An Event-B Specification of LandingSysDP_DOOR_A
Creation Date: 11Nov2013 @ 10:04:52 PM

MACHINE LandingSysDP_DOOR_A
Digital Part + Environement = DOOR
REFINES LandingSysDP_SWITCH_A
SEES Landing_DP_DOOR_Ctx
VARIABLES

$analogical\_switch$
$anomaly$
$circuit\_pressurized$
$close\_EV$
$door\_closed$
$door\_open$
$extend\_EV$
$gSensorState$
$gear\_extended$
$gear\_retracted$
$gear\_shock\_absorber$
$gears\_locked\_down$
$gears\_maneuvering$
$general\_EV$
$greenLight$
$handle$
$nextOGseq$
$nextRseq$
$open\_EV$
$orangeLight$
$order$
$redLight$
$retract\_EV$
$sequenceStep$
sw_handle the new variables are :
$doorState$ door States

INVARIANTS

`defDoorSt` : $doorState \in DOOR \rightarrow DSTATE$

EVENTS
Initialisation
extended
begin

`gl` : $greenLight := lightOFF$
`ol` : $orangeLight := lightOFF$
`rl` : $redLight := lightOFF$
`inigears` : $gears\_locked\_down := FALSE$
`initManeuver` : $gears\_maneuvering := FALSE$
`iniAnalSW` : $analogical\_switch := TRIPLE \times \{openSW\}$
`iniGE` : $gear\_extended := (TRIPLE \times GEAR) \times \{FALSE\}$
`iniGR` : $gear\_retracted := (TRIPLE \times GEAR) \times \{FALSE\}$
`iniGSK1` : $gear\_shock\_absorber := (TRIPLE \times GEAR) \times \{FALSE\}$
`iniDC` : $door\_closed := (TRIPLE \times DOOR) \times \{TRUE\}$
`iniDO` : $door\_open := (TRIPLE \times DOOR) \times \{FALSE\}$
`iniAnomaly` : $anomaly := FALSE$

iniCP : $circuit\_pressurized := TRIPLE \times \{FALSE\}$
iniNextOGseq : $nextOGseq := 1$
iniNextRseq : $nextRseq := 1$
iniSequenceStep : $sequenceStep := 1$
hs : $handle := TRIPLE \times \{hDown\}$
cm : $order := hNop$
iniGEV : $general\_EV := FALSE$
iniOEV : $open\_EV := FALSE$
iniCEV : $close\_EV := FALSE$
iniREV : $retract\_EV := FALSE$
iniEEV : $extend\_EV := FALSE$
iniVGS : $gSensorState := (TRIPLE \times GEAR) \times \{validGS\}$
@iniInvalidGS invalidGSensors := ∅
iniShdl : $sw\_handle := hDown$

```
event cockP_handleDown // the pilot moves the handle Down
  extends downG
    then
      @a3 handle := TRIPLE× { hDown}
      @a2 sw_handle := hDown
  end
  event cockP_handleUp // the pilot moves the handle Up
    then
      @a1 handle := TRIPLE× { hUp}
      @a2 sw_handle := hUp
  end
```

ini1 : $doorState := DOOR \times \{notOpenLocked\}$

end

Event $downG \mathrel{\widehat{=}}$

page 14 When the gears are locked in retracted position,
and the doors are locked in closed position,
if the pilot sets the handle to "Down",
then the software should have the outgoing sequence actions

extends $downG$

when

gGR : $ran((gSensorState^{-1}[\{validGS\}]) \lhd gear\_extended) = \{FALSE\}$
all the VALID gears are locked in retracted position,
ran((gSensorState^{-1} [{ validGS}]) ◁ gear_extended) are the valid gears
g1L : $ran(door\_closed) = \{TRUE\}$
all doors are locked in closed position,
gh : $ran(handle) = \{hDown\}$
the pilot sets the handle to "Down",
gano : $anomaly = FALSE$
no anomaly detected

then

a1 : $order := hDown$
a2 : $analogical\_switch := TRIPLE \times \{openSW\}$
a3 : $sw\_handle := hDown$

end

Event $upG \mathrel{\widehat{=}}$

when the gears are locked in down position
and the doors are loced in closed position
if the pilot sets the handle to "Up"
then the software should have the retroaction sequence

extends upG

when

g1GE : $ran((gSensorState^{-1}[\{validGS\}]) \lhd gear\_extended) = \{TRUE\}$
the VALID gears are locked in down position

g1DC : $ran(door\_closed) = \{TRUE\}$
the doors are locked in closed position
gh : $ran(handle) = \{hUp\}$
if the pilot sets the handle to "Up"
gano : $anomaly = FALSE$
no anomaly detected

then

a1 : $order := hUp$
a2 : $analogical\_switch := TRIPLE \times \{openSW\}$
a3 : $sw\_handle := hUp$

end

Event *swe_closeSwitch* $\widehat{=}$

20 second after

extends *swe_closeSwitch*

begin

a1 : $analogical\_switch := TRIPLE \times \{closedSW\}$

end

Event *cockp_emergency_detection* $\widehat{=}$

extends *cockp_emergency_detection*

when

g1 : $anomaly = TRUE$
at leats one anomaly

then

a1 : $redLight := lightON$

end

Event *stmlt_general_EV* $\widehat{=}$

Control System events -
Simulate general electro valve isolation
** action1** of of Outgoing sequence

extends *stmlt_general_EV*

when

g1 : $ran(handle) = \{hDown\}$
all 3 inputs of handle are hDown
g2 : $order = hDown$
g3 : $ran((gSensorState^{-1}[\{validGS\}]) \lhd gear\_extended) = \{FALSE\}$
all the VALID gears are locked retracted
g4 : $ran(door\_closed) = \{TRUE\}$
all the doors are closed locked
g5 : $ran(analogical\_switch) = \{closedSW\}$
the analogical circuit should be closed
gano : $anomaly = FALSE$
no anomaly detected
notLast : $nextOGseq + sequenceStep < 8$
notLeast : $1 < nextOGseq + sequenceStep$

then

a1 : $order := hDown$
a2 : $general\_EV := TRUE$
a3 : $nextOGseq := nextOGseq + sequenceStep$

end

Event *stmlt_door_Opening* $\widehat{=}$

stimulate door opening electro valve
action2 of Outgoing sequence

extends *stmlt_door_Opening*

when

g0 : $general\_EV = TRUE$

g1 : $order = hDown$
g2 : $ran(handle) = \{hDown\}$
next : $nextOGseq = 2$
gano : $anomaly = FALSE$

no anomaly detected

notLast : $nextOGseq + sequenceStep < 8$
notClose : $close\_EV = FALSE$

then

a1 : $open\_EV := TRUE$
a2 : $nextOGseq := nextOGseq + sequenceStep$

3 or 1

end

Event *stmlt_gear_outgoing* $\widehat{=}$

stimulate gear outgoing electro valve
** action 3 ** of outgoing sequence
once the three doors are in the open position

extends *stmlt_gear_outgoing*

when

g0 : $general\_EV = TRUE$
g1 : $order = hDown$
g2 : $ran(handle) = \{hDown\}$
g3 : $ran(door\_closed) = \{FALSE\}$

the three doors are in the open position

next : $nextOGseq = 3$
gano : $anomaly = FALSE$

no anomaly detected

notretract : $retract\_EV = FALSE$

then

a1 : $extend\_EV := TRUE$
a2 : $nextOGseq := nextOGseq + sequenceStep$

4 or 2

end

Event *stop_stmlt_gear_outgoing* $\widehat{=}$

stop stimulating gear outgoing electro valve
** action 4 ** of outgoing sequence
once the three gears are locked down

extends *stop_stmlt_gear_outgoing*

when

g0 : $general\_EV = TRUE$
g1 : $order = hDown$
g2 : $ran(handle) = \{hDown\}$
g3 : $ran((gSensorState^{-1}[\{validGS\}]) \lhd gear\_extended) = \{TRUE\}$

the three gears are locked down

next : $nextOGseq = 4$
gano : $anomaly = FALSE$

no anomaly detected

then

a1 : $extend\_EV := FALSE$
a2 : $nextOGseq := nextOGseq + sequenceStep$

5 or 3

end

Event *stop_stmlt_door_opening* $\widehat{=}$

stop stimulating door opening electro valve
** action 5 ** of outgoing sequence

extends *stop_stmlt_door_opening*

when

```
            g0 : $general\_EV = TRUE$
            g1 : $order = hDown$
            g2 : $ran(handle) = \{hDown\}$
            next : $nextOGseq = 5$
            gano : $anomaly = FALSE$
                  no anomaly detected
      then
            a1 : $open\_EV := FALSE$
            a2 : $nextOGseq := nextOGseq + sequenceStep$
      end
Event   $stmlt\_door\_closing \mathrel{\widehat{=}}$
                                    stimulating door closing electro valve
                                        ** action 6 ** of outgoing sequence
extends  $stmlt\_door\_closing$
      when
            g0 : $general\_EV = TRUE$
            g1 : $order = hDown$
            g2 : $ran(handle) = \{hDown\}$
            next : $nextOGseq = 6$
            notopen : $open\_EV = FALSE$
            gano : $anomaly = FALSE$
                  no anomaly detected
      then
            a1 : $close\_EV := TRUE$
            a2 : $nextOGseq := nextOGseq + sequenceStep$
                7 or 5
      end
Event   $stop\_stmlt\_door\_closing \mathrel{\widehat{=}}$
                                          stop stimulating door closing electro valve
                                              ** action 7 ** of outgoing sequence
                                              once the three doors are locked in the closed position
extends  $stop\_stmlt\_door\_closing$
      when
            g0 : $general\_EV = TRUE$
            g1 : $order = hDown$
            g2 : $ran(handle) = \{hDown\}$
            g3 : $ran(door\_closed) = \{TRUE\}$
                the three doors are locked in the closed position
            next : $nextOGseq = 7$
            gano : $anomaly = FALSE$
                  no anomaly detected
            notLast : $nextOGseq + sequenceStep < 8$
      then
            a1 : $close\_EV := FALSE$
            a2 : $nextOGseq := nextOGseq + sequenceStep$
                8 or 6
      end
Event   $stop\_stmlt\_general\_ev \mathrel{\widehat{=}}$
extends  $stop\_stmlt\_general\_ev$
      when
            g0 : $general\_EV = TRUE$
            g1 : $order = hDown$
            g2 : $ran(handle) = \{hDown\}$
            next : $nextOGseq = 8$
```

gano : $anomaly = FALSE$
no anomaly detected
EV : $\neg(open\_EV = TRUE$
$\vee close\_EV = TRUE$
$\vee extend\_EV = TRUE$
$\vee retract\_EV = TRUE)$

then

a1 : $general\_EV := FALSE$
a2 : $nextOGseq := 1$
a3 : $sequenceStep := 1$
to be checked

end

Event *orderDownInterruption* $\widehat{=}$
extends *orderDownInterruption*

when

g1 : $order = hDown$
g2 : $ran(handle) = \{hUp\}$
change or interruption of the order
gano : $anomaly = FALSE$
no anomaly detected
defseq : $-(sequenceStep) \in \{-1, 1\}$

then

a1 : $sequenceStep := -(sequenceStep)$
a2 : $order := hUp$
RIGHT ??????? To be CHECKED

end

Event *orderUpInterruption* $\widehat{=}$
extends *orderUpInterruption*

when

g1 : $order = hUp$
g2 : $ran(handle) = \{hDown\}$
change or interruption of the order
gano : $anomaly = FALSE$
no anomaly detected
defseq : $-(sequenceStep) \in \{-1, 1\}$

then

a1 : $sequenceStep := -(sequenceStep)$

end

Event *stmlt_general_ev_RSeq* $\widehat{=}$

Control System events -
Stimulate general electro valve isolation
** action1** of retraction sequence
when the gears are locked in down position, and the doors are locked in closed position
and the pilot stes the handle to UP

extends *stmlt_general_ev_RSeq*

when

g1 : $ran(handle) = \{hUp\}$
g2 : $order = hUp$
g3 : $ran((gSensorState^{-1}[\{validGS\}]) \lhd gear\_extended) = \{TRUE\}$
all the VALID gears are locked extended (down)
gSensorState[{ validGS}] ◁ gear_extended are the valid gears
g4 : $ran(door\_closed) = \{TRUE\}$
all the doors are closed locked
gano : $anomaly = FALSE$
no anomaly detected

```
            notLeast : $1 < nextRseq + sequenceStep$
            notLast : $nextRseq + sequenceStep < 8$
      then
            a2 : $general\_EV := TRUE$
            a3 : $nextRseq := nextRseq + sequenceStep$
      end
Event  stmlt_door_open_ev_RSEQ ≙
                                  Control System event
                                        Stimulate the door opening electro valve
                                        ** action 2 **
                                        order should be changed to hUp
extends  stmlt_door_open_ev_RSEQ
      when
            g1 : $order = hUp$
            g0 : $general\_EV = TRUE$
            g2 : $ran(handle) = \{hUp\}$
            g3 : $nextRseq = 2$
            gano : $anomaly = FALSE$
                  no anomaly detected
            notclose : $close\_EV = FALSE$
      then
            a1 : $order := hUp$
            a2 : $open\_EV := TRUE$
            a3 : $nextRseq := nextRseq + sequenceStep$
      end
Event  stmlt_gear_retraction_RSEQ ≙
                                  stimulate the gear retraction
                                        ** action 3 ** retraction sequence
                                        once the three doors are in open position, if the three
      shock absorbers are relaxed
extends  stmlt_gear_retraction_RSEQ
      when
            g1 : $order = hUp$
            g0 : $general\_EV = TRUE$
            g2 : $ran(handle) = \{hUp\}$
            g3 : $ran(door\_closed) = \{FALSE\}$
                  three doors open
            g4 : $nextRseq = 3$
            gano : $anomaly = FALSE$
                  no anomaly detected
            notExtend : $extend\_EV = FALSE$
      then
            a1 : $retract\_EV := TRUE$
            a2 : $nextRseq := nextRseq + sequenceStep$
      end
Event  stop_stmlt_gear_rectaction_RSEQ ≙
                                  stop stimulation of gear retraction
                                        ** action 4 **
                                        once the three gears are locked up
extends  stop_stmlt_gear_rectaction_RSEQ
      when
            g1 : $order = hUp$
            g0 : $general\_EV = TRUE$
            g2 : $ran(handle) = \{hUp\}$
```

g3 : $ran((gSensorState^{-1}[\{validGS\}]) \lhd gear\_extended) = \{FALSE\}$

once the three VALID gears are locked up

ran((gSensorState^{-1} [{ validGS}]) ◁ gear_extended) are the valid gears

g4 : $nextRseq = 5$

gano : $anomaly = FALSE$

no anomaly detected

then

a1 : $retract\_EV := FALSE$

a2 : $nextRseq := nextRseq + sequenceStep$

end

Event $stop\_stmlt\_door\_opening\_RSEQ \;\widehat{=}$

stop stimulation of door opening

** action 5 **

extends $stop\_stmlt\_door\_opening\_RSEQ$

when

g1 : $order = hUp$

g0 : $general\_EV = TRUE$

g2 : $ran(handle) = \{hUp\}$

g3 : $nextRseq = 5$

gano : $anomaly = FALSE$

no anomaly detected

then

a1 : $open\_EV := FALSE$

a2 : $nextRseq := nextRseq + sequenceStep$

end

Event $stmlt\_door\_closing\_RSEQ \;\widehat{=}$

stimulation of door closing

** action 6 **

extends $stmlt\_door\_closing\_RSEQ$

when

g0 : $general\_EV = TRUE$

g1 : $ran(handle) = \{hUp\}$

g3 : $nextRseq = 5$

gano : $anomaly = FALSE$

no anomaly detected

notOpen : $open\_EV = FALSE$

then

a1 : $close\_EV := TRUE$

a2 : $nextRseq := nextRseq + sequenceStep$

end

Event $stop\_stmlt\_door\_closing\_RSEQ \;\widehat{=}$

stop stimulation of door closing

** action 7 **

once the three doors are locked in the closed position

extends $stop\_stmlt\_door\_closing\_RSEQ$

when

g1 : $order = hUp$

g0 : $general\_EV = TRUE$

g2 : $ran(handle) = \{hUp\}$

g3 : $nextRseq = 5$

g4 : $ran(door\_closed) = \{TRUE\}$

the three doors are locked in the closed position

gano : $anomaly = FALSE$

no anomaly detected

```
          notOpen : $open\_EV = FALSE$
     then
          a1 : $close\_EV := TRUE$
          a2 : $nextRseq := nextRseq + sequenceStep$
     end
Event  stop_stmlt_general_RSEQ ≙
                                        the three doors are locked in the closed position
                                            finally stop stimulation of the general EV
                                            ** action 8 **
extends  stop_stmlt_general_RSEQ
     when
          g1 : $order = hUp$
          g0 : $general\_EV = TRUE$
          g2 : $ran(handle) = \{hUp\}$
          g3 : $nextRseq = 8$
          g4 : $ran(door\_closed) = \{TRUE\}$
                { notOpenLocked}
          gano : $anomaly = FALSE$
                  no anomaly detected
          notOpen : $open\_EV = FALSE$
     then
          a1 : $close\_EV := TRUE$
          a2 : $nextRseq := 1$
     end
Event  monitor_gears_locked_Down ≙
                                          page 7 : the outputs are synthesized by each module
                                              from sensors data and from the situation awareness
                                              page 15 : gear_locked_down = true iff the 3 gears are
     seen as locked

                                              in extended position
extends  monitor_gears_locked_Down
     when
          g1 : $ran(gear\_extended) = \{TRUE\}$
                the 3 gears are seen as locked
          gano : $anomaly = FALSE$
                  no anomaly detected
     then
          a1 : $gears\_locked\_down := TRUE$
          a2 : $greenLight := lightON$
          a3 : $orangeLight := lightOFF$
          a4 : $redLight := lightOFF$
     end
Event  monitor_gears_maneuvering ≙
                                          page 7 : the outputs are synthesized by each module
                                              from sensors data and from the situation awareness
                                              page 15 : gear_maneuvering = true iff at least one
     door or one gear is maneuvering

                                              i.e., at least one door is not locked in closed position
                                              or one gear is not locked in extension or retraction
     position
extends  monitor_gears_maneuvering
     when
          gano : $anomaly = FALSE$
                  no anomaly detected
```

g1 : $\neg(\neg(\{TRUE\} = ran(door\_closed)) \wedge \neg(\{TRUE\} = ran(gear\_extended)) \wedge \neg(\{FALSE\} = ran(gear\_extended)))$

one door is ¬ locked in closed position
OR at least one gear is not locked in extended position
OR at least one gear is not locked in retracted position
it is FALSE ∈ ran(gear_extended)
OR FALSE : ran(gear_extended)
OR at least one gear is not locked in extended position
@g3 (card(ran(gear_extended)) > 1) ∧ (TRUE ∈ ran(gear_extended)) // OR at least one gear is not locked in retracted position

then

a1 : $gears\_maneuvering := TRUE$
a2 : $orangeLight := lightON$
a3 : $redLight := lightOFF$
a4 : $greenLight := lightOFF$

end

Event *monitor_anomaly* $\widehat{=}$

page 7 : the outputs are synthesized by each module
from sensors data and from the situation awareness
page 15 : ref to section 4.3

extends *monitor_anomaly*

when

gano : $anomaly = FALSE$
no anomaly detected
finit : $finite(ran(gear\_extended))$
g2 : $card(ran(gear\_extended)) > 1$
at leats two different values

then

a1 : $anomaly := TRUE$
a2 : $redLight := lightON$
a3 : $orangeLight := lightOFF$
a4 : $greenLight := lightOFF$

end

Event *sense1_FDO_OK* $\widehat{=}$

——-DOOR OPEN—— sensor1,2,3 of FDoor open/nnot

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
g3 : $ndo = door\_open \Lleftarrow \{(1 \mapsto FD) \mapsto TRUE\}$
g1 : $doorState(FD) = open$

then

a1 : $door\_open := ndo$
(1 ↦ FD) := TRUE

end

Event *sense1_FDO_KO* $\widehat{=}$

sensor1 of FDoor (simulating malfunctioning)

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
g3 : $ndo = door\_open \Lleftarrow \{(1 \mapsto FD) \mapsto FALSE\}$

```
            g1 : $doorState(FD) = open$
    then
            a1 : $door\_open := ndo$
                 $(1 \mapsto \mathrm{FD}) := \mathrm{FALSE}$
    end
Event  $sense2\_FDO\_OK$ $\widehat{=}$
                                seonsor2 of FDoor
refines  $sense\_door$
    any
          $ndo$
    where
            g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
            g3 : $ndo = door\_open \mathbin{\lhd\!\!\!+} \{(2 \mapsto FD) \mapsto TRUE\}$
            g1 : $doorState(FD) = open$
    then
            a1 : $door\_open := ndo$
                 $(2 \mapsto \mathrm{FD}) := \mathrm{TRUE}$
    end
Event  $sense2\_FDO\_KO$ $\widehat{=}$
                                sensor2 of FDoor (simulating malfunctioning)
refines  $sense\_door$
    any
          $ndo$
    where
            g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
            g3 : $ndo = door\_open \mathbin{\lhd\!\!\!+} \{(2 \mapsto FD) \mapsto FALSE\}$
            g1 : $doorState(FD) = open$
    then
            a1 : $door\_open := ndo$
                 $(2 \mapsto \mathrm{FD}) := \mathrm{FALSE}$
    end
Event  $sense3\_FDO\_OK$ $\widehat{=}$
                                seonsor3 of FGear
refines  $sense\_door$
    any
          $ndo$
    where
            g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
            g3 : $ndo = door\_open \mathbin{\lhd\!\!\!+} \{(3 \mapsto FD) \mapsto TRUE\}$
            g1 : $doorState(FD) = open$
    then
            a1 : $door\_open := ndo$
                 $(3 \mapsto \mathrm{FD}) := \mathrm{TRUE}$
    end
Event  $sense3\_FDO\_KO$ $\widehat{=}$
                                sensor3 of FGear (simulating malfunctioning)
refines  $sense\_door$
    any
          $ndo$
    where
            g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
            g3 : $ndo = door\_open \mathbin{\lhd\!\!\!+} \{(3 \mapsto FD) \mapsto FALSE\}$
```

g1 : $doorState(FD) = open$

then

a1 : $door\_open := ndo$

(3 ↦ FD) := FALSE

end

Event $sense3\_FDC\_OK \mathrel{\widehat{=}}$

seonsor3 of FGear

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \to BOOL$

g3 : $ndc = door\_closed \mathbin{\lhd\mkern-9mu-} \{(3 \mapsto FD) \mapsto TRUE\}$

g1 : $doorState(FD) = open$

then

a1 : $door\_closed := ndc$

(3 ↦ FD) := TRUE

end

Event $sense3\_FDC\_KO \mathrel{\widehat{=}}$

sensor3 of FGear (simulating malfunctioning)

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \to BOOL$

g3 : $ndc = door\_closed \mathbin{\lhd\mkern-9mu-} \{(3 \mapsto FD) \mapsto FALSE\}$

g1 : $doorState(FD) = open$

then

a1 : $door\_closed := ndc$

(3 ↦ FD) := FALSE

end

Event $sense1\_RDC\_OK \mathrel{\widehat{=}}$

———- sensor1,2,3 of FDoor closed/notCLosed locked

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \to BOOL$

g3 : $ndc = door\_closed \mathbin{\lhd\mkern-9mu-} \{(1 \mapsto FD) \mapsto TRUE\}$

g1 : $doorState(FD) = notOpenLocked$

then

a1 : $door\_closed := ndc$

(1 ↦ FD) := TRUE

end

Event $sense1\_RDC\_KO \mathrel{\widehat{=}}$

sensor1 of FDoor (simulating malfunctioning)

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \to BOOL$

g3 : $ndc = door\_closed \mathbin{\lhd\mkern-9mu-} \{(1 \mapsto RD) \mapsto FALSE\}$

g1 : $doorState(RD) = notOpenLocked$

then

a1 : $door\_closed := ndc$

(1 $\mapsto$ RD) := FALSE

end

Event $sense2\_RDC\_OK \mathrel{\widehat{=}}$

seonsor2 of FDoor

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!+} \{(2 \mapsto RD) \mapsto TRUE\}$

g1 : $doorState(RD) = notOpenLocked$

then

a1 : $door\_closed := ndc$

(2 $\mapsto$ RD) := TRUE

end

Event $sense2\_RDC\_KO \mathrel{\widehat{=}}$

sensor2 of FDoor (simulating malfunctioning)

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!+} \{(2 \mapsto RD) \mapsto FALSE\}$

g1 : $doorState(RD) = notOpenLocked$

then

a1 : $door\_closed := ndc$

(2 $\mapsto$ RD) := FALSE

end

Event $sense3\_RDC\_OK \mathrel{\widehat{=}}$

seonsor3 of RDoor

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!+} \{(3 \mapsto RD) \mapsto TRUE\}$

g1 : $doorState(RD) = notOpenLocked$

then

a1 : $door\_closed := ndc$

(3 $\mapsto$ RD) := TRUE

end

Event $sense3\_RDC\_KO \mathrel{\widehat{=}}$

sensor3 of RDoor (simulating malfunctioning)

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!+} \{(3 \mapsto RD) \mapsto FALSE\}$

g1 : $doorState(RD) = notOpenLocked$

then

a1 : $door\_closed := ndc$

(3 $\mapsto$ RD) := FALSE

end

Event *sense1_LDO_OK* $\widehat{=}$

———— - sensor1,2,3 of LDoor open/nnot

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndo = door\_open \mathbin{\lhd\!\!\!+} \{(1 \mapsto LD) \mapsto TRUE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_open := ndo$

(1 $\mapsto$ LD) := TRUE

end

Event *sense1_LDO_KO* $\widehat{=}$

sensor1 of LDoor (simulating malfunctioning)

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndo = door\_open \mathbin{\lhd\!\!\!+} \{(1 \mapsto LD) \mapsto FALSE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_open := ndo$

(1 $\mapsto$ LD) := FALSE

end

Event *sense2_LDO_OK* $\widehat{=}$

seonsor2 of LDoor

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndo = door\_open \mathbin{\lhd\!\!\!+} \{(2 \mapsto LD) \mapsto TRUE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_open := ndo$

(2 $\mapsto$ LD) := TRUE

end

Event *sense2_LDO_KO* $\widehat{=}$

sensor2 of FDoor (simulating malfunctioning)

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndo = door\_open \mathbin{\lhd\!\!\!+} \{(2 \mapsto LD) \mapsto FALSE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_open := ndo$

(2 ↦ LD) := FALSE

end

Event *sense3_LDO_OK* $\widehat{=}$

seonsor3 of LDoo

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndo = door\_open \mathbin{<\!\!+} \{(3 \mapsto LD) \mapsto TRUE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_open := ndo$

(3 ↦ LD) := TRUE

end

Event *sense3_LDO_KO* $\widehat{=}$

sensor3 of LDoor (simulating malfunctioning)

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndo = door\_open \mathbin{<\!\!+} \{(3 \mapsto LD) \mapsto FALSE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_open := ndo$

(3 ↦ LD) := FALSE

end

Event *sense3_LDC_OK* $\widehat{=}$

seonsor3 of LDoo

refines *sense_door_close*

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!+} \{(3 \mapsto LD) \mapsto TRUE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_closed := ndc$

(3 ↦ LD) := TRUE

end

Event *sense3_LDC_KO* $\widehat{=}$

sensor3 of LDoor (simulating malfunctioning)

refines *sense_door_close*

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!+} \{(3 \mapsto LD) \mapsto FALSE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_closed := ndc$

(3 ↦ LD) := FALSE

end

Event $Door\_openDoor\_cl2cu \mathrel{\widehat{=}}$

the three doors

Door's Behaviour

first transition of the Door automata

when the action open_EV is performed by the control system

when

g1 : $open\_EV = TRUE$

all doors EV are on

g2 : $ran(doorState) = \{notOpenLocked\}$

then

a1 : $doorState := DOOR \times \{notOpenNotLocked\}$

door is being opened

end

Event $Door\_cu2ou \mathrel{\widehat{=}}$

door's behaviour

closed unlocked to open unlock

when

g1 : $ran(doorState) = \{notOpenNotLocked\}$

g2 : $open\_EV = TRUE$

all doors EV remains on

then

a1 : $doorState := DOOR \times \{open\}$

end

Event $Door\_ou2ou \mathrel{\widehat{=}}$

when

g1 : $open\_EV = TRUE$

all doors EV are on

g2 : $ran(doorState) = \{open\}$

then

a1 : $doorState := DOOR \times \{open\}$

stay open unlocked

end

Event $ou2cu \mathrel{\widehat{=}}$

door's behaviour

open unlocked to closed unlock

when

g1 : $ran(doorState) = \{open\}$

g2 : $close\_EV = TRUE$

then

a1 : $doorState := DOOR \times \{notOpenNotLocked\}$

end

Event $cu2cl \mathrel{\widehat{=}}$

door's behaviour

closed unlocked to closed locked

when

g1 : $ran(doorState) = \{notOpenNotLocked\}$

g2 : $close\_EV = TRUE$

then

a1 : $doorState := DOOR \times \{notOpenLocked\}$

last back transition

end

Event $sense1\_FDC\_OK \mathrel{\widehat{=}}$

——-DOOR CLOSED—— sensor1,2,3 of FDoor open/nnot

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{\triangleleft\!\!\!-} \{(1 \mapsto FD) \mapsto TRUE\}$

g1 : $doorState(FD) = open$

then

a1 : $door\_closed := ndc$

$(1 \mapsto \text{FD}) := \text{TRUE}$

end

Event $sense1\_FDC\_KO \mathrel{\widehat{=}}$

sensor1 of FDoor (simulating malfunctioning)

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{\triangleleft\!\!\!-} \{(1 \mapsto FD) \mapsto FALSE\}$

g1 : $doorState(FD) = open$

then

a1 : $door\_closed := ndc$

$(1 \mapsto \text{FD}) := \text{FALSE}$

end

Event $sense2\_FDC\_OK \mathrel{\widehat{=}}$

seonsor2 of FDoor

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{\triangleleft\!\!\!-} \{(2 \mapsto FD) \mapsto TRUE\}$

g1 : $doorState(FD) = open$

then

a1 : $door\_closed := ndc$

$(2 \mapsto \text{FD}) := \text{TRUE}$

end

Event $sense2\_FDC\_KO \mathrel{\widehat{=}}$

sensor2 of FDoor (simulating malfunctioning)

refines $sense\_door\_close$

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{\triangleleft\!\!\!-} \{(2 \mapsto FD) \mapsto FALSE\}$

g1 : $doorState(FD) = open$

then

a1 : $door\_closed := ndc$

(2 ↦ FD) := FALSE

end

Event *sense1_LDC_OK* $\widehat{=}$

——CLOSED——- sensor1,2,3 of LDoor open/nnot

refines *sense_door_close*

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!\!+} \{(1 \mapsto LD) \mapsto TRUE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_closed := ndc$

(1 ↦ LD) := TRUE

end

Event *sense1_LDC_KO* $\widehat{=}$

sensor1 of LDoor (simulating malfunctioning)

refines *sense_door_close*

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!\!+} \{(1 \mapsto LD) \mapsto FALSE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_closed := ndc$

(1 ↦ LD) := FALSE

end

Event *sense2_LDC_OK* $\widehat{=}$

seonsor2 of LDoor

refines *sense_door_close*

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!\!+} \{(2 \mapsto LD) \mapsto TRUE\}$

g1 : $doorState(LD) = open$

then

a1 : $door\_closed := ndc$

(2 ↦ LD) := TRUE

end

Event *sense2_LDC_KO* $\widehat{=}$

sensor2 of FDoor (simulating malfunctioning)

refines *sense_door_close*

any

ndc

where

g2 : $ndc \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndc = door\_closed \mathbin{<\!\!\!+} \{(2 \mapsto LD) \mapsto FALSE\}$

g1 : $doorState(LD) = open$
then

a1 : $door\_closed := ndc$
(2 ↦ LD) := FALSE
end

Event *sense1_RDO_OK* $\widehat{=}$
——-DOOR OPEN—— sensor1,2,3 of FDoor closed/notCLosed locked
refines *sense_door*
any

ndo
where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
g3 : $ndo = door\_open \mathbin{<\!\!+} \{(1 \mapsto FD) \mapsto TRUE\}$
g1 : $doorState(FD) = notOpenLocked$
then

a1 : $door\_open := ndo$
(1 ↦ FD) := TRUE
end

Event *sense1_RDO_KO* $\widehat{=}$
sensor1 of FDoor (simulating malfunctioning)
refines *sense_door*
any

ndo
where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
g3 : $ndo = door\_open \mathbin{<\!\!+} \{(1 \mapsto RD) \mapsto FALSE\}$
g1 : $doorState(RD) = notOpenLocked$
then

a1 : $door\_open := ndo$
(1 ↦ RD) := FALSE
end

Event *sense2_RDO_OK* $\widehat{=}$
seonsor2 of FDoor
refines *sense_door*
any

ndo
where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
g3 : $ndo = door\_open \mathbin{<\!\!+} \{(2 \mapsto RD) \mapsto TRUE\}$
g1 : $doorState(RD) = notOpenLocked$
then

a1 : $door\_open := ndo$
(2 ↦ RD) := TRUE
end

Event *sense2_RDO_KO* $\widehat{=}$
sensor2 of FDoor (simulating malfunctioning)
refines *sense_door*
any

ndo
where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$
g3 : $ndo = door\_open \mathbin{<\!\!+} \{(2 \mapsto RD) \mapsto FALSE\}$

g1 : $doorState(RD) = notOpenLocked$

then

a1 : $door\_open := ndo$

(2 ↦ RD) := FALSE

end

Event *sense3_RDO_OK* $\widehat{=}$

seonsor3 of RDoor

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndo = door\_open \mathbin{<\!\!+} \{(3 \mapsto RD) \mapsto TRUE\}$

g1 : $doorState(RD) = notOpenLocked$

then

a1 : $door\_open := ndo$

(3 ↦ RD) := TRUE

end

Event *sense3_RDO_KO* $\widehat{=}$

sensor3 of RDoor (simulating malfunctioning)

refines *sense_door*

any

ndo

where

g2 : $ndo \in (TRIPLE \times DOOR) \rightarrow BOOL$

g3 : $ndo = door\_open \mathbin{<\!\!+} \{(3 \mapsto RD) \mapsto FALSE\}$

g1 : $doorState(RD) = notOpenLocked$

then

a1 : $door\_open := ndo$

(3 ↦ RD) := FALSE

end

Event *eliminateOneOf3FGearSensor* $\widehat{=}$

when one of the three micro sensor has a value different from the other at time t

the common value of the others is used and the wrong sensor is invalid

extends *eliminateOneOf3FGearSensor*

any

gear

ci

cj

ck

validFGESensors

where

gano : $anomaly = FALSE$

no anomaly detected

defg : $gear \in GEAR$

gear is either FG, RG, LG

dvfg : $validFGESensors \subseteq TRIPLE \land finite(validFGESensors)$

vfges : $validFGESensors = dom(gSensorState^{-1}[\{validGS\}] \rhd \{gear\})$

ici FG au debut

g1 : $card(validFGESensors) = 3$

g2 : $ci \in TRIPLE \land ci \in validFGESensors$

channel ci is being invalid

g3 : $cj \in TRIPLE \wedge cj \in validFGESensors$
g4 : $ck \in TRIPLE \wedge ck \in validFGESensors$
g8 : $(ci \mapsto gear) \in dom(gear\_extended)$
g9 : $(ck \mapsto gear) \in dom(gear\_extended)$
g10 : $(cj \mapsto gear) \in dom(gear\_extended)$
dgs : $(ci \mapsto gear) \in dom(gSensorState)$
g5 : $gear\_extended(ci \mapsto gear) \neq gear\_extended(cj \mapsto gear)$
g6 : $gear\_extended(ci \mapsto gear) \neq gear\_extended(ck \mapsto gear)$
g7 : $gear\_extended(cj \mapsto gear) = gear\_extended(ck \mapsto gear)$

then

a1 : $gSensorState(ci \mapsto gear) := invalidGS$
@a2 validFGESensors := gSensorState^{-1} [{ validGS}]

end

Event *eliminateAGearSensor* $\widehat{=}$
extends *eliminateAGearSensor*

any

ci
cj
$gear$
$validFGESensors$

where

defg : $gear \in GEAR$
gear is either FG, RG, LG
dvfg : $validFGESensors \subseteq TRIPLE \wedge finite(validFGESensors)$
vfges : $validFGESensors = dom(gSensorState^{-1}[\{validGS\}] \rhd \{gear\})$
g1 : $card(validFGESensors) = 2$
there are already one invalid
g2 : $ci \in TRIPLE \wedge ci \in validFGESensors$
channel ci is being invalid
g3 : $cj \in TRIPLE \wedge cj \in validFGESensors$
g4 : $(ci \mapsto gear) \neq (cj \mapsto gear)$
g5 : $(ci \mapsto gear) \in dom(gSensorState) \wedge (cj \mapsto gear) \in dom(gSensorState)$
g6 : $gSensorState(ci \mapsto gear) \neq gSensorState(cj \mapsto gear)$
both values are different

then

a1 : $gSensorState := \{(ci \mapsto gear) \mapsto invalidGS, (cj \mapsto gear) \mapsto invalidGS\}$
@a3 gear_extended := (gSensorState^{-1} [{ validGS}]) $\lhd$ gear_extended

end

Event *anyInvalidGearSensor* $\widehat{=}$
extends *anyInvalidGearSensor*

any

$gear$
$validFGESensors$

where

defg : $gear \in GEAR$
dvfg : $validFGESensors \subseteq TRIPLE \wedge finite(validFGESensors)$
vfges : $validFGESensors = dom(gSensorState^{-1}[\{validGS\}] \rhd \{gear\})$
g1 : $card(validFGESensors) < 2$
less than two sensors valid
gano : $anomaly = FALSE$
no anomaly detected

then

a1 : $anomaly := TRUE$

end

Event *sense_gear* $\widehat{=}$

extends *sense_gear*

begin

a0 : $gear\_extended :\in (TRIPLE \times GEAR) \rightarrow BOOL$

end

Event *sense_gear_retracted* $\widehat{=}$

extends *sense_gear_retracted*

begin

a0 : $gear\_retracted :\in (TRIPLE \times GEAR) \rightarrow BOOL$

end

Event *monitor_door_motion* $\widehat{=}$

anticipated

extends *monitor_door_motion*

begin

a1 : $anomaly :\in BOOL$

end

END