OETCS

openETCS

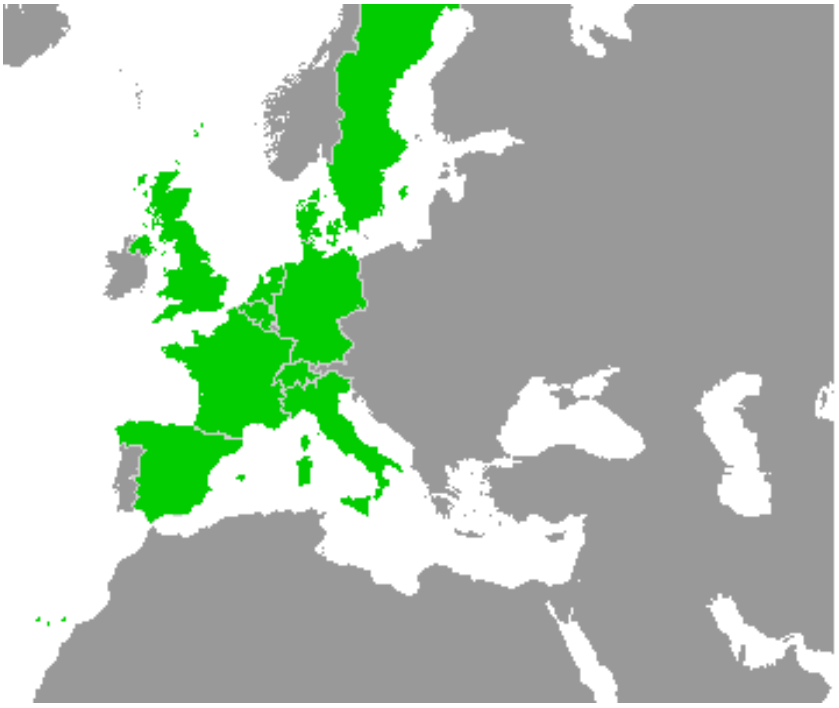
ITEA2 Project
Call 6 11025
2012 – 2015

Work-Package 7: “Toolchain”

Event-B Model of Subset 026, Section 3.13

Matthias Güdemann
Systerel, France

March 2013



This page is intentionally left blank

Work-Package 7: “Toolchain”

OETCS
March 2013

Event-B Model of Subset 026, Section 3.13

Matthias Güdemann
Systerel, France

Systerel

Model Description

Prepared for ITEA2 openETCS consortium
Europa

Table of Contents

Figures and Tables

Figures

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.

Table 1. Glossary

1 Short Introduction to Event-B

The formal language Event-B is based on a set-theoretic approach. It is a variant of the B language, with a focus on system level modeling [Abr10]. An Event-B model is separated into a static and a dynamic part.

The dynamic part of an Event-B model describes abstract state machines. The state is represented by a set of state variables. A transition from one state to another is represented by parametrized events which assign new values to the state variables. Event-B allows unbounded state spaces. They are constrained by invariants expressed in first order logic with equality which must be fulfilled in any case. The initial state is created by a special initialization event.

The static part of an Event-B model is represented by contexts. These consist of carrier sets, constants and axioms. The type system of a model is described by means of carrier sets and constraints expressed by axioms.

Event-B is not only comprised of descriptions of abstract state machines and contexts, but also includes a development approach. This approach consists of iterative refinement of the machines until the desired level of detail is reached. In the Rodin tool, proof obligations are automatically created which ensure correct refinement.

Together with the machine invariants, the proof obligations for the refinement are formally proven, creating proof trees. To accomplish this, there are different options: many proof obligations can be discharged by automated provers (e.g., AtelierB, NewPP, Rodin's SMT-plugin), but as the underlying logic is in general undecidable, it is sometimes necessary to use the interactive proof support of Rodin.

Any external actions, e.g., mode changes by the driver or train level changes are modeled via parametrized events. Only events can modify the variables of a machine. An Event-B model is on the system level, events are assumed to be called from a software system into which the

functional model is embedded. The guards of the events assure that any event can only be called when appropriate.

2 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.

3 Model Overview

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

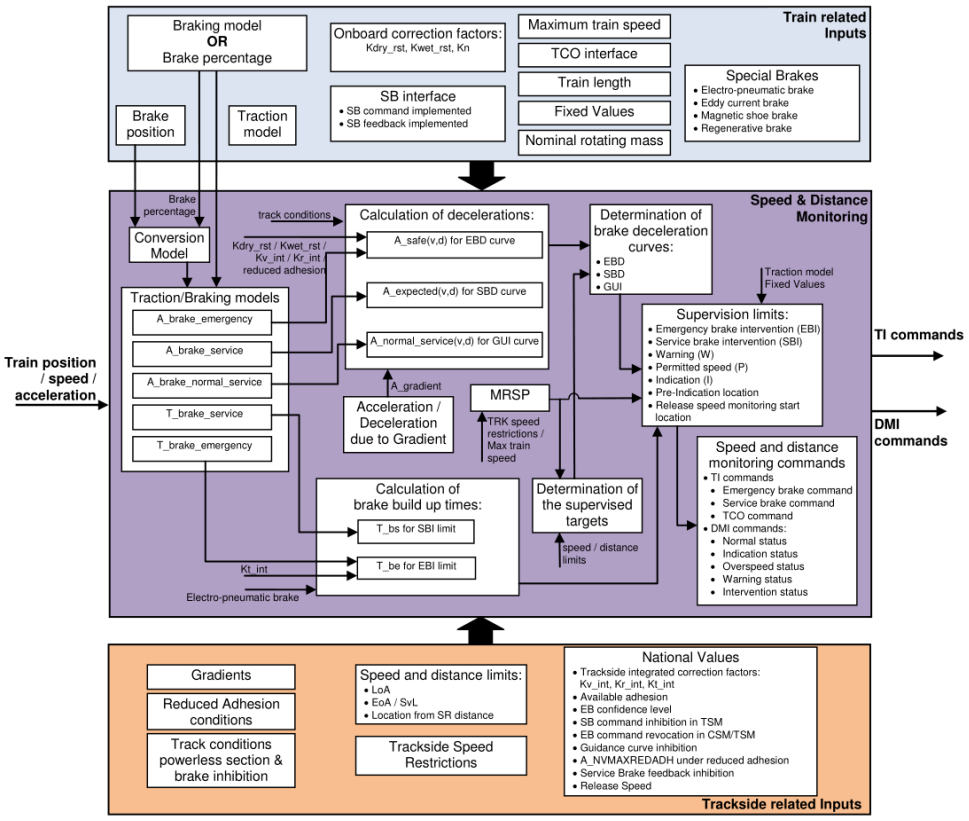


Figure 1. Speed and Distance Monitoring Overview ([Eur12] p. 85)

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 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.

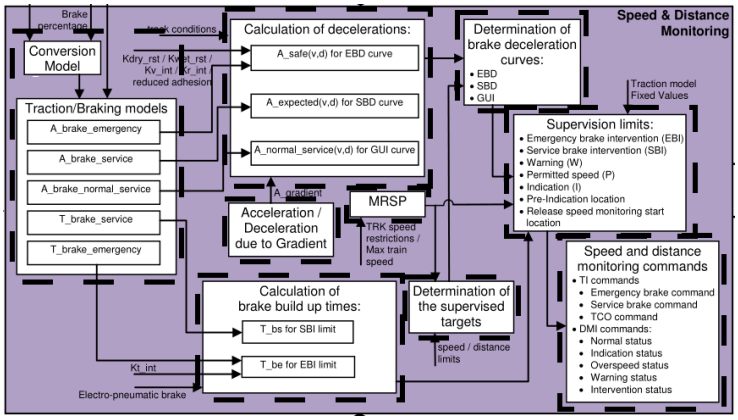


Figure 2. Decomposition of System

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.

Figure 3. Machine Decomposition Overview

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.

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

4 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] 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.

5 Detailed Model Description

5.1 Context 0 - Train Inputs, TI and DMI command

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

5.2 Machine 0 - Basic Communication

Implemented Requirements

- each session allows for communication between two entities (cf. §3.5.2.1)

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, 2013.

[Mat13b] Matthias Güdemann. Event-B Model of Subset 026, Section 4.6, 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.

