Gestion de processus

Catherine Faron Zucker, UNSA, EPU SI1
faron@polytech.unice.fr

Gestion de processus

- Définition d'un processus
- Attributs et espace d'adressage
- Ordonnancement des processus

Définition d'un processus (1)

- Aspect dynamique d'un programme
  - Programme
    - Ensemble de procédures,
    - Résultat statique d'une compilation/édition de liens (binaire exécutable)
  - Processus
    - Activité dynamique résultant de l'exécution d'un programme
    - Enchaînement d'exécutions séquentielles de procédures

Définition d'un processus (2)

- Unité de gestion d'activité
  - Unité d'allocation des ressources système
    - Fichiers, évènements, périphériques, mémoire, …
  - Espace d'adressage
- Unité d'ordonnancement
  - Entité affectable à un processeur
  - Flot de contrôle séquentiel

Processus sous Unix

- Unix traditionnels
  - Processus =
    unité de gestion d'activité + unité d'ordonnancement
- Unix modernes
  - Processus « lourd » : unité de gestion d'activité
  - Processus « légers » : unités d'ordonnancement
    - Partagent un même espace d'adressage

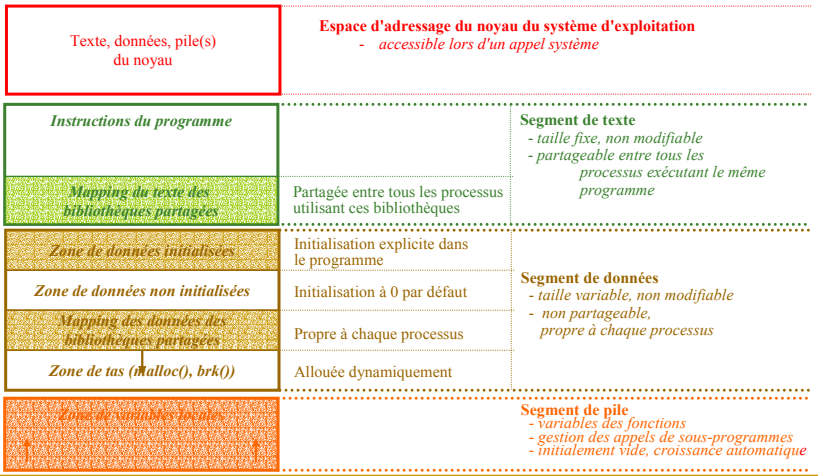
Attributs d'un processus

- Informations nécessaires à la gestion d'un processus par l'OS
- Sauvegardés lors de la suspension du processus et restauré lors de sa reprise
  - Identification du processus,
  - Identification de l'utilisateur,
  - Ressources possédées (fichiers ouverts...)
  - Contexte matériel
    - compteur ordinal, registres de l'unité centrale

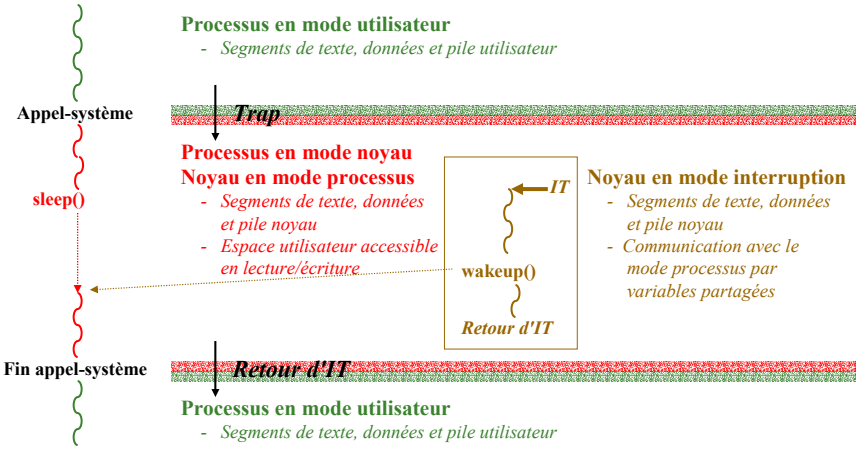
Espace d'adressage d'un processus

- Espace d'adressage logique d'un processus
  - Zone de données système (attributs) *protégé*
    - Contexte matériel et logiciel
  - Zone de texte *taille fixe*
    - Code, instructions
  - Zone de pile *taille variable*
    - Variables locale, gestion des sous-programmes
  - Zone de données *taille variable*

Espace d'adressage virtuel d'un processus sous unix

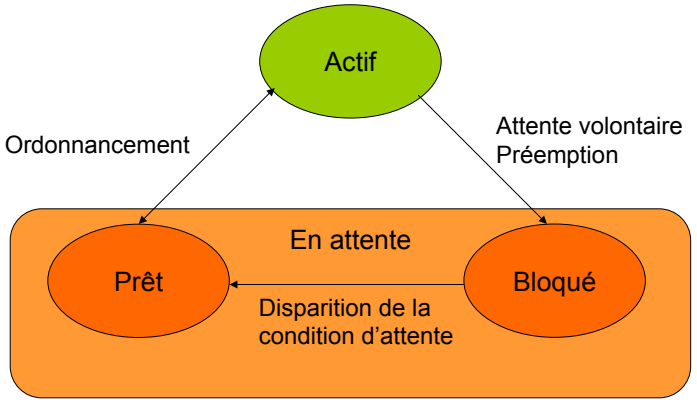


Exécution d'un appel système sous unix



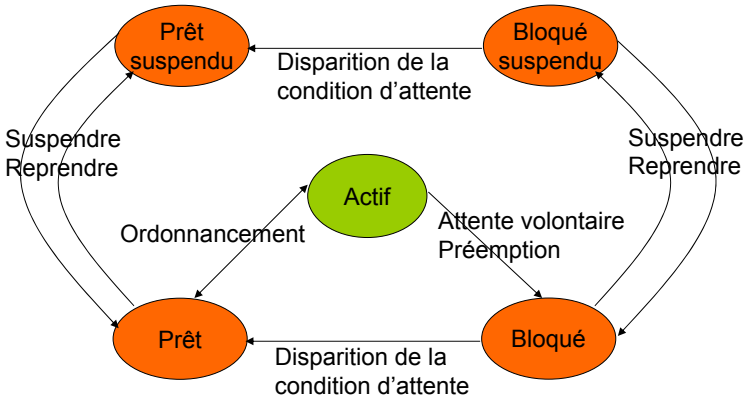
Ordonnancement de processus

- Etats d'un processus



Ordonnancement de processus

- Etats d'un processus sous Unix



Ordonnancement de processus

- Choix du processus actif
  - Lequel parmi les processus prêts?
  - Quand?
- Ordonnancement préemptif
  - L'OS peut retirer la CPU au processus actif à tout moment
  - Garantie que le processus « le plus prioritaire » sera actif dès qu'il sera prêt
- Ordonnancement non préemptif
  - Run to completion: le processus actif laisse volontairement la CPU
  - Dans les premiers systèmes de batch
  - Plus simple à programmer, et moins coûteux

Ordonnancement de processus

- Objectifs
  - Equité
    - Chaque processus a sa part de CPU, une classe de processus n'est pas trop favorisés par rapport à une autre
  - Efficacité et rendement
    - Maximiser l'activité de la CPU
    - Maximiser le nombre de jobs exécuté en un temps donné
  - Temps de réponse, interactivité
    - Minimiser le temps d'attente pour une sortie de programme
    - Minimiser les temps de réponse pour les utilisateurs interactifs
  - Dégradation gracieuse sous forte charge
  - Prédictabilité
    - Les processus ne sont pas prédictibles, quand l'OS lance un processus il ne sait pas quand celui-ci sera bloqué (préemption préconisée)

Algorithmes d'ordonnancement (1)

- Algorithme FIFO
  - Non préemptif
  - Formellement équitable, pratiquement non équitable
- Algorithme à tourniquet (« Round Robin »)
  - Préemptif par tranche de temps (quantum)
  - Si le quantum est trop court, le temps passé en changements de contexte est trop long et réduit l'efficacité de la CPU
  - Si le quantum est trop long, les temps de réponse sont trop longs pour des processus interactifs courts

Algorithmes d'ordonnancement (2)

- Le plus court d'abord (« Shortest Job First »)
  - Connaissance *a priori* du temps d'exécution
  - Le temps d'attente *moyen* est minimisé si les processus les plus courts sont exécutés avant les plus longs
  - Non préemptif donc imprévisible
- Le temps restant le plus court d'abord (« Shortest Remaining Time »)
  - Amélioration de SJF qui devient préemptif
  - Plus coûteux

Algorithmes d'ordonnancement (3)

- Ordonnancement garanti & Ordonnancement temps réel
  - si n utilisateurs sont logés, chacun peut être assuré de recevoir 1/n de temps de CPU (taux monotone)
  - En temps réel, la plus courte échéance est prioritaire sur la plus longue

Algorithmes d'ordonnancement (4)

- Ordonnancement à priorité
  - Une priorité (un entier) est attribuée à chaque processus
  - L'ordonnanceur choisit le processus de plus haute priorité parmi les processus prêts
    - Si l'ordonnancement est préemptif, le processus actif est toujours celui de plus haute priorité
- Priorités fixes versus priorités variables
  - Fixées par le programmeur (temps réel)
  - L'ordonnanceur diminue la priorité du processus actif (de plus haute priorité) à chaque interruption d'horloge pour empêcher qu'il ne monopolise la CPU

Algorithmes d'ordonnancement (5)

- Ordonnancement à priorité et tourniquet
  - Les processus sont groupés par classes de priorité
  - Un ordonnancement par priorité est appliqué entre les classes de priorité
  - Un ordonnancement à tourniquet est appliqué entre les processus d'une même classe de priorité
- UNIX BSD & Linux jusqu'à 2.4

Algorithmes d'ordonnancement (5)