Theorie et pratique de la concurrence { Master 1 Informatique
TP 6 : Concurrence et Distribution

Dans ce Tp on fera quelques exercices de programmation distribuee sans l'utilisation de variables partagees. La communication entre threads se fera a l'aide de passage de message en utilisant le protocole TCP. Pour cela on utilisera les classe `Socket` et `ServerSocket` du package `java.net` Les messages seront de type `OutputStream()` et `InputStream()`.

Exercice 1:

Programmez une classe `ConcurrentBuffer` avec un champ de type `LinkedList<String>` et deux methodes `put` et `get` qui agissent en exclusion mutuelle :

- La methode `put` ajoute une cha^ne de caracteres a la liste et ensuite reveille les threads en attente sur cette liste.
- La methode `get`, si la liste est vide, met en attente le thread appelant, sinon elle renvoie la premiere cha^ne de caracteres de la liste, et la retire de la liste.

Exercice 2:

Implementez en Java une version distribuee de l'algorithme producteur/consommateur . Pour ce faire, un serveur ecoute sur deux ports. Sur un port, le serveur attend les connexions d'un client producteur et sur l'autre port les connexions d'un client consommateur. Un client producteur envoie au serveur une cha^ne de caracteres qu'il aura recu par l'entree standard. Le serveur stocke alors cette cha^ne de caracteres dans un objet de type `ConcurrentBuffer` Un client consommateur attend que le serveur lui envoie des cha^nes de caracteres qui ont ete places dans le bu er.

Attention : le serveur attende des connexion sur deux ports di erents.

Exercice 3:

Modi ez le programme de l'exercice precedent pour avoir plusieurs producteurs et plusieurs consommateurs

Exercice 4:

Implementez une version distribuee du d^ner des philosophes. Les baguettes ne sont pas representees explicitement. Quand un philosophe veut prendre une baguette, demande la permission a son voisin. Si celui ci est en train de manger, le premier philosophe attend que son voisin lui envoie le message en disant qu'il a termine.