

Definizione e storia dei sistemi operativi

Dipartimento di Informatica
Università di Verona, Italy



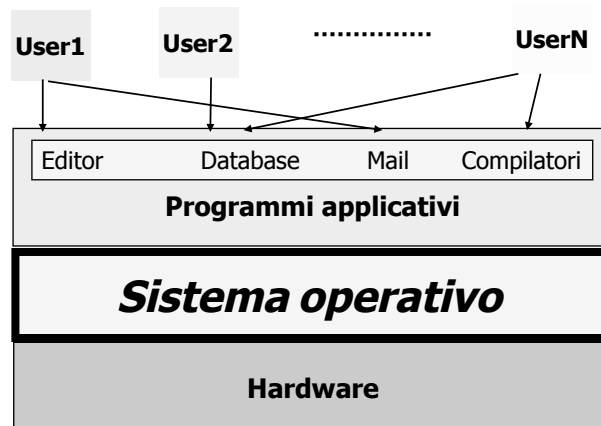
Che cos'è un Sistema Operativo?

- E' un insieme di programmi
 - agisce come intermediario tra HW e uomo
 - per facilitare l'uso del computer
 - per rendere efficiente l'uso dell'HW
 - per evitare conflitti nella allocazione delle risorse HW/SW
- offre un ambiente per *controllare e coordinare* l'utilizzo dell'HW (CPU, memoria, I/O, ...) da parte dei programmi applicativi

Che cos'è un Sistema Operativo?

- Viste di un S.O.
 - Gestore di risorse
 - Risorse HW
 - Dischi, memoria, I/O,...
 - Risorse SW
 - File, programmi,...
 - Programma di controllo
 - Controllo dell'esecuzione dei programmi e del corretto utilizzo del sistema

Il S.O. nel sistema di calcolo



Obiettivi del S.O.

- Astrazione
 - Semplificazione dell'uso del sistema
- Efficienza
 - Quanto costa astrarre?
- Tipicamente in contrasto!
 - Windows (molto astratto, meno efficiente)
 - Unix (meno astratto, più efficiente)

STORIA DEI SISTEMI OPERATIVI

Storia dei S.O.

- 4 generazioni legate ai calcolatori
- Principio ispiratore
 - Aumento dell'utilizzo del processore

1a Generazione (1946-1955)

- Enormi calcolatori a valvole
- Non esiste S.O.
- Operatore = Programmatore
- Accesso alla macchina tramite prenotazione
- Esecuzione da console
 - Programma caricato in memoria un'istruzione alla volta agendo su interruttori
 - Controllo errori su spie della console

1a Generazione (Evoluzione)

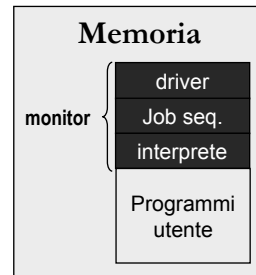
- Diffusione di periferiche (lettore/perforatore di schede, nastri, stampanti)
 - Programmi di interazione con periferiche (device driver)
- Sviluppo di “software”
 - “Librerie” di funzioni comuni
 - Compilatori, linker, loader
- Scarsa efficienza
 - Programmazione facilitata, ma operazioni complesse!
 - tempi di setup elevati → basso utilizzo

2a Generazione (1955-1965)

- Introduzione dei transistor nei calcolatori
- Separazione di operatore e programmatore
 - Eliminazione dello schema a prenotazione
 - Operatore elimina parte dei tempi morti
- Batching
 - Batch = lotto
 - Raggruppamento di programmi (job) simili nell'esecuzione
 - Es:
 - Sequenza job: Fortran, Cobol, Fortran, Cobol
 - Comp. Fortran, linker Fortran, comp. Cobol, linker Cobol, Comp. Fortran, ...
 - Sequenza job: Fortran, Fortran, Cobol, Cobol
 - Comp. Fortran, linker Fortran, comp. Cobol, linker Cobol.
 - Problemi in caso di errori/malfunzionamenti

2a Generazione (Evoluzione)

- Automatic job sequencing
 - Il sistema si occupa di passare da un job all'altro
 - S.O. fa il lavoro dell'operatore e rimuove tempi morti
- Monitor residente
 - primo vero esempio di S.O.
 - monitor = "gestore"
 - residente = perennemente caricato in memoria
 - Componenti del monitor:
 - Driver per dispositivi di I/O
 - Sequenzializzatore dei job
 - Interprete di schede di controllo (lettura ed esecuzione)

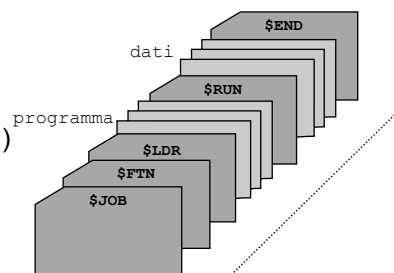


Graziano Pravadelli (2011)

11

2a Generazione (Evoluzione)

- Sequenzializzazione realizzata tramite un linguaggio di controllo (Job Control Language)
 - Schede di controllo
 - Record di controllo (nastri)
 - Esempio:
 - inizia job (\$JOB)
 - fine job (\$END)
 - compilatore (es. Fortran) (\$FTN)
 - linker (\$LNK)
 - loader (\$LDR)
 - ...



Graziano Pravadelli (2011)

12

2a Generazione (Limitazioni)

- Problema:
 - Utilizzo del sistema ancora basso
 - Velocità I/O \ll Velocità CPU
- Es:
 - Compilatore può processare 300 schede/sec
 - Velocità lettore di schede: 20 schede/sec
 - Lettura di un programma di 1200 schede:
 - 4 sec CPU
 - 60 sec I/O
 - Utilizzazione CPU = $4/64 \rightarrow 6.25\%$
- Osservazione: CPU mai attiva durante I/O
 - E' possibile superare questa limitazione?

2a Generazione (Evoluzione)

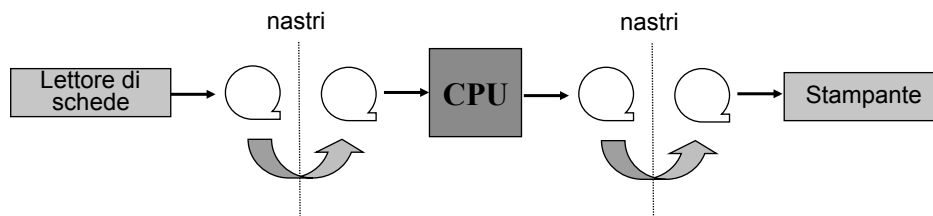
- Soluzione: Sovrapposizione delle operazioni di I/O ed elaborazione (CPU)
- Elaborazione off-line
 - Diffusione dei nastri magnetici
 - più capienti e più veloci
 - Sovrapposizione di I/O e CPU su "macchine" indipendenti
 - Da scheda a nastro su una macchina
 - Da nastro a CPU su un'altra macchina
- CPU ora limitata dalla velocità dei nastri!
 - Vero vantaggio nel caso di più lettori di nastro

Batch vs. operazioni off-line

- Batch tradizionale



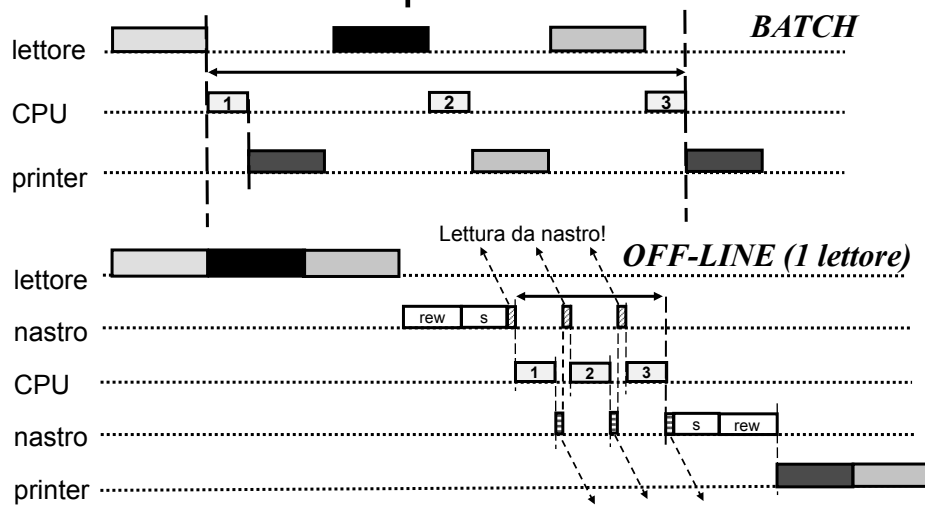
- Operazioni off-line (1 lettore)



Graziano Pravadelli (2011)

15

Batch vs. operazioni off-line



Graziano Pravadelli (2011)

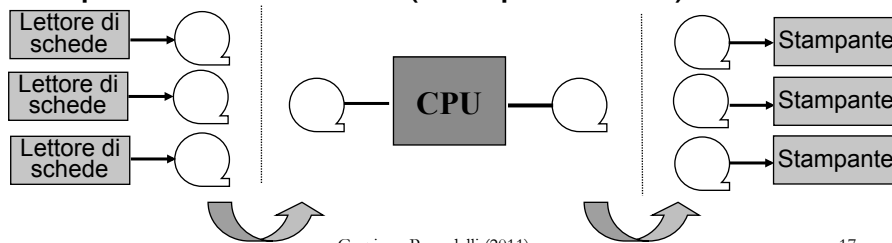
16

Batch vs. operazioni off-line

- Batch tradizionale



- Operazioni off-line (con più lettori)



Graziano Pravadelli (2011)

17

Sovrapposizione di CPU e I/O

- Operazioni off-line
 - sovrapposizione di I/O e CPU su macchine indipendenti
- E' possibile sovrapporli sulla stessa macchina?
 - Sì, ma serve opportuno supporto architetturale

Graziano Pravadelli (2011)

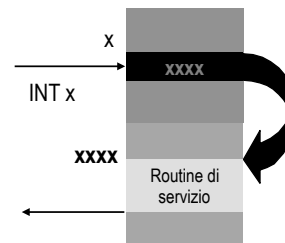
18

Sovrapposizione di CPU e I/O

- Meccanismo di interazione tradizionale tra CPU e I/O: Polling
 - Interrogazione continua del dispositivo tramite esplicite istruzioni bloccanti
- Per sovrapporre CPU e I/O è necessario un meccanismo asincrono (richiesta I/O non bloccante)
 - Interruzioni (Interrupt)
 - DMA (Direct Memory Access)

Interrupt & I/O

- Schema concettuale (semplificato)
 1. CPU (driver) “programma” il dispositivo (cosa deve fare)
 2. Contemporaneamente
 - Dispositivo (controllore) esegue
 - CPU prosegue elaborazione (se possibile)
 3. Dispositivo segnala la fine dell’elaborazione alla CPU
 4. CPU riceve segnale di **interrupt** (tramite segnale esplicito)
 - a. Interrompe l’istruzione corrente (salvando lo stato)
 - b. Salta a una locazione predefinita (X)
 - c. “Serve” l’interruzione (trasferimento dati)
 - d. Riprende l’istruzione interrotta



DMA e I/O

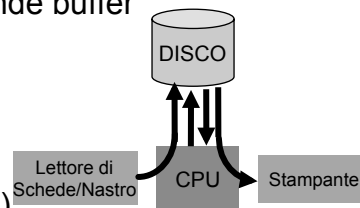
- Nel caso di dispositivi veloci (es. dischi), gli interrupt sono molto frequenti → inefficienza
- Soluzione: DMA (Direct Memory Access)
 - Uno specifico controllore HW (DMA controller) si occupa del trasferimento di blocchi di dati tra I/O e memoria senza interessare la CPU
 - Un solo interrupt per blocco di dati

Buffering & spooling

- Buffering = sovrapposizione di CPU e I/O dello stesso job
 - Dispositivo di I/O legge/scrive più dati di quanti richiesti
 - Utile quando la velocità dell'I/O e della CPU sono simili
 - Nella realtà i dispositivi di I/O sono più lenti della CPU
 - miglioramento marginale
- Spooling = sovrapposizione di CPU e I/O di job diversi
 - Problema:
 - i nastri magnetici sono sequenziali (lettore di schede non può scrivere su un'estremità del nastro mentre la CPU legge dall'altra)
 - Soluzione:
 - introduzione dei dischi magnetici ad accesso casuale

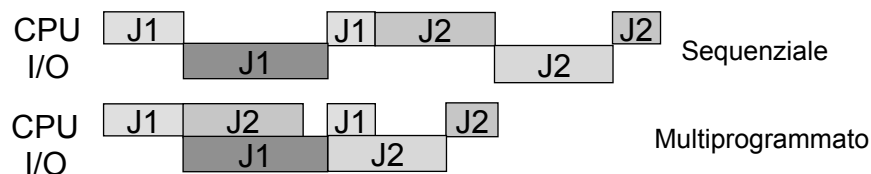
Spooling

- Simultaneous Peripheral Operations On-line
- Possibile grazie “ad accesso casuale” dei dischi
 - Utilizzo del disco come un grande buffer
 - Buffer unico per tutti i job
- Concetto un “pool di job”
 - Nasce il paradigma “moderno” di programma su disco (che viene caricato in memoria)
 - Nasce il concetto di job scheduling
 - Chi deve/può essere caricato sul disco?



3a Generazione (1965-1980)

- Introduzione della multiprogrammazione e dei circuiti integrati
 - Un singolo job non potrà mai tener sufficientemente occupata la CPU
 - Necessaria la “competizione” di più job
 - presenza di più job in memoria
 - Sfruttamento delle fasi di attesa (I/O) per l'esecuzione di un nuovo job



Batch vs. Multiprogrammazione

- Con la presenza di più job nel sistema è possibile modificare la “natura” dei S.O.
 - Sistemi tradizionali (batch):
 - tendenza alla non interattività
 - importante il tempo di completamento di un job
 - Quanto ci vuole per eseguirlo tutto
 - Sistemi multiprogrammati:
 - tendenza a soddisfare molti utenti che operano interattivamente
 - importante il tempo di risposta di un job
 - Quanto ci vuole per iniziare l'esecuzione

Time sharing (o multitasking)

- Time sharing = condivisione del tempo
 - Estensione logica della multiprogrammazione
 - L'utente ha l'impressione di avere la macchina solo per sé
 - Migliora la interattività (gestione errori, analisi risultati, ...)
- Nascita dei sistemi “moderni”
 - Tastiera
 - Decisioni sull'evoluzione del sistema basate su comandi utente
 - Comandi brevi vs. job lunghi
 - Monitor
 - Output immediato durante l'esecuzione
 - File system
 - Astrazione del sistema operativo per accedere a dati e programmi

Protezione

- Nei sistemi originari non si poneva il problema della condivisione
- In generale, l'esecuzione di un programma può influenzare l'esecuzione degli altri
- Esempio:
 - Ciclo infinito
 - previene l'uso della CPU di altri programmi
 - oppure il programma legge più dati di quanti dovrebbe
 - ...
- Tre tipi fondamentali di protezione
 - I/O (programmi diversi non devono usare I/O contemporaneamente)
 - Memoria (un programma non può leggere/scrivere in una zona di memoria che non gli "appartiene")
 - CPU (prima o poi il controllo della CPU deve tornare al S.O.)

Protezione dell'I/O

- Realizzata tramite il meccanismo del modo duale di esecuzione (dual mode)
 - Modo USER
 - I job NON possono accedere direttamente alle risorse di I/O
 - Modo SUPERVISOR (o KERNEL)
 - Il sistema operativo può accedere alle risorse di I/O
- Tutte le operazioni di I/O sono privilegiate
 - Sequenza di operazioni (per accesso ad I/O)
 - Istruzioni per accesso ad I/O invocano delle system call
 - System call = interrupt software che cambia la modalità di esecuzione da USER a SUPERVISOR
 - Al termine della system call il S.O. ripristina la modalità USER

Protezione della memoria

- Protezione dello spazio dei vari processi
- Protezione del monitor
- Realizzata:
 - associando dei registri limite ad ogni processo
 - I registri limite possono essere modificati solo dal S.O con istruzioni privilegiate

Protezione della CPU

- Garanzia che il S.O. mantenga il controllo del sistema
- Realizzata tramite timer
 - Associato ad ogni job
 - Alla scadenza, il controllo ritorna al monitor

4a Generazione (1980-????)

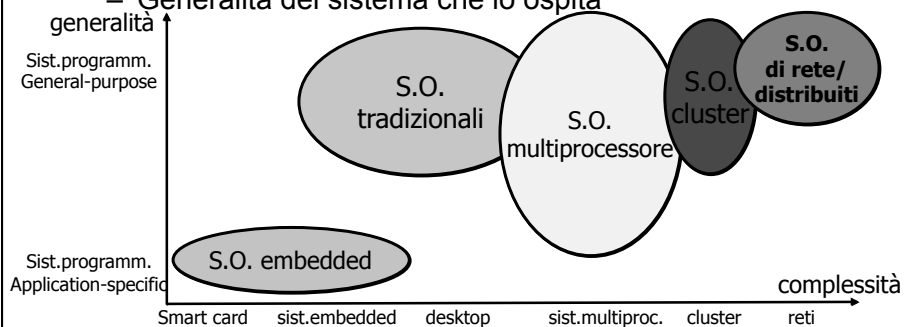
- S.O. per PC e workstation
 - Uso “personale” dell’elaboratore
- S.O. di rete
 - Separazione logica delle risorse remote
 - Accesso risorse remote ≠ accesso risorse locali
- S.O. distribuiti
 - Non-separazione logica delle risorse remote
 - Accesso risorse remote = accesso risorse locali
- S.O. real-time
 - Vincoli sui tempi di risposta del sistema
- S.O. embedded
 - Per sistemi per applicazioni specifiche

Riassunto

1 ^a generazione	<ul style="list-style-type: none">• Device driver• Librerie
2 ^a generazione	<ul style="list-style-type: none">• Batching• Automatic job sequencing• Off-line processing• Sovrapposizione CPU e I/O (buffering e spooling)
3 ^a generazione	<ul style="list-style-type: none">• Multiprogrammazione• Time sharing

Lo spazio dei S.O.

- Dimensioni
 - Complessità del sistema che lo ospita
 - Generalità del sistema che lo ospita



Real-time = dimensione ulteriore!

Graziano Pravadelli (2011)

33

Esempi di sistemi operativi

- OS/360
 - S.O. per sistemi batch multiprogrammati (IBM 360)
- CP/M
 - S.O. monoprogrammato, primo sistema per PC IBM
- MS-DOS
 - S.O. monoprogrammato, rimpiazzò CP/M
- UNIX
 - S.O. multiprogrammato, time-sharing, multiutente
- MacOS/Windows XX
 - S.O. multiprogrammati, basati su interfaccia grafiche (paradigma WIMP - Window Icon Mouse Pointer)

Graziano Pravadelli (2011)

34