I PROCESSI

A.S. 2022/2023

Prof. Verga – Prof.ssa Dalbesio

Introduzione

Le capacità computazionali sono migliorate grazie all'aumento della velocità della CPU, ma la gestione del processore deve essere ancora ottimizzata, perché presenta spesso delle criticità.

I moderni S.O. cercano di sfruttare al massimo le potenzialità di parallelismo fisico dell'hw per minimizzare i tempi di risposta ed aumentare il **throughput** (*numero di programmi eseguiti per unità di tempo*)

I processi

Il programma composto da un insieme di byte contenente le istruzioni che dovranno essere eseguite è un'***entità passiva*** e resta tale finché non viene caricato in memoria e mandato in esecuzione. In quel momento diventa un *processo* che evolve man mano che le istruzioni vengono eseguite dalla *CPU* e si trasforma in un'***entità attiva***.

I processi

Il processo pertanto è:

- Programma in esecuzione
- Individuato dal programma e dall'ambiente di esecuzione
- Costituito da stato e dati

I processi

Nel modello a processi tutto il software che può essere eseguito su un calcolatore, compreso il S.O., è organizzato in ***processi sequenziali***.

Un unico processore può essere condiviso tra molti processi usando un **algoritmo di schedulazione** (**scheduling**) per determinare quando interrompere l'evoluzione di un processo e servirne un altro.

I processi

Questa tecnica di gestione della CPU si chiama **multiprogrammazione**, la quale permette l'evoluzione contemporanea di più processi, limitando al minimo i tempi morti e sfruttando appieno le potenzialità di calcolo del processore.

I processi

Un processo impegna processore, memoria, connessioni di rete, dispositivi di I/O, file aperti, ecc..

Il **gestore dei processi** (**Scheduler**) ha il compito di scegliere quale processo deve essere eseguito, ovvero quale tra i processi attivi deve essere scelto per l'esecuzione

I processi

I processi possono essere ***indipendenti***, ***cooperanti***, oppure ***in competizione***.

Nel caso di processi indipendenti, i processi evolvono **autonomamente** e non c'è necessità di comunicazione con gli altri processi per scambiarsi dati.

In caso di processi cooperanti oppure in competizione, è necessaria una **sincronizzazione**.

I processi

Un processo si dice **cooperante** se influenza o può essere influenzato da altri processi in esecuzione nel sistema (il processo condivide dati con altri processi).

Con questi processi si può ottenere la **parallelizzazione** dell'esecuzione, la **replicazione** di un servizio, la **modularità**, la **condivisione** delle informazioni.

I processi

Un processo si dice in **competizione** quando può evolvere indipendentemente da un altro processo, ma entra in **conflitto** sulla ripartizione delle risorse.

Un esempio concreto è proprio lo **scheduling** dei processi, dove tutti competono per la CPU, oppure la condivisione della **stampante**.

Stato dei processi

Durante il ciclo di vita del processo è possibile individuare un insieme di situazioni in cui il processo può trovarsi, che vengono definiti ***stati di un processo***.

Un processo può assumere una sola volta lo stato di ***nuovo*** e di ***terminato***, mentre può assumere per più volte gli altri possibili stati.

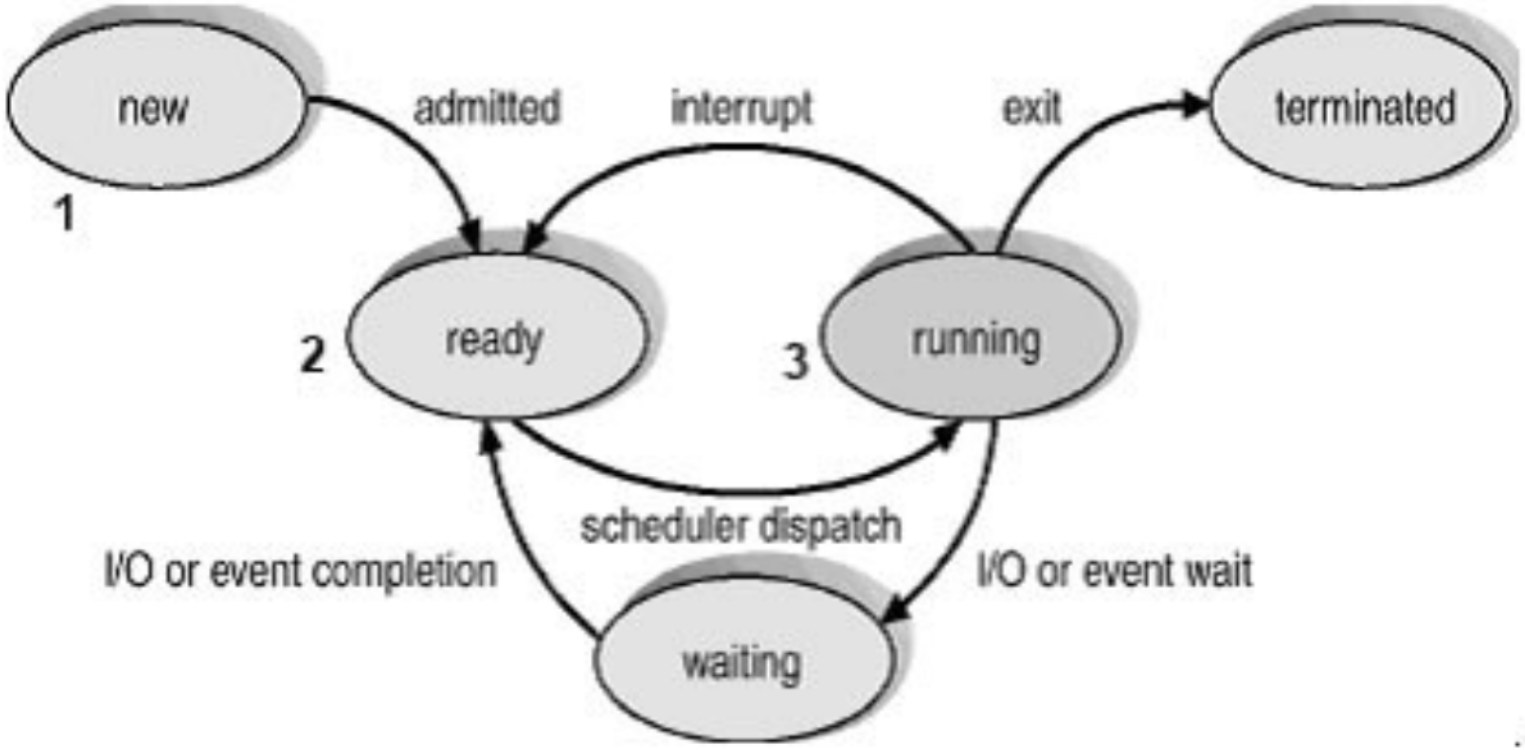
Stato dei processi

- **Nuovo (new)**: Il processo viene creato, cioè l'utente richiede l'esecuzione di un programma che risiede sul disco.
- **Esecuzione (running)**: Il processo è in esecuzione, quindi la CPU sta eseguendo le sue istruzioni.
- **Attesa (waiting)**: Il processo è in attesa di una risorsa, quindi è in attesa che si verifichi un dato evento (es. il rilascio della risorsa)

Stato dei processi

- **Pronto (ready)**: Il processo è pronto per essere eseguito. In questo stato il processo ha tutte le risorse necessarie alla sua evoluzione, tranne la CPU.
- **Finito (terminated)**: Il processo ha completato la sua esecuzione, tutto il suo codice è stato eseguito. Il Sistema Operativo può rilasciare le risorse che utilizzava.

Stato dei processi



Processi

Il processo è identificato da:

- ***PID***: identificativo processo - assegnato dal S.O. al processo al momento del lancio
- ***UID***: identificativo dell'utente che ha richiesto il processo
- Posizione in memoria e dimensione di memoria assegnata
- ***PPID***: identificativo del processo padre (quello che ha generato il processo corrente)
- eventuali id di processi figli

Come si crea un processo

Il comando utilizzato per creare processi è ***fork()***.

La fork() permette di creare in memoria una *copia del processo stesso*, generando un processo figlio

La fork ha dei ***valori di ritorno***

- se è uguale a 0, siamo nel processo **figlio**.
- se è maggiore di 0, siamo nel processo **padre**.
- se è minore di 0 si è verificato un errore.

Altri riferimenti

http://it.wikitolearn.org/Corso:Programmazione_C/Processi_e_Thread/Processi