

# aircraft

Tight Rope v0.6

November 1, 2015

## 1 Network

**section** *NetworkChannels* **parents** *scj\_prelude, MissionId, MissionIds, SchedulableId, SchedulableIds, MissionChan, SchedulableChan, TopLevelMissionSequencerFWChan, FrameworkChan, SafeletChan*

**channelset** *TerminateSync* ==  
    { *schedulables\_terminated, schedulables\_stopped, get\_activeSchedulables* }

**channelset** *ControlTierSync* ==  
    { *start\_toplevel\_sequencer, done\_toplevel\_sequencer, done\_safeletFW* }

**channelset** *TierSync* ==  
    { *start\_mission ., done\_mission ., done\_safeletFW, done\_toplevel\_sequencer* }

**channelset** *MissionSync* ==  
    { *done\_safeletFW, done\_toplevel\_sequencer, register, signalTerminationCall, signalTerminationRet, activate\_schedulables, done\_schedulable, cleanupSchedulableCall, cleanupSchedulableRet* }

**channelset** *SchedulablesSync* ==  
    { *activate\_schedulables, done\_safeletFW, done\_toplevel\_sequencer* }

**channelset** *ClusterSync* ==  
    { *done\_toplevel\_sequencer, done\_safeletFW* }

**channelset** *AppSync* ==  
    { *SafeltAppSync, MissionSequencerAppSync, MissionAppSync, MTAppSync, OSEHSync, APEHSync, getSequencer, end\_mission\_app, end\_managedThread\_app, setCeilingPriority, requestTerminationCall, requestTerminationRet, terminationPendingCall, terminationPendingRet, handleAsyncEventCall, handleAsyncEventRet* }

**channelset** *ObjectSync* ==  
    { }

**channelset** *ThreadSync* ==  
    { }

**channelset** *LockingSync* ==  
    { *lockAcquired, startSyncMeth, endSyncMeth, waitCall, waitRet, notify* }

```

channelset Tier0Sync ==
    { done_toplevel_sequencer, done_safeletFW,
      start_mission . , done_mission . ,
        initializeRet . , requestTermination . . ,
      start_mission . , done_mission . ,
        initializeRet . , requestTermination . . ,
      start_mission . , done_mission . ,
        initializeRet . , requestTermination . . }

```

**section** *Program* **parents** *scj\_prelude, MissionId, MissionIds,*  
*SchedulableId, SchedulableIds, MissionChan, SchedulableMethChan, MissionFW,*  
*SafeletFW, TopLevelMissionSequencerFW, NetworkChannels, ManagedThreadFW,*  
*SchedulableMissionSequencerFW, PeriodicEventHandlerFW, OneShotEventHandlerFW,*  
*AperiodicEventHandlerFW, ACSafeletApp, MainMissionSequencerApp,*  
*ObjectFW, ThreadFW, MainMissionApp, ACModeChangerApp, ControlHandlerApp, CommunicationsHandlerApp*

**process** *ControlTier*  $\hat{=}$   

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

**process** *Tier0*  $\hat{=}$   

$$\left( \begin{array}{l} \text{MissionFW}(\text{MainMission}) \\ \llbracket \text{MissionSync} \rrbracket \\ \left( \begin{array}{l} \text{SchedulableMissionSequencerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \left( \begin{array}{l} \text{AperiodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \text{AperiodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \left( \begin{array}{l} \text{PeriodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \text{PeriodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \text{PeriodicEventHandlerFW}() \end{array} \right) \end{array} \right) \end{array} \right)$$

**process** *Tier1*  $\hat{=}$   

$$\left( \begin{array}{l} \text{MissionFW}(\text{TakeOffMission}) \\ \llbracket \text{MissionSync} \rrbracket \\ \left( \begin{array}{l} \left( \begin{array}{l} \text{AperiodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \end{array} \right) \\ \text{AperiodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \text{PeriodicEventHandlerFW}() \end{array} \right) \\ \llbracket \text{ClusterSync} \rrbracket \\ \left( \begin{array}{l} \text{MissionFW}(\text{CruiseMission}) \\ \llbracket \text{MissionSync} \rrbracket \\ \left( \begin{array}{l} \text{AperiodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \end{array} \right) \\ \text{PeriodicEventHandlerFW}() \end{array} \right) \\ \llbracket \text{ClusterSync} \rrbracket \\ \left( \begin{array}{l} \text{MissionFW}(\text{LandMission}) \\ \llbracket \text{MissionSync} \rrbracket \\ \left( \begin{array}{l} \left( \begin{array}{l} \text{AperiodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \end{array} \right) \\ \text{AperiodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \\ \left( \begin{array}{l} \text{PeriodicEventHandlerFW}() \\ \llbracket \text{SchedulablesSync} \rrbracket \end{array} \right) \\ \text{PeriodicEventHandlerFW}() \end{array} \right) \end{array} \right)$$

**process** *Framework*  $\hat{=}$   

$$\left( \begin{array}{l} \text{ControlTier} \\ \llbracket \text{TierSync} \rrbracket \\ \left( \begin{array}{l} \text{Tier0} \\ \llbracket \text{Tier0Sync} \rrbracket \end{array} \right) \\ \text{Tier1} \end{array} \right)$$

**process** *Application*  $\hat{=}$

(*ACSafeletApp*  
 |||  
*MainMissionSequencerApp*  
 |||  
*MainMissionApp*(*hijac.tools.tightrope.environments.VariableEnv* • 71c27ee8, *hijac.tools.tightrope.environments.VariableEnv* • 71c27ee8)  
 |||  
*ACModeChangerApp*(*MainMission*)  
 |||  
*ControlHandlerApp*  
 |||  
*CommunicationsHandlerApp*  
 |||  
*EnvironmentMonitorApp*(*MainMission*)  
 |||  
*FlightSensorsMonitorApp*(*MainMission*)  
 |||  
*AperiodicSimulatorApp*(*AperiodicEventHandler*)  
 |||  
*TakeOffMissionApp*(*hijac.tools.tightrope.environments.VariableEnv* • 6e950bcf, *hijac.tools.tightrope.environments.VariableEnv* • 6e950bcf)  
 |||  
*LandingGearHandlerTakeOffApp*(*TakeOffMission*)  
 |||  
*TakeOffFailureHandlerApp*(*TakeOffMission*)  
 |||  
*TakeOffMonitorApp*(*TakeOffMission*, *AperiodicEventHandler*)  
 |||  
*CruiseMissionApp*(*hijac.tools.tightrope.environments.VariableEnv* • 46dffd3, *hijac.tools.tightrope.environments.VariableEnv* • 46dffd3)  
 |||  
*BeginLandingHandlerApp*(*Mission*)  
 |||  
*NavigationMonitorApp*(*CruiseMission*)  
 |||  
*LandMissionApp*(*hijac.tools.tightrope.environments.VariableEnv* • 53dbe163, *hijac.tools.tightrope.environments.VariableEnv* • 53dbe163)  
 |||  
*LandingGearHandlerLandApp*(*LandMission*)  
 |||  
*SafeLandingHandlerApp*(*LandMission*)  
 |||  
*GroundDistanceMonitorApp*(*LandMission*)  
 |||  
*InstrumentLandingSystemMonitorApp*(*LandMission*))

$$\begin{aligned}
\text{Locking} \hat{=} & \left( \begin{array}{l} \text{ThreadFW}(\text{LandingGearHandlerLandThread}, \text{MinPriority}) \\ \quad \llbracket \text{ThreadSync} \rrbracket \\ \text{ThreadFW}(\text{SafeLandingHandlerThread}, \text{MinPriority}) \\ \quad \llbracket \text{ThreadSync} \rrbracket \\ \text{ThreadFW}(\text{GroundDistanceMonitorThread}, \text{MinPriority}) \\ \quad \llbracket \text{ThreadSync} \rrbracket \\ \text{ThreadFW}(\text{InstrumentLandingSystemMonitorThread}, \text{MinPriority}) \end{array} \right) \\
& ||| \\
& \left( \begin{array}{l} \text{ObjectFW}(\text{ACSafeletObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{MainMissionObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{TakeOffMissionObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{CruiseMissionObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{LandMissionObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{ACModeChangerObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{EnvironmentMonitorObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{ControlHandlerObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{FlightSensorsMonitorObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{CommunicationsHandlerObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{AperiodicSimulatorObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{LandingGearHandlerTakeOffObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{TakeOffMonitorObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{TakeOffFailureHandlerObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{BeginLandingHandlerObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{NavigationMonitorObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{GroundDistanceMonitorObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{LandingGearHandlerLandObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{InstrumentLandingSystemMonitorObject}) \\ \quad \llbracket \text{ObjectSync} \rrbracket \\ \text{ObjectFW}(\text{SafeLandingHandlerObject}) \end{array} \right)
\end{aligned}$$

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

## 2 ID Files

### 2.1 MissionIds

**section** *MissionIds* **parents** *scj\_prelude, MissionId*

*MainMission* : *MissionID*  
*TakeOffMission* : *MissionID*  
*CruiseMission* : *MissionID*  
*LandMission* : *MissionID*

---

*distinct*(*nullMissionId*, *MainMission*,  
*TakeOffMission*,  
*CruiseMission*,  
*LandMission*)

### 2.2 SchedulablesIds

**section** *SchedulableIds* **parents** *scj\_prelude, SchedulableId*

*MainMissionSequencer* : *SchedulableID*  
*ACModeChanger* : *SchedulableID*  
*EnvironmentMonitor* : *SchedulableID*  
*ControlHandler* : *SchedulableID*  
*FlightSensorsMonitor* : *SchedulableID*  
*CommunicationsHandler* : *SchedulableID*  
*AperiodicSimulator* : *SchedulableID*  
*LandingGearHandlerTakeOff* : *SchedulableID*  
*TakeOffMonitor* : *SchedulableID*  
*TakeOffFailureHandler* : *SchedulableID*  
*BeginLandingHandler* : *SchedulableID*  
*NavigationMonitor* : *SchedulableID*  
*GroundDistanceMonitor* : *SchedulableID*  
*LandingGearHandlerLand* : *SchedulableID*  
*InstrumentLandingSystemMonitor* : *SchedulableID*  
*SafeLandingHandler* : *SchedulableID*

---

*distinct*(*nullSequencerId*, *nullSchedulableId*, *ACModeChanger*,  
*EnvironmentMonitor*,  
*ControlHandler*,  
*FlightSensorsMonitor*,  
*CommunicationsHandler*,  
*AperiodicSimulator*,  
*LandingGearHandlerTakeOff*,  
*TakeOffMonitor*,  
*TakeOffFailureHandler*,  
*BeginLandingHandler*,  
*NavigationMonitor*,  
*GroundDistanceMonitor*,  
*LandingGearHandlerLand*,  
*InstrumentLandingSystemMonitor*,  
*SafeLandingHandler*)

### 2.3 ThreadIds

**section** *ThreadId* **parents** *scj\_prelude, GlobalTypes*

*ACModeChangerThread : ThreadID*  
*EnvironmentMonitorThread : ThreadID*  
*ControlHandlerThread : ThreadID*  
*FlightSensorsMonitorThread : ThreadID*  
*CommunicationsHandlerThread : ThreadID*  
*AperiodicSimulatorThread : ThreadID*  
*LandingGearHandlerTakeOffThread : ThreadID*  
*TakeOffMonitorThread : ThreadID*  
*TakeOffFailureHandlerThread : ThreadID*  
*BeginLandingHandlerThread : ThreadID*  
*NavigationMonitorThread : ThreadID*  
*GroundDistanceMonitorThread : ThreadID*  
*LandingGearHandlerLandThread : ThreadID*  
*InstrumentLandingSystemMonitorThread : ThreadID*  
*SafeLandingHandlerThread : ThreadID*

---

*distinct(SafeletThreadId, nullThreadId,*  
*ACModeChangerThread,*  
*EnvironmentMonitorThread,*  
*ControlHandlerThread,*  
*FlightSensorsMonitorThread,*  
*CommunicationsHandlerThread,*  
*AperiodicSimulatorThread,*  
*LandingGearHandlerTakeOffThread,*  
*TakeOffMonitorThread,*  
*TakeOffFailureHandlerThread,*  
*BeginLandingHandlerThread,*  
*NavigationMonitorThread,*  
*GroundDistanceMonitorThread,*  
*LandingGearHandlerLandThread,*  
*InstrumentLandingSystemMonitorThread,*  
*SafeLandingHandlerThread)*

## 2.4 ObjectIds

**section** *ObjectIds* **parents** *scj\_prelude, GlobalTypes*

*ACSafeletObject* : *ObjectID*  
*MainMissionObject* : *ObjectID*  
*TakeOffMissionObject* : *ObjectID*  
*CruiseMissionObject* : *ObjectID*  
*LandMissionObject* : *ObjectID*  
*ACModeChangerObject* : *ObjectID*  
*EnvironmentMonitorObject* : *ObjectID*  
*ControlHandlerObject* : *ObjectID*  
*FlightSensorsMonitorObject* : *ObjectID*  
*CommunicationsHandlerObject* : *ObjectID*  
*AperiodicSimulatorObject* : *ObjectID*  
*LandingGearHandlerTakeOffObject* : *ObjectID*  
*TakeOffMonitorObject* : *ObjectID*  
*TakeOffFailureHandlerObject* : *ObjectID*  
*BeginLandingHandlerObject* : *ObjectID*  
*NavigationMonitorObject* : *ObjectID*  
*GroundDistanceMonitorObject* : *ObjectID*  
*LandingGearHandlerLandObject* : *ObjectID*  
*InstrumentLandingSystemMonitorObject* : *ObjectID*  
*SafeLandingHandlerObject* : *ObjectID*

---

*distinct* ( *ACSafeletObject*,  
*MainMissionObject*,  
*TakeOffMissionObject*,  
*CruiseMissionObject*,  
*LandMissionObject*,  
*ACModeChangerObject*,  
*EnvironmentMonitorObject*,  
*ControlHandlerObject*,  
*FlightSensorsMonitorObject*,  
*CommunicationsHandlerObject*,  
*AperiodicSimulatorObject*,  
*LandingGearHandlerTakeOffObject*,  
*TakeOffMonitorObject*,  
*TakeOffFailureHandlerObject*,  
*BeginLandingHandlerObject*,  
*NavigationMonitorObject*,  
*GroundDistanceMonitorObject*,  
*LandingGearHandlerLandObject*,  
*InstrumentLandingSystemMonitorObject*,  
*SafeLandingHandlerObject* )



### 3 Safelet

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

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

*InitializeApplication*  $\hat{=}$   
 $\left( \begin{array}{l} \textit{initializeApplicationCall} \longrightarrow \\ \textit{initializeApplicationRet} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

*GetSequencer*  $\hat{=}$   
 $\left( \begin{array}{l} \textit{getSequencerCall} \longrightarrow \\ \textit{getSequencerRet} ! \textit{MainMissionSequencer} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

*Methods*  $\hat{=}$   
 $\left( \begin{array}{l} \textit{GetSequencer} \\ \square \\ \textit{InitializeApplication} \end{array} \right); \textit{Methods}$

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

**end**

## 4 Top Level Mission Sequencer

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

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

<i>State</i> <i>this</i> : <b>ref</b> <i>MainMissionSequencerClass</i>
---

**state** *State*

<i>Init</i> <i>State</i> '
<i>this</i> ' = <b>new</b> <i>MainMissionSequencerClass</i> ()

*GetNextMission*  $\hat{=}$  **var** *ret* : *MissionID* •  
 $\left( \begin{array}{l} \textit{getNextMissionCall} . \textit{MainMissionSequencer} \longrightarrow \\ \textit{ret} := \textit{this} . \textit{getNextMission}(); \\ \textit{getNextMissionRet} . \textit{MainMissionSequencer} ! \textit{ret} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

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

•  $(\textit{Init} ; \textit{Methods}) \triangle (\textit{end\_sequencer\_app} . \textit{MainMissionSequencer} \longrightarrow \mathbf{Skip})$

**end**

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

<b>state</b> <i>State</i> <i>returnedMission</i> : $\mathbb{B}$
--

**state** *State*

<b>initial</b> <i>Init</i> <i>State</i> '
<i>returnedMission</i> ' = <i>false</i>

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

$$\left( \begin{array}{l} \text{if } (\neg \text{returnedMission} = \mathbf{True}) \longrightarrow \\ \quad \left( \begin{array}{l} \text{returnedMission} := \mathbf{True}; \\ \text{ret} := \text{MainMission} \end{array} \right) \\ \parallel \neg (\neg \text{returnedMission} = \mathbf{True}) \longrightarrow \\ \quad \left( \text{ret} := \text{nullMissionId} \right) \\ \text{fi} \end{array} \right)$$

• **Skip**

**end**

## 5 Missions

### 5.1 MainMission

**section** *MainMissionApp* **parents** *scj\_prelude*, *MissionId*, *MissionIds*,  
*SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *MainMissionClass* , *MainMissionMethChan*

**process** *MainMissionApp*  $\hat{=}$  *storageParameters* : *MissionID*, *storageParametersSchedulable* : *MissionID*, *aCModeChange*

*State*

*this* : **ref** *MainMissionClass*

**state** *State*

*Init*

*State'*

*this'* = **new** *MainMissionClass*()

*InitializePhase*  $\hat{=}$

$$\left( \begin{array}{l} \textit{initializeCall} . \textit{MainMission} \longrightarrow \\ \textit{register} ! \textit{ACModeChanger} ! \textit{MainMission} \longrightarrow \\ \textit{register} ! \textit{EnvironmentMonitor} ! \textit{MainMission} \longrightarrow \\ \textit{register} ! \textit{ControlHandler} ! \textit{MainMission} \longrightarrow \\ \textit{register} ! \textit{FlightSensorsMonitor} ! \textit{MainMission} \longrightarrow \\ \textit{register} ! \textit{CommunicationsHandler} ! \textit{MainMission} \longrightarrow \\ \textit{register} ! \textit{AperiodicSimulator} ! \textit{MainMission} \longrightarrow \\ \textit{initializeRet} . \textit{MainMission} \longrightarrow \\ \textbf{Skip} \end{array} \right)$$

*CleanupPhase*  $\hat{=}$

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

*getAirSpeedMeth*  $\hat{=}$  **var** *ret* : *double* •

$$\left( \begin{array}{l} \textit{getAirSpeedCall} . \textit{MainMission} \longrightarrow \\ \textit{ret} := \textit{this} . \textit{getAirSpeed}(); \\ \textit{getAirSpeedRet} . \textit{MainMission} ! \textit{ret} \longrightarrow \\ \textbf{Skip} \end{array} \right)$$

*getAltitudeMeth*  $\hat{=}$  **var** *ret* : *double* •

$$\left( \begin{array}{l} \textit{getAltitudeCall} . \textit{MainMission} \longrightarrow \\ \textit{ret} := \textit{this} . \textit{getAltitude}(); \\ \textit{getAltitudeRet} . \textit{MainMission} ! \textit{ret} \longrightarrow \\ \textbf{Skip} \end{array} \right)$$

*getCabinPressureMeth*  $\hat{=}$  **var** *ret* : *double* •

$$\left( \begin{array}{l} \textit{getCabinPressureCall} . \textit{MainMission} \longrightarrow \\ \textit{ret} := \textit{this} . \textit{getCabinPressure}(); \\ \textit{getCabinPressureRet} . \textit{MainMission} ! \textit{ret} \longrightarrow \\ \textbf{Skip} \end{array} \right)$$

$$\text{getEmergencyOxygenMeth} \hat{=} \mathbf{var} \text{ ret} : \text{double} \bullet \left( \begin{array}{l} \text{getEmergencyOxygenCall} . \text{MainMission} \longrightarrow \\ \text{ret} := \text{this} . \text{getEmergencyOxygen}(); \\ \text{getEmergencyOxygenRet} . \text{MainMission} ! \text{ret} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{getFuelRemainingMeth} \hat{=} \mathbf{var} \text{ ret} : \text{double} \bullet \left( \begin{array}{l} \text{getFuelRemainingCall} . \text{MainMission} \longrightarrow \\ \text{ret} := \text{this} . \text{getFuelRemaining}(); \\ \text{getFuelRemainingRet} . \text{MainMission} ! \text{ret} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{getHeadingMeth} \hat{=} \mathbf{var} \text{ ret} : \text{double} \bullet \left( \begin{array}{l} \text{getHeadingCall} . \text{MainMission} \longrightarrow \\ \text{ret} := \text{this} . \text{getHeading}(); \\ \text{getHeadingRet} . \text{MainMission} ! \text{ret} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{setAirSpeedMeth} \hat{=} \left( \begin{array}{l} \text{setAirSpeedCall} . \text{MainMission} ? \text{airSpeed} \longrightarrow \\ \text{this} . \text{setAirSpeed}(\text{airSpeed}); \\ \text{setAirSpeedRet} . \text{MainMission} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{setAltitudeMeth} \hat{=} \left( \begin{array}{l} \text{setAltitudeCall} . \text{MainMission} ? \text{altitude} \longrightarrow \\ \text{this} . \text{setAltitude}(\text{altitude}); \\ \text{setAltitudeRet} . \text{MainMission} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{setCabinPressureMeth} \hat{=} \left( \begin{array}{l} \text{setCabinPressureCall} . \text{MainMission} ? \text{cabinPressure} \longrightarrow \\ \text{this} . \text{setCabinPressure}(\text{cabinPressure}); \\ \text{setCabinPressureRet} . \text{MainMission} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{setEmergencyOxygenMeth} \hat{=} \left( \begin{array}{l} \text{setEmergencyOxygenCall} . \text{MainMission} ? \text{emergencyOxygen} \longrightarrow \\ \text{this} . \text{setEmergencyOxygen}(\text{emergencyOxygen}); \\ \text{setEmergencyOxygenRet} . \text{MainMission} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{setFuelRemainingMeth} \hat{=} \left( \begin{array}{l} \text{setFuelRemainingCall} . \text{MainMission} ? \text{fuelRemaining} \longrightarrow \\ \text{this} . \text{setFuelRemaining}(\text{fuelRemaining}); \\ \text{setFuelRemainingRet} . \text{MainMission} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{setHeadingMeth} \hat{=} \left( \begin{array}{l} \text{setHeadingCall} . \text{MainMission} ? \text{heading} \longrightarrow \\ \text{this} . \text{setHeading}(\text{heading}); \\ \text{setHeadingRet} . \text{MainMission} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

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

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

**end**

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

**state** *State*

---

*ALTITUDE\_READING\_ON\_GROUND* : *double*  
*cabinPressure* : *double*  
*emergencyOxygen* : *double*  
*fuelRemaining* : *double*  
*altitude* : *double*  
*airSpeed* : *double*  
*heading* : *double*

---

**state** *State*

**initial** *Init*

---

*State*'

---

*ALTITUDE\_READING\_ON\_GROUND*' = 0.0

---

**public** *getAirSpeed*  $\hat{=}$  **var** *ret* : *double* •  
(*ret* := *airSpeed*)

**public** *getAltitude*  $\hat{=}$  **var** *ret* : *double* •  
(*ret* := *altitude*)

**public** *getCabinPressure*  $\hat{=}$  **var** *ret* : *double* •  
(*ret* := *cabinPressure*)

**public** *getEmergencyOxygen*  $\hat{=}$  **var** *ret* : *double* •  
(*ret* := *emergencyOxygen*)

**public** *getFuelRemaining*  $\hat{=}$  **var** *ret* : *double* •  
(*ret* := *fuelRemaining*)

**public** *getHeading*  $\hat{=}$  **var** *ret* : *double* •  
(*ret* := *heading*)

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

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

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

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

```
public setFuelRemaining  $\hat{=}$   
(this.this.fuelRemaining := fuelRemaining)
```

```
public setHeading  $\hat{=}$   
(this.this.heading := heading)
```

- **Skip**

```
end
```



**section** *MainMissionMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

**channel** *getAirSpeedCall* : *MissionID*  
**channel** *getAirSpeedRet* : *MissionID*  $\times$  *double*

**channel** *getAltitudeCall* : *MissionID*  
**channel** *getAltitudeRet* : *MissionID*  $\times$  *double*

**channel** *getCabinPressureCall* : *MissionID*  
**channel** *getCabinPressureRet* : *MissionID*  $\times$  *double*

**channel** *getEmergencyOxygenCall* : *MissionID*  
**channel** *getEmergencyOxygenRet* : *MissionID*  $\times$  *double*

**channel** *getFuelRemainingCall* : *MissionID*  
**channel** *getFuelRemainingRet* : *MissionID*  $\times$  *double*

**channel** *getHeadingCall* : *MissionID*  
**channel** *getHeadingRet* : *MissionID*  $\times$  *double*

**channel** *setAirSpeedCall* : *MissionID*  $\times$  *double*  
**channel** *setAirSpeedRet* : *MissionID*

**channel** *setAltitudeCall* : *MissionID*  $\times$  *double*  
**channel** *setAltitudeRet* : *MissionID*

**channel** *setCabinPressureCall* : *MissionID*  $\times$  *double*  
**channel** *setCabinPressureRet* : *MissionID*

**channel** *setEmergencyOxygenCall* : *MissionID*  $\times$  *double*  
**channel** *setEmergencyOxygenRet* : *MissionID*

**channel** *setFuelRemainingCall* : *MissionID*  $\times$  *double*  
**channel** *setFuelRemainingRet* : *MissionID*

**channel** *setHeadingCall* : *MissionID*  $\times$  *double*  
**channel** *setHeadingRet* : *MissionID*

## 5.2 Schedulables of

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

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

*State*

---

```

modesLeft :  $\mathbb{Z}$ 
ref currentModeClass : ModeClass
ref launchModeClass : ModeClass
ref cruiseModeClass : ModeClass
ref landModeClass : ModeClass

```

---

**state** *State*

*Init*

---

```

State'
modesLeft' = 3
ref currentModeClass' = new ModeClass()
ref launchModeClass' = new ModeClass()
ref cruiseModeClass' = new ModeClass()
ref landModeClass' = new ModeClass()

```

---

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

*changeToMeth*  $\hat{=}$   
 $\left( \begin{array}{l} \textit{changeToCall} . \textit{ACModeChanger} ? \textit{newMode} \longrightarrow \\ (\textit{this} . \textit{currentMode} := \textit{newMode}); \\ \textit{changeToRet} . \textit{ACModeChanger} \longrightarrow \\ \textbf{Skip} \end{array} \right)$

$$\begin{aligned}
& \text{getNextMissionMeth} \hat{=} \mathbf{var} \text{ ret} : \text{MissionID} \bullet \\
& \left( \begin{array}{l}
\text{getNextMissionCall} . \text{ACModeChanger} \longrightarrow \\
\left( \begin{array}{l}
\mathbf{if} (\text{modesLeft} = 3) \longrightarrow \\
\quad \left( \begin{array}{l} \text{modesLeft} := \text{modesLeft} - 1; \\ \text{ret} := \text{TakeOffMission} \end{array} \right) \\
\Box \neg (\text{modesLeft} = 3) \longrightarrow \\
\quad \mathbf{if} (\text{modesLeft} = 2) \longrightarrow \\
\quad \quad \left( \begin{array}{l} \text{modesLeft} := \text{modesLeft} - 1; \\ \text{ret} := \text{CruiseMission} \end{array} \right) \\
\Box \neg (\text{modesLeft} = 2) \longrightarrow \\
\quad \mathbf{if} (\text{modesLeft} = 1) \longrightarrow \\
\quad \quad \left( \begin{array}{l} \text{modesLeft} := \text{modesLeft} - 1; \\ \text{ret} := \text{LandMission} \end{array} \right) \\
\Box \neg (\text{modesLeft} = 1) \longrightarrow \\
\quad (\text{ret} := \text{nullMissionId}) \\
\mathbf{fi} \\
\mathbf{fi} \\
\mathbf{fi}
\end{array} \right) ; \\
\text{getNextMissionRet} . \text{ACModeChanger} ! \text{ret} \longrightarrow \\
\mathbf{Skip}
\end{array} \right)
\end{aligned}$$

$$\begin{aligned}
& \text{advanceModeSyncMeth} \hat{=} \\
& \left( \begin{array}{l}
\text{advanceModeCall} . \text{ACModeChanger} ? \text{thread} \longrightarrow \\
\left( \begin{array}{l}
\text{startSyncMeth} . \text{ACModeChangerObject} . \text{thread} \longrightarrow \\
\text{lockAcquired} . \text{ACModeChangerObject} . \text{thread} \longrightarrow \\
\left( \begin{array}{l}
\mathbf{Skip}; \\
\mathbf{if} (\text{modesLeft} = 3) \longrightarrow \\
\quad \left( \begin{array}{l} \text{modesLeft} := \text{modesLeft} - 1; \\ \text{changeTo}(\text{launchMode}) \end{array} \right) \\
\Box \neg (\text{modesLeft} = 3) \longrightarrow \\
\quad \mathbf{if} (\text{modesLeft} = 2) \longrightarrow \\
\quad \quad \left( \begin{array}{l} \text{modesLeft} := \text{modesLeft} - 1; \\ \text{changeTo}(\text{cruiseMode}) \end{array} \right) \\
\Box \neg (\text{modesLeft} = 2) \longrightarrow \\
\quad \mathbf{if} (\text{modesLeft} = 1) \longrightarrow \\
\quad \quad \left( \begin{array}{l} \text{modesLeft} := \text{modesLeft} - 1; \\ \text{changeTo}(\text{landMode}) \end{array} \right) \\
\Box \neg (\text{modesLeft} = 1) \longrightarrow \\
\quad (\text{changeTo}(\mathbf{null})) \\
\mathbf{fi} \\
\mathbf{fi} \\
\mathbf{fi}
\end{array} \right) ; \\
\text{endSyncMeth} . \text{ACModeChangerObject} . \text{thread} \longrightarrow \\
\text{advanceModeRet} . \text{ACModeChanger} . \text{thread} \longrightarrow \\
\mathbf{Skip}
\end{array} \right)
\end{aligned}$$

$$\begin{aligned}
& \text{Methods} \hat{=} \\
& \left( \begin{array}{l}
\text{GetNextMission} \\
\Box \\
\text{changeToMeth} \\
\Box \\
\text{getNextMissionMeth} \\
\Box \\
\text{advanceModeSyncMeth}
\end{array} \right) ; \text{Methods}
\end{aligned}$$

$$\bullet (\text{Init} ; \text{Methods}) \triangle (\text{end\_sequencer\_app} . \text{ACModeChanger} \longrightarrow \mathbf{Skip})$$

**end**

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

◦ **sync** *advanceMode*  $\hat{=}$

$$\left( \begin{array}{l} ; \\ \textbf{if} (modesLeft = 3) \longrightarrow \\ \quad \left( \begin{array}{l} modesLeft := modesLeft - 1; \\ changeTo(launchMode) \end{array} \right) \\ \parallel \neg (modesLeft = 3) \longrightarrow \\ \quad \textbf{if} (modesLeft = 2) \longrightarrow \\ \quad \quad \left( \begin{array}{l} modesLeft := modesLeft - 1; \\ changeTo(cruiseMode) \end{array} \right) \\ \parallel \neg (modesLeft = 2) \longrightarrow \\ \quad \textbf{if} (modesLeft = 1) \longrightarrow \\ \quad \quad \left( \begin{array}{l} modesLeft := modesLeft - 1; \\ changeTo(landMode) \end{array} \right) \\ \parallel \neg (modesLeft = 1) \longrightarrow \\ \quad \quad (changeTo(\textbf{null})) \\ \textbf{fi} \\ \textbf{fi} \\ \textbf{fi} \end{array} \right)$$

• **Skip**

**end**

**section** *ACModeChangerMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

**channel** *changeToCall* : *SchedulableID* ×  
**channel** *changeToRet* : *SchedulableID*

**channel** *getNextMissionCall* : *SchedulableID*  
**channel** *getNextMissionRet* : *SchedulableID* × *MissionID*

**channel** *advanceModeCall* : *SchedulableID* × *ThreadID*  
**channel** *advanceModeRet* : *SchedulableID* × *ThreadID*

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

- **Skip**

**end**

**section** *ControlHandlerMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

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

- **Skip**

**end**



**section** *CommunicationsHandlerMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

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

- **Skip**

**end**

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

- **Skip**

**end**

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

- **Skip**

**end**

### 5.3 TakeOffMission

**section** *TakeOffMissionApp* **parents** *scj\_prelude*, *MissionId*, *MissionIds*,  
*SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *TakeOffMissionClass* , *TakeOffMissionMethC*

**process** *TakeOffMissionApp*  $\hat{=}$  *storageParametersSchedulable* : *MissionID*, *landingGearHandler* : *MissionID*, *takeOffMon*

<i>State</i> <i>this</i> : <b>ref</b> <i>TakeOffMissionClass</i>
---

**state** *State*

<i>Init</i> <i>State'</i>
<i>this'</i> = <b>new</b> <i>TakeOffMissionClass</i> ()

*InitializePhase*  $\hat{=}$   
 $\left( \begin{array}{l} \text{initializeCall} . \text{TakeOffMission} \longrightarrow \\ \text{register} ! \text{LandingGearHandlerTakeOff} ! \text{TakeOffMission} \longrightarrow \\ \text{register} ! \text{TakeOffMonitor} ! \text{TakeOffMission} \longrightarrow \\ \text{register} ! \text{TakeOffFailureHandler} ! \text{TakeOffMission} \longrightarrow \\ \text{initializeRet} . \text{TakeOffMission} \longrightarrow \\ \text{Skip} \end{array} \right)$

*CleanupPhase*  $\hat{=}$   
 $\left( \begin{array}{l} \text{cleanupMissionCall} . \text{TakeOffMission} \longrightarrow \\ \text{cleanupMissionRet} . \text{TakeOffMission} ! \text{True} \longrightarrow \\ \text{Skip} \end{array} \right)$

*abortMeth*  $\hat{=}$   
 $\left( \begin{array}{l} \text{abortCall} . \text{TakeOffMission} \longrightarrow \\ \text{this} . \text{abort}(); \\ \text{abortRet} . \text{TakeOffMission} \longrightarrow \\ \text{Skip} \end{array} \right)$

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

*setControllingMissionMeth*  $\hat{=}$   
 $\left( \begin{array}{l} \text{setControllingMissionCall} . \text{TakeOffMission} ? \text{controllingMission} \longrightarrow \\ \text{this} . \text{setControllingMission}(\text{controllingMission}); \\ \text{setControllingMissionRet} . \text{TakeOffMission} \longrightarrow \\ \text{Skip} \end{array} \right)$

*cleanUpMeth*  $\hat{=}$  **var** *ret* :  $\mathbb{B}$  •  
 $\left( \begin{array}{l} \text{cleanUpCall} . \text{TakeOffMission} \longrightarrow \\ \text{ret} := \text{this} . \text{cleanUp}(); \\ \text{cleanUpRet} . \text{TakeOffMission} ! \text{ret} \longrightarrow \\ \text{Skip} \end{array} \right)$

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

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

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

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

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

**end**

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

**state** *State*

---

*SAFE\_AIRSPEED\_THRESHOLD* : *double*  
*TAKEOFF\_ALTITUDE* : *double*  
*abort* :  $\mathbb{B}$   
*landingGearDeployed* :  $\mathbb{B}$

---

**state** *State*

**initial** *Init*

---

*State'*  
  
*SAFE\_AIRSPEED\_THRESHOLD'* = 10.0  
*TAKEOFF\_ALTITUDE'* = 10.0  
*abort'* = *false*

---

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

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

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

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

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

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

• **Skip**

**end**

**section** *TakeOffMissionMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

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

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

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

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

**channel** *stowLandingGearCall* : *MissionID*  
**channel** *stowLandingGearRet* : *MissionID*

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

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



5.4 Schedulables of

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

- **Skip**

**end**

**section** *LandingGearHandlerTakeOffMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

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

- **Skip**

**end**

**section** *TakeOffFailureHandlerMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

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

- **Skip**

**end**

## 5.5 CruiseMission

**section** *CruiseMissionApp* **parents** *scj\_prelude*, *MissionId*, *MissionIds*,  
*SchedulableId*, *SchedulableIds*, *MissionChan*, *SchedulableMethChan*, *CruiseMissionClass* , *CruiseMissionMethChan*

**process** *CruiseMissionApp*  $\hat{=}$  *storageParametersSchedulable* : *MissionID*, *beginLandingHandler* : *MissionID*, *navigationM*

---

*State*  
*this* : **ref** *CruiseMissionClass*

---

**state** *State*

---

*Init*  
*State'*  
*this'* = **new** *CruiseMissionClass*()

---

*InitializePhase*  $\hat{=}$   
 $\left( \begin{array}{l} \textit{initializeCall} . \textit{CruiseMission} \longrightarrow \\ \textit{register} ! \textit{BeginLandingHandler} ! \textit{CruiseMission} \longrightarrow \\ \textit{register} ! \textit{NavigationMonitor} ! \textit{CruiseMission} \longrightarrow \\ \textit{initializeRet} . \textit{CruiseMission} \longrightarrow \\ \textbf{Skip} \end{array} \right)$

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

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

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

• (*Init* ; *Methods*)  $\triangle$  (*end\_mission\_app* . *CruiseMission*  $\longrightarrow$  **Skip**)

**end**

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

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

• **Skip**

**end**



**section** *CruiseMissionMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

**channel** *getControllingMissionCall* : *MissionID*

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

5.6 Schedulables of

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

- **Skip**

**end**

**section** *BeginLandingHandlerMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

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

- **Skip**

**end**

## 5.7 LandMission

**section** *LandMissionApp* **parents** *scj\_prelude, MissionId, MissionIds, SchedulableId, SchedulableIds, MissionChan, SchedulableMethChan, LandMissionClass* , *LandMissionMethChan*

**process** *LandMissionApp*  $\hat{=}$  *storageParametersSchedulable : MissionID, groundDistanceMonitor : MissionID, landingHar*

*State*

*this* : **ref** *LandMissionClass*

**state** *State*

*Init*

*State'*

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

*InitializePhase*  $\hat{=}$

$$\left( \begin{array}{l} \text{initializeCall} . \text{LandMission} \longrightarrow \\ \text{register} ! \text{GroundDistanceMonitor} ! \text{LandMission} \longrightarrow \\ \text{register} ! \text{LandingGearHandlerLand} ! \text{LandMission} \longrightarrow \\ \text{register} ! \text{InstrumentLandingSystemMonitor} ! \text{LandMission} \longrightarrow \\ \text{register} ! \text{SafeLandingHandler} ! \text{LandMission} \longrightarrow \\ \text{initializeRet} . \text{LandMission} \longrightarrow \\ \text{Skip} \end{array} \right)$$

*CleanupPhase*  $\hat{=}$

$$\left( \begin{array}{l} \text{cleanupMissionCall} . \text{LandMission} \longrightarrow \\ \text{cleanupMissionRet} . \text{LandMission} ! \text{True} \longrightarrow \\ \text{Skip} \end{array} \right)$$

*stowLandingGearMeth*  $\hat{=}$

$$\left( \begin{array}{l} \text{stowLandingGearCall} . \text{LandMission} \longrightarrow \\ \text{this} . \text{stowLandingGear}(); \\ \text{stowLandingGearRet} . \text{LandMission} \longrightarrow \\ \text{Skip} \end{array} \right)$$

*isLandingGearDeployedMeth*  $\hat{=}$  **var** *ret* :  $\mathbb{B}$  •

$$\left( \begin{array}{l} \text{isLandingGearDeployedCall} . \text{LandMission} \longrightarrow \\ \text{ret} := \text{this} . \text{isLandingGearDeployed}(); \\ \text{isLandingGearDeployedRet} . \text{LandMission} ! \text{ret} \longrightarrow \\ \text{Skip} \end{array} \right)$$

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

$$\left( \begin{array}{l} \text{getControllingMissionCall} . \text{LandMission} \longrightarrow \\ \text{ret} := \text{this} . \text{getControllingMission}(); \\ \text{getControllingMissionRet} . \text{LandMission} ! \text{ret} \longrightarrow \\ \text{Skip} \end{array} \right)$$

*abortMeth*  $\hat{=}$

$$\left( \begin{array}{l} \text{abortCall} . \text{LandMission} \longrightarrow \\ \text{this} . \text{abort}(); \\ \text{abortRet} . \text{LandMission} \longrightarrow \\ \text{Skip} \end{array} \right)$$

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

$$\text{deployLandingGearSyncMeth} \hat{=} \left( \begin{array}{l} \text{deployLandingGearCall} . \text{LandMission} ? \text{thread} \longrightarrow \\ \left( \begin{array}{l} \text{startSyncMeth} . \text{LandMissionObject} . \text{thread} \longrightarrow \\ \text{lockAcquired} . \text{LandMissionObject} . \text{thread} \longrightarrow \\ (\text{this} . \text{landingGearDeployed} := \text{true}); \\ \text{endSyncMeth} . \text{LandMissionObject} . \text{thread} \longrightarrow \\ \text{deployLandingGearRet} . \text{LandMission} . \text{thread} \longrightarrow \end{array} \right) \\ \mathbf{Skip} \end{array} \right)$$

$$\text{Methods} \hat{=} \left( \begin{array}{l} \text{InitializePhase} \\ \square \\ \text{CleanupPhase} \\ \square \\ \text{stowLandingGearMeth} \\ \square \\ \text{isLandingGearDeployedMeth} \\ \square \\ \text{getControllingMissionMeth} \\ \square \\ \text{abortMeth} \\ \square \\ \text{cleanUpMeth} \\ \square \\ \text{deployLandingGearSyncMeth} \end{array} \right); \text{Methods}$$

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

**end**

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

**state** *State*

*SAFE\_LANDING\_ALTITUDE* : *double*

*abort* :  $\mathbb{B}$

*landingGearDeployed* :  $\mathbb{B}$

**state** *State*

**initial** *Init*

*State'*

*SAFE\_LANDING\_ALTITUDE'* = 10.0

*abort'* = *false*

**public** *stowLandingGear*  $\hat{=}$

(*this* . *landingGearDeployed* := *false*)

**public** *isLandingGearDeployed*  $\hat{=}$  **var** *ret* :  $\mathbb{B}$  •

(*ret* := *landingGearDeployed* = **True**)

**public** *getControllingMission*  $\hat{=}$  **var** *ret* : *MissionID* •

(*ret* := *controllingMission*)

**public** *abort*  $\hat{=}$

(*this* . *abort* := *true*)

**public** *cleanUp*  $\hat{=}$  **var** *ret* :  $\mathbb{B}$  •

(**Skip**;  
*ret* := ( $\neg$  *abort* = **True**))

• **Skip**

**end**



**section** *LandMissionMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

**channel** *stowLandingGearCall* : *MissionID*  
**channel** *stowLandingGearRet* : *MissionID*

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

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

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

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

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

5.8 Schedulables of

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

- **Skip**

**end**

**section** *LandingGearHandlerLandMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

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

- **Skip**

**end**

**section** *SafeLandingHandlerMethChan* **parents** *scj\_prelude, GlobalTypes, MissionId, SchedulableId*

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

- **Skip**

**end**

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

- **Skip**

**end**