Monitores

Ao usar semáforos ou mutexes, um programador precisa identificar explicitamente os pontos de sincronização necessários em seu programa, essa abordagem é eficaz para programas pequenos e problemas de sincronização simples, mas se torna inviável e suscetível a erros em sistemas mais complexos, para evitar esses erros, a a exclusão mútua é obrigatória e uma maneira de se fazer isso é colocar as seções críticas em uma área acessível somente a um processo de cada vez.

A ideia central dos monitores é que em vez de codificar as seções críticas dentro de cada processo, podemos modificá-las como procedimentos (procedure entries) do monitor. Assim, aos dados compartilhados são declarados como parte do monitor e para acessá-los, o processo deve fazer chamada um procedimento deste monitor e assim, o código da seção crítica não é mais duplicado em cada processo.

Sendo assim, um monitor pode ser visto como uma estrutura ou um objeto que contém internamente dados a serem compartilhados e procedimentos para manipular esses dados, que quando declarados dentro do monitor só podem ser acessados através dos procedimentos do monitor, isto é, a única maneira pela qual um processo pode acessar os dados compartilhados é indiretamente, por meio das procedure entries (procedimentos de entrada) que são executadas de forma mutuamente exclusiva. A forma de implementação do monitor já garante a exclusão mútua no acesso às procedure entries.

.

Como Processos Acessam um Monitor?

```
Processo P1
  Begin
  ...
  myMonitor.proced1(...)
  ...
  End
```

Variáveis de Condição

São variáveis especiais utilizadas para provocar bloqueio e desbloqueio de processos, declaradas dentro do monitor e são sempre acessadas por meio de dois comandos especiais:

- Wait (ou delay)
  Faz com que o monitor bloqueie o processo que fez a chamada. O monitor armazena as informações sobre o processo bloqueado em uma estrutura de dados (fila) associada à variável de condição.

- Signal (ou continue)
  Faz com que o monitor reative UM dos processos suspensos na fila associada à variável de condição

O que acontece após um processo P fazer um Signal (condition)?

- Hoare propôs deixar o processo Q recentemente acordado executar, bloqueando o processo P sinalizador. P deve esperar em uma fila pelo término da operação de monitor realizada por Q.
- Brinch Hansen propôs que o processo P concluísse a operação em curso, uma vez que já estava em execução no monitor (i.e., Q deve esperar). Neste caso, a condição lógica pela qual o processo Q estava esperando pode não ser mais verdadeira quando Q for reiniciado.

Produtor-Consumidor com Buffer Circular

```
Monitor buffercircular;
  buffer matriz(0..n) of "coisa";
  i: integer; //índice de remoção
  j: integer; //índice de inserção
  buffcheio: condition;
  buffvazio: condition;
  ocupado: integer;

Entry Procedure Coloca(AlgumDado: coisa)
Begin
  if ocupado = n then wait(buffcheio);
  buffer[j] := AlgumDado; //insere
  j := (j+ 1) MOD n ;
  ocupado:= ocupado + 1;
  signal(bufvazio);
End

Entry Procedure Retira(AlgumDado: coisa)
Begin
  if ocupado = 0 then wait(buffvazio);
  remove AlgumDado de buffer[i];
  i := (i+ 1) MOD n ;
  ocupado:= ocupado - 1;
  signal(buffcheio);
End

Begin
  i := 0; j :=0; ocupado := 0
End
```