

Exercice 1 : Reprendre les exercices de la série 2 (TD n°2) en utilisant les moniteurs.

Exercice 2 : Modèle du Problème des Lecteurs-Rédacteurs

Ce problème, dit de Lecteurs-Rédacteurs (cou et al. 1971), constitue un modèle ou classe de problèmes de synchronisation des processus séquentiels spécifiques à une politique particulière d'accès à un objet partagé. Tout autre problème de synchronisation qui s'apparente à cette politique d'accès forme une variante de ce modèle.

Ce problème traite du partage d'un objet, ici le fichier, par un ensemble d'utilisateurs (ou processus). Il s'agit d'autoriser plusieurs accès simultanés en lecture à un fichier, et de garantir un accès exclusif relativement aux écritures dans ce même fichier. Deux types de processus communément appelés lecteurs et rédacteurs se trouvent donc en compétition pour l'utilisation d'une ressource commune: le fichier. Cette ressource possède donc N points d'accès relativement aux lecteurs et un point d'accès unique vis-à-vis des rédacteurs. Pour cela on définit plusieurs stratégies d'utilisation possibles du fichier.

a) Priorité aux lecteurs

Un rédacteur ne peut obtenir l'accès au fichier que lorsque aucun lecteur n'ait manifesté le besoin d'y accéder.

Une Solution possible

Soient n_{lect} et n_{red} le nombre respectif de lecteurs et rédacteurs utilisant une procédure d'accès au fichier. Le maintien de la cohérence des informations du fichier impose de définir les contraintes suivantes:

Pour pouvoir utiliser le fichier en lecture on doit satisfaire la condition suivante: $n_{red} = 0$ i.e: aucun rédacteur n'est entrain d'écrire.

La relation suivante doit être vérifiée pour pouvoir accéder au fichier en écriture: ($n_{red} = 0$) et ($n_{lect} = 0$). Les accès au fichier sont soumis au respect du protocole suivant:

Processus Lecteur:

debut

< demande de lecture >;

< lecture >;

< fin de lecture >;

fin

Processus Rédacteur:

debut

< demande d'écriture >;

< écriture >;

< fin d'écriture >;

fin

Il est à remarquer que: l'accès au fichier en écriture est conditionné par la satisfaction de la relation précédente, donc exclusif. On peut penser à mettre l'opération d'écriture dans une section critique et la cohérence des informations du fichier est ainsi garantie. Toutefois, cette solution ne peut convenir car cela contredirait le principe selon lequel une section critique doit être aussi courte que possible (nécessite le moins possible de temps d'exécution)!

b) Priorité relative des lecteurs sur les rédacteurs

Priorité des lecteurs sur les rédacteurs si et seulement si un lecteur occupe déjà le fichier. Quand il n'y a aucun lecteur en lecture, il y a égalité des priorités. Par contre dès que un lecteur obtienne l'accès, tous les autres lecteurs peuvent y accéder et ce, quelque soit le nombre de rédacteurs en attente (possibilité de monopolisation du fichier par les lecteurs pouvant ainsi créer la famine des rédacteurs).

La solution est identique qu'en a) sans mutexred.

c) Priorité absolue des rédacteurs sur les lecteurs

Dès qu'un rédacteur réclame l'accès au fichier, il doit l'obtenir à la fin de l'utilisation du processus en cours. Il y a attente des lecteurs arrivant après la demande d'un rédacteur. Il y a possibilité de privation des lecteurs provoquée par la coalition des rédacteurs.

Questions :

Résoudre le problème dans chacun des cas à l'aide des:

1. Sémaphores
2. Régions critiques
3. Moniteurs

Exercice 3 : Modèle du rendez-vous

On appelle point de rendez-vous (un point de synchronisation) un point dans le programme d'un processus P_i , $i \in [1..N]$, que ce dernier ne peut franchir avant que son (ou ses) partenaire(s) P_j , $j \neq i$, $j \in [1..N]$ aient atteint les leurs.

Cette notion a été introduite par Hoare dans le langage CSP [Hoa78]. Le mécanisme de rendez-vous revêt un caractère important dans la mesure où il convient aisément à la fois dans un environnement centralisé et réparti.

En effet, l'absence de partage de mémoire commune en réparti, impose une communication interprocessus à travers messages; ce qui permet, grâce au rdv, une évolution d'exécution cadencée au rythme d'émission de messages et attente de réponse (synchronisation entre l'émission et la réception de messages). Cette synchronisation est difficilement réalisable à l'aide d'outils tels que les sémaphores ou les moniteurs où les réveils sont assujettis aux signaux externes.

Question: Ecrire un algorithme réalisant la coordination de N processus sous forme d'un rendez-vous, en utilisant:

1. Sémaphores
2. régions critiques
3. Moniteurs