**Cos’è un processo?**

Un processo è un programma in esecuzione. Un processo non è solo il codice di un programma è qualcosa di più, esso comprende l’attività corrente, rappresentata dal valore del program counter e dal contenuto dei registri della CPU. Un processo normalmente comprende anche uno stack (contente a sua volta i dati temporanei come i parametri di un metodo, gli indirizzi di rientro e le variabili locali) e una sezione di dati contenete le variabili globali. Un processo può includere un heap, ossia della memoria allocata dinamicamente.

Un programma è un’entità passiva come il contenuto di un file memorizzato su un disco mentre un processo è un’entità attiva.

Un programma diventa un processo quando il suo file eseguibile viene caricato in memoria.

Un processo durante la sua esecuzione può trovarsi in diversi stati. Gli stati che può assumere un processo sono:

* Nuovo: Creazione del processo
* Esecuzione: Un’unità di elaborazione esegue le istruzioni del relativo programma
* Attesa: Il processo attende che si verifichi qualche evento (es. completamento di una richiesta di I/O)
* Pronto: Il processo attende di essere eseguito da un’unità d’elaborazione
* Terminato: Il processo ha terminato l’esecuzione.

In ciascuna unità di elaborazione può essere presente un solo processo alla volta.

Ogni processo è rappresentato all’interno del sistema operativo attraverso il blocco di controllo del processo (PCB).

Il blocco di controllo di un processo contiene molte informazioni connesse al processo le quali sono:

* Stato del processo
* Program counter (contiene la successiva istruzione che si deve eseguire per tale processo)
* Registri di CPU (Quando si verifica un’interruzione della CPU si devono salvare tutti i contenuti dei registri e il program counter cosicché da permettere una corretta esecuzione del processo in un momento successivo)
* Informazioni sullo scheduling della CPU

Informazioni che comprendono la priorità del processo

* Informazioni sulla gestione della memoria

Informazioni relative alle aree assegnate al processo

* Informazioni sulla contabilizzazione delle risorse

Informazioni che comprendono il tempo d’uso della CPU

* Informazioni sullo stato dell’I/O

Queste informazioni comprendono la lista dei dispositivi di I/O assegnati ad un determinato processo.

**Cos’è un sotto processo?**

Un sotto processo è un nuovo processo generato da un altro processo, il processo creante viene detto processo genitore mentre il nuovo processo viene detto processo figlio.

Quando un processo crea un sotto processo quest’ultimo per eseguire il proprio compito può ottenere le risorse direttamente dal sistema operativo o può essere vincolato ad un sottoinsieme delle risorse del processo genitore. Un processo genitore può avere la necessità di spartire le proprie risorse con i suoi processi figli oppure può condividerne solo alcune. Limitando le risorse dei processi figli a un sottoinsieme di risorse del processo genitore si può evitare che un processo sovraccarichi il sistema generando troppi sotto processi.

Quando un processo ne crea un nuovo per quanto riguarda l’esecuzione si hanno due possibilità:

* Il processo genitore continua la sua esecuzione in modo concorrente con i propri processi figli
* Il processo genitore attende che alcuni o tutti i suoi processi figli siamo terminati

Si hanno due possibilità anche per quanto riguarda lo spazio degli indirizzi:

* Il processo figlio è un duplicato del processo genitore
* Nel processo figlio si carica un nuovo programma

In alcuni sistemi quando un processo genitore termina terminano anche i processi figli ottenendo così un effetto a cascata mentre in altri sistemi se un processo genitore termina i processi figli vengono affidati al processo init.

**Cosa fa una syscall fork()?**

La syscall fork() è una chiamate di sistema che permette di creare un nuovo processo.

**Scheduler di un S.O batch?**

In un sistema a batch accade che si sottopongono più processi di quanti se ne possano eseguire immediatamente. Questi lavori si trasferiscono in dispositivi di memoria secondari e si tengono fino al momento dell’esecuzione (spooling).

In un sistema batch abbiamo uno scheduler a lungo termine e a breve termine.

Lo scheduler a lungo termine preleva i lavori dalla memoria secondaria e le carica in memoria affinché vengano eseguiti e controlla il grado di multiprogrammazione, cioè il numero di processi presenti in memoria.

Lo scheduler a breve termine fa selezione tra i lavori presenti nella coda dei processi pronti e ne assegna la CPU ad un di loro

Infine è importante che lo scheduler a lungo termina faccia un’accurata selezione dei processi.

I processi si possono caratterizzare per prevalenza d’elaborazione (impiega la maggior parte del tempo in operazioni d’elaborazione) e con prevalenza di I/O (impiega la maggior parte del tempo nell’esecuzione di operazioni di I/O).

È fondamentale che lo scheduler a lungo termine selezioni una buona combinazione di processi con prevalenza di I/O e con prevalenza de’elaborazione.

**Scheduler Preemptive e non preemptive (Capitolo 5 SO)?**

Scheduling senza diritto di prelazione, quando si assegna la CPU ad un processo questo ne rimane in possesso fin quando non termina o passa nello stato d’attesa.

Scheduling con diritto di prelazione, un processo non rimane in possesso della CPU fin quando non termina o passa nello stato d’attesa, ma la sua esecuzione può essere interrotta in favore di un altro processo.

Ad un processo può essere revocata la CPU. Il SO può in base ad alcuni criteri, sottrare ad esso la CPU per assegnarla ad un nuovo processo.

**Aggiunte di Johnny**

Scheduler dei processi selezione un processo da eseguire dall’insieme di quelli disponibili.

Dispatcher passa effettivamente il controlla della CPU al processo selezionato dallo scheduler.

Il successo dello scheduling della CPU dipende dal fatto che l’esecuzione di un processo consiste in un ciclo d’elaborazione e d’attesa del completamento delle operazioni di I/O.

Un processo termina quando finisce l’esecuzione della sua ultima istruzione e inoltra la richiesta al sistema operativo di essere cancellato usando la chiamata al sistema exit() a questo punto un processo figlio può riportare alcuni dati al processo genitore che li riceve attraverso una chiamata al sistema wait();

Processo Zombie è un processo che è terminato ma non è stato ancora espulso perché deve restituire il proprio stato a qualcuno;

Normalmente se lo prenderebbe il processo padre con una wait ma se il processo padre è terminato la pulizia degli zombie (Che si chiama reaping) viene fatta dal processo init

Spooling i lavori si trasferiscono in dispositivi di memoria secondaria dove si tengono fino al momento dell’esecuzione.