Université Paris 7
M2 II
Programmation Répartie

TP

Verrous pour les lecteurs/redacteurs

Dans le package `java.util.concurrent.locks` se trouve l'interface ReadWriteLock.

```
public interface ReadWriteLock{
 public Lock readLock();//Lock utilise pout lire
 public Lock writelock();//Lock utilise pour ecrire
 }
```

qui permet de résoudre le problème des lecteurs écrivains. On rappelle que dans ce problème où lecteurs et écrivains veulent accéder à une "ressource", un écrivain travaille en exclusion mutuelle avec les lecteurs et les écrivains. Un lecteur est en exclusion mutuelle avec les écrivains, mais plusieurs lecteurs peuvent accéder en même temps à la "ressource".

Le but de ce TP est d'implementer cette interface avec différents priorités pour les lecteurs et les écrivains.

Pour ce faire on utiliser la classe `java.util.concurrent.Semaphore`

1. Pour pouvoir tester l'implémentation du `ReadWriteLock` écrire le code d'une thread lectrice qui consulte et d'une thread écrivain qui modifie une base de données (un tableau d'entiers).

   Testez vos classes avec l'implémentation `ReentrantReadWriteLock` de l'interface `ReadWriteLock` .

2. Réalisez l'implémentation avec la *Priorité des lecteurs sur les rédacteurs.* Si un lecteur se présente alors que des lecteurs sont en train de lire la base de données, le lecteur peut consulter la base. Les lecteurs peuvent donc occuper indéfiniment la base de données. Il y a famine des écrivains.

3. Réalisez l'implémentation avec la *Priorité des rédacteurs sur les lecteurs.* Dés qu'un rédacteur réclame l'accés à la base de données, il doit l'obtenir le plus tôt possible sans réquisition (i.e. sans arrêter les utilisateurs actuels de la base de données).

4. Réalisez l'implémentation avec la *Priorité égale pour les rédacteurs et les lecteurs.* Si un lecteur utilise la base de données, tous les lecteurs nouveaux peuvent y accéder jusqu'à l'arrivée d'un rédacteur. A partir de ce moment tous les nouveaux arrivant attendent. Quand le dernier lecteur a fini, il réveille l'écrivain en attente. Quand cet écrivain a terminé il réveille un des processus en attente.