openETCS

OETCS/WP2/D2.3.0 – 00/00

ITEA2 Project
2012 – 2015

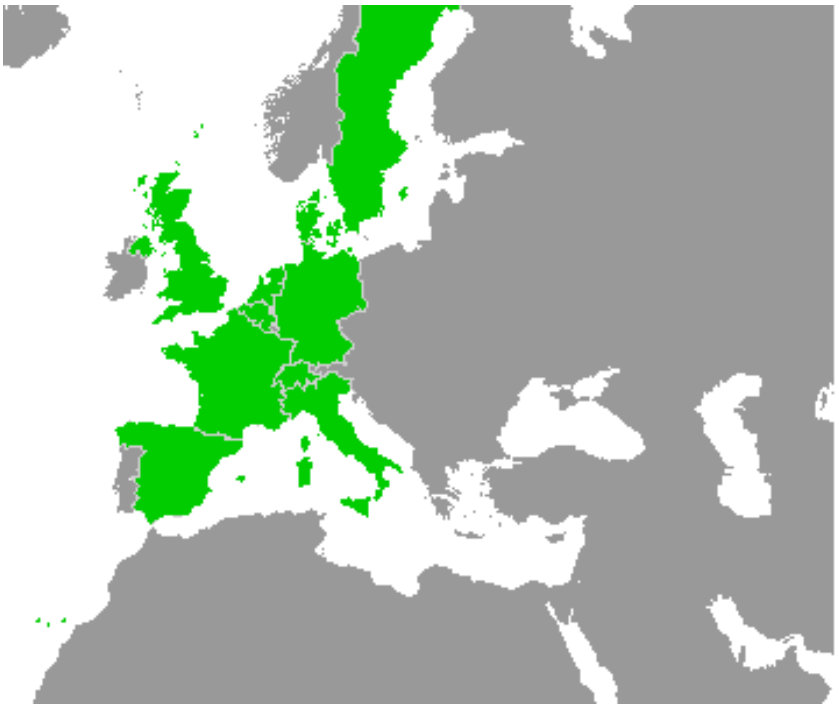
Work-Package 2: “Definition”

OpenETCS process

Definition of the overall process for the formal description of ETCS and the rail system it works in

Marielle Petit-Doche

February 2013



This page is intentionally left blank

OpenETCS process

Definition of the overall process for the formal description of ETCS and the rail system it works in

Marielle Petit-Doche

Systerel

Definition

Prepared for ITEA2 openETCS consortium
Europa

Abstract: This document give a description of the process to be applied in the OpenETCS project. It gives a description of the activities to specify and design a critical system in a first part. The second part presents an abstract description of the case study issued from subset 26.

Table of Contents

Figures and Tables **Figures**

Tables

1 Introduction

The purpose of this document is to describe, for the OpenETCS project, the activities of specification and design. However the activities of verification and validation are not in the scope of this document and will be described in %%WP4/D4.x.x%% .

These activities shall follow the requirements of EN 50126 and EN 50128 and reflect usual activities for the development of railway critical systems (see D2.1.0 and D2.2.0).

This document contents two parts :

- the description of the process to applied in the OpenETCS project
- the abstract description and the limit of the case study

2 Reference documents

- CENELEC EN 50126-1 — 01/2000 — *Railways applications — The specification and demonstration of Reliability, Availability, Maintenability and Safety (RAMS) — Part 1: Basic requirements and generic process*
- CENELEC EN 50128 — 10/2011 — *Railway applications – Communication, signalling and processing systems – Software for railway control and protection systems*
- CENELEC EN 50129 — 05/2003 — *Railway applications — Communication, signalling and processing systems — Safety related electronic systems for signalling*
- FPP — *Project Outline Full Project Proposal Annex OpenETCS* – v2.2
- SUBSET-026 3.3.0 — *System Requirement Specification*
- SUBSET-076-x 2.3.y — Test related ERTMS documentation
- SUBSET-088 2.3.0 — *ETCS Application Levels 1 & 2 - Safety Analysis*
- SUBSET-091 2.5.0 — *Safety Requirements for the Technical Interoperability of ETCS in Levels 1 & 2*
- CCS TSI — *CCS TSI for HS and CR transeuropean rail has been adopted by a Commission Decision 2012/88/EU on the 25th January 2012*

3 Conventions

The requirements are prefixed by “R-zz-x-y”, and are written in a roman typeface, where “R” stands for “Requirement”, “zz” identifies the source document,“x” is the version number and“y” is the identifier of the requirement. All the text written in italics is not a requirement: it may be a note, an open issue, an explanation of the requirements, or an example.

The placeholder “ %%xxx%% ” is used to indicates that a paragraph or section is not finished, to be defined or to be confirmed.

4 Glossary

API Application Programming Interface

FME(C)A Failure Mode Effect (and Criticity) Analysis

I/O Input/Output

OBU OnBoard Unit

QA Quality Analysis

RBC Radio Block Center

RTM RunTime Model

SIL Safety Integrity Level

THR Tolerable Hazard Rate

V&V Verification & Validation

5 OpenETCS process

5.1 Motivation

This document describes the process to be applied during the OpenETCS project to achieve the following goals of the OpenETCS project :

A formal reference specification for the ETCS requirements and architecture

The first goal of the project is to propose a formalization of a subset of the on-board subsystem, as defined in the SUBSET-26.

The purpose of the formalization is:

- to enhance the understanding of modelled subset;
- to allow formal analysis of the modelled subset;
- to be able to animate the model for testing an analyzing purpose;
- to provide information on the completeness and soundness of the SUBSET-26;
- to be used as a reference formal specification for the implementation of an OBU (by the OpenETCS project team and by industrial actors);
- ...

The output of this goal is a formal specification, understandable by many tools (SCADE, Simulink, B tools, OpenETCS tool chain...) that can be given to all railway actors, and if possible associated to SRS documents in the ERA database. The final goal is that industrial actors work with this formal specification instead of natural language specification.

Definition of a tool chain and process/methodologies for developing on onboard software that can fulfill the EN 50128 requirements

The process and the associated tools, shall provide a certifiable product. For this purpose all the step of the process and the choice of methods and tools shall be justifyed to ensure a safe approach to build a system.

The full safety process needed for the OpenETCS to be *certifiable* according to CENELEC 50126 and 50128 shall be described in details. This safety plan will detail precisely which activities are required or not, why, and the choices that are made that allows to claim that safety is guaranteed.

Building an implementation of the subset of an onboard ETCS using the system model and the tool chain

It is the demonstration that all the work done in the OpenETCS project is coherent, and that the tool chain is operational.

The output is the result of an implementation for the ETCS requirements and architecture which can be used by the industrial as references.

Define the safety properties at the model level

In order to comply the CENELEC standards, it is necessary to conduct safety activities to identify errors and anomalies in the process. One important step for this is to define safety properties which are on the same level than the formal model.

These safety properties:

- will be used for the validation of the model itself;
- will be used as reference proof obligations for the subsequent activies.

Because the full design, development, validation and safety analysis process for a SIL4 OBU is a huge task far beyond the project possibilities, the full safety activities will not be conducted on the whole subsystem (see below). Nevertheless the safety process description shall be complete according to CENELEC requirements.

However this safety analysis is out of the scope of the OpenETCS project. SUBSET-088 2.3.0 and SUBSET-091 2.5.0 will provide elements of safety analysis.

5.2 Overall description

%%highlevel description of the full process for OpenETCS : main step as defined in the synthesis of requirement. %

%%To check there is no conflict with req on CENELEC (D.2.2) and QA plan%%

%%Elements are issue form req_synthesis of D2.6, D2.7, D2.8, D2.9%%

In order to pursue this goals, the development cycle for the project may be presented as follow. The proposed process shall comply the CENELEC standard EN 50126, EN 50128 and EN50129. References to the chapter of these standard will be given in the sequel. However proof that the process satisfies the requirements of the standard is out of the scope of this document and shall

be manage by the Safety Case (WP4). Amount the requested quality assurance Plan request by EN 50128, the current document only contributes to the definition of the life-cycle model by giving a description of each step of the process (activities, input and output documents, entry and exit of each activity).

Fig. 1 shows the main part of the development process. This process may be seen as a “triple-V”. The smaller V corresponds to the development of the formal model.

It starts by the SRS which is not part of the project (SUBSET-26), then outlines the boundaries and the applicable requirements from the SUBSET-26 that will be used in the formal model. %%to link to system development phase (see EN50129)%%

The next step is the creation of the formal model itself. Because this model is executable, it can be validated as itself, thus the first “closing branch” of the V.

%%to link to software design phases : SW requirement, SW Architecture and Design, SW component Design (see

From the model can be derived some “abstract” code. The word “abstract” is used to emphasize that this code is not necessarily capable of running of a full SIL4 platform. This code can be validated in the second “closing branch”, possibly using some of the work done in the first branch. %%to link to software design phases : SW component implementation Design (see EN50128)%%

A project demonstrator may be derived from this code (or may be the “abstract” code itself).

The third “closing branch” corresponds to the production of code capable of running on a given SIL4 platform, and the associated validation activities. This is not part of the project.

The yellow boxes corresponds to activities that should be covered completely in order to produce a certifiable product, but of which only a subset will be conducted in order to demonstrate the capabilities of the product.

Fig. 2 shows activities that are needed for the safety analyses. It should be considered in parallel of the descending branch of the V, but has been put on a separate diagram for the sake of clarity.

High level safety properties are provided, which must be refined side-to-side with each step on the descending branch of the V. These properties are then used for the safety analysis of the model. The validation (safety analyses) boxes are yellow because the full activity will not be conducted. Only a subset of the safety properties will be proved.

Regarding the process, WP2 shall issue (through this document) the requirements on V&V, including safety activities. From these requirements, WP4 shall propose the corresponding plans (Safety, Verification and Validation). This plans will then be reviewed by WP2 to check conformance to the requirements. The reviewed plans will then be used as reference for the activities of the WPs.

5.3 System specification

%%To develop%%

Comment. This section as to define the activities links to the system specification and model.

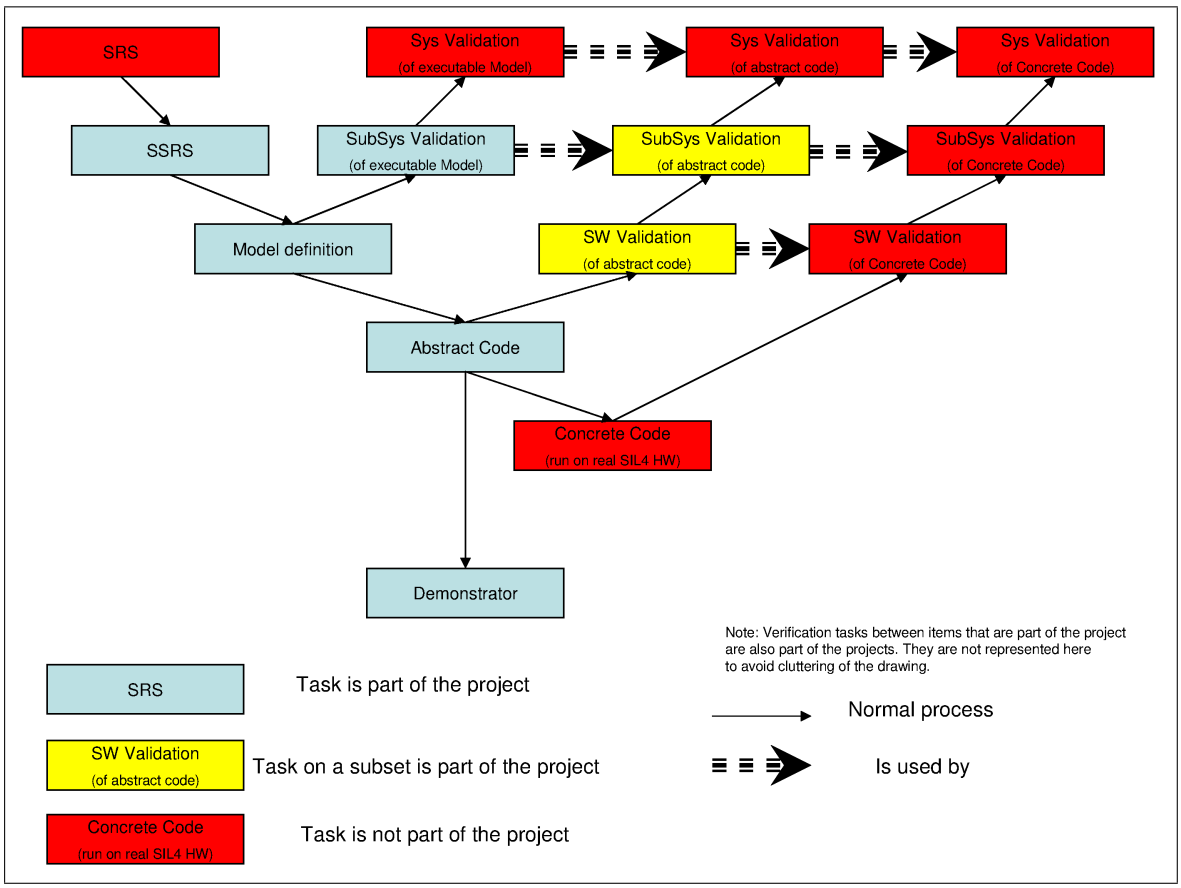


Figure 1. Main process

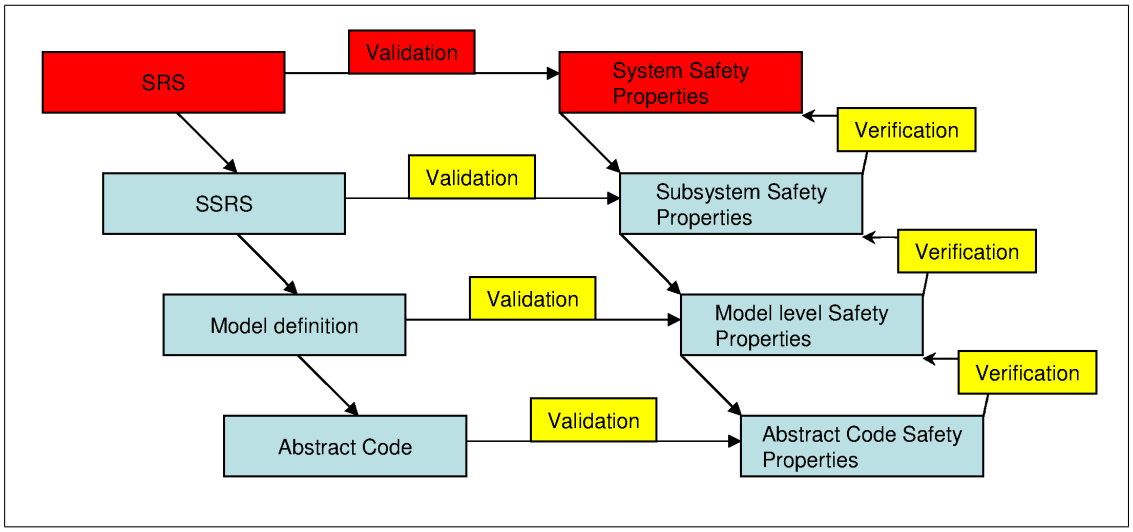


Figure 2. Safety analyses

This step is not explicitly defined in EN 50128 but is defined in EN 50129. However EN 50128 give the expected output from this step for the software activities : System Requirements Specification, System Safety Requirements Specification, System Architecture Description, External Interface Specifications

Open Issue. To discuss with all partners : Do we need a SSRS ? This is expected by EN50129.

S. Baro comments (15/02/2013) :

SSRS

We had a un-conclusive discussion on the necessity to insert a SubSystem Requirement Specification betwen the SRS (subset 26) and the formal model. The reason of this is that the model we want corresponds to a subsystem (part of the OBU), but the SRS corresponds to the whole system. The SRS also does not provide any functional architecture, and we think it is necessary to provide one. In the other hand, it adds one document between the SRS and the model. The question is thus to know if this document will remove more errors than it will add. Proposal: To provide a SSRS which contains: - a formal or semi-formal functional architecture of the OBU part which will be modeled, with function "boxes" and I/O "arrows"; - the requirements allocated to the functions and I/O, rewritten but still in natural language; - the tagging Safety/Non Safety of these requirements. This document should be seen as an important step of the modeling process, and would be under the responsibility of WP3.

5.4 Model definition

%%To have a discussion with Sylvain Baro on his choice of step for the process%%

Comment. Some requirements to take into account :

R-WP2/D2.3.0-X-1 The model shall be consistent with the SRS level and shall yield as few as possible “design choices”.

R-WP2/D2.3.0-X-1.1 Traceability with the SRS shall be provided.

R-WP2/D2.3.0-X-1.2 Each interpretation of the SRS shall be indicated precisely.

R-WP2/D2.3.0-X-1.3 Each SRS requirement not formalized in the model (*e.g.* allocated to RBC) should be traced and justified.

R-WP2/D2.3.0-X-2 When the boundary of the formalized subsystem corresponds to a FIS or FFFIS, the Functional Architecture shall try to comply to it even when it is not mandatory.

R-WP2/D2.3.0-X-3 The Functional Architecture shall split the KERNEL into independent functions.

R-WP2/D2.3.0-X-4 The Functional Architecture shall identify a subset of these functions that will be modeled.

R-WP2/D2.3.0-X-5 The Functional Architecture shall allow a universal method of adding function (modularity).

5.4.1 Software requirements

%%To detail according § 7.2 of EN 50128%%

5.4.2 Software architecture and design

%%To detail according § 7.3 of EN 50128%%

5.4.3 Software component design

%%To detail according § 7.4 of EN 50128%%

5.5 Abstract Code

%%To detail according § 7.5 of EN 50128%%

5.6 Safety activities

%%this activity is not explicitly detailed in 50128 (except some task in different activities) but in 50126 and 50129

Comment. from D.2.6,7,8,9 :

Justification. Side to side with the model (which should be a dynamic model), should lay a set of static safety properties on the model. The higher level properties will be provided by the WP2 (equivalent to a preliminary hazard analysis) from the SUBSET-91 document,

They will be refined by the safety analysis process (WP4) into properties of the same level than the model. The process of doing so shall be described in the Safety Plan.

This will provide Safety Properties on the model (or Dread Events). The lower level Safety Properties/ Dread Events shall address variables, state and interfaces used in the formal model.

Formal proof would then be used to prove that the OpenETCS model never enter a Dread State, as long as the other subsystem (RBC, communication layer...) fulfill their own safety properties (axiom describing the environment).

R-WP2/D2.3.0-X-6 A safety plan shall be provided and complied with.

R-WP2/D2.3.0-X-7 The Functional Architecture shall identify the Vital and Non Vital functions.

Comment. MPD : Identification of Vital and non vital can be done only if the safety analysers have provided safety properties and have allocated them to functions.

R-WP2/D2.3.0-X-8 The subsystem shall be compatible with the THR required in the SUBSET-091.

R-WP2/D2.3.0-X-9 The safety analysis shall consider the Dread Event of the SUBSET-091, restricted to the scope of the subsystem.

R-WP2/D2.3.0-X-10 The model-level safety properties shall be written in a formal language.

Open Issue. To discuss with all partners : what is expected on this activity in the process.

S. Baro comments (15/02/2013) :

Safety

Are safety activities required in this project? The project require "certifiability" of some items (toolchain? model?) and compliance to 50128. For the toolchain, it is quite clear what it means, but for the model it is much more complicated. Is it really required for the model? If safety activities are required on the model, I think it makes no sense to refer only to 50128: it should refer to 50126 and 50129 too (excluding the hardware part which is not in the scope of the project). This pulls system safety analysis, safety properties,... but provides the initial properties required for formal proof on the model (once refined through system safety analysis). Proposal: The "Safety" WP provides a safety case concept and safety plan according to Merlin's document and to the requirements. It is not realistic to expect that the full safety case will be provided for OpenETCS, but I think it will be useful to conduct a sample of the tasks, or all tasks but on a sample of the model. The higher level properties will be provided by subset 91, and refined by the safety analysis to properties on the model (I provided an example on how to do this in another document). Please note that even covering the whole subset 91 properties is certainly not sufficient to ensure the safety of the model. Question: who will provide the manpower for safety activities? WP4?

5.7 Verification and Validation

The Verification and Validation activities have to be planned on the whole process according the requirement of EN50128 § 6.2 and 6.3.

This plan is out of the scope of the document and shall be defined in the WP4 process.

%%To check the existing draft on this subject%%

6 OpenETCS case study

6.1 Aim of the OpenETCS project

Comment. The goal of The OpenETCS project is to provide a formal model of the On board Unit from the Subset 26 specification.

The following sections are going to give a high level description of this case study and expected elements.

Detailled specification of the system will be given during WP3 activities.

6.2 High level description of the case study

high level description of the subset + link to reference documentation (subset 26)

6.2.1 Model and Architecture

R-WP2/D2.3.0-X-11 The model shall comply all OBU ETCS mandatory requirements for level upto 2, in the functional perimeter provided by the Functional Architecture.

R-WP2/D2.3.0-X-11.1 The model shall comply the OBU part of SUBSET-26-3.3.0.

R-WP2/D2.3.0-X-11.2 The reference ETCS baseline shall be modified only by project decision, according to the QA Plan.

R-WP2/D2.3.0-X-11.3 All divergences against the chosen baseline shall be documented and tracked, according to the QA Plan.

6.3 Environement and abstract architecture

high level description of the environement of the system and main function

6.4 Runtime Model & API

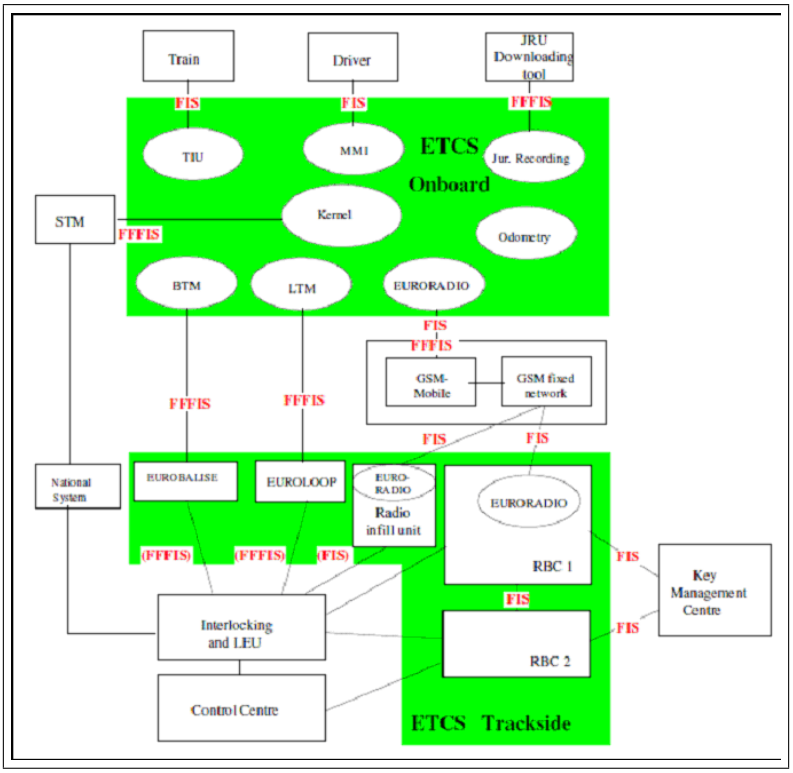


Figure 3. Architecture

In order to avoid ambiguities, we will define the following.

Runtime model. This is the abstract layer required to “run” the formal model. It shall provide “hardcoded” in the formalism part or all of the following (but not restricted to):

- memory management,
- execution of state machines (or of the chosen formal objects),

- failures,
- communication between processes and concurrence.

All these can be provided with or without safety properties. This corresponds in fact to the services provided by the "abstract machine(s)" which runs the models.

API. This is the functions/primitives required to complete the *Runtime model*. It shall provide the remaining of the features listed hereabove which are not provided by the Runtime model.

All these can be provided with or without safety properties.

RTM/API. This corresponds to the Runtime Model *plus* API. Therefore it should provide all the services needed to emulate at abstract level the hardware platform that could run the software.

Functional Architecture. This corresponds to the functional boundaries between the ETCS KERNEL and the other functional components (JRU, DMI, Odometry, Eurobalise, Euroradio…). These boundaries are described in the FIS or FFFIS. It also includes the parting of the KERNEL into different functions.

In the following requirements, we will not discriminate what is required from the API and from the RTM. This is the definition of these components that will allocate the requirements to the different parts. Hence we will only state requirements on the RTM/API.

6.4.1 RTM/API

This chapter has to be completed by the leader of D2.3.2

R-WP2/D2.3.0-X-12 The RTM/API model shall provide an abstraction layer of the hardware architecture.

R-WP2/D2.3.0-X-13 The RTM/API shall make possible to refine the software into final code able to run on hardware complying the EN 50129 standard for the requested SIL.

R-WP2/D2.3.0-X-14 The RTM/API shall allow discriminating Vital processing, data and I/O from Non Vital.

R-WP2/D2.3.0-X-15 The RTM/API shall provide a way of communication between Vital processes and Non Vital processes.

Justification. The purpose of these requirements is to be able to discriminate the safety part from the non safety part. It should be made possible to have it run on a proprietary architecture with both software on the same computer (with for example 2oo3, or coded monoprocessor) or on two different computers. One way of doing this, for example is to have some critical state machines with their data on one side, and the non critical part on the other side, with API channels to make them communicate.

Open Issue. Should it be done for just SIL4 and SIL0, or should there be all levels from SIL0 to SIL4? It should be sufficient to distinguish only between safety and non safety, but an intermediate level could be useful (for SIL2 I/O for example). I have the feeling that it should not be more complicated to have one cluster of state machine per SIL, even if we only use SIL0 and SIL4.

R-WP2/D2.3.0-X-16 The RTM/API shall allow to introduce failures for test purpose.

R-WP2/D2.3.0-X-17 The RTM/API shall provide a way of reading configuration data (*e.g.* constants,. . .)

R-WP2/D2.3.0-X-18 The RTM/API shall provide an abstraction layer of the communication and interfaces with other components.

*Justification. Even if the FIS or FFFIS requires a specific protocol (*e.g. *Profibus), this protocol will not be implemented in the high level model. It will be considered that low level communication issues are taken into account (= emulated) by the RTM/API.*

6.5 Safety properties

reference document subset 091 to the list of safety properties

7 Goals

7.1 Goal 5: Provide a subset of the safety case

Selected part of the safety process shall be applied (either by applying the full process on a small subset of the development, or by applying a part of the activities on the whole project development).

Open Issue. This point has to be discussed. The purpose here is of course to demonstrate the feasability of the safety process, but I am not convinced that it proves anything if the whole process is not applied.

7.2 Goal 6: Promote OpenETCS

Promote this work to push it to become a *de facto* standard for the industrial actors, (like *e.g.* the AUTOSAR standard in the automotive world).

8 Requirements

9 Verification, Validation and Safety issues

10 Language and formalism

This chapter has to be completed by the leader of D2.3.1 Some of the requirements in this section could suit in the Tool Chain section, but for the sake of clarity I preferred to keep them near other language requirements.

R-WP2/D2.3.0-X-19 The model formalism shall be easily understandable by the domain experts.

R-WP2/D2.3.0-X-20 The safety properties should be provided in a declarative, simple and formal language.

R-WP2/D2.3.0-X-21 The formal model shall be understandable by or exportable to many tools (SCADE, Simulink, B tools, OpenETCS tool chain. . .)

R-WP2/D2.3.0-X-22 Formal specifications should be able to formalize:

R-WP2/D2.3.0-X-22.1 State machines,

R-WP2/D2.3.0-X-22.2 Time-outs,

R-WP2/D2.3.0-X-22.3 Truth tables,

R-WP2/D2.3.0-X-22.4 Arithmetics,

R-WP2/D2.3.0-X-22.5 Braking curves,

R-WP2/D2.3.0-X-22.6 Logical statements.

Comment. This requirement does not state that all these objects need to be first order objects *of the language. It only state that it should be possible (easy?) to formalize and manipulate them.*

It is to be noted that if (for example) braking curves are objects of the language, it shall be proved that they are sound, and that the code generation for these objects is also sound.

R-WP2/D2.3.0-X-23 The formal model shall be executable.

R-WP2/D2.3.0-X-23.1 The formal model shall be executable in debug mode (step-by-step), allowing inspection of states, variables and I/O.

R-WP2/D2.3.0-X-23.2 The environment shall be emulated by high level construction of the inputs.

Justification. "High level" means that it will not be necessary to define bitwise the inputs at each cycle. On the contrary, some motorization will be available to define the behavior of the inputs.

R-WP2/D2.3.0-X-24 It shall be possible to assert logical properties on the model (*i.e. invariants*).

R-WP2/D2.3.0-X-24.1 It shall be able to check the conformance of these properties at runtime.

R-WP2/D2.3.0-X-24.2 It shall be able to prove the conformance of the model to these properties.

11 Tool chain

This chapter has to be completed by the leader of D2.3.2

R-WP2/D2.3.0-X-25 The tool chain shall be composed only of Open Source components.

R-WP2/D2.3.0-X-25.1 Closed source components may be used, but only if their use is not mandatory in the process, or if an open source counterpart is provided.

R-WP2/D2.3.0-X-26 The tool chain shall be sufficiently robust to allow large software management.

R-WP2/D2.3.0-X-26.1 It shall allow modularity at any level (proof, model, software).

R-WP2/D2.3.0-X-26.2 It shall allow the management of documentation within the same tool.

R-WP2/D2.3.0-X-26.3 It shall allow distributed software development.

R-WP2/D2.3.0-X-26.4 It shall include an *issue-tracking system*, in order to allow change management and errors/bugs management.

R-WP2/D2.3.0-X-26.5 It shall allow to document/track the differences between the model and the ERTMS reference.

*Justification. In case where errors are found in the specification, or reducting choices are to be made in the model (*e.g. *in case of non-determinism).*

R-WP2/D2.3.0-X-26.6 It shall allow concurrent version development, or be compatible with tools allowing concurrent version development.

R-WP2/D2.3.0-X-26.7 In particular, it shall be made easy [1] to track the differences between two releases of a model and to manage conflicts.

R-WP2/D2.3.0-X-26.8 In particular it shall allow to track the roles and responsabilities of each participant on a configuration item, at each step of the project lifecycle.

R-WP2/D2.3.0-X-26.9 In particular, version management shall allow to track version of the safety properties together with the model.

R-WP2/D2.3.0-X-27 The tool chain shall allow traceability between the documentation (in particular the specification) and the models and safety properties.

[1] Especially in the case of a graphical language.

R-WP2/D2.3.0-X-28 The tool chain shall allow traceability between the different layers of model and safety properties.

R-WP2/D2.3.0-X-29 The tools used in the tool chain shall be able to cooperate, *i.e.* the outputs of one tool will be suitable to be used as the inputs of the other tool.

R-WP2/D2.3.0-X-30 The tool chain shall conform to 50128 requirements, for the corresponding SIL and tool class.

R-WP2/D2.3.0-X-30.1 For T2 and T3 tools[2], the choice of tools shall be justified, and the justification shall include how the tool's failures are covered, avoided or taken into account (ref. to EN 50128 6.7.4.2).

R-WP2/D2.3.0-X-30.2 All T2 and T3 tools must be provided with their user manuals.

R-WP2/D2.3.0-X-30.3 For all T3 tool, the proof of correctness or the measure taken to guarantee the correctness of the output w.r.t. their specification and the inputs shall be provided.

R-WP2/D2.3.0-X-30.3.1 . . . for data transformation,

R-WP2/D2.3.0-X-30.3.2 . . . for software transformation (*e.g.* translation, compilation. . .).

R-WP2/D2.3.0-X-31 The tool chain shall allow to write and store *test cases* and *use cases* for the model.

R-WP2/D2.3.0-X-31.1 Version management will allow to map test cases version to model versions.

R-WP2/D2.3.0-X-32 The tool chain shall allow to generate test cases for the model.

Open Issue. Is it really necessary? If we have formal proofs on the models, the tests should stay at a functional level. Therefore generated test cases should not be interesting in this context.

Open Issue. TBD requirements on the prover. Should it verify De Bruijn's criterion[3]*. Should at least the proof tree be exportable and checkable in another tool? (if the proof tree itself is mandatory).*

%%Editors: Tree-Based, Form-Based, Text-Based, Graphical?%%
%%Code Generation%%

[2] T2: Tools contributing to the test or verification of the code or design *e.g.* static analyzers, test generators. . .)
T3: tools contributing directly or indirectly to the final code or data (*e.g.* compilers, code translator. . .)

[3] *I.e.* to be able to produce a proof tree that could be verified by a simple proof checker.

