

threeOneShots Report

4th October 2016

1 ID Files

1.1 MissionIds

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

<i>MissionAMID</i> : <i>MissionID</i>
<i>distinct</i> (<i>nullMissionId</i> , <i>MissionAMID</i>)

1.2 SchedulablesIds

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

mainSequencerSID : SchedulableID

OSEH1SID : SchedulableID

OSEH2SID : SchedulableID

OSEH3SID : SchedulableID

distinct(*nullSequencerId, nullSchedulableId, mainSequencerSID,*
OSEH1SID, OSEH2SID,
OSEH3SID)

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 Locking

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

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

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

process *Locking* $\hat{=}$ *Threads* [*ThreadSync*] *Objects*

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, OSEH1App, OSEH2App, OSEH3App*

process *ControlTier* $\hat{=}$

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

process *Tier0* $\hat{=}$

$$\left(\begin{array}{l} \text{MissionFW}(\text{MissionAID}) \\ \llbracket \text{MissionSync} \rrbracket \\ \left(\begin{array}{l} \text{OneShotEventHandlerFW}(\text{OSEH1ID}, (\text{time}(60, 0)), (\text{time}(5, 0), \text{null})) \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \text{OneShotEventHandlerFW}(\text{OSEH2ID}, (\text{time}(60, 0)), (\text{time}(5, 0), \text{null})) \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \text{OneShotEventHandlerFW}(\text{OSEH3ID}, (\text{time}(60, 0)), (\text{time}(5, 0), \text{null})) \end{array} \right) \end{array} \right)$$

process *Framework* $\hat{=}$

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

process *Application* $\hat{=}$

$$\left(\begin{array}{l} \text{MyAppApp} \\ ||| \\ \text{mainSequencerApp} \\ ||| \\ \text{MissionAApp} \\ ||| \\ \text{OSEH1App}(\text{MissionAID}) \\ ||| \\ \text{OSEH2App}(\text{MissionAID}) \\ ||| \\ \text{OSEH3App}(\text{MissionAID}) \end{array} \right)$$

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

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* $\hat{=}$
name : *String* • **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* •
 $\left(\begin{array}{l} \textit{getNextMissionCall} . \textit{mainSequencerSID} \longrightarrow \\ \textit{ret} := \textit{this} . \textit{getNextMission}(); \\ \textit{getNextMissionRet} . \textit{mainSequencerSID} ! \textit{ret} \longrightarrow \\ \textbf{Skip} \end{array} \right)$

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

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

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> = True

protected *getNextMission* $\hat{=}$ **var** *ret* : *MissionID* •

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

• **Skip**

end

5 Missions

5.1 MissionA

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

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

InitializePhase $\hat{=}$

$$\left(\begin{array}{l} \text{initializeCall} . \text{MissionAMID} \longrightarrow \\ \text{register} ! \text{ OSEH1SID} ! \text{MissionAMID} \longrightarrow \\ \text{register} ! \text{ OSEH2SID} ! \text{MissionAMID} \longrightarrow \\ \text{register} ! \text{ OSEH3SID} ! \text{MissionAMID} \longrightarrow \\ \text{initializeRet} . \text{MissionAMID} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

CleanupPhase $\hat{=}$

$$\left(\begin{array}{l} \mathbf{var} \ \mathbb{B} : \text{ret} \bullet \text{cleanupMissionCall} . \text{MissionAMID} \longrightarrow \\ \text{cleanupMissionRet} . \text{MissionAMID} ! \mathbf{True} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

Methods $\hat{=}$ $\left(\begin{array}{c} \text{InitializePhase} \\ \square \\ \text{CleanupPhase} \end{array} \right) ; \text{Methods}$

$\bullet (\text{Init} ; \text{Methods}) \triangle (\text{end_mission_app} . \text{MissionAMID} \longrightarrow \mathbf{Skip})$

end

5.2 Schedulables of MissionA

section *OSEH1App* **parents** *OneShotEventHandlerChan*, *SchedulableId*, *SchedulableIds*, *MethodCallBindingChannels*

process *OSEH1App* $\hat{=}$
 controllingMission : *MissionID* • **begin**

handleAsyncEvent $\hat{=}$

$$\left(\begin{array}{l} \textit{handleAsyncEventCall} . \textit{OSEH1SID} \longrightarrow \\ \left(\begin{array}{l} \textit{requestTerminationCall} . \textit{controllingMission} . \textit{OSEH1SID} \longrightarrow \\ \textit{requestTerminationRet} . \textit{controllingMission} . \textit{OSEH1SID} ? \textit{requestTermination} \longrightarrow \end{array} \right) ; \\ \mathbf{Skip} \\ \textit{handleAsyncEventRet} . \textit{OSEH1SID} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

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

• (*Methods*) \triangle (*end_oneShot_app* . *OSEH1SID* \longrightarrow **Skip**)

end

section *OSEH2App* **parents** *OneShotEventHandlerChan*, *SchedulableId*, *SchedulableIds*, *MethodCallBindingChannels*

process *OSEH2App* $\hat{=}$
 controllingMission : *MissionID* • **begin**

handleAsyncEvent $\hat{=}$

$$\left(\begin{array}{l} \textit{handleAsyncEventCall} . \textit{OSEH2SID} \longrightarrow \\ \left(\begin{array}{l} \textit{requestTerminationCall} . \textit{controllingMission} . \textit{OSEH2SID} \longrightarrow \\ \textit{requestTerminationRet} . \textit{controllingMission} . \textit{OSEH2SID} ? \textit{requestTermination} \longrightarrow \end{array} \right) ; \\ \mathbf{Skip} \\ \textit{handleAsyncEventRet} . \textit{OSEH2SID} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

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

• (*Methods*) \triangle (*end_oneShot_app* . *OSEH2SID* \longrightarrow **Skip**)

end

section *OSEH3App* **parents** *OneShotEventHandlerChan, SchedulableId, SchedulableIds, MethodCallBindingChannels*

process *OSEH3App* $\hat{=}$
 controllingMission : *MissionID* • **begin**

handleAsyncEvent $\hat{=}$

$$\left(\begin{array}{l} \textit{handleAsyncEventCall} . \textit{OSEH3SID} \longrightarrow \\ \left(\begin{array}{l} \textit{requestTerminationCall} . \textit{controllingMission} . \textit{OSEH3SID} \longrightarrow \\ \textit{requestTerminationRet} . \textit{controllingMission} . \textit{OSEH3SID} ? \textit{requestTermination} \longrightarrow \end{array} \right) ; \\ \mathbf{Skip} \\ \textit{handleAsyncEventRet} . \textit{OSEH3SID} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

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

• (*Methods*) \triangle (*end_oneShot_app* . *OSEH3SID* \longrightarrow **Skip**)

end