

Lecteurs-Rédacteurs avec moniteur de Hoare

a) Priorité **ABSOLUE** des lecteurs sur les rédacteurs (priorité aux lecteurs):

LectRed_PAL.Monitor

Begin

Var nblect,nbrect : integer;

lect, red : condition;

Procedure DebutLire()

Begin

 Nblect++;

 if(nbrect>0)

 begin

 wait(lect);

 signal(lect); //reveil en cascade des lecteurs
 attendant une fin d'écriture

 end

End;

Procédure FinLire()

Begin

 Nblect--;

 if (nblect=0) signal(red); //reveille
 éventuellement un red car aucun lecteur en attente

End;

Procedure DebutEcrire()

Begin

 if(nbrect>0 OR nblect >0) wait(red);
 nbrect++;

End;

Procédure FinEcrire()

Begin

 nbrect--;

 if (nblect>0) signal(lect)

 else signal(red); //reveille un rédacteur si un en
 attente sinon le signal est perdu

End;

%initialisation %

nblect=nbrect=0 ;

END.LectRed_PAL ;

b) Priorité **Relative** aux lecteurs (priorité égale):

Le problème de la priorité relative apparait dans la situation suivante : un rédacteur en cours avec lecteurs en attente, puis rédacteurs puis lecteurs. Normalement, le premier lecteur qui passe reveille ts les lecteurs en attente même s'ils sont arrivés après des rédacteurs, et si on les met tous sur la même file d'attente, ce reveil sera impossible. Il faut

donc mettre le 1er lecteur en attente avec les rédacteurs, pour qu'ils soient de même priorité, et mettre les lecteurs suivants en attente sur une autre condition pour pouvoir les réveiller tous par le 1er lecteur réveillé.

LectRed_PRL.Monitor

Begin

Var nblect,nblect : integer;

lect, LecRed: condition;

```

Procedure DebutLire()
Begin
    Nblecatt ++;
    if(nbred>0)
    begin
        if (nblecatt=1)
            wait(LecRed);
        else
            wait(lect);
        endif
        Nblect ++; Nblecatt --
        signal(lect);//reveil en cascade des lecteurs
        attendant une fin d'écriture
    end
End;

Procédure FinLire ()
Begin
    Nblect --;
    if (nblect=0) signal(LecRed);
End;
    
```

```

Procedure DebutEcrire()
Begin
    if(nbred>0 OR nblect >0)
        wait(LecRed);
    endif
    nbred++;
End;

Procédure FinEcrire()
Begin
    nbred--;
    signal(LecRed);//veille un rédacteur ou le 1er
    lecteur ou se perd si aucun en attente
End;
    
```

%initialisation %

nblect=nbred=0 ;

END.LectRed_PRL ;

b) Priorité aux rédacteurs :

LectRed_PAR.Monitor

Begin

Var nblect,nblect, nbredatt: integer;

lect, red : condition;

//nbredatt permet de bloquer les lecteurs si des rédacteurs sont en attente

Procedure DebutLire()

Begin

if(nbred>0 OR nbredatt>0)
begin
wait(lect);
signal(lect);//reveil en cascade des lecteurs
attendant une fin d'écriture
end
Nblect++;

End;

Procédure FinLire ()

Begin

Nblect--;
if (nblect=0) signal(red)
//soit des red et des lect en attente=> reveiller
red si pas de red en attente=>pas de lect en
attente

End;

Procedure DebutEcrire()

Begin

Nbredatt++;
if(nbred>0 OR nblect >0)
wait(red);
Nbredatt--;
nbred++;

End;

Procédure FinEcrire()

Begin

nbred--;
if (nbredatt>0) signal(red)
else signal(lect);

End;

%initialisation %

nblect=nbred=nbredatt=0 ;

END.LectRed_PAR ;