


Appunti di sistemi operativi

Appunti per il corso universitario di sistemi operativi, riferito a sistemi Unix/Windows.
Si discute su problemi di sincronizzazione, memoria e scheduling dei processi.

Il nuovo One X™ Con Schermo HD da 4,7" HDMI, Fotocamera Istantanea e Beats Audio HTC.com/OfficialSite

Sistemi di Gestione Ohsas 18001 - ISO 14001 Modello Organizzativo 231 - OdV www.idramanagement.com/

Com'è il tuo Inglese? Fai ora il test online, è gratis! Con noi imparare inglese è facile. www.wallstreet.it 

ARGOMENTI

[INTRODUZIONE](#)

[INPUT/OUTPUT](#)

[GESTIONE DEI
PROCESSI](#)

[ALGORITMI DI
SCHEDULING](#)

[SCHEDULING
MULTICPU](#)

[SISTEMI REAL TIME](#)

[SCHEDULING SU
LINUX](#)

[SCHEDULING SU
WINDOWS](#)

[OPERAZIONI SUI
PROCESSI](#)

[COMUNICAZIONE
TRA PROCESSI](#)

[THREAD](#)

[SINCRONIZZAZIONE
TRA PROCESSI](#)

[GESTIONE
MEMORIA](#)

GESTIONE DEI PROCESSI

$P = (C, S)$

processo = **programma in esecuzione**; è formato da una sezione di testo (C) codice del programma (parte statica del programma che non varia nel tempo), e da uno stato di esecuzione (S) composto da:

- program counter
- valori dei registri della CPU
- stack con i dati relativi all'esecuzione

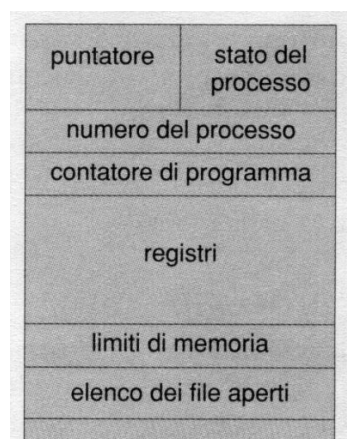
non va quindi confuso con il concetto di **programma semplice** (passivo, non in esecuzione, ossia l'insieme di **codice contenuto in un file nel supporto di massa**); diverse istanze di uno stesso programma sono da considerare infatti diversi processi

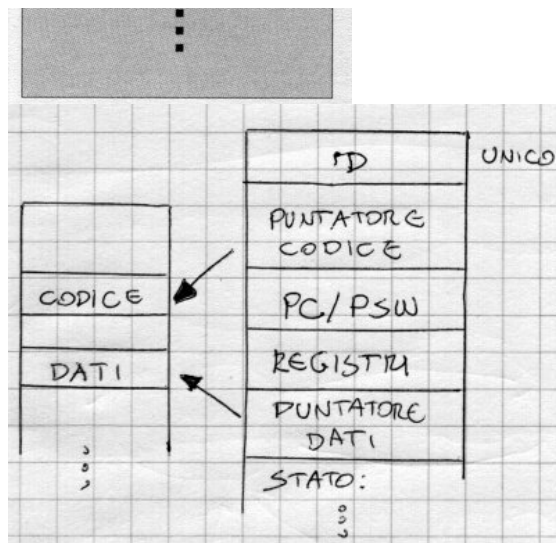
un processo è rappresentato nel sistema operativo da un **descrittore di processo** (PD = process descriptor, chiamato anche PCB = process control block), che contiene tutte le informazioni relative al processo stesso

il termine processo è ormai universale, ma per la precisione:

- lavoro (**job**) in caso di sistemi **batch**
- **processo** in caso di sistemi **time-sharing**

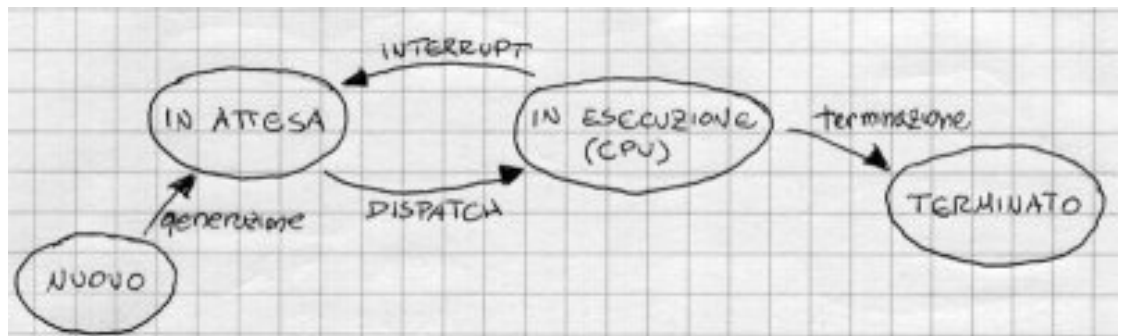
rappresentazione di un PCB:





STATO DI UN PROCESSO

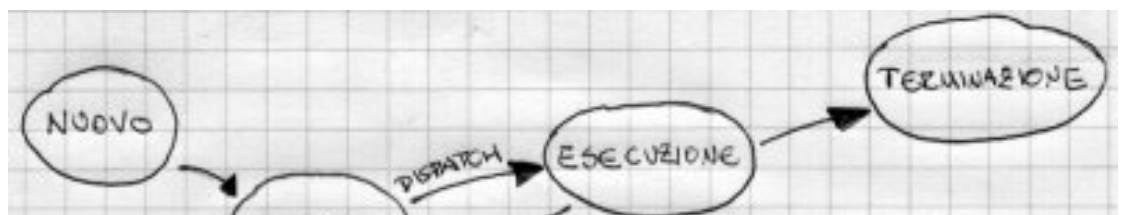
(dal punto di vista del sistema operativo)

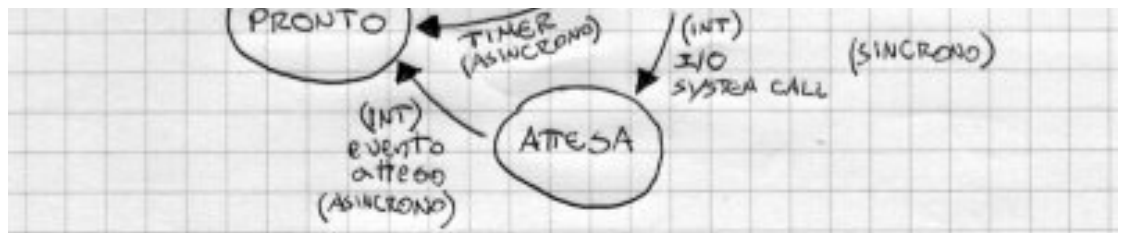


un processo passa dall'esecuzione all'attesa in presenza di un interrupt

abbiamo una sorta di ciclo in cui i processi passano da uno stato all'altro: **interrupt** - **dispatch** ("sbrigare")

schema classico a 5 stati:





abbiamo un interrupt che fa passare i processi dallo stato in attesa allo stato pronto; •

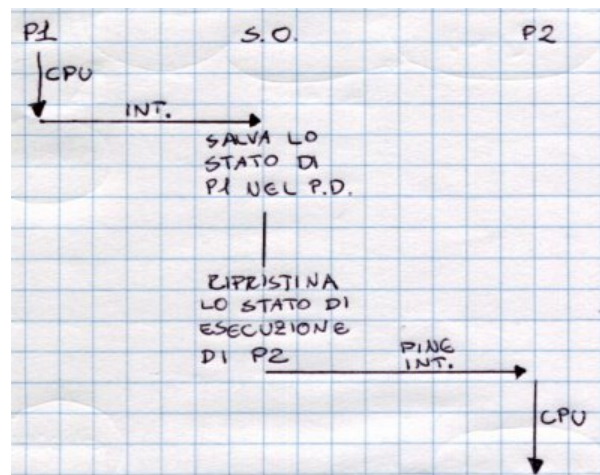
- un altro interrupt fa passare dallo stato di esecuzione allo stato di attesa (asincrono);
- nel time-sharing si aggiunge un altro interrupt (timer) che si occupa di alternare i vari processi nello stato di esecuzione periodicamente e indipendentemente dal fatto che siano stati terminati o meno (asincrono)

le code di processi nel sistema operativo consistono in code di PCB

CAMBIO DI CONTESTO

cambio di contesto = operazione tramite la quale il sistema operativo passa dall'esecuzione di un processo all'esecuzione di un altro processo; fasi:

1. il sistema si occupa di salvare al momento dell'interrupt lo stato di esecuzione di P1 nel PCB
2. ripristina lo stato di esecuzione di P2
- 3.



finisce l'interruzione e inizia l'esecuzione di P2

SCHEDULING

iter di un processo:

1. **scheduler a lungo termine** li sceglie e li carica in **memoria (nei sistemi batch)**
2. **scheduler di CPU** assegna la CPU a uno dei **processi pronti**

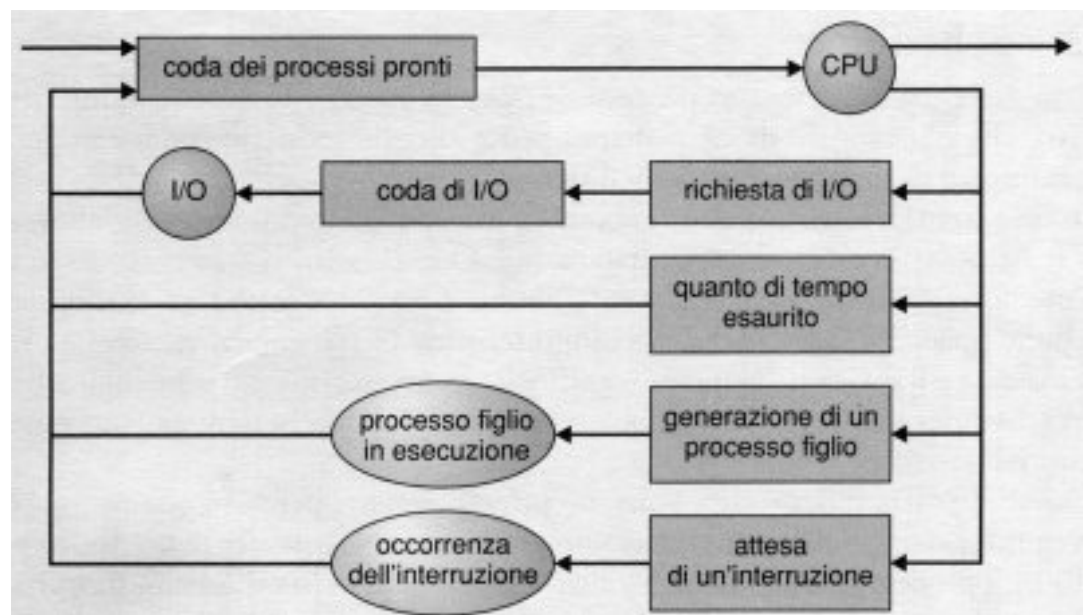
obiettivi della progettazione di uno scheduler (l'idea di base è che un solo processo è in esecuzione su una CPU, ma più processi la devono usare):

- **massimizzare** l'utilizzo della **CPU** (vero per quel che riguarda i sistemi batch)

- **massimizzare l'utilizzo della CPU** (vero per quei che riguarda i sistemi batch multiprogrammati)
- **garantire l'interattività** (time-sharing)

Ogni processo è inserito in una coda di scheduling (esistono molte code, es. tutti i processi passano attraverso la coda dei processi pronti; ogni dispositivo di I/O ha associata una coda di processi che hanno richiesto il suo utilizzo)

diagramma di accodamento:



scheduler a lungo termine = tradizionalmente legato ai sistemi batch; non molto presente in quelli attuali, dove è l'utente a decidere se aprire altri processi (facendo la funzione dello scheduler a lungo termine); si occupa di scegliere quali processi ammettere nel sistema, seguendo come criteri:

- **bilanciamento** fra numero di processi in entrata in memoria e numero di processi terminati (mantiene costante il **grado di multiprogrammazione**)
- **equilibrio** fra processi con prevalenza di I/O (**I/O bound**) e processi con prevalenza di elaborazione (**CPU bound**), per massimizzare le risorse

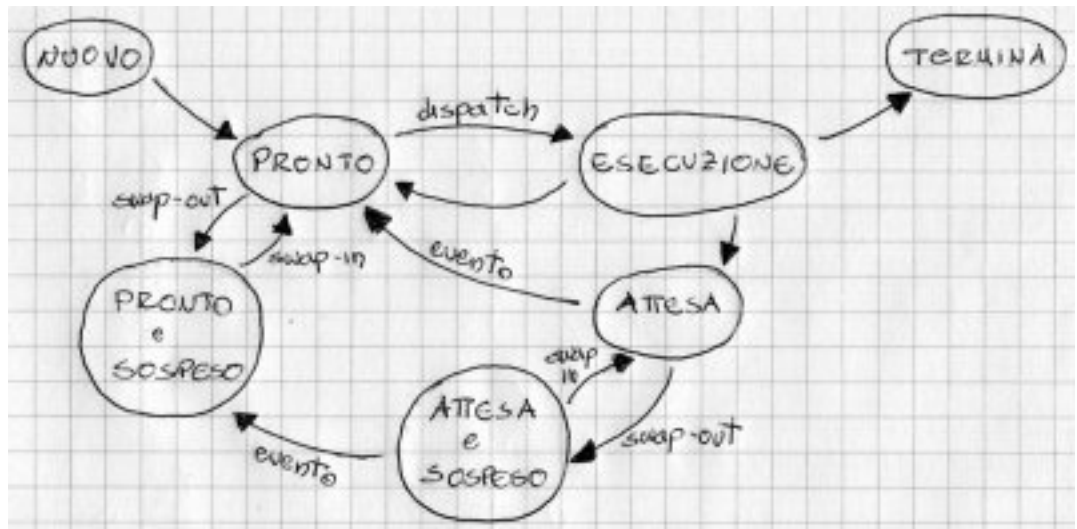
es. UNIX non ha lo scheduler a lungo termine -> sta all'utente diminuire i processi in esecuzione manualmente quando le prestazioni calano troppo

scheduler a medio termine = (presente in certi sistemi) nel momento in cui il grado di

multiprogrammazione è troppo alto, sospende temporaneamente dei processi pronti o ancora meglio in attesa, spostandoli su disco (swap-out), e in seguito li ripristina da disco alla coda dei processi pronti o in attesa (swap-in)

lo scheduler medio per effettuare le sue scelte si basa su questi dati:

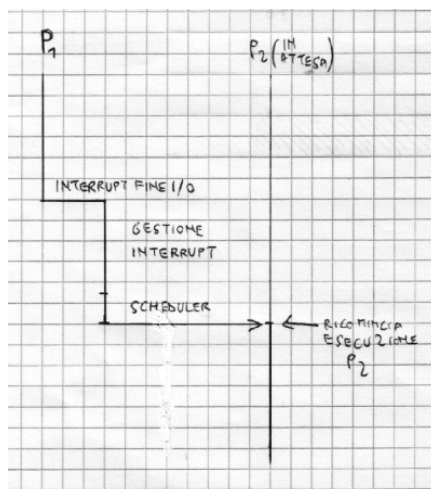
- **turnaround** = tempo di completamento di un processo
- **tempo di attesa** (nello stato di pronto), **tempo di interattività** (time-sharing)



QUANDO INTERVIENE LO SCHEDULING A BREVE TERMINE?

- **timer** (per i time-sharing, nei sistemi batch no)
- **terminazione** (è necessario che intervenga perché se un processo lascia libera la CPU è necessario eseguire un altro processo)
- **I/O, attesa**: se il processo in attesa si blocca per un I/O
- **fine I/O, fine attesa, evento**: possibile per processi real-time o comunque ad alta priorità

lo scheduler viene invocato alla fine della gestione dell'interruzione (e decide che processo eseguire)



P2 era in attesa di un I/O e aveva lasciato l'esecuzione a P1, quando avviene la fine dell'I/O vi è un interrupt e lo scheduler interrompe P1 e lascia l'esecuzione a P2

altre caratteristiche degli scheduler:

1. **scheduling non-preemptive** = non fanno prelazione sul processo in esecuzione = mettono in esecuzione un nuovo processo solo quando la CPU è già stata liberata
2. **scheduling preemptive** (con diritto di prelazione) = possono forzare il rilascio della CPU
3. **con priorità**
4. **senza priorità**

queste 4 si possono combinare

es. fila a uno sportello

fila classica, tutti in fila nessuno passa davanti = non preemptive senza priorità

fila classica + fila prioritaria per quelli che hanno un conto presso la posta = fila non-preemptive con priorità

[continua..](#)