twoSequentialMissions

Tight Rope v0.88

4th March 2017

1 ID Files

1.1 MissionIds

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

$$MissionAMID : MissionID$$
$$MissionBMID : MissionID$$

$$distinct\langle nullMissionId, MissionAMID, MissionBMID\rangle$$

1.2 SchedulablesIds

section *SchedulableIds* **parents** *scj_prelude*, *SchedulableId*

$$
\begin{array}{|l}
mainSequencerSID : SchedulableID \\
MT1SID : SchedulableID \\
MT2SID : SchedulableID \\
\hline
distinct\langle nullSequencerId, nullSchedulableId, mainSequencerSID, \\
MT1SID, MT2SID\rangle
\end{array}
$$

1.3 Non-Paradigm Objects

1.4 ThreadIds

section *ThreadIds* **parents** *scj_prelude*, *GlobalTypes*

$$SafeletTId : ThreadID$$
$$nullThreadId : ThreadID$$

$$distinct\langle SafeletTId, nullThreadId\rangle$$

1.5 ObjectIds

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

$$\vdash \; distinct\langle\rangle$$

2 Network

2.1 Network Channel Sets

section *NetworkChannels* **parents** *scj_prelude*, *MissionId*, *MissionIds*,
SchedulableId, *SchedulableIds*, *MissionChan*, *TopLevelMissionSequencerFWChan*,
FrameworkChan, *SafeletChan*, *AperiodicEventHandlerChan*, *ManagedThreadChan*,
OneShotEventHandlerChan, *PeriodicEventHandlerChan*, *MissionSequencerMethChan*

channelset *TerminateSync* ==
{| *schedulables_terminated*, *schedulables_stopped*, *get_activeSchedulables* |}

channelset *ControlTierSync* ==
{| *start_toplevel_sequencer*, *done_toplevel_sequencer*, *done_safeletFW* |}

channelset *TierSync* ==
{| *start_mission* . *MissionA*, *done_mission* . *MissionA*,
done_safeletFW, *done_toplevel_sequencer* |}

channelset *TierSync* ==
{| *start_mission* . *MissionB*, *done_mission* . *MissionB*,
done_safeletFW, *done_toplevel_sequencer* |}

channelset *MissionSync* ==
{| *done_safeletFW*, *done_toplevel_sequencer*, *register*,
signalTerminationCall, *signalTerminationRet*, *activate_schedulables*, *done_schedulable*,
cleanupSchedulableCall, *cleanupSchedulableRet* |}

channelset *SchedulablesSync* ==
{| *activate_schedulables*, *done_safeletFW*, *done_toplevel_sequencer* |}

channelset *ClusterSync* ==
{| *done_toplevel_sequencer*, *done_safeletFW* |}

channelset *SafeltAppSync* $\widehat{=}$
{| *getSequencerCall*, *getSequencerRet*, *initializeApplicationCall*, *initializeApplicationRet*, *end_safelet_app* |}

channelset *MissionSequencerAppSync* ==
{| *getNextMissionCall*, *getNextMissionRet*, *end_sequencer_app* |}

channelset *MissionAppSync* ==
{| *initializeCall*, *register*, *initializeRet*, *cleanupMissionCall*, *cleanupMissionRet* |}

channelset *AppSync* ==
$\bigcup\{$*SafeltAppSync*, *MissionSequencerAppSync*, *MissionAppSync*,
MTAppSync, *OSEHSync*, *APEHSync*, *PEHSync*,
{| *getSequencer*, *end_mission_app*, *end_managedThread_app*,
setCeilingPriority, *requestTerminationCall*, *requestTerminationRet*, *terminationPendingCall*,
terminationPendingRet, *handleAsyncEventCall*, *handleAsyncEventRet* |}$\}$

channelset *ThreadSync* ==
{| *raise_thread_priority*, *lower_thread_priority*, *isInterruptedCall*, *isInterruptedRet*, *get_priorityLevel* |}

channelset *LockingSync* ==
{| *lockAcquired*, *startSyncMeth*, *endSyncMeth*, *waitCall*, *waitRet*, *notify*, *isInterruptedCall*, *isInterruptedRet*,
interruptedCall, *interruptedRet*, *done_toplevel_sequencer*, *get_priorityLevel* |}

2.2 Locking

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

process *Threads* $\widehat{=}$
$(\mathbf{Skip})$

process *Objects* $\widehat{=}$
$(\mathbf{Skip})$

process *Locking* $\widehat{=}$ $(Threads \llbracket ThreadSync \rrbracket Objects) \triangle (done\_toplevel\_sequencer \longrightarrow \mathbf{Skip})$

2.3 Program

section *Program* **parents** *scj_prelude*, *MissionId*, *MissionIds*,
SchedulableId, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *MissionFW*,
SafeletFW, *TopLevelMissionSequencerFW*, *NetworkChannels*, *ManagedThreadFW*,
SchedulableMissionSequencerFW, *PeriodicEventHandlerFW*, *OneShotEventHandlerFW*,
AperiodicEventHandlerFW, *ObjectFW*, *ThreadFW*,
MyAppApp, *mainSequencerApp*, *MissionAApp*, $MT1App$, *MissionBApp*, $MT2App$

process *ControlTier* $\widehat{=}$
$$\left(\begin{array}{l} SafeletFW \\ \quad [\![ControlTierSync]\!] \\ TopLevelMissionSequencerFW(mainSequencer) \end{array}\right)$$

process $Tier0$ $\widehat{=}$
$$\left(\begin{array}{l} MissionFW(MissionAID) \\ \quad [\![MissionSync]\!] \\ \big(ManagedThreadFW(MT1ID)\big) \end{array}\right)$$
$$[\![ClusterSync]\!]$$
$$\left(\begin{array}{l} MissionFW(MissionBID) \\ \quad [\![MissionSync]\!] \\ \big(ManagedThreadFW(MT2ID)\big) \end{array}\right)$$

process *Framework* $\widehat{=}$
$$\left(\begin{array}{l} ControlTier \\ \quad [\![TierSync]\!] \\ \big(Tier0\big) \end{array}\right)$$

process *Application* $\widehat{=}$
$$\left(\begin{array}{l} MyAppApp \\ ||| \\ mainSequencerApp \\ ||| \\ MissionAApp \\ ||| \\ MT1App \\ ||| \\ MissionBApp \\ ||| \\ MT2App \end{array}\right)$$

process *Program* $\widehat{=}$ $\big(Framework\ [\![\ AppSync\ ]\!]\ Application\big)\ [\![\ LockingSync\ ]\!]\ Locking$

3 Safelet

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

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

$$InitializeApplication \mathrel{\widehat=} \begin{pmatrix} initializeApplicationCall \longrightarrow \\ initializeApplicationRet \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

$$GetSequencer \mathrel{\widehat=} \begin{pmatrix} getSequencerCall \longrightarrow \\ getSequencerRet\,!\,mainSequencerSID \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

$$Methods \mathrel{\widehat=} \begin{pmatrix} GetSequencer \\ \Box \\ InitializeApplication \end{pmatrix} ;\ Methods$$

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

end

4 Top Level Mission Sequencer

section *mainSequencerApp* **parents** *TopLevelMissionSequencerChan*, *MissionId*, *MissionIds*, *SchedulableId*, *SchedulableIds*, *mainSequencerClass*, *MethodCallBindingChannels*

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

State

$this : \mathbf{ref}\ mainSequencerClass$

state *State*

Init

$State'$

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

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

$Methods \mathrel{\widehat=}$
$\big(GetNextMission\big)\ ;\ Methods$

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

end

section *mainSequencerClass* **parents** *scj_prelude*, *SchedulableId*, *SchedulableIds*, *SafeletChan*
, *MethodCallBindingChannels*, *MissionId*, *MissionIds*

class *mainSequencerClass* $\widehat{=}$ **begin**

state *State*

$$releases : \mathbb{Z}$$

state *State*

initial *Init*

$$State'$$

$$releases' = 0$$

protected *getNextMission* $\widehat{=}$

$$\left(\begin{array}{l}
\textbf{if } (releases = 0) \longrightarrow \\
\quad \left(\begin{array}{l} releases := releases + 1; \\ ret := MissionAMID \end{array}\right) \\
[]\ \neg\ (releases = 0) \longrightarrow \\
\quad \textbf{if } (releases = 1) \longrightarrow \\
\quad \left(\begin{array}{l} releases := releases + 1; \\ ret := MissionBMID \end{array}\right) \\
[]\ \neg\ (releases = 1) \longrightarrow \\
\quad \big(ret := nullMissionId\big) \\
\textbf{fi} \\
\textbf{fi}
\end{array}\right)$$

- **Skip**

end

5 Missions

5.1 MissionA

section *MissionAApp* **parents** *scj_prelude*, *MissionId*, *MissionIds*, *SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *MissionAMethChan*, *MethodCallBindingChannels*

process $MissionAApp \mathrel{\widehat=} \textbf{begin}$

$$InitializePhase \mathrel{\widehat=} \begin{pmatrix} initializeCall\,.\,MissionAMID \longrightarrow \\ register\,!\,MT1SID\,!\,MissionAMID \longrightarrow \\ initializeRet\,.\,MissionAMID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

$$CleanupPhase \mathrel{\widehat=} \begin{pmatrix} cleanupMissionCall\,.\,MissionAMID \longrightarrow \\ cleanupMissionRet\,.\,MissionAMID\,!\,\textbf{True} \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

$$Methods \mathrel{\widehat=} \begin{pmatrix} InitializePhase \\ \Box \\ CleanupPhase \end{pmatrix} ;\ Methods$$

$\bullet\ (Methods) \triangle (end\_mission\_app\,.\,MissionAMID \longrightarrow \textbf{Skip})$

end

5.2 Schedulables of MissionA

section $MT1App$ **parents** $ManagedThreadChan, SchedulableId, SchedulableIds, MethodCallBindingChannels$

process $MT1App \mathrel{\widehat=}$ **begin**

$$Run \mathrel{\widehat=} \left(\begin{array}{l} runCall\,.\,MT1SID \longrightarrow \\ (\mathbf{Skip})\,; \\ runRet\,.\,MT1SID \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$Methods \mathrel{\widehat=} (Run)\,;\ Methods$$

$$\bullet\ (Methods) \triangle (end\_managedThread\_app\,.\,MT1SID \longrightarrow \mathbf{Skip})$$

end

5.3 MissionB

section *MissionBApp* **parents** *scj_prelude*, *MissionId*, *MissionIds*, *SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *MissionBMethChan*, *MethodCallBindingChannels*

process $MissionBApp \mathrel{\widehat=} \textbf{begin}$

$$InitializePhase \mathrel{\widehat=} \begin{pmatrix} initializeCall\,.\,MissionBMID \longrightarrow \\ register\,!\,MT2SID\,!\,MissionBMID \longrightarrow \\ initializeRet\,.\,MissionBMID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

$$CleanupPhase \mathrel{\widehat=} \begin{pmatrix} cleanupMissionCall\,.\,MissionBMID \longrightarrow \\ cleanupMissionRet\,.\,MissionBMID\,!\,\textbf{True} \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

$$Methods \mathrel{\widehat=} \begin{pmatrix} InitializePhase \\ \Box \\ CleanupPhase \end{pmatrix} ;\ Methods$$

$\bullet\ (Methods) \triangle (end\_mission\_app\,.\,MissionBMID \longrightarrow \textbf{Skip})$

end

5.4 Schedulables of MissionB

section $MT2App$ **parents** $ManagedThreadChan, SchedulableId, SchedulableIds, MethodCallBindingChannels$

process $MT2App \mathrel{\widehat=}$ **begin**

$$Run \mathrel{\widehat=} \left(\begin{array}{l} runCall\,.\,MT2SID \longrightarrow \\ (\mathbf{Skip})\,; \\ runRet\,.\,MT2SID \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$Methods \mathrel{\widehat=} (Run)\,;\ Methods$$

$\bullet\ (Methods) \triangle (end\_managedThread\_app\,.\,MT2SID \longrightarrow \mathbf{Skip})$

end