

Work-Package 7: "Toolchain"

Event-B Model of Subset 026, Section 3.13

Matthias Güdemann

April 2013



Funded by:


 Federal Ministry
 of Education
 and Research

 Région de
 Bruxelles-
 Capitale

 GOBIERNO DE ESPAÑA
 MINISTERIO DE INDUSTRIA, ENERGÍA
 Y TURISMO

This page is intentionally left blank

Work-Package 7: “Toolchain”

**OETCS
April 2013**

Event-B Model of Subset 026, Section 3.13

Matthias Güdemann

Systemel
Les Portes de l'Arbois, Bâtiment A
1090 rue René Descartes
13857 Aix-en-Provence Cedex 3, France

Model Description

Prepared for openETCS@ITEA2 Project

Disclaimer: This work is licensed under the "openETCS Open License Terms" (oOLT) dual Licensing: European Union Public Licence (EURL v.1.1+) AND Creative Commons Attribution-ShareAlike 3.0 – (cc by-sa 3.0)

THE WORK IS PROVIDED UNDER openETCS OPEN LICENSE TERMS (oOLT) WHICH IS A DUAL LICENSE AGREEMENT INCLUDING THE TERMS OF THE EUROPEAN UNION PUBLIC LICENSE (VERSION 1.1 OR ANY LATER VERSION) AND THE TERMS OF THE CREATIVE COMMONS PUBLIC LICENSE ("CCPL"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS OLT LICENSE OR COPYRIGHT LAW IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. TO THE EXTENT THIS LICENSE MAY BE CONSIDERED TO BE A CONTRACT, THE LICENSOR GRANTS YOU THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS.

<http://creativecommons.org/licenses/by-sa/3.0/>

<http://joinup.ec.europa.eu/software/page/eupl/licence-eupl>

Table of Contents

1	Modeling Strategy	5
2	Model Overview	5
3	Model Benefits	7
4	Detailed Model Description.....	8
4.1	Context 0 - Train Inputs, TI and DMI command.....	8
4.2	Machine 0 - Train Status and Commands	9
4.3	Machine 1 - Brake Model.....	10
4.4	Context 1 - Decelerations	11
4.5	Machine 2 - Calculate Decelerations	11
4.6	Machine 3 - Calculation of Brake Buildup Time.....	12
4.7	Machine 4 - Acceleration due to Gradient.....	13
4.8	Context 3 - Speed Profiles	13
4.9	Machine 5 - Most Restrictive Speed Profile	14
4.10	Context 4 - Targets.....	14
4.11	Machine 6 - Supervised Targets.....	14
4.12	Context 5 - Braking Curves	15
4.13	Machine 7 - Braking Curves	15
4.14	Context 6 - Supervision Limits	16
4.15	Machine 8 - Supervision Limit.....	17
5	Model Decomposition.....	19
5.1	dcmp - Braking Model.....	19
5.2	dcmp - Calc Deceleration	25
5.3	dcmp - Brake Buildup	26
5.4	dcmp - Acceleration Gradient	30
5.5	dcmp - MRSP	34
5.6	dcmp - Supervised Targets	36
5.7	dcmp - Braking Curves	38
5.8	dcmp - Supervision Limits.....	41
5.9	dcmp - Monitoring Commands.....	44
	References	47

Figures and Tables

Figures

Figure 1. Speed and Distance Monitoring Overview ([Eur12] p. 85) 6

Figure 2. Decomposition of System 7

Figure 3. Machine Decomposition Overview 7

Figure 4. Decomposition Configuration..... 19

Tables

This document describes a formal model of the requirements of section 3.13 of the subset 026 of the ETCS specification 3.3.0 [Eur12]. This section describes the speed and distance monitoring subsystem of ETCS.

The model is expressed in the formal language Event-B [Abr10] and developed within the Rodin tool [Jas12]. This formalism allows an iterative modeling approach. In general, one starts with a very abstract description of the basic functionality and step-wise adds additional details until the desired level of accuracy of the model is reached. Rodin provides the necessary proof support to ensure the correctness of the refined behavior.

In this document we present an Event-B model of the speed and distance monitoring subsystem of ETCS. At first, we describe shortly the background of Event-B, then the overall approach taken to model this section and finally present the model in detail.

The section 3.13 of the SRS gives a very detailed description of the calculation of many necessary values for speed and distance monitoring. As Event-B is a system modeling approach, we give an abstract model of the system. The calculations are abstracted as functions and the system ensures the correct parameter flow to the functions. We illustrate the model decomposition capabilities of Event-B and Rodin by decomposing the overall model into different functional parts.

For a short introduction on Event-B and the usage of Rodin with models on github see https://github.com/openETCS/model-evaluation/blob/master/model/B-Systemrel/Event_B/rodin-projects-github.pdf?raw=true

1 Modeling Strategy

The section 3.13 of the SRS describes the speed and distance monitoring together with the necessary parameters and data. The model starts with an abstract modeling of dataflow of the various intermediate calculated values. This model is partitioned into functional parts, the model is decomposed using shared variables and the respective sub-models are refined until the basic calculation functions are reached.

2 Model Overview

The overview of the speed and distance monitoring is shown in Fig. 1 from the SRS.

The on-board system comprises only the middle layer. The upper layer gives train related inputs as parameters, the lower layer track related inputs. The system itself takes the current position, speed and acceleration of the train and computes commands for the train interface and for the driver machine interface. For the train interface, this consists of the command for the service and emergency brakes. For the driver machine interface this consists of the status indication for the driver.

The Event-B modeling starts with machines describing the dataflow of all inputs, outputs and intermediate values of the model. For example, the values that are calculated for $T_{brake_service}$ in *Traction / Braking Models* are written into a variable by an event that calculates then and these values are read as input by the event that calculates T_{bs} for *SBI* limit.

This approach is conducted for each intermediate value of the system until a single machine is created with one variable for each intermediate value as well as for each input and output. On

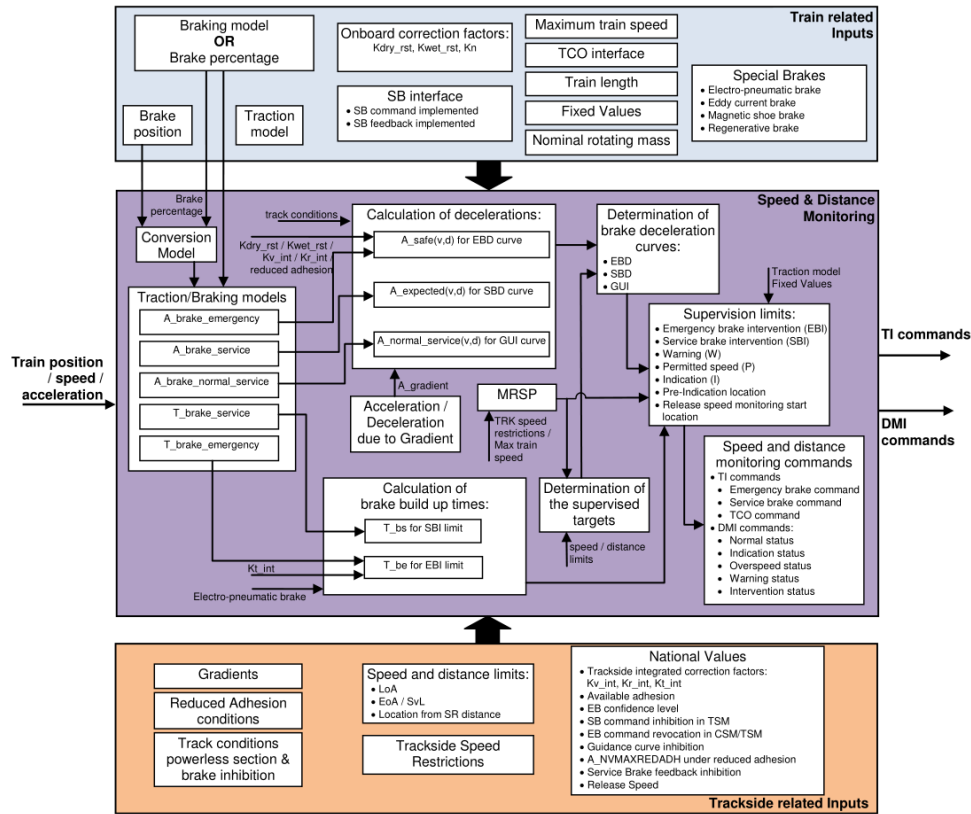


Figure 1. Speed and Distance Monitoring Overview ([Eur12] p. 85)

this level of modeling, all events only define the necessary input values and write a new value to their output variable. This value is provided as event parameter on this abstraction level.

The next step is to decompose the single machine into different sub-machines, in general one machine for each functional part of the model. This allows for model structuring and complexity reduction for each machine. For this we use the Rodin decomposition plug-in¹ using the shared-variable decomposition approach [SPHB11]. This approach splits the set of events of a machine into several disjoint sets and assigns one such set to each sub-machine. It also allows to distribute the variables over several machines, effectively implementing a shared variable distributed system.

The borders for the subsystem decomposition are shown in Fig. 2. The dashed lines show the separate sub-machines. The dataflows that cross these lines are represented by the shared variables of the decomposed model.

Each of the sub-machines with its shared variables is then further refined until the desired level of detail is reached. The overview of these refinements is shown in Fig. 3.

This refinement and context overview is very different from the others, as first an abstract global model was developed and then this model was decomposed into sub-models which are further refined. The contexts are shared between the decomposed models as far as possible. In this case, all resulting contexts and machines are kept in the same Rodin project. It is also possible to create a new project for each sub-machine which will reduce the complexity of each single project.

¹http://wiki.event-b.org/index.php/Decomposition_Plug-in_User_Guide

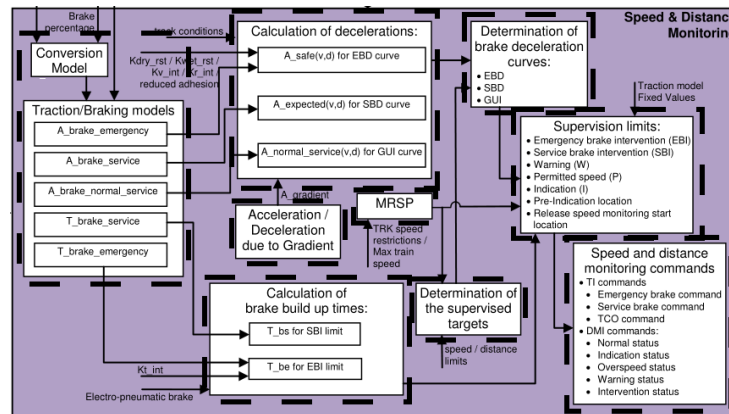


Figure 2. Decomposition of System

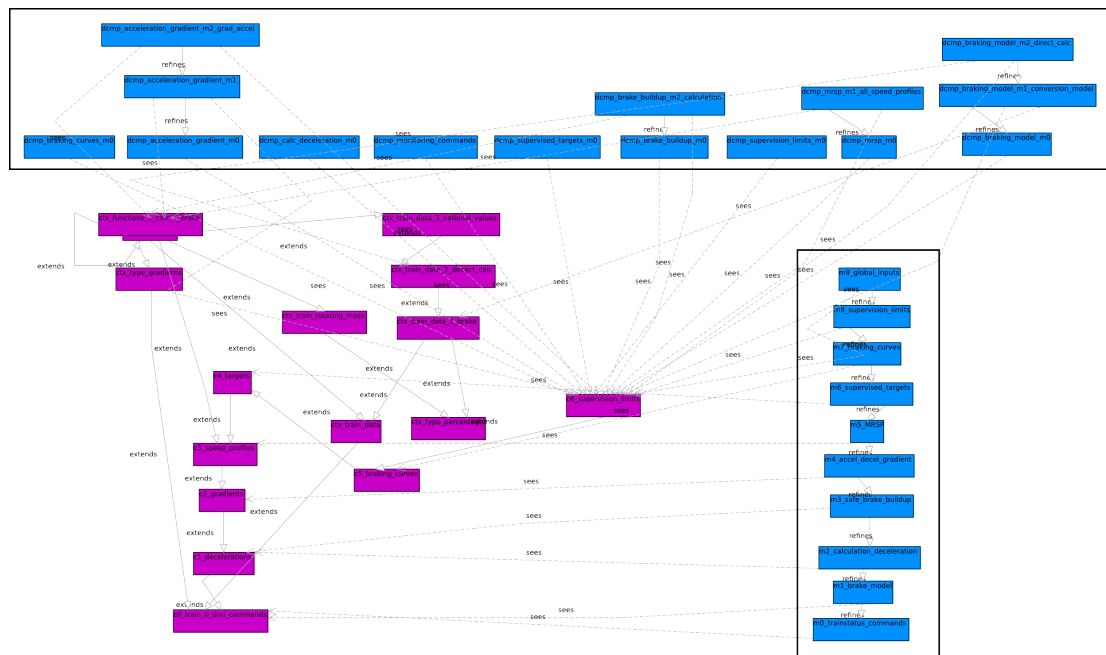


Figure 3. Machine Decomposition Overview

The global model is shown in the lower right. The first machine describes the global input and output variables of the system. The further refinements represent the iterative addition of more functions as shown in Fig. 2. For example the machine 1 adds the brake model with its inputs and outputs and the machine 2 adds the calculation of deceleration which uses the outputs of the braking model.

The last machine is then decomposed into the nine machines representing each a single functional block. This structure is shown in the upper part of Fig. 3, this also illustrates the further independent refinement of the decomposed sub-machine.

The context hierarchy also reflects this structuring. The contexts define the data types for the intermediate values, as well as the functions that calculate these values. These functions are generally not further refined in the Event-B model, as this is not part of the system modeling.

3 Model Benefits

The modeled section of the SRS provides many details for calculation of various values. The main content from a system modeling point of view is the model. So while in this case the same benefits from using Rodin as for [Mat13a, Mat13b, Mat13c] are present, the main advantage here is the model structuring facility.

- **Model Decomposition** The shared variable model decomposition [SPHB11] allows for decomposing an Event-B model and for separate refining of the machines of the resulting sub-models while retaining correctness of the refinement proofs.

4 Detailed Model Description

4.1 Context 0 - Train Inputs, TI and DMI command

The first context introduces many basic type for the model, *t_locations*, *t_speed*, *t_acceleration*, *t_TI_commands*, *t_DMI_commands*, *t_time* and *t_train_modes*.

The commands for the train interface (TI) are represented by the constants *c_emergency_brake*, *c_service_brake*, *c_TCO*, *c_no_command*. For the driver machine interface (DMI) the commands are represented by the constants *c_normal*, *c_indication*, *c_overspeed*, *c_warning* and *c_intervention*.

The other constants provide default values for the initialization of variables of that type.

CONTEXT c0_train_ti_dmi_commands
SETS

t_locations all possible locations on track
t_speed train speed measurement
t_acceleration train acceleration
t_TI_commands track interface commands
t_DMI_commands driver machine interface commands
t_time
t_train_modes

CONSTANTS

c_emergency_brake
c_service_brake
c_TCO traction cut off
c_no_command empty command
c_normal
c_indication
c_overspeed
c_warning
c_intervention
c_v0
c_a0
c_l0
c_a_brake0
c_T_brake0

AXIOMS

```

axm1 : partition(t_TI_commands, {c_no_command}, {c_emergency_brake},
               {c_service_brake}, {c_TCO})

axm2 : partition(t_DMI_commands, {c_normal}, {c_indication},
               {c_overspeed}, {c_warning}, {c_intervention})
axm3 : c_v0 ∈ t_speed
axm4 : c_a0 ∈ t_acceleration

axm5 : c_l0 ∈ t_locations
axm6 : c_a_brake0 ∈ t_speed → t_acceleration
               default brake profile

axm7 : c_T_brake0 ∈ t_time
               default brake buildup time

```

END

4.2 Machine 0 - Train Status and Commands

This first machine introduces the external input variables, i.e., the position, speed and acceleration of the train as well as the output variables, i.e., the TI commands and the DMI commands. The input variables are read by the event *update_train_style* and the output variables by the event *new_outputs*.

MACHINE m0_trainstatus_commands

SEES c0_train_ti_dmi_commands

VARIABLES

```

v_current  current speed of train
a_current  current acceleration of train
loc_current current position of train as track location
cmd_current current TI command
status_current current DMI status

```

INVARIANTS

```

inv1 : v_current ∈ t_speed
inv2 : a_current ∈ t_acceleration
inv3 : loc_current ∈ t_locations
inv4 : cmd_current ∈ t_TI_commands
inv5 : status_current ∈ t_DMI_commands

```

EVENTS

Initialisation

begin

```

act1 : v_current := c_v0
act2 : a_current := c_a0
act3 : loc_current := c_l0
act4 : cmd_current := c_no_command
act5 : status_current := c_normal

```

end

Event *update_train_state* ≡

any

```

l_speed
l_accel
l_loc

```

where

```

grd1 : l_speed ∈ t_speed
grd2 : l_accel ∈ t_acceleration

```

```

    then
        grd3 :  $l\_loc \in t\_locations$ 

        act1 :  $v\_current := l\_speed$ 
        act2 :  $a\_current := l\_accel$ 
        act3 :  $loc\_current := l\_loc$ 
    end
Event new_outputs  $\hat{=}$ 
    any
        l_ti_cmd
        l_dmi_status
    where
        grd1 :  $l\_ti\_cmd \in t\_TI\_commands$ 
        grd2 :  $l\_dmi\_status \in t\_DMI\_commands$ 
    then
        act1 :  $cmd\_current := l\_ti\_cmd$ 
        act2 :  $status\_current := l\_dmi\_status$ 
    end
END

```

4.3 Machine 1 - Brake Model

The first refinement adds the notion of the brake model. This is represented by the variables describing the speed dependent acceleration functions for emergency, service and normal service braking. The variables $T_brake_service$ and $T_brake_emergency$ describe the brake build-up times for the brakes.

```

MACHINE m1_brake_model
REFINES m0_trainstatus_commands
SEES c0_train_ti_dmi_commands
VARIABLES

    A_brake_emergency emergency brake acceleration
    A_brake_service service brake acceleration
    A_brake_normal_service
    T_brake_service
    T_brake_emergency
EVENTS
Event set_A_brake_emergency  $\hat{=}$ 
    any
        l_a_brake
    where
        then
            grd1 :  $l\_a\_brake \in t\_speed \rightarrow t\_acceleration$ 
        then
            act1 :  $A\_brake\_emergency := l\_a\_brake$ 
        end
Event set_A_brake_service  $\hat{=}$ 
    any
        l_a_brake
    where
        then
            grd1 :  $l\_a\_brake \in t\_speed \rightarrow t\_acceleration$ 
        then
            act1 :  $A\_brake\_service := l\_a\_brake$ 
        end
Event set_A_brake_normal_service  $\hat{=}$ 

```

```

    any
    where  $l\_a\_brake$ 
    then  $grd1 : l\_a\_brake \in t\_speed \rightarrow t\_acceleration$ 
    end  $act1 : A\_brake\_normal\_service := l\_a\_brake$ 
Event  $set\_T\_brake\_service \hat{=}$ 
    any
    where  $l\_T\_brake$ 
    then  $grd1 : l\_T\_brake \in t\_time$ 
    end  $act1 : T\_brake\_service := l\_T\_brake$ 
Event  $set\_T\_brake\_emergency \hat{=}$ 
    any
    where  $l\_T\_brake$ 
    then  $grd1 : l\_T\_brake \in t\_time$ 
    end  $act1 : T\_brake\_emergency := l\_T\_brake$ 
END

```

4.4 Context 1 - Decelerations

This context extension adds a distance type and a function that maps the speed and distance to an acceleration.

```

CONTEXT c1_decelerations
EXTENDS c0_train_ti_dmi_commands
SETS

     $t\_distance$ 
CONSTANTS

     $f\_A\_deceleration0$ 
AXIOMS

     $axm1 : f\_A\_deceleration0 \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
END

```

4.5 Machine 2 - Calculate Decelerations

This refinement adds the calculation of deceleration to the model. This is represented by three variables which are functions that map speed and distance to an acceleration. There is one function for each on of EBD, SBD and GUI.

```

MACHINE m2_calculation_deceleration
REFINES m1_brake_model
SEES c1_decelerations
VARIABLES

     $A\_safe$ 

```

```

    A_expected
    A_normal_service
EVENTS
Event set_A_safe  $\hat{=}$ 
  any
    where
      l_a_decel
    then
      grd1 : l_a_decel  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
      grd2 : A_brake_emergency  $\in$  t_speed  $\rightarrow$  t_acceleration
    then
      act1 : A_safe := l_a_decel
    end
Event set_A_expected  $\hat{=}$ 
  any
    where
      l_a_decel
    then
      grd1 : l_a_decel  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
      grd2 : A_brake_service  $\in$  t_speed  $\rightarrow$  t_acceleration
    then
      act1 : A_expected := l_a_decel
    end
Event set_A_normal_service  $\hat{=}$ 
  any
    where
      l_a_decel
    then
      grd1 : l_a_decel  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
      grd2 : A_brake_normal_service  $\in$  t_speed  $\rightarrow$  t_acceleration
    then
      act1 : A_normal_service := l_a_decel
    end
END

```

4.6 Machine 3 - Calculation of Brake Buildup Time

The next machine refinement adds the brake buildup calculation to the model. This is represented by two variables, T_{be} for the emergency brake and T_{se} for the service brake.

```

MACHINE m3_safe_brake_buildup
REFINES m2_calculation_deceleration
SEES c1_decelerations
VARIABLES
  T_be
  T_bs
EVENTS
Event set_T_be  $\hat{=}$ 
  any
    where
      l_t_be
    then
      grd1 : l_t_be  $\in$  t_time
      grd2 : T_brake_emergency  $\in$  t_time
    then
      act1 : T_be := l_t_be
    end

```

```

Event set_T_bs  $\hat{=}$ 
  any
    l_t_bs
  where
    grd1 : l_t_bs  $\in$  t_time
    grd2 : T_brake_service  $\in$  t_time
  then
    act1 : T_bs := l_t_bs
  end
END

```

4.7 Machine 4 - Acceleration due to Gradient

The refinement adds the notion of the acceleration due to gradient. This is represented by the variable *A_gradient* which is a function that maps speed to acceleration.

```

MACHINE m4_accel_decel_gradient
REFINES m3_safe_brake_buildup
SEES c2_gradients
VARIABLES
  A_gradient
EVENTS
Event set_A_gradient  $\hat{=}$ 
  any
    l_a_gradient
  where
    grd1 : l_a_gradient  $\in$  t_acceleration
  then
    act1 : A_gradient := l_a_gradient
  end
END

```

4.8 Context 3 - Speed Profiles

This context extension introduces the type *speed_profiles* which maps locations to speeds. It also defines one constant value of that type which is used as default value for variables of that type.

```

CONTEXT c3_speed_profiles
EXTENDS c2_gradients
CONSTANTS
  c_speed_profile0
  t_speed_profiles
AXIOMS
  axm1 : t_speed_profiles  $\subseteq$  t_locations  $\times$  t_speed
  axm2 : c_speed_profile0  $\in$  t_speed_profiles
END

```

4.9 Machine 5 - Most Restrictive Speed Profile

This machine refinement introduces the most restrictive speed profile to the model. This is represented by the variable *MRSP* of the type *speed_profile*.

```

MACHINE m5_MRSP
REFINES m4_accel_decel_gradient
SEES c3_speed_profiles
VARIABLES

    MRSP
EVENTS
Event set_MRSP  $\triangleq$ 
    any

    where

    then    grd1 :  $l_{sp} \in t\_speed\_profiles$ 

    end     act1 :  $MRSP := l_{sp}$ 
END

```

4.10 Context 4 - Targets

This context extension introduces the type *t_targets* which represents a target, i.e., a pair of location and speed.

```

CONTEXT c4_targets
EXTENDS c3_speed_profiles
CONSTANTS

    t_targets
    c_target0
AXIOMS

    axm1 :  $t\_targets \subseteq t\_locations \times t\_speed$ 

    axm2 :  $c\_target0 \in t\_targets$ 
END

```

4.11 Machine 6 - Supervised Targets

This refinement adds limit of authority, end of authority and supervision limit to the model. For each there exists one variable of type *t_targets*.

```

MACHINE m6_supervised_targets
REFINES m5_MRSP
SEES c4_targets
VARIABLES

    LOA
    EOA
    SvL
EVENTS
Event set_EOA  $\triangleq$ 

```



```

    any
       $l_{target}$ 
    where
       $grd1 : l_{target} \in t_{targets}$ 
       $grd2 : MRS P \in t_{speed\_profiles}$ 
    then
       $act1 : EOA := l_{target}$ 
    end
  Event  $set\_LOA \hat{=}$ 
    any
       $l_{target}$ 
    where
       $grd1 : l_{target} \in t_{targets}$ 
       $grd2 : MRS P \in t_{speed\_profiles}$ 
    then
       $act1 : LOA := l_{target}$ 
    end
  Event  $set\_SvL \hat{=}$ 
    any
       $l_{target}$ 
    where
       $grd1 : l_{target} \in t_{targets}$ 
       $grd2 : MRS P \in t_{speed\_profiles}$ 
    then
       $act1 : SvL := l_{target}$ 
    end
END

```

4.12 Context 5 - Braking Curves

This context extension introduces the type $t_{braking_curves}$ and a constant of that type.

```

CONTEXT c5_braking_curves
EXTENDS c4_targets
SETS
   $t_{braking\_curves}$ 
CONSTANTS
   $c\_curve0$ 
AXIOMS
   $axm1 : c\_curve0 \in t_{braking\_curves}$ 
END

```

4.13 Machine 7 - Braking Curves

This machine refinement adds the braking curves to the model, these are represented by the three variables EBD , SBD and GVI of the appropriate type.

```

MACHINE m7_braking_curves
REFINES m6_supervised_targets
SEES c5_braking_curves
VARIABLES

```

EBD

```

    SBD
    GUI
EVENTS
Event set_EBD  $\hat{=}$ 
    any
        l_curve
    where
        grd1 : l_curve  $\in$  t_braking_curves
        grd2 : A_safe  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd3 : A_expected  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd4 : A_normal_service  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd5 : LOA  $\in$  t_targets
        grd6 : EOA  $\in$  t_targets
        grd7 : SvL  $\in$  t_targets
    then
        act1 : EBD := l_curve
    end
Event set_SBD  $\hat{=}$ 
    any
        l_curve
    where
        grd1 : l_curve  $\in$  t_braking_curves
        grd2 : A_safe  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd3 : A_expected  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd4 : A_normal_service  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd5 : LOA  $\in$  t_targets
        grd6 : EOA  $\in$  t_targets
        grd7 : SvL  $\in$  t_targets
    then
        act1 : SBD := l_curve
    end
Event set_GUI  $\hat{=}$ 
    any
        l_curve
    where
        grd1 : l_curve  $\in$  t_braking_curves
        grd2 : A_safe  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd3 : A_expected  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd4 : A_normal_service  $\in$  t_speed  $\times$  t_distance  $\rightarrow$  t_acceleration
        grd5 : LOA  $\in$  t_targets
        grd6 : EOA  $\in$  t_targets
        grd7 : SvL  $\in$  t_targets
    then
        act1 : GUI := l_curve
    end
END

```

4.14 Context 6 - Supervision Limits

This context adds the type *t_supervision_limits* to the model, as well as a constant value of that type.

```

CONTEXT c6_supervision_limits
EXTENDS c5_braking_curves
SETS

```

```

    t_supervision_limits
CONSTANTS

```

```

    c_slimit0
AXIOMS

    axm1 : c_slimit0 ∈ t_supervision_limits
END

```

4.15 Machine 8 - Supervision Limit

This machine refinement adds the supervision limits to the model, emergency brake intervention (*EBI*), service brake intervention (*SBI*), warning limit (*warning_limit*), permitted speed (*P_limit*), indication limit (*I_limit*), pre-indication location (*PI_limit*) and the release start speed monitoring location (*RSM_start*).

```

MACHINE m8_supervision_limits
REFINES m7_braking_curves
SEES c6_supervision_limits
VARIABLES

    EBI emergency brake intervention
    SBI service brake intervention
    W_limit warning limit
    P_limit permitted speed
    I_limit indication limit
    PI pre-indication_location
    RSM_start release speed monitoring start location
EVENTS
Event set_EBI ≡
    any
    where
        I_limit
        grd1 : I_limit ∈ t_supervision_limits
        grd2 : MRS P ∈ t_speed_profiles
        grd3 : EBD ∈ t_braking_curves
        grd4 : SBD ∈ t_braking_curves
        grd5 : GUI ∈ t_braking_curves
        grd6 : T_bs ∈ t_time
        grd7 : T_be ∈ t_time
    then
        act1 : EBI := I_limit
    end
Event set_SBI ≡
    any
    where
        I_limit
        grd1 : I_limit ∈ t_supervision_limits
        grd2 : MRS P ∈ t_speed_profiles
        grd3 : EBD ∈ t_braking_curves
        grd4 : SBD ∈ t_braking_curves
        grd5 : GUI ∈ t_braking_curves
        grd6 : T_bs ∈ t_time
        grd7 : T_be ∈ t_time
    then
        act1 : SBI := I_limit
    end
Event set_W_limit ≡
    any
    where
        I_limit

```

```

        grd1 :  $l\_limit \in t\_supervision\_limits$ 
        grd2 :  $MRS P \in t\_speed\_profiles$ 
        grd3 :  $EBD \in t\_braking\_curves$ 
        grd4 :  $SBD \in t\_braking\_curves$ 
        grd5 :  $GUI \in t\_braking\_curves$ 
        grd6 :  $T\_bs \in t\_time$ 
        grd7 :  $T\_be \in t\_time$ 
    then
        act1 :  $W\_limit := l\_limit$ 
    end
Event set_I_limit  $\hat{=}$ 
    any
        where
             $l\_limit$ 
            grd1 :  $l\_limit \in t\_supervision\_limits$ 
            grd2 :  $MRS P \in t\_speed\_profiles$ 
            grd3 :  $EBD \in t\_braking\_curves$ 
            grd4 :  $SBD \in t\_braking\_curves$ 
            grd5 :  $GUI \in t\_braking\_curves$ 
            grd6 :  $T\_bs \in t\_time$ 
            grd7 :  $T\_be \in t\_time$ 
        then
            act1 :  $I\_limit := l\_limit$ 
        end
Event set_P_limit  $\hat{=}$ 
    any
        where
             $l\_limit$ 
            grd1 :  $l\_limit \in t\_supervision\_limits$ 
            grd2 :  $MRS P \in t\_speed\_profiles$ 
            grd3 :  $EBD \in t\_braking\_curves$ 
            grd4 :  $SBD \in t\_braking\_curves$ 
            grd5 :  $GUI \in t\_braking\_curves$ 
            grd6 :  $T\_bs \in t\_time$ 
            grd7 :  $T\_be \in t\_time$ 
        then
            act1 :  $P\_limit := l\_limit$ 
        end
Event set_PIl_limit  $\hat{=}$ 
    any
        where
             $l\_limit$ 
            grd1 :  $l\_limit \in t\_supervision\_limits$ 
            grd2 :  $MRS P \in t\_speed\_profiles$ 
            grd3 :  $EBD \in t\_braking\_curves$ 
            grd4 :  $SBD \in t\_braking\_curves$ 
            grd5 :  $GUI \in t\_braking\_curves$ 
            grd6 :  $T\_bs \in t\_time$ 
            grd7 :  $T\_be \in t\_time$ 
        then
            act1 :  $PIl := l\_limit$ 
        end
Event set_RSM_start_limit  $\hat{=}$ 
    any
        where
             $l\_limit$ 
            grd1 :  $l\_limit \in t\_supervision\_limits$ 
            grd2 :  $MRS P \in t\_speed\_profiles$ 

```

```

    grd3 :  $EBD \in t\_braking\_curves$ 
    grd4 :  $SBD \in t\_braking\_curves$ 
    grd5 :  $GUI \in t\_braking\_curves$ 
    grd6 :  $T\_bs \in t\_time$ 
    grd7 :  $T\_be \in t\_time$ 
  then
    act1 :  $RS\ M\_start := l\_limit$ 
  end
END

```

5 Model Decomposition

The decomposition strategy chosen for this model is “A-style” (for Abrial) which means shared variable decomposition [SPHB11]. In this case the events of an Event-B machine are separated into n disjoint sets. Each of these sets represents one decomposed sub-machine. Variables can be shared between machines, if they are read / written by them. In this case-study, all events that write a single variable are in one machine, events that read this variable are in another machine. Only the sub-machine in which a variable is written can do data-refinement on these variables, in the other machines, these variables are marked as shared and cannot be refined.

For the model decomposition, an additional refinement of the model is necessary. The machine m9 does not really add detail to the refined machine m8. The only changes are that the variables which are read by an event are explicitly added to the conditions by specifying a typing condition for them. This assures that the model decomposition preserves these necessary variables in the sub-machines and only removes the unneeded ones.

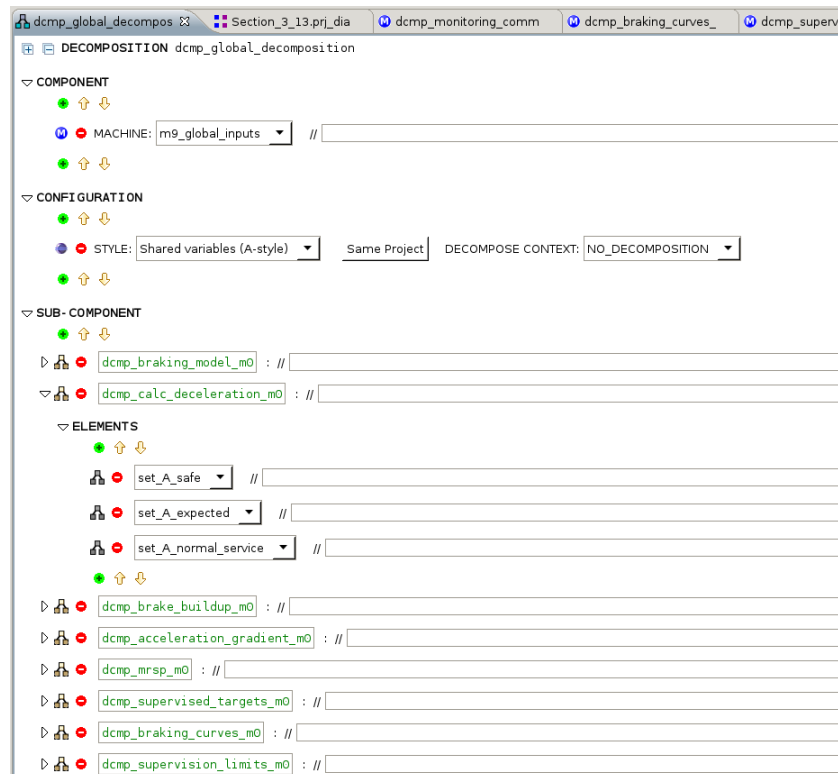


Figure 4. Decomposition Configuration

The overview of the decomposition in the Rodin tool using the decomposition plug-in is shown in Fig. 4. It lists the decomposed machine, the decomposition style, the sub-components and the events in each sub-machine (only shown for a single one).

In the following sections, the sub-components, their respective refinement and the required context definitions will be explained.

5.1 dcmp - Braking Model

The braking model has the following input variables: *a_current*, *v_current* and *loc_current*. These variables are written by the event *update_train_state*. The variables and this event are marked as “DO NOT REFINE”, i.e., they are shared variables and an external event.

The output variables are *T_brake_service*, *T_brake_emergency*, *A_brake_normal_service*, *A_brake_service* and *A_brake_emergency*.

MACHINE dcmp_braking_model_m0

SEES c6_supervision_limits

VARIABLES

A_brake_normal_service Private variable
A_brake_service Private variable
a_current Shared variable, DO NOT REFINE
v_current Shared variable, DO NOT REFINE
T_brake_service Shared variable, DO NOT REFINE
A_brake_emergency Private variable
loc_current Shared variable, DO NOT REFINE
T_brake_emergency Shared variable, DO NOT REFINE

EVENTS

Event *update_train_state* $\hat{=}$
 External event, DO NOT REFINE

any

l_speed
l_accel
l_loc

where

grd1 : *l_speed* \in *t_speed*
grd2 : *l_accel* \in *t_acceleration*
grd3 : *l_loc* \in *t_locations*

then

act1 : *v_current* := *l_speed*
act2 : *a_current* := *l_accel*
act3 : *loc_current* := *l_loc*

end

Event *set_A_brake_emergency* $\hat{=}$

any

l_a_brake

where

grd1 : *l_a_brake* \in *t_speed* \rightarrow *t_acceleration*

then

act1 : *A_brake_emergency* := *l_a_brake*

end

Event *set_A_brake_service* $\hat{=}$

any

l_a_brake

where

grd1 : *l_a_brake* \in *t_speed* \rightarrow *t_acceleration*

then

act1 : *A_brake_service* := *l_a_brake*

end

Event *set_A_brake_normal_service* $\hat{=}$

any

l_a_brake

where

grd1 : *l_a_brake* \in *t_speed* \rightarrow *t_acceleration*

```

    then
        act1 : A_brake_normal_service := l_a_brake
    end
Event set_T_brake_service ≡
    any
        l_T_brake
    where
        grd1 : l_T_brake ∈ t_time
    then
        act1 : T_brake_service := l_T_brake
    end
Event set_T_brake_emergency ≡
    any
        l_T_brake
    where
        grd1 : l_T_brake ∈ t_time
    then
        act1 : T_brake_emergency := l_T_brake
    end
END

```

5.1.1 Context - Train Data 1 Brake

This context introduces the type *t_brake_percentage* and a function that takes the train speed, the brake percentage, the train length and the brake position as arguments and returns a Boolean value indicating whether the conversion model is applicable. It also introduces functions that calculate the conversion model for the emergency and for the service brake, as well as the conversion model for the brake buildup time of the service and emergency brake.

On the system level description of this Event-B model, these functions are not implemented in more detail. Their specification and implementation details can be found in §3.13.3 of the SRS.

CONTEXT ctx_train_data_1_brake

EXTENDS ctx_train_data

SETS

t_brake_position

CONSTANTS

t_brake_percentage

c_brake_percentage0

f_conversion_model_applicable

f_conversion_model_A_emergency

f_conversion_model_A_service

f_conversion_model_T_brake_service

f_conversion_model_T_brake_emergency

AXIOMS

axm1 : *t_brake_percentage* ⊆ *t_percentage*

axm2 : *c_brake_percentage0* ∈ *t_brake_percentage*

axm3 : *f_conversion_model_applicable* ∈ *t_speed* × *t_brake_percentage* × *t_length* × *t_brake_position* → *BOOL*

cf. 3.13.3.2.1

axm4 : *f_conversion_model_A_emergency* ∈ *t_brake_percentage* → (*t_speed* → *t_acceleration*)

cf. 3.13.3.3

axm5 : *f_conversion_model_A_service* ∈ *t_brake_percentage* → (*t_speed* → *t_acceleration*)

axm6 : $f_conversion_model_T_brake_service \in t_brake_position \times t_length \times t_speed \rightarrow t_time$

cf. 3.13.3.4

axm7 : $f_conversion_model_T_brake_emergency \in t_brake_position \times t_length \times t_speed \rightarrow t_time$

END

5.1.2 dcmp - Braking Model Conversion Model

In the first refinement of the sub-machine adds the variables *brake_percentage* of type brake percentage and *brake_percentage_via_train_data* of Boolean type. These variables are modified by the event *specify_brake_percentage* and *remove_brake_percentage_data*. The Boolean variable signals the specification of the brake percentage via train data.

The value of the brake percentage is used in some events to compute concrete values for the conversion model instead of using an event parameter. For example in the event *calc_A_brake_emergency_conversion_model*, the function *f_conversion_model_A_emergency* is used with the value of *brake_percentage* to compute the conversion model. This replaces the event parameter *l_a_brake* by defining a *witness* for it.

MACHINE dcmp_braking_model_m1_conversion_model

REFINES dcmp_braking_model_m0

SEES c6_supervision_limits, ctx_train_data_l_brake

VARIABLES

brake_percentage

brake_percentage_via_train_data

EVENTS

Event *calc_A_brake_emergency_conversion_model* $\hat{=}$

refines *set_A_brake_emergency*

any

l_brake_position

where

grd2 : *brake_percentage_via_train_data* = TRUE

grd3 : $f_conversion_model_applicable(c_train_max_speed \mapsto brake_percentage \mapsto c_train_length \mapsto l_brake_position) = TRUE$

grd4 : *l_brake_position* $\in t_brake_position$

with

l_a_brake : *l_a_brake* = *f_conversion_model_A_emergency*(*brake_percentage*)

then

act1 : *A_brake_emergency* := *f_conversion_model_A_emergency*(*brake_percentage*)

end

Event *calc_A_brake_service_conversion_model* $\hat{=}$

refines *set_A_brake_service*

any

l_brake_position

where

grd2 : *brake_percentage_via_train_data* = TRUE

grd3 : $f_conversion_model_applicable(c_train_max_speed \mapsto brake_percentage \mapsto c_train_length \mapsto l_brake_position) = TRUE$

grd4 : *l_brake_position* $\in t_brake_position$

with

l_a_brake : *l_a_brake* = *f_conversion_model_A_service*(*brake_percentage*)

then

act1 : *A_brake_service* := *f_conversion_model_A_service*(*brake_percentage*)

end

Event *calc_T_brake_service_conversion_model* $\hat{=}$

refines *set_T_brake_service*

any


```

    l_brake_position
    l_target_speed
  where
    grd2 : brake_percentage_via_train_data = TRUE
    grd3 : f_conversion_model_applicable(c_train_max_speed  $\mapsto$  brake_percentage  $\mapsto$  c_train_length  $\mapsto$ 
      l_brake_position) = TRUE
    grd4 : l_brake_position  $\in$  t_brake_position
    grd5 : l_target_speed  $\in$  t_speed
  with
    l_T_brake : l_T_brake = f_conversion_model_T_brake_service(l_brake_position  $\mapsto$  c_train_length  $\mapsto$ 
      l_target_speed)
  then
    act1 : T_brake_service := f_conversion_model_T_brake_service(l_brake_position  $\mapsto$  c_train_length  $\mapsto$ 
      l_target_speed)
  end
refines set_T_brake_emergency
  any
    l_brake_position
    l_target_speed
  where
    grd1 : brake_percentage_via_train_data = TRUE
    grd2 : f_conversion_model_applicable(c_train_max_speed  $\mapsto$  brake_percentage  $\mapsto$  c_train_length  $\mapsto$ 
      l_brake_position) = TRUE
    grd3 : l_brake_position  $\in$  t_brake_position
    grd4 : l_target_speed  $\in$  t_speed
  with
    l_T_brake : l_T_brake = f_conversion_model_T_brake_emergency(l_brake_position  $\mapsto$  c_train_length  $\mapsto$ 
      l_target_speed)
  then
    act1 : T_brake_emergency := f_conversion_model_T_brake_emergency(l_brake_position  $\mapsto$ 
      c_train_length  $\mapsto$  l_target_speed)
  end
Event specify_brake_percentage  $\hat{=}$ 
  any
    l_brake
  where
    grd1 : l_brake  $\in$  t_brake_percentage
    grd2 : brake_percentage_via_train_data = FALSE
  then
    act1 : brake_percentage := l_brake
    act2 : brake_percentage_via_train_data := TRUE
  end
Event remove_brake_percentage_data  $\hat{=}$ 
  when
    then
      grd1 : brake_percentage_via_train_data = TRUE
    then
      act1 : brake_percentage := c_brake_percentage0
      act2 : brake_percentage_via_train_data := FALSE
    end
  end
END

```

5.1.3 Context - Train Data 2 Direct Calculation

The next context introduces the type representing the combination of the different brake types of the train. It also defines functions that calculate the braking models for different brake combinations for the service brake and the emergency brake, as well as functions to calculate the brake buildup time for the service and emergency brake with the brake combination as argument. It also defines a function that calculates the normal brake function dependent on the brake position and the acceleration constants c_{SB01} and c_{SB02} (cf. §3.13.2.2.3.1.10).

CONTEXT ctx_train_data_2_dircect_calc

EXTENDS ctx_train_data_1_brake

SETS

$t_brake_combination$ cf. 3.13.2.2.3.1.7, i.e., combination of regenerative, eddy current and magnetic brake

CONSTANTS

c_v_zero target speed zero

$f_calc_A_brake_service_direct$

$f_calc_A_brake_emergency_direct$

$f_calc_A_brake_normal_direct$ cf. 3.13.2.2.3.1.9

c_SB01

c_SB02

$f_calc_T_brake_service$

$f_calc_T_brake_emergency$

AXIOMS

axm1 : $c_v_zero \in t_speed$

axm2 : $f_calc_A_brake_service_direct \in t_brake_combination \rightarrow (t_speed \rightarrow t_acceleration)$

axm3 : $f_calc_A_brake_emergency_direct \in t_brake_combination \rightarrow (t_speed \rightarrow t_acceleration)$

axm4 : $f_calc_A_brake_normal_direct \in t_brake_position \times t_acceleration \times t_acceleration \rightarrow (t_speed \rightarrow t_acceleration)$

cf. 3.13.2.2.3.1.10

the accelerations are the c_SB01 and c_SB02 constants

axm5 : $c_SB01 \in t_acceleration$

axm6 : $c_SB02 \in t_acceleration$

axm7 : $f_calc_T_brake_service \in t_brake_combination \rightarrow t_time$

axm8 : $f_calc_T_brake_emergency \in t_brake_combination \rightarrow t_time$

END

5.1.4 dcmp - Braking Model Direct Calculation

This machine refinement uses the brake combinations and the corresponding functions from the context to produce more concrete events. For events that compute a function mapping speed to acceleration, the functions of the context and the constants for SB01 and SB02 are used to eliminate the event parameters by providing witnesses for them.

MACHINE dcmp_braking_model_m2_direct_calc

REFINES dcmp_braking_model_m1_conversion_model

SEES c6_supervision_limits, ctx_train_data_2_dircect_calc

EVENTS

Event $calc_A_brake_emergency_direct \hat{=}$

refines $set_A_brake_emergency$

any

$l_brake_combination$

where

grd1 : $l_brake_combination \in t_brake_combination$

with

l_a_brake : $l_a_brake = f_calc_A_brake_emergency_direct(l_brake_combination)$

then

act1 : $A_brake_emergency := f_calc_A_brake_emergency_direct(l_brake_combination)$

end

Event $calc_A_brake_service_direct \hat{=}$

refines $set_A_brake_service$

any

$l_brake_combination$

where

```

    with      grd2 : l_brake_combination ∈ t_brake_combination

    then      l_a_brake : l_a_brake = f_calc_A_brake_service_direct(l_brake_combination)

    end       act1 : A_brake_service := f_calc_A_brake_service_direct(l_brake_combination)
  end
Event set_A_brake_normal_service ≡
refines set_A_brake_normal_service
  any
    l_brake_position
  where
    with      grd1 : l_brake_position ∈ t_brake_position

    then      l_a_brake : l_a_brake = f_calc_A_brake_normal_direct(l_brake_position ↦ c_S B01 ↦ c_S B02)

    end       act1 : A_brake_normal_service := f_calc_A_brake_normal_direct(l_brake_position ↦ c_S B01 ↦ c_S B02)
  end
Event set_T_brake_service ≡
refines set_T_brake_service
  any
    l_brake_combination
  where
    with      grd1 : l_brake_combination ∈ t_brake_combination

    then      l_T_brake : l_T_brake = f_calc_T_brake_service(l_brake_combination)

    end       act1 : T_brake_service := f_calc_T_brake_service(l_brake_combination)
  end
Event set_T_brake_emergency ≡
refines set_T_brake_emergency
  any
    l_brake_combination
  where
    with      grd1 : l_brake_combination ∈ t_brake_combination

    then      l_T_brake : l_T_brake = f_calc_T_brake_emergency(l_brake_combination)

    end       act1 : T_brake_emergency := f_calc_T_brake_emergency(l_brake_combination)
  end
END

```

5.2 dcmp - Calc Deceleration

The deceleration calculation model has the following input variables: *a_current*, *v_current* and *loc_current*. The output variables it calculates are *A_normal_service*, *A_expected* and *A_safe* which each map train speed and distance to an acceleration.

```

MACHINE dcmp_calc_deceleration_m0
SEES c6_supervision_limits
EVENTS
Event set_A_safe ≡
  any
    l_a_decel
  where

```

```

    then      grd1 :  $l\_a\_decel \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
  end
    act1 :  $A\_safe := l\_a\_decel$ 
end
Event set_A_expected  $\hat{=}$ 
  any
    where
       $l\_a\_decel$ 
    then      grd1 :  $l\_a\_decel \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
    end
    act1 :  $A\_expected := l\_a\_decel$ 
end
Event set_A_normal_service  $\hat{=}$ 
  any
    where
       $l\_a\_decel$ 
    then      grd1 :  $l\_a\_decel \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
    end
    act1 :  $A\_normal\_service := l\_a\_decel$ 
end
END

```

5.3 dcmp - Brake Buildup

The brake buildup time calculation has the following input variables: $a_current$, $v_current$ and $loc_current$. The output variables are T_be , T_se , $T_brake_service$ and $T_brake_emergency$.

MACHINE dcmp_brake_buildup_m0
SEES c6_supervision_limits
VARIABLES

T_be Shared variable, DO NOT REFIN
 T_bs Shared variable, DO NOT REFIN
 $a_current$ Shared variable, DO NOT REFIN
 $v_current$ Shared variable, DO NOT REFIN
 $T_brake_service$ Shared variable, DO NOT REFIN
 $loc_current$ Shared variable, DO NOT REFIN
 $T_brake_emergency$ Shared variable, DO NOT REFIN

INVARIANTS

typing_T_be : $T_be \in t_time$
 typing_T_bs : $T_bs \in t_time$
 typing_a_current : $a_current \in t_acceleration$
 typing_v_current : $v_current \in t_speed$
 typing_T_brake_service : $T_brake_service \in t_time$
 typing_loc_current : $loc_current \in t_locations$
 typing_T_brake_emergency : $T_brake_emergency \in t_time$

EVENTS

Initialisation

```

begin
  act1 :  $v\_current := c\_v0$ 
  act2 :  $a\_current := c\_a0$ 
  act3 :  $loc\_current := c\_l0$ 
  act9 :  $T\_brake\_service := c\_T\_brake0$ 
  act10 :  $T\_brake\_emergency := c\_T\_brake0$ 
  act14 :  $T\_be := c\_T\_brake0$ 

```

```

    act15 : T_bs := c_T_brake0
end
Event update_train_state ≡
    External event, DO NOT REFINE
    any
        l_speed
        l_accel
        l_loc
    where
        grd1 : l_speed ∈ t_speed
        grd2 : l_accel ∈ t_acceleration
        grd3 : l_loc ∈ t_locations
    then
        act1 : v_current := l_speed
        act2 : a_current := l_accel
        act3 : loc_current := l_loc
    end
Event set_T_brake_service ≡
    External event, DO NOT REFINE
    any
        l_T_brake
    where
        grd1 : l_T_brake ∈ t_time
    then
        act1 : T_brake_service := l_T_brake
    end
Event set_T_brake_emergency ≡
    External event, DO NOT REFINE
    any
        l_T_brake
    where
        grd1 : l_T_brake ∈ t_time
    then
        act1 : T_brake_emergency := l_T_brake
    end
Event set_T_be ≡
    any
        l_t_be
    where
        grd1 : l_t_be ∈ t_time
    then
        act1 : T_be := l_t_be
    end
Event set_T_bs ≡
    any
        l_t_bs
    where
        grd1 : l_t_bs ∈ t_time
    then
        act1 : T_bs := l_t_bs
    end
END

MACHINE dcmp_brake_buildup_m2_calculation
REFINES dcmp_brake_buildup_m0

```

SEES c6_supervision_limits, ctx_functions_l_calc_T_brake

VARIABLES

T_be Shared variable, DO NOT REFINE
T_bs Shared variable, DO NOT REFINE
a_current Shared variable, DO NOT REFINE
v_current Shared variable, DO NOT REFINE
T_brake_service Shared variable, DO NOT REFINE
loc_current Shared variable, DO NOT REFINE
T_brake_emergency Shared variable, DO NOT REFINE
T_brake_service_data
T_brake_service_via_train_data
T_brake_emergency_data
T_brake_emergency_via_train_data

INVARIANTS

inv1 : *T_brake_service_data* ∈ *t_time*
inv2 : *T_brake_service_via_train_data* ∈ *BOOL*
inv3 : *T_brake_emergency_data* ∈ *t_time*
inv4 : *T_brake_emergency_via_train_data* ∈ *BOOL*

EVENTS

Initialisation

extended

begin

act1 : *v_current* := *c_v0*
act2 : *a_current* := *c_a0*
act3 : *loc_current* := *c_l0*
act9 : *T_brake_service* := *c_T_brake0*
act10 : *T_brake_emergency* := *c_T_brake0*
act14 : *T_be* := *c_T_brake0*
act15 : *T_bs* := *c_T_brake0*
act16 : *T_brake_service_data* := *c_T_brake0*
act17 : *T_brake_service_via_train_data* := *FALSE*
act18 : *T_brake_emergency_data* := *c_T_brake0*
act19 : *T_brake_emergency_via_train_data* := *FALSE*

end

Event *update_train_state* ≡

External event, DO NOT REFINE

extends *update_train_state*

any

l_speed

l_accel

l_loc

where

grd1 : *l_speed* ∈ *t_speed*
grd2 : *l_accel* ∈ *t_acceleration*
grd3 : *l_loc* ∈ *t_locations*

then

act1 : *v_current* := *l_speed*
act2 : *a_current* := *l_accel*
act3 : *loc_current* := *l_loc*

end

Event *set_T_brake_service* ≡

External event, DO NOT REFINE

extends *set_T_brake_service*

any

l_T_brake

where

grd1 : *l_T_brake* ∈ *t_time*

```

    then
        act1 : T_brake_service := l_T_brake
    end
Event set_T_brake_emergency  $\hat{=}$ 
    External event, DO NOT REFINE
extends set_T_brake_emergency
    any
        where
            l_T_brake
        then
            grd1 : l_T_brake  $\in$  t_time
        then
            act1 : T_brake_emergency := l_T_brake
        end
Event set_T_be  $\hat{=}$ 
refines set_T_be
    any
        where
            l_brake_combination
        then
            grd1 : T_brake_emergency_via_train_data = TRUE
            grd2 : l_brake_combination  $\in$  t_brake_combination
        with
        then
            l_t_be : l_t_be = f_T_brake_emergency(T_brake_emergency  $\mapsto$  l_brake_combination)
        then
            act1 : T_be := f_T_brake_emergency(T_brake_emergency  $\mapsto$  l_brake_combination)
        end
Event calc_T_be_conversion_model  $\hat{=}$ 
refines set_T_brake_emergency
    when
        with
            grd1 : T_brake_emergency_via_train_data = FALSE
        then
            l_T_brake : l_T_brake = f_T_brake_emergency_conversion(T_brake_service  $\mapsto$  c_Kt_int)
        then
            act1 : T_brake_emergency := f_T_brake_emergency_conversion(T_brake_service  $\mapsto$  c_Kt_int)
        end
Event set_T_bs  $\hat{=}$ 
refines set_T_bs
    any
        where
            l_brake_combination
        then
            grd1 : T_brake_service_via_train_data = TRUE
            grd2 : l_brake_combination  $\in$  t_brake_combination
        with
        then
            l_t_bs : l_t_bs = f_T_brake_service(T_brake_service_data  $\mapsto$  l_brake_combination)
        then
            act1 : T_bs := f_T_brake_service(T_brake_service_data  $\mapsto$  l_brake_combination)
        end
Event calc_T_bs_conversion_model  $\hat{=}$ 
refines set_T_bs
    any
        where
            l_brake_combination
        then
            grd1 : T_brake_service_via_train_data = FALSE
            grd2 : l_brake_combination  $\in$  t_brake_combination
        with
        then
            l_t_bs : l_t_bs = f_T_brake_emergency(T_brake_service  $\mapsto$  l_brake_combination)

```

```

    then
        act1 :  $T_{bs} := f_{T\_brake\_emergency}(T\_brake\_service \mapsto l\_brake\_combination)$ 
    end
Event specify_T_brake_service_train_data  $\hat{=}$ 
    any
        where
            l_T_brake
        then
            grd1 :  $l\_T\_brake \in t\_time$ 
            grd2 :  $T\_brake\_service\_via\_train\_data = FALSE$ 
        then
            act1 :  $T\_brake\_service\_data := l\_T\_brake$ 
            act2 :  $T\_brake\_service\_via\_train\_data := TRUE$ 
        end
Event remove_T_brake_service_data  $\hat{=}$ 
    when
        then
            grd1 :  $T\_brake\_service\_via\_train\_data = TRUE$ 
        then
            act1 :  $T\_brake\_service\_via\_train\_data := FALSE$ 
        end
Event specify_T_brake_emergency_train_data  $\hat{=}$ 
    any
        where
            l_T_brake
        then
            grd1 :  $l\_T\_brake \in t\_time$ 
            grd2 :  $T\_brake\_emergency\_via\_train\_data = FALSE$ 
        then
            act1 :  $T\_brake\_emergency\_data := l\_T\_brake$ 
            act2 :  $T\_brake\_emergency\_via\_train\_data := TRUE$ 
        end
Event remove_T_brake_emergency_data  $\hat{=}$ 
    when
        then
            grd1 :  $T\_brake\_emergency\_via\_train\_data = TRUE$ 
        then
            act1 :  $T\_brake\_emergency\_via\_train\_data := FALSE$ 
        end
END

```

5.4 dcmp - Acceleration Gradient

MACHINE dcmp_acceleration_gradient_m0

SEES c6_supervision_limits

VARIABLES

a_current Shared variable, DO NOT REFINE
v_current Shared variable, DO NOT REFINE
loc_current Shared variable, DO NOT REFINE
A_gradient Private variable

INVARIANTS

typing_a_current : $a_current \in t_acceleration$
 typing_v_current : $v_current \in t_speed$
 typing_loc_current : $loc_current \in t_locations$
 typing_A_gradient : $A_gradient \in t_acceleration$
 m4_accel_decel_gradient_inv1 : $A_gradient \in t_acceleration$

EVENTS


```

Initialisation
  begin
    act1 : v_current := c_v0
    act2 : a_current := c_a0
    act3 : loc_current := c_l0
    act16 : A_gradient := c_a0
  end
Event update_train_state  $\hat{=}$ 
  External event, DO NOT REFINE

  any
    l_speed
    l_accel
    l_loc
  where
    grd1 : l_speed  $\in$  t_speed
    grd2 : l_accel  $\in$  t_acceleration
    grd3 : l_loc  $\in$  t_locations
  then
    act1 : v_current := l_speed
    act2 : a_current := l_accel
    act3 : loc_current := l_loc
  end
Event set_A_gradient  $\hat{=}$ 
  any
    l_a_gradient
  where
    grd1 : l_a_gradient  $\in$  t_acceleration
  then
    act1 : A_gradient := l_a_gradient
  end
END

```

```

MACHINE dcmp_acceleration_gradient_m1
REFINES dcmp_acceleration_gradient_m0
SEES c6_supervision_limits, ctx_functions_0
VARIABLES

```

```

  a_current  Shared variable, DO NOT REFINE
  v_current  Shared variable, DO NOT REFINE
  loc_current Shared variable, DO NOT REFINE
  A_gradient Private variable
  rotating_mass_specified
  M_rotating_nom

```

INVARIANTS

```

  inv1 : rotating_mass_specified  $\in$  BOOL
  inv2 : M_rotating_nom  $\in$  t_mass_percentage

```

EVENTS

```

Initialisation
  extended
  begin
    act1 : v_current := c_v0
    act2 : a_current := c_a0
    act3 : loc_current := c_l0
    act16 : A_gradient := c_a0
    act17 : rotating_mass_specified := FALSE
    act18 : M_rotating_nom := c_mass_percentage0
  end

```

```

Event update_train_state  $\hat{=}$ 
    External event, DO NOT REFINE
extends update_train_state
    any
        l_speed
        l_accel
        l_loc
    where
        grd1 : l_speed  $\in$  t_speed
        grd2 : l_accel  $\in$  t_acceleration
        grd3 : l_loc  $\in$  t_locations
    then
        act1 : v_current := l_speed
        act2 : a_current := l_accel
        act3 : loc_current := l_loc
    end
Event set_A_gradient  $\hat{=}$ 
refines set_A_gradient
    any
        l_mass
        l_SvL
        l_grad
        l_TSR
    where
        grd2 : l_mass  $\in$  t_mass_percentage
        grd3 : l_SvL  $\in$  t_locations
        grd4 : l_grad  $\in$  t_gradients
        grd5 : l_TSR  $\in$  t_speed_profiles
    with
        l_a_gradient : l_a_gradient = f_accel_due_gradient(loc_current  $\mapsto$  l_SvL  $\mapsto$  l_grad  $\mapsto$  l_mass  $\mapsto$  l_TSR)
    then
        act1 : A_gradient := f_accel_due_gradient(loc_current  $\mapsto$  l_SvL  $\mapsto$  l_grad  $\mapsto$  l_mass  $\mapsto$  l_TSR)
    end
Event specify_rotating_mass  $\hat{=}$ 
    any
        l_mass
    where
        grd1 : rotating_mass_specified = FALSE
        grd2 : l_mass  $\in$  t_mass_percentage
    then
        act1 : M_rotating_nom := l_mass
        act2 : rotating_mass_specified := TRUE
    end
Event delete_rotating_mass_data  $\hat{=}$ 
    when
        grd1 : rotating_mass_specified = TRUE
    then
        act1 : rotating_mass_specified := FALSE
        act2 : M_rotating_nom := c_mass_percentage0
    end
END

```

```

MACHINE dcmp_acceleration_gradient_m2_grad_accel
REFINES dcmp_acceleration_gradient_m1
SEES c6_supervision_limits, ctx_functions_0
VARIABLES

```

```

    a_current  Shared variable, DO NOT REFINE
    v_current  Shared variable, DO NOT REFINE
    loc_current Shared variable, DO NOT REFINE
    A_gradient Private variable
    rotating_mass_specified
    M_rotating_nom
EVENTS
Initialisation
    extended
    begin
        act1 : v_current := c_v0
        act2 : a_current := c_a0
        act3 : loc_current := c_l0
        act16 : A_gradient := c_a0
        act17 : rotating_mass_specified := FALSE
        act18 : M_rotating_nom := c_mass_percentage0
    end
Event update_train_state  $\hat{=}$ 
    External event, DO NOT REFINE
extends update_train_state
    any
        l_speed
        l_accel
        l_loc
    where
        grd1 : l_speed  $\in$  t_speed
        grd2 : l_accel  $\in$  t_acceleration
        grd3 : l_loc  $\in$  t_locations
    then
        act1 : v_current := l_speed
        act2 : a_current := l_accel
        act3 : loc_current := l_loc
    end
Event calc_A_gradient_mass_known  $\hat{=}$ 
refines set_A_gradient
    any
        l_SvL
        l_grad
        l_TSR
    where
        grd3 : l_SvL  $\in$  t_locations
        grd4 : l_grad  $\in$  t_gradients
        grd5 : l_TSR  $\in$  t_speed_profiles
        grd6 : rotating_mass_specified = TRUE
    with
        l_mass : l_mass = M_rotating_nom
    then
        act1 : A_gradient := f_accel_due_gradient(loc_current  $\mapsto$  l_SvL  $\mapsto$  l_grad  $\mapsto$  M_rotating_nom  $\mapsto$  l_TSR)
    end
Event calc_A_gradient_mass_unknown_uphill  $\hat{=}$ 
refines set_A_gradient
    any
        l_SvL
        l_grad
        l_TSR
    where
        grd3 : l_SvL  $\in$  t_locations
        grd4 : l_grad  $\in$  t_gradients

```

```

    grd5 : l_TSR ∈ t_speed_profiles
    grd6 : f_grad_uphill(l_grad) = TRUE
    grd7 : rotating_mass_specified = FALSE
with
    l_mass : l_mass = c_M_rotating_max
then
    act1 : A_gradient := f_accel_due_gradient(loc_current ↦ l_SvL ↦ l_grad ↦ c_M_rotating_max ↦
l_TSR)
end
Event calc_A_gradient_mass_unknown_downhill ≡
refines set_A_gradient
any
    l_SvL
    l_grad
    l_TSR
where
    grd3 : l_SvL ∈ t_locations
    grd4 : l_grad ∈ t_gradients
    grd5 : l_TSR ∈ t_speed_profiles
    grd6 : f_grad_uphill(l_grad) = FALSE
    grd7 : rotating_mass_specified = FALSE
with
    l_mass : l_mass = c_M_rotating_min
then
    act1 : A_gradient := f_accel_due_gradient(loc_current ↦ l_SvL ↦ l_grad ↦ c_M_rotating_min ↦ l_TSR)
end
Event specify_rotating_mass ≡
extends specify_rotating_mass
any
    l_mass
where
    grd1 : rotating_mass_specified = FALSE
    grd2 : l_mass ∈ t_mass_percentage
then
    act1 : M_rotating_nom := l_mass
    act2 : rotating_mass_specified := TRUE
end
Event delete_rotating_mass_data ≡
extends delete_rotating_mass_data
when
    then
        grd1 : rotating_mass_specified = TRUE
    then
        act1 : rotating_mass_specified := FALSE
        act2 : M_rotating_nom := c_mass_percentage0
    end
END

```

5.5 dcmp - MRSP

MACHINE dcmp_mrsp_m0
SEES c6_supervision_limits
VARIABLES

MRS P Shared variable, DO NOT REFINE
a_current Shared variable, DO NOT REFINE
v_current Shared variable, DO NOT REFINE
loc_current Shared variable, DO NOT REFINE

INVARIANTS

`typing_MRSP` : $MRS P \in t_speed_profiles$
`typing_a_current` : $a_current \in t_acceleration$
`typing_v_current` : $v_current \in t_speed$
`typing_loc_current` : $loc_current \in t_locations$

EVENTS**Initialisation****begin**

`act1` : $v_current := c_v0$
`act2` : $a_current := c_a0$
`act3` : $loc_current := c_l0$
`act17` : $MRS P := c_speed_profile0$

end**Event** `update_train_state` $\hat{=}$

External event, DO NOT REFINE

any

l_speed
 l_accel
 l_loc

where

`grd1` : $l_speed \in t_speed$
`grd2` : $l_accel \in t_acceleration$
`grd3` : $l_loc \in t_locations$

then

`act1` : $v_current := l_speed$
`act2` : $a_current := l_accel$
`act3` : $loc_current := l_loc$

end**Event** `set_MRSP` $\hat{=}$ **any** l_sp **where**`grd1` : $l_sp \in t_speed_profiles$ **then**`act1` : $MRS P := l_sp$ **end****END****MACHINE** `dcmp_mrsp_m1_all_speed_profiles`**REFINES** `dcmp_mrsp_m0`**SEES** `c6_supervision_limits`, `ctx_functions_0`**VARIABLES**

$MRS P$ Shared variable, DO NOT REFINE
 $a_current$ Shared variable, DO NOT REFINE
 $v_current$ Shared variable, DO NOT REFINE
 $loc_current$ Shared variable, DO NOT REFINE

EVENTS**Initialisation***extended***begin**

`act1` : $v_current := c_v0$
`act2` : $a_current := c_a0$
`act3` : $loc_current := c_l0$
`act17` : $MRS P := c_speed_profile0$

end

```

Event update_train_state  $\hat{=}$ 
    External event, DO NOT REFINE
extends update_train_state
    any
        l_speed
        l_accel
        l_loc
    where
        grd1 : l_speed  $\in$  t_speed
        grd2 : l_accel  $\in$  t_acceleration
        grd3 : l_loc  $\in$  t_locations
    then
        act1 : v_current := l_speed
        act2 : a_current := l_accel
        act3 : loc_current := l_loc
    end
Event calculate_MRSP_from_all_speed_restrictions  $\hat{=}$ 
refines set_MRSP
    any
        l_SSP
        l_AxleLoadSP
        l_TSR
        l_SignalRelatedSP
        l_mode
        l_modeRelatedSP
        l_LXSP
        l_OverrideFuncSP
        l_ensureBrakingDistanceSP
    where
        grd2 : l_SSP  $\in$  t_speed_profiles
        grd3 : l_AxleLoadSP  $\in$  t_speed_profiles
        grd4 : l_TSR  $\in$  t_speed_profiles
        grd5 : l_SignalRelatedSP  $\in$  t_speed_profiles
        grd6 : l_mode  $\in$  t_train_modes
        grd7 : l_modeRelatedSP  $\in$  t_speed_profiles
        grd8 : l_LXSP  $\in$  t_speed_profiles
        grd9 : l_OverrideFuncSP  $\in$  t_speed_profiles
        grd10 : l_ensureBrakingDistanceSP  $\in$  t_speed_profiles
    with
        l_sp : l_sp = f_mrsp(l_SSP  $\mapsto$  l_AxleLoadSP  $\mapsto$  l_TSR  $\mapsto$  l_SignalRelatedSP
             $\mapsto$  l_modeRelatedSP  $\mapsto$  l_LXSP  $\mapsto$  l_OverrideFuncSP  $\mapsto$ 
            l_ensureBrakingDistanceSP
             $\mapsto$  l_mode  $\mapsto$  c_train_max_speed  $\mapsto$  c_train_length)
    then
        act1 : MRS P := f_mrsp(l_SSP  $\mapsto$  l_AxleLoadSP  $\mapsto$  l_TSR  $\mapsto$  l_SignalRelatedSP
             $\mapsto$  l_modeRelatedSP  $\mapsto$  l_LXSP  $\mapsto$  l_OverrideFuncSP  $\mapsto$ 
            l_ensureBrakingDistanceSP
             $\mapsto$  l_mode  $\mapsto$  c_train_max_speed  $\mapsto$  c_train_length)
    end
END

```

5.6 dcmp - Supervised Targets

MACHINE dcmp_supervised_targets_m0

SEES c6_supervision_limits

VARIABLES

SvL Shared variable, DO NOT REFINE

LOA Shared variable, DO NOT REFINE

EOA Shared variable, DO NOT REFINE
MRS P Shared variable, DO NOT REFINE
a_current Shared variable, DO NOT REFINE
v_current Shared variable, DO NOT REFINE
loc_current Shared variable, DO NOT REFINE

INVARIANTS

typing_SvL : $SvL \in t_locations \times t_speed$
typing_LOA : $LOA \in t_locations \times t_speed$
typing_EOA : $EOA \in t_locations \times t_speed$
typing_MRSP : $MRS P \in t_speed_profiles$
typing_a_current : $a_current \in t_acceleration$
typing_v_current : $v_current \in t_speed$
typing_loc_current : $loc_current \in t_locations$
m6_supervised_targets_inv1 : $LOA \in t_targets$
m6_supervised_targets_inv2 : $EOA \in t_targets$
m6_supervised_targets_inv3 : $SvL \in t_targets$

EVENTS

Initialisation

begin

act1 : $v_current := c_v0$
act2 : $a_current := c_a0$
act3 : $loc_current := c_l0$
act17 : $MRS P := c_speed_profile0$
act18 : $EOA := c_target0$
act19 : $LOA := c_target0$
act20 : $SvL := c_target0$

end

Event *update_train_state* $\hat{=}$

External event, DO NOT REFINE

any

l_speed
l_accel
l_loc

where

grd1 : $l_speed \in t_speed$
grd2 : $l_accel \in t_acceleration$
grd3 : $l_loc \in t_locations$

then

act1 : $v_current := l_speed$
act2 : $a_current := l_accel$
act3 : $loc_current := l_loc$

end

Event *set_MRSP* $\hat{=}$

External event, DO NOT REFINE

any

l_sp

where

grd1 : $l_sp \in t_speed_profiles$

then

act1 : $MRS P := l_sp$

end

Event *set_EOA* $\hat{=}$

any

l_target

where

```

    then      grd1 :  $l\_target \in t\_targets$ 
  end
    act1 :  $EOA := l\_target$ 
end
Event set_LOA  $\hat{=}$ 
  any
     $l\_target$ 
  where
    then      grd1 :  $l\_target \in t\_targets$ 
    end
    act1 :  $LOA := l\_target$ 
end
Event set_SvL  $\hat{=}$ 
  any
     $l\_target$ 
  where
    then      grd1 :  $l\_target \in t\_targets$ 
    end
    act1 :  $SvL := l\_target$ 
end
END

```

5.7 dcmp - Braking Curves

MACHINE dcmp_braking_curves_m0

SEES c6_supervision_limits

VARIABLES

A_normal_service Shared variable, DO NOT REFINE

SBD Shared variable, DO NOT REFINE

A_expected Shared variable, DO NOT REFINE

A_safe Shared variable, DO NOT REFINE

GUI Shared variable, DO NOT REFINE

LOA Shared variable, DO NOT REFINE

EOA Shared variable, DO NOT REFINE

SvL Shared variable, DO NOT REFINE

EBD Shared variable, DO NOT REFINE

a_current Shared variable, DO NOT REFINE

v_current Shared variable, DO NOT REFINE

loc_current Shared variable, DO NOT REFINE

INVARIANTS

typing_A_normal_service : $A_normal_service \in \mathbb{P}(t_speed \times t_distance \times t_acceleration)$

typing_SBD : $SBD \in t_braking_curves$

typing_A_expected : $A_expected \in \mathbb{P}(t_speed \times t_distance \times t_acceleration)$

typing_A_safe : $A_safe \in \mathbb{P}(t_speed \times t_distance \times t_acceleration)$

typing_GUI : $GUI \in t_braking_curves$

typing_LOA : $LOA \in t_locations \times t_speed$

typing_EOA : $EOA \in t_locations \times t_speed$

typing_SvL : $SvL \in t_locations \times t_speed$

typing_EBD : $EBD \in t_braking_curves$

typing_a_current : $a_current \in t_acceleration$

typing_v_current : $v_current \in t_speed$

typing_loc_current : $loc_current \in t_locations$

m2_calculation_deceleration_inv1 : $A_safe \in t_speed \times t_distance \rightarrow t_acceleration$


```

m2_calculation_deceleration_inv2 :  $A_{expected} \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
m2_calculation_deceleration_inv3 :  $A_{normal\_service} \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
m6_supervised_targets_inv1 :  $LOA \in t\_targets$ 
m6_supervised_targets_inv2 :  $EOA \in t\_targets$ 
m6_supervised_targets_inv3 :  $SvL \in t\_targets$ 
EVENTS
Initialisation
  begin
    act1 :  $v\_current := c\_v0$ 
    act2 :  $a\_current := c\_a0$ 
    act3 :  $loc\_current := c\_l0$ 
    act11 :  $A\_safe := f\_A\_deceleration0$ 
    act12 :  $A\_expected := f\_A\_deceleration0$ 
    act13 :  $A\_normal\_service := f\_A\_deceleration0$ 
    act18 :  $EOA := c\_target0$ 
    act19 :  $LOA := c\_target0$ 
    act20 :  $SvL := c\_target0$ 
    act21 :  $EBD := c\_curve0$ 
    act22 :  $SBD := c\_curve0$ 
    act23 :  $GUI := c\_curve0$ 
  end
Event update_train_state  $\hat{=}$ 
                                     External event, DO NOT REFINE
  any
    l_speed
    l_accel
    l_loc
  where
    grd1 :  $l\_speed \in t\_speed$ 
    grd2 :  $l\_accel \in t\_acceleration$ 
    grd3 :  $l\_loc \in t\_locations$ 
  then
    act1 :  $v\_current := l\_speed$ 
    act2 :  $a\_current := l\_accel$ 
    act3 :  $loc\_current := l\_loc$ 
  end
Event set_A_safe  $\hat{=}$ 
                                     External event, DO NOT REFINE
  any
  where l_a_decel
  then
    grd1 :  $l\_a\_decel \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
  then
    act1 :  $A\_safe := l\_a\_decel$ 
  end
Event set_A_expected  $\hat{=}$ 
                                     External event, DO NOT REFINE
  any
  where l_a_decel
  then
    grd1 :  $l\_a\_decel \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
  then
    act1 :  $A\_expected := l\_a\_decel$ 
  end
Event set_A_normal_service  $\hat{=}$ 
                                     External event, DO NOT REFINE
  any

```

```

    where  $l\_a\_decel$ 
  where
    grd1 :  $l\_a\_decel \in t\_speed \times t\_distance \rightarrow t\_acceleration$ 
  then
    act1 :  $A\_normal\_service := l\_a\_decel$ 
  end
Event set_EOA  $\hat{=}$ 
  External event, DO NOT REFINE
  any
    where  $l\_target$ 
  where
    grd1 :  $l\_target \in t\_targets$ 
  then
    act1 :  $EOA := l\_target$ 
  end
Event set_LOA  $\hat{=}$ 
  External event, DO NOT REFINE
  any
    where  $l\_target$ 
  where
    grd1 :  $l\_target \in t\_targets$ 
  then
    act1 :  $LOA := l\_target$ 
  end
Event set_SvL  $\hat{=}$ 
  External event, DO NOT REFINE
  any
    where  $l\_target$ 
  where
    grd1 :  $l\_target \in t\_targets$ 
  then
    act1 :  $SvL := l\_target$ 
  end
Event set_EBD  $\hat{=}$ 
  any
    where  $l\_curve$ 
  where
    grd1 :  $l\_curve \in t\_braking\_curves$ 
  then
    act1 :  $EBD := l\_curve$ 
  end
Event set_SBD  $\hat{=}$ 
  any
    where  $l\_curve$ 
  where
    grd1 :  $l\_curve \in t\_braking\_curves$ 
  then
    act1 :  $SBD := l\_curve$ 
  end
Event set_GUI  $\hat{=}$ 
  any
    where  $l\_curve$ 
  where
    grd1 :  $l\_curve \in t\_braking\_curves$ 

```

```

    then
        act1 : GUI := I_curve
    end
END

```

5.8 dcmp - Supervision Limits

MACHINE dcmp_supervision_limits_m0

SEES c6_supervision_limits

VARIABLES

SBD Shared variable, DO NOT REFINE
T_bs Shared variable, DO NOT REFINE
SBI Shared variable, DO NOT REFINE
I_limit Shared variable, DO NOT REFINE
GUI Shared variable, DO NOT REFINE
MRSP Shared variable, DO NOT REFINE
Pll Shared variable, DO NOT REFINE
W_limit Shared variable, DO NOT REFINE
EBD Shared variable, DO NOT REFINE
T_be Shared variable, DO NOT REFINE
RSM_start Shared variable, DO NOT REFINE
a_current Shared variable, DO NOT REFINE
v_current Shared variable, DO NOT REFINE
P_limit Shared variable, DO NOT REFINE
loc_current Shared variable, DO NOT REFINE
EBI Shared variable, DO NOT REFINE

INVARIANTS

typing_SBD : *SBD* ∈ *t_braking_curves*
 typing_T_bs : *T_bs* ∈ *t_time*
 typing_SBI : *SBI* ∈ *t_supervision_limits*
 typing_I_limit : *I_limit* ∈ *t_supervision_limits*
 typing_GUI : *GUI* ∈ *t_braking_curves*
 typing_MRSP : *MRSP* ∈ *t_speed_profiles*
 typing_Pll : *Pll* ∈ *t_supervision_limits*
 typing_W_limit : *W_limit* ∈ *t_supervision_limits*
 typing_EBD : *EBD* ∈ *t_braking_curves*
 typing_T_be : *T_be* ∈ *t_time*
 typing_RSM_start : *RSM_start* ∈ *t_supervision_limits*
 typing_a_current : *a_current* ∈ *t_acceleration*
 typing_v_current : *v_current* ∈ *t_speed*
 typing_P_limit : *P_limit* ∈ *t_supervision_limits*
 typing_loc_current : *loc_current* ∈ *t_locations*
 typing_EBI : *EBI* ∈ *t_supervision_limits*

EVENTS

Initialisation

begin

```

    act1 : v_current := c_v0
    act2 : a_current := c_a0
    act3 : loc_current := c_l0
    act14 : T_be := c_T_brake0
    act15 : T_bs := c_T_brake0
    act17 : MRSP := c_speed_profile0
    act21 : EBD := c_curve0
    act22 : SBD := c_curve0

```

```

    act23 : GUI := c_curve0
    act24 : EBI := c_slimit0
    act25 : SBI := c_slimit0
    act26 : W_limit := c_slimit0
    act27 : P_limit := c_slimit0
    act28 : I_limit := c_slimit0
    act29 : PII := c_slimit0
    act30 : RS M_start := c_slimit0
end
Event update_train_state  $\hat{=}$ 
    External event, DO NOT REFINE
    any
        l_speed
        l_accel
        l_loc
    where
        grd1 : l_speed  $\in$  t_speed
        grd2 : l_accel  $\in$  t_acceleration
        grd3 : l_loc  $\in$  t_locations
    then
        act1 : v_current := l_speed
        act2 : a_current := l_accel
        act3 : loc_current := l_loc
    end
Event set_T_be  $\hat{=}$ 
    External event, DO NOT REFINE
    any
        l_t_be
    where
        grd1 : l_t_be  $\in$  t_time
    then
        act1 : T_be := l_t_be
    end
Event set_T_bs  $\hat{=}$ 
    External event, DO NOT REFINE
    any
        l_t_bs
    where
        grd1 : l_t_bs  $\in$  t_time
    then
        act1 : T_bs := l_t_bs
    end
Event set_MRSP  $\hat{=}$ 
    External event, DO NOT REFINE
    any
        l_sp
    where
        grd1 : l_sp  $\in$  t_speed_profiles
    then
        act1 : MRS P := l_sp
    end
Event set_EBD  $\hat{=}$ 
    External event, DO NOT REFINE
    any
        l_curve
    where

```

```

    then      grd1 :  $l\_curve \in t\_braking\_curves$ 

    end      act1 :  $EBD := l\_curve$ 
end
Event set_SBD  $\hat{=}$ 
    External event, DO NOT REFINE

    any

    where  $l\_curve$ 

    then      grd1 :  $l\_curve \in t\_braking\_curves$ 

    end      act1 :  $SBD := l\_curve$ 
end
Event set_GUI  $\hat{=}$ 
    External event, DO NOT REFINE

    any

    where  $l\_curve$ 

    then      grd1 :  $l\_curve \in t\_braking\_curves$ 

    end      act1 :  $GUI := l\_curve$ 
end
Event set_EBI  $\hat{=}$ 
    any

    where  $l\_limit$ 

    then      grd1 :  $l\_limit \in t\_supervision\_limits$ 

    end      act1 :  $EBI := l\_limit$ 
end
Event set_SBI  $\hat{=}$ 
    any

    where  $l\_limit$ 

    then      grd1 :  $l\_limit \in t\_supervision\_limits$ 

    end      act1 :  $SBI := l\_limit$ 
end
Event set_W_limit  $\hat{=}$ 
    any

    where  $l\_limit$ 

    then      grd1 :  $l\_limit \in t\_supervision\_limits$ 

    end      act1 :  $W\_limit := l\_limit$ 
end
Event set_I_limit  $\hat{=}$ 
    any

    where  $l\_limit$ 

    then      grd1 :  $l\_limit \in t\_supervision\_limits$ 

    end      act1 :  $I\_limit := l\_limit$ 
end

```

```

Event set_P_limit  $\hat{=}$ 
  any
    l_limit
  where
    grd1 : l_limit  $\in$  t_supervision_limits
  then
    act1 : P_limit := l_limit
  end
Event set_PII_limit  $\hat{=}$ 
  any
    l_limit
  where
    grd1 : l_limit  $\in$  t_supervision_limits
  then
    act1 : PII := l_limit
  end
Event set_RSM_start_limit  $\hat{=}$ 
  any
    l_limit
  where
    grd1 : l_limit  $\in$  t_supervision_limits
  then
    act1 : RSM_start := l_limit
  end
END

```

5.9 dcmp - Monitoring Commands

MACHINE *dcmp_monitoring_commands*

SEES *c6_supervision_limits*

VARIABLES

status_current Private variable
SBI Shared variable, DO NOT REFINE
I_limit Shared variable, DO NOT REFINE
PII Shared variable, DO NOT REFINE
W_limit Shared variable, DO NOT REFINE
RSM_start Shared variable, DO NOT REFINE
a_current Shared variable, DO NOT REFINE
cmd_current Private variable
v_current Shared variable, DO NOT REFINE
P_limit Shared variable, DO NOT REFINE
loc_current Shared variable, DO NOT REFINE
EBI Shared variable, DO NOT REFINE

INVARIANTS

typing_status_current : *status_current* \in *t_DMI_commands*
typing_SBI : *SBI* \in *t_supervision_limits*
typing_I_limit : *I_limit* \in *t_supervision_limits*
typing_PII : *PII* \in *t_supervision_limits*
typing_W_limit : *W_limit* \in *t_supervision_limits*
typing_RSM_start : *RSM_start* \in *t_supervision_limits*
typing_a_current : *a_current* \in *t_acceleration*
typing_cmd_current : *cmd_current* \in *t_TI_commands*
typing_v_current : *v_current* \in *t_speed*

```

    typing_P_limit :  $P\_limit \in t\_supervision\_limits$ 
    typing_loc_current :  $loc\_current \in t\_locations$ 
    typing_EBI :  $EBI \in t\_supervision\_limits$ 
EVENTS
Initialisation
    begin
        act1 :  $v\_current := c\_v0$ 
        act2 :  $a\_current := c\_a0$ 
        act3 :  $loc\_current := c\_l0$ 
        act4 :  $cmd\_current := c\_no\_command$ 
        act5 :  $status\_current := c\_normal$ 
        act24 :  $EBI := c\_slimit0$ 
        act25 :  $SBI := c\_slimit0$ 
        act26 :  $W\_limit := c\_slimit0$ 
        act27 :  $P\_limit := c\_slimit0$ 
        act28 :  $I\_limit := c\_slimit0$ 
        act29 :  $PII := c\_slimit0$ 
        act30 :  $RS M\_start := c\_slimit0$ 
    end
Event update_train_state  $\hat{=}$ 
    any
         $l\_speed$ 
         $l\_accel$ 
         $l\_loc$ 
    where
        grd1 :  $l\_speed \in t\_speed$ 
        grd2 :  $l\_accel \in t\_acceleration$ 
        grd3 :  $l\_loc \in t\_locations$ 
    then
        act1 :  $v\_current := l\_speed$ 
        act2 :  $a\_current := l\_accel$ 
        act3 :  $loc\_current := l\_loc$ 
    end
Event new_outputs  $\hat{=}$ 
    any
         $l\_ti\_cmd$ 
         $l\_dmi\_status$ 
    where
        grd1 :  $l\_ti\_cmd \in t\_TI\_commands$ 
        grd2 :  $l\_dmi\_status \in t\_DMI\_commands$ 
    then
        act1 :  $cmd\_current := l\_ti\_cmd$ 
        act2 :  $status\_current := l\_dmi\_status$ 
    end
Event set_EBI  $\hat{=}$ 
    External event, DO NOT REFINe
    any
         $l\_limit$ 
    where
        grd1 :  $l\_limit \in t\_supervision\_limits$ 
    then
        act1 :  $EBI := l\_limit$ 
    end
Event set_SBI  $\hat{=}$ 
    External event, DO NOT REFINe
    any
         $l\_limit$ 

```

```

    where
    then    grd1 : l_limit ∈ t_supervision_limits
    end      act1 : SBI := l_limit
Event set_W_limit ≡
    External event, DO NOT REFINE
    any
    where l_limit
    then    grd1 : l_limit ∈ t_supervision_limits
    end      act1 : W_limit := l_limit
Event set_I_limit ≡
    External event, DO NOT REFINE
    any
    where l_limit
    then    grd1 : l_limit ∈ t_supervision_limits
    end      act1 : I_limit := l_limit
Event set_P_limit ≡
    External event, DO NOT REFINE
    any
    where l_limit
    then    grd1 : l_limit ∈ t_supervision_limits
    end      act1 : P_limit := l_limit
Event set_PII_limit ≡
    External event, DO NOT REFINE
    any
    where l_limit
    then    grd1 : l_limit ∈ t_supervision_limits
    end      act1 : PII := l_limit
Event set_RSM_start_limit ≡
    External event, DO NOT REFINE
    any
    where l_limit
    then    grd1 : l_limit ∈ t_supervision_limits
    end      act1 : RSM_start := l_limit
END

```


References

- [Abr10] Jean-Raymond Abrial. *Modeling in Event-B - System and Software Engineering*. Cambridge University Press, 2010.
- [Eur12] European Railway Agency (ERA). System Requirements Specification - ETCS Subset 026. <http://www.era.europa.eu/Document-Register/Documents/Index00426.zip>, 2012.
- [Jas12] Michael Jastram, editor. *Rodin User's Handbook*. DEPLOY Project, 2012.
- [Mat13a] Matthias Güdemann. Event-B Model of Subset 026, Section 3.5. github - openETCS, 2013.
- [Mat13b] Matthias Güdemann. Event-B Model of Subset 026, Section 4.6. github - openETCS, 2013.
- [Mat13c] Matthias Güdemann. Event-B Model of Subset 026, Section 5.9. github - openETCS, 2013.
- [SPHB11] Renato Silva, Carine Pascal, Thai Son Hoang, and Michael Butler. Decomposition tool for event-B. *Software: Practice and Experience*, 41(2):199–208, 2011.