MACHINE CruiseControl
SEES CruiseCtx2,discCtx
VARIABLES

cruise_mode booleana care indica daca modul cruise este activ
follow_mode booleana care indica daca modul follow este activ
pedal_command
emergency_mode booleana care indica daca modul emergency este activ
cruise_speed viteza curenta a vehiculului in modul cruise (limitata intre CRUISE_MIN_SPEED si CRUISE_MAX_SPEED)
engine_state booleana care indica daca motorul este pornit sau oprit
distance_sensor valoare intreaga care reprezinta distanta pana la un obstacol (-1 indica lipsa datelor)
vehicle_speed valoare intreaga care reprezinta viteza curenta a vehiculului (limitata intre 0 si VEHICLE_MAX_SPEED)
safety_distance distanta de siguranta pentru modul follow (numar natural)
warning_alert booleana care indica daca un avertisment este activ
t variabila de timp folosita pentru esantionare (numar natural)
canRead booleana care indica daca datele pot fi citite
workingClock booleana care indica daca ceasul de esantionare functioneaza
inactivity_timer

INVARIANTS

inv1: $cruise\_mode \in BOOL$
Modul de croaziera este activ sau inactiv
inv2: $follow\_mode \in BOOL$
Modul de urmarire este activ sau inactiv
inv3: $emergency\_mode \in BOOL$
Modul de urgenta este activ sau inactiv
inv4: $cruise\_speed \in CRUISE\_MIN\_SPEED \mathrel{..} CRUISE\_MAX\_SPEED$
Viteza de croaziera trebuie sa fie intre limitele stabilite
inv5: $engine\_state \in BOOL$
Motorul poate fi pornit sau oprit
inv6: $distance\_sensor \in -1 \mathrel{..} SAFETY\_DISTANCE$
Senzorul de distanta poate fi intre -1 si distanta de siguranta
inv7: $vehicle\_speed \in 0 \mathrel{..} VEHICLE\_MAX\_SPEED$
Viteza vehiculului trebuie sa fie intre 0 si viteza maxima permisa
inv8: $safety\_distance \in \mathbb{N}$
Distanta de siguranta este un numar natural
inv9: $warning\_alert \in BOOL$
Alerta de avertizare este activa sau inactiva
inv10: $t \in \mathbb{N}$
Cronometrul de esantionare este un numar natural
inv11: $canRead \in BOOL$
Indica daca se poate efectua o citire
inv12: $workingClock \in BOOL$
Starea ceasului de lucru
inv13: $pedal\_command \in PEDAL\_COMMANDS$
Comenzile pedalei (0: fara actiune, 1: accelereaza, 2: franeaza, 3: reseteaza)
inv14: $inactivity\_timer \in 0 \mathrel{..} MAX\_INACTIVITY$
Cronometrul de inactivitate este intre 0 si limita maxima

EVENTS

Initialisation

configureaza toate variabilele sistemului in starile lor initiale

begin

act1: $cruise\_mode := FALSE$
Modul de croaziera este dezactivat
act2: $follow\_mode := FALSE$
Modul de urmarire este dezactivat

act3: $emergency\_mode := FALSE$
 Modul de urgenta este dezactivat
act4: $cruise\_speed := 50000$
 Viteza de croaziera implicita
act5: $engine\_state := FALSE$
 Motorul este oprit
act6: $distance\_sensor := -1$
 Senzorul de distanta este inactiv
act7: $vehicle\_speed := 0$
 Viteza vehiculului este 0
act8: $safety\_distance := 50$
 Distanta de siguranta initiala este 50
act9: $warning\_alert := FALSE$
 Alerta este dezactivata
act10: $t := 0$
 Cronometrul incepe de la 0
act11: $canRead := FALSE$
 Nu este permisa citirea initial
act12: $workingClock := FALSE$
 Ceasul de lucru este dezactivat
act13: $pedal\_command := 0$
 Fara nicio comanda activa pe pedala
act14: $inactivity\_timer := 0$
 Cronometrul de inactivitate este initializat
end

Event StartEngine ⟨ordinary⟩ $\widehat{=}$
porneste motorul vehiculului daca acesta este oprit
when
grd1: $engine\_state = FALSE$
 Motorul trebuie sa fie oprit pentru a putea fi pornit
then
act1: $engine\_state := TRUE$
 Motorul este pornit
end

Event StopEngine ⟨ordinary⟩ $\widehat{=}$
opreste motorul vehiculului daca acesta este pornit. Toate modurile sunt dezactivate
when
grd1: $engine\_state = TRUE$
 Motorul trebuie sa fie pornit pentru a putea fi oprit
then
act1: $engine\_state := FALSE$
 opreste motorul
act2: $cruise\_mode := FALSE$
 reseteaza cruise_mode
act3: $follow\_mode := FALSE$
 reseteaza follow_mode
act4: $emergency\_mode := FALSE$
 reseteaza emergency_mode
end

Event EnterCruiseMode ⟨ordinary⟩ $\widehat{=}$
activeaza modul de croaziera atunci cand motorul este pornit, nu exista vehicule în apropiere (distance_sensor = -1), iar vehiculul are o viteza mai mare de 0
when
grd1: $engine\_state = TRUE$
 Motorul este pornit
grd2: $distance\_sensor = -1$
grd3: $vehicle\_speed > 0$
then

```
        act1: $cruise\_mode := TRUE$
            Modul de croaziera este activat
        act2: $follow\_mode := FALSE$
            Modul de urmarire este dezactivat
        act3: $emergency\_mode := FALSE$
            Modul de urgenta este dezactivat
        act4: $inactivity\_timer := 0$
            reseteaza cronometrul de inactivitate
    end
Event EnterFollowMode ⟨ordinary⟩ $\widehat{=}$
    ajusteaza comportamentul vehiculului pentru a mentine distanta fata de alte vehicule
    when
        grd1: $engine\_state = TRUE$
        grd2: $distance\_sensor \leq SAFETY\_DISTANCE$
        grd3: $distance\_sensor > 0$
    then
        act1: $cruise\_mode := FALSE$
        act2: $follow\_mode := TRUE$
        act3: $emergency\_mode := FALSE$
        act4: $inactivity\_timer := 0$
    end
Event EnterEmergencyMode ⟨ordinary⟩ $\widehat{=}$
    activeaza modul de urgenta atunci când distanta fata de un alt vehicul este 0 (distance_sensor = 0), iar
    sistemul emite o alerta
    when
        grd1: $engine\_state = TRUE$
            conditii pentru a intra in modul emergency
        grd2: $distance\_sensor = 0$
    then
        act1: $cruise\_mode := FALSE$
        act2: $follow\_mode := FALSE$
        act3: $emergency\_mode := TRUE$
        act4: $warning\_alert := TRUE$
        act5: $inactivity\_timer := 0$
    end
Event IncreaseCruiseSpeed ⟨ordinary⟩ $\widehat{=}$
    when
        grd1: $cruise\_mode = TRUE$
        grd2: $cruise\_speed + 2500 \leq CRUISE\_MAX\_SPEED$
    then
        act1: $cruise\_speed := cruise\_speed + 2500$
            creste cruise_speed cu 2500 unitati
    end
Event DecreaseCruiseSpeed ⟨ordinary⟩ $\widehat{=}$
    when
        grd1: $cruise\_mode = TRUE$
        grd2: $cruise\_speed - 2500 \geq CRUISE\_MIN\_SPEED$
    then
        act1: $cruise\_speed := cruise\_speed - 2500$
    end
Event UpdateSafetyDistance ⟨ordinary⟩ $\widehat{=}$
    when
        grd1: $engine\_state = TRUE$
            actualizeaza distanta de siguranta doar cand motorul este pornit
    then
        act1: $safety\_distance := SAFETY\_DISTANCE$
            seteaza safety_distance la constanta predefinita
    end
```

Event TriggerDriverWarning ⟨ordinary⟩ $\widehat{=}$
    when
        grd1: $warning\_alert = TRUE$
            declansare doar daca warning_alert este activ
    then
        act1: $warning\_alert := TRUE$
            mentine avertismentul activ
    end

Event MonitorDistanceAboveSafety ⟨ordinary⟩ $\widehat{=}$
    distanta mai mare decat pragul de siguranta
    when
        grd1: $engine\_state = TRUE$
        grd2: $distance\_sensor > SAFETY\_DISTANCE$
    then
        act1: $cruise\_mode := TRUE$
        act2: $follow\_mode := FALSE$
        act3: $emergency\_mode := FALSE$
        act4: $warning\_alert := FALSE$
    end

Event MonitorDistanceWithinSafety ⟨ordinary⟩ $\widehat{=}$
    when
        grd1: $engine\_state = TRUE$
        grd2: $distance\_sensor \leq SAFETY\_DISTANCE$
        grd3: $distance\_sensor > 0$
    then
        act1: $cruise\_mode := FALSE$
        act2: $follow\_mode := TRUE$
        act3: $emergency\_mode := FALSE$
        act4: $warning\_alert := FALSE$
    end

Event MonitorDistanceCritical ⟨ordinary⟩ $\widehat{=}$
    when
        grd1: $engine\_state = TRUE$
        grd2: $distance\_sensor = 0$
    then
        act1: $cruise\_mode := FALSE$
        act2: $follow\_mode := FALSE$
        act3: $emergency\_mode := TRUE$
        act4: $warning\_alert := TRUE$
    end

Event StartSampling ⟨ordinary⟩ $\widehat{=}$
    when
        grd1: $workingClock = FALSE$
            porneste esantionarea doar daca ceasul nu functioneaza si datele nu pot fi citite
        grd2: $canRead = FALSE$
    then
        act1: $workingClock := TRUE$
            porneste ceasul de esantionare
    end

Event StopSampling ⟨ordinary⟩ $\widehat{=}$
    when
        grd1: $workingClock = TRUE$
            opreste esantionarea cand ceasul functioneaza si timpul ajunge la pragul definit DT
        grd2: $t = DT$
    then
        act1: $canRead := TRUE$
            permite citirea datelor

```
            act2: $t := 0$
               reseteaza contorul de timp t la 0
            act3: $workingClock := FALSE$
               opreste ceasul de esantionare
      end
Event Sampling ⟨ordinary⟩ $\widehat{=}$
      when
            grd1:  $workingClock = TRUE$
               esantioneaza cat timp ceasul functioneaza si timpul este mai mic decat DT
            grd2:  $t < DT$
      then
            act1: $t := t + 1$
               incrementeaza contorul de timp t
      end
Event ApplyPedalCommand ⟨ordinary⟩ $\widehat{=}$
      any
            cmd
      where
            grd1:  $cmd \in \{0, 1, 2, 3\}$
               0: no action, 1: accelerate, 2: brake, 3: reset
      then
            act1: $pedal\_command := cmd$
      end
Event Accelerate ⟨ordinary⟩ $\widehat{=}$
      when
            grd1:  $pedal\_command = 1$
            grd2:  $vehicle\_speed + 5000 \leq VEHICLE\_MAX\_SPEED$
      then
            act1: $vehicle\_speed := vehicle\_speed + 5000$
            act2: $inactivity\_timer := 0$
      end
Event BrakeDecrease ⟨ordinary⟩ $\widehat{=}$
      when
            grd1:  $pedal\_command = 2$
            grd2:  $vehicle\_speed > 5000$
      then
            act1: $vehicle\_speed := vehicle\_speed - 5000$
            act2: $inactivity\_timer := 0$
      end
Event ResetPedalCommand ⟨ordinary⟩ $\widehat{=}$
      reseteaza comenzile pedalei (pedal_command) la starea initiala
      when
            grd1:  $pedal\_command = 3$
      then
            act1: $pedal\_command := 0$
            act2: $inactivity\_timer := 0$
      end
Event BrakeStop ⟨ordinary⟩ $\widehat{=}$
      when
            grd1:  $pedal\_command = 2$
            grd2:  $vehicle\_speed \leq 5000$
      then
            act1: $vehicle\_speed := 0$
            act2: $inactivity\_timer := 0$
      end
Event CheckInactivity ⟨ordinary⟩ $\widehat{=}$
      daca vehiculul este inactiv pentru o perioada maxima definita (MAX_INACTIVITY), revine automat la
      modul de croaziera
```

```
    when
        grd1:  $inactivity\_timer = MAX\_INACTIVITY$
    then
        act1: $cruise\_mode := TRUE$
        act2: $follow\_mode := FALSE$
        act3: $emergency\_mode := FALSE$
        act4: $warning\_alert := FALSE$
        act5: $inactivity\_timer := 0$
    end
END
```