

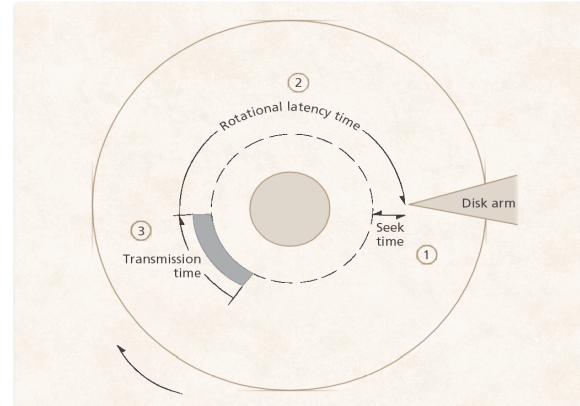
Necessità dello Scheduling del disco

- Scheduling **First-come-first-served** (FCFS) : notevoli svantaggi
 - La ricerca di dati con posizioni distribuite in modo casuale produce lunghi tempi di attesa e scarso throughput
 - Sotto carico pesante, il sistema può comportarsi in modo critico (*thrashing*)
- Le richieste devono essere **servite in ordine logico** per **minimizzare i ritardi**
 - Servire le richieste che richiedono il **minimo** movimento meccanico
- I primi algoritmi di scheduling del disco si sono concentrati sulla **minimizzazione del tempo di seek**, la componente del tempo di accesso al disco con maggior latenza
- I moderni sistemi ottimizzano **anche il tempo di rotazione**

S. Balsamo – Università Ca' Foscari Venezia – SO.5.40

Necessità dello Scheduling del disco

Componenti del tempo di accesso ad un disco



S. Balsamo – Università Ca' Foscari Venezia – SO.5.41

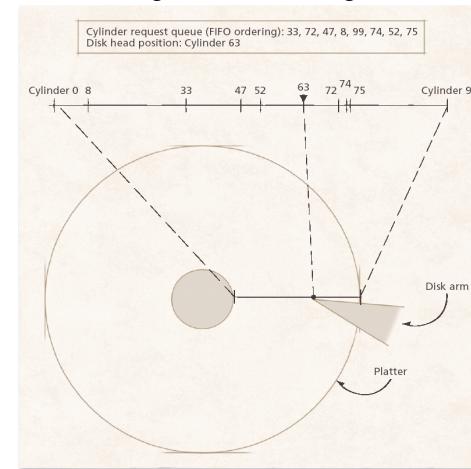
Strategie di Scheduling del disco

- Tre criteri per misurare le strategie
 - Throughput**
 - Numero di richieste servite per unità di tempo
 - Tempo medio di risposta**
 - Tempo medio di attesa che la richiesta sia servita e di servizio
 - Varianza** del tempo di risposta
 - Misura della prevedibilità del tempo di risposta
- Obiettivi generali
 - Massimizzare** il throughput
 - Minimizzare** il tempo di risposta e la varianza di tempi di risposta

S. Balsamo – Università Ca' Foscari Venezia – SO.5.42

Strategie di Scheduling del disco

Cylinder request queue (FIFO ordering): 33, 72, 47, 8, 99, 74, 52, 75
 Disk head position: Cylinder 63



Modello
di richieste
di accesso
ad un disco

S. Balsamo – Università Ca' Foscari Venezia – SO.5.43

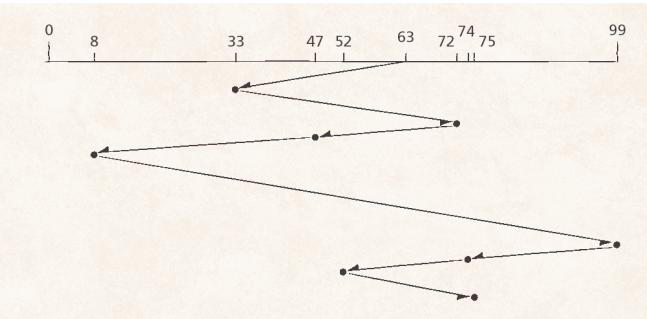
Scheduling del disco First-Come-First-Served (FCFS)

- Richieste servite in ordine di arrivo
 - Vantaggi
 - Equo
 - Previene l'attesa infinita
 - basso overhead
 - Svantaggi
 - Possibile throughput estremamente basso
 - FCFS in genere porta ad un modello di ricerca di seek casuale perché non riordina le richieste per minimizzare il ritardo di servizio

S. Balsamo – Università Ca' Foscari Venezia – SO.5.44

Scheduling del disco First-Come-First-Served (FCFS)

Operazioni di seek con la strategia FCFS



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63

S. Balsamo – Università Ca' Foscari Venezia – SO.5.45

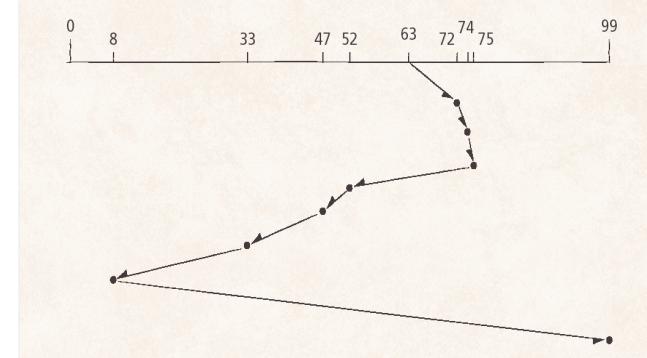
Shortest-Seek-Time-First

- SSTF: richiesta di servizio più vicina alla testina di lettura-scrittura
 - vantaggi
 - throughput maggiore e tempi di risposta inferiori rispetto a FCFS
 - soluzione ragionevole per i sistemi di elaborazione batch
 - svantaggi
 - Non garantisce equità
 - Possibilità di **attesa infinita**
 - Alta varianza dei tempi di risposta
 - Il tempo di risposta generalmente inaccettabile per sistemi interattivi

S. Balsamo – Università Ca' Foscari Venezia – SO.5.46

Shortest-Seek-Time-First

Operazioni di seek con la strategia SSTF



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63

S. Balsamo – Università Ca' Foscari Venezia – SO.5.47

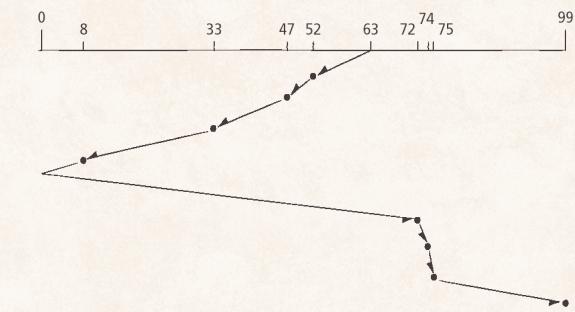
Scheduling del disco SCAN

- SCAN: tempo più breve di seek **in una direzione preferita**
 - Non cambia direzione fino a quando non si è raggiunto il limite del disco
 - Algoritmo dell'**ascensore**
 - Caratteristiche simili a SSTF
 - Attesa infinita ancora possibile
 - Migliora la **varianza dei tempi di risposta**

S. Balsamo – Università Ca' Foscari Venezia – SO.5.48

Scheduling del disco SCAN

Operazioni di seek con la strategia SCAN



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63

S. Balsamo – Università Ca' Foscari Venezia – SO.5.49

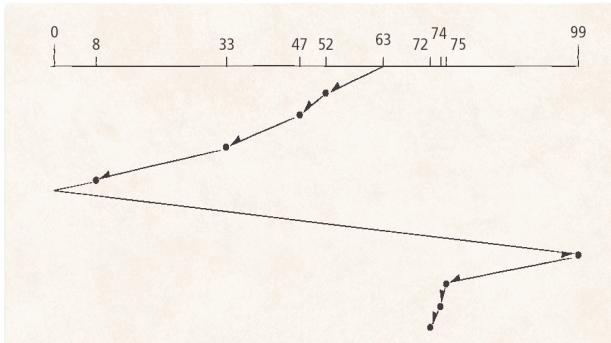
Scheduling del disco C-SCAN

- C-SCAN: (Circolare):
 - simile a SCAN, ma alla fine di una scansione verso l'interno, il braccio del disco salta (senza servire richieste) al cilindro più esterno
 - si muove sempre nella stessa direzione per servire le richieste
 - Ulteriore **riduzione della varianza dei tempi di risposta**, a scapito del throughput e del tempo medio di risposta

S. Balsamo – Università Ca' Foscari Venezia – SO.5.50

Scheduling del disco C-SCAN

Operazioni di seek con la strategia C-SCAN



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63

S. Balsamo – Università Ca' Foscari Venezia – SO.5.51

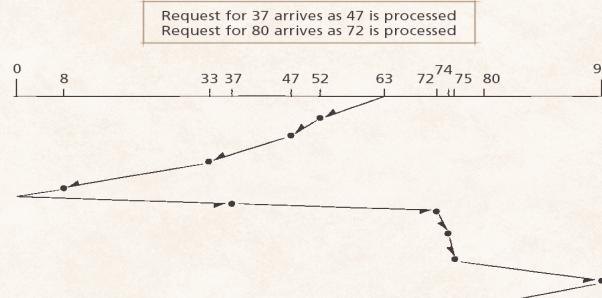
Scheduling del disco FSSCAN e N-Step SCAN

- Gruppi batch di richieste in
- **FSCAN**
"congela" periodicamente la coda di richieste al disco e serve solo le richieste in coda in quel momento
- **N-Step SCAN:**
serve solo le prime n richieste nella coda in quel momento
 - Entrambe le strategie prevengono l'attesa infinita
 - Entrambe riducono la varianza dei tempi di risposta rispetto a SCAN

S. Balsamo – Università Ca' Foscari Venezia – SO.5.52

Scheduling del disco FSSCAN e N-Step SCAN

Operazioni di seek con la strategia FSCAN

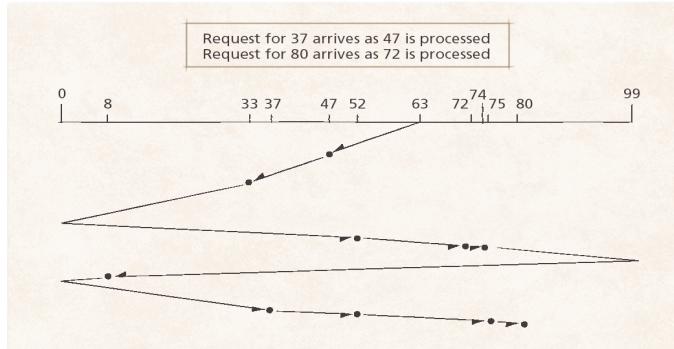


Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63

..37, ...80

© Deitel & Ass. Inc. S. Balsamo – Università Ca' Foscari Venezia – SO.5.53

Scheduling del disco FSSCAN e N-Step SCAN

Operazioni di seek con la strategia N-Step SCAN ($n=3$)

Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63

..37, ...80

S. Balsamo – Università Ca' Foscari Venezia – SO.5.54

Scheduling del disco LOOK e C-LOOK

- **LOOK:** Migliora lo scheduling SCAN

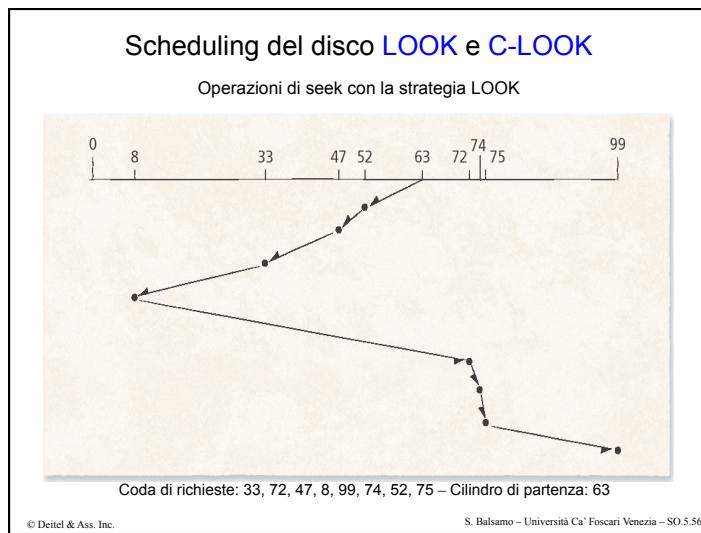
- (*look ahead*) continua fino al termine dell'attraversamento attuale per servire richieste, se non ci sono cambia direzione

- Muove il braccio del disco verso il bordo esterno del disco se non ci sono richieste pendenti per tali regioni
- Migliora l'efficienza evitando operazioni inutili di ricerca
- Throughput elevato

- **C-LOOK** migliora lo scheduling C-SCAN

- Combinazione di LOOK e C-SCAN
- Quando non ci sono richieste nell'attraversamento verso l'interno si sposta verso le richieste posizionate più all'esterno senza servirne altre in mezzo e inizia un nuovo attraversamento
- Minor varianza dei tempi di risposta di LOOK, a scapito della throughput

S. Balsamo – Università Ca' Foscari Venezia – SO.5.55



Scheduling del disco – sintesi e confronto

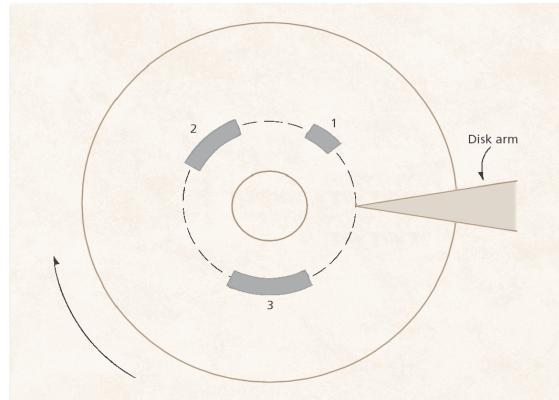
Strategia	Descrizione
FCFS	Serve le richieste in ordine di arrivo
SSTF	Serve per prime le richieste con minor distanza di seek
SCAN	La testina si sposta avanti e indietro e serve secondo SSTF
C-SCAN	La testina si sposta avanti e serve secondo SSTF in quella direzione, arrivata all'interno salta a quella più esterna e ripete
FSCAN	Come SCAN eccetto le nuove rinviate al successivo attraversamento
SCAN n-STEPS	Come FSCAN ma serve solo n richieste per attraversamento. Evita l'attesa infinita
LOOK	Come SCAN, ma la testina cambia direzione quando raggiunge l'ultima richiesta nella direzione preferenziale
C-LOOK	Come C-SCAN, ma la testina si ferma quando raggiunge l'ultima richiesta nella direzione preferenziale, serve la richiesta al cilindro più vicino al lato opposto del disco

- Ottimizzazione rotazionale**
- Il tempo di ricerca (*seek*) precedentemente determinava i problemi di prestazioni
 - Se i tempi di ricerca e la latenza rotazionale sono dello stesso ordine di grandezza
 - strategie sviluppate di recente tentano di ottimizzare le prestazioni del disco riducendo la latenza rotazionale
 - Importante quando si accede a piccoli porzioni di dati distribuiti casualmente in tutto il disco
- S. Balsamo – Università Ca' Foscari Venezia – SO.5.58

- Scheduling SLTF**
- Shortest-latency-time-first scheduling
 - In un dato cilindro, serve le richieste con la minima latenza di rotazione
 - Facile da implementare
 - Accodamento dei settori
 - Raggiunge prestazioni quasi ottimali per la latenza rotazionale
- S. Balsamo – Università Ca' Foscari Venezia – SO.5.59

Scheduling SLTF

SLTF scheduling: le richieste sono servite nell'ordine indicato senza considerare l'ordine di arrivo



© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.60

Scheduling SPTF e SATF

- Shortest-positioning-time-first scheduling
 - Tempo di posizionamento: somma di tempo di ricerca e la latenza di rotazione
 - SPTF serve per prima la richiesta con il minimo tempo di posizionamento
 - Buone prestazioni
 - Può causare attesa infinita (cilindri più sui bordi)

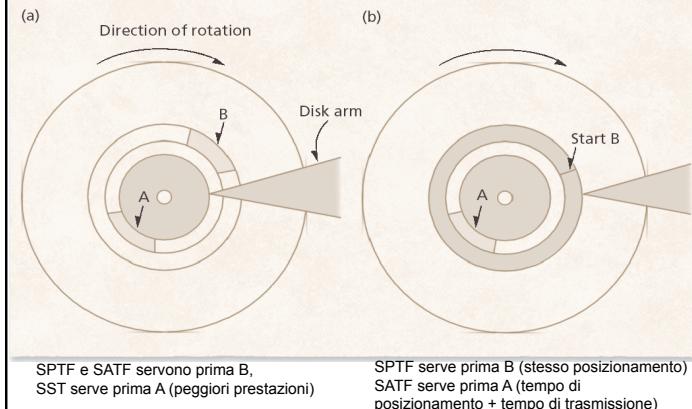
S. Balsamo – Università Ca' Foscari Venezia – SO.5.61

Scheduling SPTF e SATF

- Shortest-access-time-first scheduling
 - Variante di SPTF
 - **Tempo di accesso:** tempo di posizionamento più il tempo di trasmissione
 - Throughput elevato
 - Anche in questo caso, può causare attesa infinita
- Sia SPTF e SATF possono implementare LOOK (*ahead*) per migliorare le prestazioni
- Svantaggio
 - Sia SPTF e SATF richiedono la conoscenza delle caratteristiche di prestazioni del disco che potrebbero non essere immediatamente disponibili (tempi di seek, latenza, posizioni dei settori)
 - Possibile mascheramento al S.O. per controllo degli errori, correzione dei dati e riassegnazione trasparente dei settori danneggiati

S. Balsamo – Università Ca' Foscari Venezia – SO.5.62

Scheduling SPTF e SATF - esempio



SPTF e SATF servono prima B,
SST serve prima A (peggiori prestazioni)

SPTF serve prima B (stesso posizionamento)
SATF serve prima A (tempo di posizionamento + tempo di trasmissione)

S. Balsamo – Università Ca' Foscari Venezia – SO.5.63

Considerazioni di sistema

- Lo scheduling del disco è spesso utile, ma non sempre
 - Non aiuta sensibilmente nei sistemi processor-bound
 - Tipologie di **carico**: beneficio all'aumentare della multiprogrammazione e casualità
 - Richieste al disco in sequenze imprevedibili
 - Esempi**: archiviazione in reti locali, uso di database e web server, molti utenti e richieste piccole (Online Transactions Processing)
 - Per **distribuzioni non uniformi delle richieste**, l'overhead dello scheduling può ridurre le prestazioni
 - Esempio**: archiviazione di file posizionati su cilindri adiacenti
 - Le tecniche di **organizzazione dei file** a volte contrastano algoritmi di scheduling
 - Geometria reale e **geometria virtuale** possono vanificare i vantaggi degli algoritmi di scheduling

S. Balsamo – Università Ca' Foscari Venezia – SO.5.64

Caching e Buffering

- Buffer Cache**: memorizzazione di una copia dei dati su disco in memoria più veloce
 - Situato in **memoria principale**, onboard cache, o sul controller del disco
 - Tempi di accesso molto minori dell'accesso al disco
 - Può essere usato come un **buffer** per ritardare la scrittura dei dati finché disco è sotto carico leggero
- Potenziale incoerenza
 - Il contenuto della memoria principale potrebbe essere perso per mancanza di corrente o guasto del sistema

S. Balsamo – Università Ca' Foscari Venezia – SO.5.65

Caching e Buffering

Gestione della possibile incoerenza

- Cache write-back** (scrivi alla fine)
 - i dati non sono scritti su disco immediatamente, ma accorpati
 - Migliora le prestazioni
 - Periodicamente flushed verso il disco
- Write-through caching** (scrivi subito)
 - Scrive contemporaneamente su disco e cache
 - Riduce le prestazioni rispetto al write-back, ma garantisce la coerenza

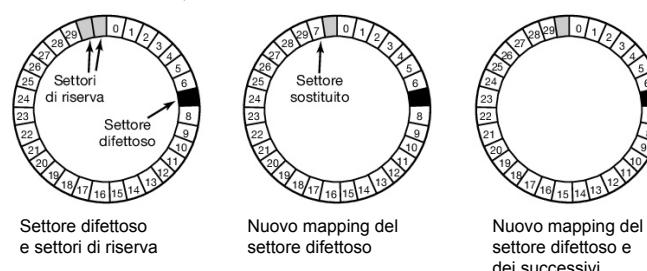
Possibile **cache preventivo** di più settori da parte del controller

Nota: **cache del controller** indipendente dalla cache del sistema operativo per blocchi non richiesti ma letti per convenienza della posizione

S. Balsamo – Università Ca' Foscari Venezia – SO.5.66

Gestione degli errori

>Errori su una traccia, di settore



Traccia di un disco con un settore con errore
Ridefinizione del mapping dei settori. Uso di tabelle interne, una per traccia.

>Errori su un cilindro, di posizionamento

Possibile ricalibratura

A. Tanenbaum – Modern Operating Systems

Gestione degli errori

Problemi di affidabilità - uso di RAID

Problemi: **crash e errori durante la lettura** con corruzione dei dati

Memoria stabile: sottosistema disco che o scrive correttamente o non esegue niente.

Assumendo di disporre di una coppia di dischi identici D1 e D2 e corrispondenti.

- Operazione dei **scrittura stabile**

scrive il blocco in D1, se non è corretto ripete n volte
se fallisce n volte mappa il blocco con uno di riserva e ripete
eventualmente ripete finché non completa
copia sul blocco corrispondente di D2
(scrittura corretta senza *crash* della CPU)

- Operazione dei **lettura stabile**

legge il blocco da D1, se non è corretto ripete n volte
se fallisce n volte legge da D2
(probabilità di doppio errore trascurabile)

- Ripristino da **crash**

scansione e confronto dei blocchi da D1 e D2
se uguali e validi è completato
se uno è errato viene riscritto con l'altro corrispondente
se validi ma diversi D1 sovrascrive D2

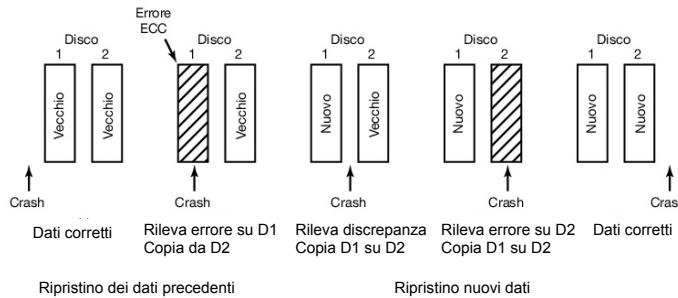
A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.68

Gestione degli errori

Senza crash sono presenti due copie valide

In caso di **crash della CPU** diverse possibili situazioni per il ripristino



A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.69

Altre tecniche di ottimizzazione di prestazioni del disco

- Frammentazione di file e record a seguito di modifiche (aggiunte/rimozioni)
- **Deframmentazione** (riorganizzazione del disco)
 - Applicati **periodicamente**
 - Inserire i dati in relazione in **settori contigui**
 - Diminuisce il numero di operazioni di ricerca richiesto
 - Usare allocazione adiacente a spazio libero per dati frequenti o in espansione
 - Il partizionamento può aiutare a ridurre la frammentazione (file memorizzati in partizioni di disco)
- **Compressione**
 - I dati consumano meno spazio su disco
 - Migliora i tempi di trasferimento e di accesso
 - Maggiore overhead del tempo di esecuzione per la compressione / decompressione

S. Balsamo – Università Ca' Foscari Venezia – SO.5.70

Altre tecniche di ottimizzazione di prestazioni del disco

- **Copie multiple** di dati richiesti più frequentemente
 - Diverse posizioni del disco
 - Accesso alla **copia più vicina** alla testina di lettura-scrittura
 - Minor tempo di ricerca e di rotazione
 - Può comportare **overhead** significativi di memoria
 - Adatta per dati in sola lettura o rare modifiche (congruenza delle copie)
- Accorpamento di record (**blocking**)
 - Leggere / scrivere più record come **unico blocco** di dati
 - Riduce i tempi
- **Anticipazione del braccio del disco**
 - Quando inattivo, sposta il braccio del disco nella **posizione** dove è **maggior probabilità del prossimo accesso** ai dati, o al centro
 - Minor tempo di attesa specie in caso di località nella zona prevista
 - Se il braccio del disco predice in modo non corretto il prossimo accesso al disco, le prestazioni possono subire forti degradazioni
 - Meno efficace con la multiprogrammazione

S. Balsamo – Università Ca' Foscari Venezia – SO.5.71

Software per I/O

Input: da [tastiera](#) e da [mouse](#)

Output: verso [finestre](#) di testo, [interfacce grafiche \(GUI\)](#)

Il driver della tastiera fornisce un numero
il driver converte in caratteri (*orientata a carattere*)
usa una tabella ASCII
li passa al programma

oppure gestisce una riga (*orientata a riga*) (*in POSIX modalità canonica*)
la passa al programma
caratteri speciali (comandi di controllo e gestione I/O)

Eccezioni, adattamenti necessari per altre lingue
molti sistemi operativi forniscono keymap o codici caricabili

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.5.72

Software per I/O

Carattere	Nome POSIX	Commento
CTRL-H	ERASE	Cancella un carattere prima del cursore
CTRL-U	KILL	Cancella l'intera linea digitata
CTRL-V	LNEXT	Interpreta letteralmente il carattere successivo
CTRL-S	STOP	Ferma l'output
CTRL-Q	START	Avvia l'output
DEL	INTR	Interrompe il processo (SIGINT)
CTRL-\	QUIT	Forza il core dump (SIGQUIT)
CTRL-D	EOF	Fine del file
CTRL-M	CR	A capo (non modificabile)
CTRL-J	NL	Nuova riga (non modificabile)

Caratteri speciali in modalità canonica POSIX

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.5.73

Software per I/O

Output: verso [finestre](#) di testo, [interfacce grafiche \(GUI\)](#)

Finestra di testo
blocco di caratteri (es. una linea)
[editori](#) di schermo più complessi – comandi per gestire il cursore
sequenze di escape – *termcap* pacchetto sw uniforme standard ANSI

Sequenza di escape	Significato
ESC\nA	Muovi verso l'alto di n linee
ESC\nB	Muovi verso il basso di n linee
ESC\nC	Muovi a destra di n spazi
ESC\nD	Muovi a sinistra di n spazi
ESC[m;nH	Muovi il cursore a (m, n)
ESC[sJ	Cancello lo schermo a partire dal cursore (0 fino alla fine, 1 dall'inizio, 2 tutto)
ESC[sK	Cancello la linea a partire dal cursore (0 fino alla fine, 1 dall'inizio, 2 tutto)
ESC[nL	Inserisci n linee a partire dal cursore
ESC[nM	Cancello n linee a partire dal cursore
ESC[nP	Cancello n caratteri a partire dal cursore
ESC[n@	Inserisci n caratteri a partire dal cursore
ESC[nm	Abilita l'interpretazione n (0 = normale, 4 = grassetto, 5 = lampeggiante, 7 = in negativo)
ESCM	Fa scorrere lo schermo indietro se il cursore è alla prima linea

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.5.74

Software per I/O

Sistema [X Window \(X\)](#)
interfaccia per sistemi Unix
sviluppato al M.I.T., portabile, eseguito nello [spazio utente](#)
sistemi sw cliente-servente
possono essere eseguiti sulla stessa macchina
o su macchine diverse
guidato da [eventi](#)
input: tastiera e mouse (X client)
output: schermo (X server)
es. su Linux ambienti desktop Gnome e KDE eseguiti su X

Sistema X non è una GUI completa
Struttura a livelli

Xlib	libreria X: procedure per accedere alle funzioni di X
protocollo X	dialogo fra X client e X server
Intrinsics	strato con strumenti per accedere e usare Xlib <i>es. gestione dei testi, barre di scorrimento, etc – widget</i>
Motif	rende uniforme l'accesso alle funzioni

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.5.75

