```
GuiStates
SETS
    Modes
CONSTANTS
    {\sf ModesG}
   StartUp
    Start
    SelfTest
   Menu
    Settings
    Ventilation
    0ff
    possTransG
    PCV
    PSV
    batteryRange
AXIOMS
    axm1 : ModesG⊆Modes
    \verb|axm2| : partition(ModesG, {StartUp}, {Start}, {Menu}, {SelfTest}, {Settings}, {Ventilation}, {Off})| \\
           : partition(Ventilation,{PCV},{PSV})
    axm3
    axm4 :
                possTransG∈ B00L\rightarrowP(ModesG×ModesG)
                possTransG={TRUE→{
                         Off⇔StartUp,
                         StartUp \mapsto Start
                         }υ
                          ({StartUp}\times Ventilation) u
    axm5
                         {Start→Menu,Start→SelfTest,
                         {\tt Menu}{\mapsto} {\tt Settings}, \ {\tt SelfTest}{\mapsto} {\tt SelfTest},
                         SelfTest\mapstoMenu,Settings\mapstoMenu\} \cup ({Off}\timesVentilation) \cup
                          ({Menu,Settings}×Ventilation)υ
                             (Ventilation×({Menu,Settings}uVentilation)),
                         FALSE → (ModesG×{Off}))
           : batteryRange=0.120*60*10
    axm6
```

END

```
MACHINE
   GuiSM
SEES
   GuiStates
VARIABLES
   power
   onAC
   batLev
   switchover
   crashed
   modeG
   modeGP
   curTime
   batFail
INVARIANTS
              power∈ BOOL ∧ crashed ∈ BOOL ∧ batLev∈ batteryRange ∧ curTime∈N
   inv1 :
              ∧ batFail∈ B00L
               {\tt crashed=TRUE \ v \ power=FALSE \ v \ (onAC=FALSE \ v \ (switchover=FALSE \ v \ batLev=0 \ v \ batFail=TRUE))} 
   inv2
              modeG=Off
   inv3 :
              modeG∈ ModesG ∧ modeGP ∈ ModesG
              modeGP \neq modeG \Rightarrow modeGP \Rightarrow modeG \in
   inv4 :
                          possTransG(bool(power=TRUE ^ crashed=FALSE ^
                              (onAC=TRUE v (switchover=TRUE \( \text{batLev>0 \( \text{batFail=FALSE}))))
   inv5 : modeG=Settings⇒ modeGP∈{Menu}uVentilation
EVENTS
   INITIALISATION ≜
   STATUS
     ordinary
   BEGIN
               power≔FALSE
     act1
     act2 : batLev:∈batteryRange
     act3 : onAC≔TRUE
     act4 : switchover≔TRUE
     act5 : crashed≔FALSE
     act6 : modeG≔Off
     act7
           :
               modeGP≔Off
     act8 :
               curTime≔0
     act9 : batFail:∈B00L
   powerON ≜
                         // CONT.2
   STATUS
     ordinary
   WHEN
     grd1 : power=FALSE
     \verb|grd2|: onAC=TRUE v (switchover=TRUE \land batLev>0 \land batFail=FALSE)|
   THEN
     act1 :
                power≔TRUE
     act2 :
               modeG≔{FALSE→StartUp, TRUE→Off}(crashed)
     act3 : modeGP≔modeG
   END
   powerOff ≜
   STATUS
     ordinary
   WHEN
                power=TRUE
     grd1 :
   THEN
     act1
                power≔FALSE
           :
     act2 :
               modeG≔Off
     act3 :
               modeGP≔modeG
   startUpEndedGui ≜
   STATUS
     ordinary
   ANY
     modeg
```

```
WHERE
 \verb"grd1": \verb"modeG=StartUp"
 grd2 :
           modeg∈{Start}uVentilation
THEN
            modeG≔modeg
 act1
           modeGP≔modeG
 act2
END
crash
STATUS
 ordinary
WHEN
            crashed=FALSE
 grd1 :
THEN
 act1 :
           crashed≔TRUE
 act2 :
           modeG≔Off
 act3 : modeGP≔Off
END
repare
STATUS
 ordinary
ANY
 modeg
WHERE
 grd1
            {\tt crashed=TRUE}
            power=FALSE v (onAC=FALSE \( \) (switchover=FALSE v batLev=0 v batFail=TRUE))
 grd2
            modeg=Off
            power=TRUE ^ (onAC=TRUE v (switchover=TRUE ^ batLev>0 ^ batFail=FALSE))
 grd3 :
            \Rightarrow
            modeg=StartUp
THEN
 act1 :
           crashed≔FALSE
 act2 :
           modeG:=modeg
 act3 : modeGP≔modeG
END
newPatient
STATUS
 ordinary
WHEN
 grd1 :
            modeG=Start
THEN
 act1 : modeG≔SelfTest
 act2 : modeGP≔modeG
END
resumeVent
STATUS
 ordinary
WHEN
 grd1 :
            modeG=Start
THEN
 act1 :
            modeG≔Menu
 act2 :
           modeGP:=modeG
END
runAbortSelfTest ≜
STATUS
 ordinary
WHEN
 grd1 :
           modeG=SelfTest
THEN
           modeG≔SelfTest
 act1 :
 act2 : modeGP≔modeG
END
                           // CONT4-4.1
selfTestPassed ≜
STATUS
 ordinary
```

```
WHEN
             modeG=SelfTest
 grd1
THEN
 act1
              modeG≔Menu
             modeGP≔modeG
 act2
setParam
STATUS
 ordinary
WHEN
 grd1
              modeG∈{Menu}∪Ventilation
THEN
 act1
             modeG≔Settings
 act2
             modeGP:=modeG
END
saveBackAbort ≜
STATUS
 ordinary
ANY
 modeg
WHERE
              modeG=Settings \land modeg \in ModesG
 grd1
 grd2
              modeg∈Ventilation∪{Menu}
THEN
              modeG = modeg
 act1
             modeGP≔modeG
 act2
END
startStopPCVPSV
STATUS
 ordinary
ANY
 modeg
WHERE
 grd1
              modeG∈{Menu}∪Ventilation
 grd2
              modeg∈Ventilationu{Menu}
              modeG=Menu⇒modeg∈Ventilation
 grd3
 grd4
              modeG∈Ventilation⇒modeg=Menu
THEN
 act1
         :
              modeG≔modeg
 act2
              modeGP:=modeG
END
{\bf change Mode}
STATUS
 ordinary
ANY
 modeg
WHERE
 grd1
              modeG∈ Ventilation
 grd2
              modeg∈Ventilationu{Settings}
THEN
 act1
              modeG:=modeg
             modeGP≔modeG
 act2
END
failExternalPower
STATUS
 ordinary
ANY
 modeg
 modegp
WHERE
 grd1
              onAC=TRUE
              modeg \in ModesG \land modegp \in ModesG
 grd2
 grd3
              \neg(power=TRUE \land crashed=FALSE) \Rightarrow modeg=0ff \land modegp=0ff
 grd4
              power=TRUE \land crashed=FALSE \Rightarrow
                 ((switchover=TRUE \land batLev>0 \land batFail=FALSE \Rightarrow modeg=modeG \land modegp=modeGP) \land
```

```
(\neg(switchover=TRUE \ \land \ batLev>0 \ \land \ batFail=FALSE) \Rightarrow \ modeg=0ff \ \land \ modegp=modeG))
THEN
             onAC:=FALSE
 act1
             modeG:=modeg
 act2 :
 act3 : modeGP≔modegp
END
progress
STATUS
 ordinary
ANY
 step
 ι
 modeg
 batf
WHERE
 grd1
             step∈N1 ∧ l∈batteryRange ∧ batf∈B00L
 grd2
             batFail=TRUE ⇒ l=batLev
  grd3
             onAC=TRUE ∧ batFail=FALSE ⇔ l=batLev+step
 grd4
             onAC=FALSE \land batFail=FALSE \Leftrightarrow l=max({0,batLev-step})
              (onAC=TRUE v (l>0 \Lambda switchover=TRUE \Lambda batf=FALSE)) \Lambda
              power=TRUE \land modeG=Off \land
 grd5
              {\tt crashed=FALSE}
              modeg=StartUp
 grd6
             (l=0 v batf=TRUE v switchover=FALSE) ∧ onAC=FALSE ⇒modeg=Off
              onAC=TRUE v
              ((l>0 \land batLev>0) \land switchover=TRUE \land batf=FALSE)
 grd7
              modeg = modeG
THEN
  act1 : curTime≔curTime+step
 act2 :
             batLev≔l
 act3
         :
             modeG≔modeg
             modeGP = \{TRUE \mapsto modeGP, FALSE \mapsto modeG\} (bool(modeg = modeG))
 act4
 act5
             batFail≔batf
END
switchoverFail
STATUS
 ordinary
ANY
 modeg
WHERE
 grd1
         .
             switchover=TRUE
 grd2
             onAC=FALSE⇒modeg=Off
 grd3
             onAC=TRUE⇒modeg=modeG
THEN
 act1 :
             switchover≔FALSE
 act2 : modeG:=modeg
        : modeGP≔{TRUE→modeG,FALSE→modeGP}(bool(modeG≠modeg))
 act3
END
```

END

ContStates **EXTENDS** GuiStates **CONSTANTS** ${\tt ModesC}$ FailSafe VentilationOff ${\sf possTransC}$ AXIOMS $partition(Modes, \{StartUp\}, \{Start\}, \{Menu\}, \{SelfTest\}, \{FailSafe\}, \{Settings\}, \{PSV\}, \{PCV\}, \{0ff\}, \{Ventilation0ff\})\}$ axm1 axm2 : $ModesC \subseteq Modes$ axm3 partition(ModesC, {StartUp}, {SelfTest}, {FailSafe},{VentilationOff}, {Off},{PCV},{PSV}) axm4 partition(Ventilation, {PCV}, {PSV}) axm5 possTransC∈ B00L→P(ModesC×ModesC) // CONT2:Off⇒StartUp possTransC={ TRUE→ {Off⇔StartUp, StartUp⇔SelfTest, StartUp⇔FailSafe, SelfTest+FailSafe, SelfTest+VentilationOff,VentilationOff+FailSafe, PSV+PCV,PCV+PSV} axm6 : $\label{lem:condition} $$(\operatorname{Ventilation}(f)) \ (\operatorname{Ventilation}(f)^*\operatorname{Ventilation}(\operatorname{Ventilation}(f)^*\operatorname{Ven$ **END**

CONTEXT

```
MACHINE
   ContSM
REFINES
   GuiSM
SEES
   ContStates
VARIABLES
   power
   onAC
   batLev
   switchover
   crashed
   modeG
   modeGP
   batFail
   modeC
   modeCP
   curTime
INVARIANTS
              power=FALSE v (onAC=FALSE \( \) (switchover=FALSE v batLev=0 v batFail=TRUE))
   inv1 :
              modeC=Off
              modeCP \neq modeC \Rightarrow modeCP \mapsto modeC \in
   inv2
              possTransC(bool(power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))))
   inv3
              modeC∈ ModesC ∧ modeCP ∈ ModesC
              power=TRUE ∧ (onAC=TRUE v (switchover=TRUE ∧ batLev>0)) ∧ modeCP=Off⇒modeC=StartUp
   inv4
              modeG∈{Settings,Menu}∪Ventilation
   inv5
              modeC∈Ventilation υ{FailSafe,VentilationOff}
              modeG \in Ventilation \Rightarrow
   inv6
              modeC∈Ventilationu{FailSafe}
   inv7
         : modeC=StartUp⇒modeG≠Settings
EVENTS
   INITIALISATION ≜
     extended
   STATUS
    ordinary
   BEGIN
     act1 : power≔FALSE
    act2 :
act3 :
               batLev:∈batteryRange
               onAC≔TRUE
     act4 :
               switchover≔TRUE
     act5 : crashed≔FALSE
     act6 : modeG≔Off
     act7 : modeGP≔Off
     act8
               curTime:=0
           : batFail:∈B00L
     act9
     act10 : modeC≔Off
    act11 : modeCP≔Off
   END
   powerON ≜
                        // CONT.2
    extended
   STATUS
     ordinary
   REFINES
     powerON
   WHEN
     grd1 : power=FALSE
     grd2 : onAC=TRUE v (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE)
   THEN
     act1
               power:=TRUE
     act2
               modeG={FALSE→StartUp, TRUE→Off}(crashed)
     act3 :
               modeGP:=modeG
     act4 : modeC≔StartUp
     act5 : modeCP≔modeC
   END
   powerOff ≜
```

```
extended
STATUS
 ordinary
REFINES
 powerOff
WHEN
 grd1
            power=TRUE
THEN
 act1
            power:=FALSE
        1
            modeG≔Off
 act2
           modeGP:=modeG
 act3
 act4 : modeC≔Off
 act5 :
           modeCP≔modeC
END
startUpEndedGui ≜
 extended
STATUS
 ordinary
REFINES
 startUpEndedGui
ANY
 modeg
WHERE
        : modeG=StartUp
 grd1
 grd2
           modeg∈{Start}uVentilation
 grd3 :
           modeC∈Ventilation ⇒ modeg=modeC
           modeC∉Ventilation ⇒ modeg=Start
 grd4
THEN
 act1 :
           modeG≔modeg
 act2 : modeGP≔modeG
startUpEndedCont =
STATUS
 ordinary
WHEN
            modeC=StartUp
 grd1
THEN
 act1 :
           modeC≔SelfTest
 act2 : modeCP≔modeC
END
crash
 extended
STATUS
 ordinary
REFINES
 crash
WHEN
 grd1 :
           crashed=FALSE
THEN
           crashed=TRUE
 act1
        1
 act2
           modeG≔Off
 act3 : modeGP:=Off
END
repare
 extended
STATUS
 ordinary
REFINES
 repare
ANY
 modeg
WHERE
 grd1 :
           crashed=TRUE
            power=FALSE v (onAC=FALSE \( \) (switchover=FALSE v batLev=0 v batFail=TRUE))
 grd2 :
            modeg=Off
 grd3 :
            power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))
```

```
modeg=StartUp
 grd4 :
            modeC \in Ventilation \implies modeg = modeC
            power=TRUE \wedge (onAC=TRUE \vee (switchover=TRUE \wedge batLev>0 \wedge batFail=FALSE))\wedge
            modeC∉Ventilation
 grd5 :
            modeg=StartUp
THEN
 act1 : crashed=FALSE
 act2 : modeG=modeg
 act3 : modeGP≔modeG
END
newPatient ≜
 extended
STATUS
 ordinary
REFINES
 newPatient
WHEN
 grd1 : modeG=Start
 grd2 : modeC=SelfTest
THEN
 act1 : modeG≔SelfTest
act2 : modeGP≔modeG
END
resumeVent ≜
 extended
STATUS
 ordinary
REFINES
 resumeVent
WHEN
 grd1 : modeG=Start
 grd2 : modeC=SelfTest
THEN
 act1 : modeG:=Menu
 act2 : modeGP≔modeG
 act3 : modeC≔VentilationOff
act4 : modeCP≔modeC
END
runAbortSelfTest ≜
 extended
STATUS
 ordinary
REFINES
 runAbortSelfTest
WHEN
 grd1
           modeG=SelfTest
       1
 grd2 : modeC=SelfTest
THEN
 act1 : modeG=SelfTest
 act2 : modeGP≔modeG
END
selfTestPassed ≜
                            // CONT4-4.1
 extended
STATUS
 ordinary
REFINES
 selfTestPassed
 grd1 : modeG=SelfTest
 \verb"grd2": \verb"modeC=SelfTest"
THEN
 act1 : modeG≔Menu
 act2 : modeGP≔modeG
 act3 : modeC≔VentilationOff
 act4 : modeCP≔VentilationOff
```

END

```
setParam
 extended
STATUS
 ordinary
REFINES
  setParam
WHEN
             modeG∈{Menu}uVentilation
  grd1 :
THEN
  act1 : modeG≔Settings
 act2 : modeGP≔modeG
saveBackAbort ≜
 extended
STATUS
 ordinary
REFINES
 saveBackAbort
ANY
 modeg
 modec
WHERE
             modeG=Settings ∧ modeg∈ModesG
 grd1
            modeg∈Ventilationu{Menu}
 grd2
 grd3 :
            modeC≠FailSafe
 grd4 :
             modec∈ModesC
 \verb"grd5": modeC=Ventilation" \Rightarrow \verb"modec=Ventilation" \land \verb"modeg=modec"
 grd6
        : modeC∉Ventilation ⇒modec=modeC ∧ modeg=Menu
THEN
 act1 : modeG≔modea
 act2 : modeGP≔modeG
 act3 : modeC≔modec
 act4 : modeCP≔modeC
END
startStopPCVPSV
  extended
STATUS
 ordinary
REFINES
 startStopPCVPSV
ANY
 modeg
 modec
WHERE
 grd1
            modeG∈{Menu}uVentilation
 grd2 :
            modeg∈Ventilationu{Menu}
  grd3 : modeG=Menu⇒modeg∈Ventilation
  grd4 :
            modeG∈Ventilation⇒modeg=Menu
 grd5
             modeC≠FailSafe
 grd6
             modeg \in Ventilation \implies modec = modeg
 grd7
             {\tt modeg \not\in Ventilation} \implies {\tt modec = Ventilation 0ff}
THEN
  act1 : modeG≔modeg
  act2 : modeGP≔modeG
        :
 act3
             modeC≔modec
  act4
             modeCP = \{TRUE \mapsto modeC, FALSE \mapsto modeCP\} (bool(modeC \in Ventilation \ \cup \{VentilationOff\})) \}
END
changeMode
             4
 extended
STATUS
 ordinary
REFINES
  changeMode
ANY
 modeg
 modec
WHERE
```

```
grd1 : modeG∈ Ventilation
  grd2
             modeg∈Ventilationu{Settings}
  grd3
             modeC∈Ventilation\{FailSafe}
  grd4
             modec∈Ventilation
THEN
             modeG:=modeg
  act1
  act2
             modeGP≔modeG
             modeC:=modec
 act3
         :
             modeCP:=modeC
 act4
END
failExternalPower ≜
 extended
STATUS
 ordinary
REFINES
  failExternalPower
ANY
 modeg
  modegp
 modec
 modecp
WHERE
             onAC=TRUE
  grd1
  grd2
             modeg ∈ ModesG ∧ modegp ∈ ModesG
         1
  grd3
             \neg (power=TRUE \land crashed=FALSE) \Rightarrow modeg=Off \land modegp=Off
             power=TRUE ∧ crashed=FALSE ⇒
  grd4
                 ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE ⇒ modeg=modeG ∧ modegp=modeGP) ∧
                 (¬(switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)⇒ modeg=Off ∧ modegp=modeG))
  grd5
             modec ∈ ModesC ∧ modecp ∈ ModesC
             power= FALSE ⇒ modec=modeC ∧ modecp=modeCP
  grd6
              power= TRUE ⇒
  grd7
                  ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE ⇒modec=modeC ∧ modecp=modeCP) ∧
                   (\neg(switchover=TRUE \ \land \ batLev>0 \ \land \ batFail=FALSE) \Longrightarrow modec=Off \ \land \ modecp=modeC))
THEN
             onAC:=FALSE
  act1
         :
  act2
             modeG:=modeg
             modeGP:=modegp
  act3
 act4
             modeC≔modec
  act5
             modeCP:=modecp
END
failSelfTest ≜
STATUS
 ordinary
WHEN
             modeC=SelfTest
 grd1
THEN
 act1
             {\tt modeC}{\coloneqq}{\sf FailSafe}
             modeCP≔modeC
 act2
END
progress
 extended
STATUS
 ordinary
REFINES
 progress
ANY
 step
  Z
  modeg
  batf
 modec
WHERE
             step∈N1 ∧ l∈batteryRange ∧ batf∈B00L
  grd1
             batFail=TRUE ⇒ l=batLev
  grd2
  grd3 : onAC=TRUE \land batFail=FALSE \Leftrightarrow l=batLev+step
```

```
: onAC=FALSE ∧ batFail=FALSE ⇔ l=max({0,batLev-step})
 grd4
            (onAC=TRUE v (l>0 \( \) switchover=TRUE \( \) batf=FALSE)) \( \)
            power=TRUE A modeG=Off A
 grd5
            crashed=FALSE
            modeg=StartUp
           (l=0 v batf=TRUE v switchover=FALSE) ∧ onAC=FALSE ⇒modeg=Off
 grd6
            onAC=TRUE v
            ((l>0 Λ batLev>0) Λ switchover=TRUE Λ batf=FALSE)
 grd7
            modeg=modeG
 grd8
            modec∈{FailSafe,modeC, Off,StartUp}
 grd9
            modec=FailSafe⇒modeC≠Off
             (onAC=TRUE v (l>0 ∧ switchover=TRUE ∧ batf=FALSE)) ∧
             power=TRUE ^ modeC=Off
 grd10
             modec=StartUp
             (batLev>0 \land l>0)v switchover=FALSE v onAC=TRUE v
             power=FALSE
 grd11
             modec∈{modeC,FailSafe}
             (l=0 v batf=TRUE v switchover=FALSE) ∧ onAC=FALSE
 grd12
             modec=Off
THEN
 act1 : curTime=curTime+step
 act2 : batLev≔l
        : modeG≔modeq
 act3
           modeGP:={TRUE→modeGP, FALSE→modeG}(bool(modeg=modeG))
 act4
 act5
            batFail≔batf
 act6
        : modeC≔modec
 act7 : modeCP≔{TRUE→modeCP,FALSE→modeC}(bool(modeC=modec))
END
switchoverFail ≜
 extended
STATUS
 ordinary
REFINES
 switchoverFail
ANY
 modeg
 modec
WHERE
        : switchover=TRUE
 grd1
 grd2
            onAC=FALSE⇒modeg=Off
 grd3
            onAC=TRUE⇒modeg=modeG
           onAC=FALSE⇒modec=Off
 grd4 :
 grd5 :
            onAC=TRUE⇒modec=modeC
THEN
 act1
           switchover≔FALSE
            modeG≔modeg
 act2
 act3
            modeGP:={TRUE→modeG, FALSE→modeGP}(bool(modeG≠modeg))
            modeC≔modec
 act4
            modeCP:={TRUE→modeC,FALSE→modeCP}(bool(modeC≠modec))
 act5
END
```

END

ComParams

SETS

parameters Cycles

CONSTANTS

comParams
PEEP
Fi02
PRM
timerRM
domcomParams
defcomParams
inspPauseMax
expPauseMax
resumeTime

axm15 : finite(Cycles)

AXIOMS

END

```
axm1 : comParams ⊆ parameters
axm2 : partition(comParams, {PEEP}, {Fi02}, {PRM}, {timerRM})
\verb|axm3| : domcomParams={PEEP+5..20,Fi02+21..100, PRM+1..50,timerRM+1..30}|
axm4
            defcomParams={PEEP→5,FiO2→21, PRM→20,timerRM→10}
axm5
             domcomParams \in comParams \rightarrow P1(N1)
axm6
            defcomParams∈ comParams → N1
axm7 : ∀p·p∈ comParams⇒defcomParams(p)∈domcomParams(p)
             ∀y∙ y∈comParams
axm8
       : ⇒
             (Ux \cdot x \in comParams \mid \{x \mapsto defcomParams(x)\})(y) \in domcomParams(y)
       : inspPauseMax=40*10
axm9
axm10 : expPauseMax=60*10
axm11 : resumeTime∈N1
              ∀ a,A, f,x·
axm12 ; as N \land As parameters \rightarrow N \land xe A \land fe 0..a \rightarrow A
              f∢{a↦x}∈ 0..a→A
axm13 : \forall a,b,c,d \cdot a \in \mathbb{N} \land b \in \mathbb{N} 1 \land c \in \mathbb{N} \land d \in \mathbb{N} 1 \land a \geq c \land b \leq d \Rightarrow a \div b \geq c \div d
axm14 : ∀a· a∈N \Rightarrow a÷2∈N
```

PCVParams

EXTENDS

ComParams

CONSTANTS

pcvParams dompcvParams defpcvParams RRPCV IEPCV PinspPCV ITSPCV

AXIOMS

```
\texttt{axm1} \quad : \quad \mathsf{pcvParams} \underline{\texttt{c}} \mathsf{parameters}
axm2
               partition(pcvParams,{RRPCV}, {IEPCV}, {PinspPCV}, {ITSPCV})
axm3
               dompcvParams \in pcvParams \rightarrow P1(N1)
         : dompcvParams={RRPCV→4..50,IEPCV→1..4,PinspPCV→2..50,ITSPCV→1..9}
axm4
axm5 : defpcvParams∈ pcvParams \rightarrow N1
axm6 : \forall p \cdot p \in pcvParams \Rightarrow defpcvParams(p) \in dompcvParams(p)
\texttt{axm7} \quad : \quad \mathsf{defpcvParams=} \{\mathsf{RRPCV} {\leftrightarrow} 12\,, \mathsf{IEPCV} {\leftrightarrow} 2\,, \mathsf{PinspPCV} {\leftrightarrow} 15\,, \mathsf{ITSPCV} {\leftrightarrow} 3\}
axm12 : \forall a,b \cdot a \in N \land b \in N \land a>0 \land b>0 ⇒a÷b≥0
axm8 : pcvParamsncomParams=ø
axm9 : card(pcvParams)=4
                 ∀y∙ y∈pcvParams
axm10 :
                                            (Ux \cdot x \in pcvParams \mid \{x \mapsto defpcvParams(x)\})(y) \in dompcvParams(y)
axm11 : \forall a,b \cdot a \in N \land b \in N \land a>0 \land b>0 \Rightarrow a*b>0
```

END

PSVParams

EXTENDS

PCVParams

CONSTANTS

psvParams PinspPSV **ITSPSV** ETS ApneaLag RRAP IEAP PinspAP dompsvParams defpsvParams ${\tt max_insp_time_psv}$ VERange

AXIOMS

```
\texttt{axm1} \quad : \quad \mathsf{psvParams} \underline{\texttt{c}} \mathsf{parameters}
\verb|axm2| : partition(psvParams, \{PinspPSV\}, \{ITSPSV\}, \{ETS\}, \{ApneaLag\}, \{RRAP\}, \{PinspAP\})|
axm3
               partition(parameters, comParams,pcvParams,psvParams)
axm4 :
               \texttt{dompsvParams} \in \, \texttt{psvParams} \, \to \, \mathbb{P}1(\mathbb{N}1)
axm5 :
               dompsvParams = \{PinspPSV \mapsto 2..50, ITSPSV \mapsto 1..9, ETS \mapsto 5..60, ApneaLag \mapsto 100..600, RRAP \mapsto 4..50, PinspAP \mapsto 2..50\}
axm6 : IEAP∈ N ∧ IEAP=2
axm7 : defpsvParams\in psvParams \setminus{RRAP,PinspAP}\rightarrow N1
         : \forall p \cdot p \in dom(defpsvParams) \Rightarrow defpsvParams(p) \in dompsvParams(p)
axm8
         : {PinspPSV→15, ITSPSV→3,ETS→30,ApneaLag→300}⊆defpsvParams
: max_insp_time_psv∈ N ∧ max_insp_time_psv=70
axm9
axm10
axm11 : VERange⊆N
```

END

```
MACHINE
   Ventilation
REFINES
   ContSM
SEES
   ContStates
   PSVParams
VARIABLES
   power
   onAC
   batLev
   switchover
   crashed
   modeG
   modeGP
   modeC
   modeCP
   batFail
   PCV2PSV
   comParamsValC
   comParamsValG
   pcvParamsValC
   pcvParamsValG
   psvParamsValC
   psvParamsValG
   curTime
   offT
INVARIANTS
   inv2 :
              comParamsValC∈ 0..curTime→(comParams→N1)
   inv3
               comParamsValG∈ 0..curTime +>(comParams→N1)
   inv4
               pcvParamsValC∈ 0..curTime→(pcvParams→N1)
   inv5
               pcvParamsValG∈ 0..curTime →(pcvParams→N1)
               psvParamsValC∈ 0..curTime→(psvParams→N1) ∧
   inv6
               (\forall x \cdot x \in ran(psvParamsValC) \Rightarrow psvParams \setminus \{RRAP, PinspAP\} \subseteq dom(x))
               psvParamsValG∈ 0..curTime →(psvParams→N1) ∧
   inv7
               (\forall x \cdot x \in ran(psvParamsValG) \Rightarrow psvParams \setminus \{RRAP, PinspAP\} \subseteq dom(x))
               modeG∉{Off,StartUp}
   inv8
               pcvParamsValG≠ø ∧ psvParamsValG≠ø ∧ comParamsValG≠ø
               modeC∉{Off,StartUp, FailSafe}
   inv9
               pcvParamsValC≠ø ∧ psvParamsValC≠ø ∧ comParamsValC≠ø
   inv10
                offT∈ N ∧ offT ≤curTime
                offT>0
   inv11
                comParamsValC≠ø ∧ pcvParamsValC≠ø ∧ psvParamsValC≠ø
                (comParamsValC≠ø ⇔ pcvParamsValC≠ø) ∧
   inv12
                (pcvParamsValC≠ø⇔psvParamsValC≠ø)
                dom(comParamsValC) = dom(pcvParamsValC) ^
   inv13
                dom(pcvParamsValC)=dom(psvParamsValC)
   inv14
                PCV2PSV∈ BOOL ∧ PCV2PSV=TRUE ⇒ modeG=Settings
    inv16
                PCV2PSV=TRUE \Rightarrow modeC \in \{PCV, FailSafe\}
                ∀t· t∈ dom(pcvParamsValC)
    inv17
                ((pcvParamsValC(t))(RRPCV) *(1 +(pcvParamsValC(t))(IEPCV)))>0
                modeC=PSV
   inv18
                dom(psvParamsValC(max(dom(psvParamsValC))))=psvParams
                (psvParamsValC(max(dom(psvParamsValC))))(ApneaLag)≥max_insp_time_psv÷2
                modeC=PCV⇒
                10*60*(pcvParamsValC(max(dom(pcvParamsValC)))))(IEPCV)÷
   inv19
                   ((pcvParamsValC(max(dom(pcvParamsValC))))(RRPCV)*
                (1 +(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV))) >0
EVENTS
   INITIALISATION
     extended
   STATUS
     ordinary
```

```
BEGIN
 act1 : power≔FALSE
 act2
       :
:
           batLev:∈batteryRange
 act3
           onAC:=TRUE
           switchover≔TRUE
 act4
 act5 :
          crashed:=FALSE
 act6 : modeG≔Off
       : modeGP≔Off
 act7
       : curTime≔0
: batFail:∈B00L
 act8
 act9
 act10 : modeC≔Off
 act11 : modeCP≔Off
 act12 : comParamsValC≔ø
 act13 : comParamsValG≔ø
 act14
        :
            pcvParamsValC≔ø
 act15
            pcvParamsValG≔ø
 act16 : psvParamsValG≔ø
 act17 : psvParamsValC≔ø
 act18 : offT≔0
 act19 : PCV2PSV≔FALSE
powerON ≜
                    // CONT.2
 extended
STATUS
 ordinary
REFINES
 power0N
WHEN
 grd1 : power=FALSE
 grd2 : onAC=TRUE ν (switchover=TRUE Λ batLev>0 Λ batFail=FALSE)
THEN
 act1 : power≔TRUE
 act2 : modeG≔{FALSE→StartUp, TRUE→Off}(crashed)
 act3 : modeGP≔modeG
 act4 : modeC≔StartUp
 act5 : modeCP≔modeC
END
powerOff ≜
 extended
STATUS
 ordinary
REFINES
 powerOff
ANY
 val
WHERE
           power=TRUE
 grd1
           comParamsValC=@ v comParamsValC(max(dom(comParamsValC)))=defcomParams
 grd2
           val=0
           comParamsValC≠ø ∧
           comParamsValC(max(dom(comParamsValC)))≠defcomParams
 grd3
           \Rightarrow
           val=curTime
THEN
 act1 : power≔FALSE
 act2
       .
           modeG=Off
 act3
           modeGP≔modeG
 act4
           modeC:=Off
 act5
           modeCP:=modeC
 act6 :
           offT≔val
 act7 : PCV2PSV≔FALSE
startUpEndedGui ≜
 extended
STATUS
 ordinary
REFINES
```

```
startUpEndedGui
ANY
  modeg
  pcvG
  comG
 psvG
WHERE
  grd1
              modeG=StartUp
              modeg∈{Start}uVentilation
  grd2
  grd3
              modeC \in Ventilation \Rightarrow modeg = modeC
  grd4
              modeC \not\in Ventilation \implies modeg = Start
              modeC \in Ventilation \ v \ (offT \neq 0 \ \land \ curTime-offT \le resumeTime)
                 comG=comParamsValG ❖
                                   {curTime →comParamsValC(max(dom(comParamsValC)))} ∧
  grd5
                 pcvG=pcvParamsValG ∢
                                   \{curTime \Rightarrow pcvParamsValC(max(dom(pcvParamsValC)))\} \land
                 psvG=psvParamsValG ∢
                                  {curTime →psvParamsValC(max(dom(psvParamsValC)))}
              modeC∉Ventilation ∧ (offT=0 v curTime-offT > resumeTime)
                 comG=comParamsValG ∢
                                   {curTime \mapstodefcomParams} \land
  grd6
                 pcvG=pcvParamsValG ∢
                                  {curTime →defpcvParams} ∧
                 psvG=psvParamsValG ❖
                                  {curTime →defpsvParams}
THEN
  act1
              modeG:=modeg
  act2
              modeGP:=modeG
  act3
              comParamsValG≔comG
  act4
              pcvParamsValG≔pcvG
 act5
              psvParamsValG≔psvG
END
startUpEndedCont
 extended
STATUS
 ordinary
REFINES
 startUpEndedCont
ANY
 comC
 pcvC
 psvC
WHERE
  grd1
              modeC=StartUp
              offT=0 v curTime-offT > resumeTime
              comC=comParamsValC ❖
                                   {curTime →defcomParams} ∧
  grd2
              pcvC=pcvParamsValC ❖
                                   \{curTime \mapsto defpcvParams\} \land
              psvC=psvParamsValC ∢
                                  {curTime →defpsvParams}
              offT≠0 \land curTime-offT \le resumeTime
              comC=comParamsValC ❖
                                   {curTime →comParamsValC(max(dom(comParamsValC)))} ∧
  grd3
              pcvC=pcvParamsValC
                                   {curTime →pcvParamsValC(max(dom(pcvParamsValC)))} ∧
                                  {curTime →psvParamsValC(max(dom(psvParamsValC)))}
THEN
  act1
              modeC≔SelfTest
              modeCP:=modeC
  act2
              comParamsValC≔comC
  act3
  act4
              pcvParamsValC≔pcvC
 act5
              psvParamsValC≔psvC
END
crash
 extended
```

```
STATUS
 ordinary
REFINES
 crash
WHEN
 grd1 : crashed=FALSE
THEN
 act1 : crashed≔TRUE
 act2 : modeG≔Off
act3 : modeGP≔Off
 act4 : PCV2PSV≔FALSE
END
repare ≜
 extended
STATUS
 ordinary
REFINES
 repare
ANY
 modeg
WHERE
 grd1 :
            crashed=TRUE
            power=FALSE v (onAC=FALSE \( \) (switchover=FALSE v batLev=0 v batFail=TRUE))
 grd2
            modeg=Off
            power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))
 grd3 :
            modeg=StartUp
 grd4
            modeC∈Ventilation ⇒ modeg=modeC
            power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))\( \)
            modeC∉Ventilation
 grd5 :
            \Rightarrow
            modeg=StartUp
THEN
 act1 : crashed=FALSE
 act2
            modeG≔modeg
 act3 :
            modeGP:=modeG
END
newPatient
 extended
STATUS
 ordinary
REFINES
 newPatient
 grd1 : modeG=Start
 grd2 : modeC=SelfTest
THEN
 act1 : modeG=SelfTest
 act2 : modeGP≔modeG
resumeVent ≜
 extended
STATUS
 ordinary
REFINES
 resumeVent
 grd1 : modeG=Start
 grd2 : modeC=SelfTest
grd3 : offT≠0 ∧ curTime-offT ≤ resumeTime
THEN
 act1 : modeG≔Menu
 act2 : modeGP≔modeG
 act3 : modeC≔VentilationOff
 act4 : modeCP:=modeC
END
```

```
runAbortSelfTest ≜
 extended
STATUS
 ordinary
REFINES
 {\tt runAbortSelfTest}
WHEN
       : modeG=SelfTest
 grd1
 grd2
       : modeC=SelfTest
THEN
 act1 : modeG=SelfTest
 act2 : modeGP≔modeG
END
                          // CONT4-4.1
selfTestPassed ≜
extended
STATUS
 ordinary
REFINES
 selfTestPassed
WHEN
 grd1 : modeG=SelfTest
 grd2 : modeC=SelfTest
THEN
 act1 :
           modeG≔Menu
 act2 :
          modeGP≔modeG
 act3 : modeC≔VentilationOff
 act4 : modeCP≔VentilationOff
END
setParam ≜
 extended
STATUS
 ordinary
REFINES
 setParam
WHEN
 grd1
           modeG∈{Menu}uVentilation
THEN
 act1 : modeG≔Settings
 act2 : modeGP≔modeG
END
settingParams ≜
STATUS
 ordinary
ANY
 psvP
 pcvP
 comP
WHERE
 grd1
           modeG=Settings
 grd2
           max(dom(psvParamsValG))<curTime</pre>
 grd3 :
           psvP∈ psvParams→N1
           psvParams\{RRAP,PinspAP}⊆dom(psvP)
 grd4 :
 grd5 : pcvP∈ pcvParams→N1
 grd6 : comP∈ comParams→N1
THEN
 act1
           psvParamsValG(curTime)≔psvP
           pcvParamsValG(curTime)≔pcvP
 act2
 act3
      : comParamsValG(curTime)≔comP
END
saveBackAbort ≜
 extended
STATUS
 ordinary
REFINES
 saveBackAbort
ANY
 modeg
```

```
modec
 sv
 pcvC
 psvC
 comC
 pcvG
 psvG
 comG
WHERE
 grd1
            modeG=Settings ∧ modeg∈ModesG
            modeg∈Ventilationu{Menu}
 ard2
            modeC≠FailSafe
 grd3
 grd4
            modec∈ModesC
            modeC∈Ventilation ⇒ modec∈Ventilation ∧ modeq=modec
 ard5
 grd6
            modeC∉Ventilation ⇒modec=modeC ∧ modeg=Menu
 grd7
             sv∈{TRUE, FALSE}
             psvG={TRUE→
 grd8
            psvParamsValG(max(dom(psvParamsValG))),
              FALSE⇔psvParamsValC(max(dom(psvParamsValC)))}(sv)
 grd9
            psvParamsValG(max(dom(psvParamsValG))),
              FALSE⇔psvParamsValC(max(dom(psvParamsValC)))}(sv)
 grd10
              pcvParamsValG(max(dom(pcvParamsValG))),
               FALSE→pcvParamsValC(max(dom(pcvParamsValC)))}(sv)
              pcvC={TRUE→
 grd11
              pcvParamsValG(max(dom(pcvParamsValG))),
              FALSE→pcvParamsValC(max(dom(pcvParamsValC)))}(sv)
              comG={TRUE→
 grd12
              comParamsValG(max(dom(comParamsValG))),
               FALSE→comParamsValC(max(dom(comParamsValC)))}(sv)
              comC={TRUE→
 grd13
              comParamsValG(max(dom(comParamsValG))),
               FALSE→comParamsValC(max(dom(comParamsValC)))}(sv)
              PCV2PSV=TRUE ∧ modeC=PCV⇒modec=PSV
 ard14
              PCV2PSV=FALSE v modeC≠PCV⇒modec=modeC
 grd15
              modec=PSV
 grd16
              dom(psvC)=psvParams ^
              psvC(ApneaLag) \ge max_insp_time_psv \div 2
 grd17
              modeC≠modec ⇒PCV2PSV=TRUE ∧ modeC=PCV ∧ modec=PSV
              modec=PCV∧ sv=TRUE⇒
 grd18
              10*60*pcvC(IEPCV)÷ (pcvC(RRPCV)*(1 +pcvC(IEPCV))) > 0
 grd19
              curTime∉dom(pcvParamsValC)
THEN
 act1
            modeG:=modeg
 act2
            modeGP:=modeG
 act3
            modeC≔modec
 act4
            modeCP:=modeC
            psvParamsValG(curTime)≔psvG
 act5
 act6
            psvParamsValC(curTime)≔psvC
 act7
            pcvParamsValG(curTime)≔pcvG
            pcvParamsValC(curTime)≔pcvC
 act8
 act9
            comParamsValG(curTime)≔comG
 act10
             comParamsValC(curTime)≔comC
             PCV2PSV≔FALSE
 act11
        . .
END
startStopPCVPSV
 extended
STATUS
 ordinary
REFINES
 startStopPCVPSV
ANY
 modeg
 modec
WHERE
 grd1
             modeG∈{Menu}uVentilation
 grd2
            modeg∈Ventilationu{Menu}
            modeG=Menu⇒modeg∈Ventilation
 grd3
```

```
grd4
            modeG∈Ventilation⇒modeg=Menu
  grd5
            modeC≠FailSafe
        1
  grd6
             modeg \in Ventilation \implies modec = modeg
  grd7
             modeg \not\in Ventilation \Rightarrow modec = VentilationOff
             modeg=PSV
  grd8
             dom(psvParamsValC(max(dom(psvParamsValC))))=psvParams
  grd9
             (psvParamsValC(max(dom(psvParamsValC))))(ApneaLag)≥max_insp_time_psv÷2
              10*60*(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)÷
  grd10
                ((pcvParamsValC(max(dom(pcvParamsValC))))(RRPCV)*
              (1 +(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV))) >0
THEN
             modeG:=modeg
  act1
             modeGP:=modeG
  act2
  act3
             modeC≔modec
            modeCP={TRUE→modeC, FALSE→modeCP}(bool(modeC∈Ventilation u{VentilationOff}))
  act4
END
moveToPSV
 extended
STATUS
 ordinary
REFINES
 changeMode
ANY
  modeg
  modec
WHERE
  grd1
            modeG∈ Ventilation
            modeg∈Ventilationu{Settings}
  grd2
            modeC∈Ventilation\{FailSafe}
  grd3
  grd4
            modec∈Ventilation
  grd5
             modeG=PCV
        :
  grd6
             modeC=PCV
         :
  grd7
             modeg=Settings
             modec=modeC
 grd8
THEN
  act1
             modeG:=modeg
             modeGP:=modeG
  act2
        .
  act3
             modeC≔modec
        1
  act4
             modeCP:=modeC
             PCV2PSV≔TRUE
 act5
END
changeMode
 extended
STATUS
  ordinary
REFINES
  changeMode
ANY
  modeg
  modec
 pcv
WHERE
  grd1
             modeG∈ Ventilation
  grd2
             modeg∈Ventilationu{Settings}
             modeC∈Ventilation\{FailSafe}
  grd3
  grd4
             modec∈Ventilation
  grd5
             pcv ∈ pcvParams→N1
  grd6
             modeC=PCV \implies modec=PCV
             modeC=PSV \land modec=PCV
             pcv=pcvParamsValC(max(dom(pcvParamsValC)))
  grd7
             {RRPCV→(psvParamsValC(max(dom(psvParamsValC))))(RRAP),
              PinspPCV→(psvParamsValC(max(dom(psvParamsValC))))(PinspAP),
             IEPCV→IEAP}
  grd8
             modeC=PCV v modec=PSV
```

```
pcv=pcvParamsValC(max(dom(pcvParamsValC)))
             modec=PCV
 grd9
             10*60*pcv(IEPCV)÷ (pcv(RRPCV)*(1 +pcv(IEPCV))) >0
THEN
             modeG:=modea
  act1
             modeGP:=modeG
  act2
  act3
             modeC:=modec
             modeCP:=modeC
  act4
 act5
             pcvParamsValC(curTime)≔pcv
             psvParamsValC(curTime):=psvParamsValC(max(dom(psvParamsValC)))
  act6
             comParamsValC(curTime) = comParamsValC(max(dom(comParamsValC)))
 act7
extended
STATUS
 ordinary
REFINES
 failExternalPower
ANY
  modeg
  modeap
  modec
  modecp
 pcv2psv
WHERE
  grd1
             onAC=TRUE
  grd2
             modeg ∈ ModesG ∧ modegp ∈ ModesG
             ¬(power=TRUE ∧ crashed=FALSE) ⇒ modeg=Off ∧ modegp=Off
             power=TRUE ∧ crashed=FALSE ⇒
  grd4
                 ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE ⇒ modeg=modeG ∧ modegp=modeGP) ∧
                 (\neg (switchover = TRUE \land batLev > 0 \land batFail = FALSE) \Rightarrow modeg = 0 ff \land modegp = modeG))
             modec ∈ ModesC ∧ modecp ∈ ModesC
  grd5
             power= FALSE ⇒ modec=modeC ∧ modecp=modeCP
  grd6
              power= TRUE ⇒
  grd7
                  ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE ⇒modec=modeC ∧ modecp=modeCP) ∧
                   (\neg (switchover = TRUE \land batLev > 0 \land batFail = FALSE) \Rightarrow modec = 0 ff \land modecp = modeC))
             \neg(power=TRUE \land crashed=FALSE) \Rightarrow pcv2psv=FALSE
  grd8
             power=TRUE \land crashed=FALSE \Rightarrow
  grd9
              ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE⇒ pcv2psv=PCV2PSV) ∧
              (¬(switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)⇒ pcv2psv=FALSE))
THEN
  act1
             onAC:=FALSE
  act2
             modeG≔modea
  act3
             modeGP:=modegp
  act4
             modeC≔modec
  act5
             modeCP:=modecp
         : PCV2PSV≔pcv2psv
  act6
END
failSelfTest ≜
 extended
STATUS
 ordinary
REFINES
  failSelfTest
WHEN
             modeC=SelfTest
  grd1
THEN
             modeC≔FailSafe
  act1
  act2
            modeCP:=modeC
END
progress ≜
```

```
extended
STATUS
 ordinary
REFINES
 progress
ANY
  step
  Z
  modeg
  batf
  modec
  paw
WHERE
            step∈N1 ∧ l∈batteryRange ∧ batf∈B00L
  ard1
  grd2
             batFail=TRUE ⇒ l=batLev
             onAC=TRUE ∧ batFail=FALSE ⇔ l=batLev+step
  grd3
            onAC=FALSE ∧ batFail=FALSE ⇔ l=max({0,batLev-step})
  grd4
             (onAC=TRUE v (l>0 \( \) switchover=TRUE \( \) batf=FALSE)) \( \)
             power=TRUE A modeG=Off A
  grd5
             crashed=FALSE
             modeg=StartUp
  grd6
            (l=0 v batf=TRUE v switchover=FALSE) ∧ onAC=FALSE ⇒modeg=Off
             ((l>0 Λ batLev>0) Λ switchover=TRUE Λ batf=FALSE)
  grd7
             modeg=modeG
  grd8
             modec∈{FailSafe,modeC, Off,StartUp}
  grd9
             modec=FailSafe⇒modeC≠Off
              (onAC=TRUE v (l>0 \land switchover=TRUE \land batf=FALSE)) \land
              power=TRUE A modeC=Off
  grd10
              modec=StartUp
              (batLev>0 \land l>0)\lor switchover=FALSE \lor onAC=TRUE \lor
              power=FALSE
  grd11
              modec∈{modeC,FailSafe}
              (l=0 v batf=TRUE v switchover=FALSE) \land onAC=FALSE
  grd12
              modec=Off
 grd13
             step∈N1 ∧ paw∈N
THEN
  act1
        : curTime≔curTime+step
  act2
            batLev≔l
  act3
        : modeG≔modeg
            modeGP:={TRUE→modeGP, FALSE→modeG}(bool(modeg=modeG))
  act4
  act5
            batFail≔batf
  act6
            modeC≔modec
            modeCP={TRUE→modeCP, FALSE→modeC} (bool (modeC=modec))
  act7
            PCV2PSV≔{TRUE→PCV2PSV, FALSE→FALSE}(bool(
 act8 :
             (batLev>0 \( 1>0\) switchover=FALSE) v onAC=TRUE))
END
switchoverFail ≜
 extended
STATUS
 ordinary
REFINES
 switchoverFail
ANY
 modeg
  modec
WHERE
            switchover=TRUE
  ard1
  grd2
            onAC=FALSE⇒modeg=Off
            onAC=TRUE⇒modeg=modeG
  grd3
 grd4
            onAC=FALSE⇒modec=Off
  grd5
            onAC=TRUE⇒modec=modeC
THEN
            switchover≔FALSE
  act1
            modeG:=modeg
  act2
```

```
act3 : modeGP={TRUE→modeG,FALSE→modeGP}(bool (modeG≠modeg))
act4 : modeC=modec
act5 : modeCP={TRUE→modeC,FALSE→modeCP}(bool(modeC≠modec))
act6 : PCV2PSV={TRUE→PCV2PSV, FALSE→FALSE}(bool(onAC=TRUE))
END
```

END

```
CONTEXT
                               VentilStates
EXTENDS
                               {\tt GuiStates}
SETS
                               {\tt ventSates}
CONSTANTS
                               inspBeg
                               inspEnd
                               expBeg
                               expEnd
                               inspPauseBeg
                               \verb"inspPauseEnd"
                               expPauseBeg
                               {\sf expPauseEnd}
                               rmBeg
                               rmEnd
 AXIOMS
                                                                                                                              \verb|partition(ventSates, \{inspBeg\}, \{inspEnd\}, \{expBeg\}, \{expEnd\}, \{inspPauseBeg\}, \{expEnd\}, \{ex
                               axm1
                                                                                                                                                   {inspPauseEnd}, {expPauseBeg}, {expPauseEnd}, {rmBeg}, {rmEnd})
 END
```

```
MACHINE
     VentilationPhases
REFINES
     Ventilation
SEES
     ContStates
     VentilStates
     PSVParams
VARIABLES
     power
     onAC
     batLev
     switchover
     crashed
     modeG
     modeGP
     modeC
     modeCP
     PCV2PSV
     comParamsValC
     comParamsValG
     pcvParamsValC
     pcvParamsValG
     psvParamsValC
     psvParamsValG
     batFail
     curTime
     offT
     cycles
     cycleMode
     ventilPhase
     inspEndT
     inspBegT
     inspPauseBegT
     inspPauseEndT
     PAW
     ٧E
     rmBegT
     rmEndT
     expBegT
     expEndT
     expPauseBegT
     expPauseEndT
INVARIANTS
     inv1 :
                    cycles⊆Cycles ∧ cycleMode∈cycles→Ventilation
     inv2 : cycles≠ø⇒pcvParamsValC≠ø
                    inspBegT ∈ cycles →N
     inv3
               :
     inv4
                     inspEndT ∈ cycles →N
                     ∀ c· c∈cycles
     inv5
                     (∃ x·(x∈dom(pcvParamsValC) ∧ x∈dom(comParamsValC) ∧ x≤inspBegT(c)))
                      ∀c· c∈ cycles ∧ cycleMode(c)=PSV
     inv6
                     inspEndT(c)-inspBegT(c) \le max_insp_time_psv \land
                     inspEndT(c)>inspBegT(c)
     inv7
                     ventilPhase = cycles \rightarrow ventSates
                     ∀c· c∈ cycles
                     \max(\{x \cdot x \in dom(pcvParamsValC) \land x \leq inspBegT(c) \mid x\})
     inv8
                     dom(pcvParamsValC) ^
                     \max(\{x \cdot x \in \mathsf{dom}(\mathsf{psvParamsValC}) \land x \leq \mathsf{inspBegT}(c) \ | \ x\})
                      dom(psvParamsValC)
                     \forall \texttt{t,c} \cdot \texttt{c} \in \texttt{cycles} \ \land \ \texttt{t} \in \mathbb{N} \ \land \ \texttt{t} = \texttt{max}(\{\texttt{x} | \texttt{x} \in \texttt{dom}(\texttt{pcvParamsValC}) \ \land \ \texttt{x} \leq \ \texttt{inspBegT}(\texttt{c})\}) \ \land \ \texttt{x} \in \texttt{max}(\texttt{x} | \texttt{x} \in \texttt{dom}(\texttt{pcvParamsValC}) \ \land \ \texttt{x} \leq \ \texttt{inspBegT}(\texttt{c})\}) \ \land \ \texttt{x} \in \texttt{max}(\texttt{x} | \texttt{x} \in \texttt{dom}(\texttt{pcvParamsValC}) \ \land \ \texttt{x} \leq \ \texttt{inspBegT}(\texttt{c})\})
                      cycleMode(c)=PCV
     inv9
                      inspEndT(c)-inspBegT(c)=10*60*(pcvParamsValC(t))(IEPCV)÷
                         ((pcvParamsValC(t))(RRPCV)*(1 +(pcvParamsValC(t))(IEPCV)))
     inv10
```

```
∀c· c∈ cycles ∧ ventilPhase(c)≠inspBeg
                           inspBegT(c) <curTime</pre>
                          modeG=Settings
inv11
                          modeC∈Ventilationu{VentilationOff,FailSafe}
                           \forall t, c \cdot c \in cycles \land t \in \mathbb{N} \land t = max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\}) \land x \in max(\{x | x \in dom(pcvParamsValC) \land x \leq inspBegT(c)\})
                           cycleMode(c)=PCV
inv12
                           10*60*(pcvParamsValC(t))(IEPCV)÷
                               ((pcvParamsValC(t))(RRPCV)*(1 +(pcvParamsValC(t))(IEPCV))) >0
                           ∀c· c∈cycles
inv13
                          inspBegT(c)<inspEndT(c)</pre>
                          ∀c· c∈cycles ∧ ventilPhase(c)≠inspBeg
inv14
                          inspEndT(c) \leq curTime
inv15
                           PAW∈ 0..curTime→N
inv16
                          VE∈ 0..curTime→N
                          cycles≠ø⇒ PAW≠ø ∧ VE≠ø
inv17
                           modeC∉Ventilationu{FailSafe}
inv18
                                               cycles=ventilPhase~[{expEnd,expPauseEnd}]
                           cycles≠ø
inv19
                           comParamsValC≠ø ∧ pcvParamsValC≠ø ∧ psvParamsValC≠ø
                           ∀c· c∈cycles ∧ ventilPhase(c)=inspBeg
inv20
                          inspEndT(c)≥curTime
                           ∀c· c∈cycles
inv21
                          inspBegT(c)≤curTime
inv22
                           inspPauseBegT∈ ventilPhase~[ventSates\{inspBeg}]→N
inv23
                           inspPauseEndT \in dom(inspPauseBegT) \rightarrow N
                           \forall c \cdot c \in dom(inspPauseBegT) \Rightarrow
inv24
                           inspPauseEndT(c)≥inspPauseBegT(c)∧
                           inspPauseEndT(c) - inspPauseBegT(c) \leq inspPauseMax
                           ∀c·c∈dom(inspPauseBegT)
inv25
                           inspPauseBegT(c)≥inspEndT(c)
                           ventilPhase~[{inspPauseBeg}] ⊆ dom(inspPauseBegT) ∧
                           ventilPhase \sim [\{inspPauseEnd\}] \subseteq dom(inspPauseEndT) \ \land \\
inv26
                           ventilPhase~[{inspEnd}] ∩ dom(inspPauseEndT)=ø ∧
                           ventilPhase~[{inspBeg}] n dom(inspPauseEndT)=ø
inv27
                           rmBegT \in ventilPhase \sim [ventSates \setminus \{inspBeg\}] \ \ \rightarrow \ \ \mathbb{N}
inv28
                           rmEndT \in dom(rmBegT) \rightarrow N
                           ∀c,t· c∈dom(rmBegT) ∧
                           t=max(\{x | x \in dom(comParamsValC) \land x \leq inspBegT(c)\}) \land
inv29
                                 cycleMode(c)=PCV
                           rmEndT(c)- rmBegT(c)=(comParamsValC(t))(timerRM)
                           ∀c·c∈dom(rmBegT)
inv30
                           rmBegT(c) \ge inspEndT(c)
                          \forall c \cdot c \in dom(rmBegT) \land c \in dom(inspPauseEndT)
inv31
                           rmBegT(c) \ge inspPauseEndT(c)
                           ventilPhase~[{rmBeg}]⊆dom(rmBegT) ∧
inv32
                           ventilPhase~[{inspEnd,inspBeg,inspPauseBeg,inspPauseEnd}]ndom(rmBegT)=ø
                           ∀c· c∈ ventilPhase~[{inspPauseEnd}]
inv33
                           inspPauseEndT(c)≤curTime
                           ∀c,t· c∈ cycles ∧ cycleMode(c)=PSV ∧
                           t=max(\{x | x \in dom(psvParamsValC) \land x \le inspBegT(c)\})
inv34
                           (dom(psvParamsValC(t))=psvParams ^
                           (psvParamsValC(t))(ApneaLag)≥max_insp_time_psv÷2)
                           expBegT ∈ventilPhase~[{expBeg,expEnd,expPauseBeg,expPauseEnd}]
inv35
                           0..curTime
                           \texttt{expEndT} \in \texttt{dom(expBegT)} \ {\rightarrow} \mathbb{P}1(\mathbb{N})
inv36
```

```
inv37
                 \forallt,c· c∈ cycles \land t∈ \mathbb{N} \land c∈dom(expBegT) \land
                 t=max(\{x | x \in dom(pcvParamsValC) \land x \le inspBegT(c)\}) \land
                 c∈dom(expBegT) ∧ cycleMode(c)=PCV
                 (\exists k \cdot (k \in \mathbb{N} \land expEndT(c) = \{k\} \land
                  k-expBegT(c) \le 60 \div ((pcvParamsValC(t))(RRPCV) *
                                         (1 +(pcvParamsValC(t))(IEPCV)))))
                 ∀t,c· c∈ cycles ∧ t∈ N ∧
                 t=max(\{x \mid x \in dom(pcvParamsValC) \land x \le inspBegT(c)\}) \land
                 c∈dom(expBegT) ∧ cycleMode(c)=PSV
    inv38
                 (\exists k1,k2· k1\in N \land k2\inN \land k1\lek2 \landexpEndT(c)=k1..k2 \land
                 (k1-expBegT(c) \ge (inspEndT(c)-inspBegT(c)) \div 2) \land
                 (k2-expBegT(c) \le ((psvParamsValC(t)))(ApneaLag)))
                 \verb|expPauseBegTe| ventilPhase \sim [\{expPauseBeg, expPauseEnd\}] \rightarrow \mathbb{N}
    inv39
                 expPauseEndT \in dom(expPauseBegT) \rightarrow N
    inv40
                 \forall c \cdot c \in dom(expPauseBegT) \Rightarrow
   inv41
                 expPauseEndT(c)≥expPauseBegT(c)∧
                 expPauseEndT(c)-expPauseBegT(c)≤expPauseMax
                 modeC∈Ventilation
    inv42
                 card({c|c∈cycles ∧
                          ventilPhase(c)∉{expEnd,expPauseEnd}})≤1
EVENTS
   INITIALISATION
     extended
    STATUS
     ordinary
    BEGIN
                 power≔FALSE
      act1
      act2 : batLev:∈batteryRange
      act3
                 onAC:=TRUE
      act4
                 switchover≔TRUE
      act5
                  crashed:=FALSE
      act6
                 modeG=Off
      act7 :
                 modeGP:=Off
      act8
                 curTime≔0
             : batFail:∈B00L
      act9
      act10
                  modeC≔Off
      act11 :
                  modeCP:=Off
      act12 : comParamsValC≔ø
      act13 : comParamsValG≔ø
      act14 : pcvParamsValC≔ø
              : pcvParamsValG≔ø
      act15
      act16
                   psvParamsValG≔ø
      act17
                   psvParamsValC≔ø
      act18
                  offT≔0
      act19 :
                  PCV2PSV:=FALSE
      act20 :
                   ventilPhase≔ø
                   inspBegT≔ø
      act21
      act22
                   inspEndT≔ø
      act23
                   cycles≔ø
                   cycleMode≔ø
      act24
      act25
                   PAW≔ø
      act26
                   VE≔ø
      act27
                   inspPauseBegT≔ø
      act28
                   inspPauseEndT≔ø
      act29
                   rmBegT≔ø
      act30
                   rmEndT≔ø
      act31
                   expBegT≔ø
      act32
                   expEndT≔ø
      act33
                   expPauseBegT≔ø
      act34
                   expPauseEndT≔ø
    END
    powerON
                                 CONT. 2
     extended
    STATUS
     ordinary
    REFINES
     powerON
    WHEN
```

```
grd1
                                          power=FALSE
                                            onAC=TRUE v (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE)
      grd2
THEN
      act1
                                            power:=TRUE
                                           modeG={FALSE→StartUp, TRUE→Off}(crashed)
      act2
                                          modeGP:=modeG
      act3
      act4
                            : modeC≔StartUp
      act5
                            : modeCP:=modeC
END
powerOff
      extended
STATUS
      ordinary
REFINES
     powerOff
ANY
      val
WHERE
      grd1
                                              comParamsValC=ø v comParamsValC(max(dom(comParamsValC)))=defcomParams
      ard2
                                              \Rightarrow
                                              val=0
                                              comParamsValC≠ø ∧
                                              comParamsValC(max(dom(comParamsValC)))≠defcomParams
      grd3
                                              val=curTime
THEN
      act1
                                            power:=FALSE
                                            modeG=Off
      act2
                                           modeGP:=modeG
      act3
                                           modeC:=Off
      act4
                             : modeCP:=modeC
      act5
                            : offT≔val
      act6
      act7
                                          PCV2PSV:=FALSE
      act8
                                            ventilPhase≔ventilPhase⊳{expEnd,expPauseEnd}
      act9
                                            inspBegT = ventilPhase \sim [\{expEnd, expPauseEnd\}] \triangleleft inspBegT
      act10
                                              inspEndT=ventilPhase~[{expEnd,expPauseEnd}] dinspEndT
      act11
                                               cycles≔ventilPhase~[{expEnd,expPauseEnd}]
      act12
                                                cycleMode≔ ventilPhase~[{expEnd,expPauseEnd}]⊲cycleMode
      act13
                                                inspPauseBegT = ventilPhase \sim [\{expEnd, expPauseEnd\}] \land inspPauseBegT = ventilPhase \sim [\{expEnd, expPauseBegT = ventilPhase = v
      act14
                                                inspPauseEndT = ventilPhase \sim [\{expEnd, expPauseEnd\}] \land inspPauseEndT = ventilPhase \sim [\{expEnd, expPauseEndT = ventilPhase = v
      act15
                                                 rmBegT≔ventilPhase~[{expEnd,expPauseEnd}]⊲rmBegT
      act16
                                                rmEndT≔ventilPhase~[{expEnd,expPauseEnd}]⊲rmEndT
      act17
                                                expBegT = ventilPhase \sim [\{expEnd, expPauseEnd\}] \triangleleft expBegT
      act18
                                                expEndT≔ventilPhase~[{expEnd,expPauseEnd}]⊲expEndT
      act19
                                                expPauseBegT≔ventilPhase~[{expEnd,expPauseEnd}]⊲expPauseBegT
      act20
                                                expPauseEndT := ventilPhase \sim [\{expEnd, expPauseEnd\}] \triangleleft expPauseEndT
END
startUpEndedGui
      extended
STATUS
      ordinary
REFINES
      startUpEndedGui
ANY
      modeg
      pcvG
      comG
      psvG
WHERE
      grd1
                                           modeG=StartUp
                                           modeg∈{Start}uVentilation
      grd2
      grd3
                                            modeC \in Ventilation \implies modeg = modeC
      grd4
                                            modeC \not\in Ventilation \Rightarrow modeg = Start
      grd5
                                            modeC \in Ventilation \ V \ (offT \neq 0 \ \land \ curTime-offT \leq resumeTime)
                                                        comG=comParamsValG ◆
                                                                                                                {curTime →comParamsValC(max(dom(comParamsValC)))} ∧
                                                        pcvG=pcvParamsValG <
```

```
{curTime →pcvParamsValC(max(dom(pcvParamsValC)))} ∧
                psvG=psvParamsValG ∢
                                {curTime →psvParamsValC(max(dom(psvParamsValC)))}
             modeC \not\in Ventilation \land (offT=0 \lor curTime-offT > resumeTime)
                comG=comParamsValG ∢
                                {curTime →defcomParams} ∧
 grd6
                pcvG=pcvParamsValG ◆
                                {curTime →defpcvParams} ∧
                psvG=psvParamsValG ◆
                                {curTime →defpsvParams}
THEN
 act1
            modeG≔modeg
 act2
            modeGP:=modeG
 act3
            comParamsValG=comG
        1
 act4
            pcvParamsValG≔pcvG
 act5
            psvParamsValG≔psvG
END
startUpEndedCont =
 extended
STATUS
 ordinary
REFINES
 startUpEndedCont
ANY
 comC
 рсиС
 psvC
WHERE
 grd1
            modeC=StartUp
             offT=0 \ v \ curTime-offT > resumeTime
             comC=comParamsValC ∢
                                {curTime →defcomParams} ∧
 grd2
             pcvC=pcvParamsValC
                                {curTime →defpcvParams} ∧
             psvC=psvParamsValC ∢
                                {curTime →defpsvParams}
             offT≠0 ∧ curTime-offT ≤ resumeTime
             comC=comParamsValC ∢
                                {curTime →comParamsValC(max(dom(comParamsValC)))} ∧
 grd3
             pcvC=pcvParamsValC ∢
                                {curTime →pcvParamsValC(max(dom(pcvParamsValC)))} ∧
            psvC=psvParamsValC ∢
                                {curTime →psvParamsValC(max(dom(psvParamsValC)))}
THEN
 act1
            modeC≔SelfTest
            modeCP:=modeC
 act2
 act3
            comParamsValC:=comC
 act4
        : pcvParamsValC≔pcvC
            psvParamsValC≔psvC
 act5
END
crash
 extended
STATUS
 ordinary
REFINES
 crash
WHEN
            crashed=FALSE
 grd1
THEN
 act1
            crashed≔TRUE
 act2
            modeG=Off
            modeGP:=Off
 act3
            PCV2PSV:=FALSE
 act4
END
repare ≜
 extended
STATUS
```

```
ordinary
REFINES
 repare
ANY
 modeg
WHERE
 grd1
            crashed=TRUE
             power=FALSE v (onAC=FALSE \( \) (switchover=FALSE v batLev=0 v batFail=TRUE))
  grd2
             modeg=Off
             power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))
  grd3 :
            modeg=StartUp
  grd4 : modeC \in Ventilation \Rightarrow modeg = modeC
            power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))\( \)
 grd5 : modeC∉Ventilation
             modeg=StartUp
THEN
 act1 : crashed=FALSE
act2 : modeG=modeg
 act3 : modeGP≔modeG
newPatient ≜
 extended
STATUS
 ordinary
REFINES
 newPatient
 grd1 : modeG=Start
 grd2 : modeC=SelfTest
 act1 : modeG≔SelfTest
 act2 : modeGP≔modeG
END
resumeVent ≜
 extended
STATUS
 ordinary
REFINES
 resumeVent
WHEN
 grd1 : modeG=Start
 grd2 : modeC=SelfTest
grd3 : offT≠0 ∧ curTime-offT ≤ resumeTime
THEN
 act1 : modeG≔Menu
 act2 : modeGP≔modeG
 act3 : modeC≔VentilationOff
 act4 : modeCP≔modeC
runAbortSelfTest ≜
 extended
STATUS
 ordinary
REFINES
 runAbortSelfTest
WHEN
 grd1 : modeG=SelfTest
grd2 : modeC=SelfTest
THEN
 act1 : modeG≔SelfTest
 act2 : modeGP≔modeG
END
                         // CONT4-4.1
selfTestPassed ≜
 extended
STATUS
```

```
ordinary
REFINES
 selfTestPassed
WHEN
           modeG=SelfTest
 grd1
 grd2 : modeC=SelfTest
THEN
 act1 : modeG≔Menu
 act2 : modeGP≔modeG
act3 : modeC≔VentilationOff
 act4 : modeCP:=VentilationOff
END
setParam ≜
 extended
STATUS
 ordinary
REFINES
 setParam
WHEN
            modeG∈{Menu}uVentilation
 grd1 :
THEN
 act1 : modeG=Settings
 act2 : modeGP≔modeG
settingParams
 extended
STATUS
 ordinary
REFINES
 settingParams
ANY
 psvP
 pcvP
 comP
WHERE
 grd1 : modeG=Settings
 grd2 : max(dom(psvParamsValG))<curTime</pre>
 grd3 : psvP∈ psvParams→N1
 grd4 : psvParams \setminus \{RRAP, PinspAP\} \subseteq dom(psvP)
       : pcvP∈ pcvParams→N1
 grd5
 grd6
           comP∈ comParams→N1
THEN
 act1 : psvParamsValG(curTime)≔psvP
 act2 : pcvParamsValG(curTime)≔pcvP
 act3 : comParamsValG(curTime)≔comP
END
saveBackAbort ≜
 extended
STATUS
 ordinary
REFINES
 saveBackAbort
ANY
 modeg
 modec
 pcvC
 psvC
 comC
 pcvG
 psvG
 comG
WHERE
 grd1
       : modeG=Settings ∧ modeg∈ModesG
 grd2 : modeg∈Ventilationu{Menu}
 grd3 : modeC≠FailSafe
 grd4
           modec∈ModesC
 grd5
       : modeC∈Ventilation ⇒ modec∈Ventilation ∧ modeg=modec
```

```
grd6
             modeC∉Ventilation ⇒modec=modeC ∧ modeg=Menu
             sv \in \{TRUE, FALSE\}
  grd7
              psvG={TRUE↔
  grd8
             psvParamsValG(max(dom(psvParamsValG))),
              FALSE \mapsto psvParamsValC(max(dom(psvParamsValC)))\}(sv)
              psvC={TRUE↔
  grd9
             psvParamsValG(max(dom(psvParamsValG))),
               FALSE⇔psvParamsValC(max(dom(psvParamsValC)))}(sv)
               pcvG={TRUE→
  grd10
               pcvParamsValG(max(dom(pcvParamsValG))),
                FALSE→pcvParamsValC(max(dom(pcvParamsValC)))}(sv)
               pcvC={TRUE↔
  grd11
               pcvParamsValG(max(dom(pcvParamsValG))),
                FALSE→pcvParamsValC(max(dom(pcvParamsValC)))}(sv)
               comG={TRUE→
  grd12
               comParamsValG(max(dom(comParamsValG))),
                \textit{FALSE} \mapsto \textit{comParamsValC}(\textit{max}(\textit{dom}(\textit{comParamsValC})))\}(\textit{sv})
               comC={TRUE→
  grd13
               comParamsValG(max(dom(comParamsValG))),
                FALSE⇔comParamsValC(max(dom(comParamsValC)))}(sv)
               PCV2PSV=TRUE ∧ modeC=PCV⇒modec=PSV
  grd14
  grd15
               PCV2PSV=FALSE v modeC≠PCV⇒modec=modeC
               modec=PSV
  grd16
               dom(psvC)=psvParams Λ
               psvC(ApneaLag)≥max_insp_time_psv÷2
  grd17
               modeC \neq modec \implies PCV2PSV = TRUE \land modeC = PCV \land modec = PSV
               modec=PCV∧ sv=TRUE⇒
  grd18
               10*60*pcvC(IEPCV)÷ (pcvC(RRPCV)*(1 +pcvC(IEPCV))) >0
  grd19
               curTime∉dom(pcvParamsValC)
  grd20
               \forall c \cdot c \in cycles \Rightarrow inspBegT(c) \neq curTime
THEN
  act1
             modeG:=modeg
  act2
             modeGP:=modeG
             modeC:=modec
  act3
  act4
         : modeCP:=modeC
  act5
         : psvParamsValG(curTime)≔psvG
  act6
         : psvParamsValC(curTime)≔psvC
  act7
             pcvParamsValG(curTime)≔pcvG
  act8
             pcvParamsValC(curTime)≔pcvC
  act9
             comParamsValG(curTime)≔comG
  act10
              comParamsValC(curTime)≔comC
  act11
         : PCV2PSV=FALSE
END
startStopPCVPSV
  extended
STATUS
  ordinary
REFINES
  startStopPCVPSV
ANY
  modeg
  modec
WHERE
             modeG∈{Menu}uVentilation
  grd1
  grd2
             modeg∈Ventilationu{Menu}
             modeG=Menu⇒modeg∈Ventilation
  ard3
             modeG∈Ventilation⇒modeg=Menu
  grd4
  grd5
             modeC≠FailSafe
  ard6
             modeg \in Ventilation \implies modec = modeg
  grd7
             modeg \not\in Ventilation \Rightarrow modec = VentilationOff
              modeg=PSV
  ard8
              dom(psvParamsValC(max(dom(psvParamsValC))))=psvParams
  grd9
              (psvParamsValC(max(dom(psvParamsValC))))(ApneaLag)≥max_insp_time_psv÷2
  grd10
               10*60*(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)÷
                 ((pcvParamsValC(max(dom(pcvParamsValC))))(RRPCV)*
```

```
(1 +(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV))) >0
 grd11
              cycles=ventilPhase~[{expEnd,expPauseEnd}]
THEN
  act1
             modeG:=modeg
            modeGP:=modeG
  act2
  act3
            modeC≔modec
  act4
             modeCP≔{TRUE→modeC, FALSE→modeCP} (bool (modeC∈Ventilation υ{VentilationOff}))
END
moveToPSV
 extended
STATUS
 ordinary
REFINES
 moveToPSV
ANY
  modeg
  modec
WHERE
            modeG∈ Ventilation
  grd1
            modeg∈Ventilationu{Settings}
  grd2
            modeC∈Ventilation\{FailSafe}
  ard3
  grd4
            modec∈Ventilation
  grd5
            modeG=PCV
  grd6
            modeC=PCV
  grd7
             modeg=Settings
  grd8
             modec=modeC
THEN
  act1
             modeG≔modeg
            modeGP:=modeG
  act2
  act3
            modeC:=modec
  act4
            modeCP:=modeC
            PCV2PSV:=TRIJF
  act5
END
extended
STATUS
 ordinary
REFINES
  failExternalPower
  modeg
  modegp
  modec
  modecp
  pcv2psv
  cycs
WHERE
  grd1
             onAC=TRIJF
  grd2
             modeg ∈ ModesG ∧ modegp ∈ ModesG
  grd3
             \neg (power=TRUE \land crashed=FALSE) \Rightarrow modeg=Off \land modegp=Off
             power=TRUE ∧ crashed=FALSE ⇒
                grd4
                (\neg (switchover = TRUE \ \land \ batLev > 0 \ \land \ batFail = FALSE) \Rightarrow \ modeg = 0 ff \ \land \ modegp = modeG))
  grd5
         1
             modec \in ModesC \land modecp \in ModesC
  grd6
             power= FALSE ⇒ modec=modeC ∧ modecp=modeCP
              power= TRUE ⇒
  grd7
                 ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE ⇒modec=modeC ∧ modecp=modeCP) ∧
                  (\neg (switchover = TRUE \ \land \ batLev > \theta \ \land \ batFail = FALSE) \Rightarrow modec = 0 ff \ \land \ modecp = modeC))
             ¬(power=TRUE ∧ crashed=FALSE) ⇒ pcv2psv=FALSE
  grd8
             power=TRUE ∧ crashed=FALSE ⇒
  grd9
             ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE⇒ pcv2psv=PCV2PSV) ∧
             (¬(switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)⇒ pcv2psv=FALSE))
  grd10
              (switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)
```

```
cycs=cycles
              (switchover=FALSE v batLev=0 v batFail=TRUE)
 grd11
             cycs=ventilPhase~[{expEnd,expPauseEnd}]
 grd12 : cycs⊆cycles
THEN
 act1
       : onAC≔FALSE
 act2 : modeG≔modeg
       : modeGP≔modegp
: modeC≔modec
 act3
 act4
 act5 : modeCP≔modecp
 act6 : PCV2PSV:=pcv2psv
 act7 : ventilPhase≔cycs⊲ventilPhase
 act8 : inspBegT≔cycs⊲inspBegT
 act9
        : inspEndT≔cycs⊲inspEndT
 act10
             cycles≔cycs
 act11 : cycleMode≔ cycs⊲cycleMode
 act12 : inspPauseBegT≔cycs⊲inspPauseBegT
 act13 : inspPauseEndT≔cycs⊲inspPauseEndT
 act14 : rmBegT≔cycs⊲rmBegT
 act15
             rmEndT≔cycs⊲rmEndT
 act16
             expBegT≔cycs⊲expBegT
 act17 : expEndT≔cycs⊲expEndT
 act18 : expPauseBegT≔cycs⊲expPauseBegT
 act19 : expPauseEndT≔cycs⊲expPauseEndT
END
failSelfTest ≜
 extended
STATUS
 ordinary
REFINES
 failSelfTest
WHEN
 grd1 : modeC=SelfTest
THEN
 act1 : modeC≔FailSafe
 act2
       : modeCP≔modeC
END
progress ≜
 extended
STATUS
 ordinary
REFINES
 progress
ANY
 step
 Z
 modeg
 batf
 modec
 paw
 cycs
WHERE
 grd1 : step∈N1 ∧ l∈batteryRange ∧ batf∈B00L
 grd2 : batFail=TRUE \Rightarrow l=batLev
 \textit{grd3} \quad : \quad \textit{onAC=TRUE} \; \land \; \textit{batFail=FALSE} \; \Leftrightarrow \; \textit{l=batLev+step}
        : onAC=FALSE ∧ batFail=FALSE ⇔ l=max({0,batLev-step})
 grd4
             (onAC=TRUE ν (l>0 ∧ switchover=TRUE ∧ batf=FALSE)) ∧
            power=TRUE A modeG=Off A
 grd5 :
            crashed=FALSE
            modeg=StartUp
 grd6
            (l=0 v batf=TRUE v switchover=FALSE) ∧ onAC=FALSE ⇒modeg=Off
            ((l>0 Λ batLev>0) Λ switchover=TRUE Λ batf=FALSE)
 grd7 :
            modea=modeG
 grd8
            modec∈{FailSafe,modeC, Off,StartUp}
 grd9
            modec=FailSafe⇒modeC≠Off
```

```
(onAC=TRUE v (l>0 \land switchover=TRUE \land batf=FALSE)) \land
 grd10
              power=TRUE A modeC=Off
              modec=StartUp
              (batLev>0 \land l>0)\lor switchover=FALSE \lor onAC=TRUE \lor
              power=FALSE
 grd11
             modec∈{modeC,FailSafe}
              (l=0 v batf=TRUE v switchover=FALSE) \( \lambda \) on AC=FALSE
 grd12
             modec=Off
 grd13
             step∈N1 ∧ paw∈N
             ∀c· c∈cycles ∧ ventilPhase(c)=inspBeg
 grd14
             curTime+step≤inspEndT(c)
             ∀c· c∈cycles ∧ ventilPhase(c)=inspPauseBeg
 grd15
             curTime+step≤inspPauseEndT(c)
             ∀c· c∈cycles ∧ ventilPhase(c)=rmBeg
 grd16
             curTime+step≤ rmEndT(c)
             ∀c· c∈cycles ∧ ventilPhase(c)=expBeg
 grd17
             curTime+step∈expEndT(c)
             ∀c· c∈cycles ∧ ventilPhase(c)=expPauseBeg
 grd18 :
             curTime+step ≤ expPauseEndT(c)
             \verb|modec@Ventilationu{FailSafe}| \implies \verb|cycs=ventilPhase=[\{expEnd,expPauseEnd\}]||
 grd19 :
 \verb|grd20| : modeceVentilationu{FailSafe}| \Rightarrow cycs=cycles
 grd21
             cycs⊆cycles
THEN
 act1 : curTime≔curTime+step
 act2 : batLev=l
 act3 : modeG≔modeg
        : modeGP:={TRUE→modeGP, FALSE→modeG}(bool(modeg=modeG))
 act4
 act5
            batFail≔batf
 act6
            modeC:=modec
 act7
       : modeCP:={TRUE→modeCP,FALSE→modeC}(bool(modeC=modec))
           PCV2PSV≔{TRUE+PCV2PSV, FALSE+FALSE}(bool(
 act8 :
            (batLev>0 ^ l>0^ switchover=FALSE) v onAC=TRUE))
 act9
       : ventilPhase≔cycs⊲ventilPhase
        : inspBegT≔cycs⊲inspBegT
 act10
             inspEndT≔cycs⊲inspEndT
 act11
         .
 act12
             cycles≔cycs
        : cycleMode≔cycs⊲cycleMode
 act13
 act14 : inspPauseBegT≔cycs⊲inspPauseBegT
 act15 : inspPauseEndT≔cycs⊲inspPauseEndT
        : rmBegT≔cycs⊲rmBegT
 act16
 act17
         :
             rmEndT≔cycs⊲rmEndT
 act18
             expBegT≔cycs⊲expBegT
 act19
             expEndT≔cycs⊲expEndT
 act20
             expPauseBegT:=cycs<expPauseBegT
 act21 :
             expPauseEndT≔cycs⊲expPauseEndT
END
switchoverFail ≜
 extended
STATUS
 ordinary
REFINES
 switchoverFail
ANY
 modeg
 modec
 CVCS
WHERE
 grd1
            switchover=TRUE
 grd2 : onAC=FALSE⇒modeg=Off
            onAC=TRUE⇒modeg=modeG
 ard3 :
            onAC=FALSE⇒modec=Off
 grd4
 grd5
            onAC=TRUE⇒modec=modeC
```

```
modec∉Ventilationu{FailSafe} ⇒ cycs=ventilPhase~
  grd6
              [{expEnd,expPauseEnd}]
  grd7
              modec∈Ventilationu{FailSafe} ⇒ cycs=cycles
  grd8
              cycs⊆cycles
THEN
  act1
              switchover≔FALSE
  act2
             modeG:=modea
             modeGP:={TRUE→modeG, FALSE→modeGP}(bool(modeG≠modeg))
  act3
  act4
              modeC:=modec
  act5
             modeCP:={TRUE→modeC, FALSE→modeCP}(bool(modeC≠modec))
             PCV2PSV:={TRUE→PCV2PSV, FALSE→FALSE}(bool(onAC=TRUE))
  act6
  act7
             ventilPhase≔cycs⊲ventilPhase
  act8
             inspBegT≔cycs⊲inspBegT
  act9
              inspEndT≔cycs⊲inspEndT
  act10
              cycles≔cycs
  act11
               cycleMode≔cycs⊲cycleMode
  act12
              inspPauseBegT≔cycs⊲inspPauseBegT
  act13 :
               \verb"inspPauseEndT"=cycs < \verb"inspPauseEndT"
  act14
               rmBegT≔cycs⊲rmBegT
  act15
               rmEndT≔cycs⊲rmEndT
  act16
               expBegT≔cycs⊲expBegT
               \texttt{expEndT} \small{=} \texttt{cycs} \small{\triangleleft} \texttt{expEndT}
  act17
  act18
               expPauseBegT≔cycs⊲expPauseBegT
  act19
               expPauseEndT \mathbin{\coloneqq} cycs \sphericalangle expPauseEndT
END
inspStart ≜
STATUS
  ordinary
ANY
  CV
  paw
  ve
  inspT
WHERE
             modeC∈Ventilation ∧ inspT∈N
  grd1
  grd2
              cy∈Cycles\cycles
  grd3
              ran(ventilPhase)⊆{expEnd,expPauseEnd}
              modeC=PCV⇒inspT=curTime + 10*60*
               (pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)÷
  grd4
                     ((pcvParamsValC(max(dom(pcvParamsValC))))(RRPCV)*
                   (1 +(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)))
              modeC=PSV⇒
  grd5
              inspT>curTime ∧
              \verb"inspTscurTime" + \verb"max_insp_time_psv"
  grd6
              paw∈N ∧ ve∈ N
  grd7
              pcvParamsValC≠ø ∧ psvParamsValC≠ø ∧ comParamsValC≠ø
THEN
              ventilPhase(cy)≔inspBeg
  act1
  act2
              cycles≔cyclesu{cy}
  act3
              inspBegT(cy)≔curTime
  act4
             inspEndT(cy)≔inspT
  act5
             cycleMode(cy)≔modeC
  act6
             PAW(curTime)≔paw
  act7
             VE(curTime)≔ve
END
inspEnd
STATUS
 ordinary
ANY
  су
  ve
  peakVE
WHERE
  grd1
              cy∈ cycles ∧ ventilPhase(cy)=inspBeg ∧ ve∈N
  grd2
              cycleMode(cy)=PSV⇒curTime>inspBegT(cy)
              cycleMode(cy)=PCV ⇒curTime=inspEndT(cy)
  ard3
  grd4
              peakVE=VE(max(dom(VE)))
  grd5
              (curTime=inspEndT(cy)) v
              (cycleMode(cy)=PSV \Rightarrow
```

```
peakVE>ve ^
            ve<(psvParamsValC(max(dom(psvParamsValC))))(ETS)*peakVE)</pre>
THEN
            ventilPhase(cy)≔inspEnd
 act1
            VE(curTime)≔ve
 act2
 act3 : inspEndT(cy)≔curTime
END
inspPauseStart ≜
STATUS
 ordinary
ANY
 CV
WHERE
            cy∈cycles ∧ ventilPhase(cy)=inspEnd
 grd1
THEN
 act1
       : ventilPhase(cy)≔inspPauseBeg
 \verb"act2": inspPauseBegT(cy) = \verb"curTime"
 act3 : inspPauseEndT(cy)=curTime+inspPauseMax
END
inspPauseEnd
STATUS
 ordinary
ANY
 су
WHERE
 grd1 : cy∈cycles ∧ ventilPhase(cy)=inspPauseBeg
 grd2 : curTime≤inspPauseEndT(cy)
 grd3 : curTime>inspPauseBegT(cy)
THEN
 act1 : ventilPhase(cy)≔inspPauseEnd
 act2 : inspPauseEndT(cy)≔curTime
END
rmStart
STATUS
 ordinary
ANY
 су
 t
WHERE
 grd1
            cy∈cycles
           ventilPhase(cy)∈{inspEnd, inspPauseEnd}
 grd2
           t=max(\{x|x\in dom(comParamsValC) \land x \le inspBegT(cy)\})
 grd3 :
THEN
           ventilPhase(cy)≔rmBeg
 act1
 act2
            rmBegT(cy)≔curTime
 act3
            rmEndT(cy)≔curTime+(comParamsValC(t))(timerRM)
END
rmEnd
STATUS
 ordinary
ANY
 су
WHERE
 grd1
            cy∈cycles
      :
 grd2 :
            ventilPhase(cy)=rmBeg
 grd3
            curTime=rmEndT(cy)
THEN
 act1
            ventilPhase(cy)≔rmEnd
END
expStart
STATUS
 ordinary
ANY
 су
 t
 expT
```

```
WHERE
              cy {\in} cycles \ \land \ ventilPhase(cy) {\in} \{inspEnd, inspPauseEnd, rmEnd\}
  grd1
              t=max({x|x\in dom(pcvParamsValC) \land x \le inspBegT(cy)})
  grd2
  grd3
              expT⊆N
              cycleMode(cy)=PCV \Rightarrow
              expT={curTime +60÷((pcvParamsValC(t))(RRPCV) *
  grd4
                          (1 +(pcvParamsValC(t))(IEPCV)))}
              cycleMode(cy)=PSV \Rightarrow
              expT=
  grd5
              (curTime+(inspEndT(cy)-inspBegT(cy))÷2)
              (curTime +((psvParamsValC(t)))(ApneaLag))
THEN
  act1
              ventilPhase(cy)≔expBeg
  act2
              expBegT(cy)≔curTime
              expEndT(cy)≔expT
  act3
END
expEnd
 extended
STATUS
  ordinary
REFINES
 changeMode
ANY
  modeg
  modec
  pcv
  paw
  су
  lastTime
  t
WHERE
  grd1
              modeG∈ Ventilation
  grd2
              modeg∈Ventilationu{Settings}
  grd3
              modeC∈Ventilation\{FailSafe}
  grd4
              modec \in Ventilation
              pcv ∈ pcvParams→N1
  grd5
  grd6
              modeC=PCV ⇒ modec=PCV
              modeC=PSV ^ modec=PCV
              pcv=pcvParamsValC(max(dom(pcvParamsValC)))
  grd7
              \{\textit{RRPCV} \mapsto (\textit{psvParamsValC}(\textit{max}(\textit{dom}(\textit{psvParamsValC})))) \ (\textit{RRAP}) \ ,
               PinspPCV → (psvParamsValC(max(dom(psvParamsValC))))(PinspAP),
              IEPCV→IEAP}
              modeC=PCV v modec=PSV
  grd8
              pcv=pcvParamsValC(max(dom(pcvParamsValC)))
              modec=PCV
  grd9
              10*60*pcv(IEPCV)÷ (pcv(RRPCV)*(1 +pcv(IEPCV))) >0
  grd11
               modeG \neq 0ff \Rightarrow modeg = modec
  grd12
               cy∈cycles
               ventilPhase(cy)=expBeg
  grd13
  grd14
               lastTime= max(dom(PAW))
  grd15
               paw∈N
  grd16
               t=max({x | x∈dom(psvParamsValC)∧x≤inspBegT(cy)})
               cycleMode(cy)=PCV \land
                    (curTime \in expEndT(cy) v
                      (curTime<max(expEndT(cy)) ^</pre>
                        (PAW(lastTime)<paw ^ PAW(lastTime)-paw>(pcvParamsValC(t))(ITSPCV)))
  grd17
               ٧
               cycleMode(cy)=PSV ^
                    (curTime= max(expEndT(cy)) v
                      (curTime≥min(expEndT(cy)) ∧ curTime<max(expEndT(cy)) ∧ PAW(lastTime)<paw ∧ PAW
               (lastTime)-paw>(psvParamsValC(t))(ITSPSV)))
  ard18
               curTime=max(expEndT(cy)) \( \) cycleMode(cy)=PSV
```

```
modec=PCV
             curTime≠max(expEndT(cy))∧ cycleMode(cy)=PSV
 grd19
             modec=PSV
             cycleMode(cy)=PCV
 grd20
             modec=PCV
THEN
 act1
            modeG:=modeg
           modeGP:=modeG
 act2
 act3
           modeC≔modec
 act4
        : modeCP≔modeC
 act5
            pcvParamsValC(curTime)≔pcv
 act6
            psvParamsValC(curTime):=psvParamsValC(max(dom(psvParamsValC)))
            comParamsValC(curTime) = comParamsValC(max(dom(comParamsValC)))
 act7
 act8 : ventilPhase(cy)≔expEnd
 act9 : PAW(curTime)≔paw
 act10 : expEndT(cy)≔curTime.curTime
expPauseStart ≜
STATUS
 ordinary
ANY
 су
WHERE
            cy∈cycles ∧ ventilPhase(cy)=expEnd
 grd1
 grd2 :
            modeC∈Ventilation
            ran(ventilPhase)⊆{expEnd,expPauseEnd}
 grd3
THEN
 act1
            ventilPhase(cy)≔expPauseBeg
            expPauseBegT(cy)≔curTime
 act2
            expPauseEndT(cy):=curTime+expPauseMax
 act3 :
END
expPauseEnd
STATUS
 ordinary
ANY
 су
WHERE
 grd1
            cy∈cycles ∧ ventilPhase(cy)=expPauseBeg
 grd2
            curTime≤expPauseEndT(cy)
            curTime>expPauseBegT(cy)
 grd3
            ventilPhase(cy)≔expPauseEnd
 act1
 act2
            expPauseEndT(cy)≔curTime
END
```

END

```
MACHINE
            Valves
REFINES
            VentilationPhases
SEES
            ContStates
            VentilStates
            PSVParams
VARIABLES
            power
            onAC
            batLev
            batFail
            switchover
            crashed
            modeG
            modeGP
            modeC
            modeCP
            PCV2PSV
             comParamsValC
            comParamsValG
            pcvParamsValC
            pcvParamsValG
            psvParamsValC
            psvParamsValG
            curTime
            offT
            cycles
            cycleMode
            ventilPhase
            inspEndT
            inspBegT
            inspPauseBegT
            \verb"inspPauseEndT"
            PAW
            ٧E
            rmBegT
            rmEndT
            expBegT
            expEndT
            expPauseBegT
            expPauseEndT
            inValve
            outValve
            inValveF
            outValveF
INVARIANTS
                                                     inValve ∈ BOOL ∧ outValve∈ BOOL ∧
            inv1
                                                     inValveF ∈ BOOL ∧ outValveF∈ BOOL
                                                    modeC∉Ventilation ∧ outValveF=FALSE
             inv3
                                                     outValve=TRUE
                                                     (\exists c \cdot c \in cycles \land ventilPhase(c) \in \{inspBeg, inspEnd\}) \land inValveF=FALSE \land inValveF=FALS
                                                    modeC∈Ventilation
            inv4
                                                    inValve=TRUE
                                                     (\exists c \cdot c \in cycles \land ventilPhase(c) \in \{inspBeg, inspEnd\}) \land outValveF=FALSE
                                                    \land \ \mathsf{modeC} \mathord{\in} \mathsf{Ventilation}
            inv5
                                                    outValve=FALSE
                                                     (\exists c \cdot c \in cycles \land ventilPhase(c)=expBeg) \land inValveF=FALSE
             {\tt inv6}
                                                    \Rightarrow
                                                    \verb"inValve=FALSE"
                                                     (\exists c \cdot c \in cycles \land ventilPhase(c)=expBeg) \land outValveF=FALSE
            inv7
                                                    \Rightarrow
                                                     outValve=TRUE
            inv8
                                                    (\exists c \cdot c \in cycles \land ventilPhase(c) \in
```

{expPauseBeg,inspPauseBeg,inspPauseEnd}) ∧ inValveF=FALSE

```
inValve=FALSE
              (\exists c \cdot c \in cycles \land ventilPhase(c) \in
                 \{expPauseBeg, inspPauseBeg, inspPauseEnd\}) \land
   inv9
                 outValveF=FALSE ∧ modeC∈Ventilation
               (\exists c \cdot c \in cycles \land ventilPhase(c) \in \{rmBeg, rmEnd\}) \land inValveF = FALSE \land
               modeC∈Ventilation
   inv10
               inValve=TRUE
               (\exists c \cdot c \in cycles \land ventilPhase(c) \in \{rmBeg, rmEnd\}) \land outValveF=FALSE
               ^ modeC∈Ventilation
   inv11 :
               outValve=FALSE
EVENTS
   INITIALISATION ≜
     extended
   STATUS
     ordinary
   BEGIN
     act1 :
               power≔FALSE
     act2 : batLev:∈batteryRange
     act3 : onAC≔TRUE
           : switchover≔TRUE
     act4
     act5
               crashed:=FALSE
     act6
               modeG=Off
     act7 : modeGP:=Off
     act8 : curTime≔0
     act9 : batFail:∈B00L
     act10
            : modeC≔Off
     act11
                modeCP:=Off
     act12 :
               comParamsValC≔ø
     act13 : comParamsValG≔ø
     act14 : pcvParamsValC≔ø
     act15 : pcvParamsValG≔ø
            : psvParamsValG≔ø
     act16
     act17
                psvParamsValC≔ø
     act18 :
                offT:=0
     act19 : PCV2PSV:=FALSE
     act20 : ventilPhase≔ø
     act21 : inspBegT≔ø
     act22 : inspEndT≔ø
     act23
                cycles:=ø
     act24
                cycleMode≔ø
     act25 : PAW≔ø
     act26 : VE≔ø
                inspPauseBegT≔ø
     act27 :
     act28
                inspPauseEndT≔ø
     act29
                rmBegT≔ø
                rmEndT≔ø
     act30
     act31 :
                expBegT≔ø
     act32 : expEndT≔ø
     act33 : expPauseBegT≔ø
     act34 : expPauseEndT≔ø
     act35
                inValve≔FALSE
           : outValve≔TRUE
     act36
     act37 : inValveF≔FALSE
     act38 : outValveF≔FALSE
   END
   powerON ≜
                        // CONT.2
    extended
   STATUS
     ordinary
   REFINES
     powerON
   WHEN
     grd1 : power=FALSE
     grd2 :
               onAC=TRUE v (switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)
   THEN
```

```
act1 : power≔TRUE
 act2
            modeG≔{FALSE⇔StartUp, TRUE→Off}(crashed)
 act3
 act4
            modeC≔StartUp
 act5
            modeCP:=modeC
END
powerOff
 extended
STATUS
 ordinary
REFINES
 powerOff
ANY
 val
WHERE
 grd1
             power=TRUE
             comParamsValC=@ v comParamsValC(max(dom(comParamsValC)))=defcomParams
 grd2
             val=0
             comParamsValC≠ø ∧
             comParamsValC(max(dom(comParamsValC)))≠defcomParams
 grd3
             val=curTime
THEN
 act1
             power:=FALSE
 act2
            modeG=Off
 act3
            modeGP:=modeG
 act4
            modeC≔Off
        : modeCP:=modeC
 act5
            offT≔val
 act6
        : PCV2PSV:=FALSE
 act7
 act8
        : ventilPhase≔ventilPhase⊳{expEnd,expPauseEnd}
 act9
        : inspBegT≔ventilPhase~[{expEnd,expPauseEnd}]⊲inspBegT
 act10 : inspEndT≔ventilPhase~[{expEnd,expPauseEnd}]⊲inspEndT
 act11
         : cycles≔ventilPhase~[{expEnd,expPauseEnd}]
 act12
              cycleMode= ventilPhase~[{expEnd,expPauseEnd}]<cycleMode</pre>
 act13
              inspPauseBegT = ventilPhase \sim [\{expEnd, expPauseEnd\}] < inspPauseBegT
 act14
             inspPauseEndT≔ventilPhase~[{expEnd,expPauseEnd}]⊲inspPauseEndT
 act15
              rmBegT:=ventilPhase~[{expEnd,expPauseEnd}]⊲rmBegT
 act16
              rmEndT:=ventilPhase~[{expEnd,expPauseEnd}]⊲rmEndT
 act17
              expBegT≔ventilPhase~[{expEnd,expPauseEnd}]⊲expBegT
 act18
              expEndT:=ventilPhase~[{expEnd,expPauseEnd}]⊲expEndT
 act19
              expPauseBegT = ventilPhase \sim [\{expEnd, expPauseEnd\}] \triangleleft expPauseBegT
 act20
             expPauseEndT≔ventilPhase~[{expEnd,expPauseEnd}]⊲expPauseEndT
 act21 : inValve≔FALSE
 act22 :
             outValve≔TRUE
END
startUpEndedGui ≜
 extended
STATUS
 ordinary
REFINES
 startUpEndedGui
ANY
 modeg
 pcvG
 comG
 psvG
WHERE
 ard1
            modeG=StartUp
 grd2
            modeg∈{Start}uVentilation
 grd3
            modeC \in Ventilation \implies modeg = modeC
             modeC \not\in Ventilation \Rightarrow modeg = Start
 grd4
        - 1
 grd5
             modeC \in Ventilation \ V \ (offT \neq 0 \ \land \ curTime-offT \leq resumeTime)
                comG=comParamsValG ∢
                                 {curTime →comParamsValC(max(dom(comParamsValC)))} ∧
                pcvG=pcvParamsValG ∢
                                 {curTime →pcvParamsValC(max(dom(pcvParamsValC)))} ∧
                psvG=psvParamsValG ◆
```

```
{curTime →psvParamsValC(max(dom(psvParamsValC)))}
             modeC \not\in Ventilation \land (offT=0 \lor curTime-offT > resumeTime)
                comG=comParamsValG ∢
                                {curTime →defcomParams} ∧
 grd6
                pcvG=pcvParamsValG ∢
                                {curTime →defpcvParams} ∧
                psvG=psvParamsValG ∢
                                {curTime →defpsvParams}
THEN
 act1
            modeG:=modeg
 act2
            modeGP:=modeG
            comParamsValG:=comG
 act3
 act4 : pcvParamsValG≔pcvG
 act5 : psvParamsValG≔psvG
END
startUpEndedCont =
 extended
STATUS
 ordinary
REFINES
 {\it startUpEndedCont}
ANY
 comC
 pcvC
 psvC
WHERE
            modeC=StartUp
 grd1
             offT=0 v curTime-offT > resumeTime
             comC=comParamsValC ∢
                                {curTime →defcomParams} ∧
 grd2
             pcvC=pcvParamsValC ❖
                                {curTime →defpcvParams} ∧
             psvC=psvParamsValC ∢
                                {curTime →defpsvParams}
             offT\neq 0 \land curTime-offT \leq resumeTime
             comC=comParamsValC <
                                {curTime →comParamsValC(max(dom(comParamsValC)))} ∧
 grd3
             pcvC=pcvParamsValC ∢
                                {curTime →pcvParamsValC(max(dom(pcvParamsValC)))} ∧
             psvC=psvParamsValC ∢
                                {curTime →psvParamsValC(max(dom(psvParamsValC)))}
THEN
 act1 : modeC≔SelfTest
 act2 : modeCP≔modeC
 act3
            comParamsValC:=comC
 act4
            pcvParamsValC≔pcvC
            psvParamsValC≔psvC
 act5
END
crash
 extended
STATUS
 ordinary
REFINES
 crash
WHEN
 grd1
            crashed=FALSE
THEN
 act1
            crashed:=TRUE
 act2 : modeG≔Off
        : modeGP≔Off
 act3
 act4
            PCV2PSV:=FALSE
        1
END
repare
 extended
STATUS
 ordinary
```

```
REFINES
 repare
ANY
 modeg
WHERE
            crashed=TRUE
 grd1
            power=FALSE v (onAC=FALSE \( \) (switchover=FALSE v batLev=0 v batFail=TRUE))
 grd2
            modeg=Off
            power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))
 grd3
            modeg=StartUp
 grd4
            modeC∈Ventilation ⇒ modeg=modeC
            power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))\( \)
            modeC∉Ventilation
 grd5 :
            modeg=StartUp
THEN
 act1 : crashed≔FALSE
       : modeG≔modeg
: modeGP≔mode
 act2
 act3
           modeGP:=modeG
END
newPatient
 extended
STATUS
 ordinary
REFINES
 newPatient
WHEN
       : modeG=Start
 grd1
 grd2 : modeC=SelfTest
THEN
 act1 : modeG≔SelfTest
 act2 : modeGP≔modeG
resumeVent ≜
 extended
STATUS
 ordinary
REFINES
 resumeVent
WHEN
 grd1 : modeG=Start
 grd2 : modeC=SelfTest
 grd3 : offT≠0 ∧ curTime-offT ≤ resumeTime
THEN
 act1 : modeG:=Menu
 act2 : modeGP≔modeG
 act3 : modeC≔VentilationOff
 act4 : modeCP≔modeC
END
runAbortSelfTest ≜
 extended
STATUS
 ordinary
REFINES
 {\tt runAbortSelfTest}
 grd1 : modeG=SelfTest
 grd2 : modeC=SelfTest
 act1 : modeG=SelfTest
 act2 : modeGP≔modeG
END
selfTestPassed ≜
                            // CONT4-4.1
 extended
STATUS
 ordinary
```

```
REFINES
 {\tt selfTestPassed}
 grd1 : modeG=SelfTest
 grd2 : modeC=SelfTest
 act1 : modeG≔Menu
 act2 : modeGP≔modeG
 act3 : modeC≔VentilationOff
act4 : modeCP≔VentilationOff
END
setParam ≜
 extended
STATUS
 ordinary
REFINES
 setParam
WHEN
 grd1 : modeG∈{Menu}uVentilation
THEN
 act1 : modeG=Settings
 act2 : modeGP≔modeG
END
settingParams ≜
 extended
STATUS
 ordinary
REFINES
 settingParams
ANY
 psvP
 pcvP
 comP
WHERE
 grd1 : modeG=Settings
 grd2 : max(dom(psvParamsValG))<curTime</pre>
 grd3 : psvP∈ psvParams→N1
 grd4 : psvParams \setminus \{RRAP, PinspAP\} \subseteq dom(psvP)
 grd5 : pcvP \in pcvParams \rightarrow N1
 grd6 : comP∈ comParams→N1
THEN
 act1 : psvParamsValG(curTime)≔psvP
 act2 : pcvParamsValG(curTime)≔pcvP
 act3 : comParamsValG(curTime)≔comP
END
saveBackAbort ≜
 extended
STATUS
 ordinary
REFINES
 saveBackAbort
ANY
 modeg
 modec
 SV
 pcvC
 psvC
 comC
 pcvG
 psvG
 comG
WHERE
           modeG=Settings ∧ modeg∈ModesG
 grd1
 grd2
       : modeg∈Ventilationu{Menu}
 grd3 : modeC≠FailSafe
 grd4 : modec∈ModesC
 grd5
           modeC∈Ventilation ⇒ modec∈Ventilation ∧ modeg=modec
 grd6
        : modeC∉Ventilation ⇒modec=modeC ∧ modeg=Menu
```

```
grd7
              sv∈
              {TRUE, FALSE}
              psvG={TRUE↔
  grd8
              psvParamsValG(max(dom(psvParamsValG))),
               FALSE⇔psvParamsValC(max(dom(psvParamsValC)))}(sv)
              psvC={TRUE→
  ard9
              psvParamsValG(max(dom(psvParamsValG))),
               FALSE⇒psvParamsValC(max(dom(psvParamsValC)))}(sv)
               pcvG={TRUE↔
  grd10
               pcvParamsValG(max(dom(pcvParamsValG))),
                FALSE⇔pcvParamsValC(max(dom(pcvParamsValC)))}(sv)
               pcvC={TRUE→
  ard11
               pcvParamsValG(max(dom(pcvParamsValG))),
                FALSE → pcvParamsValC(max(dom(pcvParamsValC)))}(sv)
                comG={TRIJF+>
  grd12
                comParamsValG(max(dom(comParamsValG))),
                FALSE⇔comParamsValC(max(dom(comParamsValC)))}(sv)
               comC={TRUE→
  grd13
                comParamsValG(max(dom(comParamsValG))),
                FALSE→comParamsValC(max(dom(comParamsValC)))}(sv)
  grd14
               PCV2PSV=TRUE ∧ modeC=PCV⇒modec=PSV
               PCV2PSV=FALSE v modeC≠PCV⇒modec=modeC
  grd15
               modec=PSV
  grd16
               dom(psvC)=psvParams Λ
               psvC(ApneaLag)≥max_insp_time_psv÷2
               modeC \neq modec \implies PCV2PSV = TRUE \land modeC = PCV \land modec = PSV
  grd17
               modec=PCV∧ sv=TRUE⇒
  grd18
               10*60*pcvC(IEPCV)÷ (pcvC(RRPCV)*(1 +pcvC(IEPCV))) >0
  grd19
               curTime∉dom(pcvParamsValC)
  grd20
               \forall c \cdot c \in cycles \Rightarrow inspBegT(c) \neq curTime
THEN
  act1
              modeG≔modeg
  act2
              modeGP:=modeG
  act3
             modeC:=modec
  act4
             modeCP:=modeC
         : psvParamsValG(curTime)≔psvG
  act5
  act6
              psvParamsValC(curTime):=psvC
  act7
              pcvParamsValG(curTime)≔pcvG
  act8
              pcvParamsValC(curTime)≔pcvC
  act9
              comParamsValG(curTime):=comG
  act10
              comParamsValC(curTime)≔comC
              PCV2PSV:=FALSE
  act11
END
startStopPCVPSV
  extended
STATUS
  ordinary
REFINES
  startStopPCVPSV
ANY
  modeg
  modec
WHERE
  grd1
              modeG∈{Menu}uVentilation
  grd2
             modeg∈Ventilationu{Menu}
  grd3
             modeG=Menu⇒modeg∈Ventilation
  grd4
              modeG∈Ventilation⇒modeg=Menu
  grd5
              modeC≠FailSafe
  grd6
              modeg \in Ventilation \implies modec = modeg
  grd7
              \textit{modeg} \not\in \textit{Ventilation} \implies \textit{modec} = \textit{VentilationOff}
              modeg=PSV
  grd8
              dom(psvParamsValC(max(dom(psvParamsValC))))=psvParams
  grd9
               (\textit{psvParamsValC}(\textit{max}(\textit{dom}(\textit{psvParamsValC}))))(\textit{ApneaLag}) \geq \textit{max}\_\textit{insp}\_\textit{time}\_\textit{psv} + 2
  ard10
               10*60*(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)÷
                  ((pcvParamsValC(max(dom(pcvParamsValC))))(RRPCV)*
```

```
(1 +(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV))) >0
  grd11
               cycles=ventilPhase~[{expEnd,expPauseEnd}]
THEN
  act1
              modeG:=modeg
              modeGP:=modeG
  act2
  act3
              modeC≔modec
              modeCP = \{TRUE \mapsto modeC, FALSE \mapsto modeCP\} (bool(modeC \in Ventilation \cup \{VentilationOff\})) \}
  act4
               inValve≔{TRUE>
  act18
                               {FALSE→FALSE, TRUE→inValve}(inValveF),
                         FALSE⇒inValve}(bool(modec∉Ventilation))
               outValve≔{TRUE>
  act19
                                {FALSE→TRUE, TRUE→outValve}(outValveF),
                          FALSE→outValve}(bool(modec∉Ventilation))
FND
moveToPSV
 extended
STATUS
 ordinary
REFINES
 moveToPSV
ANY
  modeg
  modec
WHERE
  grd1
              modeG∈ Ventilation
              modeg∈Ventilationu{Settings}
  grd2
              modeC∈Ventilation\{FailSafe}
  ard3
  grd4
              modec∈Ventilation
              modeG=PCV
  grd5
              modeC=PCV
  ard6
  grd7
              modeg=Settings
  grd8
              modec=modeC
THEN
             modeG:=modeg
  act1
              modeGP:=modeG
  act2
  act3
              modeC:=modec
         1
  act4
              modeCP:=modeC
              PCV2PSV≔TRUE
  act5
END
failExternalPower
  extended
STATUS
 ordinary
REFINES
 failExternalPower
ANY
  modeg
  modean
  modec
  modecp
  pcv2psv
  cycs
WHERE
  grd1
              onAC=TRUE
  grd2
              modeg ∈ ModesG ∧ modegp ∈ ModesG
  grd3
              \neg (power=TRUE \land crashed=FALSE) \Rightarrow modeg=Off \land modegp=Off
              power=TRUE ∧ crashed=FALSE ⇒
  grd4
                  \textit{((switchover=TRUE \land batLev>0 \land batFail=FALSE \Rightarrow modeg=modeG \land modegp=modeGP) \land }
                  (\neg(switchover=TRUE \land batLev>0 \land batFail=FALSE) \Rightarrow modeg=Off \land modegp=modeG))
  grd5
              modec ∈ ModesC ∧ modecp ∈ ModesC
              power= FALSE ⇒ modec=modeC ∧ modecp=modeCP
  grd6
               power= TRUE ⇒
  grd7
                   ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE ⇒modec=modeC ∧ modecp=modeCP) ∧
                    (\neg(switchover=TRUE \land batLev>0 \land batFail=FALSE) \Rightarrow modec=Off \land modecp=modeC))
              \neg (power=TRUE \land crashed=FALSE) \Rightarrow pcv2psv=FALSE
  grd8
```

```
power=TRUE ∧ crashed=FALSE ⇒
 grd9 :
            ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE⇒ pcv2psv=PCV2PSV) ∧
            (¬(switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)⇒ pcv2psv=FALSE))
             (switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)
 grd10
             cvcs=cvcles
             (switchover=FALSE v batLev=0 v batFail=TRUE)
 grd11
             cycs=ventilPhase~[{expEnd,expPauseEnd}]
 grd12
             cycs<u></u>cycles
THEN
 act1
            onAC:=FALSE
 act2
        : modeG≔modeg
 act3 : modeGP≔modegp
       : modeC:=modec
 act4
 act5
           modeCP:=modecp
 act6
           PCV2PSV≔pcv2psv
 act7
        : ventilPhase≔cycs⊲ventilPhase
 act8 : inspBegT≔cycs⊲inspBegT
 act9
        : inspEndT≔cycs⊲inspEndT
 act10 : cycles≔cycs
 act11
            cycleMode≔ cycs⊲cycleMode
 act12
             inspPauseBegT≔cycs⊲inspPauseBegT
 act13
            inspPauseEndT≔cycs⊲inspPauseEndT
 act14 : rmBegT≔cycs⊲rmBegT
 act15 : rmEndT≔cycs⊲rmEndT
 act16
        : expBegT≔cycs⊲expBegT
 act17
             expEndT:=cycs⊲expEndT
             expPauseBegT≔cycs⊲expPauseBegT
 act18
 act19
             expPauseEndT≔cycs⊲expPauseEndT
             inValve≔{TRUE↔
 act20
                           {FALSE→FALSE, TRUE→inValve}(inValveF),
                      FALSE→inValve}(bool(modec=Off))
             outValve≔{TRUE↔
 act21
                            {FALSE→TRUE, TRUE→outValve}(outValveF),
                       FALSE→outValve}(bool(modec=Off))
failSelfTest ≜
 extended
STATUS
 ordinary
REFINES
 failSelfTest
WHEN
 grd1 : modeC=SelfTest
THEN
 act1 : modeC≔FailSafe
 act2 : modeCP:=modeC
progress
 extended
STATUS
 ordinary
REFINES
 progress
ANY
 step
  7
 modeg
 batf
 modec
 paw
  cycs
WHERE
            step∈N1 ∧ l∈batteryRange ∧ batf∈BOOL
 grd1
 grd2
            batFail=TRUE \implies l=batLev
            onAC=TRUE ∧ batFail=FALSE ⇔ l=batLev+step
 grd3
```

```
grd4 : onAC=FALSE \land batFail=FALSE \Leftrightarrow l=max
             ({0,batLev-step})
             (onAC=TRUE v (l>0 Λ switchover=TRUE Λ batf=FALSE)) Λ
             power=TRUE A modeG=Off A
  grd5
             crashed=FALSE
             modeg=StartUp
            (l=0 v batf=TRUE v switchover=FALSE) ∧ onAC=FALSE ⇒modeg=Off
  grd6
             onAC=TRUE v
            ((l>0 Λ batLev>0) Λ switchover=TRUE Λ batf=FALSE)
  grd7
             modeg=modeG
  grd8
             modec∈{FailSafe,modeC, Off,StartUp}
            modec=FailSafe⇒modeC≠Off
  ard9
              (on AC=TRUE v (l>0 \Lambda switchover=TRUE \Lambda batf=FALSE)) \Lambda
             power=TRUE A modeC=Off
  ard10
              modec=StartUp
              (batLev>0 \land l>0)\lor switchover=FALSE \lor onAC=TRUE \lor
              power=FALSE
  grd11
              modec∈{modeC,FailSafe}
              (l=0 v batf=TRUE v switchover=FALSE) \( \lambda \) on AC=FALSE
  grd12
              modec=Off
  grd13
              step∈N1 ∧ paw∈N
              ∀c· c∈cycles ∧ ventilPhase(c)=inspBeg
  grd14
              curTime+step≤inspEndT(c)
              ∀c· c∈cycles ∧ ventilPhase(c)=inspPauseBeg
  grd15
              curTime + step \leq inspPause EndT(c)
              ∀c· c∈cycles ∧ ventilPhase(c)=rmBeg
  grd16
              curTime+step≤ rmEndT(c)
              ∀c· c∈cycles ∧ ventilPhase(c)=expBeg
  grd17
              curTime+step∈expEndT(c)
              ∀c· c∈cycles ∧ ventilPhase(c)=expPauseBeg
  ard18
              curTime+step \le expPauseEndT(c)
             modec \notin Ventilationu\{FailSafe\} \Rightarrow cycs = ventilPhase \sim [\{expEnd, expPauseEnd\}]
  grd19
         : modec∈Ventilationu{FailSafe} ⇒ cycs=cycles
  ard20
  grd21
             cycs<u></u>ccycles
THEN
            curTime:=curTime+step
  act1
            batLev≔l
  act2
            modeG≔modea
  act3
  act4
            modeGP:={TRUE→modeGP, FALSE→modeG}(bool(modeg=modeG))
  act5 : batFail≔batf
  act6 : modeC≔modec
  act7 : modeCP:={TRUE→modeCP, FALSE→modeC} (bool (modeC=modec))
 act8 : PCV2PSV={TRUE+PCV2PSV, FALSE+FALSE}(bool(
             (batLev>0 ^ l>0^ switchover=FALSE) v onAC=TRUE))
        : ventilPhase≔cycs⊲ventilPhase
  act9
  act10 : inspBegT≔cycs⊲inspBegT
  act11 : inspEndT≔cycs⊲inspEndT
  act12
         : cycles≔cycs
  act13
             cycleMode≔cycs⊲cycleMode
  act14
              inspPauseBegT≔cycs⊲inspPauseBegT
  act15
             inspPauseEndT≔cycs⊲inspPauseEndT
  act16
             rmBegT≔cycs⊲rmBegT
  act17
         : rmEndT≔cycs⊲rmEndT
  act18
         : expBegT≔cycs⊲expBegT
             expEndT≔cycs⊲expEndT
  act19
  act20
              expPauseBegT≔cycs⊲expPauseBegT
  act21
             expPauseEndT≔cycs⊲expPauseEndT
 act22 :
             inValve={TRUE→inValve,FALSE→FALSE}(bool(modec=modeC))
  act23 :
              outValve:={TRUE→outValve,FALSE→TRUE}(bool(modec=modeC))
END
```

```
switchoverFail
 extended
STATUS
 ordinary
REFINES
 switchoverFail
ANY
 modeg
 modec
 CVCS
WHERE
 grd1 : switchover=TRUE
 grd2 : onAC=FALSE \Rightarrow modeg=Off
 grd3
            onAC=TRUE⇒modeg=modeG
 grd4
            onAC=FALSE⇒modec=Off
 grd5
            onAC=TRUE⇒modec=modeC
       : modec∉Ventilationu{FailSafe} ⇒ cycs=ventilPhase~[{expEnd,expPauseEnd}]
 grd6
 grd7
       : modec∈Ventilationu{FailSafe} ⇒ cycs=cycles
 grd8
       : cycs<u>c</u>cycles
THEN
 act1
            switchover≔FALSE
 act2
           modeG≔modea
 act3
       : modeGP≔{TRUE→modeG, FALSE→modeGP}(bool(modeG≠modeg))
 act4
       : modeC≔modec
            modeCP:={TRUE→modeC, FALSE→modeCP}(bool(modeC≠modec))
 act5
 act6
            PCV2PSV:={TRUE→PCV2PSV, FALSE→FALSE}(bool(onAC=TRUE))
 act7
            ventilPhase≔cycs⊲ventilPhase
 act8
            inspBegT≔cycs⊲inspBegT
 act9
           inspEndT≔cycs⊲inspEndT
 act10
        : cycles≔cycs
 act11 : cycleMode≔cycs⊲cycleMode
 act12
             inspPauseBegT≔cycs⊲inspPauseBegT
 act13
             inspPauseEndT≔cycs⊲inspPauseEndT
        : rmBegT≔cycs⊲rmBegT
 act14
 act15 : rmEndT≔cycs⊲rmEndT
 act16 : expBegT≔cycs⊲expBegT
 act17 : expEndT≔cycs⊲expEndT
 act18
            expPauseBegT≔cycs⊲expPauseBegT
 act19
             expPauseEndT≔cycs⊲expPauseEndT
 act20
             inValve = \{TRUE \mapsto inValve, FALSE \mapsto FALSE\} (bool(modec=modeC))
 act21
             outValve:={TRUE→outValve,FALSE→TRUE}(bool(modec=modeC))
END
inspStart
 extended
STATUS
 ordinary
REFINES
 inspStart
ANY
 CV
 paw
 ve
 inspT
WHERE
 grd1
            modeC∈Ventilation ∧ inspT∈N
 grd2
           cy∈Cycles\cycles
 grd3
        : ran(ventilPhase)⊆{expEnd,expPauseEnd}
            modeC=PCV⇒inspT=curTime + 10*60*
             (pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)÷
 grd4
                   ((pcvParamsValC(max(dom(pcvParamsValC))))(RRPCV)*
                  (1 +(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)))
            modeC=PSV⇒
 grd5
            inspT>curTime ∧
            inspT≤curTime + max_insp_time_psv
 grd6
            paw∈N ∧ ve∈ N
 grd7
            pcvParamsValC≠ø ∧ psvParamsValC≠ø ∧ comParamsValC≠ø
THEN
            ventilPhase(cy)≔inspBeg
 act1
 act2
            cycles=cyclesu{cy}
 act3
            inspBegT(cy)≔curTime
```

```
inspEndT
  act4
              (cy)≔inspT
  act5
             cycleMode(cy)≔modeC
  act6
             PAW(curTime)≔paw
             VE(curTime)≔ve
  act7
             inValve={FALSE→TRUE, TRUE→inValve}(inValveF)
  act8 :
             \verb"outValve"={FALSE} \mapsto \verb"FALSE", TRUE \mapsto \verb"outValve"\} (\verb"outValve"F)
 act9
inspEnd
  extended
STATUS
 ordinary
REFINES
 inspEnd
ANY
  сy
  ve
  peakVE
WHERE
  grd1
             cy∈ cycles ∧ ventilPhase(cy)=inspBeg ∧ ve∈N
             cycleMode(cy)=PSV⇒curTime>inspBegT(cy)
  grd2
             cycleMode(cy) = PCV \Rightarrow curTime = inspEndT(cy)
  grd3
  grd4
             peakVE=VE(max(dom(VE)))
              (curTime=inspEndT(cy)) v
              (cycleMode(cy)=PSV \Rightarrow
  grd5
             peakVE>ve л
              ve<(psvParamsValC(max(dom(psvParamsValC))))(ETS)*peakVE)</pre>
  grd6
             modeC∈Ventilation
THEN
  act1
              ventilPhase(cy) = inspEnd
  act2
             VE(curTime)≔ve
            inspEndT(cy)≔curTime
  act3
END
inspPauseStart
 extended
STATUS
 ordinary
REFINES
  inspPauseStart
ANY
  CV
WHERE
             cy∈cycles ∧ ventilPhase(cy)=inspEnd
  grd1
             modeC∈Ventilation
 grd2
THEN
              ventilPhase(cy)≔inspPauseBeg
  act1
  act2
             inspPauseBegT(cy)≔curTime
  act3
             inspPauseEndT(cy)≔curTime+inspPauseMax
             inValve≔{FALSE→FALSE, TRUE→inValve}(inValveF)
 act4
inspPauseEnd
  extended
STATUS
 ordinary
REFINES
 inspPauseEnd
ANY
WHERE
             cy∈cycles ∧ ventilPhase(cy)=inspPauseBeg
  grd1
  grd2
             curTime≤inspPauseEndT(cy)
             curTime>inspPauseBegT(cy)
 grd3
             modeC∈Ventilation
 grd4
THEN
             ventilPhase(cy)≔inspPauseEnd
  act1
             inspPauseEndT(cy)≔curTime
  act2
END
rmStart ≜
```

```
extended
STATUS
 ordinary
REFINES
 rmStart
ANY
 сy
  t
WHERE
 grd1
            cy∈cycles
 grd2
           ventilPhase(cy)∈{inspEnd, inspPauseEnd}
 grd3 : t=max(\{x|x\in dom(comParamsValC) \land x \leq inspBegT(cy)\})
 grd4 :
            modeC∈Ventilation
THEN
 act1
            ventilPhase(cy)≔rmBeg
 act2
            rmBegT(cy)≔curTime
            rmEndT(cy)≔curTime+(comParamsValC(t))(timerRM)
 act3
            inValve≔{FALSE→TRUE, TRUE→inValve}(inValveF)
 act4 :
END
rmEnd
 extended
STATUS
 ordinary
REFINES
 rmEnd
ANY
 CV
WHERE
 grd1
            cy∈cycles
       : ventilPhase(cy)=rmBeg
 grd2
 grd3
            curTime=rmEndT(cy)
 grd4
            modeC∈Ventilation
THEN
 act1 : ventilPhase(cy)≔rmEnd
END
expStart
 extended
STATUS
 ordinary
REFINES
 expStart
ANY
 сy
 t
 expT
WHERE
 grd1
             cy∈cycles ∧ ventilPhase(cy)∈{inspEnd, inspPauseEnd,rmEnd}
 grd2
            t=max(\{x|x\in dom(pcvParamsValC) \land x \leq inspBegT(cy)\})
 grd3
             expT<u>c</u>N
             cycleMode(cy)=PCV \Rightarrow
 grd4
             expT={curTime +60÷((pcvParamsValC(t))(RRPCV) *
                        (1 +(pcvParamsValC(t))(IEPCV)))}
             cycleMode(cy)=PSV ⇒
             expT=
 grd5
            (curTime+(inspEndT(cy)-inspBegT(cy))÷2)
             (curTime +((psvParamsValC(t)))(ApneaLag))
 grd6
             modeC∈Ventilation
THEN
 act1
             ventilPhase(cy)≔expBeg
 act2
             expBegT(cy)≔curTime
 act3
            expEndT(cy):=expT
 act4
            inValve≔{FALSE→FALSE, TRUE→inValve}(inValveF)
            outValve≔{FALSE→TRUE, TRUE→outValve}(outValveF)
 act5
END
expEnd
 extended
STATUS
```

```
ordinary
REFINES
 expEnd
ANY
  modea
  modec
  pcv
 paw
  сy
  lastTime
  †
WHERE
  grd1
            modeG∈ Ventilation
  grd2
            modeg∈Ventilationu{Settings}
  grd3
            modeC∈Ventilation\{FailSafe}
  grd4
             modec∈Ventilation
  grd5
             pcv ∈ pcvParams→N1
            modeC=PCV ⇒ modec=PCV
  grd6
             modeC=PSV A modec=PCV
             pcv=pcvParamsValC(max(dom(pcvParamsValC))) 
  grd7
             {RRPCV→(psvParamsValC(max(dom(psvParamsValC))))(RRAP),
              PinspPCV→(psvParamsValC(max(dom(psvParamsValC))))(PinspAP),
             IEPCV→IEAP}
             modeC=PCV v modec=PSV
  grd8
             pcv=pcvParamsValC(max(dom(pcvParamsValC)))
             modec=PCV
  grd9
             10*60*pcv(IEPCV)÷ (pcv(RRPCV)*(1 +pcv(IEPCV))) >0
              modeG≠0ff⇒ modeg=modec
  grd11
  grd12
              cy∈cycles
  grd13
              ventilPhase(cy)=expBeg
  grd14
              lastTime= max(dom(PAW))
  grd15
              paw∈N
  ard16
              t=max(\{x \mid x \in dom(psvParamsValC) \land x \leq inspBegT(cy)\})
              cycleMode(cy)=PCV Λ
                  (curTime∈ expEndT(cy) v
                     (curTime<max(expEndT(cy)) Λ
                      (PAW(lastTime)<paw ∧ PAW(lastTime)-paw>(pcvParamsValC(t))(ITSPCV)))
  grd17
              ν
              cycleMode(cy)=PSV л
                   (curTime= max(expEndT(cy)) v
                    (curTime≥min(expEndT(cy)) ∧ curTime<max(expEndT(cy)) ∧ PAW(lastTime)<paw ∧ PAW
              (lastTime)-paw>(psvParamsValC(t))(ITSPSV)))
              curTime=max(expEndT(cy)) \( \text{cycleMode(cy)=PSV} \)
  grd18
              modec=PCV
              curTime≠max(expEndT(cy))∧ cycleMode(cy)=PSV
  grd19
              modec=PSV
              cycleMode(cy)=PCV
  grd20
              \Rightarrow
              modec=PCV
  grd21
              modeC∈Ventilation
THEN
  act1
            modeG:=modeg
  act2
            modeGP:=modeG
  act3
             modeC≔modec
  act4
             modeCP:=modeC
             pcvParamsValC(curTime)≔pcv
  act5
  act6
            psvParamsValC(curTime):=psvParamsValC(max(dom(psvParamsValC)))
  act7
            comParamsValC(curTime)≔comParamsValC(max(dom(comParamsValC)))
  act8
            ventilPhase(cy)≔expEnd
  act9
            PAW(curTime)≔paw
  act10
             expEndT(cy)=curTime..curTime
FND
```

```
expPauseStart ≜
 extended
STATUS
 ordinary
REFINES
 expPauseStart
ANY
 сy
WHERE
 grd1 : cy∈cycles ∧ ventilPhase(cy)=expEnd
 grd2 : modeC \in Ventilation
 grd3 : ran(ventilPhase)⊆{expEnd,expPauseEnd}
THEN
 act1 :
           ventilPhase(cy)≔expPauseBeg
 act2 :
           expPauseBegT(cy):=curTime
 act3 : expPauseEndT(cy)≔curTime+expPauseMax
 \verb"act4": inValve={FALSE} + FALSE, TRUE + inValve}(inValveF)
 act5
       : outValve={FALSE→FALSE, TRUE→outValve}(outValveF)
END
expPauseEnd
 extended
STATUS
 ordinary
REFINES
 expPauseEnd
ANY
 CV
WHERE
       : cy∈cycles ∧ ventilPhase(cy)=expPauseBeg
 grd1
 grd2 : curTime≤expPauseEndT(cy)
 grd3 : curTime>expPauseBegT(cy)
 \verb"grd4": \verb"modeC=Ventilation"
THEN
 act1
           ventilPhase(cy)≔expPauseEnd
 act2 : expPauseEndT(cy)≔curTime
END
inValveF ≜
STATUS
 ordinary
WHEN
 grd1 :
           inValveF=FALSE
THEN
 act1 : inValveF≔TRUE
outValveF
STATUS
 ordinary
WHEN
 grd1
            outValveF=FALSE
THEN
           outValveF≔TRUE
 act1 :
END
```

END

CONTEXT

Alarms

SETS

Alarms

CONSTANTS

patConnected
FI1Failure
FI2Failure
oxygenSensorFailure
switchoverFailure
conguiComFailure
inValveFailure
outsideValue
outValveFailure

AXIOMS

partition(Alarms,{conguiComFailure}, {inValveFailure},{patConnected},
axml : {FI1Failure}, {FI2Failure}, {oxygenSensorFailure},{switchoverFailure},
{outsideValue},{outValveFailure})

END

```
MACHINE
   MVLAlarms
REFINES
   Valves
SEES
   ContStates
   VentilStates
   PSVParams
   Alarms
VARIABLES
   power
   onAC
   batLev
   batFail
   switchover
   crashed
   modeG
   modeGP
   modeC
   modeCP
   PCV2PSV
   comParamsValC
   comParamsValG
   pcvParamsValC
   pcvParamsValG
   psvParamsValC
   psvParamsValG
   curTime
   offT
   cycles
   cycleMode
   ventilPhase
   inspEndT
   inspBegT
   inspPauseBegT
   inspPauseEndT
   PAW
   ۷E
   rmBegT
   rmEndT
   expBegT
   expEndT
   expPauseBegT
   expPauseEndT
   inValve
   outValve
   inValveF
   outValveF
   alarmRaised
   patientConnected
   FI1
   FI2
   oxygenSensor
   guiContCom
   inValveP
INVARIANTS
   inv1 : alarmRaised ∈ Alarms→BOOL ∧ inValveP∈BOOL
   inv2 :
              power=FALSE ⇒ alarmRaised=Alarms×{FALSE}
              alarmRaised(patConnected)
   inv3
              bool(patientConnected=TRUE \ \land \ modeC \in \{StartUp, SelfTest\})
   inv4 :
              alarmRaised(inValveFailure)=
              bool(∃c· (c∈cycles ∧
                        (ventilPhase(c)=inspBeg ^ curTime >inspBegT(c))v
                                   (ventilPhase(c)=rmBeg ^ curTime > rmBegT(c))
                        ) ∧ inValveP=FALSE)
```

```
((
                         (ventilPhase(c)=expPauseBeg ^ curTime >expPauseBegT(c))
                        (ventilPhase(c)=expBeg ^ curTime >expBegT(c))
                         (ventilPhase(c)=inspPauseBeg \( \text{curTime >inspPauseBegT(c)} \)
                       ∧ inValveP=TRUE))
                  )
              )
              alarmRaised(inValveFailure)=TRUE
   inv5
              modeC=FailSafe
              oxygenSensor∈ B00L ∧ FI1∈ B00L ∧ FI2∈ B00L
   inv6
              oxygenSensor=FALSE ∧ modeC∉{Off, StartUp,SelfTest}⇔alarmRaised(oxygenSensorFailure)=TRUE
   inv7
   inv8
              \verb|switchover=FALSE| \land \verb|modeC@{Off}, StartUp, SelfTest| \Leftrightarrow \verb|alarmRaised(switchoverFailure)=TRUE| \\
             FI1=FALSE ∧ modeC∉{Off, StartUp,SelfTest} ⇔alarmRaised(FI1Failure)=TRUE
   inv9
   inv10
              FI2=FALSE ∧ modeC∉{Off, StartUp, SelfTest} ⇔alarmRaised(FI2Failure)=TRUE
                   modeC∉{Off,StartUp, FailSafe} v
   inv14
                   (modeCP≠StartUp ∧ modeC=FailSafe)
               ⇒comParamsValC≠ø
               (
                   modeC∉{Off,StartUp, FailSafe} v
                   (modeCP≠StartUp ∧ modeC=FailSafe)
               )
               alarmRaised(outsideValue)=bool(
               (\exists p \cdot p \in comParams \land
   inv11
                (comParamsValC(max(dom(comParamsValC))))(p)∉domcomParams(p))
               (∃ p· p ∈ pcvParams \Lambda
                (pcvParamsValC(max(dom(pcvParamsValC))))(p)∉dompcvParams(p))
               (\exists p \cdot p \in dom((psvParamsValC(max(dom(psvParamsValC)))))) \land
                (psvParamsValC(max(dom(psvParamsValC))))(p)∉dompsvParams(p))
   inv12
               guiContCom∈B00L
               guiContCom=FALSE ∧ modeC∉{Off, StartUp, SelfTest}
   inv13
               alarmRaised(conguiComFailure)=TRUE
EVENTS
   INITIALISATION
     extended
   STATUS
     ordinary
   BEGIN
     act1
                power=FALSE
               batLev:∈batteryRange
     act2 :
     act3
           1
               onAC:=TRUE
     act4
                switchover≔TRUE
               crashed:=FALSE
     act5
     act6
           : modeG=Off
     act7 : modeGP≔Off
     act8
               curTime:=0
            : batFail:∈B00L
     act9
     act10
                 modeC≔Off
                modeCP:=Off
     act11
     act12
                comParamsValC≔ø
     act13 : comParamsValG≔ø
     act14 : pcvParamsValC≔ø
     act15
            : pcvParamsValG≔ø
     act16
                psvParamsValG=ø
     act17
                 psvParamsValC≔ø
     act18 : offT≔0
     act19 : PCV2PSV=FALSE
     act20 : ventilPhase≔ø
     act21
                 inspBeqT≔ø
     act22
                 inspEndT≔ø
     act23 :
                 cycles≔ø
```

```
act24 : cycleMode≔ø
 act25
            PAW≔ø
 act26
             VE≔ø
 act27
             inspPauseBegT≔ø
 act28
            inspPauseEndT≔ø
 act29 : rmBegT≔ø
 act30 : rmEndT≔ø
 act31 : expBegT≔ø
 act32 :
act33 :
            expEndT:=ø
            expPauseBegT≔ø
 act34 : expPauseEndT≔ø
 act35 : inValve=FALSE
 act36 : outValve≔TRUE
 act37 : inValveF≔FALSE
 act38
            outValveF≔FALSE
 act39
             alarmRaised≔Alarms×{FALSE}
 act40 : patientConnected≔FALSE
 act41 : oxygenSensor≔TRUE
 act42 : FI1≔TRUE
 act43 : FI2≔TRUE
         : guiContCom:∈B00L
 act44
 act45
             inValveP≔FALSE
END
powerON ≜
                    // CONT.2
 extended
STATUS
 ordinary
REFINES
 powerON
WHEN
 grd1 : power=FALSE
 grd2 : onAC=TRUE v (switchover=TRUE \( \text{ batLev} > 0 \( \text{ \( \text{hatFail} = FALSE \) \)
THEN
 act1 : power≔TRUE
 act2 : modeG≔{FALSE→StartUp, TRUE→Off}(crashed)
       : modeGP≔modeG
 act3
 act4
           modeC≔StartUp
 act5 :
           modeCP:=modeC
            alarmRaised≔alarmRaised∢{
 \verb"act6": patConnected" \\ \texttt{patientConnected}
END
powerOff
 extended
STATUS
 ordinary
REFINES
 powerOff
ANY
 val
WHERE
 grd1
            power=TRUE
            comParamsValC=@ v comParamsValC(max(dom(comParamsValC)))=defcomParams
 grd2 :
            val=0
            comParamsValC≠ø ∧
            comParamsValC(max(dom(comParamsValC)))≠defcomParams
 grd3 :
            val=curTime
THEN
 act1 : power≔FALSE
 act2 : modeG≔Off
        : modeGP≔modeG
 act3
 act4
            modeC≔Off
 act5
           modeCP:=modeC
 act6
       : offT≔val
 act7 : PCV2PSV≔FALSE
           ventilPhase≔ventilPhase⊳{expEnd,expPauseEnd}
 act8
       : inspBegT≔ventilPhase~[{expEnd,expPauseEnd}]⊲inspBegT
 act9
```

```
inspEndT:=ventilPhase~[{expEnd, expPauseEnd}]
 act10
              ⊲inspEndT
 act11
              cycles=ventilPhase~[{expEnd,expPauseEnd}]
 act12
              cycleMode≔ ventilPhase~[{expEnd,expPauseEnd}]⊲cycleMode
              inspPauseBegT≔ventilPhase~[{expEnd,expPauseEnd}] dinspPauseBegT
 act13
 act14
              inspPauseEndT = ventilPhase \sim [\{expEnd, expPauseEnd\}] \lor inspPauseEndT
              rmBegT≔ventilPhase~[{expEnd,expPauseEnd}]⊲rmBegT
 act15
 act16
              rmEndT≔ventilPhase~[{expEnd,expPauseEnd}]⊲rmEndT
              expBegT = ventilPhase \sim [\{expEnd, expPauseEnd\}] \triangleleft expBegT
 act17
 act18
              expEndT:=ventilPhase~[{expEnd,expPauseEnd}] dexpEndT
              act19
 act20
              expPauseEndT = ventilPhase \sim [\{expEnd, expPauseEnd\}] \triangleleft expPauseEndT
 act21
              inValve=FALSE
 act22
              outValve≔TRUE
 act23
              alarmRaised:=Alarms×{FALSE}
END
startUpEndedGui ≜
 extended
STATUS
 ordinary
REFINES
 {\it startUpEndedGui}
ANY
 modeg
 pcvG
 comG
 psvG
WHERE
             modeG=StartUp
 ard1
             modeg∈{Start}uVentilation
 grd2
             \textit{modeC} \in \textit{Ventilation} \implies \textit{modeg} = \textit{modeC}
 ard3
             modeC∉Ventilation ⇒ modeg=Start
 ard4
             modeC∈Ventilation v (offT≠0 ∧ curTime-offT ≤ resumeTime)
                comG=comParamsValG ←
                                 {curTime →comParamsValC(max(dom(comParamsValC)))} ∧
 grd5
                pcvG=pcvParamsValG <
                                 {curTime →pcvParamsValC(max(dom(pcvParamsValC)))} ∧
                psvG=psvParamsValG ∢
                                 {curTime →psvParamsValC(max(dom(psvParamsValC)))}
             modeC∉Ventilation ∧ (offT=0 v curTime-offT > resumeTime)
                comG=comParamsValG ◆
                                 {curTime →defcomParams} ∧
 ard6
                pcvG=pcvParamsValG ◆
                                 {curTime ⇔defpcvParams} ∧
                psvG=psvParamsValG ∢
                                 {curTime →defpsvParams}
 grd7
             guiContCom=TRUE
THEN
             modeG≔modea
 act1
             modeGP:=modeG
 act2
             comParamsValG:=comG
 act3
             pcvParamsValG≔pcvG
 act4
 act5
             psvParamsValG≔psvG
END
startUpEndedCont =
 extended
STATUS
 ordinary
REFINES
 {\it startUpEndedCont}
ANY
 comC
 pcvC
 psvC
 outs
WHERE
 grd1
             modeC=StartUp
 ard2
             offT=0 v curTime-offT > resumeTime
```

```
comC=comParamsValC <
                                  {curTime →defcomParams} ∧
              pcvC=pcvParamsValC
                                  {curTime →defpcvParams} ∧
              psvC=psvParamsValC ◆
                                  {curTime →defpsvParams}
              offT≠0 ∧ curTime-offT ≤ resumeTime
              comC=comParamsValC ◆
                                  {curTime →comParamsValC(max(dom(comParamsValC)))} ∧
  grd3
              pcvC=pcvParamsValC <
                                  {curTime →pcvParamsValC(max(dom(pcvParamsValC)))} ∧
              psvC=psvParamsValC ∢
                                  {curTime →psvParamsValC(max(dom(psvParamsValC)))}
             oxygenSensor=TRUE \wedge FI1=TRUE \wedge FI2=TRUE \wedge guiContCom=TRUE
  grd4
              \land inValve=FALSE \land outValve=TRUE
             offT≠0 ∧ curTime-offT ≤ resumeTime
             outs=bool(
              (\exists p \cdot p \in dom(comParamsValC(max(dom(comParamsValC)))))
               ^ (comParamsValC(max(dom(comParamsValC))))(p)∉domcomParams(p))
  grd5
              (\exists p \cdot p \in dom(pcvParamsValC(max(dom(pcvParamsValC)))))
               ∧ (pcvParamsValC(max(dom(pcvParamsValC))))(p)∉dompcvParams(p))
               (\exists p \cdot p \in dom(psvParamsValC(max(dom(psvParamsValC)))))
                            ∧ (psvParamsValC(max(dom(psvParamsValC))))(p)∉dompsvParams(p))
             )
             offT=0 v curTime-offT > resumeTime
 grd6
             outs=FALSE
THEN
  act1
             modeC:=SelfTest
             modeCP:=modeC
  act2
  act3
             comParamsValC≔comC
             pcvParamsValC≔pcvC
  act4
  act5
             psvParamsValC≔psvC
         ...
  act6
             alarmRaised(outsideValue)≔outs
END
crash
 extended
STATUS
 ordinary
REFINES
 crash
WHEN
             crashed=FALSE
 grd1
         1
THEN
             crashed:=TRUE
  act1
  act2
         : modeG=Off
  act3
        : modeGP≔Off
             PCV2PSV:=FALSE
  act4
repare
 extended
STATUS
 ordinary
REFINES
  repare
ANY
 modeg
WHERE
  grd1
              crashed=TRUE
             power=FALSE v (onAC=FALSE \( \) (switchover=FALSE v batLev=0 v batFail=TRUE))
  grd2
              modeg=Off
  grd3
             power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))
              \Rightarrow
```

```
modeg=StartUp
 grd4
            modeC \in Ventilation \Rightarrow modeg = modeC
             power=TRUE \( \) (onAC=TRUE \( \) (switchover=TRUE \( \) batLev>0 \( \) batFail=FALSE))\( \)
            modeC∉Ventilation
  grd5
             modeg=StartUp
THEN
            crashed:=FALSE
  act1
  act2
            modeG≔modeg
 act3
            modeGP:=modeG
END
newPatient
 extended
STATUS
 ordinary
REFINES
 newPatient
WHEN
 grd1 :
            modeG=Start
 grd2
            modeC=SelfTest
 act1 : modeG≔SelfTest
 act2 : modeGP≔modeG
END
resumeVent
 extended
STATUS
 ordinary
REFINES
 resumeVent
WHEN
  grd1 : modeG=Start
       : modeC=SelfTest
  grd2
  grd3
            offT≠0 ∧ curTime-offT ≤ resumeTime
THEN
            modeG:=Menu
  act1
  act2
        : modeGP≔modeG
  act3
        : modeC≔VentilationOff
  act4
            modeCP:=modeC
             alarmRaised≔alarmRaised∢(
               {patConnected⊬FALSE}u
               {oxygenSensorFailure→bool(oxygenSensor=FALSE),
 act5
                switchoverFailure⇔bool(switchover=FALSE),
                FI1Failure⇒bool(FI1=FALSE),
                FI2Failure⇔bool(FI2=FALSE).
                \verb|conguiComFailure+bool(guiContCom=FALSE)||)
END
runAbortSelfTest ≜
 extended
STATUS
 ordinary
REFINES
  {\tt runAbortSelfTest}
WHEN
 grd1
            modeG=SelfTest
            modeC=SelfTest
  grd2
            ¬(switchover=TRUE \land outValve=TRUE \land FI1=TRUE \land FI2=TRUE \land
 grd3
             oxygenSensor= TRUE)
THEN
  act1 : modeG≔SelfTest
  act2 : modeGP≔modeG
END
selfTestPassed ≜
                             // CONT4-4.1
 extended
STATUS
 ordinary
REFINES
  selfTestPassed
```

```
WHEN
 grd1 : modeG=SelfTest
 grd2 : modeC=SelfTest
            switchover=TRUE \( \text{ outValve=TRUE \( \Lambda \) FI1=TRUE \( \Lambda \) FI2=TRUE \( \Lambda \)
 grd3 :
            oxygenSensor= TRUE \land guiContCom=TRUE
THEN
 act1
            modeG≔Menu
 act2 : modeGP≔modeG
       : modeC≔VentilationOff
 act3
 act4
            modeCP≔VentilationOff
 act5 : alarmRaised(patConnected):=FALSE
END
setParam
 extended
STATUS
 ordinary
REFINES
 setParam
WHEN
 grd1
            modeG∈{Menu}uVentilation
THEN
 act1 : modeG=Settings
 act2 : modeGP≔modeG
END
settingParams
 extended
STATUS
 ordinary
REFINES
 settingParams
ANY
 psvP
 pcvP
 comP
WHERE
            modeG=Settings
 grd1
           max(dom(psvParamsValG))<curTime
 grd2 :
 grd3 : psvP∈ psvParams→N1
 grd4 : psvParams \setminus \{RRAP, PinspAP\} \subseteq dom(psvP)
 grd5
       : pcvP∈ pcvParams→N1
 grd6
        : comP∈ comParams→N1
THEN
 act1 : psvParamsValG(curTime)≔psvP
 act2 : pcvParamsValG(curTime)≔pcvP
 act3 : comParamsValG(curTime)≔comP
END
saveBackAbort ≜
 extended
STATUS
 ordinary
REFINES
 saveBackAbort
ANY
 modeg
 modec
 pcvC
 psvC
 comC
 pcvG
 psvG
 comG
 outs
WHERE
 grd1
            modeG=Settings ∧ modeg∈ModesG
 grd2 :
            modeg∈Ventilationu{Menu}
 grd3 :
            modeC≠FailSafe
 grd4 : modec∈ModesC
```

```
grd5
             modeC \in Ventilation \Rightarrow modec \in Ventilation \land modeg = modec
 grd6
             modeC∉Ventilation ⇒modec=modeC ∧ modeg=Menu
 grd7
             sv∈{TRUE, FALSE}
             psvG={TRUE↔
 grd8
             psvParamsValG(max(dom(psvParamsValG))),
              FALSE⇔psvParamsValC(max(dom(psvParamsValC)))}(sv)
             psvC={TRUE↔
 ard9
             psvParamsValG(max(dom(psvParamsValG))),
               FALSE⇔psvParamsValC(max(dom(psvParamsValC)))}(sv)
               pcvG={TRUE↔
 grd10
               pcvParamsValG(max(dom(pcvParamsValG))),
               FALSE⇒pcvParamsValC(max(dom(pcvParamsValC)))}(sv)
              pcvC={TRUE↔
 ard11
              pcvParamsValG(max(dom(pcvParamsValG))),
               FALSE⇔pcvParamsValC(max(dom(pcvParamsValC)))}(sv)
               comG={TRUE↔
 ard12
               comParamsValG(max(dom(comParamsValG))),
               FALSE⇔comParamsValC(max(dom(comParamsValC)))}(sv)
               comC={TRUE→
 ard13
               comParamsValG(max(dom(comParamsValG))),
               FALSE→comParamsValC(max(dom(comParamsValC)))}(sv)
 grd14
               PCV2PSV=TRUE \land modeC=PCV \Rightarrow modec=PSV
              PCV2PSV=FALSE v modeC≠PCV⇒modec=modeC
 grd15
               modec=PSV
 grd16
               dom(psvC)=psvParams Λ
               psvC(ApneaLag)≥max_insp_time_psv÷2
              modeC \neq modec \implies PCV2PSV = TRUE \land modeC = PCV \land modec = PSV
 grd17
               modec=PCV∧ sv=TRUE⇒
 grd18
               10*60*pcvC(IEPCV)÷ (pcvC(RRPCV)*(1 +pcvC(IEPCV))) >0
 grd19
               curTime∉dom(pcvParamsValC)
 ard20
               ∀c·c∈cycles⇒ inspBegT(c)≠curTime
               sv=TRUE
               outs=bool(
               (\exists p \cdot p \in comParams \land comC(p) \notin domcomParams(p))
 grd21
               (\exists p \cdot p \in pcvParams \land pcvC(p) \notin dompcvParams(p))
                (\exists p \cdot p \in dom(psvC) \land psvC(p) \notin dompsvParams(p))
               )
               sv=FALSE
 grd22
              outs=alarmRaised(outsideValue)
THEN
 act1
             modeG:=modeg
 act2
             modeGP:=modeG
 act3
             modeC:=modec
         ...
 act4
             modeCP:=modeC
 act5
             psvParamsValG(curTime)≔psvG
 act6
            psvParamsValC(curTime)≔psvC
 act7
        : pcvParamsValG(curTime)≔pcvG
 act8
            pcvParamsValC(curTime)≔pcvC
 act9
             comParamsValG(curTime)≔comG
 act10
              comParamsValC(curTime) = comC
          1
 act11
              PCV2PSV:=FALSE
              alarmRaised(outsideValue)≔outs
 act12
END
startStopPCVPSV
 extended
STATUS
 ordinary
REFINES
 startStopPCVPSV
ANY
 modeg
 modec
WHERE
             modeG∈{Menu}uVentilation
 grd1
```

```
grd2
        : modeg∈Ventilationu
             {Menu}
  grd3
             modeG=Menu⇒modeg∈Ventilation
  grd4
             modeG∈Ventilation⇒modeg=Menu
  grd5
             modeC≠FailSafe
  grd6
             modeg∈Ventilation ⇒ modec=modeg
             \textit{modeg} \textit{\notin} \textit{Ventilation} \, \Rightarrow \, \textit{modec} \textit{=} \textit{Ventilation0ff}
  grd7
             modeg=PSV
  grd8
             dom(psvParamsValC(max(dom(psvParamsValC))))=psvParams
             modeg=PSV
  grd9
             (psvParamsValC(max(dom(psvParamsValC))))(ApneaLag) \ge max_insp_time_psv \div 2
              modec=PCV⇒
              10*60*(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)÷
  ard10
                ((pcvParamsValC(max(dom(pcvParamsValC))))(RRPCV)*
               (1 +(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV))) >0
  grd11
              cycles=ventilPhase~[{expEnd,expPauseEnd}]
THEN
  act1
             modeG≔modeg
  act2
             modeGP:=modeG
  act3
             modeC:=modec
  act4
            modeCP≔{TRUE→modeC, FALSE→modeCP}(bool(modeC∈Ventilation υ{VentilationOff}))
              inValve≔{TRUE
  act18
                             {FALSE→FALSE, TRUE→inValve}(inValveF),
                        FALSE⇔inValve}(bool(modec∉Ventilation))
              outValve≔{TRUE↔
  act19
                              {FALSE→TRUE, TRUE→outValve}(outValveF),
                         FALSE→outValve}(bool(modec∉Ventilation))
END
moveToPSV
 extended
STATUS
 ordinary
REFINES
 moveToPSV
ANY
 modeg
  modec
WHERE
  grd1
            modeG∈ Ventilation
  grd2
             modeg∈Ventilationu{Settings}
            modeC∈Ventilation\{FailSafe}
  grd3
  grd4
            modec∈Ventilation
  grd5
         : modeG=PCV
  grd6
            modeC=PCV
  grd7
             modeg=Settings
  grd8
             modec=modeC
THEN
  act1
             modeG:=modeg
  act2 : modeGP≔modeG
  act3
        : modeC≔modec
  act4
         : modeCP:=modeC
  act5
            PCV2PSV≔TRUE
END
failExternalPower ≜
 extended
STATUS
 ordinary
REFINES
  failExternalPower
ANY
 modeg
  modegp
  modec
  modecp
  pcv2psv
  cvcs
WHERE
```

```
grd1
            onAC=TRUE
 grd2
             modeg ∈ ModesG ∧ modegp ∈ ModesG
             \neg (power=TRUE \land crashed=FALSE) \Rightarrow modeg=Off \land modegp=Off
 grd3
             power=TRUE ∧ crashed=FALSE ⇒
 grd4
                 ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE ⇒ modeg=modeG ∧ modegp=modeGP) ∧
                 (\neg(switchover=TRUE \land batLev>0 \land batFail=FALSE) \Rightarrow modeg=0ff \land modegp=modeG))
             modec ∈ ModesC ∧ modecp ∈ ModesC
 grd5
 ard6
             power= FALSE ⇒ modec=modeC ∧ modecp=modeCP
              power= TRUE ⇒
 grd7
                  ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE ⇒modec=modeC ∧ modecp=modeCP) ∧
                   (\neg(switchover=TRUE \land batLev>0 \land batFail=FALSE) \Rightarrow modec=Off \land modecp=modeC))
 grd8
             \neg (power=TRUE \land crashed=FALSE) \Rightarrow pcv2psv=FALSE
             power=TRUE ∧ crashed=FALSE ⇒
 grd9
             ((switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE⇒ pcv2psv=PCV2PSV) ∧
             (¬(switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)⇒ pcv2psv=FALSE))
              (switchover=TRUE ∧ batLev>0 ∧ batFail=FALSE)
 grd10
              cycs=cycles
              (switchover=FALSE v batLev=0 v batFail=TRUE)
 grd11
              cycs=ventilPhase~[{expEnd,expPauseEnd}]
 grd12
              cycs<u></u>cycles
THEN
             onAC:=FALSE
 act1
 act2
             modeG:=modeg
        1
 act3
             modeGP:=modegp
 act4
             modeC:=modec
 act5
             modeCP:=modecp
         1
 act6
             PCV2PSV:=pcv2psv
 act7
             ventilPhase≔cycs⊲ventilPhase
 act8
             inspBegT≔cycs⊲inspBegT
 act9
        : inspEndT≔cycs⊲inspEndT
 act10
         : cycles≔cycs
              cycleMode≔ cycs⊲cycleMode
 act11
 act12
              inspPauseBegT≔cycs⊲inspPauseBegT
             inspPauseEndT≔cycs⊲inspPauseEndT
 act13
 act14
         : rmBegT≔cycs⊲rmBegT
 act15
         : rmEndT≔cycs⊲rmEndT
 act16
              expBegT:=cycs⊲expBegT
 act17
              expEndT:=cycs⊲expEndT
 act18
              expPauseBegT:=cycs⊲expPauseBegT
 act19
              expPauseEndT = cycs \triangleleft expPauseEndT
              inValve≔{TRUE→
 act20
                             {FALSE→FALSE, TRUE→inValve}(inValveF),
                        FALSE⇔inValve}(bool(modec=Off))
              outValve≔{TRUE
 act21
                              {FALSE→TRUE, TRUE→outValve}(outValveF),
                         FALSE \mapsto outValve (bool (modec=0ff))
              alarmRaised≔{TRUE→alarmRaised, FALSE→Alarms×{FALSE}}(bool(modec≠Off))
 act22
END
failSelfTest
 extended
STATUS
 ordinary
REFINES
 failSelfTest
WHEN
 grd1
             modeC=SelfTest
             FI1=FALSE v FI2=FALSE v oxygenSensor=FALSE v switchover=FALSE v
 grd2
             guiContCom=FALSE v inValve=TRUE v outValve=FALSE
THEN
 act1
             modeC=FailSafe
 act2
             modeCP:=modeC
 act3
             alarmRaised≔ alarmRaised⊲
```

```
{patConnected→FALSE, FI1Failure→ bool(FI1=FALSE), FI2Failure→bool(FI2=FALSE),
              oxygenSensorFailure→bool(oxygenSensor=FALSE),
              switchoverFailure→bool(switchover=FALSE),
              conguiComFailure →bool(guiContCom=FALSE)}
END
progress
 extended
STATUS
 ordinary
REFINES
 progress
ANY
  step
  Z
  modea
  batf
  modec
  paw
  cycs
  alarmInV
  outside
WHERE
  grd1
             step∈N1 ∧ l∈batteryRange ∧ batf∈B00L
             batFail=TRUE ⇒ l=batLev
  grd2
             onAC=TRUE ∧ batFail=FALSE ⇔ l=batLev+step
  grd3
  grd4
             onAC=FALSE \land batFail=FALSE \Leftrightarrow l=max(\{0,batLev-step\})
             (onAC=TRUE v (l>0 Λ switchover=TRUE Λ batf=FALSE)) Λ
             power=TRUE A modeG=Off A
  grd5
             crashed=FALSE
             modeg=StartUp
  ard6
             (l=0 v batf=TRUE v switchover=FALSE) ∧ onAC=FALSE ⇒modeg=Off
             onAC=TRUE v
             ((l>0 Λ batLev>0) Λ switchover=TRUE Λ batf=FALSE)
  grd7
             modeg=modeG
            modec∈{FailSafe,modeC, Off,StartUp}
  ard8
         .
            modec=FailSafe⇒modeC≠Off
  grd9
              (onAC=TRUE v (l>0 Λ switchover=TRUE Λ batf=FALSE)) Λ
              power=TRUE ^ modeC=Off
  grd10
              modec=StartUp
              (batLev>0 ^ l>0)v switchover=FALSE v onAC=TRUE v
              power=FALSE
  grd11
              modec∈{modeC,FailSafe}
              (l=0 v batf=TRUE v switchover=FALSE) \( \lambda \) on AC=FALSE
  grd12
              \Rightarrow
              modec=Off
  grd13
              step∈N1 ∧ paw∈N
              \forall c \cdot c \in cycles \land ventilPhase(c) = inspBeg
  grd14
              curTime+step≤inspEndT(c)
              ∀c· c∈cycles ∧ ventilPhase(c)=inspPauseBeg
  grd15
              curTime+step≤inspPauseEndT(c)
              ∀c· c∈cycles ∧ ventilPhase(c)=rmBeg
  grd16
              curTime+step≤ rmEndT(c)
              ∀c· c∈cycles ∧ ventilPhase(c)=expBeg
  grd17
              curTime+step∈expEndT(c)
              ∀c· c∈cycles ∧ ventilPhase(c)=expPauseBeg
  grd18
              curTime+step \le expPauseEndT(c)
  grd19
              modec∉Ventilationu{FailSafe} ⇒ cycs=ventilPhase~[{expEnd,expPauseEnd}]
  grd20
              modec∈Ventilationu{FailSafe} ⇒ cycs=cycles
  grd21
              cycs⊆cycles
  grd22
              modec∉{Off, StartUp,SelfTest}
              alarmTnV=
```

```
bool(∃c· (c∈cycles ∧
                         (ventilPhase(c)=inspBeg ^ curTime+step >inspBegT(c))v
                                   (ventilPhase(c)=rmBeg ^ curTime+step > rmBegT(c))
                          inValve=FALSE)
                         v
                         ((
                         (ventilPhase(c)=expPauseBeg ^ curTime+step >expPauseBegT(c))
                        (ventilPhase(c)=expBeg ^ curTime+step >expBegT(c))
                         (ventilPhase(c)=inspPauseBeg ^ curTime+step >inspPauseBegT(c))
                       ∧ inValve=TRUE))
              modec∈{Off, StartUp,SelfTest}
 grd23
              alarmInV=FALSE
              alarmInV=TRUE
 ard24
              ¬(oxygenSensor=TRUE ∧ FI1=TRUE ∧ FI2=TRUE ∧ guiContCom=TRUE)
 grd25
              modec=FailSafe
              modec≠0ff
 grd26
              outside=alarmRaised(outsideValue)
              modec=Off
 ard27
              outside=FALSE
THEN
            curTime≔curTime+step
 act1
 act2
             batLev=l
 act3
             modeG:=modeg
 act4
            modeGP:={TRUE→modeGP, FALSE→modeG}(bool(modeg=modeG))
 act5
            batFail≔batf
 act6
        : modeC≔modec
        : modeCP:={TRUE→modeCP,FALSE→modeC}(bool(modeC=modec))
 act7
             PCV2PSV:={TRUE→PCV2PSV, FALSE→FALSE}(bool(
 act8
             (batLev>0 \( l > 0 \( \) switchover=FALSE) \( v \) onAC=TRUE))
 act9
             ventilPhase≔cycs⊲ventilPhase
 act10
            inspBegT≔cycs⊲inspBegT
 act11
         : inspEndT≔cycs⊲inspEndT
 act12
         : cycles≔cycs
 act13
              cycleMode≔cycs⊲cycleMode
 act14
              inspPauseBegT≔cycs⊲inspPauseBegT
 act15
              inspPauseEndT≔cycs⊲inspPauseEndT
 act16
             rmBegT≔cycs⊲rmBegT
 act17
         : rmEndT≔cycs⊲rmEndT
 act18
         : expBegT≔cycs⊲expBegT
 act19
              expEndT≔cvcs⊲expEndT
 act20
              expPauseBegT≔cycs⊲expPauseBegT
 act21
              expPauseEndT≔cycs⊲expPauseEndT
 act22
              inValve:={TRUE→inValve,FALSE→FALSE}(bool(modec=modeC))
 act23
              outValve={TRUE→outValve, FALSE→TRUE}(bool(modec=modeC))
              alarmRaised≔alarmRaised∢{patConnected→
                  bool(patientConnected=TRUE \ \land \ modec \in \{StartUp, SelfTest\})\,,
              inValveFailure → alarmInV,
              oxygenSensorFailure ⇒bool(
              oxygenSensor=FALSE ∧ modec∉{Off, StartUp,SelfTest}),
              switchoverFailure⇒bool(switchover=FALSE ∧ modec∉{Off, StartUp,SelfTest}),
FI1Failure⇒bool(FI1=FALSE ∧ modec∉{Off, StartUp,SelfTest}),
 act24
              FI2Failure→bool(FI2=FALSE ∧ modec∉{Off, StartUp,SelfTest}),
              outsideValue → outside,
              conguiComFailure⇒bool(guiContCom=FALSE ∧
                                       modec∉{Off, StartUp,SelfTest})
              inValveP≔inValve
 act25
END
```

```
switchoverFail
 extended
STATUS
 ordinary
REFINES
 switchoverFail
 modea
 modec
  cycs
WHERE
 grd1
            switchover=TRUE
 grd2
            onAC=FALSE⇒modeg=Off
            onAC=TRUE⇒modeg=modeG
 grd3
 grd4
            onAC=FALSE⇒modec=Off
 grd5
            onAC=TRUE⇒modec=modeC
        : modec∉Ventilationu{FailSafe} ⇒ cycs=ventilPhase~[{expEnd,expPauseEnd}]
 grd6
 grd7
            modec∈Ventilationu{FailSafe} ⇒ cycs=cycles
        1
 grd8
            cycs<u></u>cycles
THEN
 act1
            switchover:=FALSE
 act2
            modeG≔modeg
            modeGP≔{TRUE→modeG, FALSE→modeGP}(bool(modeG≠modeg))
 act3
 act4
            modeC:=modec
 act5
            modeCP:={TRUE→modeC, FALSE→modeCP} (bool (modeC≠modec))
            PCV2PSV:={TRUE→PCV2PSV, FALSE→FALSE}(bool(onAC=TRUE))
 act6
 act7
            ventilPhase≔cycs⊲ventilPhase
 act8
            inspBegT≔cycs⊲inspBegT
 act9
        : inspEndT≔cycs⊲inspEndT
 act10
         : cycles≔cycs
 act11
             cycleMode≔cycs⊲cycleMode
 act12
            inspPauseBegT≔cycs⊲inspPauseBegT
 act13 : inspPauseEndT≔cycs⊲inspPauseEndT
 act14 : rmBegT≔cycs⊲rmBegT
 act15
         : rmEndT≔cycs⊲rmEndT
             expBegT≔cycs⊲expBegT
 act16
 act17
              expEndT≔cycs⊲expEndT
 act18
              expPauseBegT≔cycs⊲expPauseBegT
 act19
              expPauseEndT≔cycs⊲expPauseEndT
 act20
              inValve = \{TRUE \Rightarrow inValve, FALSE \Rightarrow FALSE\} (bool(modec=modeC))
 act21
              outValve={TRUE→outValve, FALSE→TRUE} (bool (modec=modeC) )
              alarmRaised≔{FALSE→Alarms×{FALSE},
                TRUE⇒alarmRaised⊲
 act22
                          {switchoverFailure→bool(modeC∉{Off, StartUp,SelfTest})}
                }(onAC)
END
inspStart
 extended
STATUS
 ordinary
REFINES
 inspStart
ANY
 CV
 paw
  ve
 inspT
WHERE
 grd1
            modeC∈Ventilation ∧ inspT∈N
 grd2
            cv∈Cvcles\cvcles
 grd3
            ran(ventilPhase)⊆{expEnd,expPauseEnd}
            modeC=PCV⇒inspT=curTime + 10*60*
              (pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)÷
 grd4
                    ((pcvParamsValC(max(dom(pcvParamsValC))))(RRPCV)*
                  (1 +(pcvParamsValC(max(dom(pcvParamsValC))))(IEPCV)))
            modeC=PSV⇒
 grd5
            inspT>curTime A
             inspT≤curTime + max_insp_time_psv
            paw∈N ∧ ve∈ N
 grd6
```

```
grd7 :
            pcvParamsValC≠ø ∧ psvParamsValC≠ø ∧ comParamsValC≠ø
THEN
 act1
            ventilPhase(cy)≔inspBeg
 act2
            cycles≔cyclesu{cy}
            inspBegT(cy)≔curTime
 act3
            inspEndT(cy)≔inspT
 act4
 act5
            cycleMode(cy)≔modeC
 act6
            PAW(curTime)≔paw
 act7
            VE(curTime)≔ve
            inValve≔{FALSE+TRUE, TRUE+inValve}(inValveF)
 act8
 act9
        : outValve≔{FALSE→FALSE, TRUE→outValve}(outValveF)
END
inspEnd
 extended
STATUS
 ordinary
REFINES
 inspEnd
ANY
 сy
 ve
 peakVE
WHERE
 grd1
            cy∈ cycles ∧ ventilPhase(cy)=inspBeg ∧ ve∈N
 grd2
            cycleMode(cy)=PSV⇒curTime>inspBegT(cy)
            cycleMode(cy)=PCV ⇒curTime=inspEndT(cy)
 grd3
 grd4
            peakVE=VE(max(dom(VE)))
            (curTime=inspEndT(cy)) v
            (cycleMode(cy)=PSV \Rightarrow
 grd5
            peakVE>ve A
            ve<(psvParamsValC(max(dom(psvParamsValC))))(ETS)*peakVE)</pre>
 grd6
            modeC∈Ventilation
 act1 : ventilPhase(cy)≔inspEnd
 act2 : VE(curTime)≔ve
 act3
        : inspEndT(cy)≔curTime
END
inspPauseStart
 extended
STATUS
 ordinary
REFINES
 inspPauseStart
ANY
 CV
WHERE
            cy∈cycles ∧ ventilPhase(cy)=inspEnd
 grd1
 grd2 : modeC∈Ventilation
THEN
        : ventilPhase(cy)≔inspPauseBeg
 act1
        : inspPauseBegT(cy)≔curTime
: inspPauseEndT(cy)≔curTime+inspPauseMax
 act2
 act3
        : inValve≔{FALSE⇔FALSE, TRUE⇔inValve}(inValveF)
 act4
END
inspPauseEnd
 extended
STATUS
 ordinary
REFINES
 inspPauseEnd
ANY
WHERE
 grd1 : cy∈cycles ∧ ventilPhase(cy)=inspPauseBeg
 grd2 : curTime≤inspPauseEndT(cy)
       : curTime>inspPauseBegT(cy)
 grd3
 grd4
            modeC∈Ventilation
THEN
```

```
act1 : ventilPhase(cy)
             :=inspPauseEnd
  act2
            inspPauseEndT(cy)≔curTime
END
rmStart
  extended
STATUS
 ordinary
REFINES
 rmStart
ANY
  СУ
  t
WHERE
  grd1
        : cy∈cycles
  grd2 : ventilPhase(cy) \in \{inspEnd, inspPauseEnd\}
        : t=max(\{x|x\in dom(comParamsValC) \land x \le inspBegT(cy)\})
  grd3
  grd4
            modeC∈Ventilation
THEN
  act1 : ventilPhase(cy)≔rmBeg
  act2 : rmBegT(cy)≔curTime
  act3 : rmEndT(cy) = curTime + (comParamsValC(t))(timerRM)
  act4 : inValve≔{FALSE→TRUE, TRUE→inValve}(inValveF)
END
rmEnd
 extended
STATUS
  ordinary
REFINES
 rmEnd
ANY
  cy
WHERE
  grd1
            cy∈cycles
  grd2
            ventilPhase(cy)=rmBeg
 grd3
            curTime=rmEndT(cy)
  grd4
            modeC∈Ventilation
THEN
  act1
        : ventilPhase(cy)≔rmEnd
END
expStart
 extended
STATUS
 ordinary
REFINES
 expStart
ANY
  CV
  t
  expT
WHERE
  grd1
            cy∈cycles ∧ ventilPhase(cy)∈{inspEnd, inspPauseEnd,rmEnd}
  grd2
             t=max(\{x|x\in dom(pcvParamsValC) \land x \leq inspBegT(cy)\})
  grd3
            expT<u>c</u>N
             cycleMode(cy)=PCV \Rightarrow
  grd4
             expT={curTime +60÷((pcvParamsValC(t))(RRPCV) *
                        (1 +(pcvParamsValC(t))(IEPCV)))}
             cycleMode(cy)=PSV \Rightarrow
             expT=
             (curTime+(inspEndT(cy)-inspBegT(cy))÷2)
  grd5
             (curTime +((psvParamsValC(t)))(ApneaLag))
  grd6
             modeC∈Ventilation
THEN
            ventilPhase(cy)≔expBeg
  act1
  act2
             expBegT(cy)≔curTime
  act3
             expEndT(cy)≔expT
            inValve≔{FALSE→FALSE, TRUE→inValve}(inValveF)
  act4
  act5 :
            outValve={FALSE→TRUE, TRUE→outValve}(outValveF)
```

END

expEnd

```
extended
STATUS
 ordinary
REFINES
 expEnd
ANY
 modea
 modec
 pcv
 paw
 сy
 lastTime
 alarmOutValue
WHERE
             modeG∈ Ventilation
 grd1
 grd2
             modeg∈Ventilationu{Settings}
 grd3
             modeC∈Ventilation\{FailSafe}
 grd4
             modec∈Ventilation
 grd5
             pcv ∈ pcvParams→N1
 grd6
             modeC=PCV \implies modec=PCV
             modeC=PSV ^ modec=PCV
             pcv=pcvParamsValC(max(dom(pcvParamsValC)))
 grd7
              {RRPCV → (psvParamsValC(max(dom(psvParamsValC)))) (RRAP),
               PinspPCV→(psvParamsValC(max(dom(psvParamsValC))))(PinspAP),
              IEPCV→IEAP}
             modeC=PCV v modec=PSV
 grd8
             pcv=pcvParamsValC(max(dom(pcvParamsValC)))
 grd9
             \Rightarrow
              10*60*pcv(IEPCV)÷ (pcv(RRPCV)*(1 +pcv(IEPCV))) >0
 grd11
              modeG≠Off⇒ modeg=modec
 grd12
              cy∈cycles
 grd13
               ventilPhase(cy)=expBeg
 grd14
               lastTime= max(dom(PAW))
 grd15
              paw∈N
          1
 grd16
               t=max(\{x \mid x \in dom(psvParamsValC) \land x \leq inspBegT(cy)\})
               cycleMode(cy)=PCV ∧
                   (curTime∈ expEndT(cy) v
                     (curTime<max(expEndT(cy)) A
                       (PAW(lastTime)<paw ∧ PAW(lastTime)-paw>(pcvParamsValC(t))(ITSPCV)))
 grd17
               ν
               cycleMode(cy)=PSV л
                   (curTime= max(expEndT(cy)) v
                     (curTime≥min(expEndT(cy)) ∧ curTime<max(expEndT(cy)) ∧ PAW(lastTime)<paw ∧ PAW
               (lastTime)-paw>(psvParamsValC(t))(ITSPSV)))
               curTime=max(expEndT(cy)) \( \text{ cycleMode(cy)=PSV} \)
 grd18
              modec=PCV
               curTime≠max(expEndT(cy)) \( \text{cycleMode(cy)=PSV} \)
 grd19
              modec=PSV
               cycleMode(cy)=PCV
 grd20
              modec=PCV
 grd21
              modeC \in Ventilation
 grd22
              modeC=PSV ^ modec=PCV
               alarmOutValue=bool(
               (\exists p \cdot p \in pcvParams \land pcv(p) \notin dompcvParams(p))
               (∃ p· p ∈ comParams \wedge
```

```
({\tt comParamsValC(max(dom(comParamsValC))))(p) \not\in domcomParams(p))}
             (\exists p \cdot p \in dom((psvParamsValC(max(dom(psvParamsValC)))))) \land
              (psvParamsValC(max(dom(psvParamsValC))))(p)∉dompsvParams(p))
             modeC=PCV v modec=PSV
 grd23 :
             alarmOutValue=alarmRaised(outsideValue)
THEN
 act1 : modeG≔modeg
       : modeGP:=modeG
 act2
 act3
            modeC≔modec
 act4
            modeCP:=modeC
 act5 :
           pcvParamsValC(curTime)≔pcv
 act6 : psvParamsValC(curTime)≔psvParamsValC(max(dom(psvParamsValC)))
 act7 : comParamsValC(curTime)≔comParamsValC(max(dom(comParamsValC)))
        : ventilPhase(cy)≔expEnd
 act8
 act9
        : PAW(curTime)≔paw
 act10 : expEndT(cy)≔curTime..curTime
 act11 : alarmRaised(outsideValue)≔alarmOutValue
END
expPauseStart
 extended
STATUS
 ordinary
REFINES
 expPauseStart
ANY
 сy
WHERE
 grd1 : cy∈cycles ∧ ventilPhase(cy)=expEnd
 grd2 : modeC \in Ventilation
 grd3 : ran(ventilPhase) \subseteq \{expEnd, expPauseEnd\}
THEN
 act1 : ventilPhase(cy)≔expPauseBeg
 act2 : expPauseBegT(cy)≔curTime
 act3 : expPauseEndT(cy)≔curTime+expPauseMax
 act4 : inValve≔{FALSE→FALSE, TRUE→inValve}(inValveF)
 act5 : outValve={FALSE→FALSE, TRUE→outValve}(outValveF)
END
expPauseEnd
 extended
STATUS
 ordinary
REFINES
 expPauseEnd
ANY
 CV
WHERE
 grd1
       : cy∈cycles ∧ ventilPhase(cy)=expPauseBeg
 grd2 : curTime≤expPauseEndT(cy)
 grd3 : curTime>expPauseBegT(cy)
 grd4 : modeC∈Ventilation
THEN
           ventilPhase(cy)≔expPauseEnd
 act1
 act2 :
           expPauseEndT(cy)≔curTime
END
inValveF
 extended
STATUS
 ordinary
REFINES
 inValveF
WHEN
 grd1
           inValveF=FALSE
THEN
 act1 : inValveF≔TRUE
END
```

```
outValveF
 extended
STATUS
 ordinary
REFINES
 {\tt outValveF}
WHEN
 grd1 : outValveF=FALSE
THEN
 act1 : outValveF≔TRUE
END
connectPatient ≜
STATUS
 ordinary
WHEN
 grd1 : patientConnected=FALSE
THEN
 act1 : patientConnected≔TRUE
 act2 : alarmRaised(patConnected)≔bool(modeC∈{StartUp,SelfTest})
END
disconnectPatient ≜
STATUS
 ordinary
WHEN
 grd1 : patientConnected=TRUE
 \verb"act1": patientConnected:=FALSE"
 act2 : alarmRaised(patConnected)≔FALSE
END
FI1Fail ≜
STATUS
 ordinary
WHEN
 grd1 : FI1=TRUE
THEN
 act1 : FI1≔FALSE
 act2 : alarmRaised(FI1Failure)≔bool(modeC∉{Off, StartUp, SelfTest})
END
FI2Fail ≜
STATUS
 ordinary
WHEN
 grd1 : FI2=TRUE
THEN
 act1 : FI2≔FALSE
 act2 : alarmRaised(FI2Failure)≔bool(modeC∉{Off, StartUp,SelfTest})
oxygenSensorFail ≜
STATUS
 ordinary
WHEN
 grd1 : oxygenSensor=TRUE
THEN
 act1 :
           oxygenSensor≔FALSE
 act2 :
           alarmRaised(oxygenSensorFailure)≔bool(modeC∉{Off, StartUp,SelfTest})
END
```

END