UNIVERSITE DE M'SILA, Faculté des Mathématiques et de l'Informatique Département d'informatique

Module : Réseau et systèmes repartis Année : 2014-2015
1 ere année Master

TP1: Revision sur les threads et la synchronisation des process

Les sémaphores de Java

Un morceau de code est dit réentrant s'il peut être exécuté par plusieurs threads en même temps. Il peut être nécessaire de limiter le nombre de threads accédant au code.
Un sémaphore permet d'autoriser plusieurs threads à accéder à du code réentrant.
Le sémaphore gère un ensemble de permis, en nombre fixé, et accorde ces permis aux threads qui en font la demande.
Java.util.concurrent.Semaphore implémente ce mécanisme :

- **Semaphore(int i) :**crée un sémaphore avec i permis initiaux
  - Semaphore sem1=new Semaphore() // avec 1 permis
  - Semaphore sem2=new Semaphore(3) // avec 3 permis

- **Semaphore(int i,boolean fairness):**Si fairness== true alors la file d'attente est gère d'après la politique FIFO
  - Semaphore sem1=new Semaphore(1,true)//fifo avec 1permis

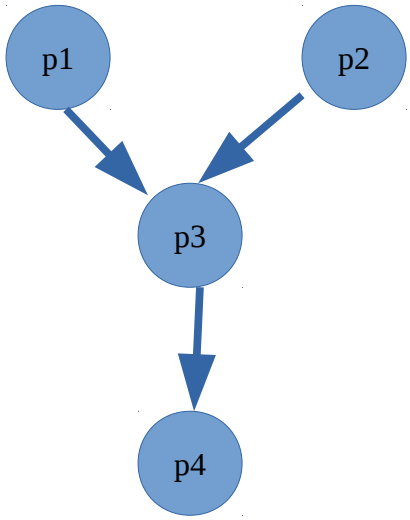
- **la méthode acquire()** :permet de requérir un permis et bloque le thread demandeur jusqu'à ce qu'un permis soit disponible ou que le thread soit interrompu**.**
  - sem1.aquire() // requérir 1 seul permis

- **la méthode acquire(int) :**permet de requérir plusieurs permis.
  - sem1.aquire(2) // requérir 2 permis
- **la méthode release()** : rendre 1 permis.
  - sem1.release() // augmente le nbr de permis de 1

- **release(int i) :** rendre i permis.
  - **sem1.release(2) // augmente le nbr de permis de 2**

Exercice 1

On veut créer les threads et gérer la synchronisation entres eux comme il montre dans la figure



de tel sorte que chaque thread va afficher son id et puis va attendre 1000ms avant le lancement du thread qui est synchronisez avec lui.

Exercice 2

On veut calculer les elements d'une matrice de taille (n*n). Pour faire ce traitement nous voulons utilises 5 thread de tels sorte que la matrice est divisée virtuellement en 4 parties et chaque thread va calculer la somme des éléments dans une partie. Le cinquième thread va afficher le résultat final