Sumário

- Gestão de memória.
- Temporização.
- Semáforos.
- Caixas do correio.
- Gestão de nomes.
- Resumo das primitivas do RMX.

©Luís E. T. Rodrigues, 1996

Gestão memória

- `CREATE$SEGMENT`

  Utilizado para reservar um segmento.

- `DELETE$SEGMENT`

  Utilizado para libertar um segmento.

©Luís E. T. Rodrigues, 1996

Temporização

- Gestão de tempo de espera.
- Funcionalidade:

  Colocar uma tarefa em espera por um determinado período de tempo.

  Incondicionalmente, ou sobre um recurso.

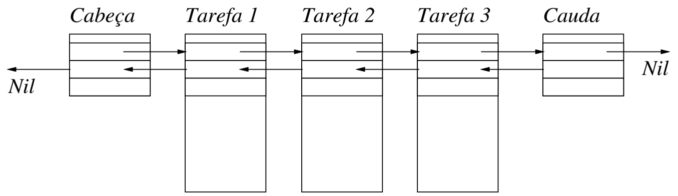
- Concretização:

  Interrupção do relógio.

  Gestor chama uma tarefa que decrementa o tempo de espera.

©Luís E. T. Rodrigues, 1996

Temporização: estruturas de dados



©Luís E. T. Rodrigues, 1996

Temporizadores: concretização

- Gestão da fila de temporização:

  A cada interrupção tira-se uma unidade a tarefa que está à cabeça da fila.

  Quando o tempo chega a zero: retira a tarefa da lista de espera: põe a tarefa na lista de executáveis; se necessário, comuta as tarefas.

- `SLEEP`: coloca a tarefa em espera pelo tempo especificado.

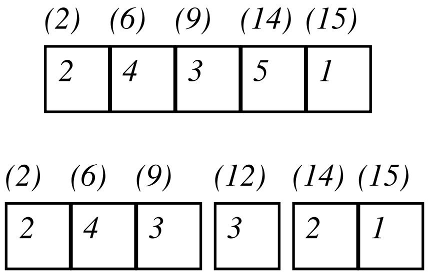
  Tira a tarefa da lista de executáveis; insere na lista de espera; comuta as tarefas.

- Primitivas de sincronização temporizadas.

©Luís E. T. Rodrigues, 1996

Temporizadores: inserção na lista de espera

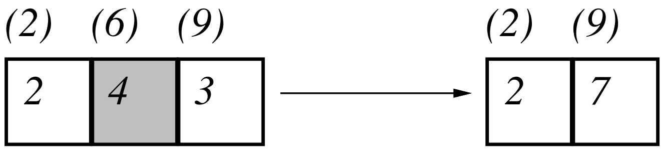
Atrasos diferenciais: só é necessário decrementar na cabeça da fila.



©Luís E. T. Rodrigues, 1996

Temporizadores: remoção da lista de espera

Ao remover é necessário repôr as diferenças.



©Luís E. T. Rodrigues, 1996

Semáforos

- Mecanismo de sincronização.
- Neste núcleo suportam:

  Número de unidades configurável.

  Espera FIFO e espera por prioridades.

  Permite especificar um tempo máximo de espera.

- Primitivas:

  `CREATE$SEMAPHORE`: *criar;* `SEND$UNITS`: *assinalar;* `RECEIVE$UNITS`: *esperar.*

©Luís E. T. Rodrigues, 1996

Semáforos: concretização do assinalar

- Actualiza as unidades.
- Move para a lista de `Ready` todas as tarefas que forem libertadas.
- Realiza a comutação de tarefas se necessário.

©Luís E. T. Rodrigues, 1996

Semáforos: concretização do esperar

- Actualiza as unidades do semáforo se houver suficientes.
- Caso contrário põe a tarefa em espera (retirando-a de `Ready`).
- Se um atraso tiver sido especificado, a tarefa é também inserida na lista de `Delay`.

  Deste modo, uma tarefa pode estar nas duas filas.

©Luís E. T. Rodrigues, 1996

Regiões

- Semáforos binários optimizados para exclusão mútua.

  Processamento mais eficiente que os semáforos.

  Espera FIFO.

- Primitivas:

  `CREATE$REGION`*: criar.*

  `RECEIVE$CONTROL`*: esperar.*

  `SEND$CONTROL`*: assinalar.*

  `ACCEPT$CONTROL`*: testa.*

©Luís E. T. Rodrigues, 1996

Caixas do correio

- Permitem a sincronização e troca de segmentos entre tarefas.

  Também designadas por semáforos com mensagem.

- Concretizadas através de:

  Listas de tarefas bloqueadas esperando segmentos.

  Listas de segmentos ainda não recebidos.

©Luís E. T. Rodrigues, 1996

Caixas do correio: primitivas

- `SEND$MESSAGE (caixa, segmento, caixaResposta, &erro)`

  Permite indicar qual a caixa de correio do cliente.

- `RECEIVE$MESSAGE (caixa, tempoMaximo, &segmento, &erro)`

©Luís E. T. Rodrigues, 1996

Gestão de nomes (só no RMX)

- Serviço que associa identificadores a cadeias de caracteres.

  Permite armazenar e procurar associações.

- Primitivas:

  `CATALOG$OBJECT (, id, "PRINTER", ).`

  `idprinter = LOOKUP$OBJECT (, "PRINTER", ).`

©Luís E. T. Rodrigues, 1996

Resumo das primitivas do RMX

- Gestão de tarefas

  createTask, deleteTask, setPriority, getPriority, getTaskToken.

- Gestão de memória:

  createSegment, deleteSegment, getSize.

- Suspensão:

  suspendTask, resumeTask, sleep.

©Luís E. T. Rodrigues, 1996

Resumo das primitivas do RMX (continuação)

- Semáforos:

  createSemaphore, deleteSemaphore, sendUnits, receiveUnits.

- Regiões:

  createRegion, deleteRegion, acceptControl, sendControl, receiveControl.

- Caixas do correio:

  createMailbox, deleteMailbox, sendMessage, receiveMessage.

©Luís E. T. Rodrigues, 1996

Resumo das primitivas do RMX (continuação)

- Filas:

  createQueue, deleteQueue, sendQueue, receiveQueue.

- Gestão das interrupções:

  setInterrupt, waitInterrupt, enterInterrupt, exitInterrupt, signalInterrupt.

©Luís E. T. Rodrigues, 1996

Resumo

- Gestão das filas de espera.
- Utilização das primitivas.

©Luís E. T. Rodrigues, 1996