Partie 5

COURS

5.1 Définitions

5.1.1 Programme et processus

L'execution d'un **programme** entraîne des lectures et écritures dans des registres et une partie de la mémoire. Le processeur réalise lui des opérations sur les données mises dans les registres.

En réalité, derrière ce mot "programme" se trouve deux notions :

- Un programme binaire (ou langage machine) :

Il s'agit d'une suite de bits directement compréhensible par le processeur. Exemple d'un ensemble de bits permettant de transférer le contenu d'une zone-mémoire dans l'un des registres de processeur : En base 02 : 0010 0010 0011 1001 0000 0000 0000 0000 0000 0000 0010 0000. On peut le traduire si on sait qu'il s'agit du jeu d'instructions 68000 de Motorola, on peut lire dans la notice que 001 en début d'instruction signifie qu'on veut déplacer des données en mémoire (dont on donne l'adresse) vers un registre. La traduction mot à mot en utilisant quelques mots clés plutôt que les valeurs réelles des bits se nomme le langage assembleur.

- Un programme source (ou code source) : *Il s'agit des instructions fournies dans un langage de programmation. Ce code-source est compréhensible par un humain. Ce code-source n'est pas compréhensible par le processeur d'un système informatique et doit être traduit en langage machine. (langages compilés/interpretés).*

(https ://www.levavasseur.xyz/NSI_T/Archi/Archi_Processus.html)

Un processus représente une instance d'exécution d'un programme dans une machine donnée. Un processus a besoin de ressources de la machine pour son execution : accès aux registres, à la RAM, au DD, à un peripherique..

Au lancement d'un processus, un numero lui est attribué par l'OS : le PID. En console, avec l'instruction `STAT`, on a la liste des processus avec leur PID mais aussi leur PPID (numero de processus parent).

5.1.2 Processeur (rappel à faire avec les SoC)

Le processeur est composé principalement des parties suivantes :

L'UDC (Unité de Commande) permettant de gérer X bits à la fois : la partie de la puce qui lit la prochaine tâche élémentaire à effectuer en mémoire et qui commande l'UAL pour effectuer cette tâche élémentaire.

L'UAL (Unité Arithmétique et Logique) permettant de traiter X bits à la fois : la partie de la puce qui effectue les calculs, les déplacements en mémoire…

Quelques registres permettent de stocker et lire très rapidement en mémoire.

Des bus permettant de transporter X bits à la fois

On nomme cela UCT (Unité Centrale de Traitement en français) ou CPU (Central Processing Unit en anglais).

5.1.3 Le systeme d'exploitation

C'est un ensemble de programmes qui dirige l'utilisation des ressources d'un ordinateur par des logiciels. Un OS est multitâche : Les systèmes d'exploitation sont capables de gérer l'execution de plusieurs processus en même temps. En réalité, ce n'est pas tout à fait *en même temps* : pour gérer ce *chacun son tour*, les **systèmes d'exploitation** attribuent des *états* au processus.

5.2 Les états d'un processus

Ces états sont résumés ci-dessous.

- Lorsqu'un processus est créé, il démarre dans l'état *prêt* : il attend de pouvoir avoir accès au processeur. (mis dans une FILE)
- Le processus obtient, grâce au systeme d'exploitation, l'accès au processeur. Il passe alors dans l'état *élu*. (élection)
- Alors qu'il est *élu*, le processus peut avoir besoin d'attendre une ressource quelconque comme, par exemple, une ressource en mémoire ou sur le disque dur. Il doit alors quitter momentanément le processeur pour que celui-ci puisse être utilisé à d'autres tâches (le processeur ne doit pas attendre !). Le processus passe donc de l'état élu à l'état *bloqué*. (c'est un *blocage*)
- Lorsque le processus a obtenu la ressource attendue mais s'est fait prendre sa place dans le processeur par un autre processus, il se met en attente : C'est l'état *prêt*, en attente que la *place se libère*. Cette étape, de bloqué à prêt est l'opération de *déblocage*.
- Le passage de l'état *prêt* vers l'état *élu* constitue l'opération *d'élection*.
- Un processus ne pourra *terminer* que s'il est déjà dans l'état *élu*.

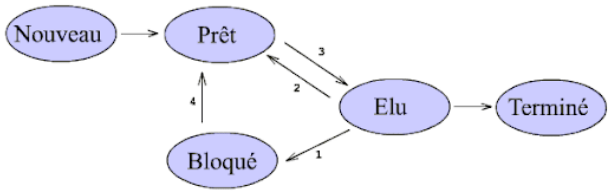

FIGURE 3 – mecanisme d'election

Un processus peut créer un ou plusieurs nouveaux processus. Il y a alors une filiation père-fils.

Chaque processus possède un numéro PID qui lui est attribué automatiquement. Il possède aussi un PPID qui est le numéro d'identification du processus père.

5.3 Ordonnancement

Plusieurs processus peuvent être dans l'état prêt : comment choisir celui qui sera élu ?

L'ordonnanceur (sheduler) classe les processus prêts dans une file et les fait passer du statut prêt à élu. Il planifie l'exécution des processus.

Dans les systèmes d'exploitation, l'ordonnanceur désigne le composant du noyau du *système d'exploitation* choisissant l'ordre d'exécution des processus sur les processeurs d'un ordinateur. Cette organisation permet d'occuper au mieux le (ou les) processeurs.

Il existe plusieurs politiques d'ordonnancement dont le choix va dépendre des objectifs du système. Voici quelques exemples :

• Premier arrivé, premier servi : (utilise une FILE). simple, mais pas adapté à toutes les situations.
• Plus court d'abord : très efficace, mais il est la plupart du temps impossible de connaître à l'avance le temps d'exécution d'un processus.
• Priorité : le système alloue un niveau de priorité aux processus (SCHED_FIFO sur Linux). Cependant des processus de faible priorité peuvent ne jamais être élus.
• Tourniquet : un quantum de temps est alloué à chaque processus (SCHED_RR sous Linux). Si le processus n'est pas terminé au bout de ce temps, il est mis en bout de file en état prêt.
• Un système hybride entre tourniquet et priorité qu'on retrouve dans les systèmes Unix.

Partie 6

Interblocage

6.1 Concurents mal synchronisés

L'ordonnanceur fait passer les processus de *élu* à *commuté*. Mais Il peut y avoir une situation où 2 processus sont **interbloqués** car :

- P1 possède la ressource R1 mais souhaite R2 : commutation pour P2
- P2 possède la ressource R2 mais souhaite R1 : commutation pour P1

Dans cette situation, les deux processus légers sont définitivement bloqués.(wiki)

Des solutions de détection/guérison peuvent être mises en place.

6.2 Définition sur Wikipedia

Deux processus en concurrence pour deux ressources dans un ordre opposé. Voici une chronologie possible qui mène à un interblocage.

A) Un seul processus se déroule.

B) Le processus ultérieur doit attendre.

C) Un blocage se produit lorsque le premier processus verrouille la première ressource en même temps que le second processus verrouille la seconde ressource.

D) Le blocage peut être résolu en annulant et en redémarrant le premier processus.

Exemple : le processus P1 utilise la ressource R2 qui est attendue par le processus P2 qui utilise la ressource R1, attendue par P1.

6.3 Modélisation par un graphe orienté

Il y a interblocage lorsque des cycles sont présents dans le graphe réalisé de la manière suivante :

- un arc de la ressource Ri au processus Pj signifie que le processus Pj a obtenu la ressource
- un arc Pj vers Ri signifie que le processus Pj demande la ressource Ri.

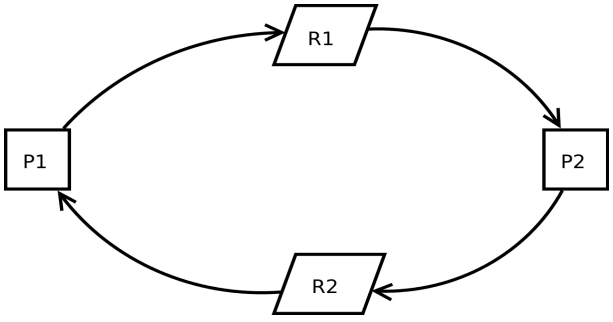

Figure 4 – interblocage - wikipedia

Partie 7

Informations sur les processus en console : STAT

La commande `ps` permet d'obtenir la liste des processus actifs dans un terminal Linux (ici en utilisant la "simulation" de terminal, l'application graphique bash en réalité).

- Le TTY : le terminal contrôlant le processus
- STAT : L'état des processus (STAT, comme Status) est ce qui nous intéresse ici :

Les status possibles

- R (Running et Runnable) : en cours d'exécution. Nous verrons que cela correspond aux états PRET (Runnable) ou ELU (Running) de la partie 2.
- S (Sleeping) : endormi. Cela correspond à l'état BLOQUE de la partie 2.
- D (Device) : en attente d'une ressource (généralement d'entrée/sortie) (le processus ne peut pas être interrompu). Cela correspond à l'état BLOQUE de la partie 2.

Les trois états terminaux FINI :

- T (sTopped) : terminé et va transmettre sa réponse à son parent. On libère une partie de la mémoire mais on garde encore des informations sur son état final.
- Z (Zombie) : processus terminé ayant répondu mais dont le parent n'a pas encore eu le temps de totalement finir la destruction.
- X (Dead) : processus terminé et détruit (vous ne devriez jamais voir de X dans votre liste). On peut trouver une deuxième lettre derrière l'état : il s'agit de la priorité du processus :
- `<` : Priorité haute
- `+` : Processus au premier plan
- `s` : Leader de session
- `l` : multi-theads
- `N` : Priorité basse
- `L` : ressources verrouillées en mémoire