PRINCIPI DI SISTEMI OPERATIVI

ESERCIZIO del 19 DICEMBRE 2003

Un **centro metereologico** utilizza **S sonde** per rilevare alcuni dati metereologici, le quali periodicamente scrivono i valori rilevati in un vettore di **N** (N<S) elementi. Ogni sonda chiede di scrivere il valore in un solo elemento, ma più sonde possono scrivere in mutua esclusione nello stesso elemento per motivi di ridondanza. I dati del vettore vengono elaborati periodicamente da un **processo elaboratore**, che può iniziare ad eseguire il suo compito solo se nessuna sonda sta scrivendo e se tutti gli elementi sono stati aggiornati dopo l'ultima elaborazione. Dopo aver eseguito l'elaborazione, le sonde possonno aggiornare gli elementi.

Si implementi una soluzione usando il costrutto monitor per modellare il **centro metereologico** e i processi per modellare le **sonde** e **il processo elaboratore** e si descriva la sincronizzazione tra i processi. Nel rispettare i vincoli richiesti, si cerchi di massimizzare l'utilizzo delle risorse. Si discuta se la soluzione proposta può presentare starvation e in caso positivo per quali processi, e si propongano modifiche e/o aggiunte per evitare starvation.

```
program CentroMetereologico

const     S = ...; { numero di sonde }
const     N = ...; { numero di elementi }

type sonda = process (e: 1..N)
begin
      repeat
            c.inizio_scrittura(e);
            vettore[e] := < valore >
            c.fine_scrittura(e);
            <attendi >
      until false
end

type elaboratore = process
begin
      repeat
            c.inizio_elaborazione;
            <elabora>
            c.fine_elaborazione;
            <attendi >
      until false
end

type centro = monitor

{ variabili del monitor }
var  elemocc: array[1..N] of boolean;
      { elementi occupati }
      elaborazione: boolean;
      { elaborazione in corso }
      numelocc: integer;
      { numero di elementi occupati }
      elemagg: array[1..N] of boolean;
      { elementi aggiornati }
```

```
        codaS : array [1..N] of condition;
        { coda su cui sospendere le sonde }
        codaE : condition;
        { coda su cui sospendere l'elaboratore }

procedure entry inizio_scrittura (e: 1..N)
begin
{ se l'elemento è occupato o l'elaborazione è in corso}
        while elemocc[e] or elaborazione do
                { sospensione }
                codaS[e].wait;
        end
        { acquisizione delle risorse }
        elemocc[e] := true;
        numelocc ++;
end

procedure entry fine_scrittura(e: 1..N)
begin
        { rilascio delle risorse }
        elemocc[e] := false;
        numelocc --;
        { dico che il valore è aggiornato }
        elemagg[e] :=  true;
        { se non ci sono sonde che scrivono }
        if numelocc = 0 then
                { risveglio l'elaboratore }
                codaE.signal;
        { risveglio una sonda nella code dell'elemento liberato }
        codaS[e].signal;
end
```

```
procedure entry inizio_elaborazione
begin
      { se ci sono sonde che stanno scrivendo
            o se non tutti i valori sono aggiornati}
      if (numelocc > 0) or (tutti_aggiornati) then
            { sospensione }
            codaE.wait;
      { acquisizione delle risorse }
      elaborazione := true;
end

procedure entry fine_elaborazione
var i: integer;
begin
      { rilascio delle risorse }
      elaborazione := false;
      for i := 1 to N do
            elemagg[i] := false;

      { risveglio ua sonda per ogni elemento }
      for i := 1 to N do
            codaS[i].signal;
end

function tutti_aggiornati: boolean
{ ritorna vero solo se tutti gli elemagg sono veri }
var i: integer;
     s: boolean;
begin
      s := true;
      for i := 1 to N do
            s := s and elemagg[i];
      tutti_aggiornati := s;
end
```

```
begin { inizializzazione delle variabili }
var i: integer;
      numelocc := 0;
      for i := 1 to N do
      begin
            elemocc[i] := false;
            elemagg[i] := true;
      end;
end

var  c: centro; { il nostro monitor }
      s1, s2, ... : sonda (j);
      e: elaboratore;

begin end.
```

Starvation

Nella soluzione proposta può esserci starvation per l'elaboratore, perché le sonde potrebbero passargli sempre davanti.
Si può risolvere imponendo che una sonda non possa acquisire la risorsa se l'elaboratore è in coda. Oppure si potrebbe usare un contatore per fare eseguire l'elaboratore ogni tot sonde.

Nota

Il contatore numelocc tiene conto degli elementi occupati. Non è essenziale, in quanto sarebbe sufficiente controllare gli elementi elemocc, ma risulta comodo.