

$$ID == \{0, 1, 2\}$$

$$ConnID == \{0, 1, 2\}$$

$$ValidIDs == \mathbb{F} ID$$

$$Response ::= success \mid fail \mid full$$

$$\mathbf{channel} link, return : ID \times Response$$

$$\mathbf{process} PoolSpec \hat{=} \mathbf{begin}$$

$$ReqGranted \hat{=} i : ID \bullet call.i \rightarrow link.i?r : (r \neq full) \rightarrow return.i.r \rightarrow PoolIntf(i)$$

$$ReqDenied \hat{=} i : ID \bullet call.i \rightarrow return.i.full \rightarrow PoolIntf(i)$$

$$PoolIntf \hat{=} i : ID \bullet ReqGranted(i) \sqcap ReqDenied(i)$$

$$\bullet \parallel id : ValidIDs \bullet PoolIntf(id)$$

$$\mathbf{end}$$

$$\mid \quad maxreq : \mathbb{N} \mid maxreq = \# ValidIDs$$

$$\mid \quad maxconn : \mathbb{N}; poolsize, queuesize : \mathbb{N} \mid maxconn = \# ValidConnIDs \wedge poolsize \leq maxconn \wedge queuesize + poolsize \leq maxreq$$

$$\mathbf{channel} enter, exit, reject : ID$$

$$\mathbf{channelset} Internals \hat{=} \{enter, exit, reject\}$$

$$\mathbf{channelset} Shared \hat{=} Internals \cup \{link\}$$

process *PoolDesign* $\hat{=}$ **begin**

state *State* $\hat{=}$ [*active* : $\mathbb{F} ID$; *queue* : seq *ID*

 | *active* \cap ran *queue* = $\emptyset \wedge \#active \leq maxconn \wedge \#queue \leq queuesize$
 $\wedge queue \neq \langle \rangle \Rightarrow \#active = maxconn$]

Init $\hat{=}$ [*State'* | *active'* = $\emptyset \wedge queue' = \langle \rangle$]

Supervisor $\hat{=}$ $\mu X \bullet (SystemCall \sqcap TransactionSystem); X$

SystemCall $\hat{=}$ *EnterActive* \sqcap *QueueOrReject*

EnterActive $\hat{=}$ $\langle \#active < maxconn \rangle \& enter?t \rightarrow AddToActive(t)$

AddToActive $\hat{=}$ [$\Delta State$; $t? : ID$ | *active* = *active* $\cup t? \wedge queue = queue$]

QueueOrReject $\hat{=}$ $\langle \#active = maxconn \rangle \& (EnterQueue \sqcap Reject)$

EnterQueue $\hat{=}$ $\langle \#queue \neq queuesize \rangle \& enter?t \rightarrow AddToQueue(t)$

AddToQueue $\hat{=}$ $t : ID \bullet queue := queue \hat{\ } \langle t \rangle$

Reject $\hat{=}$ $\langle \#queue = queuesize \rangle \& reject?t \rightarrow Skip$

TransactionSystem $\hat{=}$ *Linking* \sqcap *Managing*

Linking $\hat{=}$ $link?t : (t \in active)?r : (r \neq full) \rightarrow Skip$

Managing $\hat{=}$ $exit?t : (t \in active) \rightarrow (UpdateActive \vee UpdateAll)$

UpdateActive $\hat{=}$ [$\Delta State$; $t? : ID$ | *queue* = $\langle \rangle$

$\wedge active' = active \setminus \{t\} \wedge queue' = \langle \rangle$]

UpdateAll $\hat{=}$ [$\Delta State$; $t? : ID$ | *queue* $\neq \langle \rangle$

$\wedge active' = active \setminus \{t\} \cup \{head\ queue\} \wedge queue' = tail\ queue$]

Threads $\hat{=}$ $\parallel\parallel\parallel tid : ValidIDs \bullet Thread(tid)$

Thread $\hat{=}$ $(t : ID) \bullet call.t \rightarrow (Accept(t) \sqcap TReject(t)); Thread(t)$

Accept $\hat{=}$ $(t : ID) \bullet enter.t \rightarrow link.t?r : (r \neq full) \rightarrow exit.t \rightarrow return.t.r \rightarrow Skip$

PoolAssembly $\hat{=}$

$(Threads \parallel \{ \} \mid Shared \mid \{active, queue\}) \parallel Supervisor) \setminus Internals$

$\bullet (Init; PoolAssembly)$

end

REFINANDO PARA TRADUCAO EM JCIRCUS

```

ID == {0, 1, 2}
ValidIDs == V0 | V1 | V2 | V01 | V12 | V02 | V012
    /* O antigo era um  $\mathbb{F}1$  de ID. Refinamos para o conjunto 0, 1, 2, gerando V0, ..., V012 */
channel call : ID
Response ::= success | fail | full
channel link, return : ID  $\times$  Response
process PoolSpec  $\hat{=}$  begin
    ReqGranted  $\hat{=}$   $i : ID \bullet \text{call}.i \rightarrow \text{link}.i?r \rightarrow \text{return}.i.r \rightarrow \text{PoolIntf}(i)$ 
    PoolIntf  $\hat{=}$   $i : ID \bullet \text{ReqGranted} \sqcap \text{ReqDenied}(i)$ 
    • |||  $id : \text{ValidIDs} \bullet \text{PoolIntf}(id)$ 
end

```

```

ConnID == {0, 1, 2}
ValidConnIDs == C0 | C1 | C2 | C01 | C12 | C02 | C012
    /* O antigo era um  $\mathbb{F}1$  de ConnID. Refinamos para o conjunto 0, 1, 2, gerando C0, ..., C012 */

```

```

| maxconn :  $\mathbb{N}$ ; poolsize, queuesize :  $\mathbb{N}$  | maxconn = 3

```

```

channel enter, exit, reject : ID
channelset Internals  $\hat{=}$  {enter, exit, reject}
channelset Shared  $\hat{=}$  Internals  $\cup$  {link}

```

```

process PoolDesign  $\hat{=}$  begin
  state State  $\hat{=}$  [active :  $\mathbb{N}$ ; queue :  $\mathbb{N}$ 
     $\wedge$  queue  $\neq \langle \rangle \Rightarrow \#active = maxconn$ ]
  Init  $\hat{=}$  active := 0; queue := 0
  Supervisor  $\hat{=}$   $\mu X \bullet (SystemCall \sqcap TransactionSystem); X$ 
  SystemCall  $\hat{=}$  EnterActive  $\sqcap$  QueueOrReject
  EnterActive  $\hat{=}$   $\langle \#2 < maxconn \rangle \& enter?t \rightarrow AddToActive(t)$ 
  AddToActive  $\hat{=}$   $t : ID \bullet active := active \cup t$ 
  QueueOrReject  $\hat{=}$   $\langle \#active = maxconn \rangle \& (EnterQueue \sqcap Reject)$ 
  EnterQueue  $\hat{=}$   $\langle \#queue \neq queuesize \rangle \& enter?t \rightarrow AddToQueue(t)$ 
  AddToQueue  $\hat{=}$   $t : ID \bullet queue := queue + 1$ 
  Reject  $\hat{=}$   $\langle queue = queuesize \rangle \& reject?t \rightarrow Skip$ 
  TransactionSystem  $\hat{=}$  Linking  $\sqcap$  Managing
  Linking  $\hat{=}$  link?t?r  $\rightarrow Skip$ 
  Managing  $\hat{=}$  exit?t  $\rightarrow (UpdateActive \sqcap UpdateAll)$ 
  UpdateAll  $\hat{=}$   $t : ID \bullet \langle queue \neq 0 \rangle \&$ 
     $active := active - 1; queue := queue - 1$ 
    / * head e tail nao sao traduziveis, pois circus nao traduz tipos de sequencias
  Threads  $\hat{=}$   $\parallel tid : ValidIDs \bullet Thread(tid)$ 
  Thread  $\hat{=}$   $(t : ID) \bullet call.t \rightarrow (Accept(t) \sqcap TReject(t)); Thread(t)$ 
  Accept  $\hat{=}$   $(t : ID) \bullet enter.t \rightarrow link.t?r : (r \neq full) \rightarrow exit.t \rightarrow return.t.r \rightarrow Skip$ 
  PoolAssembly  $\hat{=}$ 
     $(Threads \parallel \{ \} \mid Shared \mid \{active, queue\} \parallel Supervisor) \setminus Internals$ 
   $\bullet (Init; PoolAssembly)$ 
end

```