Université Assane Seck de Ziguinchor

Systèmes d'Exploitation

GESTION DES PROCESSUS

I- NOTION DE PROCESSUS

Toute l'activité d'un ordinateur s'organise autour de l'exécution des instructions des programmes de celui- ci. Ces instructions exécutées peuvent venir des taches lancées par l'utilisateur comme par exemple un click de souris, une création de dossier, le démarrage d'un logiciel d'application, l'exécution d'un programme écrit en langage C etc. Le terme processus fait référence a un jeu d'instruction machine encours d'exécution. Une instruction en tant que telle est une entité statique une suite de lettres ou de mots qu'un ordinateur peut exécuter. Elle peut exister sur papier, dans un fichier, ou même en tant que concept dans l'esprit d'un individu. Un processus correspond à ce jeu d'instruction qui prend vie, c'est une entité dynamique qui réalise des actions de façon dont l'indique le code. L'objectif de la gestion des processus est de gérer l'allocation du CPU aux différents programmes. `

Remarque:

Un processus est différent d'un programme exécutable qui ne devient processus que quand son exécution est lancée**.** Un processus correspond à une suite d'actions (notion dynamique) réalisées lors d'une exécution. Alors qu'un programme est une suite d'instructions (notion statique). **Le même programme peut s'exécuter plusieurs fois donnant ainsi un processus différent pour chacune de ces exécutions.**

On distingue deux types de processus **:**

Les processus utilisateur :

Les processus système :

II- DU PROGRAMME AU PROCESSUS

Une chaîne de production d'un programme permet de passer d'un programme écrit dans un langage de haut évolué (de haut niveau) en un programme exécutable, écrit en langage machine et qui devient un processus des que son chargement en mémoire est déclenché. Cette évolution du programme passe par les étapes suivantes.

- La saisie du programme dans un éditeur de texte comme BlocNote pour produire un code appelé **code source**. Ceci génère un fichier stocké sur un support de comme le disque dur.
- Le fichier source est ensuite compilé à l'aide d'un compilateur. Ce compilateur est lié au langage de programmation utilisé et dont le role est de générer l'équivalent du programme source en langage machine absolu après avoir vérifié si la syntaxe du langage est respectée. On obtient aisnsi un fichier appelé **fichier objet** sur le disque.
- Ce fichier objet est ensuite soumis à un programme appelé **éditeur de liens** et dont le rôle est de résoudre les références externs (les appels de fonctions se trouvant dans des bibliothèques par exemple). L'éditeur de liens produit sur disque un **fichier exécutable** qu'on peut appeler fichier binaire pur.
- Lorsque lance l'exécution de son programme, le fichier exécutable est alors chargé en mémoire central, le SE alloue de la place mémoire pour placer le code et les données du programme: ainsi un processus est crée (voir la **Figure1**).

Du programme au processus

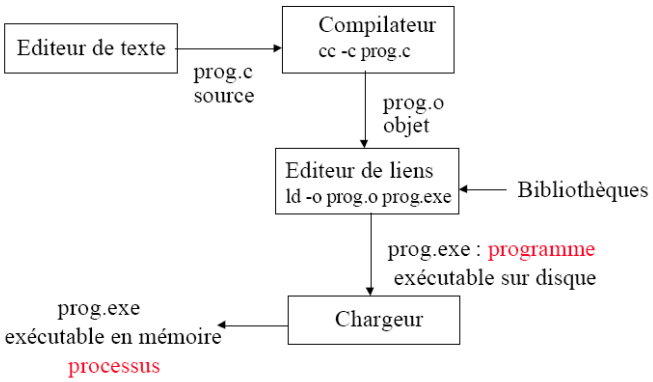

Figure1 : Production de programme

II-LES ETATS D'UN PROCESSUS

L'exécution d'un processus passe par différents états. Elle progresse séquentiellement c'est à dire en un instant donné, une et une seule instruction au plus est exécutée au nom du processus. Il faut noter également l'exécution d'une instruction est composée de différentes phases. Ainsi au fur et à mesure que le processus s'exécute, il change d'état.

Généralement, de façon simplifiée, on peut imaginer un SE dans lequel les processus pourraient être dans trois états fondamentaux (Prêt, Elu, Bloqué) et deux états secondaires (Création et Fin).

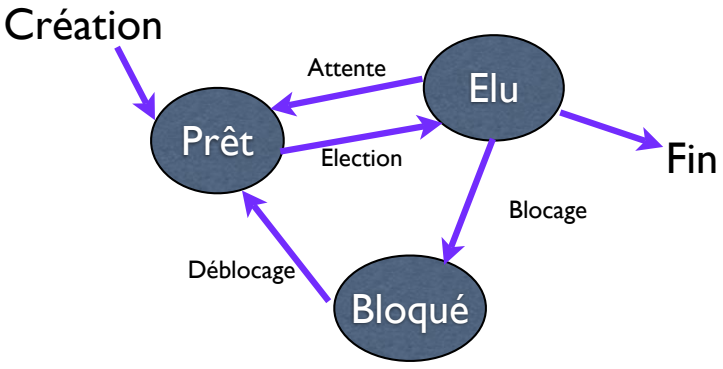

Figure2 : Etats d'un Processus

Passif (création): A l'état passif un processus lui correspondant est initialisé par le chargeur système pour exécution. Il existe des appels systèmes permettant de créer un processus, charger son contexte et lancer son exécution.

Prêt : le processus n'est pas alloué à l'UC, mais qui est prêt à être exécuté. Un processus prêt pourrait s'il était alloué à l'UC. Le processus peut être dans la file d'attente à l'état prêt à la suite d'une suspension provisoire en vue de développer en vue de développer un autre processus correspondant à un autre programme. Il peut revenir à l'état prêt à la suite d'un déblocage du à l'arrivé d'un événement attendu qui avait été la cause de son blocage.

Elu (état d'exécution): il a le contrôle du processeur. Il reste à l'état élu jusqu'à ce qu'il soit fini de s'exécuter ou bien qu'il soit suspendu et il restera dans l'état prêt, au contraire il restera à l'état bloqué.

Bloqué: il est en attente de la libération d'une ressource (qui peut être un fichier une données, une zone mémoire, un périphérique etc.) ou de la fin d'un processus.

Terminé: Lorsqu'il a fini de s'exécuter et attend la libération des ressources allouées et éventuellement sa suppression.

La possibilité qu'a un processeur de commuter d'un programme à un autre en exécutant chaque programme quelques dizaines ou quelques centaines de millisecondes s'appelle **pseudoparallélisme**. A un instant donné, le processeur n'exécute réellement qu'un seul programme. En revanche, pendant une seconde, il peut exécuter plusieurs programmes. Ceci procure aux utilisateurs l'impression de parallélisme.

.

II-LE CONTEXTE D'UN PROCESSUS

Le processeur commute entre les processus sous la direction d'un ordonnanceur. Le processus est caractérisé par les données qu'il manipule, qui sont stockées dans une zone mémoire allouée, et par l'état du processeur courant s'il est actif ou après sa dernière instruction exécutée s'il ne l'est pas. A la perte de contrôle du processeur ces informations appelées contextes sont gardées dans une zone mémoire appelé bloque de contexte ou zone de contexte. La Figure3 montre les schéma d'exécution de trois processus A, B et C.

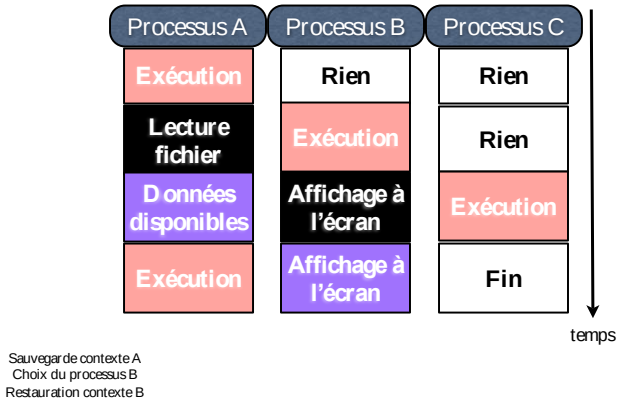

Figure3 : Schéma d'exécution de 3 processus

Les informations (contexte) doivent être restaurées lors de sa prochaine élection. **Le contexte d'un processus** est l'ensemble des informations dynamiques qui représente l'état d'exécution d'un processus (e.g. où est-ce que le processus en est de son exécution).

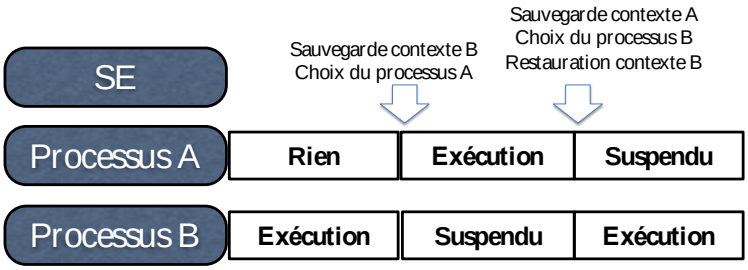

Figure5 : Sauvegarde et restauration de contexte

Le SE regroupe des informations-clés sur le fonctionnement du processeur, c'est le mot d'état du processeur (Processor Status Word, PSW) qui comporte généralement : la valeur du compteur ordinal, des informations sur les interruptions, le privilège du processeur etc.

A cet effet, à tout processus, on associe un bloc de contrôle de processus (BCP). Il comprend généralement:

- une copie du PSW au moment de la dernière interruption du processus
- l'état du processus : prêt à être exécuté, en attente, suspendu, ...
- des informations sur les ressources utilisées
    - mémoire principale
    - temps d'exécution
    - périphériques d'E/S en attente
    - files d'attente dans lesquelles le processus est inclus, etc...
    - et toutes les informations nécessaires pour assurer la reprise du processus en cas d'interruption

Les BCP sont rangés dans une table en mémoire centrale à cause de leur manipulation fréquente.

Lorsqu'un programme s'exécute, il est fait appel à un ensemble de procédures s'adressant les uns aux autres. On appel contexte d'activité la portion d'informations accessible au processus pendant un état actif; il inclut celui du processeur et celui de la mémoire. La transition d'une activité à l'autre est réalisée grâce à des instructions spéciales d'appels et de retours de procédures qui produisent un changement de contexte ou commutation de contexte. **La commutation de contexte** est le mécanisme qui permet au système d'exploitation de remplacer le processus élu par un autre processus éligible comme le montre la Figure suivante.

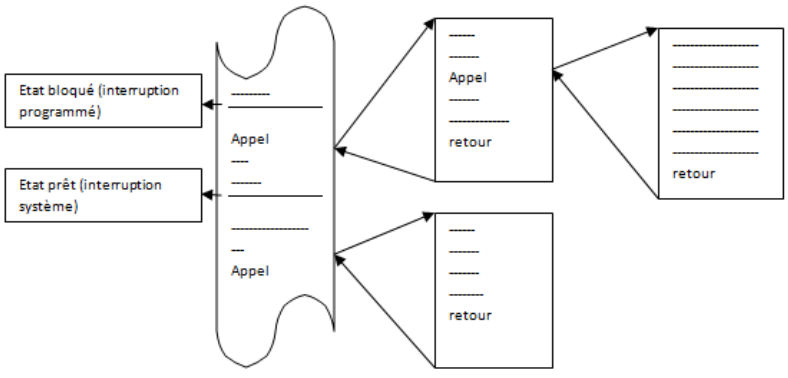

Figure6 : Commutation des processus du point de vue de l'utilisateur

Pour réaliser ce mécanisme, il faut assurer les fonctions suivantes.

- Conservation et restauration des contextes lors des appels et retours
- Transmission des paramètres entre procédures appelantes et appelées.

- Gestion des zones mémoires permettant un accès récursif.

Pour se faire on peut utiliser les structures de données comme les piles. Les paramètres peuvent ainsi se transmettre selon les valeurs ou les adresses ; les procédures s'appellent récursivement de manière implicite ou explicite. En résumé les trois paramètres suivants suffise pour caractériser ses fonctionnalités :

- L'adresse de base du premier mot du programme (stockée dans le registre Code Segment CS)
- Le pointeur sue la prochaine instruction à exécuter (Instruction Pointer registre IP)
- La pile qui empile les appels de fonctions avec leurs paramètres (registre Stack Pointer SP)

Le coefficient de **multiprogrammation** est le nombre de contexte qu'on peut garder dans la pile et ceci détermine le nombre d'appel de fonction imbriqué que la pile peut prendre.