openETCS

ITEA2

INFORMATION TECHNOLOGY FOR EUROPEAN ADVANCEMENT
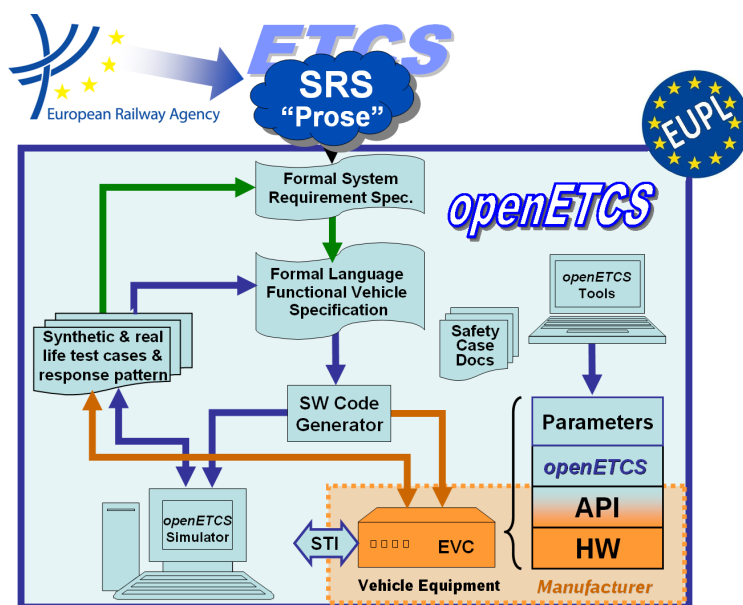
ITEA2 Project
Call 6 11025
2012 – 2015

Work Package 3: "Modelling"

openETCS Modelling Work Package

Description of Work

openETCS WP3 SRS task force, Marielle Petit-Doche, Uwe Steinke and Bernd Hekele

January 2014

This page is intentionally left blank

Work Package 3: "Modelling"   **OETCS/WP3/DescriptionOfWork**
                                                                          January 2014

openETCS Modelling Work Package
Description of Work

openETCS WP3 SRS task force

The Team

Marielle Petit-Doche

Methodology

Uwe Steinke

Objects)

Bernd Hekele

SRS Findings)

Description of work

Prepared for   openETCS@ITEA2 Project

Abstract: This work package...

Table of Contents

Figures and Tables

Figures

Tables

1   Introduction

1.1   Motivation

The openETCS work package WP3 aims to provide the architecture and design of the openETCS OBU software as mainly specified in UNISIG Subset_026 version_3.3.0.

The appropriate functionality has been divided into a list of subfunctions of different complexity (see `https://github.com/openETCS/SRS-Analysis/blob/master/SystemAnalysis/List\`
`_Functions.xlsx`).

All these functions are object of the openETCS project and have to be analyzed from their requirements and subsequently modelled and implemented. With limited manpower, a reasonable selection and order of these functions is required for the practical work that allows the distribution of the workload, more openETCS participants to join and leads to an executable– limited – kernel function as soon as possible.

While the first version of this document focuses on the first version of the limited kernel function, it is intended to grow in parallel to the growing openETCS software.

1.2   Objectives

The first objective of WP3 software shall be

- "Make the train run as soon as possible, with a very minimum functionality, and in the form of a rapid prototype."

This does not contradict the openETCS goal to provide conformance to EN50128.

- After a phase of prototyping, the openETCS software shall be implemented compliant to EN50128 for SIL4 systems.

Additional goals for this document are

- Identification of the functions required for a minimum OBU kernel

- Architecture overview regarding the minimum OBU kernel

- Technical approach: Description of the proceeding and methods to be used

- Road map of the minimum OBU kernel functions

- Road map thereafter

Note: This document will be extended according to the progress of WP3.

2   Goals of the openETCS Modelling Work

by Uwe

2.1 Functional Scope: The Minimum OBU Kernel Function

The objective "Make the train run with a very minimum functionality" shall be in terms of on ETCS OBU translated into

- The Train moves on a track equipped with balises and determines its position.

That means, for this very first step the train shall not supervise the maximum speed, shall not activate the brakes. Instead, the minimum function set shall be limited to (see `https://github.com/openETCS/SRS-Analysis/issues/9`)

- Receive, filter and manage balise information, received from track (see `https://github.com/openETCS/SRS-Analysis/issues/12`)

- Calculate the actual train position based on balise and odometry information (see `https://github.com/openETCS/SRS-Analysis/issues/8`)

- Calculate the distances between the actual train position to track elements in its front

A more detailed architectural breakdown of these functions is available in the form of a SysML model at (see `https://github.com/openETCS/modeling/tree/master/model/sysml`.

In addition, the work on this minimum functionality requires to be supported by

- The availability of the ETCS language as specified in Subset UNISIG Subset_026, chapter 7 and 8

- The abiltiy to link intermediate and final results with the requirements of the ETCS specification (subset_026, ..)

- The usability of a data dictionary (see `https://github.com/openETCS/dataDictionary`)

These supporting prerequisites are under construction and therefore not completely operable actually. How to deal with these restrictions, will be outlined in chapter ???

2.2 Actual Status

Some first analyis steps for the required minimum functionality have been gone as results from the SRS-Analysis task force. These results are available on `https://github.com/openETCS/SRS-Analysis`

2.3 Practical Approach

The architecture and design of the minimum OBU kernel shall be developed in consideration of the actual status, restricted prerequisites and limited resources as follows.

2.4 Organisation of the Work package

Due to historical reasons we have some projects linked to the openETCS modeling activities. In the progress of our task we will make a clean-up on the history. Here is an overview on the different repositories with reasonable content.

- `https://github.com/openETCS/modeling`
  The modeling repository is the place where modeling takes place. openETCS ToolChain is linked to this repository (folder "model". All code artefacts have to reside in this location.

- `https://github.com/openETCS/SRS-Analysis`
  The SRS-Analysis Repository was originally introduced to support the task force led by Alstom and intended to start with the analysis work. In this location you can find traces of the analysis work. The repository is still in use.

- `https://github.com/openETCS/dataDictionary`
  The data-Dictionary repository is related to the definition of common data and functions. The dataDictionry implementation is based on SysML/Papyrus. Therefore, not tools nor tool artefacts are stored here. Only documentation is needed. The repository is still in use.

- `https://github.com/openETCS/SSRS`
  The SSRS activities had been used before the modelling kick-off to prepare for the analysis work. There are stil some useful documents included. Especially the wiki is usefull. However, the place is no longer used for active work and the repository can - after having moved valid information to other locations - be considered to be recycled.

- Requirements on Process and Methods, Guidelines
  The following guidelines are the basis for modeling work: The openETCS development process: `https://github.com/openETCS/requirements/blob/master/D2.3/D2_3.pdf`
  The openETCS requirements on methods: `https://github.com/openETCS/requirements/blob/master/D2.4/D2_4.pdf` This document also describes naming conventions which you need to respect for working in this task.

3 Methodology

This section gives some information on how to proceed to achieve the objectives described in section 1.2. As decided previously in the project, the means chosen to design the WP3 software, compliant to SIL4 requirements of EN50128 are SysML on Papyrus and SCADE for the modelling part, C language for the executive software. The first versions of the OpenETCS toolchain already involve Papyrus.

Other tools can be involved to support the task of the workpackage (indeed to manage requirements, data, traceability,...) but will be defined latter depending of the needs and the propositions.

For further information on means selection and detailed description of the process, consider the deliverables [D7.1][1], [D7.2][2] and [D2.4][3].

---

[1]`https://github.com/openETCS/toolchain/blob/master/T7.1/D7.1/D7.1.pdf`
[2]`https://github.com/openETCS/toolchain/blob/master/T7.2/D7.2/D7_2.pdf`
[3]`https://github.com/openETCS/requirements/blob/master/D2.4/D2_4.pdf`

3.1 Main step of the analysis

The figure 1 gives the main steps of the design approach and main used and produced artefacts. All is detailed in D2.4.

Figure 1. Main steps of the design approach

Four main steps are defined:

System Analysis to identify the main functions and their interactions and to clarify the requirements allocated to them.

Architectural Modelling to specify the functional architecture of the system to design and internal and external interfaces. In parallel the main data exchanged between the functions shall be identified.

Functional and Behavioural Modelling to provide a formal model of each function.

Executable Software Design to provide executable part of software.

These steps are sharing common artifacts which going to be updated and completed during all the design phase. Some recommendations and guidelines are defined in D2.4 for naming of elements or specifing a model.

As the artifacts are shared, an iterative process can be easily applied at each level.

3.2 Shared artifacts

3.2.1 Functional architecture and API

API[4] is a document provided as input which can be updated according to the needs.

Functional architecture is specified as a SysML model, which can be automatically translated in a Scade model. An initial SysML model to cover the initial functional scope 2.1 is available on github: `https://github.com/openETCS/modeling/tree/master/model/sysml`

See D2.4 for how to use SysML to define the Functional Architecture.

3.2.2 Set of Requirements

The initial set of requirements is the contains of Subset_026 v3.3.0. These requirements are available as ReqIF format on github: `https://github.com/openETCS/modeling/tree/master/model/subset26` .

This set is going to be increased during the design with updated requirements and added requirements.

Data model of D2.4 gives the information to manage on the requirements.

For the moment, ProR is involved in the openETCS toolchain to define new requirements and links between them. It is still to clarify how to manage traceability.

3.2.3 Data Dictionary

The Data Dictionary is the set of all the types, constants and variables defined to describe the system. Data model of D2.4 describes how to define a data.

A preliminary task of the system analysis is to identified and specify the data structure which allow to describe the system.

Then the data structure and data definition shall be implemented in the data dictionary.

For the moment the mean to do this implementation is not clearly identified (UML library ? XML files ?)

3.2.4 Formal models

Two models are provided during this phase:

- a semi-formal model in SysML which gives the functional architecture of the system, including interaction between each function and definition of the data.

- a formal model in SCADE which follows and completes the same architecture and gives a behavioural description of each function. Then C code can be automatically generated from this model.

---

[4]`https://github.com/openETCS/requirements/blob/master/D2.7-Technical_Appendix/OETCS_API\Requirements_v1.0Draft_130301.pdf`

3.3   Function specification and design

For the specification and the design of a functional block, a set of subtasks can be defined:

1. Identify the function and the input document to describe it (requirements of subset 26 or other subsets, API,, functional architecture,...)

2. Specify its environment and its external interfaces with an ibd diagram in SysML

3. Define and specify its internal decomposition in subfunctions and its internal interfaces with ibd and bdd diagrams in SysML

4. Link and complete to the data dictionary

5. Allocate and manage the requirements

6. Clarify and specify the behavior of each elementary function in SCADE

7. Complete Data dictionary if necessary

8. Complete requirement sets and manage traceability

9. Provide for review

4   Verification and Validation of Modeling artefacts

The handling is based on the openETCS review process which is documented as an extension of the quality insurance plan. `https://github.com/openETCS/governance/tree/master/Review%20Process`. In short, this is done in posting an issue with the issue tracker of the modeling repository. Before passing the artefact to WP4, the WP3 product owner has to agree to the triggering of verification and validation.

5   How to Handle Findings in the SRS

The work of the openETCS project is based on several standards and standardised specifications, in particular the SRS. During modelling activities, issues with the specifications will arise. While some may be caused by insufficient domain knowledge of the modeller, others may be errors that need to be reported back to the standardisation institutions.

(Confirmed) specification findings can be classified according to the following categories:

Unclearity The specification/standard is not precise enough with an item, the specification should be detailed. The (semi-)formal model to be built in openETCS will certainly help here.

Ambiguity An item can be interpreted in different ways that can lead to erroneous, incompatible or even harmful implementations.

Error There is an error in the specification that may lead to a faulty implementation with possibly safety-critical effects.

Improvement Proposal Proposal on how to improve a certain standard/specification based on project experience/results.
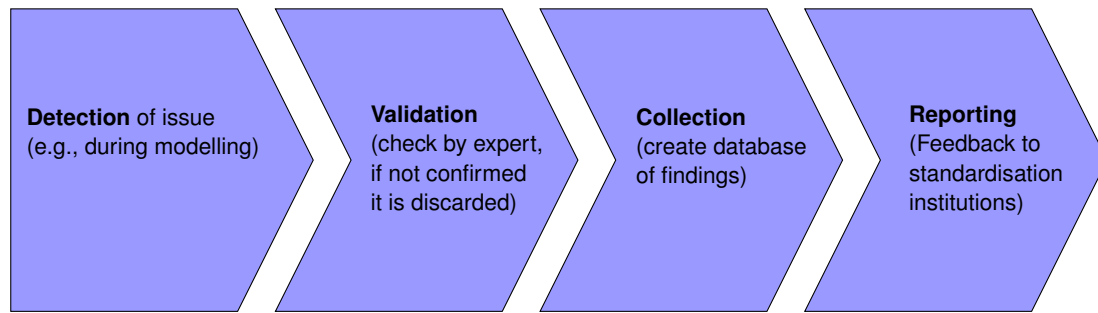
Figure 2. Process for handling specification findings

The formal model to be built in WP3 can be the basis for improving the existing standards and thus it is vital to collect findings and their (potential) resolution. Figure 2 depicts the process that is proposed for openETCS.

If an issue is detected during modelling it should be recorded. If the modeller is unable to resolve the issue it should be checked by some domain expert who can either resolve it, or confirm it as an SRS finding. If possible, a resolution of the issue should be proposed and stored along with the finding in a database of findings. At a later point in time the specification findings are then reported back to the standardisation institutions to improve the existing standards.

To implement this process in the project, the following steps are necessary:

1. Setup an infrastructure for reporting findings (GitHub-Repository?)

2. Identify experts with the standard that can help in resolving issues.

3. Collect resolved issues, transform them into change requests.

4. Designate (a) responsible person(s) for this process.