

Corso di Laurea in Matematica

I Processi - Parte 2

Annalisa Massini

Dipartimento di Informatica
Sapienza Università di Roma

Argomenti trattati

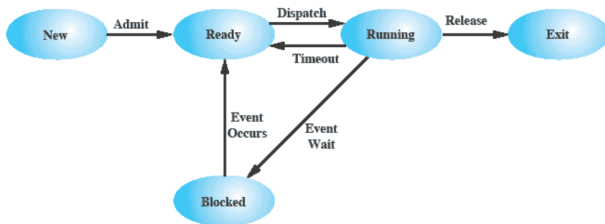
- 1 Stati di un processo
 - Processi sospesi
- 2 Descrizione del processo
 - Processi e risorse
 - Strutture di controllo del SO
 - Switching tra processi
- 3 Controllo del processo
 - Modi di esecuzione

Stati di un processo

Processi sospesi

Modello dei processi a 5 Stati

- Nel modello di processo a cinque stati si hanno:
 - Lo stato **new** per un processo appena creato
 - lo stato **ready** per un processo a cui il SO può assegnare il processore
 - lo stato **running** per il processo che sta usando il processore
 - lo stato **blocked** per i processi in attesa del completamento di un'operazione (es. di I/O)
 - lo stato **exit** per il processo che ha completato l'esecuzione



Processi Sospesi

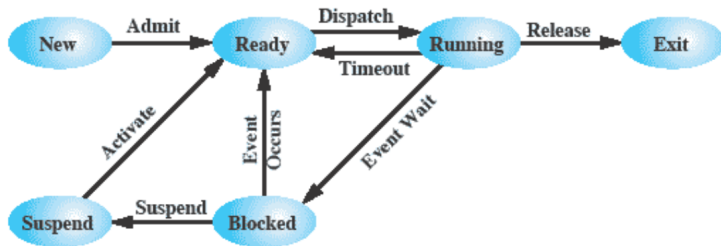
- Oltre agli stati *Ready*, *Running* e *Blocked* può essere utile usare altri stati
- Ogni processo deve essere in memoria principale (*non consideriamo per ora la memoria virtuale*)
- Il processore è più veloce dell'I/O, quindi tutti i processi attualmente in memoria potrebbero essere in attesa di I/O

Processi Sospesi

- Per non lasciare il processore inoperoso i processi vengono spostati su disco, **swap**, così da liberare memoria
- Quando il processo è *swappato* su disco lo stato da *Blocked* diventa **Suspend**
- Lo spazio liberato in memoria principale può essere usato per un altro processo:
 - un processo appena creato *oppure*
 - un processo precedentemente sospeso
 - *meglio un processo sospeso* per non sovraccaricare il sistema

Stato Suspended

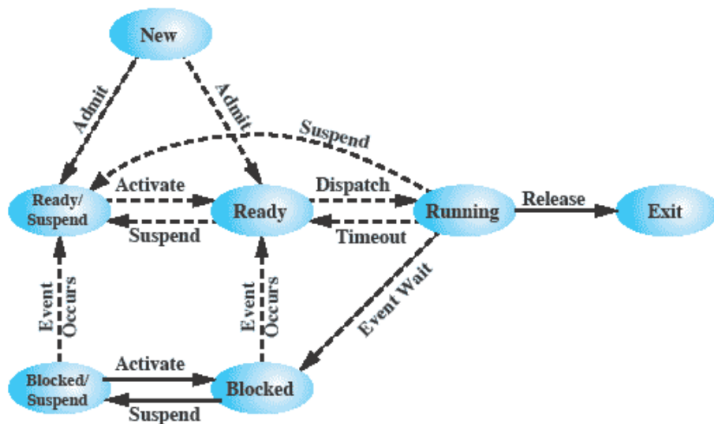
- Dallo stato *Blocked* si passa quindi allo stato **Suspend**
- E dallo stato **Suspend** si passa allo stato *Ready*



Processi Sospesi

- Non conviene riportare in memoria principale processi che non sono ancora pronti per l'esecuzione, cioè *bloccati*
- Ma se un processo era in attesa di un evento, quando l'evento si è verificato diventa pronto per l'esecuzione
- Due nuovi stati
 - **Blocked/Suspend** (swappato - in attesa dell'evento)
 - **Ready/Suspend** (swappato - pronto per l'esecuzione)

Due stati Suspend



- Si può andare ad Exit da un qualsiasi stato diverso da new
- Ad esempio se un processo ne *uccide* un altro

Motivi per Sospendere un Processo

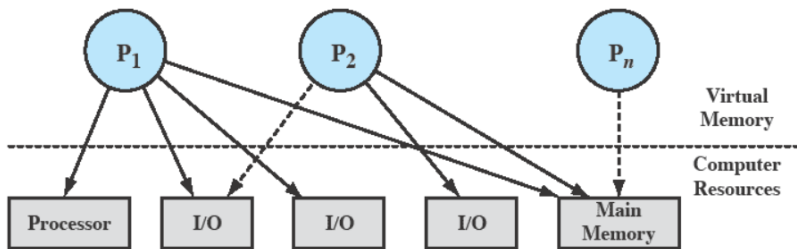
Motivo	Commento
Swapping	Il SO ha bisogno di liberare abbastanza memoria per caricare un processo ready
Interno al SO	Il SO sospetta che il processo stia causando problemi
Richiesta utente interattiva	Ad esempio: debugging o motivi legati all'uso di risorse
Periodicità	Il processo viene eseguito periodicamente (p.e. monitoraggio di sistema o accounting) e può venire sospeso in attesa della prossima esecuzione
Richiesta del padre	Il padre potrebbe sospendere l'esecuzione di un figlio per esaminarlo, modificarlo o per coordinare l'attività tra più figli

Descrizione del processo

Processi e risorse

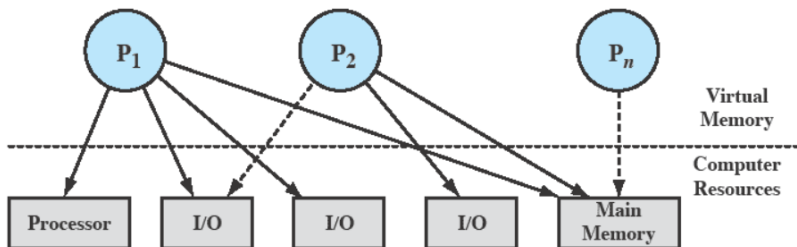
Processi e Risorse

- Il compito del sistema operativo è fondamentalmente la *gestione dell'uso delle risorse di sistema da parte dei processi*, processore *in primis*
- In un sistema multiprogrammato si ha un insieme di processi che competono per l'utilizzo delle risorse comuni



Processi e Risorse

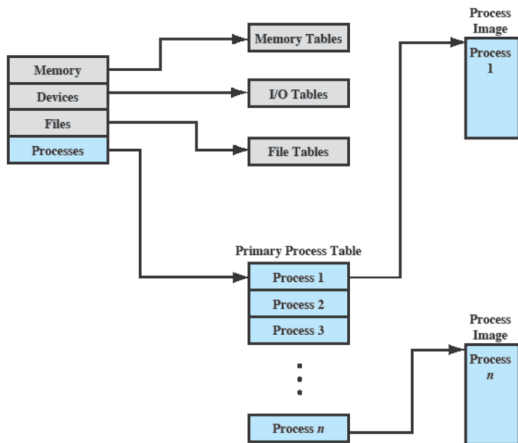
- P_1 è running, quindi almeno in parte è in memoria principale e usa processore e dispositivi di I/O
- P_2 è in attesa dell'I/O utilizzato da P_1
- P_n è stato swappato ed è sospeso



Strutture di Controllo del SO

- Per gestire sia i processi che le risorse, il SO deve conoscere lo stato di ogni processo e di ogni risorsa
- Il SO costruisce e mantiene una o più tabelle per ognuna delle entità da gestire
- I quattro tipi di tabelle mantenute dal SO sono:
 - memoria
 - I/O
 - file
 - processi

Tabelle di controllo del SO



- **Ci sono molti riferimenti incrociati:** memoria, I/O, file vengono gestiti dai processi che devono quindi avere nella propria tabella le referenze a queste risorse

Tabelle di Memoria

- Le **tabelle di memoria** sono usate per gestire sia la memoria principale che quella secondaria
 - quella secondaria serve per la memoria virtuale (lo vedremo)
- Le **tabelle di memoria** devono comprendere le seguenti informazioni:
 - allocazione di memoria principale da parte dei processi
 - allocazione di memoria secondaria da parte dei processi
 - attributi di protezione per l'accesso a zone di memoria condivisa
 - informazioni per gestire la memoria virtuale

Tabelle per l'I/O

- Le **tabelle per l'I/O** sono usate dal SO per gestire i dispositivi e i canali di I/O
- Il SO deve sapere:
 - se il dispositivo è disponibile o già assegnato
 - lo stato dell'operazione di I/O
 - la locazione in memoria principale usata come sorgente o destinazione del trasferimento di I/O

Tabelle dei File

- Le **tabelle dei file** forniscono informazioni su:
 - esistenza di files
 - locazioni in memoria secondaria
 - stato corrente
 - altri attributi
- Sono memorizzate parte su disco e parte in RAM

Tabelle dei Processi

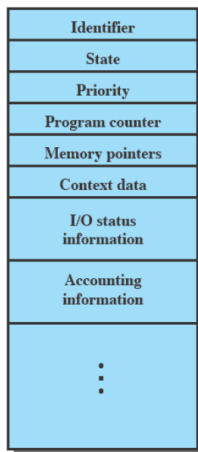
- Abbiamo già visto che ad ogni processo è associata la tabella **Process Control Block**
- PCB Un processo è composto (almeno) da:
 - un **programma** (o un insieme di programmi) da eseguire
 - un insieme di **locazioni di memoria** per le variabili locali e globali e per le costanti istanziate
 - una o più **stack** (zona di memoria ad accesso LIFO) per tenere traccia dei parametri e delle chiamate a procedura e a sistema che sono richieste durante l'esecuzione del programma
 - un insieme di **attributi** usati dal sistema operativo per il controllo del processo

Tabelle dei Processi

- L'insieme di programma sorgente, dati, stack delle chiamate e dei dati necessari al SO per il controllo del processo è detto **process image** (immagine del processo)
- Eseguire un'istruzione cambia l'immagine del processo: per esempio, modificando un registro o una cella di memoria
- Per gestire un processo il SO deve conoscere tutti i dettagli relativi all'esecuzione del programma che origina il processo che sono appunto raccolti nella tabella denominata (di solito) **Process Control Block**

Attributi dei Processi

- Le informazioni in ciascun **Process Control Block (PCB)** possono essere raggruppate in 3 categorie:
 - identificazione
 - stato
 - controllo
- Ogni sistema può organizzare queste informazioni in modo diverso



Come si Identifica un Processo

- Ad ogni processo è assegnato un numero identificativo, che quindi è unico: il **PID** (Process IDentifier)
- Molte tabelle del SO usano i PID per realizzare collegamenti tra le varie tabelle e la tabella dei processi
 - ad esempio, la tabella dei dispositivi I/O deve mantenere, per ogni dispositivo, quale processo lo sta usando
 - basta mettere il PID e implicitamente si può accedere alle informazioni sul processo corrispondente
- Nel PCB è specificato lo **stato del processo**, come *ready*, *blocked*, ecc.

Lo stato del processore

- Lo stato del processore è dato dal contenuto dei registri del processore stesso:
 - registri visibili all'utente
 - registri di controllo e di stato
 - puntatori allo stack
- Tutti i processori includono il registro per la parola dello stato di programma (*Program status word*, **PSW**)
 - contiene informazioni di stato come: riporto, overflow, segno, zero, interruzioni, priorità, ecc.

Control Block del Processo

- Quindi, il **PCB** contiene informazioni di cui il SO ha bisogno per controllare e coordinare i vari processi attivi
- Identificatori:
 - del processo (PID)
 - del processo padre (Parent PID, o PPID)
 - dell'utente proprietario
- Informazioni sullo stato del processore:
 - registri utente (accessibili in linguaggio macchina/assembly)
 - program counter
 - stack pointer
 - registri di stato: risultati di operazioni aritmetico/logiche, modalità di esecuzione, interrupt abilitati/disabilitati

Control Block del Processo

- Informazioni per il controllo del processo:
 - stato del processo (ready, suspended, blocked, ...)
 - priorità
 - informazioni sullo scheduling (ad es.: per quanto tempo è stato in esecuzione l'ultima volta)
 - l'evento da attendere per tornare ad essere ready, se attualmente in attesa
- Supporto per strutture dati
 - puntatori ad altri processi
 - per mantenere liste concatenate di processi nei casi in cui siano necessarie (es., code di processi per qualche risorsa)

Control Block del Processo

- Comunicazioni tra processi
 - flag, segnali, messaggi per supportare comunicazioni tra processi
- Permessi speciali
 - non tutti i processi possono accedere a tutto
- Gestione della memoria
 - puntatori ad aree di memoria che gestiscono l'uso della memoria virtuale
 - es: pagine virtuali attualmente in uso
- Uso delle risorse
 - file aperti
 - uso di risorse (compreso il processore) fino ad ora

Control Block del Processo

In conclusione il **PCB**:

- È la struttura dati più importante in un sistema operativo
 - definisce lo stato del SO stesso
- È una struttura che richiede protezioni
 - una funzione scritta male potrebbe danneggiare il blocco, rendendo il SO incapace di gestire i processi
 - ogni cambiamento nella progettazione del blocco ha effetti su molti moduli del SO

Controllo del processo

Switching tra processi

Creazione di un processo

- Prima di parlare di **process switching** riconsideriamo la creazione di un processo
- Per creare un processo, il SO deve:
 - Assegnargli un PID unico
 - Allocargli spazio in memoria principale
 - Inizializzare il Process Control Block
 - Inserire il processo nella giusta coda
 - ad es., ready oppure ready/suspended
 - Creare o espandere altre strutture dati
 - ad esempio, quelle per l'accounting (per richiedere pagamenti di accesso al sistema)

Process Switching

- Un *process switch* può avvenire in un qualunque momento
- Quando avviene un *process switch*
 - il SO interrompe l'esecuzione di un processo, cioè un *running process*
 - assegna l'uso del processore ad un altro processo che assume lo stato di running
- Un *process switch* si verifica quando per qualche motivo l'attuale processo non deve più usare il processore, che va concesso invece ad un altro processo
- Il **process switch** pone vari problemi
 - Quali eventi possono determinare un process switch?
 - Cosa deve fare il SO per tenere aggiornate tutte le strutture dati in seguito ad uno switch tra processi?

Quando effettuare uno switch

Un *process switch* avviene, per motivi diversi, quando il SO toglie il controllo al processo in esecuzione e se lo riprende

Meccanismo	Causa	Uso
Interruzione	Esterna all'esecuzione dell'istruzione corrente	Reazione ad un evento esterno asincrono (include <i>timeout</i> per lo scheduler)
Eccezione	Associata all'esecuzione dell'istruzione corrente	Gestione di un errore sincrono
Chiamata al SO	Richiesta esplicita	Chiamata a funzione di sistema (caso particolare di eccezione)

Process Switch: Passaggi

- 1 Salvare il contesto del programma (registri e PC)
- 2 Aggiornare il process control block del processo attualmente running ad un altro stato
- 3 Spostare il process control block nella coda appropriata: ready; blocked; ready/suspend
- 4 Scegliere un altro processo da eseguire
- 5 Aggiornare il process control block del processo selezionato mettendo lo stato a running
- 6 Aggiornare le strutture dati per la gestione della memoria
- 7 Ripristinare il contesto del processo selezionato

Avviene tutto in kernel mode

Modalità di Esecuzione

La maggior parte dei processori supporta almeno due modalità di esecuzione

- **Modo sistema - System mode**

- modalità dal SO (o dal kernel)
- il software ha pieno controllo del processore e di tutte le istruzioni, dei registri e della memoria.
- ad esempio
 - si possono eseguire istruzioni che bloccano gli interrupt
 - si può accedere a qualsiasi locazione di RAM

- **Modo utente - User mode**

- modalità usata dai programmi utente
- molte operazioni sono vietate
- per proteggere SO e tabelle
- *Linux* usa modalità *ristretta*, *modalità utente*, e modalità *senza limitazioni*, *modalità sistema o kernel*
- *Pentium* ha 4 modalità

Kernel Mode

- Il **kernel mode** è per le operazioni effettuate dal kernel:
 - Gestione dei processi (tramite PCB)
 - creazione e terminazione
 - pianificazione di lungo, medio e breve termine (*scheduling e dispatching*)
 - avvicendamento di processi (*process switching*)
 - sincronizzazione e comunicazione
 - Gestione della memoria principale
 - allocazione di spazio per i processi
 - gestione della memoria virtuale
 - Gestione dell'I/O
 - gestione dei buffer e delle cache per l'I/O
 - assegnazione risorse I/O ai processi
 - Funzioni di supporto
 - Gestione interrupt/eccezioni, accounting, monitoraggio