aircraft

Tight Rope v0.65

9th March 2016

1 ID Files

1.1 MissionIds

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

$$MainMissionMID : MissionID$$
$$TakeOffMissionMID : MissionID$$
$$CruiseMissionMID : MissionID$$
$$LandMissionMID : MissionID$$

$$distinct\langle nullMissionId, MainMissionMID, TakeOffMissionMID, CruiseMissionMID, LandMissionMID\rangle$$

1.2 SchedulablesIds

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

$$
\begin{array}{l}
MainMissionSequencerSID : SchedulableID \\
ACModeChangerSID : SchedulableID \\
EnvironmentMonitorSID : SchedulableID \\
ControlHandlerSID : SchedulableID \\
FlightSensorsMonitorSID : SchedulableID \\
CommunicationsHandlerSID : SchedulableID \\
AperiodicSimulatorSID : SchedulableID \\
LandingGearHandlerTakeOffSID : SchedulableID \\
TakeOffMonitorSID : SchedulableID \\
TakeOffFailureHandlerSID : SchedulableID \\
BeginLandingHandlerSID : SchedulableID \\
NavigationMonitorSID : SchedulableID \\
GroundDistanceMonitorSID : SchedulableID \\
LandingGearHandlerLandSID : SchedulableID \\
InstrumentLandingSystemMonitorSID : SchedulableID \\
SafeLandingHandlerSID : SchedulableID \\
\hline
distinct\langle nullSequencerId, nullSchedulableId, MainMissionSequencerSID, \\
ACModeChangerSID, EnvironmentMonitorSID, \\
ControlHandlerSID, FlightSensorsMonitorSID, \\
CommunicationsHandlerSID, AperiodicSimulatorSID, \\
LandingGearHandlerTakeOffSID, TakeOffMonitorSID, \\
TakeOffFailureHandlerSID, BeginLandingHandlerSID, \\
NavigationMonitorSID, GroundDistanceMonitorSID, \\
LandingGearHandlerLandSID, InstrumentLandingSystemMonitorSID, \\
SafeLandingHandlerSID\rangle
\end{array}
$$

1.3 ThreadIds

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

$$
\begin{array}{l}
InstrumentLandingSystemMonitorTID : ThreadID \\
SafeLandingHandlerTID : ThreadID \\
GroundDistanceMonitorTID : ThreadID \\
CommunicationsHandlerTID : ThreadID \\
ControlHandlerTID : ThreadID \\
AperiodicSimulatorTID : ThreadID \\
TakeOffFailureHandlerTID : ThreadID \\
LandingGearHandlerLandTID : ThreadID \\
EnvironmentMonitorTID : ThreadID \\
FlightSensorsMonitorTID : ThreadID \\
NavigationMonitorTID : ThreadID \\
ACModeChangerTID : ThreadID \\
BeginLandingHandlerTID : ThreadID \\
LandingGearHandlerTakeOffTID : ThreadID \\
TakeOffMonitorTID : ThreadID
\end{array}
$$

$$
\begin{array}{l}
distinct\langle SafeletTId, nullThreadId, \\
InstrumentLandingSystemMonitorTID, SafeLandingHandlerTID, \\
GroundDistanceMonitorTID, CommunicationsHandlerTID, \\
ControlHandlerTID, AperiodicSimulatorTID, \\
TakeOffFailureHandlerTID, LandingGearHandlerLandTID, \\
EnvironmentMonitorTID, FlightSensorsMonitorTID, \\
NavigationMonitorTID, ACModeChangerTID, \\
BeginLandingHandlerTID, LandingGearHandlerTakeOffTID, \\
TakeOffMonitorTID\rangle
\end{array}
$$

1.4 ObjectIds

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

$$TakeOffMissionOID : ObjectID$$
$$LandMissionOID : ObjectID$$

$$distinct\langle TakeOffMissionOID, LandMissionOID\rangle$$

2 Network

2.1 Network Channel Sets

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

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

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

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

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

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

channelset *AppSync* ==
{| *getSequencerCall*, *getSequencerRet*, *initializeApplicationCall*, *initializeApplicationRet*, *end_safelet_app*,
initializeCall, *initializeRet*, *register*, *cleanupMissionCall*, *cleanupMissionRet*, *end_mission_app*,
getNextMissionCall, *getNextMissionRet*, *end_sequencer_app*,
handleAsyncEventCall, *handleAsyncEventRet*, *end_periodic_app*,
handleAsyncLongEventCall, *handleAsyncLongEventRet*, *end_aperiodic_app*,
descheduleCall, *descheduleRet*, *scheduleNextRelease*, *getNextReleaseTimeCall*, *getNextReleaseTimeRet*, *end_oneShot_*
runCall, *runRet*, *end_managedThread_app*,
setCeilingPriority, *requestTerminationCall*, *requestTerminationRet*, *terminationPendingCall*, *terminationPendingRet*
done_safeletFW, *done_toplevel_sequencer*, *signalTerminationCall*, *signalTerminationRet*,
activate_schedulables, *done_schedulable*, *cleanupSchedulableCall*, *cleanupSchedulableRet* |}

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* |}

2.2 MethodCallBinder

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

channel $binder\_setCabinPressureCall : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$
channel $binder\_setCabinPressureRet : MissionID \times SchedulableID$

$setCabinPressureLocs == \{MainMissionMID\}$
$setCabinPressureCallers == \{EnvironmentMonitorSID\}$

channel $binder\_setEmergencyOxygenCall : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$
channel $binder\_setEmergencyOxygenRet : MissionID \times SchedulableID$

$setEmergencyOxygenLocs == \{MainMissionMID\}$
$setEmergencyOxygenCallers == \{EnvironmentMonitorSID\}$

channel $binder\_setFuelRemainingCall : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$
channel $binder\_setFuelRemainingRet : MissionID \times SchedulableID$

$setFuelRemainingLocs == \{MainMissionMID\}$
$setFuelRemainingCallers == \{EnvironmentMonitorSID\}$

channel $binder\_setAirSpeedCall : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$
channel $binder\_setAirSpeedRet : MissionID \times SchedulableID$

$setAirSpeedLocs == \{MainMissionMID\}$
$setAirSpeedCallers == \{FlightSensorsMonitorSID\}$

channel $binder\_setAltitudeCall : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$
channel $binder\_setAltitudeRet : MissionID \times SchedulableID$

$setAltitudeLocs == \{MainMissionMID\}$
$setAltitudeCallers == \{FlightSensorsMonitorSID\}$

channel $binder\_setHeadingCall : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$
channel $binder\_setHeadingRet : MissionID \times SchedulableID$

$setHeadingLocs == \{MainMissionMID\}$
$setHeadingCallers == \{FlightSensorsMonitorSID\}$

channel $binder\_isLandingGearDeployedCall : MissionID \times SchedulableID$
channel $binder\_isLandingGearDeployedRet : MissionID \times SchedulableID \times \mathbb{B}$

$isLandingGearDeployedLocs == \{TakeOffMissionMID\}$
$isLandingGearDeployedCallers == \{LandingGearHandlerTakeOffSID\}$

channel $binder\_stowLandingGearCall : MissionID \times SchedulableID$
channel $binder\_stowLandingGearRet : MissionID \times SchedulableID$

$stowLandingGearLocs == \{TakeOffMissionMID\}$
$stowLandingGearCallers == \{LandingGearHandlerTakeOffSID\}$

channel $binder\_deployLandingGearCall : MissionID \times SchedulableID \times ThreadID$
channel $binder\_deployLandingGearRet : MissionID \times SchedulableID \times ThreadID$

$deployLandingGearLocs == \{TakeOffMissionMID\}$
$deployLandingGearCallers == \{LandingGearHandlerTakeOffSID\}$

channel $binder\_getAltitudeCall : MissionID \times SchedulableID$
channel $binder\_getAltitudeRet : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$

$getAltitudeLocs == \{MainMissionMID\}$
$getAltitudeCallers == \{NavigationMonitorSID, TakeOffMonitorSID, GroundDistanceMonitorSID, SafeLandingHandlerS$

channel $binder\_getAirSpeedCall : MissionID \times SchedulableID$
channel $binder\_getAirSpeedRet : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$

$getAirSpeedLocs == \{MainMissionMID\}$
$getAirSpeedCallers == \{NavigationMonitorSID, TakeOffFailureHandlerSID\}$

channel $binder\_abortCall : MissionID \times SchedulableID$
channel $binder\_abortRet : MissionID \times SchedulableID$

$abortLocs == \{TakeOffMissionMID\}$
$abortCallers == \{TakeOffFailureHandlerSID\}$

channel $binder\_getHeadingCall : MissionID \times SchedulableID$
channel $binder\_getHeadingRet : MissionID \times SchedulableID \times \mathbb{P}\,\mathbb{A}$

$getHeadingLocs == \{MainMissionMID\}$
$getHeadingCallers == \{NavigationMonitorSID\}$

channel $binder\_isLandingGearDeployedCall : MissionID \times SchedulableID$
channel $binder\_isLandingGearDeployedRet : MissionID \times SchedulableID \times \mathbb{B}$

$isLandingGearDeployedLocs == \{LandMissionMID\}$
$isLandingGearDeployedCallers == \{LandingGearHandlerLandSID\}$

channel $binder\_stowLandingGearCall : MissionID \times SchedulableID$
channel $binder\_stowLandingGearRet : MissionID \times SchedulableID$

$stowLandingGearLocs == \{LandMissionMID\}$
$stowLandingGearCallers == \{LandingGearHandlerLandSID\}$

channel $binder\_deployLandingGearCall : MissionID \times SchedulableID \times ThreadID$
channel $binder\_deployLandingGearRet : MissionID \times SchedulableID \times ThreadID$

$deployLandingGearLocs == \{LandMissionMID\}$
$deployLandingGearCallers == \{LandingGearHandlerLandSID\}$

channelset $MethodCallBinderSync == \{\!|\ done\_toplevel\_sequencer,$
$binder\_setCabinPressureCall, binder\_setCabinPressureRet,$
$binder\_setEmergencyOxygenCall, binder\_setEmergencyOxygenRet,$
$binder\_setFuelRemainingCall, binder\_setFuelRemainingRet,$
$binder\_setAirSpeedCall, binder\_setAirSpeedRet,$
$binder\_setAltitudeCall, binder\_setAltitudeRet,$
$binder\_setHeadingCall, binder\_setHeadingRet,$
$binder\_isLandingGearDeployedCall, binder\_isLandingGearDeployedRet,$
$binder\_stowLandingGearCall, binder\_stowLandingGearRet,$
$binder\_deployLandingGearCall, binder\_deployLandingGearRet,$
$binder\_getAltitudeCall, binder\_getAltitudeRet,$
$binder\_getAirSpeedCall, binder\_getAirSpeedRet,$
$binder\_abortCall, binder\_abortRet,$
$binder\_getHeadingCall, binder\_getHeadingRet,$
$binder\_isLandingGearDeployedCall, binder\_isLandingGearDeployedRet,$
$binder\_stowLandingGearCall, binder\_stowLandingGearRet,$
$binder\_deployLandingGearCall, binder\_deployLandingGearRet\ |\!\}$

section $MethodCallBinder$ **parents** $scj\_prelude, MissionId, MissionIds,$
$SchedulableId, SchedulableIds, MethodCallBindingChannels$
$, MainMissionMethChan, TakeOffMissionMethChan, LandMissionMethChan$

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

$setCabinPressure\_MethodBinder \mathrel{\widehat=}$
$\Big($ $binder\_setCabinPressureCall\,?\,loc : (loc \in setCabinPressureLocs)\,?\,caller : (caller \in setCabinPressureCallers)\,?\,p1-$
$setCabinPressureCall\,.\,loc\,.\,caller\,!\,p1 \longrightarrow$
$setCabinPressureRet\,.\,loc\,.\,caller \longrightarrow$
$binder\_setCabinPressureRet\,.\,loc\,.\,caller \longrightarrow$
$setCabinPressure\_MethodBinder$

$setEmergencyOxygen\_MethodBinder \mathrel{\widehat=}$
$\Big($ $binder\_setEmergencyOxygenCall\,?\,loc : (loc \in setEmergencyOxygenLocs)\,?\,caller : (caller \in setEmergencyOxygenC$
$setEmergencyOxygenCall\,.\,loc\,.\,caller\,!\,p1 \longrightarrow$
$setEmergencyOxygenRet\,.\,loc\,.\,caller \longrightarrow$
$binder\_setEmergencyOxygenRet\,.\,loc\,.\,caller \longrightarrow$
$setEmergencyOxygen\_MethodBinder$

$setFuelRemaining\_MethodBinder \mathrel{\widehat=}$
$\Big($ $binder\_setFuelRemainingCall\,?\,loc : (loc \in setFuelRemainingLocs)\,?\,caller : (caller \in setFuelRemainingCallers)\,?\,p1$
$setFuelRemainingCall\,.\,loc\,.\,caller\,!\,p1 \longrightarrow$
$setFuelRemainingRet\,.\,loc\,.\,caller \longrightarrow$
$binder\_setFuelRemainingRet\,.\,loc\,.\,caller \longrightarrow$
$setFuelRemaining\_MethodBinder$

$$setAirSpeed\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_setAirSpeedCall\,?\,loc : (loc \in setAirSpeedLocs)\,?\,caller : (caller \in setAirSpeedCallers)\,?\,p1 \longrightarrow \\ setAirSpeedCall\,.\,loc\,.\,caller\,!\,p1 \longrightarrow \\ setAirSpeedRet\,.\,loc\,.\,caller \longrightarrow \\ binder\_setAirSpeedRet\,.\,loc\,.\,caller \longrightarrow \\ setAirSpeed\_MethodBinder \end{array}\right)$$

$$setAltitude\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_setAltitudeCall\,?\,loc : (loc \in setAltitudeLocs)\,?\,caller : (caller \in setAltitudeCallers)\,?\,p1 \longrightarrow \\ setAltitudeCall\,.\,loc\,.\,caller\,!\,p1 \longrightarrow \\ setAltitudeRet\,.\,loc\,.\,caller \longrightarrow \\ binder\_setAltitudeRet\,.\,loc\,.\,caller \longrightarrow \\ setAltitude\_MethodBinder \end{array}\right)$$

$$setHeading\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_setHeadingCall\,?\,loc : (loc \in setHeadingLocs)\,?\,caller : (caller \in setHeadingCallers)\,?\,p1 \longrightarrow \\ setHeadingCall\,.\,loc\,.\,caller\,!\,p1 \longrightarrow \\ setHeadingRet\,.\,loc\,.\,caller \longrightarrow \\ binder\_setHeadingRet\,.\,loc\,.\,caller \longrightarrow \\ setHeading\_MethodBinder \end{array}\right)$$

$$isLandingGearDeployed\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_isLandingGearDeployedCall\,?\,loc : (loc \in isLandingGearDeployedLocs)\,?\,caller : (caller \in isLandingGearDep \\ isLandingGearDeployedCall\,.\,loc\,.\,caller \longrightarrow \\ isLandingGearDeployedRet\,.\,loc\,.\,caller\,?\,ret \longrightarrow \\ binder\_isLandingGearDeployedRet\,.\,loc\,.\,caller\,!\,ret \longrightarrow \\ isLandingGearDeployed\_MethodBinder \end{array}\right)$$

$$stowLandingGear\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_stowLandingGearCall\,?\,loc : (loc \in stowLandingGearLocs)\,?\,caller : (caller \in stowLandingGearCallers) \longrightarrow \\ stowLandingGearCall\,.\,loc\,.\,caller \longrightarrow \\ stowLandingGearRet\,.\,loc\,.\,caller \longrightarrow \\ binder\_stowLandingGearRet\,.\,loc\,.\,caller \longrightarrow \\ stowLandingGear\_MethodBinder \end{array}\right)$$

$$deployLandingGear\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_deployLandingGearCall\,?\,loc : (loc \in deployLandingGearLocs)\,?\,caller : (caller \in deployLandingGearCallers) \\ deployLandingGearCall\,.\,loc\,.\,caller\,.\,callingThread \longrightarrow \\ deployLandingGearRet\,.\,loc\,.\,caller\,.\,callingThread \longrightarrow \\ binder\_deployLandingGearRet\,.\,loc\,.\,caller\,.\,callingThread \longrightarrow \\ deployLandingGear\_MethodBinder \end{array}\right)$$

$$getAltitude\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_getAltitudeCall\,?\,loc : (loc \in getAltitudeLocs)\,?\,caller : (caller \in getAltitudeCallers) \longrightarrow \\ getAltitudeCall\,.\,loc\,.\,caller \longrightarrow \\ getAltitudeRet\,.\,loc\,.\,caller\,?\,ret \longrightarrow \\ binder\_getAltitudeRet\,.\,loc\,.\,caller\,!\,ret \longrightarrow \\ getAltitude\_MethodBinder \end{array}\right)$$

$$getAirSpeed\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_getAirSpeedCall\,?\,loc : (loc \in getAirSpeedLocs)\,?\,caller : (caller \in getAirSpeedCallers) \longrightarrow \\ getAirSpeedCall\,.\,loc\,.\,caller \longrightarrow \\ getAirSpeedRet\,.\,loc\,.\,caller\,?\,ret \longrightarrow \\ binder\_getAirSpeedRet\,.\,loc\,.\,caller\,!\,ret \longrightarrow \\ getAirSpeed\_MethodBinder \end{array}\right)$$

$$abort\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_abortCall\,?\,loc : (loc \in abortLocs)\,?\,caller : (caller \in abortCallers) \longrightarrow \\ abortCall\,.\,loc\,.\,caller \longrightarrow \\ abortRet\,.\,loc\,.\,caller \longrightarrow \\ binder\_abortRet\,.\,loc\,.\,caller \longrightarrow \\ abort\_MethodBinder \end{array}\right)$$

$$getHeading\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_getHeadingCall\,?\,loc : (loc \in getHeadingLocs)\,?\,caller : (caller \in getHeadingCallers) \longrightarrow \\ getHeadingCall\,.\,loc\,.\,caller \longrightarrow \\ getHeadingRet\,.\,loc\,.\,caller\,?\,ret \longrightarrow \\ binder\_getHeadingRet\,.\,loc\,.\,caller\,!\,ret \longrightarrow \\ getHeading\_MethodBinder \end{array}\right)$$

$$isLandingGearDeployed\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_isLandingGearDeployedCall\,?\,loc : (loc \in isLandingGearDeployedLocs)\,?\,caller : (caller \in isLandingGearDep \\ isLandingGearDeployedCall\,.\,loc\,.\,caller \longrightarrow \\ isLandingGearDeployedRet\,.\,loc\,.\,caller\,?\,ret \longrightarrow \\ binder\_isLandingGearDeployedRet\,.\,loc\,.\,caller\,!\,ret \longrightarrow \\ isLandingGearDeployed\_MethodBinder \end{array}\right)$$

$$stowLandingGear\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_stowLandingGearCall\,?\,loc : (loc \in stowLandingGearLocs)\,?\,caller : (caller \in stowLandingGearCallers) \longrightarrow \\ stowLandingGearCall\,.\,loc\,.\,caller \longrightarrow \\ stowLandingGearRet\,.\,loc\,.\,caller \longrightarrow \\ binder\_stowLandingGearRet\,.\,loc\,.\,caller \longrightarrow \\ stowLandingGear\_MethodBinder \end{array}\right)$$

$$deployLandingGear\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_deployLandingGearCall\,?\,loc : (loc \in deployLandingGearLocs)\,?\,caller : (caller \in deployLandingGearCallers) \\ deployLandingGearCall\,.\,loc\,.\,caller\,.\,callingThread \longrightarrow \\ deployLandingGearRet\,.\,loc\,.\,caller\,.\,callingThread \longrightarrow \\ binder\_deployLandingGearRet\,.\,loc\,.\,caller\,.\,callingThread \longrightarrow \\ deployLandingGear\_MethodBinder \end{array}\right)$$

$$
\begin{array}{l}
BinderActions \mathrel{\widehat{=}} \\
\left(\begin{array}{l}
setCabinPressure\_MethodBinder \\
|||\\
setEmergencyOxygen\_MethodBinder \\
|||\\
setFuelRemaining\_MethodBinder \\
|||\\
setAirSpeed\_MethodBinder \\
|||\\
setAltitude\_MethodBinder \\
|||\\
setHeading\_MethodBinder \\
|||\\
isLandingGearDeployed\_MethodBinder \\
|||\\
stowLandingGear\_MethodBinder \\
|||\\
deployLandingGear\_MethodBinder \\
|||\\
getAltitude\_MethodBinder \\
|||\\
getAirSpeed\_MethodBinder \\
|||\\
abort\_MethodBinder \\
|||\\
getHeading\_MethodBinder \\
|||\\
isLandingGearDeployed\_MethodBinder \\
|||\\
stowLandingGear\_MethodBinder \\
|||\\
deployLandingGear\_MethodBinder
\end{array}\right)
\end{array}
$$

- $BinderActions \triangle (done\_toplevel\_sequencer \longrightarrow \mathbf{Skip})$

end

2.3 Locking

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

process $Threads \mathrel{\widehat=}$
$$\left(\begin{array}{l} ThreadFW(InstrumentLandingSystemMonitorTID, 5) \\ ||| \\ ThreadFW(SafeLandingHandlerTID, 5) \\ ||| \\ ThreadFW(GroundDistanceMonitorTID, 5) \\ ||| \\ ThreadFW(CommunicationsHandlerTID, 5) \\ ||| \\ ThreadFW(ControlHandlerTID, 5) \\ ||| \\ ThreadFW(AperiodicSimulatorTID, 5) \\ ||| \\ ThreadFW(TakeOffFailureHandlerTID, 5) \\ ||| \\ ThreadFW(LandingGearHandlerLandTID, 5) \\ ||| \\ ThreadFW(EnvironmentMonitorTID, 5) \\ ||| \\ ThreadFW(FlightSensorsMonitorTID, 5) \\ ||| \\ ThreadFW(NavigationMonitorTID, 5) \\ ||| \\ ThreadFW(ACModeChangerTID, 5) \\ ||| \\ ThreadFW(BeginLandingHandlerTID, 5) \\ ||| \\ ThreadFW(LandingGearHandlerTakeOffTID, 5) \\ ||| \\ ThreadFW(TakeOffMonitorTID, 5) \end{array}\right)$$

process $Objects \mathrel{\widehat=}$
$$\left(\begin{array}{l} ObjectFW(TakeOffMissionOID) \\ ||| \\ ObjectFW(LandMissionOID) \end{array}\right)$$

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

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

process $Framework \mathrel{\widehat=}$
$$\left(\begin{array}{l} ControlTier \\ \quad [\![TierSync]\!] \\ \left(\begin{array}{l} Tier0 \\ \quad [\![Tier0Sync]\!] \\ Tier1 \end{array}\right) \end{array}\right)$$

process $Application \mathrel{\widehat=}$
$$\left(\begin{array}{l} ACSafeletApp \\ ||| \\ MainMissionSequencerApp \\ ||| \\ MainMissionApp \\ ||| \\ ACModeChangerApp(MainMissionID) \\ ||| \\ ControlHandlerApp \\ ||| \\ CommunicationsHandlerApp \\ ||| \\ EnvironmentMonitorApp(MainMissionID) \\ ||| \\ FlightSensorsMonitorApp(MainMissionID) \\ ||| \\ AperiodicSimulatorApp(controlHandlerID) \\ ||| \\ TakeOffMissionApp \\ ||| \\ LandingGearHandlerTakeOffApp(TakeOffMissionID) \\ ||| \\ TakeOffFailureHandlerApp(MissionID, TakeOffMissionID, 10.0) \\ ||| \\ TakeOffMonitorApp(MissionID, TakeOffMissionID, 10.0, landingGearHandlerID) \\ ||| \\ CruiseMissionApp \\ ||| \\ BeginLandingHandlerApp(MissionID) \\ ||| \\ NavigationMonitorApp(MissionID) \\ ||| \\ LandMissionApp \\ ||| \\ LandingGearHandlerLandApp(LandMissionID) \\ ||| \\ SafeLandingHandlerApp(MissionID, 10.0) \\ ||| \\ GroundDistanceMonitorApp(MissionID) \\ ||| \\ InstrumentLandingSystemMonitorApp(LandMissionID) \end{array}\right)$$

section *Network* **parents** *scj_prelude*,

process $Program \mathrel{\widehat=} \big(Framework \;[\![\; AppSync \;]\!]\; ApplicationB \big) \;[\![\; LockingSync \;]\!]\; Locking$

3 Safelet

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

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

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

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

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

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

end

4 Top Level Mission Sequencer

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

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

State

$this : \textbf{ref}\ MainMissionSequencerClass$

state *State*

Init

$State'$

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

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

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

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

end

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

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

state *State*

$$returnedMission : \mathbb{B}$$

state *State*

initial *Init*

$$State'$$

$$returnedMission' = \mathbf{False}$$

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

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

$\bullet$ **Skip**

end

5 Missions

5.1 MainMission

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

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

State

$this : \mathbf{ref}\ MainMissionClass$

state *State*

Init

$State'$

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

$InitializePhase \mathrel{\widehat=}$
$\left(\begin{array}{l} initializeCall\,.\,MainMissionMID \longrightarrow \\ register\,!\,ACModeChangerSID\,!\,MainMissionMID \longrightarrow \\ register\,!\,EnvironmentMonitorSID\,!\,MainMissionMID \longrightarrow \\ register\,!\,ControlHandlerSID\,!\,MainMissionMID \longrightarrow \\ register\,!\,FlightSensorsMonitorSID\,!\,MainMissionMID \longrightarrow \\ register\,!\,CommunicationsHandlerSID\,!\,MainMissionMID \longrightarrow \\ register\,!\,AperiodicSimulatorSID\,!\,MainMissionMID \longrightarrow \\ initializeRet\,.\,MainMissionMID \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

end

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

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

state *State*

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

state *State*

initial *Init*

$State'$

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

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

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

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

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

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

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

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

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

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

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

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

- **Skip**

end

5.2 Schedulables of MainMission

section *ACModeChangerApp* **parents** *TopLevelMissionSequencerChan*,
MissionId, *MissionIds*, *SchedulableId*, *SchedulableIds*, *ACModeChangerClass*, *MethodCallBindingChannels*

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

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

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

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

end

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

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

state $State$

$controllingMission : MainMission$

$modesLeft : \mathbb{Z}$

state $State$

initial $Init$

$State'$

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

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

• **Skip**

end

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

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

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

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

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

end

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

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

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

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

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

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ *handleAsyncEventCall* . *EnvironmentMonitorSID* $\longrightarrow$
$\Big($ *binder_setCabinPressureCall* . *controllingMission* . *EnvironmentMonitorSID* ! 0 $\longrightarrow$
binder_setCabinPressureRet . *controllingMission* . *EnvironmentMonitorSID* $\longrightarrow$
Skip;
binder_setEmergencyOxygenCall . *controllingMission* . *EnvironmentMonitorSID* ! 0 $\longrightarrow$
binder_setEmergencyOxygenRet . *controllingMission* . *EnvironmentMonitorSID* $\longrightarrow$
Skip;
binder_setFuelRemainingCall . *controllingMission* . *EnvironmentMonitorSID* ! 0 $\longrightarrow$
binder_setFuelRemainingRet . *controllingMission* . *EnvironmentMonitorSID* $\longrightarrow$
Skip
$\Big)$;
handleAsyncEventRet . *EnvironmentMonitorSID* $\longrightarrow$
Skip
$\Big)$

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

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

end

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

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

state *State*

$controllingMission : MainMission$

state *State*

initial *Init*

$State'$

- **Skip**

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ *handleAsyncEventCall* . *FlightSensorsMonitorSID* $\longrightarrow$
$\Big($ *binder_setAirSpeedCall* . *controllingMission* . *FlightSensorsMonitorSID* ! 0 $\longrightarrow$
binder_setAirSpeedRet . *controllingMission* . *FlightSensorsMonitorSID* $\longrightarrow$
Skip;
binder_setAltitudeCall . *controllingMission* . *FlightSensorsMonitorSID* ! 0 $\longrightarrow$
binder_setAltitudeRet . *controllingMission* . *FlightSensorsMonitorSID* $\longrightarrow$
Skip;
binder_setHeadingCall . *controllingMission* . *FlightSensorsMonitorSID* ! 0 $\longrightarrow$
binder_setHeadingRet . *controllingMission* . *FlightSensorsMonitorSID* $\longrightarrow$
Skip $\Big)$;
handleAsyncEventRet . *FlightSensorsMonitorSID* $\longrightarrow$
Skip $\Big)$

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

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

end

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

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

state *State*

$controllingMission : MainMission$

state *State*

initial *Init*

$State'$

- **Skip**

end

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

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

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

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

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

end

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

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*, *TakeOffMissionMethChan*, *TakeOffMissionClass*, *MethodCallBindingChannels*, *ObjectFWChan*, *ObjectIds*

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

State

$this : \mathbf{ref}\ TakeOffMissionClass$

state *State*

Init

$State'$

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

$InitializePhase \mathrel{\widehat=}$
$\left(\begin{array}{l} initializeCall\,.\,TakeOffMissionMID \longrightarrow \\ register\,!\,LandingGearHandlerTakeOffSID\,!\,TakeOffMissionMID \longrightarrow \\ register\,!\,TakeOffMonitorSID\,!\,TakeOffMissionMID \longrightarrow \\ register\,!\,TakeOffFailureHandlerSID\,!\,TakeOffMissionMID \longrightarrow \\ initializeRet\,.\,TakeOffMissionMID \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

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

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

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

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

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

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

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

$$deployLandingGearSyncMeth \mathrel{\widehat=} \left(\begin{array}{l} deployLandingGearCall\ .\ TakeOffMissionMID\ ?\ caller\ ?\ thread \longrightarrow \\ \left(\begin{array}{l} startSyncMeth\ .\ TakeOffMissionOID\ .\ thread \longrightarrow \\ lockAcquired\ .\ TakeOffMissionOID\ .\ thread \longrightarrow \\ \left(this\ .\ landingGearDeployed := \mathbf{True}\right); \\ endSyncMeth\ .\ TakeOffMissionOID\ .\ thread \longrightarrow \\ deployLandingGearRet\ .\ TakeOffMissionMID\ .\ caller\ .\ thread \longrightarrow \\ \mathbf{Skip} \end{array}\right) \end{array}\right)$$

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

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

end

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

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

state *State*

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

state *State*

initial *Init*

$State'$

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

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

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

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

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

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

• **Skip**

end

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

channel $abortCall : MissionID \times SchedulableID$
channel $abortRet : MissionID \times SchedulableID$

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

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

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

channel $stowLandingGearCall : MissionID \times SchedulableID$
channel $stowLandingGearRet : MissionID \times SchedulableID$

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

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

5.4 Schedulables of TakeOffMission

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ *handleAsyncEventCall . LandingGearHandlerTakeOffSID* $\longrightarrow$
$\Big($ *binder_isLandingGearDeployedCall . mission . LandingGearHandlerTakeOffSID* $\longrightarrow$
binder_isLandingGearDeployedRet . mission . LandingGearHandlerTakeOffSID ? isLandingGearDeployed $\longrightarrow$
var $landingGearIsDeployed : \mathbb{B} \bullet landingGearIsDeployed := isLandingGearDeployed$;
if $landingGearIsDeployed =$ **True** $\longrightarrow$
$\Big($ *binder_stowLandingGearCall . mission . LandingGearHandlerTakeOffSID* $\longrightarrow$
binder_stowLandingGearRet . mission . LandingGearHandlerTakeOffSID $\longrightarrow$
Skip $\Big)$
$[]\ \neg\ landingGearIsDeployed =$ **True** $\longrightarrow$
$\Big($ *binder_deployLandingGearCall . mission . LandingGearHandlerTakeOffSID . LandingGearHandlerTakeOffTID-*
binder_deployLandingGearRet . mission . LandingGearHandlerTakeOffSID . LandingGearHandlerTakeOffTID–
Skip
fi
handleAsyncEventRet . LandingGearHandlerTakeOffSID $\longrightarrow$
Skip

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

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

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($
$handleAsyncEventCall\,.\,TakeOffFailureHandlerSID \longrightarrow$
$\Big($
$binder\_getAirSpeedCall\,.\,mainMission\,.\,TakeOffFailureHandlerSID \longrightarrow$
$binder\_getAirSpeedRet\,.\,mainMission\,.\,TakeOffFailureHandlerSID\,?\,getAirSpeed \longrightarrow$
var $currentSpeed : \mathbb{P}\,\mathbb{A} \bullet currentSpeed := getAirSpeed;$
if $(currentSpeed < threshold) \longrightarrow$
$\Big($
$binder\_abortCall\,.\,takeoffMission\,.\,TakeOffFailureHandlerSID \longrightarrow$
$binder\_abortRet\,.\,takeoffMission\,.\,TakeOffFailureHandlerSID \longrightarrow$
Skip;
$requestTerminationCall\,.\,takeoffMission\,.\,TakeOffFailureHandlerSID \longrightarrow$
$requestTerminationRet\,.\,takeoffMission\,.\,TakeOffFailureHandlerSID\,?\,requestTermination \longrightarrow$
Skip
$\Big)$
$[]\ \neg\,(currentSpeed < threshold) \longrightarrow$
Skip
fi
$\Big)$
;
$handleAsyncEventRet\,.\,TakeOffFailureHandlerSID \longrightarrow$
Skip
$\Big)$

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

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

end

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

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

state *State*

$$threshold : \mathbb{P}\,\mathbb{A}$$

state *State*

initial *Init*

$$State'$$

- **Skip**

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall\,.\,TakeOffMonitorSID \longrightarrow$
$\Big($ $binder\_getAltitudeCall\,.\,mainMission\,.\,TakeOffMonitorSID \longrightarrow$
$binder\_getAltitudeRet\,.\,mainMission\,.\,TakeOffMonitorSID\,?\,getAltitude \longrightarrow$
var $altitude : \mathbb{P}\,\mathbb{A} \bullet altitude := getAltitude;$
if $(altitude > takeOffAltitude) \longrightarrow$
$\Big($ $releaseCall\,.\,landingGearHandler \longrightarrow$
$releaseRet\,.\,landingGearHandler\,?\,release \longrightarrow$
Skip;
$requestTerminationCall\,.\,takeoffMission\,.\,TakeOffMonitorSID \longrightarrow$
$requestTerminationRet\,.\,takeoffMission\,.\,TakeOffMonitorSID\,?\,requestTermination \longrightarrow$
Skip $\Big)$
$[]\ \neg\,(altitude > takeOffAltitude) \longrightarrow$ **Skip**
fi $\Big)$;
$handleAsyncEventRet\,.\,TakeOffMonitorSID \longrightarrow$
Skip $\Big)$

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

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

end

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

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

state *State*

$takeoffMission : TakeOffMission$
$takeOffAltitude : \mathbb{P}\,\mathbb{A}$

state *State*

initial *Init*

$State'$

- **Skip**

end

5.5 CruiseMission

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

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

State

$this : \mathbf{ref}\ CruiseMissionClass$

state *State*

Init

$State'$

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

$$InitializePhase \mathrel{\widehat=} \begin{pmatrix} initializeCall\,.\,CruiseMissionMID \longrightarrow \\ register\,!\,BeginLandingHandlerSID\,!\,CruiseMissionMID \longrightarrow \\ register\,!\,NavigationMonitorSID\,!\,CruiseMissionMID \longrightarrow \\ initializeRet\,.\,CruiseMissionMID \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

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

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

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

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

end

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

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

state *State*

$$controllingMission : MainMission$$

state *State*

initial *Init*

$$State'$$

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

$\bullet$ **Skip**

end

5.6 Schedulables of CruiseMission

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

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

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

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

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

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall \,.\, NavigationMonitorSID \longrightarrow$
$\Big($ $binder\_getHeadingCall \,.\, mainMission \,.\, NavigationMonitorSID \longrightarrow$
$binder\_getHeadingRet \,.\, mainMission \,.\, NavigationMonitorSID \,?\, getHeading \longrightarrow$
var $heading : \mathbb{P}\,\mathbb{A} \bullet heading := getHeading$;
$binder\_getAirSpeedCall \,.\, mainMission \,.\, NavigationMonitorSID \longrightarrow$
$binder\_getAirSpeedRet \,.\, mainMission \,.\, NavigationMonitorSID \,?\, getAirSpeed \longrightarrow$
var $airSpeed : \mathbb{P}\,\mathbb{A} \bullet airSpeed := getAirSpeed$;
$binder\_getAltitudeCall \,.\, mainMission \,.\, NavigationMonitorSID \longrightarrow$
$binder\_getAltitudeRet \,.\, mainMission \,.\, NavigationMonitorSID \,?\, getAltitude \longrightarrow$
var $altitude : \mathbb{P}\,\mathbb{A} \bullet altitude := getAltitude$
$\Big)$;
$handleAsyncEventRet \,.\, NavigationMonitorSID \longrightarrow$
Skip
$\Big)$

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

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

end

5.7 LandMission

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

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

State

$this : \mathbf{ref}\ LandMissionClass$

state *State*

Init

$State'$

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

$InitializePhase \mathrel{\widehat=}$
$\left(\begin{array}{l} initializeCall\,.\,LandMissionMID \longrightarrow \\ register\,!\,GroundDistanceMonitorSID\,!\,LandMissionMID \longrightarrow \\ register\,!\,LandingGearHandlerLandSID\,!\,LandMissionMID \longrightarrow \\ register\,!\,InstrumentLandingSystemMonitorSID\,!\,LandMissionMID \longrightarrow \\ register\,!\,SafeLandingHandlerSID\,!\,LandMissionMID \longrightarrow \\ initializeRet\,.\,LandMissionMID \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

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

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

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

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

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

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

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

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

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

end

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

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

state *State*

$$controllingMission : MainMission$$
$$SAFE\_LANDING\_ALTITUDE : \mathbb{P}\,\mathbb{A}$$
$$abort : \mathbb{B}$$
$$landingGearDeployed : \mathbb{B}$$

state *State*

initial *Init*

$$State'$$

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

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

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

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

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

$\bullet$ **Skip**

end

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

channel *stowLandingGearCall* : *MissionID* × *SchedulableID*
channel *stowLandingGearRet* : *MissionID* × *SchedulableID*

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

channel *getControllingMissionCall* : *MissionID*
channel *getControllingMissionRet* : *MissionID* × *MissionID*

channel *abortCall* : *MissionID*
channel *abortRet* : *MissionID*

channel *cleanUpCall* : *MissionID*
channel *cleanUpRet* : *MissionID* × $\mathbb{B}$

channel *deployLandingGearCall* : *MissionID* × *SchedulableID* × *ThreadID*
channel *deployLandingGearRet* : *MissionID* × *SchedulableID* × *ThreadID*

5.8 Schedulables of LandMission

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ *handleAsyncEventCall . LandingGearHandlerLandSID* $\longrightarrow$
$\Big($ *binder_isLandingGearDeployedCall . mission . LandingGearHandlerLandSID* $\longrightarrow$
binder_isLandingGearDeployedRet . mission . LandingGearHandlerLandSID ? isLandingGearDeployed $\longrightarrow$
var $landingGearIsDeployed : \mathbb{B} \bullet landingGearIsDeployed := isLandingGearDeployed$;
if $landingGearIsDeployed =$ **True** $\longrightarrow$
$\Big($ *binder_stowLandingGearCall . mission . LandingGearHandlerLandSID* $\longrightarrow$
binder_stowLandingGearRet . mission . LandingGearHandlerLandSID $\longrightarrow$
Skip $\Big)$
$[]\ \neg\ landingGearIsDeployed =$ **True** $\longrightarrow$
$\Big($ *binder_deployLandingGearCall . mission . LandingGearHandlerLandSID . LandingGearHandlerLandTID* $\longrightarrow$
binder_deployLandingGearRet . mission . LandingGearHandlerLandSID . LandingGearHandlerLandTID $\longrightarrow$
Skip $\Big)$
fi
handleAsyncEventRet . LandingGearHandlerLandSID $\longrightarrow$
Skip $\Big)$

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

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

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall\,.\,SafeLandingHandlerSID \longrightarrow$
$\Big($ $binder\_getAltitudeCall\,.\,mainMission\,.\,SafeLandingHandlerSID \longrightarrow$
$binder\_getAltitudeRet\,.\,mainMission\,.\,SafeLandingHandlerSID\,?\,getAltitude \longrightarrow$
var $altitude : \mathbb{P}\,\mathbb{A} \bullet altitude := getAltitude;$
if $(altitude < threshold) \longrightarrow$
Skip
$[]\ \neg\,(altitude < threshold) \longrightarrow$
Skip
fi
$\Big)$;
$handleAsyncEventRet\,.\,SafeLandingHandlerSID \longrightarrow$
Skip
$\Big)$

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

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

end

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

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

state *State*

$threshold : \mathbb{P}\,\mathbb{A}$

state *State*

initial *Init*

$State'$

- **Skip**

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ *handleAsyncEventCall* . *GroundDistanceMonitorSID* $\longrightarrow$
$\Big($ *binder_getAltitudeCall* . *mainMission* . *GroundDistanceMonitorSID* $\longrightarrow$
binder_getAltitudeRet . *mainMission* . *GroundDistanceMonitorSID* ? *getAltitude* $\longrightarrow$
var $distance : \mathbb{P}\,\mathbb{A} \bullet distance := getAltitude$;
if $(distance = readingOnGround) \longrightarrow$
$\Big($ *requestTerminationCall* . *mainMission* . *GroundDistanceMonitorSID* $\longrightarrow$
requestTerminationRet . *mainMission* . *GroundDistanceMonitorSID* ? *requestTermination* $\longrightarrow$
Skip $\Big)$
$[] \neg (distance = readingOnGround) \longrightarrow$ **Skip**
fi $\Big)$;
handleAsyncEventRet . *GroundDistanceMonitorSID* $\longrightarrow$
Skip $\Big)$

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

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

end

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

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

state *State*

$$readingOnGround : \mathbb{P}\,\mathbb{A}$$

state *State*

initial *Init*

$$State'$$

- **Skip**

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\begin{pmatrix} handleAsyncEventCall\,.\,InstrumentLandingSystemMonitorSID \longrightarrow \\ \textbf{Skip}; \\ handleAsyncEventRet\,.\,InstrumentLandingSystemMonitorSID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$

$Methods \mathrel{\widehat=}$
$(handleAsyncEvent)\ ;\ Methods$

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

end