producerConsumer

Tight Rope v0.75

4th October 2016

1 ID Files

1.1 MissionIds

section *MissionIds* **parents** *scj_prelude*, *MissionId*

$$PCMissionMID : MissionID$$

$$distinct \langle nullMissionId, PCMissionMID \rangle$$

1.2 SchedulablesIds

section $\mathit{SchedulableIds}$ **parents** $\mathit{scj\_prelude}, \mathit{SchedulableId}$

$$\begin{array}{|l}
\mathit{PCMissionSequencerSID} : \mathit{SchedulableID} \\
\mathit{ProducerSID} : \mathit{SchedulableID} \\
\mathit{ConsumerSID} : \mathit{SchedulableID} \\
\hline
\mathit{distinct}\langle \mathit{nullSequencerId}, \mathit{nullSchedulableId}, \mathit{PCMissionSequencerSID}, \\
\mathit{ProducerSID}, \mathit{ConsumerSID} \rangle
\end{array}$$

1.3 Non-Paradigm Objects

section $BufferApp$ **parents** $scj\_prelude, SchedulableId, SchedulableIds, SafeletChan, MethodCallBindingChannels \quad , O$

process $BufferApp \mathrel{\widehat=}$ **begin**

$$\begin{array}{|l} \mathbf{state}\ State \\ \hline buffer : \mathbb{Z} \\ \hline \end{array}$$

state $State$

$$\begin{array}{|l} \mathbf{initial}\ Init \\ \hline State~' \\ \hline buffer' = 0 \\ \hline \end{array}$$

$$bufferEmptyMeth \mathrel{\widehat=} \mathbf{var}\ ret : \mathbb{B} \bullet \left(\begin{array}{l} bufferEmptyCall \,.\, \longrightarrow \\ \left(\begin{array}{l} \mathbf{if}\ (buffer = 0) \longrightarrow \\ \qquad ret := \mathbf{True} \\ [\!]\ \neg\ (buffer = 0) \longrightarrow \\ \qquad ret := \mathbf{False} \\ \mathbf{fi} \end{array}\right) ; \\ bufferEmptyRet \,.\, !\, ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$\begin{array}{l} writeSyncMeth \mathrel{\widehat=} \\ \left(\begin{array}{l} writeCall \,.\, ?\, thread\, ?\, update \longrightarrow \\ \left(\begin{array}{l} startSyncMeth \,.\, BufferOID \,.\, thread \longrightarrow \\ lockAcquired \,.\, BufferOID \,.\, thread \longrightarrow \\ \left(\begin{array}{l} \mathbf{var}\ bufferEmpty : \mathbb{B} \bullet bufferEmpty := bufferEmpty() \,; \\ \mu X \bullet \\ \left(\begin{array}{l} \mathbf{var}\ loopVar : \mathbb{B} \bullet loopVar := \ (\neg\ bufferEmpty = \mathbf{True}); \\ \mathbf{if}\ (loopVar = \mathbf{True}) \longrightarrow \\ \qquad \left(\begin{array}{l} waitCall \,.\, BufferOID \,.\, thread \longrightarrow \\ waitRet \,.\, BufferOID \,.\, thread \longrightarrow \\ \mathbf{Skip}; \\ bufferEmpty := bufferEmpty() \end{array}\right) ; \ X \\ [\!]\ (loopVar = \mathbf{False}) \longrightarrow \mathbf{Skip} \\ \mathbf{fi} \end{array}\right) \\ ; \\ this \,.\, buffer := update; \\ notify \,.\, BufferOID \,!\, thread \longrightarrow \\ \mathbf{Skip} \end{array}\right) ; \\ endSyncMeth \,.\, BufferOID \,.\, thread \longrightarrow \\ writeRet \,.\, .\, thread \longrightarrow \\ \mathbf{Skip} \end{array}\right) \end{array}\right) \end{array}$$

$$
\begin{array}{l}
readSyncMeth \mathrel{\widehat{=}} \mathbf{var}\ ret : \mathbb{Z} \bullet \\
\left(\begin{array}{l}
readCall\,.\,?\,thread \longrightarrow \\
\left(\begin{array}{l}
startSyncMeth\,.\,BufferOID\,.\,thread \longrightarrow \\
lockAcquired\,.\,BufferOID\,.\,thread \longrightarrow \\
\left(\begin{array}{l}
\mathbf{var}\ bufferEmpty : \mathbb{B} \bullet bufferEmpty := bufferEmpty()\ ; \\
\mu X \bullet \\
\left(\begin{array}{l}
\mathbf{var}\ loopVar : \mathbb{B} \bullet loopVar :=\ bufferEmpty; \\
\mathbf{if}\ (loopVar = \mathbf{True}) \longrightarrow \\
\qquad ;\ X \\
[\!]\ (loopVar = \mathbf{False}) \longrightarrow \mathbf{Skip} \\
\mathbf{fi}
\end{array} \right) \\
; \\
\left(\begin{array}{l}
waitCall\,.\,BufferOID\,.\,thread \longrightarrow \\
waitRet\,.\,BufferOID\,.\,thread \longrightarrow \\
\mathbf{Skip}; \\
bufferEmpty := bufferEmpty()
\end{array} \right) ; \\
\mathbf{var}\ out : \mathbb{Z} \bullet out := buffer\ ; \\
this\,.\,buffer := 0; \\
notify\,.\,BufferOID\,!\,thread \longrightarrow \\
\mathbf{Skip}; \\
ret := out
\end{array} \right) ; \\
endSyncMeth\,.\,BufferOID\,.\,thread \longrightarrow \\
readRet\,.\,.\,thread\,!\,ret \longrightarrow \\
\mathbf{Skip}
\end{array} \right)
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
Methods \mathrel{\widehat{=}} \\
\left(\begin{array}{l}
GetSequencer \\
\Box \\
InitializeApplication \\
\Box \\
bufferEmptyMeth \\
\Box \\
writeSyncMeth \\
\Box \\
readSyncMeth
\end{array} \right) ;\ Methods
\end{array}
$$

$$
\bullet\ (Methods) \mathbin{\triangle} (end\_safelet\_app \longrightarrow \mathbf{Skip})
$$

end

$\mathbf{section}\ BufferMethChan\ \mathbf{parents}\ scj\_prelude, GlobalTypes, MissionId, SchedulableId$

$bufferID : NonParadigmID$

$\mathbf{channel}\ bufferEmptyCall : NonParadigmID$
$\mathbf{channel}\ bufferEmptyRet : NonParadigmID \times \mathbb{B}$

$\mathbf{channel}\ writeCall : NonParadigmID \times ThreadID \times \mathbb{Z}$
$\mathbf{channel}\ writeRet : NonParadigmID \times ThreadID$

$\mathbf{channel}\ readCall : NonParadigmID \times ThreadID$
$\mathbf{channel}\ readRet : NonParadigmID \times ThreadID \times \mathbb{Z}$

1.4 ThreadIds

$\mathbf{section}\ ThreadIds\ \mathbf{parents}\ scj\_prelude, GlobalTypes$

$$\begin{array}{|l}
SafeletTId : ThreadID \\
nullThreadId : ThreadID \\
ProducerTID : ThreadID \\
ConsumerTID : ThreadID \\
\hline
distinct \langle SafeletTId, nullThreadId, ProducerTID, ConsumerTID \rangle
\end{array}$$

1.5 ObjectIds

section *ObjectIds* **parents** *scj_prelude*, *GlobalTypes*

$$\begin{array}{|l}
\mathit{PCMissionOID} : \mathit{ObjectID} \\
\mathit{BufferOID} : \mathit{ObjectId} \\
\hline
\mathit{distinct} \langle \mathit{PCMissionOID}, \mathit{BufferOID} \rangle
\end{array}$$

2 Network

2.1 Network Channel Sets

$$\begin{array}{l}
\mathbf{section}\ \mathit{NetworkChannels}\ \mathbf{parents}\ \mathit{scj\_prelude}, \mathit{MissionId}, \mathit{MissionIds},\\
\quad \mathit{SchedulableId}, \mathit{SchedulableIds}, \mathit{MissionChan}, \mathit{TopLevelMissionSequencerFWChan},\\
\quad \mathit{FrameworkChan}, \mathit{SafeletChan}, \mathit{AperiodicEventHandlerChan}, \mathit{ManagedThreadChan},\\
\quad \mathit{OneShotEventHandlerChan}, \mathit{PeriodicEventHandlerChan}, \mathit{MissionSequencerMethChan}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{TerminateSync} ==\\
\quad \{\!| \mathit{schedulables\_terminated}, \mathit{schedulables\_stopped}, \mathit{get\_activeSchedulables} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{ControlTierSync} ==\\
\quad \{\!| \mathit{start\_toplevel\_sequencer}, \mathit{done\_toplevel\_sequencer}, \mathit{done\_safeletFW} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{TierSync} ==\\
\quad \{\!| \mathit{start\_mission} . \mathit{PCMission}, \mathit{done\_mission} . \mathit{PCMission},\\
\quad \mathit{done\_safeletFW}, \mathit{done\_toplevel\_sequencer} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{MissionSync} ==\\
\quad \{\!| \mathit{done\_safeletFW}, \mathit{done\_toplevel\_sequencer}, \mathit{register},\\
\mathit{signalTerminationCall}, \mathit{signalTerminationRet}, \mathit{activate\_schedulables}, \mathit{done\_schedulable},\\
\mathit{cleanupSchedulableCall}, \mathit{cleanupSchedulableRet} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{SchedulablesSync} ==\\
\quad \{\!| \mathit{activate\_schedulables}, \mathit{done\_safeletFW}, \mathit{done\_toplevel\_sequencer} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{ClusterSync} ==\\
\quad \{\!| \mathit{done\_toplevel\_sequencer}, \mathit{done\_safeletFW} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{SafeltAppSync} \widehat{=}\\
\{\!| \mathit{getSequencerCall}, \mathit{getSequencerRet}, \mathit{initializeApplicationCall}, \mathit{initializeApplicationRet}, \mathit{end\_safelet\_app} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{MissionSequencerAppSync} ==\\
\{\!| \mathit{getNextMissionCall}, \mathit{getNextMissionRet}, \mathit{end\_sequencer\_app} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{MissionAppSync} ==\\
\{\!| \mathit{initializeCall}, \mathit{register}, \mathit{initializeRet}, \mathit{cleanupMissionCall}, \mathit{cleanupMissionRet} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{AppSync} ==\\
\quad \bigcup \{\mathit{SafeltAppSync}, \mathit{MissionSequencerAppSync}, \mathit{MissionAppSync},\\
\quad \mathit{MTAppSync}, \mathit{OSEHSync}, \mathit{APEHSync}, \mathit{PEHSync},\\
\quad \{\!| \mathit{getSequencer}, \mathit{end\_mission\_app}, \mathit{end\_managedThread\_app},\\
\quad \mathit{setCeilingPriority}, \mathit{requestTerminationCall}, \mathit{requestTerminationRet}, \mathit{terminationPendingCall},\\
\quad \mathit{terminationPendingRet}, \mathit{handleAsyncEventCall}, \mathit{handleAsyncEventRet} |\!\}\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{ThreadSync} ==\\
\quad \{\!| \mathit{raise\_thread\_priority}, \mathit{lower\_thread\_priority}, \mathit{isInterruptedCall}, \mathit{isInterruptedRet}, \mathit{get\_priorityLevel} |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{channelset}\ \mathit{LockingSync} ==\\
\quad \{\!| \mathit{lockAcquired}, \mathit{startSyncMeth}, \mathit{endSyncMeth}, \mathit{waitCall}, \mathit{waitRet}, \mathit{notify}, \mathit{isInterruptedCall}, \mathit{isInterruptedRet},\\
\quad \mathit{interruptedCall}, \mathit{interruptedRet}, \mathit{done\_toplevel\_sequencer}, \mathit{get\_priorityLevel} |\!\}
\end{array}$$

2.2 MethodCallBinder

$\mathbf{section}\ MethodCallBindingChannels\ \mathbf{parents}\ scj\_prelude, GlobalTypes, FrameworkChan, MissionId, MissionIds,$
$\quad SchedulableId, SchedulableIds, ThreadIds$

$\mathbf{channel}\ binder\_readCall : NonParadigmID \times SchedulableID \times ThreadID$
$\mathbf{channel}\ binder\_readRet : NonParadigmID \times SchedulableID \times ThreadID \times \mathbb{Z}$

$readLocs == \{ bufferID \}$
$readCallers == \{ ConsumerSID \}$

$\mathbf{channel}\ binder\_terminationPendingCall : \times SchedulableID$
$\mathbf{channel}\ binder\_terminationPendingRet : \times SchedulableID \times boolean$

$terminationPendingLocs == \{ PCMissionMID \}$
$terminationPendingCallers == \{ ProducerSID, ConsumerSID \}$

$\mathbf{channel}\ binder\_writeCall : NonParadigmID \times SchedulableID \times ThreadID \times \mathbb{Z}$
$\mathbf{channel}\ binder\_writeRet : NonParadigmID \times SchedulableID \times ThreadID$

$writeLocs == \{ bufferID \}$
$writeCallers == \{ ProducerSID \}$

$\mathbf{channelset}\ MethodCallBinderSync == \{\!|\ done\_toplevel\_sequencer,$
$binder\_readCall, binder\_readRet,$
$binder\_terminationPendingCall, binder\_terminationPendingRet,$
$binder\_writeCall, binder\_writeRet\ |\!\}$

$\mathbf{section}\ MethodCallBinder\ \mathbf{parents}\ scj\_prelude, MissionId, MissionIds,$
$\quad SchedulableId, SchedulableIds, MethodCallBindingChannels$
$, PCMissionMethChan$

$\mathbf{process}\ MethodCallBinder \mathrel{\widehat=} \mathbf{begin}$

$$
read\_MethodBinder \mathrel{\widehat=} \\
\quad \left(\begin{array}{l}
binder\_readCall\,?\,loc : (loc \in readLocs)\,?\,caller : (caller \in readCallers)\,?\,callingThread \longrightarrow \\
readCall\,.\,loc\,.\,caller\,.\,callingThread \longrightarrow \\
readRet\,.\,loc\,.\,caller\,.\,callingThread\,?\,ret \longrightarrow \\
binder\_readRet\,.\,loc\,.\,caller\,.\,callingThread\,!\,ret \longrightarrow \\
read\_MethodBinder
\end{array}\right)
$$

$$
terminationPending\_MethodBinder \mathrel{\widehat=} \\
\quad \left(\begin{array}{l}
binder\_terminationPendingCall \\
\qquad ?\,loc : (loc \in terminationPendingLocs) \\
\qquad ?\,caller : (caller \in terminationPendingCallers) \longrightarrow \\
terminationPendingCall\,.\,loc\,.\,caller \longrightarrow \\
terminationPendingRet\,.\,loc\,.\,caller\,?\,ret \longrightarrow \\
binder\_terminationPendingRet\,.\,loc\,.\,caller\,!\,ret \longrightarrow \\
terminationPending\_MethodBinder
\end{array}\right)
$$

$$write\_MethodBinder \mathrel{\widehat{=}}$$

$$\left(\begin{array}{l} binder\_writeCall \mathrel{?} loc : (loc \in writeLocs) \mathrel{?} caller : (caller \in writeCallers) \mathrel{?} callingThread \mathrel{?} p1 \longrightarrow \\ writeCall \,.\, loc \,.\, caller \,.\, callingThread \mathrel{!} p1 \longrightarrow \\ writeRet \,.\, loc \,.\, caller \,.\, callingThread \longrightarrow \\ binder\_writeRet \,.\, loc \,.\, caller \,.\, callingThread \longrightarrow \\ write\_MethodBinder \end{array}\right)$$

$$BinderActions \mathrel{\widehat{=}}$$

$$\left(\begin{array}{l} read\_MethodBinder \\ ||| \\ terminationPending\_MethodBinder \\ ||| \\ write\_MethodBinder \end{array}\right)$$

$$\bullet \; BinderActions \mathbin{\triangle} (done\_toplevel\_sequencer \longrightarrow \mathbf{Skip})$$

end

2.3 Locking

section *NetworkLocking* **parents** *scj_prelude*, *GlobalTypes*, *FrameworkChan*, *MissionId*, *MissionIds*, *ThreadIds*, *NetworkChannels*, *ObjectFW*, *ThreadFW*

$$\textbf{process}\ Threads \mathrel{\widehat=} \left(\begin{array}{l} ThreadFW(ProducerTID) \\ ||| \\ ThreadFW(ConsumerTID) \end{array}\right)$$

$$\textbf{process}\ Objects \mathrel{\widehat=} \left(ObjectFW(BufferOID)\right)$$

$$\textbf{process}\ Locking \mathrel{\widehat=} Threads \llbracket ThreadSync \rrbracket Objects$$

2.4 Program

$$\begin{array}{l}
\mathbf{section}\ Program\ \mathbf{parents}\ scj\_prelude, MissionId, MissionIds, \\
\quad SchedulableId, SchedulableIds, MissionChan, SchedulableMethChan, MissionFW, \\
\quad SafeletFW, TopLevelMissionSequencerFW, NetworkChannels, ManagedThreadFW, \\
\quad SchedulableMissionSequencerFW, PeriodicEventHandlerFW, OneShotEventHandlerFW, \\
\quad AperiodicEventHandlerFW, ObjectFW, ThreadFW, \\
\quad PCSafeletApp, PCMissionSequencerApp, PCMissionApp, ProducerApp, ConsumerApp
\end{array}$$

$$\begin{array}{l}
\mathbf{process}\ ControlTier \mathrel{\widehat=} \\
\left(\begin{array}{l}
SafeletFW \\
\quad \llbracket ControlTierSync \rrbracket \\
TopLevelMissionSequencerFW(PCMissionSequencer)
\end{array}\right)
\end{array}$$

$$\begin{array}{l}
\mathbf{process}\ Tier0 \mathrel{\widehat=} \\
\left(\begin{array}{l}
MissionFW(PCMissionID) \\
\quad \llbracket MissionSync \rrbracket \\
\left(\begin{array}{l}
ManagedThreadFW(ProducerID) \\
\quad \llbracket SchedulablesSync \rrbracket \\
ManagedThreadFW(ConsumerID)
\end{array}\right)
\end{array}\right)
\end{array}$$

$$\begin{array}{l}
\mathbf{process}\ Framework \mathrel{\widehat=} \\
\left(\begin{array}{l}
ControlTier \\
\quad \llbracket TierSync \rrbracket \\
(Tier0)
\end{array}\right)
\end{array}$$

$$\begin{array}{l}
\mathbf{process}\ Application \mathrel{\widehat=} \\
\left(\begin{array}{l}
PCSafeletApp \\
\interleave \\
PCMissionSequencerApp \\
\interleave \\
PCMissionApp \\
\interleave \\
ProducerApp(PCMissionID, bufferID) \\
\interleave \\
ConsumerApp(bufferID)
\end{array}\right)
\end{array}$$

$$\begin{array}{l}
\mathbf{process}\ Bound\_Application \mathrel{\widehat=} Application \llbracket MethodCallBinderSync \rrbracket MethodCallBinder \\
\mathbf{process}\ Program \mathrel{\widehat=} (Framework \llbracket AppSync \rrbracket Bound\_Application) \llbracket LockingSync \rrbracket Locking
\end{array}$$

3 Safelet

section $PCSafeletApp$ **parents** $scj\_prelude, SchedulableId, SchedulableIds, SafeletChan, MethodCallBindingChannels$

process $PCSafeletApp \mathrel{\widehat=}$ **begin**

$$InitializeApplication \mathrel{\widehat=} \left(\begin{array}{l} initializeApplicationCall \longrightarrow \\ initializeApplicationRet \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$GetSequencer \mathrel{\widehat=} \left(\begin{array}{l} getSequencerCall \longrightarrow \\ getSequencerRet\,!\,PCMissionSequencerSID \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$Methods \mathrel{\widehat=} \left(\begin{array}{l} GetSequencer \\ \Box \\ InitializeApplication \end{array}\right) ;\ Methods$$

$$\bullet\ (Methods) \mathbin{\triangle} (end\_safelet\_app \longrightarrow \mathbf{Skip})$$

end

4 Top Level Mission Sequencer

section $\mathit{PCMissionSequencerApp}$ **parents** $\mathit{TopLevelMissionSequencerChan}$,
$\mathit{MissionId}, \mathit{MissionIds}, \mathit{SchedulableId}, \mathit{SchedulableIds}, \mathit{PCMissionSequencerClass}, \mathit{MethodCallBindingChannels}$

process $\mathit{PCMissionSequencerApp} \mathrel{\widehat=}$ **begin**

State

$\mathit{this} : \mathbf{ref}\ \mathit{PCMissionSequencerClass}$

state State

Init

State'

$\mathit{this}' = \mathbf{new}\ \mathit{PCMissionSequencerClass}()$

$$\mathit{GetNextMission} \mathrel{\widehat=} \mathbf{var}\ \mathit{ret} : \mathit{MissionID} \bullet \\ \left(\begin{array}{l} \mathit{getNextMissionCall} \,.\, \mathit{PCMissionSequencerSID} \longrightarrow \\ \mathit{ret} := \mathit{this} \,.\, \mathit{getNextMission}(); \\ \mathit{getNextMissionRet} \,.\, \mathit{PCMissionSequencerSID}\, !\, \mathit{ret} \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$\mathit{Methods} \mathrel{\widehat=} \\ \left(\mathit{GetNextMission}\right) ;\ \mathit{Methods}$$

$$\bullet\ (\mathit{Init} ;\ \mathit{Methods}) \mathbin{\triangle} (\mathit{end\_sequencer\_app} \,.\, \mathit{PCMissionSequencerSID} \longrightarrow \mathbf{Skip})$$

end

section $PCMissionSequencerClass$ **parents** $scj\_prelude, SchedulableId, SchedulableIds, SafeletChan$
$, MethodCallBindingChannels, MissionId, MissionIds$

class $PCMissionSequencerClass \mathrel{\widehat=}$ **begin**

state $State$

$returnedMission : \mathbb{B}$

state $State$

initial $Init$

$State~'$

$returnedMission' = \mathbf{False}$

$$\mathbf{protected}\ getNextMission\ \mathrel{\widehat=} \mathbf{var}\ ret : MissionID \bullet$$

$$\left(\begin{array}{l}
\mathbf{if}\ (\lnot\ returnedMission = \mathbf{True}) \longrightarrow \\
\qquad \left(\begin{array}{l} this\,.\,returnedMission := \mathbf{True}; \\ ret := PCMissionMID \end{array}\right) \\
[\!]\ \lnot\ (\lnot\ returnedMission = \mathbf{True}) \longrightarrow \\
\qquad \mathbf{Skip} \\
\mathbf{fi}
\end{array}\right)$$

$\bullet$ **Skip**

end

5 Missions

5.1 PCMission

section *PCMissionApp* **parents** *scj_prelude*, *MissionId*, *MissionIds*,
SchedulableId, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *PCMissionMethChan*
, *PCMissionClass*, *MethodCallBindingChannels*, *ObjectChan*, *ObjectIds*, *ThreadIds*, *ObjectFWChan*, *ObjectIds*

process *PCMissionApp* $\widehat{=}$ **begin**

State

$this : \mathbf{ref}\ Buffer$

state *State*

Init

$State'$

$this' = \mathbf{new}\ Buffer()$

$$InitializePhase \mathrel{\widehat{=}} \left(\begin{array}{l} initializeCall\,.\,PCMissionMID \longrightarrow \\ register\,!\,ProducerSID\,!\,PCMissionMID \longrightarrow \\ register\,!\,ConsumerSID\,!\,PCMissionMID \longrightarrow \\ initializeRet\,.\,PCMissionMID \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$CleanupPhase \mathrel{\widehat{=}} \left(\begin{array}{l} \mathbf{var}\,\mathbb{B} : ret \bullet cleanupMissionCall\,.\,PCMissionMID \longrightarrow \\ cleanupMissionRet\,.\,PCMissionMID\,!\,\mathbf{True} \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$Methods \mathrel{\widehat{=}} \left(\begin{array}{l} InitializePhase \\ \Box \\ CleanupPhase \end{array}\right) \mathrel{;} Methods$$

$$\bullet\ (Init \mathrel{;} Methods) \mathrel{\triangle} (end\_mission\_app\,.\,PCMissionMID \longrightarrow \mathbf{Skip})$$

end

section $PCMissionMethChan$ **parents** $scj\_prelude, GlobalTypes, MissionId, SchedulableId$

channel $bufferEmptyCall : MissionID$
channel $bufferEmptyRet : MissionID \times \mathbb{B}$

channel $cleanUpCall : MissionID$
channel $cleanUpRet : MissionID \times \mathbb{B}$

5.2 Schedulables of PCMission

section *ProducerApp* **parents** *ManagedThreadChan*, *SchedulableId*, *SchedulableIds*, *MethodCallBindingChannels* , *MissionMethChan*, *PCMissionMethChan*, *BufferMethChan*, *ObjectIds*, *ThreadIds*

$$
\begin{array}{l}
\mathbf{process}\ \mathit{ProducerApp} \mathrel{\widehat=} \\
\qquad \mathit{pcMission} : \mathit{MissionID};\ \mathit{buffer} : \mathit{NonParadigmID} \bullet \mathbf{begin}
\end{array}
$$

$$
\begin{array}{l}
\mathit{Run} \mathrel{\widehat=} \\
\left(\begin{array}{l}
\mathit{runCall} \,.\, \mathit{ProducerSID} \longrightarrow \\
\left(\begin{array}{l}
\mu X \bullet \\
\left(\begin{array}{l}
\mathit{binder\_terminationPendingCall} \,.\, \mathit{buffer} \longrightarrow \\
\mathit{binder\_terminationPendingRet} \,.\, \mathit{buffer} \,?\, \mathit{terminationPending} \longrightarrow \\
\mathbf{var}\ \mathit{loopVar} : \mathbb{B} \bullet \mathit{loopVar} := (\neg\ \mathit{terminationPending}); \\
\mathbf{if}\ (\mathit{loopVar} = \mathbf{True}) \longrightarrow \\
\qquad \left(\begin{array}{l}
\mathit{binder\_writeCall} \,.\, \mathit{buffer} \,.\, \mathit{ProducerSID} \,.\, \mathit{ProducerTID} \,!\, i \longrightarrow \\
\mathit{binder\_writeRet} \,.\, \mathit{buffer} \,.\, \mathit{ProducerSID} \,.\, \mathit{ProducerTID} \longrightarrow \\
\mathbf{Skip}; \\
i := i + 1; \\
\mathbf{var}\ \mathit{keepWriting} : \mathbb{B} \bullet \mathit{keepWriting} := \mathit{this} \,.\, i >= 5; \\
\mathbf{if}\ \ \mathit{keepWriting} = \mathbf{True} \longrightarrow \\
\qquad \mathbf{Skip} \\
\Box\ \neg\ \mathit{keepWriting} = \mathbf{True} \longrightarrow \mathit{requestTerminationCall} \,.\, \mathit{pcMission} \,.\, \mathit{ProducerSID} \longrightarrow \\
\mathit{requestTerminationRet} \,.\, \mathit{pcMission} \,.\, \mathit{ProducerSID} \,?\, \mathit{rT} \longrightarrow \\
\mathbf{Skip} \\
\mathbf{fi}
\end{array}\right) ;\ X \\
\Box\ (\mathit{loopVar} = \mathbf{False}) \longrightarrow \mathbf{Skip} \\
\mathbf{fi}
\end{array}\right)
\end{array}\right) ; \\
\mathit{runRet} \,.\, \mathit{ProducerSID} \longrightarrow \\
\mathbf{Skip}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
\mathit{Methods} \mathrel{\widehat=} \\
\big(\mathit{Run}\big) ;\ \ \mathit{Methods}
\end{array}
$$

$$
\bullet\ (\mathit{Methods}) \mathbin{\triangle} (\mathit{end\_managedThread\_app} \,.\, \mathit{ProducerSID} \longrightarrow \mathbf{Skip})
$$

end

section *ConsumerApp* **parents** *ManagedThreadChan*, *SchedulableId*, *SchedulableIds*, *MethodCallBindingChannels*
, *MissionMethChan*, *PCMissionMethChan*, *BufferMethChan*, *ObjectIds*, *ThreadIds*

process *ConsumerApp* $\widehat{=}$
$\mathit{buffer} : \mathit{NonParadigmID} \bullet$ **begin**

$$
\mathit{Run} \widehat{=}
$$

$$
\left(\begin{array}{l}
\mathit{runCall} \,.\, \mathit{ConsumerSID} \longrightarrow \\
\left(\begin{array}{l}
\mu X \bullet \\
\left(\begin{array}{l}
\mathit{binder\_terminationPendingCall} \,.\, \mathit{buffer} \longrightarrow \\
\mathit{binder\_terminationPendingRet} \,.\, \mathit{buffer} \,?\, \mathit{terminationPending} \longrightarrow \\
\mathbf{var}\ \mathit{loopVar} : \mathbb{B} \bullet \mathit{loopVar} := (\neg\ \mathit{terminationPending}); \\
\mathbf{if}\ (\mathit{loopVar} = \mathbf{True}) \longrightarrow \\
\qquad \left(\begin{array}{l}
\mathbf{var}\ \mathit{result} : \mathbb{Z} \bullet \mathit{result} := 999; \\
\mathit{binder\_readCall} \,.\, \mathit{buffer} \,.\, \mathit{ConsumerSID} \,.\, \mathit{ConsumerTID} \longrightarrow \\
\mathit{binder\_readRet} \,.\, \mathit{buffer} \,.\, \mathit{ConsumerSID} \,.\, \mathit{ConsumerTID} \,?\, \mathit{read} \longrightarrow \\
\mathbf{Skip};
\end{array}\right) ;\ X \\
[\!]\ (\mathit{loopVar} = \mathbf{False}) \longrightarrow \mathbf{Skip} \\
\mathbf{fi}
\end{array}\right)
\end{array}\right) ; \\
\mathit{runRet} \,.\, \mathit{ConsumerSID} \longrightarrow \\
\mathbf{Skip}
\end{array}\right)
$$

$$
\mathit{Methods} \widehat{=}
$$

$$
\big(\mathit{Run}\big) ;\ \mathit{Methods}
$$

$$
\bullet\ (\mathit{Methods}) \vartriangle (\mathit{end\_managedThread\_app} \,.\, \mathit{ConsumerSID} \longrightarrow \mathbf{Skip})
$$

end