

Contents

CONTEXT CruiseCtx	2
CONTEXT CruiseCtx1	3
CONTEXT CruiseCtx2	4
MACHINE CruiseMc	5
MACHINE CruiseMc1	7
MACHINE CruiseMc2	13

CONTEXT CruiseCtx

CONSTANTS

CHANGE.BOOL.STATE procedura pentru alternarea unei valori bool

CRUISE.MAX.SPEED viteza maxima admisa a sistemului de croaziera

CRUISE.MIN.SPEED viteza minima admisa a sistemului de croaziera

AXIOMS

axm1: $CHANGE_BOOL_STATE \in BOOL \rightarrow BOOL$

axm2: $CHANGE_BOOL_STATE = \{TRUE \mapsto FALSE, FALSE \mapsto TRUE\}$

axm3: $CRUISE_MAX_SPEED \in \mathbb{N}$

axm4: $CRUISE_MAX_SPEED = 180000$

valoarea maxima a vitezei de croaziera ce poate fi setata este de 180km/h

axm5: $CRUISE_MIN_SPEED \in \mathbb{N}$

axm6: $CRUISE_MIN_SPEED = 10000$

valoarea minima a vitezei de croaziera ce poate fi setata este de 10km/h

END

CONTEXT CruiseCtx1

EXTENDS CruiseCtx

CONSTANTS

PEDAL_COMMANDS comenzi posibile pentru apasarea pedalelor

SAFETY_DISTANCE

AXIOMS

axm1: $PEDAL_COMMANDS = 0 \dots 3$

conventie: 0 - nicio comanda; 1 - crestere nivel apasare; 2 - scadere nivel apasare; 3 - scadere forta de apasare la 0

axm2: $SAFETY_DISTANCE \in \mathbb{N}$

axm3: $SAFETY_DISTANCE = 50$

distanța de siguranță are valoarea prestabilită de 50 de metri

END

CONTEXT CruiseCtx2

EXTENDS CruiseCtx1

CONSTANTS

VEHICLE_MAX_SPEED

AXIOMS

axm1: $VEHICLE_MAX_SPEED \in \mathbb{N}$

axm2: $VEHICLE_MAX_SPEED = 200000$

prin constructie vehiculul nostru are o viteza maxima de 200km/h

END

MACHINE CruiseMc**SEES** CruiseCtx**VARIABLES**

cruise_system_state

cruise_speed

engine_state

INVARIANTS*inv1: cruise_system_state* \in *BOOL**inv2: cruise_speed* \in *CRUISE_MIN_SPEED* .. *CRUISE_MAX_SPEED**inv3: engine_state* \in *BOOL***EVENTS****Initialisation****begin***act1: cruise_system_state* := *FALSE*

initial sistemul de croaziera este oprit

act2: cruise_speed := 50000

viteza de croaziera prestabilita este de 50km/h

act3: engine_state := *FALSE*

motorul este oprit

end**Event** start_engine \langle ordinary $\rangle \hat{=}$

porneste motorul daca este oprit

when*grd1: engine_state* = *FALSE***then***act1: engine_state* := *CHANGE_BOOL_STATE(engine_state)***end****Event** stop_engine \langle ordinary $\rangle \hat{=}$

opreste motorul si sistemul de croaziera daca motorul este pornit

when*grd2: engine_state* = *TRUE***then***act2: cruise_system_state* := *FALSE**act3: engine_state* := *FALSE***end****Event** input_increase_cruise_speed \langle ordinary $\rangle \hat{=}$ **when***grd1: engine_state* = *TRUE*

motorul este pornit

grd2: cruise_system_state = *FALSE*

sistemul de croaziera este oprit

grd3: cruise_speed + 2500 \leq *CRUISE_MAX_SPEED*

verificam daca cresterea vitezei de croaziera depaseste viteza maxima

then*act1: cruise_speed* := *cruise_speed* + 2500

se adauga valoarea 2.5km/h la viteza de croaziera

end**Event** input_decrease_cruise_speed \langle ordinary $\rangle \hat{=}$ **when***grd1: engine_state* = *TRUE*

motorul este pornit

grd2: cruise_system_state = *FALSE*

sistemul de croaziera este oprit

grd3: cruise_speed - 2500 \geq *CRUISE_MIN_SPEED*

verificam daca scaderea valorii vitezei scade sub limita minima

then

```
    act1: cruise_speed := cruise_speed - 2500
          reducem viteza de croaziera cu 2.5km/h
    end
Event input_change_cruise_state ⟨ordinary⟩ ≐
  porneste sistemul de croaziera odata cu pornirea motorului
  when
    grd1: engine_state = TRUE
  then
    act1: cruise_system_state := CHANGE_BOOL_STATE(cruise_system_state)
  end
END
```

MACHINE CruiseMc1**REFINES** CruiseMc**SEES** CruiseCtx1**VARIABLES**

acc_pedal_command *stocheaza semnalul pentru actuatorul de acceleratie*
 brake_pedal_command *stocheaza semnalul pentru actuatorul de franare*
 engine_state
 cruise_system_state
 cruise_speed
 distance_sensor *semnal pentru distanta curenta fata de vehiculul din fata*
 prev_distance_sensor *stocare pentru distanta anterioara fata de vehiculul din fata*
 vehicle_board_warning
 safety_distance_timer

INVARIANTS

inv1: acc_pedal_command \in *PEDAL_COMMANDS*
inv2: brake_pedal_command \in *PEDAL_COMMANDS*
inv3: distance_sensor \in $-1 \dots SAFETY_DISTANCE$
 valoarea de -1 este valoarea returnata de senzorul de distanta atat timp cat nu avem un vehicul in
 distanta de siguranta
inv4: vehicle_board_warning \in *BOOL*
inv5: prev_distance_sensor \in $-1 \dots SAFETY_DISTANCE$
inv6: safety_distance_timer \in \mathbb{N}

EVENTS**Initialisation** *<extended>***begin**

act1: cruise_system_state := FALSE
 initial sistemul de croaziera este oprit
act2: cruise_speed := 50000
 viteza de croaziera prestabilita este de 50km/h
act3: engine_state := FALSE
 motorul este oprit
act7: acc_pedal_command := 0
 nu se transmite niciun semnal actuatorului de accelerare
act8: brake_pedal_command := 0
 nu se transmite niciun semnal actuatorului de franare
act9: distance_sensor := -1
 se asuma faptul ca nu este nicio masina in disanta de siguranta
act10: prev_distance_sensor := -1
act11: vehicle_board_warning := FALSE
act12: safety_distance_timer := 0

end**Event** input_change_cruise_state *<ordinary>* $\hat{=}$ **extends** input_change_cruise_state**when***grd1: engine_state = TRUE***then***act1: cruise_system_state := CHANGE_BOOL_STATE(cruise_system_state)***end****Event** input_decrease_cruise_speed *<ordinary>* $\hat{=}$ **extends** input_decrease_cruise_speed**when***grd1: engine_state = TRUE**motorul este pornit**grd2: cruise_system_state = FALSE**sistemul de croaziera este oprit*

```

    grd3: cruise_speed - 2500 ≥ CRUISE_MIN_SPEED
           verificam daca scaderea valorii vitezei scade sub limita minima
  then
    act1: cruise_speed := cruise_speed - 2500
           reducem viteza de croaziera cu 2.5km/h
  end
Event input_increase_cruise_speed ⟨ordinary⟩ ≐
extends input_increase_cruise_speed
  when
    grd1: engine_state = TRUE
           motorul este pornit
    grd2: cruise_system_state = FALSE
           sistemul de croaziera este oprit
    grd3: cruise_speed + 2500 ≤ CRUISE_MAX_SPEED
           verificam daca cresterea vitezei de croaziera depaseste viteza maxima
  then
    act1: cruise_speed := cruise_speed + 2500
           se adauga valoarea 2.5km/h la viteza de croaziera
  end
Event start_engine ⟨ordinary⟩ ≐
extends start_engine
  when
    grd1: engine_state = FALSE
  then
    act1: engine_state := CHANGE_BOOL_STATE(engine_state)
  end
Event stop_engine ⟨ordinary⟩ ≐
extends stop_engine
  when
    grd2: engine_state = TRUE
  then
    act2: cruise_system_state := FALSE
    act3: engine_state := FALSE
  end
Event execute_increase_acc_pedal_command ⟨ordinary⟩ ≐
when
  grd1: engine_state = TRUE
  grd2: cruise_system_state = TRUE
  grd3: acc_pedal_command = 1
           comanda curenta este de accelerare
then
  act2: acc_pedal_command := 0
           reseteaza comanda
end
Event execute_increase_brake_pedal_command ⟨ordinary⟩ ≐
when
  grd1: engine_state = TRUE
  grd2: cruise_system_state = TRUE
  grd3: brake_pedal_command = 1
           comanda curenta pentru actuatorul de franare este apasare
then
  act2: brake_pedal_command := 0
           reseteaza comanda
end
Event execute_decrease_acc_pedal_command ⟨ordinary⟩ ≐
when
  grd1: engine_state = TRUE

```



```

    grd2: cruise_system_state = TRUE
    grd3: acc_pedal_command = 2
           comanda curenta pentru actuatorul de acceleratie este scadere apasare
  then
    act2: acc_pedal_command := 0
           resetare comanda
  end
Event execute_decrease_brake_pedal_command ⟨ordinary⟩ ≐
  when
    grd1: engine_state = TRUE
    grd2: cruise_system_state = TRUE
    grd3: brake_pedal_command = 2
           comanda curenta pentru actuatorul de franare este apasare
  then
    act2: brake_pedal_command := 0
           reseteaza comanda
  end
Event execute_lift_acc_pedal_command ⟨ordinary⟩ ≐
  when
    grd1: engine_state = TRUE
    grd2: cruise_system_state = TRUE
    grd3: acc_pedal_command = 3
           comanda curenta pentru actuatorul de accelerare este eliminare apasare
  then
    act2: acc_pedal_command := 0
           reseteaza comanda
  end
Event execute_lift_brake_pedal_command ⟨ordinary⟩ ≐
  when
    grd1: engine_state = TRUE
    grd2: cruise_system_state = TRUE
    grd3: brake_pedal_command = 3
           comanda curenta pentru actuatorul de franare este eliminare apasare
  then
    act2: brake_pedal_command := 0
           reseteaza comanda
  end
Event vehicle_enters_safety_distance ⟨ordinary⟩ ≐
  when
    grd1: prev_distance_sensor = -1
           la momentul anterior nu exista un vehicul in fata
    grd2: distance_sensor ≠ -1
           o masina a patruns in distanta de siguranta
    grd3: cruise_system_state = TRUE
    grd4: engine_state = TRUE
  then
    act1: acc_pedal_command := 3
           se trimite comanda pentru eliminarea apasarii acceleratiei
    act2: safety_distance_timer := 0
           se seteaza cronometrul pentru distanta de siguranta
    act3: prev_distance_sensor := distance_sensor
           se stocheaza distanta curenta ca valoare anterioara
    act4: vehicle.board_warning := TRUE
           se trimite un semnal de atentionare soferului
  end
Event vehicle_in_safety_distance_before_timer ⟨ordinary⟩ ≐
  eveniment declansat cand masina se afla in distanta de siguranta de mai putin de 10 secunde
  when

```

```

    grd1: distance_sensor  $\neq$  -1
           o masina se afla in fata
    grd2: safety_distance_timer < 10
           cronometrul nu a atins 10 secunde
    grd3: cruise_system_state = TRUE
    grd4: engine_state = TRUE
    grd5: prev_distance_sensor  $\neq$  -1
           vehiculul a fost detectat anterior
  then
    act1: safety_distance_timer := safety_distance_timer + 1
           se incrementeaza cronometrul cu o secunda
    act2: prev_distance_sensor := distance_sensor
           actualizam distanta pana la vehicul
  end
Event vehicle_in_safety_distance_increased ⟨ordinary⟩  $\hat{=}$ 
  vehiculul din fata se distanteaza
when
  grd1: distance_sensor  $\neq$  -1
  grd2: safety_distance_timer  $\geq$  10
           vehiculul se afla in distanta de siguranta de mai mult de 10 secunde
  grd3: cruise_system_state = TRUE
  grd4: engine_state = TRUE
  grd5: distance_sensor > prev_distance_sensor
           vehiculul din fata se distanteaza de vehiculul nostru
then
  act1: safety_distance_timer := safety_distance_timer + 1
           se incrementeaza cronometrul
  act2: prev_distance_sensor := distance_sensor
  act3: brake_pedal_command := 2
           se trimite comanda de scadere a apasarii franei
end
Event vehicle_in_safety_distance_decreased ⟨ordinary⟩  $\hat{=}$ 
  vehiculul din fata se apropie
when
  grd1: distance_sensor  $\neq$  -1
  grd2: safety_distance_timer  $\geq$  10
           vehiculul din fata se afla acolo de cel putin 10 secunde
  grd3: cruise_system_state = TRUE
  grd4: distance_sensor  $\leq$  prev_distance_sensor
           vehiculul din fata se apropie de vehiculul nostru
  grd5: engine_state = TRUE
then
  act1: safety_distance_timer := safety_distance_timer + 1
           se incrementeaza cronometrul
  act2: prev_distance_sensor := distance_sensor
  act3: brake_pedal_command := 1
           se trimite comanda de crestere a franarii
end
Event vehicle_exits_safety_distance ⟨ordinary⟩  $\hat{=}$ 
  vehiculul din fata iese din detectia senzorului
when
  grd1: distance_sensor = -1
  grd2: prev_distance_sensor  $\neq$  -1
  grd3: cruise_system_state = TRUE
  grd4: engine_state = TRUE
then
  act1: prev_distance_sensor := distance_sensor
  act2: brake_pedal_command := 3
           se trimite comanda de eliminare a apasarii franei

```

```

    act3: vehicle_board_warning := FALSE
        se sterge avertizarea trimisa soferului
    end
Event input_increase_distance_sensor <ordinary>  $\hat{=}$ 
    vehiculul din fata se distanteaza (update valoare senzor)
    when
        grd1: cruise_system_state = TRUE
        grd2: engine_state = TRUE
        grd3: distance_sensor < SAFETY_DISTANCE
            distanta pana la vehiculul din fata este mai mica decat valoarea prestabilita a distantei de sig-
            ranta
        grd4: distance_sensor  $\neq$  -1
    then
        act1: distance_sensor := distance_sensor + 1
            creste valoarea detectata cu 1 metru (masina din fata s-a distantat cu 1 metru)
    end
Event input_decrease_distance_sensor <ordinary>  $\hat{=}$ 
    vehiculul din fata se apropie (update valoare senzor)
    when
        grd1: cruise_system_state = TRUE
        grd2: engine_state = TRUE
        grd3: distance_sensor > 0
    then
        act1: distance_sensor := distance_sensor - 1
            scade valoarea detectata cu 1 metru (masina din fata s-a distantat cu 1 metru)
    end
Event input_exits_range_distance_sensor <ordinary>  $\hat{=}$ 
    resetare senzor cand masina din fata iese din zona de detectie
    when
        grd1: cruise_system_state = TRUE
        grd2: engine_state = TRUE
        grd3: distance_sensor = SAFETY_DISTANCE
            exista o distanta de cel putin SAFETY_DISTANCE metri intre masina noastra si cea din fata
    then
        act1: distance_sensor := -1
            reseteaza senzor pe valoarea "nedetectat"
    end
Event cruise_accelerate_vehicle <ordinary>  $\hat{=}$ 
    when
        grd1: cruise_system_state = TRUE
        grd2: engine_state = TRUE
        grd3: distance_sensor = -1
            nu este nicio masina detectata in fata
        grd4: acc_pedal_command = 0
    then
        act1: acc_pedal_command := 1
            accelereaza vehiculul
    end
Event cruise_decelerate_vehicle <ordinary>  $\hat{=}$ 
    when
        grd1: cruise_system_state = TRUE
        grd2: engine_state = TRUE
        grd3: distance_sensor = -1
            nu este nicio masina in fata
    then
        act1: acc_pedal_command := 2
            scade nivelul de accelerare
    end

```

```
Event input_enters_range_distance_sensor <ordinary>  $\hat{=}$   
  when  
    grd1: cruise_system_state = TRUE  
    grd2: engine_state = TRUE  
    grd3: distance_sensor = -1  
  then  
    act1: distance_sensor := SAFETY_DISTANCE  
  end  
END
```

MACHINE CruiseMc2**REFINES** CruiseMc1**SEES** CruiseCtx2**VARIABLES**

distance_sensor
 prev_distance_sensor
 vehicle_board_warning
 safety_distance_timer
 acc_pedal_command
 brake_pedal_command
 cruise_speed
 cruise_system_state
 engine_state
 acc_pedal_perc
 brake_pedal_perc
 vehicle_speed
 ONE_PERC_SPEED

INVARIANTS

inv1: $acc_pedal_perc \in 0..100$
 procentajul apasarii pedalei de acceleratie
inv2: $brake_pedal_perc \in 0..100$
 procentajul apasarii pedalei de franare
inv3: $vehicle_speed \in \mathbb{Z}$
inv4: $ONE_PERC_SPEED \in \mathbb{N}$

EVENTS**Initialisation** ⟨extended⟩**begin**

act1: $cruise_system_state := FALSE$
 initial sistemul de croaziera este oprit
act2: $cruise_speed := 50000$
 viteza de croaziera prestabilita este de 50km/h
act3: $engine_state := FALSE$
 motorul este oprit
act7: $acc_pedal_command := 0$
 nu se transmite niciun semnal actuatorului de accelerare
act8: $brake_pedal_command := 0$
 nu se transmite niciun semnal actuatorului de franare
act9: $distance_sensor := -1$
 se asuma faptul ca nu este nicio masina in disanta de siguranta
act10: $prev_distance_sensor := -1$
act11: $vehicle_board_warning := FALSE$
act12: $safety_distance_timer := 0$
act13: $acc_pedal_perc := 0$
 initial pedala de acceleratie este neapasata
act14: $brake_pedal_perc := 0$
 initial pedala de franare este neapasata
act15: $vehicle_speed := 0$
 viteza initiala a vehiculului este 0
act16: $ONE_PERC_SPEED := VEHICLE_MAX_SPEED/100$
 aici definim ce inseamna un procent al vitezei masinii

end**Event** input_change_cruise_state ⟨ordinary⟩ $\hat{=}$ **extends** input_change_cruise_state**when***grd1:* $engine_state = TRUE$

```

    then
        act1: cruise_system_state := CHANGE_BOOL_STATE(cruise_system_state)
    end
Event execute_decrease_acc_pedal_command ⟨ordinary⟩ ≐
extends execute_decrease_acc_pedal_command
    when
        grd1: engine_state = TRUE
        grd2: cruise_system_state = TRUE
        grd3: acc_pedal_command = 2
            comanda curenta pentru actuatorul de acceleratie este scadere apasare
        grd4: acc_pedal_perc > 0
    then
        act2: acc_pedal_command := 0
            resetare comanda
        act3: acc_pedal_perc := acc_pedal_perc - 1
            se scade procentajul apasarii pedalei de acceleratie cu 1%
    end
Event execute_decrease_brake_pedal_command ⟨ordinary⟩ ≐
extends execute_decrease_brake_pedal_command
    when
        grd1: engine_state = TRUE
        grd2: cruise_system_state = TRUE
        grd3: brake_pedal_command = 2
            comanda curenta pentru actuatorul de franare este apasare
        grd4: brake_pedal_perc > 0
    then
        act2: brake_pedal_command := 0
            reseteaza comanda
        act3: brake_pedal_perc := brake_pedal_perc - 1
            se scade procentajul apasarii pedalei de frana cu 1%
    end
Event input_decrease_cruise_speed ⟨ordinary⟩ ≐
extends input_decrease_cruise_speed
    when
        grd1: engine_state = TRUE
            motorul este pornit
        grd2: cruise_system_state = FALSE
            sistemul de croaziera este oprit
        grd3: cruise_speed - 2500 ≥ CRUISE_MIN_SPEED
            verificam daca scaderea valorii vitezei scade sub limita minima
    then
        act1: cruise_speed := cruise_speed - 2500
            reducem viteza de croaziera cu 2.5km/h
    end
Event execute_increase_acc_pedal_command ⟨ordinary⟩ ≐
extends execute_increase_acc_pedal_command
    when
        grd1: engine_state = TRUE
        grd2: cruise_system_state = TRUE
        grd3: acc_pedal_command = 1
            comanda curenta este de accelerare
        grd4: acc_pedal_perc < 100
    then
        act2: acc_pedal_command := 0
            reseteaza comanda
        act3: acc_pedal_perc := acc_pedal_perc + 1
            se creste procentajul apasarii pedalei de acceleratie cu 1%
    end
end

```

Event execute_increase_brake_pedal_command $\langle \text{ordinary} \rangle \hat{=}$

extends execute_increase_brake_pedal_command

when

grd1: *engine_state* = *TRUE*

grd2: *cruise_system_state* = *TRUE*

grd3: *brake_pedal_command* = 1

comanda curenta pentru actuatorul de franare este apasare

grd4: *brake_pedal_perc* < 100

then

act2: *brake_pedal_command* := 0

reseteaza comanda

act3: *brake_pedal_perc* := *brake_pedal_perc* + 1

se creste procentajul apasarii pedalei de franare cu 1%

end

Event input_increase_cruise_speed $\langle \text{ordinary} \rangle \hat{=}$

extends input_increase_cruise_speed

when

grd1: *engine_state* = *TRUE*

motorul este pornit

grd2: *cruise_system_state* = *FALSE*

sistemul de croaziera este oprit

grd3: *cruise_speed* + 2500 ≤ *CRUISE_MAX_SPEED*

verificam daca cresterea vitezei de croaziera depaseste viteza maxima

then

act1: *cruise_speed* := *cruise_speed* + 2500

se adauga valoarea 2.5km/h la viteza de croaziera

end

Event execute_lift_acc_pedal_command $\langle \text{ordinary} \rangle \hat{=}$

extends execute_lift_acc_pedal_command

when

grd1: *engine_state* = *TRUE*

grd2: *cruise_system_state* = *TRUE*

grd3: *acc_pedal_command* = 3

comanda curenta pentru actuatorul de accelerare este eliminare apasare

then

act2: *acc_pedal_command* := 0

reseteaza comanda

act3: *acc_pedal_perc* := 0

end

Event execute_lift_brake_pedal_command $\langle \text{ordinary} \rangle \hat{=}$

extends execute_lift_brake_pedal_command

when

grd1: *engine_state* = *TRUE*

grd2: *cruise_system_state* = *TRUE*

grd3: *brake_pedal_command* = 3

comanda curenta pentru actuatorul de franare este eliminare apasare

then

act2: *brake_pedal_command* := 0

reseteaza comanda

act3: *brake_pedal_perc* := 0

end

Event start_engine $\langle \text{ordinary} \rangle \hat{=}$

extends start_engine

when

grd1: *engine_state* = *FALSE*

then

act1: *engine_state* := *CHANGE_BOOL_STATE*(*engine_state*)

act2: *vehicle_speed* := 1000

```

end
Event stop_engine ⟨ordinary⟩ ≐
extends stop_engine
when
  grd2: engine_state = TRUE
  grd3: vehicle_speed < 1000
then
  act2: cruise_system_state := FALSE
  act3: engine_state := FALSE
end
Event vehicle_enters_safety_distance ⟨ordinary⟩ ≐
extends vehicle_enters_safety_distance
when
  grd1: prev_distance_sensor = -1
    la momentul anterior nu exista un vehicul in fata
  grd2: distance_sensor ≠ -1
    o masina a patruns in distanta de siguranta
  grd3: cruise_system_state = TRUE
  grd4: engine_state = TRUE
then
  act1: acc_pedal_command := 3
    se trimite comanda pentru eliminarea apasarii acceleratiei
  act2: safety_distance_timer := 0
    se seteaza cronometrul pentru distanta de siguranta
  act3: prev_distance_sensor := distance_sensor
    se stocheaza distanta curenta ca valoare anterioara
  act4: vehicle_board_warning := TRUE
    se trimite un semnal de atentionare soferului
end
Event vehicle_in_safety_distance_before_timer ⟨ordinary⟩ ≐
extends vehicle_in_safety_distance_before_timer
when
  grd1: distance_sensor ≠ -1
    o masina se afla in fata
  grd2: safety_distance_timer < 10
    cronometrul nu a atins 10 secunde
  grd3: cruise_system_state = TRUE
  grd4: engine_state = TRUE
  grd5: prev_distance_sensor ≠ -1
    vehiculul a fost detectat anterior
then
  act1: safety_distance_timer := safety_distance_timer + 1
    se incrementeaza cronometrul cu o secunda
  act2: prev_distance_sensor := distance_sensor
    actualizam distanta pana la vehicul
end
Event vehicle_in_safety_distance_increased ⟨ordinary⟩ ≐
extends vehicle_in_safety_distance_increased
when
  grd1: distance_sensor ≠ -1
  grd2: safety_distance_timer ≥ 10
    vehiculul se afla in distanta de siguranta de mai mult de 10 secunde
  grd3: cruise_system_state = TRUE
  grd4: engine_state = TRUE
  grd5: distance_sensor > prev_distance_sensor
    vehiculul din fata se distanteaza de vehiculul nostru
  grd6: brake_pedal_perc > 100
then

```



```

    act1: safety_distance_timer := safety_distance_timer + 1
           se incrementeaza cronometrul
    act2: prev_distance_sensor := distance_sensor
    act3: brake_pedal_command := 2
           se trimite comanda de scadere a apasarii franei
end
Event vehicle_in_safety_distance_decreased ⟨ordinary⟩ ≐
extends vehicle_in_safety_distance_decreased
when
    grd1: distance_sensor ≠ -1
    grd2: safety_distance_timer ≥ 10
           vehiculul din fata se afla acolo de cel putin 10 secunde
    grd3: cruise_system_state = TRUE
    grd4: distance_sensor ≤ prev_distance_sensor
           vehiculul din fata se apropie de vehiculul nostru
    grd5: engine_state = TRUE
    grd6: brake_pedal_perc < 100
then
    act1: safety_distance_timer := safety_distance_timer + 1
           se incrementeaza cronometrul
    act2: prev_distance_sensor := distance_sensor
    act3: brake_pedal_command := 1
           se trimite comanda de crestere a franarii
end
Event vehicle_exits_safety_distance ⟨ordinary⟩ ≐
extends vehicle_exits_safety_distance
when
    grd1: distance_sensor = -1
    grd2: prev_distance_sensor ≠ -1
    grd3: cruise_system_state = TRUE
    grd4: engine_state = TRUE
then
    act1: prev_distance_sensor := distance_sensor
    act2: brake_pedal_command := 3
           se trimite comanda de eliminare a apasarii franei
    act3: vehicle_board_warning := FALSE
           se sterge avertizarea trimisa soferului
end
Event input_increase_distance_sensor ⟨ordinary⟩ ≐
extends input_increase_distance_sensor
when
    grd1: cruise_system_state = TRUE
    grd2: engine_state = TRUE
    grd3: distance_sensor < SAFETY_DISTANCE
           distanta pana la vehiculul din fata este mai mica decat valoarea prestabilita a distantei de siguranta
    grd4: distance_sensor ≠ -1
then
    act1: distance_sensor := distance_sensor + 1
           creste valoarea detectata cu 1 metru (masina din fata s-a distantat cu 1 metru)
end
Event input_decrease_distance_sensor ⟨ordinary⟩ ≐
extends input_decrease_distance_sensor
when
    grd1: cruise_system_state = TRUE
    grd2: engine_state = TRUE
    grd3: distance_sensor > 0
then

```

```

    act1: distance_sensor := distance_sensor - 1
           scade valoarea detectata cu 1 metru (masina din fata s-a distantat cu 1 metru)
end
Event input_exits_range_distance_sensor ⟨ordinary⟩ ≐
extends input_exits_range_distance_sensor
when
    grd1: cruise_system_state = TRUE
    grd2: engine_state = TRUE
    grd3: distance_sensor = SAFETY_DISTANCE
           exista o distanta de cel putin SAFETY_DISTANCE metri intre masina noastra si cea din fata
then
    act1: distance_sensor := - 1
           reseteaza senzor pe valoarea "nedetectat"
end
Event cruise_accelerate_vehicle ⟨ordinary⟩ ≐
extends cruise_accelerate_vehicle
when
    grd1: cruise_system_state = TRUE
    grd2: engine_state = TRUE
    grd3: distance_sensor = -1
           nu este nicio masina detectata in fata
    grd4: acc_pedal_command = 0
    grd5: vehicle_speed < cruise_speed
    grd6: brake_pedal_perc = 0
    grd7: acc_pedal_perc < 100
then
    act1: acc_pedal_command := 1
           accelereaza vehiculul
end
Event cruise_decelerate_vehicle ⟨ordinary⟩ ≐
extends cruise_decelerate_vehicle
when
    grd1: cruise_system_state = TRUE
    grd2: engine_state = TRUE
    grd3: distance_sensor = -1
           nu este nicio masina in fata
    grd4: vehicle_speed > cruise_speed
    grd5: brake_pedal_perc = 0
    grd6: acc_pedal_perc > 0
then
    act1: acc_pedal_command := 2
           scade nivelul de accelerare
end
Event input_enters_range_distance_sensor ⟨ordinary⟩ ≐
extends input_enters_range_distance_sensor
when
    grd1: cruise_system_state = TRUE
    grd2: engine_state = TRUE
    grd3: distance_sensor = -1
then
    act1: distance_sensor := SAFETY_DISTANCE
end
Event update_speed_with_acc_below ⟨ordinary⟩ ≐
           accelereaza vehiculul cand pedala de acceleratie este apasata
when
    grd1: acc_pedal_perc > 0
    grd2: engine_state = TRUE

```

```

    grd3:  $vehicle\_speed \leq acc\_pedal\_perc * ONE\_PERC\_SPEED$ 
           vedem daca viteza curenta a vehiculului e mai mica decat trebuie pentru procentul de apasare a
           acceleratiei
  then
    act1:  $vehicle\_speed := vehicle\_speed + 50 * (acc\_pedal\_perc - (vehicle\_speed * 100 / VEHICLE\_MAX\_SPEED))$ 

  end

Event update_speed_no_acc ⟨ordinary⟩  $\hat{=}$ 
  decelereaza vehiculul cand acceleratia nu este apasata (pedala de frana poate fi apasata)
  when
    grd1:  $acc\_pedal\_perc = 0$ 
    grd2:  $vehicle\_speed > 0$ 
  then
    act1:  $vehicle\_speed := vehicle\_speed - 50 * (((vehicle\_speed * 100 + ONE\_PERC\_SPEED) / VEHICLE\_MAX\_SPEED) -$ 
            $brake\_pedal\_perc)$ 
           actualizeaza viteza vehiculului cand se decelereaza
  end

Event update_speed_with_acc_above ⟨ordinary⟩  $\hat{=}$ 
  decelereaza vehiculul atunci cand pedala de acceleratie este apasata dar viteza depaseeste viteza maxima a
  vehiculului
  when
    grd1:  $acc\_pedal\_perc > 0$ 
    grd2:  $engine\_state = TRUE$ 
    grd3:  $vehicle\_speed > acc\_pedal\_perc * ONE\_PERC\_SPEED$ 
  then
    act1:  $vehicle\_speed := vehicle\_speed - 50 * ((vehicle\_speed * 100 / VEHICLE\_MAX\_SPEED) +$ 
            $1 - acc\_pedal\_perc)$ 
  end
end
END

```