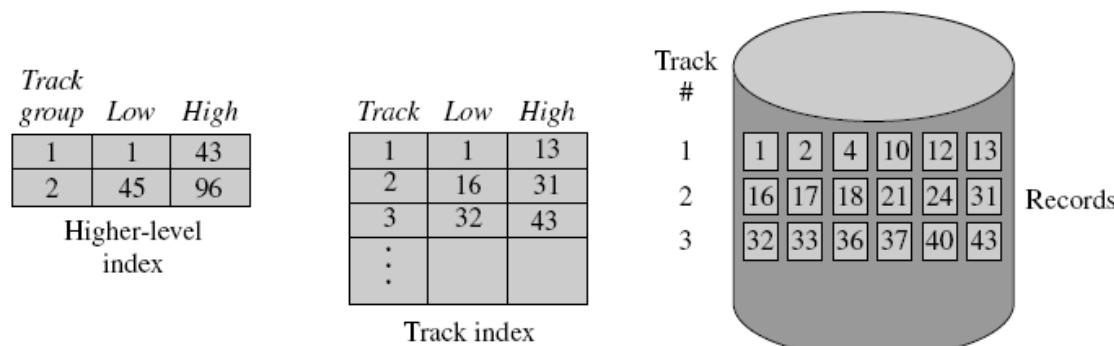


Figure 13.5 Track index and higher-level index in an index sequential file.

Directory

<i>File name</i>	<i>Type and size</i>	<i>Location info</i>	<i>Protection info</i>	<i>Open count</i>	<i>Lock</i>	<i>Flags</i>	<i>Misc info</i>
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 consentire 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

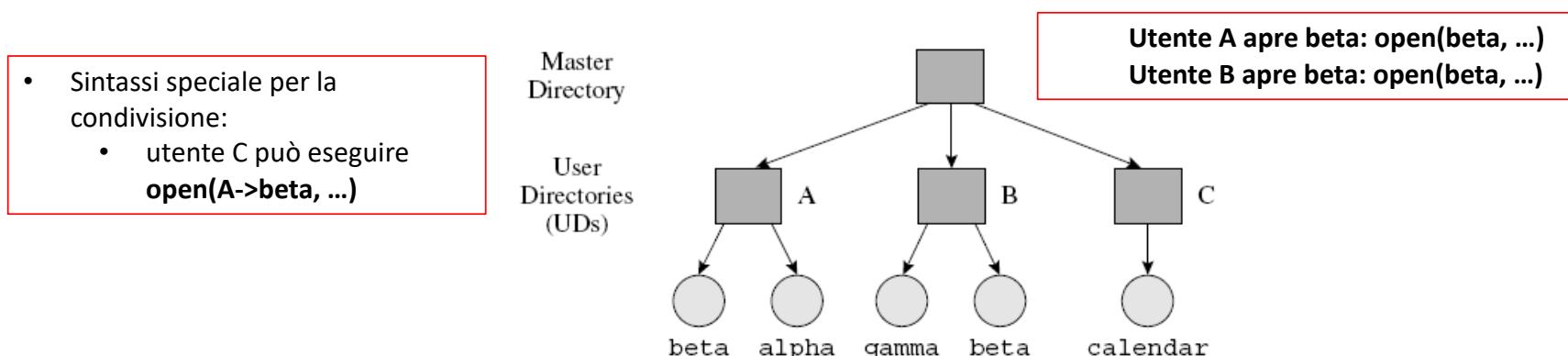


Figure 13.7 A directory structure composed of master and user directories.

Alberi delle directory

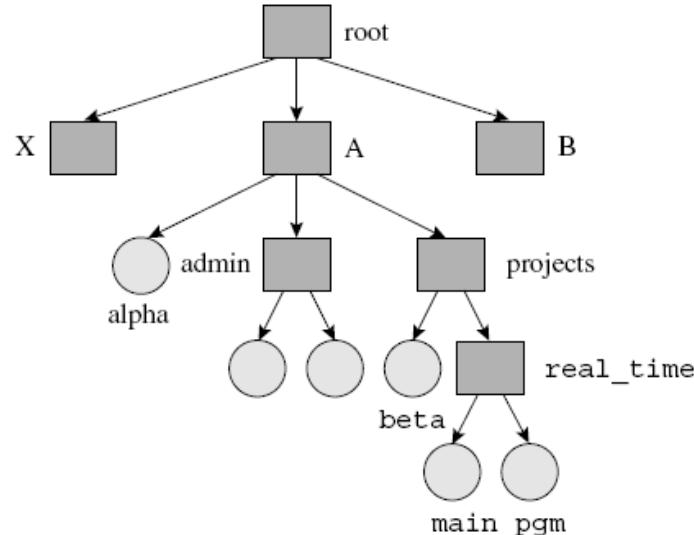


Figure 13.8 Directory trees of the file system and of user A.

- 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
 - Forma generale di un link: (<from_file_name>, <to_file_name>, <link_name>)

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

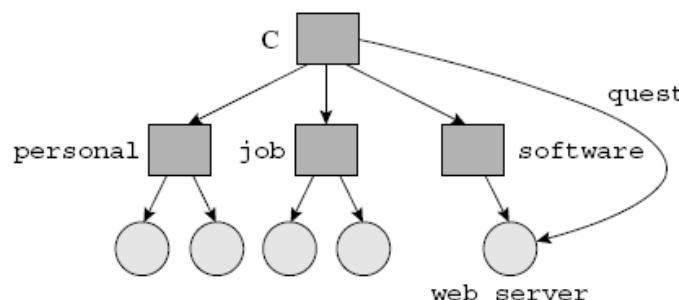


Figure 13.9 A link in the directory structure.

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 (ACL)**
 - 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 disco possono risiedere più FS
 - 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
 - Necessario gestire tre 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, detti anche cluster) o gruppi di cilindri (cilindri consecutivi su disco)
 - La DSM semplifica il prelievo di blocchi da un cluster o gruppo di cilindri
 - 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.)

- La 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
- La DSM è consultata ogni volta deve essere allocato ad un file un nuovo blocco del disco

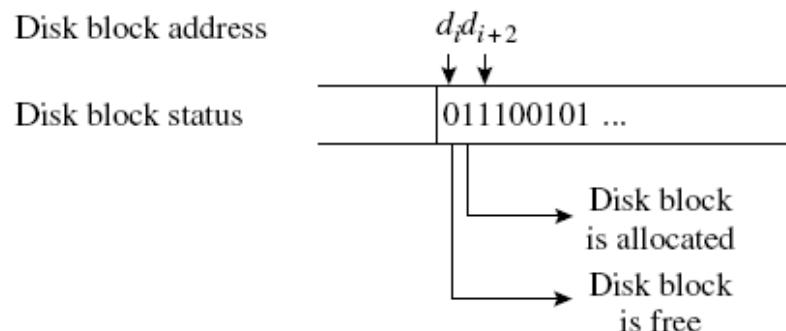
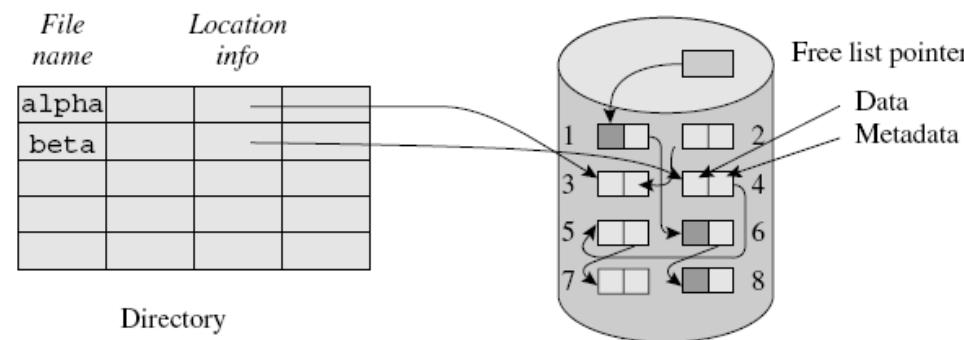


Figure 13.12 Disk status map (DSM).

Allocazione concatenata



- Ogni blocco contiene dati e 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: File Allocation Table (FAT)

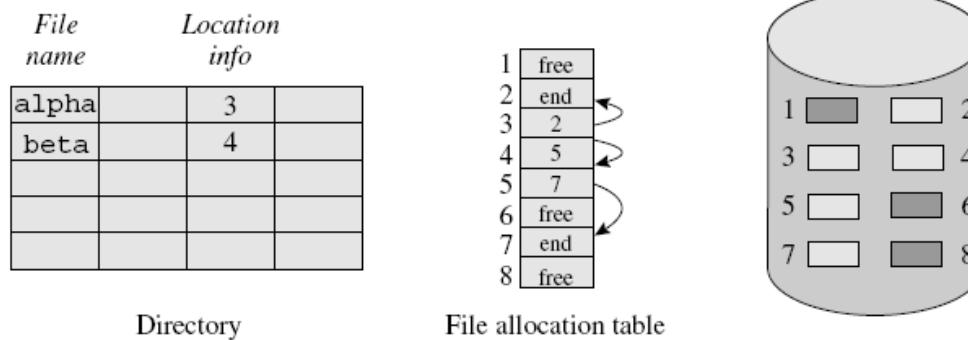
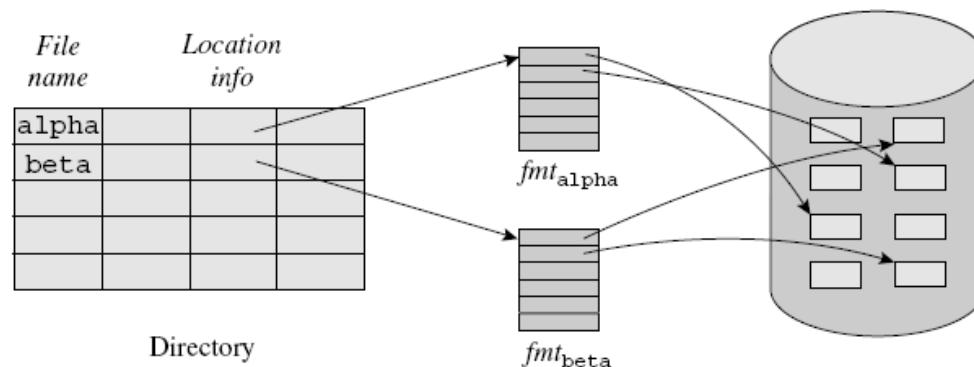


Figure 13.14 File Allocation Table (FAT).

- 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

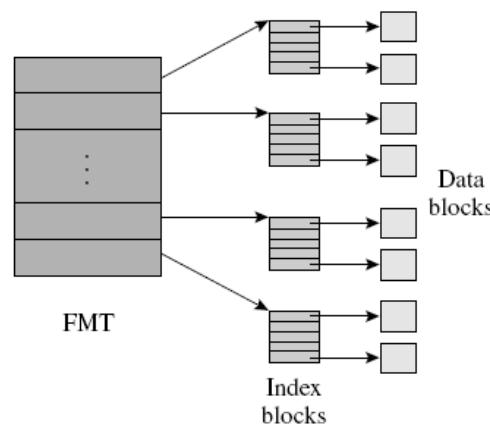
Allocazione indicizzata



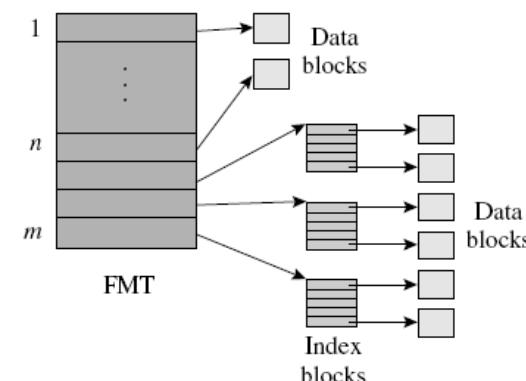
- E' gestito un indice (**File Map Table (FMT)**) per memorizzare gli indirizzi dei blocchi del disco allocati ad un file
 - Forma più semplice: FMT può essere un array di indirizzi di blocchi del disco
- Se i file sono piccoli, l'intera fmt può essere tenuta nell'elemento della directory relativa al file. Altrimenti ...

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

- La consistenza del FS implica la correttezza dei metadati e delle operazioni del file system
- Un guasto può causare i seguenti malfunzionamenti:
 - a. Qualche dato di un file aperto può essere perso (es., aggiunta di un blocco e guasto)
 - b. Parte di un file aperto può diventare inaccessibile
 - c. I contenuti di due file possono essere scambiati
- Per esempio (caso b), consideriamo l'aggiunta di un blocco del disco ad un file e un guasto al passo 3:

1. $d_j.\text{next} := d_1.\text{next}$
2. $d_1.\text{next} := \text{address}(d_j)$
3. Write d_1 to disk
4. Write d_j to disk

// aggiunge il blocco d_j tra d_1 e d_2

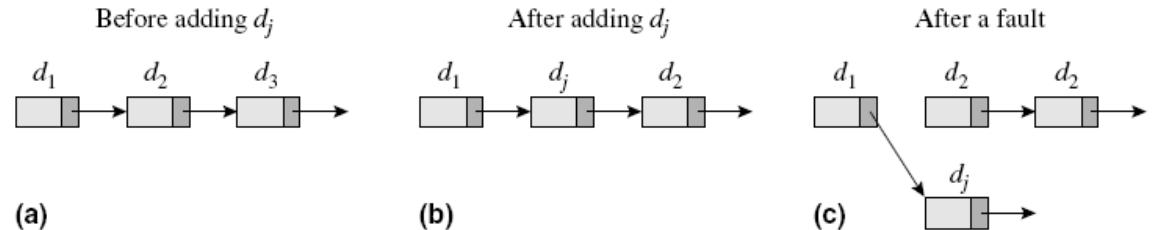


Figure 13.24 Inconsistencies in metadata due to faults: (a)–(b) before and after adding d_j during normal operation; (c) after a fault.

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 corrette 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 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

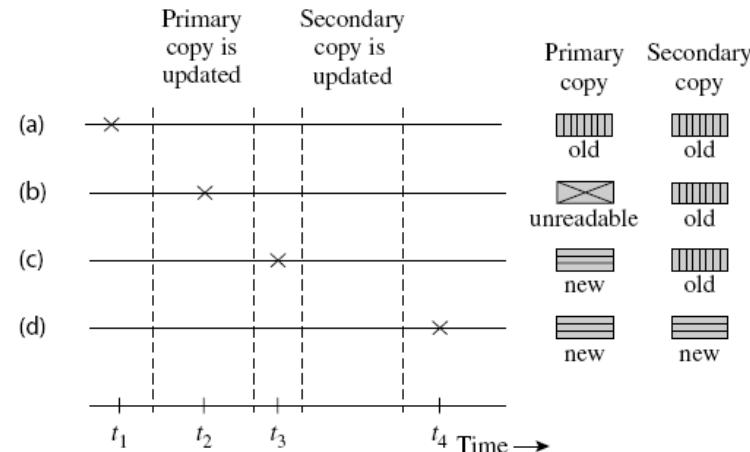


Figure 13.28 Fault tolerance using the stable storage technique.

Azioni atomiche

- Un'azione comporta la manipolazione di molte strutture dati e tali strutture possono diventare inconsistenti se un guasto interrompe l'esecuzione dell'azione

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 memorizzazione stabile)
 - *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 tutte le modifiche della *intention list* sono eseguite



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.

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.)

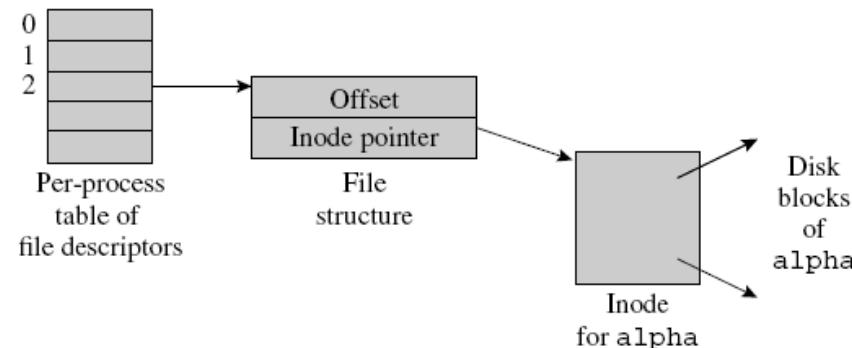


Figure 13.32 Unix file system data structures.

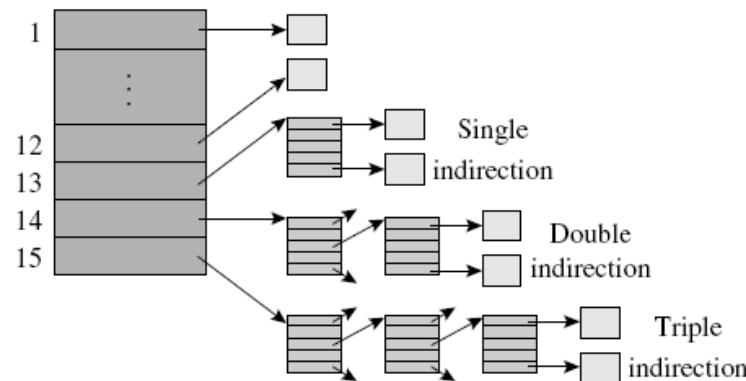


Figure 13.33 Unix file allocation table.

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
- Include 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



UNIVERSITÀ DEGLI STUDI DI NAPOLI
PARTHENOPE

Sistemi Operativi

Input-Output Control System

LEZIONE 24

Livelli IOCS

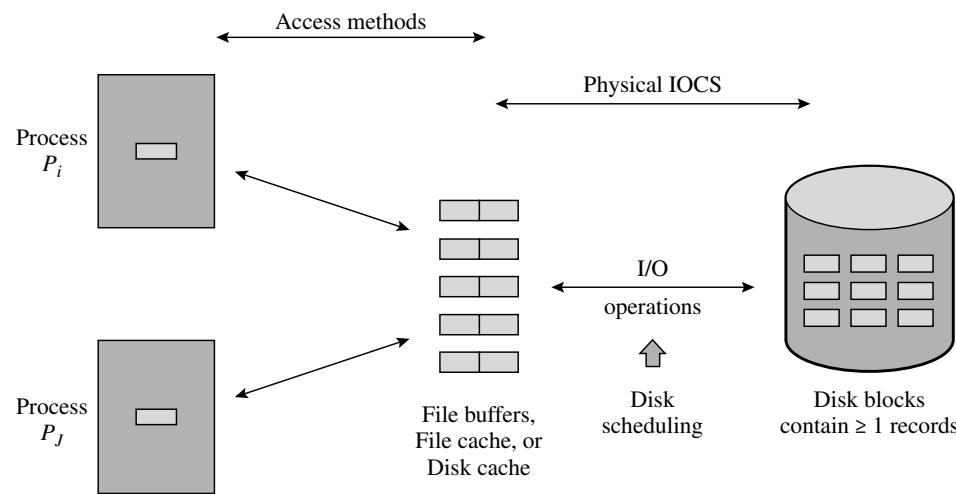


Figure 14.1 Implementation of file operations by the IOCS.

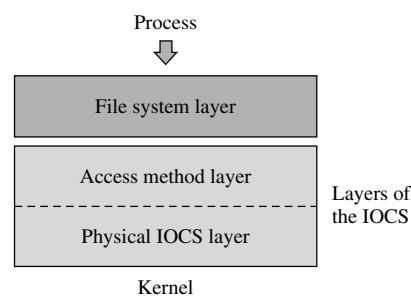
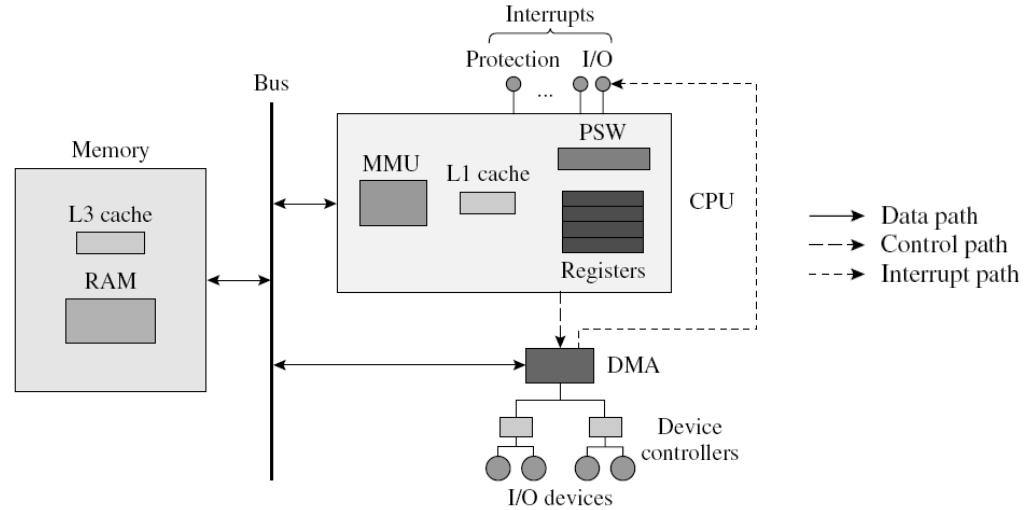


Figure 14.2 Layers of the file system and the IOCS.

Table 14.1 Mechanisms and Policies in File System and IOCS Layers

Physical IOCS	<ul style="list-style-type: none"> <i>Mechanisms:</i> I/O initiation, providing I/O operation status, I/O completion processing, error recovery. <i>Policy:</i> Optimization of I/O device performance through a <i>disk scheduler</i> and a <i>disk cache</i>.
Access methods	<ul style="list-style-type: none"> <i>Mechanisms:</i> File open and close, read and write. <i>Policy:</i> Optimization of file access performance through <i>buffering</i> and <i>blocking</i> of file data and use of a <i>file cache</i>.
File System	<ul style="list-style-type: none"> <i>Mechanisms:</i> Allocation of disk blocks, directory maintenance, setting and checking of file protection information. <i>Policies:</i> Disk space allocation for access efficiency, sharing and protection of files.

Organizzazione dell'I/O



- Il sottosistema di I/O ha un percorso ai dati in memoria indipendente
- I dispositivi sono connessi ai controller dei dispositivi che sono connessi al DMA (Direct Memory Access)
 - un dispositivo è identificato dalla coppia (*controller id, device id*)
- Il DMA, un controller di dispositivo e il dispositivo implementano un'operazione di I/O



Operazioni di I/O

- Un'operazione di I/O coinvolge:
 - Operazione da eseguire: read, write ecc
 - Indirizzo del dispositivo di I/O
 - Numero di byte di dati da trasferire
 - Indirizzi delle aree in memoria e sul dispositivo di I/O coinvolte nel trasferimento
- La CPU avvia l'operazione di I/O mediante l'esecuzione di un'istruzione di I/O, ma non è coinvolta nel trasferimento
 - L'istruzione di I/O punta ad un insieme di comandi di I/O
 - Singole azioni coinvolte nel trasferimento dati
 - L'esecuzione di tali azioni è compito del DMA, del controller del dispositivo e del dispositivo di I/O
 - La CPU è libera di fare altro mentre l'operazione di I/O è in atto

Operazioni di I/O (cont.)

- Operazione di I/O *read* da un blocco del disco (*track_id*, *block_id*) eseguita mediante l'istruzione
 - I/O-init(controller_id, device_id), I/O_command_addr*
 - *I/O_command_addr* è l'indirizzo di partenza dell'area di memoria che contiene i seguenti comandi:
 1. Posiziona la testina del disco sulla traccia *track_id*
 2. Leggi il record *record_id* nell'area di memoria con indirizzo di partenza *memory_add*

Third Party DMA

- Quando è eseguita un'istruzione di I/O
 - Il controller del DMA passa i dettagli dei comandi di I/O al controller del dispositivo di I/O
 - Il dispositivo consegna i dati al controller di dispositivo
 - Il trasferimento dei dati da controller del dispositivo a memoria avviene come segue
 - Il controller del dispositivo invia un segnale **DMA request** quando è pronto al trasferimento
 - Il DMA, ricevuto il segnale, ottiene il controllo del bus e vi pone l'indirizzo di memoria che partecipa al trasferimento. Infine, invia un **DMA ack** al controller del dispositivo
 - Il controller del dispositivo trasferisce i dati verso o dalla memoria
 - Alla fine del trasferimento, il controller del DMA genera un interrupt di completamento I/O con codice uguale all'indirizzo del dispositivo
 - La routine di servizio degli interrupt analizza il codice per trovare quale dispositivo ha completato la sua operazione di I/O e intraprende le azioni appropriate

Dispositivi di I/O

- Esistono differenti tipologie di dispositivi di I/O che funzionano sulla base di vari principi fisici
 - Generazione di segnali elettromeccanici
 - Memorizzazione dati ottica o elettromagnetica
- I dispositivi di I/O possono essere classificati sulla base dei seguenti criteri:
 - **Scopo**: dispositivi di input, di stampa e di memorizzazione
 - **Natura dell'accesso**
 - Sequentiale: tastiera, mouse, rete, nastro
 - Casuale: dischi
 - **Modalità trasferimento dati**: caratteri o blocchi
- L'informazione letta o scritta in un comando di I/O costituisce un record

Dispositivi di I/O: modalità trasferimento dati

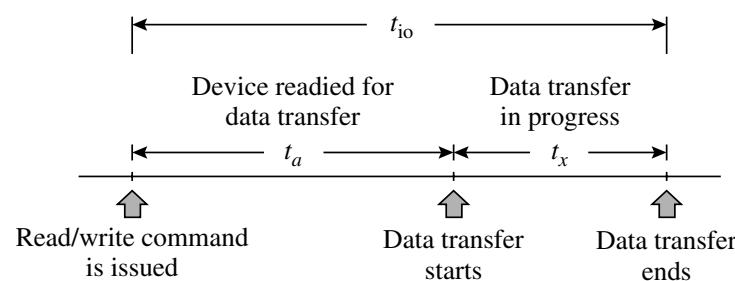
- Dipende dalla velocità di trasferimento
 - Dispositivo di **I/O lento** (tastiera, mouse e stampante sono dispositivi a carattere)
 - **Lavora nella modalità carattere**: è trasferito un carattere per volta tra memoria e periferica
 - Contiene un buffer che memorizza il carattere
 - Il controller genera un interrupt in conseguenza di una lettura dal buffer (dispositivo di input) o di una scrittura nel buffer (dispositivo di output)
 - I **controller** possono essere connessi direttamente al bus
 - Dispositivi di **I/O veloci** (nastri, dischi)
 - **Lavora in modalità a blocco**
 - **Connesso ad un controller di DMA**
 - Devono trasferire i dati a specifiche velocità
 - I dati sono trasferiti tra la periferica di I/O e un *buffer del DMA*



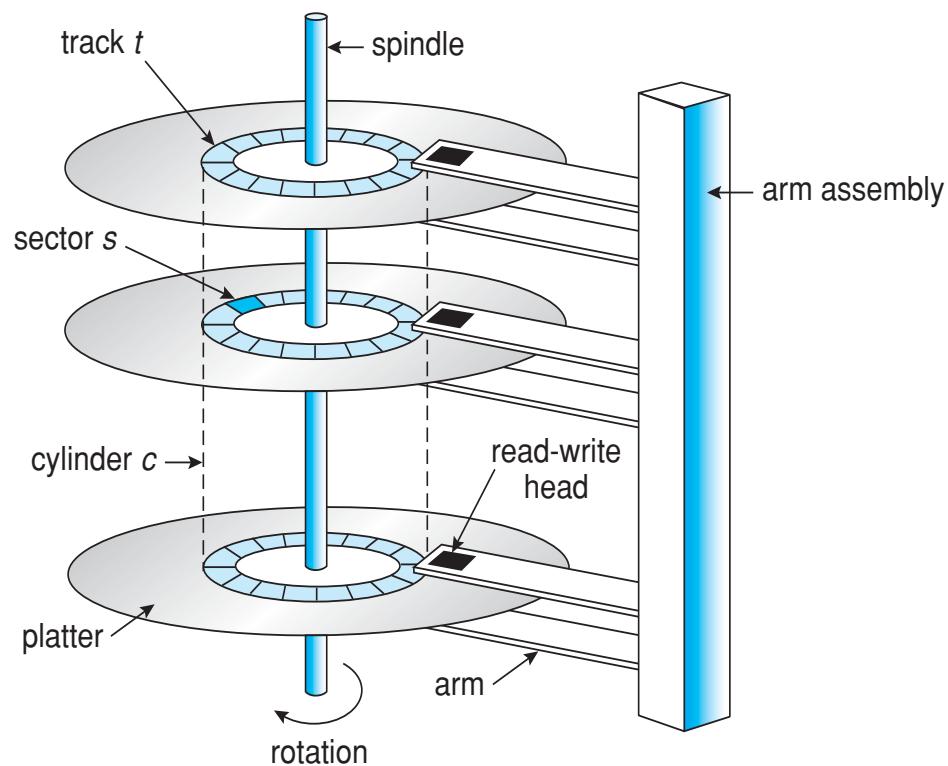
Dispositivi di I/O: tempo di accesso e tempo di trasferimento

- t_{io} (tempo di I/O) -> intervallo tra esecuzione istruzione inizio I/O e completamento operazione
- t_a (tempo di accesso) -> intervallo tra un comando read o write e l'inizio del trasferimento
- t_x (tempo di trasferimento) -> tempo necessario per trasferire dati da/verso una periferica durante un'operazione read o write (inizio trasferimento primo byte, fine trasferimento ultimo byte)

$$t_{io} = t_a + t_x$$



Struttura di un Hard Disk



Disco magnetico

- L'elemento di memorizzazione è un oggetto circolare chiamato **piatto** che ruota intorno al proprio **asse**
 - La superficie circolare è ricoperta di materiale magnetico
- Una **singola testina di lettura-scrittura** registra e legge dalla superficie
 - Un byte è memorizzato in modo seriale lungo una **traccia circolare** sulla superficie del disco
 - La testina può muoversi **radialmente** lungo il piatto
 - E' marcata la posizione di avvio su ogni traccia ed ai record di una traccia sono attribuiti numeri seriali rispetto a tale posizione
 - Il disco può accedere ad ogni record con indirizzo (**numero traccia, numero record**)

Disco magnetico (cont.)

- Il tempo di accesso è:
 - $t_a = t_s + t_r$
 - t_s **tempo di ricerca**, tempo per posizionare la testina sulla traccia richiesta
 - t_r **latenza rotazionale**, tempo per accedere al settore desiderato sulla traccia
- La latenza rotazionale media è il tempo richiesto per una rivoluzione di metà del disco
 - 3-4 ms
- Maggiori capacità sono ottenute montando molti piatti
 - Una testina di lettura e scrittura per ogni superficie circolare del piatto
 - Una testina sopra ed una sotto
 - Tutte le testine sono montate su un singolo braccio (attuatore)
 - Tutte le testine sono posizionate sulle stesse tracce di superfici diverse



Disco magnetico (cont.)

- Cilindro
 - Consiste di tracce posizionate in modo uguale su tutti i piatti di un disco
 - Tutte le sue tracce possono essere accedute dalla stessa posizione della testina
 - L'uso riduce il movimento della testina
 - Pone dati adiacenti di un file su tracce dello stesso cilindro
- Indirizzo di un record: (*numero cilindro, numero superficie, numero record*)
- Per ottimizzare l'uso della superficie del disco le tracce sono organizzate in settori
 - Slot di dimensione standard in una traccia per un record
 - Dimensione scelta per minimizzare lo spreco di capacità di memorizzazione
- La suddivisione in settori può essere parte dello hw (hard sectoring) o implementata da software (soft sectoring)

Organizzazione dei dati su disco

- I dati devono essere organizzati in modo da garantire un accesso efficiente
- Il tempo di trasferimento, di commutazione e di ricerca possono crescere se non si adottano misure preventive
 - Tecniche di distribuzione dei dati

Scheduling del Disco

- First-come, First-Served (FCFS)
 - Seleziona l'operazione di I/O richiesta prima
- Shortest Seek Time First (SSTF)
 - Seleziona l'operazione di I/O il cui tempo di ricercar dalla posizione corrente della testina è più breve
- SCAN
 - Sposta la testina da un'estremità all'altra del piatto, servendo le operazioni di I/O per i blocchi sulla traccia o cilindro durante lo spostamento. Raggiunta un'estremità procede nel senso inverso
- Circular SCAN (C-SCAN)
 - come SCAN, tuttavia quando arriva all'estremità, non esegue la scansione inversa ma riposiziona la testina all'estremità opposta
- LOOK e C-LOOK
 - come SCAN e C-SCAN, con la differenza che non si raggiunge l'estremità del piatto ma solo l'ultima richiesta in quella direzione



Richieste di I/O per lo scheduling del disco

$$t_{\text{hm}} = t_{\text{const}} + |\text{track}_1 - \text{track}_2| \times t_{\text{pt}}$$

t_{const} and t_{pt}	= 0 ms and 1 ms, respectively
Current head position	= Track 65
Direction of last movement	= Toward higher numbered tracks
Current clock time	= 160 ms

Requested I/O operations:

Serial number	1	2	3	4	5
Track number	12	85	40	100	75
Time of arrival	65	80	110	120	175

- Le testine del disco si muovono verso le tracce con numeri maggiori
- Le richieste sono fatte in tempi diversi

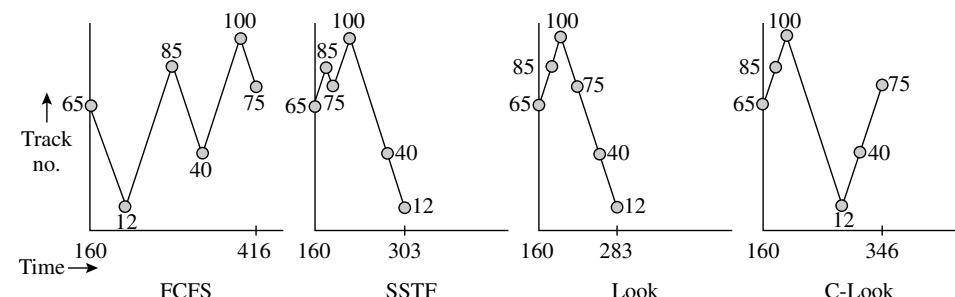
Dettagli scheduling del disco

Policy	Details	Scheduling decisions					Σ Seek time
		1	2	3	4	5	
FCFS	Time of decision	160	213	286	331	391	256
	Pending requests	1, 2, 3, 4	2, 3, 4, 5	3, 4, 5	4, 5	5	
	Head position	65	12	85	40	100	
	Selected request	1	2	3	4	5	
	Seek time	53	73	45	60	25	
SSTF	Time of decision	160	180	190	215	275	143
	Pending requests	1, 2, 3, 4	1, 3, 4, 5	1, 3, 4	1, 3	1	
	Head position	65	85	75	100	40	
	Selected request	2	5	4	3	1	
	Seek time	20	10	25	60	28	
Look	Time of decision	160	180	195	220	255	123
	Pending requests	1, 2, 3, 4	1, 3, 4, 5	1, 3, 5	1, 3	1	
	Head position	65	85	100	75	40	
	Selected request	2	4	5	3	1	
	Seek time	20	15	25	35	28	
C-Look	Time of decision	160	180	195	283	311	186
	Pending requests	1, 2, 3, 4	1, 3, 4, 5	1, 3, 5	3, 5	5	
	Head position	65	85	100	12	40	
	Selected request	2	4	1	3	5	
	Seek time	20	15	88	28	35	

t_{const} and t_{pt} = 0 ms and 1 ms, respectively
 Current head position = Track 65
 Direction of last movement = Toward higher numbered tracks
 Current clock time = 160 ms

Requested I/O operations:

Serial number	1	2	3	4	5
Track number	12	85	40	100	75
Time of arrival	65	80	110	120	175



$$t_{\text{hm}} = t_{\text{const}} + |\text{track}_1 - \text{track}_2| \times t_{\text{pt}}$$

Selezione di un Algoritmo di Scheduling

- SSTF è piuttosto comune e attrattivo
- SCAN e C-SCAN sono più performanti per i sistemi che impongono un carico elevato sul disco
- Le prestazioni dipendono dal numero e dal tipo di richieste
- L'algoritmo di scheduling del disco dovrebbe essere scritto come modulo separato del sistema operativo, in modo da poterlo sostituire con un algoritmo diverso, se necessario
- Un algoritmo predefinito ragionevole può essere SSTF o LOOK