



1 Présentation

Votre travail consistera à écrire un programme illustrant la mise en place d'un ordonnanceur. Ce travail donnera lieu à un compte-rendu et à un programme qui sera noté.

Pour le compte-rendu, vous devrez mettre en avant les concepts du cours que vous aurez appliqué ici. Vous indiquerez également les structures de donnée utilisées. La clarté, la pertinence et les justifications serviront de base à la notation.

Pour le programme, vous devrez être le plus fidèle possible aux spécifications énoncées ci-dessous. Vous ferez attention à la cohérence entre les éléments décrits dans le compte-rendu et les éléments de votre programme. La clarté, la propreté et le fonctionnement de votre programme serviront de base à la notation. En fonction de votre niveau, vous pouvez commencer en simplifiant le problème mais précisez vos hypothèses de départ. Le programme devra impérativement être écrit en langage java.

2 L'ordonnanceur

Il s'agit de simuler l'ordonnancement de processus. Un tick horloge, simulé par une boucle dans le programme principal, donne la main à l'ordonnanceur qui choisit la tâche à exécuter en fonction de l'état du système. Suivant l'algorithme d'ordonnancement choisi, il choisira soit de laisser s'exécuter une tâche précédemment élue, soit d'élire une nouvelle tâche, soit d'élire la tâche IDLE correspondant au système au repos.

Il sera possible de choisir l'algorithme d'ordonnancement parmi les suivants:

- Premier arrive premier servi (First Come First Serve)
- Le plus petit travail en premier (Short Job First)
- Le plus court temps restant (Shortest Remaining Time)
- Tourniquet (Round robin)
- Priorité

Le temps est simulé et relatif au programme, c'est à dire qu'un passage dans la boucle principale du programme correspond à un tick d'horloge, durant lequel on suppose qu'une action atomique est effectuée par la tâche élue.

Suivant l'algorithme d'ordonnancement choisi, on doit pouvoir choisir le quantum de temps durant lequel une tâche peut s'exécuter.

3 Les tâches

Des tâches seront définies au début d'une simulation puis seront exécutées et ordonnancées en fonction de l'ordonnanceur et de l'état de la machine.

Deux types de tâches seront définis :

1. Le premier type de tâche incrémente à chaque « action atomique » un compteur.
2. Le deuxième type de tâche devra en fonction de paramètres définis à leur création accéder à un sémaphore d'exclusion mutuel avant de pouvoir incrémenter un compteur partagé par

l'ensemble des tâches. Cette tâche aura donc quatre types d'action atomique : incrémenter son compteur, réserver le sémaphore, incrémenter le compteur privé et relâcher le sémaphore.

Il faudra donc définir à la création d'une tâche et en fonction de l'algorithme d'ordonnancement choisi :

- Le type de tâche
- Le temps d'exécution de la tâche
- Le temps d'arrivée de la tâche
- Sa priorité
- Le temps d'arrivée de la réservation du sémaphore d'exclusion mutuel
- Le temps d'exécution avant la libération du sémaphore d'exclusion mutuel, une fois celui-ci acquit.

4 Le sémaphore d'exclusion mutuel

Les tâches de type deux se partageront un sémaphore d'exclusion mutuel.

Le sémaphore devra implémenter les fonctionnalités classiques d'un sémaphore, c'est-à-dire une méthode de réservation, une méthode de libération... On pourra également savoir quelle tâche à obtenir l'accès exclusif et quelles sont les autres tâches qui sont bloquées à cause de cet accès exclusif.

5 Statistiques

A la fin d'une simulation, le programme devra afficher des statistiques.

La première sera un graphe temporel permettant de visualiser la succession d'exécution des tâches. A chaque temps processus devra être associée une tâche. Par exemple :

T0 => Processus 1

T1 => Processus 1

T2 => Processus 4

...

Tn => Processus x

Le deuxième jeu de statistiques concerne :

- Temps moyen de séjour d'une tâche
- Temps moyen d'attente d'une tâche
- Nombre de changement de contexte

6 Affichage en cours de simulation

Pendant l'exécution de la simulation, l'interface utilisateur permettra de visualiser la tâche élue, l'état des tâches, de leur compteur, l'état du sémaphore et l'état du compteur partagé ainsi que toutes autres informations que vous jugerez utile d'afficher.

7 Indications supplémentaires

- Il est conseillé de créer le modèle objet, d'implémenter les méthodes avant de commencer l'interface graphique.

- Ne négliger pas le rapport, un bon rapport indique un bon programme. Cela fonctionne aussi dans l'autre sens.
- Il n'est pas interdit d'aider ces camarades mais il est interdit de s'échanger du code ou des bouts de code entre binôme.
- L'utilisation de la classe `java.lang.Thread` n'est pas obligatoire ici. Tout comme la classe `java.util.concurrent.Semaphore`.