OETCS

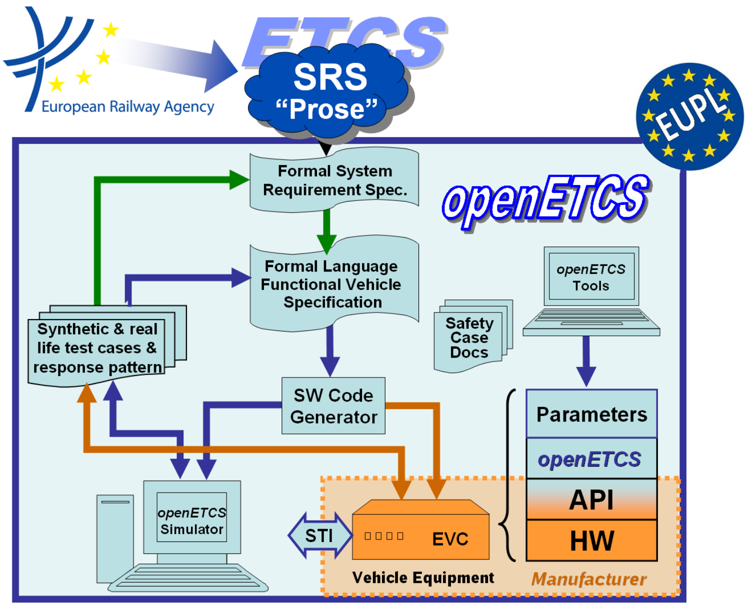
ITEA2 Project
Call 6 11025
2012 – 2015

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

openETCS

This page is intentionally left blank

Work-Package 7: "Toolchain"

OETCS
April 2013

Event-B Model of Subset 026, Section 3.13

Matthias Güdemann

Systerel
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

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
References 19

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

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-Systerel/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 and 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

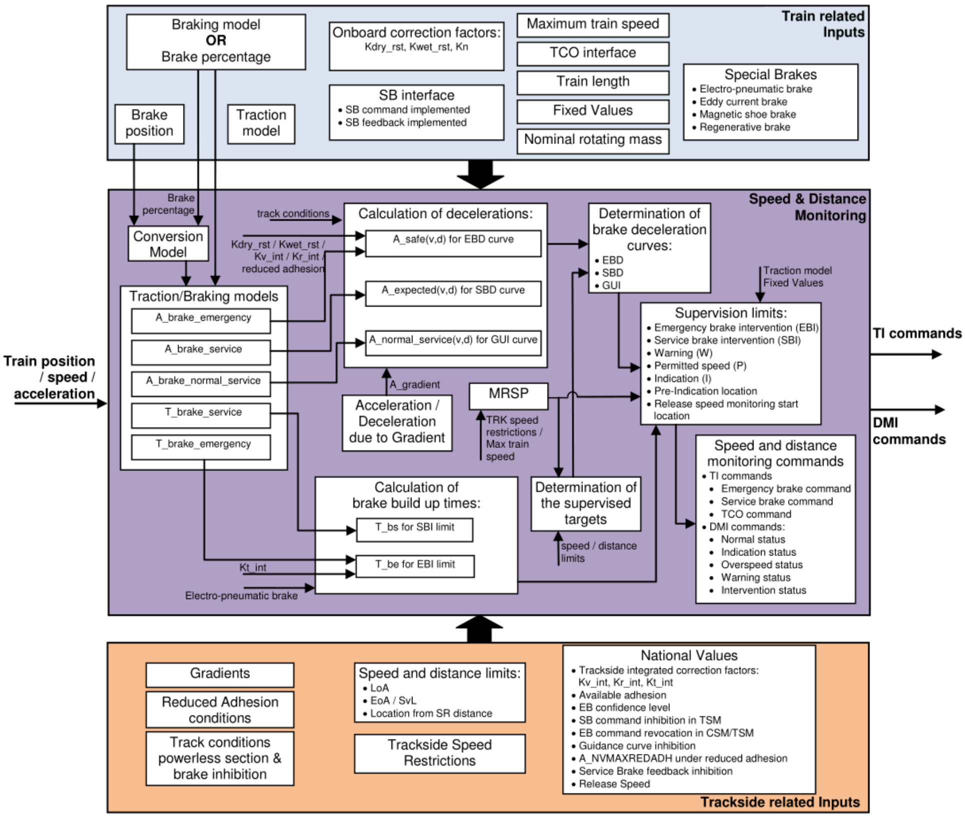


Figure 1. Speed and Distance Monitoring Overview ([Eur12] p. 85)

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-in [1] 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.

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

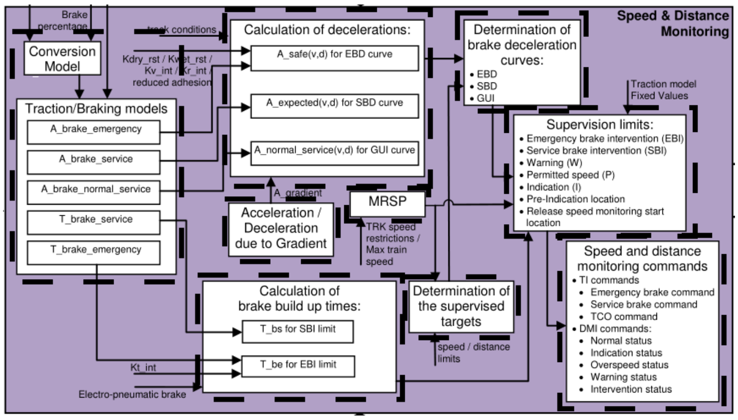


Figure 2. Decomposition of System

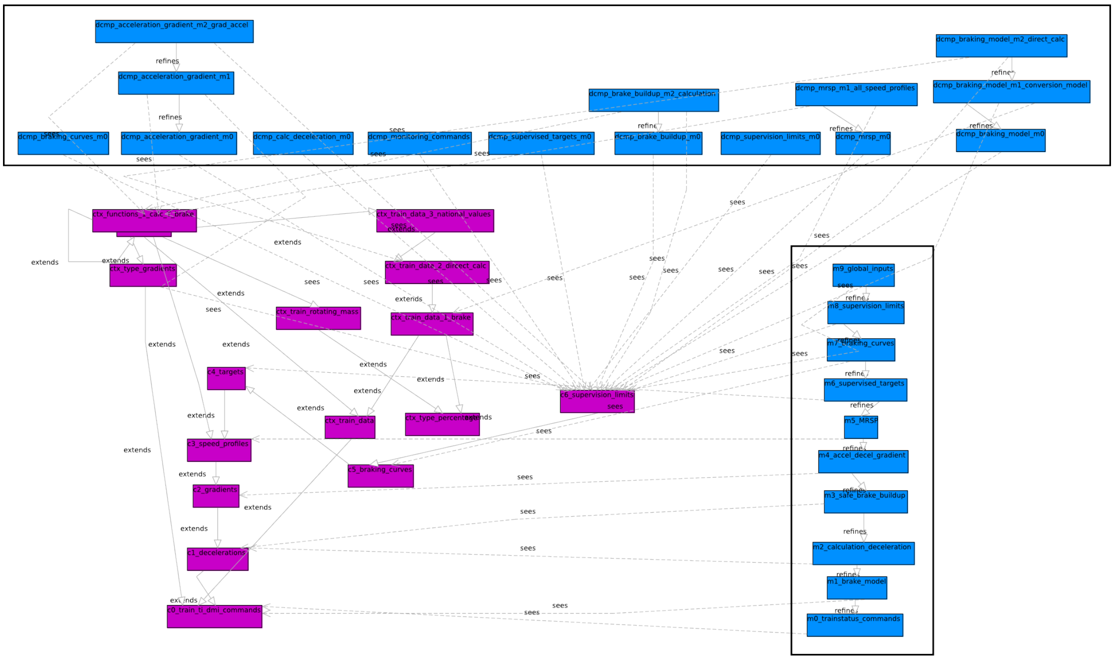


Figure 3. Machine Decomposition Overview

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_v*0
- *c_a*0
- *c_l*0
- *c_a_brake*0
- *c_T_brake*0

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 : $c\_v0 \in t\_speed$
axm4 : $c\_a0 \in t\_acceleration$

axm5 : $c\_l0 \in t\_locations$
axm6 : $c\_a\_brake0 \in t\_speed \rightarrow t\_acceleration$
default brake profile

axm7 : $c\_T\_brake0 \in t\_time$
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 : $v\_current \in t\_speed$
inv2 : $a\_current \in t\_acceleration$
inv3 : $loc\_current \in t\_locations$
inv4 : $cmd\_current \in t\_TI\_commands$
inv5 : $status\_current \in t\_DMI\_commands$

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* $\widehat{=}$
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 $new\_outputs \widehat{=}$
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 $set\_A\_brake\_emergency \widehat{=}$
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 $set\_A\_brake\_service \widehat{=}$
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 $set\_A\_brake\_normal\_service \widehat{=}$

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 $set\_T\_brake\_service \widehat{=}$
any
l_T_brake
where
grd1 : $l\_T\_brake \in t\_time$
then
act1 : $T\_brake\_service := l\_T\_brake$
end

Event $set\_T\_brake\_emergency \widehat{=}$
any
l_T_brake
where
grd1 : $l\_T\_brake \in t\_time$
then
act1 : $T\_brake\_emergency := l\_T\_brake$
end

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 $set\_A\_safe \widehat{=}$

any

l_a_decel

where

grd1 : $l\_a\_decel \in t\_speed \times t\_distance \to t\_acceleration$
grd2 : $A\_brake\_emergency \in t\_speed \to t\_acceleration$

then

act1 : $A\_safe := l\_a\_decel$

end

Event $set\_A\_expected \widehat{=}$

any

l_a_decel

where

grd1 : $l\_a\_decel \in t\_speed \times t\_distance \to t\_acceleration$
grd2 : $A\_brake\_service \in t\_speed \to t\_acceleration$

then

act1 : $A\_expected := l\_a\_decel$

end

Event $set\_A\_normal\_service \widehat{=}$

any

l_a_decel

where

grd1 : $l\_a\_decel \in t\_speed \times t\_distance \to t\_acceleration$
grd2 : $A\_brake\_normal\_service \in t\_speed \to t\_acceleration$

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 $set\_T\_be \widehat{=}$

any

l_t_be

where

grd1 : $l\_t\_be \in t\_time$
grd2 : $T\_brake\_emergency \in t\_time$

then

act1 : $T\_be := l\_t\_be$

end

Event $set\_T\_bs \widehat{=}$
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 $set\_A\_gradient \widehat{=}$
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 $set\_MRSP \mathrel{\widehat{=}}$

any

l_sp

where

grd1 : $l\_sp \in t\_speed\_profiles$

then

act1 : $MRSP := l\_sp$

end

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 $set\_EOA \mathrel{\widehat{=}}$

any

l_target

where

`grd1` : $l\_target \in t\_targets$
`grd2` : $MRSP \in t\_speed\_profiles$

then

`act1` : $EOA := l\_target$

end

Event $set\_LOA \widehat{=}$

any

l_target

where

`grd1` : $l\_target \in t\_targets$
`grd2` : $MRSP \in t\_speed\_profiles$

then

`act1` : $LOA := l\_target$

end

Event $set\_SvL \widehat{=}$

any

l_target

where

`grd1` : $l\_target \in t\_targets$
`grd2` : $MRSP \in t\_speed\_profiles$

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 *GUI* of the appropriate type.

MACHINE m7_braking_curves
REFINES m6_supervised_targets
SEES c5_braking_curves
VARIABLES

EBD

SBD
GUI

EVENTS

Event $set\_EBD \widehat{=}$

any

l_curve

where

grd1 : $l\_curve \in t\_braking\_curves$
grd2 : $A\_safe \in t\_speed \times t\_distance \to t\_acceleration$
grd3 : $A\_expected \in t\_speed \times t\_distance \to t\_acceleration$
grd4 : $A\_normal\_service \in t\_speed \times t\_distance \to t\_acceleration$
grd5 : $LOA \in t\_targets$
grd6 : $EOA \in t\_targets$
grd7 : $SvL \in t\_targets$

then

act1 : $EBD := l\_curve$

end

Event $set\_SBD \widehat{=}$

any

l_curve

where

grd1 : $l\_curve \in t\_braking\_curves$
grd2 : $A\_safe \in t\_speed \times t\_distance \to t\_acceleration$
grd3 : $A\_expected \in t\_speed \times t\_distance \to t\_acceleration$
grd4 : $A\_normal\_service \in t\_speed \times t\_distance \to t\_acceleration$
grd5 : $LOA \in t\_targets$
grd6 : $EOA \in t\_targets$
grd7 : $SvL \in t\_targets$

then

act1 : $SBD := l\_curve$

end

Event $set\_GUI \widehat{=}$

any

l_curve

where

grd1 : $l\_curve \in t\_braking\_curves$
grd2 : $A\_safe \in t\_speed \times t\_distance \to t\_acceleration$
grd3 : $A\_expected \in t\_speed \times t\_distance \to t\_acceleration$
grd4 : $A\_normal\_service \in t\_speed \times t\_distance \to t\_acceleration$
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 : $c\_slimit0 \in t\_supervision\_limits$

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
PIl pre-indication_location
RSM_start release speed monitoring start location

EVENTS

Event $set\_EBI \triangleq$

any

l_limit

where

grd1 : $l\_limit \in t\_supervision\_limits$
grd2 : $MRSP \in t\_speed\_profiles$
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 : $EBI := l\_limit$

end

Event $set\_SBI \triangleq$

any

l_limit

where

grd1 : $l\_limit \in t\_supervision\_limits$
grd2 : $MRSP \in t\_speed\_profiles$
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 : $SBI := l\_limit$

end

Event $set\_W\_limit \triangleq$

any

l_limit

where

grd1 : $l\_limit \in t\_supervision\_limits$
grd2 : $MRSP \in t\_speed\_profiles$
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* $\widehat{=}$

any

l_limit

where

grd1 : $l\_limit \in t\_supervision\_limits$
grd2 : $MRSP \in t\_speed\_profiles$
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* $\widehat{=}$

any

l_limit

where

grd1 : $l\_limit \in t\_supervision\_limits$
grd2 : $MRSP \in t\_speed\_profiles$
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* $\widehat{=}$

any

l_limit

where

grd1 : $l\_limit \in t\_supervision\_limits$
grd2 : $MRSP \in t\_speed\_profiles$
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* $\widehat{=}$

any

l_limit

where

grd1 : $l\_limit \in t\_supervision\_limits$
grd2 : $MRSP \in t\_speed\_profiles$

```
            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 : RSM_start := l_limit
      end
END
```

5 Model Decomposition

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.

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.