Event-B model of Subset 026, Section 3.5.3

Matthias Güdemann

January 22, 2013

This document describes a formal model of the requirements of section 3.5.3 of the ETCS specification 3.3.0 [**?**]. This section describes the establishing of a communication connection between on-board and on-track equipment.

The model is expressed in the formal language Event-B [**?**] and developed in the RODIN tool [**?**]. In this documentation we present the Event-B machines and the contexts of the model. Each section introduces one refinement step, discusses the reasoning of that step and the newly introduced variables and invariants. Later sections refine the machine, introducing more detailed behavior.

The machines are not presented in full, only the relevant parts that were changed or added will be shown to make it more readable. In particular the initialization is not shown for the refined machines. If not mentioned explicitly, sets are initialized empty, integers with value 0 and Booleans with false.

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 development [**?**]. An Event-B model describes abstract state machines, transitions describe changes to the state variables of the machine and pre-conditions of the events are expressed in first order logic with equality. The Event-B machines also contain invariant specifications in this logic. These represent requirements assuring the correctness of the model.

While Event-B machines describe the dynamic aspect of a model, contexts described the static part. In particular, they describe the type system of a model by means of carrier sets. Contexts also allow the definition of constants and axioms, in general these axioms define constraints on the types.

Event-B is not only the description of an abstract state machine and its type properties, but is also comprised of a development approach. This approach consists of iterative refinement of the machines until the desired level of detail is reached. To ensure correct refinement, i.e., that the abstract model has a more general behavior, proof obligations are created. This can be automated by special tools like Rodin. For refinement of abstract data structures, the necessary proof obligations must be created manually.

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

2 Modeling Strategy

The section 3.5.3 of the SRS describes how a communication session is established. In its context, the low level EURORADIO network connection (cf. §3.5.1.1) are considered basic functionality and are not part of the modeling.

The model is constructed from the local point of view of an OBU entity, it does not consider modeling any on-track unit. On track entities are only modeled as possible communication partners.

Established communication sessions are modeled as sets of entities, the events that modify these sets have an event parameter that represents one entity

3 Model Overview

Figure 1 shows the structure of the Event-B model. The blue boxes represent the abstract state machines, the magenta boxes the contexts. An arrow from one machine to another machine represents a refinement relation, an arrow from a machine to a context represents a sees relation and arrow from one context to another represents an extension relation.

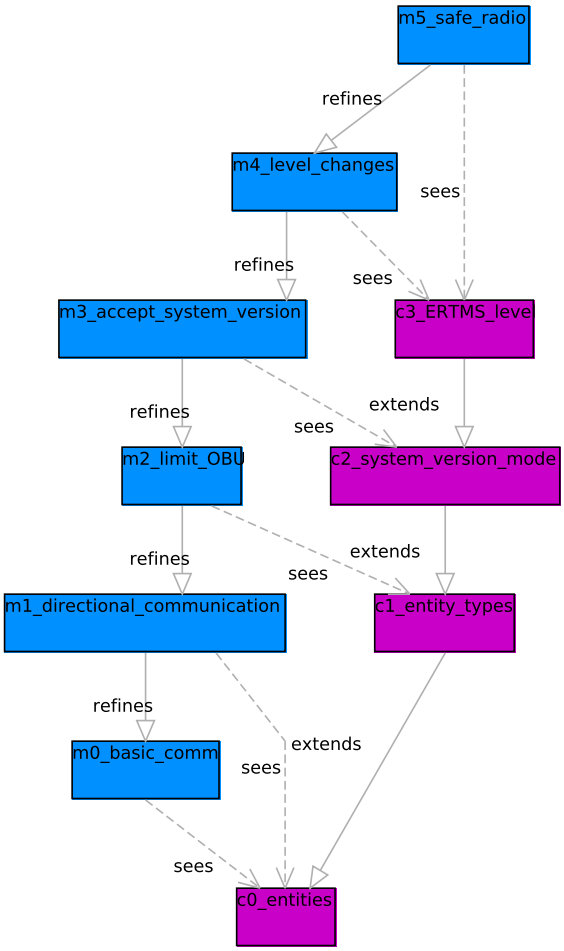

Figure 1: Overview on State Machine and Context Hierarchy

The modeling starts with the very abstract possibility to establish and to terminate a communication session in the machine $m0$, the set of entities is defined in the context $c0$. This basic functionality is refined in the succeeding machines to incorporate the different stages of the protocol to establish a session. The contexts further refine the entities to on-track and on-board entities and limit the modeling to the point of view of an OBU.

4 Detailed Model Description

4.1 Context 0 - Entities

This context defines the type of entities with whom a communication session can be established. *my_entity* represents the piece of equipment which is modeled.

CONTEXT c0_entities
SETS

entities

CONSTANTS

my_entity

AXIOMS

axm1 : $my\_entity \in entities$

END

4.2 Machine 0 - Basic Communication

This abstract state machine represents the basic functionality. Its allows for the creation and the destruction of a communication session with another entity.

MACHINE m0_basic_comm
SEES c0_entities
VARIABLES

comm_sessions

INVARIANTS

inv1 : $comm\_sessions \subseteq entities \setminus \{my\_entity\}$

EVENTS

Initialisation

begin

act1 : $comm\_sessions := \varnothing$

end

Event *establish_communication* $\widehat{=}$

any

l_partner

where

grd1 : $l\_partner \notin comm\_sessions$
grd2 : $l\_partner \in entities \setminus \{l\_partner\}$

then

act1 : $comm\_sessions := comm\_sessions \cup \{l\_partner\}$

end

Event *terminate_communication* $\widehat{=}$

any

l_partner

where

grd1 : $l\_partner \in comm\_sessions$

then

act1 : $comm\_sessions := comm\_sessions \setminus \{l\_partner\}$

end

END