aircraft

Tight Rope v0.6

9th December 2015

1 ID Files

1.1 MissionIds

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

$$MainMissionID : MissionID$$
$$TakeOffMissionID : MissionID$$
$$CruiseMissionID : MissionID$$
$$LandMissionID : MissionID$$

$$distinct\langle nullMissionId, MainMissionID, TakeOffMissionID, CruiseMissionID, LandMissionID\rangle$$

1.2 SchedulablesIds

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

$$
\begin{array}{l}
MainMissionSequencerID : SchedulableID \\
ACModeChangerID : SchedulableID \\
EnvironmentMonitorID : SchedulableID \\
ControlHandlerID : SchedulableID \\
FlightSensorsMonitorID : SchedulableID \\
CommunicationsHandlerID : SchedulableID \\
AperiodicSimulatorID : SchedulableID \\
LandingGearHandlerTakeOffID : SchedulableID \\
TakeOffMonitorID : SchedulableID \\
TakeOffFailureHandlerID : SchedulableID \\
BeginLandingHandlerID : SchedulableID \\
NavigationMonitorID : SchedulableID \\
GroundDistanceMonitorID : SchedulableID \\
LandingGearHandlerLandID : SchedulableID \\
InstrumentLandingSystemMonitorID : SchedulableID \\
SafeLandingHandlerID : SchedulableID
\end{array}
$$

$$
\begin{array}{l}
distinct\langle nullSequencerId, nullSchedulableId, MainMissionSequencerID, \\
ACModeChangerID, EnvironmentMonitorID, \\
ControlHandlerID, FlightSensorsMonitorID, \\
CommunicationsHandlerID, AperiodicSimulatorID, \\
LandingGearHandlerTakeOffID, TakeOffMonitorID, \\
TakeOffFailureHandlerID, BeginLandingHandlerID, \\
NavigationMonitorID, GroundDistanceMonitorID, \\
LandingGearHandlerLandID, InstrumentLandingSystemMonitorID, \\
SafeLandingHandlerID\rangle
\end{array}
$$

1.3 ThreadIds

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

$$
\begin{array}{l}
SafeLandingHandlerThreadID : ThreadID \\
ACModeChangerThreadID : ThreadID \\
TakeOffFailureHandlerThreadID : ThreadID \\
InstrumentLandingSystemMonitorThreadID : ThreadID \\
FlightSensorsMonitorThreadID : ThreadID \\
TakeOffMonitorThreadID : ThreadID \\
AperiodicSimulatorThreadID : ThreadID \\
LandingGearHandlerLandThreadID : ThreadID \\
LandingGearHandlerTakeOffThreadID : ThreadID \\
GroundDistanceMonitorThreadID : ThreadID \\
ControlHandlerThreadID : ThreadID \\
CommunicationsHandlerThreadID : ThreadID \\
BeginLandingHandlerThreadID : ThreadID \\
NavigationMonitorThreadID : ThreadID \\
EnvironmentMonitorThreadID : ThreadID
\end{array}
$$

$$
\begin{array}{l}
distinct\langle SafeletThreadId, nullThreadId, \\
SafeLandingHandlerThreadID, ACModeChangerThreadID, \\
TakeOffFailureHandlerThreadID, InstrumentLandingSystemMonitorThreadID, \\
FlightSensorsMonitorThreadID, TakeOffMonitorThreadID, \\
AperiodicSimulatorThreadID, LandingGearHandlerLandThreadID, \\
LandingGearHandlerTakeOffThreadID, GroundDistanceMonitorThreadID, \\
ControlHandlerThreadID, CommunicationsHandlerThreadID, \\
BeginLandingHandlerThreadID, NavigationMonitorThreadID, \\
EnvironmentMonitorThreadID\rangle
\end{array}
$$

1.4 ObjectIds

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

$$
\begin{array}{l}
ACSafeletObjectID : ObjectID \\
MainMissionObjectID : ObjectID \\
ACModeChangerObjectID : ObjectID \\
EnvironmentMonitorObjectID : ObjectID \\
ControlHandlerObjectID : ObjectID \\
FlightSensorsMonitorObjectID : ObjectID \\
CommunicationsHandlerObjectID : ObjectID \\
AperiodicSimulatorObjectID : ObjectID \\
TakeOffMissionObjectID : ObjectID \\
LandingGearHandlerTakeOffObjectID : ObjectID \\
TakeOffMonitorObjectID : ObjectID \\
TakeOffFailureHandlerObjectID : ObjectID \\
CruiseMissionObjectID : ObjectID \\
BeginLandingHandlerObjectID : ObjectID \\
NavigationMonitorObjectID : ObjectID \\
LandMissionObjectID : ObjectID \\
GroundDistanceMonitorObjectID : ObjectID \\
LandingGearHandlerLandObjectID : ObjectID \\
InstrumentLandingSystemMonitorObjectID : ObjectID \\
SafeLandingHandlerObjectID : ObjectID
\end{array}
$$

$$
\begin{array}{l}
distinct\langle ACSafeletObjectID, MainMissionObjectID, \\
ACModeChangerObjectID, EnvironmentMonitorObjectID, \\
ControlHandlerObjectID, FlightSensorsMonitorObjectID, \\
CommunicationsHandlerObjectID, AperiodicSimulatorObjectID, \\
TakeOffMissionObjectID, LandingGearHandlerTakeOffObjectID, \\
TakeOffMonitorObjectID, TakeOffFailureHandlerObjectID, \\
CruiseMissionObjectID, BeginLandingHandlerObjectID, \\
NavigationMonitorObjectID, LandMissionObjectID, \\
GroundDistanceMonitorObjectID, LandingGearHandlerLandObjectID, \\
InstrumentLandingSystemMonitorObjectID, SafeLandingHandlerObjectID\rangle
\end{array}
$$

2 Network

section *NetworkChannels* **parents** *scj_prelude*, *MissionId*, *MissionIds*,
SchedulableId, *SchedulableIds*, *MissionChan*, *SchedulableChan*, *TopLevelMissionSequencerFWChan*,
FrameworkChan, *SafeletChan*

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

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

channelset *TierSync* ==
{| *start_mission* . , *done_mission* . ,
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 *AppSync* ==
$\bigcup$ {*SafeltAppSync*, *MissionSequencerAppSync*, *MissionAppSync*,
MTAppSync, *OSEHSync*, *APEHSync*,
{| *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* |}

channelset *Tier*0*Sync* ==
{| *done_toplevel_sequencer*, *done_safeletFW*,
start_mission . *TakeOffMission*, *done_mission* . *TakeOffMission*,
initializeRet . *TakeOffMission*, *requestTermination* . *TakeOffMission* . *MainMissionSequencer*,
start_mission . *CruiseMission*, *done_mission* . *CruiseMission*,
initializeRet . *CruiseMission*, *requestTermination* . *CruiseMission* . *MainMissionSequencer*,
start_mission . *LandMission*, *done_mission* . *LandMission*,
initializeRet . *LandMission*, *requestTermination* . *LandMission* . *MainMissionSequencer* |}

section *Program* **parents** *scj_prelude*, *MissionId*, *MissionIds*,
SchedulableId, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *MissionFW*,
SafeletFW, *TopLevelMissionSequencerFW*, *NetworkChannels*, *ManagedThreadFW*,
SchedulableMissionSequencerFW, *PeriodicEventHandlerFW*, *OneShotEventHandlerFW*,
AperiodicEventHandlerFW, *ObjectFW*, *ThreadFW*,
ACSafeletApp, *MainMissionSequencerApp*, *MainMissionApp*, *ACModeChangerApp*, *ControlHandlerApp*,
CommunicationsHandlerApp, *EnvironmentMonitorApp*, *FlightSensorsMonitorApp*,
AperiodicSimulatorApp, *TakeOffMissionApp*, *LandingGearHandlerTakeOffApp*, *TakeOffFailureHandlerApp*,
TakeOffMonitorApp, *CruiseMissionApp*, *BeginLandingHandlerApp*, *NavigationMonitorApp*
, *LandMissionApp*, *LandingGearHandlerLandApp*, *SafeLandingHandlerApp*, *GroundDistanceMonitorApp*,
InstrumentLandingSystemMonitorApp

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

process $Tier0 \mathrel{\widehat=}$
$$\left(\begin{array}{l} MissionFW(MainMissionID) \\ \quad [\![MissionSync]\!] \\ \left(\begin{array}{l} SchedulableMissionSequencerFW(ACModeChangerID) \\ \quad [\![SchedulablesSync]\!] \\ \left(\begin{array}{l} AperiodicEventHandlerFW(ControlHandlerID, (time(10,0), null)) \\ \quad [\![SchedulablesSync]\!] \\ AperiodicEventHandlerFW(CommunicationsHandlerID, (NULL, nullSchedulableId)) \end{array}\right) \\ \quad [\![SchedulablesSync]\!] \\ \left(\begin{array}{l} PeriodicEventHandlerFW(EnvironmentMonitorID, (time(10,0), NULL, NULL, nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ PeriodicEventHandlerFW(FlightSensorsMonitorID, (time(10,0), NULL, NULL, nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ PeriodicEventHandlerFW(AperiodicSimulatorID, (time(10,0), NULL, NULL, nullSchedulableId)) \end{array}\right) \end{array}\right) \end{array}\right)$$

process $Tier1 \mathrel{\widehat=}$
$$\left(\begin{array}{l} MissionFW(TakeOffMissionID) \\ \quad [\![MissionSync]\!] \\ \left(\begin{array}{l} \left(\begin{array}{l} AperiodicEventHandlerFW(LandingGearHandlerTakeOffID, (NULL, nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ AperiodicEventHandlerFW(TakeOffFailureHandlerID, (NULL, nullSchedulableId)) \end{array}\right) \\ \quad [\![SchedulablesSync]\!] \\ PeriodicEventHandlerFW(TakeOffMonitorID, (time(0,0), time(500,0), NULL, nullSchedulableId)) \end{array}\right) \end{array}\right)$$
$$[\![ClusterSync]\!]$$
$$\left(\begin{array}{l} MissionFW(CruiseMissionID) \\ \quad [\![MissionSync]\!] \\ \left(\begin{array}{l} AperiodicEventHandlerFW(BeginLandingHandlerID, (NULL, nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ PeriodicEventHandlerFW(NavigationMonitorID, (time(0,0), time(10,0), NULL, nullSchedulableId)) \end{array}\right) \end{array}\right)$$
$$[\![ClusterSync]\!]$$
$$\left(\begin{array}{l} MissionFW(LandMissionID) \\ \quad [\![MissionSync]\!] \\ \left(\begin{array}{l} \left(\begin{array}{l} AperiodicEventHandlerFW(LandingGearHandlerLandID, (NULL, nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ AperiodicEventHandlerFW(SafeLandingHandlerID, (NULL, nullSchedulableId)) \end{array}\right) \\ \quad [\![SchedulablesSync]\!] \\ \left(\begin{array}{l} PeriodicEventHandlerFW(GroundDistanceMonitorID, (time(0,0), time(10,0), NULL, nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ PeriodicEventHandlerFW(InstrumentLandingSystemMonitorID, (time(0,0), time(10,0), NULL, nullSchedulableId) \end{array}\right. \end{array}\right. \end{array}\right.$$

$$
\begin{array}{l}
\textbf{process}\ \mathit{Framework} \mathrel{\widehat{=}} \\
\left(\begin{array}{l}
\mathit{ControlTier} \\
\quad [\![\mathit{TierSync}]\!] \\
\left(\begin{array}{l}
\mathit{Tier}0 \\
\quad [\![\mathit{Tier}0\mathit{Sync}]\!] \\
\mathit{Tier}1
\end{array}\right)
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
\textbf{process}\ \mathit{Application} \mathrel{\widehat{=}} \\
\left(\begin{array}{l}
\mathit{ACSafeletApp} \\
|||\ \\
\mathit{MainMissionSequencerApp} \\
|||\ \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
\mathit{MainMissionApp} \\
\mathit{MissionSync} \\
\left(\begin{array}{l}
\mathit{ACModeChangerApp}(\mathit{MainMissionID}) \\
|||\ \\
\mathit{ControlHandlerApp} \\
|||\ \\
\mathit{CommunicationsHandlerApp} \\
|||\ \\
\mathit{EnvironmentMonitorApp}(\mathit{MainMissionID}) \\
|||\ \\
\mathit{FlightSensorsMonitorApp}(\mathit{MainMissionID}) \\
|||\ \\
\mathit{AperiodicSimulatorApp}(\mathit{controlHandlerID})
\end{array}\right)
\end{array}\right)
\end{array}\right) \\
\mathit{InterfaceSync} \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\mathit{TakeOffMissionApp} \\
\mathit{MissionSync} \\
\left(\begin{array}{l}
\mathit{LandingGearHandlerTakeOffApp}(\mathit{TakeOffMissionID}) \\
|||\ \\
\mathit{TakeOffFailureHandlerApp}(\mathit{MainMission}, \mathit{TakeOffMissionID},) \\
|||\ \\
\mathit{TakeOffMonitorApp}(\mathit{MainMission}, \mathit{TakeOffMissionID}, , \mathit{landingGearHandlerID})
\end{array}\right)
\end{array}\right) \\
|||\ \\
\left(\begin{array}{l}
\mathit{CruiseMissionApp} \\
\mathit{MissionSync} \\
\left(\begin{array}{l}
\mathit{BeginLandingHandlerApp}(\mathit{MainMission}) \\
|||\ \\
\mathit{NavigationMonitorApp}(\mathit{MainMission})
\end{array}\right)
\end{array}\right) \\
|||\ \\
\left(\begin{array}{l}
\mathit{LandMissionApp} \\
\mathit{MissionSync} \\
\left(\begin{array}{l}
\mathit{LandingGearHandlerLandApp}(\mathit{LandMissionID}) \\
|||\ \\
\mathit{SafeLandingHandlerApp}(\mathit{MainMission},) \\
|||\ \\
\mathit{GroundDistanceMonitorApp}(\mathit{MainMission}) \\
|||\ \\
\mathit{InstrumentLandingSystemMonitorApp}(\mathit{LandMissionID})
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}
$$

$$
\begin{aligned}
&Threads \mathrel{\widehat{=}} \\
&\left(\begin{array}{l}
ThreadFW(SafeLandingHandlerThreadID, 5) \\
\quad ||| \\
ThreadFW(ACModeChangerThreadID, 5) \\
\quad ||| \\
ThreadFW(TakeOffFailureHandlerThreadID, 5) \\
\quad ||| \\
ThreadFW(InstrumentLandingSystemMonitorThreadID, 5) \\
\quad ||| \\
ThreadFW(FlightSensorsMonitorThreadID, 5) \\
\quad ||| \\
ThreadFW(TakeOffMonitorThreadID, 5) \\
\quad ||| \\
ThreadFW(AperiodicSimulatorThreadID, 5) \\
\quad ||| \\
ThreadFW(LandingGearHandlerLandThreadID, 5) \\
\quad ||| \\
ThreadFW(LandingGearHandlerTakeOffThreadID, 5) \\
\quad ||| \\
ThreadFW(GroundDistanceMonitorThreadID, 5) \\
\quad ||| \\
ThreadFW(ControlHandlerThreadID, 5) \\
\quad ||| \\
ThreadFW(CommunicationsHandlerThreadID, 5) \\
\quad ||| \\
ThreadFW(BeginLandingHandlerThreadID, 5) \\
\quad ||| \\
ThreadFW(NavigationMonitorThreadID, 5) \\
\quad ||| \\
ThreadFW(EnvironmentMonitorThreadID, 5)
\end{array}\right)
\end{aligned}
$$

$$
\begin{aligned}
&Objects \mathrel{\widehat{=}} \\
&\left(\begin{array}{l}
ObjectFW(ACSafeletObjectID) \\
\quad |||\ ObjectFW(MainMissionObjectID) \\
\quad |||\ ObjectFW(ACModeChangerObjectID) \\
\quad |||\ ObjectFW(EnvironmentMonitorObjectID) \\
\quad |||\ ObjectFW(ControlHandlerObjectID) \\
\quad |||\ ObjectFW(FlightSensorsMonitorObjectID) \\
\quad |||\ ObjectFW(CommunicationsHandlerObjectID) \\
\quad |||\ ObjectFW(AperiodicSimulatorObjectID) \\
\quad |||\ ObjectFW(TakeOffMissionObjectID) \\
\quad |||\ ObjectFW(LandingGearHandlerTakeOffObjectID) \\
\quad |||\ ObjectFW(TakeOffMonitorObjectID) \\
\quad |||\ ObjectFW(TakeOffFailureHandlerObjectID) \\
\quad |||\ ObjectFW(CruiseMissionObjectID) \\
\quad |||\ ObjectFW(BeginLandingHandlerObjectID) \\
\quad |||\ ObjectFW(NavigationMonitorObjectID) \\
\quad |||\ ObjectFW(LandMissionObjectID) \\
\quad |||\ ObjectFW(GroundDistanceMonitorObjectID) \\
\quad |||\ ObjectFW(LandingGearHandlerLandObjectID) \\
\quad |||\ ObjectFW(InstrumentLandingSystemMonitorObjectID) \\
\quad |||\ ObjectFW(SafeLandingHandlerObjectID)
\end{array}\right)
\end{aligned}
$$

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

$$\textbf{process}\ Program \mathrel{\widehat{=}} \big(Framework \llbracket\ AppSync \ \rrbracket Application\big) \llbracket\ LockingSync \ \rrbracket Locking$$

3 Safelet

section $ACSafeletApp$ **parents** $scj\_prelude, SchedulableId, SchedulableIds, SafeletChan$

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

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

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

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

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

end

4 Top Level Mission Sequencer

section *MainMissionSequencerApp* **parents** *TopLevelMissionSequencerChan*, *MissionId*, *MissionIds*, *SchedulableId*, *MainMissionSequencerClass*

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

State

$this : \textbf{ref}\ MainMissionSequencerClass$

state *State*

Init

$State'$

$this' = \textbf{new}\ MainMissionSequencerClass()$

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

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

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

end

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

state *State*

$$returnedMission : \mathbb{B}$$

state *State*

initial *Init*

$$State'$$

$$returnedMission' = false$$

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

$$\left(\begin{array}{l} \textbf{if } (\neg\ returnedMission = \textbf{True}) \longrightarrow \\ \quad \left(\begin{array}{l} this\ .\ returnedMission := true; \\ ret := MainMission \end{array}\right) \\ [] \neg\ (\neg\ returnedMission = \textbf{True}) \longrightarrow \\ \quad \left(ret := nullMissionId\right) \\ \textbf{fi} \end{array}\right)$$

• **Skip**

end

5 Missions

5.1 MainMission

section *MainMissionApp* **parents** *scj_prelude*, *MissionId*, *MissionIds*, *SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *MainMissionClass*, *MainMissionMethChan*

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

State

$this : \mathbf{ref}\ MainMissionClass$

state *State*

Init

$State'$

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

$$InitializePhase \mathrel{\widehat=} \left(\begin{array}{l} initializeCall\,.\,MainMission \longrightarrow \\ register\,!\,ACModeChanger\,!\,MainMission \longrightarrow \\ register\,!\,EnvironmentMonitor\,!\,MainMission \longrightarrow \\ register\,!\,ControlHandler\,!\,MainMission \longrightarrow \\ register\,!\,FlightSensorsMonitor\,!\,MainMission \longrightarrow \\ register\,!\,CommunicationsHandler\,!\,MainMission \longrightarrow \\ register\,!\,AperiodicSimulator\,!\,MainMission \longrightarrow \\ initializeRet\,.\,MainMission \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$CleanupPhase \mathrel{\widehat=} \left(\begin{array}{l} cleanupMissionCall\,.\,MainMission \longrightarrow \\ cleanupMissionRet\,.\,MainMission\,!\,\mathbf{True} \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$getAirSpeedMeth \mathrel{\widehat=} \mathbf{var}\ ret : \mathbb{R} \bullet \left(\begin{array}{l} getAirSpeedCall\,.\,MainMission \longrightarrow \\ ret := this\,.\,getAirSpeed(); \\ getAirSpeedRet\,.\,MainMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$getAltitudeMeth \mathrel{\widehat=} \mathbf{var}\ ret : \mathbb{R} \bullet \left(\begin{array}{l} getAltitudeCall\,.\,MainMission \longrightarrow \\ ret := this\,.\,getAltitude(); \\ getAltitudeRet\,.\,MainMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$getCabinPressureMeth \mathrel{\widehat=} \mathbf{var}\ ret : \mathbb{R} \bullet \left(\begin{array}{l} getCabinPressureCall\,.\,MainMission \longrightarrow \\ ret := this\,.\,getCabinPressure(); \\ getCabinPressureRet\,.\,MainMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$getEmergencyOxygenMeth \mathrel{\widehat=} \textbf{var}\ ret : \mathbb{R} \bullet \left(\begin{array}{l} getEmergencyOxygenCall\ .\ MainMission \longrightarrow \\ ret := this\ .\ getEmergencyOxygen();\\ getEmergencyOxygenRet\ .\ MainMission\,!\,ret \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$getFuelRemainingMeth \mathrel{\widehat=} \textbf{var}\ ret : \mathbb{R} \bullet \left(\begin{array}{l} getFuelRemainingCall\ .\ MainMission \longrightarrow \\ ret := this\ .\ getFuelRemaining();\\ getFuelRemainingRet\ .\ MainMission\,!\,ret \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$getHeadingMeth \mathrel{\widehat=} \textbf{var}\ ret : \mathbb{R} \bullet \left(\begin{array}{l} getHeadingCall\ .\ MainMission \longrightarrow \\ ret := this\ .\ getHeading();\\ getHeadingRet\ .\ MainMission\,!\,ret \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$setAirSpeedMeth \mathrel{\widehat=} \left(\begin{array}{l} setAirSpeedCall\ .\ MainMission\,?\,airSpeed \longrightarrow \\ this\ .\ setAirSpeed(airSpeed);\\ setAirSpeedRet\ .\ MainMission \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$setAltitudeMeth \mathrel{\widehat=} \left(\begin{array}{l} setAltitudeCall\ .\ MainMission\,?\,altitude \longrightarrow \\ this\ .\ setAltitude(altitude);\\ setAltitudeRet\ .\ MainMission \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$setCabinPressureMeth \mathrel{\widehat=} \left(\begin{array}{l} setCabinPressureCall\ .\ MainMission\,?\,cabinPressure \longrightarrow \\ this\ .\ setCabinPressure(cabinPressure);\\ setCabinPressureRet\ .\ MainMission \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$setEmergencyOxygenMeth \mathrel{\widehat=} \left(\begin{array}{l} setEmergencyOxygenCall\ .\ MainMission\,?\,emergencyOxygen \longrightarrow \\ this\ .\ setEmergencyOxygen(emergencyOxygen);\\ setEmergencyOxygenRet\ .\ MainMission \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$setFuelRemainingMeth \mathrel{\widehat=} \left(\begin{array}{l} setFuelRemainingCall\ .\ MainMission\,?\,fuelRemaining \longrightarrow \\ this\ .\ setFuelRemaining(fuelRemaining);\\ setFuelRemainingRet\ .\ MainMission \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$setHeadingMeth \mathrel{\widehat=} \left(\begin{array}{l} setHeadingCall\ .\ MainMission\,?\,heading \longrightarrow \\ this\ .\ setHeading(heading);\\ setHeadingRet\ .\ MainMission \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$Methods \mathrel{\widehat{=}} \left(\begin{array}{l} InitializePhase \\ \Box \\ CleanupPhase \\ \Box \\ getAirSpeedMeth \\ \Box \\ getAltitudeMeth \\ \Box \\ getCabinPressureMeth \\ \Box \\ getEmergencyOxygenMeth \\ \Box \\ getFuelRemainingMeth \\ \Box \\ getHeadingMeth \\ \Box \\ setAirSpeedMeth \\ \Box \\ setAltitudeMeth \\ \Box \\ setCabinPressureMeth \\ \Box \\ setEmergencyOxygenMeth \\ \Box \\ setFuelRemainingMeth \\ \Box \\ setHeadingMeth \end{array} \right) ; \; Methods$$

• $(Init \; ; \; Methods) \triangle (end\_mission\_app \,.\, MainMission \longrightarrow \mathbf{Skip})$

end

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

state *State*

$ALTITUDE\_READING\_ON\_GROUND : \mathbb{R}$
$test : \mathbb{Z}$
$cabinPressure : \mathbb{R}$
$emergencyOxygen : \mathbb{R}$
$fuelRemaining : \mathbb{R}$
$altitude : \mathbb{R}$
$airSpeed : \mathbb{R}$
$heading : \mathbb{R}$

state *State*

initial *Init*

$State'$

$ALTITUDE\_READING\_ON\_GROUND' = 0.0$
$test' = 0$

public *getAirSpeed* $\widehat{=}$ **var** $ret : \mathbb{R} \bullet$
$(ret := airSpeed)$

public *getAltitude* $\widehat{=}$ **var** $ret : \mathbb{R} \bullet$
$(ret := altitude)$

public *getCabinPressure* $\widehat{=}$ **var** $ret : \mathbb{R} \bullet$
$(ret := cabinPressure)$

public *getEmergencyOxygen* $\widehat{=}$ **var** $ret : \mathbb{R} \bullet$
$(ret := emergencyOxygen)$

public *getFuelRemaining* $\widehat{=}$ **var** $ret : \mathbb{R} \bullet$
$(ret := fuelRemaining)$

public *getHeading* $\widehat{=}$ **var** $ret : \mathbb{R} \bullet$
$(ret := heading)$

public *setAirSpeed* $\widehat{=}$
$(this\,.\,this.airSpeed := airSpeed)$

public *setAltitude* $\widehat{=}$
$(this\,.\,this.altitude := altitude)$

public *setCabinPressure* $\widehat{=}$
$(this\,.\,this.cabinPressure := cabinPressure)$

public *setEmergencyOxygen* $\widehat{=}$
$(this\,.\,this.emergencyOxygen := emergencyOxygen)$

public $setFuelRemaining \mathrel{\widehat{=}}$
$\big(this\,.\,this.fuelRemaining := fuelRemaining\big)$

public $setHeading \mathrel{\widehat{=}}$
$\big(this\,.\,this.heading := heading\big)$

- **Skip**

end

5.2 Schedulables of MainMission

section $ACModeChangerApp$ **parents** $TopLevelMissionSequencerChan$, $MissionId$, $MissionIds$, $SchedulableId$, $ACModeChangerClass$

process $ACModeChangerApp \mathrel{\widehat=}$
$controllingMission : MissionID \bullet$ **begin**

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

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

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

end

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

state *State*

$controllingMission : MainMission$
$modesLeft : \mathbb{Z}$

state *State*

initial *Init*

$State'$

$modesLeft' = 3$

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

$$
\left(\begin{array}{l}
\textbf{if } (modesLeft = 3) \longrightarrow \\
\quad \left(\begin{array}{l} modesLeft := modesLeft - 1; \\ ret := TakeOffMission \end{array}\right) \\
[] \neg (modesLeft = 3) \longrightarrow \\
\quad \textbf{if } (modesLeft = 2) \longrightarrow \\
\quad \left(\begin{array}{l} modesLeft := modesLeft - 1; \\ ret := CruiseMission \end{array}\right) \\
[] \neg (modesLeft = 2) \longrightarrow \\
\quad \textbf{if } (modesLeft = 1) \longrightarrow \\
\quad \left(\begin{array}{l} modesLeft := modesLeft - 1; \\ ret := LandMission \end{array}\right) \\
[] \neg (modesLeft = 1) \longrightarrow \\
\quad \big(ret := nullMissionId\big) \\
\textbf{fi} \\
\textbf{fi} \\
\textbf{fi}
\end{array}\right)
$$

• **Skip**

end

section *ControlHandlerApp* **parents** *AperiodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*

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

$$handlerAsyncEvent \mathrel{\widehat=} \left(\begin{array}{l} handleAsyncEventCall\,.\,ControlHandler \longrightarrow \\ (\textbf{Skip})\,; \\ handleAsyncEventRet\,.\,ControlHandler \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

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

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

end

section $CommunicationsHandlerApp$ **parents** $AperiodicEventHandlerChan, SchedulableId, SchedulableIds$

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

$$handlerAsyncEvent \mathrel{\widehat=} \left(\begin{array}{l} handleAsyncEventCall\,.\,CommunicationsHandler \longrightarrow \\ (\textbf{Skip})\,; \\ handleAsyncEventRet\,.\,CommunicationsHandler \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

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

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

end

section *EnvironmentMonitorApp* **parents** *PeriodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*
,
MainMissionMethChan

process $EnvironmentMonitorApp \mathrel{\widehat=}$
$mainMission : MissionID \bullet$ **begin**

$handlerAsyncEvent \mathrel{\widehat=}$
$\Big($ *handleAsyncEventCall* . *EnvironmentMonitor* $\longrightarrow$
$\Big($ **Skip**;
setCabinPressureCall . *controllingMission* ! 0 $\longrightarrow$
setCabinPressureRet . *controllingMission* $\longrightarrow$
Skip;
setEmergencyOxygenCall . *controllingMission* ! 0 $\longrightarrow$
setEmergencyOxygenRet . *controllingMission* $\longrightarrow$
Skip;
setFuelRemainingCall . *controllingMission* ! 0 $\longrightarrow$
setFuelRemainingRet . *controllingMission* $\longrightarrow$
Skip $\Big)$;
handleAsyncEventRet . *EnvironmentMonitor* $\longrightarrow$
Skip $\Big)$

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

$\bullet$ $(Methods) \triangle (end\_periodic\_app \,.\, EnvironmentMonitor \longrightarrow$ **Skip**$)$

end

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

state *State*

$controllingMission : MainMission$

state *State*

initial *Init*

$State'$

- **Skip**

end

section *FlightSensorsMonitorApp* **parents** *PeriodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*
,
MainMissionMethChan

process *FlightSensorsMonitorApp* $\widehat{=}$
mainMission : *MissionID* • **begin**

$$
\begin{array}{l}
handlerAsyncEvent \mathrel{\widehat{=}} \\
\left(\begin{array}{l}
handleAsyncEventCall\,.\,FlightSensorsMonitor \longrightarrow \\
\left(\begin{array}{l}
\textbf{Skip}; \\
setAirSpeedCall\,.\,controllingMission\,!\,0 \longrightarrow \\
setAirSpeedRet\,.\,controllingMission \longrightarrow \\
\textbf{Skip}; \\
setAltitudeCall\,.\,controllingMission\,!\,0 \longrightarrow \\
setAltitudeRet\,.\,controllingMission \longrightarrow \\
\textbf{Skip}; \\
setHeadingCall\,.\,controllingMission\,!\,0 \longrightarrow \\
setHeadingRet\,.\,controllingMission \longrightarrow \\
\textbf{Skip}
\end{array}\right) \\
handleAsyncEventRet\,.\,FlightSensorsMonitor \longrightarrow \\
\textbf{Skip}
\end{array}\right)
\end{array}
$$

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

• $(Methods) \triangle (end\_periodic\_app\,.\,FlightSensorsMonitor \longrightarrow \textbf{Skip})$

end

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

state *State*

$controllingMission : MainMission$

state *State*

initial *Init*

$State'$

- **Skip**

end

section *AperiodicSimulatorApp* **parents** *PeriodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*

process $AperiodicSimulatorApp \mathrel{\widehat=}$
$aperiodicEvent : SchedulableID \bullet$ **begin**

$$handlerAsyncEvent \mathrel{\widehat=}$$
$$\left(\begin{array}{l} handleAsyncEventCall\,.\,AperiodicSimulator \longrightarrow \\ \left(\begin{array}{l} \textbf{Skip}; \\ releaseCall\,.\,event \longrightarrow \\ releaseRet\,.\,event\,?\,release \longrightarrow \\ \textbf{Skip} \end{array}\right); \\ handleAsyncEventRet\,.\,AperiodicSimulator \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

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

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

end

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

state *State*

$event : AperiodicEventHandler$

state *State*

initial *Init*

$State'$

- **Skip**

end

5.3 TakeOffMission

section *TakeOffMissionApp* **parents** *scj_prelude*, *MissionId*, *MissionIds*, *SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *TakeOffMissionClass*, *TakeOffMissionMethChan*

process $TakeOffMissionApp \mathrel{\widehat=}$
$controllingMission : MissionID \bullet$ **begin**

State

$this : \mathbf{ref}\ TakeOffMissionClass$

state *State*

Init

$State'$

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

$InitializePhase \mathrel{\widehat=}$
$\left(\begin{array}{l} initializeCall\,.\,TakeOffMission \longrightarrow \\ register\,!\,LandingGearHandlerTakeOff\,!\,TakeOffMission \longrightarrow \\ register\,!\,TakeOffMonitor\,!\,TakeOffMission \longrightarrow \\ register\,!\,TakeOffFailureHandler\,!\,TakeOffMission \longrightarrow \\ initializeRet\,.\,TakeOffMission \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$CleanupPhase \mathrel{\widehat=}$
$\left(\begin{array}{l} cleanupMissionCall\,.\,TakeOffMission \longrightarrow \\ cleanupMissionRet\,.\,TakeOffMission\,!\,\mathbf{True} \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$abortMeth \mathrel{\widehat=}$
$\left(\begin{array}{l} abortCall\,.\,TakeOffMission \longrightarrow \\ this\,.\,abort(); \\ abortRet\,.\,TakeOffMission \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$getControllingMissionMeth \mathrel{\widehat=} \mathbf{var}\ ret : MissionID \bullet$
$\left(\begin{array}{l} getControllingMissionCall\,.\,TakeOffMission \longrightarrow \\ ret := this\,.\,getControllingMission(); \\ getControllingMissionRet\,.\,TakeOffMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$setControllingMissionMeth \mathrel{\widehat=}$
$\left(\begin{array}{l} setControllingMissionCall\,.\,TakeOffMission\,?\,controllingMission \longrightarrow \\ this\,.\,setControllingMission(controllingMission); \\ setControllingMissionRet\,.\,TakeOffMission \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$$cleanUpMeth \mathrel{\widehat=} \mathbf{var}\ ret : \mathbb{B} \bullet \left(\begin{array}{l} cleanUpCall\ .\ TakeOffMission \longrightarrow \\ ret := this\ .\ cleanUp(); \\ cleanUpRet\ .\ TakeOffMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$stowLandingGearMeth \mathrel{\widehat=} \left(\begin{array}{l} stowLandingGearCall\ .\ TakeOffMission \longrightarrow \\ this\ .\ stowLandingGear(); \\ stowLandingGearRet\ .\ TakeOffMission \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$isLandingGearDeployedMeth \mathrel{\widehat=} \mathbf{var}\ ret : \mathbb{B} \bullet \left(\begin{array}{l} isLandingGearDeployedCall\ .\ TakeOffMission \longrightarrow \\ ret := this\ .\ isLandingGearDeployed(); \\ isLandingGearDeployedRet\ .\ TakeOffMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$deployLandingGearSyncMeth \mathrel{\widehat=} \left(\begin{array}{l} deployLandingGearCall\ .\ TakeOffMission\,?\,thread \longrightarrow \\ \left(\begin{array}{l} startSyncMeth\ .\ TakeOffMissionObject\ .\ thread \longrightarrow \\ lockAcquired\ .\ TakeOffMissionObject\ .\ thread \longrightarrow \\ \left(this\ .\ landingGearDeployed := true\right); \\ endSyncMeth\ .\ TakeOffMissionObject\ .\ thread \longrightarrow \\ deployLandingGearRet\ .\ TakeOffMission\ .\ thread \longrightarrow \\ \mathbf{Skip} \end{array}\right) \end{array}\right)$$

$$Methods \mathrel{\widehat=} \left(\begin{array}{l} InitializePhase \\ \Box \\ CleanupPhase \\ \Box \\ abortMeth \\ \Box \\ getControllingMissionMeth \\ \Box \\ setControllingMissionMeth \\ \Box \\ cleanUpMeth \\ \Box \\ stowLandingGearMeth \\ \Box \\ isLandingGearDeployedMeth \\ \Box \\ deployLandingGearSyncMeth \end{array}\right) ;\ Methods$$

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

end

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

state *State*

$SAFE\_AIRSPEED\_THRESHOLD : \mathbb{R}$
$TAKEOFF\_ALTITUDE : \mathbb{R}$
$controllingMission : MainMission$
$abort : \mathbb{B}$
$landingGearDeployed : \mathbb{B}$

state *State*

initial *Init*

$State'$

$SAFE\_AIRSPEED\_THRESHOLD' = 10.0$
$TAKEOFF\_ALTITUDE' = 10.0$
$abort' = false$

public *abort* $\widehat{=}$
$\big(this\,.\,abort := true\big)$

public *getControllingMission* $\widehat{=}$ **var** $ret : MissionID \bullet$
$\big(ret := controllingMission\big)$

public *setControllingMission* $\widehat{=}$
$\big(this\,.\,this.controllingMission := controllingMission\big)$

public *cleanUp* $\widehat{=}$ **var** $ret : \mathbb{B} \bullet$
$\begin{pmatrix}\textbf{Skip};\\ ret := (\neg\ abort = \textbf{True})\end{pmatrix}$

public *stowLandingGear* $\widehat{=}$
$\big(this\,.\,landingGearDeployed := false\big)$

public *isLandingGearDeployed* $\widehat{=}$ **var** $ret : \mathbb{B} \bullet$
$\big(ret := landingGearDeployed = \textbf{True}\big)$

• **Skip**

end

section *TakeOffMissionMethChan* **parents** *scj_prelude*, *GlobalTypes*, *MissionId*, *SchedulableId*

channel $abortCall : SchedulableID$
channel $abortRet : SchedulableID$

channel $getControllingMissionCall : SchedulableID$
channel $getControllingMissionRet : SchedulableID \times MissionID$

channel $setControllingMissionCall : SchedulableID \times MissionID$
channel $setControllingMissionRet : SchedulableID$

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

channel $stowLandingGearCall : SchedulableID$
channel $stowLandingGearRet : SchedulableID$

channel $isLandingGearDeployedCall : SchedulableID$
channel $isLandingGearDeployedRet : SchedulableID \times \mathbb{B}$

channel $deployLandingGearCall : SchedulableID \times ThreadID$
channel $deployLandingGearRet : SchedulableID \times ThreadID$

5.4 Schedulables of TakeOffMission

section *LandingGearHandlerTakeOffApp* **parents** *AperiodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*
,
TakeOffMissionMethChan, *ObjectIds*, *ThreadIds*

process *LandingGearHandlerTakeOffApp* $\widehat{=}$
mission : *MissionID* • **begin**

$$
\begin{array}{l}
handlerAsyncEvent \mathrel{\widehat{=}} \\
\left(\begin{array}{l}
handleAsyncEventCall\,.\,LandingGearHandlerTakeOff \longrightarrow \\
\left(\begin{array}{l}
\textbf{Skip}; \\
isLandingGearDeployedCall\,.\,mission \longrightarrow \\
isLandingGearDeployedRet\,.\,mission\,?\,isLandingGearDeployed \longrightarrow \\
\\
\textbf{var}\ landingGearIsDeployed : \mathbb{B} \bullet landingGearIsDeployed := isLandingGearDeployed \\
\textbf{if}\ landingGearIsDeployed = \textbf{True} \longrightarrow \\
\qquad \left(\begin{array}{l} stowLandingGearCall\,.\,mission \longrightarrow \\ stowLandingGearRet\,.\,mission \longrightarrow \\ \textbf{Skip} \end{array}\right) \\
[]\ \neg\ landingGearIsDeployed = \textbf{True} \longrightarrow \\
\qquad \left(\begin{array}{l} deployLandingGearCall\,.\,mission\,.\,LandingGearHandlerTakeOffThread \longrightarrow \\ deployLandingGearRet\,.\,mission\,.\,LandingGearHandlerTakeOffThread \longrightarrow \\ \textbf{Skip} \end{array}\right) \\
\textbf{fi}
\end{array}\right) \\
handleAsyncEventRet\,.\,LandingGearHandlerTakeOff \longrightarrow \\
\textbf{Skip}
\end{array}\right) \\
\\
Methods \mathrel{\widehat{=}} \\
\left(handlerAsyncEvent\right)\ ;\ Methods
\end{array}
$$

• $(Methods) \triangle (end\_aperiodic\_app\,.\,LandingGearHandlerTakeOff \longrightarrow \textbf{Skip})$

end

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

state *State*

$mission : TakeOffMission$

state *State*

initial *Init*

$State'$

- **Skip**

end

section *TakeOffFailureHandlerApp* **parents** *AperiodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*
,
MainMissionMethChan, *TakeOffMissionMethChan*

process $TakeOffFailureHandlerApp \mathrel{\widehat=}$
$mainMission : MissionID,$
$takeoffMission : MissionID,$
$threshold : Double \bullet$ **begin**

$handlerAsyncEvent \mathrel{\widehat=}$
$\Big(\Big($
$handleAsyncEventCall\,.\,TakeOffFailureHandler \longrightarrow$
$\Big($
$\Big($
$getAirSpeedCall\,.\,mainMission \longrightarrow$
$getAirSpeedRet\,.\,mainMission\,?\,getAirSpeed \longrightarrow$
$\Big)$
$\mathbf{var}\ currentSpeed : \mathbb{R} \bullet currentSpeed := getAirSpeed$
$\mathbf{if}\ (currentSpeed < threshold) \longrightarrow$
$\Big($
$\Big($
$\mathbf{Skip};$
$abortCall\,.\,takeoffMission \longrightarrow$
$abortRet\,.\,takeoffMission \longrightarrow$
$\mathbf{Skip};$
$requestTerminationCall\,.\,takeoffMission \longrightarrow$
$requestTerminationRet\,.\,takeoffMission\,?\,requestTermination \longrightarrow$
$\mathbf{Skip}$
$\Big)$
$[]\ \neg\ (currentSpeed < threshold) \longrightarrow$
$\big(\mathbf{Skip}\big)$
$\mathbf{fi}\ \mathbf{Skip}$
$\Big)$
$\Big)$
;
$handleAsyncEventRet\,.\,TakeOffFailureHandler \longrightarrow$
$\mathbf{Skip}$
$\Big)\Big)$

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

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

end

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

state *State*

$mainMission : MainMission$
$takeoffMission : TakeOffMission$
$threshold : \mathbb{R}$

state *State*

initial *Init*

$State'$

- **Skip**

end

section *TakeOffMonitorApp* **parents** *PeriodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*
,
MainMissionMethChan

process $TakeOffMonitorApp \mathrel{\widehat=}$
$mainMission : MissionID,$
$takeOffMission : MissionID,$
$takeOffAltitude : \mathbb{R},$
$landingGearHandler : SchedulableID \bullet$ **begin**

$handlerAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall \,.\, TakeOffMonitor \longrightarrow$
$\Big($ **Skip**;
$getAltitudeCall \,.\, mainMission \longrightarrow$
$getAltitudeRet \,.\, mainMission \,?\, getAltitude \longrightarrow$

var $altitude : \mathbb{R} \bullet altitude := getAltitude$
if $(altitude > takeOffAltitude) \longrightarrow$
$\Big($ **Skip**;
$releaseCall \,.\, landingGearHandler \longrightarrow$
$releaseRet \,.\, landingGearHandler \,?\, release \longrightarrow$
$requestTerminationCall \,.\, takeoffMission \longrightarrow$
$requestTerminationRet \,.\, takeoffMission \,?\, requestTermination \longrightarrow$
Skip $\Big)$
$[] \neg (altitude > takeOffAltitude) \longrightarrow$ **Skip**
fi ;
Skip $\Big)$
$handleAsyncEventRet \,.\, TakeOffMonitor \longrightarrow$
Skip $\Big)$;

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

$\bullet$ $(Methods) \triangle (end\_periodic\_app \,.\, TakeOffMonitor \longrightarrow$ **Skip**$)$

end

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

state *State*

$mainMission : MainMission$
$takeoffMission : TakeOffMission$
$takeOffAltitude : \mathbb{R}$
$landingGearHandler : AperiodicEventHandler$

state *State*

initial *Init*

$State'$

- **Skip**

end

5.5 CruiseMission

section *CruiseMissionApp* **parents** *scj_prelude*, *MissionId*, *MissionIds*, *SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *CruiseMissionClass*, *CruiseMissionMethChan*

process $CruiseMissionApp \mathrel{\widehat=}$
$controllingMission : MissionID \bullet$ **begin**

State

$this : \mathbf{ref}\ CruiseMissionClass$

state *State*

Init

$State'$

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

$$InitializePhase \mathrel{\widehat=} \begin{pmatrix} initializeCall\,.\,CruiseMission \longrightarrow \\ register\,!\,BeginLandingHandler\,!\,CruiseMission \longrightarrow \\ register\,!\,NavigationMonitor\,!\,CruiseMission \longrightarrow \\ initializeRet\,.\,CruiseMission \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

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

$$getControllingMissionMeth \mathrel{\widehat=} \mathbf{var}\ ret : MissionID \bullet \begin{pmatrix} getControllingMissionCall\,.\,CruiseMission \longrightarrow \\ ret := this\,.\,getControllingMission(); \\ getControllingMissionRet\,.\,CruiseMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

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

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

end

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

state *State*

$controllingMission : MainMission$

state *State*

initial *Init*

$State'$

public *getControllingMission* $\widehat{=}$ **var** $ret : MissionID \bullet$
$\big(ret := controllingMission\big)$

• **Skip**

end

5.6 Schedulables of CruiseMission

section *BeginLandingHandlerApp* **parents** *AperiodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*

process $BeginLandingHandlerApp \mathrel{\widehat=}$
$controllingMission : MissionID \bullet$ **begin**

$$handlerAsyncEvent \mathrel{\widehat=}$$
$$\left(\begin{array}{l} handleAsyncEventCall\,.\,BeginLandingHandler \longrightarrow \\ \left(\begin{array}{l} \textbf{Skip}; \\ requestTerminationCall\,.\,controllingMission \longrightarrow \\ requestTerminationRet\,.\,controllingMission\,?\,requestTermination \longrightarrow \\ \textbf{Skip} \end{array}\right); \\ handleAsyncEventRet\,.\,BeginLandingHandler \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

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

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

end

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

state *State*

controllingMission : *Mission*

state *State*

initial *Init*

State'

- **Skip**

end

section *NavigationMonitorApp* **parents** *PeriodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*
,
MainMissionMethChan

process $NavigationMonitorApp \mathrel{\widehat=}$
$mainMission : MissionID \bullet$ **begin**

$handlerAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall \,.\, NavigationMonitor \longrightarrow$
$\Big($ $getHeadingCall \,.\, mainMission \longrightarrow$
$getHeadingRet \,.\, mainMission \,?\, getHeading \longrightarrow$
$\Big)$
var $heading : \mathbb{R} \bullet heading := getHeading$
$getAirSpeedCall \,.\, mainMission \longrightarrow$
$getAirSpeedRet \,.\, mainMission \,?\, getAirSpeed \longrightarrow$
var $airSpeed : \mathbb{R} \bullet airSpeed := getAirSpeed$
$getAltitudeCall \,.\, mainMission \longrightarrow$
$getAltitudeRet \,.\, mainMission \,?\, getAltitude \longrightarrow$
var $altitude : \mathbb{R} \bullet altitude := getAltitude$
Skip
$\Big)$;
$handleAsyncEventRet \,.\, NavigationMonitor \longrightarrow$
Skip
$\Big)$

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

$\bullet$ $(Methods) \triangle (end\_periodic\_app \,.\, NavigationMonitor \longrightarrow$ **Skip**$)$

end

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

state *State*

$mainMission : MainMission$

state *State*

initial *Init*

$State'$

- **Skip**

end

5.7 LandMission

section *LandMissionApp* **parents** *scj_prelude*, *MissionId*, *MissionIds*, *SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *LandMissionClass*, *LandMissionMethChan*

process $LandMissionApp \mathrel{\widehat=}$
$controllingMission : MissionID \bullet$ **begin**

State

$this : \mathbf{ref}\ LandMissionClass$

state *State*

Init

$State'$

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

$InitializePhase \mathrel{\widehat=}$
$\left(\begin{array}{l} initializeCall\,.\,LandMission \longrightarrow \\ register\,!\,GroundDistanceMonitor\,!\,LandMission \longrightarrow \\ register\,!\,LandingGearHandlerLand\,!\,LandMission \longrightarrow \\ register\,!\,InstrumentLandingSystemMonitor\,!\,LandMission \longrightarrow \\ register\,!\,SafeLandingHandler\,!\,LandMission \longrightarrow \\ initializeRet\,.\,LandMission \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$CleanupPhase \mathrel{\widehat=}$
$\left(\begin{array}{l} cleanupMissionCall\,.\,LandMission \longrightarrow \\ cleanupMissionRet\,.\,LandMission\,!\,\mathbf{True} \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$stowLandingGearMeth \mathrel{\widehat=}$
$\left(\begin{array}{l} stowLandingGearCall\,.\,LandMission \longrightarrow \\ this\,.\,stowLandingGear(); \\ stowLandingGearRet\,.\,LandMission \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$isLandingGearDeployedMeth \mathrel{\widehat=} \mathbf{var}\ ret : \mathbb{B} \bullet$
$\left(\begin{array}{l} isLandingGearDeployedCall\,.\,LandMission \longrightarrow \\ ret := this\,.\,isLandingGearDeployed(); \\ isLandingGearDeployedRet\,.\,LandMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$getControllingMissionMeth \mathrel{\widehat=} \mathbf{var}\ ret : MissionID \bullet$
$\left(\begin{array}{l} getControllingMissionCall\,.\,LandMission \longrightarrow \\ ret := this\,.\,getControllingMission(); \\ getControllingMissionRet\,.\,LandMission\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$$abortMeth \mathrel{\widehat=} \left(\begin{array}{l} abortCall\,.\,LandMission \longrightarrow \\ this\,.\,abort(); \\ abortRet\,.\,LandMission \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$cleanUpMeth \mathrel{\widehat=} \textbf{var}\ ret : \mathbb{B} \bullet \left(\begin{array}{l} cleanUpCall\,.\,LandMission \longrightarrow \\ ret := this\,.\,cleanUp(); \\ cleanUpRet\,.\,LandMission\,!\,ret \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$deployLandingGearSyncMeth \mathrel{\widehat=} \left(\begin{array}{l} deployLandingGearCall\,.\,LandMission\,?\,thread \longrightarrow \\ \left(\begin{array}{l} startSyncMeth\,.\,LandMissionObject\,.\,thread \longrightarrow \\ lockAcquired\,.\,LandMissionObject\,.\,thread \longrightarrow \\ \big(this\,.\,landingGearDeployed := true\big)\,; \\ endSyncMeth\,.\,LandMissionObject\,.\,thread \longrightarrow \\ deployLandingGearRet\,.\,LandMission\,.\,thread \longrightarrow \\ \textbf{Skip} \end{array}\right) \end{array}\right)$$

$$Methods \mathrel{\widehat=} \left(\begin{array}{l} InitializePhase \\ \Box \\ CleanupPhase \\ \Box \\ stowLandingGearMeth \\ \Box \\ isLandingGearDeployedMeth \\ \Box \\ getControllingMissionMeth \\ \Box \\ abortMeth \\ \Box \\ cleanUpMeth \\ \Box \\ deployLandingGearSyncMeth \end{array}\right) ;\ Methods$$

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

end

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

state *State*

$controllingMission : MainMission$
$SAFE\_LANDING\_ALTITUDE : \mathbb{R}$
$abort : \mathbb{B}$
$landingGearDeployed : \mathbb{B}$

state *State*

initial *Init*

$State'$

$SAFE\_LANDING\_ALTITUDE' = 10.0$
$abort' = false$

public *stowLandingGear* $\widehat{=}$
$(this\,.\,landingGearDeployed := false)$

public *isLandingGearDeployed* $\widehat{=}$ **var** $ret : \mathbb{B} \bullet$
$(ret := landingGearDeployed = \mathbf{True})$

public *getControllingMission* $\widehat{=}$ **var** $ret : MissionID \bullet$
$(ret := controllingMission)$

public *abort* $\widehat{=}$
$(this\,.\,abort := true)$

public *cleanUp* $\widehat{=}$ **var** $ret : \mathbb{B} \bullet$
$\begin{pmatrix} \mathbf{Skip}; \\ ret := (\neg\ abort = \mathbf{True}) \end{pmatrix}$

• **Skip**

end

section *LandMissionMethChan* **parents** *scj_prelude*, *GlobalTypes*, *MissionId*, *SchedulableId*

channel *stowLandingGearCall* : *SchedulableID*
channel *stowLandingGearRet* : *SchedulableID*

channel *isLandingGearDeployedCall* : *SchedulableID*
channel *isLandingGearDeployedRet* : *SchedulableID* $\times$ $\mathbb{B}$

channel *getControllingMissionCall* : *SchedulableID*
channel *getControllingMissionRet* : *SchedulableID* $\times$ *MissionID*

channel *abortCall* : *SchedulableID*
channel *abortRet* : *SchedulableID*

channel *cleanUpCall* : *SchedulableID*
channel *cleanUpRet* : *SchedulableID* $\times$ $\mathbb{B}$

channel *deployLandingGearCall* : *SchedulableID* $\times$ *ThreadID*
channel *deployLandingGearRet* : *SchedulableID* $\times$ *ThreadID*

5.8 Schedulables of LandMission

section *LandingGearHandlerLandApp* **parents** *AperiodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*, *LandMissionMethChan*, *ObjectIds*, *ThreadIds*

process $LandingGearHandlerLandApp \mathrel{\widehat=}$
$mission : MissionID \bullet$ **begin**

$$
\begin{array}{l}
handlerAsyncEvent \mathrel{\widehat=} \\
\left(\begin{array}{l}
handleAsyncEventCall\,.\,LandingGearHandlerLand \longrightarrow \\
\left(\begin{array}{l}
\textbf{Skip}; \\
isLandingGearDeployedCall\,.\,mission \longrightarrow \\
isLandingGearDeployedRet\,.\,mission\,?\,isLandingGearDeployed \longrightarrow \\
\\
\textbf{var}\ landingGearIsDeployed : \mathbb{B} \bullet landingGearIsDeployed := isLandingGearDeployed \\
\textbf{if}\ landingGearIsDeployed = \textbf{True} \longrightarrow \\
\qquad \left(\begin{array}{l} stowLandingGearCall\,.\,mission \longrightarrow \\ stowLandingGearRet\,.\,mission \longrightarrow \\ \textbf{Skip} \end{array}\right) \\
[\!]\ \neg\, landingGearIsDeployed = \textbf{True} \longrightarrow \\
\qquad \left(\begin{array}{l} deployLandingGearCall\,.\,mission\,.\,LandingGearHandlerLandThread \longrightarrow \\ deployLandingGearRet\,.\,mission\,.\,LandingGearHandlerLandThread \longrightarrow \\ \textbf{Skip} \end{array}\right) \\
\textbf{fi}
\end{array}\right) \\
handleAsyncEventRet\,.\,LandingGearHandlerLand \longrightarrow \\
\textbf{Skip}
\end{array}\right)
\end{array}
$$

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

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

end

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

state *State*

$mission : LandMission$

state *State*

initial *Init*

$State'$

- **Skip**

end

section *SafeLandingHandlerApp* **parents** *AperiodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*
,
MainMissionMethChan

process $SafeLandingHandlerApp \mathrel{\widehat=}$
$mainMission : MissionID,$
$threshold : Double \bullet$ **begin**

$handlerAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall \,.\, SafeLandingHandler \longrightarrow$
$\Big($ $getAltitudeCall \,.\, mainMission \longrightarrow$
$getAltitudeRet \,.\, mainMission \,?\, getAltitude \longrightarrow$

var $altitude : \mathbb{R} \bullet altitude := getAltitude$
if $(altitude < threshold) \longrightarrow$
$\big($**Skip**$\big)$
$[] \neg (altitude < threshold) \longrightarrow$
$\big($**Skip**$\big)$
fi
$\Big)$;
$handleAsyncEventRet \,.\, SafeLandingHandler \longrightarrow$
Skip
$\Big)$

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

$\bullet$ $(Methods) \triangle (end\_aperiodic\_app \,.\, SafeLandingHandler \longrightarrow$ **Skip**$)$

end

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

state *State*

$mainMission : MainMission$

$threshold : \mathbb{R}$

state *State*

initial *Init*

$State'$

- **Skip**

end

section *GroundDistanceMonitorApp* **parents** *PeriodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*, *MainMissionMethChan*

process $GroundDistanceMonitorApp \mathrel{\widehat=}$
$mainMission : MissionID \bullet$ **begin**

$handlerAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall \,.\, GroundDistanceMonitor \longrightarrow$
$\Big($ **Skip**;
$getAltitudeCall \,.\, mainMission \longrightarrow$
$getAltitudeRet \,.\, mainMission \,?\, getAltitude \longrightarrow$

var $distance : \mathbb{R} \bullet distance := getAltitude$
if $(distance = readingOnGround) \longrightarrow$
$\Big($ **Skip**;
$requestTerminationCall \,.\, mainMission \longrightarrow$
$requestTerminationRet \,.\, mainMission \,?\, requestTermination \longrightarrow$
Skip $\Big)$
$[\!] \neg (distance = readingOnGround) \longrightarrow$ **Skip**
fi ;
Skip $\Big)$;
$handleAsyncEventRet \,.\, GroundDistanceMonitor \longrightarrow$
Skip $\Big)$

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

$\bullet$ $(Methods) \triangle (end\_periodic\_app \,.\, GroundDistanceMonitor \longrightarrow$ **Skip**$)$

end

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

state *State*

$mainMission : MainMission$

$readingOnGround : \mathbb{R}$

state *State*

initial *Init*

$State'$

- **Skip**

end

section *InstrumentLandingSystemMonitorApp* **parents** *PeriodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*

process *InstrumentLandingSystemMonitorApp* $\widehat{=}$
$\quad$ *mission* : *MissionID* • **begin**

$$
\begin{array}{l}
handlerAsyncEvent \mathrel{\widehat{=}} \\
\left(\begin{array}{l}
handleAsyncEventCall\,.\,InstrumentLandingSystemMonitor \longrightarrow \\
(\mathbf{Skip})\,; \\
handleAsyncEventRet\,.\,InstrumentLandingSystemMonitor \longrightarrow \\
\mathbf{Skip}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
Methods \mathrel{\widehat{=}} \\
(handlerAsyncEvent)\,;\ Methods
\end{array}
$$

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

end

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

state *State*

$mission : LandMission$

state *State*

initial *Init*

$State'$

- **Skip**

end