Université Toulouse III – Paul Sabatier
118 route de Narbonne
31062 Toulouse cedex 9

Travaux dirigés – n°1 à 2 – Moniteurs de Hoare

Quelques rappels

Un moniteur de Hoare est un objet évolué de synchronisation encapsulant :

- des opérations privées
- des opérations publiques qu'utiliseront des processus pour accéder à la ressource critique protégée par M
- des variables d'état, pour exprimer des prédicats notamment
- des variables « conditions » pour exprimer les blocages/déblocages des processus

Propriété fondamentale :

Les **opérations** du moniteur sont **exécutées en exclusion mutuelle.**

Si un processus exécute une opération, aucun autre processus ne peut exécuter une autre opération en même temps (premier niveau de blocage possible pour un processus : avoir l'accès à une opération du moniteur).

Sur une **variable condition** C, trois opérations sont possibles :

- **wait(C)**Le processus appelant est bloqué dans le file d'attente associée à C (deuxième niveau de blocage possible pour un processus : accéder à la ressource critique).
- **signal(C)** Le premier processus, s'il existe, de la file d'attente associée à C est débloqué (celui qui réveille perd l'accès au moniteur). Si aucun processus n'était bloqué, cette opération n'a aucun effet (pas de mémorisation comme avec l'opération V des sémaphores).
- **vide(C)**Savoir si la file d'attente associée à C est vide ou non.

Exercice 1 – Modèle des producteurs/consommateurs

On considère le modèle des producteurs-consommateurs dans lequel deux familles de processus accèdent à un buffer partagé pour y déposer (processus Producteurs) ou retirer des messages (processus Consommateurs). Le buffer est géré de manière circulaire. Les retraits s'effectuent dans l'ordre des dépôts.

On se propose d'écrire un moniteur de Hoare gérant l'accès à la ressource commune selon les différentes variantes énoncées ci-après.

La spécification du moniteur est la suivante :

```
Moniteur Prod_Conso {
  void deposer(. . . );    -- utilisée par un processus producteur
  void retirer(. . . );    -- utilisée par un processus consommateur
end Prod_Conso ;
```

Variante 1 – Buffer de N cases, messages d'un type unique

C'est la variante de base, dans laquelle la capacité du buffer est de N messages et la politique appliquée est celle décrite plus haut.

Variante 2 – Message de deux types, dépôts alternés

Dans cette variante, les messages considérés peuvent être de deux types (par exemple, noir/blanc ou recto/verso…). Un producteur dépose des messages d'un certain type. Les dépôts des messages se font de manière alternée. Les retraits se font selon la même politique que précédemment.

Variante 3 – Messages de deux types, choix du type de message retiré

Dans cette variante, on a toujours des producteurs de deux types mais les dépôts ne sont plus forcément alternés. On a aussi des consommateurs de deux types et un consommateur spécifie le type du message qu'il désire retirer. Les retraits se font toujours dans l'ordre des dépôts.

Questions

Pour chacune des variantes :

- Préciser la spécification du moniteur.
- Préciser les conditions de blocage et de réveil d'un processus producteur et d'un processus consommateur.
- En déduire les variables d'état et les variables « condition » gérées dans le moniteur.
- Donner le code du moniteur.