aircraft

Tight Rope v0.65

9th May 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*

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*

distinct⟨*nullSequencerId*, *nullSchedulableId*, *MainMissionSequencerSID*,
ACModeChangerSID, *EnvironmentMonitorSID*,
ControlHandlerSID, *FlightSensorsMonitorSID*,
CommunicationsHandlerSID, *AperiodicSimulatorSID*,
LandingGearHandlerTakeOffSID, *TakeOffMonitorSID*,
TakeOffFailureHandlerSID, *BeginLandingHandlerSID*,
NavigationMonitorSID, *GroundDistanceMonitorSID*,
LandingGearHandlerLandSID, *InstrumentLandingSystemMonitorSID*,
SafeLandingHandlerSID⟩

1.3 ThreadIds

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

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*

---

distinct⟨SafeletTId, *nullThreadId*,
InstrumentLandingSystemMonitorTID, *SafeLandingHandlerTID*,
GroundDistanceMonitorTID, *CommunicationsHandlerTID*,
ControlHandlerTID, *AperiodicSimulatorTID*,
TakeOffFailureHandlerTID, *LandingGearHandlerLandTID*,
EnvironmentMonitorTID, *FlightSensorsMonitorTID*,
NavigationMonitorTID, *ACModeChangerTID*,
BeginLandingHandlerTID, *LandingGearHandlerTakeOffTID*,
TakeOffMonitorTID⟩

1.4 ObjectIds

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

TakeOffMissionOID : *ObjectID*
LandMissionOID : *ObjectID*

distinct⟨TakeOffMissionOID, LandMissionOID⟩

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* ≙
⦃| *getSequencerCall, getSequencerRet, initializeApplicationCall, initializeApplicationRet, end_safelet_app* |⦄

channelset *MissionSequencerAppSync* ==
⦃| *getNextMissionCall, getNextMissionRet, end_sequencer_app* |⦄

channelset *MissionAppSync* ==
⦃| *initializeCall, register, initializeRet, cleanupMissionCall, cleanupMissionRet* |⦄

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

channelset *ThreadSync* ==
  ⦃| *raise_thread_priority, lower_thread_priority, isInterruptedCall, isInterruptedRet, get_priorityLevel* |⦄

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

channelset *Tier0Sync* ==
  ⦃| *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_isLandingGearDeployedCall* : *MissionID* × *SchedulableID*
channel *binder_isLandingGearDeployedRet* : *MissionID* × *SchedulableID* × $\mathbb{B}$

isLandingGearDeployedLocs == {*TakeOffMissionMID*}
isLandingGearDeployedCallers == {*LandingGearHandlerTakeOffSID*}

channel *binder_stowLandingGearCall* : *MissionID* × *SchedulableID*
channel *binder_stowLandingGearRet* : *MissionID* × *SchedulableID*

stowLandingGearLocs == {*TakeOffMissionMID*}
stowLandingGearCallers == {*LandingGearHandlerTakeOffSID*}

channel *binder_deployLandingGearCall* : *MissionID* × *SchedulableID* × *ThreadID*
channel *binder_deployLandingGearRet* : *MissionID* × *SchedulableID* × *ThreadID*

deployLandingGearLocs == {*TakeOffMissionMID*}
deployLandingGearCallers == {*LandingGearHandlerTakeOffSID*}

channel *binder_getAltitudeCall* : *MissionID* × *SchedulableID*
channel *binder_getAltitudeRet* : *MissionID* × *SchedulableID* × $\mathbb{P}\,\mathbb{A}$

getAltitudeLocs == {*MainMissionMID*}
getAltitudeCallers == {*NavigationMonitorSID*, *TakeOffMonitorSID*, *GroundDistanceMonitorSID*, *SafeLandingHandlerS*

channel *binder_getAirSpeedCall* : *MissionID* × *SchedulableID*
channel *binder_getAirSpeedRet* : *MissionID* × *SchedulableID* × $\mathbb{P}\,\mathbb{A}$

getAirSpeedLocs == {*MainMissionMID*}
getAirSpeedCallers == {*NavigationMonitorSID*, *TakeOffFailureHandlerSID*}

channel *binder_getHeadingCall* : *MissionID* × *SchedulableID*
channel *binder_getHeadingRet* : *MissionID* × *SchedulableID* × $\mathbb{P}\,\mathbb{A}$

getHeadingLocs == {*MainMissionMID*}
getHeadingCallers == {*NavigationMonitorSID*}

channel *binder_getAirSpeedCall* : *MissionID* × *SchedulableID*
channel *binder_getAirSpeedRet* : *MissionID* × *SchedulableID* × $\mathbb{P}\,\mathbb{A}$

getAirSpeedLocs == {*MainMissionMID*}
getAirSpeedCallers == {*NavigationMonitorSID*, *TakeOffFailureHandlerSID*}

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

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

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

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

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

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

channelset $MethodCallBinderSync == \{\!|\ done\_toplevel\_sequencer,$
$binder\_isLandingGearDeployedCall, binder\_isLandingGearDeployedRet,$
$binder\_stowLandingGearCall, binder\_stowLandingGearRet,$
$binder\_deployLandingGearCall, binder\_deployLandingGearRet,$
$binder\_getAltitudeCall, binder\_getAltitudeRet,$
$binder\_getAirSpeedCall, binder\_getAirSpeedRet,$
$binder\_getHeadingCall, binder\_getHeadingRet,$
$binder\_getAirSpeedCall, binder\_getAirSpeedRet,$
$binder\_getAltitudeCall, binder\_getAltitudeRet,$
$binder\_getAltitudeCall, binder\_getAltitudeRet,$
$binder\_isLandingGearDeployedCall, binder\_isLandingGearDeployedRet,$
$binder\_stowLandingGearCall, binder\_stowLandingGearRet,$
$binder\_deployLandingGearCall, binder\_deployLandingGearRet,$
$binder\_getAltitudeCall, binder\_getAltitudeRet\ |\!\}$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$BinderActions \mathrel{\widehat{=}}$

$$
\begin{pmatrix}
isLandingGearDeployed\_MethodBinder \\
||| \\
stowLandingGear\_MethodBinder \\
||| \\
deployLandingGear\_MethodBinder \\
||| \\
getAltitude\_MethodBinder \\
||| \\
getAirSpeed\_MethodBinder \\
||| \\
getHeading\_MethodBinder \\
||| \\
getAirSpeed\_MethodBinder \\
||| \\
getAltitude\_MethodBinder \\
||| \\
getAltitude\_MethodBinder \\
||| \\
isLandingGearDeployed\_MethodBinder \\
||| \\
stowLandingGear\_MethodBinder \\
||| \\
deployLandingGear\_MethodBinder \\
||| \\
getAltitude\_MethodBinder
\end{pmatrix}
$$

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

end

2.3   Locking

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

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

process *Objects* $\widehat{=}$
$$\begin{pmatrix} ObjectFW(TakeOffMissionOID) \\ ||| \\ ObjectFW(LandMissionOID) \end{pmatrix}$$

process *Locking* $\widehat{=}$ *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* $\widehat{=}$
$$\begin{pmatrix} SafeletFW \\ \quad [\![ControlTierSync]\!] \\ TopLevelMissionSequencerFW(MainMissionSequencer) \end{pmatrix}$$

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

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

process $Framework \mathrel{\widehat{=}}$
$$\begin{pmatrix} ControlTier \\ \quad [\![TierSync]\!] \\ \begin{pmatrix} Tier0 \\ \quad [\![Tier0Sync]\!] \\ Tier1 \end{pmatrix} \end{pmatrix}$$

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

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

3 Safelet

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

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

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

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

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

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

end

4 Top Level Mission Sequencer

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

process *MainMissionSequencerApp* $\widehat{=}$ **begin**

State
$this$: **ref** *MainMissionSequencerClass*

---

state *State*

Init
State'

$this' = $ **new** *MainMissionSequencerClass*()

GetNextMission $\widehat{=}$ **var** *ret* : *MissionID* $\bullet$
$$\begin{pmatrix} getNextMissionCall \,.\, MainMissionSequencerSID \longrightarrow \\ ret := this \,.\, getNextMission(); \\ getNextMissionRet \,.\, MainMissionSequencerSID \,!\, ret \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

Methods $\widehat{=}$
$\begin{pmatrix} GetNextMission \end{pmatrix}$; *Methods*

$\bullet$ (*Init* ; *Methods*) $\triangle$ (*end_sequencer_app* . *MainMissionSequencerSID* $\longrightarrow$ **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* $\widehat{=}$ **var** *ret* : *MissionID* $\bullet$

$$\begin{pmatrix} \textbf{if} \, (\neg \, returnedMission = \textbf{True}) \longrightarrow \\ \qquad \begin{pmatrix} this \, . \, returnedMission := \textbf{True}; \\ ret := MainMissionMID \end{pmatrix} \\ [\!] \neg \, (\neg \, returnedMission = \textbf{True}) \longrightarrow \\ \qquad \begin{pmatrix} ret := nullMissionId \end{pmatrix} \\ \textbf{fi} \end{pmatrix}$$

$\bullet$ **Skip**

end

5 Missions

5.1 MainMission

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

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

```
┌─ State ────────────────────────────────
│  this : ref MainMissionClass
└────────────────────────────────────────
```

state *State*

```
┌─ Init ─────────────────────────────────
│  State′
│ ───────────────────────────────────────
│  this′ = new MainMissionClass()
└────────────────────────────────────────
```

InitializePhase $\widehat{=}$
$$\begin{pmatrix} initializeCall \, . \, MainMissionMID \longrightarrow \\ register \, ! \, ACModeChangerSID \, ! \, MainMissionMID \longrightarrow \\ register \, ! \, EnvironmentMonitorSID \, ! \, MainMissionMID \longrightarrow \\ register \, ! \, ControlHandlerSID \, ! \, MainMissionMID \longrightarrow \\ register \, ! \, FlightSensorsMonitorSID \, ! \, MainMissionMID \longrightarrow \\ register \, ! \, CommunicationsHandlerSID \, ! \, MainMissionMID \longrightarrow \\ register \, ! \, AperiodicSimulatorSID \, ! \, MainMissionMID \longrightarrow \\ initializeRet \, . \, MainMissionMID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

CleanupPhase $\widehat{=}$
$$\begin{pmatrix} cleanupMissionCall \, . \, MainMissionMID \longrightarrow \\ cleanupMissionRet \, . \, MainMissionMID \, ! \, \textbf{True} \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

getAirSpeedMeth $\widehat{=}$ **var** *ret* : $\mathbb{P} \, \mathbb{A} \, \bullet$
$$\begin{pmatrix} getAirSpeedCall \, . \, MainMissionMID \, ? \, caller \longrightarrow \\ ret := this \, . \, getAirSpeed(); \\ getAirSpeedRet \, . \, MainMissionMID \, . \, caller \, ! \, ret \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

getAltitudeMeth $\widehat{=}$ **var** *ret* : $\mathbb{P} \, \mathbb{A} \, \bullet$
$$\begin{pmatrix} getAltitudeCall \, . \, MainMissionMID \, ? \, caller \longrightarrow \\ ret := this \, . \, getAltitude(); \\ getAltitudeRet \, . \, MainMissionMID \, . \, caller \, ! \, ret \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

getCabinPressureMeth $\widehat{=}$ **var** *ret* : $\mathbb{P} \, \mathbb{A} \, \bullet$
$$\begin{pmatrix} getCabinPressureCall \, . \, MainMissionMID \longrightarrow \\ ret := this \, . \, getCabinPressure(); \\ getCabinPressureRet \, . \, MainMissionMID \, ! \, ret \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

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

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

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

$setAirSpeedMeth \mathrel{\widehat{=}}$
$$\left(\begin{aligned} &setAirSpeedCall \,.\, MainMissionMID \,?\, airSpeed \longrightarrow \\ &this \,.\, setAirSpeed(airSpeed); \\ &setAirSpeedRet \,.\, MainMissionMID \longrightarrow \\ &\mathbf{Skip} \end{aligned} \right)$$

$setAltitudeMeth \mathrel{\widehat{=}}$
$$\left(\begin{aligned} &setAltitudeCall \,.\, MainMissionMID \,?\, altitude \longrightarrow \\ &this \,.\, setAltitude(altitude); \\ &setAltitudeRet \,.\, MainMissionMID \longrightarrow \\ &\mathbf{Skip} \end{aligned} \right)$$

$setCabinPressureMeth \mathrel{\widehat{=}}$
$$\left(\begin{aligned} &setCabinPressureCall \,.\, MainMissionMID \,?\, cabinPressure \longrightarrow \\ &this \,.\, setCabinPressure(cabinPressure); \\ &setCabinPressureRet \,.\, MainMissionMID \longrightarrow \\ &\mathbf{Skip} \end{aligned} \right)$$

$setEmergencyOxygenMeth \mathrel{\widehat{=}}$
$$\left(\begin{aligned} &setEmergencyOxygenCall \,.\, MainMissionMID \,?\, emergencyOxygen \longrightarrow \\ &this \,.\, setEmergencyOxygen(emergencyOxygen); \\ &setEmergencyOxygenRet \,.\, MainMissionMID \longrightarrow \\ &\mathbf{Skip} \end{aligned} \right)$$

$setFuelRemainingMeth \mathrel{\widehat{=}}$
$$\left(\begin{aligned} &setFuelRemainingCall \,.\, MainMissionMID \,?\, fuelRemaining \longrightarrow \\ &this \,.\, setFuelRemaining(fuelRemaining); \\ &setFuelRemainingRet \,.\, MainMissionMID \longrightarrow \\ &\mathbf{Skip} \end{aligned} \right)$$

$setHeadingMeth \mathrel{\widehat{=}}$
$$\left(\begin{aligned} &setHeadingCall \,.\, MainMissionMID \,?\, heading \longrightarrow \\ &this \,.\, setHeading(heading); \\ &setHeadingRet \,.\, MainMissionMID \longrightarrow \\ &\mathbf{Skip} \end{aligned} \right)$$

$$
Methods \mathrel{\widehat{=}} \begin{pmatrix} InitializePhase \\ \square \\ CleanupPhase \\ \square \\ getAirSpeedMeth \\ \square \\ getAltitudeMeth \\ \square \\ getCabinPressureMeth \\ \square \\ getEmergencyOxygenMeth \\ \square \\ getFuelRemainingMeth \\ \square \\ getHeadingMeth \\ \square \\ setAirSpeedMeth \\ \square \\ setAltitudeMeth \\ \square \\ setCabinPressureMeth \\ \square \\ setEmergencyOxygenMeth \\ \square \\ setFuelRemainingMeth \\ \square \\ setHeadingMeth \end{pmatrix} \; ; \; Methods
$$

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

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* : $\mathbb{P}\,\mathbb{A}$ •
$(\,ret := airSpeed\,)$

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

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

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

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

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

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

GetNextMission $\widehat{=}$ **var** *ret* : *MissionID* •
$$\begin{pmatrix} getNextMissionCall \, . \, ACModeChangerSID \longrightarrow \\ ret := this \, . \, getNextMission(); \\ getNextMissionRet \, . \, ACModeChangerSID \, ! \, ret \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

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

• (*Methods*) $\triangle$ (*end_sequencer_app* . *ACModeChangerSID* $\longrightarrow$ **Skip**)

end

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

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

---
state *State*
$controllingMission : MainMission$
$modesLeft : \mathbb{Z}$

---

state *State*

---
initial *Init*
$State'$

---

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

$$
\begin{pmatrix}
\textbf{if}\,(modesLeft = 3) \longrightarrow \\
\qquad \begin{pmatrix} modesLeft := modesLeft - 1; \\ ret := TakeOffMissionMID \end{pmatrix} \\
[\!]\, \neg\,(modesLeft = 3) \longrightarrow \\
\qquad \textbf{if}\,(modesLeft = 2) \longrightarrow \\
\qquad \begin{pmatrix} modesLeft := modesLeft - 1; \\ ret := CruiseMissionMID \end{pmatrix} \\
[\!]\, \neg\,(modesLeft = 2) \longrightarrow \\
\qquad \textbf{if}\,(modesLeft = 1) \longrightarrow \\
\qquad \begin{pmatrix} modesLeft := modesLeft - 1; \\ ret := LandMissionMID \end{pmatrix} \\
[\!]\, \neg\,(modesLeft = 1) \longrightarrow \\
\qquad \begin{pmatrix} ret := nullMissionId \end{pmatrix} \\
\textbf{fi} \\
\textbf{fi} \\
\textbf{fi}
\end{pmatrix}
$$

$\bullet$ **Skip**

end

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

process *ControlHandlerApp* $\widehat{=}$ **begin**

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

$Methods \;\widehat{=}$
$\begin{pmatrix} handleAsyncEvent \end{pmatrix} ; \; Methods$

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

end

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

process *CommunicationsHandlerApp* $\widehat{=}$ **begin**

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

$Methods \,\widehat{=}$
$\begin{pmatrix} handleAsyncEvent \end{pmatrix} \,;\; Methods$

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

end

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

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

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

Methods $\widehat{=}$
$\begin{pmatrix} handleAsyncEvent \end{pmatrix}$; *Methods*

• (*Methods*) $\triangle$ (*end_periodic_app* . *EnvironmentMonitorSID* $\longrightarrow$ **Skip**)

end

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

27

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

```
__ state State _____
  controllingMission : MainMission
```

state *State*

```
__ initial Init _____
  State′
```

- **Skip**

end

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

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

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

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

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

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*, *MethodCallBindingC*

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

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

Methods $\widehat{=}$
$\begin{pmatrix} handleAsyncEvent \end{pmatrix}$; *Methods*

• (*Methods*) $\triangle$ (*end_periodic_app*.*AperiodicSimulatorSID* $\longrightarrow$ **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* $\widehat{=}$
  controllingMission : *MissionID* • **begin**

State
  this : **ref** *TakeOffMissionClass*

---

state *State*

Init
  State'
  ---
  this' = **new** *TakeOffMissionClass*()

InitializePhase $\widehat{=}$
$\begin{pmatrix} initializeCall \. TakeOffMissionMID \longrightarrow \\ register \! LandingGearHandlerTakeOffSID \! TakeOffMissionMID \longrightarrow \\ register \! TakeOffMonitorSID \! TakeOffMissionMID \longrightarrow \\ register \! TakeOffFailureHandlerSID \! TakeOffMissionMID \longrightarrow \\ initializeRet \. TakeOffMissionMID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$

CleanupPhase $\widehat{=}$
$\begin{pmatrix} cleanupMissionCall \. TakeOffMissionMID \longrightarrow \\ cleanupMissionRet \. TakeOffMissionMID \! \textbf{True} \longrightarrow \\ \textbf{Skip} \end{pmatrix}$

abortMeth $\widehat{=}$
$\begin{pmatrix} abortCall \. TakeOffMissionMID \longrightarrow \\ this \. abort(); \\ abortRet \. TakeOffMissionMID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$

getControllingMissionMeth $\widehat{=}$ **var** *ret* : *MissionID* •
$\begin{pmatrix} getControllingMissionCall \. TakeOffMissionMID \longrightarrow \\ ret := this \. getControllingMission(); \\ getControllingMissionRet \. TakeOffMissionMID \! ret \longrightarrow \\ \textbf{Skip} \end{pmatrix}$

setControllingMissionMeth $\widehat{=}$
$\begin{pmatrix} setControllingMissionCall \. TakeOffMissionMID \? controllingMission \longrightarrow \\ this \. setControllingMission(controllingMission); \\ setControllingMissionRet \. TakeOffMissionMID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$

$cleanUpMeth \cong \mathbf{var}\ ret : \mathbb{B} \bullet$
$$\begin{pmatrix} cleanUpCall \,.\, TakeOffMissionMID \longrightarrow \\ ret := this \,.\, cleanUp(); \\ cleanUpRet \,.\, TakeOffMissionMID \,!\, ret \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

$stowLandingGearMeth \cong$
$$\begin{pmatrix} stowLandingGearCall \,.\, TakeOffMissionMID \,?\, caller \longrightarrow \\ this \,.\, stowLandingGear(); \\ stowLandingGearRet \,.\, TakeOffMissionMID \,.\, caller \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

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

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

$$Methods \cong \begin{pmatrix} InitializePhase \\ \square \\ CleanupPhase \\ \square \\ abortMeth \\ \square \\ getControllingMissionMeth \\ \square \\ setControllingMissionMeth \\ \square \\ cleanUpMeth \\ \square \\ stowLandingGearMeth \\ \square \\ isLandingGearDeployedMeth \\ \square \\ deployLandingGearSyncMeth \end{pmatrix} ; \ Methods$$

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

$\mathbf{end}$

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

class *TakeOffMissionClass* $\hat{=}$ **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* $\hat{=}$
$\big(this \,.\, abort := \textbf{True}\big)$

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

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

public *cleanUp* $\hat{=}$ **var** *ret* : $\mathbb{B}$ $\bullet$
$\big(ret := (\neg \; abort = \textbf{True})\big)$

public *stowLandingGear* $\hat{=}$
$\big(this \,.\, landingGearDeployed := \textbf{False}\big)$

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

$\bullet$ **Skip**

end

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

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

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

channel *setControllingMissionCall* : *MissionID* × *MissionID*
channel *setControllingMissionRet* : *MissionID*

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

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

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

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

5.4 Schedulables of TakeOffMission

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

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

handleAsyncEvent $\widehat{=}$
$\Bigg(\begin{array}{l}\end{array}$ *handleAsyncEventCall* . *LandingGearHandlerTakeOffSID* $\longrightarrow$
 $\bigg(\begin{array}{l}\end{array}$ *binder_isLandingGearDeployedCall* . *mission* . *LandingGearHandlerTakeOffSID* $\longrightarrow$
   binder_isLandingGearDeployedRet . *mission* . *LandingGearHandlerTakeOffSID* ? *isLandingGearDeployed* $\longrightarrow$
   Skip var *landingGearIsDeployed* : $\mathbb{B}$ • *landingGearIsDeployed* := *isLandingGearDeployed*;
   if *landingGearIsDeployed* = **True** $\longrightarrow$
   $\left(\begin{array}{l} binder\_stowLandingGearCall . mission . LandingGearHandlerTakeOffSID \longrightarrow \\ binder\_stowLandingGearRet . mission . LandingGearHandlerTakeOffSID \longrightarrow \\ \textbf{Skip} \end{array}\right)$
   $\llbracket \neg$ *landingGearIsDeployed* = **True** $\longrightarrow$
   $\left(\begin{array}{l} binder\_deployLandingGearCall . mission . LandingGearHandlerTakeOffSID . LandingGearHandlerTakeOffTID\text{-} \\ binder\_deployLandingGearRet . mission . LandingGearHandlerTakeOffSID . LandingGearHandlerTakeOffTID\text{-} \\ \textbf{Skip} \end{array}\right)$
   fi
  handleAsyncEventRet . *LandingGearHandlerTakeOffSID* $\longrightarrow$
  Skip

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

• (*Methods*) $\triangle$ (*end_aperiodic_app* . *LandingGearHandlerTakeOffSID* $\longrightarrow$ **Skip**)

end

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

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

$handleAsyncEvent \widehat{=}$
$\begin{pmatrix} handleAsyncEventCall . TakeOffFailureHandlerSID \longrightarrow \\ \begin{pmatrix} binder\_getAirSpeedCall . mainMission . TakeOffFailureHandlerSID \longrightarrow \\ binder\_getAirSpeedRet . mainMission . TakeOffFailureHandlerSID\,?\,getAirSpeed \longrightarrow \\ \textbf{Skip var } currentSpeed : \mathbb{P}\,\mathbb{A} \bullet currentSpeed := getAirSpeed; \\ \textbf{if } (currentSpeed < threshold) \longrightarrow \\ \quad \textbf{Skip} \\ [\!]\, \neg\, (currentSpeed < threshold) \longrightarrow \\ \quad \textbf{Skip} \\ \textbf{fi} \end{pmatrix} \\ handleAsyncEventRet . TakeOffFailureHandlerSID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$;

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

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

end

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

class *TakeOffFailureHandlerClass* $\hat{=}$ **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* $\widehat{=}$
$\quad$ *mainMission* : *MissionID*,
takeOffMission : *MissionID*,
takeOffAltitude : $\mathbb{P}\,\mathbb{A}$,
landingGearHandler : *SchedulableID* • **begin**

handleAsyncEvent $\widehat{=}$
$\left(\begin{array}{l} handleAsyncEventCall . TakeOffMonitorSID \longrightarrow \\ \left(\begin{array}{l} binder\_getAltitudeCall . mainMission . TakeOffMonitorSID \longrightarrow \\ binder\_getAltitudeRet . mainMission . TakeOffMonitorSID\,? getAltitude \longrightarrow \\ \textbf{Skip var } altitude : \mathbb{P}\,\mathbb{A} \bullet altitude := getAltitude; \\ \textbf{if } (altitude > takeOffAltitude) \longrightarrow \\ \quad\quad \textbf{Skip} \\ [\!]\; \neg\, (altitude > takeOffAltitude) \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right); \\ handleAsyncEventRet . TakeOffMonitorSID \longrightarrow \\ \textbf{Skip} \end{array}\right)$

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

• (*Methods*) $\triangle$ (*end_periodic_app* . *TakeOffMonitorSID* $\longrightarrow$ **Skip**)

end

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

class *TakeOffMonitorClass* $\hat{=}$ **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* $\widehat{=}$
 controllingMission : *MissionID* • **begin**

$\boxed{\begin{array}{l} \textit{State} \\ \hline \textit{this} : \textbf{ref } \textit{CruiseMissionClass} \end{array}}$

state *State*

$\boxed{\begin{array}{l} \textit{Init} \\ \hline \textit{State}' \\ \hline \textit{this}' = \textbf{new } \textit{CruiseMissionClass}() \end{array}}$

InitializePhase $\widehat{=}$
$\left(\begin{array}{l} \textit{initializeCall . CruiseMissionMID} \longrightarrow \\ \textit{register ! BeginLandingHandlerSID ! CruiseMissionMID} \longrightarrow \\ \textit{register ! NavigationMonitorSID ! CruiseMissionMID} \longrightarrow \\ \textit{initializeRet . CruiseMissionMID} \longrightarrow \\ \textbf{Skip} \end{array} \right)$

CleanupPhase $\widehat{=}$
$\left(\begin{array}{l} \textit{cleanupMissionCall . CruiseMissionMID} \longrightarrow \\ \textit{cleanupMissionRet . CruiseMissionMID !} \textbf{True} \longrightarrow \\ \textbf{Skip} \end{array} \right)$

getControllingMissionMeth $\widehat{=}$ **var** *ret* : *MissionID* •
$\left(\begin{array}{l} \textit{getControllingMissionCall . CruiseMissionMID} \longrightarrow \\ \textit{ret} := \textit{this . getControllingMission}(); \\ \textit{getControllingMissionRet . CruiseMissionMID ! ret} \longrightarrow \\ \textbf{Skip} \end{array} \right)$

Methods $\widehat{=}$ $\left(\begin{array}{l} \textit{InitializePhase} \\ \Box \\ \textit{CleanupPhase} \\ \Box \\ \textit{getControllingMissionMeth} \end{array} \right)$; *Methods*

• (*Init* ; *Methods*) $\triangle$ (*end_mission_app . CruiseMissionMID* $\longrightarrow$ **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* $\widehat{=}$
   controllingMission : *MissionID* • **begin**

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

Methods $\widehat{=}$
$\begin{pmatrix} handleAsyncEvent \end{pmatrix}$; *Methods*

• (*Methods*) $\triangle$ (*end_aperiodic_app* . *BeginLandingHandlerSID* $\longrightarrow$ **Skip**)

end

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

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

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

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

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

end

5.7 LandMission

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

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

State

  this : **ref** *LandMissionClass*

---

state *State*

Init

  State$'$

  this$'$ = **new** *LandMissionClass*()

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

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

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

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

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

$abortMeth \;\widehat{=}$
$$\begin{pmatrix} abortCall \,.\, LandMissionMID \longrightarrow \\ this \,.\, abort(); \\ abortRet \,.\, LandMissionMID \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

$cleanUpMeth \;\widehat{=}\; \mathbf{var}\ ret : \mathbb{B}\ \bullet$
$$\begin{pmatrix} cleanUpCall \,.\, LandMissionMID \longrightarrow \\ ret := this \,.\, cleanUp(); \\ cleanUpRet \,.\, LandMissionMID \,!\, ret \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

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

$$Methods \;\widehat{=}\; \begin{pmatrix} InitializePhase \\ \square \\ CleanupPhase \\ \square \\ stowLandingGearMeth \\ \square \\ isLandingGearDeployedMeth \\ \square \\ getControllingMissionMeth \\ \square \\ abortMeth \\ \square \\ cleanUpMeth \\ \square \\ deployLandingGearSyncMeth \end{pmatrix} ;\ Methods$$

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

$\mathbf{end}$

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* $\,\widehat{=}\,$
$\big($*this* . *landingGearDeployed* := **False**$\big)$

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

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

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

public *cleanUp* $\,\widehat{=}\,$ **var** *ret* : $\mathbb{B}$ •
$\big($*ret* := $(\neg$ *abort* = **True**$))\big)$

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

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

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

• $($ *Methods* $)$ $\triangle$ $($ *end_aperiodic_app* . *LandingGearHandlerLandSID* $\longrightarrow$ **Skip** $)$

end

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

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

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

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

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

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*

52

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

$handleAsyncEvent \widehat{=}$
$\begin{pmatrix} handleAsyncEventCall \, . \, GroundDistanceMonitorSID \longrightarrow \\ \begin{pmatrix} binder\_getAltitudeCall \, . \, mainMission \, . \, GroundDistanceMonitorSID \longrightarrow \\ binder\_getAltitudeRet \, . \, mainMission \, . \, GroundDistanceMonitorSID \, ? \, getAltitude \longrightarrow \\ \textbf{Skip var } distance : \mathbb{P} \, \mathbb{A} \bullet distance := getAltitude; \\ \textbf{if } (distance = readingOnGround) \longrightarrow \\ \qquad \textbf{Skip} \\ []\, \neg \, (distance = readingOnGround) \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{pmatrix} ; \\ handleAsyncEventRet \, . \, GroundDistanceMonitorSID \longrightarrow \\ \textbf{Skip} \end{pmatrix}$

$Methods \widehat{=}$
$\begin{pmatrix} handleAsyncEvent \end{pmatrix} ; \; Methods$

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

end

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

53

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

---
state *State*
> *readingOnGround* : $\mathbb{P} \, \mathbb{A}$
---

state *State*

---
initial *Init*
> *State'*

- **Skip**

end

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

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

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

Methods $\widehat{=}$
$\begin{pmatrix} handleAsyncEvent \end{pmatrix}$; *Methods*

• (*Methods*) $\triangle$ (*end_periodic_app* . *InstrumentLandingSystemMonitorSID* $\longrightarrow$ **Skip**)

end