aircraft

Tight Rope v0.75

28th February 2017

1 ID Files

1.1 MissionIds

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

$$
\begin{array}{|l}
MainMissionMID : MissionID \\
TakeOffMissionMID : MissionID \\
CruiseMissionMID : MissionID \\
LandMissionMID : MissionID \\
\hline
distinct\langle nullMissionId, MainMissionMID, TakeOffMissionMID, \\
CruiseMissionMID, LandMissionMID\rangle
\end{array}
$$

1.2 SchedulablesIds

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

$$
\begin{array}{l}
MainMissionSequencerSID : SchedulableID \\
ACModeChanger2SID : SchedulableID \\
EnvironmentMonitorSID : SchedulableID \\
ControlHandlerSID : SchedulableID \\
FlightSensorsMonitorSID : SchedulableID \\
CommunicationsHandlerSID : SchedulableID \\
LandingGearHandlerSID : SchedulableID \\
TakeOffMonitorSID : SchedulableID \\
TakeOffFailureHandlerSID : SchedulableID \\
BeginLandingHandlerSID : SchedulableID \\
NavigationMonitorSID : SchedulableID \\
GroundDistanceMonitorSID : SchedulableID \\
LandingGearHandlerLandSID : SchedulableID \\
InstrumentLandingSystemMonitorSID : SchedulableID \\
SafeLandingHandlerSID : SchedulableID \\
\hline
distinct\langle nullSequencerId, nullSchedulableId, MainMissionSequencerSID, \\
ACModeChanger2SID, EnvironmentMonitorSID, \\
ControlHandlerSID, FlightSensorsMonitorSID, \\
CommunicationsHandlerSID, LandingGearHandlerSID, \\
TakeOffMonitorSID, TakeOffFailureHandlerSID, \\
BeginLandingHandlerSID, NavigationMonitorSID, \\
GroundDistanceMonitorSID, LandingGearHandlerLandSID, \\
InstrumentLandingSystemMonitorSID, SafeLandingHandlerSID\rangle
\end{array}
$$

1.3 Non-Paradigm Objects

1.4 ThreadIds

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

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

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

1.5 ObjectIds

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

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

2 Network

2.1 Network Channel Sets

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$stowLandingGearLocs == \{TakeOffMissionMID, LandMissionMID\}$
$stowLandingGearCallers == \{LandingGearHandlerSID, LandingGearHandlerLandSID\}$

channel $binder\_takeOffAbortCall : MissionID \times SchedulableID$
channel $binder\_takeOffAbortRet : MissionID \times SchedulableID$

$takeOffAbortLocs == \{TakeOffMissionMID\}$
$takeOffAbortCallers == \{TakeOffFailureHandlerSID\}$

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

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

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

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

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

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

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

$deployLandingGearLocs == \{TakeOffMissionMID, LandMissionMID\}$
$deployLandingGearCallers == \{LandingGearHandlerSID, LandingGearHandlerLandSID\}$

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

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

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

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

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

$isLandingGearDeployedLocs == \{TakeOffMissionMID, LandMissionMID\}$
$isLandingGearDeployedCallers == \{LandingGearHandlerSID, LandingGearHandlerLandSID\}$

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

section *MethodCallBinder* **parents** *scj_prelude*, *MissionId*, *MissionIds*,
SchedulableId, *SchedulableIds*, *MethodCallBindingChannels*
, *MainMissionMethChan*, *LandMissionMethChan*

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

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

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

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

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

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

$takeOffAbort\_MethodBinder \mathrel{\widehat=}$
$(binder\_takeOffAbortCall\,?\,loc : (loc \in takeOffAbortLocs)\,?\,caller : (caller \in takeOffAbortCallers) \longrightarrow$
$takeOffAbortCall\,.\,loc\,.\,caller \longrightarrow$
$takeOffAbortRet\,.\,loc\,.\,caller \longrightarrow$
$binder\_takeOffAbortRet\,.\,loc\,.\,caller \longrightarrow$
$takeOffAbort\_MethodBinder)$

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

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

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

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

$$setEmergencyOxygen\_MethodBinder \mathrel{\widehat{=}} \left(\begin{array}{l} binder\_setEmergencyOxygenCall\,?\,loc : (loc \in setEmergencyOxygenLocs)\,?\,caller : (caller \in setEmergencyOxygenCa \\ setEmergencyOxygenCall\,.\,loc\,.\,caller\,!\,p1 \longrightarrow \\ setEmergencyOxygenRet\,.\,loc\,.\,caller \longrightarrow \\ binder\_setEmergencyOxygenRet\,.\,loc\,.\,caller \longrightarrow \\ setEmergencyOxygen\_MethodBinder \end{array}\right)$$

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

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

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

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

end

2.3 Locking

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

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

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

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

2.4 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*, *ACModeChanger2App*, *ControlHandlerApp*,
CommunicationsHandlerApp, *EnvironmentMonitorApp*, *FlightSensorsMonitorApp*
, *TakeOffMissionApp*, *LandingGearHandlerApp*, *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 $Tier0 \mathrel{\widehat=}$
$$\left(\begin{array}{l} MissionFW(MainMissionID) \\ \quad [\![MissionSync]\!] \\ \left(\begin{array}{l} SchedulableMissionSequencerFW(ACModeChanger2ID) \\ \quad [\![SchedulablesSync]\!] \\ AperiodicEventHandlerFW(ControlHandlerID, aperiodic, (time(10,0), nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ AperiodicEventHandlerFW(CommunicationsHandlerID, aperiodic, (NULL, nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ PeriodicEventHandlerFW(EnvironmentMonitorID, (time(10,0), NULL, NULL, nullSchedulableId)) \\ \quad [\![SchedulablesSync]\!] \\ PeriodicEventHandlerFW(FlightSensorsMonitorID, (time(10,0), NULL, NULL, nullSchedulableId)) \end{array}\right) \end{array}\right)$$

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

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

$$\textbf{process}\ Application \mathrel{\widehat{=}} \left(\begin{array}{l} ACSafeletApp \\ ||| \\ MainMissionSequencerApp \\ ||| \\ MainMissionApp \\ ||| \\ ACModeChanger2App(MainMissionID) \\ ||| \\ ControlHandlerApp \\ ||| \\ CommunicationsHandlerApp \\ ||| \\ EnvironmentMonitorApp(MainMissionID) \\ ||| \\ FlightSensorsMonitorApp(MainMissionID) \\ ||| \\ TakeOffMissionApp \\ ||| \\ LandingGearHandlerApp(TakeOffMissionID) \\ ||| \\ TakeOffFailureHandlerApp(TakeOffMissionID, 10.0) \\ ||| \\ TakeOffMonitorApp(TakeOffMissionID, 10.0, landingGearHandlerID) \\ ||| \\ CruiseMissionApp \\ ||| \\ BeginLandingHandlerApp \\ ||| \\ NavigationMonitorApp \\ ||| \\ LandMissionApp \\ ||| \\ LandingGearHandlerLandApp(LandMissionID) \\ ||| \\ SafeLandingHandlerApp(10.0) \\ ||| \\ GroundDistanceMonitorApp(0.0) \\ ||| \\ InstrumentLandingSystemMonitorApp(LandMissionID) \end{array}\right)$$

$$\textbf{process}\ Bound\_Application \mathrel{\widehat{=}} Application\ [\![\ MethodCallBinderSync\ ]\!]\ MethodCallBinder$$
$$\textbf{process}\ Program \mathrel{\widehat{=}} \left(Framework\ [\![\ AppSync\ ]\!]\ Bound\_Application\right)\ [\![\ LockingSync\ ]\!]\ Locking$$

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*

$$State'$$

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

protected *getNextMission* $\widehat{=}$

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

- **Skip**

end

5 Missions

5.1 MainMission

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

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

State

$this : \mathbf{ref}\ MainMissionClass$

state *State*

Init

$State'$

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

$InitializePhase \mathrel{\widehat=}$
$\left(\begin{array}{l} initializeCall\,.\,MainMissionMID \longrightarrow \\ register\,!\,ACModeChanger2SID\,!\,MainMissionMID \longrightarrow \\ register\,!\,EnvironmentMonitorSID\,!\,MainMissionMID \longrightarrow \\ register\,!\,ControlHandlerSID\,!\,MainMissionMID \longrightarrow \\ register\,!\,FlightSensorsMonitorSID\,!\,MainMissionMID \longrightarrow \\ register\,!\,CommunicationsHandlerSID\,!\,MainMissionMID \longrightarrow \\ initializeRet\,.\,MainMissionMID \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

end

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

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

state *State*

$$
\begin{array}{l}
cabinPressure : \mathbb{P}\,\mathbb{A} \\
emergencyOxygen : \mathbb{P}\,\mathbb{A} \\
fuelRemaining : \mathbb{P}\,\mathbb{A} \\
altitude : \mathbb{P}\,\mathbb{A} \\
airSpeed : \mathbb{P}\,\mathbb{A} \\
heading : \mathbb{P}\,\mathbb{A}
\end{array}
$$

state *State*

initial *Init*

$$State'$$

public *getAirSpeed* $\widehat{=}$
$(ret := airSpeed)$

public *getAltitude* $\widehat{=}$
$(ret := altitude)$

public *getCabinPressure* $\widehat{=}$
$(ret := cabinPressure)$

public *getEmergencyOxygen* $\widehat{=}$
$(ret := emergencyOxygen)$

public *getFuelRemaining* $\widehat{=}$
$(ret := fuelRemaining)$

public *getHeading* $\widehat{=}$
$(ret := heading)$

public *setAirSpeed* $\widehat{=}$ **var** $newAirSpeed : \mathbb{P}\,\mathbb{A} \bullet$

$(airSpeed := newAirSpeed)$

public *setAltitude* $\widehat{=}$ **var** $newAltitude : \mathbb{P}\,\mathbb{A} \bullet$

$(altitude := newAltitude)$

public *setCabinPressure* $\widehat{=}$ **var** $newCabinPressure : \mathbb{P}\,\mathbb{A} \bullet$

$(cabinPressure := newCabinPressure)$

public $setEmergencyOxygen \mathrel{\widehat=} \textbf{var}\ newEmergencyOxygen : \mathbb{P}\,\mathbb{A} \bullet$

$\big(emergencyOxygen := newEmergencyOxygen\big)$

public $setFuelRemaining \mathrel{\widehat=} \textbf{var}\ newFuelRemaining : \mathbb{P}\,\mathbb{A} \bullet$

$\big(fuelRemaining := newFuelRemaining\big)$

public $setHeading \mathrel{\widehat=} \textbf{var}\ newHeading : \mathbb{P}\,\mathbb{A} \bullet$

$\big(heading := newHeading\big)$

- **Skip**

end

5.2 Schedulables of MainMission

section $ACModeChanger2App$ **parents** $TopLevelMissionSequencerChan$,
$MissionId$, $MissionIds$, $SchedulableId$, $SchedulableIds$, $ACModeChanger2Class$, $MethodCallBindingChannels$

process $ACModeChanger2App \mathrel{\widehat=}$
$controllingMission : MissionID \bullet$ **begin**

$$GetNextMission \mathrel{\widehat=} \textbf{var}\ ret : MissionID \bullet \left(\begin{array}{l} getNextMissionCall\,.\,ACModeChanger2SID \longrightarrow \\ ret := this\,.\,getNextMission(); \\ getNextMissionRet\,.\,ACModeChanger2SID\,!\,ret \longrightarrow \\ \textbf{Skip} \end{array} \right)$$

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

$$\bullet\ (Methods) \triangle (end\_sequencer\_app\,.\,ACModeChanger2SID \longrightarrow \textbf{Skip})$$

end

section $ACModeChanger2Class$ **parents** $scj\_prelude$, $SchedulableId$, $SchedulableIds$, $SafeletChan$, $MethodCallBindingChannels$, $MissionId$, $MissionIds$

class $ACModeChanger2Class \mathrel{\widehat=}$ **begin**

state $State$

$$modesLeft : \mathbb{Z}$$

state $State$

initial $Init$

$$State'$$

$$modesLeft' = 3$$

protected $getNextMission \mathrel{\widehat=}$

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

- **Skip**

end

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

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

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

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

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

end

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

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

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

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

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

end

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall\,.\,EnvironmentMonitorSID \longrightarrow$
$\Big($ **Skip**;
$binder\_setCabinPressureCall\,.\,controllingMission\,.\,EnvironmentMonitorSID\,!\,0 \longrightarrow$
$binder\_setCabinPressureRet\,.\,controllingMission\,.\,EnvironmentMonitorSID \longrightarrow$
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 $\Big)$

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

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

end

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

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

State

$controllingMission : MainMission$

state *State*

Init

$State'$

$controllingMission' =$

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall \,.\, FlightSensorsMonitorSID \longrightarrow$
$\Big($ **Skip**;
$binder\_setAirSpeedCall \,.\, controllingMission \,.\, FlightSensorsMonitorSID \,!\, 0 \longrightarrow$
$binder\_setAirSpeedRet \,.\, controllingMission \,.\, FlightSensorsMonitorSID \longrightarrow$
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 $\Big)$

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

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

end

5.3 TakeOffMission

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

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

State

$this : \mathbf{ref}\ TakeOffMissionClass$

state *State*

Init

$State'$

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

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

$CleanupPhase \mathrel{\widehat=} \mathbf{var}\ \mathbb{B} : ret \bullet$
$\left(\begin{array}{l} cleanupMissionCall\,.\,TakeOffMissionMID \longrightarrow \\ \left(\begin{array}{l} \mathbf{Skip}; \\ ret := (\neg\ this\,.\,abort) \end{array}\right) \\ cleanupMissionRet\,.\,TakeOffMissionMID\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$takeOffAbortMeth \mathrel{\widehat=}$
$\left(\begin{array}{l} takeOffAbortCall\,.\,TakeOffMissionMID\,?\,caller \longrightarrow \\ this\,.\,takeOffAbort(); \\ takeOffAbortRet\,.\,TakeOffMissionMID\,.\,caller \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$deployLandingGearMeth \mathrel{\widehat=}$
$\left(\begin{array}{l} deployLandingGearCall\,.\,TakeOffMissionMID\,?\,caller \longrightarrow \\ this\,.\,deployLandingGear(); \\ deployLandingGearRet\,.\,TakeOffMissionMID\,.\,caller \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

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

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

$$Methods \mathrel{\widehat=} \left(\begin{array}{l} InitializePhase \\ \Box \\ CleanupPhase \\ \Box \\ takeOffAbortMeth \\ \Box \\ deployLandingGearMeth \\ \Box \\ stowLandingGearMeth \\ \Box \\ isLandingGearDeployedMeth \end{array}\right) ;\ Methods$$

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

end

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

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

state *State*

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

state *State*

initial *Init*

$State'$

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

public *takeOffAbort* $\widehat{=}$
$\big(abort := \mathbf{True}\big)$

public *deployLandingGear* $\widehat{=}$
$\big(landingGearDeployed := \mathbf{True}\big)$

public *stowLandingGear* $\widehat{=}$
$\big(landingGearDeployed := \mathbf{False}\big)$

public *isLandingGearDeployed* $\widehat{=}$
$\big(ret := landingGearDeployed\big)$

- **Skip**

end

5.4 Schedulables of TakeOffMission

section *LandingGearHandlerApp* **parents** *AperiodicEventHandlerChan*, *SchedulableId*, *SchedulableIds*, *MethodCallBindi*
, *TakeOffMissionMethChan*

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

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

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

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

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

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

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

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**

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall\,.\,TakeOffMonitorSID \longrightarrow$
$\Big($ **Skip**;
$binder\_getAltitudeCall\,.\,mainMission\,.\,TakeOffMonitorSID \longrightarrow$
$binder\_getAltitudeRet\,.\,mainMission\,.\,TakeOffMonitorSID\,?\,getAltitude \longrightarrow$
Skip ; **var** $altitude : \mathbb{P}\,\mathbb{A} \bullet altitude := getAltitude$;
if $(altitude > takeOffAltitude) \longrightarrow$
$\Big($ **Skip**;
$release\,.\,landingGearHandler \longrightarrow$
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*

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

state *State*

initial *Init*

$$State'$$

- **Skip**

end

5.5 CruiseMission

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

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

State

$this : \mathbf{ref}\ CruiseMissionClass$

state *State*

Init

$State'$

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

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

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

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

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

end

5.6 Schedulables of CruiseMission

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

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

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall\,.\,BeginLandingHandlerSID \longrightarrow$
$\Big($ **Skip**;
$requestTerminationCall\,.\,controllingMission\,.\,BeginLandingHandlerSID \longrightarrow$
$requestTerminationRet\,.\,controllingMission\,.\,BeginLandingHandlerSID\,?\,requestTermination \longrightarrow$ $\Big)$;
Skip
$handleAsyncEventRet\,.\,BeginLandingHandlerSID \longrightarrow$
Skip $\Big)$

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

$\bullet\ (Methods) \triangle (end\_aperiodic\_app\,.\,BeginLandingHandlerSID \longrightarrow$ **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* $\longrightarrow$
Skip ; **var** *heading* : $\mathbb{P}\,\mathbb{A} \bullet$ *heading* := *getHeading*;
binder_getAirSpeedCall . *mainMission* . *NavigationMonitorSID* $\longrightarrow$
binder_getAirSpeedRet . *mainMission* . *NavigationMonitorSID* ? *getAirSpeed* $\longrightarrow$
Skip ; **var** *airSpeed* : $\mathbb{P}\,\mathbb{A} \bullet$ *airSpeed* := *getAirSpeed*;
binder_getAltitudeCall . *mainMission* . *NavigationMonitorSID* $\longrightarrow$
binder_getAltitudeRet . *mainMission* . *NavigationMonitorSID* ? *getAltitude* $\longrightarrow$
Skip ; **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* $\longrightarrow$ **Skip**$)$

end

5.7 LandMission

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

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

State

$this : \mathbf{ref}\ LandMissionClass$

state *State*

Init

$State'$

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

$$InitializePhase \mathrel{\widehat=} \begin{pmatrix} initializeCall\,.\,LandMissionMID \longrightarrow \\ register\,!\,GroundDistanceMonitorSID\,!\,LandMissionMID \longrightarrow \\ register\,!\,LandingGearHandlerLandSID\,!\,LandMissionMID \longrightarrow \\ register\,!\,InstrumentLandingSystemMonitorSID\,!\,LandMissionMID \longrightarrow \\ register\,!\,SafeLandingHandlerSID\,!\,LandMissionMID \longrightarrow \\ initializeRet\,.\,LandMissionMID \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

$$CleanupPhase \mathrel{\widehat=} \mathbf{var}\,\mathbb{B} : ret \bullet \begin{pmatrix} cleanupMissionCall\,.\,LandMissionMID \longrightarrow \\ \begin{pmatrix} \mathbf{Skip}; \\ ret := \mathbf{False} \end{pmatrix} \\ cleanupMissionRet\,.\,LandMissionMID\,!\,ret \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

$$deployLandingGearMeth \mathrel{\widehat=} \begin{pmatrix} deployLandingGearCall\,.\,LandMissionMID\,?\,caller \longrightarrow \\ this\,.\,deployLandingGear(); \\ deployLandingGearRet\,.\,LandMissionMID\,.\,caller \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

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

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

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

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

end

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

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

state *State*

$SAFE\_LANDING\_ALTITUDE : \mathbb{P}\,\mathbb{A}$
$ALTITUDE\_READING\_ON\_GROUND : \mathbb{P}\,\mathbb{A}$
$abort : \mathbb{B}$
$landingGearDeployed : \mathbb{B}$

state *State*

initial *Init*

$State'$

$SAFE\_LANDING\_ALTITUDE' = 10.0$
$ALTITUDE\_READING\_ON\_GROUND' = 0.0$
$abort' = false$

public *deployLandingGear* $\widehat{=}$
$\big(landingGearDeployed := \mathbf{True}\big)$

public *stowLandingGear* $\widehat{=}$
$\big(landingGearDeployed := \mathbf{False}\big)$

public *isLandingGearDeployed* $\widehat{=}$
$\big(ret := landingGearDeployed\big)$

- **Skip**

end

5.8 Schedulables of LandMission

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

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

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

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

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

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

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 $GroundDistanceMonitorApp \mathrel{\widehat=}$
$mainMission : MissionID,$
$readingOnGround : \mathbb{P}\,\mathbb{A} \bullet$ **begin**

$handleAsyncEvent \mathrel{\widehat=}$
$\Big($ $handleAsyncEventCall\,.\,GroundDistanceMonitorSID \longrightarrow$
$\Big($ **Skip**;
$binder\_getAltitudeCall\,.\,mainMission\,.\,GroundDistanceMonitorSID \longrightarrow$
$binder\_getAltitudeRet\,.\,mainMission\,.\,GroundDistanceMonitorSID\,?\,getAltitude \longrightarrow$
Skip ; **var** $distance : \mathbb{P}\,\mathbb{A} \bullet distance := getAltitude$;
if $(distance = readingOnGround) \longrightarrow$
$\Big($ **Skip**;
$requestTerminationCall\,.\,mainMission\,.\,GroundDistanceMonitorSID \longrightarrow$
$requestTerminationRet\,.\,mainMission\,.\,GroundDistanceMonitorSID\,?\,requestTermination \longrightarrow$
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* $\widehat{=}$
$\quad$ *mission* : *MissionID* • **begin**

handleAsyncEvent $\widehat{=}$
$$\begin{pmatrix} handleAsyncEventCall\,.\,InstrumentLandingSystemMonitorSID \longrightarrow \\ (\mathbf{Skip})\,; \\ handleAsyncEventRet\,.\,InstrumentLandingSystemMonitorSID \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

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

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

end