

SR2 – Partie « Systèmes » - Travaux pratiques n°4&5

Threads Posix - Sémaphores

Exercice 1 – Alternance d'affichage

On veut synchroniser l'affichage de NBT threads de manière à ce qu'ils affichent NBM messages (constitué de NBL lignes) à tour de rôle à l'écran (même principe que l'exercice vu en TD).

L'application sera paramétrée par les valeurs de NBT, NBM et NBL.

Attention : Chaque thread sera paramétré par le nombre de messages et le nombre de lignes qu'il devra afficher (ce ne seront pas des variables partagées).

Version 1. La synchronisation sera réalisée à l'aide de verrous d'exclusion mutuelle Posix (pthread_mutex_t).

Exemples d'exécution : ./exo1v1 2 2 3

```
Afficheur 0 (140370571626240), j'affiche ligne 0/2 du message 0/3
Afficheur 0 (140370571626240), j'affiche ligne 1/2 du message 0/3
Afficheur 1 (140370563172096), j'affiche ligne 0/2 du message 0/3
Afficheur 1 (140370563172096), j'affiche ligne 1/2 du message 0/3
Afficheur 0 (140370571626240), j'affiche ligne 0/2 du message 1/3
Afficheur 0 (140370571626240), j'affiche ligne 1/2 du message 1/3
Afficheur 1 (140370563172096), j'affiche ligne 0/2 du message 1/3
Afficheur 1 (140370563172096), j'affiche ligne 1/2 du message 1/3
Afficheur 0 (140370571626240), j'affiche ligne 0/2 du message 2/3
Afficheur 0 (140370571626240), j'affiche ligne 1/2 du message 2/3
Afficheur 0 (140370571626240), je me termine
Afficheur 1 (140370563172096), j'affiche ligne 0/2 du message 2/3
Afficheur 1 (140370563172096), j'affiche ligne 1/2 du message 2/3
Afficheur 1 (140370563172096), je me termine
Fin de l'execution du thread principal
```

Version 2. La synchronisation sera réalisée à l'aide de sémaphores Posix (sem_t).

Exemples d'exécution : ./exo1v2 2 2 3

La même chose que ci-dessus.

Exercice 2 – Compte bancaire

Ajoutez la synchronisation nécessaire pour résoudre les problèmes constatés lors de l'exécution de l'exercice 2 du TP 3.

Exercice 3 – Modèle du producteurs-consommateurs

Si ce n'est fait, terminez l'exercice 3 du TP3. Ajoutez la synchronisation nécessaire pour résoudre les problèmes de synchronisation et permettre aux threads de déposer et retirer leurs messages de manière cohérente.

Rappel du principe : Les dépôts se font de manière circulaire dans l'ordre des indices croissants et les retraits se font dans l'ordre des dépôts.

Exemples d'exécution : ./exo3 2 2 2 2 1

➔ 2 producteurs et 2 consommateurs déposant ou retirant chacun 2 messages, tableau de taille 1

```

Prod 0 (140442678724352) : Message depose = Bonjour 1 de prod 0
Conso 0 (140442661816064) : Message retire = Bonjour 1 de prod 0
Prod 1 (140442670270208) : Message depose = Bonjour 1 de prod 1
Conso 1 (140442653361920) : Message retire = Bonjour 1 de prod 1
Prod 0 (140442678724352) : Message depose = Bonjour 2 de prod 0
Conso 0 (140442661816064) : Message retire = Bonjour 2 de prod 0
Prod 1 (140442670270208) : Message depose = Bonjour 2 de prod 1
Conso 1 (140442653361920) : Message retire = Bonjour 2 de prod 1

```

Fin de l'exécution du main

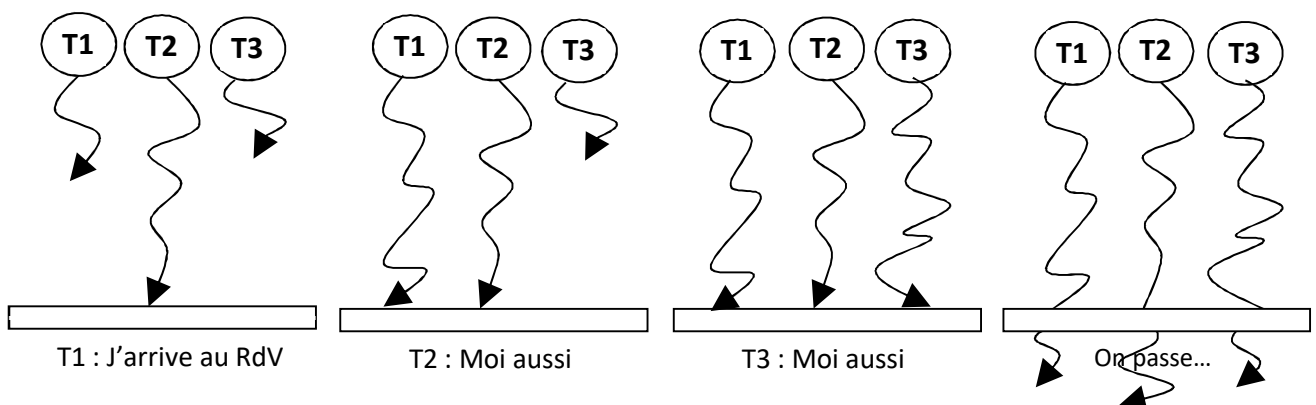
Exercice 4 – Rendez-vous entre threads

En vous basant sur ce qui a été évoqué éventuellement en CTD, programmez le rendez-vous entre N threads.

L'application sera paramétrée par le nombre N de threads.

Rappel du principe du rendez-vous : un thread arrivant au point de rendez-vous se met en attente s'il existe au moins un autre thread qui n'y est pas arrivé. Tous les threads bloqués sur cette « barrière » peuvent la franchir lorsque le dernier y est arrivé.

La figure ci-dessous illustre ce comportement pour un rendez-vous à 3 :



Un thread aura le comportement suivant :

Début

```

Je fais un certain traitement ;
J'arrive au point de rendez-vous
    et j'attends que tous les autres y soient aussi... ;
...Avant de pouvoir continuer mon traitement ;

```

Fin

Exemples d'exécution : ./exo4 3

```
Thread 0 (139981516506880) : J'effectue un traitement en parallele avec les autres
Thread 0 (139981516506880) : J'arrive au RdV
Thread 0 (139981516506880) : Je ne suis pas le dernier au RdV
Thread 1 (139981508052736) : J'effectue un traitement en parallele avec les autres
Thread 1 (139981508052736) : J'arrive au RdV
Thread 1 (139981508052736) : Je ne suis pas le dernier au RdV
Thread 2 (139981499598592) : J'effectue un traitement en parallele avec les autres
Thread 2 (139981499598592) : J'arrive au RdV
Thread 2 (139981499598592) : Je suis le dernier au RdV
Thread 2 (139981499598592) : Je passe le point de RdV
Thread 2 (139981499598592) : Je continue un traitement en parallele avec les autres
Thread 0 (139981516506880) : Je passe le point de RdV
Thread 0 (139981516506880) : Je continue un traitement en parallele avec les autres
Thread 1 (139981508052736) : Je passe le point de RdV
Thread 1 (139981508052736) : Je continue un traitement en parallele avec les autres
Fin de l'execution du thread principal
```

Pour aller plus loin...

Programmez les différents exercices vus en CTD.