USTHB
Faculté d'Electronique et d'Informatique
Département Informatique

SERIE N° 3
Synchronisation (sémaphores, moniteurs & régions critiques)
Année : 2021/2022 le 12/10/2021

Exercice N° 1 : Soit un système composé de trois processus fumeurs et d'un processus agent.
Chaque fumeur roule continuellement une cigarette et la fume. Pour pouvoir rouler une cigarette, trois ingrédients sont nécessaires : le tabac, le papier et les allumettes.
Un des processus fumeurs détient le papier, le deuxième processus le tabac et le troisième les allumettes.
Le processus agent a un approvisionnement infini des trois ingrédients. L'agent place deux ingrédients distincts sur la table ainsi le fumeur qui possède le troisième ingrédient peut rouler une cigarette, la fumer puis signale à l'agent la fin de son opération. L'agent remet sur la table deux autres ingrédients et ainsi le cycle se répète.
- Synchroniser les processus à l'aide des sémaphores
- Modifier la solution pour permettre à plus d'un fumeur d'opérer en même temps.

Exercice N° 2 : Des voitures venant du nord et du sud doivent traverser un pont. Sur ce pont ne peut passer qu'une seule file de voitures à la fois et dans un même sens.
- Ecrire un algorithme qui permet aux voitures de passer du nord vers le sud ou du sud vers le nord en se synchronisant à l'aide des sémaphores au niveau du pont.
- Discuter votre solution et proposer d'éventuelles modifications pour diverses situations.

Exercice N° 3 : Soit un système constitué d'un processus producteur et un processus consommateur qui se partagent deux tampons T1 et T2 de tailles respectives M et N, comme suit :
- Le producteur, à tout moment, ne peut déposer dans T2 que si le tampon T1 est plein.
- Le consommateur, à tout moment, ne peut prélever du tampon T2 que si le tampon T1 est vide.
    1- Synchroniser les deux processus en utilisant les sémaphores.
    2- Généraliser cette solution au cas de plusieurs producteurs et plusieurs consommateurs.

Exercice N°4 : Soient 2 processus producteurs P_1, P_2 et deux processus consommateurs C_1, C_2 qui partagent un tampon de taille fixe. On définit les contraintes suivantes d'accès au tampon :
- P_i constitue une classe CL_i,
- C_i ne peut consommer que les messages produits par le processus de la classe CL_i,
- A tout instant, le tampon ne contient que les messages produits par le processus d'une seule classe.

A/ A l'aide des sémaphores, synchroniser l'ensemble de ces processus parallèles pour l'accès au tampon.
B/ Quelles seront les modifications nécessaires à apporter à la solution,
a- si on ajoute des processus producteurs à chaque classe CLi ?
b- et de plus, si on ajoute une classe CL_3 et un consommateur C_3 ?

Exercice N°5 : Des processus « utilisation» et des processus « systèmes » se partagent n imprimantes. Les processus systèmes ont la priorité pour l'acquisition d'une imprimante.
- Décrire le comportement de ces deux classes de processus pour leur synchronisation en utilisant les moniteurs dans deux cas de demandes :
    A/ Une imprimante à la fois.
    B/ k imprimantes à la fois.

Exercice N° 6 : On considère 2 ressources appelées R1, R2. La ressource R1 existe en N1 exemplaires et la ressource R2 en N2 exemplaires.
On supposant que chaque processus peut demander :
- Soit 1 exemplaire de la ressource R1,
- Soit 1 exemplaire de la ressource R2,
- Soit 1 exemplaire de R1 et 1 exemplaire de R2

A/ Ecrire un moniteur gérant l'accès à ces ressources en donnant la priorité à celui qui exprime le troisième type de demande.
B/ Modifier la solution pour permettre l'accès FIFO.

Exercice n°7 : Etant donné un système contenant N processus évoluant de manière parallèle et dont le fonctionnement de chacun est donné comme suit :

```
Processus P (i : entier) ;   /* i : identité du processus pouvant prendre une valeur de 1 à N */
Var j : entier ; T : Tableau [1..N] de Booleen :=Faux ;
Debut -
Pour j :=1 à N Faire
        Envoyer (B, 'présent', i)
Fait ;
Repeter
        Recevoir (B , m, k) ;
        Si T[k]= Faux Alors T[k] := Vrai  Sinon Envoyer (B, m, k) Fsi ;
        j :=1 ;  Tantque (j<=N) et (T[j]= Vrai Faire j :=j+1 Fait
Jusqu'à (j>N)
Fin.
```

où B est une boite aux lettres commune (peut être utilisée par chaque processus en émission et en réception) supposée de capacité infinie.
- Envoyer (B, m, i) : permet d'envoyer un message m accompagné d'un entier i vers la boite aux lettres B.
- Recevoir (B, m, k) : permet de recevoir un message m accompagné d'un entier k dans la boite aux lettres B. cette primitive bloque le processus appelant jusqu'à la réception du message.

A/ Discuter cette solution.
- Si on suppose que chaque processus P(i) i=1, N possède sa propre boite aux lettres B(i) (donc utilisée uniquement pour la réception de messages),

B/ Réécrire la solution précédente avec cette nouvelle considération.
C/ Comparer les deux solutions.

Exercice 8: Dans un système d'exploitation, on dispose des primitives de communication par boites aux lettres suivantes :
- Send (B, message) : Dépôt de message dans la boite aux lettres B.
- Receive (B, message) : Attente et Retrait de message de la boite aux lettres B.

- Soit le processus suivant:

```
Processus P ;
Struct m, mess : ….. ;
Debut
    -
    Send( B1, ''vide'') ; Send (B2,''vide'') ;
    Répéter
        Receive (B1, m) ;
        Si (m<>''vide'')  Alors mess ← m
                          Sinon  Send (B1,''vide'') ;
                                 Receive (B2, m) ;
                                 Si (m<>''vide'') Alors mess ←m
                                                  Sinon Send (B2,''vide'')
                                 Fsi
        Fsi
    Jusqu'à (mess<>''vide'')
Fin.
```

B1 et B2 étant deux boites aux lettres communes à P et à d'autres processus.
1/ Expliquer le fonctionnement de la solution, déduire sa fonction.

- Remplaçons la primitive *Receive ()* par la fonction *Read (B)* qui retourne le premier message de la boite aux lettres B s'il existe sinon vide.
2/ Réécrire le processus P.

Exercice n°9

- Ecrire une solution à l'aide des régions critiques qui permet de réaliser le RDV de N processus parallèles de telle sorte qu' à l'arrivée de tous ces processus à ce point de rendez-vous, ils reprennent leurs exécutions dans un ordre prédéfini lié à leur ordre d'arrivé (on retiendra l'ordre inverse de leur ordre d'arrivée).