


2009 - 2010

A set of four horizontal lines, with the second line from the top being a darker teal color and the others being a lighter teal.

Systemes d'exploitation 2

TD n 3 : Applications classiques

A set of four horizontal lines, with the second line from the top being a darker teal color and the others being a lighter teal.

Mme Lilia SFAXI

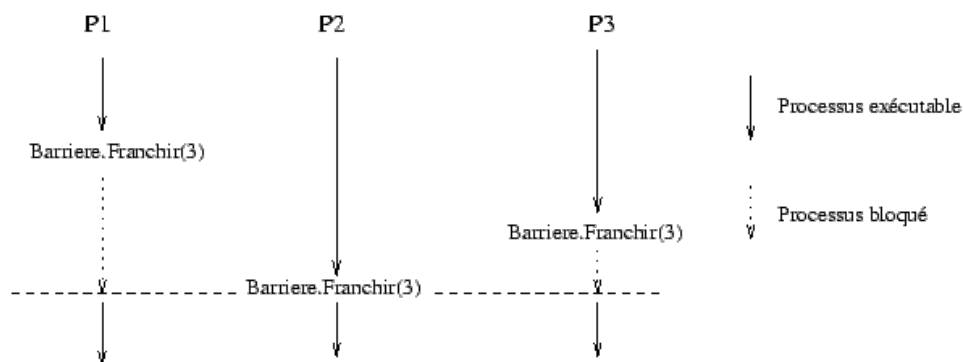
Systèmes d'exploitation 2

TD n 3 : Applications classiques

Exercice 1 : Les Barrières

Les barrières représentent un mécanisme de synchronisation destiné aux groupes de processus. Certaines applications sont décomposées en phases, elles ont pour règle qu'aucun processus ne peut entrer dans la phase suivante tant que les autres processus ne sont pas prêts à y entrer. Ce comportement peut se produire si on place une barrière à la fin de chaque phase. Lorsqu'un processus atteint la barrière, il est bloqué jusqu'à ce que tous les processus l'atteignent également.

Soient N processus parallèles ayant un point de rendez-vous. Un processus arrivant au point de rendez-vous se met en attente s'il existe au moins un autre processus qui n'y est pas arrivé. Le dernier arrivé réveillera les processus bloqués.



Ecrire l'algorithme réalisant ce scénario.

Exercice 2 : Le dîner des philosophes

Le dîner des philosophes est un problème classique de synchronisation des processus proposé et résolu par Dijkstra. La situation est la suivante :

- cinq philosophes se trouvent autour d'une table ;
- chacun des philosophes a devant lui un plat de spaghetti ;
- à gauche de chaque assiette se trouve une fourchette.



Un philosophe n'a que trois états possibles :

- penser pendant un temps indéterminé ;
- être affamé (pendant un temps déterminé et fini sinon il y a famine) ;
- manger pendant un temps déterminé et fini.

Des contraintes extérieures s'imposent à cette situation :

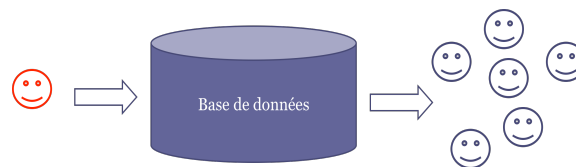
- quand un philosophe a faim, il va se mettre dans l'état « affamé » et attendre que les fourchettes soient libres ;
- pour manger, un philosophe a besoin de deux fourchettes : celle qui se trouve à sa droite, et celle qui se trouve à sa gauche ;
- si un philosophe n'arrive pas à s'emparer d'une fourchette, il reste affamé pendant un temps déterminé, en attendant de renouveler sa tentative.

Le problème consiste à trouver un ordonnancement des philosophes tel qu'ils puissent tous manger, chacun à leur tour.

Exercice 3 : Problème des lecteurs-rédacteurs

Le problème des lecteurs et des rédacteurs est un problème classique en théorie informatique, qui permet de modéliser les accès à des bases de données.

Soit un système de réservation de billets d'avion, par exemple, où de nombreux processus entrent en concurrence pour effectuer des lectures et des écritures. Il est acceptable que plusieurs processus puissent lire la base de données en même temps. Mais l'accès en écriture et les accès en lecture sont mutuellement exclusifs.



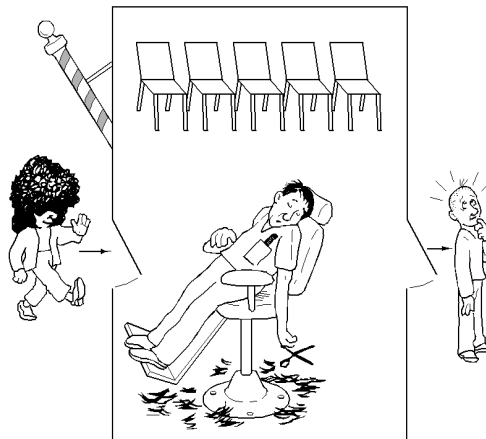
Ecrire le pseudo-code du processus lecteur et du processus rédacteur dans les deux cas suivant :

- version simplifiée : un seul lecteur est autorisé à se connecter à la base à la fois
- version généralisée : plusieurs lecteurs sont autorisés à se connecter en même temps.

Exercice 4 : Le coiffeur endormi

Il s'agit encore d'un de ces problèmes de synchronisation mis sous une forme "plaisante". Mais celui-ci est encore plus sérieux que le problème des philosophes, car on en trouve une application presque directe dans certains mécanismes des systèmes d'exploitation (comme l'ordonnancement des accès disque).

Un coiffeur possède un salon avec un siège de coiffeur et une salle d'attente comportant un nombre fixe F de fauteuils.



S'il n'y a pas de clients, le coiffeur se repose sur son siège de coiffeur. Si un client arrive et trouve le coiffeur endormi, il le réveille, s'assoit sur le siège de coiffeur et attend la fin de sa coupe de cheveux. Si le coiffeur est occupé lorsqu'un client arrive, le client s'assoit et s'endort sur une des chaises de la salle d'attente ; si la salle d'attente est pleine, le client repasse plus tard. Lorsque le coiffeur a terminé une coupe de cheveux, il fait sortir son client courant et va réveiller un des clients de la salle d'attente. Si la salle d'attente est vide, il se rendort sur son siège jusqu'à ce qu'un nouveau client arrive.

Ecrire un pseudo-code permettant de synchroniser entre les deux processus : le coiffeur et le client.