

mission2

Tight Rope v0.88

4th March 2017

1 ID Files

1.1 MissionIds

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

MissionAMID : *MissionID*

distinct $\langle \text{nullMissionId}, \text{MissionAMID} \rangle$

1.2 SchedulablesIds

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

mainSequencerSID : *SchedulableID*

OSEHSID : *SchedulableID*

MTSID : *SchedulableID*

distinct (*nullSequencerId, nullSchedulableId, mainSequencerSID,*
OSEHSID, MTSID)

1.3 Non-Paradigm Objects

1.4 ThreadIds

section *ThreadId* **parents** *scj_prelude, GlobalTypes*

SafeletTid : *ThreadID*
nullThreadId : *ThreadID*

distinct(*SafeletTid*, *nullThreadId*)

1.5 ObjectIds

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

$\text{distinct} \langle \rangle$

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 *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* $\hat{=}$
{*getSequencerCall, getSequencerRet, initializeApplicationCall, initializeApplicationRet, end_safelet_app*}

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

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

channelset *AppSync* ==
{*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 MethodCallBinder

section *MethodCallBindingChannels* **parents** *scj_prelude*, *GlobalTypes*, *FrameworkChan*, *MissionId*, *MissionIds*,
SchedulableId, *SchedulableIds*, *ThreadId*

channel *binder_systemActionCall* : *MissionID* \times *SchedulableID*

channel *binder_systemActionRet* : *MissionID* \times *SchedulableID*

systemActionLocs == {*MissionAMID*}

systemActionCallers == {*MTSID*}

channelset *MethodCallBinderSync* == { *done_toplevel_sequencer*,
binder_systemActionCall, *binder_systemActionRet* }

section *MethodCallBinder* **parents** *scj_prelude*, *MissionId*, *MissionIds*,
SchedulableId, *SchedulableIds*, *MethodCallBindingChannels*
, *MissionAMethChan*

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

systemAction_MethodBinder $\hat{=}$

$$\left(\begin{array}{l} \text{binder_systemActionCall} ? \text{loc} : (\text{loc} \in \text{systemActionLocs}) ? \text{caller} : (\text{caller} \in \text{systemActionCallers}) \longrightarrow \\ \text{systemActionCall} . \text{loc} . \text{caller} \longrightarrow \\ \text{systemActionRet} . \text{loc} . \text{caller} \longrightarrow \\ \text{binder_systemActionRet} . \text{loc} . \text{caller} \longrightarrow \\ \text{systemAction_MethodBinder} \end{array} \right)$$

BinderActions $\hat{=}$
(*systemAction_MethodBinder*)

• *BinderActions* \triangle (*done_toplevel_sequencer* \longrightarrow **Skip**)

end

2.3 Locking

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

process *Threads* $\hat{=}$
(**Skip**)

process *Objects* $\hat{=}$
(**Skip**)

process *Locking* $\hat{=}$ (*Threads* \llbracket *ThreadSync* \rrbracket *Objects*) \triangle (*done_toplevel_sequencer* \longrightarrow **Skip**)

2.4 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, MTApp, OSEHApp*

process *ControlTier* $\hat{=}$

$$\left(\begin{array}{c} \text{SafeletFW} \\ \llbracket \text{ControlTierSync} \rrbracket \\ \text{TopLevelMissionSequencerFW}(\text{mainSequencer}) \end{array} \right)$$

process *Tier0* $\hat{=}$

$$\left(\begin{array}{c} \text{MissionFW}(\text{MissionAID}) \\ \llbracket \text{MissionSync} \rrbracket \\ \left(\begin{array}{c} \text{ManagedThreadFW}(\text{MTID}) \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \text{OneShotEventHandlerFW}(\text{OSEHID}, (\text{time}(60, 0)), (\text{time}(5, 0), \text{nullSchedulableId})) \end{array} \right) \end{array} \right)$$

process *Framework* $\hat{=}$

$$\left(\begin{array}{c} \text{ControlTier} \\ \llbracket \text{TierSync} \rrbracket \\ (\text{Tier0}) \end{array} \right)$$

process *Application* $\hat{=}$

$$\left(\begin{array}{c} \text{MyAppApp} \\ ||| \\ \text{mainSequencerApp} \\ ||| \\ \text{MissionAApp} \\ ||| \\ \text{MTApp}(\text{MissionAID}) \\ ||| \\ \text{OSEHApp}(\text{MissionAID}) \end{array} \right)$$

process *Bound_Application* $\hat{=}$ *Application* $\llbracket \text{MethodCallBinderSync} \rrbracket \text{MethodCallBinder}$
process *Program* $\hat{=}$ $(\text{Framework} \llbracket \text{AppSync} \rrbracket \text{Bound_Application}) \llbracket \text{LockingSync} \rrbracket \text{Locking}$

3 Safelet

section *MyAppApp* **parents** *scj_prelude, SchedulableId, SchedulableIds, SafeletChan, MethodCallBindingChannels*

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

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

GetSequencer $\hat{=}$
 $\left(\begin{array}{l} \textit{getSequencerCall} \longrightarrow \\ \textit{getSequencerRet} \text{ ! } \textit{mainSequencerSID} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

Methods $\hat{=}$
 $\left(\begin{array}{l} \textit{GetSequencer} \\ \square \\ \textit{InitializeApplication} \end{array} \right); \textit{Methods}$

• $(\textit{Methods}) \triangle (\textit{end_safelet_app} \longrightarrow \mathbf{Skip})$

end

4 Top Level Mission Sequencer

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

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

<i>State</i> <i>this</i> : ref <i>mainSequencerClass</i>
--

state *State*

<i>Init</i> <i>State</i> '
<i>this</i> ' = new <i>mainSequencerClass</i> ()

GetNextMission $\hat{=}$ **var** *ret* : *MissionID* •
 $\left(\begin{array}{l} \text{getNextMissionCall} . \text{mainSequencerSID} \longrightarrow \\ \text{ret} := \text{this} . \text{getNextMission}(); \\ \text{getNextMissionRet} . \text{mainSequencerSID} ! \text{ret} \longrightarrow \\ \text{Skip} \end{array} \right)$

Methods $\hat{=}$
 $(\text{GetNextMission}) ; \text{Methods}$

• $(\text{Init} ; \text{Methods}) \triangle (\text{end_sequencer_app} . \text{mainSequencerSID} \longrightarrow \text{Skip})$

end

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

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

state <i>State</i> <i>notReleased</i> : \mathbb{B}
--

state *State*

initial <i>Init</i> <i>State</i> '
<i>notReleased</i> ' = <i>true</i>

protected *getNextMission* $\hat{=}$

$$\left(\begin{array}{l} \text{if } notReleased \longrightarrow \\ \quad \left(notReleased := \mathbf{False}; \right. \\ \quad \left. ret := MissionAMID \right) \\ \square \neg notReleased \longrightarrow \\ \quad (ret := nullMissionId) \\ \text{fi} \end{array} \right)$$

• **Skip**

end

5 Missions

5.1 MissionA

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

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

<i>State</i> <i>this</i> : ref <i>MissionAClass</i>

state *State*

<i>Init</i> <i>State'</i>
<i>this'</i> = new <i>MissionAClass</i> ()

InitializePhase $\hat{=}$

$$\left(\begin{array}{l} \textit{initializeCall} . \textit{MissionAMID} \longrightarrow \\ \textit{register} ! \textit{OSEHSID} ! \textit{MissionAMID} \longrightarrow \\ \textit{register} ! \textit{MTSID} ! \textit{MissionAMID} \longrightarrow \\ \textit{initializeRet} . \textit{MissionAMID} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

CleanupPhase $\hat{=}$

$$\left(\begin{array}{l} \textit{cleanupMissionCall} . \textit{MissionAMID} \longrightarrow \\ \textit{cleanupMissionRet} . \textit{MissionAMID} ! \mathbf{True} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

systemActionMeth $\hat{=}$

$$\left(\begin{array}{l} \textit{systemActionCall} . \textit{MissionAMID} ? \textit{caller} \longrightarrow \\ \textit{this} . \textit{systemAction}(); \\ \textit{systemActionRet} . \textit{MissionAMID} . \textit{caller} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

Methods $\hat{=}$
$$\left(\begin{array}{l} \textit{InitializePhase} \\ \square \\ \textit{CleanupPhase} \\ \square \\ \textit{systemActionMeth} \end{array} \right); \textit{Methods}$$

• (*Init* ; *Methods*) \triangle (*end_mission_app* . *MissionAMID* \longrightarrow **Skip**)

end

section *MissionAClass* **parents** *scj_prelude*, *SchedulableId*, *SchedulableIds*, *SafeletChan*
, MethodCallBindingChannels

class *MissionAClass* $\hat{=}$ **begin**

public *systemAction* $\hat{=}$
Skip

• **Skip**

end

5.2 Schedulables of MissionA

section *MTApp* **parents** *ManagedThreadChan*, *SchedulableId*, *SchedulableIds*, *MethodCallBindingChannels*, *MissionAMethChan*

process *MTApp* $\hat{=}$
 controllingMission : *MissionID* • **begin**

Run $\hat{=}$

$$\left(\begin{array}{l} \text{runCall} . \text{MTSID} \longrightarrow \\ \left(\begin{array}{l} \text{binder_systemActionCall} . \text{controllingMission} . \text{MTSID} \longrightarrow \\ \text{binder_systemActionRet} . \text{controllingMission} . \text{MTSID} \longrightarrow \end{array} \right) ; \\ \text{Skip} \\ \text{runRet} . \text{MTSID} \longrightarrow \\ \text{Skip} \end{array} \right)$$

Methods $\hat{=}$
 (*Run*) ; *Methods*

• (*Methods*) \triangle (*end_managedThread_app* . *MTSID* \longrightarrow **Skip**)

end

section *OSEHApp* **parents** *OneShotEventHandlerChan*, *SchedulableId*, *SchedulableIds*, *MethodCallBindingChannels*

process *OSEHApp* $\hat{=}$
controllingMission : *MissionID* • **begin**

<i>State</i> <i>controllingMission</i> : <i>Mission</i>
--

state *State*

<i>Init</i> <i>State</i> '
<i>controllingMission</i> ' =

handleAsyncEvent $\hat{=}$

$$\left(\begin{array}{l} \text{handleAsyncEventCall} . \text{OSEHSID} \longrightarrow \\ \left(\begin{array}{l} \text{requestTerminationCall} . \text{controllingMission} . \text{OSEHSID} \longrightarrow \\ \text{requestTerminationRet} . \text{controllingMission} . \text{OSEHSID} ? \text{requestTermination} \longrightarrow \end{array} \right) ; \\ \mathbf{Skip} \\ \text{handleAsyncEventRet} . \text{OSEHSID} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

Methods $\hat{=}$
(*handleAsyncEvent*) ; *Methods*

• (*Init* ; *Methods*) \triangle (*end_oneShot_app* . *OSEHSID* \longrightarrow **Skip**)

end